Modifications to Reduce the Minimum Thermal Source Temperature of the ST05G-CNC Stirling Engine

by

Connor Paul Speer

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering
University of Alberta

© Connor Paul Speer, 2018

Abstract

An experimental study has been executed with two main objectives. The applicability of the second order modeling approach for low thermal source temperature Stirling engines was assessed, and modifications to improve the low thermal source temperature performance of an ST05G-CNC Stirling engine were studied.

A test rig was fabricated, which featured a modified ST05G-CNC Stirling engine equipped with instrumentation to measure gas and coolant temperatures, gas pressures, output torque, crankshaft position, and coolant flow rate. Novel aspects of the test rig include high speed buffer pressure measurements, and separation of the water jacket into two independent zones.

A second order mathematical model was assembled drawing from the literature. The model could use either the ideal isothermal model or the ideal adiabatic model as a reference cycle. Imperfect heat transfer, regenerator enthalpy loss, flow friction, mechanical friction, appendix gap loss, and conduction loss were calculated based on the reference cycle results. It was shown that the two reference cycles studied give increasingly similar results as the thermal source temperature declines.

Three modifications to the engine were tested experimentally: The piston diameter was decreased from 85 mm to 44 mm, the crankcase volume was increased from 3.20 L to 7.83 L, and the dead volume of the working space was reduced from 0.877 L to 0.745 L. All together the modifications reduced the minimum thermal source temperature from 242 °C to 145 °C, with a constant thermal sink temperature of 21 °C. The reduced piston diameter and increased

crankcase volume were shown experimentally to increase the shaft power of the engine at low thermal source temperatures by reducing forced work and crankcase gas spring hysteresis. The reduction in dead volume had an immeasurable influence on the shaft power, but shifted some heat rejection duty from the connecting pipe and power cylinder water jackets to the cooler water jacket.

Agreement between experimental results and model predictions was evaluated individually for sub-components of the mathematical model. Correspondence between predicted and measured indicator diagrams was found to worsen with decreasing thermal source temperature. The trend was attributed to the greater influence of losses neglected by the reference cycles at lower thermal source temperatures. Since all other components of the model rely on the reference cycle for their input parameters, the implications of reference cycle errors are far-reaching. Neglect of losses in the reference cycle simulation is fundamental to the second order modeling approach. The validity of this assumption has been shown to break down as thermal source temperature declines.

With consideration of the experimental data, replacement of the heater head was recommended as a means of reducing the minimum thermal source temperature further. By replacing the heater with something similar to the current cooler, low thermal source temperature performance could be improved by reducing temperature drop between the thermal source and the engine working fluid, and by reducing conduction loss.

Preface

This thesis is an original work by Connor Speer. No part of this thesis has been previously published.

Acknowledgements

The author gratefully acknowledges the support of his supervisor,

Dr. Nobes

his lab mates (Team Stirling),

David Miller

Calynn Stumpf

Jason Michaud

Steven Middleton

Michael Nicol-Seto

Alex Hunt

Michael Bayans

Jackson Kutzner

and the co-op students and volunteers,

Sam Tseung

Anders Carlstad

Yanwei Zhang

Jiacheng Yao

Spencer O'Donnell

Jakub Piwowarczyk

Shahzeb Mirza

who all helped to make this project a success.

Financial support for this research was provided by the following funding bodies:

Natural Sciences and Engineering Research Council (NSERC) of Canada

Canadian Foundation of Innovation (CFI)

Alberta Innovates Energy and Environment Solutions

Innovative Solar Power

Terrapin Geothermics

Table of Contents

Abstract ii

Preface iv

Acknowledgements v

Table of Contents vii

List of Tables xii

List of Figures xiv

List of Symbols xx

Chapter 1. Introduction and Literature Review 1

1.1 Motivation 2

1.2 Stirling Engine Operating Principles 4

1.3 Types of Stirling Engines 10

1.4 Model Classification/Overview 12

1.4.1 Conservation Laws 12

1.4.2 Classification of Models 13

1.4.3 Beale and West Number Correlations 14

1.4.4 Reference Cycles 15

1.5 Decoupled Losses 27

1.5.1 Regenerator Enthalpy Loss 27

1.5.2 Conduction Loss 27

1.5.3 Appendix Gap Loss 27

1.5.4 Mechanical Friction 28

1.5.5 Flow Friction 30

1.5.6 Gas Spring Hysteresis 30
1.5.7 Heat Transfer Hysteresis 31
1.5.8 Seal Leakage 31
1.5.9 Piston Finite Speed Loss 31
1.5.10 Auxiliary Component Losses 32
1.6 The Simple Heat Exchanger Model 33
1.7 Second Order Models from the Literature 39
1.8 The ST05G Stirling Engine 43
1.9 Low Temperature Difference Stirling Engines 49
1.10 Thesis Objectives and Structure 52
Chapter 2. Description of the Experimental Set-up 53
2.1 Overview 54
2.2 Modified ST05G-CNC Stirling Engine 56
2.3 Heating System 60
2.4 Cooling System 63
2.5 Braking System 66
2.6 Motoring Test System 67
2.7 Instrumentation System 68
2.7.1 Gas Temperature Measurement System 69
2.7.2 Pressure Measurement System 74
2.7.3 Angular Position Measurement System 83
2.7.4 Torque Measurement System 85
2.7.5 Heat Input Measurement System 85
2.7.6 Heat Rejection Measurement System 87
2.7.7 Data Acquisition Software 88

2.8 Experimental Procedures 90
2.8.1 Warm-Up Procedure 90
2.8.2 Performance Testing Procedure 90
2.8.3 Efficiency Measurement 91
2.8.4 Conduction Loss Measurement 91
2.8.5 Cool-Down Procedure 91
2.8.6 Cold Motoring Test Procedure 91
2.9 Experimental Uncertainty 93
2.10 Conclusions 96
Chapter 3. Development of a Second Order Mathematical Model 97
3.1 Calculation of Constant Volumes 98
3.2 Calculation of Volume Variations 100
3.3 Calculation of Working Fluid Mass 105
3.4 Ideal/Reference Cycle 107
3.5 Decoupled Loss Calculations 110
3.5.1 Regenerator Enthalpy Loss 110
3.5.2 Conduction Loss 111
3.5.3 Appendix Gap Loss 112
3.5.4 Mechanical Friction and Forced Work 114
3.5.5 Flow Friction 115
3.5.6 Losses Neglected in the Mathematical Model 116
3.6 Addition of Losses to the Reference Cycle Results 118
3.7 Sample Calculations 120
3.8 Conclusions 123
Chapter 4. Modifications to Reduce Minimum Thermal Source Temperature 124

4.1 Preliminary Results 125
4.2 Engine Modifications 127
4.2.1 Modification 1: Reduced Piston Diameter 127
4.2.2 Modification 2: Crankcase Extension 130
4.2.3 Modification 3: Dead Volume Reduction Parts 131
4.3 Experimental Results – Effect of Modifications 133
4.4 Experimental Results – Cool Down Tests 141
4.5 Conclusions 149
Chapter 5. Performance Evaluation and Model Assessment for Fully Modified Engine 150
5.1 Reference Cycle Indicator Diagram 151
5.2 Crankcase Gas Spring Hysteresis 155
5.3 Mechanical Friction 158
5.4 Flow Friction 161
5.5 Heat Exchanger Performance 166
5.6 Evidence of Preferential Flow 169
5.7 Heat Lost Through the Heating Cap Insulation 171
5.8 Conduction loss 172
5.9 Relative Influence of the Cooling Zones 174
5.10 Discussion of Losses Not Measured 176
5.11 Overall Engine Performance 178
5.12 Conclusions 187
Chapter 6. Conclusions and Future Work 188
References 190
Appendix A: Analysis of Experimental Uncertainty 199
Appendix B: Optimum Swept Volume Ratio Based on Equal Pressure Swings 215

Appendix C: MATLAB Source Code for Mathematical Models 222

Appendix D: MATLAB Source Code for Data Processing 323

Appendix E: Drawing Package for Modified Stirling Engine 405

List of Tables

Table 1.1: Ideal Isothermal Model Equation Set [10] 19
Table 1.2: Ideal Adiabatic Model Equation Set 26
Table 1.3 Design Data for Original ST05G Stirling Engine [53] 43
Table 1.4: Summary of Power Measurements of Engines Similar to the ST05G-CNC Reported in the Literature 47
Table 2.1: Estimated Overall Uncertainties for Base Measurements 94
Table 2.2: Propagated Uncertainties for Calculated Quantities 95
Table 3.1: Information Associated with Calculation of the Regenerator Volume 99
Table 3.2: Gamma-Type Slider-Crank Volume Variation Equation Set 102
Table 3.3: Expansion Space Volume Variation Derivative Equation Set 102
Table 3.4: Compression Space Volume Variation Derivative Equation Set 103
Table 3.5: Definition of Isothermal and Adiabatic Engine Components 107
Table 3.6: Information Associated with Calculation of the Conduction Loss 112
Table 3.7: Scheme for Addition of Losses to Reference Cycle Results 118
Table 4.1: Optimum Piston Diameters for Thermal Source Temperature of 95 °C and Thermal Sink Temperature of 5 °C 129
Table 4.2: Numerical Results of Second Order Model Simulation for Reduced Piston Diameter 130
Table 4.3: Modeling Results for Addition of Dead Volume Reduction Components 132
Table 4.4: Minimum Thermal Source Temperature Reached During Cool Down Tests for All Engine Configurations 141
Table 5.1: Second Order Polynomial Fit Coefficients for Crankcase Gas Spring Hysteresis Empirical Formula 157
Table A.1 Estimated Uncertainties for Thermocouple Measurements 200
Table A.2 Estimated Uncertainties for RTD Measurements 201
Table A.3: Inputs and Results for Uncertainty Propagation in Temperature Difference 202
Table A.4 Estimated Uncertainties for Validyne Pressure Measurements 203

Table A.5: Estimated Uncertainties for PCB Pressure Measurements 204
Table A.6 Estimated Uncertainties for Crankshaft Position Measurement 205
Table A.7 Estimated Uncertainties for Time Measurement 205
Table A.8: Inputs and Results for Uncertainty Propagation in Angular Frequency 206
Table A.9 Estimated Uncertainties for Torque Measurement 207
Table A.10: Inputs and Results for Uncertainty Propagation in Shaft Power 208
Table A.11 Estimated Uncertainties for Current Measurements 209
Table A.12 Estimated Uncertainties for Resistance Measurements 209
Table A.13: Inputs and Results for Uncertainty Propagation in Heat Input Rate 210
Table A.14 Estimated Uncertainties for Volume Flowrate Measurements 211
Table A.15 Estimated Uncertainties for Coolant Density, ρ 211
Table A.16 Estimated Uncertainties for Coolant Specific Heat Capacity 211
Table A.17: Inputs and Results for Uncertainty Propagation in Heat Rejection Rate 212
Table A.18: Inputs and Results for Uncertainty Propagation in Thermal Efficiency 213

List of Figures

Figure 1.1: CAD Model of Displacer Cylinder, after West 4
Figure 1.2: CAD Model of a Displacer Cylinder (a) Working Fluid Displaced to the Compression Space (b) Working Fluid Displaced to the Expansion Space 5
Figure 1.3: CAD Model of a Displacer Cylinder and Power Cylinder 6
Figure 1.4: Four Processes Comprising the Ideal Stirling Cycle (a) Isochoric Heat Addition (b) Isothermal Expansion (c) Isochoric Heat Rejection (d) Isothermal Compression 7
Figure 1.5: CAD Model of Displacer Cylinder, Power Cylinder, and Heat Exchangers 8
Figure 1.6: CAD Model of Five Component Stirling Machine 9
Figure 1.7: Three Main Stirling Engine Layouts (a) Alpha Type (b) Beta Type (c) Gamma Type 10
Figure 1.8: Schematic and Temperature Profile for the Ideal Isothermal Model 16
Figure 1.9: Schematic and Temperature Profile for the Ideal Adiabatic Model 21
Figure 1.10: Indicator Diagram Showing Forced Work 29
Figure 1.11: Fluid and Matrix Temperatures Defining Nomenclature for Regenerator Effectiveness, after Urieli and Berchowitz [10] 33
Figure 1.12: Algorithmic Approach for the Simple Model 37
Figure 1.13: (a) Original ST05G Stirling Engine [53] (b) Sectioned Solid Model of ST05G-CNC Stirling Engine 44
Figure 2.1: Image of Overall Experimental Set-up 54
Figure 2.2: Top View of Engine Output Shaft 55
Figure 2.3: Rendered Solid Models of (a) Original ST05G-CNC Design (b) As-Built Engine for Solar Power Application 56
Figure 2.4: Image of Liquid Nitrogen Shrink Fit Process for Heater Head Assembly 57
Figure 2.5: (a) Rendered Section View of Heating Cap CAD Model (b) Image of Heating Cap and Cartridge Heaters As-Built 60
Figure 2.6: Schematic of Circuit for Heat Input Measurement System 62
Figure 2.7: Schematic Defining the Two Cooling Zones 64

Figure 2.8: Top View of Test Rig Set-Up for Cold Motoring Tests 67
Figure 2.9: Block Diagram of Instrumentation System 68
Figure 2.10: Locations of Gas Temperature Measurements 69
Figure 2.11: Image of Gas Thermocouple Tip 70
Figure 2.12: Example of Raw Data Set for Gas Temperature Measurements 71
Figure 2.13: Measured Expansion Space Gas Temperature as a Function of Crank Angle 72
Figure 2.14: Difference between Temperature Measured by Thermocouples and Overall Average Temperature (a) Before Applying Correction Terms and (b) After Applying Correction Terms 73
Figure 2.15: Locations of Gas Pressure Measurements 74
Figure 2.16: Image of Tube Set-Up for Regenerator Differential Pressure Measurement 75
Figure 2.17: Regenerator Pressure Drop as a Function of Crank Angle for (a) Several Rotational Speeds at a Constant Fill Pressure of 414 kPa (b) Several Fill Pressures at a Constant Rotational Speed of 5 Hz 76
Figure 2.18: Potential Impact of Lag in Pressure Measurement on Indicator Diagram Shape 77
Figure 2.19: Tubes Tested to Confirm Pressure Measurement Lag 78
Figure 2.20: Normalized Power Cylinder Pressures for Fill Pressures of (a) 138 kPa (20 PSI) and (b) 827 kPa (120 PSI) 78
Figure 2.21: Difference between Measured Pressure and Calibration Machine Setpoint (a) Before Applying Correction Terms and (b) After Applying Correction Terms for Outboard Transducers at the Crankcase and Power Cylinder Locations 80
Figure 2.22: Difference between Measured Pressure and Calibration Machine Setpoint (a) Before Applying Correction Terms and (b) After Applying Correction Terms for a Single Outboard Transducer Connected between the Regenerator Hot and Regenerator Cold Locations 81
Figure 2.23: Example of Raw Data for Pressure Measurement 82
Figure 2.24: Comparison Between Validyne and PCB Pressure Transducers for Measuring Indicated Work 82
Figure 2.25: Image of the Angular Position Measurement Set-Up 83
Figure 2.26: Example of Raw Data Collected from the Rotary Encoder 84
Figure 2.27: Example of Raw Data for Torque Measurement 85
Figure 2.28: Example of Raw Data for Heater Control Signal 86

Figure 2.29: Difference between Measured Water Temperature and Overall Average Temperature (a) Before Applying Correction Terms and (b) After Applying Correction Terms 87
Figure 2.30: Example of Raw Water Temperatures as a Function of Time 88
Figure 2.31: Screen Shot of Graphical User Interface for DAQ Program 89
Figure 3.1: CAD Model of Engine Internal Volume and Displacer 98
Figure 3.2: Schematic of Generic Slider-Crank Mechanism (after Cleghorn and Dechev [89]) 100
Figure 3.3: Slider-Crank Mechanism Pair Equivalent to Modified ST05G-CNC Drive Mechanism 101
Figure 3.4: Comparison Between Sinusoidal Motion and Slider-Crank Mechanism Motion for (a) the Piston and (b) the Displacer 104
Figure 3.5: Comparison Between Ideal Isothermal Model and Ideal Adiabatic Model for (a) Indicated Work and (b) Thermal Efficiency as Functions of Heater Gas Temperature 108
Figure 3.6: Ideal Isothermal and Adiabatic Model Indicator Diagrams for Heater Gas Temperatures of (a) 500 °C and (b) 50 °C 109
Figure 3.7: Variation in Regenerator Enthalpy Loss as a Function of (a) Regenerator Wire Diameter and (b) Regenerator Porosity 111
Figure 3.8: Calculated Appendix Gap Loss as a Function of Appendix Gap Width 114
Figure 3.9: Example Forced Work Plot for Adiabatic Buffer Space 115
Figure 3.10: (a) Calculated Pressure Drops as a Function of Crank Angle for 10 Hz Engine Frequency (b) Flow Friction Losses as a Function of Engine Frequency 116
Figure 3.11: (a) Power and Efficiency Curves and (b) Normalized Power Losses as a Function of Frequency 120
Figure 3.12: (a) Power and Efficiency Curves and (b) Normalized Power Losses as a Function of Mean Pressure 121
Figure 3.13: (a) Power and Efficiency Curves and (b) Normalized Power Losses as a Function of Heater Wall Temperature 122
Figure 4.1: Experimental Indicator Diagram Taken Near the Cool Down Test Stall Point of the Engine As-Built 125
Figure 4.2: Rendered CAD Models of (a) As-Built Engine for Solar Power Application (b) Further Modifications for Low Hot Source Temperature Application 127
Figure 4.3: Annotated Section View of Glass Cylinder Holder 128

Figure 4.4: Calculated Forced Work Plots for (a) 85 mm Piston and (b) 44 mm Piston at 200 °C Heater Gas Temperature, 5 Hz Operating Frequency, and 10 bar Mean Pressure 129
Figure 4.5: Sectioned CAD Model of Crankcase Volume Extension 130
Figure 4.6: Sectioned Solid Model of Dead Volume Reduction Parts 131
Figure 4.7: Experimental Indicated Power as a Function of Frequency for Three Modifications 134
Figure 4.8: Experimental Torque as a Function of Frequency for Three Modifications 135
Figure 4.9: Experimental Shaft Power as a Function of Frequency for Three Modifications 136
Figure 4.10: Measured Indicator Diagrams with and without the Crankcase Extension Modification 137
Figure 4.11: Crankcase Gas Spring Hysteresis Loss for the Three Modifications 138
Figure 4.12: Experimental Forced Work for the Three Modifications 139
Figure 4.13: (a) Indicated Power and (b) Shaft Power as a Function of Frequency with and without the Dead Volume Reduction Parts 140
Figure 4.14: Heat Rejection Rates from (a) the Connecting Pipe and Power Cylinder and (b) the Cooler 140
Figure 4.15: Loaded Cool Down Test Results for (a) Torque and (b) Shaft Power 142
Figure 4.16: Mean Crankcase and Power Cylinder Pressures as a Function of Expansion Space Temperature for the Loaded Cool Down Test 143
Figure 4.17: (a) Indicated Work and (b) Forced Work as a Function of Expansion Space Temperature for Loaded Cool Down Test with Fully Modified Engine 144
Figure 4.18: Engine Frequency as a Function of Expansion Space Temperature for the Loaded Cool Down Test 145
Figure 4.19: Series of Indicator Diagrams for Loaded Cool Down Test of Fully Modified Engine 146
Figure 4.20: Forced Work Ratio as a Function of Expansion Space Temperature for Loaded Cool Down Test 147
Figure 4.21: Gas Spring Hysteresis Loss as a Function of Expansion Space Temperature for the Loaded Cool Down Test 148

Figure 5.1: Experimental Indicator Diagrams with Overlaid Reference Cycle Models (a) As-Built Engine Running with a Heating Cap Temperature of 400 °C (b) Fully Modified Engine Running with a Heating Cap Temperature of 300 °C 152
Figure 5.2: Indicated Work Error for Isothermal and Adiabatic Reference Cycle Models as a Function of Engine Frequency and Heating Cap Temperature 153
Figure 5.3: Measured Crankcase Gas Spring Hysteresis Loss for the Fully Modified Engine as a Function of Engine Frequency and Mean Pressure 155
Figure 5.4: Second Order Polynomial Surface Fit to Measured Crankcase Gas Spring Hysteresis Loss Data 156
Figure 5.5: Measured Buffer Pressure with Overlaid Adiabatic and Isothermal Buffer Pressure Models..... 159
Figure 5.6: Measured Power Consumption During Cold Motoring Tests..... 162
Figure 5.7: Measured Regenerator Pressure Drop as a Function of Crank Angle with Overlaid Quasi-Steady Model for (a) Three Mean Pressures at an Engine Frequency of 1.5 Hz and (b) Three Engine Frequencies at a Mean Pressure of 562 kPa (Positive Direction is from the Cold to the Hot Side of the Regenerator) 163
Figure 5.8: Measured Regenerator Pressure Drop Amplitude as a Function of Speed and Mean Pressure for a Heating Cap Temperature of 200 °C 164
Figure 5.9: Regenerator Pressure Drop Amplitude Error as a Function of Engine Frequency and Mean Pressure..... 165
Figure 5.10: (a) Measured Temperature Drop between the Heater Gas Temperature and the Heating Cap Setpoint and (b) Measured Temperature Drop between the Cooler Gas Temperature and the Water Bath Setpoint 167
Figure 5.11: Measured Gas Temperatures at the Interface between the Regenerator and the Cooler with Engine Running at a Heating Cap Temperature of 200 °C..... 169
Figure 5.12: Measured Heater Pressure Drop Amplitude (in kPa) as a Function of Azimuthal Position 170
Figure 5.13: Measured Heat Loss Rate Through the Heating Cap Insulation Expressed as (a) An Energy Transfer Rate and (b) A Fraction of the Heat Input Rate..... 171
Figure 5.14: (a) Comparison between Measured and Calculated Conduction Loss Rate and (b) Measured Conduction Loss Rate Expressed as a Fraction of the Measured Heat Input 173

Figure 5.15: Portion of Heat Rejected in the Connecting Pipe and Power Cylinder as a Function of Engine Frequency and Mean Pressure for a Heating Cap Temperature of 200 °C 175
Figure 5.16: Measured Torque as a Function of Engine Frequency and Mean Pressure for a Heating Cap Temperature of 200 °C 179
Figure 5.17: Measured Shaft Power as a Function of Engine Frequency and Mean Pressure for a Heating Cap Temperature of 200 °C 180
Figure 5.18: Measured Thermal Efficiency as a Function of Engine Frequency for a Heating Cap Temperature of 300 °C and a Mean Pressure of 430 kPa 181
Figure 5.19: Measured Heat Input Rate with Overlaid Second Order Model for a Heating Cap Temperature of 300 °C and a Mean Pressure of 430 kPa 183
Figure 5.20: Measured Thermal Efficiency with Overlaid Second Order Model for a Heating Cap Temperature of 300 °C and a Mean Pressure of 430 kPa 184
Figure 5.21: Measured Shaft Power as a Function of Engine Frequency and Mean Pressure with Overlaid Second Order Model. Data was collected at a heating cap temperature of 200 °C. 185
Figure 5.22: Measured Heat Rejection Rate as a Function of Engine Frequency and Mean Pressure with Overlaid Second Order Model. Data was collected at a heating cap temperature of 200 °C. 186

List of Symbols

Roman Alphabet Symbols

a	parameter in finite piston speed loss calculation, $a \equiv \sqrt{3\gamma}$	-
A	surface area	m^2
A_{cond}	cross-sectional area of conduction path	m^2
A_r	regenerator surface area (includes matrix and inside of housing)	m^2
A_{flow}	cross-sectional area in the flow direction	m^2
b	appendix gap width	m
c	specific heat capacity	J/kgK
c_p	specific heat at constant pressure	J/kgK
c_v	specific heat at constant volume	J/kgK
c_{FPS}	average gas molecule speed for finite speed loss calculation	m/s
Dr_1	length of the displacer desaxe offset	m
Dr_2	length of the displacer crank arm	m
Dr_3	length of the displacer connecting rod	m
Dr_4	distance between crankshaft rotational axis and wrist pin for displacer slider crank mechanism	m
DV_c	portion of variable compression space volume controlled by the displacer	m^3
d	diameter, or hydraulic diameter	m
d_{wire}	diameter of regenerator wire	m
E	mechanism effectiveness	-
f	frequency	Hz
f_{Re}	Reynolds friction factor, equal to the Fanning friction factor multiplied by the Reynolds number	-
F	force	N

g	gravitational acceleration	m/s^2
h	specific enthalpy	J/kg
h_{conv}	convective heat transfer coefficient	W/m^2K
k	thermal conductivity	W/mK
L	length	m
M	total mass in the engine	kg
m	mass	kg
$\dot{m}$	mass flow rate	kg/s
m_{flux}	mass flux	kg/m^2s
N_B	Beale number, $N_B \equiv \frac{\dot{W}_{shaft}}{fV_{swp}P_{mean}}$	-
N_{Pr}	Prandtl number, $N_{Pr} \equiv \frac{c_p \mu}{k}$	-
N_{Re}	Reynolds number, $N_{Re} \equiv \frac{\rho u d}{\mu}$	-
N_{St}	Stanton number, $N_{St} \equiv \frac{h}{\rho u c_p} = \frac{N_{Nu}}{N_{Re}N_{Pr}}$	-
NTU	number of transfer units, $NTU \equiv \frac{hA_{regen}}{\dot{m}c_p}$	-
N_W	West number, $N_W \equiv N_B \frac{(T_H+T_C)}{(T_H-T_C)}$	-
p	pressure	Pa
p_i	average pressure for current volume change increment	Pa
p_{mean}	mean cycle pressure	Pa
Pr_1	length of the piston desaxe offset	m
Pr_2	length of the piston crank arm	m
Pr_3	length of the piston connecting rod	m
Pr_4	distance between crankshaft rotational axis and wrist pin for piston slider crank mechanism	m
PV_c	portion of variable compression space volume controlled by the piston	m^3
¶	volumetric porosity	-
$\dot{q}$	heat transfer rate per unit mass	W/kg

Q	heat	J
Q_{rloss}	regenerator enthalpy loss per cycle	J
$\dot{Q}$	heat transfer rate	W
$\dot{Q}_{app}$	rate of energy loss in the appendix gap	W
$\dot{Q}_{cond}$	conductive heat transfer rate	W
$\dot{Q}_{rej}$	heat rejection rate	W
R	specific ideal gas constant	J/kgK
S	stroke	m
t	time	s
T	temperature	K
T_{error}	temperature error in simple heat exchanger model iterative loop	K
u	specific internal energy, or mean gas velocity	J/kg or m/s
U	internal energy	J
V	volume	m^3
$\dot{V}$	volume flow rate	m^3/s
V_{swp}	swept volume of the piston	m^3
w	piston speed	m/s
W	work	J
W_f	forced work	J
W_{flow}	work lost to flow friction	J
W_i	indicated work, calculated area of the indicator diagram	J
δW_{FPS}	incremental work lost due to finite piston speed	J
$\dot{W}$	power	W
$\dot{W}_{shaft}$	shaft power, measured at the output shaft	W
x	linear distance	m
z	height	m

Greek Alphabet Symbols

α	thermal diffusivity	m^2/s
Δ	prefix to indicate a change	-
γ	specific heat ratio	-
ε	regenerator effectiveness	-
η_{th}	thermal efficiency	-
θ	crank angle (maximum engine volume occurs at $\theta = 0$)	rad
$D\theta_2$	angle measured from right-side horizontal to displacer crank arm	rad
$P\theta_2$	angle measured from right-side horizontal to piston crank arm	rad
$D\theta_3$	angle measured from right-side horizontal to displacer connecting rod	rad
$P\theta_3$	angle measured from right-side horizontal to piston connecting rod	rad
μ	dynamic viscosity	$Pas = kg/ms$
ρ	density	kg/m^3
τ	torque	Nm
φ	phase angle between engine pressure variations and displacer motion	rad
ω	angular frequency	rad/s

Subscripts

0	value at a reference point	-
b	buffer space (crankcase)	-
c	compression space	-
ck	compression space-cooler interface	-
$cycle$	pertaining to the engine cycle	-
C	thermal sink	-
CC	crankcase	-
CV	control volume	-
$disp$	displacer	-
e	expansion space	-
g	prefix to indicate a gas temperature	-
h	heater	-
he	heater-expansion space interface	-
h_1	hot gas inlet temperature for simple heat exchanger model	-
h_2	hot gas outlet temperature for simple heat exchanger model	-
H	thermal source	-
HEX	heat exchanger	-
i	place holder for engine component subscript	-
$ideal$	pertaining to an assumed ideal scenario	-
in	into the control volume	-
k	cooler	-
kr	cooler-regenerator interface	-
k_1	cold gas inlet temperature for simple heat exchanger model	-
k_2	cold gas outlet temperature for simple heat exchanger model	-
max	maximum value	-
min	minimum value	-
net	net value considering inflow and outflow	-
new	newly calculated value in iterative loop	-
old	previously calculated value in iterative loop	-

out	out of the control volume	-
$pist$	piston	-
r	regenerator	-
rej	rejection	-
rh	regenerator-heater interface	-
$shaft$	shaft work or power after all losses are included	-
w	prefix to indicate a wall temperature	-
x	along the x-direction	-

Chapter 1. Introduction and Literature Review

The following chapter begins with a brief discussion of the motivation for this research. Next, background information relevant to this thesis is explained. This includes a targeted description of Stirling engine operating principles, types, and mathematical models. The open literature is then surveyed regarding three main areas: Second order mathematical models, the ST05G Stirling engine, and low temperature difference Stirling engines. The chapter concludes by outlining the remainder of the thesis.

1.1 Motivation

Global concern for the health of the environment, coupled with rising fuel costs, are currently motivating research in electricity production from unconventional sources. An abundant, but largely untapped, energy resource is heat at temperatures below 232 °C [1]. Sources of this low-grade heat include industrial process waste heat [1], flat plate solar collectors, and geothermal reservoirs.

To utilize low grade heat resources for electricity production, a heat engine which can run on a low thermal source temperature is required. Since heat engines operating on small temperature differences are inherently inefficient, candidate engines must be inexpensive to build and maintain, to be commercially viable [1]. A modular design, which can be sized to match a variety of heat sources, is also desirable.

The Stirling engine is a closed cycle, externally heated, reciprocating engine, with attributes that make it a potential candidate for distributed electricity production from low grade heat sources [2]. Stirling engines typically use non-toxic, inert working fluids and may be pressurized to increase their power density [3]. With no valves, pumps, or turbines, the Stirling is relatively simple compared to other externally heated engines such as the Rankine or Ericsson. A group of Stirling units may be mechanically coupled together to create an engine of any desired power output; however, above a threshold power the multi-unit Stirling engine will be equal in complexity to its competitors, and thus lose its main advantage.

Competing technologies for low grade heat recovery include the organic Rankine cycle and the thermoelectric generator. Current technology for the organic Rankine cycle is suitable for electrical power outputs in the range of 200 – 2 000 MW [4]. This leaves room for application of the Stirling engine to smaller scale low grade heat recovery systems. Thermoelectric generators have the advantage of simplicity over the Stirling engine, as they have no moving parts; however, their high cost and relatively low efficiency make them commercially unfeasible for most applications [5].

To date, most of the Stirling engine literature deals with engines built for source temperatures above 500 °C. It is unclear whether theory developed for high temperature engines will be directly applicable to low source temperature designs. Understanding the differences between

high and low thermal source temperature Stirling engines will allow knowledge developed for high temperature designs to be applied to low temperature difference designs with confidence.

1.2 Stirling Engine Operating Principles

The following section is based on the explanation of Stirling engine operation given by West [6].

Consider the device depicted in Figure 1.1 below. A reciprocating element called the displacer separates a hot expansion space from a cold compression space. The cylinder walls of the expansion space are maintained at a high temperature relative to the walls of the compression space, which are maintained at a low temperature. A pressure gauge shows the current pressure of the gaseous working fluid in this sealed system. A narrow duct provides a connection between the compression and expansion spaces, and will allow gas to pass between them if the displacer is moved.

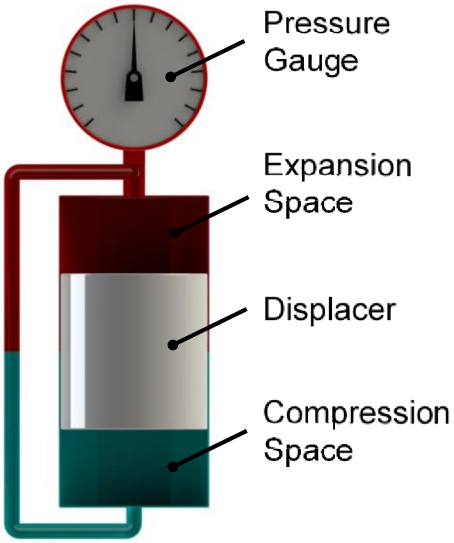
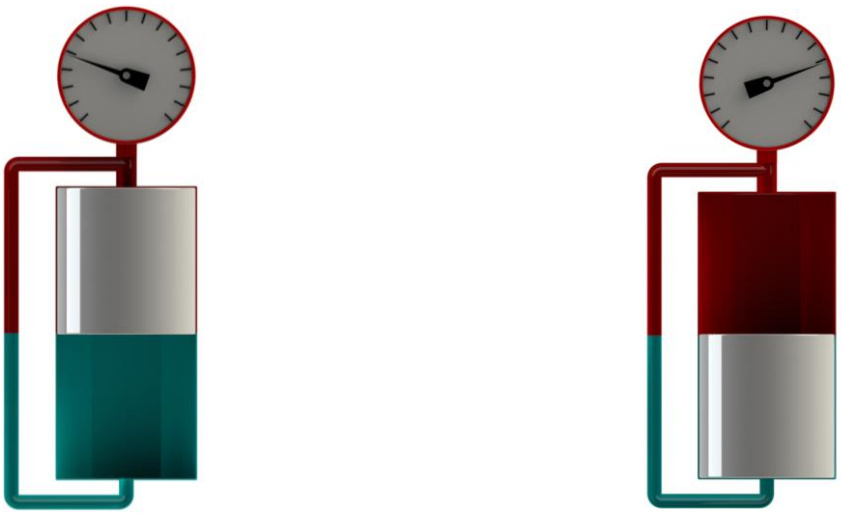


Figure 1.1: CAD Model of Displacer Cylinder, after West

In Figure 1.2, the effect of moving the displacer is illustrated. With most of the working fluid displaced to the cold compression space, as shown in Figure 1.2 (a), the average temperature of the working fluid is decreased relative to the case of Figure 1.1. Since the volume has not changed and the system is sealed, the system pressure decreases as indicated by the pressure gauge. Figure 1.2 (b) displays the inverse situation, in which the pressure has increased due to an increase of the average temperature of the working fluid. Heat has been transferred through the cylinder walls to bring about these average temperature changes.



(a) (b)

Figure 1.2: CAD Model of a Displacer Cylinder (a) Working Fluid Displaced to the Compression Space (b) Working Fluid Displaced to the Expansion Space

Although this device is an interesting way to observe ideal gas behavior, it is not an engine. True, reciprocating the displacer will move heat through the device, but since there is no change in volume, no work can be extracted. In fact, a small amount of work must be expended to overcome mechanical friction between the displacer and the cylinder, and fluid friction in the connecting duct.

To produce an engine, a piston and power cylinder are added, as presented in Figure 1.3 below. The piston separates the working space from the buffer space, which is usually the open atmosphere or a pressurized crankcase. Motion of the piston changes the volume of the working space, allowing work to be transferred to and from the engine.

With the displacer held stationary, and infinite heat transfer capability assumed for the displacer cylinder walls, moving the piston back and forth will isothermally expand and compress the working fluid. The pressure of the system will rise and fall proportionally with the changing volume, but no net work will be done in a complete cycle of the piston. Any work extracted from the system will need to be returned to bring the piston back to its original position.

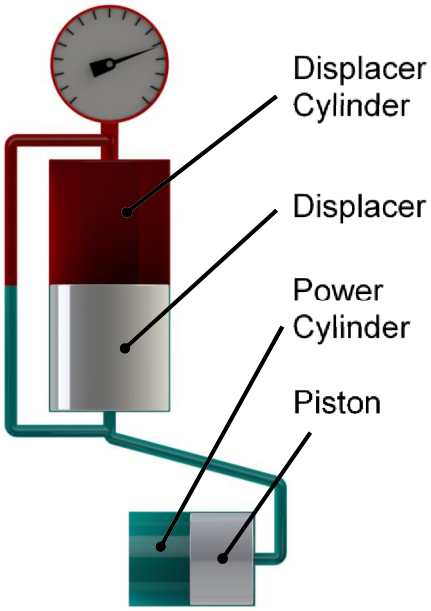


Figure 1.3: CAD Model of a Displacer Cylinder and Power Cylinder

To operate as an engine, the components shown in Figure 1.3 move through the cycle shown in Figure 1.4 below. This is known as the ideal Stirling cycle. Since all four processes are reversible if carried out at infinitesimal speed in the absence of friction, the ideal Stirling cycle has the same efficiency as the Carnot cycle. This is the textbook definition, and it offers little insight regarding the operation of real Stirling machines. The realities of friction, finite time, and finite heat transfer negate any aspiration of reaching the Carnot efficiency.

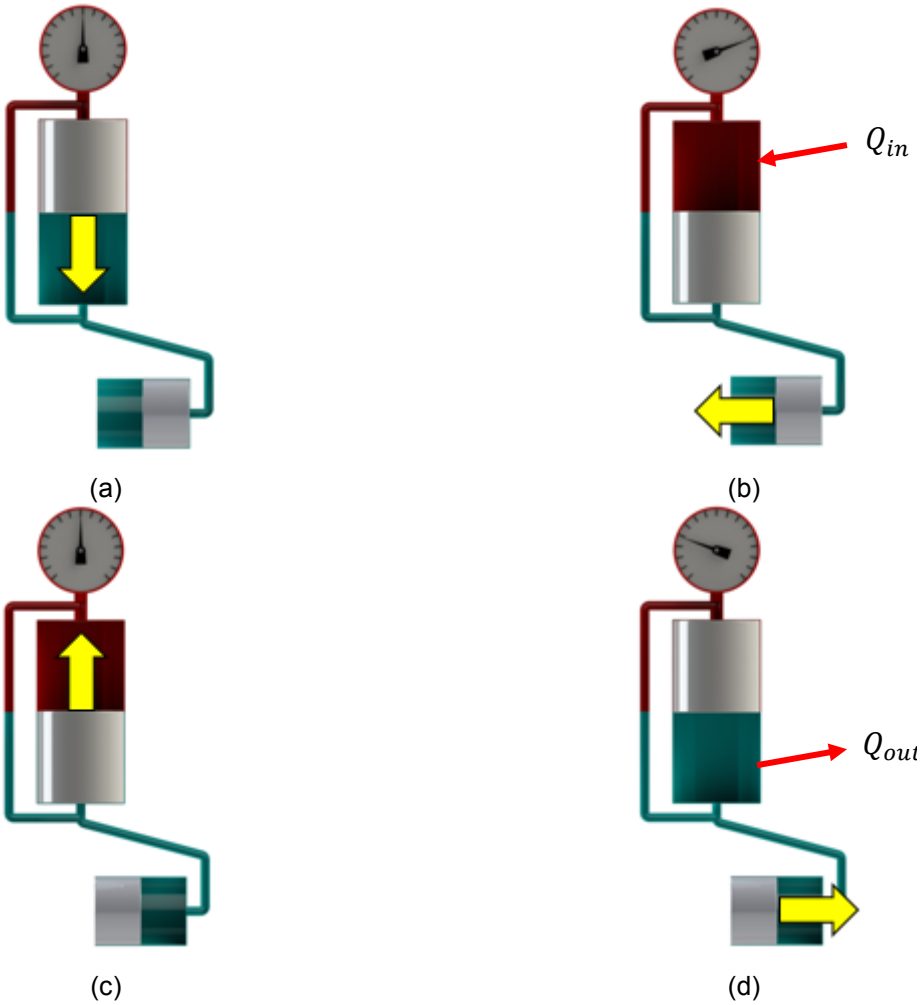


Figure 1.4: Four Processes Comprising the Ideal Stirling Cycle (a) Isochoric Heat Addition (b) Isothermal Expansion (c) Isochoric Heat Rejection (d) Isothermal Compression

In practice, the cylinder walls make very poor heat transfer surfaces, because their surface area to volume ratio is low. Dedicated heat exchangers are typically added to Stirling machines to move heat through the engine more quickly. These are shown in Figure 1.5. More effective heat exchangers allow a faster running speed and hence, a greater power output. The heater conveys heat from the thermal source to the working fluid, while the cooler conveys heat from the working fluid to the thermal sink.

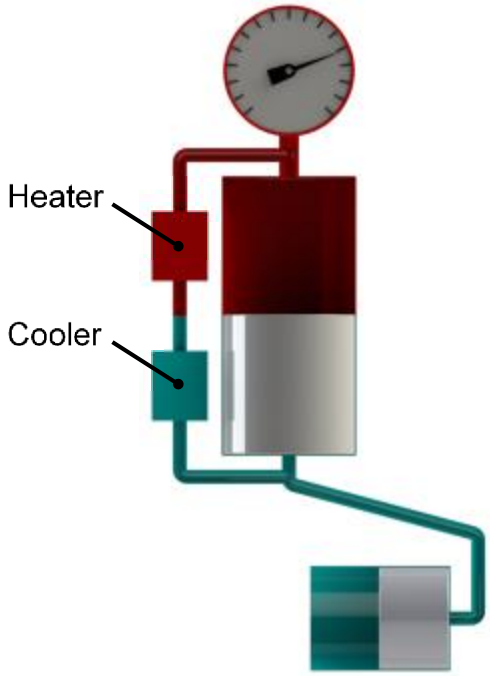


Figure 1.5: CAD Model of Displacer Cylinder, Power Cylinder, and Heat Exchangers

The optimal heat exchanger design will weigh the heat transfer benefit against the negative effects of flow friction and internal volume. Typical Stirling engine heat exchanger geometries include slots, tubes, annular gaps, and flat plates. Since the fluid properties and the heat transfer requirements of the heater and cooler are different, they are often geometrically different from each other.

Another component that is often added to real Stirling machines is the regenerator. It is placed between the heater and the cooler as indicated by Figure 1.6. The regenerator absorbs heat as the working fluid passes through it on its journey from the expansion space to the compression space. This reduces the amount of heat that must be rejected in the cooler to bring the working fluid down to the compression space temperature. The stored heat is returned to the working fluid as it moves back into the expansion space, reducing the load on the heater.

If the heater and cooler alone are effective enough to change the temperature of the working fluid to the desired limits, then the addition of a regenerator will increase the efficiency of the engine by decreasing the heat transferred between the thermal source and sink, but decrease the power output by adding flow friction. More often, the heat exchangers alone are not enough to

change the temperature of the working fluid as desired. In this case, adding a regenerator can increase both the power and the efficiency of the engine.

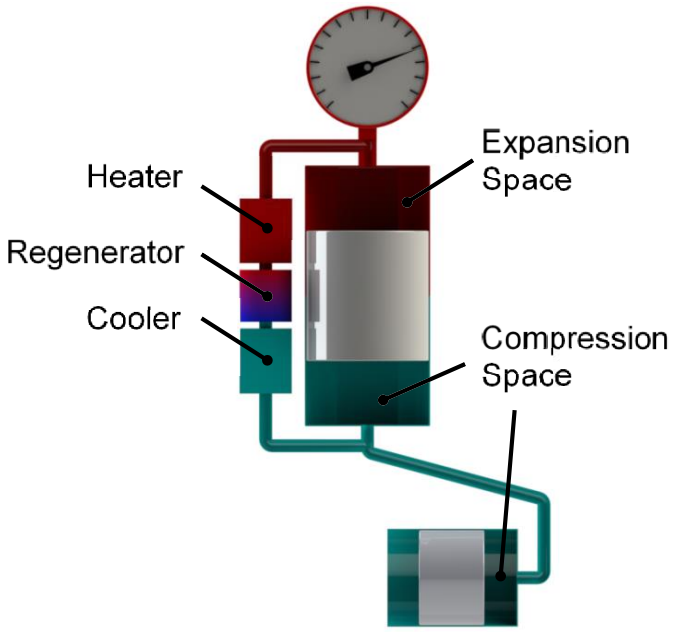


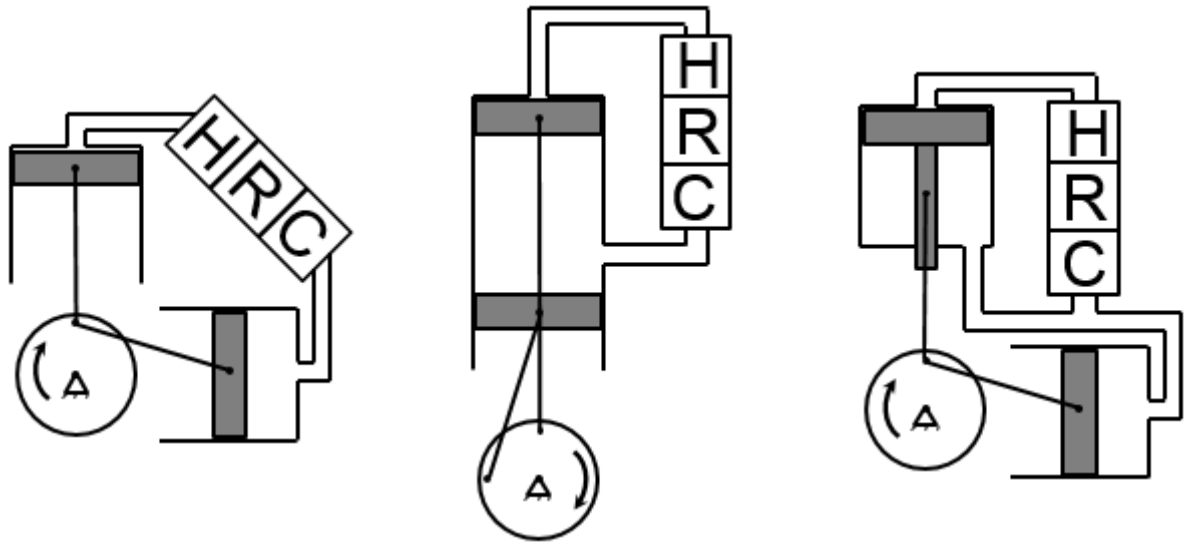
Figure 1.6: CAD Model of Five Component Stirling Machine

Like the heater and cooler, thorough regenerator design will consider the effects of heat transfer/storage, flow friction, and internal volume on the engine. Examples of regenerator geometries include densely packed fine wire mesh, randomly stacked fibers, stacked sintered wire screens, and layered foil.

Real Stirling engines also need some method of moving the piston and the displacer to drive the working fluid through an approximate Stirling cycle. For kinematic Stirling engines, this is done using a drive mechanism. A rich variety of drive mechanisms have been explored in the literature. Nearly all of them stray from the ideal cycle, by imparting continuous, near-sinusoidal motion onto the piston and displacer.

1.3 Types of Stirling Engines

Stirling engines are generally classified into three categories based on the arrangement of their components. A schematic of each type is given in Figure 1.7 below.



(a) (b) (c)

Figure 1.7: Three Main Stirling Engine Layouts (a) Alpha Type (b) Beta Type (c) Gamma Type

Both the beta and gamma types employ a piston and a displacer to move the working fluid through its cycle. The example engine used in previous figures would be considered a gamma type Stirling machine, because the displacer and the piston are in separate cylinders. In beta type engines, the piston and displacer share a common cylinder.

Alpha engines differ from betas and gammas in that they have two pistons rather than a piston and a displacer. The two pistons move together to displace the gas, and they move individually to expand and compress the gas. Alphas have the advantage of simpler flow geometry; however, they also have the disadvantage of a high temperature seal in the expansion space. Betas and gammas can easily avoid this high temperature seal by using a long displacer with the seal at the cold end. Alpha units can also be arranged to form a double acting engine, with a Stirling cycle running on either side of each piston.

Betas simplify manufacturing by requiring only one cylinder. The piston and displacer strokes often overlap, using the engine volume efficiently. Fitting the displacer drive rod through the piston in a low friction leak-free way presents a challenge for beta designs. For low

temperature difference engines, in which the displacement process involves a much greater volume of gas than the volume change process, betas can become impractical, because the displacer stroke becomes much larger than the piston stroke.

Since gamma machines have separate cylinders for the piston and the displacer, the bore of each need not be the same. This simplifies the mechanism for low temperature difference engines.

1.4 Model Classification/Overview

The next section reviews the Stirling engine modeling approaches available in the literature. The first sub-section states the conservation laws on which Stirling engine models are based, and relates them to the categories in which Stirling engine models are typically classified. Models relevant to this research are then explained in further detail.

1.4.1 Conservation Laws

While Stirling engine models range in scope and complexity, they are built using a common set of tools: Conservation laws for mass, momentum, and energy, and an equation of state for the working fluid [7]. The one dimensional mathematical statements for each of these are given below for future reference [7]. For the conservation laws, both the engineering thermodynamics (Eulerian), and the flow field (Lagrangian) descriptions are given. The most complicated Stirling engine models include all three conservation laws, an equation of state, and a statement of conservation of energy for the engine walls [7]. The simplest models use only mass conservation, and an equation of state [7].

Equations (1.1a) and (1.1b) are expressions of mass conservation in Eulerian and Lagrangian reference frames, respectively.

$$\sum_{in} \dot{m} - \sum_{out} \dot{m} = \frac{dm}{dt} \tag{1.1a}$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) = 0 \tag{1.1b}$$

Expressions of momentum conservation are given below in Eqs. (1.2a) and (1.2b). Equation (1.2a) is the Eulerian description commonly used in engineering thermodynamics, while Eq. (1.2b) is the Lagrangian description used in fluid mechanics. Equation (1.2b) is known as the one-dimensional Navier-Stokes equation.

$$\sum_{x} F + \sum_{in} (\dot{m} u_x) - \sum_{out} (\dot{m} u_x) = \frac{\partial}{\partial t}(M u_x)_{CV} \tag{1.2a}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{1}{\rho}\frac{\partial p}{\partial x} - g_x - \frac{\mu}{\rho}\frac{\partial^2 u}{\partial x^2} = 0 \tag{1.2b}$$

The law of conservation of energy is listed below. Equation (1.3a) gives the engineering thermodynamics interpretation and Eq. (1.3b) gives the flow field description.

$$\sum_{in} \dot{m}\left(h + \frac{1}{2}u^2 + gz\right) - \sum_{out} \dot{m}\left(h + \frac{1}{2}u^2 + gz\right) + \dot{Q} - \dot{W} = \frac{\partial E}{\partial t} \tag{1.3a}$$

$$\frac{\partial}{\partial t}\left[\rho\left(c_v T + \frac{1}{2}u^2\right)\right] - \dot{q}\rho = -\frac{\partial}{\partial x}\left[\rho u\left(c_v T + \frac{p}{\rho} + \frac{1}{2}u^2\right)\right] \tag{1.3b}$$

Finally, Eq. (1.4) gives the ideal gas equation of state, commonly used in Stirling engine analysis.

$$pV = mRT \tag{1.4}$$

1.4.2 Classification of Models

Martini [8], and Chen and Griffin [9] classified Stirling engine modeling approaches in the early 1980s. Their distinctions are in common use today. First order models can be solved without the aid of a computer and require few details of the specific design [8]. They are semi-empirical, and are typically used only to get a first-cut understanding of the capability of a proposed Stirling engine [9]. Examples include the Beale and West number correlations, and the Schmidt model derated by an empirical loss factor. Second order models begin by assuming that the losses present in real Stirling engines can be calculated individually, and algebraically added to the results of an idealized reference cycle to give the final power and efficiency [8], [9]. Common reference cycles are the ideal isothermal model, and the ideal adiabatic model [10]. Third order models begin by discretizing the engine into a series of control volumes or nodes [9]. The one dimensional conservation laws are then solved numerically, often with further simplifications [8]. This approach is more computationally expensive than the second order approach, but accounts for the interactions between loss mechanisms more faithfully [8].

Scaling using dynamic similarity has been proposed by Organ as an alternative design approach for Stirling engines [7], [11], [12]. In this method, the variables governing the Stirling

engine problem are combined into dimensionless groups using the Buckingham-Pi theorem. Use of the Buckingham-Pi theorem for Stirling engines is also demonstrated clearly by Prieto [13]. By keeping the dimensionless groups constant, it is possible to scale an engine of known performance, called the prototype, into a derivative engine of a different physical size, or working fluid [7], [11], [12]. Strict dynamic similarity requires perfect geometric similarity, i.e. all linear dimensions of the prototype are multiplied by the same factor to obtain the linear dimensions of the derivative [7], [11], [12].

1.4.3 Beale and West Number Correlations

The Beale and West number correlations are based on trends observed in performance data collected for a group of engines [6]. They are both considered first order models [8].

The Beale number was determined by plotting the power outputs of a group of engines as a function of the product of their mean cycle pressure, p_{mean}, frequency, f, and piston swept volume, V_{swp} [6]. Most of the engines chosen had heater temperatures of ~650 °C [6]. A line of best fit was established for the plot, and the slope of that line was called the Beale number [14]. The Beale number of an engine is calculated based on measured performance using

$$N_B = \frac{\dot{W}_{shaft}}{f V_{swp} p_{mean}} \tag{1.5}$$

Note that the Beale number depends on the operating point chosen. In SI units, the Beale number based on the original group of engines, is 0.15 [6]. Using this value, the potential shaft power of an engine can be roughly estimated for a given combination of frequency, piston swept volume, and mean pressure.

West suggested an additional factor for the correlation, to account for engines having different source and sink temperatures [6]. Applying this factor yields

$$N_W = N_B \frac{(T_H + T_C)}{(T_H - T_C)} \tag{1.6}$$

The West number, N_W, in SI units was determined to be 0.25 using the same group of engines as used for the Beale number [6].

The lowest source temperature of the engines used in either of the correlations was 340 °C. Their applicability to engines of lower source temperatures is questionable.

1.4.4 Reference Cycles

Second order modeling approaches, by definition, rely on reference cycles to determine a baseline power and efficiency, from which individually calculated losses may be subtracted. The two most commonly used reference cycles are the ideal isothermal model, and the ideal adiabatic model described by Urieli and Berchowitz [10].

1.4.4.1 Ideal Isothermal Model [10]

The ideal isothermal model considers a five-component model of the engine, as shown in Figure 1.8 below. From left to right the components are called the compression space, cooler, regenerator, heater, and expansion space. It is assumed that each component is maintained at a constant temperature during the cycle. This implies infinite heat transfer rates are possible or that speed is infinitesimal. The control volume formulations for conservation of mass and energy, and the ideal gas equation of state are applied to calculate pressure, work, heat transfer, and efficiency. The following assumptions are made [10]:

1 Heat exchangers and regenerator have infinite convective heat transfer coefficients and negligible flow friction.
2 Constant pressure spatially throughout the engine. For a given engine geometry, working fluid and operating point, pressure is a function of crank angle only.
3 Perfect sealing. No mass transfer across sealing surfaces.
4 The working fluid behaves as an ideal gas.
5 Engine is at cyclic steady state. Each complete cycle matches the previous one.
6 The kinetic and potential energies of the working fluid are negligible.
7 The working fluid has constant specific heats.

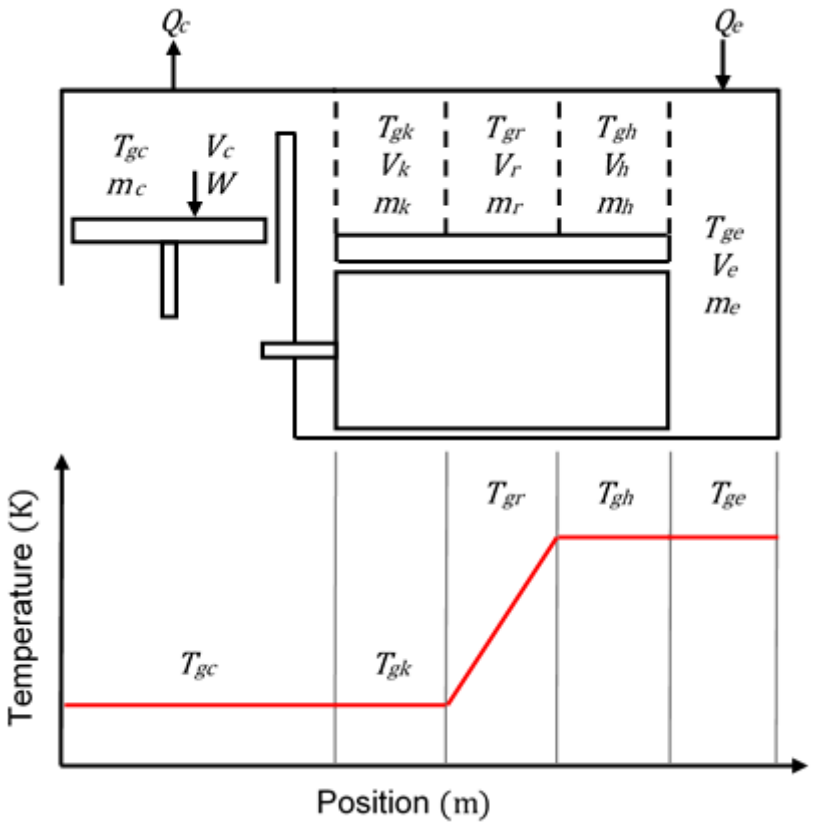


Figure 1.8: Schematic and Temperature Profile for the Ideal Isothermal Model

The following ideal gas relations for enthalpy and internal energy will be used in the derivation.

$$dh = c_p(T)dT \cong c_p dT \qquad du = c_v(T)dT \cong c_v dT \tag{1.7}$$

The derivation begins with a statement of mass balance.

$$M = m_c + m_k + m_r + m_h + m_e \tag{1.8}$$

Substituting the ideal gas law, Eq (1.4), and isolating the pressure yields

$$p = MR\left(\frac{V_c(\theta)}{T_{gk}} + \frac{V_k}{T_{gk}} + \frac{V_r}{T_{gr}} + \frac{V_h}{T_{gh}} + \frac{V_e(\theta)}{T_{gh}}\right)^{-1} \tag{1.9}$$

where $V_c(\theta)$ and $V_e(\theta)$ are the crank angle dependent volumes of the compression and expansion spaces and T_r is the regenerator mean effective temperature. Assuming a linear temperature profile in the regenerator, the regenerator mean effective temperature can be shown to be

$$T_{gr} = \frac{T_{gh} - T_{gk}}{\ln\left(\frac{T_{gh}}{T_{gk}}\right)} \tag{1.10}$$

which takes the variation of density in the regenerator into account [10].

Since the heater, cooler, and regenerator have fixed volumes, work can only be transferred through the compression and expansion spaces. The total work is the sum of the work done in these two spaces, i.e.

$$W = W_c + W_e = \oint p dV_c(\theta) + \oint p dV_e(\theta) = \oint p\left(\frac{dV_c(\theta)}{d\theta} + \frac{dV_e(\theta)}{d\theta}\right) d\theta \tag{1.11}$$

To determine the heat transfer during the cycle, consider the control volume formulation of conservation of energy, Eq. (1.3a). Assuming kinetic and potential energies of the working gas are negligible, and realizing that the engine components have at most one inlet and one outlet, Eq. (1.3a) reduces to

$$\dot{m}_{in} h_{in} - \dot{m}_{out} h_{out} + \dot{Q} - \dot{W} = \frac{\partial U}{\partial t}$$

Substituting the ideal gas relations for internal energy and enthalpy, Eqs. (1.7), yields

$$\dot{m}_{in} c_{p,in} T_{g,in} - \dot{m}_{out} c_{p,out} T_{g,out} + \dot{Q} - \dot{W} = c_v \frac{\partial (m_{net} T_{g,CV})}{\partial t}$$

In the heat exchangers and working spaces, the temperature is constant, so

$$T_{g,in} = T_{g,out} = T_{g,CV} = T$$

$$c_{p,in} = c_{p,out}$$

and

$$c_v \frac{\partial (m_{net} T)}{\partial t} = c_v T \frac{\partial m_{net}}{\partial t} = c_v T \dot{m}_{net}$$

These allow further simplification of the conservation of energy equation to

$$c_p T(\dot{m}_{in} - \dot{m}_{out}) + \dot{Q} - \dot{W} = c_v T \dot{m}_{net}$$

Solving for $\dot{Q}$ gives

$$\dot{Q} = c_v T \dot{m}_{net} - c_p T(\dot{m}_{in} - \dot{m}_{out}) + \dot{W}$$

Noting that

$$\dot{m}_{net} = \dot{m}_{in} - \dot{m}_{out} = \dot{m}$$

The equation for $\dot{Q}$ can be written as

$$\dot{Q} = \dot{m} T(c_v - c_p) + \dot{W}$$

or

$$\dot{Q} = -\dot{m} R T + \dot{W}$$

since for an ideal gas

$$c_v - c_p = -R$$

Of interest, is the net heat transferred over a complete cycle. To calculate this, the simplified conservation of energy equation must be integrated.

$$Q = \oint \dot{Q} = -RT \oint \dot{m} + \oint \dot{W}$$

Since the engine is assumed to be at cyclic steady state, there is no net mass transfer over a complete cycle,

$$\oint \dot{m} = 0$$

and what remains is

$$Q = \oint \dot{W} = W$$

So, in the compression and expansion spaces

$$Q_c = W_c \qquad \text{and} \qquad Q_e = W_e$$

Since no work is done in the heat exchangers,

$$Q_h = Q_k = 0$$

The ideal isothermal model equation set is summarized in Table 1.1 below. Note that the efficiency for the ideal isothermal model is equal to the Carnot efficiency, because all losses have been ignored, and isothermal expansion and compression processes are reversible. Volumes unswept by the piston and displacer are commonly referred to as "dead volume" and are typically kept to a minimum when designing Stirling machine heat exchangers. Inspection of the pressure equation below reveals that if the volumes of the heat exchangers and regenerator are large relative to the swept volumes of the compression and expansion spaces, the pressure variation amplitude will be low. Also, since component volumes are divided by their corresponding temperatures, dead volume in cold components of the engine will have a greater influence than dead volume in hot components. Experiments have identified operating conditions under which an increase in dead volume leads to an increase in indicated power [12]. The ideal isothermal model is blind to this phenomenon.

Table 1.1: Ideal Isothermal Model Equation Set [10]

$p(\theta) = MR\left(\frac{V_c(\theta)}{T_{gk}} + \frac{V_k}{T_{gk}} + \frac{V_r \ln\left(\frac{T_{gh}}{T_{gk}}\right)}{T_{gh} - T_{gk}} + \frac{V_h}{T_{gh}} + \frac{V_e(\theta)}{T_{gh}}\right)^{-1}$	Pressure
$Q_c = W_c = \oint p \frac{dV_c}{d\theta} d\theta$	Heat Transferred
$Q_e = W_e = \oint p \frac{dV_e}{d\theta} d\theta$	
$W = W_c + W_e$	Work Done
$\eta = \frac{W}{Q_e} = 1 - \frac{T_{gc}}{T_{gk}}$	Efficiency

For the special case of sinusoidal volume variations, the ideal isothermal model can be integrated analytically to give a closed form solution. This analysis was first performed by Gustav Schmidt in 1871 [15] and now bears his name. The volume variations differ for the three engine layouts. The Schmidt analysis multiplied by a loss factor of 0.3 to 0.4 has been used as a first order model [8].

The assumptions used in the derivation of the ideal isothermal model are unrealistic for most real Stirling engines. A direct result of the isothermal model is that all heat transfer occurs in the cylinders. If this were the case, the heat exchangers would be redundant [10]. In most Stirling machines, the cylinders are not designed for heat transfer. Cylinders typically have a low surface area to volume ratio, and expansion and compression processes are typically very short. For example, the ST05G Stirling engine expansion space has a surface area to volume ratio of 0.208 cm^{-1}, compared to 22.0 cm^{-1} in the cooler, and at 500 RPM, the expansion process happens in 0.06 s. Thus, in most Stirling machines, adding heat exchangers is an improvement over relying on heat transfer in the cylinders. The ideal isothermal model also ignores important losses that occur in real engines; hence, alone, it gives an incomplete picture of Stirling engine operation.

1.4.4.2 Ideal Adiabatic Model [10]

The ideal adiabatic model is thought to be a more realistic reference cycle for most practical Stirling engines [10]. The same five component model is considered; however, the compression and expansion spaces are assumed to be adiabatic, so their temperatures vary with the engine pressure. The schematic and temperature profile for the ideal adiabatic model are shown in Figure 1.9. Again, the ideal gas equation of state, and the conservation laws for mass and energy are applied to produce the equation set. The result is a system of ordinary differential equations that must be solved numerically. Other than adiabatic working spaces, the ideal adiabatic model begins with the same assumptions as the ideal isothermal model.

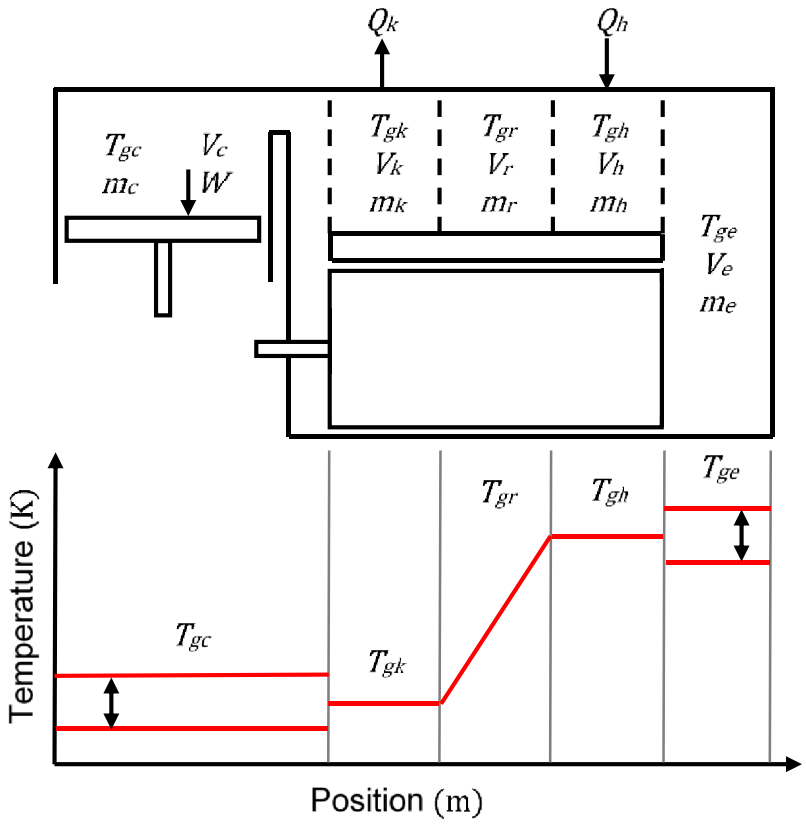


Figure 1.9: Schematic and Temperature Profile for the Ideal Adiabatic Model

First, conditional temperatures are defined for the inlets to the compression and expansion spaces.

$$T_{gck} = \begin{cases} T_{gc}(\theta) & \frac{dm_{ck}}{d\theta} > 0 \\ T_{gk} & \frac{dm_{ck}}{d\theta} \leq 0 \end{cases} \qquad T_{ghe} = \begin{cases} T_{gh} & \frac{dm_{he}}{d\theta} \geq 0 \\ T_{ge}(\theta) & \frac{dm_{he}}{d\theta} < 0 \end{cases} \tag{1.12}$$

These temperatures depend on the direction of flow. The interface temperatures are equal to the upstream temperatures.

The derivative of the ideal gas equation of state with respect to time will also be needed. It is calculated as follows

$$\frac{\partial}{\partial t}\{pV = mRT\}$$

$$\frac{\partial}{\partial t}\{ln(pV) = ln(mRT)\}$$

$$\frac{\partial}{\partial t}\{ln(p) + ln(V) = ln(m) + ln(R) + ln(T)\}$$

$$\frac{1}{p}\frac{\partial p}{\partial t} + \frac{1}{V}\frac{\partial V}{\partial t} = \frac{1}{m}\frac{\partial m}{\partial t} + \frac{1}{T}\frac{\partial T}{\partial t} \tag{1.13}$$

The following ideal gas relations will also be used

$$R = c_p - c_v \qquad \gamma = \frac{c_p}{c_v} \qquad c_p = \frac{R\gamma}{(\gamma - 1)} \qquad c_v = \frac{R}{(\gamma - 1)} \tag{1.14}$$

Consider a statement of conservation of mass for the engine, as in the isothermal model.

$$M = m_c + m_k + m_r + m_h + m_e$$

This time, take the derivative of the mass equation with respect to time

$$0 = \dot{m}_c + \dot{m}_k + \dot{m}_r + \dot{m}_h + \dot{m}_e \tag{1.15}$$

For the heat exchangers and regenerator, the volumes and temperatures have been assumed to be constant. This allows simplification of Eq. (1.13) to

$$\frac{1}{p}\frac{\partial p}{\partial t} = \frac{1}{m}\frac{\partial m}{\partial t}$$

$$\frac{\partial m}{\partial t} = \dot{m} = \frac{m}{p}\frac{\partial P}{\partial t} = \frac{V}{RT}\frac{\partial p}{\partial t}$$

$$\dot{m} = \frac{V}{RT}\frac{\partial p}{\partial t} \tag{1.16}$$

Substituting Eq. (1.16) into Eq. (1.15) generates

$$0 = \dot{m}_c + \frac{1}{R}\frac{\partial p}{\partial t}\left(\frac{V_k}{T_{gk}} + \frac{V_r}{T_{gr}} + \frac{V_h}{T_{gh}}\right) + \dot{m}_e \tag{1.17}$$

Next, the conservation of energy is applied to the compression and expansion spaces. Since it is assumed that there is no heat transfer in these spaces, and they each only have one inlet/outlet, Eq. (1.3a) reduces to

$$\dot{m}c_p T - \dot{W} = c_v \frac{\partial (mT)}{\partial t} \tag{1.18}$$

Using the ideal gas relations and the definition of the rate of work, and simplifying, this becomes

$$\frac{dm}{d\theta} = \frac{1}{RT}\left(p\frac{dV}{d\theta} + \frac{V}{\gamma}\frac{dp}{d\theta}\right) \tag{1.19}$$

where the time derivatives have been changed to theta derivatives by multiplying through by $\frac{dt}{d\theta}$. Substituting Eq. (1.19) into Eq. (1.17) and rearranging provides a differential equation for pressure as a function of crank angle.

$$\frac{dp}{d\theta} = \frac{-\gamma p(\theta)\left(\frac{1}{T_{gck}}\frac{dV_c}{d\theta} + \frac{1}{T_{ghe}}\frac{dV_e}{d\theta}\right)}{\left(\frac{V_c(\theta)}{T_{gck}} + \gamma\left(\frac{V_k}{T_{gk}} + \frac{V_r}{T_{gr}} + \frac{V_h}{T_{gh}}\right) + \frac{V_e(\theta)}{T_{ghe}}\right)} \tag{1.20}$$

To calculate the mass accumulation and mass flow for each component, the following equations, based on conservation of mass, are used

$$\frac{dm_c}{d\theta} = -\frac{dm_{ck}}{d\theta} \tag{1.21}$$

$$\frac{dm_k}{d\theta} = \frac{dm_{ck}}{d\theta} - \frac{dm_{kr}}{d\theta} \tag{1.22}$$

$$\frac{dm_r}{d\theta} = \frac{dm_{kr}}{d\theta} - \frac{dm_{rh}}{d\theta} \tag{1.23}$$

$$\frac{dm_h}{d\theta} = \frac{dm_{rh}}{d\theta} - \frac{dm_{he}}{d\theta} \tag{1.24}$$

$$\frac{dm_e}{d\theta} = \frac{dm_{he}}{d\theta} \tag{1.25}$$

Note that mass flows towards the expansion space are considered positive.

The differential equation for the work done by the engine is

$$\frac{dW}{d\theta} = p\frac{dV_c}{d\theta} + p\frac{dV_e}{d\theta} \tag{1.26}$$

To calculate the heat transferred in the heat exchangers and regenerator, the conservation of energy equation is applied. Since no work is done in these spaces, and kinetic and potential energy are neglected, Eq. (1.3a) simplifies to

$$\dot{m}_{in}c_{p,in}T_{in} - \dot{m}_{out}c_{p,out}T_{out} + \dot{Q} = c_v\frac{\partial(m_{net}T)}{\partial t} \tag{1.27}$$

Using the ideal gas equation and realizing that the volume is constant in the heat exchangers and regenerator, the right side of the equation may be rewritten as

$$c_v\frac{\partial(m_{net}T)}{\partial t} = c_v\frac{\partial(\frac{pV}{RT}T)}{\partial t} = \frac{c_v}{R}\frac{\partial(pV)}{\partial t} = \frac{c_v}{R}\frac{\partial p}{\partial t}V + \frac{c_v}{R}\frac{\partial V}{\partial t}p = \frac{c_v}{R}\frac{\partial p}{\partial t}V \tag{1.28}$$

Applying Eq. (1.28) to Eq. (1.27), isolating the heat transfer variable, and writing in terms of crank angle rather than time gives the following heat transfer equations for the cooler, regenerator, and heater.

$$\frac{dQ_k}{d\theta} = c_v\frac{V_k}{R}\frac{dp}{d\theta} - c_p\left(\frac{dm_{ck}}{d\theta}T_{gck} - \frac{dm_{kr}}{d\theta}T_{gkr}\right) \tag{1.29}$$

$$\frac{dQ_r}{d\theta} = c_v\frac{V_r}{R}\frac{dp}{d\theta} - c_p\left(\frac{dm_{kr}}{d\theta}T_{gkr} - \frac{dm_{rh}}{d\theta}T_{grh}\right) \tag{1.30}$$

$$\frac{dQ_h}{d\theta} = c_v\frac{V_h}{R}\frac{dp}{d\theta} - c_p\left(\frac{dm_{rh}}{d\theta}T_{grh} - \frac{dm_{he}}{d\theta}T_{ghe}\right) \tag{1.31}$$

where constant specific heats have been assumed, and the time derivatives have been converted to crank angle derivatives.

Finally, since the heat exchangers and regenerator have been assumed to be isothermal

$$T_{gkr} = T_{gk} \qquad\qquad T_{grh} = T_{gh} \tag{1.32}$$

The ideal adiabatic model equation set is summarized in Table 1.2 below. It consists of six ordinary differential equations and 15 algebraic equations. Urieli and Berchowitz [10] solved the differential equations simultaneously using the fourth-order Runge-Kutta method. The Runge-Kutta method requires that the system of equations is presented as an initial value problem. The method of Urieli and Berchowitz [10], is to set the initial compression and expansion space temperatures to the cooler and heater temperatures, respectively. The equations are then solved

for several complete cycles until the values of the variables at the beginning of the cycle match those at the end of the cycle, within a specified tolerance.

While the ideal adiabatic model offers more detailed information than the ideal isothermal model and is thought to be more realistic for most Stirling engines, it still ignores losses which degrade the power and efficiency of real engines. To form a complete second-order model, these losses must be quantified individually and added to the ideal cycle results.

Table 1.2: Ideal Adiabatic Model Equation Set

$\frac{dp}{d\theta} = \frac{-\gamma p(\theta)\left(\frac{1}{T_{gck}}\frac{dV_c}{d\theta} + \frac{1}{T_{ghe}}\frac{dV_e}{d\theta}\right)}{\left(\frac{V_c(\theta)}{T_{gck}} + \gamma\left(\frac{V_k}{T_{gk}} + \frac{V_r}{T_{gr}} + \frac{V_h}{T_{gh}}\right) + \frac{V_e(\theta)}{T_{ghe}}\right)}$			Pressure
$\frac{dm_c}{d\theta} = \frac{1}{RT_{gck}}\left(p\frac{dV_c}{d\theta} + \frac{V_c(\theta)}{\gamma}\frac{dp}{d\theta}\right)$			Masses
$m_k = \frac{pV_k}{RT_{gk}}$	$m_r = \frac{pV_r}{RT_{gr}}$	$m_h = \frac{pV_h}{RT_{gh}}$	
$m_e = M - (m_c + m_k + m_h + m_e)$			
$T_{gc} = \frac{pV_c}{Rm_c}$	$T_{ge} = \frac{pV_e}{Rm_e}$	$T_{gr} = \frac{T_{gh} - T_{gk}}{\ln\left(\frac{T_{gh}}{T_{gk}}\right)}$	Constant Temperatures
$T_{gck} = \begin{cases} T_{gc}(\theta) & \frac{dm_{ck}}{d\theta} > 0 \\ T_{gk} & \frac{dm_{ck}}{d\theta} \le 0 \end{cases}$	$T_{ghe} = \begin{cases} T_{gh} & \frac{dm_{he}}{d\theta} \ge 0 \\ T_{ge}(\theta) & \frac{dm_{he}}{d\theta} < 0 \end{cases}$		Conditional Temperatures
$\frac{dm_k}{d\theta} = \frac{m_k}{p}\frac{dp}{d\theta}$	$\frac{dm_r}{d\theta} = \frac{m_r}{p}\frac{dp}{d\theta}$	$\frac{dm_h}{d\theta} = \frac{m_h}{p}\frac{dp}{d\theta}$	Mass Accumulation and Mass Flows
$\frac{dm_{ck}}{d\theta} = -\frac{dm_c}{d\theta}$			
$\frac{dm_{kr}}{d\theta} = \frac{dm_{ck}}{d\theta} - \frac{dm_k}{d\theta}$			
$\frac{dm_{rh}}{d\theta} = \frac{dm_{kr}}{d\theta} - \frac{dm_r}{d\theta}$			
$\frac{dm_{he}}{d\theta} = \frac{dm_{rh}}{d\theta} - \frac{dm_h}{d\theta}$			
$\frac{dW}{d\theta} = p\left(\frac{dV_c}{d\theta} + \frac{dV_e}{d\theta}\right)$			Work and Heat Transfer
$\frac{dQ_k}{d\theta} = c_v\frac{V_k}{R}\frac{dp}{d\theta} - c_p\left(\frac{dm_{ck}}{d\theta}T_{gck} - \frac{dm_{kr}}{d\theta}T_{gkr}\right)$			
$\frac{dQ_r}{d\theta} = c_v\frac{V_r}{R}\frac{dp}{d\theta} - c_p\left(\frac{dm_{kr}}{d\theta}T_{gkr} - \frac{dm_{rh}}{d\theta}T_{grh}\right)$			
$\frac{dQ_h}{d\theta} = c_v\frac{V_h}{R}\frac{dP}{d\theta} - c_p\left(\frac{dm_{rh}}{d\theta}T_{grh} - \frac{dm_{he}}{d\theta}T_{ghe}\right)$			

1.5 Decoupled Losses

Losses are added to the heat input rate, power output, and heat rejection rate of the reference cycle to obtain more accurate results. Martini classified losses as either heat losses, which are added to the reference cycle heat input rate, or power losses which are subtracted from the reference cycle power output [8]. Urieli and Berchowitz point out that some losses can affect both the power output and the heat transfer rates [10]. For example, flow friction impedes the motion of the displacer leading to a power loss, but it also produces heat which affects the heat input and rejection rates. Calculation of decoupled losses, and methods for adding them to the reference cycle, is typically what differentiates second order models in the literature. The following subsections introduce the loss mechanisms that are typically accounted for in second order Stirling engine models.

1.5.1 Regenerator Enthalpy Loss

An ideal regenerator will heat the working fluid to the heater temperature on the cold blow, and cool the working fluid to the cooler temperature on the hot blow [10]. Real regenerators do not achieve these temperatures. The difference between the actual enthalpy change of the working fluid as it passes through the regenerator, and the theoretical maximum enthalpy change is called the regenerator enthalpy loss. A common method of calculating this loss is using the simple heat exchanger model, which will be described in detail below.

1.5.2 Conduction Loss

Ideally, all the heat supplied to a Stirling engine passes through the working fluid and contributes to the thermodynamic cycle before being rejected to the thermal sink. In reality, a portion of this heat bypasses the thermodynamic cycle entirely, conducting through the engine walls, the displacer, or the working fluid itself. These conduction paths are typically minimized in Stirling engine design, with focus on the regenerator housing and displacer, since they have the most severe thermal gradients [3].

1.5.3 Appendix Gap Loss

It is common for displacers to have an overall length greater than their stroke, with a seal at the cold end [6], [10]. This is done to reduce the material requirements for the displacer seal [6]. The resulting annular gap that surrounds the displacer is called the appendix gap.

Heat and mass transfer occurring in the appendix gap induce losses. Second order models typically consider the appendix gap loss to be comprised of two components: shuttle loss, and pumping loss [6], [10]. Shuttle loss is the conduction of heat from the hot cylinder walls of the expansion space to the cold cylinder walls of the compression space, enhanced by the motion of the displacer [6], [8], [10]. The pumping loss is associated with the gas flow in and out of the appendix gap driven by the changing engine pressure [6], [10]. An optimum appendix gap width exists at which the sum of these loss mechanisms is minimized [6], [10], [16], [17].

1.5.4 Mechanical Friction

Mechanical friction refers to friction within the bearings and seals of the drive mechanism, defined here as the device which transmits power from the working fluid to the output shaft, and imparts motion on the displacer and piston. Two methods of quantifying mechanical friction are prominent in the Stirling engine literature. The first forms a component of the finite speed thermodynamics treatment of the Stirling cycle introduced by Petrescu et al., and contains empirical factors that are not given [18]. The second was developed by James R. Senft and considers not only the friction inherent in the mechanism, but also the amount of energy transferred through the mechanism [19]–[27]. Senft's methods will now be explained, as extensive use of them is made throughout this thesis.

When energy is transmitted through the drive mechanism, a portion of it is lost due to friction. The ratio of the energy transmitted by the mechanism, to the energy supplied to the mechanism, is termed the mechanism effectiveness, E.

$$E = \frac{\text{energy transmitted by mechanism}}{\text{energy supplied to mechanism}} \tag{1.33}$$

The mechanism effectiveness depends on the loads and speeds of the mechanism components, which vary with the position of the mechanism; however, for slider-crank mechanisms common in Stirling engines, the mechanism effectiveness is approximately constant [20].

The pressure on the backside of the piston will be referred to as the buffer pressure. The net force on the piston is determined by considering both the workspace pressure and the buffer space pressure. In general, there are periods of the cycle in which the net force on the piston is opposite to its direction of travel. The work done by the piston in these scenarios is called forced

work. Areas of forced work for a Stirling engine cycle are highlighted in red on the indicator diagram in Figure 1.10 below. The black line represents the buffer pressure, and the blue loop represents the engine pressure. Total volume of the working space is plotted on the horizontal axis.

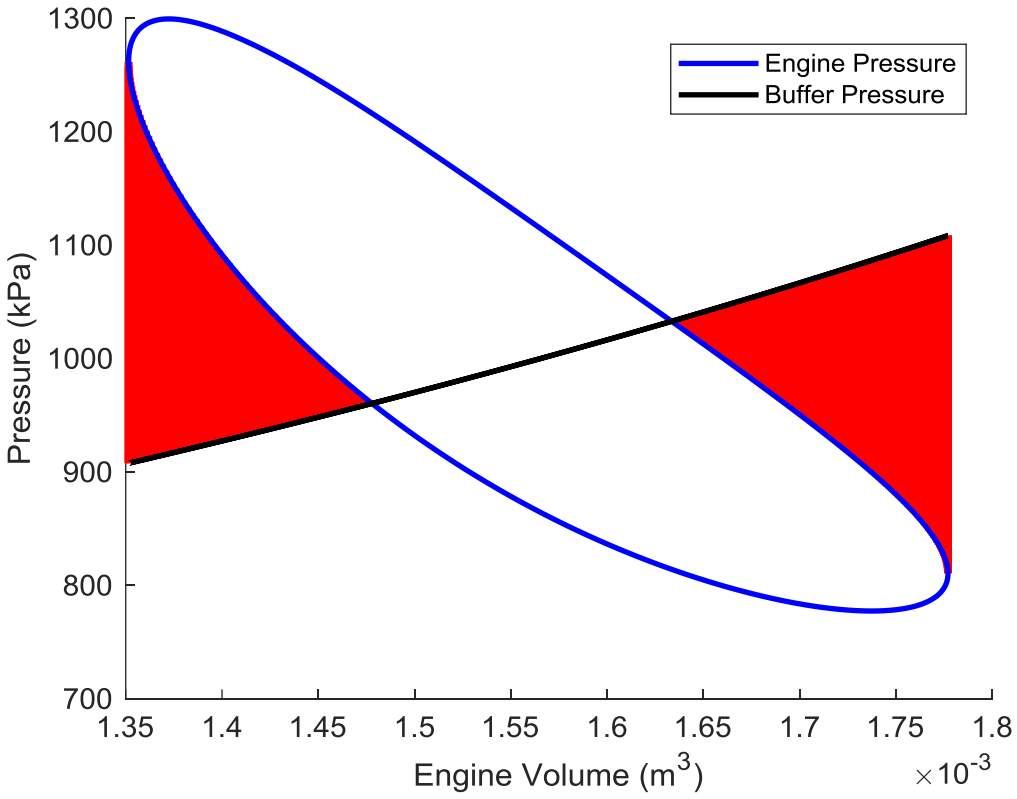


Figure 1.10: Indicator Diagram Showing Forced Work

The energy required to perform forced work comes from the flywheel. During periods of forced work, extra energy is removed from the flywheel to overcome friction in the mechanism. Similarly, the engine must supply extra energy to overcome friction while replenishing the flywheel. Areas of forced work are not directly subtracted from the engine cycle; they are subject to friction twice. The amount of forced work depends on the buffer pressure, and on the cycle shape. The negative effect of forced work depends on the mechanism effectiveness.

Assuming a constant mechanism effectiveness, E, the shaft work, W_{shaft}, is related to the indicated work, W_i, and the forced work, W_f, by Eq. (1.34) below [27].

$$W_{shaft} = EW_i - \frac{W_f}{E} + EW_f \tag{1.34}$$

Each term on the right-hand side has a clear physical significance. The first term represents the case absent of forced work, in which the indicated work is directly reduced by the mechanism effectiveness. In this case, energy flows through the mechanism in one direction only: from the engine to the flywheel. The second term describes work which is stored in the flywheel by the engine, for later use in overcoming forced work. The negative sign indicates energy removed from the engine. The third term is the energy which returns from the flywheel to perform forced work.

Nearly all Stirling engine optimization studies focus on indicated work, rather than shaft work. It is possible to have cycles with high indicated work, that will not run due to the presence of excessive forced work [27]. It is also possible to have cycles with no forced work at all. A simple way to reduce the amount of forced work for a Stirling engine is to reduce the overall volume change by reducing the piston swept volume. If an engine is optimized for maximum shaft work, some forced work will be present. The optimum amount of forced work depends on the mechanism effectiveness. Senft recommends erring on the side of less forced work, to improve the life of the mechanism [27].

1.5.5 Flow Friction

Flow friction reduces the power of real Stirling engines by resisting the flow of working fluid through the engine [8]. Prediction of this loss is difficult due to the flow conditions present in Stirling engines: Unsteady, compressible flow with sudden changes in temperature and geometry. Approaches in the literature include direct use of steady flow correlations, steady flow correlations with a quasi-steady approach, oscillating flow correlations, and computational fluid dynamics simulations. The simple model described by Urieli and Berchowitz includes a quasi-steady method for estimating the flow friction loss, and will be described in further detail below.

1.5.6 Gas Spring Hysteresis

Gas springs are sometimes found in Stirling engines, and come with inherent losses [10]. For example, in the ST05G Stirling engine, the crankcase acts as a gas spring when it is compressed and expanded by motion of the piston. At steady state, the walls of a gas spring will equilibrate at an intermediate average temperature [6]. When gas is compressed, its temperature rises, and it

gives up heat to the walls of its container. During expansion, the gas cools and heat is absorbed from the walls at a lower temperature than it was given up. In a complete cycle, there is no net heat flow between the gas and the walls, but there is a loss of work [6]. Equations for prediction of gas spring hysteresis loss derived from first principles are provided by Urieli and Berchowitz [10]; however, the assumptions used to derive them are invalid in the case of the ST05G engine. Gas spring hysteresis can be measured as the area of the gas spring indicator diagram [28].

1.5.7 Heat Transfer Hysteresis

Heat transfer hysteresis loss is the same phenomenon as gas spring hysteresis, occurring inside the working space [6]. Calculation of this loss would involve determination of finite heat transfer rates in all engine components. It is typically not included in second order models.

1.5.8 Seal Leakage

Three types of seal leakage can reduce the power output of real Stirling engines: Leakage past the displacer, leakage past the piston, and leakage to the outside. Leakage past the displacer results in a decrease in the temperature difference between the expansion and compression spaces. This reduces the displacer's ability to change the engine pressure, and results in a decrease in power. Piston seal leakage decreases the net force across the piston, reducing the work extracted, and ultimately the engine power. Leakage from inside the engine to the environment reduces the engine power by necessitating the use of an auxiliary compressor to maintain the engine fill pressure.

1.5.9 Piston Finite Speed Loss

The idea of finite piston speed loss in Stirling engines seems to have originated with Petrescu et al. [18], [29]. They refer to change in the effective pressure felt by the piston due to pressure waves generated by piston motion [29]. According to Petrescu et al., the work lost due to finite piston speed, δW_{FPS}, for an incremental volume change, dV, is given by

$$\delta W_{FPS} = p_i \left(\pm \frac{aw}{c_{FPS}} \right) dV \tag{1.35}$$

where

$$a = \sqrt{3\gamma} \quad \text{and} \quad c_{FPS} = \sqrt{3RT} \tag{1.36}$$

Here, P_i and w are the average pressure and piston speed during the incremental volume change, and c is the average speed of the gas molecules. The sign of the term in brackets is chosen so that the equation increases the work of compression and decreases the work of expansion [18].

1.5.10 Auxiliary Component Losses

A complete Stirling engine system would include auxiliary components such as a coolant pump, fuel pump, working fluid compressor, power control system, etc. These components would reduce the power output if driven by the engine. They are typically not included in academic Stirling engine research, and will not be considered here.

1.6 The Simple Heat Exchanger Model

The Simple model, described by Urieli and Berchowitz [10], [30], is a popular approach for including the flow friction and imperfect heat transfer losses of the heater, cooler, and regenerator in a second order model based on the ideal adiabatic reference cycle.

The heat transfer performance of a regenerator may be described by a parameter called the regenerator effectiveness, ε, which varies between 1, for a perfect regenerator, and 0, for no regeneration [10]. Urieli and Berchowitz define the regenerator effectiveness with reference to the ideal adiabatic model as follows [10].

$$\varepsilon \equiv \frac{\left(\begin{array}{c}\text{Actual heat transferred unidirectionally from the} \\ \text{regenerator to the working fluid in a complete cycle}\end{array}\right)}{\left(\begin{array}{c}\text{Unidirectional heat transfer from the regenerator to} \\ \text{the working fluid predicted by the ideal adiabatic model}\end{array}\right)} \tag{1.37}$$

Nomenclature is defined below in terms of a counter flow heat exchanger analogy.

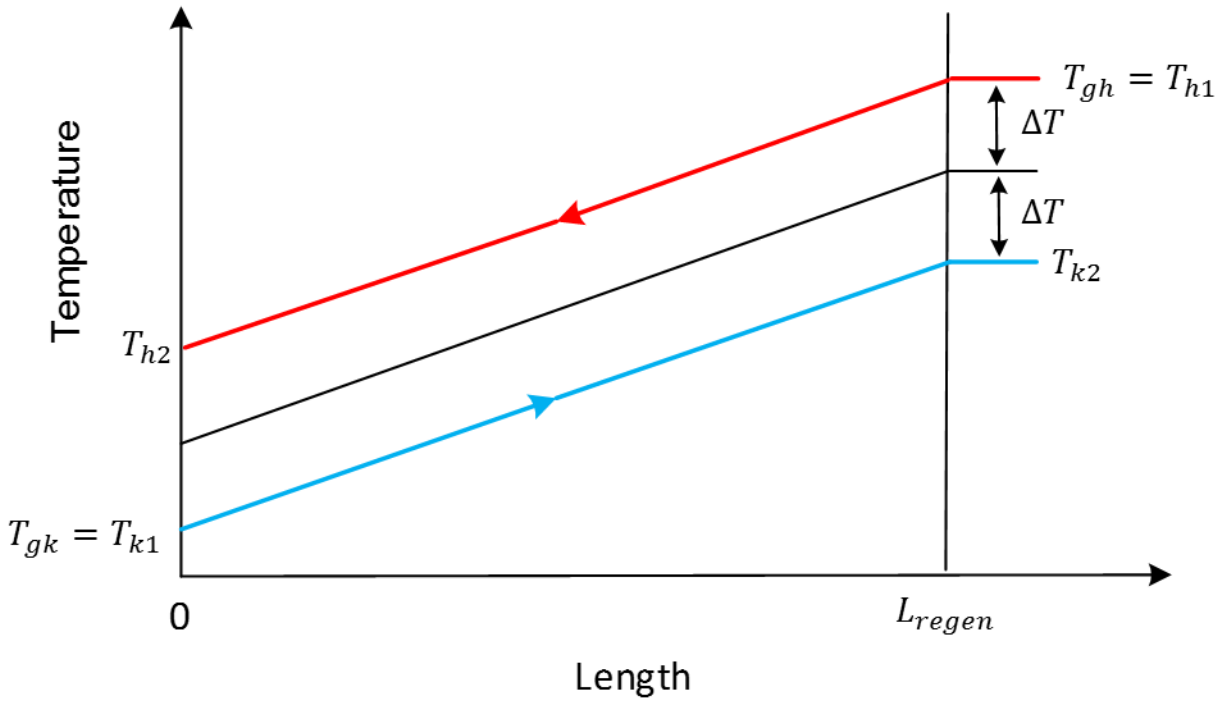


Figure 1.11: Fluid and Matrix Temperatures Defining Nomenclature for Regenerator Effectiveness, after Urieli and Berchowitz [10]

With reference to Figure 1.11, the regenerator effectiveness may also be written as

$$\varepsilon = \frac{(T_{h1} - T_{h2})}{(T_{h1} - T_{k2})} \tag{1.38}$$

Since the mass flows and heat capacities in both directions are equal over a complete cycle,

$$2\Delta T = T_{h2} - T_{k2} \tag{1.39}$$

Substituting Eq. (1.39) into Eq. (1.38) yields

$$\varepsilon = \frac{1}{1 + \frac{2\Delta T}{T_{h1} - T_{h2}}} \tag{1.40}$$

Now, the conservation of energy, Eq. (1.3a), is applied to the hot stream, and simplifies to

$$\dot{Q} = \dot{m} c_p (T_{h1} - T_{h2}) \tag{1.41}$$

Using Newton's Law of Cooling, $\dot{Q}$, can also be written as

$$\dot{Q} = 2hA_r \Delta T \tag{1.42}$$

where h is a convective heat transfer coefficient which describes the process of heat transfer from the hot stream to the regenerator matrix to the cold stream. Substitution of Eqs. (1.41) and (1.42) into Eq. (1.40) gives

$$\varepsilon = \frac{1}{1 + \frac{\dot{m} c_p}{h_{conv} A_r}} \tag{1.43}$$

or

$$\varepsilon = \frac{NTU}{1 + NTU} \tag{1.44}$$

where NTU represents the number of transfer units defined as

$$NTU = \frac{N_{St} A_r}{2A_{flow,r}} \tag{1.45}$$

The factor of two in Eq. (1.45) is added because the Stanton number, N_{St}, is typically defined for only a single gas to matrix heat transfer process rather than for two. Urieli used the following steady flow Stanton number correlation from Kays and London for mesh regenerators [10], [31].

$$N_{St} = \frac{0.46{N_{Re}}^{-0.4}}{N_{Pr}} \tag{1.46}$$

Using the regenerator effectiveness, and the reference cycle result for heat transfer in the regenerator, the regenerator enthalpy loss is calculated as

$$Q_{rloss} = (1-\varepsilon)(Q_{r,ideal,max} - Q_{r,ideal,min}) \tag{1.47}$$

where the second factor on the right-hand side represents the unidirectional heat transfer from the regenerator to the working gas predicted by the ideal adiabatic model.

The convective heat transfers in the heater and cooler may be written as

$$\begin{aligned} Q_{k,ideal} &= h_{conv,k}A_k\left(T_{wk} - T_{gk}\right) \\ Q_{h,ideal} &= h_{conv,h}A_h\left(T_{wh} - T_{gh}\right) \end{aligned} \tag{1.48}$$

The heater and cooler must transfer additional heat to overcome the regenerator enthalpy loss. Including this, Eqs. (1.48) become

$$\begin{aligned} (Q_{k,ideal} - Q_{rloss}) &= h_{conv,k}A_k\left(T_{wk} - T_{gk}\right) \\ (Q_{h,ideal} + Q_{rloss}) &= h_{conv,h}A_h\left(T_{wh} - T_{gh}\right) \end{aligned} \tag{1.49}$$

Equations (1.49) can be rearranged to give the mean effective gas temperatures of the heater and cooler as

$$\begin{aligned} T_{gk} &= T_{wk} - \frac{(Q_{k,ideal} - Q_{rloss})}{\overline{h_{conv,k}}A_k} \\ T_{gh} &= T_{wh} - \frac{(Q_{h,ideal} + Q_{rloss})}{\overline{h_{conv,h}}A_h} \end{aligned} \tag{1.50}$$

where the overbar indicates that the convective heat transfer coefficient is averaged spatially across the heat exchanger.

To obtain the convective heat transfer coefficients, the Blasius turbulent steady flow friction factor correlation

$$f_{Re} = 0.0791{N_{Re}}^{0.75} \tag{1.51}$$

is used with Reynolds simple analogy:

$$h = \frac{f_{Re}\mu c_p}{2dN_{Pr}} \tag{1.52}$$

The simple model also includes an estimation of the power lost due to flow friction. Taking the compression space temperature as the reference point, pressure drops of the heat exchangers and regenerator are subtracted from the expansion space temperature. This leads to the following expression for the work output of the engine in one cycle [30].

$$W = W_e + W_c = \oint p\frac{dV_c}{d\theta}d\theta + \oint (p - \sum \Delta p)\frac{dV_e}{d\theta}d\theta$$

$$W = \oint p\left(\frac{dV_c}{d\theta} + \frac{dV_e}{d\theta}\right)d\theta - \oint \sum \Delta p\frac{dV_e}{d\theta}d\theta = W_{ideal} - W_{flow} \tag{1.53}$$

So, the total work lost per cycle due to flow friction is [30]

$$W_{flow} = \int_0^{2\pi}\left[(\Delta p_k + \Delta p_r + \Delta p_h)\frac{dV_e}{d\theta}\right]d\theta \tag{1.54}$$

The individual pressure drops for the cooler, regenerator, and heater are calculated using the following equation [30].

$$\Delta p_i = -2\left(\frac{f_{Re}\mu V m_{flux} L}{md^2}\right)_i \tag{1.55}$$

Reynolds friction factors for the heater and cooler are calculated using Eq. (1.51). For mesh regenerators. Eq. (1.56) below is used [31].

$$f_{Re} = 54 + 1.43{N_{Re}}^{0.78} \tag{1.56}$$

The Reynolds number, N_{Re}, is calculated using the following equation.

$$N_{Re} = \frac{|\rho u|d}{\mu} \tag{1.57}$$

The factor in absolute value brackets is calculated at each crank angle increment from the results of the reference cycle calculation. The hydraulic diameter, d, is calculated as

$$d = \frac{4V_{HEX}}{A_{HEX}} \tag{1.58}$$

for slotted heat exchangers and

$$d = \frac{d_{wire}¶}{1 - ¶} \tag{1.59}$$

for wire mesh regenerators [30].

Numerical solution of the simple model follows the algorithmic approach shown in Figure 1.12 below.

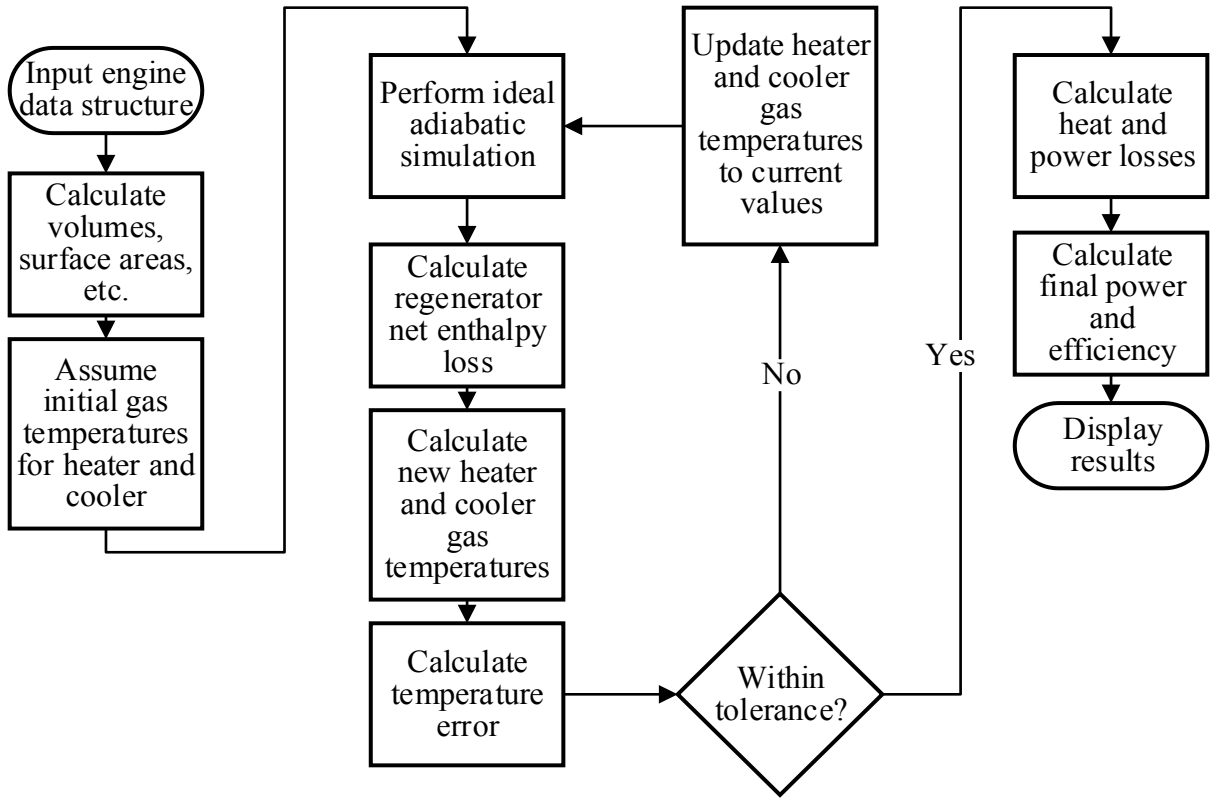


Figure 1.12: Algorithmic Approach for the Simple Model

Written in steps, the solution procedure is:

1. Set the heater and cooler wall temperatures to the thermal source and sink temperatures respectively.

2. Perform an ideal adiabatic simulation.
3. Using the mass flow results from the ideal adiabatic simulation, calculate the average Reynolds number in the regenerator.
4. Use the average Reynolds number to calculate the Stanton number in the regenerator via Eq. (1.46).
5. Calculate the number of transfer units in the regenerator using the Stanton number and Eq. (1.45).
6. Use the number of transfer units to calculate the regenerator effectiveness with Eq. (1.44).
7. Calculate the regenerator enthalpy loss by substituting the ideal adiabatic model results and the regenerator effectiveness from the previous step into Eq. (1.47).
8. Use the regenerator enthalpy loss and the convective heat transfer coefficients to calculate the mean effective gas temperatures in the heater and cooler using Eqs. (1.50).
9. Calculate the "temperature error" using the following formula.

$$T_{error} = \left|T_{gh,old} - T_{gh,new}\right| + \left|T_{gk,old} - T_{gk,new}\right| \tag{1.60}$$

where the suffixes old and new refer to the previous and current iterations of the loop.

10. If the temperature error is below a specified tolerance, move on to step 11; otherwise, update the heater and cooler gas temperatures to the current values and rerun the loop starting from step 2.
11. Once the loop has converged, calculate the work lost due to pressure drops across the heat exchangers and regenerator.
12. Add the losses to the reference cycle results to obtain the final power and efficiency. In the original code offered on Urieli's website, [30], the regenerator enthalpy loss was added to the heat input and the pressure drop work loss was subtracted from the power output.

1.7 Second Order Models from the Literature

The literature was surveyed with a focus on second order modeling approaches. Works using the ideal isothermal model as a reference cycle will be discussed first, followed by models derived from the principles of non-equilibrium thermodynamics. Models which use the ideal adiabatic reference cycle, as is done in the present study, will be examined in detail. Lastly, shortcomings of validation schemes in the current literature regarding second order models will be highlighted.

While the ideal isothermal model provides a first-cut understanding of the Stirling cycle, its use for detailed design is limited. Kongtragool and Wongwises, [32], studied the ideal isothermal model, and extended it by adding imperfect heat transfer from the regenerator. Regenerator pressure drop was not included. They observed that a decrease in regenerator effectiveness led to a decrease in efficiency, due to an increased heat input, but left the power output unchanged. Practical regenerators influence the engine power output in two ways. They introduce a significant pressure drop, which negatively affects the engine power. Regenerators also reduce the heat transfer requirements for the heater and cooler, often leading to a larger temperature difference between the compression and expansion spaces, which increases the power output. A more detailed model by Formosa and Despesse, [33], added conduction loss, regenerator enthalpy loss, mechanical and flow friction, and heat exchanger effectiveness. They compared their results to published performance data for the GPU-3, a well-documented beta-type Stirling engine, with good agreement. Their description left out important details such as their assumed mechanism effectiveness and flow friction correlations. These parameters can easily be adjusted to "validate" a model, especially when comparing results to only one engine. A second order model which agreed perfectly with experimental data would justifiably raise suspicion due to the assumptions involved. Ahmadi et al., [34], used an isothermal model, with regenerator and heat exchanger effectiveness added as losses, in an optimization study. While optimization is the ultimate goal of any Stirling engine model, performing it on an incomplete model can lead to a non-optimum design.

Another category of second order models are those based on ideas of non-equilibrium thermodynamics. The genre is further subdivided into finite time thermodynamics and finite speed thermodynamics. Finite time approaches consider finite heat transfer rates in the engine

components, with are evaluated as functions of time. Bert et al., [35], used a finite time approach to develop a three-component numerical model of a gamma-type engine. Nusselt numbers for each component were calculated using the Colburn equation for turbulent flow in smooth tubes, corrected using an unspecified function of the length to diameter ratio of each component. Model results were overlaid on indicated power measurements with good agreement. Ding, Chen, and Sun, [36], used an analytical scheme which combined finite time thermodynamics, and the mechanical friction ideas of Senft. They studied the influence of conduction loss, regenerator enthalpy loss, and mechanical friction on predicted power and efficiency, but did not compare their results to experimental data. Punnathanam and Kotecha, [37], and Fan et al., [38], both used finite time approaches for optimizations of overall power generation systems. Finite speed thermodynamics was applied to Stirling engine analysis by Petrescu and Costea et al. [18], [29]. The group considered pressure losses due to fluid friction, mechanical friction, and pressure waves generated by finite piston speed [29]. The fluid and mechanical friction calculations were both empirical models based on experimental data from the literature [29]. Li et al., [39], formed a second order model using the ideal isothermal reference cycle, and several losses including finite piston speed. They calculated heat transfer hysteresis loss in the working space using the gas spring hysteresis equations presented by Urieli and Berchowitz [10]. The relative influence of the losses on the energy flows through the engine are presented as calculated by the model. Experimental validation considered overall performance only. Ahmadi et al., [40], argue that the finite speed thermodynamic method is superior to the finite time method, because it considers irreversibilities internal to the engine such as flow friction and mechanical friction. They compared their model to the overall power and efficiency of the GPU-3 Stirling engine at one operating point, and displayed predicted performance trends for variable source temperature, regenerator effectiveness, swept volume ratio, piston stroke, and rotational speed. Overall, the finite time and finite speed formulations have the advantage of short computation time, making them popular for optimization studies, but their semi-empirical nature limits their wide applicability.

Several examples of second order models based on the ideal adiabatic model exist in the literature. Strauss and Dobson, [41], treated the regenerator and flow friction losses differently than in the original description by Urieli and Berchowitz [10]. While the same methods were used to calculate the magnitude of each loss, the way they were added to the reference cycle

results differed. Strauss and Dobson, [41], argue that, for real engines, the heat input rate is limited by the heat source temperature; thus, it is more realistic to add the regenerator enthalpy and conduction losses to the heat rejection rate rather than the heat input rate. The gas temperature in the cooler must increase to increase the rate of heat rejection. Strauss and Dobson, [41], only considered flow friction losses in the cooler, and these were added to the heat rejection rate only. They, [41], compared their results to Urieli and Berchowitz by applying their model to the GPU-3, and found better agreement with experimental data.

Araoz et al., [42]–[45], developed a second order model and compared it to both published GPU-3 data and to data they collected from a two-unit gamma-type engine. To the original Simple model, they added mechanical friction losses using Senft's forced work approach. They also used different heat transfer correlations for the regenerator [46].Their two-unit gamma engine performed poorly relative to the manufacturer's specifications [44]. Mechanical friction was the dominant power loss mechanism according to their simulation [44]. An earlier paper, [45], reveals that their engine is buffered from above, i.e. the buffer pressure is higher than the workspace pressure during the entire cycle. This leads to excessive forced work and degrades the mechanical efficiency of the engine significantly. They assumed a constant mechanism effectiveness of 0.8 [44], which is relatively high compared to the 0.7 recommended by Senft [27].

Babaelahi and Sayyaadi, [47], [48], extended the Simple model by adding additional losses and by rederiving the original equation set with fewer assumptions. Their 2014 publication, [47], added the effect of piston seal leakage and appendix gap shuttle loss to the differential equation set. They also calculated the mechanical friction as a decoupled loss loss using finite speed thermodynamics [47]. Their later paper, [48], considered polytropic processes in the compression and expansion spaces, and solved iteratively for the polytropic index as part of its convergence scheme. Their models were compared to published results for the GPU-3.

Hosseinzade and Sayyaadi, [49], produced a second-order model by combining the ideal adiabatic reference cycle with the finite speed formulations for mechanical friction, flow friction, and finite piston speed losses integrated into the Simple heat exchanger model. Using the GPU-3 as a case study, they plotted the pressure drop due to finite piston speed as a function of crank angle and showed that it is similar in magnitude to the regenerator pressure drop. The overall

power and efficiency of their model was compared to published data on the GPU-3 at a single operating point.

Paul, [50], [51], developed a second order model for the GPU-3 as part of his PhD thesis. His additions to the Simple model included using an iterative scheme to calculate the mass of working fluid, adding heater and cooler manifolds and the appendix gap as separate components, and adding minor losses to the flow friction calculation [50]. He suggests a rule of thumb that for Stirling engines operating at a mean pressure below 3.0 MPa and at a speed above 1500 RPM, the steady, incompressible friction factor should be multiplied by 4/3 to correct for the effects of oscillating, compressible flow. Heat transfer hysteresis and flow friction were the most influential losses for the GPU-3, according to his calculations [50]. Comparisons to the measured power and heat transfer rates over a range of operating conditions were made to validate the model.

Ni et al., [52], performed experiments on a 100 W beta-type Stirling engine using air and helium working fluids and compared the results to a second order model. The model used the ideal adiabatic reference cycle and the Simple heat exchanger approach, with the addition of heat transfer hysteresis, appendix gap, piston seal leakage, and conduction losses. They compared their calculations to overall power and efficiency data over a range of operating conditions for both the GPU-3 and their 100 W beta-type engine. Measured gas temperatures were also compared to calculated results.

Second order models have the advantages of simplicity and transparency over higher order techniques. They are composed of many separate calculations, which are often semi-empirical in nature. Most published second order model validations are done by comparison with overall power and efficiency data. This makes it difficult to diagnose errors. Ideally, the individual components of a second order model should be validated individually.

1.8 The ST05G Stirling Engine

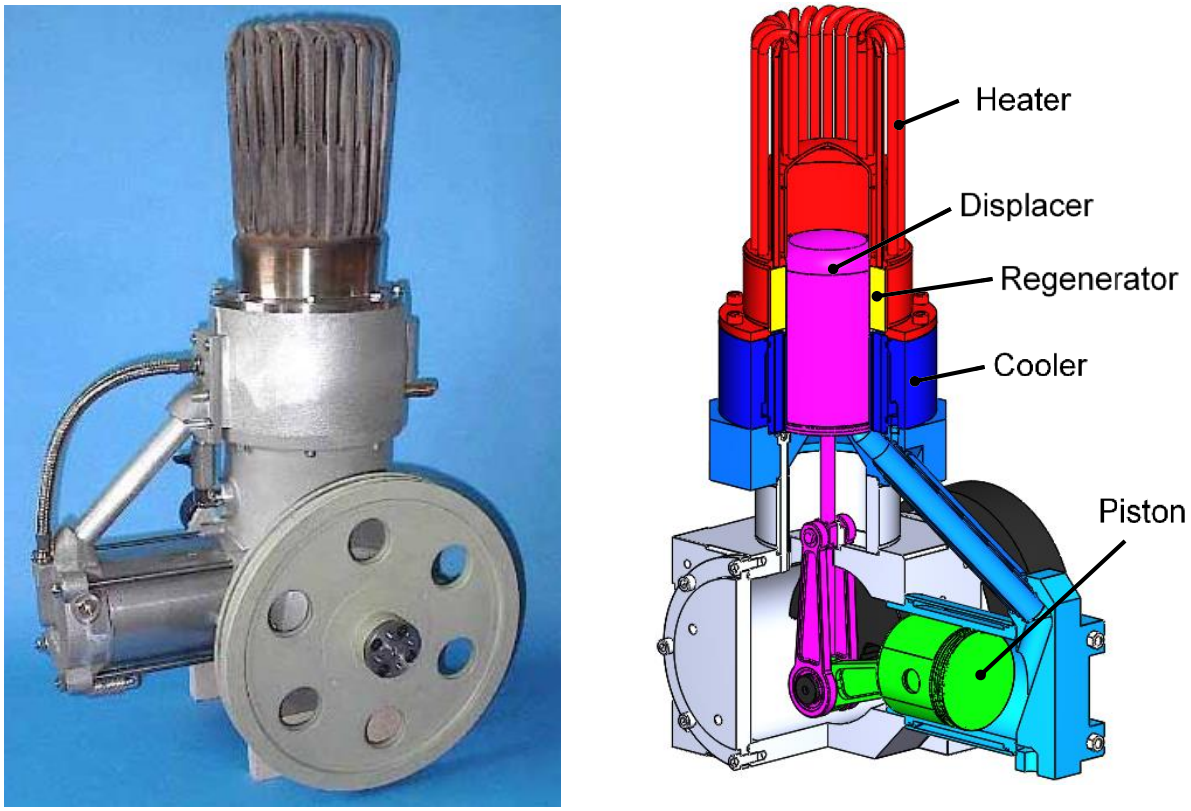
The work presented here involves a gamma type Stirling engine, designed in 1992 by Dieter Viebach to promote Stirling engine technology for electricity production from biomass combustion [53]. Ve-Ingenieure, [53], offers a set of castings for the original ST05G design, shown in Figure 1.13. They also have a version designed for CNC machine manufacturing, called the ST05G-CNC. The full drawing package for each of these versions is available on their website [53].

Table 1.3 shows the specifications of the ST05G-CNC Stirling engine from the drawing package preamble. Figure 1.13(a) shows an image of the original ST05G Stirling engine, while Figure 1.13(b) labels the main components on a sectioned solid model.

Table 1.3 Design Data for Original ST05G Stirling Engine [53]

Power Output	500 W at 8.33 Hz (500 RPM)
Maximum Engine Speed	12.5 Hz (750 RPM)
Maximum Heater Head Temperature	550 °C
Working Gas Pressure	1000 kPa
Working Gas	Dry nitrogen or oil-free compressed air
Stroke	0.075 m
Displacer Bore	0.096 m
Piston Bore	0.085 m
Main Dimensions	0.410 x 0.375 x 0.780 m
Total Weight	33.5 kg

The engine is rated at 500 W of shaft power when spinning at 500 RPM. It features a pressurized crankcase, which facilitates fill pressures up to 1000 kPa. The heater is comprised of twenty stainless steel tubes welded to the outside of the displacer cylinder. A water jacket surrounds the cooler, connecting pipe, and power cylinder. The engine is designed to run oil-free for two reasons: To eliminate the risk of combustion of atomized oil particles in the engine which could lead to an explosion, and to prevent clogging of the regenerator.



(a) (b)

Figure 1.13: (a) Original ST05G Stirling Engine [53] (b) Sectioned Solid Model of ST05G-CNC Stirling Engine

The ST05G Stirling engine made its debut at the 1993 ISEC Stirling engine conference, [53], and since then it has been studied by both academics and Stirling engine enthusiasts. Gheith et al. contributed several publications regarding an engine similar to the ST05G. Experiments reported in their 2011 paper, [54], showed the effects of varying fill pressure, speed, coolant flow rate, and thermal source temperature. Their setup included four thermocouples on each side of the regenerator, which revealed that the average regenerator temperature was significantly higher on the power cylinder side of the engine compared to the opposite side [54]. Using the phase shift between the top and bottom thermocouples, they estimated that the axial velocity of the working fluid was twice as large on the power cylinder side, as it was on the opposite side [54]. They claimed that the difference in average pressure between the expansion and compression spaces represents the pressure drop loss of the engine, and that the difference in area between the expansion space and compression space indicator diagrams is equal to the work lost due to flow friction in a complete cycle [54]. The same group also studied the influence of the regenerator

properties on the engine performance by testing several materials and porosities [55], [56]. They recommended stainless steel with a porosity of 85 % as the best compromise between heat transfer, pressure drop, and oxidation resistance [56]. In a 2015 contribution, [57], Gheith et al. experimentally studied the effects of source temperature, fill pressure, engine speed, and coolant flow rate, with emphasis on the heater performance. The engine tested used a heater made up of 20 tubes similar to the original ST05G. Results showed that there is an optimum combination of temperature, pressure, speed, and coolant flow rate which maximizes the heat transfer rate through the heater [57]. Hachem et al., [58], compared the Stirling engine tested by Gheith et al. to an open cycle Ericsson engine, and found that the Stirling outperformed the Ericsson in specific indicated work, thermal efficiency, and exergetic efficiency. The regenerator of the Stirling engine was the reason its performance was better, and Hachem et al. point out that the Ericsson engine could achieve similar performance to the Stirling if fitted with an air preheater [58]. Another work by Hachem et al., [59], presents second order modeling results for the engine tested by Gheith et al. The model is based on the simple model of Urieli and Berchowitz [10], and includes shuttle losses and mechanical friction losses [59]. Trends predicted by the model are shown, but comparison to experimental data is minimal [59].

Bert et al., [35], [60], modeled, conducted experiments, and numerically optimized the kinematics of an engine design based on the ST05G. Their engine had a drilled-block heater head rather than the original tube cage heater [60]. Their model divided the engine into three zones and made use of the mass and energy conservation laws [35]. Only indicated power was reported for their ST05G engine [60]. Their kinematic optimization suggested discontinuous piston and displacer motion would increase power at low speed, but would decrease the maximum power at high speed relative to the original volume variations [60].

Hooshang et al., [61], [62], offer two publications concerning the ST05G Stirling engine. The first optimized a third order model of the engine for maximum power and efficiency by varying displacer stroke, frequency, and phase angle [61]. Calculations were done using helium as a working fluid [61]. The second paper reported measured crankshaft speed variations and compared them to thermodynamic and rigid body dynamic modeling results [62]. Neither paper gave a power output measurement.

Alfarawi et al., [63]–[65], analyzed the original ST05G-CNC Stirling engine through modeling and experiments. Their test rig included temperature measurements of both ends of the regenerator, the compression space, and the heater wall, pressure measurements inside the compression space, and a dynamometer for power output measurement [64]. They began by comparing their experimental results to a second order model based on the ideal adiabatic reference cycle [64]. Slider-crank volume variations, mechanical friction, appendix gap losses, and conduction losses were added to Urieli and Berchowitzs' simple heat exchanger analysis to complete their model [64]. They used Senft's forced work approach to calculate mechanical friction, and determined a constant mechanism effectiveness of 0.75 through comparison with their experimental results. They assumed a constant buffer pressure equal to the mean pressure, and their work made no mention of gas spring hysteresis loss in the crankcase. Empirical heat transfer and flow friction correlations based on steady flow experiments, [66], were used for the heat exchangers while the regenerator calculations were based on correlations specific to oscillating flow [67]. Next, they subjected the ST05G-CNC to a computational fluid dynamics simulation [63]. Their CFD results suggested that indicated power was maximized at a phase angle of 105 °, and a connecting pipe internal diameter of 14 mm [63]. In their most recent paper, [65], the group performed 3D CFD calculations on mini-channel regenerators for the ST05G-CNC engine. The regenerators they simulated approached, but did not exceed, the performance of random fiber regenerators.

Table 1.4: Summary of Power Measurements of Engines Similar to the ST05G-CNC Reported in the Literature

Research Group	Engine Description	Maximum Indicated Power (W)	Maximum Shaft Power (W)	Operating Conditions at Maximum Shaft Power	Minimum Temperature Difference Tested (°C)
Gheith et al. [54]	Original heater head, copper wire regenerator with 90 % porosity, slightly different dimensions than the ST05G-CNC	Not reported	320	5.5 bar, Hot Source = 550 °C, Expansion Space Gas = 428.5 °C, 450 RPM	~380
Gheith et al. [55]	Same setup as above, but with a stainless-steel regenerator with 85 % porosity	Not reported	308	8 bar, Hot Source = 400 °C	275
Bert et al. [60]	Drilled block heater head, same bores and strokes as ST05G-CNC	850	Not reported	11 bar, Heater wall temperature = 700 °C, 1000 RPM, Coolant temperature = 25 °C	275
Alfarawi et al. [64]	Stock ST05G-CNC	740	470	10 bar, Hot end temperature = 650 °C, Coolant temperature = 15 °C	435

The ST05G Stirling engine has been studied by several academic groups, yet it still offers research opportunities. Pressure and temperature measurements in the crankcase have not been

performed; hence, the role of the buffer pressure has not been quantified. Experiments testing the effects of physical modifications to the ST05G-CNC Stirling engine have not been reported in the open literature. An alternative to the hot side heat exchanger would allow new heat source possibilities. Options could be expanded further by improving the engine's low temperature difference performance. Further experimental data will benefit a community of researchers currently studying the ST05G.

1.9 Low Temperature Difference Stirling Engines

Wang et al., [2], presented a recent review paper on low temperature difference Stirling engines for low grade heat recovery. They summarized experimental work done and concluded that kinematic and thermoacoustic Stirling engines had the greatest potential for low grade heat recovery applications.

Most low temperature Stirling engine prototypes use a gamma layout, and produce less than 10 W of shaft power. Pioneers of the field appear to be Ivo Kolin [68] and James R. Senft [69]. They both built small demonstration engines with power outputs in the milliwatt range. The engines of Kolin and Senft engines both used flat plate heat exchangers and operated on minimum temperature differences of 15 °C and 0.5 °C, respectively. Cinar et al., [70], investigated the effect of changing displacer material on a 3 W engine with a thermal source temperature of 160 °C. His plots demonstrate that maximum engine torque occurs at relatively low speed. He found that medium density fiberboard displacer was a superior displacer material relative to aluminum alloy due to its lower thermal conductivity and density. Kato, [71], presented indicator diagrams for a 3.34 mW gamma-type Stirling engine at thermal source temperatures of 75 - 95 °C. Results indicated that the flat plate heat exchangers performed poorly when the displacer was far away.

Larger scale low temperature difference Stirling engines have also been built and tested. Boutammachte and Knorr, [72], field-tested two solar-powered engines with 1.6 m diameter displacers that doubled as solar energy absorbers. They employed a cam mechanism to impart discontinuous motion on the displacer, and measured a performance benefit in terms of indicated work, at the expense of a more complicated mechanism and higher accelerations on the displacer. They reported a power output of 30-40 W at absorber temperatures of 50-80 °C, speeds of 30-40 RPM, and mean pressure equal to atmospheric pressure. Iwamoto et al., [73], tested a gamma-type Stirling engine with a 0.8 m displacer diameter, finned-tube heat exchangers, copper mesh regenerator, and pressurized crankcase. Their engine produced a maximum power of 146 W at 143 RPM at a gas temperature difference of 73 °C.

Kongtragool and Wongwises, [74], performed experiments on two gamma-type Stirling engines: one with two pistons and one with four. Their two and four piston engines achieved maximum power outputs of 11.8 W and 32.7 W at thermal source temperatures of 316 °C and

498 °C. They present torque and power curves as a function of speed which show similar trends to those of Cinar. Their work presents examples of engines intended for low temperature difference operation running at relatively high temperature differences.

A group led by Seth Sanders at the University of California-Berkeley, [28], [75]–[82], made significant progress towards low temperature Stirling engines for solar-thermal power generation systems. They targeted hot side temperatures in the 120 °C to 150 °C range, and built several configurations, including a crankshaft-driven gamma [78], [80], and a multi-unit free piston Stirling engine design with integrated linear alternators [28]. Der Minassians and Sanders, [28], emphasized that the cost of low temperature designs may be significantly reduced relative to high temperature designs by replacing metal parts with plastics. Their multi-unit design featured custom silicone rubber diaphragms molded in 3D-printed wax molds [77]. They used a second order modeling approach based on the ideal adiabatic model [28]. To quantify frictional losses experimentally, they performed ring-down tests on the piston and displacer [28]. These consisted of displacing the piston and displacer individually from their equilibrium positions and tracking their oscillations when released [28]. Gas spring hysteresis losses were calculated as the area of the indicator diagram measured for the gas springs [28]. Formosa et al., [82], performed experiments on a similar 3-unit free piston Stirling engine, and compared their results to an electrical network representation of the ideal isothermal model.

First order mathematical models for low temperature difference Stirling engines have been explored by Kongtragool and Wongwises [83]. They compared first order approaches for estimating the power output to experimental data from Senft's low temperature difference engines, [69], and concluded that the West number correlation was the most useful since it includes a factor that depends on the temperature difference. They obtained closest agreement when the West number was taken to be between 0.025 and 0.035.

Hoegel et al., [84], [85], modeled low temperature difference Stirling engines using a commercial third order analysis software. They compared two alpha-type Stirling engines with hot source temperatures of 750 °C and 150 °C, respectively, and identified the following differences [85]:

1. The optimum phase angle for alpha engines is higher for low temperature difference engines than for high temperature difference engines. This translates to gamma and beta engines having a smaller piston swept volume relative to displacer swept volume for low temperature differences.
2. Low temperature difference engines run slower due to lower heat transfer rates and since flow friction has a greater influence on the total power output.
3. Optimum regenerator porosity is higher for low temperature difference engines.
4. Heat exchangers can be made identical in low temperature difference engines without reducing performance.
5. Heat exchangers for the low temperature engine were optimal when they had more tubes and were shorter in length than for the high temperature engine (tube diameter was fixed). This result was attributed to the following:
    a. Flow friction influence is greater for low temperature engines, so optimum length of heat exchangers is shorter.
    b. Heat transfer surface area is more important for low temperature engines, so the optimum number of tubes was greater.

The body of literature concerning high temperature difference Stirling engines is vast, and a substantial amount of work has also been devoted to low temperature difference designs. Relatively few works attempt to bridge the gap between the two genres, and important questions remain unaddressed: What modifications are necessary to convert a high temperature difference engine to a low temperature difference one? Are models validated at high temperature operating points still relevant at low temperature ones? Understanding the similarities and differences between these two engine categories will help solidify the Stirling engine literature, and provide insight on the design of each.

1.10 Thesis Objectives and Structure

The remainder of this thesis is divided into five chapters. Their overarching aims are to explore modifications for reduction of the minimum thermal source temperature of the ST05G-CNC Stirling engine, and to assess the validity of common second order models for analysis of low temperature difference machines.

In Chapter 2, a measurement system is developed to allow a modified ST05G-CNC Stirling engine to be evaluated experimentally. The computer controlled system includes temperature and pressure measurements at several locations on the engine, as well as torque and crankshaft position measurements. The engine is heated electrically, and cooled by a two-zone water jacket with controlled coolant flowrate and temperature. The experimental procedures, and uncertainties are stated to conclude the chapter.

The third chapter develops a second order model to predict performance of the modified ST05G-CNC engine. The model is written such that the isothermal and adiabatic reference cycles may be used interchangeably. It includes flow friction, mechanical friction, gas spring hysteresis, appendix gap loss, conduction loss, and regenerator enthalpy loss. The method of combining losses with the reference cycle results is described and sample calculations are presented to demonstrate the capabilities of the model.

Chapter 4 describes modifications made to the ST05G-CNC Stirling engine to reduce its minimum thermal source temperature. Specifically, the effects of reduced piston swept volume, increased crankcase volume, and reduced dead volume are revealed through experimentation.

In Chapter 5, the fully modified engine is compared to predictions made using the second order model developed in Chapter 3. Individual components of the model are isolated and validated against relevant experimental data as much as possible.

The final chapter presents overall conclusions, and suggests potential future work.

Chapter 2. Description of the Experimental Set-up

The following chapter provides a detailed description of the experimental set-up. It begins with a general overview of the test rig. Each subsystem is then delineated individually. Lastly, testing procedures are outlined, and experimental uncertainties are assessed.

2.1 Overview

The experimental set-up is shown in Figure 2.1 and Figure 2.2 below. A modified ST05G-CNC Stirling engine, using compressed air as a working fluid, is heated electrically and water-cooled using a refrigerated bath and two peristaltic pumps. Variable loads are applied to the engine using a strap brake draped over a plastic spool. Measurements of temperature, pressure, crankshaft position, and torque are collected using a computer controlled data acquisition system.



Figure 2.1: Image of Overall Experimental Set-up

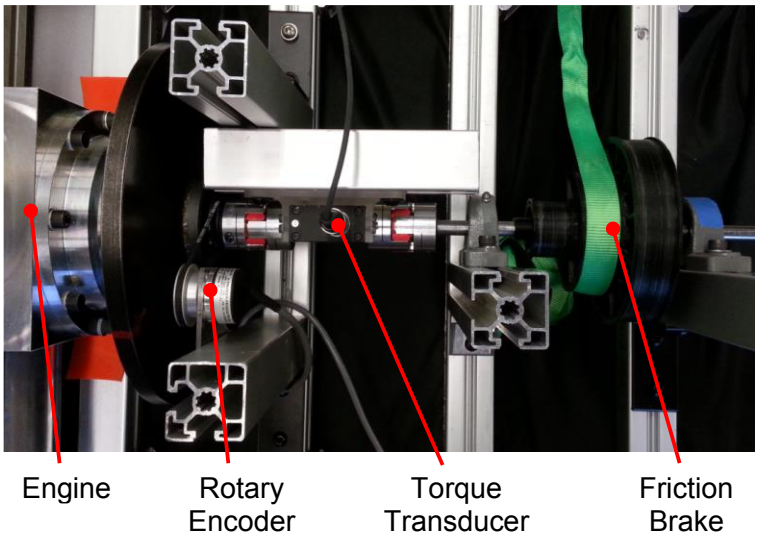
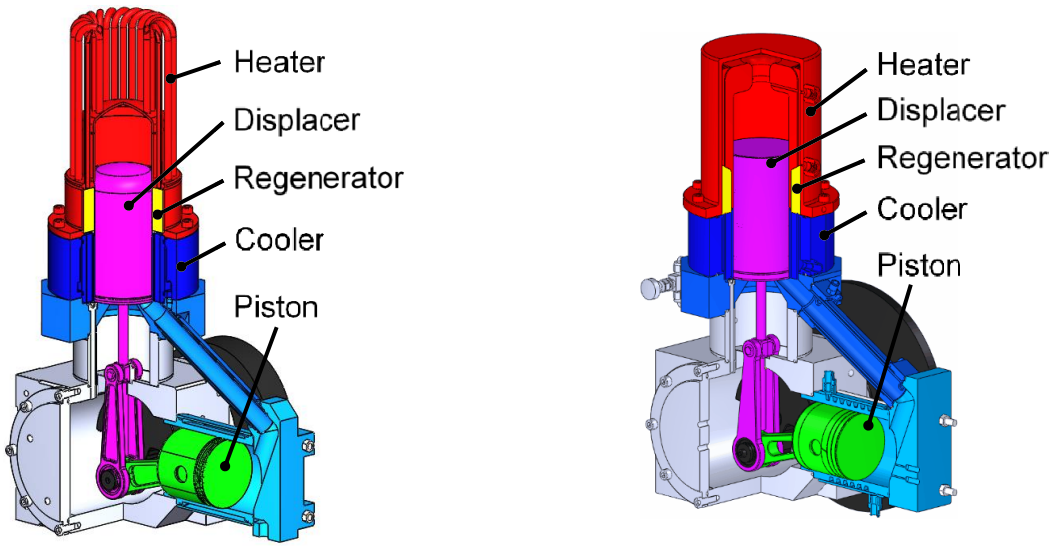


Figure 2.2: Top View of Engine Output Shaft

The test rig has several unique features which distinguish it from those of other research groups. The water jacket has been divided into two separate zones, each with their own flow rate and temperature measurements. This allows quantification of the relative influence of each zone on the overall heat rejection. The number and placement of instruments is also unique. The working fluid temperature is measured at the interfaces between the five components: compression space, cooler regenerator, heater, expansion space. The temperature and pressure of the crankcase is measured, so that the forced work and gas spring hysteresis loss can be quantified. Cold motoring tests can be performed by spinning the engine with an electric motor. Perhaps most important, the engine itself has been modified from the original ST05G-CNC design. Several modifications have been tested to determine their effects.

2.2 Modified ST05G-CNC Stirling Engine

The ST05G-CNC Stirling engine studied here was initially built with modifications to suit a solar power application. The overall solar power system was to include a Fresnel mirror solar concentrator, a thermal reservoir, a Stirling engine, and an electrical generator. The thermal reservoir was to be a solid graphite block, with a hole on one side into which the Stirling engine heater head would be inserted. The thermal reservoir was to achieve a maximum temperature of 650 °C. Figure 2.3 compares the original ST05G-CNC engine to the one built here. The drawing package for the modified design is provided in Appendix E.



(a) (b)

Figure 2.3: Rendered Solid Models of (a) Original ST05G-CNC Design (b) As-Built Engine for Solar Power Application

The most significant design change is the heater head, shown in red in Figure 2.3, which has been redesigned to mate with a solid thermal reservoir. The original heater head was intended to absorb heat from gaseous combustion products, [53], and consisted of 20 stainless steel tubes welded to the displacer cylinder. Stainless steel was likely used in the original design for its corrosion resistance and creep strength - important properties in a combustion environment. The modified heater head is built of mild steel for machinability, and painted for corrosion resistance. It passes working fluid through a slotted annular gap which surrounds the displacer cylinder. Two pieces are shrink-fit together to form the heater head assembly. The inner piece was turned,

then slotted with a CNC milling machine fitted with a slitting saw. The outer piece interference fit over the inner piece, and included weld-on NPT fittings for inserting instruments. Relative to the original design, the as-built heater head has a larger internal surface area, and a smaller internal volume.



Figure 2.4: Image of Liquid Nitrogen Shrink Fit Process for Heater Head Assembly

While the basic geometry was retained, numerous minor changes were made to the engine to facilitate manufacturing, and improve sealing. These included removal of aesthetic features, consolidation of assemblies into single parts, and adjustment of static seal designs. The number of O-rings was reduced from 21 to 15 by incorporating permanent seals using silicone. All radial seal O-rings were switched to face seals to simplify assembly, and all O-rings used standard sizes. The bypass valve was brought outside the engine so that a standard valve could be used. The piston seal was changed to allow use of standard PTFE sealing rings and wear rings.

Instrumentation related changes included adding ports for working fluid pressure and temperature measurement devices. All instrument holes were made to fit 5/16”-24 straight tap

Swagelok® tube fittings, except for the heater head ports which used NPT Swagelok® tube fittings instead, since temperatures there were too high to use an O-ring seal. Sheathed thermocouples were inserted to the desired depth and swaged in place to form a seal. Outboard pressure transducers were connected to the engine volume via 1/8" stainless steel tubes swaged in place. Additional holes were added to the crankcase cover and power cylinder head for flush-mount piezoelectric pressure transducers. The crankshaft was extended to facilitate connection to power measurement equipment.

Several adhesives were used during the assembly process. The three water jackets were sealed permanently using black silicone (Adhesive Sealant Black Silicone #81158, Permatex). The inlet/outlets for the power cylinder water jacket were attached using a combination of black silicone and acrylic adhesive (Scotch-Weld™ Acrylic Adhesive DP 820, 3M), because the 3M DP 820 only worked well if it stayed dry. Flange sealant (Loctite® 510™, Henkel) was tried as a sealant for the water jackets without success. Press fit retaining compound (Loctite® 603™, Henkel) was used for the big end bearings of the displacer connecting rods.

During the assembly and commissioning process, the most significant problem was the seal on the piston. The rings specified in the original ST05G-CNC design were difficult to obtain, so the piston geometry was altered to allow common pneumatic cylinder sealing rings to be substituted. The new sealing rings were made of glass reinforced PTFE, and used a nitrile O-ring backer to provide force normal to the cylinder wall. The rings were continuous, and were heated and stretched into the grooves in the piston with considerable difficulty. Once the sealing rings were fitted, inserting the piston into its cylinder was a challenge, due to the absence of a chamfer on the rim of the power cylinder bore. Ultimately, the sealing rings were too tight in the cylinder bore, and attempts to wear them in scratched the honed aluminum surface of the bore. It was determined that the sealing rings were unsuitable for the application. The author would advise against use of glass-filled PTFE in 6061 aluminum cylinders, for risk of scratching the cylinder surface. The displacer cylinder suffered a similar fate, prompting the search for alternatives. 3D-printed nylon split rings were tested next, with disappointing results. It was decided that a 3D-printed nylon ring would be sufficient for the displacer, but not for the piston. The final solution for the piston was a 50-durometer nitrile O-ring lubricated with PTFE grease (Magnalube-G,

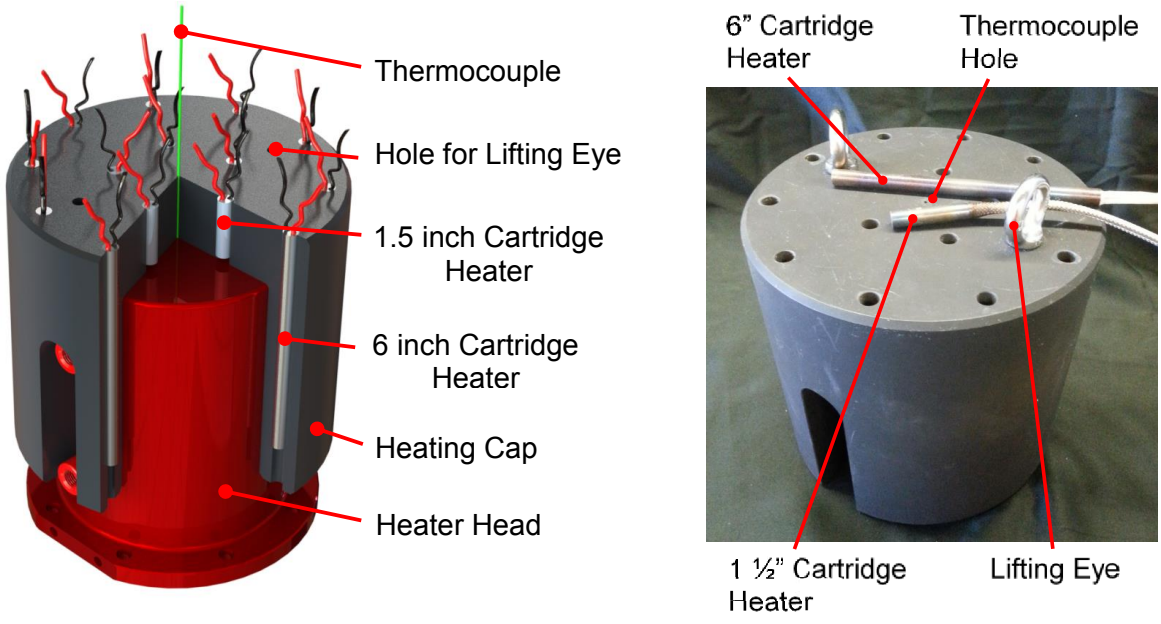
Saunders Enterprises Inc). While the O-ring sealed well, its durability was poor and frequent replacements were necessary during the testing campaign.

An additional sealing problem occurred during testing that proved difficult to diagnose. The displacer rod had been lightly greased during a recent reassembly, ironically, in an effort to improve its seal. The displacer rod seal is a plastic bushing press-fit into the aluminum displacer mount. The grease eventually found its way between the outside of the bushing, and its aluminum housing. The bushing slid out of the displacer mount causing the pressure between the crankcase and the working space to equalize. The crosshead on the displacer rod subsequently pushed the bushing back into place, restoring the seal. When the problem was finally pin-pointed, it was remedied by removing all grease from the displacer rod, and pressing in a new bushing.

2.3 Heating System

Figure 2.5 displays the heating system used on the Stirling engine test rig. The system is comprised of a mild steel heating cap, electric cartridge heaters, power electronics, thermal insulation, and a temperature controller which receives feedback from a thermocouple mounted in the center of the cap (shown in green in Figure 2.5 (a)).

The heating cap was designed with the solar power system in mind. Its large thermal inertia was intended to simulate the graphite thermal reservoir. It has been coated with high temperature paint for corrosion resistance. Holes for the heaters are reamed only to the correct insertion depth, but are drilled all the way through. This allows the cartridge heaters to be pushed out from the bottom if necessary. A 1/16" hole in the center of the cap houses a K-type thermocouple to provide feedback for the temperature controller. Lifting eyes may be attached to the cap, as shown in Figure 2.5 (b), to facilitate removal and installation. Once assembled onto the engine, the heating cap is covered with several layers of alumina silica ceramic fiber insulation.



(a) (b)

Figure 2.5: (a) Rendered Section View of Heating Cap CAD Model (b) Image of Heating Cap and Cartridge Heaters As-Built

In total, there are 15 cartridge heaters, with a combined power rating of 5 000 W. When tested, it was found that the system draws ~3 291 W of electrical power when running. The heaters are rated to withstand temperatures above 650 °C. There is an outer ring of ten 3/8" x 6" heaters and an inner ring of five 3/8" x 1.5" heaters. The leads of the cartridge heaters are insulated for temperatures up to 450 °C and are routed normal to the cap to keep them as cool as possible. The heaters receive single phase power at 208 Volts.

The power supply circuit for the cartridge heaters is housed in an electrical box mounted on the test rig cart. The box is visible in Figure 2.1. It contains fuses on each of the incoming hot wires, a solid-state relay driven by the temperature controller, and bus bars to distribute power evenly to the cartridge heaters. Power to the heaters is either on or off; there is no intermediate control. All cartridge heaters receive the same voltage, since they are all connected to the same two bus bars. Both the lid and the box itself have been grounded for safety. An emergency cut-off switch has been wired into the main power cable, so that power can be quickly shut-off without going near the electrical box itself.

A temperature controller (CN8DPT-440-C24, Omega Engineering Inc.) is used to signal the solid-state relay which supplies power to the cartridge heaters. It receives feedback from a single K-type grounded junction thermocouple (KMTSS-062G-6, Omega Engineering Inc.) mounted in the center of the heating cap as shown in Figure 2.5 (a). It uses a simple on/off control scheme. The power is turned off when the cap temperature reaches the set point. Power is restored when the cap temperature drops 2 °C below the setpoint.

Experience has shown that the cap temperature will overshoot the setpoint by roughly five degrees, even though power is removed at the setpoint temperature. It has been inferred that during the heating process, there are parts of the cap which reach temperatures above that measured by the thermocouple. When the power supply is turned off, the temperature equalizes throughout the cap, bringing the temperature of the thermocouple above the setpoint.

It is useful to know the average amount of time that the heating system is on during a test, to estimate the average heat input rate. For this purpose, the voltage signal from the temperature controller is sent to a second solid state relay, in addition to the one which switches the heater power supply. A schematic of the circuit associated with this second relay, is provided in Figure

2.6 below. When the relay is closed, the voltage from the power supply is measured by the high speed DAQ device (USB-6211, National Instruments Inc.). When the relay is open, the high speed DAQ is pulled down to ground voltage through its connection to the negative terminal of the power supply. The power supply voltage was set to 8 V, with the current set as low as possible.

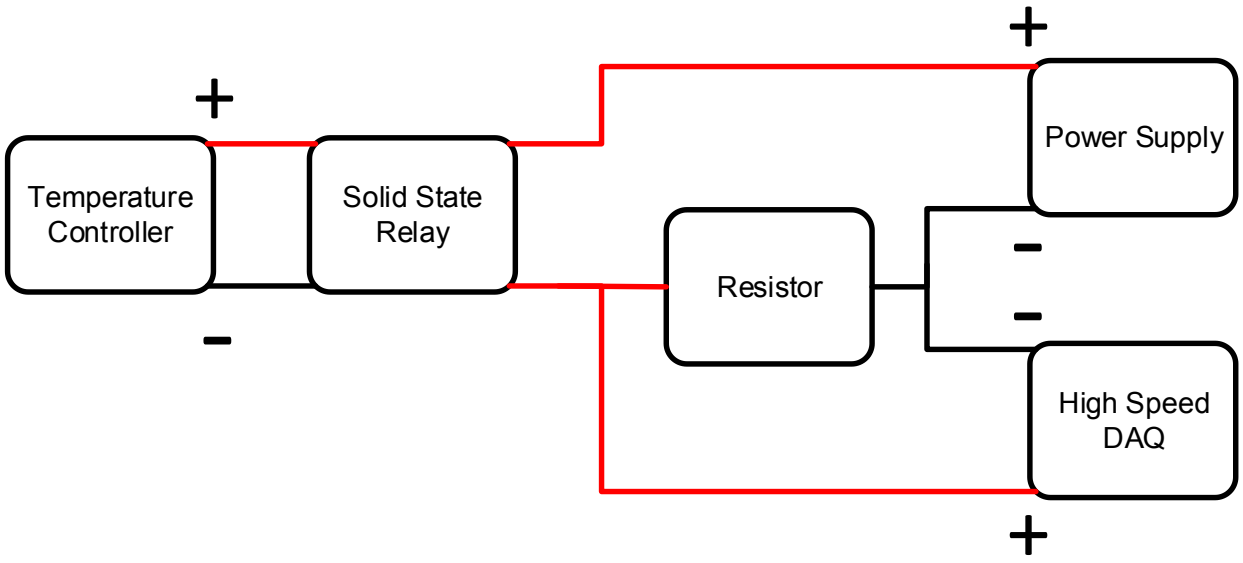


Figure 2.6: Schematic of Circuit for Heat Input Measurement System

2.4 Cooling System

The engine is cooled by a recirculating water loop. The temperature of the water is maintained by a refrigerated water bath. The engine water jackets are divided into two separate cooling zones. The flow rate through each zone is controlled by peristaltic pumps. Inlet and outlet temperatures of each zone are measured using resistance temperature detectors (RTDs). Using these measurements, the heat rejection rate may be calculated.

The water bath (12101-41, Cole-Parmer Canada Company) can heat or cool the water to maintain the setpoint. It has a cooling capacity of 900 W, which is not sufficient for heater cap temperatures above 400 °C. As a temporary solution, ice was used to maintain the water bath temperature if its cooling capacity was exceeded. The water bath has a built-in pump. This pump was used to circulate water through the engine if the peristaltic pumps were not being used.

The water jacket was divided into two separate cooling zones: The cooler, and the connecting pipe and power cylinder. The zones are shown in Figure 2.7. The cooler water jacket surrounds the internally finned tube, that contains the gaseous working fluid. The connecting pipe is double-walled, with water flowing between the inner pipe and the outer pipe. Inlet/outlets for the connecting pipe water jacket are offset to encourage water to flow in a helical pattern around the inner pipe. The power cylinder water jacket contains a large thread profile, to guide water in a helical path around the cylinder. The flow of water is from bottom to top, to aid in the removal of air bubbles when pumping begins.

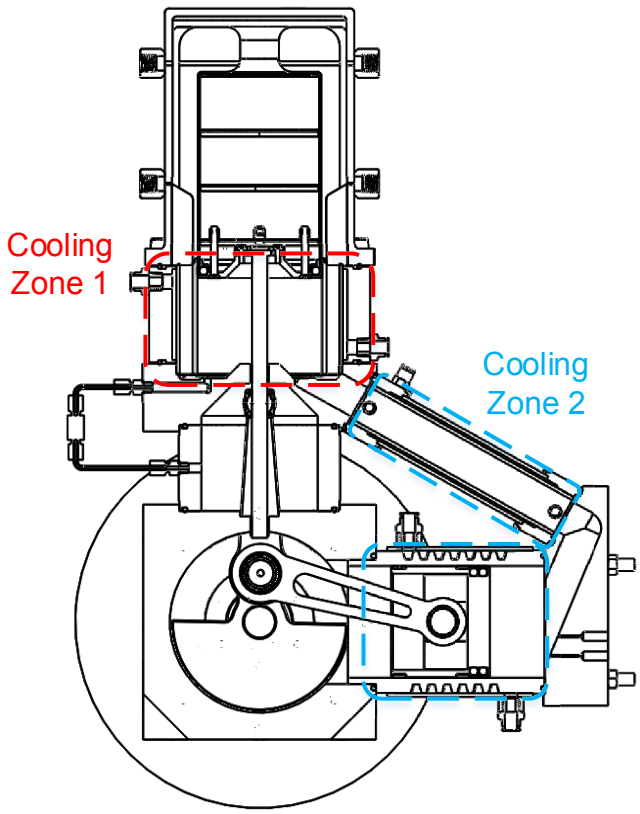


Figure 2.7: Schematic Defining the Two Cooling Zones

For preliminary tests, control of flow rate for each cooling zone was provided by acrylic flowmeters (75302113C08, King Instrument Company). The acrylic flow meters had maximum capacities of 3.5 GPM for the connecting pipe and power cylinder, and 5 GPM for the cooler. In practice, it was found that flow rates below 1 GPM were needed to obtain a measurable temperature rise across the water jackets. At these flow rates, the accuracy and consistency of the flow meters was poor.

In cases when the flow meter accuracy was insufficient, two peristaltic pumps were used (Model # 07551-20, Cole-Parmer® Masterflex® L/S® pump drives fitted with Model # 77200-62, Easy-load® II pump heads). One supplied water to the cooler, and the other supplied water to the power cylinder and connecting pipe. Both pumps were calibrated at the constant flow rates used for all tests, which were 0.800 L/min and 0.200 L/min for the cooler and connecting pipe/power cylinder, respectively. The suction line for each pump drew water from the refrigerated water bath, and water was returned to the bath after passing through the engine.

Water temperature measurements were taken at the inlet and outlet of each cooling zone using 3-wire RTDs (RTD-810, Omega Engineering Inc.), which were housed in T-fittings placed in the water hoses as close to the engine as practical. The RTD signals were conditioned using an RTD-specific data acquisition device (NI 9217 RTD, National Instruments Inc.), before reaching the data acquisition computer. Excessive noise problems during preliminary tests using grounded junction K-type thermocouples, prompted the switch to RTDs for the final version of the test rig.

2.5 Braking System

The test rig includes a braking system, used to apply various loads to the engine. Figure 2.2 shows the braking system coupled to the engine output shaft. A 3D-printed spool-shaped part is rigidly connected to the output shaft. A length of 1-inch nylon webbing passes over the spool, and rubs against it to apply a load to the engine. Weights are hung from the webbing to alter the amount of braking force. The spool also has three running surfaces, each with a different radius, which allow further load adjustment.

Initial tests were done using a bicycle disk brake to apply loads. A similar system of hanging weights was used to vary the resistance against the engine. The inconsistency of the disk brake at light loads made it unsuitable for low power testing. Light loads could not be applied without inadvertently stalling the engine.

2.6 Motoring Test System

Cold motoring tests can be performed with slight adjustments to the test rig. Figure 2.8 provides a top view of the test rig configured for a cold motoring test. An electric motor (CPM-MCVC-3441S-RLN, Teknic Inc.) is used to drive the engine crankshaft. The torque-limiting coupling protects the torque transducer from torques above 10 Nm. Torque and speed are measured by the torque transducer and rotary encoder. Cold motoring tests are safer and take less time than hot tests. They have been used to measure friction, and pressure drops across heat exchanger components.

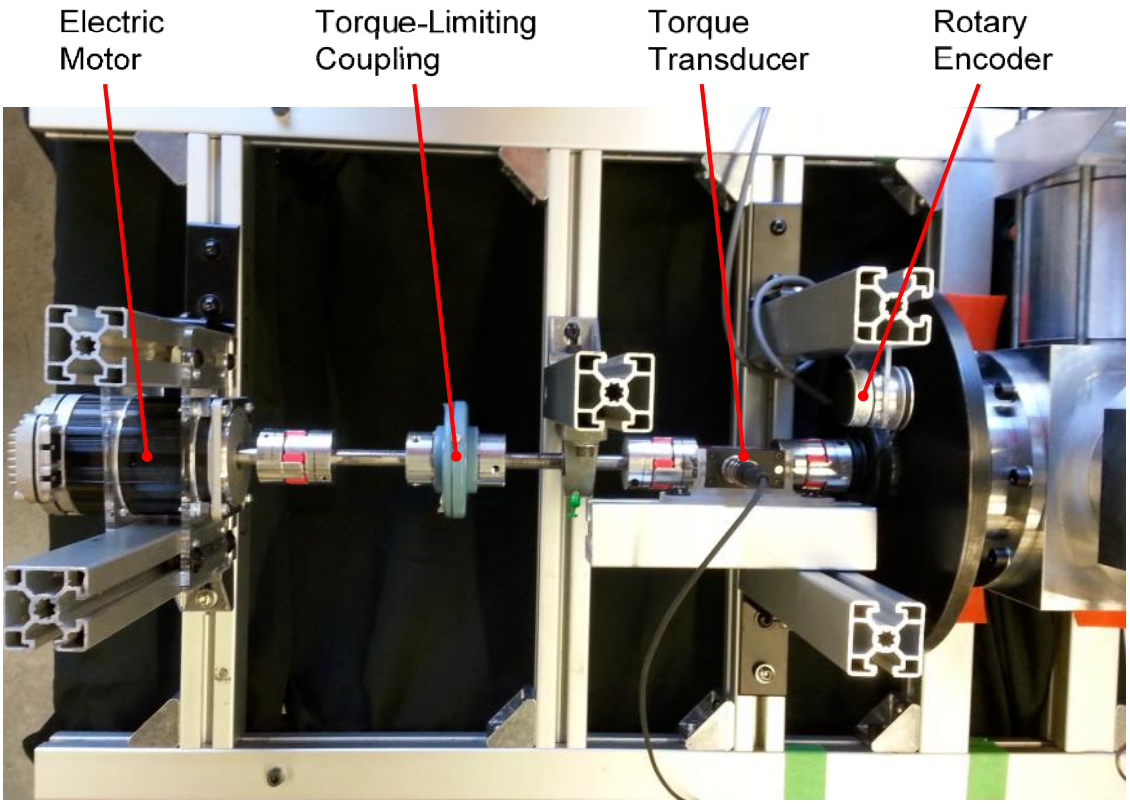


Figure 2.8: Top View of Test Rig Set-Up for Cold Motoring Tests

2.7 Instrumentation System

A block diagram of the instrumentation system is given in Figure 2.9 below. Measurements of gas temperature, gas pressure, angular position, torque, heat input rate, and heat rejection rate will be discussed in the following sub-sections. Relevant sources of error will be discussed for each.

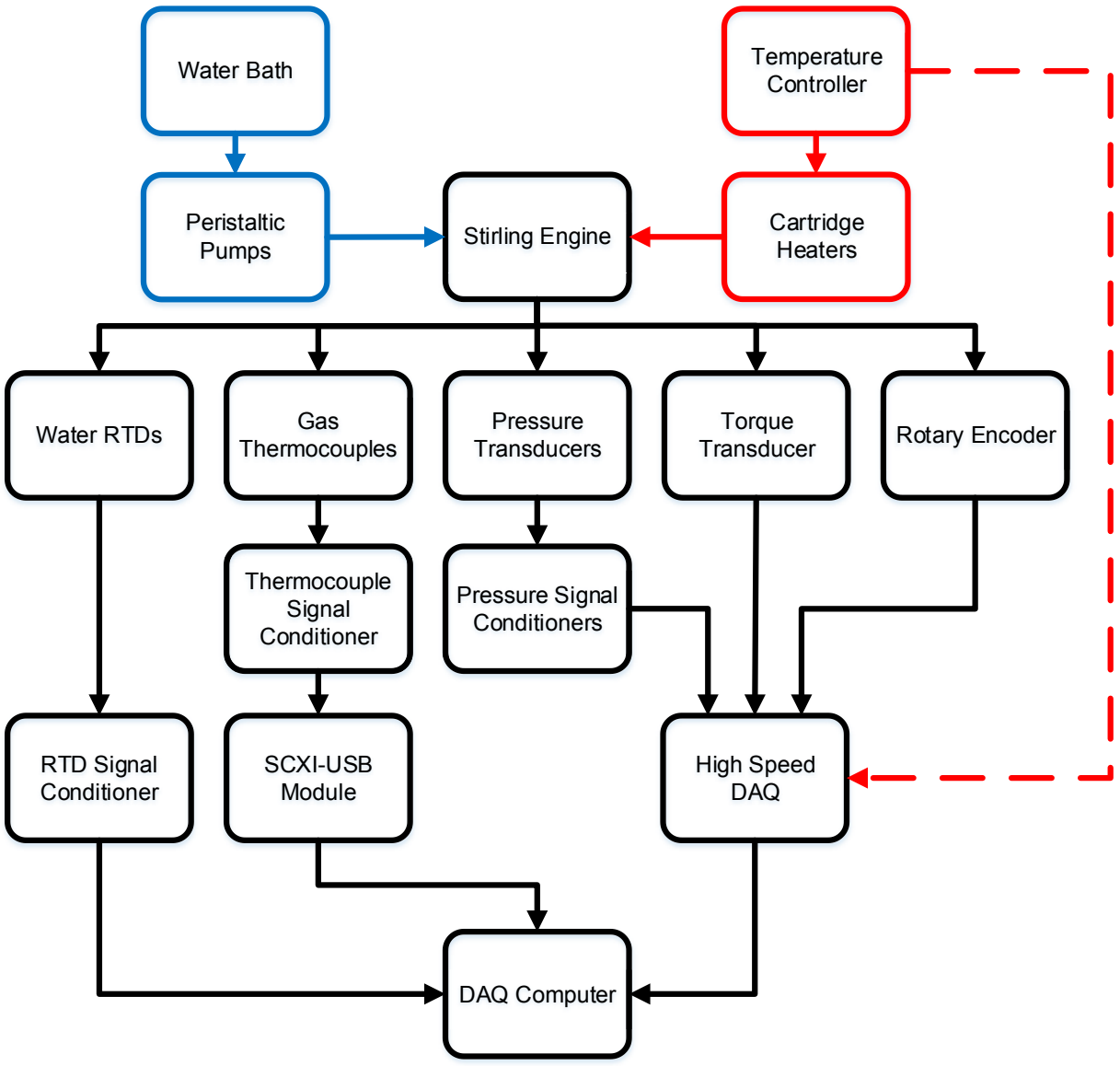


Figure 2.9: Block Diagram of Instrumentation System

2.7.1 Gas Temperature Measurement System

A total of seven gas temperature measurement locations are monitored during a typical experiment. The locations of these measurements are shown by the red stars in Figure 2.10. The temperatures of the interfaces between the five engine components are all measured. The cold side of the regenerator is measured on the right and the left side, with the aim of indicating flow asymmetry. The crankcase temperature is measured for comparison with buffer pressure calculations. The heater head may be rotated to eight different positions, to change the circumferential position of the instruments, if desired.

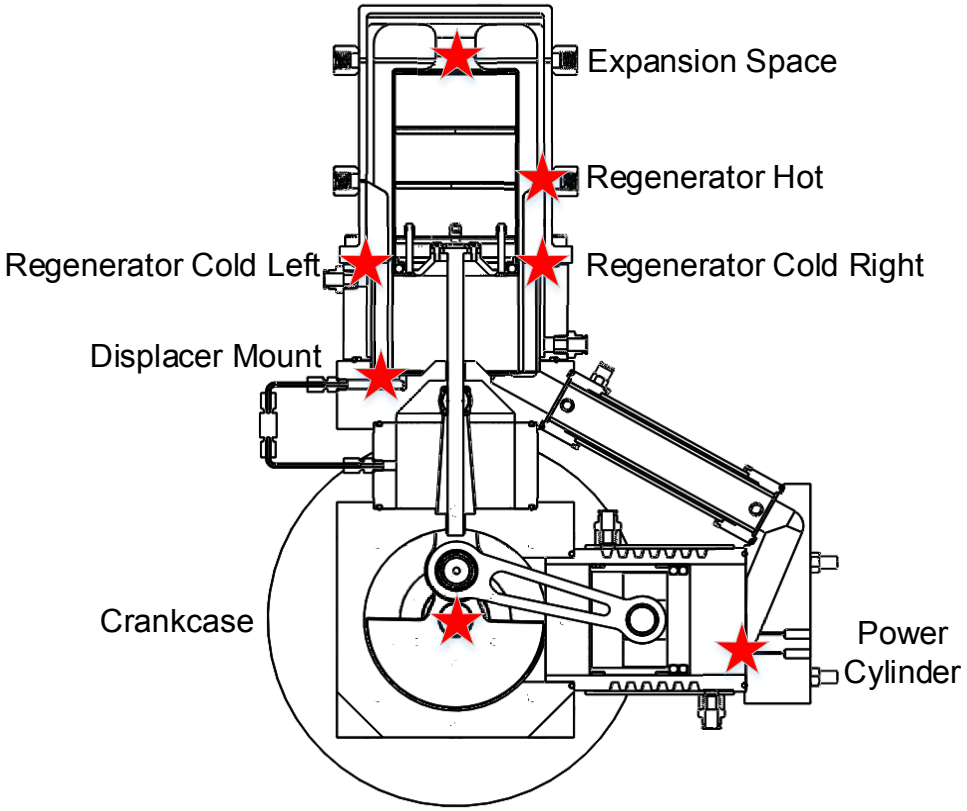


Figure 2.10: Locations of Gas Temperature Measurements

Gas temperatures were measured using exposed-junction K-type thermocouples (HKMTSS-062E-6, Omega Engineering Inc.). An example of a thermocouple used to measure gas temperature is given in Figure 2.11. The thermocouples were housed in 1/16" stainless steel sheaths. Swagelok® tube fittings sealed onto these sheaths to prevent leakage. The average junction diameter of the seven gas temperature thermocouples was measured to be 0.77 mm.

Based on this, the time response of the thermocouples was estimated to be 3.6 s, and a full rotation takes 0.6 s at 100 RPM. Thus, only average gas temperatures could be used.

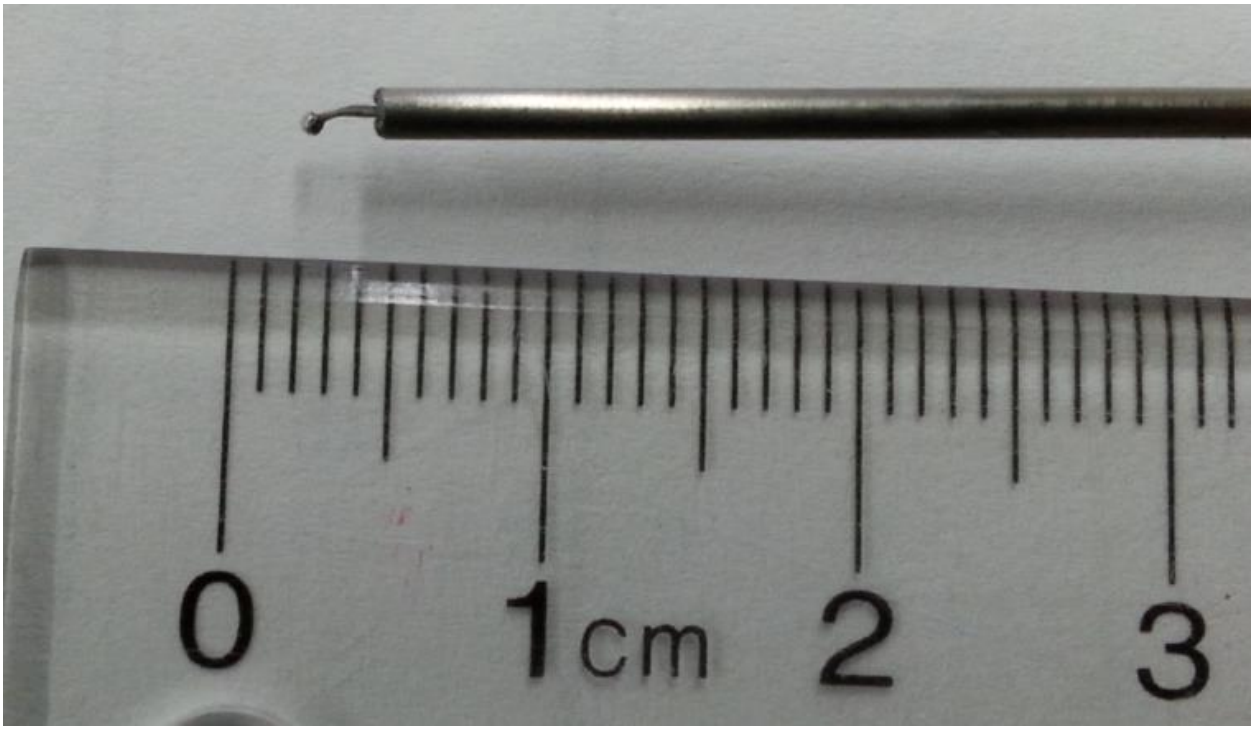


Figure 2.11: Image of Gas Thermocouple Tip

Thermocouple signals are received by a 32-channel thermocouple terminal block (TC-2095, National Instruments Inc.), passed to a DAQ chassis (SCXI-1000, National Instruments Inc.) by a thermocouple module (SCXI-1102B, National Instruments Inc.), and converted from analog to digital by a USB module (SCXI-1600, National Instruments Inc.), before reaching the DAQ computer. Gas temperature signals are typically sampled at 5 000 Hz.

Figure 2.12 is a specimen raw data set of gas temperature measurements collected while the engine was running at steady state. Regular fluctuations in gas temperature are visible in the raw data. Figure 2.13 reveals that gas temperatures vary periodically with crankshaft position. The measured amplitude and phase of these temperature swings is not expected to be accurate due to the slow response time of the thermocouples relative to the cycle frequency. All analysis here is done using gas temperatures averaged over the entire data set.

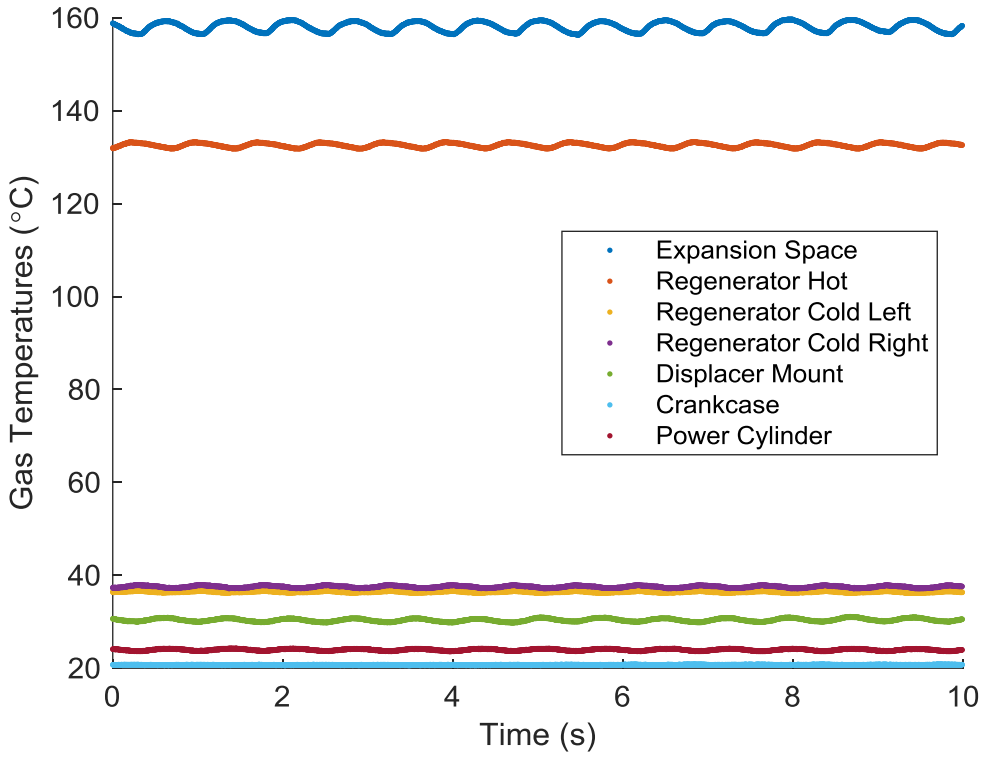


Figure 2.12: Example of Raw Data Set for Gas Temperature Measurements

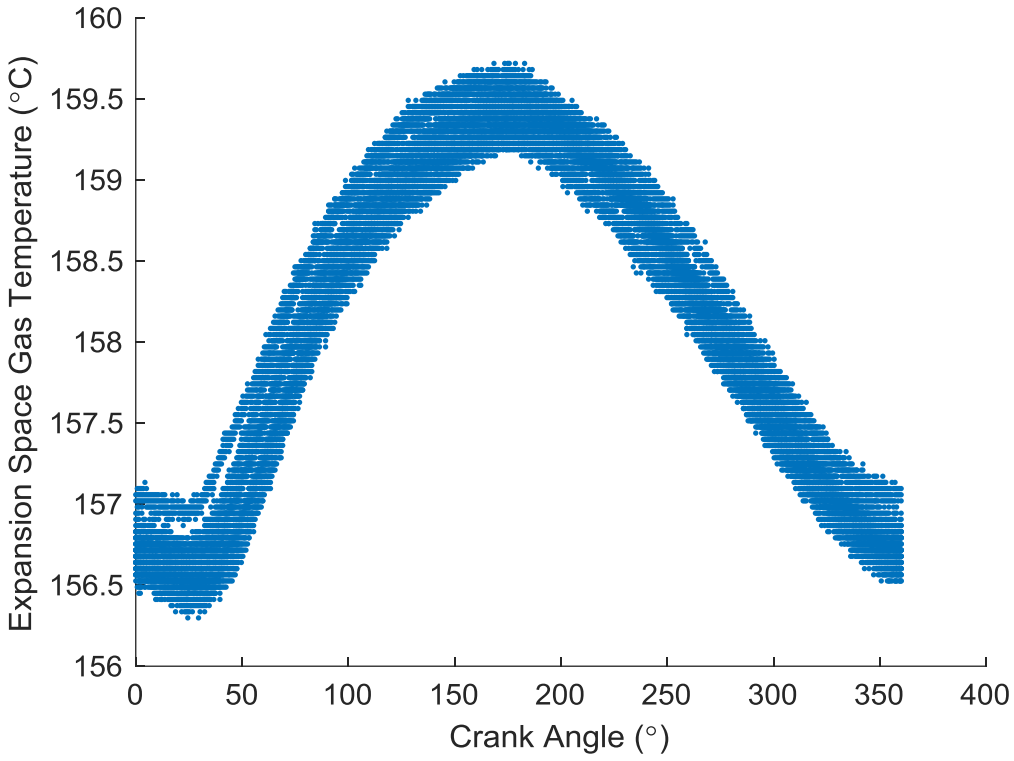


Figure 2.13: Measured Expansion Space Gas Temperature as a Function of Crank Angle

During testing, it was discovered that thermocouples would exhibit excessive electrical noise if their exposed junctions were in contact with solid parts of the engine. This was a problem for the regenerator thermocouples, which would often touch the regenerator matrix when inserted. The issue was remedied by fashioning a small wire cage to hold the regenerator material away from the thermocouple junction. Wire cages were used at the regenerator cold right and regenerator cold left thermocouple locations.

The thermocouples were calibrated by immersing them in a water bath at several different temperatures. At each temperature, a 10-second data set was taken at 5 000 Hz. The average temperature for the group of thermocouples was taken to be the "true" temperature. For each individual thermocouple, the average deviation from the true temperature was calculated. This was repeated for several temperatures. Then, a third order polynomial curve was fitted to the deviations for each individual thermocouple. Correction terms calculated by these polynomial curves were used to adjust the raw thermocouple data.

This procedure was used for all thermocouples except for those at the expansion space and regenerator hot locations. These two were omitted because they are typically measuring gas temperatures that are much higher than those reached during the calibration tests. Extrapolation of the correction term curves to these temperatures is likely to increase the error relative to taking the temperatures raw.

Figure 2.14 compares the measured temperature error before and after application of the correction terms. The marker colors in the plots identify individual thermocouples tested. The error was calculated as the difference between the measurement in question, and the overall average for that test calculated using all the thermocouples. Since the thermocouples are only being corrected relative to each other, the error shown in the plots is relative not absolute. The correction procedure improves measurements of temperature differences between thermocouples, but does not necessarily decrease the uncertainty in absolute temperature measurements.

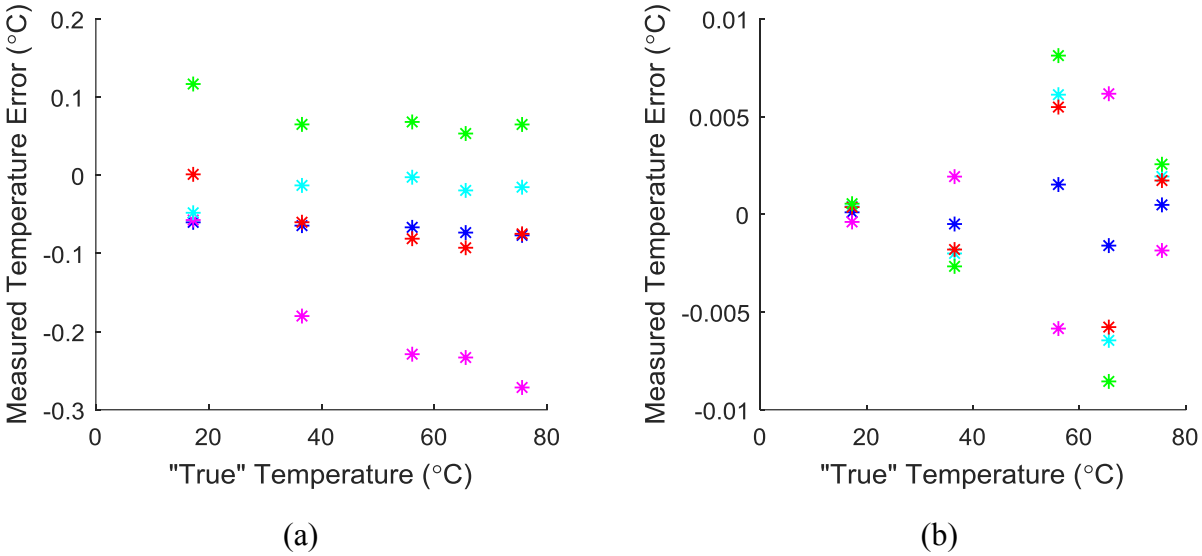


Figure 2.14: Difference between Temperature Measured by Thermocouples and Overall Average Temperature (a) Before Applying Correction Terms and (b) After Applying Correction Terms

2.7.2 Pressure Measurement System

Figure 2.15 indicates the six locations at which gas pressures may be measured. The expansion space pressure port includes a 1/8” stainless steel tube, which passes from the weld-on NPT fitting to the inlet of the expansion space. All other ports have 1/16” or 1/8” holes drilled through the engine walls. Differential measurements across the heater, cooler, and regenerator are possible using a single transducer configured as shown in Figure 2.16.

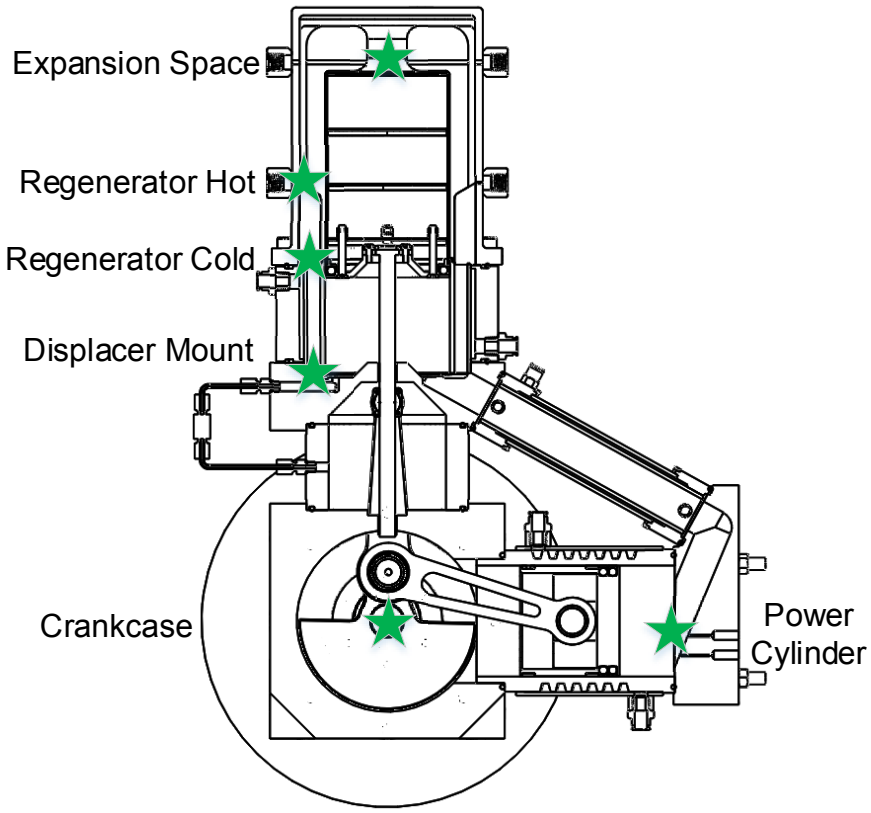


Figure 2.15: Locations of Gas Pressure Measurements

During the first round of tests, only outboard pressure transducers (DP15-50, Validyne Engineering) were used. The transducers were connected to the engine using 1/8” stainless steel tubes, which were kept as short as possible. In general, they were about 2 cm in length; however, they could be up to 12 cm if used for a differential measurement as in Figure 2.16.

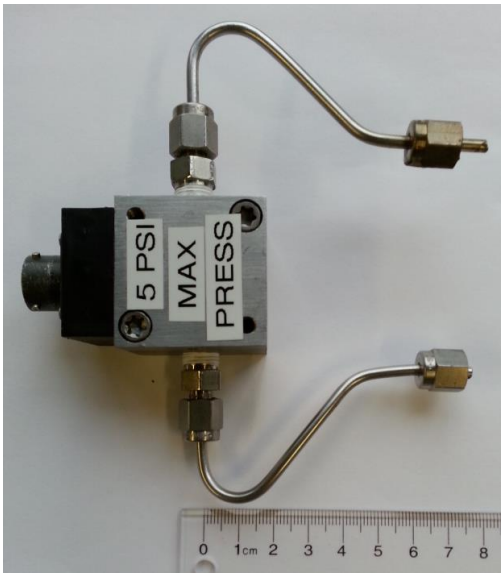


Figure 2.16: Image of Tube Set-Up for Regenerator Differential Pressure Measurement

Analysis of regenerator pressure drop data, via Figure 2.17, revealed a significant problem. The figure plots data collected during a cold motoring test. Figure 2.17 (a) shows the effect of rotational speed and Figure 2.17 (b) shows the effect of fill pressure, on the measured differential pressure across the regenerator. A pressure gradient which is descending away from the crankcase appears positive in the plots. It was expected that the measured pressure curves would match each other in terms of phase, because the pressure drop is thought to be a strong function of flow velocity. Since the maximum flow velocity will occur at approximately the same crank angle for a range of speeds, the maximum pressure drop should follow. Figure 2.17 shows a significant phase difference in the regenerator pressure drop for different speeds. Losses in the holes/tubes associated with the outboard transducer system were suspected of introducing measurement lag.

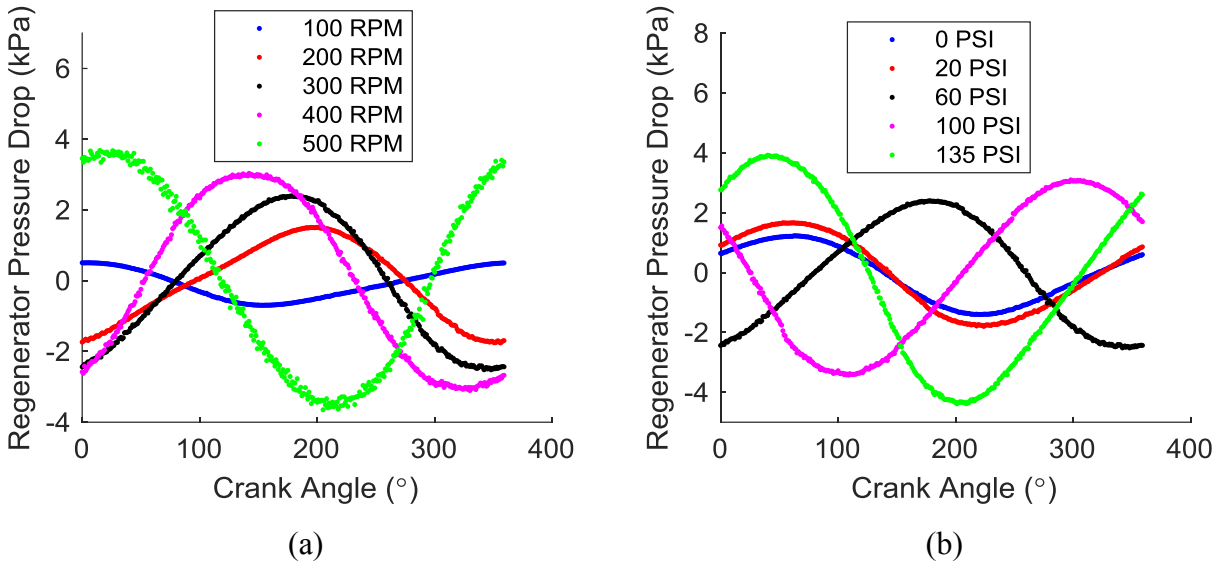


Figure 2.17: Regenerator Pressure Drop as a Function of Crank Angle for (a) Several Rotational Speeds at a Constant Fill Pressure of 414 kPa (b) Several Fill Pressures at a Constant Rotational Speed of 5 Hz

Lag in the pressure measurements could significantly skew the experimental results. To illustrate, Figure 2.18 was produced. The figure shows an experimental indicator diagram. The area inside the loop represents the work done by the engine in a single rotation of the crankshaft. The shape of the loop, when combined with the buffer pressure measurement, can be used to quantify the forced work. The black curve shows unaltered experimental data. The red and blue curves were artificially phase shifted to show the impact of pressure measurement lag. Evident from the figure, is that any phase shift in the pressure measurement will change the indicator diagram shape and area, making comparison to modeling results dubious.

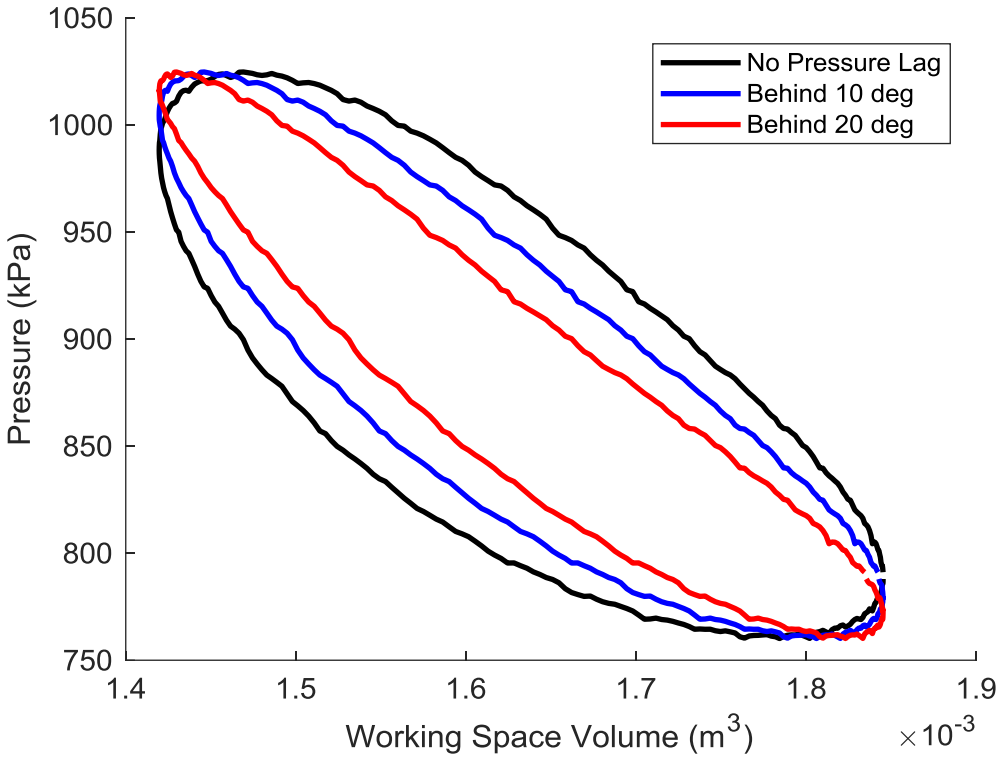


Figure 2.18: Potential Impact of Lag in Pressure Measurement on Indicator Diagram Shape

Suspicions were confirmed by performing tests with two different tube lengths, shown in Figure 2.19. The two tubes were each connected to the power cylinder measurement port and put through a series of cold motoring tests. The same transducer was used for both sets of tests. Each tube was tested at engine speeds of 100 RPM and 400 RPM, and engine pressures of 138 kPa (20 PSI) and 827 kPa (120 PSI).

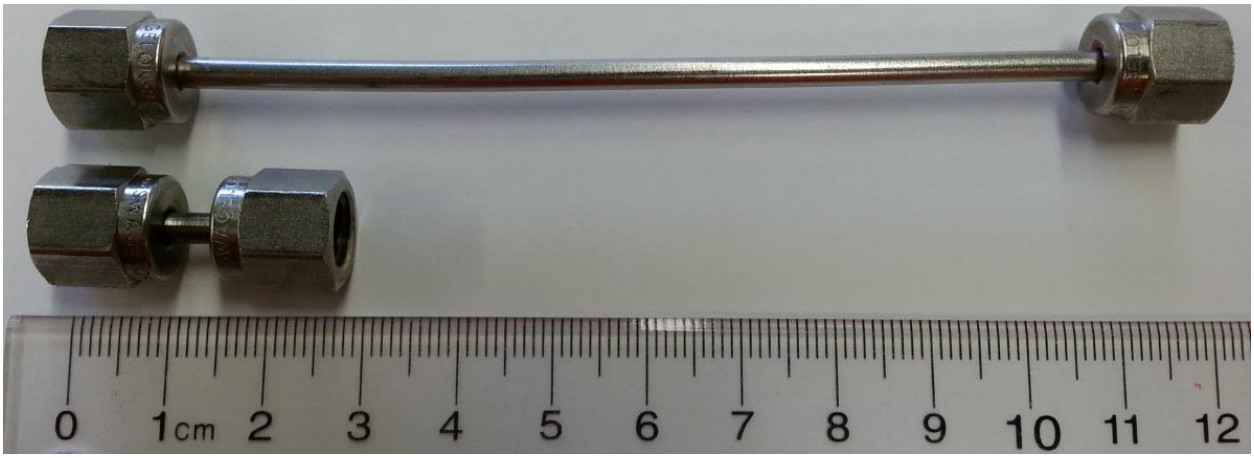


Figure 2.19: Tubes Tested to Confirm Pressure Measurement Lag

Figure 2.20 displays the results of the pressure tube test. Pressure values have been normalized to remove slight differences in fill pressure, between the two tests, from the plots. To normalize a set of data for a given speed, fill pressure, and tube length, the average of the data set was subtracted from the individual pressure samples. Inspection of the plots reveals that the tube length affects the phase lag by an amount that depends on both the speed and the pressure.

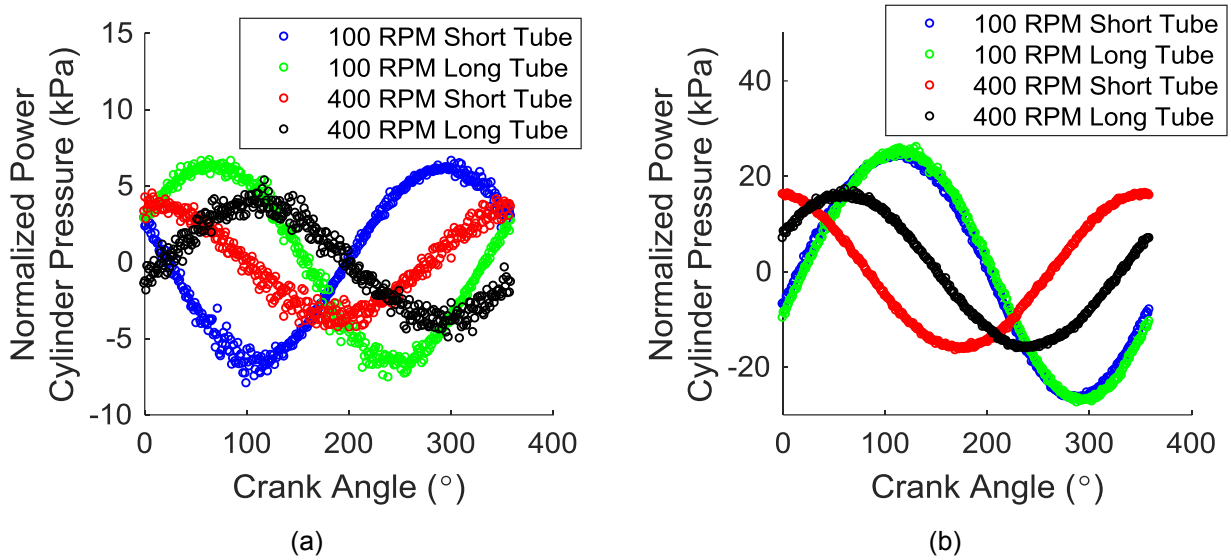


Figure 2.20: Normalized Power Cylinder Pressures for Fill Pressures of (a) 138 kPa (20 PSI) and (b) 827 kPa (120 PSI)

Yang et al., [86], describe a method of correcting for distortion caused by tubing in outboard transducer unsteady pressure measurements. They point out that the tubes may amplify the

pressure fluctuations, due to resonance effects, or attenuate them, due to viscous effects. Which occurs will depend on the frequency of the unsteady pressure. They also state that time-lag is introduced by using tubes, which increases the phase offset of the pressure signal for increasing frequency. The correction method that Yang et al., [86], describe, requires that a transfer function for the tube is determined experimentally, by comparison with a reference transducer mounted at the measurement location directly. Since every pressure port on the engine has a different hole and tube geometry, each would have its own unique transfer function. For many of the ports, the experiments required to determine the transfer function are not possible with the current set-up. For the power cylinder and crankcase, new pressure transducers were installed to circumvent the problem.

The new pressure transducers were mounted inside the engine, flush with the engine wall. The first transducer was mounted flush with the inside of the power cylinder head, and the second was mounted flush with the inside of the crankcase cover. Separate holes were cut to allow the new transducers to be installed alongside the old transducers, so that measurements could be compared.

The final pressure measurement system included two types of pressure transducers, their associated signal conditioners, and an analog input DAQ device. The outboard transducers (DP15-50, Validyne Engineering) could be used as single point or differential pressure measurements, depending if one or two of the transducer's ports were used. The outboard transducers also had replaceable diaphragms, which allowed the measurement range to be adjusted. For single point measurements, diaphragms rated to 1 034 kPa (150 PSI) were used. A diaphragm rated to 34.5 kPa (5 PSI) was used for differential measurements. Signals from the transducers were conditioned by a demodulator (CD280-8, Validyne Engineering), before reaching the analog input DAQ device. The flush mounted transducers (113B21, PCB Piezotronics Inc.) use the piezoelectric properties of quartz crystals to convert the pressure signal to an electrical signal. The electrical outputs pass through a signal conditioner (482C05, PCB Piezotronics Inc.) on their way to the analog input DAQ. The analog input DAQ (USB-6211, National Instruments Inc.) samples the pressure signals at cumulative rates up to 250 000 Hz. This allows resolution of the cyclic pressure variation up to the maximum rated engine speed of 12.5 Hz (750 RPM).

The outboard pressure transducers were calibrated with reference to a pressure calibration machine (DPI 603, Druck). The zero and span were adjusted on the signal conditioner for each outboard transducer individually. The transducers were then connected to the pressure calibration machine as a group, and data was collected at several static pressures. The reading on the calibration machine was taken to be the true pressure, and correction terms were determined for each transducer at each static pressure. Third order polynomial curves were fitted to the correction factors, and the equations of the curves were used to adjust the raw data. Figure 2.21 shows the reduction in measured pressure error obtained by applying the correction terms to the raw data of the outboard transducers measuring absolute pressures. Figure 2.22 presents analogous plots for the outboard transducer which measures the regenerator pressure drop.

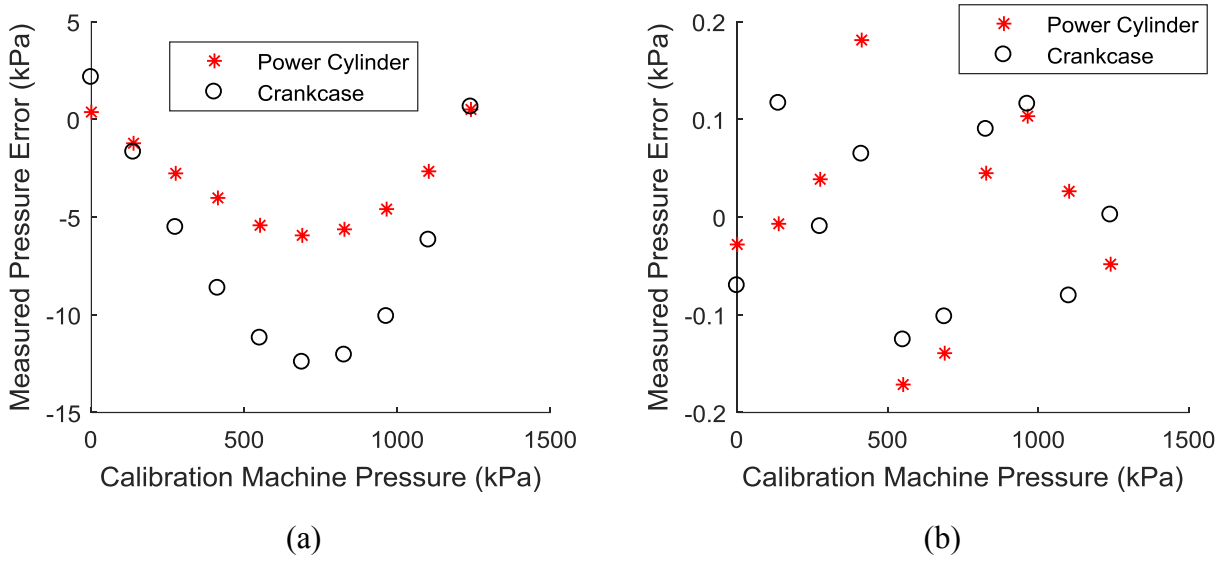
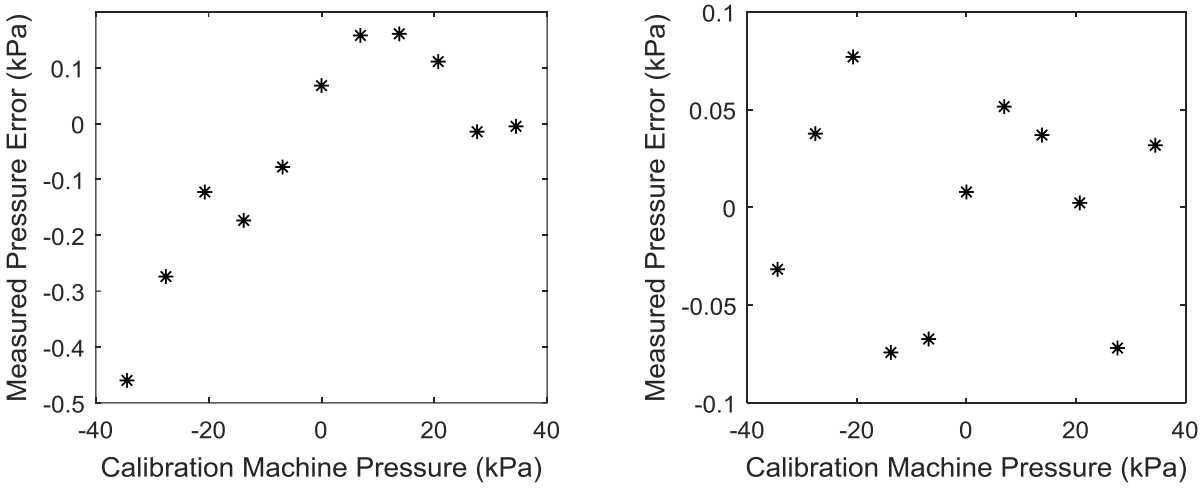


Figure 2.21: Difference between Measured Pressure and Calibration Machine Setpoint (a) Before Applying Correction Terms and (b) After Applying Correction Terms for Outboard Transducers at the Crankcase and Power Cylinder Locations



(a) (b)

Figure 2.22: Difference between Measured Pressure and Calibration Machine Setpoint (a) Before Applying Correction Terms and (b) After Applying Correction Terms for a Single Outboard Transducer Connected between the Regenerator Hot and Regenerator Cold Locations

The flush-mount piezoelectric pressure transducers were calibrated at the factory. They are only capable of measuring dynamic pressures. When exposed to a periodic pressure, their voltage outputs will quickly center on a mean value of zero volts; hence they provide no information on the absolute pressure. Here, the outboard transducers were used to determine the absolute mean pressure, and the flush-mount transducers were used to measure the dynamic pressure. Voltages of the flush mount transducers were converted to pressures using the sensitivities provided in the factory calibration documents.

An example of raw pressure data is given in Figure 2.23. A slight difference between the two transducer types is visible. Figure 2.24 shows that the Validyne outboard transducers tend to over predict the indicated work relative to the PCB flush mount transducers.

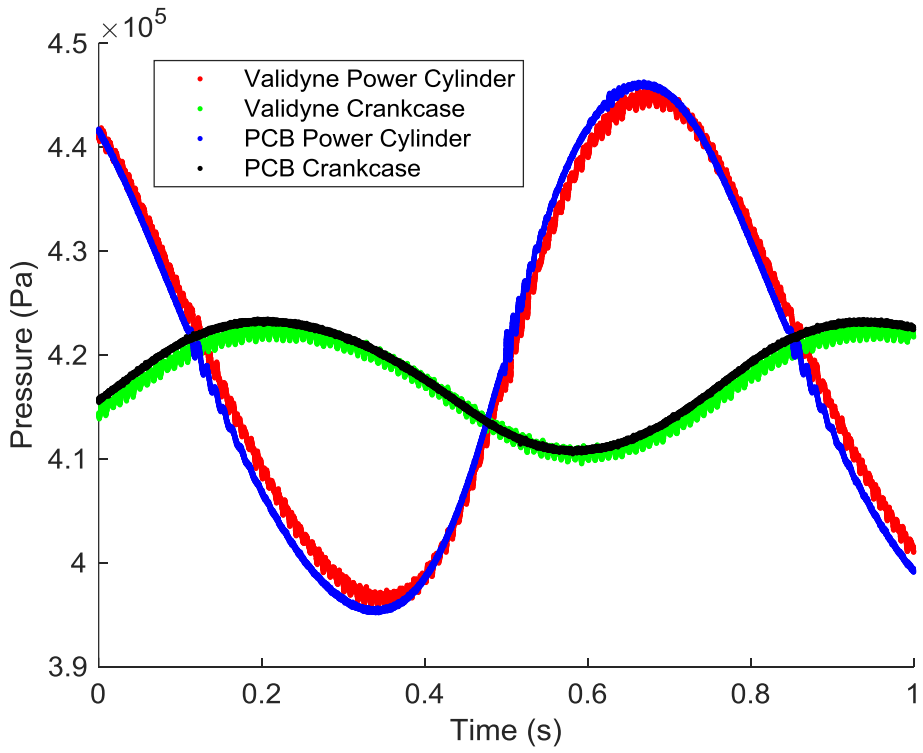


Figure 2.23: Example of Raw Data for Pressure Measurement

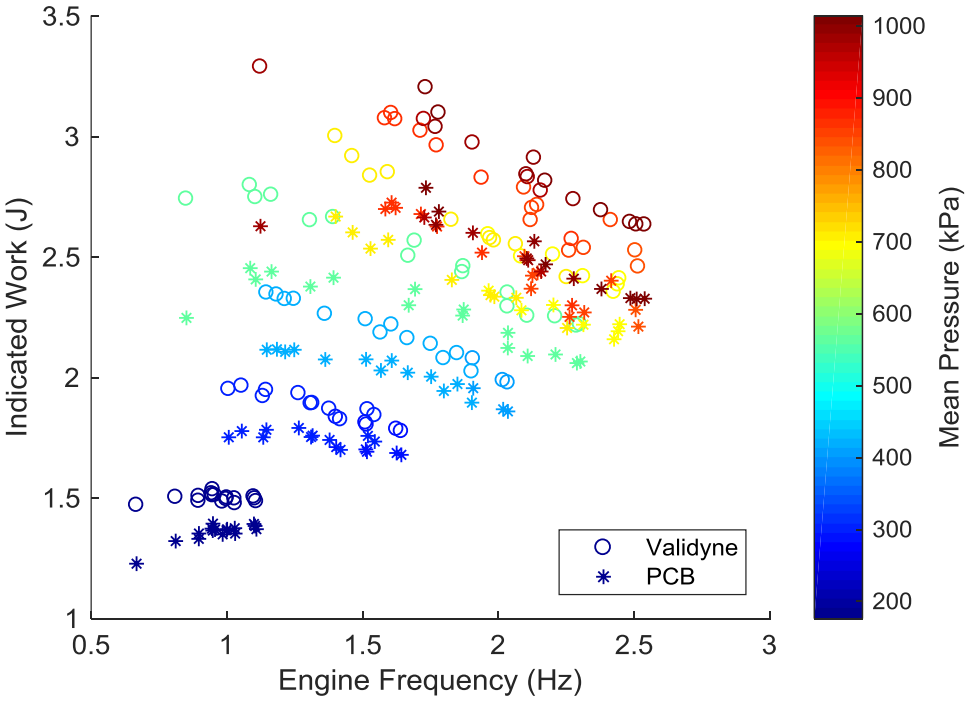


Figure 2.24: Comparison Between Validyne and PCB Pressure Transducers for Measuring Indicated Work

2.7.3 Angular Position Measurement System

The angular position of the crankshaft was measured using a 500 pulse per revolution (PPR) incremental rotary encoder (15S-19M1-0500NV1ROC-F03S1, Encoder Products Company), connected to the crankshaft with an MXL timing belt and a pair of toothed pulleys. The rotary encoder has three analog voltage outputs: A, B, and Z. The Z-output emits one PPR, and is used as a reference point. The A and B outputs each emit 500 PPR, but have a slight phase offset relative to each other. This allows rotation direction to be discerned if all three outputs are used. Since the rotation direction is always the same for this application, only the A and Z outputs were needed. The pulley on the encoder was purchased, while the pulley on the crankshaft was 3D-printed. The outer diameter and number of teeth were matched for the two pulleys.



Figure 2.25: Image of the Angular Position Measurement Set-Up

To align the angular position measurement system, the following procedure was followed. First, the crankcase cover was removed, providing a clear view of the crankshaft position. The engine was then positioned so that the working space volume was maximum (i.e. the piston was at bottom dead center). With the belt removed, the rotary encoder pulley was rotated until the Z output voltage increased to 5 V, indicating the reference point had been reached. The timing belt was then installed, and tightened by sliding the acrylic rotary encoder mount. The engine was then rotated to an arbitrary position, and returned to the position of maximum engine volume. If

the Z output was confirmed to again show 5 V, calibration was complete. Otherwise, the relative position of the pulleys was re-adjusted.

The rotary encoder signals were collected using the same DAQ device used for the pressure signals (USB-6211, National Instruments Inc.). Only the A and Z-outputs were collected. An example of the raw data is presented in Figure 2.26. The post processing scheme started from the Z-pulses, and counted the A-pulses forward and backward from there. The result was that every sample had an A-output pulse count associated with it. The pulse count was converted to crank angle degrees with the knowledge that there are 500 A-output pulses per revolution, and that crank angle is defined to be zero when the engine volume is maximum, which coincides with the Z-pulse.

Since the rotary encoder was incremental, rather than absolute, it was crucial that the sampling rate was high enough so that no pulses were missed. At the maximum rated engine speed of 750 RPM, the A-output delivers 6 250 pulses/s. Voltages were generally sampled at 30 000 Hz, which gives 4.8 samples per pulse in this worst-case scenario.

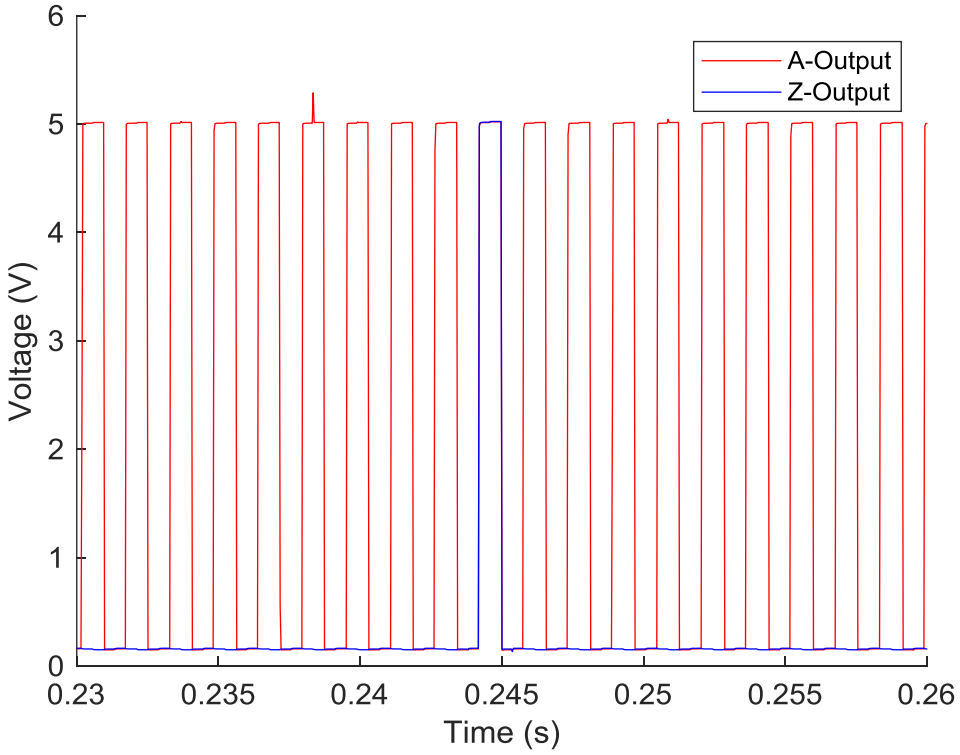


Figure 2.26: Example of Raw Data Collected from the Rotary Encoder

2.7.4 Torque Measurement System

Torque was measured using a non-contact rotary torque transducer (TRS600, FUTEK Advanced Sensor Technology, Inc.). The set-up is visible in Figure 2.2. The torque transducer was calibrated by the manufacturer. They performed a five-point calibration in both directions and included the results in the transducer shipment. A linear fit was applied to their calibration data, and the resulting equation was used to convert measured raw voltages into torques.

Figure 2.27 provides an example of raw data collected using the torque transducer, expressed as a function of crank position. The wide band of data results from plotting several engine rotations on the same axis. Periodic torque fluctuations are visible in the raw data, and their regular shape indicates a consistent load.

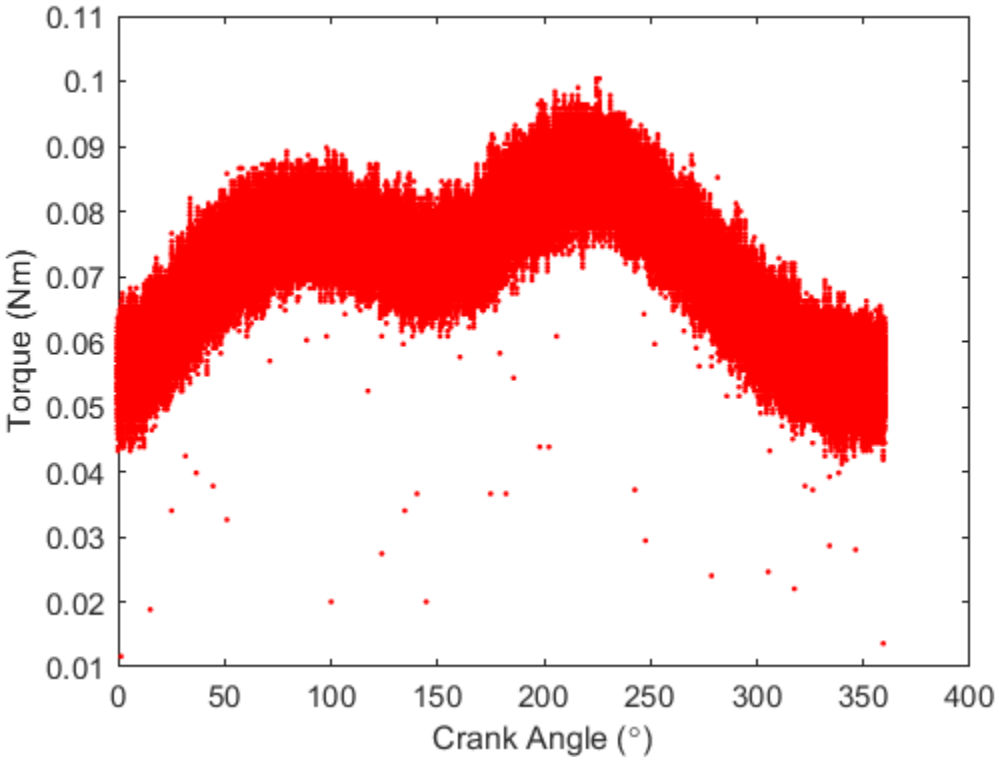


Figure 2.27: Example of Raw Data for Torque Measurement

2.7.5 Heat Input Measurement System

The heat input rate is measured by tracking the average amount of time that the heaters are on during a steady state test. This is done by logging the signal voltage emitted by the

temperature controller. It was assumed that all the electrical power consumed by the heater is either absorbed by the engine or lost to the room through the insulation.

The electrical power consumed by the heating system was calculated based on the measured current and resistance of the cartridge heaters. The power consumption will vary with temperature, since the resistance of the heaters increases when the temperature increases. The test was performed at room temperature and at 60 °C, yielding power values of 3 235 W and 3 291 W, respectively. The higher value was taken as the constant power consumption of the heating system.

For tests including heat input measurements, data was collected for 30 minutes. The average heat input rate was then calculated using the following equation.

$$\dot{Q}_{in} = \begin{pmatrix}\text{Fraction of time}\\ \text{heater is on}\end{pmatrix}\begin{pmatrix}\text{Electric power consumption}\\ \text{when heater is on}\end{pmatrix} \quad (2.1)$$

Figure 2.28 exemplifies the heater control signal raw data. A voltage above 7 V indicates that the heaters are switched on.

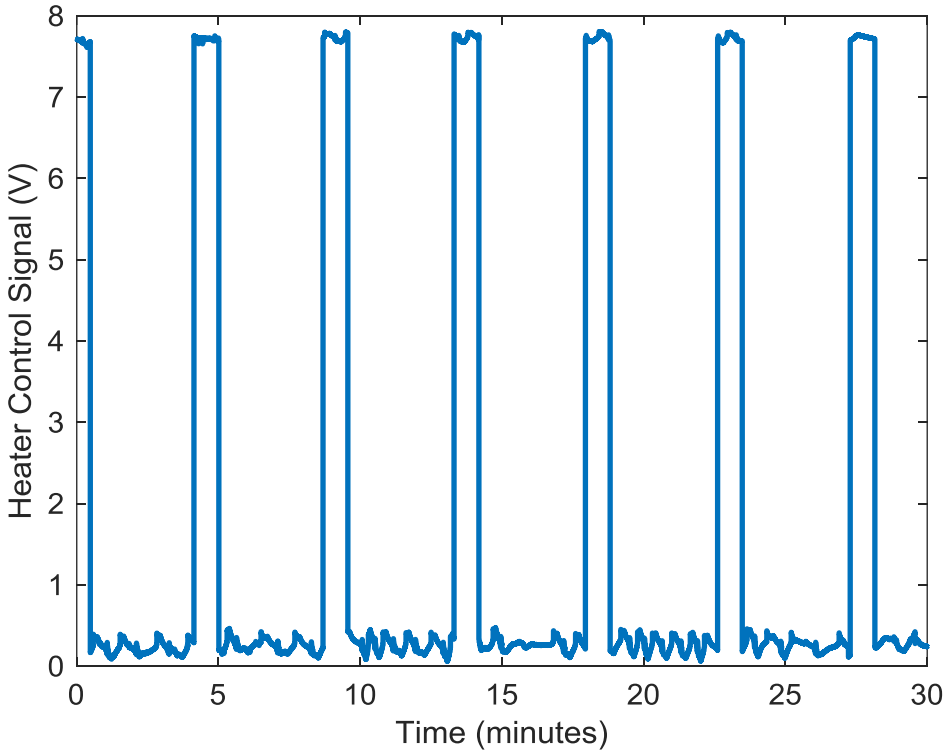


Figure 2.28: Example of Raw Data for Heater Control Signal

2.7.6 Heat Rejection Measurement System

To measure the rate of heat rejection, coolant flow rates and temperature changes were measured. The heat rejection rate was then calculated using Eq. (2.2) below.

$$\dot{Q}_{rej} = \dot{m}c\Delta T = \dot{V}\rho c\Delta T \tag{2.2}$$

The heat rejection system calibration involved calibrating the peristaltic pumps and the RTDs. The pumps were calibrated using the procedure built into the pump electronics, which involved collecting water from the pump and measuring its volume using a graduated cylinder. The RTDs were calibrated the same way as the gas temperature thermocouples. Calibration data was collected by immersing them as a group in the water bath at several temperatures. Correction terms were calculated such that the RTDs would read common temperatures at the calibration data points. By fitting third order polynomials to the calibration terms, equations were created which could be used to estimate the appropriate correction term for any desired temperature. These correction terms were applied to the raw data to decrease the uncertainty in relative temperature measurements taken between two RTDs. The effect of applying the correction terms to the raw RTD data is shown in Figure 2.29. Like the thermocouples, the correction strategy only improves the error between RTDs, and does not affect the absolute error.

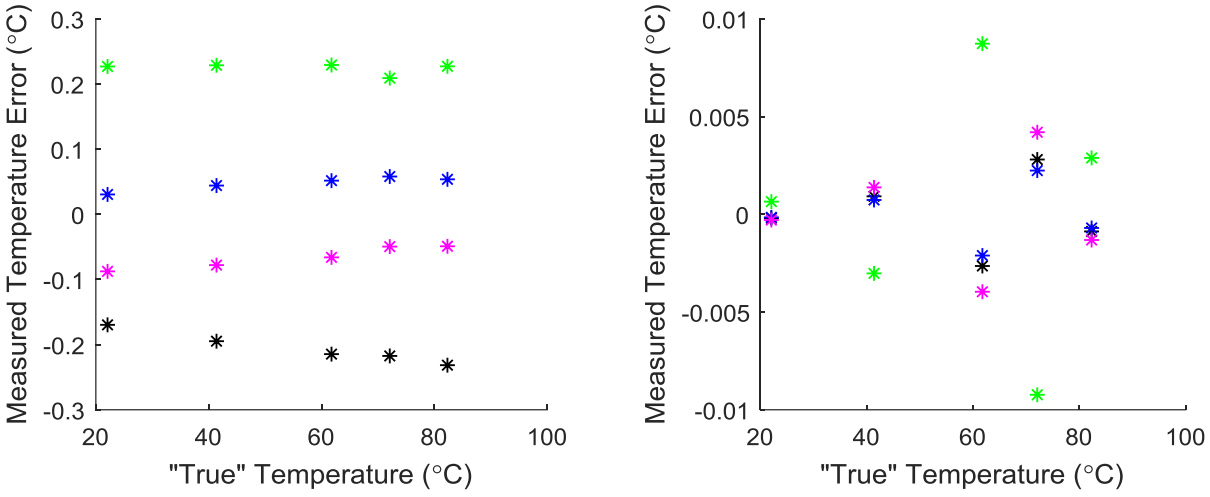


Figure 2.29: Difference between Measured Water Temperature and Overall Average Temperature (a) Before Applying Correction Terms and (b) After Applying Correction Terms

All experiments were run with water temperatures of 21 °C, which was approximately the temperature of the air in the room. This minimized the heat transferred between the outside of the engine and the surroundings during the test, which would skew the measurements.

Figure 2.30 gives measured water temperatures as a function of time for an example data set. Flow rates were adjusted to obtain a clear temperature rise across both cooling zones. Water mperatures were averaged over the data sets in the post processing routine.

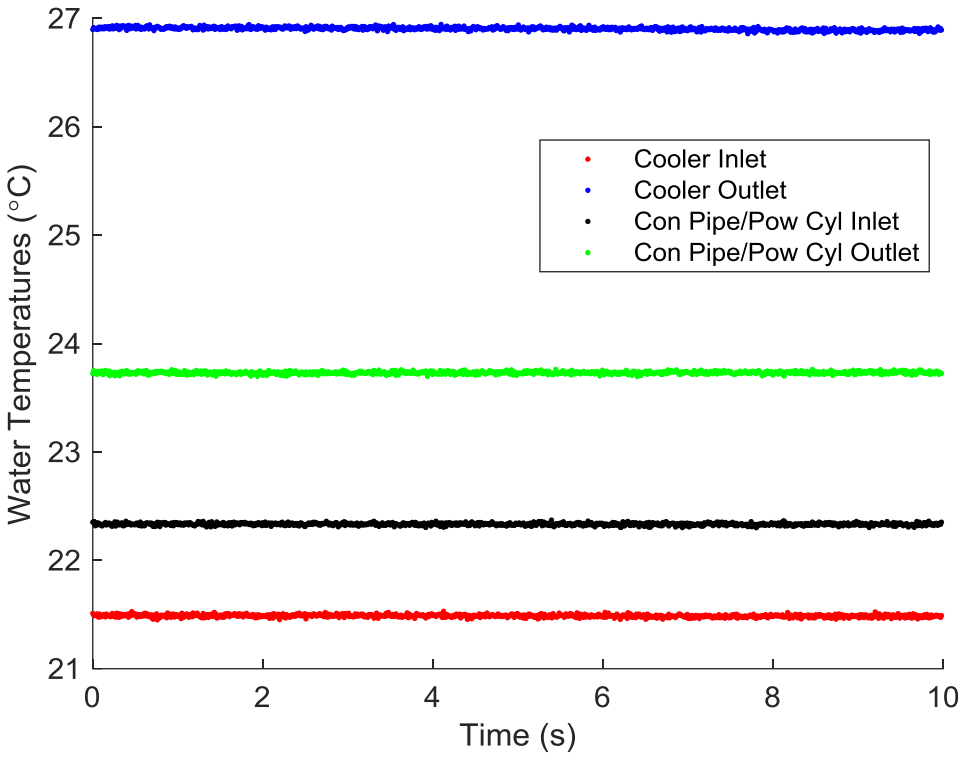


Figure 2.30: Example of Raw Water Temperatures as a Function of Time

2.7.7 Data Acquisition Software

Experimental data was recorded by an in-house DAQ software program written in LabWindows™ CVI. A screen shot of the user interface is shown in Figure 2.31. The program features live plots of temperature, voltage, speed, torque, and power, as well as a live indicator diagram. Voltage, thermocouple, and RTD data may be sampled at different rates, and measured data is stored as text files.

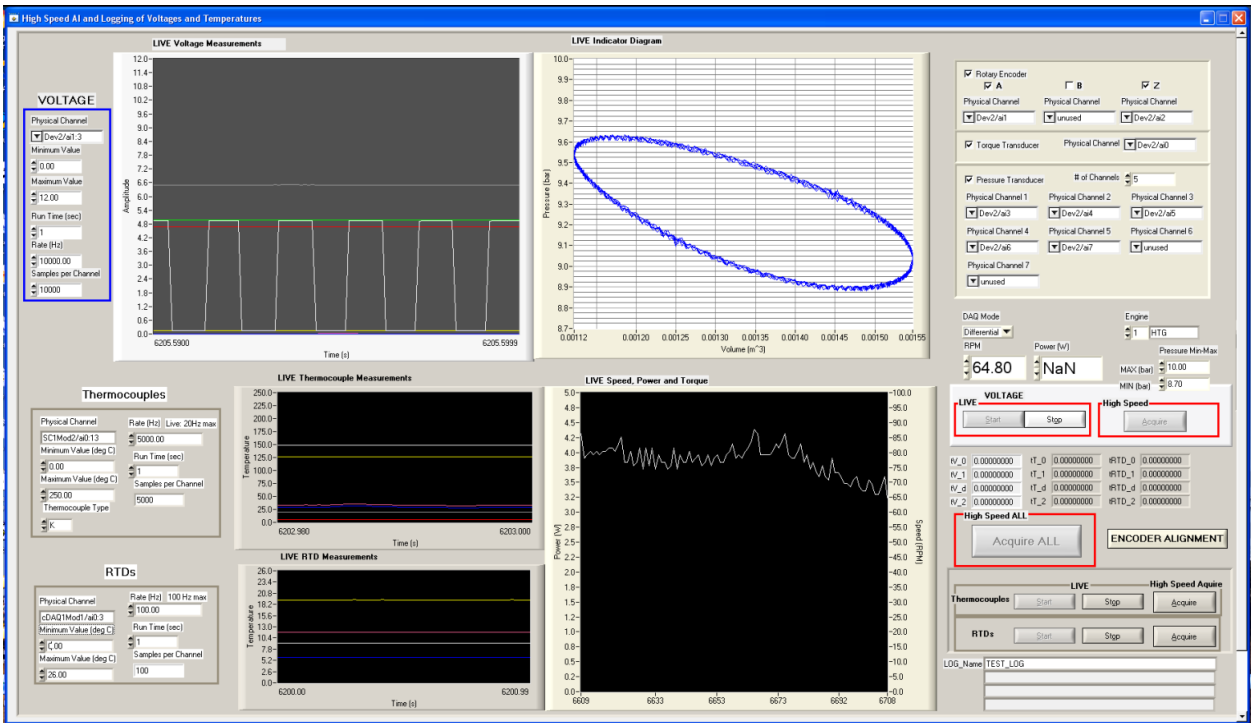


Figure 2.31: Screen Shot of Graphical User Interface for DAQ Program

2.8 Experimental Procedures

The following section lists procedures for the various tests performed with the test rig. The warm-up procedure describes how to warm-up the heating system. This is typically done with the engine stationary, to minimize wear on the seals. Procedures for general performance tests, efficiency tests, and conduction loss tests are listed next, followed by the heating system cool-down procedure. Finally, the procedure for performing a cold motoring test is described.

In general, the heating system should not be switched on until correct operation of the cooling system has been verified. Engine fill pressure changes should always be done slowly to allow the pressure inside the displacer to equilibrate. Hair should be tied back, and loose clothing should not be worn, when operating the engine.

2.8.1 Warm-Up Procedure

1 Check that work area is tidy and free of trip hazards.
2 Ensure that water bath and air compressor are plugged into separate circuits. Otherwise, they may trip the breaker.
3 Start the cooling system pump and check for leaks.
4 Start the heating system and verify correct operation at some temperature below 100 °C.
5 Heat engine in steps of less than 100 °C, to minimize thermal shock on the engine components.
6 While the engine is warming up, test the data acquisition system by collecting some sample data sets, and examining the log files.

2.8.2 Performance Testing Procedure

1 Set heater cap temperature and allow temperature to stabilize (this takes at least half an hour).
2 Use the air compressor to slowly fill the engine to the desired pressure. To avoid collapsing the displacer, all fill pressure changes should be performed at less than 2 bar per minute, according to the original ST05G-CNC drawing package from Ve-Ingenieure.
3 Start engine by hand while wearing a leather glove. The glove is worn to prevent the operator from scratching their hand on other test rig components when starting the engine. The engine should start very easily, with one light pull of the flywheel.

4 Once the engine is running, allow it to run freely until it reaches steady-state. When the engine first starts, it will rapidly consume heat stored in the heating cap. After about 10 minutes, the heating system will have cycled on and off several times, and the temperatures and engine speed should stabilize.

5 Collect data at various engine speeds. The engine speed is controlled using the braking system. Since both speed and torque are measured, it is not necessary to precisely control the engine speed. The engine should be allowed to reach a constant speed at each load, before the measurement is taken.

6 Repeat for several fill pressures, as needed.

7 Repeat for several heating cap temperatures, as needed.

2.8.3 Efficiency Measurement

1 Ensure that the water flow rate is consistent, and that the known flow rate is low enough that a measurable temperature rise exists across the engine water jackets.

2 Start the engine and apply a consistent load.

3 Allow the engine to reach steady-state.

4 Acquire a long data set, which includes several on/off cycles of the heating system.

2.8.4 Conduction Loss Measurement

1 Ensure that the water flow rate is consistent, and that the known flow rate is low enough that a measurable temperature rise exists across the engine water jackets.

2 Position the crankshaft at 0 ° (maximum working space volume).

3 Collect water temperature data with the engine stationary.

4 Repeat for several heating system temperatures.

2.8.5 Cool-Down Procedure

1 Slowly drain pressure from the engine, by loosening an instrument fitting. Depressurization is always done slowly to protect the displacer.

2 Reduce the setpoint of the water bath to 1 °C to speed up the cooling process.

3 Constantly supervise until the heater cap temperature is below 200 °C.

4 Check regularly until the heater cap temperature is below 100 °C.

5 Shut down water bath, temperature controller, computer, and all DAQ equipment.

2.8.6 Cold Motoring Test Procedure

1 Run the calibration routine for the electric motor (CPM-MCVC-3441S-RLN, Tecknic Inc.) using the Teknic ClearPath software.
2 Start the cooling system pump and check for leaks.
3 Verify correct operation of the DAQ system by taking a measurement and examining the log file.
4 Slowly fill engine to desired pressure.
5 Use motor to slowly accelerate engine to desired test speed. Allow engine temperatures to reach steady state. When driven by the motor, the engine will act as a refrigerator.
6 Collect data.
7 Repeat for several speeds, pressures, differential pressure measurements, heater head orientations, or regenerators, as needed.

2.9 Experimental Uncertainty

Sources of experimental uncertainty are discussed and summarized in this section. Appendix A contains more detailed information regarding error sources and uncertainty propagation calculations. Uncertainties in base measurements were propagated into calculated results using the procedure outlined by Wheeler and Ganji [87].

Table 2.1 outlines the estimated uncertainties for the base measurements taken using the current test rig. Uncertainties have been estimated conservatively using the manufacturer specifications. It is expected that the uncertainties will be reduced once the experimental data has been calibrated.

The most significant uncertainty is that of the torque measurements. These are of concern because the magnitude of the uncertainty is comparable to the measured result. For the torque measurement, the torque may be amplified using a gear or pulley ratio to bring it out of the noise range. This has not been done at this stage. Torque data collected to date should be viewed critically, especially in terms of absolute values.

Table 2.1: Estimated Overall Uncertainties for Base Measurements

Measurement	Instrument	Estimated Overall Uncertainty
Time	computer	0.001 s
Gas Temperatures	thermocouple (absolute measurement)	5 °C
	thermocouple (relative measurement)	0.01 °C
Water Temperatures	RTD (absolute measurement)	1 °C
	RTD (relative measurement)	0.01 °C
Gas Pressures	outboard diaphragm pressure transducers (absolute measurement)	5 kPa
	outboard diaphragm pressure transducers (differential measurement)	0.01 kPa
	flush mount piezoelectric pressure transducers	5.01 kPa
Torque	rotary torque transducer	0.05 Nm
Crank Position	incremental rotary encoder	5 °
Coolant Flow Rate	peristaltic pump	0.01 L/min
Coolant Density	N/A	5 kg/m^3
Coolant Specific Heat Capacity	N/A	5 J/kgK
Current Supplied to Heaters	multimeter	0.4 A
Resistance of Heaters	multimeter	0.1 Ω
Heating Cap Temperature	thermocouple	10 °C

Table 2.2 sums up the uncertainty propagation results for the quantities calculated here. The uncertainty in the torque measurement has led to a relatively high uncertainty in the shaft power and efficiency. The heat input rate uncertainty is also significant. It is increased by the uncertainties in the measured current and resistance of the cartridge heaters.

Table 2.2: Propagated Uncertainties for Calculated Quantities

Calculated Quantity	Dependent Variables	Propagated Uncertainty
Temperature Difference	gas temperatures, water temperatures	gas: 0.02 °C
		water: 0.02 °C
Engine Frequency	crank position, time	0.0087 rad/s (0.00138 Hz)
Shaft Power	torque, engine frequency	0.25 W
Heat Rejection Rate	coolant flow rate, coolant density, coolant specific heat capacity, water temperature difference	2.4 W
Heat Input Rate	current supplied to heaters, resistance of heaters, fraction of time heaters are on	49 W
Efficiency	heat input rate, shaft power	0.20 %

2.10 Conclusions

A test apparatus has been assembled to study the performance of a modified ST05G-CNC Stirling engine. In addition to the engine, the set-up is comprised of heating, cooling, motoring, braking, and instrumentation systems. Relative to comparable test rigs in the literature, the one presented here has the following main differences:

1. The engine has been modified to suit a solid thermal source.
2. The cooling jacket has been separated into two zones.
3. Pressure and temperature measurements are taken in the crankcase, as well as in several locations in the working space.

Endeavors to improve the test system could include:

1. Measurements of solid surface temperatures inside the engine.
2. Faster time response measurements of gas temperatures.
3. Better control of fill pressure to make tests more repeatable.
4. Better resolution in the torque measurements for low torque values.

Chapter 3. Development of a Second Order Mathematical Model

Drawing from the literature, a second order mathematical model was assembled for comparison with experimental results. Israel Urieli offers an open source MATLAB code on his website [30]. His code contains the Schmidt analysis, the ideal adiabatic reference cycle, and the simple heat exchanger model, all written for alpha-type Stirling engines. Here, the code has been rewritten and extended to include slider crank volume variations for gamma engines, interchangability of the isothermal and adiabatic reference cycles, additional loss mechanisms, and features to allow virtual parameteric studies. The following chapter details these changes and presents sample calculations to show the capabilities of the model.

3.1 Calculation of Constant Volumes

The calculation of constant volumes was performed using the CAD models. Specifically, the intersect, and mass properties tools in Solidworks® were used. The CAD representation of the engine internal volume is the light blue object in Figure 3.1. The displacer and its drive rod appear in pink. The volumes of the displacer, displacer drive rod, and regenerator material are subtracted from the total engine internal volume. The volume of the power cylinder changes depending on the position of the piston. Figure 3.1 shows the minimum power cylinder volume case.

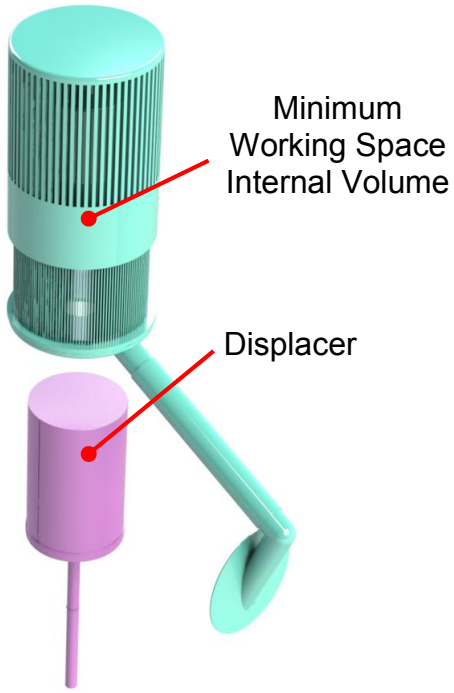


Figure 3.1: CAD Model of Engine Internal Volume and Displacer

The validity of the calculation relies on close agreement between the CAD model and the real engine, and on the accuracy of the calculation in the software. Measurements confirmed that the CAD model and real engine are a close dimensional match. Volumes calculated by the software were also measured by displacing water, and good agreement was found. This measurement was done on the dead volume reduction parts which are described in Chapter 4. The measured volume was 132 mL and the calculated volume was 131 mL.

The regenerator volume was estimated by subtracting the volume of the regenerator material from the total volume of the regenerator cavity. Information used in the regenerator volume calculation is displayed in Table 3.1 below.

Table 3.1: Information Associated with Calculation of the Regenerator Volume

Measured Mass of Regenerator Material	302.7 g
Density of 316 Stainless Steel [88]	8 g/mL
Calculated Volume of Regenerator Material	37.8 mL
Calculated Volume of Regenerator Cavity	354 mL
Calculated Regenerator Porosity	89.3 %

The volumes and surface areas of the heater and cooler were calculated in the model based on the slot dimensions and number of slots of each. These volumes were both treated as isothermal by the model. For the heater, the slot length in the model was adjusted so that the total volume calculated by the model matched the true heater volume. The resulting heater slot length was longer than the true slot length to include the volume of the disk-shaped volume at the top of the heater, and the bell-mouth entrance of the displacer cylinder. The true cooler slot geometry was input as is.

3.2 Calculation of Volume Variations

Motion of the piston and displacer leads to volume variations in the compression and expansion spaces. A mathematical description of these is required as an input to the both the ideal isothermal and the ideal adiabatic reference cycle models. The aim is to express the volumes of the compression space and expansion space as a function of crank angle, defined as zero when the working space volume is maximum. The volume of the displacer rod has been neglected in the volume variation calculations.

Cleghorn and Dechev, [89], present volume variations based on the kinematics of the slider crank mechanism. Figure 3.2 defines nomenclature for a generic slider-crank mechanism.

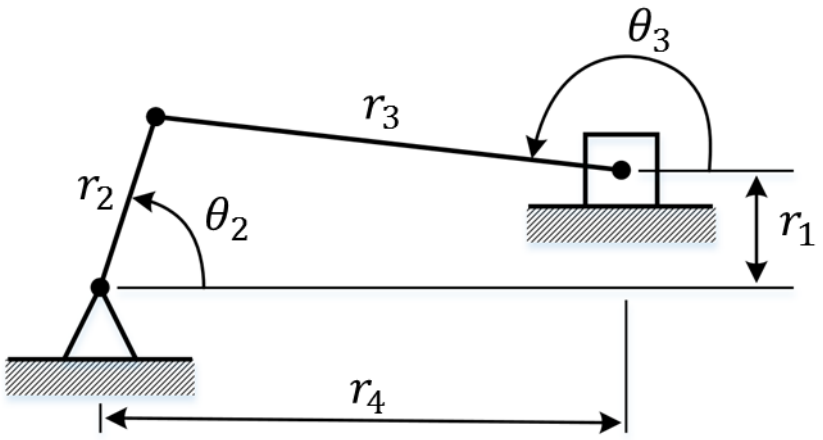


Figure 3.2: Schematic of Generic Slider-Crank Mechanism (after Cleghorn and Dechev [89])

It can be shown, [89], that for the slider-crank mechanism defined above

$$r_4 = r_2 cos\ \theta_2 - r_3 cos\ \theta_3 \tag{3.1}$$

where

$$\theta_3 = 180\ ° - sin^{-1}\left(\frac{-r_1 + r_2 sin\ \theta_2}{r_3}\right) \tag{3.2}$$

The piston and displacer are both driven by slider-crank mechanisms in the engine studied here. They share a crank, and the axes of their respective cylinders form an angle, β. Use of Eqs. (3.1) and (3.2) above requires careful consideration of this angle. The difference between θ_2,

defined above, and the crank angle, θ, measured clockwise from the point at which the workspace volume is maximum must also be understood. The real mechanism can be represented by the pair of slider-crank mechanisms shown in Figure 3.3 below. Variables with the prefix "D" are associated with the displacer and variables with the prefix "P" are associated with the piston.

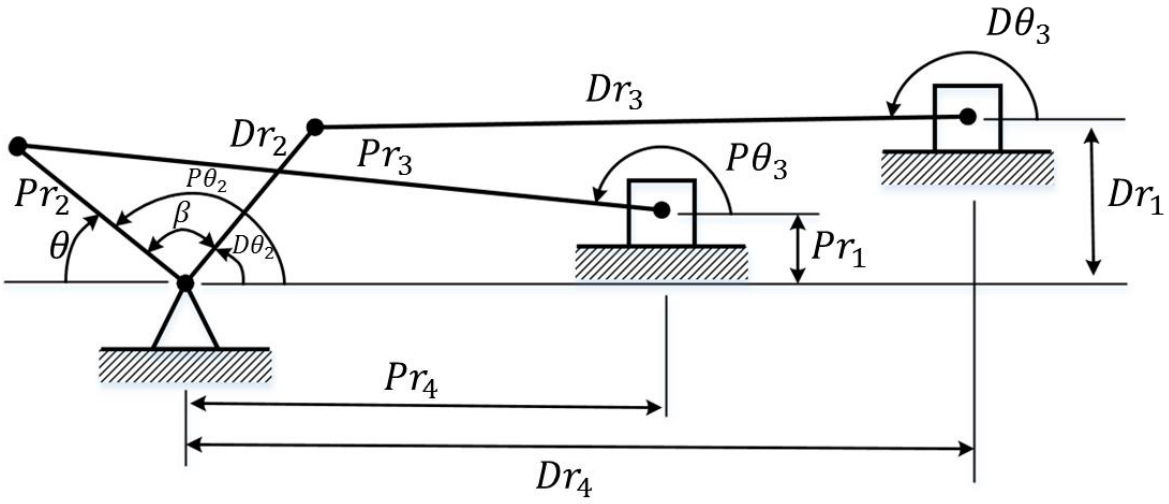


Figure 3.3: Slider-Crank Mechanism Pair Equivalent to Modified ST05G-CNC Drive Mechanism

By applying Eqs. (3.1) and (3.2) to the mechanism shown in Figure 3.3, and using geometry, the equation set shown in Table 3.2 was derived for gamma-type engines using slider-crank drive mechanisms. These equations were programmed into the thermodynamic models, and were used to convert measured crank angles into volumes when post-processing experimental data.

Table 3.2: Gamma-Type Slider-Crank Volume Variation Equation Set

$P\theta_2 = 180° - \theta$	$D\theta_2 = P\theta_2 - \beta$
$Dr_{4,max} = \sqrt{(Dr_2 + Dr_3)^2 - Dr_1{}^2}$	$Pr_{4,max} = \sqrt{(Pr_2 + Pr_3)^2 - Pr_1{}^2}$
$Dr_{4,min} = \sqrt{(Dr_3 - Dr_2)^2 - Dr_1{}^2}$	
$D\theta_3 = 180° - sin^{-1}\left(\frac{-Dr_1 + Dr_2 sin\, D\theta_2}{Dr_3}\right)$	
$P\theta_3 = 180° - sin^{-1}\left(\frac{-Pr_1 + Pr_2 sin\, P\theta_2}{Pr_3}\right)$	
$Dr_4 = Dr_2 cos\, D\theta_2 - Dr_3 cos\, D\theta_3$	
$Pr_4 = Pr_2 cos\, P\theta_2 - Pr_3 cos\, P\theta_3$	
$DV_c = \frac{\pi}{4} d_{disp}{}^2 (Dr_{4,max} - Dr_{4,min}) - V_e$	$PV_c = \frac{\pi}{4} d_{pist}{}^2 (Pr_{4,max} - Pr_4)$
$V_e = \frac{\pi}{4} d_{disp}{}^2 (Dr_{4,max} - Dr_4)$	$V_c = DV_c + PV_c$

These equations were programmed into the thermodynamic models, and were used to convert measured crank angles to volumes when post-processing experimental data.

For the ideal adiabatic model, the derivatives of the volume variations were also needed. These were evaluated, and the results are given in Table 3.3 and Table 3.4 below.

Table 3.3: Expansion Space Volume Variation Derivative Equation Set

$\frac{\partial D\theta_3}{\partial \theta} = \frac{\frac{Dr_2}{Dr_3} cos D\theta_2}{\sqrt{1 - \left(\frac{-Dr_1 + Dr_2 sin D\theta_2}{Dr_3}\right)^2}}$
$\frac{\partial Dr_4}{\partial \theta} = Dr_2 sin\, D\theta_2 + Dr_3 sin\, D\theta_3 \frac{\partial D\theta_3}{\partial \theta}$
$\frac{\partial V_e}{\partial \theta} = -\frac{\pi}{4} d_{disp}{}^2 \frac{\partial Dr_4}{\partial \theta}$

Table 3.4: Compression Space Volume Variation Derivative Equation Set

$$\frac{\partial P\theta_3}{\partial \theta} = \frac{\frac{Pr_2}{Pr_3} cos P\theta_2}{\sqrt{1 - \left(\frac{-Pr_1 + Pr_2 sin P\theta_2}{Pr_3}\right)^2}}$$
$$\frac{\partial Pr_4}{\partial \theta} = Pr_2 sin\, P\theta_2 + Pr_3 sin\, P\theta_3 \frac{\partial P\theta_3}{\partial \theta}$$
$$\frac{\partial PV_c}{\partial \theta} = -\frac{\pi}{4} {d_{pist}}^2 \frac{\partial Pr_4}{\partial \theta}$$
$$\frac{\partial DV_c}{\partial \theta} = -\frac{\partial V_e}{\partial \theta}$$
$$\frac{\partial V_c}{\partial \theta} = \frac{\partial PV_c}{\partial \theta} + \frac{\partial DV_c}{\partial \theta}$$

For buffer-pressure calculations, the volume variation equation of the buffer space was also required. This equation is given below.

$$V_b = V_{b,max} - PV_c \tag{3.3}$$

Application of the volume variation equations presented above to the engine used in this research reveals that the piston and displacer motion is nearly sinusoidal. This is shown graphically in Figure 3.4 below.

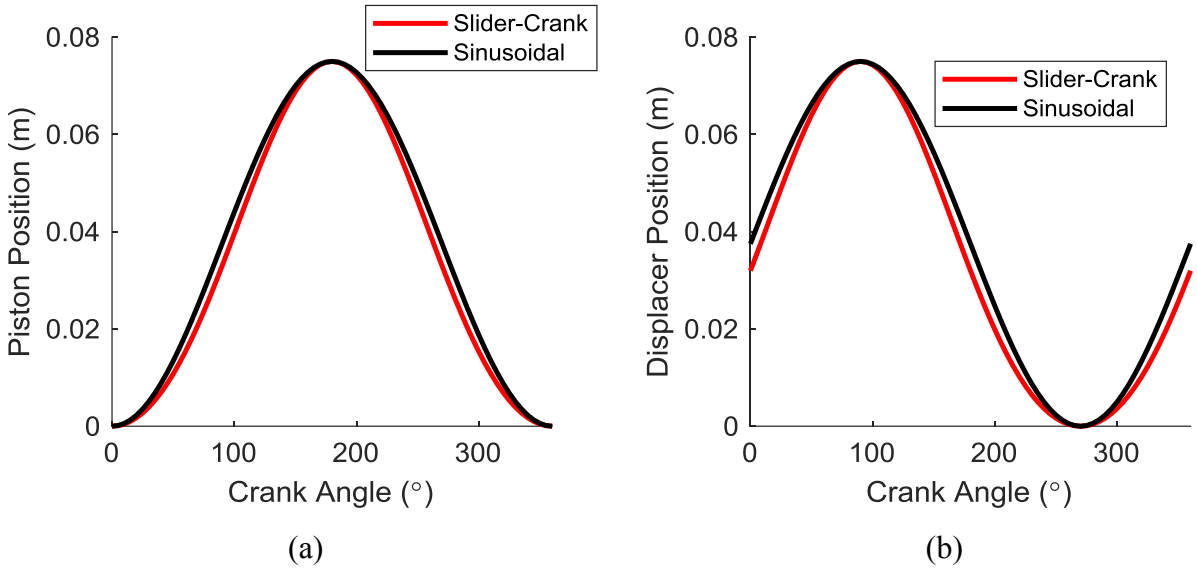


Figure 3.4: Comparison Between Sinusoidal Motion and Slider-Crank Mechanism Motion for (a) the Piston and (b) the Displacer

3.3 Calculation of Working Fluid Mass

Both the adiabatic and isothermal models take the total mass of working fluid as an input parameter. Since this is difficult to measure in the lab, it is instead calculated using the fill pressure, temperature, and engine volume. Using this approach, the calculated mass depends on the crank position, since the engine volume is crank angle dependent.

Urieli, [30], calculated the working fluid mass using the Schmidt analysis in his original code. If this mass is used as an input for a reference cycle simulation that does not have sinusoidal volume variations, the resulting mean cycle pressure will differ from that used to calculate the mass in the first place.

Paul, [50], provides an iterative method for adjusting the working fluid mass. Following Paul, the method used here begins by calculating an initial mass of working fluid using the Schmidt analysis equation and the desired mean engine pressure. Specifically, the gamma Schmidt analysis formulation given by Senft, [27], is evaluated at each crank angle degree and the average is taken to be the initial working fluid mass. This mass is then used as an input to the reference cycle model (isothermal, adiabatic, or simple), to obtain pressure as a function of crank angle. The average of this pressure is calculated and compared to the desired pressure. If the difference exceeds 0.1 %, the mass is adjusted according to Paul's formula, which reads

$$m_{new} = \left(1 + \frac{p_{mean,desired} - \bar{p}}{p_{mean,desired}}\right) m_{old} \tag{3.4}$$

where $\bar{p}$ is the average engine pressure calculated from the reference cycle simulation results. The process is iterative, and typically converges in less than five iterations.

As an example, the as-built engine was simulated using the full second order model detailed in this chapter. The working fluid mass was reduced by 2.34 % using Paul's algorithm, to obtain the correct mean pressure from the simple model reference cycle. The reduced mass led to a shaft power reduction of 14.9 %.

Thus, calculation of the mass of working fluid is an important detail of any Stirling engine model and should be given consideration when comparing model performance. Ahmadi et al., [90], offer a review paper in which they compare several second order models in terms of their

prediction of the performance of the GPU-3. The mass calculation procedure of each model was not investigated, and differences may have skewed the results.

3.4 Ideal/Reference Cycle

The model is written to allow reference cycles to be interchanged. For the ideal adiabatic model, the engine was comprised of five components as defined in Table 3.5 below. If the isothermal model was being used, the adiabatic components shown below were instead assumed to be isothermal.

Table 3.5: Definition of Isothermal and Adiabatic Engine Components

Engine Component	Assumption	Description
Compression Space	Adiabatic	Includes piston swept volume, piston clearance volume, cylinder head, connecting pipe, displacer swept volume (on cold side of displacer), and displacer cold side clearance volume.
Cooler	Isothermal	Includes the annular slotted heat exchanger only.
Regenerator	Isothermal (Linear temperature profile)	Starts at the top of the cooler fins and ends at the bottom of the heater fins. The volume of the stainless-steel wool is subtracted from the volume of the empty cavity.
Heater	Isothermal	Includes annular slots, disk-shaped volume at the top of the heater head, and the bell-mouth entrance to the expansion space. The length of the slots was adjusted in the model to obtain the correct overall volume.
Expansion Space	Adiabatic	Includes the swept volume of the displacer (hot side) and the hot side displacer clearance volume.

Results from the isothermal and adiabatic reference cycles are shown as functions of heater gas temperature in Figure 3.5. The mean pressure and cooler gas temperature, were held constant at 1000 kPa and 21 °C, respectively, to produce the plots.

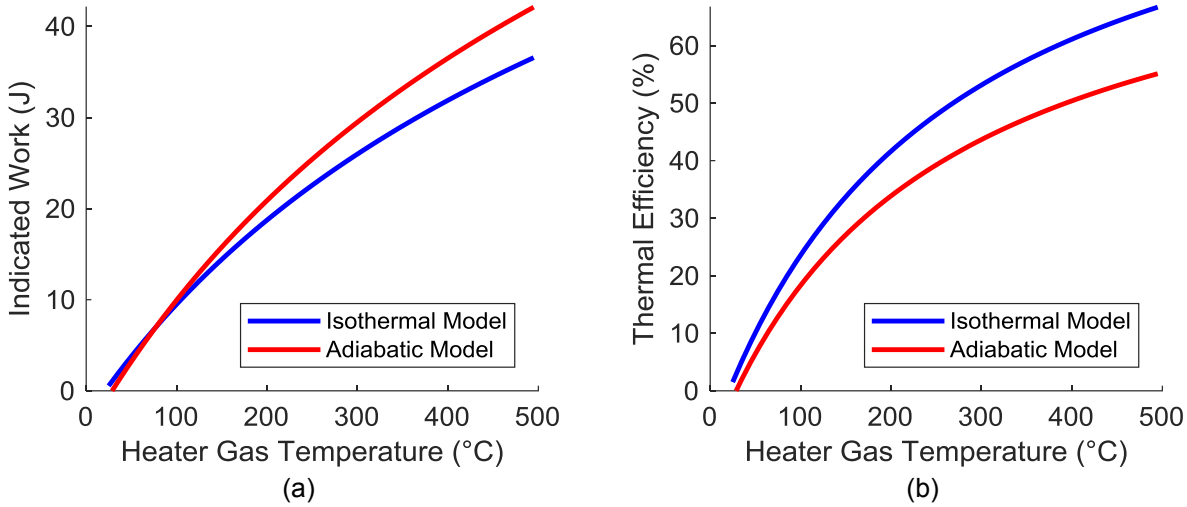


Figure 3.5: Comparison Between Ideal Isothermal Model and Ideal Adiabatic Model for (a) Indicated Work and (b) Thermal Efficiency as Functions of Heater Gas Temperature

The plots show that, at low heater gas temperatures, the two reference cycle models give similar predictions for indicated work and thermal efficiency. As the heater gas temperature increases, the ideal adiabatic model forecasts a higher indicated work and a lower thermal efficiency than the ideal isothermal model.

The temperature swings in the compression and expansion spaces of the adiabatic model lead to higher pressure swings, than in the isothermal model [10]. These pressure swings are illustrated in Figure 3.6, which gives indicator diagrams calculated by both methods, at heater gas temperatures of 500 °C and 50 °C.

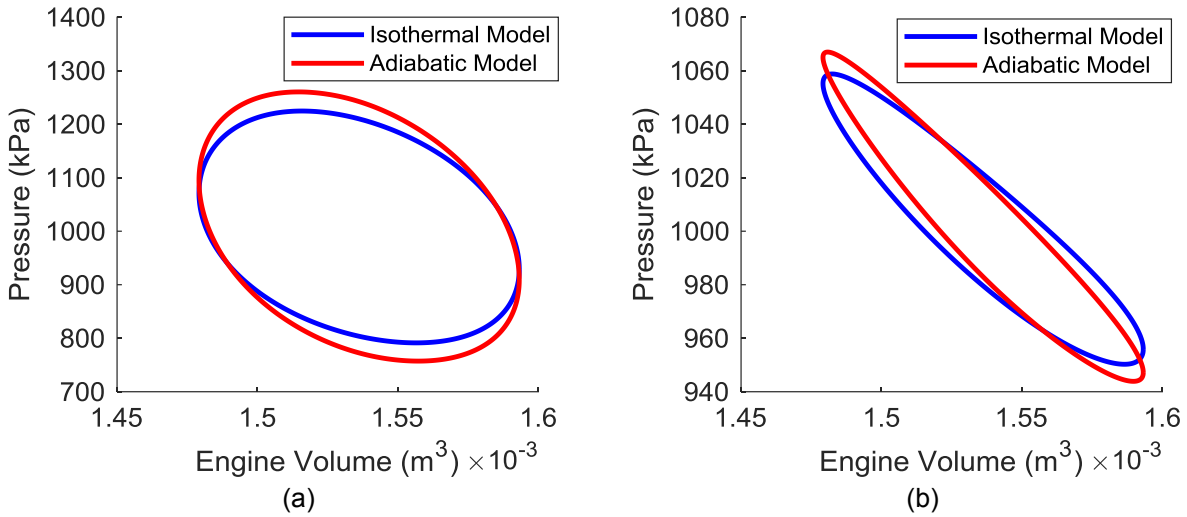


Figure 3.6: Ideal Isothermal and Adiabatic Model Indicator Diagrams for Heater Gas Temperatures of (a) 500 °C and (b) 50 °C

The lower efficiency of the ideal adiabatic model results from the mean temperatures of the expansion and compression space being below and above the heater and cooler temperatures, respectively [6], [10]. For example, gas in the adiabatic compression space, which is initially at the cooler temperature, will rise in temperature as it is compressed and make the effective temperature of the compression space higher than the cooler temperature [6].

3.5 Decoupled Loss Calculations

3.5.1 Regenerator Enthalpy Loss

The regenerator enthalpy loss was calculated following Urieli's, [30], procedure for the simple heat exchanger model exactly as described in Chapter 1. The simplified approach does not consider the properties or temperature changes of the regenerator material. It also uses steady flow heat transfer correlations.

The hydraulic diameter of the regenerator was calculated based on the porosity, ¶, and the wire diameter, d_{wire}, using [30].

$$d_r = d_{wire}\left(\frac{¶}{1-¶}\right) \tag{3.5}$$

Figure 3.7 shows regenerator enthalpy loss dependence on these two parameters as calculated using the simple heat exchanger model. According to the figure, regenerator enthalpy loss is minimized for small wire diameter, high porosity regenerator matrices. Both parameters are involved in the hydraulic diameter calculation for the regenerator matrix, Eq. (1.59), which is one reason they affect the enthalpy loss. They also affect the loss by influencing the mass flux through the regenerator, shown in absolute value brackets in Eq. (1.57). The wire diameter and porosity of the regenerator matrix used in experiments here were 0.0508 mm, and 89 %, respectively.

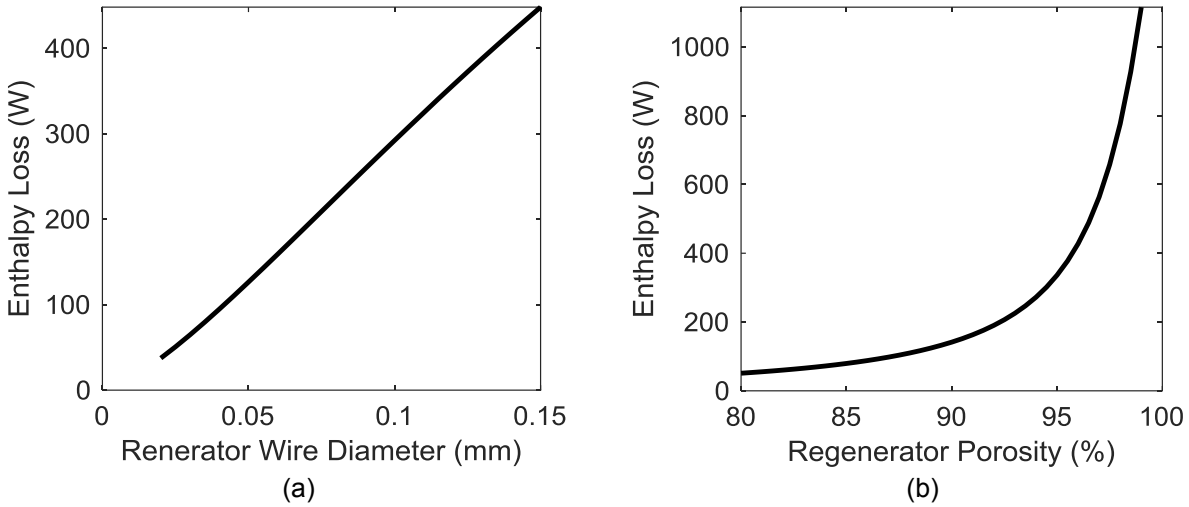


Figure 3.7: Variation in Regenerator Enthalpy Loss as a Function of (a) Regenerator Wire Diameter and (b) Regenerator Porosity

3.5.2 Conduction Loss

A one-dimensional Fourier's Law approach was used to estimate the conduction loss. A linear temperature profile in the heat flow direction and constant thermal conductivity were assumed. Radiation and convection were assumed to be negligible. A total of three conduction paths were considered. Listed in order of significance, these were

1. Conduction through the walls of the regenerator cavity.
2. Conduction through the regenerator.
3. Conduction through the displacer.

For each path, a representative thermal conductivity, cross-sectional area, length, and temperature drop was used. These values are provided in Table 3.6 below.

Table 3.6: Information Associated with Calculation of the Conduction Loss

Conduction Path	Thermal Conductivity (W/mK)	Cross-Sectional Area (m^2)	Length (m)	Temperature Difference
Walls of regenerator cavity	47 (Solidworks® data for AISI 1020 steel)	0.0036	0.055	Difference between heater and cooler wall temperatures
Regenerator	1.94 (Volume weighted combination of 316 stainless steel and working fluid)	0.0061	0.055	Difference between heater and cooler gas temperatures
Displacer	0.684 (Area weighted combination of 316 stainless steel and working fluid)	0.0069	0.151	Difference between expansion space and compression space average gas temperatures

Heat flow rates for each conduction path were obtained via

$$\dot{Q}_{cond} = kA_{cond}\Delta T \tag{3.6}$$

were summed to give the total conduction loss.

3.5.3 Appendix Gap Loss

Appendix gap losses occur in the annular gap around the displacer. According to Urieli and Berchowitz, [10], the appendix gap loss may be estimated using

$$\dot{Q}_{app} = -\pi \frac{d_{disp}}{2} k_g {S_{disp}}^2 \frac{(T_{ge} - T_{gc})}{L_{disp} b} + \pi d_{disp} \Delta p_{cycle} b S_{disp} \omega \left[\frac{\gamma}{\gamma - 1} ln\left(\frac{T_{ge}}{T_{gc}}\right) \left(\frac{1}{2} - \frac{1}{\sqrt{\frac{\omega}{2\alpha_{wall}}} h} \frac{k_g}{k_w} \right) - \frac{1}{2} \right] sin\,\varphi$$

where

b	is the appendix gap width	m
k_g	is the thermal conductivity of the working fluid	W/mK
k_w	is the thermal conductivity of the displacer and cylinder walls, taken to be 47 W/mK in this case.	W/mK
S_{disp}	is the displacer stroke	m
Δp_{cycle}	is the difference between the maximum and minimum cycle pressures	Pa
α_w	is the thermal diffusivity of the displacer and cylinder walls, taken to be 1.172e-05 m^2/s in this case	m2/s
φ	phase angle by which cycle pressure variation leads displacer motion	rad
ω	is the angular frequency of the engine	rad/s

The calculated appendix gap loss is plotted as a function of appendix gap width in Figure 3.8. An optimum value of the gap width occurs due to the competing influences of the shuttle loss component and the net enthalpy transport component. The shuttle loss component, which refers to the transport of heat down the cylinder walls enhanced by the displacer motion [10], decreases with increasing gap width. The net enthalpy transport component, which is related to the pumping of gas in and out of the gap caused by pressure changes in the engine [6], increases with increasing gap width. The plot predicts an appendix gap width of 1 mm will minimize the appendix gap loss for the engine studied here. This is the gap of the displacer currently installed.

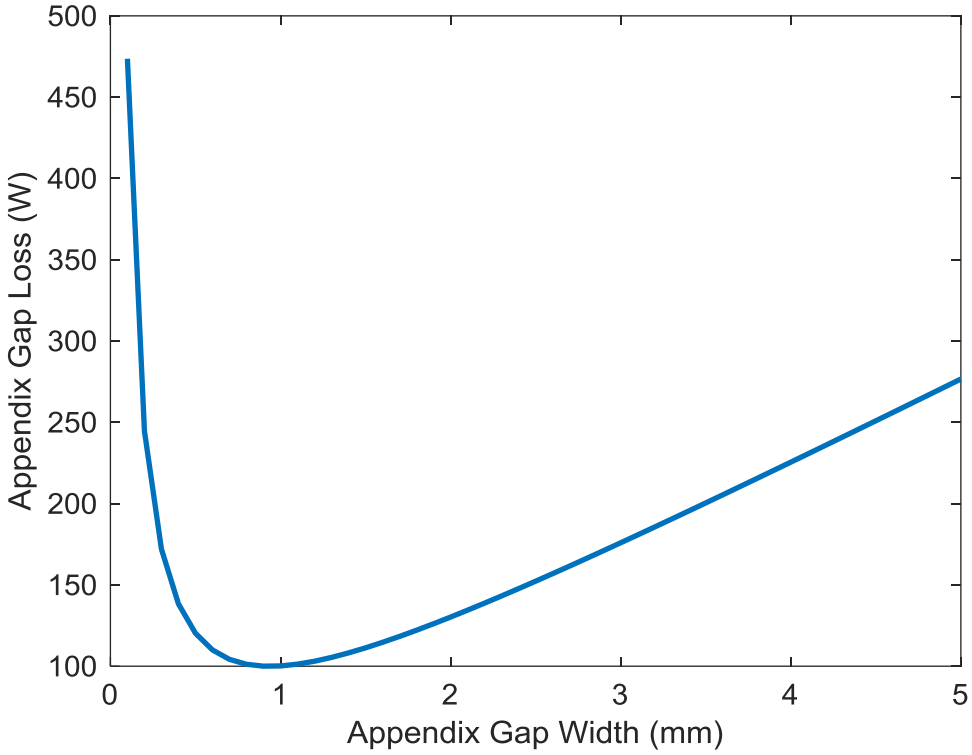


Figure 3.8: Calculated Appendix Gap Loss as a Function of Appendix Gap Width

3.5.4 Mechanical Friction and Forced Work

Senft's method, [27], of calculating mechanical friction using a constant mechanism effectiveness and the concept of forced work was implemented in the mathematical model. The mechanism effectiveness was taken to be 0.7, at the low end of Senft's suggested range for slider crank mechanisms. The pressure in the crankcase was assumed to vary adiabatically with the crankcase volume. Figure 3.9 provides an example indicator diagram with an overlaid buffer pressure curve. The areas of forced work have been highlighted in red. Areas of indicated work and forced work were numerically integrated from the results of the reference cycle calculations.

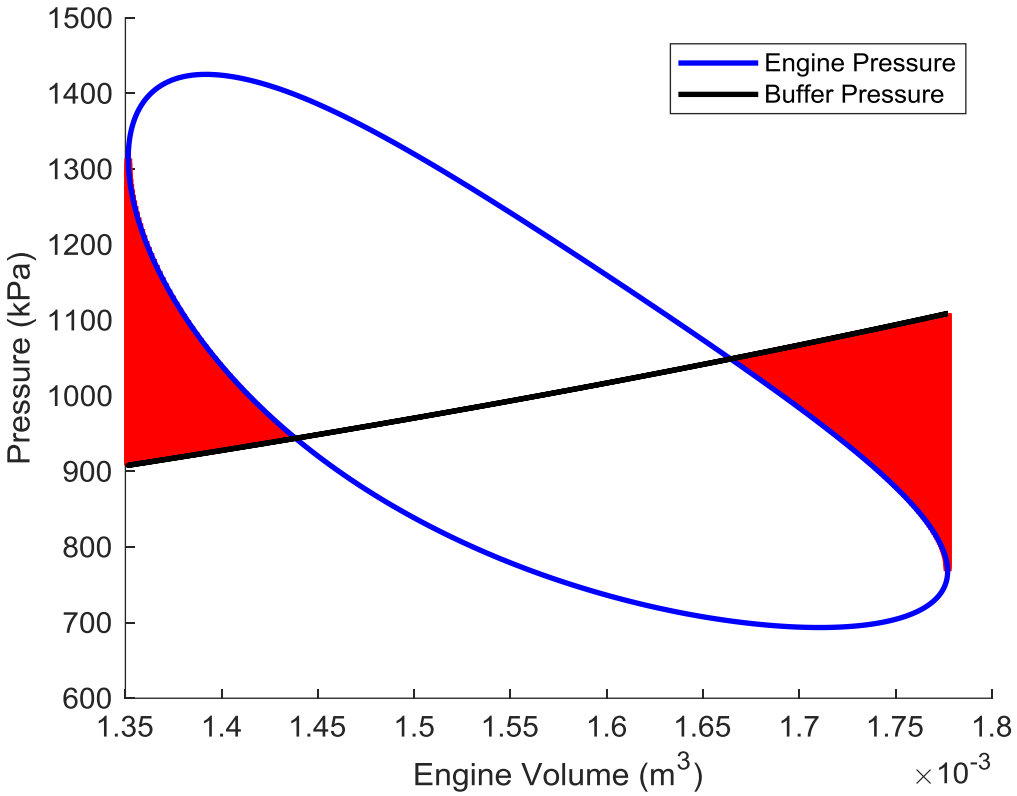


Figure 3.9: Example Forced Work Plot for Adiabatic Buffer Space

It is evident from Figure 3.9 that the amount of forced work depends on the shape of the indicator diagram and the buffer pressure curve, as well as their relative positions. The mechanical friction loss associated with forced work depends on the mechanism effectiveness.

3.5.5 Flow Friction

Flow friction in the heat exchangers and regenerator was estimated using the simple heat exchanger modeling approach, as described in Chapter 1 [30]. Entrance effects were not included. Only flow friction in the heater, cooler and regenerator were calculated. Specimen results are shown in Figure 3.10. The plots convey that the regenerator has a much greater contribution to the flow friction loss than the heater and cooler.

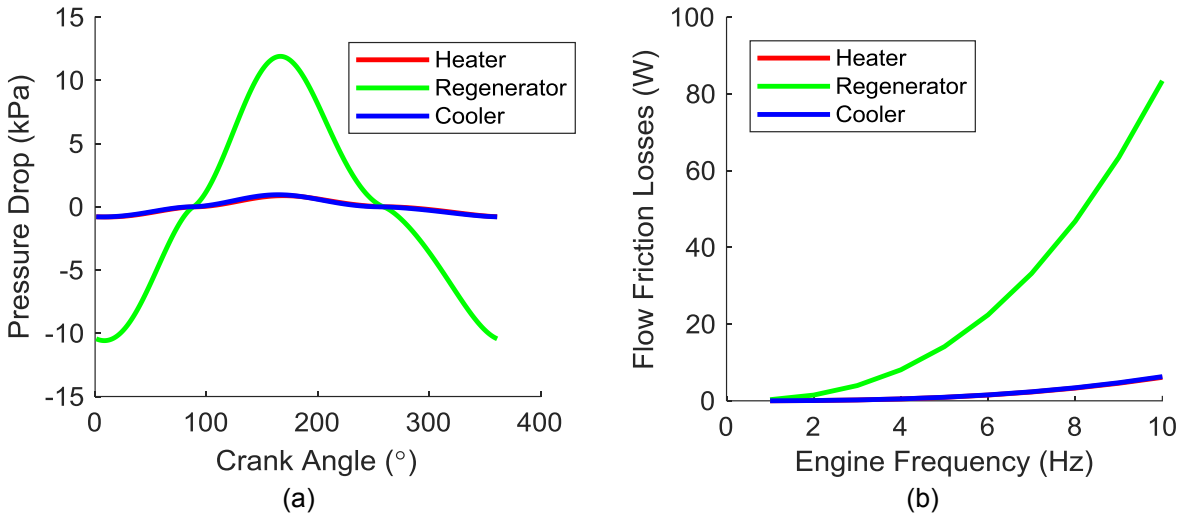


Figure 3.10: (a) Calculated Pressure Drops as a Function of Crank Angle for 10 Hz Engine Frequency (b) Flow Friction Losses as a Function of Engine Frequency

3.5.6 Losses Neglected in the Mathematical Model

Inherently, second order models overlook interactions between loss mechanisms. The calculations presented here consider the engine to be comprised of 5 homogeneous components. Effects of momentum, 3-dimensional flows, temperature and pressure dependent fluid properties, and temperature distributions across solid surfaces have not been included. Potentially computable decoupled loss mechanisms are discussed in this section.

Accurate calculation of the gas spring hysteresis loss would require knowledge of the heat transfer processes occurring in the crankcase. Urieli and Berchowitz, [10], offer a calculation for gas spring hysteresis in their book; however, application to the present problem violates several of their assumptions. The loss may be characterized empirically based on experimental data [28]. This was the approach taken here.

The effects of seal leakage have been discussed qualitatively in Chapter 1. The mathematical model developed here currently does not account for seal leakage. The most significant source of seal leakage is expected to be the displacer rod bearing. There is no dedicated seal on the displacer rod, only a plastic bushing which closely fits the rod. Maximum leakage at this location would occur at low engine speeds, when the time for leakage to occur was greatest.

Heat transfer hysteresis has been omitted due to the error involved in calculating it with the current model. Finite heat transfer rates would need to be estimated for all internal surfaces of

the engine. All wall temperatures would also need to be considered known. The loss is expected to increase with engine pressure swing amplitude.

Pressure losses due to finite speed of the pistons was not considered here. References were found which provided the equations necessary to perform the calculation, [18], [29], [49], but the derivation of these equations could not be found. The loss increases with increasing engine speed.

Energy consumed by the engine accessories would reduce the power output of a complete system. Accessories could include pumps for fuel or coolant, fans, or an alternator, depending on the application. These losses have not been considered in the modeling or experimental results presented here.

Temperature drops associated with heat transfer between the working fluid and the thermal source and sink decrease the efficiency and power output of the engine. These, like losses associated with the engine accessories would be evaluated separately from the thermodynamic model, which only considers working gas temperatures and assumed constant heat exchanger wall temperatures. Temperature drop losses have been neglected at this stage.

3.6 Addition of Losses to the Reference Cycle Results

The procedure for combining the decoupled losses with the reference cycle results is non-trivial and often differs between second order models in the literature. Losses can be added to the heat input and rejection rates, and to the power output of the reference cycle. Table 3.7 summarizes the scheme used in the current model.

Table 3.7: Scheme for Addition of Losses to Reference Cycle Results

Added to Reference Cycle …	Decoupled Loss	Notes
Heat Input Rate	Heater flow friction and half of regenerator flow friction (subtract)	Heat produced by flow friction in the heater decreases the amount of heat it must take from the thermal source
	Regenerator enthalpy loss	Increases the load on both the heater and the cooler
	Conduction loss	
	Appendix gap loss	
Power Output	Mechanical friction	Depends on amount of forced work and the mechanism effectiveness
	Flow friction	Heater, regenerator, and cooler.
Heat Rejection Rate	Mechanical friction	Heat produced by mechanical friction leaves the engine as reject heat.
	Cooler flow friction and half of regenerator flow friction	Heat produced by flow friction in the cooler is removed as reject heat
	Regenerator enthalpy loss	Increases the load on both the heater and the cooler
	Conduction loss	
	Appendix gap loss	

Note that flow friction in the heater and hot side of the regenerator decrease the power output of the cycle, but they also reduce the heat input. Regenerator enthalpy loss is calculated during the reference cycle simulation using the simple heat exchange model, so its ramifications on the power output have already been accounted for. The loss is also added to the heat input and rejection rates to reflect the additional load on the heater and cooler due to imperfect regeneration. Heat produced by mechanical friction is removed from the engine as reject heat.

Conduction and appendix gap losses are assumed only to affect the heat transfer rates; hence, they decrease the efficiency but not the power output.

3.7 Sample Calculations

Using the mathematical model, virtual parametric studies are possible. Examples of model results for changing speed, pressure, and heater wall temperature are presented in Figure 3.11, Figure 3.12, and Figure 3.13, respectively. The adiabatic reference cycle model was used with a constant cooler wall temperature of 21 °C. In the plots, losses have been normalized by dividing them by the reference cycle power output to show changes in their relative influence. Imperfect heat transfer has been isolated from the simple heat exchanger model by comparing the reference cycle power output before and after the simple model has adjusted the heater and cooler gas temperatures. At this stage, trends in the model results are the focus, rather than numerical values. Quantitatively accurate results await comparison with experimental data, and adjustment of the appropriate calculation methods.

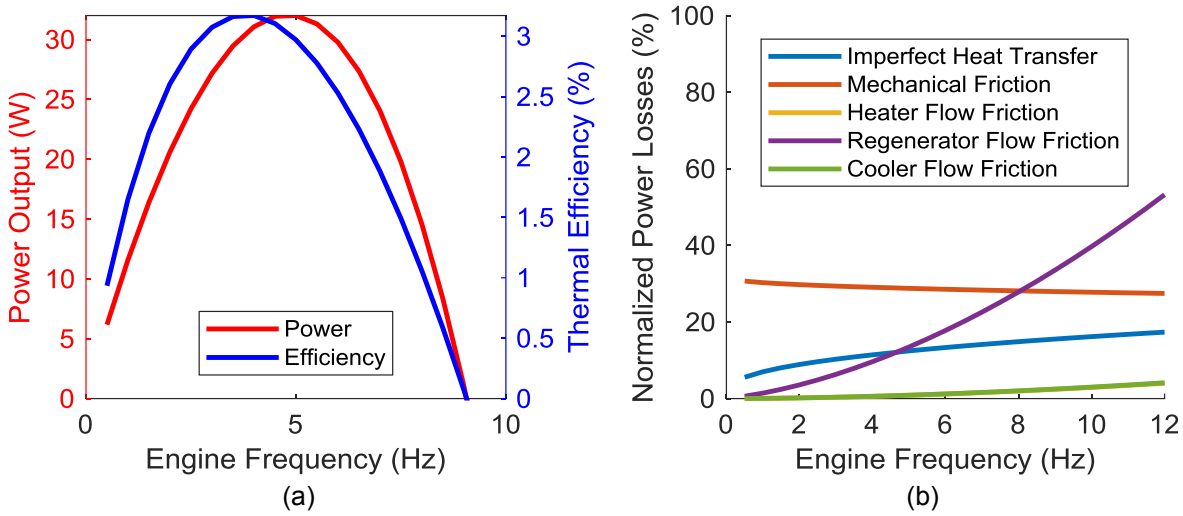


Figure 3.11: (a) Power and Efficiency Curves and (b) Normalized Power Losses as a Function of Frequency

In Figure 3.11(a), power and efficiency are observed to reach maxima, at different frequencies, before falling to zero at a common frequency. The point at which the curves cross the horizontal axis is the theoretical free running speed of the engine. Here, the power produced by the ideal reference cycle is equal to that taken away by the various power losses. Inspection of Figure 3.11(b), reveals that flow friction and imperfect heat transfer losses conspire to limit the free running speed of the engine. Mechanical friction decreases slightly due to a change in indicator diagram shape, brought about by the different gas temperatures caused by the imperfect

heat transfer loss. Regenerator flow friction is the loss which dominates at high engine speeds according to the model.

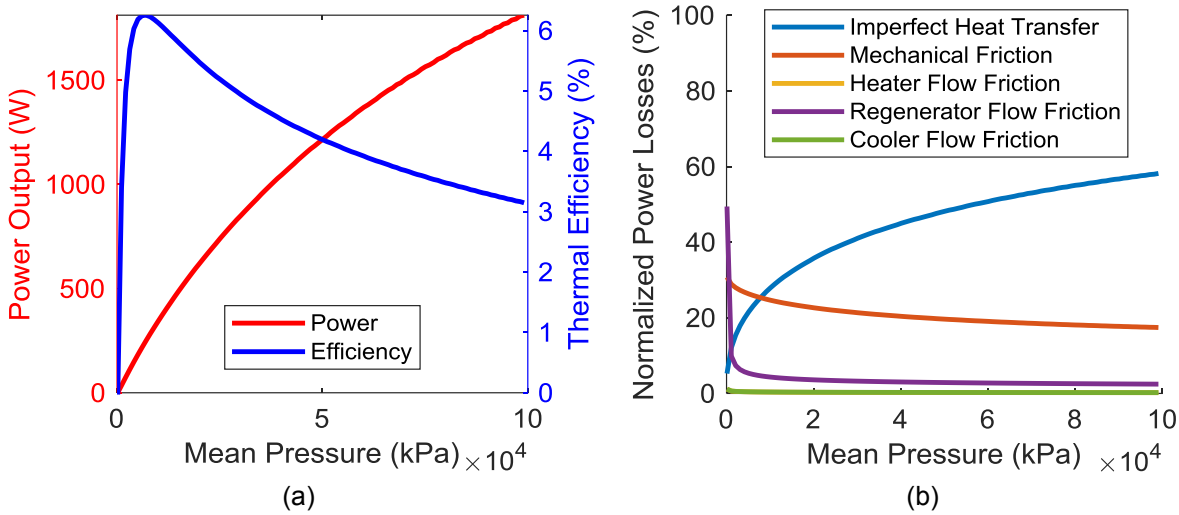


Figure 3.12: (a) Power and Efficiency Curves and (b) Normalized Power Losses as a Function of Mean Pressure

Figure 3.12 illustrates the effect of mean pressure on the power, efficiency, and power losses according to the current model. Efficiency is seen to peak, but power continues to increase as pressures climb far above what is possible in the laboratory. Inclusion of heat transfer hysteresis loss in the working space and buffer space as decoupled losses would limit the power of the engine as pressure was increased. Neglect of these loss mechanisms has led to this unrealistic trend in the power curve.

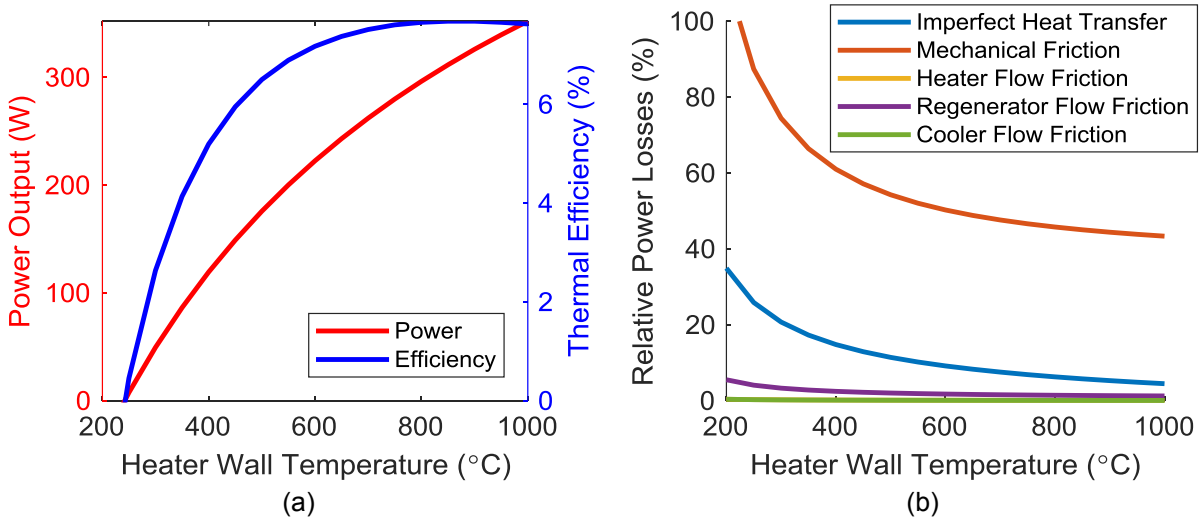


Figure 3.13: (a) Power and Efficiency Curves and (b) Normalized Power Losses as a Function of Heater Wall Temperature

Variations in engine performance as a function of heater wall temperature have been calculated and displayed in Figure 3.13. The power and efficiency curves cross the horizontal axis at the theoretical stall temperature of the engine. As heater wall temperature increases, power and efficiency follow with decreasing slope. In the case of heater wall temperature, the maximum power is limited by the material properties of the heater itself rather than by losses. As the heater wall temperature declines, all losses are seen to increase relative to the reference cycle power. Mechanical friction dominates in this case, increasing rapidly due to change in indicator diagram shape.

3.8 Conclusions

A preliminary second order model has been assembled which provides insight on engine performance trends as operating conditions vary. Advantages of the model include interchangeability of reference cycles, and isolation of the individual contributions of the decoupled losses. The model is missing several loss mechanisms, and needs to be compared with experimental data to identify areas of inaccuracy.

Chapter 4. Modifications to Reduce Minimum Thermal Source Temperature

The following chapter presents the experimental results of modifications made with the aim of reducing the minimum thermal source temperature of the engine. Three main changes were made: The piston diameter was reduced from 85 mm to 44 mm, the crankcase volume was increased from 3.20 L to 7.83 L, and the dead volume of the working space was reduced from 0.877 L to 0.745 L. The chapter begins by reporting the low thermal source temperature performance of the engine as it was originally built. The modifications are then described in detail. Finally, experimental results are presented which show the effects of the modifications on the low thermal source temperature engine performance. Throughout the chapter, and for the entirety of this thesis, the thermal sink temperature is constant at 21 °C.

4.1 Preliminary Results

The baseline low thermal source temperature performance of the engine was tested before modifications were made. With the heating system switched off, and the coolant temperature maintained at 21 °C, the engine ran until the heating cap temperature fell to 242 °C. Figure 4.1 is an indicator diagram measured near the stall point of this cool down test. The blue curve represents the measured pressure in the power cylinder, and the black curve corresponds to the measured crankcase pressure. Areas of forced work have been highlighted in red. Pressures were measured using the outboard transducers only at this stage of the experimental campaign. Pressure data from several crankshaft rotations has been averaged at each crank angle degree to produce the plot.

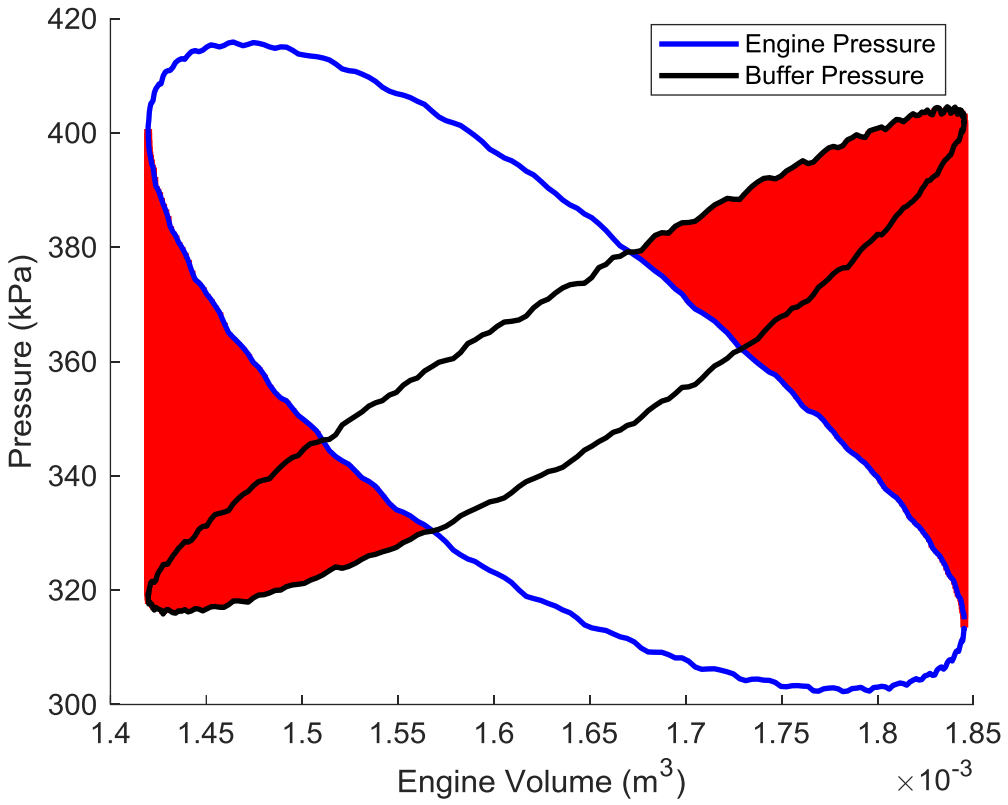
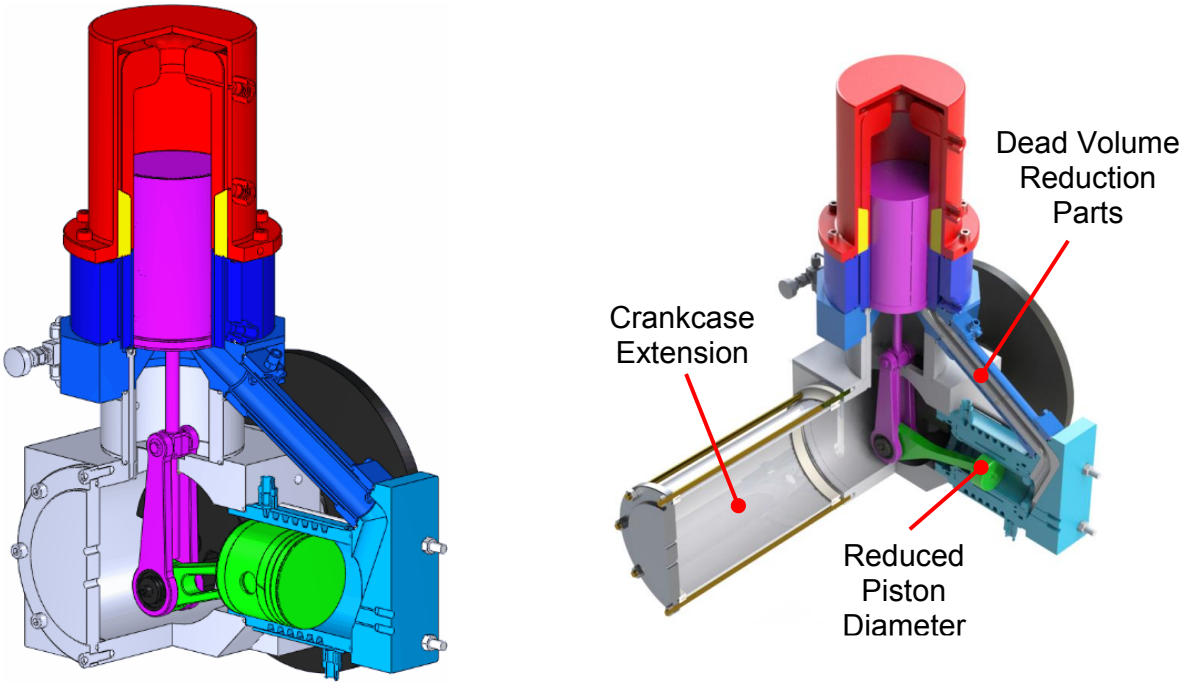


Figure 4.1: Experimental Indicator Diagram Taken Near the Cool Down Test Stall Point of the Engine As-Built

Two observations can be made regarding the buffer pressure curve in Figure 4.1. First, the pressure swings in the buffer space are seen to be of similar magnitude to those in the working

space. The resulting slope of the buffer pressure loop increases the amount of forced work relative to the constant buffer pressure case. Second, the area of the buffer pressure loop, which represents the work lost to gas spring hysteresis in the crankcase, is significant when compared to the area of the engine pressure curve. Thus, a buffer pressure curve with a slope closer to horizontal and a smaller area would lead to improved engine performance at low thermal source temperatures due to a reduction in mechanical friction and gas spring hysteresis losses.

4.2 Engine Modifications



(a) (b)

Figure 4.2: Rendered CAD Models of (a) As-Built Engine for Solar Power Application (b) Further Modifications for Low Hot Source Temperature Application

4.2.1 Modification 1: Reduced Piston Diameter

Mechanical friction and gas spring hysteresis in the crankcase have been identified as dominant losses at low source temperature operating points. Reducing the piston diameter addresses both loss mechanisms simultaneously. The smaller engine volume change results in a more circular indicator diagram with less forced work, and hence less mechanical friction. Crankcase volume changes are also reduced which decreases both the slope and the area of the buffer pressure loop on the indicator diagram, decreasing the amount of forced work and the gas spring hysteresis loss.

To reduce the piston diameter, the original aluminum piston was replaced with a graphite/glass piston-cylinder set (2KS444-3.0CP, Airpot® Corp). The piston-cylinder set was held in place by 3D printed components as shown in Figure 4.3. A new 3D-printed connecting rod was added to complete the system. The original aluminum cylinder was retained to preserve the original pressurization capability.

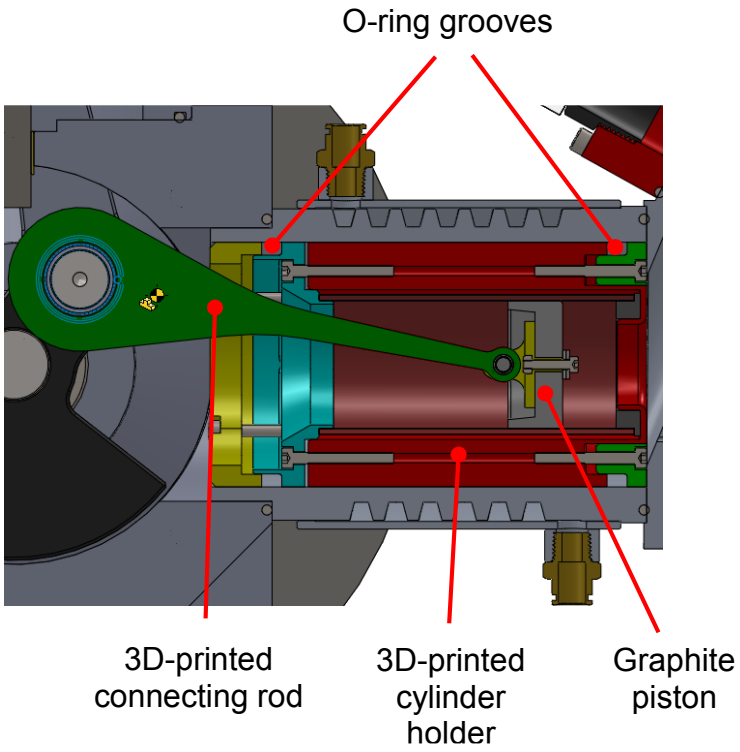


Figure 4.3: Annotated Section View of Glass Cylinder Holder

Several avenues for calculating the optimum piston diameter were explored. The results are summarized in Table 4.1 below. The first was using an empirical formula [68],

$$V_{swp} = \frac{V_{min}\Delta T}{1100} \tag{4.1}$$

presented by Ivo Kolin. The second method matched the pressure swing amplitudes induced by the displacer to those induced by the piston, while assuming ideal gas behavior and discontinuous piston and displacer motion. Further description of this method is included in Appendix B. Lastly, the second order model described in Chapter 3 was run, using the ideal adiabatic reference cycle, for a range of piston sizes to determine the size which maximized power and efficiency at a typical engine frequency of 2.5 Hz.

Table 4.1: Optimum Piston Diameters for Thermal Source Temperature of 95 °C and Thermal Sink Temperature of 5 °C

Method	Ivo Kolin's Empirical Formula [68]	Equal Pressure Swing Amplitude from Piston and Displacer	Second Order Model Maximum Power and Efficiency at 2.5 Hz
Optimum Piston Diameter (mm)	12 mm	47 mm	34 mm

The optimum piston diameter varies widely depending on the method used to calculate it. A 44 mm piston diameter was ultimately chosen for the prototype. This allowed use of a standard size piston-cylinder set, and left clearance for the 3D-printed connecting rod to pass through.

The engine was simulated using the second order model, for each piston size, at a heater gas temperature of 200 °C. The resulting forced work plots are displayed in Figure 4.4, while the numerical results appear in Table 4.2. While reducing the piston size has decreased the indicated work of the cycle, it has reduced the forced work by a greater margin, leading to a positive shaft work at a heater temperature where the as built engine theoretically will not run. The calculation illustrates the danger of optimizing indicated work without considering the influence of the cycle shape on mechanical friction losses.

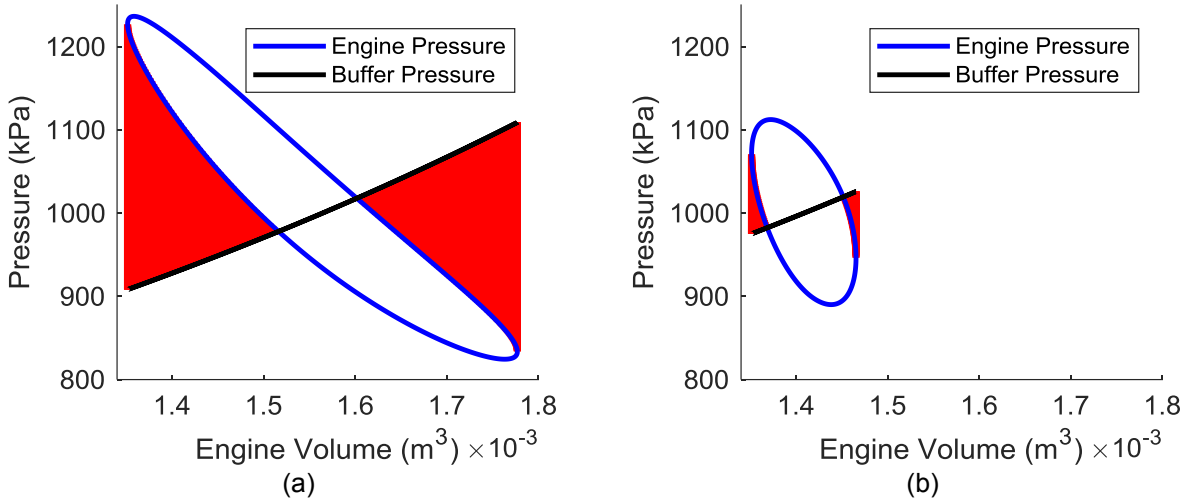


Figure 4.4: Calculated Forced Work Plots for (a) 85 mm Piston and (b) 44 mm Piston at 200 °C Heater Gas Temperature, 5 Hz Operating Frequency, and 10 bar Mean Pressure

Table 4.2: Numerical Results of Second Order Model Simulation for Reduced Piston Diameter

Piston Diameter (mm)	85	44
Indicated Work (J)	40.4	16.5
Forced Work (J)	45.2	0.995
Shaft Work Assuming a Mechanism Effectiveness of 0.7 (J)	-4.67	10.8

4.2.2 Modification 2: Crankcase Extension

Reducing crankcase pressure swings has been shown to reduce the amount of forced work and the amount of gas spring hysteresis. These pressure swings may be reduced further by increasing the crankcase volume. To accomplish this, the crankcase was extended as shown in Figure 4.5. The extension was comprised of a 6 inch O.D. mild steel pipe fitted with 3D-printed end pieces and fastened to the crankcase using the original crankcase cover, and eight threaded brass rods. The 3D-printed end pieces and the acrylic flange were fastened to the steel pipe using epoxy (Part #83000, Loctite®). The acrylic flange had an O-ring groove milled into the side facing the original crankcase. Prior to use, the extension was tested by filling it with water and pressurizing it to 10 bar for several hours.

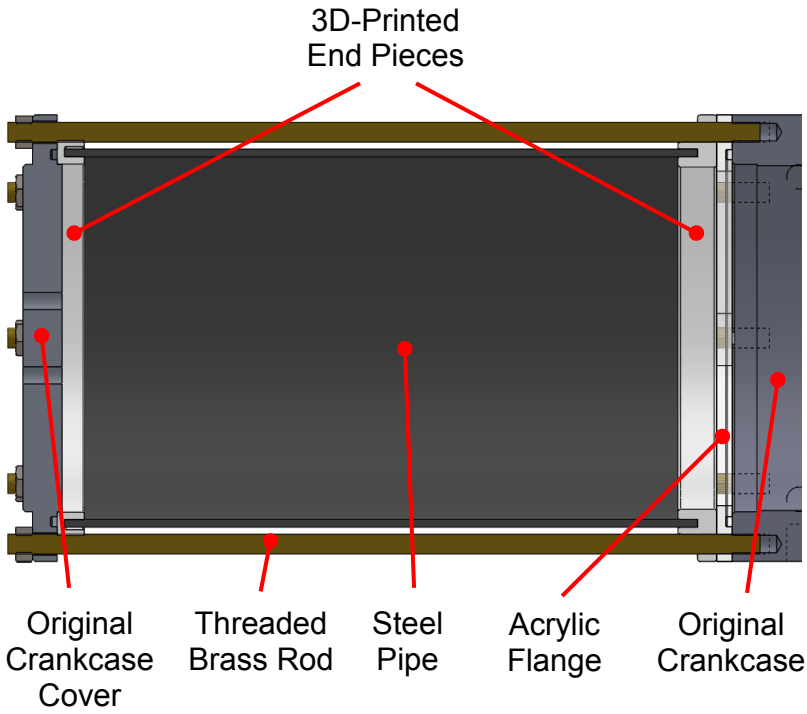


Figure 4.5: Sectioned CAD Model of Crankcase Volume Extension

4.2.3 Modification 3: Dead Volume Reduction Parts

Alfarawi et al., [63], showed, using their CFD results, that reducing the connecting pipe diameter could lead to an increase in power output. Here, dead volume reduction components were manufactured via 3D-printing and installed inside the connecting pipe, cylinder head, and displacer mount. A sectioned solid model of the parts is provided in Figure 4.6. The parts reduce the dead volume of the engine by 15 %.

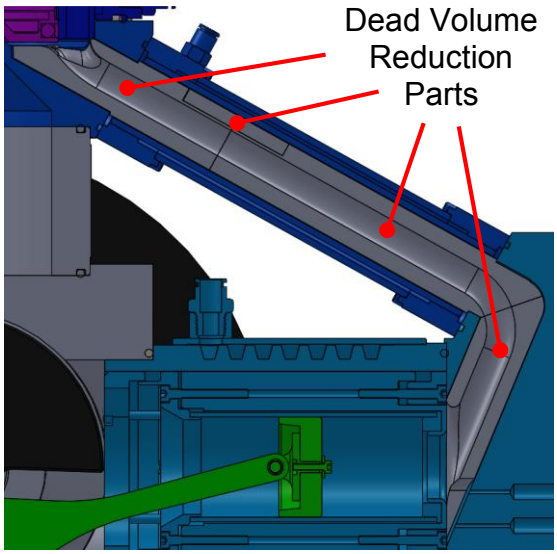


Figure 4.6: Sectioned Solid Model of Dead Volume Reduction Parts

The addition of the dead volume reduction parts has at least three potential effects on the engine. The reduction in dead volume increases the ability of both the piston and the displacer to change the pressure of the engine. Higher pressure swings should increase the indicated work, but may also influence mechanical friction, seal leakage, and heat transfer hysteresis losses. The decrease in connecting pipe diameter is also expected to increase the amount of flow friction. Finally, the plastic parts inhibit heat transfer from the working fluid to the water jacket that surrounds the connecting pipe. These conflicting effects make it unclear whether the dead volume reduction parts will lead to an increase, or a decrease, in low thermal source temperature performance.

The second order model was run with and without the dead volume reduction modification, recognizing that the effects would not be fully captured. The results are listed in Table 4.3 below. The model predicts an increase in both indicated work and forced work after installation of the dead volume reduction components. The amplitudes of the pressure and temperature swings in the adiabatic components have also been heightened. These effects are expected to change the amount of heat transfer hysteresis in the working space, which is not considered in this preliminary model. In the bottom row of the table, the dead volume reduction modification is observed to reduce the effective temperature difference between the expansion and compression spaces – a detrimental effect.

Table 4.3: Modeling Results for Addition of Dead Volume Reduction Components

	Modifications 1 and 2 only	Modifications 1, 2, and 3
Indicated Work (J)	16.5	18.5
Forced Work (J)	0.631	0.717
Pressure Swing Amplitude (kPa)	222	252
Temperature Swing Amplitude in Expansion Space (°C)	29.9	34.0
Effective Mean Temperature Difference Between Expansion and Compression Space (°C)	153	152

4.3 Experimental Results – Effect of Modifications

Experiments were performed to study the effects of the three modifications. These included power measurements, and cool down tests. For cool down tests, the heating cap was switched off with the engine running, and data was gathered until the engine stalled, with the intent of determining the minimum thermal source temperature.

The reduction in piston diameter allowed the engine to run at lower temperatures and pressures than it could with the original piston. Ideally, the engine power output would have been measured at the same operating point with both piston sizes. The limited strength of the 3D-printed components made this difficult to achieve. In fact, the small plastic part on the backside of the piston failed several times during attempts to reach the operating temperatures and pressures at which the as-built configuration ran well. It is expected that the larger piston will outperform the smaller one at high thermal source temperatures, but that the smaller one will produce more power at low thermal source temperatures. The cool down test results showed a clear difference between the two piston sizes. The 85 mm piston configuration stalled at a thermal source temperature of 242 °C, while the 44 mm piston version stalled at a thermal source temperature of 185 °C.

The remaining two modifications, crankcase extension and dead volume reduction, were tested with the 44 mm piston installed. Preliminary tests were executed at three fill pressures below 500 kPa to reduce the risk of breaking the 3D-printed components. The thermal source temperature was set to 300 °C. Power tests were repeated for three engine configurations:

1. 44 mm piston.
2. 44 mm piston, and crankcase extension.
3. 44 mm piston, crankcase extension, and dead volume reduction components.

The experimental indicated power, shaft power, and torque as a function of speed and pressure for the three configurations are plotted in Figure 4.7, Figure 4.9, and Figure 4.8, respectively.

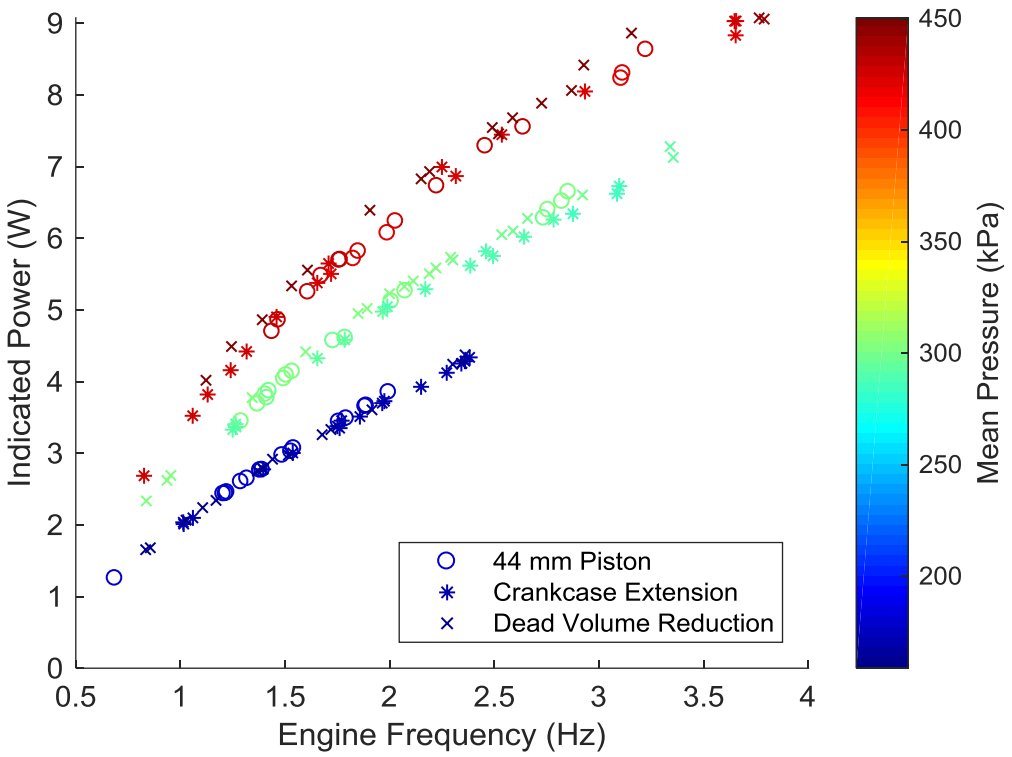


Figure 4.7: Experimental Indicated Power as a Function of Frequency for Three Modifications

The indicated power is calculated as the area of the indicator diagram multiplied by the engine frequency. As shown in Figure 4.7, the difference in indicated power was negligible between the three configurations. The addition of the crankcase extension was not expected to affect the indicated power, but the dead volume reduction parts were. The indicated power is observed to increase with frequency and mean pressure. The plot also shows that the engine was able to reach higher speeds at higher mean pressures. For the data corresponding to the highest mean pressure, differences in color between the data points indicate that mean pressure varied somewhat between tests.

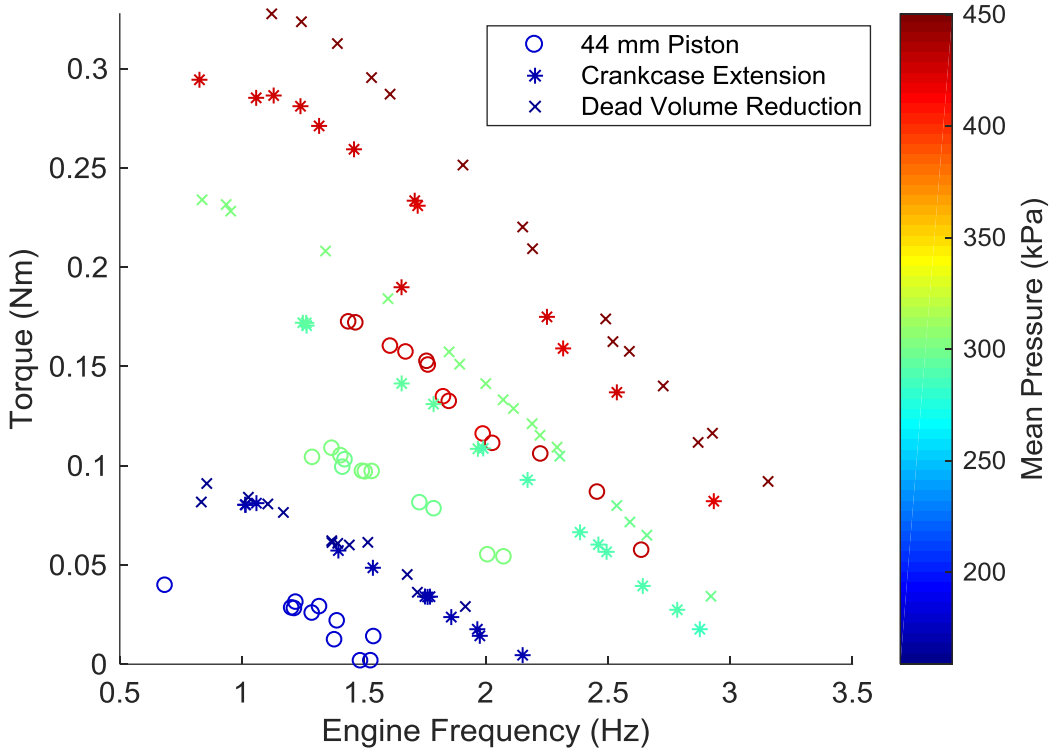


Figure 4.8: Experimental Torque as a Function of Frequency for Three Modifications

Figure 4.8 plots measured torque against engine frequency and mean pressure. Torque has clearly been improved by the addition of the crankcase extension, but the consequence of the dead volume reduction is unclear pending further test results.

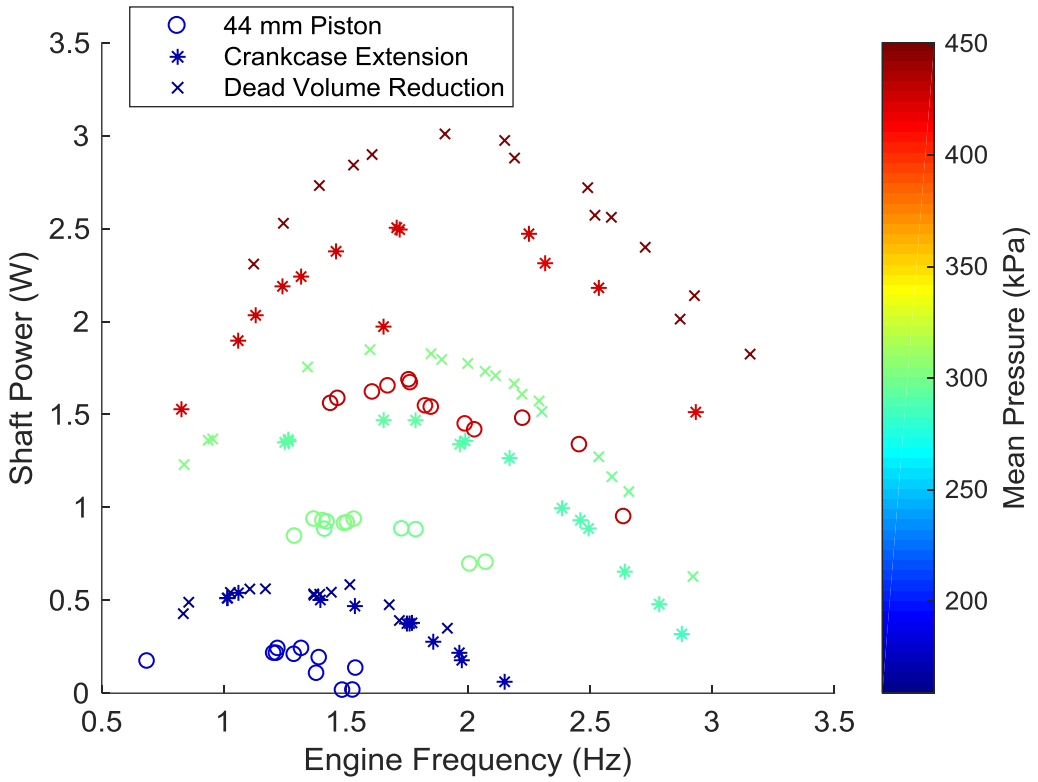


Figure 4.9: Experimental Shaft Power as a Function of Frequency for Three Modifications

In Figure 4.9, measured shaft power is plotted as a function of engine frequency and mean pressure. The shaft power is calculated by multiplying the measured torque by the measured angular frequency in radians per second. Maximum power is observed to occur at a moderate frequency which changes with mean pressure. The crankcase extension increased shaft power relative to the 44 mm piston configuration; however, the influence of the dead volume reduction parts is unclear at this stage due to inconsistencies in mean pressure between tests. For the low mean pressure case, the shaft power is unaffected, while for the higher mean pressure cases, the dead volume reduction parts appear to improve the shaft power. The marker colours indicate inconsistency in the mean pressures for the two higher pressure tests.

The benefit of the crankcase extension is further illustrated in Figure 4.10, below. The large loops correspond to the engine pressure measured in the power cylinder, while the small loops represent the crankcase pressure. As expected, adding the crankcase extension left the engine pressure diagram unchanged. The crankcase pressure swing, and the area of the crankcase pressure loop, were both reduced by extending the crankcase volume. For the example in Figure

4.10, the addition of the crankcase extension reduced the forced work by 39.1 % and the gas spring hysteresis loss by 65.8 %. The figure also reveals that these effects could be reduced further if the crankcase were made even larger.

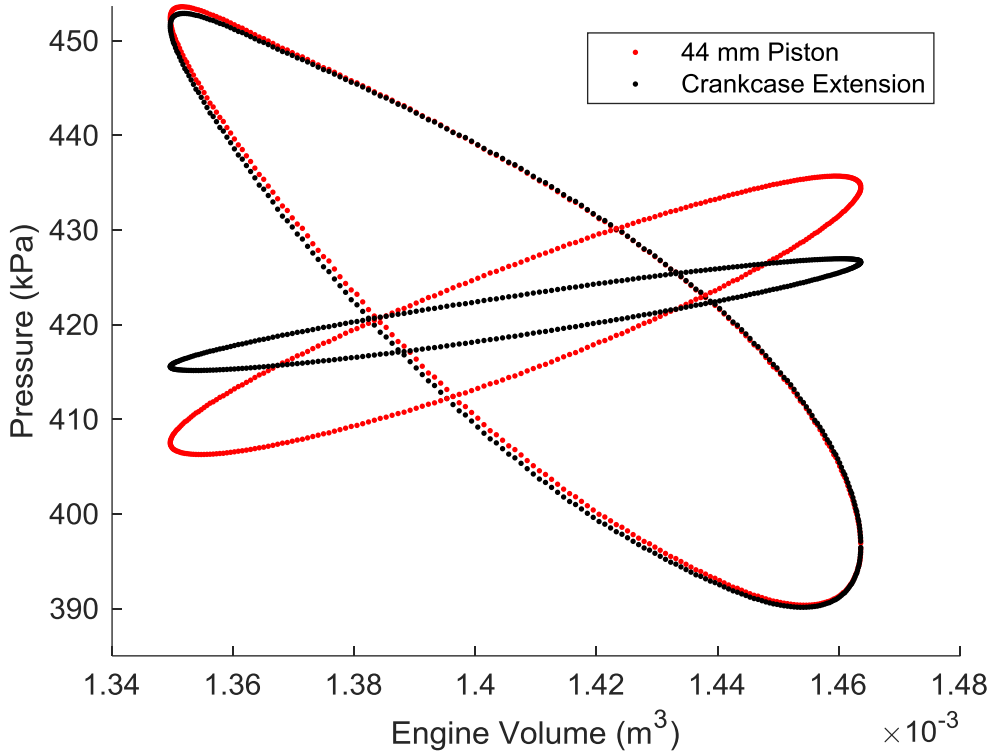


Figure 4.10: Measured Indicator Diagrams with and without the Crankcase Extension Modification

Figure 4.11 and Figure 4.12 plot measure crankcase gas spring hysteresis loss and forced work, respectively, as functions of engine frequency and mean pressure. The plots confirm that the extension of the crankcase volume reduced forced work and gas spring hysteresis for all conditions tested.

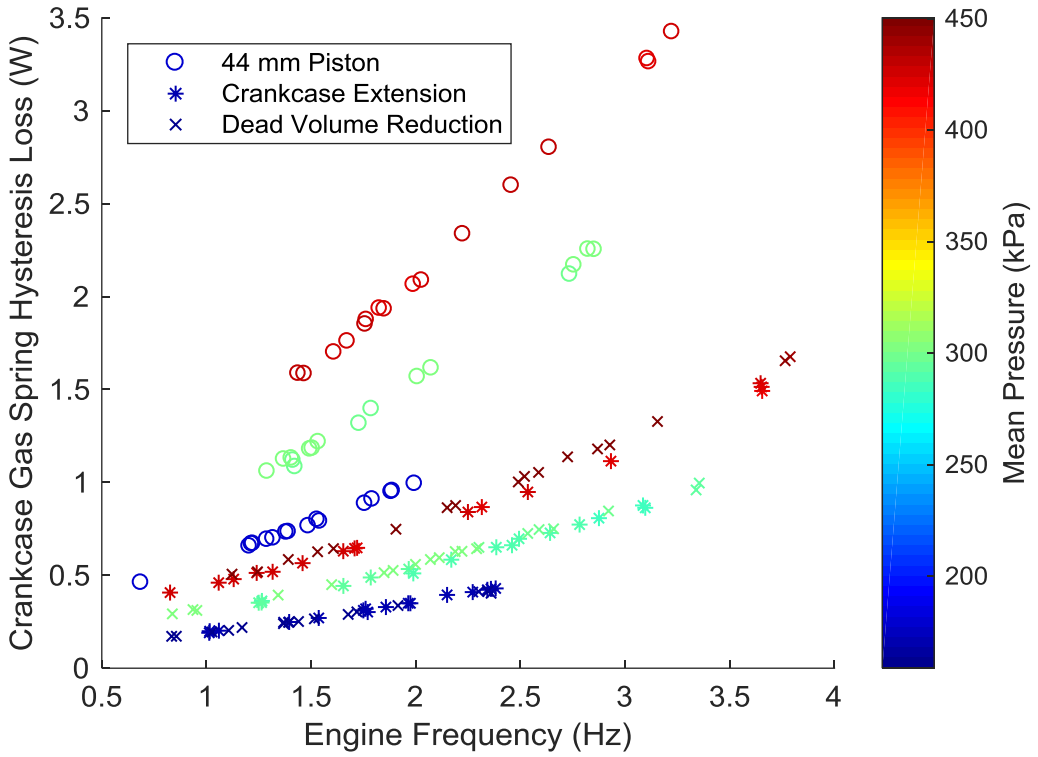


Figure 4.11: Crankcase Gas Spring Hysteresis Loss for the Three Modifications

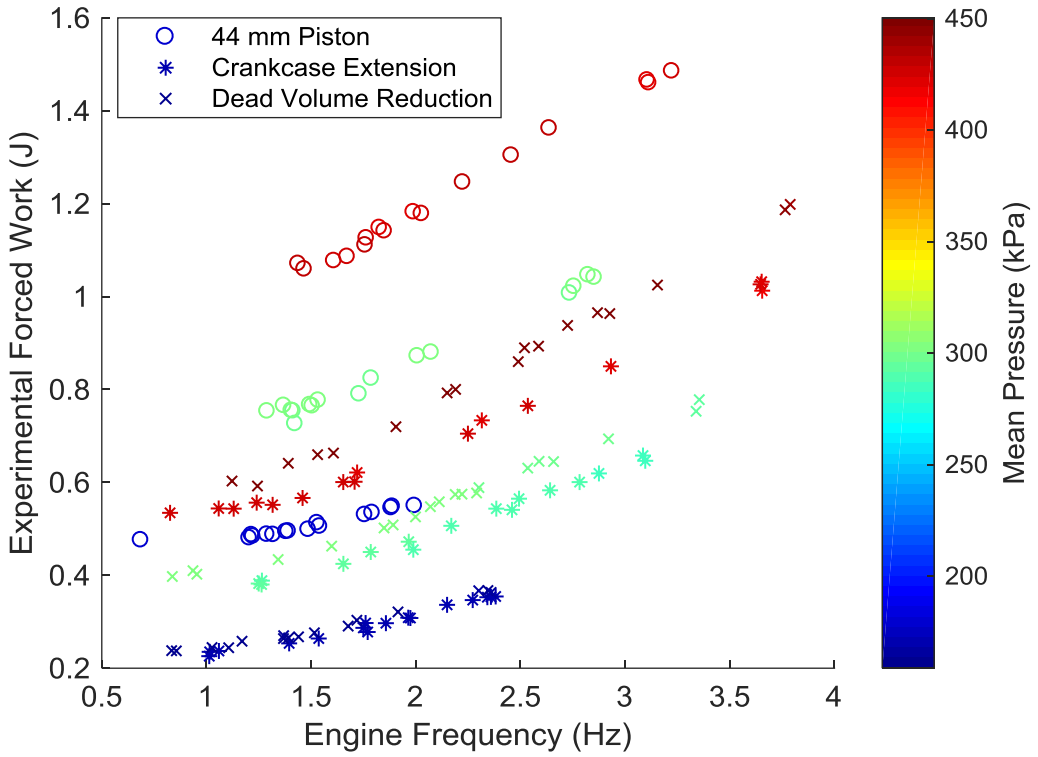


Figure 4.12: Experimental Forced Work for the Three Modifications

Further tests were deemed necessary to flush out the effect of the dead volume reduction parts. For this purpose, a second data set was collected with special attention payed to the engine fill pressure. These tests were completed at a thermal source temperature of 200 °C. The fill pressure was still difficult to duplicate between tests with, and without, the dead volume reduction parts installed. Post processing included removing outlier mean pressure data files so that mean pressures were within a range of 6 000 Pa (0.87 PSI) for the data presented below. The tests were performed with the 44 mm piston and the crankcase extension installed.

Figure 4.13 presents the indicated and shaft power results for the second data set. The addition of the dead volume reduction parts appears to have no influence on either of these results for the conditions tested.

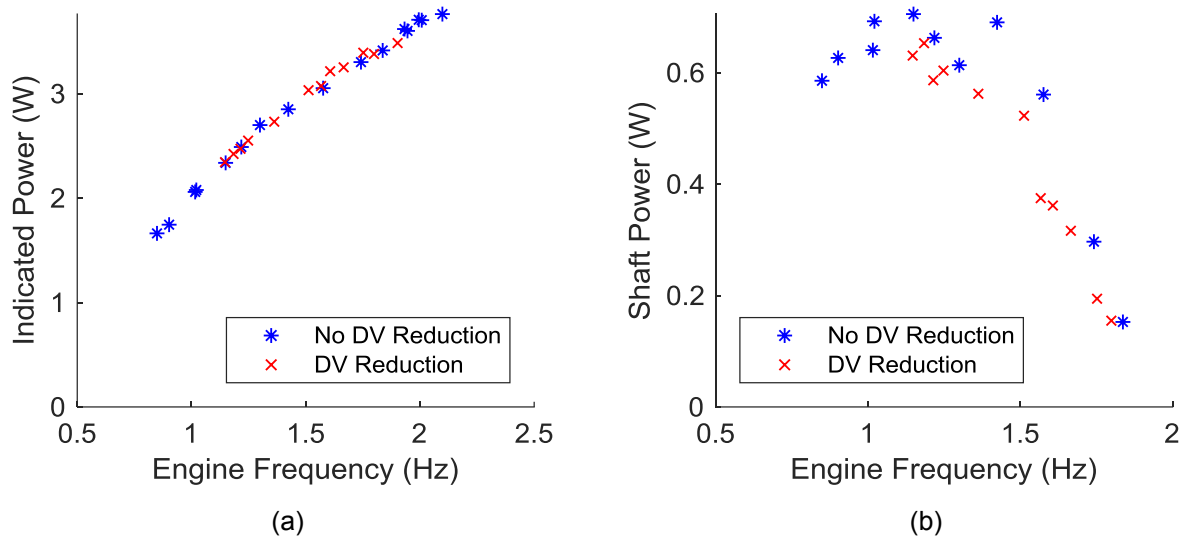


Figure 4.13: (a) Indicated Power and (b) Shaft Power as a Function of Frequency with and without the Dead Volume Reduction Parts

The only measurement that showed a distinct difference caused by the dead volume reduction is the heat rejection measurement. Heat rejection data is shown in Figure 4.14 below. The dead volume reduction components reduced the heat rejection rate in the connecting pipe and power cylinder water jacket, and increased the heat rejection rate in the cooler water jacket.

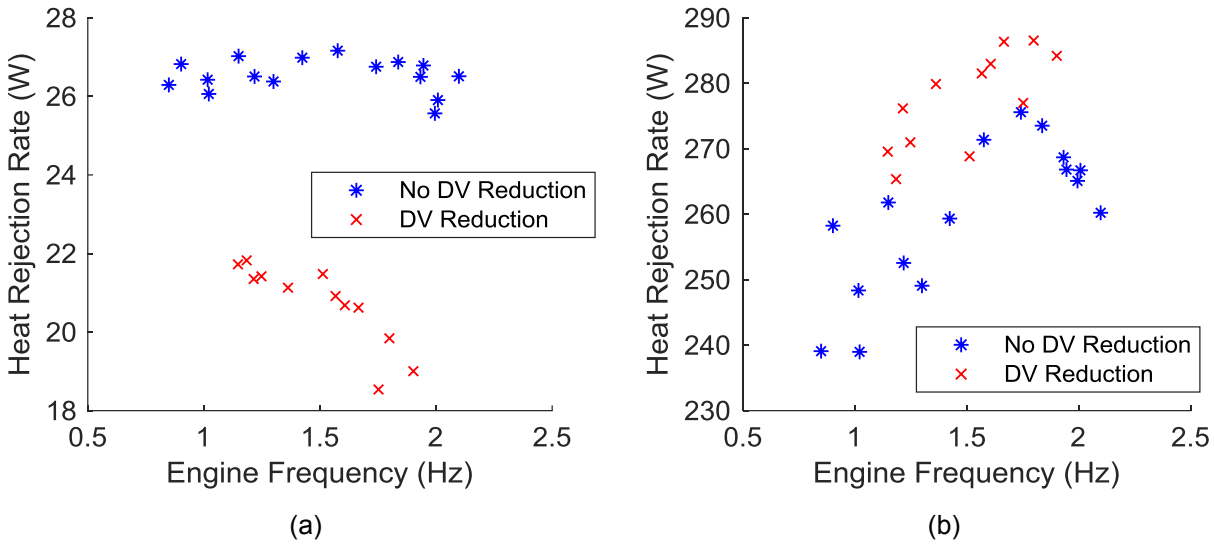


Figure 4.14: Heat Rejection Rates from (a) the Connecting Pipe and Power Cylinder and (b) the Cooler

4.4 Experimental Results – Cool Down Tests

Cool down tests were conducted to characterize the low thermal source temperature performance of the modified engine. These involved switching off the heating system with the engine running, and recording data until the engine stalled. Table 4.4 lists the minimum thermal source temperatures reached by the four engine configurations tested.

Table 4.4: Minimum Thermal Source Temperature Reached During Cool Down Tests for All Engine Configurations

Engine Configuration	Minimum Thermal Source Temperature (°C)
As-Built	242
44 mm Piston	185
44 mm Piston, and Crankcase Extension	144
44 mm Piston, Crankcase Extension, and Dead Volume Reduction	145

The results show that all modifications, except for the dead volume reduction, were successful in reducing the minimum thermal source temperature. The results shown in the table above were obtained with no load on the engine.

Additional cool down experiments were carried out in which the engine was working against a constant load. For these tests, the fully modified engine was used. Data was collected intermittently as the engine cooled. The cool down process took about an hour, and started at a heating cap temperature of 300 °C.

As in the modification comparison tests, maintaining a constant fill pressure was challenging with the current set-up. The engine had a slow leak, so it was left connected to the air compressor during the cool down test. Even in a perfectly sealed engine, the ideal gas behavior of the working fluid would decrease the mean pressure as the thermal source temperature was reduced. Imperfections in the pressure regulator on the compressor, and the check valve on the filling tube of the engine, led to an inconsistent flow rate of air into the engine to maintain the mean pressure. At the low power levels being measured during cool down experiments, these fill pressure fluctuations may have affected the data. For example, Figure 4.15 shows that the torque, frequency, and power all decrease with decreasing thermal source temperature. The trend is

approximately linear, with a steep drop off just before the stall temperature is reached. The accompanying fill pressure data was examined while investigating this drop off, and is displayed in Figure 4.16.

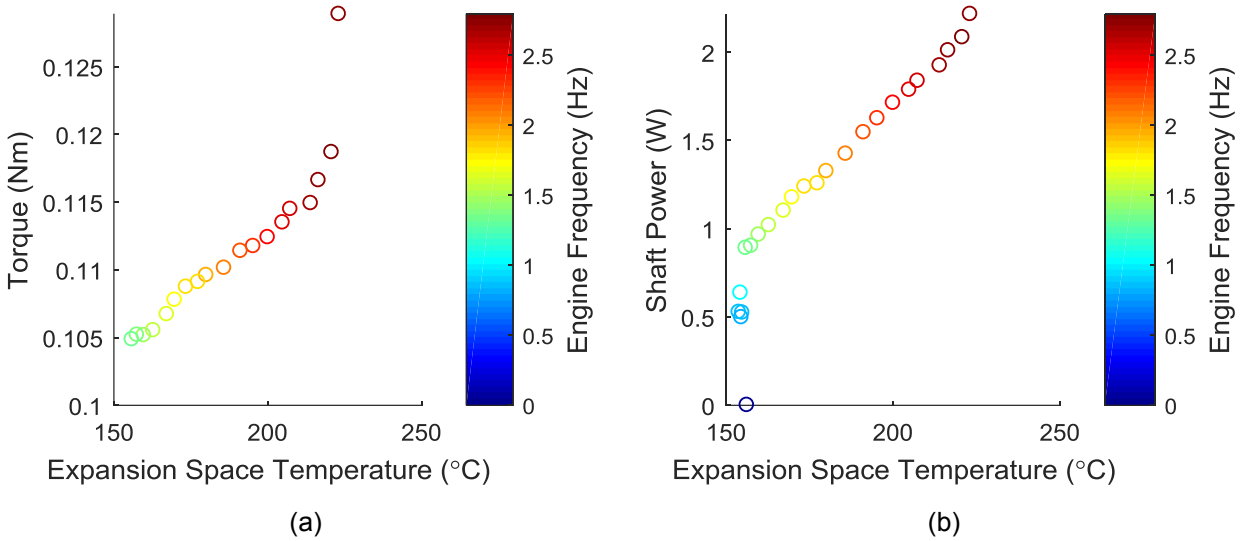


Figure 4.15: Loaded Cool Down Test Results for (a) Torque and (b) Shaft Power

In Figure 4.16, the red circles represent the mean power cylinder pressure and the black stars represent the mean crankcase pressure. Starting at an expansion space temperature of 180 °C and continuing downward, the mean pressures are observed to increase. Their rates of increase are steepest at the lowest expansion space temperatures, and the crankcase mean pressure changes at a different rate than the power cylinder mean pressure. The difference between the mean pressures of the crankcase and working space affect the amount of forced work in the cycle. Gas spring hysteresis in the crankcase also changes with crankcase mean pressure. Seal leakage characteristics are also expected to be affected by the pressure difference between the working space and the crankcase. So, the steep drop-offs in the torque and power plots of Figure 4.15 are probably caused by the steep changes in pressure visible in Figure 4.16. Further, the data collected above 180 °C expansion space temperature should show trends in expansion space temperature more faithfully, since the mean pressures were more consistent for that part of the experiment.

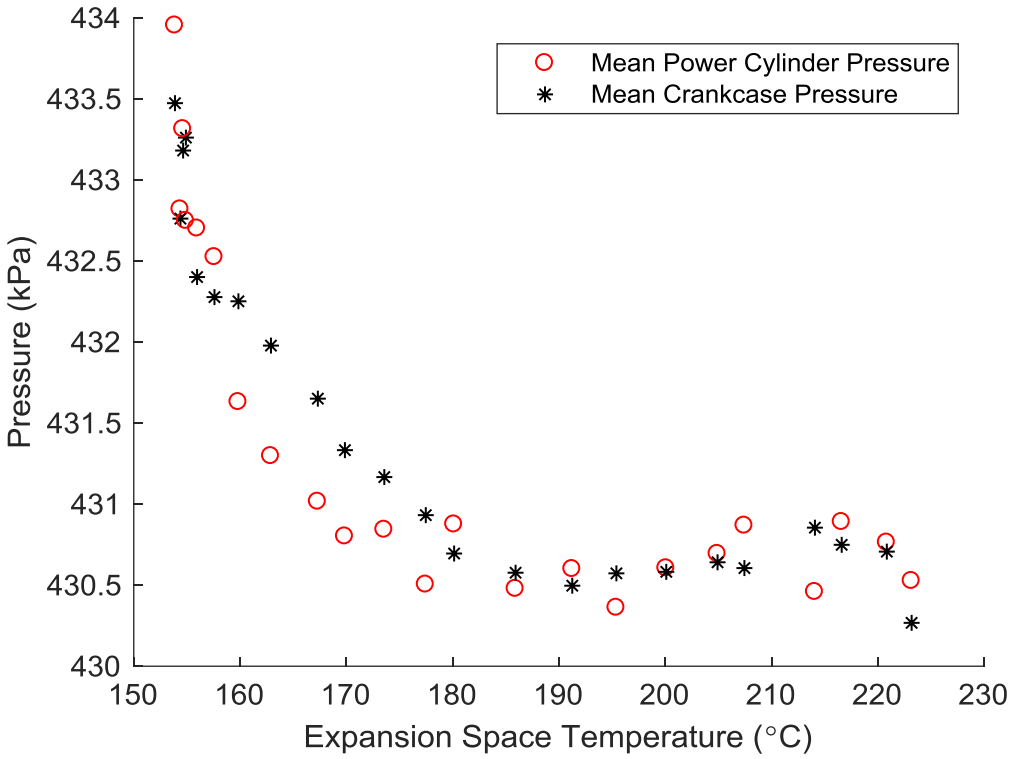


Figure 4.16: Mean Crankcase and Power Cylinder Pressures as a Function of Expansion Space Temperature for the Loaded Cool Down Test

In Figure 4.17, indicated work and forced work have been plotted for the same cool down trial. As expected, Figure 4.17 (a) shows that the indicated work diminishes with expansion space temperature. The adjacent plot indicates that the forced work has also decreased. The trends are approximately linear down to the expansion space temperature of 180 °C.

The steep drop in indicated work at an expansion space temperature of about 160 °C is counter-intuitive considering the mean pressure data of Figure 4.16. It is expected that an increase in pressure will increase the indicated work. This phenomenon is evidenced in Figure 4.7. The trend could be related to the engine speed, and the loss associated with seal leakage. Seal leakage is expected to be more severe at lower engine speeds since there is more time for mass transfer in a single cycle. There could be a speed at which seal leakage becomes the dominant loss mechanism. When this speed was reached, seal leakage losses would reduce the indicated work and the engine would slow further, leading to even more severe leakage. This compounding effect would quickly stall the engine.

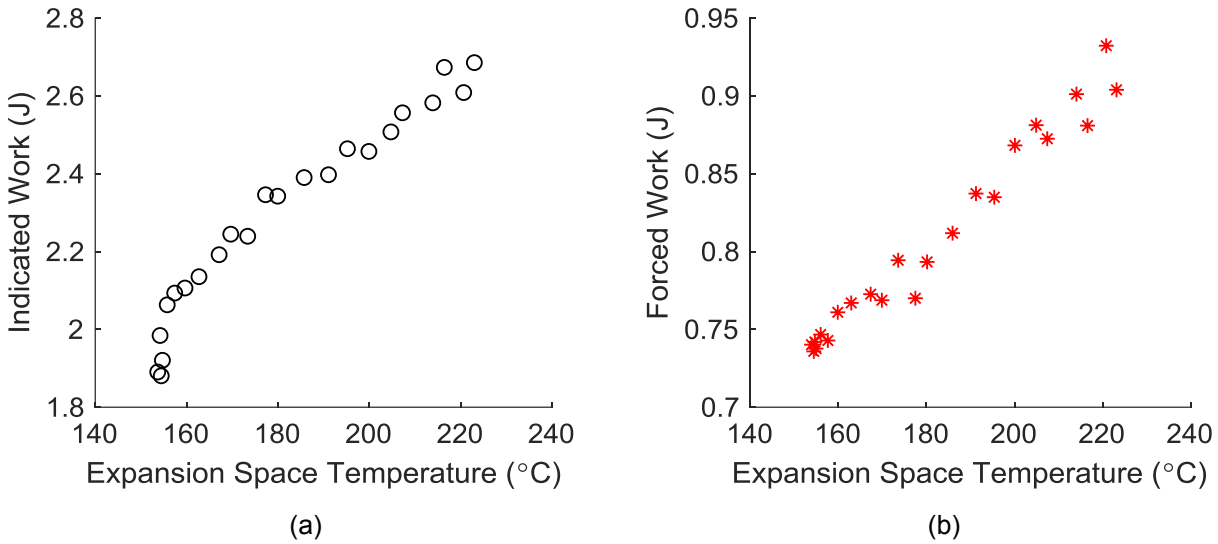


Figure 4.17: (a) Indicated Work and (b) Forced Work as a Function of Expansion Space Temperature for Loaded Cool Down Test with Fully Modified Engine

Figure 4.18 was produced with the seal leakage hypothesis in mind. The engine frequency is observed to follow a similar trend to the indicated work. Seal leakage is a plausible link between the two plots; however, verification work would require additional experiments in which the indicated power was measured for a series of speeds and leakage rates.

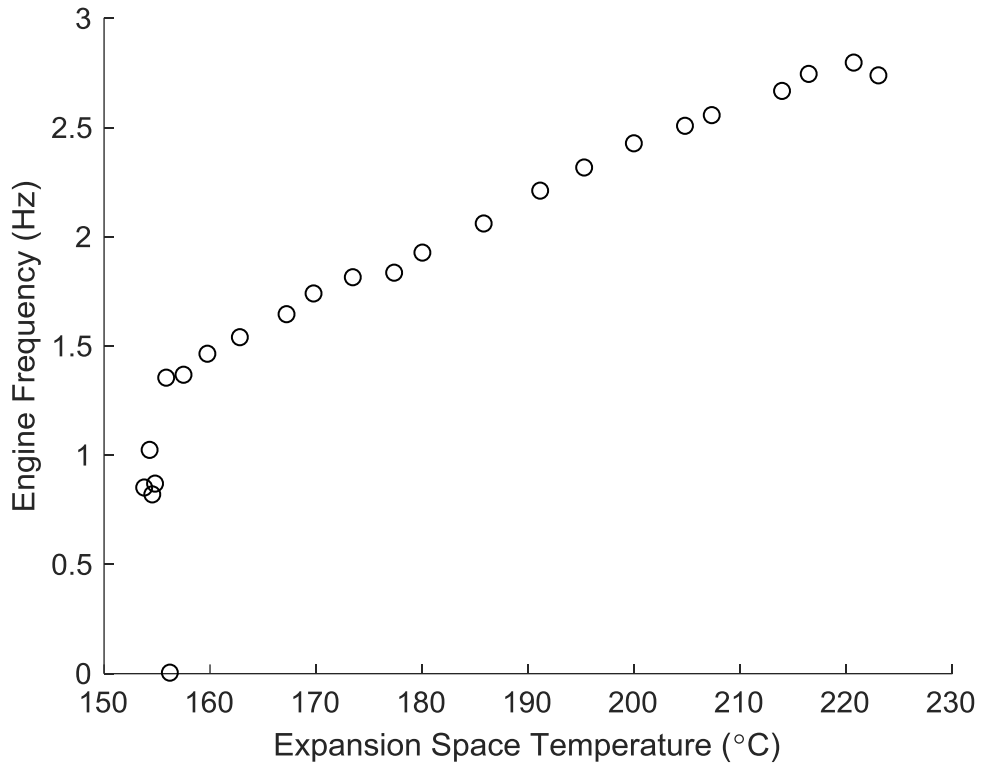


Figure 4.18: Engine Frequency as a Function of Expansion Space Temperature for the Loaded Cool Down Test

Figure 4.19 shows the indicator diagrams corresponding to the same cool down test, colored according to their expansion space temperature. Only data sets in which the expansion space temperature was greater than 180 °C were included, as these had the most consistent mean pressure according to Figure 4.16. The area of the indicator diagrams is observed to decrease with expansion space temperature. The orientation of the diagrams rotates counter-clockwise with decreasing expansion space temperature, which explains the reduction in forced work.

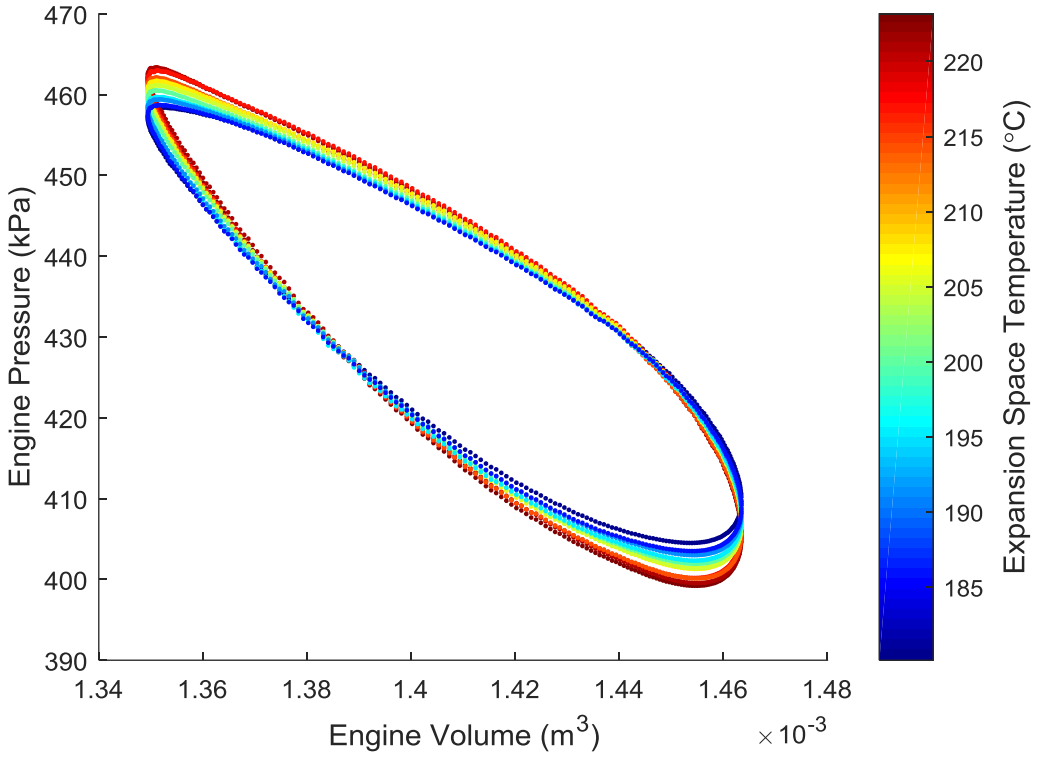


Figure 4.19: Series of Indicator Diagrams for Loaded Cool Down Test of Fully Modified Engine

It is the relative amount of forced work, compared to indicated work, which is the important quantity for engine performance. The ratio of forced work to indicated work has been plotted in Figure 4.20, as a function of expansion space temperature. The forced work ratio maintains an approximately constant average down to 180 °C, where it begins to increase rapidly. This is likely caused by the change in mean pressures indicated by Figure 4.16. The difference between the crankcase and power cylinder mean pressures strongly affects the amount of forced work. This difference changes, according to Figure 4.16, because the crankcase mean pressure changes at a different rate than the power cylinder mean pressure, when the expansion space temperature declines past 180 °C.

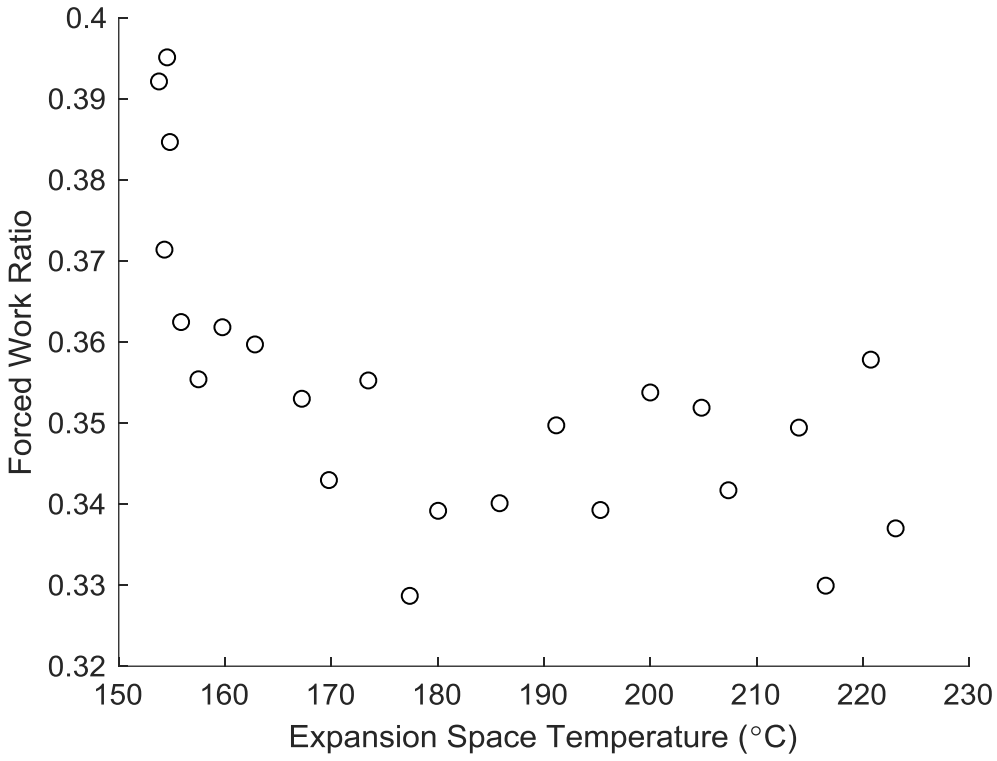


Figure 4.20: Forced Work Ratio as a Function of Expansion Space Temperature for Loaded Cool Down Test

Figure 4.21 shows that the gas spring hysteresis loss per cycle follows a similar trend to the forced work ratio for this cool down test. Again, the sudden change at 180 °C is plausibly due to the change in mean pressure, since gas spring hysteresis loss is strongly dependent on the mean pressure. This is because a larger mean pressure leads to larger pressure swings for a given change in crankcase volume. Larger pressure swings are accompanied by larger temperature swings, which encourage more heat transfer between the gas and the crankcase walls, and lead to higher hysteresis loss.

Another possibility is that the non-linear decrease in speed, which occurred at an expansion space temperature of 160 °C according to Figure 4.18, was the main cause of the abrupt changes in forced work ratio and gas spring hysteresis loss. A decrease in speed could change the indicator diagram shape due to changes in flow speed altering heat transfer rates, and due to an increase in seal leakage. This change in indicator diagram shape would influence the amount of

forced work. In the case of gas spring hysteresis, slower speed means more time for heat transfer between the gas and the crankcase walls, and a higher resulting loss.

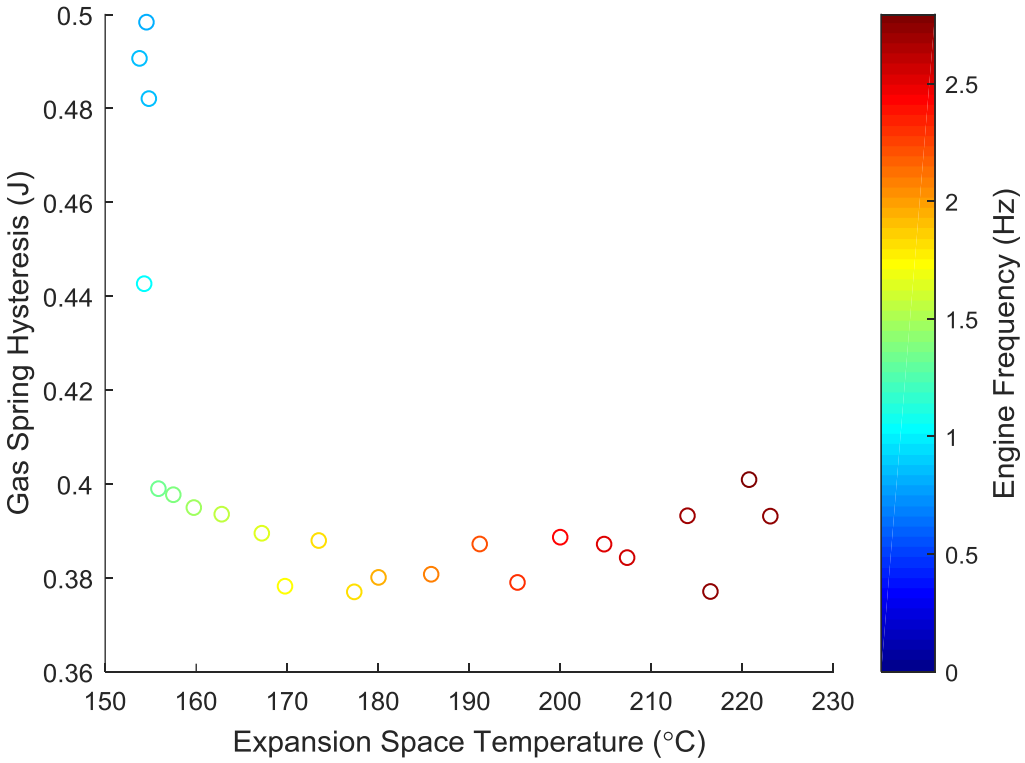


Figure 4.21: Gas Spring Hysteresis Loss as a Function of Expansion Space Temperature for the Loaded Cool Down Test

4.5 Conclusions

The as-built Stirling engine has been successfully modified to reduce its minimum required thermal source temperature. The decreased piston size and increased crankcase volume modifications improved low temperature performance by decreasing mechanical friction and crankcase gas spring hysteresis losses. The dead volume reduction parts changed the power output immeasurably, but did shift some heat rejection duty from the power cylinder/connecting pipe water jacket to the cooler water jacket. A loaded cool down test was performed with the aim of identifying barriers inhibiting lower thermal source temperature operation. Inconsistency in fill pressure, and coupled changes in temperature and speed, made the results difficult to interpret.

Stirling engines in the literature have operated on temperature differences as little as 0.5 °C, [69], so the temperatures achieved here do not represent a fundamental limit. Reducing the minimum thermal source temperature further will require further modifications, which could include

1. Further reductions in piston size and/or dead volume or a further increase in crankcase volume.
2. Changes in regenerator properties and/or heater head geometry.
3. Refinement of the mechanism to reduce mechanical friction.

The model needs to be improved and validated so it can be used to make more informed choices about which modifications to pursue for a desired performance change.

Chapter 5. Performance Evaluation and Model Assessment for Fully Modified Engine

In this chapter, an assessment of the second order model is made based on comparison to measured data pertaining to the fully modified engine and the final version of the test rig. The assessment follows a systematic approach, in which each sub-component of the model is evaluated independently by comparison to relevant experimental results. The chapter concludes with a comparison between measured and predicted overall performance.

5.1 Reference Cycle Indicator Diagram

Accurate prediction of the indicator diagram is the foundation of a second order model. The area of the indicator diagram represents the work done in one rotation of a lossless engine. The shape of the indicator diagram affects the calculation of decoupled losses. It is a prerequisite for the forced work method of estimating mechanical friction. Pressure swings from the indicator diagram are needed for calculation of heat transfer hysteresis, appendix gap, flow friction, regenerator enthalpy, imperfect heat transfer, and seal leakage losses. Experimentally, these losses affect the indicator diagram; hence, the true problem is a coupled one, which goes against the assumptions of the second order modeling approach.

The experimental indicator diagram was produced from data collected using the power cylinder pressure transducers, and the rotary encoder. Encoder pulses were converted to crank angles as described in Chapter 2, and rounded to the nearest degree. The result was a list of crank angles spanning several complete rotations over a ten second data set, each adjacent to a corresponding power cylinder pressure measurement. Then, all pressure measurements with the same crank angle value were averaged to produce a vector of 360 pressures, one for each crank angle degree. Volumes for each crank angle degree were calculated using the equations introduced in Chapter 3. Using these volumes and averaged pressures, the indicator diagram could be plotted and compared easily with outputs from the reference cycle models.

To compare experimental data to the reference cycle models, the measured gas temperatures and mean pressure were used as model inputs. Specifically, the average measured power cylinder pressure was taken to be the mean pressure in the model. The heater gas temperature was taken to be the average of the temperatures measured by the expansion space and regenerator hot thermocouples shown in Figure 2.10. To calculate the measured cooler gas temperature, the regenerator cold right and left thermocouple readings were averaged, and the result was averaged with the displacer mount thermocouple reading.

Figure 5.1 exhibits two experimental indicator diagrams, and their associated ideal isothermal and adiabatic model counterparts. Figure 5.1 (a) was produced using data collected with the as-built engine configuration at a heating cap temperature of 400 °C. The models are observed to approximate the measured data more closely than in Figure 5.1 (b), where the data comes from the fully modified engine at a heating cap temperature of 300 °C. Note that the mean

pressure is the same for the two plots, but the engine volume variations are different due to the difference in piston diameter. Taken together, the plots reveal that agreement between the reference cycle models and the experimental data is significantly better for the high temperature as-built engine than for the lower temperature fully modified engine. The plots typify those found for the high and low temperature testing campaigns, respectively.

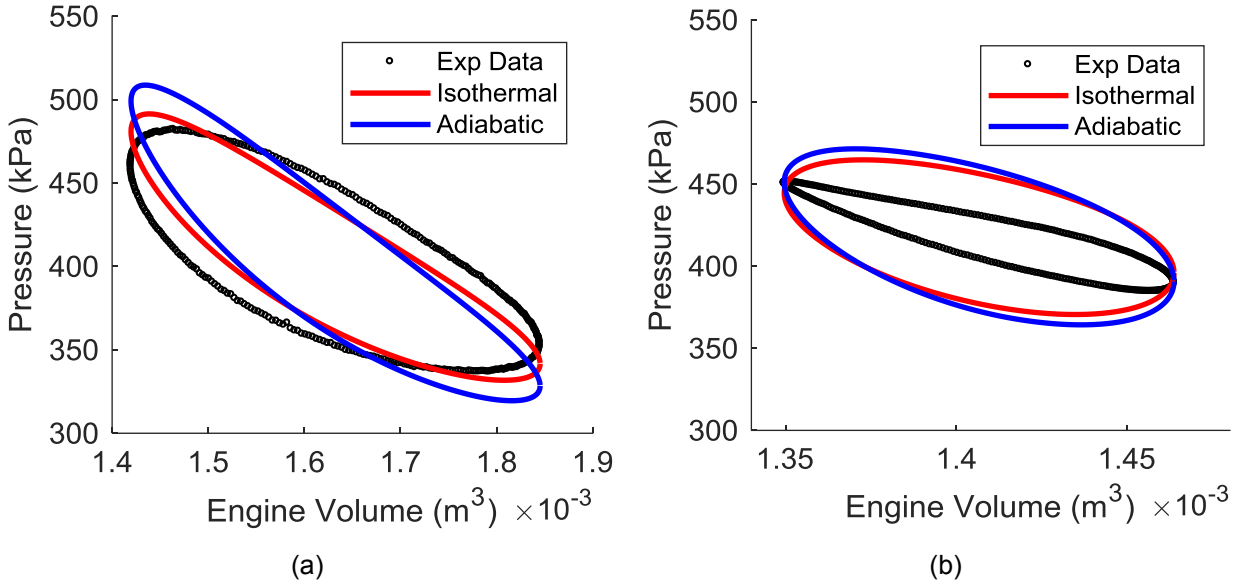


Figure 5.1: Experimental Indicator Diagrams with Overlaid Reference Cycle Models (a) As-Built Engine Running with a Heating Cap Temperature of 400 °C (b) Fully Modified Engine Running with a Heating Cap Temperature of 300 °C

Figure 5.2 presents data collected using the fully modified engine and two different heating cap temperatures. Both data sets were collected with a mean engine pressure of 414 kPa. The vertical axis gives the relative error in indicated work predicted by the isothermal and adiabatic reference cycle models. The figure demonstrates that at the lower heating cap temperature, the error in indicated work is higher for a given engine frequency. The isothermal model overshoots the measured value by a smaller amount than the adiabatic model for the conditions tested. For both heating cap temperatures, the indicated work error increases with engine speed.

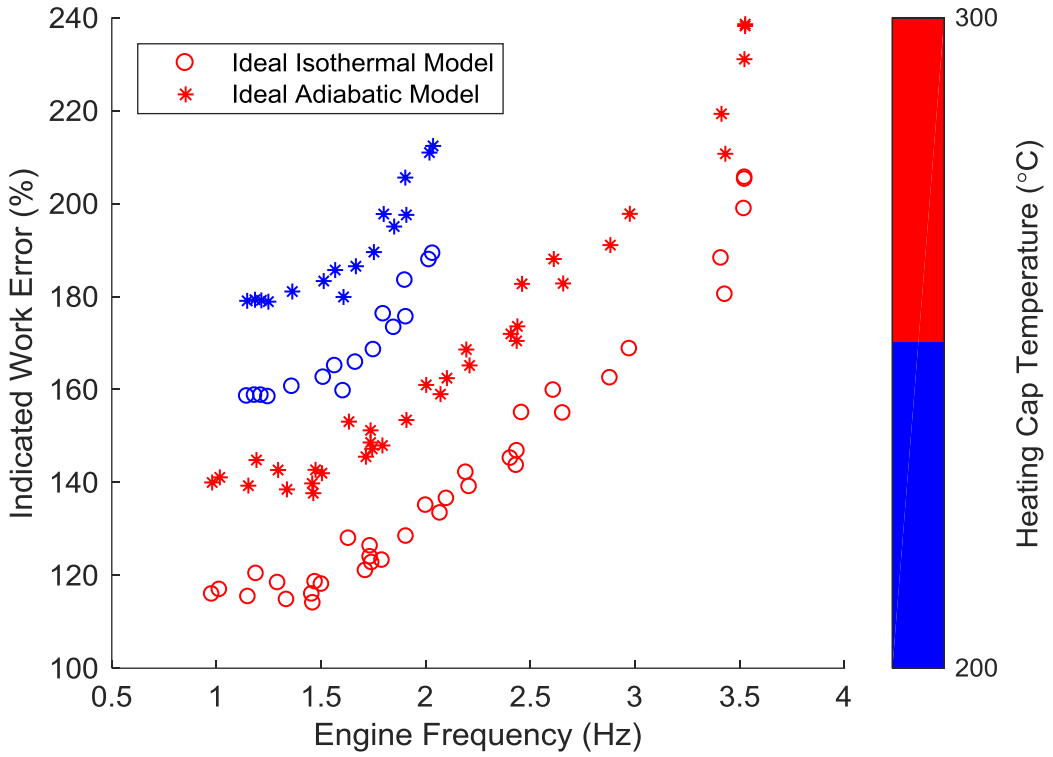


Figure 5.2: Indicated Work Error for Isothermal and Adiabatic Reference Cycle Models as a Function of Engine Frequency and Heating Cap Temperature

The reference cycle models do not include seal leakage, flow friction, heat transfer hysteresis, and other losses which influence the indicator diagram in the real system. The indicator diagram error is more severe for low temperature, low power operating points because losses neglected in the reference cycle models are more influential there, relative to the indicated work being produced. The increase in indicated work error with speed implies that, as a group, the neglected losses have a greater influence on the indicator diagram at higher speeds. The behavior is indicative of flow friction losses, which have not been accounted for in the reference cycle calculations.

The second order modeling approach assumes that the losses do not affect the indicator diagram. In fact, results from the reference cycle indicator diagram are used as inputs for the decoupled loss calculations; thus, any errors in the reference cycle indicator diagram propagate into the loss calculations. Figure 5.1 and Figure 5.2 demonstrate that, all else being equal, the

assumption of decoupled losses becomes increasingly unsound as the thermal source temperature of the engine decreases.

5.2 Crankcase Gas Spring Hysteresis

Gas spring hysteresis loss in the crankcase is calculated as the area of the crankcase indicator diagram. Since the volume amplitude is the same for the crankcase and the engine, the crankcase indicator diagram can be plotted as a function of either crankcase volume or engine volume, without affecting the area. Here, measured crankcase pressures were averaged for each crank angle degree and plotted as a function of engine volume. The area of the resulting loop was calculated numerically, and taken to be the crankcase gas spring hysteresis loss per cycle.

The measured crankcase gas spring hysteresis loss per cycle is plotted as a function of engine speed and mean pressure in Figure 5.3 below. The loss decreases slightly with speed and increases with mean pressure. Heat transfer between the gas and the crankcase walls is one of the main irreversibilities which causes gas spring hysteresis [6]. Lower speeds allow more time for heat transfer, resulting in a higher loss. Higher mean pressures lead to higher pressure swing amplitudes in the crankcase, which produce higher temperature swings, more heat transfer, and higher loss.

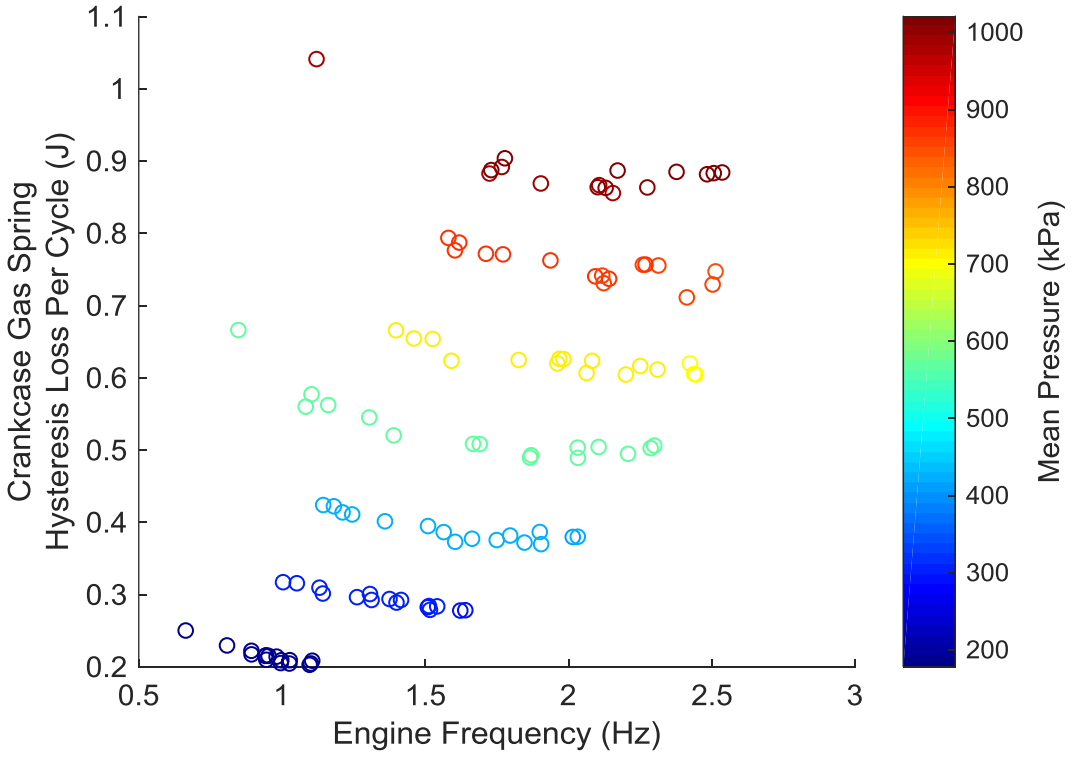


Figure 5.3: Measured Crankcase Gas Spring Hysteresis Loss for the Fully Modified Engine as a Function of Engine Frequency and Mean Pressure

A second order polynomial surface was fit to the gas spring hysteresis loss rate plotted against engine speed and mean pressure as shown in Figure 5.4. The coefficients of the fit equation, Eq. (5.1), were recorded and could be used to estimate the loss for arbitrary speed and pressure combinations. The fit was repeated for the three crankcase configurations tested: As-built, 44 mm piston, and 44 mm piston with crankcase volume extension. The fit coefficients, and R^2 values are listed in

Table 5.1.

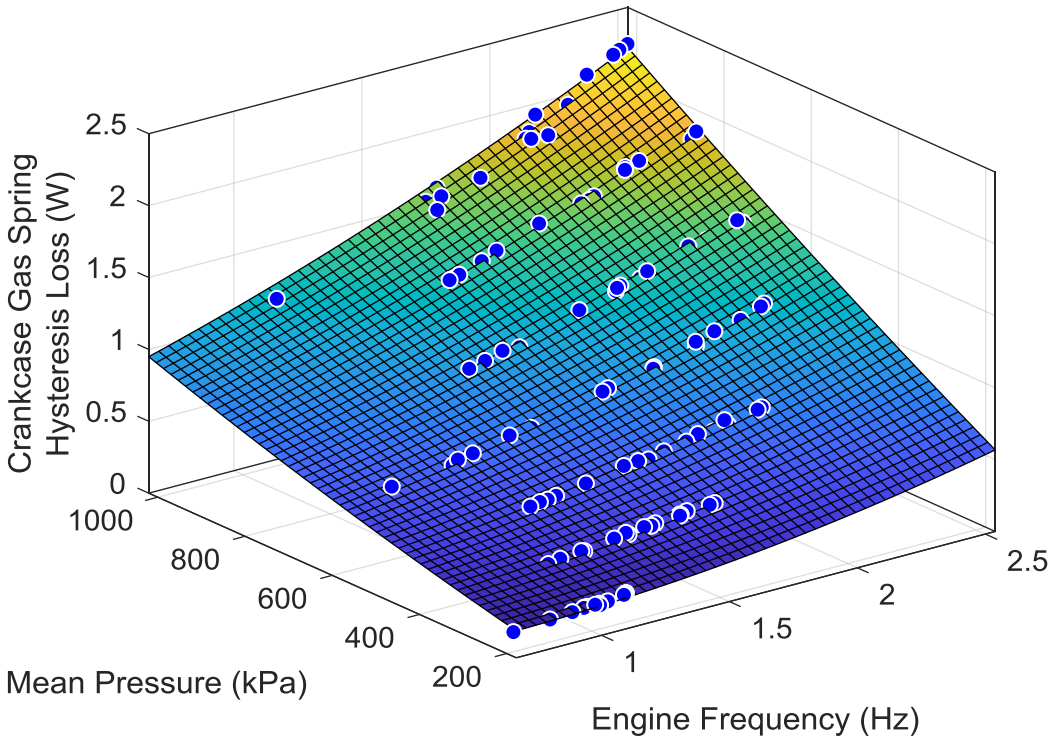


Figure 5.4: Second Order Polynomial Surface Fit to Measured Crankcase Gas Spring Hysteresis Loss Data

$$PL_{GSH} = c_1 + c_2 f + c_3 p_{mean} + c_4 f^2 + c_5 f p_{mean} + c_6 {p_{mean}}^2 \quad (5.1)$$

Table 5.1: Second Order Polynomial Fit Coefficients for Crankcase Gas Spring Hysteresis Empirical Formula

Crankcase Configuration	c_1	c_2	c_3	c_4	c_5	c_6
As-Built ($R^2 = 0.9991$)	-2.855	4.902	-1.239e-5	-0.4064	2.327e-5	7.373e-12
44 mm Piston ($R^2 = 0.9997$)	0.2473	-0.1534	2.311e-7	0.07564	2.076e-6	1.975e-13
44 mm Piston + Crankcase Extension ($R^2 = 0.9994$)	0.1625	-0.2039	2.503e-7	0.09528	5.668e-7	2.293e-13

5.3 Mechanical Friction

Mechanical friction is modeled using the forced work approach developed by James R. Senft. Models for the buffer pressure, engine pressure, and a value of mechanism effectiveness are prerequisites. It has already been shown that both the isothermal and adiabatic reference cycles represent the measured engine pressure variations poorly. This shortcoming will need to be improved for successful use of the forced work approach. Efforts to measure and model the remaining parameters will be discussed in this subsection.

Since the volume of the sealed crankcase varies while the engine is running, its pressure is not constant. Preliminary buffer pressure models assumed that the process could be represented by compression and expansion of an ideal gas in a container that was either isothermal or adiabatic. For the isothermal assumption, the crankcase pressure was calculated using the ideal gas law as

$$p_{CC,isothermal}(\theta) = \frac{m_{CC} R T_{CC}}{V_{CC}(\theta)} \tag{5.2}$$

For the adiabatic crankcase assumption, the crankcase pressure was calculated as

$$p_{CC,adiabatic}(\theta) = \overline{p_{CC}} \left(\frac{\overline{V_{CC}}}{V_{CC}(\theta)} \right)^{\gamma} \tag{5.3}$$

where the overbars denote mean values.

Figure 5.5 shows a typical crankcase indicator diagram overlaid with the two buffer pressure models shown above. The adiabatic buffer pressure assumption was found to better represent the crankcase pressure swings than the isothermal assumption. Even the adiabatic buffer pressure model falls short of the measured pressure swing amplitude. This could be the result of neglecting the volume of the displacer drive rod in the calculation of the crankcase volume variations. Seal leakage between the working space and the crankcase could also increase the crankcase pressure swings. Since the buffer pressure forms a loop, the forced work will be increased relative to that calculated using a single-line model of the buffer pressure. This adds additional error to the forced work model.

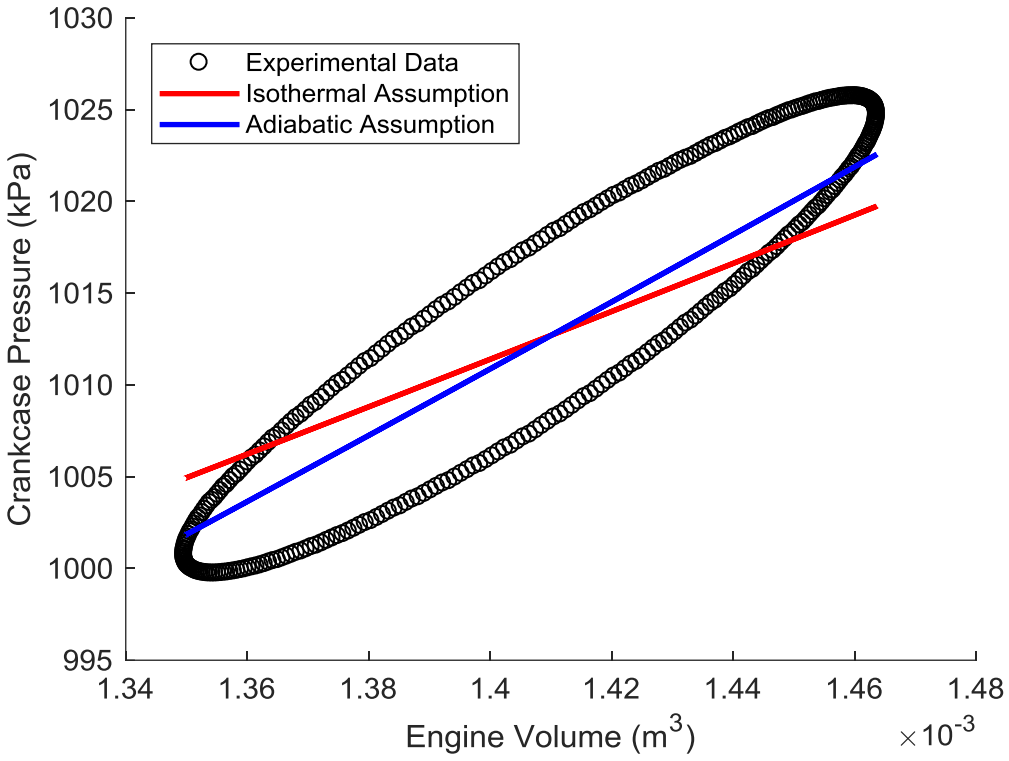


Figure 5.5: Measured Buffer Pressure with Overlaid Adiabatic and Isothermal Buffer Pressure Models

With models and experiments to determine the engine and buffer pressure curves, it is possible to calculate the amount of forced work. Graphically, this amounts to calculating the area of the red regions highlighted in Figure 3.9. Numerically, the forced work is calculated using the following algorithm[1]. The inputs are vectors of engine volume, engine pressure, and buffer pressure as functions of crank angle.

1. Apply a central differencing scheme to the engine volume vector to calculate the volume change increment that corresponds to each engine and buffer pressure value.
2. For each pair of corresponding engine and buffer pressure values, calculate the difference in pressure.
3. If the sign of a pressure difference vector element is opposite to its corresponding element in the volume change vector, calculate an incremental area of forced work by multiplying the two elements.
4. The sum of the forced work area elements is the total forced work.

[1] Personal communication with Steven Middleton.

The above method is the result of considerable effort by the author and his colleagues and has been validated by comparison to known areas.

The forced work approach assumes a constant mechanism effectiveness which represents frictional losses in the drive mechanism. The real mechanism effectiveness depends on crank position, due to the changing loads and geometry of the mechanism during operation. The most direct method of measuring the mechanism effectiveness would be to apply a known force to the piston and measure the resulting torque on the crankshaft for a series of crank positions. This could potentially be done using weights if the engine orientation was changed. The mechanism effectiveness has not been measured at this stage.

5.4 Flow Friction

The flow friction was measured using two methods. The first was a series of cold motoring tests with the engine in various stages of disassembly. Flow friction was also quantified by measuring the regenerator pressure drop, using an outboard pressure transducer, while the engine was running. This subsection examines the results of these two flow friction measurement approaches and, for the second approach only, agreement with the mathematical model.

Figure 5.6 presents the results of the cold motoring friction tests. These were performed using the test rig configuration shown in Figure 2.8. The engine was fitted with the 44 mm piston, but had the original crankcase and no dead volume reduction components during the cold motoring experiments. Input shaft power to the engine was measured, using the rotary encoder and torque transducer, for a range of engine frequencies at atmospheric pressure. The procedure was repeated for the following engine states:

1. Fully assembled (44 mm piston, original crankcase, no dead volume reduction components).
2. Crankcase cover removed.
3. Crankcase cover and heater head removed.
4. Crankcase cover, heater head, and power cylinder head removed. A 3-D printed part was manufactured to hold the power cylinder on in the absence of the power cylinder head.
5. Crankcase cover, heater head, power cylinder head, and cooler removed.

The plot reveals the relative influence of mechanical friction, flow friction in different components, and gas spring hysteresis. The difference between the blue circles and the red starts is small, indicating that the gas spring hysteresis loss in the crankcase is insignificant compared to mechanical and flow friction at high engine frequencies. Removal of the heater head, which also contains the regenerator, drops the power consumption significantly, showing that flow friction in these components is a relatively large contribution to the overall friction. It could be argued that, since elimination of the heater head has broken the seal between the working space and the surroundings, any gas spring hysteresis occurring in the working space has also been jettisoned. Considering the relative influence of the gas spring hysteresis in the crankcase, the contribution of the loss in the working space is assumed to also be relatively small; hence, the

large drop in power consumption after elimination of the heater head is attributed to flow friction in the heater and regenerator. The magenta x's show the measured power consumption with the cylinder head removed. The slight decrease in power is due to the absence of flow friction occurring in the cylinder head and connecting pipe. The green dots show a further decrease in power consumption when the flow friction of the cooler is taken away. The green dots represent the mechanical friction power consumption of the engine, as all flow friction and gas spring hysteresis causing components have been removed.

Figure 5.6 proves that mechanical friction and flow friction in the heater and regenerator are the two main contributions to the overall friction at atmospheric pressure. At higher pressures, gas spring hysteresis could play a more influential role.

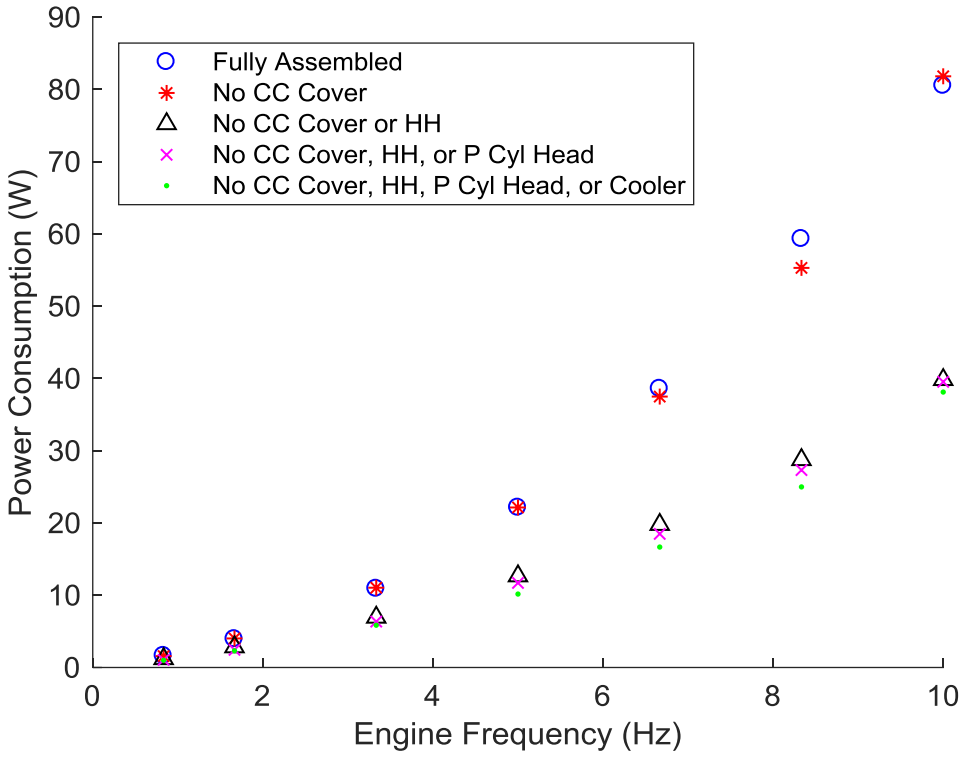


Figure 5.6: Measured Power Consumption During Cold Motoring Tests

The pressure drop across the regenerator was measured using an outboard pressure transducer connected via 1/8" stainless steel tubes to ports above and below the regenerator. Data was collected with the engine running at a heating cap temperature of 200 °C, and at a range of

frequencies and mean pressures. The effect of changing frequency and mean pressure on the measured regenerator pressure drop are exemplified in Figure 5.7 below. On the plots, measured data is represented by dashed lines, while results of the quasi-steady flow friction model are represented by solid lines. Positive pressure values indicate a pressure gradient which is declining away from the crankcase.

Figure 5.7 shows that regenerator pressure drop amplitude increases with mean pressure and engine frequency. The shape of the curves predicted by the model are similar to those measured. Pressure drops predicted by the model are symmetric about the horizontal axis, while the mean values of the measured pressure drops are below the x-axis. This could be a result of the calibration scheme used for the outboard differential pressure transducer. The curve fitted to the correction terms may have resulted in a zero offset since it was based on positive and negative pressure calibration data. The tube length of the outboard transducers has been shown to influence the measurement, in some cases, in Chapter 2. Here, since there is little phase offset between the pressure curves, it seems unlikely that the influence of the tubes was significant for these operating conditions.

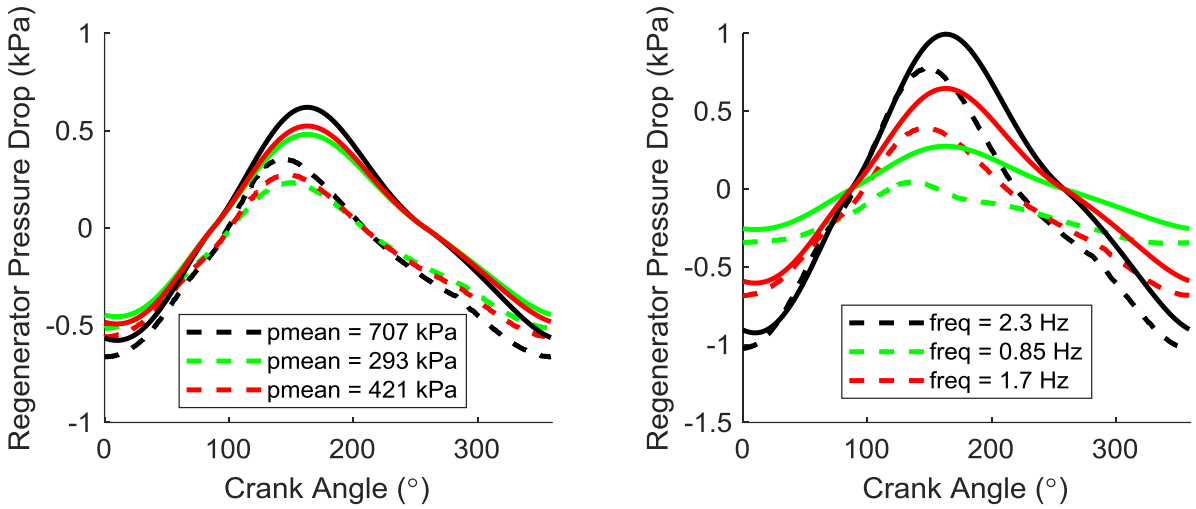


Figure 5.7: Measured Regenerator Pressure Drop as a Function of Crank Angle with Overlaid Quasi-Steady Model for (a) Three Mean Pressures at an Engine Frequency of 1.5 Hz and (b) Three Engine Frequencies at a Mean Pressure of 562 kPa (Positive Direction is from the Cold to the Hot Side of the Regenerator)

Figure 5.8 plots regenerator pressure drop amplitude as a function of engine frequency and mean pressure for the full data set collected at 200 °C heating cap temperature. The plot confirms the trend that regenerator pressure drop amplitude increases with engine frequency and mean pressure.

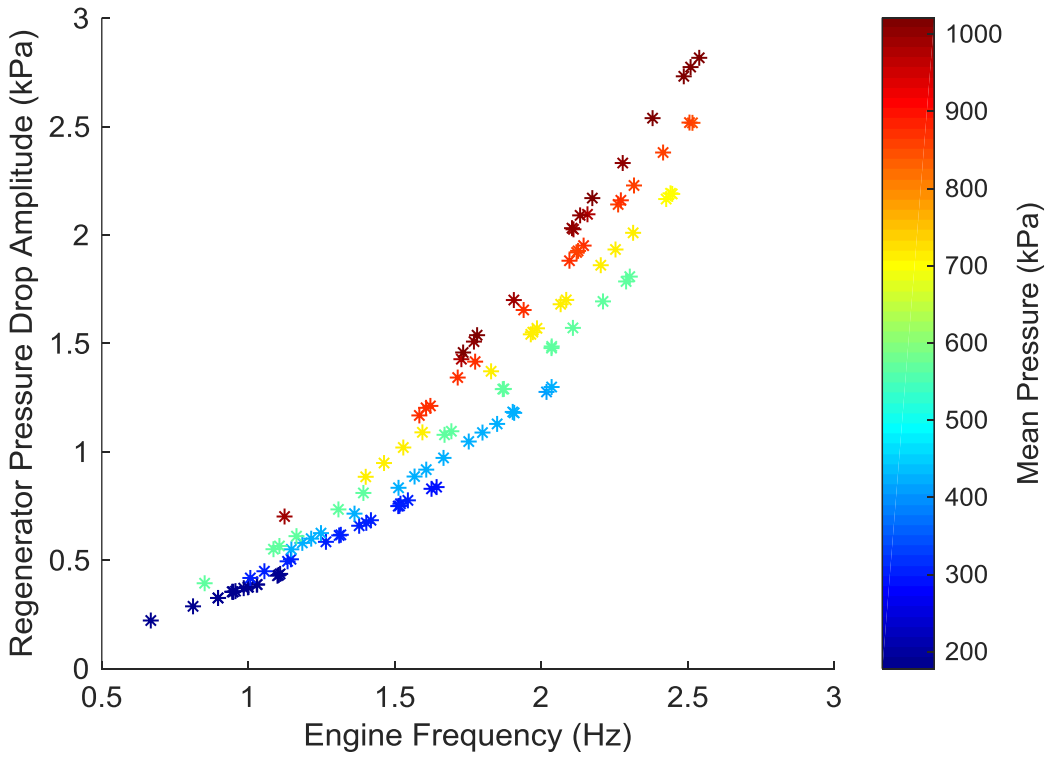


Figure 5.8: Measured Regenerator Pressure Drop Amplitude as a Function of Speed and Mean Pressure for a Heating Cap Temperature of 200 °C

In Figure 5.9, the error in regenerator pressure drop amplitude between the model and the experimental data is plotted as a function of engine frequency and mean pressure for the same data set plotted in Figure 5.8. The error was calculated as the difference between the modeled and measured pressure drop amplitude divided by the measured pressure drop amplitude, and is expressed as a percent. The plot shows that the agreement between the model and the data is closest at high speed, high pressure operating points. At low speed, low pressure operating points, agreement is relatively poor. Many low temperature difference Stirling engines operate at low speeds and mean pressures. Significant overestimation of the regenerator pressure drop

could occur when using this simple heat exchanger model to predict performance of these engines.

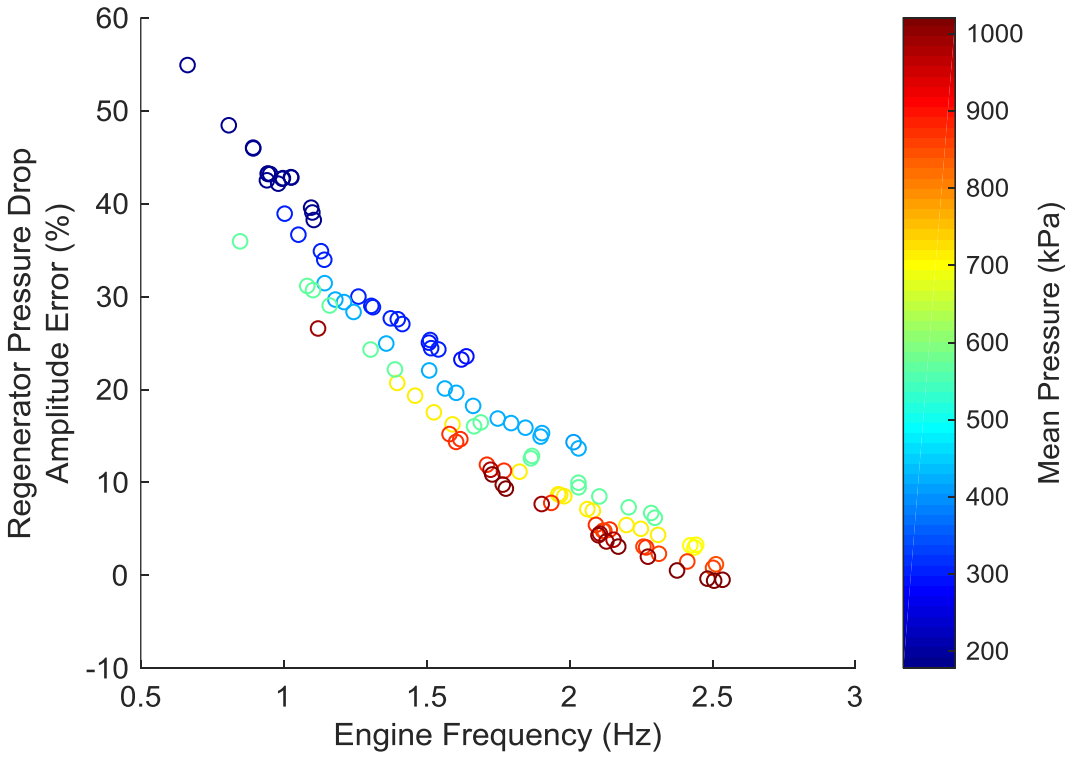


Figure 5.9: Regenerator Pressure Drop Amplitude Error as a Function of Engine Frequency and Mean Pressure

Note that, in the model, the regenerator pressure drop calculation is based on results from the ideal adiabatic reference cycle model, which has already been shown to represent the data inaccurately regarding the indicator diagram. The pressure drop calculation is also based on steady flow correlations with questionable applicability to the current problem. Further assessment of this flow friction modeling approach should continue after prediction of the indicator diagram has been improved.

5.5 Heat Exchanger Performance

The overall heat exchanger performance includes heat transfer rate, flow friction, and dead volume contribution; however, this section will focus on the heat transfer rate only. Regarding heat transfer rate, a perfect cooler would have an average gas temperature equal to the thermal sink temperature, and a perfect heater would have an average gas temperature equal to the thermal source temperature. For this research, the heating cap and water bath setpoints were taken to be the thermal source and sink temperatures, respectively.

The heat transfer performance of the heater and cooler was quantified by comparing measured gas temperatures to the heating cap and water bath setpoints. For the heater, the gas temperature was taken as the average of the expansion space and regenerator hot thermocouple measurement locations shown in Figure 2.10. To calculate the measured cooler gas temperature, the regenerator cold left and regenerator cold right thermocouple readings were averaged, and the result was averaged with the displacer mount reading. The difference between the heating cap setpoint and the average heater gas temperature is denoted the heater temperature drop, and the difference between the average cooler gas temperature and the water bath setpoint is denoted the cooler temperature drop. Ideally, these temperature drops will be as low as possible.

Figure 5.10 plots the measured heater and cooler temperature drops for a data set collected at a heating cap temperature of 200 °C using the fully modified engine. Note that the vertical axis limits are different between the two plots. Temperature drops are expressed as a function of engine frequency and mean pressure.

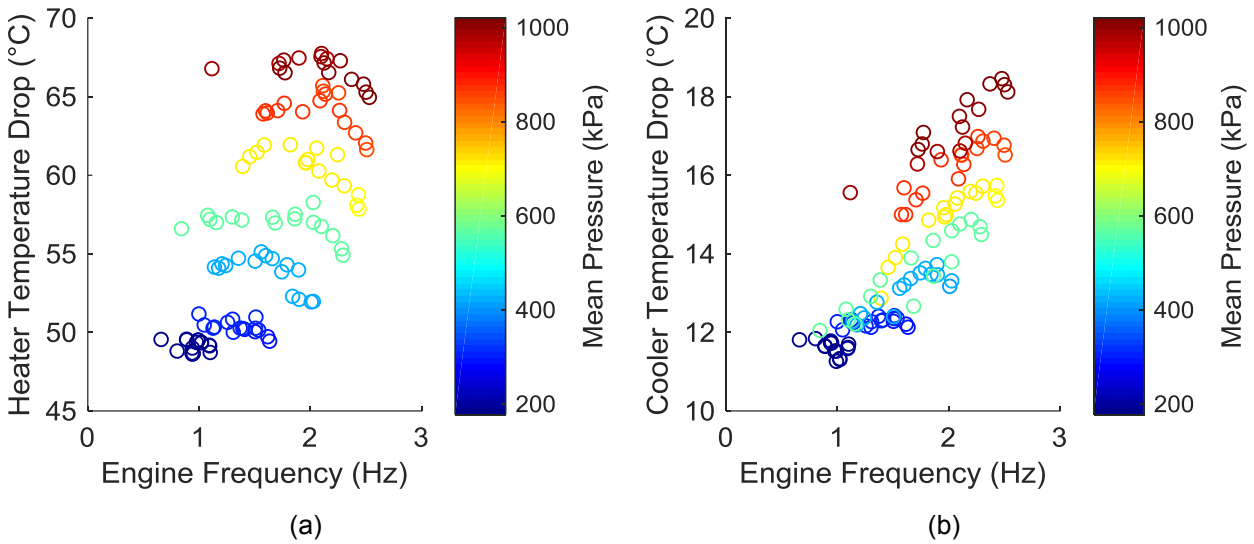


Figure 5.10: (a) Measured Temperature Drop between the Heater Gas Temperature and the Heating Cap Setpoint and (b) Measured Temperature Drop between the Cooler Gas Temperature and the Water Bath Setpoint

In both plots of Figure 5.10, the temperature drops are observed to increase with mean pressure. This trend is what would eventually lead to a peak in shaft power as the mean pressure was increased. As the pressure is increased, the engine is effectively working across a lower temperature difference.

The temperature curves in Figure 5.10 slope downwards near the end of their frequency ranges. In these areas, the temperature drops decrease even though the average mass flow rate through the heat exchangers has increased. The trend could indicate a laminar to turbulent transition or some other change in the fluid flow phenomena, which improves the heat transfer rate above a threshold frequency.

The measured temperature drop combines a series of heat transfer processes between the working fluid, and the thermal source and sink. For the cooler, heat is transferred from the coolant to the aluminum internally finned tube via convection. The heat then conducts thorough the tube walls and fins and is finally transferred to the gaseous working fluid by convection. The heat from the heating cap must conduct from the steel heating cap to the steel heater head of the engine. It then conducts through the heater head and is convected into the working fluid of the engine.

The plots in Figure 5.10 show that the temperature drop of the heater is larger than that of the cooler for a given frequency and mean pressure. The cooler has the advantage of being made of aluminum rather than low carbon steel, giving it an approximately three times higher thermal conductivity and reducing the temperature drop during the conduction process. The walls of the cooler are also 2.5 mm compared to a thickness of 6 mm for the heater head walls. The clearance fit between the heating cap and the heater head is expected to have a large contact resistance associated with it, adding to the temperature drop from the thermal source. For convective heat transfer to the working fluid, the cooler has finer, larger aspect ratio slots than the heater, which may improve its convective heat transfer performance.

Pending further experiments in which the temperatures of solid surfaces are measured, comparison with calculations is of limited use. The current model assumes a wall temperature in the heater and cooler and models the convection process using the simple heat exchanger model described in Chapter 1. Using internal and external wall temperature measurements, the conduction and convection processes could be individually modelled and validated.

Considering Figure 5.10, the minimum thermal source temperature could potentially be reduced significantly by replacing the heater with something that closely resembled the current cooler and using a liquid thermal source. The changes in flow friction and dead volume associated with the suggested heater replacement would also affect the performance, so the resulting reduction in minimum thermal source temperature may not directly correspond to Figure 5.10. The idea is a promising, but expensive, option for reducing the minimum thermal source temperature further.

5.6 Evidence of Preferential Flow

Gheith et al., [54], noted that measured regenerator temperatures were warmer on the power cylinder side of the engine than on the bypass side during their experiments. They attributed the difference in temperature to a preferential flow on the power cylinder side of the engine [54].

Figure 5.11 presents gas temperature data taken at the cold end of the regenerator. Each column of data in the plot represents a range of engine frequencies tested at a common mean pressure. Like in Gheith et al.'s experiments, the power cylinder side of the regenerator, shown by the red stars, is typically warmer than the opposite side. The plot reveals that this difference in temperature is more pronounced at low mean pressures than at high mean pressures. In the previous section, it was shown that the pressure drop across the regenerator increases with mean pressure. This high pressure drop is expected to even out the flow more effectively, and could be the reason that the temperatures are more even on opposite sides of the regenerator at higher mean pressures.

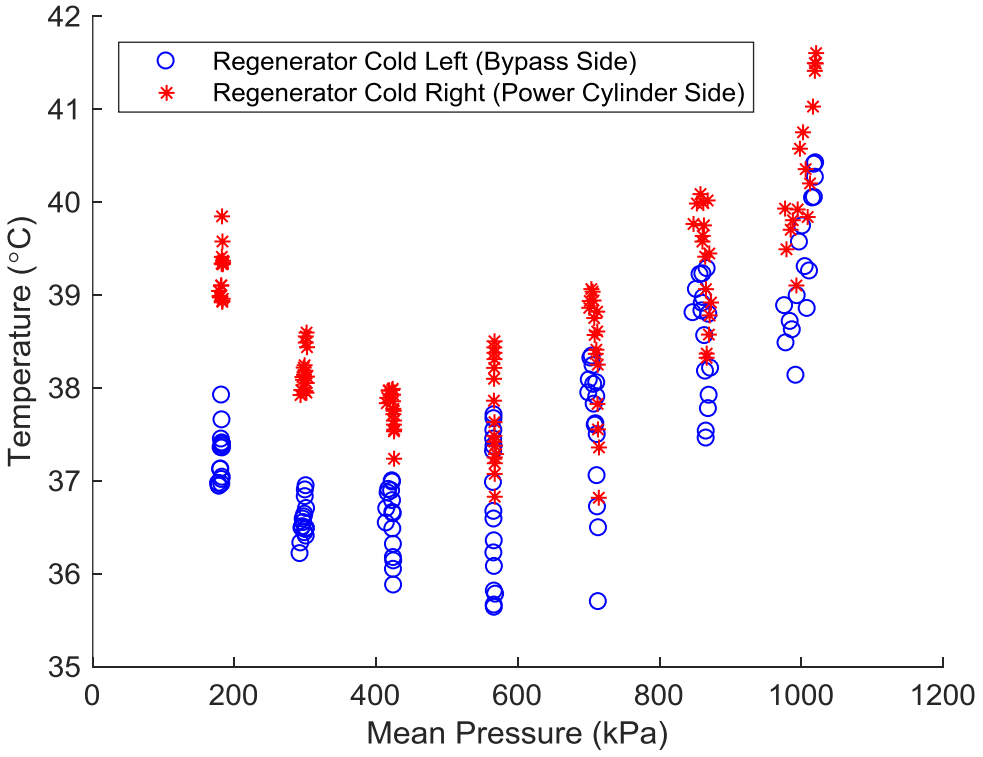


Figure 5.11: Measured Gas Temperatures at the Interface between the Regenerator and the Cooler with Engine Running at a Heating Cap Temperature of 200 °C

Evidence of preferential flow was also recorded during a cold motoring test. The engine was rotated using an electric motor at 10 Hz with a mean pressure of 414 kPa, and the pressure drop across the heater was measured. An outboard pressure transducer was connected via 1/8" stainless steel tubes to the expansion space and regenerator hot locations indicated on Figure 2.15. The test was repeated with the heater head rotated to eight azimuthal positions in 45° increments. Figure 5.12 is a polar plot showing the measured heater pressure drops in kPa. The higher pressure drop on the power cylinder side of the engine suggests a higher flow rate of gas through the regenerator on that side.

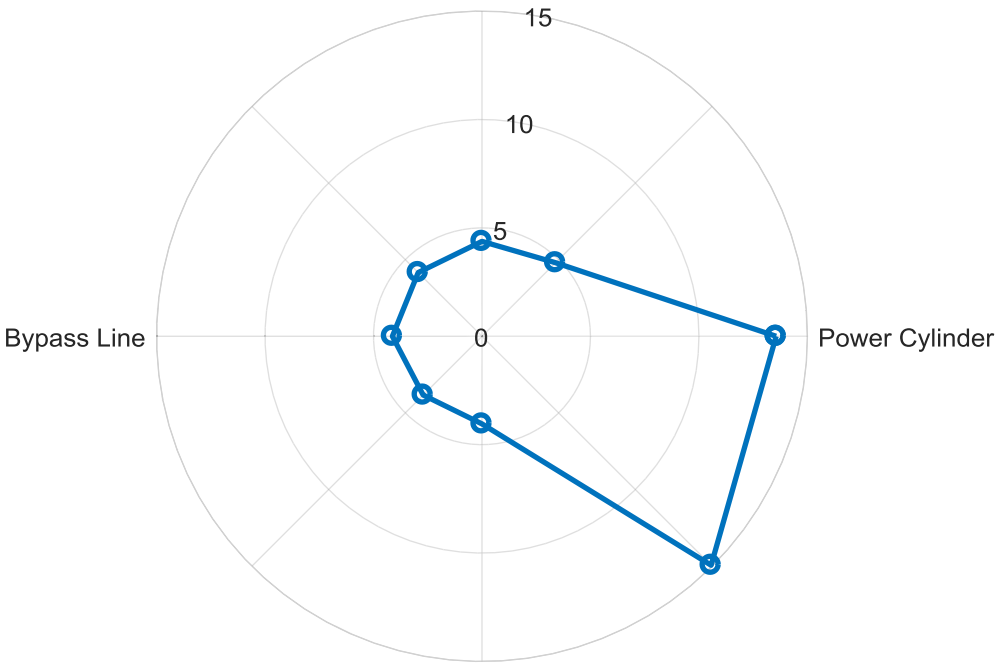


Figure 5.12: Measured Heater Pressure Drop Amplitude (in kPa) as a Function of Azimuthal Position

The preferential flow found by Gheith et al., [54], has been confirmed by measurements here. It has been shown that the issue is more significant at lower mean pressures. There is a larger opening at the bottom of the cooler on the power cylinder side due to the presence of the connecting pipe. This is expected to be the cause of the preferential flow towards that side.

5.7 Heat Lost Through the Heating Cap Insulation

Some of the heat supplied by the cartridge heaters does not enter the engine at all, rather it escapes into the surrounding air through the insulation. This loss mechanism was not accounted for in the model, but it was measured by comparing the heat input and rejection rates with the engine stationary. The measured heat loss rate is displayed as a function of heating cap setpoint in Figure 5.13 below.

It is expected that the heat loss rate will monotonically increase as the heating cap setpoint increases. The plot shows a slight decline in the heat loss rate between the heating cap setpoints of 250 °C and 300 °C. This is because the insulation was removed and refitted during the conduction loss testing which spanned several days. The 300 °C data point was collected before the insulation was removed and reinstalled, and the remaining 4 data points were collected post reinstallation.

The results convey that the heat lost through the insulation is significant. At best, it is about 31 % of the heat input for this stationary test. The problem can be reduced by adding additional layers of insulation or by substituting an alternative product with a higher thermal resistance.

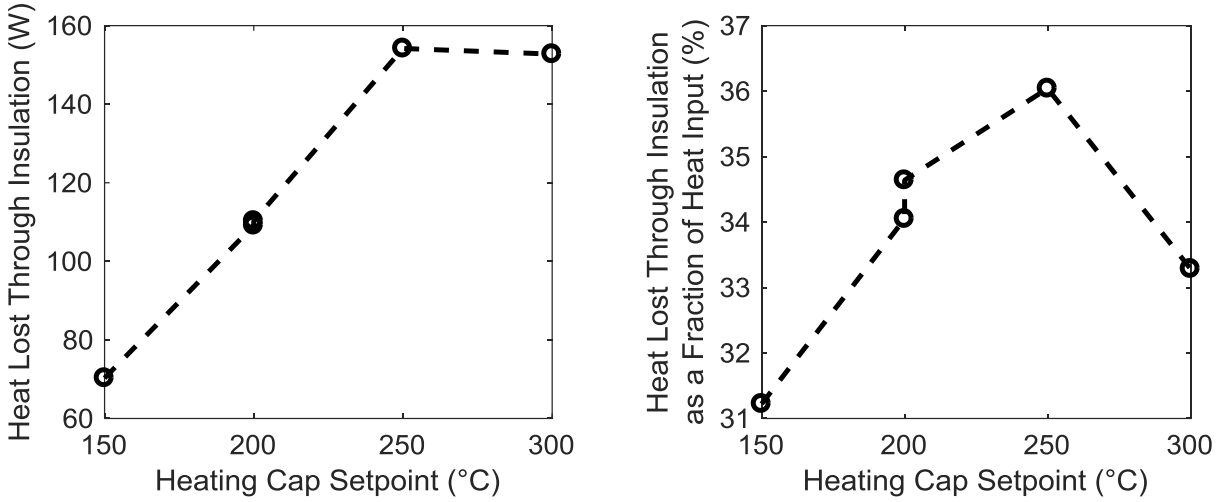


Figure 5.13: Measured Heat Loss Rate Through the Heating Cap Insulation Expressed as (a) An Energy Transfer Rate and (b) A Fraction of the Heat Input Rate

5.8 Conduction loss

The conduction loss was measured using the same procedure as the heat loss through the heating cap insulation. The engine was held stationary at operating temperature and the heat input and output rates were measured. The rate of heat removed by the coolant during the test represents the conduction loss of the engine.

In the model, the conduction loss was estimated using a one-dimensional Fourier's Law approach based on the average thermal conductivities of the relevant engine components. To compare the model to experimental results, the measured gas temperatures above and below the regenerator were used as the hot and cold side temperatures for the calculation of conduction through the regenerator and walls of the regenerator cavity. The measured gas temperatures of the expansion space and displacer mount were used for the calculation of heat conduction through the displacer.

Figure 5.14 (a) shows the measured conduction loss with the model overlaid. The model is observed to overestimate the conduction loss for the heating cap temperatures studied. A more complete experiment would include measurements of the wall temperatures during the experiment, and a more complete model would include multi-dimensional conduction, contact resistances, and temperature dependent material properties. Figure 5.14 (b) expresses the measured conduction loss rate as a fraction of the measured heat input. The fraction of input heat conducted away increases as the heating cap setpoint declines.

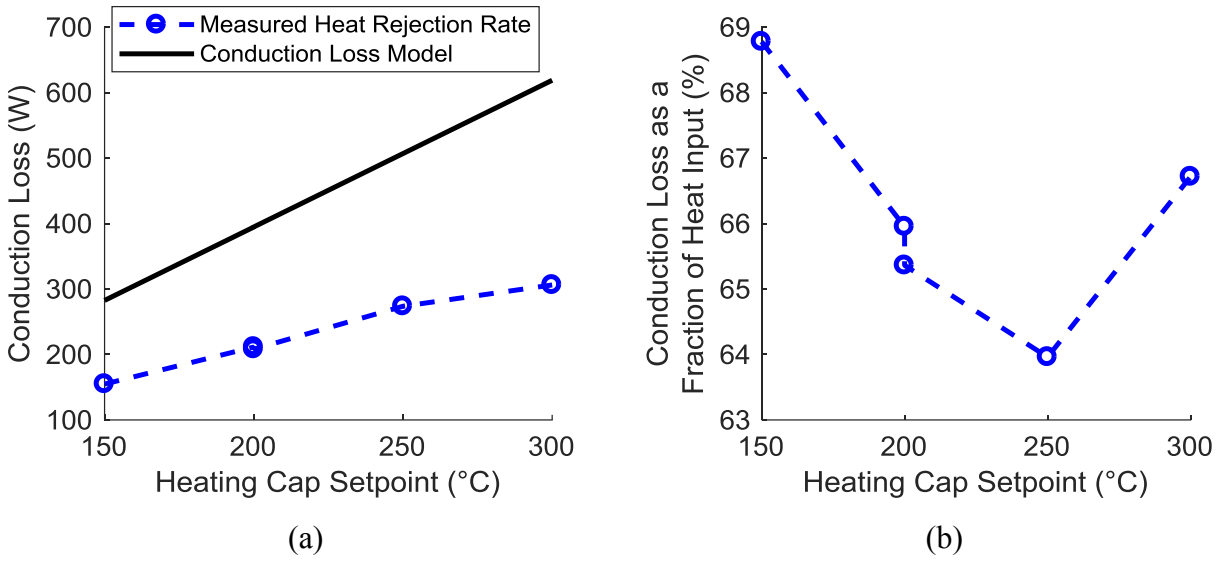


Figure 5.14: (a) Comparison between Measured and Calculated Conduction Loss Rate and (b) Measured Conduction Loss Rate Expressed as a Fraction of the Measured Heat Input

Since the gas and wall temperatures are different when the engine is running from when it is stationary, the results of this experiment will inherently have some error. It is expected that the temperature differences across the conduction paths will be less when the engine is running and so the true conduction loss will be less than that measured in a stationary test.

5.9 Relative Influence of the Cooling Zones

The water jackets of the engine were grouped into two cooling zones for testing. The first zone was the cooler water jacket, and the second zone was comprised of the connecting pipe and power cylinder water jackets connected in series. In the second zone, water flowed through the power cylinder before flowing through the connecting pipe. In both zones, water flowed from bottom to top. Each zone was circulated using an peristaltic pump, and had temperature measurements at the inlet and outlet.

Provision for water jackets complicates the engine design considerably. It is desirable to know the relative influence of the two zones under different operating conditions, so that informed decisions can be made during the water jacket design of future engines. Figure 5.15 plots the portion of the total heat rejected in the connecting pipe/power cylinder as a function of engine frequency and mean pressure. Data for the plot was collected at a heating cap temperature of 200 °C. As the pressure increases, a larger portion of the total rejected heat is removed by the connecting pipe/power cylinder zone. The trend can be explained with reference to Figure 5.10. At higher mean pressures, the average gas temperature in the cooler is higher and the working gas exits the cooler at a higher temperature. This allows the connecting pipe and power cylinder to remove a greater share of the reject heat.

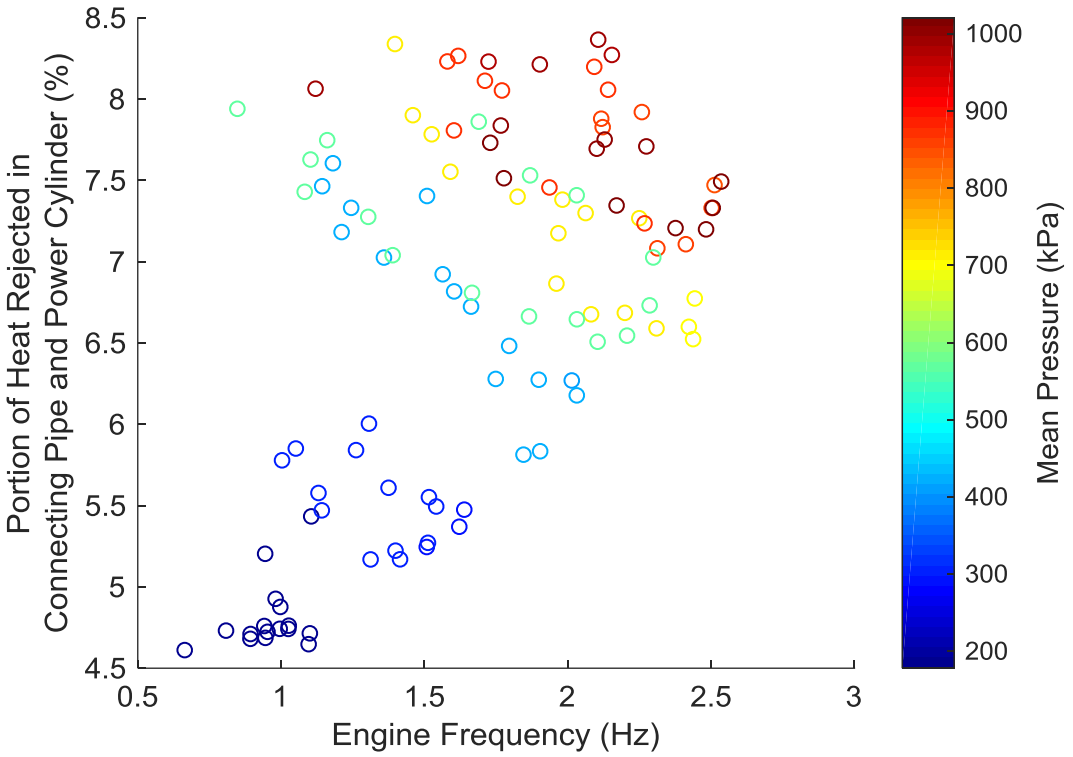


Figure 5.15: Portion of Heat Rejected in the Connecting Pipe and Power Cylinder as a Function of Engine Frequency and Mean Pressure for a Heating Cap Temperature of 200 °C

5.10 Discussion of Losses Not Measured

Ideally, each sub-component of a second order model would be validated independently to obtain improved agreement between predicted and measured overall engine performance. Using this approach, sources of error in the model are isolated; therefore, subsequent efforts to improve the model may be focused on components which contribute the largest errors. The research presented here followed this scheme as much as possible when comparing models to experimental data. The losses that were not measured are discussed in this subsection.

Regenerator enthalpy loss is calculated using the simple heat exchanger model described in Chapter 1, but it has not been measured. Measurement of this loss would require instantaneous temperature measurements collected at the hot and cold ends of the regenerator while the engine was running. From these the regenerator effectiveness could be calculated and used to quantify the regenerator enthalpy loss. The response time of the current thermocouples is too slow to use this approach.

Validation of the appendix gap loss would involve instantaneous temperature and flow field measurements in the appendix gap. This could potentially be achieved using optical diagnostic techniques, but replicating the engine conditions while remaining optically transparent would be difficult. Recall that the purpose of the appendix gap is to prevent the seal on the displacer from getting too hot. In low thermal source temperature applications, this is less of a concern. The appendix gap could potentially be removed altogether, eliminating the loss mechanism.

Seal leakage was not investigated through modeling or experiments in this research. Leakage past the displacer, leakage past the piston and displacer drive rod, and leakage of working fluid into the surrounding room are all expected to affect the performance of the real engine. Numerically, seal leakage could be accounted for during the reference cycle calculations, provided that the relation between leakage mass flow rate and pressure difference across the seal was known. Experimentally, the effects of seal leakage could be measured by introducing leaks of known size into the engine and measuring the resulting performance.

Heat transfer hysteresis, which is gas spring hysteresis in the working space, has not been calculated or measured. Like the gas spring hysteresis of the crankcase measured earlier, its calculation would involve evaluating the heat transfer and viscous dissipation processes

occurring inside the engine as it was running. The calculations await the creation of a far more detailed model of the engine. In the meantime, an empirical approach could be taken, like that used for the crankcase. An option would be to spin the engine at operating temperature and pressure with the displacer held stationary, and measure the indicator diagram in the power cylinder. If this was done for a range of frequencies, temperatures, and pressures an empirical formula could be obtained via curve fitting.

Imperfect heat transfer losses are calculated using the simple heat exchanger model described in Chapter 1. Wall temperatures in the heater and cooler are assumed to be constant throughout the calculation. Experimental validation would involve gas, and wall temperature measurements. Ideally, these would be instantaneous. Using the calculated mass flow rate and surface area, and the measured wall and gas temperatures, the convective heat transfer coefficients of the heater and cooler could be determined, and their performance could be assessed.

Flow friction in the connecting pipe and cylinder head is absent from the model and the experimental data. A quasi-steady approach like that used for the heater, cooler, and regenerator could be implemented in the model as a starting point. Measurement would be possible using high speed differential pressure measurements across the associated components.

Piston finite speed loss was not included in the model nor was it measured. It is difficult to devise a measurement technique which could isolate losses caused by the speed of the piston from gas spring hysteresis and flow friction. The derivation of the equations used by other researchers could not be found by the current author, leading to skepticism of their validity. Since second order models typically overshoot experimental data, the addition of any small loss mechanism will often improve agreement if only overall shaft power and efficiency are used in the validation.

5.11 Overall Engine Performance

The section below presents plots related to the overall engine performance. Specifically, measurements of the torque, shaft power, heat rejection rate, and thermal efficiency will be presented. Comparisons will also be made to the results of the second order model. Significant errors in the model have already been identified in previous subsections. Discrepancies in overall performance results represent the accumulation of these sub-component errors.

Torque measurements as a function of engine frequency and mean pressure are plotted in Figure 5.16 below. Data for the plot was collected at a heating cap temperature of 200 °C. It is observed that peak torque occurs at minimum engine frequency. This is where the effective temperature difference is the highest and the frequency dependent losses of flow friction are the lowest. If seal leakage was significant, the torque would be expected to drop off at low speeds due to increased mass leakage per cycle. The torque curves increase with mean pressure at a diminishing rate, with the highest two mean pressures giving torque curves that are nearly identical. Increasing the pressure allows the working gas to carry more energy per unit volume. This benefit comes at the cost of reduced heat exchanger performance, as demonstrated in Figure 5.10. Eventually, further elevation of the mean pressure will decrease the torque, and an optimum mean pressure will be discovered. This optimum pressure lies above the maximum rated pressure of the engine for the conditions tested.

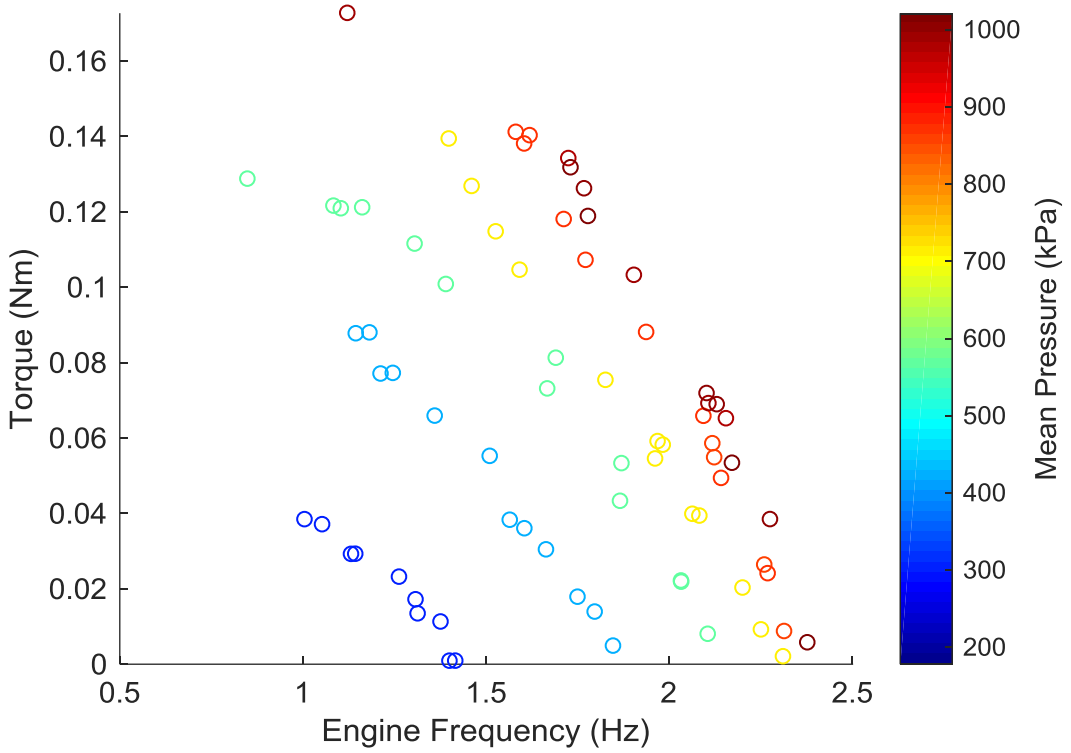


Figure 5.16: Measured Torque as a Function of Engine Frequency and Mean Pressure for a Heating Cap Temperature of 200 °C

Figure 5.17 plots measured shaft power as a function of engine frequency and mean pressure for the same data set. Shaft power declines sharply with engine frequency at this heating cap temperature as a direct result of the sharp decline in torque shown on the previous figure. The peak power frequency increases with mean pressure because flow friction and other speed dependent losses constitute a reduced fraction of the power output. Again, the benefit of increasing mean pressure is less pronounced towards the upper limit of the pressures tested, forecasting an optimum will be reached if pressure is elevated further.

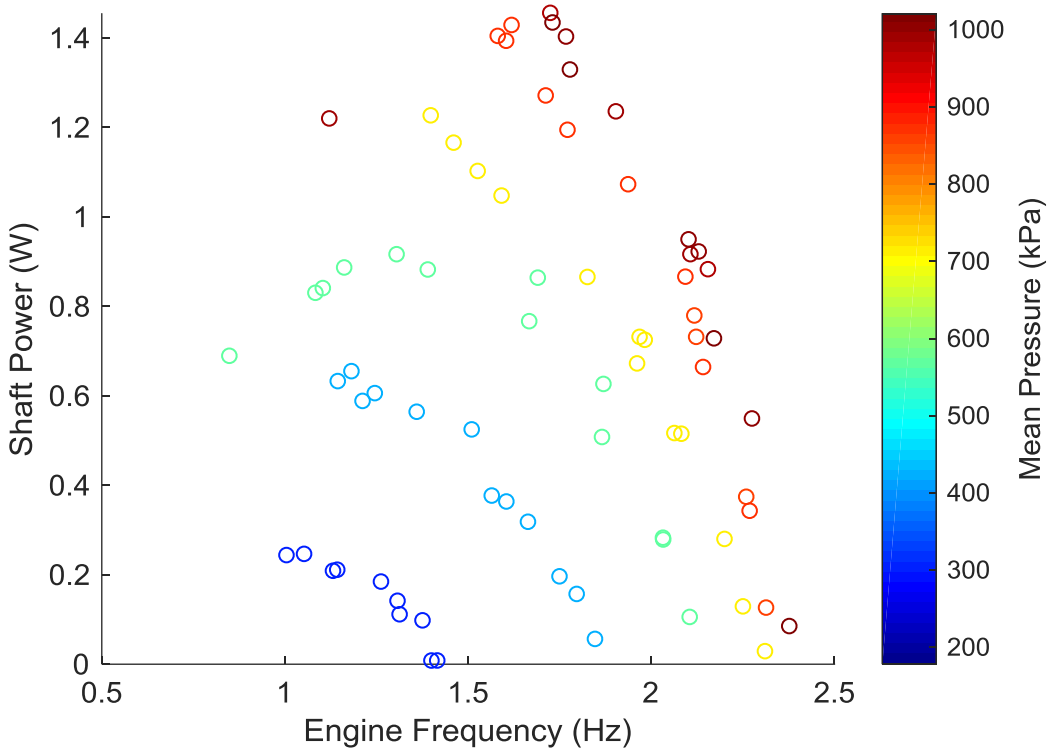


Figure 5.17: Measured Shaft Power as a Function of Engine Frequency and Mean Pressure for a Heating Cap Temperature of 200 °C

The thermal efficiency of the engine was measured by recording the average torque and heat input over a 30-minute run at approximately constant speed. For these experiments, the sampling rate for all measurements was reduced to 10 Hz, to reduce the size of the data files. At this sampling rate, speed could not be measured using the rotary encoder because missed pulses made crank position uncertain. Instead, the speed was measured before and after the test using the normal voltage sampling rate of 30 000 Hz. If the two speeds differed, the average was taken to be the speed of the test. The maximum difference between the start and end speeds of the seven efficiency tests was 0.2 Hz. Speeds were adjusted before each test using the brake. All efficiency tests were performed with a heating cap temperature of 300 °C, and a mean pressure of 430 $\pm$ 10 kPa.

Figure 5.18 plots the results of the thermal efficiency tests as a function of engine frequency. The measured thermal efficiency is less than 0.5 % under these conditions. The low efficiency values are attributed to the high heat losses through conduction and the heating cap insulation.

Appendix gap losses, and regenerator enthalpy losses also reduce the efficiency, but have not been experimentally quantified. For comparison, the Carnot efficiency at the same source and sink temperatures is 49 %. The Carnot efficiency at maximum power, calculated using the Curzon-Ahlborn relation [91], is 28 %. The two lowest frequency data points have different measured thermal efficiencies even though they were collected under the same conditions. The speeds between the two tests were the same on average, but they had different start and end speeds. One test had a frequency of 1.25 Hz at the beginning and the end of the run, while the other started the run at 1.2 Hz and finished it at 1.3 Hz. This speed difference could be the cause of the discrepancy between the two efficiency data points.

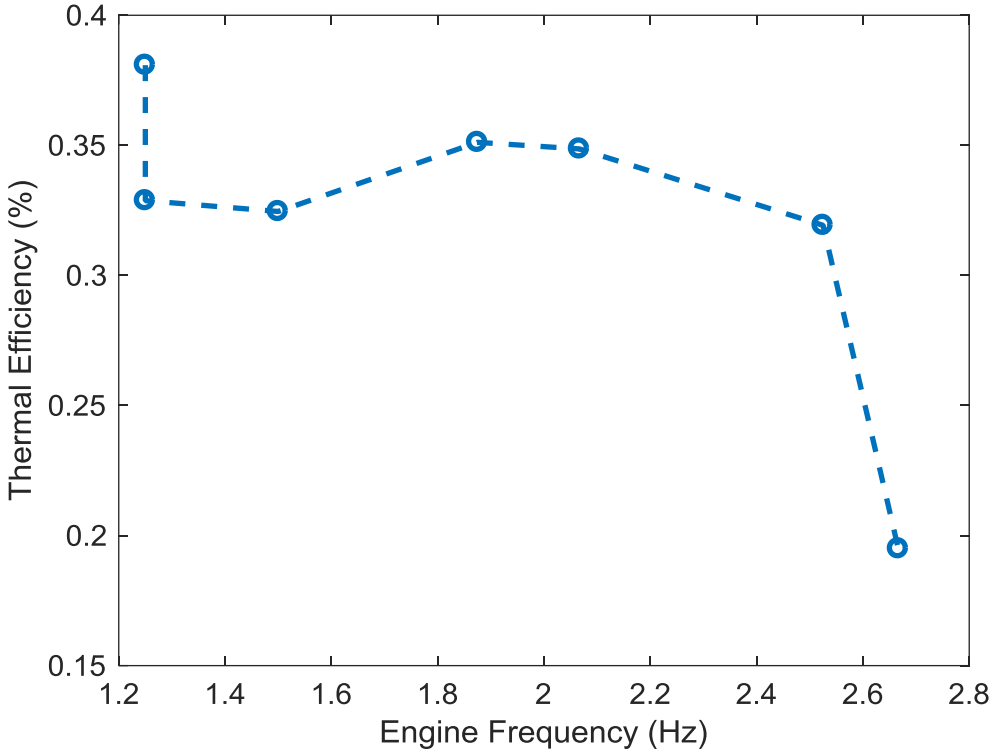


Figure 5.18: Measured Thermal Efficiency as a Function of Engine Frequency for a Heating Cap Temperature of 300 °C and a Mean Pressure of 430 kPa

The remaining plots in this section will compare experimental results to those found using the final version of the second order model. The model used the ideal adiabatic reference cycle and the decoupled losses were calculated and incorporated as described in Chapter 3, except for the following modifications:

1. The heater and cooler gas temperatures were used in the walls of the regenerator cavity conduction loss calculation, rather than the thermal source and sink temperatures. The resulting conduction loss was still an overestimate compared to what was measured, but the error was reduced.
2. The empirical model for the crankcase hysteresis loss was implemented.
3. The measured heat loss through the insulation was included as a constant added to the reference cycle heat input. 135 W was added for a heating cap temperature of 300 °C, and 90 W was added for a heating cap temperature of 200 °C.

Operating conditions from the experiments were input to the model in the following way.

1. The average pressure measured at the power cylinder outboard pressure transducer was taken to be the mean cycle pressure in the model.
2. The measured engine frequency was taken to be the frequency in the model.
3. The measured gas temperature at the regenerator hot thermocouple location was taken to be the heater wall temperature in the model.
4. The average of the measured gas temperatures at the regenerator cold left and the regenerator cold right locations was taken to be the cooler wall temperature in the model.
5. The measured gas temperature at the expansion space thermocouple location was taken as the expansion space temperature in the model.
6. The measured gas temperature at the displacer mount thermocouple location was taken to be the compression space gas temperature in the model.
7. The average of the measured gas temperatures at the expansion space and regenerator hot thermocouple locations was taken to be the heater gas temperature in the model.
8. The measured gas temperatures at the regenerator cold left and regenerator cold right thermocouple locations were averaged, and the result was averaged with the measured gas temperature at the displacer mount thermocouple location to calculate the cooler gas temperature for input to the model.

In Figure 5.19, the measured heat input rate has been plotted with corresponding results from the second order model as a function of engine frequency. The red stars represent experimental data and the blue circles represent model results. The plot was produced using the same data set

as for Figure 5.18. The model overestimates the heat input rate for all conditions tested. This is partly due to the overestimation of the conduction loss highlighted in a previous section, and may also be the result of losses neglected in the model. There is no clear trend with frequency in either the data or the model, and the error in the model is approximately constant for the conditions tested.

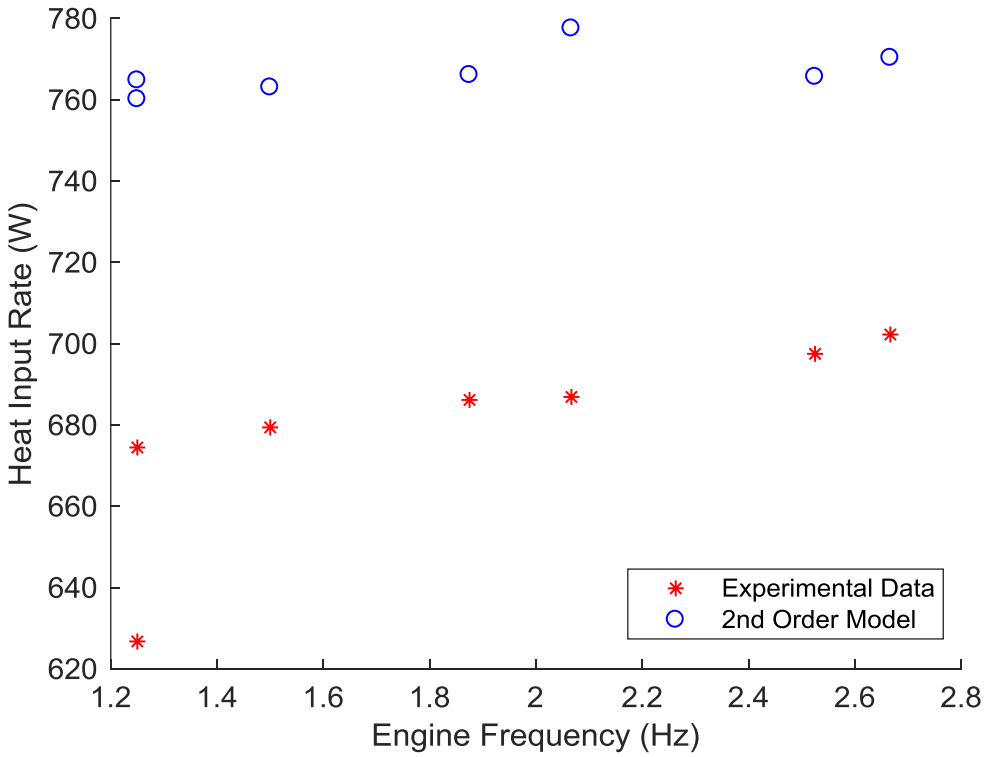


Figure 5.19: Measured Heat Input Rate with Overlaid Second Order Model for a Heating Cap Temperature of 300 °C and a Mean Pressure of 430 kPa

In Figure 5.20, the measured thermal efficiency has been overlaid by corresponding second order model results. The data and the model follow a similar trend, both indicating an optimum speed of approximately 2.1 Hz, where the thermal efficiency is maximized. The second order model over-predicts the measured thermal efficiency for the conditions tested. Since it has already been shown in Figure 5.19 that the model overestimates the heat input, this means that the power output must also be overstated by the second order model.

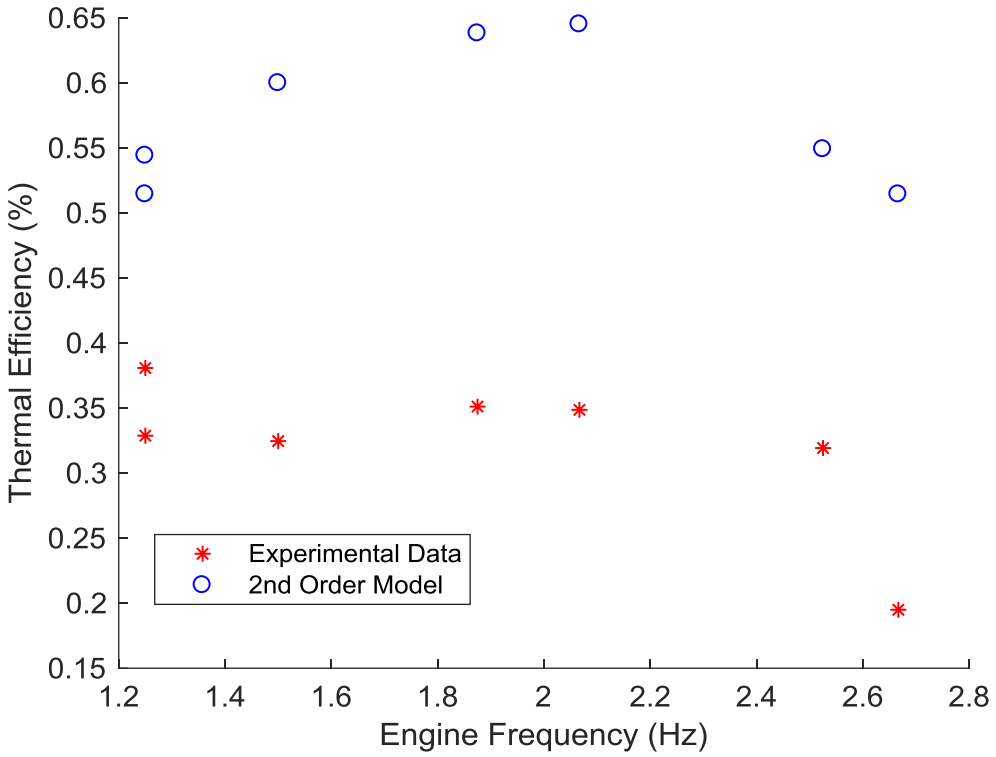


Figure 5.20: Measured Thermal Efficiency with Overlaid Second Order Model for a Heating Cap Temperature of 300 °C and a Mean Pressure of 430 kPa

Figure 5.21 plots measured shaft power overlaid with corresponding results from the second order model. Data for the plot was recorded with the heating cap temperature at 200 °C. Data and model follow similar trends and indicate similar maximum power frequencies. The model overestimates the experimental results under all conditions tested. Causes of shaft power overestimation include neglect of losses such as seal leakage, heat transfer hysteresis, and flow friction in the connecting pipe and power cylinder head, as well as inaccurate calculation of losses such as mechanical friction and flow friction, due to unrealistic reference cycle results and simplifying assumptions.

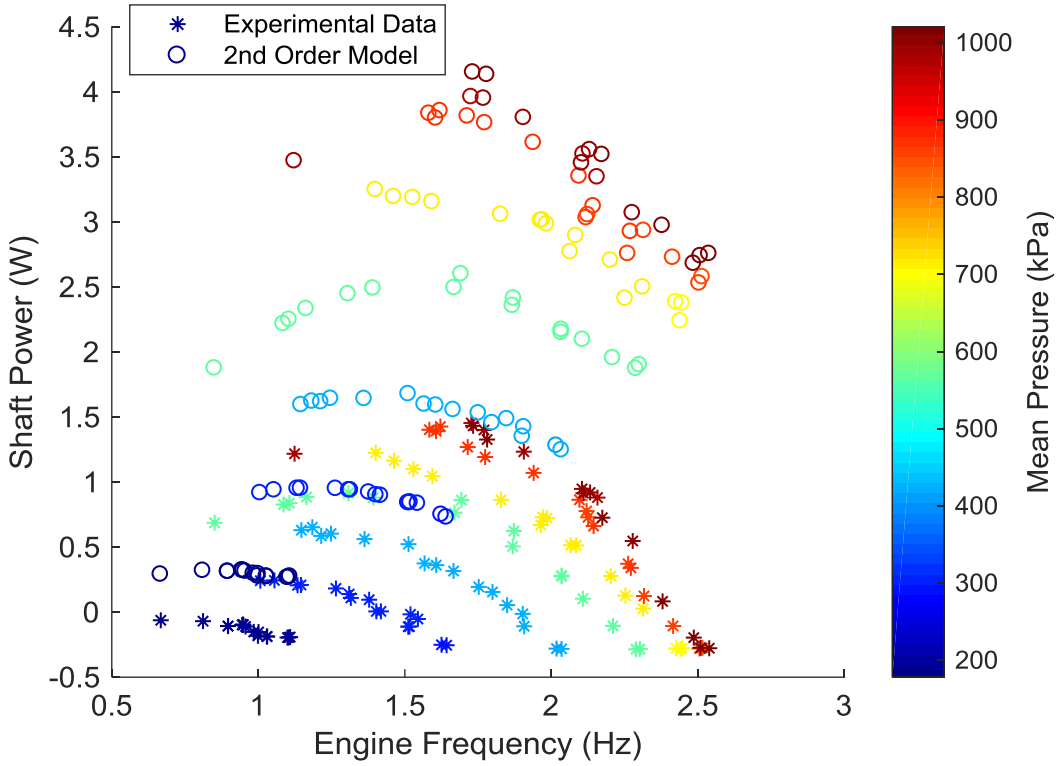


Figure 5.21: Measured Shaft Power as a Function of Engine Frequency and Mean Pressure with Overlaid Second Order Model. Data was collected at a heating cap temperature of 200 °C.

The measured total heat rejection rate is compared with results from the second order model in Figure 5.22. The plot shows that the total heat rejection rate is overestimated by the model for all conditions tested. The trends followed differ between the data and the model. The experimental data shows a clear increase in heat rejection rate with increasing mean pressure. Mean pressure dependence is not exhibited by the second order model results. A pressure dependent loss mechanism may be missing from the heat rejection rate calculation, or the pressure dependence in one of the included loss mechanisms may be underrepresented due to the assumptions made. The experimental data also shows a decrease in the heat rejection rate above a threshold engine frequency which depends on the mean pressure. This is indicative of a change in flow phenomena on the gas side of the heat exchanger. The model uses Reynolds analogy with a turbulent steady flow friction factor correlation for all engine frequencies; hence, changes in flow regime are not represented.

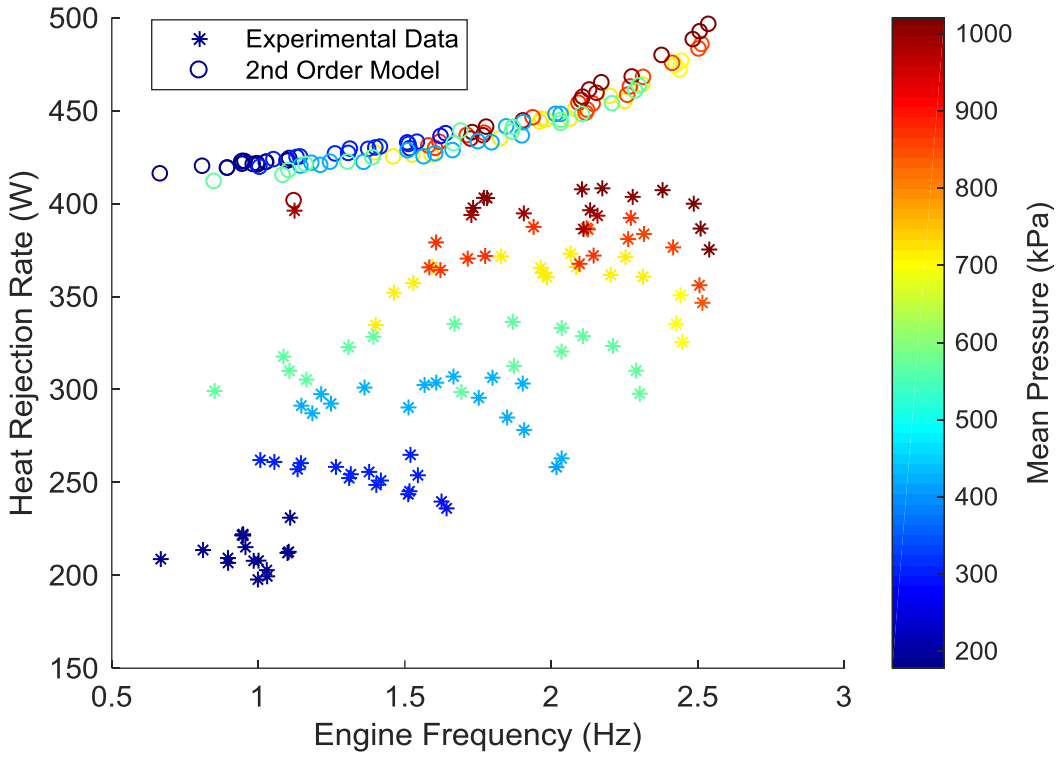


Figure 5.22: Measured Heat Rejection Rate as a Function of Engine Frequency and Mean Pressure with Overlaid Second Order Model. Data was collected at a heating cap temperature of 200 °C.

5.12 Conclusions

Experimental results have been compared to model predictions to assess agreement. Effort was focused on evaluating individual components of the model, in addition to overall results. Significant discrepancies between measured and modeled results were discovered in several areas. Of these, indicator diagram prediction was the most influential, since most other calculations are based on its results. Indicator diagram prediction accuracy was observed to decline with thermal source temperature, due to the greater influence of neglected decoupled losses at lower temperature difference operating points.

To further improve the low thermal source temperature performance of the engine, it is recommended that the heater head be replaced with something similar to the cooler. The aims of the modification would be to reduce temperature drop from the thermal source to the heater gas, and decrease conduction loss and heat loss through the insulation. Resulting changes in dead volume and flow geometry will have unknown effects.

Chapter 6. Conclusions and Future Work

This research was conducted with the intent of addressing two main issues. The first was the applicability of commonly used second order models to Stirling engines with thermal source temperatures less than 150 °C. The second was the identification of modifications necessary to improve the low thermal source temperature performance of the ST05G-CNC Stirling engine. The problems were investigated experimentally.

It was shown that the assumption of decoupled losses, fundamental to the second order modeling approach, breaks down at low thermal source temperatures. Under these conditions, both the ideal adiabatic and the ideal isothermal reference cycles significantly overestimate the measured indicated work of the cycle. The discrepancy was attributed to the absence of losses in the reference cycle calculation. The influence of these losses on the reference cycle results becomes greater as the thermal source temperature declines. Since nearly all the decoupled losses rely on the reference cycle for their input parameters, inaccuracies are far-reaching. Further model validation awaits improvement of the reference cycle simulation.

Experiments proved that the minimum thermal source temperature of the engine could be reduced by reducing the piston diameter and increasing the crankcase volume. These modifications were shown to reduce the amount of forced work and crankcase gas spring hysteresis, leading to a performance improvement at low thermal source temperatures. The measured shaft power was unaffected by the addition of dead volume reduction components in the connecting pipe, displacer mount, and power cylinder head. A suggested further modification is to replace the heater head with something resembling the cooler, with the aim of reducing temperature drop between the thermal source and the heater gas, the conduction loss, and the heat lost through the insulation. The significance of these loss mechanisms was shown using experimental results. The change would also facilitate a liquid thermal source.

Future work on these topics could include:

1. Measurements of wall temperatures, and faster response gas temperatures inside the engine for characterization of convective heat transfer processes.
2. Experiments and modeling to study the effects of seal leakage, heat transfer hysteresis, and changes in thermal sink temperature.
3. Optimization of the regenerator properties for low thermal source temperatures.

References

[1] B. Inc, “Waste Heat Recovery: Technology Opportunities in the US Industry,” 2008.

[2] K. Wang, S. R. Sanders, S. Dubey, F. H. Choo, and F. Duan, “Stirling cycle engines for recovering low and moderate temperature heat: A review,” *Renew. Sustain. Energy Rev.*, vol. 62, no. May, pp. 89–108, 2016.

[3] C. M. Hargreaves, *The Philips Stirling Engine*. Amsterdam, The Netherlands: Elsevier Science Publishers B.V., 1991.

[4] F. Vélez, J. J. Segovia, M. C. Martín, G. Antolín, F. Chejne, and A. Quijano, “A technical, economical and market review of organic Rankine cycles for the conversion of low-grade heat for power generation,” *Renew. Sustain. Energy Rev.*, vol. 16, no. 6, pp. 4175–4189, 2012.

[5] C. B. Vining, “An inconvenient truth about thermoelectrics TL - 8,” *Nat. Mater.*, vol. 8 VN-re, no. 2, pp. 83–85, 2009.

[6] C. D. West, *Principles and Applications of Stirling Engines*. New York: Van Nostrand Reinhold Company Inc., 1986.

[7] A. J. Organ, *Thermodynamics and Gas Dynamics of the Stirling Cycle Machine*. Cambridge: Cambridge University Press, 1992.

[8] W. R. Martini, *Stirling Engine Design Manual*. Washington D.C.: NASA Lewis Research Center, 1983.

[9] N. C. J. Chen and F. P. Griffin, “A Review of Stirling Engine Mathematical Models,” pp. 1–37, 1983.

[10] I. Urieli and B. D. M., *Stirling Cycle Engine Analysis*. Bristol: Adam Hilger Ltd, 1984.

[11] A. J. Organ, “Intimate thermodynamic design of the Stirling engine gas circuit without the computer,” *J. Mech. Eng. Sci.*, vol. 205, 1991.

[12] A. J. Organ, *Stirling Cycle Engines: Inner Workings and Design*. West Sussex, UK: John Wiley and Sons, Ltd, 2014.

[13] J.-I. Prieto and R. Diaz, "Isothermal simulation with decoupled losses for kinematic Stirling engine design," *JSME Int. Journal;*, vol. 36, no. 4, pp. 153–158, 1993.

[14] G. Walker, "Elementary Design Guidelines for Stirling Engines," in *14th IECEC*, 1979, pp. 1066–1068.

[15] G. Schmidt, "The Theory of Lehmann's Calorimetric Machine," *Z. Ver. Dtsch. Ing.*, vol. 15, p. Part 1.

[16] M. T. Mabrouk, A. Kheiri, and M. Feidt, "Displacer gap losses in beta and gamma Stirling engines," *Energy*, vol. 72, pp. 135–144, 2014.

[17] J. Sauer and H. D. Kuehl, "Numerical model for Stirling cycle machines including a differential simulation of the appendix gap," *Appl. Therm. Eng.*, vol. 111, pp. 819–833, 2017.

[18] S. Petrescu, M. Costea, C. Harman, and T. Florea, "Application of the Direct Method to irreversible Stirling cycles with finite speed," *Int. J. Energy Res.*, vol. 26, no. 7, pp. 589–609, 2002.

[19] J. R. Senft, "Mechanical efficiency of kinematic heat engines," *J. Franklin Inst.*, vol. 324, no. 2, pp. 273–290, 1987.

[20] J. R. Senft, "A General Formula For The Work Output Of Reciprocating Heat Engines," *Interscociety Energy Convers. Eng. Conf.*, pp. 179–184, 1990.

[21] J. R. Senft, "Pressurization effects in kinematic heat engines," *J. Franklin Inst.*, vol. 328, no. 2–3, pp. 255–279, 1991.

[22] J. R. Senft, "General Analysis of Reciprocating," *J. Franklin Inst.*, vol. 330, no. 5, pp. 967–984, 1993.

[23] J. R. Senft, "Brake Power Maxima of Engines With Limited Heat Transfer," *Interscociety Energy Convers. Eng. Conf.*, pp. 973–978, 1997.

[24] J. R. Senft, “Theoretical limits on the performance of Stirling engines,” *Int. J. Energy Res.*, vol. 22, no. May, pp. 991–1000, 1998.

[25] J. R. Senft, “Extended mechanical efficiency theorems for engines and heat pumps,” *Int. J. Energy Res.*, vol. 24, no. 8, pp. 679–693, 2000.

[26] J. R. Senft, “Optimum Stirling engine geometry,” *Int. J. Energy Res.*, vol. 26, no. 12, pp. 1087–1101, 2002.

[27] J. R. Senft, *Mechanical Efficiency of Heat Engines*. New York: Cambridge University Press, 2007.

[28] A. Der Minassians and S. R. Sanders, “Stirling Engines for Distributed Low-Cost Solar-Thermal-Electric Power Generation,” *J. Sol. Energy Eng.*, vol. 133, no. 1, p. 11015, 2011.

[29] M. Costea, S. Petrescu, and C. Harman, “Effect of irreversibilities on solar Stirling engine cycle performance,” *Energy Convers. Manag.*, vol. 40, no. 15, pp. 1723–1731, 1999.

[30] I. Urieli, “Stirling Cycle Machine Analysis.” [Online]. Available: https://www.ohio.edu/mechanical/stirling/index.html. [Accessed: 14-Jun-2017].

[31] W. M. Kays and A. L. London, *Compact Heat Exchangers*. New York: McGraw-Hill, 1955.

[32] B. Kongtragool and S. Wongwises, “Thermodynamic analysis of a Stirling engine including dead volumes of hot space, cold space and regenerator,” *Renew. Energy*, vol. 31, no. 3, pp. 345–359, 2006.

[33] F. Formosa and G. Despesse, “Analytical model for Stirling cycle machine design,” *Energy Convers. Manag.*, vol. 51, pp. 1855–1863, 2010.

[34] M. H. Ahmadi, A. H. Mohammadi, and S. M. Pourkiaei, “Optimisation of the thermodynamic performance of the Stirling engine,” *Int. J. Ambient Energy*, vol. 37, no. 2, pp. 149–161, 2016.

[35] J. Bert, D. Chrenko, T. Sophy, L. Le Moyne, and F. Sirot, “Zero dimensional finite-time thermodynamic, three zones numerical model of a generic Stirling and its experimental

validation," *Renew. Energy*, vol. 47, pp. 167–174, 2012.

[36] Z. M. Ding, L. G. Chen, and F. R. Sun, "Performance optimization of a linear phenomenological law system Stirling engine," *J. Energy Inst.*, vol. 88, no. 1, pp. 36–42, 2015.

[37] V. Punnathanam and P. Kotecha, "Effective multi-objective optimization of Stirling engine systems," *Appl. Therm. Eng.*, vol. 108, pp. 261–276, 2016.

[38] S. Fan, M. Li, S. Li, T. Zhou, Y. Hu, and S. Wu, "Thermodynamic analysis and optimization of a Stirling cycle for lunar surface nuclear power system," *Appl. Therm. Eng.*, vol. 111, pp. 60–67, 2017.

[39] R. Li, L. Grosu, and D. Queiros-Condé, "Losses effect on the performance of a Gamma type Stirling engine," *Energy Convers. Manag.*, vol. 114, pp. 28–37, 2016.

[40] M. H. Ahmadi, M. A. Ahmadi, F. Pourfayaz, M. Bidi, H. Hosseinzade, and M. Feidt, "Optimization of powered Stirling heat engine with finite speed thermodynamics," *Energy Convers. Manag.*, vol. 108, pp. 96–105, 2016.

[41] J. M. Strauss and R. T. Dobson, "Evaluation of a second order simulation for Sterling engine design and optimisation," *Energy South. Africa*, vol. 21, no. 2, pp. 17–29, 2010.

[42] E. Cardozo, C. Erlich, A. Malmquist, and L. Alejo, "Integration of a wood pellet burner and a Stirling engine to produce residential heat and power," *Appl. Therm. Eng.*, vol. 73, no. 1, pp. 669–678, 2014.

[43] J. A. Araoz, M. Salomon, L. Alejo, and T. H. Fransson, "Non-ideal Stirling engine thermodynamic model suitable for the integration into overall energy systems," *Appl. Therm. Eng.*, vol. 73, no. 1, pp. 203–219, 2014.

[44] J. A. Araoz, M. Salomon, L. Alejo, and T. H. Fransson, "Numerical simulation for the design analysis of kinematic Stirling engines," *Appl. Energy*, vol. 159, pp. 633–650, 2015.

[45] J. Araoz, E. Cardozo, M. Salomon, L. Alejo, and T. Fransson, "Development and validation of a thermodynamic model for the perfomance analysis of a gamma Stirling

engine prototype," *Appl. Therm. Eng.*, vol. submitted, pp. 16–30, 2015.

[46] B. Thomas and D. Pittman, "Update on the evaluation of different correlations for the flow friction factor and heat transfer of Stirling engine regenerators," in *Collection of Technical Papers. 35th Intersociety Energy Conversion Engineering Conference and Exhibit (IECEC)*, 2000, p. 76e84.

[47] M. Babaelahi and H. Sayyaadi, "Simple-II: A new numerical thermal model for predicting thermal performance of Stirling engines," *Energy*, vol. 69, pp. 873–890, 2014.

[48] M. Babaelahi and H. Sayyaadi, "A new thermal model based on polytropic numerical simulation of Stirling engines," *Appl. Energy*, vol. 141, pp. 143–159, 2015.

[49] H. Hosseinzade and H. Sayyaadi, "CAFS: The Combined Adiabatic-Finite Speed thermal model for simulation and optimization of Stirling engines," *Energy Convers. Manag.*, vol. 91, pp. 32–53, 2015.

[50] C. J. Paul and A. Engeda, "Modeling a complete Stirling engine," *Energy*, vol. 80, pp. 85–97, 2015.

[51] C. J. Paul, "A Stirling engine for use with lower quality fuels," Michigan State University, 2014.

[52] M. Ni, B. Shi, G. Xiao, H. Peng, U. Sultan, S. Wang, Z. Luo, and K. Cen, "Improved Simple Analytical Model and experimental study of a 100W β-type Stirling engine," *Appl. Energy*, vol. 169, pp. 768–787, 2016.

[53] D. Viebach, "The ST05G Stirling Engine Project." [Online]. Available: http://ve-ingenieure.de/projekt_st05g_cnc_engl.html. [Accessed: 11-Dec-2016].

[54] R. Gheith, F. Aloui, M. Tazerout, and S. Ben Nasrallah, "Experimental investigations of a gamma Stirling engine," *Int. J. energy Res.*, vol. 31, no. June 2011, pp. 1175–1182, 2011.

[55] R. Gheith, F. Aloui, and S. Ben Nasrallah, "Study of the regenerator constituting material influence on a gamma type Stirling engine," *J. Mech. Sci. Technol.*, vol. 26, no. 4, pp. 1251–1255, 2012.

[56] R. Gheith, F. Aloui, and S. Ben Nasrallah, “Determination of adequate regenerator for a Gamma-type Stirling engine,” *Appl. Energy*, vol. 139, pp. 272–280, 2015.

[57] R. Gheith, H. Hachem, F. Aloui, and S. Ben Nasrallah, “Experimental and theoretical investigation of Stirling engine heater: Parametrical optimization,” *Energy Convers. Manag.*, vol. 105, pp. 285–293, 2015.

[58] H. Hachem, M. Creyx, R. Gheith, E. Delacourt, C. Morin, F. Aloui, and S. Ben Nasrallah, “Comparison based on exergetic analyses of two hot air engines: A Gamma type Stirling engine and an open joule cycle Ericsson engine,” *Entropy*, vol. 17, no. 11, pp. 7331–7348, 2015.

[59] H. Hachem, R. Gheith, F. Aloui, and S. Ben Nasrallah, “Numerical characterization of a γ-Stirling engine considering losses and interaction between functioning parameters,” *Energy Convers. Manag.*, vol. 96, pp. 532–543, 2015.

[60] J. Bert, D. Chrenko, T. Sophy, L. Le Moyne, and F. Sirot, “Simulation, experimental validation and kinematic optimization of a Stirling engine using air and helium,” *Energy*, vol. 78, pp. 701–712, 2014.

[61] M. Hooshang, R. Askari Moghadam, S. Alizadeh Nia, and M. T. Masouleh, “Optimization of Stirling engine design parameters using neural networks,” *Renew. Energy*, vol. 74, pp. 855–866, 2015.

[62] M. Hooshang, R. Askari Moghadam, and S. AlizadehNia, “Dynamic response simulation and experiment for gamma-type Stirling engine,” *Renew. Energy*, vol. 86, pp. 192–205, 2016.

[63] S. Alfarawi, R. AL-Dadah, and S. Mahmoud, “Influence of phase angle and dead volume on gamma-type Stirling engine power using CFD simulation,” *Energy Convers. Manag.*, vol. 124, pp. 130–140, 2016.

[64] S. Alfarawi, R. Al-Dadah, and S. Mahmoud, “Enhanced thermodynamic modelling of a gamma-type Stirling engine,” *Appl. Therm. Eng.*, vol. 106, pp. 1380–1390, 2016.

[65] S. Alfarawi, R. AL-Dadah, and S. Mahmoud, “Potentiality of new miniature-channels

Stirling regenerator," *Energy Convers. Manag.*, vol. 133, pp. 264–274, 2017.

[66] W. M. Kays and A. L. London, *Compact Heat Exchangers*. Krieger Pub Co, 1998.

[67] D. R. Gedeon and J. G. Wood, "Oscillating-flow regenerator test rig: hardware and theory with derived correlations for screens and felts," 1996.

[68] I. Kolin, *Stirling motor: history-theory-practice*. Zagreb University Publications, 1991.

[69] J. R. Senft, *Ringbom Stirling Engines*. USA: Oxford University Press, 1993.

[70] C. Çinar, F. Aksoy, and D. Erol, "The effect of displacer material on the performance of a low temperature differential Stirling engine," *Int. J. Energy Res.*, vol. 36, no. 8, pp. 911–917, 2012.

[71] Y. Kato, "Indicated diagrams of a low temperature differential Stirling engine using flat plates as heat exchangers," *Renew. Energy*, vol. 85, pp. 973–980, 2016.

[72] N. Boutammachte and J. Knorr, "Field-test of a solar low delta-T Stirling engine," *Sol. Energy*, vol. 86, no. 6, pp. 1849–1856, 2012.

[73] S. Iwamoto, F. Toda, K. Hirata, M. Takeuchi, and T. Yamamoto, "Comparison of Low- and High Temperature Differential Stirling Engines," *Proc. 8th Int. Stirling Engine Conf.*, pp. 29–38, 1997.

[74] B. Kongtragool and S. Wongwises, "Performance of low-temperature differential Stirling engines," *Renew. Energy*, vol. 32, no. 4, pp. 547–566, 2007.

[75] A. Der Minassians and S. Sanders, "A Magnetically-Actuated Resonant-Displacer Free-Piston Stirling Machine," *5th Int. Energy Convers. Eng. Conf. Exhib.*, no. June, pp. 25–27, 2007.

[76] A. Der Minassians and S. R. Sanders, "Multiphase Free-Piston Stirling Engine for Solar-Thermal-Electric Power Generation Applications," in *5th International Energy Conversion Engineering Conference and Exhibit (IECEC)*, 2007, no. June, pp. 25–27.

[77] A. Der Minassians, "Stirling Engines for Low-Temperature Solar-Thermal-Electric Power

Generation," University of California, Berkeley, 2007.

[78] M. He, S. Sanders, and C. Berkeley, "Design of a 2.5 kW Low Temperature Stirling Engine for Distributed Solar Thermal Generation," in *American Institute of Aeronautics and Astronautics*, 2011, no. August, pp. 1–8.

[79] A. Asnaghi, S. M. Ladjevardi, P. Saleh Izadkhast, and A. H. Kashani, "Thermodynamics Performance Analysis of Solar Stirling Engines," *ISRN Renew. Energy*, vol. 2012, pp. 1–14, 2012.

[80] M. He, N. Beutler, D. Loeder, S. Sanders, and E. Engineering, "Testing of 2 . 5kW Low Temperature Stirling Engine for Distributed Solar Thermal Generation," *IEEE Energytech*, pp. 1–6, 2012.

[81] A. Madduri, D. Loeder, N. Beutler, M. He, and S. Sanders, "Concentrated evacuated tubes for solar-thermal energy generation using stirling engine," *2012 IEEE Energytech, Energytech 2012*, no. Equation 1, pp. 1–6, 2012.

[82] F. Formosa, A. Badel, and J. Lottin, "Equivalent electrical network model approach applied to a double acting low temperature differential Stirling engine," *Energy Convers. Manag.*, vol. 78, pp. 753–764, 2014.

[83] B. Kongtragool and S. Wongwises, "Investigation on power output of the gamma-configuration low temperature differential Stirling engines," *Renew. Energy*, vol. 30, no. 3, pp. 465–476, 2005.

[84] B. Hoegel, "Thermodynamics Based Design of Stirling Engines for Low-Temperature Heat Sources," University of Canterbury, 2014.

[85] B. Hoegel, D. Pons, M. Gschwendtner, A. Tucker, and M. Sellier, "Thermodynamic peculiarities of alpha-type Stirling engines for low-temperature difference power generation: Optimisation of operating parameters and heat exchangers using a third-order model," *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.*, vol. 228, no. 11, pp. 1936–1947, 2014.

[86] H. Yang, D. B. Sims-Williams, and L. He, "Unsteady Pressure Measurement With

Correction on Tubing Distortion," *Unsteady Aerodyn. Aeroacoustics Aeroelasticity Turbomachines*, pp. 521–529, 2006.

[87] A. J. Wheeler and A. R. Ganji, *Introduction to Engineering Experimentation*, Third Edit. New Jersey: Pearson Higher Education, 2004.

[88] A. A. S. M. Inc., "AISI Type 316 Stainless Steel, annealed sheet." [Online]. Available: http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=mq316a. [Accessed: 28-Nov-2017].

[89] W. L. Cleghorn and N. Dechev, *Mechanics of Machines*, 2nd ed. New York: Oxford University Press, 2015.

[90] M. H. Ahmadi, M.-A. Ahmadi, and F. Pourfayaz, "Thermal models for analysis of performance of Stirling engine: A review," *Renew. Sustain. Energy Rev.*, vol. 68, no. February 2016, pp. 168–184, 2017.

[91] F. L. Curzon and B. Ahlborn, "Efficiency of a Carnot engine at maximum power output," *Am. J. Phys.*, vol. 43, no. 1, pp. 22–24, 1975.

Appendix A: Analysis of Experimental Uncertainty

Introduction

The following uncertainty analysis follows the procedure documented by Wheeler and Ganji [1]. The propagation of uncertainty may be quantified using the root sum of squares error given by

$$w_Z = \left(\sum_{i=1}^{n} \left[w_{x_i} \frac{\partial Z}{\partial x_i} \right]^2 \right)^{\frac{1}{2}} \tag{A.1}$$

where Z is the result calculated from n measured variables, x_i, and w is the uncertainty associated with the variable indicated by the subscript.

If Z is calculated by multiplying measured variables only, i.e.

$$Z = C {x_1}^a {x_2}^b \ldots {x_n}^N \tag{A.2}$$

then Eq. (A.1) reduces to

$$\frac{w_Z}{Z} = \left\{ \left(a \frac{w_1}{x_1} \right)^2 + \left(b \frac{w_2}{x_2} \right)^2 + \cdots + \left(N \frac{w_n}{x_n} \right)^2 \right\}^{\frac{1}{2}} \tag{A.3}$$

The following sections give the error propagation equation for each calculated result. They also list potential error sources and their potential contributions to the uncertainty for each type of base measurement. Data is presented in tables for simplicity.

Temperature Drop

Definition of Functional Relationship

Temperature drops are calculated using equations of the form

$$\Delta T = T_1 - T_2 \tag{A.4}$$

where T_1 and T_2 are the measured working fluid or coolant temperatures.

Estimation of Uncertainties in Measured Variables

Table A.1 Estimated Uncertainties for Thermocouple Measurements

Sources of Error	Error Type	Uncertainty Contribution (°C)	Comments
standard deviation/noise	random	0.03	Calculated using calibration data
spatial variation	systematic	-	Most significant for heating cap thermocouple.
conduction	systematic	-	Only significant for heater head thermocouples. More significant at higher heating cap temperatures.
quantization	random	-	Mitigated by taking the average at steady state.
time constant	systematic	-	Mitigated by taking the average at steady state.
radiation	systematic	-	Only significant for heater head thermocouples.
differences between thermocouples	systematic	-	Mitigated by applying calibration terms to the raw data
Standard Accuracy from Manufacturer [2]	random	4	Greater of $\pm$ 2.2 °C or $\pm$ 0.75 % [2]

Total Uncertainty	5 °C (10 °C for heating cap temperature due to increased spatial variation)

Table A.2 Estimated Uncertainties for RTD Measurements

Sources of Error	**Error Type**	**Uncertainty Contribution (°C)**	**Comments**
standard deviation/noise	random	0.03	Calculated using calibration data
spatial variation	systematic	-	Not significant since cross-sectional area of stream is small
conduction	systematic	-	Not significant since water is near ambient temperature
quantization	random	-	Mitigated by taking the average at steady state.
time constant	systematic	-	Mitigated by taking the average at steady state.
radiation	systematic	-	Not significant since water is near ambient temperature
differences between RTDs	systematic	-	Mitigated by applying calibration terms to the raw data
Accuracy stated by manufacturer [3]	random	0.8	Class B Accuracy [3]
Total Uncertainty		1 °C	

Propagation of Uncertainty

Applying Eq. (A.1) to Eq. (A.4) yields

$$w_{\Delta T} = \left\{\left(w_{T_1}\right)^2 + \left(-w_{T_2}\right)^2\right\}^{\frac{1}{2}} \tag{B.5}$$

Since the same instruments and procedures are used to measure temperatures T_1 and T_2, it will be assumed that they have the same uncertainty, i.e.

$$w_{T_1} = w_{T_2} = w_T$$

This allows simplification of Eq. (A.5)

$$w_{\Delta T} = \{(w_T)^2 + (-w_T)^2\}^{\frac{1}{2}} = \{2(w_T)^2\}^{\frac{1}{2}} = \sqrt{2}w_T \tag{A.6}$$

Table A.3: Inputs and Results for Uncertainty Propagation in Temperature Difference

w_T (gas temperatures)	0.01 °C
w_T (water temperatures)	0.01 °C
Propagated Uncertainty, $w_{\Delta T}$	0.014 °C

Pressure

The pressure measurements are not used for further calculations, so no uncertainty propagation calculations are shown in this section.

Estimation of Uncertainties in Measured Variables

Table A.4 Estimated Uncertainties for Validyne Pressure Measurements

Sources of Error	Error Type	Uncertainty Contribution (kPa)	Comments
noise	random	-	Mitigated by taking the average at steady state.
spatial variation	systematic	-	Assumed to be negligible since pressure drops are small in regions characterized by a single pressure measurement.
connection tube losses	systematic	-	Mitigated by taking the average at steady state.
calibration machine	random	0.07	0.07 kPa = 0.01 PSI, which is the resolution of the machine display.
quantization	random	-	Mitigated by taking the average at steady state.
time constant	systematic	-	Mitigated by taking the average at steady state.
Accuracy stated by manufacturer [4]	random	$\pm$ 0.25 % Full Scale = 2.5 kPa for absolute = 0.0862 kPa for differential	Calculated using 10 bar as full scale value for absolute transducers and 5 psi for differential transducer

Total Uncertainty	5 kPa for absolute measurement, 0.01 kPa for differential measurement

Table A.5: Estimated Uncertainties for PCB Pressure Measurements

Sources of Error	**Error Type**	**Uncertainty Contribution (kPa)**	**Comments**
standard deviation/noise	random	-	Assumed negligible.
spatial variation	systematic	-	Assumed to be negligible since pressure drops are small in regions characterized by a single pressure measurement.
leak down time	random	-	Mitigated by taking measurements with engine at steady state.
quantization	random	-	Assumed negligible.
mean pressure determined from Validyne transducers	random	5	
Resolution stated by the manufacturer [5]	random	$\pm$ 0.007	
Total Uncertainty		5.01 kPa	

Engine Frequency

Definition of Functional Relationship

$$f = \frac{\Delta\theta}{t} \tag{A.7}$$

Estimation of Uncertainties in Measured Variables

Table A.6 Estimated Uncertainties for Crankshaft Position Measurement

Sources of Error	Error Type	Uncertainty Contribution (°)	Comments
noise	random	-	not significant since only counting pulses of 5 V.
belt stretch/alignment	random	± 5	Alignment is done by looking at the crankshaft with thecrankcase cover removed.
quantization	random	-	not significant since only counting pulses of 5 V.
Total Uncertainty		5 °	

Table A.7 Estimated Uncertainties for Time Measurement

Sources of Error	Error Type	Uncertainty Contribution (s)	Comments
electrical/software lag	systematic	0.001	
Total Uncertainty		0.001 s	

Propagation of Uncertainty

$$w_\omega = \left\{\left(w_{\Delta\theta}\frac{\partial f}{\partial \Delta\theta}\right)^2 + \left(w_t\frac{\partial f}{\partial t}\right)^2\right\}^{\frac{1}{2}} = \left\{\left(w_{\Delta\theta}\frac{1}{t}\right)^2 + \left(-w_t\frac{1}{t^2}\right)^2\right\}^{\frac{1}{2}} \quad \text{(A.8)}$$

Table A.8: Inputs and Results for Uncertainty Propagation in Angular Frequency

$w_{\Delta\theta}$ (uncertainty in crank angle increment)	0.0873 rad (5 °)
w_t (uncertainty in time)	0.001 s
t (maximum measured time)	10 s
Propagated Uncertainty, w_ω	0.0087 rad/s

Power Measurement

Definition of Functional Relationship

$$\dot{W} = \tau\omega \tag{A.9}$$

Estimation of Uncertainties in Measured Variables

Table A.9 Estimated Uncertainties for Torque Measurement

Sources of Error	Error Type	Uncertainty Contribution (Nm)	Comments
quantization	random		Mitigated by taking the average at steady state.
response time	systematic		Mitigated by taking the average at steady state.
Inconsistent speed during power measurement	random		Inertia of the engine and output shaft can affect torque measurement during changes in speed
Manufacturer Data Sheet [6]	random	0.05 Nm	Nonlinearity $\pm$ 0.2% of RO Hysteresis $\pm$ 0.1% of RO Nonrepeatability $\pm$ 0.2% of RO [6] added together to get full uncertainty
Total Uncertainty		0.05 Nm	

Propagation of Uncertainty

$$w_{\dot{W}} = \dot{W}\left\{\left(\frac{w_\tau}{\tau}\right)^2 + \left(\frac{w_\omega}{\omega}\right)^2\right\}^{\frac{1}{2}} \tag{A.10}$$

Table A.10: Inputs and Results for Uncertainty Propagation in Shaft Power

w_τ (uncertainty in torque)	0.05 Nm
τ (maximum measured torque)	2 Nm
w_ω (uncertainty in angular frequency)	0.0087 rad/s
ω (maximum measured angular frequency)	31.4 rad/s (10 Hz)
Propagated Uncertainty, $w_{\dot{W}}$	0.25 W

Heat Input Rate

Definition of Functional Relationship

$$\dot{Q}_{in} = I^2 R \left(\frac{time\ on}{total\ time} \right) = I^2 R t_{frac} \tag{A.11}$$

Estimation of Uncertainties in Measured Variables

Table A.11 Estimated Uncertainties for Current Measurements

Sources of Error	Error Type	Uncertainty Contribution (A)	Comments
Manufacturer Specified Accuracy [7]	random	0.33	2 %
Total Uncertainty		0.4 A	

Table A.12 Estimated Uncertainties for Resistance Measurements

Sources of Error	Error Type	Uncertainty Contribution (Ω)	Comments
temperature dependent resistance	systematic	-	Resistance depends on temperature. Heater wire material is unknown and high temperature measurements were deemed too dangerous.
Manufacturer data sheet [8]	random	0.058	0.05 Ohms + 8 digits [8]
Total Uncertainty		0.1 Ω	

Propagation of Uncertainty

$$w_{t_{frac}} = \left\{ \left(\frac{w_t}{t_{eff}} \right)^2 + \left(w_t \frac{t_{on}}{{t_{eff}}^2} \right)^2 \right\}^{\frac{1}{2}} \tag{A.12}$$

$$w_{P_{heater}} = \{(2w_I IR)^2 + (w_R I^2)^2\}^{\frac{1}{2}} \tag{A.13}$$

$$w_{\dot{Q}_{in}} = \dot{Q}_{in} \left\{ \left(\frac{w_{P_{heater}}}{P_{heater}} \right)^2 + \left(\frac{w_{t_{frac}}}{t_{frac}} \right)^2 \right\}^{\frac{1}{2}} \tag{A.14}$$

Table A.13: Inputs and Results for Uncertainty Propagation in Heat Input Rate

w_t (uncertainty in time)	0.001 s
t_{eff} (total time for heat input measurement)	1 800 s
t_{on} (maximum time heater is on during heat input measurement)	280 s
$w_{t_{frac}}$ (uncertainty in fraction of time heater is on)	5.62e-7 s
w_I (uncertainty in current)	0.4 A
I (maximum measured current)	16.5 A
R (maximum measured resistance)	12.088 Ω
w_R (uncertainty in resistance)	1 Ω
$w_{P_{heater}}$ (uncertainty in heater power consumption)	316 W
t_{frac} (maximum fraction of time heater is on)	0.156
Propagated Uncertainty, $w_{\dot{Q}_{in}}$	48.5 W

Heat Rejection Rate

Definition of Functional Relationship

$$\dot{Q}_{out} = \dot{V}\rho c\Delta T \tag{A.15}$$

Estimation of Uncertainties in Measured Variables

Table A.14 Estimated Uncertainties for Volume Flowrate Measurements

Sources of Error	**Error Type**	**Uncertainty Contribution (L/min)**	**Comments**
Calibration, air bubbles, tubing alignment, etc.	random	0.01	Estimate based on calibration
Total Uncertainty		0.01 L/min.	

Table A.15 Estimated Uncertainties for Coolant Density, ρ

Sources of Error	**Error Type**	**Uncertainty Contribution (kg/m^3)**	**Comments**
temperature			
impurities			
air bubbles			
pressure			
Total Uncertainty		$5\ kg/m^3$	

Table A.16 Estimated Uncertainties for Coolant Specific Heat Capacity

Sources of Error	**Error Type**	**Uncertainty Contribution (J/kgK)**	**Comments**
temperature			
impurities			

air bubbles			
pressure			
Total Uncertainty		5 J/kgK	

Propagation of Uncertainty

$$w_{\dot{Q}_{out}} = \dot{Q}_{out}\left\{\left(\frac{w_{\dot{V}}}{\dot{V}}\right)^2 + \left(\frac{w_{\rho}}{\rho}\right)^2 + \left(\frac{w_c}{c}\right)^2 + \left(\frac{w_{\Delta T}}{\Delta T}\right)^2\right\}^{\frac{1}{2}} \quad \text{(A.16)}$$

Table A.17: Inputs and Results for Uncertainty Propagation in Heat Rejection Rate

$w_{\dot{V}}$ (uncertainty in volume flow rate)	1.67e-8 m^3/s
$\dot{V}$ (maximum measured volume flow rate)	1.33e-5 m^3/s
w_{ρ} (uncertainty in coolant density)	5 kg/m^3
ρ (maximum coolant density)	1000 kg/m^3
w_c (uncertainty in coolant specific heat capacity)	5 J/kgK
c (maximum coolant specific heat capacity)	4184 J/kgK
$w_{\Delta T}$ (uncertainty in coolant temperature rise)	0.014 °C
ΔT (maximum measured coolant temperature rise)	5 °C
Propagated Uncertainty, $w_{\dot{Q}_{out}}$	2.40 W

Thermal Efficiency

Definition of Functional Relationship

$$\eta_{th} = \frac{\dot{W}}{\dot{Q}_{in}} \tag{A.17}$$

Propagation of Uncertainty

$$w_{\eta_{th}} = \left\{ \left(w_{\dot{W}} \frac{\partial \eta_{th}}{\partial \dot{W}} \right)^2 + \left(w_{\dot{Q}_{in}} \frac{\partial \eta_{th}}{\partial \dot{Q}_{in}} \right)^2 \right\}^{\frac{1}{2}} = \left\{ \left(\frac{w_{\dot{W}}}{\dot{Q}_{in}} \right)^2 + \left(w_{\dot{Q}_{in}} \frac{-\dot{W}}{\dot{Q}_{in}{}^2} \right)^2 \right\}^{\frac{1}{2}} \tag{A.18}$$

Table A.18: Inputs and Results for Uncertainty Propagation in Thermal Efficiency

$w_{\dot{W}}$ (uncertainty in shaft power)	0.25 W
$\dot{W}$ (maximum measured shaft power)	10 W
$w_{\dot{Q}_{in}}$ (uncertainty in heat input)	48.5 W
$\dot{Q}_{in}$ (maximum measured heat input)	500 W
Propagated Uncertainty, $w_{\eta_{th}}$	0.002

References

[1] A. J. Wheeler and A. R. Ganji, *Introduction to Engineering Experimentation*, Third Edit. New Jersey: Pearson Higher Education, 2004.

[2] O. E. Inc., "Thermocouples." [Online]. Available: https://www.omega.ca/prodinfo/thermocouples.html. [Accessed: 26-Nov-2017].

[3] O. E. Inc., "Resistance Elements and RTDs." [Online]. Available: https://www.omega.com/techref/ResistanceElements.html. [Accessed: 26-Nov-2017].

[4] V. E. Inc., "DP15 Variable Reluctance Pressure Sensor Capable of Range Changes." [Online]. Available: http://validyne.com/product/DP15_Variable_Reluctance_Pressure_Sensor_Capable_of_Range_Changes#sthash.eEhUF8Te.dpbs. [Accessed: 26-Nov-2017].

[5] P. Piezotronics Co., "Model: 113B21." [Online]. Available: http://www.pcb.com/products.aspx?m=113b21. [Accessed: 26-Nov-2017].

[6] F. A. S. T. Inc., "Rotary Torque Sensor – Non-Contact Shaft-to-Shaft." [Online]. Available: http://www.futek.com/product.aspx?t=torque&m=trs600. [Accessed: 26-Nov-2017].

[7] F. Co., "Fluke 325 True-rms Clamp Meter." [Online]. Available: http://www.fluke.com/fluke/caen/clamp-meters/fluke-325-true-rms-clamp-meter.htm?PID=74609. [Accessed: 26-Nov-2017].

[8] L. Good Will Instrument Co., "GDM-8351." [Online]. Available: http://www.gwinstek.com/en-global/products/Digital_Meters/Benchtop_Digital_Multimeters/GDM-8351. [Accessed: 26-Nov-2017].

Appendix B: Optimum Swept Volume Ratio Based on Equal Pressure Swings

The swept volume ratio has been estimated on the assumption that the displacer and piston should each be able to change the pressure by the same amount. The formulae of West [6] were repurposed to determine the pressure swing of each reciprocating component individually. The displacer pressure swing is calculated assuming the piston is at midstroke, and vise versa.

Assumptions

1. The pressure swing induced by the displacer should be equal to that induced by the piston.
2. The ideal gas law is applicable.
3. Piston and displacer move individually.
4. No leakage.
5. Pressure throughout the engine is constant.

Displacer Induced Pressure Swing:

From West page 29 [6], the displacer induced pressure swing ΔP_D is given by

$$\Delta P_D \approx P_m \frac{V_D}{V_m} \frac{(T_H - T_C)}{T_m} \tag{B.1}$$

where

P_m is the mean pressure (displacer at midstroke)

V_D is the displacer swept volume

V_m is the total engine volume with the piston at midstroke

T_H is the hot space gas temperature

T_C is the cold space gas temperature

T_m is the effective average gas temperature with the displacer at midstroke

Piston Induced Pressure Swing:

From the ideal gas law we have

$$P_{min} = \frac{mRT_m}{V_{max}} \qquad P_{max} = \frac{mRT_m}{V_{min}} \qquad P_m = \frac{mRT_m}{V_m}$$

where m is the mass of the working fluid

R is the ideal gas constant

V_{max} is the maximum total engine volume (piston at bottom dead center)

V_{min} is the minimum total engine volume (piston at top dead center)

P_{min} is the minimum engine pressure (piston at BDC displacer at midstroke)

P_{max} is the maximum engine pressure (piston at TDC displacer at midstroke)

Considering the piston induced pressure swing we have

$$\frac{\Delta P_P}{P_m} = \frac{P_{max} - P_{min}}{P_m} = \frac{\frac{mRT_m}{V_{min}} - \frac{mRT_m}{V_{max}}}{\frac{mRT_m}{V_m}} = \frac{\frac{1}{V_{min}} - \frac{1}{V_{max}}}{\frac{1}{V_m}}$$

$$\frac{\Delta P_P}{P_m} = V_m\left(\frac{1}{V_{min}} - \frac{1}{V_{max}}\right) = V_m\left(\frac{V_{max} - V_{min}}{V_{max}V_{min}}\right) = V_m\left(\frac{V_P}{V_{max}V_{min}}\right)$$

$$\Delta P_P = P_m V_m\left(\frac{V_P}{V_{max}V_{min}}\right) \tag{B.2}$$

where V_P is the piston swept volume

Equating the two pressure swings we have

$$\Delta P_D = \Delta P_P$$

$$P_m \frac{V_D}{V_m}\frac{(T_H - T_C)}{T_m} = P_m V_m\left(\frac{V_P}{V_{max}V_{min}}\right)$$

$$\frac{(T_H - T_C)}{T_m} = \frac{{V_m}^2}{V_D}\left(\frac{V_P}{V_{max}V_{min}}\right) \tag{B.3}$$

The volumes V_{max}, V_{min}, and V_m can be rewritten in terms of the piston swept volume and the constant volume V_C, which is the sum of the displacer swept volume and the dead volume.

$$V_m = (V_D + V_{dead}) + \frac{1}{2}V_P = V_C + \frac{1}{2}V_P \tag{B.4}$$

$$V_{min} = (V_D + V_{dead}) = V_C \tag{B.5}$$

$$V_{max} = (V_D + V_{dead} + V_P) = V_C + V_P \tag{B.6}$$

Substituting (B.4), (B.5), and (B.6) into (B.3) yields

$$\frac{(T_H - T_C)}{T_m} = \frac{V_P}{V_D V_C}\frac{\left(V_C + \frac{1}{2}V_P\right)^2}{V_C + V_P} \tag{B.7}$$

To facilitate a numerical solution, equation (B.7) may be written as a third order polynomial as follows.

$$\frac{(T_H - T_C)}{T_m} = \frac{V_P}{V_D V_C}\frac{\left(V_C + \frac{1}{2}V_P\right)\left(V_C + \frac{1}{2}V_P\right)}{V_C + V_P} = \frac{V_P}{V_D V_C}\frac{\left({V_C}^2 + V_C V_P + \frac{1}{4}{V_P}^2\right)}{V_C + V_P}$$

$$\frac{(T_H - T_C)}{T_m} = \frac{\left(V_P {V_C}^2 + V_C {V_P}^2 + \frac{1}{4}{V_P}^3\right)}{V_D {V_C}^2 + V_D V_C V_P}$$

$$T_{factor} V_D {V_C}^2 + T_{factor} V_D V_C V_P = V_P {V_C}^2 + V_C {V_P}^2 + \frac{1}{4}{V_P}^3$$

$$\left(\frac{1}{4}\right){V_P}^3 + (V_C){V_P}^2 + \left({V_C}^2 - T_{factor} V_D V_C\right)V_P + \left(-T_{factor} V_D {V_C}^2\right) = 0 \tag{B.8}$$

where $$T_{factor} = \frac{(T_H - T_C)}{T_m}$$

Lastly, an expression for T_m is needed. Assume the isothermal model temperature profile, as shown below.

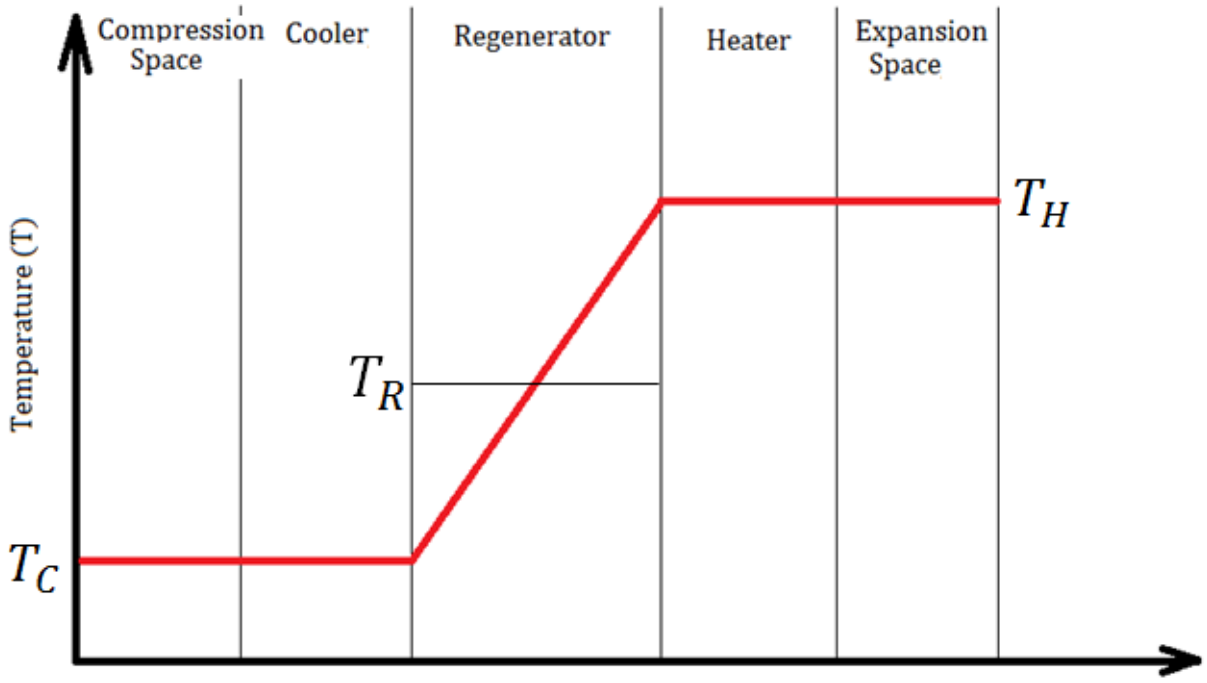


Figure A3.1.1: Isothermal Model Assumed Temperature Profile

Now consider an average temperature weighted by mass.

$$T_m = \frac{\frac{1}{2}m_D T_C + m_{cooler}T_C + m_{regen}T_R + m_{heater}T_H + \frac{1}{2}m_D T_H}{m_D + m_{cooler} + m_{regen} + m_{heater}} \qquad \text{(B.9)}$$

where m_D is the mass of gas in the displacer cylinder

m_{cooler} is the mass of gas in the cooler

m_{regen} is the mass of gas in the regenerator

m_{heater} is the mass of gas in the heater

Note that the mass of gas in the piston cylinder has not been considered here.

Using the ideal gas law to substitute for mass, reduces equation (B.9) to

$$T_m = \frac{\left(\frac{P\frac{1}{2}V_D}{RT_C}\right) + \left(\frac{PV_{cooler}}{RT_C}\right)T_C + \left(\frac{PV_{regen}}{RT_R}\right)T_R + \left(\frac{PV_{heater}}{RT_H}\right)T_H + \left(\frac{P\frac{1}{2}V_D}{RT_H}\right)T_H}{\left(\frac{P\frac{1}{2}V_D}{RT_C}\right) + \left(\frac{PV_{cooler}}{RT_C}\right) + \left(\frac{PV_{regen}}{RT_R}\right) + \left(\frac{PV_{heater}}{RT_H}\right) + \left(\frac{P\frac{1}{2}V_D}{RT_H}\right)}$$

$$T_m = \frac{V_D + V_{cooler} + V_{regen} + V_{heater}}{\left(\frac{\frac{1}{2}V_D}{T_C}\right) + \left(\frac{V_{cooler}}{T_C}\right) + \left(\frac{V_{regen}}{T_R}\right) + \left(\frac{V_{heater}}{T_H}\right) + \left(\frac{\frac{1}{2}V_D}{T_H}\right)} \tag{B.10}$$

For the regenerator temperature, the log-mean temperature will be used as recommended by Urieli and Berchowitz [10].

$$T_R = \frac{(T_H - T_C)}{\ln\left(\frac{T_H}{T_C}\right)} \tag{B.11}$$

References

[1] C. D. West, *Principles and Applications of Stirling Engines*. New York: Van Nostrand Reinhold Company Inc., 1986.

Appendix C: MATLAB Source Code for Mathematical Models

sea.m

```
function [REF_CYCLE_DATA, LOSSES_DATA, PARTICLE_TRAJECTORY_DATA, ENGINE_DATA]
= ...
    sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off)

% Israel Urieli 7/20/02 - Modified by Connor Speer April 2017
% To allow automatic iteration of the SEA code, this script has been
% converted to a function that may be called by the iteration script.

% Modified October 2017 by Connor Speer to use structures throughout and
% allow easy choice of models

%% Inputs:
% ENGINE_DATA --> Structure specifying engine geometry and operating point.
% crank_inc --> Crank angle step size for numerical integration in [degrees]
% Model_Code --> Code specifying which model to run.
% 1 = Ideal Isothermal Model
% 2 = Ideal Adiabatic Model
% 3 = Simple Model
% Losses_Code --> Code specifying which losses subfunction to run
% 0 = no loss calculations
% 1 = parasitic_losses_GEN.m
% 2 = parasitic_losses_HTG_A.m
% 3 = parasitic_losses_OPA.m
% 4 = parasitic_losses_TER.m
% 5 = parasitic_losses_LTG.m
% 6 = parasitic_losses_FSE.m
% 7 = parasitic_losses_CC.m
% 8 = parasitic_losses_HTG_B.m
% part_traj_on_off --> 1 if you want to run the particle trajectory
% function, 0 if you don't.

%% Outputs
% ISOTHERMAL_DATA --> Structure containing results from the ideal
% isothermal model
% ADIABATIC_DATA --> Structure containing results from the ideal adiabatic
% model
% SIMPLE_DATA --> Structure containing results from the simple model
% LOSSES_DATA --> Structure containing results from decoupled loss
% calculations

%% Set Default Output Structures to Zero
LOSSES_DATA = [];
PARTICLE_TRAJECTORY_DATA = [];
```

```
%% Do preliminary calculations, define all engine parameters.
ENGINE_DATA = define(ENGINE_DATA);

%% Correct working fluid mass according to Paul 2015
ENGINE_DATA = mass_correction(ENGINE_DATA,crank_inc,Model_Code);

switch Model_Code
    case 1
    % Solve the ideal isothermal model
    REF_CYCLE_DATA = isothermal(ENGINE_DATA,crank_inc);

    case 2
    % Solve the ideal adiabatic model
    REF_CYCLE_DATA = adiabatic(ENGINE_DATA,crank_inc);

    case 3
    % Solve the simple model with regenerator enthalpy loss only
    REF_CYCLE_DATA = simple(ENGINE_DATA,crank_inc);

    otherwise
    disp('ERROR: Invalid model code.')
end

%% Calculate parasitic losses and add them to the reference cycle results
switch Losses_Code
    case 0
        % Do NOTHING
    case 1
        % Use the general losses subfunction to calculate decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_GEN(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 2
        % Use the unvalidated HTG engine specific losses subfunction to
calculate decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_HTG(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 3
        % Use the OPA engine specific losses subfunction to calculate
decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_OPA(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 4
        % Use the Terrapin engine specific losses subfunction to calculate
decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_TER(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 5
        % Use the low temperature gamma engine specific losses subfunction to
calculate decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_LTG(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 6
        % Use the Franchot engine specific losses subfunction to calculate
decoupled losses
```

```
        LOSSES_DATA                                                    =
parasitic_losses_FSE(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 7
        % Use the coffee cup engine specific losses subfunction to calculate
decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_CC(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    case 8
        % Use the "validated" HTG engine specific losses subfunction to
calculate decoupled losses
        LOSSES_DATA                                                    =
parasitic_losses_HTG_B(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);
    otherwise
        disp('ERROR: Invalid losses code')
end


%% Calculate 1-D particle trajectories and flush ratios
Num_Mass_Inc = 10;
if part_traj_on_off == 1
    PARTICLE_TRAJECTORY_DATA = ...
        Gen_Part_Traj(ENGINE_DATA,REF_CYCLE_DATA,Num_Mass_Inc);
end
```

The Define Function Set

define.m

```
function ENGINE_DATA = define(ENGINE_DATA)
% define the stirling engine geometric
% and operational parameters
% Israel Urieli 4/1/02 (April Fool's Day)
% Modified 2/12/2010 to include no-matrix regenerator awgr0

% Modified by Connor Speer April 2017. This function has been changed to
% accept an engine data structure as an input.
% Modified by Connor Speer October 2017. No more global variables.

% Calculate and display drive mechanism information
ENGINE_DATA = engine(ENGINE_DATA);

% Calculate and display heat exchanger and regenerator information
ENGINE_DATA = heatex(ENGINE_DATA);

% Calculate and display working fluid information
ENGINE_DATA = gas(ENGINE_DATA);

% Calculate mas of working fluid and display Schmidt analysis results
ENGINE_DATA = operat(ENGINE_DATA);
```

engine.m

```
function ENGINE_DATA = engine(ENGINE_DATA)
% Define engine configuration and drive geometric parameters.
% Israel Urieli 4/14/02

% Modified by Connor Speer May 2017, October 2017

engine_type = ENGINE_DATA.engine_type; % Engine layout and drive mechanism,
see below.

if(strncmp(engine_type,'s',1)) % Sinusoidal alpha
    ENGINE_DATA = sindrive(ENGINE_DATA);
elseif(strncmp(engine_type,'y',1)) % Ross yoke mechanism alpha
    ENGINE_DATA = yokedrive(ENGINE_DATA);
elseif(strncmp(engine_type,'r',1)) % Ross rocker V-drive alpha
    ENGINE_DATA = rockerVdrive(ENGINE_DATA);
elseif(strncmp(engine_type,'g',1)) % Sinusoidal gamma
    ENGINE_DATA = gammasindrive(ENGINE_DATA);
elseif(strncmp(engine_type,'x',1)) % Slider-crank mechanism gamma
    ENGINE_DATA = gammacrankdrive(ENGINE_DATA);
elseif(strncmp(engine_type,'a',1)) % Slider-crank mechanism alpha
    ENGINE_DATA = alphacrankdrive(ENGINE_DATA);
else
    fprintf('engine type is undefined\n')
end
%=============================================================

function ENGINE_DATA = sindrive(ENGINE_DATA)
% Sinusoidal drive engine configuration
% Israel Urieli 4/14/02
% Connor Speer, October 2017

% SET ENGINE VOLUME AND PHASE ANGLE PARAMETERS;
Vclc = ENGINE_DATA.Vclc;
Vcle = ENGINE_DATA.Vcle;
Vswe = ENGINE_DATA.Vswe;
Vswc = ENGINE_DATA.Vswc;
alpha_deg = ENGINE_DATA.alpha_deg;

fprintf('Sinusoidal alpha engine configuration\n')
fprintf('\nsinusoidal drive engine data summary:\n');
fprintf('\t    comp    clearence,swept    vols    %.1f,    %.1f    [cm^3]\n',
Vclc*1e6,Vswc*1e6);
fprintf('\t    exp    clearence,swept    vols    %.1f,    %.1f    [cm^3]\n',
Vcle*1e6,Vswe*1e6);
fprintf('\t expansion phase angle advance %.1f[degrees]\n', alpha_deg);
%=============================================================

function ENGINE_DATA = yokedrive(ENGINE_DATA)
% Ross yoke drive engine configuration
% Israel Urieli 4/14/02
% Connor Speer, October 2017

fprintf('Ross yoke drive engine configuration\n');
% Compression, expansion swept volumes [m^3]
Vclc = ENGINE_DATA.Vclc;
```

```
Vcle = ENGINE_DATA.Vcle;

b1 = ENGINE_DATA.b1; % Ross yoke length (1/2 yoke base) [m]
b2 = ENGINE_DATA.b2; % Ross yoke height [m]
crank = ENGINE_DATA.crank; % crank radius [m]

% Diameter of compression/expansion pistons [m]
dcomp = ENGINE_DATA.dcomp;
dexp = ENGINE_DATA.dexp;

% Area of compression/expansion pistons [m^2]
acomp = pi*dcomp^2/4.0;
aexp = pi*dexp^2/4.0;

% Minimum yoke vertical displacement [m]
yoke = sqrt(b1^2 + b2^2);
ymax = sqrt((yoke + crank)^2 - b2^2);
ymin = sqrt((yoke - crank)^2 - b2^2);

% Compression, expansion swept volumes [m^3]
Vswc = acomp*(ymax - ymin);
Vswe = aexp*(ymax - ymin);

% Phase angle advance of expansion space [radians]
thmaxe = asin(ymax/(yoke + crank));
thmaxc = pi - thmaxe;
thmine = pi + asin(ymin/(yoke - crank));
thminc = 3*pi - thmine;
alpha = 0.5*(thmaxc - thmaxe) + 0.5*(thminc - thmine);
alpha_deg = alpha*180/pi;

fprintf('\nRoss yoke drive engine data summary:\n');
fprintf(' yoke length b1 (1/2 yoke base) %.1f [mm]\n', b1*1e3);
fprintf(' yoke height b2 %.1f [mm]\n', b2*1e3);
fprintf(' crank radius %.1f [mm]\n', crank*1e3);
fprintf(' compression piston diameter %.1f [mm]\n', dcomp*1e3);
fprintf(' expansion piston diameter %.1f [mm]\n', dexp*1e3);
fprintf(' comp clearence,swept vols %.1f, %.1f [cm^3]\n', Vclc*1e6,Vswc*1e6);
fprintf(' exp clearence,swept vols %.1f, %.1f [cm^3]\n', Vcle*1e6,Vswe*1e6);
fprintf(' ymin = %.1f(cm), ymax = %.1f(cm)\n',ymin*1e2,ymax*1e2)
fprintf(' alpha = %.1f(degrees)\n',alpha_deg);
%=============================================================

function ENGINE_DATA = rockerVdrive(ENGINE_DATA)
% Ross rocker-V drive engine configuration
% Israel Urieli 4/14/02 & Martine Long 2/14/05
% Connor Speer, October 2017

fprintf('Ross rocker-V drive engine configuration\n');

% Compression,expansion clearence vols [m^3]
Vclc = ENGINE_DATA.Vclc;
Vcle = ENGINE_DATA.Vcle;

crank = ENGINE_DATA.crank; % crank radius [m]
```

```
% Length of comp/exp piston connecting rods [m]
conrodc = ENGINE_DATA.conrodc;
conrode = ENGINE_DATA.conrode;

% Diameter of compression/expansion pistons [m]
dcomp = ENGINE_DATA.dcomp;
dexp = ENGINE_DATA.dexp;

% Phase angle advance of expansion space [radians]
alpha_deg = ENGINE_DATA.alpha_deg;

% Area of compression/expansion pistons [m^2]
acomp = pi*dcomp^2/4.0;
aexp = pi*dexp^2/4.0;

% Maximum comp/exp piston vertical displacement [m]
ycmax = conrodc + crank;
ycmin = conrodc - crank;
yemax = conrode + crank;
yemin = conrode - crank;

% Compression, expansion swept volumes [m^3]
Vswc = acomp*(ycmax - ycmin);
Vswe = aexp*(yemax - yemin);

fprintf('\nRoss rocker-V drive engine data summary:\n');
fprintf(' crank radius %.1f [mm]\n', crank*1e3);
fprintf('   compression   piston   connecting   rod   length   %.1f   [mm]\n',
conrodc*1e3);
fprintf(' expansion piston connecting rod length %.1f [mm]\n', conrode*1e3);
fprintf(' compression piston diameter  %.1f [mm]\n', dcomp*1e3);
fprintf(' expansion piston diameter %.1f [mm]\n', dexp*1e3);
fprintf(' comp clearence,swept vols %.1f, %.1f [cm^3]\n', Vclc*1e6,Vswc*1e6);
fprintf(' exp clearence,swept vols %.1f, %.1f [cm^3]\n', Vcle*1e6,Vswe*1e6);
fprintf('      COMPRESSION      ymin      =      %.1f(cm),      ymax      =
%.1f(cm)\n',ycmin*1e2,ycmax*1e2)
fprintf(' EXPANSION ymin = %.1f(cm), ymax = %.1f(cm)\n',yemin*1e2,yemax*1e2)
fprintf(' expansion phase angle advance %.1f[degrees]\n', alpha_deg);
%=============================================================

%=============================================================
function ENGINE_DATA = gammasindrive(ENGINE_DATA)
% Gamma sinusoidal drive engine configuration
% Added by Connor Speer 01/06/17
% Connor Speer, October 2017

fprintf('gamma sinusoidal drive engine configuration\n')

% Piston, displacer clearence vols [m^3]
Vcld = ENGINE_DATA.Vcld;
Vclp = ENGINE_DATA.Vclp;

% Piston, displacer swept volumes [m^3]
Vswp = ENGINE_DATA.Vswp;
```

```
Vswd = ENGINE_DATA.Vswd;

% Displacer Phase Angle Advance for Gamma
beta_deg = ENGINE_DATA.beta_deg; %[deg]

fprintf('\n gamma sinusoidal drive engine data summary:\n');
fprintf('    piston    clearence,swept    vols    %.1f,    %.1f    [cm^3]\n',
Vclp*1e6,Vswp*1e6);
fprintf('    displacer    clearence,swept    vols    %.1f,    %.1f    [cm^3]\n',
Vcld*1e6,Vswd*1e6);
fprintf(' displacer phase angle advance %.1f[degrees]\n', beta_deg);

%=============================================================
function ENGINE_DATA = gammacrankdrive(ENGINE_DATA)
% Gamma crankshaft drive engine configuration
% Added by Connor Speer 02/14/17
% Connor Speer, October 2017

% Piston, displacer clearence vols [m^3]
Vclp = ENGINE_DATA.Vclp;
Vcld = ENGINE_DATA.Vcld;

% Piston, displacer bores [m]
Pbore = ENGINE_DATA.Pbore;
Dbore = ENGINE_DATA.Dbore;

% Piston, displacer desaxe offset in [m]
Pr1 = ENGINE_DATA.Pr1;
Dr1 = ENGINE_DATA.Dr1;

% Piston, displacer crank radius in [m]
Pr2 = ENGINE_DATA.Pr2;
Dr2 = ENGINE_DATA.Dr2;

% Piston, displacer connecting rod lengths [m]
Pr3 = ENGINE_DATA.Pr3;
Dr3 = ENGINE_DATA.Dr3;

% Displacer Phase Angle Advance for Gamma
beta_deg = ENGINE_DATA.beta_deg; %[deg]

fprintf('gamma crankshaft drive engine configuration\n')

% Calculate the swept volumes (NEED TO FIX THESE TO WORK FOR CASES WHERE
% THERE IS A DESAXE OFFSET!!!!)
Vswp = (pi/4)*(Pbore^2)*(2*Pr2);
Vswd = (pi/4)*(Dbore^2)*(2*Dr2);

fprintf('\n Gamma crankshaft drive engine data summary:\n');
fprintf('    piston    clearence,swept    vols    %.1f,    %.1f    [cm^3]\n',
Vclp*1e6,Vswp*1e6);
fprintf('    displacer    clearence,swept    vols    %.1f,    %.1f    [cm^3]\n',
Vcld*1e6,Vswd*1e6);
fprintf(' displacer phase angle advance %.1f[degrees]\n', beta_deg);
```

```
%=============================================================
function ENGINE_DATA = alphacrankdrive(ENGINE_DATA)
% Alpha crankshaft drive engine configuration
% Added by Connor Speer 02/15/17
% Connor Speer, October 2017

% Expansion Space Phase Angle Advance [deg]
alpha_deg = ENGINE_DATA.alpha_deg;

% Compression, expansion clearence vols [m^3]
Vclc = ENGINE_DATA.Vclc;
Vcle = ENGINE_DATA.Vcle;

% Compression, expansion piston bores [m]
Cbore = ENGINE_DATA.Cbore;
Ebore = ENGINE_DATA.Ebore;

% Compression, expansion desaxe offset in [m]
Cr1 = ENGINE_DATA.Cr1;
Er1 = ENGINE_DATA.Er1;

% Compression, expansion crank radius in [m]
Cr2 = ENGINE_DATA.Cr2;
Er2 = ENGINE_DATA.Er2;

% Compression, expansion connecting rod lengths [m]
Cr3 = ENGINE_DATA.Cr3;
Er3 = ENGINE_DATA.Er3;

% Calculate the swept volumes (NEED TO FIX THESE TO WORK FOR CASES WHERE
% THERE IS A DESAXE OFFSET!!!!)
Vswc = (pi/4)*(Cbore^2)*(2*Cr2);
Vswe = (pi/4)*(Ebore^2)*(2*Er2);

ENGINE_DATA.Vswc = Vswc;
ENGINE_DATA.Vswe = Vswe;

% %% DATA PRINTING
% fprintf('\nAlpha crankshaft drive engine data summary:\n');
%  fprintf('\t compression space clearence,swept vols %.1f, %.1f [cm^3]\n',
Vclc*1e6,Vswc*1e6);
%  fprintf('\t expansion space clearence,swept vols %.1f, %.1f [cm^3]\n',
Vcle*1e6,Vswe*1e6);
%  fprintf('\t expansion space phase angle advance %.1f[degrees]\n',
alpha_deg);
```

gas.m

```
function ENGINE_DATA = gas(ENGINE_DATA)
% specifies the working gas properties (he, h2, air)
% Israel Urieli 4/20/02
```

```
gas_type = ENGINE_DATA.gas_type;

if(strncmp(gas_type,'hy',2))
%    fprintf('gas type is hydrogen\n')
   ENGINE_DATA.gamma = 1.4;   % Specific heat ratio: cp/cv
   ENGINE_DATA.R = 4157.2; % Specific gas constant in [J/kgK]
   ENGINE_DATA.kgas = 0.2843; % Thermal conductivity in [W/mK]
   ENGINE_DATA.mu0  =  8.35e-6;  %  Dynamic  viscosity  at  reference  temp  t0
[kg.m/s]
   ENGINE_DATA.t_suth = 84.4; % Sutherland constant [K]
elseif(strncmp(gas_type,'he',2))
%     fprintf('gas type is helium\n')
    ENGINE_DATA.gamma = 1.67;  % Specific heat ratio: cp/cv
    ENGINE_DATA.R = 2078.6; % Specific gas constant in [J/kgK]
    ENGINE_DATA.kgas = 0.142; % Thermal conductivity in [W/mK]
    ENGINE_DATA.mu0  =  18.85e-6;  %  Dynamic  viscosity  at  reference  temp  t0
[kg.m/s]
    ENGINE_DATA.t_suth = 80.0; % Sutherland constant [K]
elseif(strncmp(gas_type,'ai',2))
%     fprintf('gas type is air\n')
    ENGINE_DATA.gamma = 1.4;   % Specific heat ratio: cp/cv
    ENGINE_DATA.R = 287.0;  % Specific gas constant in [J/kgK]
    ENGINE_DATA.kgas = 0.04418; % Thermal conductivity in [W/mK]
    ENGINE_DATA.mu0  =  17.08e-6;  %  Dynamic  viscosity  at  reference  temp  t0
[kg.m/s]
    ENGINE_DATA.t_suth = 112.0; % Sutherland constant [K]
else
%     fprintf('gas type is undefined\n')
end
```

heatex.m

```
function ENGINE_DATA = heatex(ENGINE_DATA)
% Specify heat exchanger geometric parameters
% Israel Urieli 3/31/02 (modified 12/01/03)
% Modified 2/14/2010 annulus and slots wetted area
% Modified by Connor Speer, October 2017, no globals


ENGINE_DATA = cooler(ENGINE_DATA);

ENGINE_DATA = regen(ENGINE_DATA);

ENGINE_DATA = heater(ENGINE_DATA);
%=======================================================

function ENGINE_DATA = cooler(ENGINE_DATA)
% Specify cooler geometric parameters
% Israel Urieli 4/15/02

cooler_type = ENGINE_DATA.cooler_type;

if(strncmp(cooler_type,'p',1)) % Pipes
```

```
    ENGINE_DATA.dk  =  ENGINE_DATA.cooler_pipe_diameter;  % cooler hydraulic
diameter [m]
    ENGINE_DATA.lk  =  ENGINE_DATA.cooler_pipe_length;  % cooler effective
length [m]
    ENGINE_DATA.ak                                                    =
ENGINE_DATA.cooler_num_pipes*pi*ENGINE_DATA.dk*ENGINE_DATA.dk/4;   %   cooler
internal free flow area [m^2]
    ENGINE_DATA.Vk  =  ENGINE_DATA.ak*ENGINE_DATA.lk;  % cooler void volume
[m^3]
    ENGINE_DATA.awgk                                                    =
ENGINE_DATA.cooler_num_pipes*pi*ENGINE_DATA.dk*ENGINE_DATA.lk;    %    cooler
internal wetted area [m^2]

elseif(strncmp(cooler_type,'a',1)) % Annulus
    % Inner and outer diameters of the annular heater [m]
    dout = ENGINE_DATA.cooler_outer_diameter;
    din = ENGINE_DATA.cooler_inner_diameter;

    ENGINE_DATA.dk = dout - din; % cooler hydraulic diameter [m]
    ENGINE_DATA.lk = ENGINE_DATA.cooler_length; % cooler effective length [m]
    ENGINE_DATA.ak = pi*(dout*dout - din*din)/4; % cooler internal free flow
area [m^2]
    ENGINE_DATA.Vk = ENGINE_DATA.ak*ENGINE_DATA.cooler_length; % cooler void
volume [m^3]
    ENGINE_DATA.awgk =  pi*dout*ENGINE_DATA.cooler_length; % cooler internal
wetted area [m^2]

elseif(strncmp(cooler_type,'s',1)) % Slots
    % cooler internal free flow area [m^2]
    ENGINE_DATA.ak                                                    =
ENGINE_DATA.cooler_num_slots*ENGINE_DATA.cooler_slot_width*ENGINE_DATA.cooler
_slot_height;

    % cooler void volume [m^3]
    ENGINE_DATA.Vk = ENGINE_DATA.ak*ENGINE_DATA.cooler_length;

    % cooler internal wetted area [m^2]
    ENGINE_DATA.awgk                                                    =
ENGINE_DATA.cooler_num_slots*(ENGINE_DATA.cooler_slot_width                +
2*ENGINE_DATA.cooler_slot_height)*ENGINE_DATA.cooler_length;

    ENGINE_DATA.dk  =  4*ENGINE_DATA.Vk/ENGINE_DATA.awgk;  % cooler hydraulic
diameter [m]
    ENGINE_DATA.lk = ENGINE_DATA.cooler_length; % cooler effective length [m]

elseif(strncmp(cooler_type,'c',1)) % Custom
    ENGINE_DATA.ak  =  ENGINE_DATA.cooler_free_flow_area;  % cooler internal
free flow area [m^2]
    ENGINE_DATA.Vk = ENGINE_DATA.ak*ENGINE_DATA.cooler_length; % cooler void
volume [m^3]
    ENGINE_DATA.awgk  =  ENGINE_DATA.cooler_wetted_surface_area;  % cooler
internal wetted area [m^2]
    ENGINE_DATA.dk  =  4*ENGINE_DATA.Vk/ENGINE_DATA.awgk;  % cooler hydraulic
diameter [m]
    ENGINE_DATA.lk = ENGINE_DATA.cooler_length; % cooler effective length [m]
```

```
else
    fprintf('ERROR: Cooler type is undefined.\n')
end
%% PRINTING
% fprintf('Cooler data summary:\n');
% fprintf('\t void volume(cc) %.2f\n', ENGINE_DATA.Vk*1e6)
% fprintf('\t free flow area (cm^2) %.2f\n', ENGINE_DATA.ak*1e4)
% fprintf('\t wetted area (cm^2) %.2f\n', ENGINE_DATA.awgk*1e4)
% fprintf('\t hydraulic diameter(mm) %.2f\n', ENGINE_DATA.dk*1e3)
% fprintf('\t cooler length (cm) %.2f\n', ENGINE_DATA.lk*1e2)

%=========================================================
function ENGINE_DATA = heater(ENGINE_DATA)
% Specify heater geometric parameters
% Israel Urieli 4/15/02
% Modified by Connor Speer, October 2017. No globals.

heater_type = ENGINE_DATA.heater_type;

if(strncmp(heater_type,'p',1)) % Pipes
    ENGINE_DATA.dh  =  ENGINE_DATA.heater_pipe_diameter;  % heater hydraulic
diameter [m]
    ENGINE_DATA.lh  =  ENGINE_DATA.heater_pipe_length;  % heater effective
length [m]
    ENGINE_DATA.ah                                                    =
ENGINE_DATA.heater_num_pipes*pi*ENGINE_DATA.dh*ENGINE_DATA.dh/4;    %    heater
internal free flow area [m^2]
    ENGINE_DATA.Vh  =  ENGINE_DATA.ah*ENGINE_DATA.lh;  % heater void volume
[m^3]
    ENGINE_DATA.awgh                                                    =
ENGINE_DATA.heater_num_pipes*pi*ENGINE_DATA.dh*ENGINE_DATA.lh;     %     heater
internal wetted area [m^2]

elseif(strncmp(heater_type,'a',1)) % Annulus
    % Inner and outer diameters of the annular heater [m]
    dout = ENGINE_DATA.heater_outer_diameter;
    din = ENGINE_DATA.heater_inner_diameter;

    ENGINE_DATA.dh = dout - din; % heater hydraulic diameter [m]
    ENGINE_DATA.lh = ENGINE_DATA.heater_length; % heater effective length [m]
    ENGINE_DATA.ah = pi*(dout*dout - din*din)/4; % heater internal free flow
area [m^2]
    ENGINE_DATA.Vh = ENGINE_DATA.ah*ENGINE_DATA.heater_length; % heater void
volume [m^3]
    ENGINE_DATA.awgh =  pi*dout*ENGINE_DATA.heater_length; % heater internal
wetted area [m^2]

elseif(strncmp(heater_type,'s',1)) % Slots
    % heater internal free flow area [m^2]
    ENGINE_DATA.ah                                                    =
ENGINE_DATA.heater_num_slots*ENGINE_DATA.heater_slot_width*ENGINE_DATA.heater
_slot_height;

    % heater void volume [m^3]
```

```
    ENGINE_DATA.Vh = ENGINE_DATA.ah*ENGINE_DATA.heater_length;

    % heater internal wetted area [m^2]
    ENGINE_DATA.awgh                                                    =
ENGINE_DATA.heater_num_slots*(ENGINE_DATA.heater_slot_width                 +
2*ENGINE_DATA.heater_slot_height)*ENGINE_DATA.heater_length;

    ENGINE_DATA.dh  =  4*ENGINE_DATA.Vh/ENGINE_DATA.awgh;  % heater hydraulic
diameter [m]
    ENGINE_DATA.lh = ENGINE_DATA.heater_length; % heater effective length [m]

elseif(strncmp(heater_type,'c',1)) % Custom
    ENGINE_DATA.ah  =  ENGINE_DATA.heater_free_flow_area;  % heater internal
free flow area [m^2]
    ENGINE_DATA.Vh  =  ENGINE_DATA.ah*ENGINE_DATA.heater_length;  % heater void
volume [m^3]
    ENGINE_DATA.awgh   =   ENGINE_DATA.heater_wetted_surface_area;   %   heater
internal wetted area [m^2]
    ENGINE_DATA.dh  =  4*ENGINE_DATA.Vh/ENGINE_DATA.awgh;  % heater hydraulic
diameter [m]
    ENGINE_DATA.lh = ENGINE_DATA.heater_length; % heater effective length [m]

else
    fprintf('ERROR: Heater type is undefined.\n')
end

% %% PRINTING
% fprintf('Heater data summary:\n');
% fprintf('\t void volume(cc) %.2f\n', ENGINE_DATA.Vh*1e6)
% fprintf('\t free flow area (cm^2) %.2f\n', ENGINE_DATA.ah*1e4)
% fprintf('\t wetted area (cm^2) %.2f\n', ENGINE_DATA.awgh*1e4)
% fprintf('\t hydraulic diameter(mm) %.2f\n', ENGINE_DATA.dh*1e3)
% fprintf('\t heater length (cm) %.2f\n', ENGINE_DATA.lh*1e2)
```

operat.m

```
function ENGINE_DATA = operat(ENGINE_DATA)
% Determine operating parameters and do Schmidt anlysis
% Israel Urieli 4/20/02

% Modified by Connor Speer, October 2017. No globals.

% Calculate regenerator mean effective temperature
ENGINE_DATA.Tgr                =                (ENGINE_DATA.Tgh                -
ENGINE_DATA.Tgk)/log(ENGINE_DATA.Tgh/ENGINE_DATA.Tgk);

ENGINE_DATA.omega = 2*pi*ENGINE_DATA.freq; % Engine angular frequency (rad/s)

% fprintf('Operating parameters:\n');
% fprintf('\t mean pressure (kPa): %.3f\n',ENGINE_DATA.Pmean*1e-3);
% fprintf('\t cold sink temperature (K): %.1f\n',ENGINE_DATA.TC);
% fprintf('\t hot source temperature (K): %.1f\n',ENGINE_DATA.TH);
% fprintf('\t effective regenerator temperature (K): %.1f\n',ENGINE_DATA.TR);
```

```
% fprintf('\t operating frequency (hertz): %.1f\n',ENGINE_DATA.freq);

ENGINE_DATA = Schmidt(ENGINE_DATA); % Do Schmidt analysis
%=============================================================
function ENGINE_DATA = Schmidt(ENGINE_DATA)
% Schmidt anlysis
% Israel Urieli 3/31/02 - Extended by Connor Speer January 2017
% Modified by Connor Speer, October 2017. No globals.

engine_type = ENGINE_DATA.engine_type;

if strcmp(engine_type,'g') || strcmp(engine_type,'x')
    % Gamma Schmidt analysis
    % Constant Calculations
    Vdead = ENGINE_DATA.Vh + ENGINE_DATA.Vr + ENGINE_DATA.Vk; % Dead Volume
    tao = ENGINE_DATA.Tgk/ENGINE_DATA.Tgh; % Temperature ratio
    kappa = ENGINE_DATA.Vswp/ENGINE_DATA.Vswd; % Ratio of piston swept volume
to displacer swept volume
    chi = Vdead/ENGINE_DATA.Vswd; % Dead volume ratio
    freq = ENGINE_DATA.freq; % Engine Frequency [Hz]
    ENGINE_DATA.beta = ENGINE_DATA.beta_deg*(pi/180); %(rad)

    % Senft Pressure calculations
    Y = 1 + tao + kappa + ((4*chi*tao)/(1+tao));
    A = kappa - ((1-tao)*cosd(ENGINE_DATA.beta_deg));
    B = (1-tao)*sind(ENGINE_DATA.beta_deg);
    X = sqrt((A^2) + (B^2));
    angle = acos(A/(sqrt((A^2) + (B^2)))); % Is this the same as pressure
phase below?

    % Here the mass of gas in the engine is calculated as the mean of the
    % mass calculated at each crank angle degree.
    theta = 0:(pi/180):2*pi; %(rad)
    Ve = (0.5*ENGINE_DATA.Vcld)+(ENGINE_DATA.Vswd*0.5)*...
        (1+cos(theta+ENGINE_DATA.beta)); % Expansion space volume in [m^3].
    Vc = (0.5*ENGINE_DATA.Vcld)+ENGINE_DATA.Vclp+ENGINE_DATA.Vswd-Ve+...
        (ENGINE_DATA.Vswp*0.5*(1+cos(theta))); % Compression space volume in
[m^3].

    % Mass of gas in the engine [kg].
    ENGINE_DATA.mgas                                                    =
ENGINE_DATA.pmean/(mean(ENGINE_DATA.R*((Vc./ENGINE_DATA.Tgk)...

+(ENGINE_DATA.Vk/ENGINE_DATA.Tgk)+((ENGINE_DATA.Vr*log(ENGINE_DATA.Tgh/ENGINE
_DATA.Tgk))/...
        (ENGINE_DATA.Tgh-
ENGINE_DATA.Tgk))+(ENGINE_DATA.Vh/ENGINE_DATA.Tgh)+(Ve./ENGINE_DATA.Tgh)).^-
1));

    % Calculation of heat and work.
    W_Schmidt = (pi*(1-tao)*(ENGINE_DATA.Vswd+ENGINE_DATA.Vswp)...

*ENGINE_DATA.pmean*kappa*sin(ENGINE_DATA.beta))/((kappa+1)*(sqrt((Y^2)-
(X^2))+Y));
```

```
    Power_Schmidt = W_Schmidt*freq;

    eff_Schmidt = 1-tao; % Carnot efficency

else
    ENGINE_DATA.alpha = ENGINE_DATA.alpha_deg*(pi/180); %(rad)

    % Alpha Schmidt analysis
    c           =           (((ENGINE_DATA.Vswe/ENGINE_DATA.Tgh)^2           +
(ENGINE_DATA.Vswc/ENGINE_DATA.Tgk)^2 +...

2*(ENGINE_DATA.Vswe/ENGINE_DATA.Tgh)*(ENGINE_DATA.Vswc/ENGINE_DATA.Tgk)*cos(E
NGINE_DATA.alpha))^0.5)/2;

    s       =       (ENGINE_DATA.Vswc/2       +       ENGINE_DATA.Vclc       +
ENGINE_DATA.Vk)/ENGINE_DATA.Tgk +...
        ENGINE_DATA.Vr/ENGINE_DATA.Tgr      +      (ENGINE_DATA.Vswe/2      +
ENGINE_DATA.Vcle + ENGINE_DATA.Vh)/ENGINE_DATA.Tgh;
    b = c/s;
    sqrtb = (1 - b^2)^0.5;
    bf = (1 - 1/sqrtb);

    pressure_phase                                                          =
atan(ENGINE_DATA.Vswe*sin(ENGINE_DATA.alpha)/ENGINE_DATA.Tgh/...
        (ENGINE_DATA.Vswe*cos(ENGINE_DATA.alpha)/ENGINE_DATA.Tgh            +
ENGINE_DATA.Vswc/ENGINE_DATA.Tgk));

%                           fprintf('     pressure     phase     angle     beta
%.1f(degrees)\n',pressure_phase*180/pi)

    % total mass of working gas in engine
    ENGINE_DATA.mgas = ENGINE_DATA.pmean*s*sqrtb/ENGINE_DATA.R;
%     fprintf(' total mass of gas:  %.3f(gm)\n',ENGINE_DATA.mgas*1e3)

    % work output
    Wc                                                              =
(pi*ENGINE_DATA.Vswc*ENGINE_DATA.mgas*ENGINE_DATA.R*sin(pressure_phase)*bf/c)
;
    We                                                              =
(pi*ENGINE_DATA.Vswe*ENGINE_DATA.mgas*ENGINE_DATA.R*sin(pressure_phase       -
ENGINE_DATA.alpha)*bf/c);
    W_Schmidt = (Wc + We);
    Power_Schmidt = W_Schmidt*ENGINE_DATA.freq;
    eff_Schmidt = W_Schmidt/We; % qe = we (Carnot efficiency)
end
% Printout Schmidt analysis results
fprintf('====================  Schmidt analysis  ==============\n')
fprintf('      Work(joules)      %.3f      \n      Power(watts)      %.3f\n',
W_Schmidt,Power_Schmidt);
% fprintf(' Qexp(joules) %.3e,  Qcom(joules) %.3e\n', we,wc);
fprintf('    Indicated    efficiency    (Carnot    efficiency)    %.3f\n    %%',
eff_Schmidt*100);
fprintf('============================================================\n')
```

regen.m

```
function ENGINE_DATA = regen(ENGINE_DATA)
% Specifies regenerator geometric and thermal properties
% Israel Urieli 04/20/02 (modified 12/01/03)
% modified 2/12/2010 to include awgr0 (wetted area)
% modified 11/27/2010 to include 'no regenerator matrix'
% modified 10/5/2017 by Connor Speer and Jason Michaud
% modified 10/15/2017 by Connor Speer, no globals.

regen_type = ENGINE_DATA.regen_config;

% Available regenerator configurations are:
% t for tubular regenerator housing
% a for annular regenerator housing
% r for rectangular regenerator housing

if(strncmp(regen_type,'t',1))
%     fprintf('Tubular regenerator housing\n')
    ENGINE_DATA.lr = ENGINE_DATA.regen_length; % regenerator effective length
[m]

    domat = ENGINE_DATA.regen_housing_ID;
    dimat = 0;

    % no matrix regenerator wetted area [m^2]
    ENGINE_DATA.awgr0 = ENGINE_DATA.regen_num_tubes*pi*domat*ENGINE_DATA.lr;

    ENGINE_DATA.amat    =    ENGINE_DATA.regen_num_tubes*pi*(domat*domat    -
dimat*dimat)/4; % regen matrix area

elseif(strncmp(regen_type,'a',1))
%     fprintf('Annular regenerator housing\n')
    ENGINE_DATA.lr = ENGINE_DATA.regen_length; % regenerator effective length
[m]

    domat = ENGINE_DATA.regen_housing_ID;
    dimat = ENGINE_DATA.regen_matrix_ID;

    % no matrix regenerator wetted area [m^2]
    ENGINE_DATA.awgr0                                                    =
ENGINE_DATA.regen_num_tubes*pi*(domat+dimat)*ENGINE_DATA.lr;

    ENGINE_DATA.amat    =    ENGINE_DATA.regen_num_tubes*pi*(domat*domat    -
dimat*dimat)/4; % regen matrix area

 elseif(strncmp(regen_type,'r',1))
%     fprintf('Rectangular regenerator housing\n')
    ENGINE_DATA.lr = ENGINE_DATA.regen_length; % regenerator effective length
[m]

    % no matrix regenerator wetted area [m^2]
```

```
    ENGINE_DATA.awgr0                                                    =
((2*ENGINE_DATA.regen_height)+(2*ENGINE_DATA.regen_width))*ENGINE_DATA.lr*ENG
INE_DATA.regen_num;

    ENGINE_DATA.amat                                                     =
ENGINE_DATA.regen_num*ENGINE_DATA.regen_height*ENGINE_DATA.regen_width;      %
regen matrix area

 else
%   fprintf('regenerator configuration is undefined\n')
end

% Call regenerator matrix subfunction
% Available regenerator materials are:
% m for mesh regenerator material
% f for foil regenerator material
% n for  regenerator material
ENGINE_DATA = matrix(ENGINE_DATA);

%=============================================================
function ENGINE_DATA = matrix(ENGINE_DATA)
% Specifies regenerator matrix geometric and thermal properties
% Israel Urieli 03/31/02
% modified 11/27/10 for no regenerator matrix
% Modified 10/5/2017 by Connor Speer
% Modified 10/15/2017 by Connor Speer, no globals


matrix_type = ENGINE_DATA.regen_matrix_type;

if(strncmp(matrix_type,'m',1)) % Mesh regenerator
    ENGINE_DATA.ar = ENGINE_DATA.amat*ENGINE_DATA.regen_matrix_porosity;
    ENGINE_DATA.lr = ENGINE_DATA.regen_length;
    ENGINE_DATA.Vr = ENGINE_DATA.ar*ENGINE_DATA.lr;
    ENGINE_DATA.dr                                                       =
ENGINE_DATA.regen_wire_diameter*ENGINE_DATA.regen_matrix_porosity/(1          -
ENGINE_DATA.regen_matrix_porosity);
    ENGINE_DATA.awgr = 4*ENGINE_DATA.Vr/ENGINE_DATA.dr + ENGINE_DATA.awgr0;

%                           fprintf('\t     matrix     porosity:     %.3f\n',
ENGINE_DATA.regen_matrix_porosity)
%                       fprintf('\t    matrix    wire    diam    %.2f(mm)\n',
ENGINE_DATA.regen_wire_diameter*1e3)
%     fprintf('\t hydraulic diam %.3f(mm)\n', ENGINE_DATA.dr*1e3)
%     fprintf('\t total wetted area %.3e(sq.m)\n', ENGINE_DATA.awgr)
%     fprintf('\t regenerator length %.1f(mm)\n', ENGINE_DATA.lr*1e3)
%     fprintf('\t void volume %.2f(cc)\n', ENGINE_DATA.Vr*1e6)

elseif(strncmp(matrix_type,'f',1))
%     fprintf('Wrapped foil matrix\n')

    ENGINE_DATA.am                                                       =
ENGINE_DATA.foil_thickness*ENGINE_DATA.unrolled_foil_length;
    ENGINE_DATA.ar = ENGINE_DATA.amat - ENGINE_DATA.am;
    ENGINE_DATA.Vr = ENGINE_DATA.ar*ENGINE_DATA.lr;
```

```
    ENGINE_DATA.awgr   =   2*ENGINE_DATA.lr*ENGINE_DATA.unrolled_foil_length   +
awgr0;
    ENGINE_DATA.dr = 4*ENGINE_DATA.Vr/ENGINE_DATA.awgr;
    porosity = ENGINE_DATA.ar/ENGINE_DATA.amat;
%
%     fprintf('\t unrolled foil length: %.3f(m)\n', fl)
%     fprintf('\t foil thickness %.3f(mm)\n',ENGINE_DATA.foil_thickness*1e3)
%     fprintf('\t hydraulic diam %.3f(mm)\n', ENGINE_DATA.dr*1e3)
%     fprintf('\t total wetted area %f(sq.m)\n', ENGINE_DATA.awgr)
%     fprintf('\t void volume %.2f(cc)\n', ENGINE_DATA.Vr*1e6)
%     fprintf('\t porosity %.3f\n', porosity)

elseif(strncmp(matrix_type,'n',1))
%     fprintf('No regenerator matrix\n')

    ENGINE_DATA.ar = ENGINE_DATA.amat;
    ENGINE_DATA.Vr = ENGINE_DATA.ar*ENGINE_DATA.lr;
    ENGINE_DATA.awgr = ENGINE_DATA.awgr0;
    ENGINE_DATA.dr = 4*ENGINE_DATA.Vr/ENGINE_DATA.awgr;
%
%     fprintf('\t hydraulic diam %.3f(mm)\n', ENGINE_DATA.dr*1e3)
%     fprintf('\t total wetted area %f(sq.m)\n', ENGINE_DATA.awgr)
%     fprintf('\t void volume %.2f(cc)\n', ENGINE_DATA.Vr*1e6)
else
%     fprintf('matrix configuration is undefined\n')
end
```

The Isothermal Function Set

derivative.m

```
function dx = derivative(x,dy)
%derivative Take derivative of one variable 'x' with respect to dy
%   Developed Oct 2016 - Shahzeb Mirza
%   Takes 'derivative' assuming the array x is one cycle of a periodic
%   function with respect to y. Uses FORWARD FINITE DIFFERENCE METHOD.
dx = diff(x);
dx(end+1) = x(1) - x(end);
dx = dx/dy;
end
```

isothermal.m

```
function ISOTHERMAL_DATA = isothermal(ENGINE_DATA,crank_inc)
% Isothermal Function that when given an ENGINE_DATA data structure returns
% performance results based on the Ideal Isothermal Model
%   Developed by Shahzeb Mirza, October 2017
% Edited by Connor Speer, October 2017

%{
VARIABLE NAMES AND MEANINGS
theta - Array of sequential crank angles in [rad]

Vc - Compression Space Volume (function of angle) [m^3]
Ve - Expansion Space Volume (function of angle) [m^3]
Vk - Cold Space Volume (const.) [m^3]
Vh - Hot Space Volume (const.) [m^3]
Vr - Regenerator Volume (const.) [m^3]

Tgk - Hot-side GAS Temperature (const.) [K]
Tgh - Cold-side GAS Temperature (const.) [K]
Tr - Regenerator Temperature (const.) [K]

pmean - Initial Fill Pressure (const.) [Pa]
p - Engine Pressure (function of angle) [Pa]

R - Gas Specific Constant [J/kgK]

M - mass of gas
%}

%% Create array of crank increments in RADIANS
theta = linspace(0,2*pi, 360/crank_inc + 1);
dtheta = theta(2)-theta(1);
nEl = length(theta);

%% Call on separate 'volume' function to get volume variations
% dVc, dVe returned here are derivatives with respect to theta
[Vc, Ve, dVc, dVe] = volume(theta, ENGINE_DATA);
```

```
% % dVc and dVe returned here are just the differences between adjacent
% % entries
% dVc = diff(Vc);
% dVc(end+1) = Vc(1) - Vc(end);
% dVe = diff(Ve);
% dVe(end+1) = Ve(1) - Ve(end);

%% Find or Calculate engine CONSTANTS
Vk = ENGINE_DATA.Vk;
Vh = ENGINE_DATA.Vh;
Vr = ENGINE_DATA.Vr;
Tgk = ENGINE_DATA.Tgk;
Tgh = ENGINE_DATA.Tgh;
% pmean = ENGINE_DATA.pmean;
R = ENGINE_DATA.R;

tau = Tgh/Tgk;
deltaT = Tgh - Tgk;
Tr = deltaT/log(tau);

M = ENGINE_DATA.mgas;

% Engine pressure as a function of crank angle
p = M * R ./ ( Vc/Tgk + Vk/Tgk + Vr/Tr + Vh/Tgh + Ve/Tgh );

% Pressure Derivative
dp = derivative(p,dtheta);

% Mass functions
mc = (p .* Vc)./(R .* Tgk);
mk = (p .* Vk)./(R .* Tgk);
mr = (p .* Vr)./(R .* Tr);
mh = (p .* Vh)./(R .* Tgh);
me = (p .* Ve)./(R .* Tgh);

% Mass Derivatives
dmc = derivative(mc,dtheta);
dmk = derivative(mk,dtheta);
dmr = derivative(mr,dtheta);
dmh = derivative(mh,dtheta);
dme = derivative(me,dtheta);

% Mass flows
m_dot_ck = -dmc;
m_dot_kr = -dmc - dmk;
m_dot_rh = dmh + dme;
m_dot_he = dme;

%% Find Work Done by Expansion and Compression Spaces
% Differential Method: Produces accurate results if crank_angle increment
% is extremely fine (~0.01 degrees). Otherwise error is significant.
% Advantage is that it is sensitive to the 'orientation' of the Indicator
% Diagram
dWc = ( p .* dVc * dtheta);
```

```
dWe = ( p .* dVe * dtheta);
dW = dWc + dWe;
Wc = cumsum(dWc);
We = cumsum(dWe);
W = Wc + We;

% Polyarea Method: Accurate Results at almost any crank angle increment,
% but slower than differential method. Is not sensitive to orientation of
% the indicator diagrams
% We = repmat(polyarea(Ve, p),1,nEl);
% Wc = repmat(-polyarea(Vc, p),1,nEl);
% % dWe = derivative(We, dtheta);
% % dWc = derivative(Wc, dtheta);
% % dW = dWe + dWc;
% % Wc = cumsum(dWc);
% % We = cumsum(dWe);
% W = Wc + We;


%% PREALLOCATE ISOTHERMAL_DATA STRUCTURE;
ISOTHERMAL_DATA(nEl).theta  = [];  % Crank angle (radians)

ISOTHERMAL_DATA(nEl).p  = [];  % Pressure (Pa)
ISOTHERMAL_DATA(nEl).Vc = []; % Compression space volume (m^3)
ISOTHERMAL_DATA(nEl).Ve = []; % Expansion space volume (m^3)
ISOTHERMAL_DATA(nEl).mc = []; % Mass of gas in the compression space (kg)
ISOTHERMAL_DATA(nEl).mk = []; % Mass of gas in the cooler (kg)
ISOTHERMAL_DATA(nEl).mr = []; % Mass of gas in the regenerator (kg)
ISOTHERMAL_DATA(nEl).mh = []; % Mass of gas in the heater (kg)
ISOTHERMAL_DATA(nEl).me = []; % Mass of gas in the expansion space (kg)
ISOTHERMAL_DATA(nEl).m_dot_ck = []; % Conditional mass flow compression space
/ cooler (kg/rad)
ISOTHERMAL_DATA(nEl).m_dot_kr  =  [];  %  Conditional  mass  flow  cooler  /
regenerator (kg/rad)
ISOTHERMAL_DATA(nEl).m_dot_rh  =  []; %  Conditional  mass  flow  regenerator  /
heater (kg/rad)
ISOTHERMAL_DATA(nEl).m_dot_he  =  [];  %  Conditional  mass  flow  heater  /
expansion space (kg/rad)
ISOTHERMAL_DATA(nEl).Tgc = [];  % Compression space gas temperature (K)
ISOTHERMAL_DATA(nEl).Tge = [];  % Expansion space gas temperature (K)
ISOTHERMAL_DATA(nEl).Qk = [];  % Heat transferred to the cooler (J)
ISOTHERMAL_DATA(nEl).Qr = [];  % Heat transferred to the regenerator (J)
ISOTHERMAL_DATA(nEl).Qh = [];  % Heat transferred to the heater (J)
ISOTHERMAL_DATA(nEl).Wc = [];  % Work done by the compression space (J)
ISOTHERMAL_DATA(nEl).We = [];  % Work done by the expansion space (J)
ISOTHERMAL_DATA(nEl).W  = [];  % Total work done (WC + WE) (J)
ISOTHERMAL_DATA(nEl).Tgck  =  []; % Conditional  gas  temperature  compression
space / cooler (K)
ISOTHERMAL_DATA(nEl).Tghe  =  []; %  Conditional  gas  temperature  heater  /
expansion space (K)

ISOTHERMAL_DATA(nEl).dTgc  =  []; %  Derivative  of  compression  space  gas
temperature (K/rad)
ISOTHERMAL_DATA(nEl).dTge  =  []; %  Derivative  of  expansion  space  gas
temperature (K/rad)
```

```
ISOTHERMAL_DATA(nEl).dQk = []; % Derivative of heat transferred to the cooler
(J/rad)
ISOTHERMAL_DATA(nEl).dQr  =  []; % Derivative of heat transferred to the
regenerator (J/rad)
ISOTHERMAL_DATA(nEl).dQh = []; % Derivative of heat transferred to the heater
(J/rad)
ISOTHERMAL_DATA(nEl).dWc = []; % Derivative of work done by the compression
space (J/rad)
ISOTHERMAL_DATA(nEl).dWe  =  []; % Derivative of work done by the expansion
space (J/rad)
ISOTHERMAL_DATA(nEl).dW   = []; % Derivative of total work done (WC + WE)
(J/rad)
ISOTHERMAL_DATA(nEl).dp  = []; % Derivative of Pressure (Pa/rad)
ISOTHERMAL_DATA(nEl).dVc  =  []; % Derivative of compression space volume
(m^3/rad)
ISOTHERMAL_DATA(nEl).dVe  =  []; % Derivative of expansion space volume
(m^3/rad)
ISOTHERMAL_DATA(nEl).dmc = []; % Derivative of mass of gas in the compression
space (kg/rad)
ISOTHERMAL_DATA(nEl).dmk  =  []; % Derivative of mass of gas in the cooler
(kg/rad)
ISOTHERMAL_DATA(nEl).dmr = []; % Derivative of mass of gas in the regenerator
(kg/rad)
ISOTHERMAL_DATA(nEl).dmh  =  []; % Derivative of mass of gas in the heater
(kg/rad)
ISOTHERMAL_DATA(nEl).dme = []; % Derivative of mass of gas in the expansion
space (kg/rad)


%% OUTPUT TO ISOTHERMAL_DATA STRUCTURE:
for i = 1:nEl
    ISOTHERMAL_DATA(i).p  = p(i);  % Pressure (Pa)
    ISOTHERMAL_DATA(i).Vc = Vc(i); % Compression space volume (m^3)
    ISOTHERMAL_DATA(i).Ve = Ve(i); % Expansion space volume (m^3)
    ISOTHERMAL_DATA(i).mc  =  mc(i);  % Mass of gas in the compression space
(kg)
    ISOTHERMAL_DATA(i).mk = mk(i); % Mass of gas in the cooler (kg)
    ISOTHERMAL_DATA(i).mr = mr(i); % Mass of gas in the regenerator (kg)
    ISOTHERMAL_DATA(i).mh = mh(i); % Mass of gas in the heater (kg)
    ISOTHERMAL_DATA(i).me = me(i); % Mass of gas in the expansion space (kg)
    ISOTHERMAL_DATA(i).m_dot_ck  =  m_dot_ck(i);  % Conditional mass flow
compression space / cooler (kg/rad)
    ISOTHERMAL_DATA(i).m_dot_kr = m_dot_kr(i); % Conditional mass flow cooler
/ regenerator (kg/rad)
    ISOTHERMAL_DATA(i).m_dot_rh  =  m_dot_rh(i);  % Conditional mass flow
regenerator / heater (kg/rad)
    ISOTHERMAL_DATA(i).m_dot_he = m_dot_he(i); % Conditional mass flow heater
/ expansion space (kg/rad)
    ISOTHERMAL_DATA(i).Tgc = Tgk;  % Compression space gas temperature (K)
    ISOTHERMAL_DATA(i).Tge = Tgh;  % Expansion space gas temperature (K)
    ISOTHERMAL_DATA(i).Qk = 0;  % Heat transferred to the cooler (J)
    ISOTHERMAL_DATA(i).Qr = 0;  % Heat transferred to the regenerator (J)
    ISOTHERMAL_DATA(i).Qh = 0;  % Heat transferred to the heater (J)
    ISOTHERMAL_DATA(i).Wc = Wc(i);  % Work done by the compression space (J)
    ISOTHERMAL_DATA(i).We = We(i);  % Work done by the expansion space (J)
    ISOTHERMAL_DATA(i).W  = W(i);  % Total work done (WC + WE) (J)
```

```
    ISOTHERMAL_DATA(i).Tgck = Tgk; % Conditional gas temperature compression
space / cooler (K)
    ISOTHERMAL_DATA(i).Tghe = Tgh; % Conditional gas temperature heater /
expansion space (K)

    ISOTHERMAL_DATA(i).dTgc = 0; % Derivative of compression space gas
temperature (K/rad)
    ISOTHERMAL_DATA(i).dTge = 0; % Derivative of expansion space gas
temperature (K/rad)
    ISOTHERMAL_DATA(i).dQk = 0; % Derivative of heat transferred to the
cooler (J/rad)
    ISOTHERMAL_DATA(i).dQr = 0; % Derivative of heat transferred to the
regenerator (J/rad)
    ISOTHERMAL_DATA(i).dQh = 0; % Derivative of heat transferred to the
heater (J/rad)
    ISOTHERMAL_DATA(i).dWc = dWc(i); % Derivative of work done by the
compression space (J/rad)
    ISOTHERMAL_DATA(i).dWe = dWe(i); % Derivative of work done by the
expansion space (J/rad)
    ISOTHERMAL_DATA(i).dW = dW(i); % Derivative of total work done (WC + WE)
(J/rad)
    ISOTHERMAL_DATA(i).dp = dp(i); % Derivative of Pressure (Pa/rad)
    ISOTHERMAL_DATA(i).dVc = dVc(i); % Derivative of compression space volume
(m^3/rad)
    ISOTHERMAL_DATA(i).dVe = dVe(i); % Derivative of expansion space volume
(m^3/rad)
    ISOTHERMAL_DATA(i).dmc = dmc(i); % Derivative of mass of gas in the
compression space (kg/rad)
    ISOTHERMAL_DATA(i).dmk = dmk(i); % Derivative of mass of gas in the
cooler (kg/rad)
    ISOTHERMAL_DATA(i).dmr = dmr(i); % Derivative of mass of gas in the
regenerator (kg/rad)
    ISOTHERMAL_DATA(i).dmh = dmh(i); % Derivative of mass of gas in the
heater (kg/rad)
    ISOTHERMAL_DATA(i).dme = dme(i); % Derivative of mass of gas in the
expansion space (kg/rad)
end
ENGINE_DATA.mgas = M; % Total Mass of gas in the engine (kg)
%% PLOT
% theta = theta*180/pi;
% hold on;
% plot(theta,mc);
% plot(theta,mk);
% plot(theta,mr);
% plot(theta,mh);
% plot(theta,me);
% hold off;
% legend('mc','mk','mr','mh','me');
% xlabel('Crank Angle (deg)')
% ylabel('Mass (kg)')
% title('Engine Space Mass')
%
% figure; hold on;
% plot(theta, W);
% plot(theta, We);
% plot(theta, Wc);
% legend('W','We','Wc');
```

```
% xlabel('Crank Angle (deg)')
% ylabel('Work (J)')
% title('Cumulative Work in Cycle')
%
% figure; hold on;
% plot(theta, dW);
% plot(theta, dWe);
% plot(theta, dWc);
% legend('dW','dWe','dWc');
% xlabel('Crank Angle (deg)')
% ylabel('"POWER" (J/rad)')
% title('Power in Cycle')
end
```

The Adiabatic Function Set

adiabatic.m

```
function ADIABATIC_DATA = adiabatic(ENGINE_DATA,crank_inc)
% ideal adiabatic model simulation
% Israel Urieli, 7/6/2002

% Modified by Connor Speer October 2017 to use structures.

% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% crank_inc --> crank angle step size [degrees]

% Output:
% ADIABATIC_DATA --> Structure containing ideal adiabatic model results

% Preallocate space in a structure array for ideal adiabatic model
% results. All derivatives are with respect to crank angle.
ADIABATIC_DATA((360/crank_inc)+1).theta  = [];  % Crank angle (radians)

ADIABATIC_DATA((360/crank_inc)+1).p  = [];  % Pressure (Pa)
ADIABATIC_DATA((360/crank_inc)+1).Vc = []; % Compression space volume (m^3)
ADIABATIC_DATA((360/crank_inc)+1).Ve = []; % Expansion space volume (m^3)
ADIABATIC_DATA((360/crank_inc)+1).mc = []; % Mass of gas in the compression
space (kg)
ADIABATIC_DATA((360/crank_inc)+1).mk = []; % Mass of gas in the cooler (kg)
ADIABATIC_DATA((360/crank_inc)+1).mr = []; % Mass of gas in the regenerator
(kg)
ADIABATIC_DATA((360/crank_inc)+1).mh = []; % Mass of gas in the heater (kg)
ADIABATIC_DATA((360/crank_inc)+1).me = []; % Mass of gas in the expansion
space (kg)
ADIABATIC_DATA((360/crank_inc)+1).m_dot_ck = []; % Conditional mass flow
compression space / cooler (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).m_dot_kr = []; % Conditional mass flow
cooler / regenerator (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).m_dot_rh = []; % Conditional mass flow
regenerator / heater (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).m_dot_he = []; % Conditional mass flow
heater / expansion space (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).Tgc = []; % Compression space gas
temperature (K)
ADIABATIC_DATA((360/crank_inc)+1).Tge = []; % Expansion space gas
temperature (K)
ADIABATIC_DATA((360/crank_inc)+1).Qk = []; % Heat transferred to the cooler
(J)
ADIABATIC_DATA((360/crank_inc)+1).Qr = []; % Heat transferred to the
regenerator (J)
ADIABATIC_DATA((360/crank_inc)+1).Qh = []; % Heat transferred to the heater
(J)
ADIABATIC_DATA((360/crank_inc)+1).Wc = []; % Work done by the compression
space (J)
ADIABATIC_DATA((360/crank_inc)+1).We = []; % Work done by the expansion
space (J)
```

```
ADIABATIC_DATA((360/crank_inc)+1).W  = [];  % Total work done (WC + WE) (J)
ADIABATIC_DATA((360/crank_inc)+1).Tgck  =  []; % Conditional gas temperature
compression space / cooler (K)
ADIABATIC_DATA((360/crank_inc)+1).Tghe  =  []; % Conditional gas temperature
heater / expansion space (K)

ADIABATIC_DATA((360/crank_inc)+1).dTgc  =  []; % Derivative of compression
space gas temperature (K/rad)
ADIABATIC_DATA((360/crank_inc)+1).dTge = []; % Derivative of expansion space
gas temperature (K/rad)
ADIABATIC_DATA((360/crank_inc)+1).dQk = []; % Derivative of heat transferred
to the cooler (J/rad)
ADIABATIC_DATA((360/crank_inc)+1).dQr = []; % Derivative of heat transferred
to the regenerator (J/rad)
ADIABATIC_DATA((360/crank_inc)+1).dQh = []; % Derivative of heat transferred
to the heater (J/rad)
ADIABATIC_DATA((360/crank_inc)+1).dWc = []; % Derivative of work done by the
compression space (J/rad)
ADIABATIC_DATA((360/crank_inc)+1).dWe = []; % Derivative of work done by the
expansion space (J/rad)
ADIABATIC_DATA((360/crank_inc)+1).dW   = []; % Derivative of total work done
(WC + WE) (J/rad)
ADIABATIC_DATA((360/crank_inc)+1).dp  = []; % Derivative of Pressure (Pa/rad)
ADIABATIC_DATA((360/crank_inc)+1).dVc = []; % Derivative of compression space
volume (m^3/rad)
ADIABATIC_DATA((360/crank_inc)+1).dVe  =  []; % Derivative of expansion space
volume (m^3/rad)
ADIABATIC_DATA((360/crank_inc)+1).dmc  =  []; % Derivative of mass of gas in
the compression space (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).dmk  =  []; % Derivative of mass of gas in
the cooler (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).dmr  =  []; % Derivative of mass of gas in
the regenerator (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).dmh  =  []; % Derivative of mass of gas in
the heater (kg/rad)
ADIABATIC_DATA((360/crank_inc)+1).dme  =  []; % Derivative of mass of gas in
the expansion space (kg/rad)

% Set values of the variables in ADIABATIC_DATA at crank angle zero equal
% to zero.
ADIABATIC_DATA(1).theta  = 0;  % Crank angle (radians)

ADIABATIC_DATA(1).p  = 0;  % Pressure (Pa)
ADIABATIC_DATA(1).Vc = 0; % Compression space volume (m^3)
ADIABATIC_DATA(1).Ve = 0; % Expansion space volume (m^3)
ADIABATIC_DATA(1).mc = 0; % Mass of gas in the compression space (kg)
ADIABATIC_DATA(1).mk = 0; % Mass of gas in the cooler (kg)
ADIABATIC_DATA(1).mr = 0; % Mass of gas in the regenerator (kg)
ADIABATIC_DATA(1).mh = 0; % Mass of gas in the heater (kg)
ADIABATIC_DATA(1).me = 0; % Mass of gas in the expansion space (kg)
ADIABATIC_DATA(1).m_dot_ck  =  0;  % Conditional mass flow compression space /
cooler (kg/rad)
ADIABATIC_DATA(1).m_dot_kr = 0; % Conditional mass flow cooler / regenerator
(kg/rad)
ADIABATIC_DATA(1).m_dot_rh = 0; % Conditional mass flow regenerator / heater
(kg/rad)
```

```
ADIABATIC_DATA(1).m_dot_he = 0; % Conditional mass flow heater / expansion
space (kg/rad)
ADIABATIC_DATA(1).Tgc = 0;  % Compression space gas temperature (K)
ADIABATIC_DATA(1).Tge = 0;  % Expansion space gas temperature (K)
ADIABATIC_DATA(1).Qk = 0;  % Heat transferred to the cooler (J)
ADIABATIC_DATA(1).Qr = 0;  % Heat transferred to the regenerator (J)
ADIABATIC_DATA(1).Qh = 0;  % Heat transferred to the heater (J)
ADIABATIC_DATA(1).Wc = 0;  % Work done by the compression space (J)
ADIABATIC_DATA(1).We = 0;  % Work done by the expansion space (J)
ADIABATIC_DATA(1).W  = 0;  % Total work done (WC + WE) (J)
ADIABATIC_DATA(1).Tgck = 0; % Conditional gas temperature compression space /
cooler (K)
ADIABATIC_DATA(1).Tghe = 0; % Conditional gas temperature heater / expansion
space (K)

ADIABATIC_DATA(1).dTgc = 0; % Derivative of compression space gas temperature
(K/rad)
ADIABATIC_DATA(1).dTge = 0; % Derivative of expansion space gas temperature
(K/rad)
ADIABATIC_DATA(1).dQk = 0; % Derivative of heat transferred to the cooler
(J/rad)
ADIABATIC_DATA(1).dQr  =  0;  % Derivative of heat transferred to the
regenerator (J/rad)
ADIABATIC_DATA(1).dQh = 0; % Derivative of heat transferred to the heater
(J/rad)
ADIABATIC_DATA(1).dWc = 0; % Derivative of work done by the compression space
(J/rad)
ADIABATIC_DATA(1).dWe = 0; % Derivative of work done by the expansion space
(J/rad)
ADIABATIC_DATA(1).dW  = 0; % Derivative of total work done (WC + WE) (J/rad)
ADIABATIC_DATA(1).dp  = 0; % Derivative of Pressure (Pa/rad)
ADIABATIC_DATA(1).dVc = 0; % Derivative of compression space volume (m^3/rad)
ADIABATIC_DATA(1).dVe = 0; % Derivative of expansion space volume (m^3/rad)
ADIABATIC_DATA(1).dmc = 0; % Derivative of mass of gas in the compression
space (kg/rad)
ADIABATIC_DATA(1).dmk = 0; % Derivative of mass of gas in the cooler (kg/rad)
ADIABATIC_DATA(1).dmr = 0; % Derivative of mass of gas in the regenerator
(kg/rad)
ADIABATIC_DATA(1).dmh = 0; % Derivative of mass of gas in the heater (kg/rad)
ADIABATIC_DATA(1).dme = 0; % Derivative of mass of gas in the expansion space
(kg/rad)

%=========================================================================
fprintf('============Ideal Adiabatic Analysis==================\n')
fprintf('Cooler Tgk = %.1f[K], Heater Tgh = %.1f[K]\n', [ENGINE_DATA.Tgk],
[ENGINE_DATA.Tgh]);

%% Convergence Criteria
%%%%%%%% CHANGING
epsilon = .0001;  % Allowable error in temperature (K)


% epsilon = .01;  % Allowable error in temperature (K)
max_iteration = 10000;  % Maximum number of iterations to convergence
ninc = 360/crank_inc; % number if integration increments (every degree)
dtheta = 2.0*pi/ninc; % integration increment (radians)
```

```
%% Initial conditions:
% Preallocate space for the variable "y" which contains the 22 variable
% values for one crank angle increment.
y = zeros(1,22);

% Initial temperature at interface between heater and expansion space is
% assumed to be equal to the heater gas temperature.
ADIABATIC_DATA(1).Tghe = ENGINE_DATA.Tgh;
y(18) = ENGINE_DATA.Tgh;

% Initial temperature at interface between compression space and cooler is
% assumed to be equal to the cooler gas temperature.
ADIABATIC_DATA(1).Tgck = ENGINE_DATA.Tgk;
y(17) = ENGINE_DATA.Tgk;

% Initial expansion space temperature is assumed to be equal to the heater
% gas temperature.
ADIABATIC_DATA(1).Tge = ENGINE_DATA.Tgh;
y(2) = ENGINE_DATA.Tge;

% Initial compression space temperature is assumed to be equal to the
% cooler gas temperature.
ADIABATIC_DATA(1).Tgc = ENGINE_DATA.Tgk;
y(1) = ENGINE_DATA.Tgk;

iter = 0; % Initialize iteration counter.
Terror = 10*epsilon; % Initial error to enter the loop

% Iteration loop to cyclic convergence
 while ((Terror >= epsilon)&&(iter < max_iteration))
% cyclic initial conditions
    theta = 0;
    y(3) = 0; % 3 = QK
    y(4) = 0; % 4 = QR
    y(5) = 0; % 5 = QH
    y(6) = 0; % 6 = WC
    y(7) = 0; % 7 = WE
    y(8) = 0; % 8 = W

    fprintf('iteration  %d:  Tc(@360  dg)  =  %.1f[K],  Te(@360  dg)  =
%.1f[K]\n',iter,y(1),y(2))

    % Initialize crank increment counter variable (counter = 1 are the intial
conditions)
    counter = 2;

    for i = 1:1:ninc % For each crank angle degree
        % Make variables for most recently calculated crank angle
        % increment into the vector "y".
        ADIABATIC_DATA(counter-1).Tgc = y(1);
        ADIABATIC_DATA(counter-1).Tge = y(2);
        ADIABATIC_DATA(counter-1).Qk = y(3);
        ADIABATIC_DATA(counter-1).Qr = y(4);
        ADIABATIC_DATA(counter-1).Qh = y(5);
```

```
        ADIABATIC_DATA(counter-1).Wc = y(6);
        ADIABATIC_DATA(counter-1).We = y(7);
        ADIABATIC_DATA(counter-1).W = y(8);
        ADIABATIC_DATA(counter-1).p = y(9);
        ADIABATIC_DATA(counter-1).Vc = y(10);
        ADIABATIC_DATA(counter-1).Ve = y(11);
        ADIABATIC_DATA(counter-1).mc = y(12);
        ADIABATIC_DATA(counter-1).mk = y(13);
        ADIABATIC_DATA(counter-1).mr = y(14);
        ADIABATIC_DATA(counter-1).mh = y(15);
        ADIABATIC_DATA(counter-1).me = y(16);
        ADIABATIC_DATA(counter-1).Tgck = y(17);
        ADIABATIC_DATA(counter-1).Tghe = y(18);
        ADIABATIC_DATA(counter-1).m_dot_ck = y(19);
        ADIABATIC_DATA(counter-1).m_dot_kr = y(20);
        ADIABATIC_DATA(counter-1).m_dot_rh = y(21);
        ADIABATIC_DATA(counter-1).m_dot_he = y(22);

        % Use 4th order Runge-Kutta algorithm to calculate variable values
        % for the next crank angle increment.
       [theta,y,dy]  =  rk4(7,theta,dtheta,y,ENGINE_DATA);  % Calculate  the
variable values for the next crank angle increment

       % Store the newly calculated values in the ADIABATIC_DATA solution
       % structure
       ADIABATIC_DATA(counter).theta = theta;

       ADIABATIC_DATA(counter).Tgc = y(1);
       ADIABATIC_DATA(counter).Tge = y(2);
       ADIABATIC_DATA(counter).Qk = y(3);
       ADIABATIC_DATA(counter).Qr = y(4);
       ADIABATIC_DATA(counter).Qh = y(5);
       ADIABATIC_DATA(counter).Wc = y(6);
       ADIABATIC_DATA(counter).We = y(7);
       ADIABATIC_DATA(counter).W = y(8);
       ADIABATIC_DATA(counter).p = y(9);
       ADIABATIC_DATA(counter).Vc = y(10);
       ADIABATIC_DATA(counter).Ve = y(11);
       ADIABATIC_DATA(counter).mc = y(12);
       ADIABATIC_DATA(counter).mk = y(13);
       ADIABATIC_DATA(counter).mr = y(14);
       ADIABATIC_DATA(counter).mh = y(15);
       ADIABATIC_DATA(counter).me = y(16);
       ADIABATIC_DATA(counter).Tgck = y(17);
       ADIABATIC_DATA(counter).Tghe = y(18);
       ADIABATIC_DATA(counter).m_dot_ck = y(19);
       ADIABATIC_DATA(counter).m_dot_kr = y(20);
       ADIABATIC_DATA(counter).m_dot_rh = y(21);
       ADIABATIC_DATA(counter).m_dot_he = y(22);

       ADIABATIC_DATA(counter).dTgc = dy(1);
       ADIABATIC_DATA(counter).dTge = dy(2);
       ADIABATIC_DATA(counter).dQk = dy(3);
       ADIABATIC_DATA(counter).dQr = dy(4);
       ADIABATIC_DATA(counter).dQh = dy(5);
       ADIABATIC_DATA(counter).dWc = dy(6);
```

```
        ADIABATIC_DATA(counter).dWe = dy(7);
        ADIABATIC_DATA(counter).dW  = dy(8);
        ADIABATIC_DATA(counter).dp  = dy(9);
        ADIABATIC_DATA(counter).dVc = dy(10);
        ADIABATIC_DATA(counter).dVe = dy(11);
        ADIABATIC_DATA(counter).dmc = dy(12);
        ADIABATIC_DATA(counter).dmk = dy(13);
        ADIABATIC_DATA(counter).dmr = dy(14);
        ADIABATIC_DATA(counter).dmh = dy(15);
        ADIABATIC_DATA(counter).dme = dy(16);

        % Increment the crank increment counter variable
        counter = counter + 1;
    end
    % Check if the expansion and compression space temperatures at crank
    % angle 360 match the ones at crank angle 0.
    Terror                    =                    abs([ADIABATIC_DATA(1).Tgc]-
[ADIABATIC_DATA((360/crank_inc)+1).Tgc])...
         +                                         abs([ADIABATIC_DATA(1).Tge]-
[ADIABATIC_DATA((360/crank_inc)+1).Tge]);

    epsilon            =            abs(max([ADIABATIC_DATA.Tgc])            -
min([ADIABATIC_DATA.Tgc]))*0.001;

    iter = iter + 1; % Increment iteration counter variable
  end

 if (iter >= max_iteration)
     fprintf('No convergence within %d iteration\n',max_iteration)
 end


%% Print Out Ideal Adiabatic Analysis Results
% The heat and work quantities are cumulative, so the values at 360 degrees
% represent the total energy transferred in one complete cycle.
eff = ADIABATIC_DATA(end).W / ADIABATIC_DATA(end).Qh;
Qkpower = ADIABATIC_DATA(end).Qk * ENGINE_DATA.freq;
Qrpower = ADIABATIC_DATA(end).Qr * ENGINE_DATA.freq;
Qhpower = ADIABATIC_DATA(end).Qh * ENGINE_DATA.freq;
Wpower = ADIABATIC_DATA(end).W * ENGINE_DATA.freq;

ADIABATIC_DATA(1).eff = eff;
ADIABATIC_DATA(1).Qkpower = Qkpower;
ADIABATIC_DATA(1).Qrpower = Qrpower;
ADIABATIC_DATA(1).Qhpower = Qhpower;
ADIABATIC_DATA(1).Wpower = Wpower;

fprintf('========== ideal adiabatic analysis results ===========\n')
fprintf(' Heat transferred to the cooler: %.2f[W]\n', Qkpower);
fprintf(' Net heat transferred to the regenerator: %.2f[W]\n', Qrpower);
fprintf(' Heat transferred to the heater: %.2f[W]\n', Qhpower);
fprintf(' Indicated power output: %.2f[W]\n', Wpower);
fprintf(' Thermal efficiency : %.1f[%%]\n', eff*100);
fprintf('=====================================================\n')
```

dadiab.m

```
function [y,dy] = dadiab(theta,y,ENGINE_DATA)
% Evaluate ideal adiabatic model derivatives
% Israel Urieli, 7/6/2002
% Arguments:  theta - current cycle angle [radians]
%             y(22) - vector of current variable values
% Returned values:
%             y(22) - updated vector of current variables
%             dy(16) vector of current derivatives
% Function invoked : volume.m

% Modified by Connor Speer October 2017 to use structures.

Vk = ENGINE_DATA.Vk; % cooler void volume [m^3]
Vr = ENGINE_DATA.Vr; % regen void volume [m^3]
Vh = ENGINE_DATA.Vh;% heater void volume [m^3]
R = ENGINE_DATA.R;% gas constant [J/kg.K]
cp = ENGINE_DATA.cp;% specific heat capacity at constant pressure [J/kg.K]
cv = ENGINE_DATA.cv;% specific heat capacity at constant volume [J/kg.K]
gamma = ENGINE_DATA.gamma;% ratio: cp/cv
mgas = ENGINE_DATA.mgas;% total mass of gas in engine [kg]
Tgk = ENGINE_DATA.Tgk; % cooler gas temperature [K]
Tgr = ENGINE_DATA.Tgr; % regenerator gas temperature [K]
Tgh = ENGINE_DATA.Tgh;% heater gas temperature [K]

% Indices of the y, dy vectors:
 TC = 1;  % Compression space temperature (K)
 TE = 2;  % Expansion space temperature (K)
 QK = 3;  % Heat transferred to the cooler (J)
 QR = 4;  % Heat transferred to the regenerator (J)
 QH = 5;  % Heat transferred to the heater (J)
 WC = 6;  % Work done by the compression space (J)
 WE = 7;  % Work done by the expansion space (J)
 W  = 8;  % Total work done (WC + WE) (J)
 P  = 9;  % Pressure (Pa)
 VC = 10; % Compression space volume (m^3)
 VE = 11; % Expansion space volume (m^3)
 MC = 12; % Mass of gas in the compression space (kg)
 MK = 13; % Mass of gas in the cooler (kg)
 MR = 14; % Mass of gas in the regenerator (kg)
 MH = 15; % Mass of gas in the heater (kg)
 ME = 16; % Mass of gas in the expansion space (kg)
 TCK = 17; % Conditional temperature compression space / cooler (K)
 THE = 18; % Conditional temeprature heater / expansion space (K)
 GACK = 19; % Conditional mass flow compression space / cooler (kg/rad)
 GAKR = 20; % Conditional mass flow cooler / regenerator (kg/rad)
 GARH = 21; % Conditional mass flow regenerator / heater (kg/rad)
 GAHE = 22; % Conditional mass flow heater / expansion space (kg/rad)
%=========================================================================

% Volume and volume derivatives:
```

```
 [y(VC),y(VE),dy(VC),dy(VE)] = volume(theta, ENGINE_DATA);

% Pressure and pressure derivatives:
 vot = Vk/Tgk + Vr/Tgr + Vh/Tgh;
 y(P) = (mgas*R/(y(VC)/y(TC) + vot + y(VE)/y(TE)));
 top = -y(P)*(dy(VC)/y(TCK) + dy(VE)/y(THE));
 bottom = (y(VC)/(y(TCK)*gamma) + vot + y(VE)/(y(THE)*gamma));
 dy(P) = top/bottom;

% Mass accumulations and derivatives:
 y(MC) = y(P)*y(VC)/(R*y(TC));
 y(MK) = y(P)*Vk/(R*Tgk);
 y(MR) = y(P)*Vr/(R*Tgr);
 y(MH) = y(P)*Vh/(R*Tgh);
 y(ME) = y(P)*y(VE)/(R*y(TE));
 dy(MC) = (y(P)*dy(VC) + y(VC)*dy(P)/gamma)/(R*y(TCK));
 dy(ME) = (y(P)*dy(VE) + y(VE)*dy(P)/gamma)/(R*y(THE));
 dpop = dy(P)/y(P);
 dy(MK) = y(MK)*dpop;
 dy(MR) = y(MR)*dpop;
 dy(MH) = y(MH)*dpop;

% Mass flow between cells:
 y(GACK) = -dy(MC);
 y(GAKR) = y(GACK) - dy(MK);
 y(GAHE) = dy(ME);
 y(GARH) = y(GAHE) + dy(MH);

% Conditional temperatures between cells:
 y(TCK) = Tgk;
 if(y(GACK)>0)
    y(TCK) = y(TC);
 end
 y(THE) = y(TE);
 if(y(GAHE)>0)
    y(THE) = Tgh;
 end

% 7 derivatives to be integrated by rk4:
% Working space temperatures:
 dy(TC) = y(TC)*(dpop + dy(VC)/y(VC) - dy(MC)/y(MC));
 dy(TE) = y(TE)*(dpop + dy(VE)/y(VE) - dy(ME)/y(ME));

% Energy:
 dy(QK) = Vk*dy(P)*cv/R - cp*(y(TCK)*y(GACK) - Tgk*y(GAKR));
 dy(QR) = Vr*dy(P)*cv/R - cp*(Tgk*y(GAKR) - Tgh*y(GARH));
 dy(QH) = Vh*dy(P)*cv/R - cp*(Tgh*y(GARH) - y(THE)*y(GAHE));
 dy(WC) = y(P)*dy(VC);
 dy(WE) = y(P)*dy(VE);

% Net work done:
 dy(W) = dy(WC) + dy(WE);
 y(W) = y(WC) + y(WE);
```

volume.m

```
function [Vc,Ve,dVc,dVe] = volume(theta, ENGINE_DATA)
% determine working space volume variations and derivatives
% Israel Urieli, 7/6/2002
% Modified 2/14/2010 to include rockerV (rockdrive)
% Modified by Connor Speer October 2017

% Argument:  theta - current cycle angle [radians]
% Returned values:
%   vc, ve - compression, expansion space volumes [m^3]
%   dvc, dve - compression, expansion space volume derivatives

% *** Note: For gamma engines, the total workspace volume is maximum at
% crank angle 0. For alpha engines, the compression space volume is maximum
% at crank angle zero.

engine_type = ENGINE_DATA.engine_type; % Letter indicationg engine layout and
drive mechanism

if(strncmp(engine_type,'s',1)) % Sinusoidal alpha
    [Vc,Ve,dVc,dVe] = sinevol(theta, ENGINE_DATA);
elseif(strncmp(engine_type,'y',1)) % Ross yoke mechanism alpha
    [Vc,Ve,dVc,dVe] = yokevol(theta, ENGINE_DATA);
elseif(strncmp(engine_type,'r',1)) % Ross rocker V-drive alpha
    [Vc,Ve,dVc,dVe] = rockvol(theta, ENGINE_DATA);
elseif(strncmp(engine_type,'g',1)) % Sinusoidal gamma
    [Vc,Ve,dVc,dVe] = gammasinvol(theta, ENGINE_DATA);
elseif(strncmp(engine_type,'x',1)) % Slider-crank mechanism gamma
    [Vc,Ve,dVc,dVe] = gammacrankvol(theta, ENGINE_DATA);
elseif(strncmp(engine_type,'a',1)) % Slider-crank mechanism alpha
    [Vc,Ve,dVc,dVe] = alphacrankvol(theta, ENGINE_DATA);
end
%=============================================================

function [Vc,Ve,dVc,dVe] = sinevol(theta, ENGINE_DATA)
% sinusoidal drive volume variations and derivatives
% Israel Urieli, 7/6/2002
% Argument:  theta - current cycle angle [radians]
% Returned values:
%   vc, ve - compression, expansion space volumes [m^3]
%   dvc, dve - compression, expansion space volume derivatives

Vclc = ENGINE_DATA.Vclc;
Vcle = ENGINE_DATA.Vcle;
Vswc = ENGINE_DATA.Vswc;
Vswe = ENGINE_DATA.Vswe;
alpha = ENGINE_DATA.alpha;

% Vclc Vcle % compression,expansion clearence vols [m^3]
% Vswc Vswe % compression, expansion swept volumes [m^3]
% alpha % phase angle advance of expansion space [radians]
```

```
Vc = Vclc + 0.5*Vswc*(1 + cos(theta));
Ve = Vcle + 0.5*Vswe*(1 + cos(theta + alpha));
dVc = -0.5*Vswc*sin(theta);
dVe = -0.5*Vswe*sin(theta + alpha);
%============================================================

function [Vc,Ve,dVc,dVe] = yokevol(theta, ENGINE_DATA)
% Ross yoke drive volume variations and derivatives
% Israel Urieli, 7/6/2002
% Modified by Connor Speer, October 2017.
% Argument:  theta - current cycle angle [radians]
% Returned values:
%   Vc, Ve - compression, expansion space volumes [m^3]
%   dVc, dVe - compression, expansion space volume derivatives

% compression,expansion clearence vols [m^3]
Vclc = ENGINE_DATA.Vclc;
Vcle = ENGINE_DATA.Vcle;

b1 = ENGINE_DATA.b1; % Ross yoke length (1/2 yoke base) [m]
b2 = ENGINE_DATA.b2; % Ross yoke height [m]
crank = ENGINE_DATA.crank; % crank radius [m]

% area of compression/expansion pistons [m^2]
acomp = ENGINE_DATA.acomp;
aexp = ENGINE_DATA.aexp;

ymin = ENGINE_DATA.ymin; % minimum yoke vertical displacement [m]

sinth = sin(theta);
costh = cos(theta);
bth = (b1^2 - (crank*costh)^2)^0.5;
ye = crank*(sinth + (b2/b1)*costh) + bth;
yc = crank*(sinth - (b2/b1)*costh) + bth;

Ve = vcle + aexp*(ye - ymin);
Vc = vclc + acomp*(yc - ymin);
dVc = acomp*crank*(costh + (b2/b1)*sinth + crank*sinth*costh/bth);
dVe = aexp*crank*(costh - (b2/b1)*sinth + crank*sinth*costh/bth);
%============================================================

function [vc,ve,dvc,dve] = rockvol(theta, ENGINE_DATA)
% Ross Rocker-V drive volume variations and derivatives
% Israel Urieli, 7/6/2002 & Martine Long 2/25/2005
% Argument:  theta - current cycle angle [radians]
% Returned values:
%   vc, ve - compression, expansion space volumes [m^3]
%   dvc, dve - compression, expansion space volume derivatives

global vclc vcle % compression,expansion clearence vols [m^3]
global crank % crank radius [m]
global acomp aexp % area of compression/expansion pistons [m^2]
global conrodc conrode % length of comp/exp piston connecting rods [m]
global ycmax yemax % maximum comp/exp piston vertical displacement [m]
```

```
sinth = sin(theta);
costh = cos(theta);
beth = (conrode^2 - (crank*costh)^2)^0.5;
bcth = (conrodc^2 - (crank*sinth)^2)^0.5;
ye = beth - crank*sinth;
yc = bcth + crank*costh;

ve = vcle + aexp*(yemax - ye);
vc = vclc + acomp*(ycmax - yc);
dvc = acomp*crank*sinth*(crank*costh/bcth + 1);
dve = -aexp*crank*costh*(crank*sinth/beth - 1);


 function [vc,ve,dvc,dve] = gammasinvol(theta, ENGINE_DATA);
% gamma sinusoidal drive volume variations and derivatives
% Added by Connor Speer - January 2017
% Argument:  theta - current cycle angle [radians]
% Returned values:
%   vc, ve - compression, expansion space volumes [m^3]
%   dvc, dve - compression, expansion space volume derivatives

global vclp vcld % piston, displacer clearence vols [m^3]
global vswp vswd % compression, expansion swept volumes [m^3]
global beta % phase angle advance of displacer motion over piston [radians]

%*** Total volume is maximum at theta = 0 for gammas.
 vc   =   vcld   +   vclp   +   (vswd*0.5)*(1   +   ((vswp/vswd)*(1+cos(theta))   -
cos(theta+beta)));
 ve = vcld + (vswd*0.5)*(1 + cos(theta+beta));
 dvc = -(vswd*0.5)*(((vswp/vswd)*sin(theta)) - sin(theta+beta));
 dve = -(vswd*0.5)*sin(theta+beta);
%===============================================================

 function [vc,ve,dvc,dve] = gammacrankvol(theta, ENGINE_DATA)
% gamma crankshaft drive volume variations and derivatives
% Added by Connor Speer - February 2017
% Argument:  theta - current cycle angle [radians]
% Returned values:
%   vc, ve - compression, expansion space volumes [m^3]
%   dvc, dve - compression, expansion space volume derivatives

vclp = ENGINE_DATA.Vclp;
vcld = ENGINE_DATA.Vclp;
Dbore = ENGINE_DATA.Dbore;
Pbore = ENGINE_DATA.Pbore;
Dr1 = ENGINE_DATA.Dr1;
Dr2 = ENGINE_DATA.Dr2;
Dr3 = ENGINE_DATA.Dr3;
Pr1 = ENGINE_DATA.Pr1;
Pr2 = ENGINE_DATA.Pr2;
Pr3 = ENGINE_DATA.Pr3;
beta = ENGINE_DATA.beta_deg*(pi/180);

% vclp vcld % piston, displacer clearence vols [m^3]
% Dbore Pbore % displacer, piston bores [m]
```

```
% Dr1 Pr1 % displacer, piston desaxe offset in [m]
% Dr2 Pr2 % displacer, piston crank length (half stroke) in [m]
% Dr3 Pr3 % displacer, piston connecting rod lengths [m]
% beta % phase angle advance of displacer motion over piston [radians]

%*** Total volume is maximum at theta = 0 for gammas.
Ptheta2 = pi - theta;
Dtheta2 = Ptheta2 - beta;

Dtheta3 = pi - asin((-Dr1+(Dr2*sin(Dtheta2)))/Dr3);
Dr4 = Dr2*cos(Dtheta2) - Dr3*cos(Dtheta3);
Dr4max = sqrt(((Dr2+Dr3)^2)-(Dr1^2));
Dr4min = sqrt(((Dr3-Dr2)^2)-(Dr1^2));
ve = (vcld*0.5) + ((pi/4)*(Dbore^2))*(Dr4max-Dr4);
DVc = (((pi/4)*(Dbore^2))*(Dr4max-Dr4min)) - ve;

Ptheta3 = pi - asin((-Pr1+(Pr2*sin(Ptheta2)))/Pr3);
Pr4 = Pr2*cos(Ptheta2) - Pr3*cos(Ptheta3);
Pr4max = sqrt(((Pr2+Pr3)^2)-(Pr1^2));
PVc = (((pi/4)*(Pbore^2))*(Pr4max-Pr4));
vc = (vcld*0.5) + vclp + DVc + PVc;

dDtheta3          =                    (Dr2.*cos(Dtheta2))./(Dr3.*sqrt(1-(((-
Dr1+(Dr2.*sin(Dtheta2))).Dr3).^2)));
dDr4 = Dr2.*sin(Dtheta2) + Dr3.*sin(Dtheta3).*dDtheta3;
dve = -(pi/4)*(Dbore^2).*(dDr4);

dPtheta3               =               (Pr2.*cos(Ptheta2))./(Pr3.*sqrt(1-(((-
Pr1+(Pr2.*sin(Ptheta2))).Pr3).^2)));
dPr4 = Pr2.*sin(Ptheta2) + Pr3.*sin(Ptheta3).*dPtheta3;
dPVc = -(pi/4)*(Pbore^2).*dPr4;
dDVc = -dve;
dvc = dDVc + dPVc;
%================================================================

 function [vc,ve,dvc,dve] = alphacrankvol(theta, ENGINE_DATA)
% alpha crankshaft drive volume variations and derivatives
% Added by Connor Speer - February 2017
% Argument:  theta - current cycle angle [radians]
% Returned values:
%   vc, ve - compression, expansion space volumes [m^3]
%   dvc, dve - compression, expansion space volume derivatives

vclc = ENGINE_DATA.Vclc;
vcle = ENGINE_DATA.Vcle;
Cbore = ENGINE_DATA.Cbore;
Ebore = ENGINE_DATA.Ebore;
Cr1 = ENGINE_DATA.Cr1;
Cr2 = ENGINE_DATA.Cr2;
Cr3 = ENGINE_DATA.Cr3;
Er1 = ENGINE_DATA.Er1;
Er2 = ENGINE_DATA.Er2;
Er3 = ENGINE_DATA.Er3;
alpha = ENGINE_DATA.alpha;
```

```
% vclc vcle % compression, expansion clearence vols [m^3]
% Cbore Ebore % compression, expansion piston bores [m]
% Cr1 Er1 % compression, expansion desaxe offset in [m]
% Cr2 Er2 % compression, expansion crank length (half stroke) in [m]
% Cr3 Er3 % compression, expansion connecting rod lengths [m]
% alpha % phase angle advance of expansion space [radians]

%*** Compression space volume is maximum at theta = 0 for alphas. Be
% careful defining crank angle 0 if using a desaxe offset.
Ctheta2 = theta - pi;
Etheta2 = Ctheta2 + alpha;

Ctheta3 = pi - asin((-Cr1+(Cr2*sin(Ctheta2)))/Cr3);
Cr4 = Cr2*cos(Ctheta2) - Cr3*cos(Ctheta3);
Cr4max = sqrt(((Cr2+Cr3)^2)-(Cr1^2));
vc = vclc + ((pi/4)*(Cbore^2))*(Cr4max-Cr4);

Etheta3 = pi - asin((-Er1+(Er2*sin(Etheta2)))/Er3);
Er4 = Er2*cos(Etheta2) - Er3*cos(Etheta3);
Er4max = sqrt(((Er2+Er3)^2)-(Er1^2));
ve = vcle + ((pi/4)*(Ebore^2))*(Er4max-Er4);

dCtheta3                =                (-Cr2*cos(Ctheta2))/(Cr3*sqrt(1-(((-
Cr1+(Cr2*sin(Ctheta2)))/Cr3).^2)));
dCr4 = -Cr2*sin(Ctheta2) + Cr3*sin(Ctheta3)*dCtheta3;
dvc = -(pi/4)*(Cbore^2)*(dCr4);

dEtheta3                =                (-Er2*cos(Etheta2))/(Er3*sqrt(1-(((-
Er1+(Er2*sin(Etheta2)))/Er3).^2)));
dEr4 = -Er2*sin(Etheta2) + Er3*sin(Etheta3)*dEtheta3;
dve = -(pi/4)*(Ebore^2)*(dEr4);
%================================================================
```

The Simple Heat Exchanger Model Function Set

simple.m

```
function [SIMPLE_DATA] = simple(ENGINE_DATA,crank_inc)
% simple analysis - including heat transfer and pressure drop effects
% Israel Urieli, 7/22/2002 (modified 12/3/2003 for temp plots)
% Modified 2/7/2010 to include qrloss in hotsim & kolsim
% Modified 2/15/2010 logical reorganization

% Modified by Connor Speer, 2017 to include parasitic losses
% Modified by Connor Speer, October 2017, to use structure system.

%% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% crank_inc --> crank angle step size [degrees]

%% Outputs:
% SIMPLE_DATA --> Structure containing results from the simple model

% Initialize variables
Tgk_old = ENGINE_DATA.Twk; % Initial cooler gas temp = cooler wall temp
Tgh_old = ENGINE_DATA.Twh; % Initial heater gas temp = heater wall temp
epsilon = 0.1; % allowable temperature error bound for cyclic convergence
(originally 1)
Terror = 10*epsilon; % Initial temperature error (to enter loop)

% Store original reference cycle results (no heat exchanger losses)
REF_CYCLE_DATA = adiabatic(ENGINE_DATA,crank_inc); % Perform an ideal
adiabatic simulation


while (Terror>epsilon)
   ADIABATIC_DATA = adiabatic(ENGINE_DATA,crank_inc); % Perform an ideal
adiabatic simulation
   qrloss = regsim(ENGINE_DATA,ADIABATIC_DATA,crank_inc); % Calculate
regenerator enthalpy loss using results from ideal adiabatic simulation.
   Tgh_new =
hotsim(ENGINE_DATA,ADIABATIC_DATA,ENGINE_DATA.Twh,qrloss,crank_inc); %
Calculate new heater mean gas temperature
   Tgk_new =
kolsim(ENGINE_DATA,ADIABATIC_DATA,ENGINE_DATA.Twk,qrloss,crank_inc); %
Calculate new cooler mean gas temperature
   Terror = abs(Tgh_old - Tgh_new) + abs(Tgk_old - Tgk_new);
   Tgh_old = Tgh_new;
   Tgk_old = Tgk_new;
   Tgr = (Tgh_old-Tgk_old)/log(Tgh_old/Tgk_old);
   ENGINE_DATA.Tgh = Tgh_new;
   ENGINE_DATA.Tgk = Tgk_new;
   ENGINE_DATA.Tgr = Tgr;
end

%% Store converged solution in SIMPLE_DATA structure
SIMPLE_DATA = ADIABATIC_DATA;
```

```
SIMPLE_DATA(1).qrloss = qrloss; % Regenerator enthalpy loss per cycle in (J)

SIMPLE_DATA(1).Tgh = Tgh_new; % Converged heater gas temperature (K)
SIMPLE_DATA(1).Tgk = Tgk_new; % Converged cooler gas temperature (K)
SIMPLE_DATA(1).Tgr  =  Tgr;  %  Converged  regenerator  mean  effective  gas
temperature (K)

SIMPLE_DATA(1).W_dot_ref = REF_CYCLE_DATA(end).W*ENGINE_DATA.freq;

%% Print Out Simple Analysis Results
fprintf('============ Simple Analysis Results ============\n');
fprintf('      Regenerator      net      enthalpy      loss:      %.1f[W]\n',
SIMPLE_DATA(1).qrloss*ENGINE_DATA.freq);
fprintf('heater    wall/gas    temperatures:    Twh    =    %.1f[K],    Th    =
%.1f[K]\n',ENGINE_DATA.Twh,SIMPLE_DATA(1).Tgh);
fprintf('cooler    wall/gas    temperatures:    Twk    =    %.1f[K],    Tk    =
%.1f[K]\n',ENGINE_DATA.Twk,SIMPLE_DATA(1).Tgk);
fprintf('Power      lost      due      to      imperfect      heat      exchange:
%.1f[W]\n',SIMPLE_DATA(1).W_dot_ref-(SIMPLE_DATA(end).W*ENGINE_DATA.freq));
% Print out ideal adiabatic analysis results
eff = SIMPLE_DATA(end).W/SIMPLE_DATA(end).Qh; % Thermal efficency
Qkpower  =  SIMPLE_DATA(end).Qk*ENGINE_DATA.freq;    % Heat transferred to the
cooler (W)
Qrpower  =  SIMPLE_DATA(end).Qr*ENGINE_DATA.freq;    % Heat transferred to the
regenerator (W)
Qhpower  =  SIMPLE_DATA(end).Qh*ENGINE_DATA.freq;    % Heat transferred to the
heater (W)
Wpower  =  SIMPLE_DATA(end).W*ENGINE_DATA.freq;       % Total power output for
ideal adiabatic model (W)
fprintf('======= Converged Ideal Adiabatic Analysis  =======\n');
fprintf(' Heat transferred to the cooler: %.2f[W]\n', Qkpower);
fprintf(' Net heat transferred to the regenerator: %.2f[W]\n', Qrpower);
fprintf(' Heat transferred to the heater: %.2f[W]\n', Qhpower);
fprintf(' Total power output: %.2f[W]\n', Wpower);
fprintf(' Ideal adiabatic thermal efficiency: %.1f[%%]\n', eff*100);
```

foilfr.m

```
function [st,ht,fr] = foilfr(ENGINE_DATAmu,N_Re)
% evaluate regenerator wrapped foil stanton number, friction factor
% Israel Urieli, 7/22/2002
% Arguments:
%   mu - gas dynamic viscosity [kg.m/s]
%   N_Re - Reynolds number
%   ENGINE_DATA - structure defining engine geometry and operating point

% Returned values:
%   st - Stanton number
%   ht - heat transfer coefficient [W/m^2.K]
%   fr - Reynolds friction factor ( = re*fanning friction factor)

cp = ENGINE_DATA.cp; % specific heat capacity at constant pressure [J/kg.K]
prandtl = ENGINE_DATA.prandtl; % Prandtl number
```

```
dr = ENGINE_DATA.dr; % regenerator hydraulic diameter [m]

if (N_Re < 2000) % normally laminar flow
   fr = 24;
else
   fr = 0.0791*N_Re^0.75;
end
% From Reynolds simple analogy:
st=fr/(2*N_Re*prandtl);
ht=st*N_Re*cp*mu/dr;
```

hotsim.m

```
function Tgh = hotsim(ENGINE_DATA,ADIABATIC_DATA,Twh,qrloss,crank_inc)
% evaluate heater average heat transfer performance
% Israel Urieli, 7/22/2002
% Modified 2/6/2010 to include regenerator qrloss
% Modified by Connor Speer 2017, October 2017. No globals.

%% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% ADIABATIC_DATA --> structure of adiabatic reference cycle results
% Twh --> heater wall temperature [K]
% qrloss --> heat loss due to imperfect regenerator [J]
% crank_inc --> crank angle step size for numerical integration (degrees)

%% Output:
%   Tgh --> heater average gas temperature [K]

% Preallocate space for loop variables
gah = zeros(1,360/crank_inc);
N_Re = zeros(1,360/crank_inc);

% Calculating the Reynolds number over the cycle
for i = 1:1:(360/crank_inc)+1
   gah(i)              =              (ADIABATIC_DATA(i).m_dot_rh              +
ADIABATIC_DATA(i).m_dot_he)*ENGINE_DATA.omega/2;
   gh = gah(i)/ENGINE_DATA.ah;
   [mu,kgas,N_Re(i)] = reynum(ENGINE_DATA,ENGINE_DATA.Tgh,gh,ENGINE_DATA.dh);
end

% Average and maximum Reynolds number
N_Re_avg = mean(N_Re(1:(360/crank_inc)));
N_Re_max = max(N_Re(1:(360/crank_inc)));

[ht,fr]  =  pipefr(ENGINE_DATA,ENGINE_DATA.dh,mu,N_Re_avg);  %  Heat  transfer
coefficient

% Calculate the average gas temperature in the heater
Tgh = Twh - (ADIABATIC_DATA((360/crank_inc)+1).Qh + qrloss)*...
    ENGINE_DATA.freq/(ht*ENGINE_DATA.awgh); % Heater gas temperature [K]

fprintf('============ Heater Simple analysis ============\n')
```

```
fprintf(' Average Reynolds number : %.1f\n',N_Re_avg)
fprintf(' Maximum Reynolds number : %.1f\n',N_Re_max)
fprintf(' Convective heat transfer coefficient [W/m^2*K] : %.2f\n',ht)
fprintf('heater    wall/gas    temperatures:    Twh    =    %.1f[K],    Tgh    =
%.1f[K]\n',Twh,Tgh);
```

kolsim.m

```
function Tgk = kolsim(ENGINE_DATA,ADIABATIC_DATA,Twk,qrloss,crank_inc)
% evaluate cooler average heat transfer performance
% Israel Urieli, 7/22/2002
% Modified 2/6/2010 to include regenerator qrloss
% Modified by Connor Speer 2017, October 2017. No globals.

%% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% ADIABATIC_DATA --> structure of adiabatic reference cycle
% Twk --> cooler wall temperature [K]
% qrloss --> heat loss due to imperfect regenerator [J]
% crank_inc --> crank angle step size for numerical integration (degrees)

%% Output:
% Tgk --> cooler average gas temperature [K]

% Preallocate space for loop variables
gak = zeros(1,360/crank_inc);
N_Re = zeros(1,360/crank_inc);

% Calculating the Reynolds number over the cycle
for i = 1:1:(360/crank_inc)+1
   gak(i)              =              (ADIABATIC_DATA(i).m_dot_ck              +
ADIABATIC_DATA(i).m_dot_kr)*ENGINE_DATA.omega/2;
   gk = gak(i)/ENGINE_DATA.ak;
   [mu,kgas,N_Re(i)] = reynum(ENGINE_DATA,ENGINE_DATA.Tgk,gk,ENGINE_DATA.dk);
end

% Average and maximum Reynolds number
N_Re_avg = mean(N_Re(1:(360/crank_inc)));
N_Re_max = max(N_Re(1:(360/crank_inc)));

[ht,fr]  =  pipefr(ENGINE_DATA,ENGINE_DATA.dk,mu,N_Re_avg);  %  Heat  transfer
coefficient

Tgk      =      Twk      -      (ADIABATIC_DATA(360/crank_inc+1).Qk      -
qrloss)*ENGINE_DATA.freq/...
    (ht*ENGINE_DATA.awgk); % Cooler gas temperature [K]

fprintf('============ Cooler Simple analysis =============\n')
fprintf(' Average Reynolds number : %.1f\n',N_Re_avg)
fprintf(' Maximum Reynolds number : %.1f\n',N_Re_max)
fprintf(' Heat transfer coefficient [W/m^2*K] : %.2f\n',ht)
fprintf('cooler    wall/gas    temperatures:    Twk    =    %.1f[K],    Tgk    =
%.1f[K]\n',Twk,Tgk);
```

matrixfr.m

```
function [st,fr] = matrixfr(ENGINE_DATA,N_Re)
% evaluate regenerator mesh matrix stanton number, friction factor
% Israel Urieli, 7/22/2002
% Modified by Connor Speer, October 2017. No globals.
% Arguments:
%   N_Re - Reynolds number
%   ENGINE_DATA - structure defining engine geometry and operating point
% Returned values:
%   st - Stanton number
%   fr - Reynolds friction factor ( = re*fanning friction factor)

prandtl = ENGINE_DATA.prandtl; % Prandtl number

% equations taken from Kays & London (1955 edition)
st = 0.46*N_Re^(-0.4)/prandtl; % Stanton Number

fr = 54 + 1.43*N_Re^0.78; % Reynolds friction factor
```

pipefr.m

```
function [ht,fr] = pipefr(ENGINE_DATA,d,mu,N_Re)
% evaluate heat transfer coefficient, Reynolds friction factor
% Israel Urieli, 7/22/2002 (corrected header 2/20/2011)
% Connor Speer, October 2017. No globals.

% Arguments:
%   d - hydraulic diameter [m]
%   mu - gas dynamic viscosity [kg.m/s]
%   re - Reynolds number
% Returned values:
%   ht - heat transfer coefficient [W/m^2.K]
%   fr - Reynolds friction factor ( = re*fanning friction factor)

cp = ENGINE_DATA.cp; % specific heat capacity at constant pressure [J/kg.K]
prandtl = ENGINE_DATA.prandtl; % Prandtl number

% Personal communication with Alan Organ, because of oscillating
% flow, we assume that flow is always turbulent. Use the Blasius
% relation for all Reynolds numbers:
fr = 0.0791*N_Re^0.75;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Modified by Jason Michaud, 6/9/2017
% If the flow is laminar, use the circular pipe laminar friction factor
% equation. If the Reynolds number is between 5*10^3 and 1*10^5 use Blasius
% friction factor equation for fully developed turbulent flow in pipe.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% if re <= 2300
%     fr = 64/re;
```

```
% elseif (3000 <= re) && (re <=200000)
%     fr = 0.0791*re^(-0.25);
% else
%     return
% end

% From Reynolds simple analogy:
ht = fr*mu*cp/(2*d*prandtl);
```

regsim.m

```
function qrloss = regsim(ENGINE_DATA,ADIABATIC_DATA,crank_inc)
% Evaluate the effectiveness and performance of the regenerator
% Israel Urieli, 7/23/2002 - Modified 2/15/2010
% modified 11/27/2010 to include 'no regenerator matrix'
% Modified by Connor Speer, October 2017. No globals.

%% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% ADIABATIC_DATA --> Structure containing ideal adiabatic model results
% crank_inc --> crank angle step size for simulation in [degrees]

%% Output:
% qrloss --> regenerator net enthalpy loss [J]

% Preallocate space for loop variables
N_Re = zeros(1,360/crank_inc);
gar = zeros(1,360/crank_inc);
qreg = zeros(1,360/crank_inc);

% Reynolds number over the cycle
for i = 1:1:(360/crank_inc)+1
    gar(i)              =              (ADIABATIC_DATA(i).m_dot_kr              +
ADIABATIC_DATA(i).m_dot_rh)*ENGINE_DATA.omega/2;
    gr = gar(i)/ENGINE_DATA.ar;
    [mu,kgas,N_Re(i)]                                                          =
reynum(ENGINE_DATA,ENGINE_DATA.Tgr,gr,ENGINE_DATA.dr);
end

% Average and maximum Reynolds number
N_Re_avg = mean(N_Re(1:(360/crank_inc)));
N_Re_max = max(N_Re(1:(360/crank_inc)));

% Stanton number, number of transfer units, regenerator effectiveness
if (strncmp(ENGINE_DATA.regen_matrix_type,'m',1))
    [N_St,fr] = matrixfr(ENGINE_DATA,N_Re_avg);
elseif (strncmp(ENGINE_DATA.regen_matrix_type,'f',1))
    [N_St,ht,fr] = foilfr(ENGINE_DATA,ENGINE_DATA.dr,mu,N_Re_avg);
elseif (strncmp(ENGINE_DATA.regen_matrix_type,'n',1))
    [N_St,ht,fr] = foilfr(ENGINE_DATA,ENGINE_DATA.dr,mu,N_Re_avg);
end

NTU = N_St*ENGINE_DATA.awgr/(2*ENGINE_DATA.ar);
effect = NTU/(NTU + 1);
```

```
%% Calculate qrloss
% Find the amount of heat transferred unidirectionally to the working gas
% from the regenerator in a complete cycle. Use the effectiveness to
% calculate the regenerator enthalpy loss.
qrmin = min([ADIABATIC_DATA.Qr]);
qrmax = max([ADIABATIC_DATA.Qr]);
qrloss = (1 - effect)*(qrmax - qrmin); %[W]

% Regenerator simple analysis results:
fprintf('============ Regenerator Simple analysis ============\n')
fprintf('Average Reynolds number: %.1f\n', N_Re_avg);
fprintf('Maximum Reynolds number: %.1f\n', N_Re_max);
fprintf('Stanton number(Average Re): %.3f\n',N_St);
fprintf('Number of transfer units: %.1f\n',NTU);
fprintf('Regenerator effectiveness : %.3f\n',effect);
fprintf('Regenerator net enthalpy loss: %.1f[W]\n', qrloss*ENGINE_DATA.freq);
```

reynum.m

```
function [mu,kgas,re] = reynum(ENGINE_DATA,t,g,d)
% evaluate dynamic viscosity, thermal conductivity, Reynolds number
% Israel Urieli, 7/22/2002 (mu units correction 2/13/2011)
% Modified by Connor Speer, October 2017. No globals.
% Arguments:
%   t - gas temperature [K]
%   g - mass flux [kg/m^2.s]
%   d - hydraulic diameter [m]
%   ENGINE_DATA - structure defining engine geometry and operating point
% Returned values:
%   mu - gas dynamic viscosity [kg/m.s]
%   kgas - gas thermal conductivity [W/m.K]
%   re - Reynolds number

cp = ENGINE_DATA.cp; % specific heat capacity at constant pressure [J/kg.K]
mu0 = ENGINE_DATA.mu0; % dynamic viscosity at reference temp t0 [kg.m/s]
t0 = ENGINE_DATA.t0; % reference temperature [K]
t_suth = ENGINE_DATA.t_suth; % Sutherland constant [K]
prandtl = ENGINE_DATA.prandtl; % Prandtl number

mu = mu0*(t0 + t_suth)/(t + t_suth)*(t/t0)^1.5;
kgas = cp*mu/prandtl;
re = abs(g)*d/mu;

if(re < 1)
   re = 1;
end
```

The Parasitic Losses Function Set

worksim.m

```
function            [dwork_h,dwork_r,dwork_k,pdropk,pdroph,pdropr]            =
worksim(ENGINE_DATA,REF_CYCLE_DATA,crank_inc)
% Evaluate the pressure drop available work loss [J]
% Israel Urieli, 7/23/2002, Modified by Connor Speer - June 2017
% Modified by Connor Speer, October 2017. No globals.

% Inputs:
% crank_inc --> crank angle step size for simulation in [degrees]

% Outputs:
% dwork --> pressure drop available work loss [J]
% pdropk --> pressure drop across the cooler [Pa]
% pdroph --> pressure drop across the heater [Pa]
% pdropr --> pressure drop across the regenerator [Pa]

dtheta = 2*pi/(360/crank_inc);
dwork_h = 0; % initialize pumping work loss
dwork_r = 0; % initialize pumping work loss
dwork_k = 0; % initialize pumping work loss

% Preallocate space for loop variables
pdropk = zeros(1,360/crank_inc);
pdroph = zeros(1,360/crank_inc);
pdropr = zeros(1,360/crank_inc);
dp = zeros(1,360/crank_inc);
pc = zeros(1,360/crank_inc);
pe = zeros(1,360/crank_inc);
N_Re = zeros(1,360/crank_inc);

for i = 1:1:(360/crank_inc)
    % Cooler
    m_dot_k             =             ([REF_CYCLE_DATA(i).m_dot_ck]             +
[REF_CYCLE_DATA(i).m_dot_kr])*ENGINE_DATA.omega/(2*ENGINE_DATA.ak);    %    mass
flux in [kg/s*m^2]
    [mu,kgas,N_Re(i)]                                                          =
reynum(ENGINE_DATA,ENGINE_DATA.Tgk,m_dot_k,ENGINE_DATA.dk); % Reynolds number
for crank angle increment
    [ht,fr]    =    pipefr(ENGINE_DATA,ENGINE_DATA.dk,mu,N_Re(i));    %    Reynolds
friction factor for each crank angle increment
    pdropk(i)                                                                  =
2*fr*mu*ENGINE_DATA.Vk*m_dot_k*ENGINE_DATA.lk/(REF_CYCLE_DATA(i).mk*ENGINE_DA
TA.dk^2); % Pressure drop across cooler in [Pa].

    % Work Lost due to pressure drop in the cooler
    dwork_k  =  dwork_k  +  dtheta*pdropk(i)*REF_CYCLE_DATA(i).dVe;  %  pumping
work [J]

    % Regenerator
    m_dot_r             =             (REF_CYCLE_DATA(i).m_dot_kr             +
REF_CYCLE_DATA(i).m_dot_rh)*ENGINE_DATA.omega/(2*ENGINE_DATA.ar);
```

```
    [mu,kgas,N_Re(i)]                                                    =
reynum(ENGINE_DATA,ENGINE_DATA.Tgr,m_dot_r,ENGINE_DATA.dr);
    if(strncmp(ENGINE_DATA.regen_matrix_type,'m',1))
         [st,fr] = matrixfr(ENGINE_DATA,N_Re(i));
    elseif (strncmp(ENGINE_DATA.regen_matrix_type,'f',1))
         [st,ht,fr] = foilfr(ENGINE_DATA.dr,mu,N_Re(i));
    end
    pdropr(i)                                                            =
2*fr*mu*ENGINE_DATA.Vr*m_dot_r*ENGINE_DATA.lr/(REF_CYCLE_DATA(i).mr*ENGINE_DA
TA.dr^2);

    % Work Lost due to pressure drop in the regenerator
    dwork_r  =  dwork_r  +  dtheta*pdropr(i)*REF_CYCLE_DATA(i).dVe;  % pumping
work [J]

    % Heater
    m_dot_h             =             (REF_CYCLE_DATA(i).m_dot_rh             +
REF_CYCLE_DATA(i).m_dot_he)*ENGINE_DATA.omega/(2*ENGINE_DATA.ah);
    [mu,kgas,N_Re(i)]                                                    =
reynum(ENGINE_DATA,ENGINE_DATA.Tgh,m_dot_h,ENGINE_DATA.dh);

    [ht,fr] = pipefr(ENGINE_DATA,ENGINE_DATA.dh,mu,N_Re(i));
    pdroph(i)                                                            =
2*fr*mu*ENGINE_DATA.Vh*m_dot_h*ENGINE_DATA.lh./(REF_CYCLE_DATA(i).mh*ENGINE_D
ATA.dh^2);

    % Work Lost due to pressure drop in the heater
    dwork_h  =  dwork_h  +  dtheta*pdroph(i)*[REF_CYCLE_DATA(i).dVe];  % pumping
work [J]

    % Overall Pressure Drop
    dp(i) = pdropk(i) + pdropr(i) + pdroph(i);

    pc(i)  =  REF_CYCLE_DATA(i).p;  % Baseline pressure is defined to be the
compression space pressure.
    pe(i)  =  pc(i)  +  dp(i);  % Expansion space pressure is the compression
space pressure plus the total pressure drop.
end

%% Add one last value to the results so the length matches the other
variables
pdropk((360/crank_inc)+1) = pdropk(1);
pdropr((360/crank_inc)+1) = pdropr(1);
pdroph((360/crank_inc)+1) = pdroph(1);
dp((360/crank_inc)+1) = dp(1);
pc((360/crank_inc)+1) = pc(1);
pe((360/crank_inc)+1) = pe(1);
```

FW_Subfunction_v3.m

```
function             [W_ind,             FW,             W_shaft]             =
FW_Subfunction_v3(theta,P_engine,P_buffer,V_engine,effect, wantPlot)

% Forced Work subfunction updated to run faster without plotting
```

```
% functionality

% Forced Work 2 - Written by Connor Speer, November 2016
% *** Forced Work modificiations - April 2017 - Calynn Stumpf

% *** Modified in June 2017 to work as a subfunction.

% Modified August 2017 by Steven Middleton and Shahzeb Mirza

% See Senft Pg 17 for a definition of forced work.

% Inputs:
% theta --> Vector of engine crank angles, Vmax = 0 deg, [deg]
% P_engine --> Vector of engine pressures corresponding to theta vector [Pa]
% P_buffer --> Vector of buffer pressures corresponding to theta vector [Pa]
% V_engine --> Vector of engine volumes corresponding to theta vector [m^3]
% effect --> Constant mechanism effectiveness
% wantPlot --> True or False

% Outputs:
% W_ind --> Indicated work in [J]
% FW --> Forced work in [J]
% W_shaft --> Shaft work out in [J]

% Notes:
% Orientation of Pressure and Volume arrays must be correct (Indicator
% diagram must 'turn' the desired way

E = effect; % Mechanism effectiveness

P_cycle = P_engine; % Engine Pressure = Cycle Pressure

Vtotal = V_engine; % Engine Volume = Total Volume

W_ind = polyarea(Vtotal,P_cycle); % Calculate Indicated Work by Integrating
the Indicator Diagram

%% Plot Set-Up
set(0,'defaultfigurecolor',[1 1 1])

% For Paper with 1 Column %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 8;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 326;
% height = 275;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% For Connor's Thesis Paper (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 326;
height = 275;

% Font For Figures
font = 'Arial';
font_size = 11;

% To change line width for plots, say "'LineWidth',2"
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Forced Work Calculations
% Calculate the difference b/w cycle and buffer pressure at each point
P_diff = P_cycle - P_buffer;

dV = delta(V_engine);

FW = 0;

if wantPlot
    figure('Position', [x y width height]);
    hold on;
end
```

```
nLines = 350;
spacing = floor(length(dV)/nLines);

% Use DEFINITION of Forced work and Riemann Sums to find final FW
for i = 1:length(dV)
    if (sign(dV(i)) ~= sign(P_diff(i))) % If they are of opposite sign
        FW = FW + abs(P_diff(i)*dV(i));
        if (wantPlot) && (mod(i,spacing) == 0)
            plot([V_engine(i)                    V_engine(i)],[P_engine(i)./1000
P_buffer(i)./1000], 'r','LineWidth',3);
        end
    end
end
W_shaft = (E*W_ind) - (((1/E) - E)*FW); % Shaft Work

if wantPlot
    P1 = plot(V_engine, P_engine./1000,'b','LineWidth',2);
    P2 = plot(V_engine, P_buffer./1000,'k','LineWidth',2);
    xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
    ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size)
    legend([P1 P2],{'Engine Pressure','Buffer Pressure'})
    set(gca,'fontsize',font_size);
    set(gca,'FontName',font)
%     xlim([1.34e-3 1.8e-3])
    ylim([800 1250])
    hold off
end

% Display Indicated Work
Text1 = ['Indicated Cycle Work: ',num2str(W_ind),' J'];
disp(Text1);

% Total Forced Work
Text2 = ['Total Forced Work:    ',num2str(FW),' J'];
disp(Text2);

% Display Shaft Work (See Senft pg 106)
Text3 = ['Shaft Work:           ',num2str(W_shaft),' J'];
disp(Text3);
end
```

delta.m

```
function [ DELTA ] = delta( Vector)
% This function calculates the wrapping derivative of each element in the
% Array (assumes cyclic)
%   Takes an vector, Outputs an vector of the same size

if (isvector(Vector))
    DELTA = Vector*0;
    DELTA(2:end-1) = (Vector(3:end)-Vector(1:end-2));
    DELTA(1) = (Vector(2)-Vector(end));
    DELTA(end) = (Vector(1)-Vector(end-1));
```

```
    DELTA = DELTA/2;
else
    % Array is not a vector
    fprintf('Provided number to delta function is not a vector');

end
end
```

Engine Specific Functions

Chapter_3_Plots.m

```
% Chapter 3 Plots.m - Written by Connor Speer - October 2017

clear,clc,close all;

%% Plot Aesthetics
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% FIGURE 3.X Isothermal Model vs. Adiabatic Model
% % Thermal Source Temperature Variation
% Min_TH = 273 + 25; % Minimum thermal source temp to be simulated in [K].
% Max_TH = 273 + 500; % Maximum thermal source temp to be simulated in [K].
% TH_inc = 10; % Thermal source temp increment in [K].
%
% Vdead = 874.35*1e-6;
%
% % Isothermal Model Set-Up
% crank_inc = 1;
% Model_Code = 1; % Code specifying which model to run.
% Losses_Code = 0; % Code specifying which losses subfunction to run
% part_traj_on_off = 0; % No particle trajectory calculation
%  ENGINE_DATA  =  HTG_ENGINE_DATA;  %  Structure  defining  engine  geometry  and
operating conditions.
%
% counter = 1;
% % counter_max = length(Min_TH:TH_inc:Max_TH);
%
% for TH = Min_TH:TH_inc:Max_TH
%     ENGINE_DATA.Tsource = TH;
%     ENGINE_DATA.Tgh = TH;
%     ENGINE_DATA.Twh = TH;
%     ENGINE_DATA.Tge = TH;
%
%      [ISOTHERMAL_DATA, LOSSES_DATA, PARTICLE_TRAJECTORY_DATA, ENGINE_DATA] =
...
%     sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);
%
```

```
%     ISOTHERMAL_SOL(counter).W_ind = ISOTHERMAL_DATA(end).W;
%                                     ISOTHERMAL_SOL(counter).efficiency       =
ISOTHERMAL_DATA(end).W/ISOTHERMAL_DATA(end).We;
%     ISOTHERMAL_SOL(counter).p = [ISOTHERMAL_DATA.p];
%                 ISOTHERMAL_SOL(counter).Vtotal    =    [ISOTHERMAL_DATA.Ve]    +
[ISOTHERMAL_DATA.Vc] + ...
%         repelem(Vdead,length([ISOTHERMAL_DATA.Ve]));
%
%     counter = counter + 1;
% end
%
% % Adiabatic Model Set-Up
% crank_inc = 1;
% Model_Code = 2; % Code specifying which model to run.
% Losses_Code = 0; % Code specifying which losses subfunction to run
% part_traj_on_off = 0; % No particle trajectory calculation
%  ENGINE_DATA  =  HTG_ENGINE_DATA;  %  Structure  defining  engine  geometry  and
operating conditions.
%
% counter = 1;
% % counter_max = length(Min_TH:TH_inc:Max_TH);
%
% for TH = Min_TH:TH_inc:Max_TH
%     ENGINE_DATA.Tsource = TH;
%     ENGINE_DATA.Tgh = TH;
%     ENGINE_DATA.Twh = TH;
%     ENGINE_DATA.Tge = TH;
%
%      [ADIABATIC_DATA, LOSSES_DATA, PARTICLE_TRAJECTORY_DATA, ENGINE_DATA] =
...
%     sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);
%
%     ADIABATIC_SOL(counter).W_ind = ADIABATIC_DATA(end).W;
%                                     ADIABATIC_SOL(counter).efficiency       =
ADIABATIC_DATA(end).W/ADIABATIC_DATA(end).Qh;
%     ADIABATIC_SOL(counter).p = [ADIABATIC_DATA.p];
%                 ADIABATIC_SOL(counter).Vtotal    =    [ADIABATIC_DATA.Ve]    +
[ADIABATIC_DATA.Vc] + ...
%         repelem(Vdead,length([ADIABATIC_DATA.Ve]));
%
%     counter = counter + 1;
% end
% % Indicated Work vs TH
% figure('Position', [x y 326 275])
% TH_Range = Min_TH:TH_inc:Max_TH;
% TH_Range_Celcius = TH_Range - 273.15;
% hold on
% plot(TH_Range_Celcius, [ISOTHERMAL_SOL.W_ind],'b','LineWidth',2);
% plot(TH_Range_Celcius, [ADIABATIC_SOL.W_ind],'r','LineWidth',2);
% xlabel('Heater Gas Temperature (°C)','FontName',font,'FontSize',font_size);
% ylabel('Indicated Work (J)','FontName',font,'FontSize',font_size);
% ylim([0 inf])
% legend('Isothermal Model','Adiabatic Model','Location','SouthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
% hold off
%
```

```
% % Efficiency vs TH
% figure('Position', [x y 326 275])
% TH_Range = Min_TH:TH_inc:Max_TH;
% TH_Range_Celcius = TH_Range - 273.15;
% hold on
% plot(TH_Range_Celcius, [ISOTHERMAL_SOL.efficiency].*100,'b','LineWidth',2);
% plot(TH_Range_Celcius, [ADIABATIC_SOL.efficiency].*100,'r','LineWidth',2);
% xlabel('Heater Gas Temperature (°C)','FontName',font,'FontSize',font_size);
% ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size);
% ylim([0 inf])
% legend('Isothermal Model','Adiabatic Model','Location','SouthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
% hold off
%
% % High Temperature Indicator Diagram Comparison
% figure('Position', [x y 326 275])
% hold on
%                                                plot([ISOTHERMAL_SOL(end).Vtotal],
[ISOTHERMAL_SOL(end).p]./1000,'b','LineWidth',2);
%                                                 plot([ADIABATIC_SOL(end).Vtotal],
[ADIABATIC_SOL(end).p]./1000,'r','LineWidth',2);
% xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size);
% ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size);
% ylim([700 1400])
% legend('Isothermal Model','Adiabatic Model','Location','NorthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
% hold off
%
% % Low Temperature Indicator Diagram Comparison
% figure('Position', [x y 326 275])
% hold on
%                                                  plot([ISOTHERMAL_SOL(1).Vtotal],
[ISOTHERMAL_SOL(1).p]./1000,'b','LineWidth',2);
%                                                   plot([ADIABATIC_SOL(1).Vtotal],
[ADIABATIC_SOL(1).p]./1000,'r','LineWidth',2);
% xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size);
% ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size);
% legend('Isothermal Model','Adiabatic Model','Location','NorthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
% hold off

% % Mean Pressure Variation
% Min_pmean = 100000; % Minimum mean pressure to be simulated in [Pa].
% Max_pmean = 1000000; % Maximum mean pressure to be simulated in [Pa].
% pmean_inc = 100000; % mean pressure increment in [Pa].
%
% % Isothermal Model Set-Up
% crank_inc = 1;
% Model_Code = 1; % Code specifying which model to run.
% Losses_Code = 0; % Code specifying which losses subfunction to run
% part_traj_on_off = 0; % No particle trajectory calculation
%  ENGINE_DATA  =  HTG_ENGINE_DATA;  %  Structure  defining  engine  geometry  and
operating conditions.
%
```

```
% counter = 1;
% % counter_max = length(Min_TH:TH_inc:Max_TH);
%
% for pmean = Min_pmean:pmean_inc:Max_pmean
%     ENGINE_DATA.pmean = pmean;
%
%      [ISOTHERMAL_DATA, LOSSES_DATA, PARTICLE_TRAJECTORY_DATA, ENGINE_DATA] =
...
%     sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);
%
%     ISOTHERMAL_SOL(counter).W_ind = ISOTHERMAL_DATA(end).W;
%                                     ISOTHERMAL_SOL(counter).efficiency       =
ISOTHERMAL_DATA(end).W/ISOTHERMAL_DATA(end).We;
%
%     counter = counter + 1;
% end
%
% % Adiabatic Model Set-Up
% crank_inc = 1;
% Model_Code = 2; % Code specifying which model to run.
% Losses_Code = 0; % Code specifying which losses subfunction to run
% part_traj_on_off = 0; % No particle trajectory calculation
%  ENGINE_DATA  =  HTG_ENGINE_DATA;  %  Structure  defining  engine  geometry  and
operating conditions.
%
% counter = 1;
% % counter_max = length(Min_TH:TH_inc:Max_TH);
%
% for pmean = Min_pmean:pmean_inc:Max_pmean
%     ENGINE_DATA.pmean = pmean;
%
%      [ADIABATIC_DATA, LOSSES_DATA, PARTICLE_TRAJECTORY_DATA, ENGINE_DATA] =
...
%     sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);
%
%     ADIABATIC_SOL(counter).W_ind = ADIABATIC_DATA(end).W;
%                                     ADIABATIC_SOL(counter).efficiency       =
ADIABATIC_DATA(end).W/ADIABATIC_DATA(end).Qh;
%
%     counter = counter + 1;
% end
%
% figure('Position', [x y width height])
% pmean_Range_kPa = Min_pmean:pmean_inc:Max_pmean;
% hold on
% plot(pmean_Range_kPa, [ISOTHERMAL_SOL.W_ind],'LineWidth',2);
% plot(pmean_Range_kPa, [ADIABATIC_SOL.W_ind],'LineWidth',2);
% xlabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size);
% ylabel('Indicated Work (J)','FontName',font,'FontSize',font_size);
% legend('Isothermal Model','Adiabatic Model','Location','Northwest')
%
% figure('Position', [x y width height])
% pmean_Range_kPa = Min_pmean:pmean_inc:Max_pmean;
% hold on
% plot(pmean_Range_kPa, [ISOTHERMAL_SOL.efficiency].*100,'LineWidth',2);
% plot(pmean_Range_kPa, [ADIABATIC_SOL.efficiency].*100,'LineWidth',2);
% xlabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size);
```

```
% ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size);
% legend('Isothermal Model','Adiabatic Model','Location','Northwest')

%% FIGURE 3.X Regenerator Enthalpy Loss
% % crank_inc = 1; % Crank angle step size for model output [degrees].
% %
% % % Regenerator Porosity Variation
% % Min_porosity = 0.8; % Minimum porosity to be simulated.
% % Max_porosity = 0.99; % Maximum porosity to be simulated.
% % porosity_inc = 0.005; % Porosity increment.
% %
% % Model_Code = 3; % Code specifying which model to run (3 = Simple Model).
% % Losses_Code = 2; % Code specifying which losses subfunction to run
% % part_traj_on_off = 0; % No particle trajectory calculation
% %
% % ENGINE_DATA = HTG_ENGINE_DATA; % Structure defining engine geometry and
operating conditions.
% %
% % % Vary Porosity in 2nd Order Model:
%                     %                       POROSITY_DATA                       =
HTG_2nd_Order_Vary('regen_matrix_porosity',Min_porosity,Max_porosity,porosity
_inc,ENGINE_DATA);
% %
% % % Find porosity at maximum power
% % [Max_Power,max_power_index] = max([POROSITY_DATA.W_dot]);
% % porosity_at_Max_Power = Min_porosity + (max_power_index-1)*porosity_inc;
% %
% % % Find porosity at maximum efficiency
% % [Max_Efficiency,max_efficiency_index] = max([POROSITY_DATA.eff_thermal]);
%  %  porosity_at_Max_Efficiency  =  Min_porosity  +  (max_efficiency_index-
1)*porosity_inc;
% %
% % % Find porosity at stall
% % [Min_Power,free_run_index] = min(abs([POROSITY_DATA.W_dot]));
% % porosity_at_Free_Run = Min_porosity + (free_run_index-1)*porosity_inc;
% %
% % % Power and Efficiency vs. Regenerator Porosity
% % figure('Position', [x y 326 275])
% % porosity_Range = Min_porosity:porosity_inc:Max_porosity;
% % porosity_Range_percent = porosity_Range.*100;
% %
% % porosity_Power_Range = [POROSITY_DATA.W_dot];
% % porosity_Efficiency_Range = [POROSITY_DATA.eff_thermal].*100;
% % %
% % % % Plot Power on the left axis
% % % yyaxis left
% % % plot(porosity_Range_percent,porosity_Power_Range,'r','LineWidth',2)
% % % ylim([0 inf])
% % % ylabel('Power Output (W)','FontName',font,'FontSize',font_size)
% % % set(gca,'ycolor','r')
% % % % Plot efficiency on the right axis
% % % yyaxis right
%                                     %                                       %
plot(porosity_Range_percent,porosity_Efficiency_Range,'b','LineWidth',2)
% % % ylim([0 inf])
% % % ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size)
% % % set(gca,'ycolor','b')
```

```
% % % xlabel('Renerator Porosity (%)','FontName',font,'FontSize',font_size)
% % % legend('Power','Efficiency','Location','South')
% % % set(gca,'fontsize',font_size);
% % % set(gca,'FontName',font)
% %
% % % Regen flow friction and enthalpy Losses vs Porosity
% % figure('Position', [x y 326 275])
% % plot(porosity_Range_percent,[POROSITY_DATA.Q_qrloss],'k','LineWidth',2)
% % ylim([0 inf])
% % ylabel('Enthalpy Loss (W)','FontName',font,'FontSize',font_size)
% % xlabel('Regenerator Porosity (%)','FontName',font,'FontSize',font_size)
% % set(gca,'fontsize',font_size);
% % set(gca,'FontName',font)
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% crank_inc = 1; % Crank angle step size for model output [degrees].
%
% % Regenerator Wire Diameter Variation
% Min_regen_wire = 2e-05; % Minimum regenerator wire diameter to be simulated
in [m].
%  Max_regen_wire  =  15e-05;  %  Maximum  regenerator  wire  diameter  to  be
simulated in [m].
% regen_wire_inc = 0.5e-05; % Regenerator wire diameter increment in [m].
%
% Model_Code = 3; % Code specifying which model to run (3 = Simple Model).
% Losses_Code = 2; % Code specifying which losses subfunction to run
% part_traj_on_off = 0; % No particle trajectory calculation
%
%  ENGINE_DATA  =  HTG_ENGINE_DATA;  %  Structure  defining  engine  geometry  and
operating conditions.
%
% % Vary Porosity in 2nd Order Model:
%                                 REGEN_WIRE_DATA                                =
HTG_2nd_Order_Vary('regen_wire_diameter',Min_regen_wire,Max_regen_wire,regen_
wire_inc,ENGINE_DATA);
%
% % % Power and Efficiency vs. Regenerator Wire Diameter
% % figure('Position', [x y 326 275])
% % regen_wire_Range = Min_regen_wire:regen_wire_inc:Max_regen_wire;
% % regen_wire_Range_mm = regen_wire_Range.*1000;
% %
% % regen_wire_Power_Range = [REGEN_WIRE_DATA.W_dot];
% % regen_wire_Efficiency_Range = [REGEN_WIRE_DATA.eff_thermal].*100;
% %
% % % Plot Power on the left axis
% % yyaxis left
% % plot(regen_wire_Range_mm,regen_wire_Power_Range,'r','LineWidth',2)
% % ylim([0 inf])
% % ylabel('Power Output (W)','FontName',font,'FontSize',font_size)
% % set(gca,'ycolor','r')
% % % Plot efficiency on the right axis
% % yyaxis right
% % plot(regen_wire_Range_mm,regen_wire_Efficiency_Range,'b','LineWidth',2)
% % ylim([0 inf])
% % ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size)
% % set(gca,'ycolor','b')
```

```
%           %           xlabel('Renerator           Wire           Diameter
(mm)','FontName',font,'FontSize',font_size)
% % legend('Power','Efficiency','Location','South')
% % set(gca,'fontsize',font_size);
% % set(gca,'FontName',font)
%
% % Regen enthalpy Losses vs Porosity
% figure('Position', [x y 326 275])
% plot(regen_wire_Range_mm,[REGEN_WIRE_DATA.Q_qrloss],'k','LineWidth',2)
% ylim([0 inf])
% ylabel('Enthalpy Loss (W)','FontName',font,'FontSize',font_size)
%              xlabel('Regenerator               Wire               Diameter
(mm)','FontName',font,'FontSize',font_size)
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)

%% Figure 3.X: Power and Efficiency vs. Frequency Using 2nd Order Model
% crank_inc = 1; % Crank angle step size for model output [degrees].
%
% % Mean Pressure Variation
% Min_freq = 0.5; % Minimum frequency to be simulated in [Hz].
% Max_freq = 12; % Maximum frequency to be simulated in [Hz].
% freq_inc = 0.5; % Frequency increment in [Hz].
%
% Model_Code = 3; % Code specifying which model to run (3 = Simple Model).
% Losses_Code = 2; % Code specifying which losses subfunction to run
% part_traj_on_off = 0; % No particle trajectory calculation
%
% ENGINE_DATA = HTG_ENGINE_DATA; % Structure defining engine geometry and
operating conditions.
% ENGINE_DATA.Pbore = 0.085;
% ENGINE_DATA.V_buffer_max = 0.0032;
% ENGINE_DATA.Vclp = 1.733e-04;
% ENGINE_DATA.GSH_config = 1;

% % Vary Frequency with 85 mm Piston:
%                                 FREQ_DATA                                =
HTG_2nd_Order_Vary('freq',Min_freq,Max_freq,freq_inc,ENGINE_DATA);
%
% % Find mean pressure at maximum power
% [Max_Power,max_power_index] = max([FREQ_DATA.W_dot]);
% freq_at_Max_Power = Min_freq + (max_power_index-1)*freq_inc;
%
% % Find mean pressure at maximum efficiency
% [Max_Efficiency,max_efficiency_index] = max([FREQ_DATA.eff_thermal]);
% freq_at_Max_Efficiency = Min_freq + (max_efficiency_index-1)*freq_inc;
%
% % Find mean pressure at stall
% [Min_Power,free_run_index] = min(abs([FREQ_DATA.W_dot]));
% freq_at_Free_Run = Min_freq + (free_run_index-1)*freq_inc;
%
% % Power and Efficiency vs. Frequency
% figure('Position', [x y 326 275])
% freq_Range = Min_freq:freq_inc:Max_freq;
%
% freq_Power_Range = [FREQ_DATA.W_dot];
```

```
% freq_Efficiency_Range = [FREQ_DATA.eff_thermal].*100;
%
% % Plot Power on the left axis
% yyaxis left
% plot(freq_Range,freq_Power_Range,'r','LineWidth',2)
% ylim([0 inf])
% ylabel('Power Output (W)','FontName',font,'FontSize',font_size)
% set(gca,'ycolor','r')
% % Plot efficiency on the right axis
% yyaxis right
% plot(freq_Range,freq_Efficiency_Range,'b','LineWidth',2)
% ylim([0 inf])
% ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size)
% set(gca,'ycolor','b')
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% legend('Power','Efficiency','Location','South')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
%
% % Normalized Power Losses vs Frequency
% figure('Position', [x y 326 275])
% P_HEX_rel = [FREQ_DATA.P_HEX]./[FREQ_DATA.ref_cycle_power];
% P_mech_rel = [FREQ_DATA.P_mech]./[FREQ_DATA.ref_cycle_power];
% P_flow_h_rel = [FREQ_DATA.P_flow_h]./[FREQ_DATA.ref_cycle_power];
% P_flow_r_rel = [FREQ_DATA.P_flow_r]./[FREQ_DATA.ref_cycle_power];
% P_flow_k_rel = [FREQ_DATA.P_flow_k]./[FREQ_DATA.ref_cycle_power];
% % P_GSH_rel = [FREQ_DATA.P_GSH]./[FREQ_DATA.ref_cycle_power];
% hold on
% plot(freq_Range,P_HEX_rel.*100,'LineWidth',2)
% plot(freq_Range,P_mech_rel.*100,'LineWidth',2)
% plot(freq_Range,P_flow_h_rel.*100,'LineWidth',2)
% plot(freq_Range,P_flow_r_rel.*100,'LineWidth',2)
% plot(freq_Range,P_flow_k_rel.*100,'LineWidth',2)
% % plot(freq_Range,P_GSH_rel.*100,'LineWidth',2)
% hold off
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% ylabel('Normalized Power Losses (%)','FontName',font,'FontSize',font_size)
%   legend('Imperfect   Heat   Transfer','Mechanical   Friction','Heater   Flow
Friction','Regenerator Flow Friction',...
%     'Cooler Flow Friction','Gas Spring Hysteresis','Location','North')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
% ylim([0 100])
% xlim([0 12])
%
% % Flow Friction losses vs Frequency
% figure('Position', [x y 326 275])
% hold on
% plot(freq_Range,[FREQ_DATA.P_flow_h],'r','LineWidth',2)
% plot(freq_Range,[FREQ_DATA.P_flow_r],'g','LineWidth',2)
% plot(freq_Range,[FREQ_DATA.P_flow_k],'b','LineWidth',2)
% hold off
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% ylabel('Flow Friction Losses (W)','FontName',font,'FontSize',font_size)
% legend('Heater','Regenerator','Cooler','Location','NorthWest')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
```

```
%
% % Pressure Drops vs Frequency
% figure('Position', [x y 326 275])
% hold on
% plot([FREQ_DATA(end).pdroph]./1000,'r','LineWidth',2)
% plot([FREQ_DATA(end).pdropr]./1000,'g','LineWidth',2)
% plot([FREQ_DATA(end).pdropk]./1000,'b','LineWidth',2)
% hold off
% xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
% ylabel('Pressure Drop (kPa)','FontName',font,'FontSize',font_size)
% legend('Heater','Regenerator','Cooler','Location','NorthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)

%% Figure 3.X: Power and Efficiency vs. Mean Pressure Using 2nd Order Model
crank_inc = 1; % Crank angle step size for model output [degrees].

% Mean Pressure Variation
Min_pmean = 100000; % Minimum mean pressure to be simulated in [Pa].
Max_pmean = 100000000; % Maximum mean pressure to be simulated in [Pa].
pmean_inc = 1000000; % Mean pressure increment in [Pa].

Model_Code = 3; % Code specifying which model to run (3 = Simple Model).
Losses_Code = 2; % Code specifying which losses subfunction to run
part_traj_on_off = 0; % No particle trajectory calculation

ENGINE_DATA = HTG_ENGINE_DATA; % Structure defining engine geometry and
operating conditions.
ENGINE_DATA.Pbore = 0.085;
ENGINE_DATA.V_buffer_max = 0.0032;
ENGINE_DATA.Vclp = 1.733e-04;
ENGINE_DATA.GSH_config = 1;

% Vary Frequency with 85 mm Piston:
PMEAN_DATA                                                    =
HTG_2nd_Order_Vary('pmean',Min_pmean,Max_pmean,pmean_inc,ENGINE_DATA);

% Find mean pressure at maximum power
[Max_Power,max_power_index] = max([PMEAN_DATA.W_dot]);
pmean_at_Max_Power = Min_pmean + (max_power_index-1)*pmean_inc;

% Find mean pressure at maximum efficiency
[Max_Efficiency,max_efficiency_index] = max([PMEAN_DATA.eff_thermal]);
pmean_at_Max_Efficiency = Min_pmean + (max_efficiency_index-1)*pmean_inc;

% Find mean pressure at stall
[Min_Power,free_run_index] = min(abs([PMEAN_DATA.W_dot]));
pmean_at_Free_Run = Min_pmean + (free_run_index-1)*pmean_inc;

% Power and Efficiency vs. Mean Pressure
figure('Position', [x y 326 275])
pmean_Range = Min_pmean:pmean_inc:Max_pmean;
pmean_Range_kPa = pmean_Range./1000;

pmean_Power_Range = [PMEAN_DATA.W_dot];
```

```
pmean_Efficiency_Range = [PMEAN_DATA.eff_thermal].*100;

% Plot Power on the left axis
yyaxis left
plot(pmean_Range_kPa,pmean_Power_Range,'r','LineWidth',2)
ylim([0 inf])
ylabel('Power Output (W)','FontName',font,'FontSize',font_size)
set(gca,'ycolor','r')
% Plot efficiency on the right axis
yyaxis right
plot(pmean_Range_kPa,pmean_Efficiency_Range,'b','LineWidth',2)
ylim([0 inf])
ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size)
set(gca,'ycolor','b')
xlabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
legend('Power','Efficiency','Location','South')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

% Normalized Power Losses vs Mean Pressure
figure('Position', [x y 326 275])
P_HEX_rel = [PMEAN_DATA.P_HEX]./[PMEAN_DATA.ref_cycle_power];
P_mech_rel = [PMEAN_DATA.P_mech]./[PMEAN_DATA.ref_cycle_power];
P_flow_h_rel = [PMEAN_DATA.P_flow_h]./[PMEAN_DATA.ref_cycle_power];
P_flow_r_rel = [PMEAN_DATA.P_flow_r]./[PMEAN_DATA.ref_cycle_power];
P_flow_k_rel = [PMEAN_DATA.P_flow_k]./[PMEAN_DATA.ref_cycle_power];
% P_GSH_rel = [PMEAN_DATA.P_GSH]./[PMEAN_DATA.ref_cycle_power];
hold on
plot(pmean_Range_kPa,P_HEX_rel.*100,'LineWidth',2)
plot(pmean_Range_kPa,P_mech_rel.*100,'LineWidth',2)
plot(pmean_Range_kPa,P_flow_h_rel.*100,'LineWidth',2)
plot(pmean_Range_kPa,P_flow_r_rel.*100,'LineWidth',2)
plot(pmean_Range_kPa,P_flow_k_rel.*100,'LineWidth',2)
% plot(pmean_Range_kPa,P_GSH_rel.*100,'LineWidth',2)
hold off
xlabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
ylabel('Normalized Power Losses (%)','FontName',font,'FontSize',font_size)
legend('Imperfect   Heat   Transfer','Mechanical   Friction','Heater   Flow
Friction','Regenerator Flow Friction',...
    'Cooler Flow Friction','Gas Spring Hysteresis','Location','North')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
ylim([0 100])

%% Figure 3.X: Power and Efficiency vs. Thermal Source Temperature Using 2nd
Order Model
% % Thermal Source Temperature Variation
% Min_TH = 273 + 50; % Minimum thermal source temperature to be simulated in
[K].
% Max_TH = 273 + 1000; % Maximum thermal source temperature to be simulated
in [K].
% TH_inc = 50; % Thermal source temperature increment in [K].
%
% % Plot 1 - Power and Efficiency vs. Hot Source Temperature, As-Built
% ENGINE_DATA = HTG_ENGINE_DATA;
% ENGINE_DATA.Pbore = 0.085;
```

```
% ENGINE_DATA.V_buffer_max = 0.0032;
% ENGINE_DATA.Vclp = 1.733e-04;
% ENGINE_DATA.GSH_config = 1;
%
% counter = 1;
% for TH = Min_TH:TH_inc:Max_TH
%     ENGINE_DATA.Tsource = TH;
%     ENGINE_DATA.Tge = TH;
%     ENGINE_DATA.Tgh = TH;
%     ENGINE_DATA.Twh = TH;
%
%     [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
%         = HTG_2nd_Order(ENGINE_DATA);
%
%     SOL(counter).second_order_power = SECOND_ORDER_DATA.W_dot;
%     SOL(counter).second_order_efficiency = SECOND_ORDER_DATA.eff_thermal;
%                                         SOL(counter).ref_cycle_power        =
[REF_CYCLE_DATA(end).W].*[ENGINE_DATA.freq];
%                       SOL(counter).P_HEX    =    REF_CYCLE_DATA(1).W_dot_ref-
(REF_CYCLE_DATA(end).W*ENGINE_DATA.freq);
%     SOL(counter).P_mech = LOSSES_DATA.P_mech;
%     SOL(counter).P_flow_h = LOSSES_DATA.P_flow_h;
%     SOL(counter).P_flow_r = LOSSES_DATA.P_flow_r;
%     SOL(counter).P_flow_k = LOSSES_DATA.P_flow_k;
%     SOL(counter).P_GSH = LOSSES_DATA.P_GSH;
%     counter = counter + 1;
% end
%
% figure('Position', [x y 326 275])
%
% TH_Range = Min_TH:TH_inc:Max_TH;
% TH_Range_Celcius = TH_Range - 273.15;
%
% % Plot Power on the left axis
% yyaxis left
% plot(TH_Range_Celcius,[SOL.second_order_power],'r','LineWidth',2)
% ylim([0 inf])
% ylabel('Power Output (W)','FontName',font,'FontSize',font_size)
% set(gca,'ycolor','r')
% % Plot efficiency on the right axis
% yyaxis right
% plot(TH_Range_Celcius,[SOL.second_order_efficiency]*100,'b','LineWidth',2)
% ylim([0 inf])
% ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size)
% set(gca,'ycolor','b')
%
%                xlabel('Heater                Wall                Temperature
(\circC)','FontName',font,'FontSize',font_size)
% legend('Power','Efficiency','Location','SouthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
%
% % Plot 2 - Relative Power Losses As-Built
% P_HEX_rel = [SOL.P_HEX]./[SOL.ref_cycle_power];
% P_mech_rel = [SOL.P_mech]./[SOL.ref_cycle_power];
% P_flow_h_rel = [SOL.P_flow_h]./[SOL.ref_cycle_power];
% P_flow_r_rel = [SOL.P_flow_r]./[SOL.ref_cycle_power];
```

```
% P_flow_k_rel = [SOL.P_flow_k]./[SOL.ref_cycle_power];
% % P_GSH_rel = [SOL.P_GSH]./[SOL.ref_cycle_power];
%
% figure('Position', [x y 326 275])
% hold on
% plot(TH_Range_Celcius,P_HEX_rel.*100,'LineWidth',2)
% plot(TH_Range_Celcius,P_mech_rel.*100,'LineWidth',2)
% plot(TH_Range_Celcius,P_flow_h_rel.*100,'LineWidth',2)
% plot(TH_Range_Celcius,P_flow_r_rel.*100,'LineWidth',2)
% plot(TH_Range_Celcius,P_flow_k_rel.*100,'LineWidth',2)
% % plot(TH_Range_Celcius,P_GSH_rel.*100,'LineWidth',2)
% ylim([0 100])
% xlim([200 1000])
% hold off
%                 xlabel('Heater                Wall                Temperature
(\circC)','FontName',font,'FontSize',font_size)
% ylabel({'Relative Power Losses (%)'},'FontName',font,'FontSize',font_size)
%   legend('Imperfect   Heat   Transfer','Mechanical   Friction','Heater   Flow
Friction','Regenerator Flow Friction',...
%     'Cooler Flow Friction','Gas Spring Hysteresis','Location','NorthEast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
```

Chapter_4_Model_Plots.m

```
% Chapter 4 Plots.m - Written by Connor Speer - October 2017

clear,clc,close all;


%% Input Parameters
% crank_inc = 1; % Crank angle step size for model output [degrees].

% Engine Hot Source Temperature Variation
Min_TH = 373.15; % Minimum hot source temperature to be simulated in [K].
Max_TH = 773.15; % Maximum hot source temperature to be simulated in [K].
TH_Increment = 100; % Hot source temperature increment in [K].


%% Plot Aesthetics
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
```

```
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 326;
% height = 275;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% % For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% Pro Tips %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% To change line width for plots, say "'LineWidth',2".

% To include math symbols with their own font size and font type in the
% axes labels, do this:
%  xlabel({'\fontsize{11} Engine Speed \fontsize{11} \fontname{Cambria Math}
\omega \fontsize{11} \fontname{Times New Roman} [RPM]'});

% Here is the degree symbol in case you need it °C. Alternatively, use
% \circ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Plot 1 - Power and Efficiency vs. Hot Source Temperature, As-Built
ENGINE_DATA = HTG_ENGINE_DATA;
ENGINE_DATA.Pbore = 0.085;
ENGINE_DATA.V_buffer_max = 0.0032;
ENGINE_DATA.Vclp = 1.733e-04;
ENGINE_DATA.GSH_config = 1;

counter = 1;
for TH = Min_TH:TH_Increment:Max_TH
    ENGINE_DATA.Tsource = TH;
    ENGINE_DATA.Tge = TH;
    ENGINE_DATA.Tgh = TH;
    ENGINE_DATA.Twh = TH;
```

```
    [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
        = HTG_2nd_Order(ENGINE_DATA);

    SOL(counter).second_order_power = SECOND_ORDER_DATA.W_dot;
    SOL(counter).second_order_efficiency = SECOND_ORDER_DATA.eff_thermal;
    SOL(counter).ref_cycle_power                                          =
[REF_CYCLE_DATA(end).W].*[ENGINE_DATA.freq];
    SOL(counter).P_HEX             =             REF_CYCLE_DATA(1).W_dot_ref-
(REF_CYCLE_DATA(end).W*ENGINE_DATA.freq);
    SOL(counter).P_mech = LOSSES_DATA.P_mech;
    SOL(counter).P_flow_h = LOSSES_DATA.P_flow_h;
    SOL(counter).P_flow_r = LOSSES_DATA.P_flow_r;
    SOL(counter).P_flow_k = LOSSES_DATA.P_flow_k;
    SOL(counter).P_GSH = LOSSES_DATA.P_GSH;
    counter = counter + 1;
end

figure('Position', [x y width height])

TH_Range = Min_TH:TH_Increment:Max_TH;
TH_Range_Celcius = TH_Range - 273.15;

% Plot Power on the left axis
yyaxis left
plot(TH_Range_Celcius,[SOL.second_order_power],'LineWidth',2)
ylim([0 inf])
ylabel('Power Output,    (W)','FontName',font,'FontSize',font_size)
% Plot efficiency on the right axis
yyaxis right
plot(TH_Range_Celcius,[SOL.second_order_efficiency]*100,'LineWidth',2)
ylim([0 inf])
ylabel('Thermal Efficiency,    (%)','FontName',font,'FontSize',font_size)

xlabel('Thermal                        Source                        Temperature,
(^\circC)','FontName',font,'FontSize',font_size)
legend('Power','Efficiency','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

%% Plot 2 - Relative Power Losses As-Built
P_mech_rel = [SOL.P_mech]./[SOL.ref_cycle_power];
P_flow_h_rel = [SOL.P_flow_h]./[SOL.ref_cycle_power];
P_flow_r_rel = [SOL.P_flow_r]./[SOL.ref_cycle_power];
P_flow_k_rel = [SOL.P_flow_k]./[SOL.ref_cycle_power];
P_GSH_rel = [SOL.P_GSH]./[SOL.ref_cycle_power];
P_HEX_rel = [SOL.P_HEX]./[SOL.ref_cycle_power];

figure('Position', [x y width height])
hold on
plot(TH_Range_Celcius,P_mech_rel.*100,'LineWidth',2,'Color','m')
plot(TH_Range_Celcius,P_flow_h_rel.*100,'LineWidth',2,'Color','r')
plot(TH_Range_Celcius,P_flow_r_rel.*100,'LineWidth',2,'Color','g')
plot(TH_Range_Celcius,P_flow_k_rel.*100,'LineWidth',2,'Color','b')
plot(TH_Range_Celcius,P_GSH_rel.*100,'LineWidth',2,'Color','k')
```

```
plot(TH_Range_Celcius,P_HEX_rel.*100,'LineWidth',2,'Color','c')
% xlim([100 500])
% ylim([0 350])
hold off
xlabel('Thermal                    Source                    Temperature,
(^\circC)','FontName',font,'FontSize',font_size)
ylabel({'Relative     Power     Losses';'(%     of     Reference     Cycle
Power)'},'FontName',font,'FontSize',font_size)
legend('Mechanical   Friction','Heater   Flow   Friction','Regenerator   Flow
Friction'...
    ,'Cooler Flow Friction','Crankcase Gas Spring Hysteresis','Imperfect Heat
Transfer','Position',[0.1 50 550 400])
set(gca,'fontsize',font_size);
set(gca,'FontName',font)


%% Plot 3 - Power and Efficiency vs. Hot Source Temperature, All Mods
ENGINE_DATA = HTG_ENGINE_DATA;
ENGINE_DATA.Pbore = 0.044;
ENGINE_DATA.V_buffer_max = 0.0032 + 0.004633333;
ENGINE_DATA.Vclp = 1.733e-04 - 0.000132;
ENGINE_DATA.GSH_config = 4;

counter = 1;
for TH = Min_TH:TH_Increment:Max_TH
    ENGINE_DATA.Tsource = TH;
    ENGINE_DATA.Tge = TH;
    ENGINE_DATA.Tgh = TH;
    ENGINE_DATA.Twh = TH;

    [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
        = HTG_2nd_Order(ENGINE_DATA);

    SOL(counter).second_order_power = SECOND_ORDER_DATA.W_dot;
    SOL(counter).second_order_efficiency = SECOND_ORDER_DATA.eff_thermal;
    SOL(counter).ref_cycle_power                                          =
[REF_CYCLE_DATA(end).W].*[ENGINE_DATA.freq];
    SOL(counter).P_HEX             =             REF_CYCLE_DATA(1).W_dot_ref-
(REF_CYCLE_DATA(end).W*ENGINE_DATA.freq);
    SOL(counter).P_mech = LOSSES_DATA.P_mech;
    SOL(counter).P_flow_h = LOSSES_DATA.P_flow_h;
    SOL(counter).P_flow_r = LOSSES_DATA.P_flow_r;
    SOL(counter).P_flow_k = LOSSES_DATA.P_flow_k;
    SOL(counter).P_GSH = LOSSES_DATA.P_GSH;
    counter = counter + 1;
end

figure('Position', [x y width height])

TH_Range = Min_TH:TH_Increment:Max_TH;
TH_Range_Celcius = TH_Range - 273.15;

% Plot Power on the left axis
yyaxis left
plot(TH_Range_Celcius,[SOL.second_order_power],'LineWidth',2)
```

```
ylim([0 inf])
ylabel('Power Output,     (W)','FontName',font,'FontSize',font_size)
% Plot efficiency on the right axis
yyaxis right
plot(TH_Range_Celcius,[SOL.second_order_efficiency]*100,'LineWidth',2)
ylim([0 inf])
ylabel('Thermal Efficiency,     (%)','FontName',font,'FontSize',font_size)

xlabel('Thermal                          Source                          Temperature,
(^\circC)','FontName',font,'FontSize',font_size)
legend('Power','Efficiency','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

%% Plot 2 - Relative Power Losses All Mods
P_mech_rel = [SOL.P_mech]./[SOL.ref_cycle_power];
P_flow_h_rel = [SOL.P_flow_h]./[SOL.ref_cycle_power];
P_flow_r_rel = [SOL.P_flow_r]./[SOL.ref_cycle_power];
P_flow_k_rel = [SOL.P_flow_k]./[SOL.ref_cycle_power];
P_GSH_rel = [SOL.P_GSH]./[SOL.ref_cycle_power];
P_HEX_rel = [SOL.P_HEX]./[SOL.ref_cycle_power];

figure('Position', [x y width height])
hold on
plot(TH_Range_Celcius,P_mech_rel.*100,'LineWidth',2,'Color','m')
plot(TH_Range_Celcius,P_flow_h_rel.*100,'LineWidth',2,'Color','r')
plot(TH_Range_Celcius,P_flow_r_rel.*100,'LineWidth',2,'Color','g')
plot(TH_Range_Celcius,P_flow_k_rel.*100,'LineWidth',2,'Color','b')
plot(TH_Range_Celcius,P_GSH_rel.*100,'LineWidth',2,'Color','k')
plot(TH_Range_Celcius,P_HEX_rel.*100,'LineWidth',2,'Color','c')
% xlim([100 500])
% ylim([0 350])
hold off
xlabel('Thermal                          Source                          Temperature,
(^\circC)','FontName',font,'FontSize',font_size)
ylabel({'Relative      Power      Losses';'(%      of      Reference      Cycle
Power)'},'FontName',font,'FontSize',font_size)
legend('Mechanical    Friction','Heater    Flow    Friction','Regenerator    Flow
Friction'...
    ,'Cooler Flow Friction','Crankcase Gas Spring Hysteresis','Imperfect Heat
Transfer','Position',[0.1 50 550 400])
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

%% Optimum Piston Size for 95/5 Temperature Difference
% Input Parameters
Tgh = 273 + 95; %(K)
Tgk = 273 + 5; %(K)
Vmin = (874.35 + 542.87)*1e-6; %(m^3)
Vswd = 542.87*1e-6; %(m^3)
Sp = 0.075; %(m)

% Kolin's Formula
Vswp_Kolin = (Vmin*(Tgh-Tgk))/1100; %(m^3)

Pbore_Kolin = sqrt((4/pi)*Vswp_Kolin); %(m)
```

```
% Connor's Equal Pressure Swings Method
Tgr = (Tgh-Tgk)/log(Tgh/Tgk);

% Mean temperature (displacer at mid-stroke)
Tmean = (Vmin)/(((0.5*Vswd)/Tgh)+((Vh)/Tgh)+ ...
     ((Vr)/Tgr)+((Vk)/Tgk)+((0.5*Vswd)/Tgk));

Tfactor = (Tgh-Tgk)/Tmean; % Temperature difference factor [dimensionless].

C1 = 0.25; % Polynomial coefficient 1.
C2 = Vmin; % Polynomial coefficient 2.
C3 = (Vmin^2) - (Tfactor*Vswd*Vmin); % Polynomial coefficient 3.
C4 = -Tfactor*Vswd*(Vmin^2); % Polynomial coefficient 4.

p = [C1 C2 C3 C4];
r = roots(p);       % Find roots of the polynomial.

Sort_roots = sort(r); % Order roots from smallest to largest.
Vswp = Sort_roots(3); % Choose the largest root to be the piston swept vol.

Pbore_Connor = sqrt((4*Vswp)/(pi*Sp)); % Piston diameter in [mm].

% Second Order Model Optimization
ENGINE_DATA = HTG_ENGINE_DATA;

ENGINE_DATA.freq = 2;

ENGINE_DATA.V_buffer_max = 0.0032 + 0.004633333;
ENGINE_DATA.Vclp = 1.733e-04 - 0.000132;
ENGINE_DATA.GSH_config = 4;

ENGINE_DATA.Tsource = 273 + 95;
ENGINE_DATA.Tge = 273 + 95;
ENGINE_DATA.Tgh = 273 + 95;
ENGINE_DATA.Twh = 273 + 95;

ENGINE_DATA.Tsink = 273 + 5;
ENGINE_DATA.Tgc = 273 + 5;
ENGINE_DATA.Tgk = 273 + 5;
ENGINE_DATA.Twk = 273 + 5;

counter = 1;
Pbore = 0.01:0.001:0.09;
for Dp = Pbore
    ENGINE_DATA.Pbore = Dp;

    [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
        = HTG_2nd_Order(ENGINE_DATA);

    SOL(counter).Power_2nd_order = SECOND_ORDER_DATA.W_dot;
    SOL(counter).Efficiency_2nd_order = SECOND_ORDER_DATA.eff_thermal;
    SOL(counter).ref_cycle_power                                        =
[REF_CYCLE_DATA(end).W].*[ENGINE_DATA.freq];
```

```
    SOL(counter).REF_CYCLE_DATA = REF_CYCLE_DATA;
    SOL(counter).P_HEX              =              REF_CYCLE_DATA(1).W_dot_ref-
(REF_CYCLE_DATA(end).W*ENGINE_DATA.freq);
    SOL(counter).P_mech = LOSSES_DATA.P_mech;
    SOL(counter).P_flow_h = LOSSES_DATA.P_flow_h;
    SOL(counter).P_flow_r = LOSSES_DATA.P_flow_r;
    SOL(counter).P_flow_k = LOSSES_DATA.P_flow_k;
    SOL(counter).P_GSH = LOSSES_DATA.P_GSH;
    counter = counter + 1;
end

figure('Position', [x y width height])
% Plot Power on the left axis
yyaxis left
plot(Pbore,[SOL.Power_2nd_order],'LineWidth',2)
ylabel('Power Output,    (W)','FontName',font,'FontSize',font_size)
% Plot efficiency on the right axis
yyaxis right
plot(Pbore,[SOL.Efficiency_2nd_order]*100,'LineWidth',2)
ylabel('Thermal Efficiency,    (%)','FontName',font,'FontSize',font_size)

xlabel('Piston Diameter,     (m)','FontName',font,'FontSize',font_size)
legend('Power','Efficiency','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

[Max_power,Max_power_index] = max([SOL.Power_2nd_order]);
Pbore_2nd_order_power = Pbore(Max_power_index);

[Max_efficiency,Max_efficiency_index] = max([SOL.Efficiency_2nd_order]);
Pbore_2nd_order_eff = Pbore(Max_efficiency_index);

figure('Position', [x y width height])
hold on
plot(Pbore,[SOL.ref_cycle_power])
plot(Pbore,[SOL.P_HEX])
plot(Pbore,[SOL.P_mech])
plot(Pbore,[SOL.P_flow_h])
plot(Pbore,[SOL.P_flow_r])
plot(Pbore,[SOL.P_flow_k])
plot(Pbore,[SOL.P_GSH])
hold off
xlabel('Piston Diameter,     (m)','FontName',font,'FontSize',font_size)
ylabel('Power Losses','FontName',font,'FontSize',font_size)
legend('Reference   Cycle   Power','Imperfect   Heat   Transfer','Mechanical
Friction','Heater Flow Friction','Regenerator Flow Friction'...
    ,'Cooler Flow Friction','Crankcase Gas Spring Hysteresis','Position',[0.1
50 550 400])
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
```

HTG_2nd_Order.m

```
function            [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA]            =
HTG_2nd_Order(ENGINE_DATA)

% Written by Connor Speer, October 2017.
% This function contains a scheme for adding decoupled losses to the
% reference cycle results.

%% Input:
% ENGINE_DATA --> Engine geometry and operating point.

%% Preamble
% A heat engine communicates with its environment in three ways: Heat
% transfer between the thermal source and sink, and work transfer through
% the output shaft. Decoupled losses can affect any of these three energy
% transfer mechanisms.

%% Call SEA code to calculate reference cycle and decoupled losses
crank_inc = 1; % Crank angle step size for numerical integration in [deg]
Model_Code = 3; % Code specifying which model to run (3 = Simple Model).
Losses_Code = 8; % Code specifying which losses subfunction to run
part_traj_on_off = 0; % 0 = no particle trajectory calculation.

[REF_CYCLE_DATA, LOSSES_DATA, ...
    PARTICLE_TRAJECTORY_DATA] = sea(ENGINE_DATA,crank_inc,Model_Code,...
    Losses_Code,part_traj_on_off);


%% Heat Transfer Between the Engine and the Thermal Source
SECOND_ORDER_DATA.Qh_dot  =  REF_CYCLE_DATA(end).Qh*ENGINE_DATA.freq  +  ...  %
Reference cycle heat input rate
    (-LOSSES_DATA.P_flow_h) + ... % Flow friction in the heater
    (-0.5*LOSSES_DATA.P_flow_r)  +  ...  %  Half  of  flow  friction  in  the
regenerator
    LOSSES_DATA.Q_cond + ... % Conduction loss
    LOSSES_DATA.Q_seals + ... % Seal leakage
    LOSSES_DATA.Q_app + ... % Appendix gap loss
    LOSSES_DATA.Q_insulation + ... % Heat lost through the insulation
    LOSSES_DATA.Q_qrloss;  % Regenerator enthalpy loss

%% Work Transfer Between the Engine and the Load
SECOND_ORDER_DATA.W_dot  =  REF_CYCLE_DATA(end).W*ENGINE_DATA.freq  -  ...  %
Reference cycle power output
    LOSSES_DATA.P_mech - ... % Mechanical friction
    LOSSES_DATA.P_flow_h - ... % Flow friction in the heater
    LOSSES_DATA.P_flow_r - ... % Flow friction in the regenerator
    LOSSES_DATA.P_flow_k - ... % Flow friction in the cooler
    LOSSES_DATA.P_GSH - ... % Gas spring hysteresis
    LOSSES_DATA.P_seals - ... % Seal leakage
    LOSSES_DATA.P_HTH - ... % Heat transfer hysteresis
    LOSSES_DATA.P_FPS - ... % Finite piston speed
    LOSSES_DATA.P_pump_cool - ... % Coolant pump
    LOSSES_DATA.P_pump_hot; % Heating system pump


%% Heat Transfer Between the Engine and the Thermal Sink
```

```
SECOND_ORDER_DATA.Qk_dot  =  -REF_CYCLE_DATA(end).Qk*ENGINE_DATA.freq  +  ...  %
Reference cycle heat rejection rate
    LOSSES_DATA.P_mech + ... % Mechanical friction
    0.5*LOSSES_DATA.P_flow_r + ... % Half of flow friction in the regenerator
    LOSSES_DATA.P_flow_k + ... % Flow friction in the cooler
    LOSSES_DATA.Q_cond + ... % Conduction loss
    LOSSES_DATA.Q_seals + ... % Seal leakage
    LOSSES_DATA.Q_app + ... % Appendix gap loss
    LOSSES_DATA.Q_qrloss; % Regenerator enthalpy loss

%% Thermal Efficiency
SECOND_ORDER_DATA.eff_thermal                                                    =
SECOND_ORDER_DATA.W_dot/SECOND_ORDER_DATA.Qh_dot;

%% Overall Energy Balance
SECOND_ORDER_DATA.Energy_Imbalance = SECOND_ORDER_DATA.Qh_dot - ...
    SECOND_ORDER_DATA.W_dot - SECOND_ORDER_DATA.Qk_dot;

%% Display Results
fprintf('========== 2nd Order Model Results ============\n')
fprintf(' Heat input rate: %.2f[W]\n', SECOND_ORDER_DATA.Qh_dot);
fprintf(' Heat rejection rate: %.2f[W]\n', SECOND_ORDER_DATA.Qk_dot);
fprintf(' Power output: %.2f[W]\n', SECOND_ORDER_DATA.W_dot);
fprintf('             Thermal             efficiency:            %.1f[%%]\n',
SECOND_ORDER_DATA.eff_thermal*100);
fprintf(' Energy Imbalance: %.1f[W]\n', SECOND_ORDER_DATA.Energy_Imbalance);
fprintf('=============================================================\n')
%    fprintf('    Exergy    destruction    at    thermal    source:    %.1f[W]\n',
LOSSES_DATA.X_dest_source);
%    fprintf('    Exergy    destruction    at    thermal    sink:    %.1f[W]\n',
LOSSES_DATA.X_dest_sink);

fprintf('========== Summary of Losses ============\n')
fprintf(' Mechanical friction: %.2f[W]\n', LOSSES_DATA.P_mech);
fprintf(' Flow friction in the heater: %.2f[W]\n', LOSSES_DATA.P_flow_h);
fprintf('     Flow     friction     in     the     regenerator:     %.2f[W]\n',
LOSSES_DATA.P_flow_r);
fprintf(' Flow friction in the cooler: %.2f[W]\n', LOSSES_DATA.P_flow_k);
fprintf(' Gas spring hysteresis: %.2f[W]\n', LOSSES_DATA.P_GSH);
fprintf(' Seal leakage: %.1f[W]\n', LOSSES_DATA.P_seals);
fprintf(' Heat transfer hysteresis: %.1f[W]\n', LOSSES_DATA.P_HTH);
fprintf(' Finite piston speed: %.1f[W]\n', LOSSES_DATA.P_FPS);
fprintf(' Coolant pump: %.1f[W]\n', LOSSES_DATA.P_pump_cool);
fprintf(' Heating system pump: %.1f[W]\n', LOSSES_DATA.P_pump_hot);
fprintf(' Regenerator enthalpy loss: %.1f[W]\n', LOSSES_DATA.Q_qrloss);
fprintf(' Conduction loss: %.1f[W]\n', LOSSES_DATA.Q_cond);
fprintf(' Appendix gap loss: %.1f[W]\n', LOSSES_DATA.Q_app);
fprintf('       Heat       lost       through       insulation:       %.1f[W]\n',
LOSSES_DATA.Q_insulation);
fprintf('=============================================================\n')
```

HTG_2nd_Order_Vary.m

```
function                                DATA_STRUCTURE                                  =
HTG_2nd_Order_Vary(varName,Min,Max,Increment,ENGINE_DATA)

% Written by Connor Speer - April 2017
% Editted by Shahzeb Mirza - October 2017

% The purpose of this script is to run the SEA code repeatedly for a
% variety of any given variable, and return results.

% Inputs:
% varName --> String with the exact variable name

% Min --> Lower bound of the variable range being considered [any unit]
% Max --> Upper bound of the variable range being considered [any unit]
% Increment --> Increment size for variable range [any unit

% ENGINE_DATA --> Structure which defines engine geometry and operating
conditions

% Crank_Angle_Increment --> Crank angle step size for numerical integration
in [degrees]

% Model_Code --> Code specifying which model to run.
% 1 = Ideal Isothermal Model
% 2 = Ideal Adiabatic Model
% 3 = Simple Model
% Losses_Code --> Code specifying which losses subfunction to run
% 1 = parasitic_losses_GEN.m
% 2 = parasitic_losses_HTG.m


range = Min:Increment:Max;

% Preallocate space
DATA_STRUCTURE(length(range)).Qh_dot = [];
DATA_STRUCTURE(length(range)).Qk_dot = [];
DATA_STRUCTURE(length(range)).W_dot = [];
DATA_STRUCTURE(length(range)).eff_thermal = [];

counter = 1; % Initialize counter variable
counter_max = length(range);
h = waitbar(0,'Running 2nd order model...');

for i = range
    command = strcat('ENGINE_DATA.',varName,' = i;');
    eval(command);

    % Call the second order model subfunction
    [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA]                              =
HTG_2nd_Order(ENGINE_DATA);

    % Store outputs from the model
    DATA_STRUCTURE(counter).Qh_dot = SECOND_ORDER_DATA.Qh_dot;
    DATA_STRUCTURE(counter).Qk_dot = SECOND_ORDER_DATA.Qk_dot;
```

```
    DATA_STRUCTURE(counter).W_dot = SECOND_ORDER_DATA.W_dot;
    DATA_STRUCTURE(counter).eff_thermal = SECOND_ORDER_DATA.eff_thermal;

    DATA_STRUCTURE(counter).ref_cycle_power = REF_CYCLE_DATA(1).W_dot_ref;
    DATA_STRUCTURE(counter).ref_cycle_heat_in                                    =
REF_CYCLE_DATA(end).Qh*ENGINE_DATA.freq;
    DATA_STRUCTURE(counter).ref_cycle_heat_out                                   =
REF_CYCLE_DATA(end).Qk*ENGINE_DATA.freq;

    DATA_STRUCTURE(counter).FW = LOSSES_DATA.FW;
    DATA_STRUCTURE(counter).P_HEX        =        REF_CYCLE_DATA(1).W_dot_ref-
(REF_CYCLE_DATA(end).W*ENGINE_DATA.freq);
    DATA_STRUCTURE(counter).P_mech = LOSSES_DATA.P_mech;
    DATA_STRUCTURE(counter).pdroph = LOSSES_DATA.pdroph;
    DATA_STRUCTURE(counter).pdropr = LOSSES_DATA.pdropr;
    DATA_STRUCTURE(counter).pdropk = LOSSES_DATA.pdropk;
    DATA_STRUCTURE(counter).P_flow_h = LOSSES_DATA.P_flow_h;
    DATA_STRUCTURE(counter).P_flow_r = LOSSES_DATA.P_flow_r;
    DATA_STRUCTURE(counter).P_flow_k = LOSSES_DATA.P_flow_k;
    DATA_STRUCTURE(counter).P_GSH = LOSSES_DATA.P_GSH;
    DATA_STRUCTURE(counter).Q_qrloss = LOSSES_DATA.Q_qrloss;
    DATA_STRUCTURE(counter).Q_cond = LOSSES_DATA.Q_cond;
    DATA_STRUCTURE(counter).Q_app = LOSSES_DATA.Q_app;
    DATA_STRUCTURE(counter).P_seals = LOSSES_DATA.P_seals;
    DATA_STRUCTURE(counter).Q_seals = LOSSES_DATA.Q_seals;
    DATA_STRUCTURE(counter).P_HTH = LOSSES_DATA.P_HTH;
    DATA_STRUCTURE(counter).P_FPS = LOSSES_DATA.P_FPS;
    DATA_STRUCTURE(counter).P_pump_cool = LOSSES_DATA.P_pump_cool;
    DATA_STRUCTURE(counter).P_pump_hot = LOSSES_DATA.P_pump_hot;
    DATA_STRUCTURE(counter).X_dest_source = LOSSES_DATA.X_dest_source;
    DATA_STRUCTURE(counter).X_dest_sink = LOSSES_DATA.X_dest_sink;
    DATA_STRUCTURE(counter).Q_insulation = LOSSES_DATA.Q_insulation;

    % Close the temporary data file to prevent Matlab errors
    fclose('all');

    counter = counter + 1;

    waitbar(counter/ counter_max)
end

close(h)
```

HTG_ENGINE_DATA.m

```
function ENGINE_DATA = HTG_ENGINE_DATA

% This function will define a structure that describes the geometric data,
% operating conditions, and working fluid properties for the HTG engine.

% Default values are for the fully modified version of the HTG engine.

%% Drive Mechanism:
```

```
% Engine Type
% s --> Alpha sinusoidal
% a --> Alpha slider crank
% y --> Alpha Ross yoke drive
% r --> Alpha Ross rocker-V drive
% g --> Gamma sinusoidal
% x --> Gamma slider crank
ENGINE_DATA.engine_type = 'x';

% Slider Crank Mechanism Dimensions
ENGINE_DATA.Pbore = 0.044; %[m]
ENGINE_DATA.Pr1 = 0; %[m]
ENGINE_DATA.Pr2 = 0.0375; %[m]
ENGINE_DATA.Pr3 = 0.156; %[m]

ENGINE_DATA.Dbore = 0.096; %[m]
ENGINE_DATA.Dr1 = 0; %[m]
ENGINE_DATA.Dr2 = 0.0375; %[m]
ENGINE_DATA.Dr3 = 0.130; %[m]

%% Volumes: WHAT ABOUT THE APPENDIX GAP?????!
% Power Piston Clearance Volume for Gamma Engines
% --> Includes the Cylinder Head and Connecting Pipe in this case.
% Dead volume reduction parts are subtracted
ENGINE_DATA.Vclp = 1.733e-04 - 0.000132; %[m^3]

% Power Piston Swept Volume for Gammas
ENGINE_DATA.Vswp = 4.256e-04; %[m^3]

% Total Displacer Clearance Volume (above and below) for Gammas
ENGINE_DATA.Vcld = 1.448e-05; %[m^3]

% Displacer Swept Volume for Gammas
ENGINE_DATA.Vswd = 5.429e-04; %[m^3]

% Displacer Phase Angle Advance for Gamma
ENGINE_DATA.beta_deg = 90.0; %[deg]

%% Cooler:
% Cooler Type
% p --> smooth pipes
% a --> smooth annulus
% s --> slots
ENGINE_DATA.cooler_type = 's';

% Cooler Slot Width for Slot Cooler
ENGINE_DATA.cooler_slot_width = 1.00e-03; %[m]

% Cooler Slot Height for Slot Cooler
ENGINE_DATA.cooler_slot_height = 1.00e-02; %[m]

% Cooler Heat Exchanger Length
ENGINE_DATA.cooler_length = 8.400e-02; %[m]
```

```
% Cooler Number of Slots
ENGINE_DATA.cooler_num_slots = 144; %[m]

%% Regenerator:
% Regenerator Configuration
% t --> tubular regenerator
% a --> annular regenerator
ENGINE_DATA.regen_config = 'a';

% Regen Housing O.D. for Annular Regenerator (May not be needed since
% conduction loss calc was redone)
ENGINE_DATA.regen_housing_OD = 1.465e-01; %[m]

% Regen Housing I.D. for Annular Regenerator (May not be needed since
% conduction loss calc was redone)
ENGINE_DATA.regen_housing_ID = 1.355e-01; %[m]

% Matrix I.D. for Annular Regenerator
ENGINE_DATA.regen_matrix_ID = 1.030e-01;

% Regenerator Length
ENGINE_DATA.regen_length = 5.481e-02; %[m]

% Regenerator Number of Tubes
ENGINE_DATA.regen_num_tubes = 1;

% Regenerator Matrix Type
% m --> mesh
% f --> foil
% n --> no matrix
ENGINE_DATA.regen_matrix_type = 'm';

% Matrix Porosity for Mesh Matrix
ENGINE_DATA.regen_matrix_porosity = 0.893;

% Matrix Wire Diameter for Mesh Matrix
ENGINE_DATA.regen_wire_diameter = 5.08e-05; %[m]

%% Heater:
% Heater Type
% p --> smooth pipes
% a --> smooth annulus
% s --> slots
ENGINE_DATA.heater_type = 's';

% Pipe I.D. for Smooth Pipe Heater
% ENGINE_DATA.heater_pipe_ID = ; %[m]

% Heater Slot Width for Slot Heater
ENGINE_DATA.heater_slot_width = 3.180e-03; %[m]

% Heater Slot Height for Slot Heater
ENGINE_DATA.heater_slot_height = 5.170e-03; %[m]
```

```
% Heater Heat Exchanger Length
ENGINE_DATA.heater_length = 1.233e-01; %[m]

% Heater Number of Slots for Slot Heater
ENGINE_DATA.heater_num_slots = 64;

%% Operating Conditions:
% Working Fluid
% hy --> hydrogen
% he --> helium
% ai --> air
ENGINE_DATA.gas_type = 'ai';

% Mean Pressure
ENGINE_DATA.pmean = 1000000.0; %[Pa]  (= 10 bar)

% Cold Sink Temperature
ENGINE_DATA.Tsink = 294.0; %[K]

% Cooler Gas Temperature
ENGINE_DATA.Tgk = 294.0; %[K]

% Cooler Wall Temperature
ENGINE_DATA.Twk = 294.0; %[K]

% Compression Space Temperature
ENGINE_DATA.Tgc = 294.0; %[K]

% Hot Source Temperature
ENGINE_DATA.Tsource = 273 + 200; %[K]

% Heater Gas Temperature
ENGINE_DATA.Tgh = 273 + 200; %[K]

% Heater Wall Temperature
ENGINE_DATA.Twh = 273 + 200; %[K]

% Expansion Space Temperature
ENGINE_DATA.Tge = 273 + 200; %[K]

% Operating Frequency
ENGINE_DATA.freq = 4.2; %[Hz]

%% Data for Engine Specific Loss Calculations:
% Maximum volume of the buffer space
% Volume of crankcase extension has been added
ENGINE_DATA.V_buffer_max = 0.0032 + 0.004633333; %[m^3]

% Constant mechanism effectiveness
ENGINE_DATA.effect = 0.7; % [unitless]

% Configuration code for GSH calculation
ENGINE_DATA.GSH_config = 4; % All mods
```

HTG_GSH.m

```
function P_hys = HTG_GSH(RPM,pmean,tk)

% Inputs:
% RPM --> engine speed in [rev/min]
% pmean --> engine mean pressure in [Pa]
% tk --> cooler temperature in [K]

omega = RPM*(1/60)*(2*pi); %[rad/s]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Gas Spring Hysteresis Losses (Subtract from work output)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Input parameters needed:
% omega --> angular frequency of the engine [rad/s]
% gama --> specific heat ratio of the spring gas
% Tw --> gas spring wall temperature [K]
% pmean --> mean pressure in the gas spring [Pa] (assumed to be the mean
% cycle pressure)
% kgas --> thermal conductivity of the spring gas [W/mK] (assumed to be the
% working fluid)
% delV --> volume amplitude of the spring cavity [m^3]
% VB --> mean volume of the gas spring cavity [m^3]
% Aw --> mean wetted area of the gas spring [m^2]
% delA --> wetted area amplitude [m^2]

% Inputs for Original ST05G-CNC Stirling Engine
Tw = tk; %[K] (assumed to be equal to the cooler temperature)
delV = 0.25*pi*(0.085^2)*0.075; %[m^3]
VB = 0.5*delV + 0.002488647 + 0.000091248 + 0.000419874 + 0.000186558 + ...
    0.000182367 - 0.000229935 - 0.000061378 - 0.000059277 - ...
    0.000007376 - 0.000021882 - 0.000013195 - 0.000009817 - ...
    0.000009173 - 0.000231607; %[m^3]
delA = pi*0.085*0.075; %[m^2]
Aw = 0.5*delA + 0.05853681 + 0.00804849 + 0.01866106 + 0.01030158 + ...
    0.02145481 + 0.03279471 + 0.01778246 + 0.02301812 + 0.00211262 + ...
    0.007294024 + 0.00026389 + 0.00045239 + 0.02173007; %[m^2]

% Calculation of the gas spring hysteresis loss
F1 = sqrt(0.125*omega*gama*(gama-1)*Tw*pmean*kgas);
F2 = (delV/VB)*((0.5*gama*(delV/VB)*Aw) - delA);

P_hys = F1*F2*freq; % Power lost to gas spring hysteresis in [W].
```

HTG_GSH_Empirical.m

```
function P_hys = HTG_GSH_Empirical(Hz,pmean,Config_code)
% Written by Connor Speer - August 2017

% Calculates the gas spring hysteresis loss for the the high temperature
```

```
% gamma engine with the 85mm piston.
% Based on a surface fit that is a second degree polynomial in the x and y
% directions. 'poly22'.

% Inputs:
% Hz --> engine speed in [Hz]
% pmean --> engine mean pressure in [Pa]
% Config_code --> code specifying the buffer space configuration
% 1 --> 85 mm piston, original crankcase volume
% 2 --> 85 mm piston, extended crankcase volume
% 3 --> 44 mm piston, original crankcase volume
% 4 --> 44 mm piston, extended crankcase volume

x = Hz;
y = pmean;

switch Config_code
    case 1 % 85 mm piston, original crankcase volume
    p00 = -2.855;
    p10 = 4.902;
    p01 = -1.239e-05;
    p20 = -0.4064;
    p11 = 2.327e-05;
    p02 = 7.373e-12;

    case 2 % 85 mm piston, extended crankcase volume
    p00 = 0;
    p10 = 0;
    p01 = 0;
    p20 = 0;
    p11 = 0;
    p02 = 0;

    case 3 % 44 mm piston, original crankcase volume
    p00 = 0.2473;
    p10 = -0.1534;
    p01 = 2.311e-07;
    p20 = 0.07564;
    p11 = 2.076e-06;
    p02 = 1.975e-13;

    case 4 % 44 mm piston, extended crankcase volume
    p00 = 0.1611;
    p10 = -0.2116;
    p01 = 2.969e-07;
    p20 = 0.09854;
    p11 = 5.766e-07;
    p02 = 1.987e-13;
end


P_hys = p00 + p10.*x + p01.*y + p20.*x.^2 + p11.*x.*y + p02.*y.^2; %[W]
```

parasitic_losses_HTG.m

```
function                                LOSSES_DATA                                =
parasitic_losses_HTG(ENGINE_DATA,REF_CYCLE_DATA,crank_inc)
% Written by Connor Speer - January 2017
% Calculates the parasitic losses.
% Modified by Connor Speer, October 2017. No globals.

% Losses already accounted for by the "Simple" simulation:
% - Regenerator Enthalpy Loss

% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% REF_CYCLE_DATA --> Structure containing data from the reference cycle
% crank_inc --> crank angle step size in [degrees]

% Output:
% LOSSES_DATA --> Structure containing parasitic loss information


%% Mechanical Friction (Forced Work Method, See Senft's Book)
vtot = ([REF_CYCLE_DATA.Vc] + ENGINE_DATA.Vk + ENGINE_DATA.Vr + ...
    ENGINE_DATA.Vh + [REF_CYCLE_DATA.Ve]); % cubic meters

p_engine = [REF_CYCLE_DATA.p]; % Pascals

Pbore = ENGINE_DATA.Pbore; % piston bore [m]
Pr1 = ENGINE_DATA.Pr1; % piston desaxe offset in [m]
Pr2 = ENGINE_DATA.Pr2;% piston crank length (half stroke) in [m]
Pr3 = ENGINE_DATA.Pr3;% piston connecting rod lengths [m]

theta = 0:crank_inc*(pi/180):2*pi;
Ptheta2 = pi - theta;

Ptheta3 = pi - asin((-Pr1+(Pr2*sin(Ptheta2)))/Pr3);
Pr4 = Pr2*cos(Ptheta2) - Pr3*cos(Ptheta3);
Pr4max = sqrt(((Pr2+Pr3)^2)-(Pr1^2));
Pr4min = sqrt(((Pr3-Pr2)^2)-(Pr1^2));
Pstroke = Pr4max - Pr4min;

% Crankcase Volume Variations in (m^3)
V_buffer = ENGINE_DATA.V_buffer_max - ((Pr4max-Pr4)*(((pi/4)*(Pbore^2))));

% Crankcase Pressure Models %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
p0 = mean([REF_CYCLE_DATA.p]); % Mean crankcase pressure in (Pa).

% Mean crankcase volume in (m^3).
V0 = ENGINE_DATA.V_buffer_max - 0.5*Pstroke*(((pi/4)*(Pbore^2)));

% % Constant Buffer Pressure Equal to the Mean Pressure
% p_buff_const = mean([REF_CYCLE_DATA.p]); % Pascals
% p_buffer = repelem(p_buff_const,length(vtot));

% % Buffer Pressure Assuming Isothermal Compression and Expansion
% T_crankcase = ENGINE_DATA.Tgc; % Mean crankcase temperature in [K].
% % Mass of gas in the crankcase in [kg].
```

```
% m_crankcase = (p0*V0)/(ENGINE_DATA.rgas*T_crankcase);
% p_buffer = (m_crankcase*R*T_crankcase)./V_buffer;

% Buffer Pressure Assuming Adiabatic Compression and Expansion in (Pa).
p_buffer = p0.*((V0./V_buffer).^ENGINE_DATA.gamma);

% Run Forced Work Subfunction %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[W_ind, FW, W_shaft] = ...
    FW_Subfunction_v3(1,p_engine,p_buffer,vtot,ENGINE_DATA.effect,0);

% Store forced work in LOSSES_DATA structure
LOSSES_DATA.FW = FW; %(J)

% Net output shaft power (W) (Senft's formula)
P_shaft = W_shaft*ENGINE_DATA.freq;

% Power lost to mechanical friction in [W]
LOSSES_DATA.P_mech = (W_ind*ENGINE_DATA.freq) - P_shaft;


%% Flow Friction
% Calculate pressure drops
[dwork_h,dwork_r,dwork_k,pdropk,pdroph,pdropr] = ...
    worksim(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);

LOSSES_DATA.pdropk = pdropk; % Pressure drop across the cooler in (Pa)
LOSSES_DATA.pdroph = pdroph; % Pressure drop across the heater in (Pa)
LOSSES_DATA.pdropr = pdropr; % Pressure drop across the regenerator in (Pa)

% Pressure drop power loss in the heater
LOSSES_DATA.P_flow_h = dwork_h*ENGINE_DATA.freq; %(W)

% Pressure drop power loss in the regenerator
LOSSES_DATA.P_flow_r = dwork_r*ENGINE_DATA.freq; %(W)

% Pressure drop power loss in the cooler
LOSSES_DATA.P_flow_k = dwork_k*ENGINE_DATA.freq; %(W)


%% Regenerator Enthalpy Loss
A = isfield(REF_CYCLE_DATA, 'qrloss');
if A == 1
    % Convert regenerator enthalpy loss per cycle in (J) to a rate in (W)
    LOSSES_DATA.Q_qrloss = REF_CYCLE_DATA(1).qrloss*ENGINE_DATA.freq; %(W)
else
    LOSSES_DATA.Q_qrloss = 0;
end


%% Conduction Loss (1-D Fourier Law)
% Input parameters needed:
% A_wall --> Average cross-sectional area of the wall conduction path [m^2]
% k_wall --> Average thermal conductivity of the wall conduction material
% [W/mK]
```

```
% L_wall --> Length of the wall conduction path [m]
% A_regen --> Regenerator free flow area [m^2]
% kgas --> Thermal conductivity of the working gas [W/mK]
% L_regen --> Length of the regenerator [m]
% A_displacer --> Cross-sectional area of the displacer [m^2]
% k_displacer --> Effective thermal conductivity of the displacer [W/m^2]
% L_displacer --> Length of the displacer [m]

% % Inputs for Original ST05G-CNC Stirling Engine
% A_wall = 0.25*pi*((0.1397^2)-(0.134^2)+(0.0976^2)-(0.096^2)); % [m^2]
% k_wall = 25; % Stainless steel [W/m^2K] (Temperature dependent)
% L_wall = 0.05651; %[m] Length of regenerator cavity in this case
% A_regen = 0.1*0.25*pi*((0.134^2)-(0.0976^2)); %[m^2] Area x porosity
% L_regen = 0.05651; %[m]
% A_displacer = 0.25*pi*(0.094^2); %[m^2]
%  k_displacer  =  (0.25*pi)*(1/A_displacer)*(((0.0924^2)*(kgas))+(((0.094^2)-
(0.0924^2))*k_wall)); %[W/mK]
% L_displacer = 0.15405; %[m]

% Inputs for As Built HTG Stirling Engine
A_wall = 0.25*pi*((0.1465^2)-(0.1355^2)+(0.103^2)-(0.0956^2)); % [m^2]
k_wall = 47; % AISI 1020 steel [W/m^2K] (From Solidworks)
L_wall = 0.055; %[m] Length of regenerator cavity in this case
A_regen = 0.25*pi*((0.1355^2)-(0.103^2)); %[m^2] Area or regenerator
k_regen = 0.9*ENGINE_DATA.kgas + 0.1*(19); % Porosity weighted working gas +
316 stainless steel.
L_regen = 0.055; %[m]
A_displacer = 0.25*pi*(0.094^2); %[m^2]
k_displacer                                                    =
(0.25*pi)*(1/A_displacer)*(((0.0924^2)*(ENGINE_DATA.kgas))+(((0.094^2)-
(0.0924^2))*19)); %[W/mK]
L_displacer = (0.15613 + 0.14513)/2; %[m]

% Conduction through the engine walls
Q_wall = (A_wall*k_wall*(ENGINE_DATA.Twh-ENGINE_DATA.Twk))/L_wall; %[W]

% Conduction through the regenerator
Q_fluid = (A_regen*k_regen*(ENGINE_DATA.Tgh-ENGINE_DATA.Tgk))/L_regen; %[W]

% Conduction through the displacer
Q_displacer = (A_displacer*k_displacer*(mean([REF_CYCLE_DATA.Tge])...
    -mean([REF_CYCLE_DATA.Tgc])))/L_displacer; %[W]

% Total Conduction Loss [W]
LOSSES_DATA.Q_cond = Q_wall + Q_fluid + Q_displacer;


%% Gas Spring Hysteresis Losses
% Input parameters needed:
% Tw --> gas spring wall temperature [K]
% delV --> volume amplitude of the spring cavity [m^3]
% VB --> mean volume of the gas spring cavity [m^3]
% Aw --> mean wetted area of the gas spring [m^2]
% delA --> wetted area amplitude [m^2]
```

```
% Inputs for Original ST05G-CNC Stirling Engine
% Tw = ENGINE_DATA.Tgk; %[K] (assumed to be equal to the cooler temperature)
% delV = 0.25*pi*(0.085^2)*0.075; %[m^3]
% VB = 0.5*delV + 0.002488647 + 0.000091248 + 0.000419874 + 0.000186558 + ...
%     0.000182367 - 0.000229935 - 0.000061378 - 0.000059277 - ...
%     0.000007376 - 0.000021882 - 0.000013195 - 0.000009817 - ...
%     0.000009173 - 0.000231607; %[m^3]
% delA = pi*0.085*0.075; %[m^2]
% Aw = 0.5*delA + 0.05853681 + 0.00804849 + 0.01866106 + 0.01030158 + ...
%     0.02145481 + 0.03279471 + 0.01778246 + 0.02301812 + 0.00211262 + ...
%     0.007294024 + 0.00026389 + 0.00045239 + 0.02173007; %[m^2]
%
% % Calculation of the gas spring hysteresis loss
% F1 = sqrt(0.125*ENGINE_DATA.omega*ENGINE_DATA.gamma*...
% (ENGINE_DATA.gamma-1)*Tw*ENGINE_DATA.pmean*ENGINE_DATA.kgas);
% F2 = (delV/VB)*((0.5*ENGINE_DATA.gamma*(delV/VB)*Aw) - delA);
%
%  LOSSES_DATA.P_GSH  =  F1*F2*ENGINE_DATA.freq;  %  Power  lost  to  gas  spring
hysteresis in [W].
LOSSES_DATA.P_GSH = 0;

%% Seal Leakage Losses
% Input parameters needed:
%  mloss_outside  -->  Working  fluid  loss  rate  due  to  leakage  to  the  outside
[Pa/s]
% isentropic_eff_comp --> Isentropic efficiency of the compressor
% h_ambient --> Enthalpy of the air entering the compressor in [J/kg]
% h2_isentropic --> Discharge enthalpy of the compressor in [J/kg]

% Inputs for HTG Stirling Engine
mloss_outside = 0; % [kg/s]
isentropic_eff_comp = 0.75; % [unitless]
h_ambient = 298180; % [J/kg]
h2_isentropic = 575590; % [J/kg]

% Piston Leakage

% Displacer Leakage

% Leakage to the outside (adds compressor work for a pressurized engine)
% *** This will not affect the measured engine output if the engine is not
% driving the compressor.
% Power consumed by the compressor to maintain engine pressure
P_compressor  =  (mloss_outside*(h2_isentropic-h_ambient))/isentropic_eff_comp;
%[W]

% Total leakage loss effects
LOSSES_DATA.P_seals = P_compressor; % Power lost due to seal leakage in [W].
LOSSES_DATA.Q_seals = 0; % Heat lost due to seal leakage in [W].


%% Appendix Gap Losses (Pg 140 of Urieli and Berchowitz)
% Input parameters needed:
% D --> diameter of the displacer cylinder ??? [m]
% kgas --> thermal conductivity of the working fluid [W/mK]
```

```
% Xd --> displacer motion amplitude (displacer stroke) [m]
% upsilon --> linear axial temperature gradient (bad assumption in this case
since cooler is a different material) [K/m]
% h --> appendix gap width(distance b/w displacer wall and cylinder wall) [m]
% omega --> engine angular frequency [rad/s]
% phi --> phase angle b/w displacer motion and pressure variation [rad]
% alpha_solid --> thermal diffusivity of the displacer and cylinder material
in [m^2/s]
% k_solid --> thermal conductivity of the displacer and cylinder material in
[W/m.K]

% Inputs for HTG Stirling Engine
D = 0.096; %[m]
Xd = 0.075; %[m]
upsilon = (ENGINE_DATA.Tgh-ENGINE_DATA.Tgk)/0.220; %[K/m] (Divided by length
of displacer cylinder)
h = 0.001; %[m]
alpha_solid = 1.172e-05; %[m^2/s] (Steel from Wikipedia)
k_solid = 47; %[W/m.K] (AISI 1020 Steel from Solidworks - Displacer is
Stainless so this should be refined)

% Crank angle at which minimum workspace pressure occurs
[~,pmax_index] = max([REF_CYCLE_DATA.p]);
pmaxAngle = pmax_index*crank_inc - crank_inc; %[degrees]

% Crank angle at which expansion space volume is minimum
[~,Vemin_index] = min([REF_CYCLE_DATA.Ve]);
DispTDCAngle = Vemin_index*crank_inc - crank_inc; %[degrees]

% Crank angle by which pressure variation leads the displacer motion
phi = (pmaxAngle-DispTDCAngle); %[degrees]

pressure = [REF_CYCLE_DATA.p];
delp = max(pressure) - min(pressure); %[Pa]

w = sqrt(ENGINE_DATA.omega/(2*alpha_solid));

A1 = -pi*0.5*D*ENGINE_DATA.kgas*(Xd^2)*(upsilon/h);
A2 = pi*D*delp*h*Xd*ENGINE_DATA.omega*sin(phi);
A3                  =                   (ENGINE_DATA.gamma/(ENGINE_DATA.gamma-
1))*log(ENGINE_DATA.Tgh/ENGINE_DATA.Tgk)*...
    (0.5-(ENGINE_DATA.kgas/(w*h*k_solid))-0.5);

LOSSES_DATA.Q_app = abs(A1 + (A2*A3)); % Heat lost due to appendix gap in
[W].


%% Heat Transfer Hysteresis
LOSSES_DATA.P_HTH = 0; % Power lost to heat transfer hysteresis in [W].


%% Finite Piston Speed Loss
% % Equations come from Petrescu 2002 and quasi-steady approach was inspired
% % by Hosseinzade 2015
```

```
%
% % Calculate the constant quantity "a"
% a = sqrt(3*ENGINE_DATA.gamma);
%
% % Calculate vectors of average molecular speeds corresponding to the
% % vector of crank angle increments in (m/s)
% cc = sqrt(3*ENGINE_DATA.R.*[REF_CYCLE_DATA.Tgc]); % Compression space
% ce = sqrt(3*ENGINE_DATA.R.*[REF_CYCLE_DATA.Tge]); % Expansion space
%
% % Calculate vectors of piston speeds in (m/s)
% Dr1 = ENGINE_DATA.Dr1;
% Dr2 = ENGINE_DATA.Dr2;
% Dr3 = ENGINE_DATA.Dr3;
% Dbore = ENGINE_DATA.Dbore;
%
% Pr1 = ENGINE_DATA.Pr1;
% Pr2 = ENGINE_DATA.Pr2;
% Pr3 = ENGINE_DATA.Pr3;
% Pbore = ENGINE_DATA.Pbore;
%
% Ptheta2 = pi - [REF_CYCLE_DATA.theta]; %(rad)
% Dtheta2 = Ptheta2 - ENGINE_DATA.beta; %(rad)
%
% Ptheta2_dot = -ENGINE_DATA.omega; %(rad/s)
% Dtheta2_dot = -ENGINE_DATA.omega; %(rad/s)
%
% Ptheta3 = pi - asin((-Pr1+(Pr2.*sin(Ptheta2)))./Pr3); %(rad)
% Dtheta3 = pi - asin((-Dr1+(Dr2.*sin(Dtheta2)))./Dr3); %(rad)
%
% wp = (Pr2*Ptheta2_dot.*sin(Ptheta3-Ptheta2))./cos(Ptheta3); %(m/s)
% wd = (Dr2*Dtheta2_dot.*sin(Dtheta3-Dtheta2))./cos(Dtheta3); %(m/s)
%
% % Calculate incremental volume changes caused by the piston and displacer
%          dPtheta3          =          (Pr2.*cos(Ptheta2))./(Pr3.*sqrt(1-(((-
Pr1+(Pr2.*sin(Ptheta2)))./Pr3).^2)));
% dPr4 = Pr2.*sin(Ptheta2) + Pr3.*sin(Ptheta3).*dPtheta3;
% dVp = -(pi/4)*(Pbore^2).*dPr4;
%
%       dDtheta3       =               (Dr2.*cos(Dtheta2))./(Dr3.*sqrt(1-(((-
Dr1+(Dr2.*sin(Dtheta2)))./Dr3).^2)));
% dDr4 = Dr2.*sin(Dtheta2) + Dr3.*sin(Dtheta3).*dDtheta3;
% dVe = -(pi/4)*(Dbore^2).*(dDr4);
%
% % Calculate vectors of lost work
% dWp = [REF_CYCLE_DATA.p].*(-a*wp./cc).*dVp;
%
% % Calculate total work lost to finite piston speed
% W_FPS = sum(dWp);
%
% % Calculate total power lost to finite piston speed
% LOSSES_DATA.P_FPS = W_FPS*ENGINE_DATA.freq; %(W)
LOSSES_DATA.P_FPS = 0; % Power loss due to relative motion between piston and
gas molecules in [W].


%% Cooling/Heating System Pumping Power
```

```
% Some of the power produced by the engine must be used to drive the
% coolant pump (unless a heat pipe or other passive cooling method is used).
% A fuel pump or hot side water pump may also be needed.

% Inputs:
% rho_cool --> density of coolant in (kg/m^3)
% g --> gravitational acceleration in (m/s^2)
% V_dot_cool --> volume flow rate of coolant in (m^3/s)
% H_cool --> coolant pump net head in (m)
% eff_pump_cool --> coolant pump efficiency

% P_pump_cool = (rho_cool*g*V_dot_cool*H_cool)/eff_pump; %(W)

LOSSES_DATA.P_pump_cool = 0; % Power required to drive the coolant pump in
(W).

LOSSES_DATA.P_pump_hot = 0; % Power required to drive fuel pump or hot water
pump in [W].


%% Exergy Destruction due to Heat Conduction at the Thermal Source/Sink
Interfaces
% The temperature drop between the heater/cooler gas temperature and the
% thermal source/sink causes exergy destruction.

% Calculations follow an example on Pg 446 of Cengel Thermodynamics
% Assumptions:
% 1. Steady state conduction process.
% 2. 1-dimensional heat conduction.
% 3. The exergy of the wall itself is constant, since it does not change
% state.

% Average heat transfer rates in the heater and cooler in (W)
Qh_dot_avg = mean([REF_CYCLE_DATA.Qh]*ENGINE_DATA.freq);
Qk_dot_avg = mean([REF_CYCLE_DATA.Qk]*ENGINE_DATA.freq);

% Average gas temperatures in the heater and cooler in (K)
Tgh_avg = mean([ENGINE_DATA.Tgh]);
Tgk_avg = mean([ENGINE_DATA.Tgk]);

% Temperatures of the thermal source and sink in (K)
Tsource = ENGINE_DATA.Tsource;
Tsink = ENGINE_DATA.Tsink;

T0 = ENGINE_DATA.Tsink; % Temperature of the environment in (K)

% Calculate exergy destruction in the heater and cooler in (W)
LOSSES_DATA.X_dest_source  =  Qh_dot_avg*(1-(T0/Tsource))  -  Qh_dot_avg*(1-
(T0/Tgh_avg));

LOSSES_DATA.X_dest_sink  =  Qk_dot_avg*(1-(T0/Tsink))  -  Qk_dot_avg*(1-
(T0/Tgk_avg));
```

```
%% Heat lost through the heating cap insulation
LOSSES_DATA.Q_insulation = 0;
```

parasitic_losses_HTG_B.m

```
function                                LOSSES_DATA                                =
parasitic_losses_HTG_B(ENGINE_DATA,REF_CYCLE_DATA,crank_inc)
% Written by Connor Speer - January 2017
% Calculates the parasitic losses.
% Modified by Connor Speer, October 2017. No globals.

% Losses already accounted for by the "Simple" simulation:
% - Regenerator Enthalpy Loss

% Inputs:
% ENGINE_DATA --> Structure containing engine geometry and operating point
% REF_CYCLE_DATA --> Structure containing data from the reference cycle
% crank_inc --> crank angle step size in [degrees]

% Output:
% LOSSES_DATA --> Structure containing parasitic loss information


%% Mechanical Friction (Forced Work Method, See Senft's Book)
vtot = ([REF_CYCLE_DATA.Vc] + ENGINE_DATA.Vk + ENGINE_DATA.Vr + ...
    ENGINE_DATA.Vh + [REF_CYCLE_DATA.Ve]); % cubic meters

p_engine = [REF_CYCLE_DATA.p]; % Pascals

Pbore = ENGINE_DATA.Pbore; % piston bore [m]
Pr1 = ENGINE_DATA.Pr1; % piston desaxe offset in [m]
Pr2 = ENGINE_DATA.Pr2;% piston crank length (half stroke) in [m]
Pr3 = ENGINE_DATA.Pr3;% piston connecting rod lengths [m]

theta = 0:crank_inc*(pi/180):2*pi;
Ptheta2 = pi - theta;

Ptheta3 = pi - asin((-Pr1+(Pr2*sin(Ptheta2)))/Pr3);
Pr4 = Pr2*cos(Ptheta2) - Pr3*cos(Ptheta3);
Pr4max = sqrt(((Pr2+Pr3)^2)-(Pr1^2));
Pr4min = sqrt(((Pr3-Pr2)^2)-(Pr1^2));
Pstroke = Pr4max - Pr4min;

% Crankcase Volume Variations in (m^3)
V_buffer = ENGINE_DATA.V_buffer_max - ((Pr4max-Pr4)*(((pi/4)*(Pbore^2))));

% Crankcase Pressure Models %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
p0 = mean([REF_CYCLE_DATA.p]); % Mean crankcase pressure in (Pa).

% Mean crankcase volume in (m^3).
V0 = ENGINE_DATA.V_buffer_max - 0.5*Pstroke*(((pi/4)*(Pbore^2)));

% % Constant Buffer Pressure Equal to the Mean Pressure
```

```
% p_buff_const = mean([REF_CYCLE_DATA.p]); % Pascals
% p_buffer = repelem(p_buff_const,length(vtot));

% % Buffer Pressure Assuming Isothermal Compression and Expansion
% T_crankcase = ENGINE_DATA.Tgc; % Mean crankcase temperature in [K].
% % Mass of gas in the crankcase in [kg].
% m_crankcase = (p0*V0)/(ENGINE_DATA.rgas*T_crankcase);
% p_buffer = (m_crankcase*R*T_crankcase)./V_buffer;

% Buffer Pressure Assuming Adiabatic Compression and Expansion in (Pa).
p_buffer = p0.*((V0./V_buffer).^ENGINE_DATA.gamma);

% % Buffer Pressure Determined By Shazeb's Empirical Ellipse Function
% p_buffer = getBufferPressure(ENGINE_DATA.pmean,crank_inc);

% Run Forced Work Subfunction %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[W_ind, FW, W_shaft] = ...
    FW_Subfunction_v3(1,p_engine,p_buffer,vtot,ENGINE_DATA.effect,0);

% Store forced work in LOSSES_DATA structure
LOSSES_DATA.FW = FW; %(J)

% Net output shaft power (W) (Senft's formula)
P_shaft = W_shaft*ENGINE_DATA.freq;

% Power lost to mechanical friction in [W]
LOSSES_DATA.P_mech = (W_ind*ENGINE_DATA.freq) - P_shaft;


%% Flow Friction
regen_corr_factor = 1;

% Calculate pressure drops
[~,~,~,pdropk,pdroph,pdropr] = ...
    worksim(ENGINE_DATA,REF_CYCLE_DATA,crank_inc);

LOSSES_DATA.pdropk = pdropk; % Pressure drop across the cooler in (Pa)
LOSSES_DATA.pdroph = pdroph; % Pressure drop across the heater in (Pa)
LOSSES_DATA.pdropr = pdropr; % Pressure drop across the regenerator in (Pa)

dwork_h                                                         =
sum(crank_inc*(pi/180).*pdropr(1:360).*[REF_CYCLE_DATA(1:360).dVe]);         %
pumping work [J]
dwork_r                                                         =
sum(crank_inc*(pi/180).*regen_corr_factor.*pdropr(1:360).*[REF_CYCLE_DATA(1:3
60).dVe]); % pumping work [J]
dwork_k                                                         =
sum(crank_inc*(pi/180).*pdropr(1:360).*[REF_CYCLE_DATA(1:360).dVe]);         %
pumping work [J]

% Pressure drop power loss in the heater
LOSSES_DATA.P_flow_h = dwork_h*ENGINE_DATA.freq; %(W)

% Pressure drop power loss in the regenerator
```

```
LOSSES_DATA.P_flow_r = dwork_r*ENGINE_DATA.freq; %(W)

% Pressure drop power loss in the cooler
LOSSES_DATA.P_flow_k = dwork_k*ENGINE_DATA.freq; %(W)

%% Regenerator Enthalpy Loss
A = isfield(REF_CYCLE_DATA, 'qrloss');
if A == 1
    % Convert regenerator enthalpy loss per cycle in (J) to a rate in (W)
    LOSSES_DATA.Q_qrloss = REF_CYCLE_DATA(1).qrloss*ENGINE_DATA.freq; %(W)
else
    LOSSES_DATA.Q_qrloss = 0;
end

%% Conduction Loss (1-D Fourier Law)
% Input parameters needed:
% A_wall --> Average cross-sectional area of the wall conduction path [m^2]
% k_wall --> Average thermal conductivity of the wall conduction material
% [W/mK]
% L_wall --> Length of the wall conduction path [m]
% A_regen --> Regenerator free flow area [m^2]
% kgas --> Thermal conductivity of the working gas [W/mK]
% L_regen --> Length of the regenerator [m]
% A_displacer --> Cross-sectional area of the displacer [m^2]
% k_displacer --> Effective thermal conductivity of the displacer [W/m^2]
% L_displacer --> Length of the displacer [m]

% % Inputs for Original ST05G-CNC Stirling Engine
% A_wall = 0.25*pi*((0.1397^2)-(0.134^2)+(0.0976^2)-(0.096^2)); % [m^2]
% k_wall = 25; % Stainless steel [W/m^2K] (Temperature dependent)
% L_wall = 0.05651; %[m] Length of regenerator cavity in this case
% A_regen = 0.1*0.25*pi*((0.134^2)-(0.0976^2)); %[m^2] Area x porosity
% L_regen = 0.05651; %[m]
% A_displacer = 0.25*pi*(0.094^2); %[m^2]
%  k_displacer  =  (0.25*pi)*(1/A_displacer)*(((0.0924^2)*(kgas))+(((0.094^2)-
(0.0924^2))*k_wall)); %[W/mK]
% L_displacer = 0.15405; %[m]

% % Inputs for As Built HTG Stirling Engine
A_wall = 0.25*pi*((0.1465^2)-(0.1355^2)+(0.103^2)-(0.0956^2)); % [m^2]
k_wall = 47; % AISI 1020 steel [W/m^2K] (From Solidworks)
L_wall = 0.055; %[m] Length of regenerator cavity in this case
A_regen = 0.25*pi*((0.1355^2)-(0.103^2)); %[m^2] Area or regenerator
k_regen = 0.9*ENGINE_DATA.kgas + 0.1*(19); % Porosity weighted working gas +
316 stainless steel.
L_regen = 0.055; %[m]
A_displacer = 0.25*pi*(0.094^2); %[m^2]
k_displacer                                                    =
(0.25*pi)*(1/A_displacer)*(((0.0924^2)*(ENGINE_DATA.kgas))+(((0.094^2)-
(0.0924^2))*19)); %[W/mK]
L_displacer = (0.15613 + 0.14513)/2; %[m]

% Conduction through the engine walls
Q_wall = (A_wall*k_wall*(ENGINE_DATA.Tgh-ENGINE_DATA.Tgk))/L_wall; %[W]
```

```
% Conduction through the regenerator
Q_fluid = (A_regen*k_regen*(ENGINE_DATA.Tgh-ENGINE_DATA.Tgk))/L_regen; %[W]

% Conduction through the displacer
Q_displacer = (A_displacer*k_displacer*(mean([REF_CYCLE_DATA.Tge])...
    -mean([REF_CYCLE_DATA.Tgc])))/L_displacer; %[W]

% Conduction through the regenerator material

% Total Conduction Loss [W]
LOSSES_DATA.Q_cond = Q_wall + Q_fluid + Q_displacer;


%% Gas Spring Hysteresis Losses
% Input parameters needed:
% Tw --> gas spring wall temperature [K]
% delV --> volume amplitude of the spring cavity [m^3]
% VB --> mean volume of the gas spring cavity [m^3]
% Aw --> mean wetted area of the gas spring [m^2]
% delA --> wetted area amplitude [m^2]

% % Inputs for Original ST05G-CNC Stirling Engine
% Tw = ENGINE_DATA.Tgk; %[K] (assumed to be equal to the cooler temperature)
% delV = 0.25*pi*(0.085^2)*0.075; %[m^3]
% VB = 0.5*delV + 0.002488647 + 0.000091248 + 0.000419874 + 0.000186558 + ...
%     0.000182367 - 0.000229935 - 0.000061378 - 0.000059277 - ...
%     0.000007376 - 0.000021882 - 0.000013195 - 0.000009817 - ...
%     0.000009173 - 0.000231607; %[m^3]
% delA = pi*0.085*0.075; %[m^2]
% Aw = 0.5*delA + 0.05853681 + 0.00804849 + 0.01866106 + 0.01030158 + ...
%     0.02145481 + 0.03279471 + 0.01778246 + 0.02301812 + 0.00211262 + ...
%     0.007294024 + 0.00026389 + 0.00045239 + 0.02173007; %[m^2]
%
% % Calculation of the gas spring hysteresis loss
% F1 = sqrt(0.125*ENGINE_DATA.omega*ENGINE_DATA.gamma*...
% (ENGINE_DATA.gamma-1)*Tw*ENGINE_DATA.pmean*ENGINE_DATA.kgas);
% F2 = (delV/VB)*((0.5*ENGINE_DATA.gamma*(delV/VB)*Aw) - delA);
%
% P_hys = F1*F2*ENGINE_DATA.freq; % Power lost to gas spring hysteresis in
[W].

% Empirical Formula Based on Experimental Data
LOSSES_DATA.P_GSH                                                    =
HTG_GSH_Empirical(ENGINE_DATA.freq,ENGINE_DATA.pmean,ENGINE_DATA.GSH_config);
%[W]


%% Seal Leakage Losses
% Input parameters needed:
% mloss_outside --> Working fluid loss rate due to leakage to the outside
[Pa/s]
% isentropic_eff_comp --> Isentropic efficiency of the compressor
% h_ambient --> Enthalpy of the air entering the compressor in [J/kg]
% h2_isentropic --> Discharge enthalpy of the compressor in [J/kg]
```

```
% Inputs for HTG Stirling Engine
mloss_outside = 0; % [kg/s]
isentropic_eff_comp = 0.75; % [unitless]
h_ambient = 298180; % [J/kg]
h2_isentropic = 575590; % [J/kg]

% Piston Leakage

% Displacer Leakage

% Leakage to the outside (adds compressor work for a pressurized engine)
% *** This will not affect the measured engine output if the engine is not
% driving the compressor.
% Power consumed by the compressor to maintain engine pressure
P_compressor  =  (mloss_outside*(h2_isentropic-h_ambient))/isentropic_eff_comp;
%[W]

% Total leakage loss effects
LOSSES_DATA.P_seals = 0; % Power lost due to seal leakage in [W].
LOSSES_DATA.Q_seals = 0; % Heat lost due to seal leakage in [W].


%% Appendix Gap Losses (Pg 140 of Urieli and Berchowitz)
% Input parameters needed:
% D --> diameter of the displacer cylinder ??? [m]
% kgas --> thermal conductivity of the working fluid [W/mK]
% Xd --> displacer motion amplitude (displacer stroke) [m]
% upsilon --> linear axial temperature gradient (bad assumption in this case
since cooler is a different material) [K/m]
% h --> appendix gap width(distance b/w displacer wall and cylinder wall) [m]
% omega --> engine angular frequency [rad/s]
% phi --> phase angle b/w displacer motion and pressure variation [rad]
% alpha_solid --> thermal diffusivity of the displacer and cylinder material
in [m^2/s]
% k_solid --> thermal conductivity of the displacer and cylinder material in
[W/m.K]

% Inputs for HTG Stirling Engine
D = 0.096; %[m]
Xd = 0.075; %[m]
upsilon  =  (ENGINE_DATA.Tgh-ENGINE_DATA.Tgk)/0.220;  %[K/m]  (Divided by length
of displacer cylinder)
h = 0.001; %[m]
alpha_solid = 1.172e-05; %[m^2/s] (Steel from Wikipedia)
k_solid  =  47;  %[W/m.K]  (AISI  1020  Steel  from  Solidworks  -  Displacer  is
Stainless so this should be refined)

% Crank angle at which minimum workspace pressure occurs
[~,pmax_index] = max([REF_CYCLE_DATA.p]);
pmaxAngle = pmax_index*crank_inc - crank_inc; %[degrees]

% Crank angle at which expansion space volume is minimum
[~,Vemin_index] = min([REF_CYCLE_DATA.Ve]);
DispTDCAngle = Vemin_index*crank_inc - crank_inc; %[degrees]
```

```
% Crank angle by which pressure variation leads the displacer motion
phi = (pmaxAngle-DispTDCAngle); %[degrees]

pressure = [REF_CYCLE_DATA.p];
delp = max(pressure) - min(pressure); %[Pa]

w = sqrt(ENGINE_DATA.omega/(2*alpha_solid));

A1 = -pi*0.5*D*ENGINE_DATA.kgas*(Xd^2)*(upsilon/h);
A2 = pi*D*delp*h*Xd*ENGINE_DATA.omega*sin(phi);
A3                  =                  (ENGINE_DATA.gamma/(ENGINE_DATA.gamma-
1))*log(ENGINE_DATA.Tgh/ENGINE_DATA.Tgk)*...
    (0.5-(ENGINE_DATA.kgas/(w*h*k_solid))-0.5);

LOSSES_DATA.Q_app = abs(A1 + (A2*A3)); % Heat lost due to appendix gap in
[W].


%% Heat Transfer Hysteresis
LOSSES_DATA.P_HTH = 0; % Power lost to heat transfer hysteresis in [W].


%% Finite Piston Speed Loss
% % Equations come from Petrescu 2002 and quasi-steady approach was inspired
% % by Hosseinzade 2015
%
% % Calculate the constant quantity "a"
% a = sqrt(3*ENGINE_DATA.gamma);
%
% % Calculate vectors of average molecular speeds corresponding to the
% % vector of crank angle increments in (m/s)
% cc = sqrt(3*ENGINE_DATA.R.*[REF_CYCLE_DATA.Tgc]); % Compression space
% ce = sqrt(3*ENGINE_DATA.R.*[REF_CYCLE_DATA.Tge]); % Expansion space
%
% % Calculate vectors of piston speeds in (m/s)
% Dr1 = ENGINE_DATA.Dr1;
% Dr2 = ENGINE_DATA.Dr2;
% Dr3 = ENGINE_DATA.Dr3;
% Dbore = ENGINE_DATA.Dbore;
%
% Pr1 = ENGINE_DATA.Pr1;
% Pr2 = ENGINE_DATA.Pr2;
% Pr3 = ENGINE_DATA.Pr3;
% Pbore = ENGINE_DATA.Pbore;
%
% Ptheta2 = pi - [REF_CYCLE_DATA.theta]; %(rad)
% Dtheta2 = Ptheta2 - ENGINE_DATA.beta; %(rad)
%
% Ptheta2_dot = -ENGINE_DATA.omega; %(rad/s)
% Dtheta2_dot = -ENGINE_DATA.omega; %(rad/s)
%
% Ptheta3 = pi - asin((-Pr1+(Pr2.*sin(Ptheta2)))./Pr3); %(rad)
% Dtheta3 = pi - asin((-Dr1+(Dr2.*sin(Dtheta2)))./Dr3); %(rad)
%
% wp = (Pr2*Ptheta2_dot.*sin(Ptheta3-Ptheta2))./cos(Ptheta3); %(m/s)
```

```
% wd = (Dr2*Dtheta2_dot.*sin(Dtheta3-Dtheta2))./cos(Dtheta3); %(m/s)
%
% % Calculate incremental volume changes caused by the piston and displacer
%          dPtheta3          =          (Pr2.*cos(Ptheta2))./(Pr3.*sqrt(1-(((-
Pr1+(Pr2.*sin(Ptheta2)))./Pr3).^2)));
% dPr4 = Pr2.*sin(Ptheta2) + Pr3.*sin(Ptheta3).*dPtheta3;
% dVp = -(pi/4)*(Pbore^2).*dPr4;
%
%       dDtheta3       =               (Dr2.*cos(Dtheta2))./(Dr3.*sqrt(1-(((-
Dr1+(Dr2.*sin(Dtheta2)))./Dr3).^2)));
% dDr4 = Dr2.*sin(Dtheta2) + Dr3.*sin(Dtheta3).*dDtheta3;
% dVe = -(pi/4)*(Dbore^2).*(dDr4);
%
% % Calculate vectors of lost work
% dWp = [REF_CYCLE_DATA.p].*(-a*wp./cc).*dVp;
%
% % Calculate total work lost to finite piston speed
% W_FPS = sum(dWp);
%
% % Calculate total power lost to finite piston speed
% LOSSES_DATA.P_FPS = W_FPS*ENGINE_DATA.freq; %(W)
LOSSES_DATA.P_FPS = 0; % Power loss due to relative motion between piston and
gas molecules in [W].


%% Cooling/Heating System Pumping Power
% Some of the power produced by the engine must be used to drive the
% coolant pump (unless a heat pipe or other passive cooling method is used).
% A fuel pump or hot side water pump may also be needed.

% Inputs:
% rho_cool --> density of coolant in (kg/m^3)
% g --> gravitational acceleration in (m/s^2)
% V_dot_cool --> volume flow rate of coolant in (m^3/s)
% H_cool --> coolant pump net head in (m)
% eff_pump_cool --> coolant pump efficiency

% P_pump_cool = (rho_cool*g*V_dot_cool*H_cool)/eff_pump; %(W)

LOSSES_DATA.P_pump_cool = 0; % Power required to drive the coolant pump in
(W).

LOSSES_DATA.P_pump_hot = 0; % Power required to drive fuel pump or hot water
pump in [W].


%% Exergy Destruction due to Heat Conduction at the Thermal Source/Sink
Interfaces
% The temperature drop between the heater/cooler gas temperature and the
% thermal source/sink causes exergy destruction.

% Calculations follow an example on Pg 446 of Cengel Thermodynamics
% Assumptions:
% 1. Steady state conduction process.
% 2. 1-dimensional heat conduction.
```

```
% 3. The exergy of the wall itself is constant, since it does not change
% state.

% Average heat transfer rates in the heater and cooler in (W)
Qh_dot_avg = mean([REF_CYCLE_DATA.Qh]*ENGINE_DATA.freq);
Qk_dot_avg = mean([REF_CYCLE_DATA.Qk]*ENGINE_DATA.freq);

% Average gas temperatures in the heater and cooler in (K)
Tgh_avg = mean([ENGINE_DATA.Tgh]);
Tgk_avg = mean([ENGINE_DATA.Tgk]);

% Temperatures of the thermal source and sink in (K)
Tsource = ENGINE_DATA.Tsource;
Tsink = ENGINE_DATA.Tsink;

T0 = ENGINE_DATA.Tsink; % Temperature of the environment in (K)

% Calculate exergy destruction in the heater and cooler in (W)
LOSSES_DATA.X_dest_source   =   Qh_dot_avg*(1-(T0/Tsource))   -   Qh_dot_avg*(1-
(T0/Tgh_avg));

LOSSES_DATA.X_dest_sink    =    Qk_dot_avg*(1-(T0/Tsink))    -    Qk_dot_avg*(1-
(T0/Tgk_avg));


%% Heat lost through the heating cap insulation
% LOSSES_DATA.Q_insulation = 135; % Measured at 300 deg C Cap Temperature (W)
LOSSES_DATA.Q_insulation = 90; % Measured at 200 deg C Cap Temperature (W)
```

Vary.m

```
function DATA_STRUCTURE = Vary(varName,Min,Max,Increment,ENGINE_DATA,...
    crank_inc,Model_Code,Losses_Code,part_traj_on_off)

% Written by Connor Speer - April 2017
% Editted by Shahzeb Mirza - October 2017

% The purpose of this script is to run the SEA code repeatedly for a
% variety of any given variable, and return results.

% Inputs:
% varName --> String with the exact variable name

% Min --> Lower bound of the variable range being considered [any unit]
% Max --> Upper bound of the variable range being considered [any unit]
% Increment --> Increment size for variable range [any unit

%  ENGINE_DATA  -->  Structure  which  defines  engine  geometry  and  operating
conditions

% Crank_Angle_Increment --> Crank angle step size for numerical integration
in [degrees]
```

```
% Model_Code --> Code specifying which model to run.
% 1 = Ideal Isothermal Model
% 2 = Ideal Adiabatic Model
% 3 = Simple Model
% Losses_Code --> Code specifying which losses subfunction to run
% 1 = parasitic_losses_GEN.m
% 2 = parasitic_losses_HTG.m

range = Min:Increment:Max;

% Preallocate space
DATA_STRUCTURE(length(range)).REF_CYCLE_DATA = [];
DATA_STRUCTURE(length(range)).LOSSES_DATA = [];

counter = 1; % Initialize counter variable

for i = range
    command = strcat('ENGINE_DATA.',varName,' = i;');
    eval(command);

    % Call the SEA code as a subfunction
    [REF_CYCLE_DATA,  LOSSES_DATA,  PARTICLE_TRAJECTORY_DATA,  ENGINE_DATA]  =
...
    sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);

    % Store outputs from the SEA code
    DATA_STRUCTURE(counter).REF_CYCLE_DATA = REF_CYCLE_DATA;
    DATA_STRUCTURE(counter).LOSSES_DATA = LOSSES_DATA;

    % Close the temporary data file to prevent Matlab errors
    fclose('all');

    counter = counter + 1;
end
```

Miscellaneous Functions

rk4.m

```
function [x, y, dy] = rk4(n,x,dx,y,ENGINE_DATA)
% Classical fourth order Runge-Kutta method
% Integrates n first order differential equations
% dy(x,y) over interval x to x+dx
% Israel Urieli - Jan 21, 2002
x0 = x;
y0 = y;
% Modifications made by Steven Middleton
% (1) eliminated for loops in favor of matrix operations (faster)
% ... previous code maintained in comments

% Comments: Is it necassary to pass y variable through deriv? It is
% unused. Also, if one has the function handle feval is unnecessary.

% Modifications made by Connor Speer, October 2017:
% Got rid of feval in favour of calling dadiab directly.
% Now uses the ENGINE_DATA structure.

[y,dy1] = dadiab(x0,y,ENGINE_DATA);

y(1:n) = y0(1:n) + 0.5*dx*dy1(1:n);

xm = x0 + 0.5*dx;
[y,dy2] = dadiab(xm,y,ENGINE_DATA);

y(1:n) = y0(1:n) + 0.5*dx*dy2(1:n);

[y,dy3] = dadiab(xm,y,ENGINE_DATA);
y(1:n) = y0(1:n) + dx*dy3(1:n);

x = x0 + dx;
[y,dy] = dadiab(x,y,ENGINE_DATA);
dy = (dy1 + 2*(dy2 + dy3) + dy)/6;
y(1:n) = y0(1:n) + dx*dy(1:n);
%
% function [x, y, dy] = rk4(deriv,n,x,dx,y)
% % Classical fourth order Runge-Kutta method
% % Integrates n first order differential equations
% % dy(x,y) over interval x to x+dx
% % Israel Urieli - Jan 21, 2002
% x0 = x;
% y0 = y;
%
% [y,dy1] = feval(deriv,x0,y);
% for i = 1:n
%   y(i) = y0(i) + 0.5*dx*dy1(i);
% end
% xm = x0 + 0.5*dx;
% [y,dy2] = feval(deriv,xm,y);
% for i = 1:n
```

```
%   y(i) = y0(i) + 0.5*dx*dy2(i);
% end
% [y,dy3] = feval(deriv,xm,y);
% for i = 1:n
%   y(i) = y0(i) + dx*dy3(i);
% end
% x = x0 + dx;
% [y,dy] = feval(deriv,x,y);
% for i = 1:n
%   dy(i) = (dy1(i) + 2*(dy2(i) + dy3(i)) + dy(i))/6;
%   y(i) = y0(i) + dx*dy(i);
% end
```

mass_correction.m

```
function ENGINE_DATA = mass_correction(ENGINE_DATA,crank_inc,Model_Code)

% Written by Connor Speer and Shahzeb Mirza, October 2017.

% This function uses the iterative scheme proposed by "2015(Paul) Modeling a
% complete Stirling engine", to correct the mass of working fluid so that
% the reference cycle simulation has the same mean pressure as the one
% specified by the user in ENGINE_DATA.


% Find pressure as a function of crank angle using the initial mass value
% in the ENGINE_DATA structure, which is calculated in define.m using the
% Schmidt analysis.
if Model_Code == 1
    % Solve the ideal isothermal model
    ISOTHERMAL_DATA = isothermal(ENGINE_DATA,crank_inc);
    REF_CYCLE_DATA = ISOTHERMAL_DATA;
elseif Model_Code == 2
    % Solve the ideal adiabatic model
    ADIABATIC_DATA = adiabatic(ENGINE_DATA,crank_inc);
    REF_CYCLE_DATA = ADIABATIC_DATA;
elseif Model_Code == 3
    % Solve the simple model (no flow friction)
    SIMPLE_DATA = simple(ENGINE_DATA,crank_inc);
    REF_CYCLE_DATA = SIMPLE_DATA;
else
    disp('ERROR: Invalid model code.')
end

% Find the mean pressure of the reference cycle
pbar = mean([REF_CYCLE_DATA.p]); %(Pa)

% Recall the mean pressure specified in the ENGINE_DATA structure.
pmean = ENGINE_DATA.pmean; %(Pa)

% If this 'found' mean pressure is significantly different from the
% 'desired' mean pressure specified in the ENGINE_DATA structure, then
% iteratively loop through an algorithm that changes the mass value until
% the mean pressure calculated by the reference cycle matches the desired
```

```
% value.

Mold = ENGINE_DATA.mgas; %(kg)

while( abs((pmean - pbar)/pmean) > 0.001 )
    Mnew = (1 + (pmean - pbar)/pmean) * Mold;
    Mold = Mnew;
    ENGINE_DATA.mgas = Mnew;
    if Model_Code == 1
        % Solve the ideal isothermal model
        ISOTHERMAL_DATA = isothermal(ENGINE_DATA,crank_inc);
        disp(abs((pmean - pbar)/pmean));
        REF_CYCLE_DATA = ISOTHERMAL_DATA;
    elseif Model_Code == 2
        % Solve the ideal adiabatic model
        ADIABATIC_DATA = adiabatic(ENGINE_DATA,crank_inc);
        REF_CYCLE_DATA = ADIABATIC_DATA;
    elseif Model_Code == 3
        % Solve the simple model (no flow friction)
        SIMPLE_DATA = simple(ENGINE_DATA,crank_inc);
        REF_CYCLE_DATA = SIMPLE_DATA;
    else
        disp('ERROR: Invalid model code.')
    end

    % Find the mean pressure of the reference cycle
    pbar = mean([REF_CYCLE_DATA.p]);
end
```

Revised_Crank_Vol.m

```
% Crank_Vol.m - Written by Connor Speer - February 2017
% This script calculates the volume variations for engines which use slider
% crank mechanisms.
% See page 162 in Cleghorn's second edition of "Mechanics of Machines".

% Edited by Connor Speer - September 2017
% Changed angle definitions in the gamma part to make them more clear for
thesis
% paper.

clear, clc, close all;

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Gamma Engines
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Gamma Engine Input Parameters
% Dr1 --> Displacer desaxe offset in [m]
% Dr2 --> Displacer crank length in [m]
% Dr3 --> Displacer connecting rod length in [m]
% Dbore --> Displacer bore in [m]
% Pr1 --> Piston desaxe offset in [m]369
% Pr2 --> Piston crank length in [m]
% Pr3 --> Piston connecting rod length in [m]
```

```
% Pbore --> Piston bore in [m]
% Vdead --> Total dead volume in [m^3]
% V_crankcase_max --> Maximum crankcase volume in [m^3]
% beta_deg --> Phase angle advance of displacer motion over piston motion
[degrees]

% Input Parameters for HTG Engine
Dr1 = 0; %[m]
Dr2 = 0.0375; %[m]
Dr3 = 0.130; %[m]
Dbore = 0.096; %[m]
Pr1 = 0; %[m]
Pr2  = 0.0375; %[m]
Pr3 = 0.156; %[m]
Pbore = 0.085; %[m]
Vdead = 579.09*1e-6; %[m^3]
V_crankcase_max = 0.0032; %[m^3]
beta_deg = 90; %[degrees]

% Specify the crank angle range and increment
%*** Crank angle is zero when total volume is maximum for gamma engines
theta_deg = 0:5:360; %[degrees]
Ptheta2 = pi - theta_deg*(pi/180); %[rad]
Dtheta2 = Ptheta2 - beta_deg*(pi/180); %[rad]

% Working Space Volume Variations

Dr4max = sqrt(((Dr2+Dr3)^2)-(Dr1^2));
Dr4min = sqrt(((Dr3-Dr2)^2)-(Dr1^2));
Dstroke = Dr4max - Dr4min;
Vswd = ((pi/4)*(Dbore^2))*Dstroke;

Pr4max = sqrt(((Pr2+Pr3)^2)-(Pr1^2));
Pr4min = sqrt(((Pr3-Pr2)^2)-(Pr1^2));
Pstroke = Pr4max - Pr4min;
Vswp = ((pi/4)*(Pbore^2))*Pstroke;

Dtheta3 = pi - asin((-Dr1+(Dr2*sin(Dtheta2)))/Dr3);
Dr4 = Dr2*cos(Dtheta2) - Dr3*cos(Dtheta3);
Ve = ((pi/4)*(Dbore^2))*(Dr4max-Dr4);
DVc = (((pi/4)*(Dbore^2))*Dstroke) - Ve;

Ptheta3 = pi - asin((-Pr1+(Pr2*sin(Ptheta2)))/Pr3);
Pr4 = Pr2*cos(Ptheta2) - Pr3*cos(Ptheta3);
PVc = (((pi/4)*(Pbore^2))*(Pr4max-Pr4));
Vc = DVc + PVc;

Vtotal = Vdead + (((pi/4)*(Dbore^2))*2*Dr2) + PVc;

% Crankcase Volume Variations
V_crankcase = V_crankcase_max - PVc;

% Sinusoidal Volume Variations for Comparison
theta = theta_deg*(pi/180);
Ve_sine = Vswd*0.5*(1+cos(theta+(beta_deg*(pi/180))));
```

```
Vc_sine = (Vswd-Ve_sine) + Vswp*0.5*(1+cos(theta));
Vtotal_sine = Vdead + Ve_sine + Vc_sine;

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Crankcase Pressure Models
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SHR = 1.4; % Specific heat ratio of air.
% P0 = 1000000; % Engine fill pressure in [Pa].
% T_crankcase = 298; % Mean crankcase temperature in [K].
% R = 287; % Gas constant for air in [J/kgK].
%
% % Mean crankcase volume in [m^3].
% V0 = V_crankcase_max - 0.5*Pstroke*(((pi/4)*(Pbore^2)));
%
% % Mass of gas in the crankcase in [kg].
% m_crankcase = (P0*V0)/(R*T_crankcase);
%
% % Crankcase Pressure Assuming Adiabatic Compression and Expansion
% P_crankcase_adiabatic = P0.*((V0./V_crankcase).^SHR); % Crankcase pressure
in [Pa].
%
% % Crankcase Pressure Assuming Isothermal Compression and Expansion
% P_crankcase_isothermal = (m_crankcase*R*T_crankcase)./V_crankcase;

% figure
% hold on
% plot(theta_deg, P_crankcase_adiabatic*1e-5)
% plot(theta_deg, P_crankcase_isothermal*1e-5)
% hold off
% xlabel('Crank Angle [degrees]')
% ylabel('Crankcase Pressure [Pa]')
% legend('Adiabatic Crankcase', 'Isothermal Crankcase')

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Alpha Engines
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Alpha Engine Input Parameters
% % Cr1 --> Compression piston desaxe offset in [m]
% % Cr2 --> Compression piston crank length in [m]
% % Cr3 --> Compression piston connecting rod length in [m]
% % Cbore --> Compression piston bore in [m]
% % Er1 --> Expansion piston desaxe offset in [m]
% % Er2 --> Expansion piston crank length in [m]
% % Er3 --> Expansion piston connecting rod length in [m]
% % Ebore --> Expansion piston bore in [m]
% % Vdead --> Total dead volume in [m^3]
% % V_crankcase_max --> Maximum crankcase volume in [m^3] (i.e. both pistons
% % at TDC. This may not be possible when assembled)
% % alpha_deg --> Phase angle advance of expansion space over compression
space [degrees]
%
% % Made up numbers for testing
% Cr1 = 0; %[m]
% Cr2  = 0.0375; %[m]
% Cr3 = 0.156; %[m]
% Cbore = 0.085; %[m]
```

```
% Er1 = 0; %[m]
% Er2 = 0.0375; %[m]
% Er3 = 0.156; %[m]
% Ebore = 0.096; %[m]
% Vdead = 579.09*1e-6; %[m^3]
% V_crankcase_max = 0.0032; %[m^3]
% alpha_deg  = 90; %[degrees]
%
% % Specify the crank angle range and increment
% % *** Crank angle is zero when Vc is maximum for alpha engines
% % *** Be careful defining crank angle 0 for desaxe offset alphas
% theta_deg = 0:5:360;
% theta = theta_deg*(pi/180); %[rad]
%
% Ctheta2 = theta - pi;
% Etheta2 = Ctheta2 + alpha_deg*(pi/180);
%
% % Working Space Volume Variations
% Cr4max = sqrt(((Cr2+Cr3)^2)-(Cr1^2));
% Cr4min = sqrt(((Cr3-Cr2)^2)-(Cr1^2));
% Cstroke = Cr4max - Cr4min;
% CVswept = ((pi/4)*(Cbore^2))*Cstroke;
%
% Er4max = sqrt(((Er2+Er3)^2)-(Er1^2));
% Er4min = sqrt(((Er3-Er2)^2)-(Er1^2));
% Estroke = Er4max - Er4min;
% EVswept = ((pi/4)*(Ebore^2))*Estroke;
%
% Ctheta3 = pi - asin((-Cr1+(Cr2*sin(Ctheta2)))/Cr3);
% Cr4 = Cr2*cos(Ctheta2) - Cr3*cos(Ctheta3);
% Vc = (((pi/4)*(Cbore^2))*(Cr4max-Cr4));
%
% Etheta3 = pi - asin((-Er1+(Er2*sin(Etheta2)))/Er3);
% Er4 = Er2*cos(Etheta2) - Er3*cos(Etheta3);
% Ve = (((pi/4)*(Ebore^2))*(Er4max-Er4));
%
% Vtotal = Vdead + Ve + Vc;
%
% % Crankcase Volume Variations
% V_crankcase = V_crankcase_max - Vc - Ve;
%
% % Sinusoidal Volume Variations for Comparison
% Vc_sine = CVswept*0.5*(1+cos(theta));
% Ve_sine = EVswept*0.5*(1+cos(theta+(alpha_deg*(pi/180))));
% Vtotal_sine = Vdead + Ve_sine + Vc_sine;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plots
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
set(0,'defaultfigurecolor',[1 1 1])

% % For Connor's Thesis Paper
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
```

```
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Arial';
% font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 326;
height = 275;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% % For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% Pro Tips %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% To change line width for plots, say "'LineWidth',2".

% To include math symbols with their own font size and font type in the
% axes labels, do this:
% xlabel({'\fontsize{11} Engine Speed \fontsize{11} \fontname{Cambria Math}
\omega \fontsize{11} \fontname{Times New Roman} [RPM]'});

% Here is the degree symbol in case you need it °C. Alternatively, use
% \circ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot 1 - Total Working Space Volume as a Function of Crank Angle
figure('Position', [x y width height])
hold on
plot(theta_deg, Vtotal,'r','linewidth',2)
plot(theta_deg,Vtotal_sine,'k','linewidth',2)
xlim([0 360])
hold off
```

```
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
ylabel('Total Workspace Volume (m^3)','FontName',font,'FontSize',font_size)
legend('Slider-Crank','Sinusoidal','Location','Southeast')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

% Plot 2 - Piston Position as a Function of Crank Angle
figure('Position', [x y width height])
hold on
plot(theta_deg, max(Vtotal)/((pi/4)*(Pbore^2)) -
Vtotal/((pi/4)*(Pbore^2)),'r','linewidth',2)
plot(theta_deg,max(Vtotal_sine)/((pi/4)*(Pbore^2)) -
Vtotal_sine/((pi/4)*(Pbore^2)),'k','linewidth',2)
xlim([0 360])
hold off
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
ylabel('Piston Position (m)','FontName',font,'FontSize',font_size)
legend('Slider-Crank','Sinusoidal','Location','Northeast')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

% Plot 3 - Displacer Position as a Function of Crank Angle
figure('Position', [x y width height])
hold on
plot(theta_deg,max(Ve)/((pi/4)*(Dbore^2)) -
(Ve/((pi/4)*(Dbore^2))),'r','linewidth',2)
plot(theta_deg,max(Ve_sine)/((pi/4)*(Dbore^2)) -
(Ve_sine/((pi/4)*(Dbore^2))),'k','linewidth',2)
xlim([0 360])
hold off
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
ylabel('Displacer Position (m)','FontName',font,'FontSize',font_size)
legend('Slider-Crank','Sinusoidal','Location','Northeast')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
%
% figure('Position', [x y width height])
% plot(theta_deg, V_crankcase*1000)
% xlabel('Crank Angle [degrees]','FontName',font,'FontSize',font_size)
% ylabel('Crankcase Volume [L]','FontName',font,'FontSize',font_size)

% % Plot 1 - Total Working Space Volume as a Function of Crank Angle
% figure('Position', [x y width height])
% hold on
% plot(theta_deg, Vtotal*1000,'r','linewidth',2)
% plot(theta_deg,Vtotal_sine*1000,'k','linewidth',2)
% xlim([0 360])
% hold off
% xlabel('Crank Angle [\circ]','FontName',font,'FontSize',font_size)
% ylabel('Total Workspace Volume [L]','FontName',font,'FontSize',font_size)
% legend('Slider-Crank','Sinusoidal','Location','Southeast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
%
% % Plot 2 - Compression Piston Position as a Function of Crank Angle
% figure('Position', [x y width height])
```

```
% hold on
% plot(theta_deg, max(Vc)/((pi/4)*(Cbore^2)) -
Vc/((pi/4)*(Cbore^2)),'r','linewidth',2)
% plot(theta_deg,max(Vc_sine)/((pi/4)*(Cbore^2)) -
Vc_sine/((pi/4)*(Cbore^2)),'k','linewidth',2)
% xlim([0 360])
% hold off
% xlabel('Crank Angle [\circ]','FontName',font,'FontSize',font_size)
% ylabel('Compression Piston Position
[m]','FontName',font,'FontSize',font_size)
% legend('Slider-Crank','Sinusoidal','Location','Northeast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
%
% % Plot 3 - Expansion Piston Position as a Function of Crank Angle
% figure('Position', [x y width height])
% hold on
% plot(theta_deg,max(Ve)/((pi/4)*(Ebore^2)) -
(Ve/((pi/4)*(Ebore^2))),'r','linewidth',2)
% plot(theta_deg,max(Ve_sine)/((pi/4)*(Ebore^2)) -
(Ve_sine/((pi/4)*(Ebore^2))),'k','linewidth',2)
% xlim([0 360])
% hold off
% xlabel('Crank Angle [\circ]','FontName',font,'FontSize',font_size)
% ylabel('Expansion Piston Position
[m]','FontName',font,'FontSize',font_size)
% legend('Slider-Crank','Sinusoidal','Location','Northeast')
% set(gca,'fontsize',font_size);
% set(gca,'FontName',font)
%
% figure('Position', [x y width height])
% plot(theta_deg, V_crankcase*1000)
% xlabel('Crank Angle [degrees]','FontName',font,'FontSize',font_size)
% ylabel('Crankcase Volume [L]','FontName',font,'FontSize',font_size)
```

Appendix D: MATLAB Source Code for Data Processing

General Data Processing Functions

Process_Data.m

```
function DATA_STRUCTURE = Process_Data(log_file_folder)

% Written by Connor Speer - October 2017
% Process experimental data and return it in a structure for plotting
% elsewhere.

% Calculate Correction Terms Using Calibration Data
Abs_Val_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Calibration Codes\Validyne
Calibration Oct. 22';
[ABS_VAL_FIT_COEFFICIENTS]...
    = ABS_VAL_Correction_Terms(Abs_Val_Cal_Folder, 0);

Diff_Val_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Calibration Codes\Differential
Validyne Calibration Oct. 22';
[DIFF_VAL_FIT_COEFFICIENTS]...
    = DIFF_VAL_Correction_Terms(Diff_Val_Cal_Folder, 0);

TC_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Calibration Codes\Thermocouple Calibration
Oct. 22';
[TC_FIT_COEFFICIENTS]...
    = TC_Correction_Terms(TC_Cal_Folder, 0);

RTD_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Calibration Codes\RTD Calibration Oct.
22';
[RTD_FIT_COEFFICIENTS]...
    = RTD_Correction_Terms(RTD_Cal_Folder, 0);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cooling System Input Parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
V_dot_Cooler = 0.800*1.66667e-5; % Cooler water volume flow rate [m^3/s]
V_dot_ConPipePowCyl = 0.200*1.66667e-5; % Connecting Pipe and Power Cylinder
water volume flow rate [m^3/s]
c_water = 4184; % Specific heat capacity of water in [J/kgK]
dens_water = 1000; % Water density [kg/m^3]
```

```
% Calculations
m_dot_Cooler = V_dot_Cooler*dens_water; % Cooler water mass flow rate [kg/s]
m_dot_ConPipePowCyl = V_dot_ConPipePowCyl*dens_water; % Connecting Pipe water
mass flow rate [kg/s]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Collect all the log file names from the test data folder
log_files_info = dir(fullfile(log_file_folder, '*.log'));

%% Preallocate space for the structure array
DATA_STRUCTURE(length(log_files_info)/3).filename = [];
DATA_STRUCTURE(length(log_files_info)/3).Filename_Hz = [];
DATA_STRUCTURE(length(log_files_info)/3).Filename_P_shaft = [];
DATA_STRUCTURE(length(log_files_info)/3).time_TC = [];
DATA_STRUCTURE(length(log_files_info)/3).time_RTD = [];
DATA_STRUCTURE(length(log_files_info)/3).time_Volt = [];
DATA_STRUCTURE(length(log_files_info)/3).theta_deg = [];
DATA_STRUCTURE(length(log_files_info)/3).T_0 = [];
DATA_STRUCTURE(length(log_files_info)/3).T_1 = [];
DATA_STRUCTURE(length(log_files_info)/3).T_2 = [];
DATA_STRUCTURE(length(log_files_info)/3).T_3 = [];
DATA_STRUCTURE(length(log_files_info)/3).T_4 = [];
DATA_STRUCTURE(length(log_files_info)/3).T_5 = [];
DATA_STRUCTURE(length(log_files_info)/3).T_6 = [];
DATA_STRUCTURE(length(log_files_info)/3).RTD_0 = [];
DATA_STRUCTURE(length(log_files_info)/3).RTD_1 = [];
DATA_STRUCTURE(length(log_files_info)/3).RTD_2 = [];
DATA_STRUCTURE(length(log_files_info)/3).RTD_3 = [];
DATA_STRUCTURE(length(log_files_info)/3).P1_Pa = [];
DATA_STRUCTURE(length(log_files_info)/3).P2_Pa = [];
DATA_STRUCTURE(length(log_files_info)/3).P3_Pa = [];
DATA_STRUCTURE(length(log_files_info)/3).P4_Pa = [];
DATA_STRUCTURE(length(log_files_info)/3).P5_Pa = [];
DATA_STRUCTURE(length(log_files_info)/3).A = [];
DATA_STRUCTURE(length(log_files_info)/3).Z = [];
DATA_STRUCTURE(length(log_files_info)/3).TOR = [];
DATA_STRUCTURE(length(log_files_info)/3).pmean = [];
DATA_STRUCTURE(length(log_files_info)/3).Cap_Setpoint = [];
DATA_STRUCTURE(length(log_files_info)/3).Bath_Setpoint = [];
DATA_STRUCTURE(length(log_files_info)/3).Tge = [];
DATA_STRUCTURE(length(log_files_info)/3).Tgc = [];
DATA_STRUCTURE(length(log_files_info)/3).Tgk = [];
DATA_STRUCTURE(length(log_files_info)/3).Tgh = [];
DATA_STRUCTURE(length(log_files_info)/3).T_cooler_inlet = [];
DATA_STRUCTURE(length(log_files_info)/3).T_cooler_outlet = [];
DATA_STRUCTURE(length(log_files_info)/3).Torque_avg = [];
DATA_STRUCTURE(length(log_files_info)/3).Engine_RPM = [];
DATA_STRUCTURE(length(log_files_info)/3).Engine_Hz = [];
DATA_STRUCTURE(length(log_files_info)/3).P_shaft = [];
DATA_STRUCTURE(length(log_files_info)/3).P_ind_Validyne = [];
DATA_STRUCTURE(length(log_files_info)/3).P_ind_PCB = [];
DATA_STRUCTURE(length(log_files_info)/3).N_Beale = [];
DATA_STRUCTURE(length(log_files_info)/3).N_West = [];
DATA_STRUCTURE(length(log_files_info)/3).N_T = [];
DATA_STRUCTURE(length(log_files_info)/3).Qdot_Cooler = [];
DATA_STRUCTURE(length(log_files_info)/3).Qdot_ConPipePowCyl = [];
```

```
DATA_STRUCTURE(length(log_files_info)/3).Error_isothermal = [];
DATA_STRUCTURE(length(log_files_info)/3).Loss_Factor_isothermal = [];
DATA_STRUCTURE(length(log_files_info)/3).Error_adiabatic = [];
DATA_STRUCTURE(length(log_files_info)/3).Power_adiabatic = [];
DATA_STRUCTURE(length(log_files_info)/3).P_GSH_Loss_Validyne = [];
DATA_STRUCTURE(length(log_files_info)/3).P_GSH_Loss_PCB = [];
DATA_STRUCTURE(length(log_files_info)/3).Exp_FW = [];
DATA_STRUCTURE(length(log_files_info)/3).Tdrop_Cap_to_Expansion_Space = [];
DATA_STRUCTURE(length(log_files_info)/3).P_2nd_order = [];
DATA_STRUCTURE(length(log_files_info)/3).Q_dot_in_2nd_order = [];
DATA_STRUCTURE(length(log_files_info)/3).Q_dot_rej_2nd_order = [];
DATA_STRUCTURE(length(log_files_info)/3).eff_thermal_2nd_order = [];
DATA_STRUCTURE(length(log_files_info)/3).Vtotal_rounded = [];
DATA_STRUCTURE(length(log_files_info)/3).Vtotal = [];
DATA_STRUCTURE(length(log_files_info)/3).p_engine_experimental = [];
DATA_STRUCTURE(length(log_files_info)/3).p_engine_isothermal = [];
DATA_STRUCTURE(length(log_files_info)/3).p_engine_adiabatic = [];
DATA_STRUCTURE(length(log_files_info)/3).p_buffer_experimental = [];
DATA_STRUCTURE(length(log_files_info)/3).p_buffer_isothermal = [];
DATA_STRUCTURE(length(log_files_info)/3).p_buffer_adiabatic = [];
DATA_STRUCTURE(length(log_files_info)/3).pdrop_regen_experimental = [];
DATA_STRUCTURE(length(log_files_info)/3).pdrop_regen_model = [];
DATA_STRUCTURE(length(log_files_info)/3).T_crankcase = [];
DATA_STRUCTURE(length(log_files_info)/3).Q_dot_in = [];
DATA_STRUCTURE(length(log_files_info)/3).eff_thermal = [];
DATA_STRUCTURE(length(log_files_info)/3).Conduction_Loss_Model = [];
DATA_STRUCTURE(length(log_files_info)/3).P_lost_regen_FF = [];
DATA_STRUCTURE(length(log_files_info)/3).P_lost_regen_FF_model = [];

% Initialize counter variable
counter = 1;
counter_max = (1/3)*length(log_files_info);

WaitBar = waitbar(0,'Processing experimental data ...');

%% Open triplets of log files
for i = 1:3:length(log_files_info)

    % Open the RTD log file.
    filename_RTD = strcat(log_file_folder,'\',log_files_info(i).name);
    [RTD_0,RTD_1,RTD_2,RTD_3] = importfile_RTD(filename_RTD);
%     RTD_0 --> Cooler water inlet
%     RTD_1 --> Cooler water outlet
%     RTD_2 --> Con Pipe/Pow Cyl inlet
%     RTD_3 --> Con Pipe/Pow Cyl outlet

    % Open the thermocouple log file.
    filename_TC = strcat(log_file_folder,'\',log_files_info(i+1).name);
    [T_0,T_1,T_2,T_3,T_4,T_5,T_6,T_7,T_8,T_9,T_10,T_11,T_12,T_13] = ...
    importfile_TC(filename_TC);
%     T_0 --> Expansion Space
%     T_1 --> HH-Regen
%     T_2 --> HH Flange - Bypass Side
%     T_3 --> HH Flange - Power Piston Side
%     T_4 --> Displacer Mount
%     T_5 --> Crankcase
```

```
%     T_6 --> Power Piston

    % Open the voltage log file.
    filename_Volt = strcat(log_file_folder,'\',log_files_info(i+2).name);
    [A,Z,TOR,P1,P2,P3,P4,P5] = importfile_P(filename_Volt);
%     A --> 500 PPR Rotary Encoder Output
%     Z --> 1 PPR Rotary Encoder Output
%     TOR --> Torque Transducer Output
%     P1 --> Validyne Regenerator Differential Pressure
%     P2 --> Validyne Power Cylinder
%     P3 --> Validyne Crankcase
%     P4 --> PCB Power Cylinder
%     P5 --> PCB Crankcase

    %% Apply Calibration
    RTD_0 = calibrate_RTD(0,RTD_0,RTD_FIT_COEFFICIENTS); %(deg C)
    RTD_1 = calibrate_RTD(1,RTD_1,RTD_FIT_COEFFICIENTS); %(deg C)
    RTD_2 = calibrate_RTD(2,RTD_2,RTD_FIT_COEFFICIENTS); %(deg C)
    RTD_3 = calibrate_RTD(3,RTD_3,RTD_FIT_COEFFICIENTS); %(deg C)

    % Don't adjust first two thermocouples because calibration data range
    % is too low, and extrpolating it will lead to more error.
%     T_0 = calibrate_TC(0,T_0_raw,TC_FIT_COEFFICIENTS); %(deg C)
%     T_1 = calibrate_TC(1,T_1_raw,TC_FIT_COEFFICIENTS); %(deg C)
    T_2 = calibrate_TC(2,T_2,TC_FIT_COEFFICIENTS); %(deg C)
    T_3 = calibrate_TC(3,T_3,TC_FIT_COEFFICIENTS); %(deg C)
    T_4 = calibrate_TC(4,T_4,TC_FIT_COEFFICIENTS); %(deg C)
    T_5 = calibrate_TC(5,T_5,TC_FIT_COEFFICIENTS); %(deg C)
    T_6 = calibrate_TC(6,T_6,TC_FIT_COEFFICIENTS); %(deg C)

    P1_Pa_raw = P1*0.5*6894.76; %(Pa)
    P1_Pa = calibrate_diff_VAL(1,P1_Pa_raw,DIFF_VAL_FIT_COEFFICIENTS); %(Pa)

    P2_Pa_raw = P2*20*6894.76; %(Pa)
    P3_Pa_raw = P3*20*6894.76; %(Pa)
    P2_Pa = calibrate_abs_VAL(2,P2_Pa_raw,ABS_VAL_FIT_COEFFICIENTS);
    P3_Pa = calibrate_abs_VAL(3,P3_Pa_raw,ABS_VAL_FIT_COEFFICIENTS);

    % Calculate mean pressure from the Validyne transducers
    pmean = mean(P2_Pa); %[Pa]

    [P4, P5] = PCB_Pressure_Calibration(P4.*0.5, P5.*0.5);
    P4_Pa = mean(P2_Pa) + P4; % PCB Power Cylinder
    P5_Pa = mean(P3_Pa) + P5; % PCB Crankcase

    TOR = Futek_Torque_Calibration(TOR);
    Torque_avg = mean(TOR); %[Nm]

    %%  Collect the Start and End Times of the voltage log file
    volt_log_file = fileread(filename_Volt);
    start_index = regexp(volt_log_file,'Start');
    end_index = regexp(volt_log_file,'End');
    start_line = volt_log_file(start_index:end_index-1);
    end_line = volt_log_file(end_index:end_index+14);
    [token_start,remain_start] = strtok(start_line);
```

```
    [token_end,remain_end] = strtok(end_line);
    start_time = str2double(strip(remain_start));
    end_time = str2double(strip(remain_end));

    % Time for voltages
    time_inc_Volt = (end_time-start_time)/length(A);
    N_Volt = length(A);
    time_Volt = (0:N_Volt-1)*time_inc_Volt;
    time_Volt = time_Volt(:);

    % Time for thermocouples
    time_inc_TC = (end_time-start_time)/length(T_0);
    N_TC = length(T_0);
    time_TC = (0:N_TC-1)*time_inc_TC;
    time_TC = time_TC(:);

    % Time for RTDs
    time_inc_RTD = (end_time-start_time)/length(RTD_0);
    N_RTD = length(RTD_0);
    time_RTD = (0:N_RTD-1)*time_inc_RTD;
    time_RTD = time_RTD(:);

    %% Record Data from the name
    % Heater Cap Temperature Setpoint
    [token1,remain1] = strtok(log_files_info(i).name,'TH_');
    Cap_Setpoint = str2double(token1); %[deg C]

    % Coolant Water Bath Setpoint
    [token3,remain3] = strtok(log_files_info(i).name,'TC');
    [token4,remain4] = strtok(remain3,'_');
    token5 = strtok(remain4,'_');
    Bath_Setpoint = str2double(token5); %[deg C]

    % Engine Speed
    [token6,remain6] = strtok(log_files_info(i).name,'D');
    [token7,remain7] = strtok(remain6,'D');
    [token8,remain8] = strtok(remain7,'_');
    [token9,remain9] = strtok(remain8,'_');
    Filename_RPM = str2double(token9); %[RPM]
    Filename_Hz = Filename_RPM/60; %[Hz]

    %% Calculate Data from the file
%     % Convert pressures and average pressures into units of Pa
%     p_scale1 = 0.5*0.0689476*100000; % (1V = 0.5 PSI)
%     p_scale2 = 20*0.0689476*100000; % (1V = 20 PSI)
%     p_scale3 = (1/0.024)*0.0689476*100000*0.5; % 24 mV/PSI, Gain = 2
%
%     P1_Pa = P1.*p_scale1; % Regenerator Differential Validyne
%
%     P2_Pa = P2.*p_scale2; % Validyne Power Cylinder
%     P3_Pa = P3.*p_scale2; % Validyne Crankcase
%
%     % Calculate mean pressure from Validyne transducers
%     pmean = mean(P2_Pa); %[Pa]
%
```

```
%     P4_Pa = mean(P2_Pa) + P4.*p_scale3; % PCB Power Cylinder
%     P5_Pa = mean(P3_Pa) + P5.*p_scale3; % PCB Crankcase

    % Calculate average expansion space gas temperature
    Tge = mean(T_0); %[deg C]

    % Calculate average compression space gas temperature
    Tgc = (mean(T_6)+mean(T_4))/2; %[deg C]

    % Calculate average Cooler Inlet Water Temperature
    T_cooler_inlet = mean(RTD_0); %[deg C]

    % Calculate average Cooler Outlet Water Temperature
    T_cooler_outlet = mean(RTD_1); %[deg C]

%     % Calculate average torque
%     Torque_avg = mean(TOR)*2; %[Nm]

    % Calculate crank angles
    [theta_deg] = Encoder_2_Angle(A,Z); %[deg]
    theta = theta_deg*(pi/180); % [rad]

    % Calculate  Average Crankshaft Speed
    refs = find(A < 0.5);

    % Initialize variables
    pulse_count = 0;

    for inc = 1:(length(refs)-1)
        if refs(inc+1) ~= (refs(inc) + 1)
            pulse_count = pulse_count + 1;
        end
    end

    Engine_Ang_Freq = ((pulse_count/(end_time-start_time))/500)*(2*pi); %
[rad/sec]
    Engine_Hz = Engine_Ang_Freq/(2*pi); %[Hz]
    Engine_RPM = Engine_Ang_Freq*(60/(2*pi)); % [rev/min]

    % Calculate the engine power output
    P_shaft = Torque_avg*Engine_Ang_Freq; %[W]

    % Calculate the engine power output using the speed in the filename
    % (for efficiency data files)
    Filename_P_shaft = Torque_avg*Filename_Hz*2*pi; %[W]

    % Calculate the heat input
    Num_Samples_On =  sum(P1 > 6.99);
    Num_Samples_Total = length(P1);
    OnFraction = Num_Samples_On/Num_Samples_Total;
    Q_dot_in = OnFraction*3290.958; %(W)

    % Calculate the engine thermal efficiency
    eff_thermal = Filename_P_shaft/Q_dot_in;
```

```
    % Calculate the Beale number
    VsweptPP = (426)*1e-6; % Power Piston swept volume in [m^3]
    N_Beale = P_shaft/((pmean)*(Engine_Ang_Freq/(2*pi))*VsweptPP);

    % Calculate the West Number
    T_factor = (Tge+Tgc)/(Tge-Tgc);
    N_West = N_Beale*T_factor;

    % Calculate the Characteristic Temperature Ratio
    N_T = Cap_Setpoint/Bath_Setpoint;

    % Average the Crank Angles and Pressures
    [P1_avg_Pa] = PV_data_avg(theta,P1_Pa);
    [P2_avg_Pa] = PV_data_avg(theta,P2_Pa);
    [P3_avg_Pa] = PV_data_avg(theta,P3_Pa);
    [P4_avg_Pa] = PV_data_avg(theta,P4_Pa);
    [P5_avg_Pa] = PV_data_avg(theta,P5_Pa);

    % Calculate volumes at rounded crank angles to go with average pressures
    %% Use crank angles to calculate volumes
    % Input Parameters for HTG Engine
    Vdead = 876.79*1e-6 - 132*1e-6; %[m^3] (With DV Reduction Parts)
    ENGINE_DATA = HTG_ENGINE_DATA;
    ENGINE_DATA.Pbore = 0.044; %[m]

    theta_deg_rounded = 0:1:359;
    theta_rounded = theta_deg_rounded*(pi/180);
    theta_rounded = theta_rounded(:);
    [Vc_rounded,Ve_rounded,dVc_rounded,dVe_rounded] = volume(theta_rounded,
ENGINE_DATA);

    [Vc,Ve,dVc,dVe] = volume(theta, ENGINE_DATA);

    % Calculate Volumes
    Vtotal_rounded = Vdead + Vc_rounded + Ve_rounded;
    Vtotal_rounded = Vtotal_rounded(:);

    Vtotal = Vdead + Vc + Ve;
    Vtotal = Vtotal(:);

    % Crankcase Volume Variations
    Pbore = ENGINE_DATA.Pbore; % piston bore [m]
    Pr1 = ENGINE_DATA.Pr1; % piston desaxe offset in [m]
    Pr2 = ENGINE_DATA.Pr2; % piston crank radius in [m]
    Pr3 = ENGINE_DATA.Pr3; % piston connecting rod length [m]

    Ptheta2 = pi - theta_rounded;

    Ptheta3 = pi - asin((-Pr1+(Pr2*sin(Ptheta2)))/Pr3);
    Pr4 = Pr2*cos(Ptheta2) - Pr3*cos(Ptheta3);
    Pr4max = sqrt(((Pr2+Pr3)^2)-(Pr1^2));

    % Crankcase Volume Variations in (m^3)
```

```
    V_crankcase_rounded = ENGINE_DATA.V_buffer_max - ((Pr4max-
Pr4)*(((pi/4)*(Pbore^2)))); %(m^3)
    V_crankcase_rounded = V_crankcase_rounded(:);

    % Buffer Pressure Models
    R = 287; % Ideal gas constant for air in (J/kgK)
    gamma = 1.4; % Specific heat ratio for air.
    p0 = mean(P3_Pa); % Mean crankcase pressure in (Pa).
    V0 = mean(V_crankcase_rounded); % Mean crankcase volume in (m^3).
    T_crankcase = mean(T_5) + 273.15; % Mean crankcase temperature in [K].
    m_crankcase = (p0*V0)/(R*T_crankcase); % Mass of gas in the crankcase in
[kg].

    p_buffer_isothermal = (m_crankcase*R*T_crankcase)./V_crankcase_rounded;
%(Pa)
    p_buffer_isothermal = p_buffer_isothermal(1:360);

    p_buffer_adiabatic = p0.*((V0./V_crankcase_rounded).^gamma); %(Pa)
    p_buffer_adiabatic = p_buffer_adiabatic(1:360);

    % Calculate the work lost to regenerator flow friction
    dW_regen_FF = P1_avg_Pa.*(pi/4).*(0.096.^2).*dVe_rounded; %[J]
    W_lost_regen_FF = sum(dW_regen_FF); %[J]
    P_lost_regen_FF = W_lost_regen_FF*Engine_Hz; %[W]



    % Calculate Experimental Indicated Work and Power in the Compression
    % Space according to Validynes
    W_ind_Validyne = polyarea(Vtotal_rounded,P2_avg_Pa); %[J]
    P_ind_Validyne = W_ind_Validyne*Engine_Hz; %[W]

    % Calculate Experimental Indicated Work and Power in the Compression
    % Space according to PCB
    W_ind_PCB = polyarea(Vtotal_rounded,P4_avg_Pa); %[J]
    P_ind_PCB = W_ind_PCB*Engine_Hz; %[W]

    % Calculate Crankcase Gas Spring Hysteresis Loss according to Validyne
    W_GSH_Loss_Validyne = polyarea(Vtotal_rounded,P3_avg_Pa); %[J]
    P_GSH_Loss_Validyne = W_GSH_Loss_Validyne*Engine_Hz; %[W]

    % Calculate Crankcase Gas Spring Hysteresis Loss according to PCB
    W_GSH_Loss_PCB = polyarea(Vtotal_rounded,P5_avg_Pa); %[J]
    P_GSH_Loss_PCB = W_GSH_Loss_PCB*Engine_Hz; %[W]

    % Calculate the rate of heat removed by the cooling system
    Qdot_Cooler = m_dot_Cooler*c_water*(mean(RTD_1)-mean(RTD_0)); %[W]
    Qdot_ConPipePowCyl = m_dot_ConPipePowCyl*c_water*(mean(RTD_3)-
mean(RTD_2)); %[W]

    %% Calculate the Ideal Isothermal Model Indicated power
    ENGINE_DATA = HTG_ENGINE_DATA;
    crank_inc = 1; %[deg]
    Model_Code = 1; % Ideal isothermal model
    Losses_Code = 0; % No loss calculations
    part_traj_on_off = 0; % No particle trajectory calculations
```

```
% Change operating conditions of the engine data structure to match the
% current data set
ENGINE_DATA.Tgk = (((mean(T_2)+mean(T_3))/2)+mean(T_4))/2 + 273.15; %[K]
ENGINE_DATA.Tgh = (mean(T_0)+mean(T_1))/2 + 273.15; %[K]

ENGINE_DATA.Twk = Bath_Setpoint + 273.15; %[K]
ENGINE_DATA.Twh = Cap_Setpoint + 273.15; %[K]

ENGINE_DATA.Tgc = (mean(T_6)+mean(T_4))/2 + 273.15; %[K]
ENGINE_DATA.Tge = mean(T_0) + 273.15; %[K]

ENGINE_DATA.pmean = pmean; %[Pa]

[ISOTHERMAL_DATA, ~, ~] = ...
    sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);

p_engine_isothermal = [ISOTHERMAL_DATA.p]; %(Pa)
p_engine_isothermal = p_engine_isothermal(1:360);

W_ind_isothermal = ISOTHERMAL_DATA(end).W; %(J)
P_ind_isothermal = W_ind_isothermal*Engine_Hz; %(W)

% Calculate the ideal isothermal model Error in Predicting Indicated
% Work
Error_isothermal = ((W_ind_isothermal-W_ind_PCB)/W_ind_PCB)*100; %(%)

% Calculate the ideal isothermal model loss factor
Loss_Factor_isothermal = P_shaft/P_ind_isothermal;

%% Calculate the Ideal Adiabatic Model Indicated Power
ENGINE_DATA = HTG_ENGINE_DATA;
crank_inc = 1; %[deg]
Model_Code = 2; % Ideal adiabatic model
Losses_Code = 0; % HTG specific loss calculations
part_traj_on_off = 0; % No particle trajectory calculations

% Change operating conditions of the engine data structure to match the
% current data set
ENGINE_DATA.Tgk = (((mean(T_2)+mean(T_3))/2)+mean(T_4))/2 + 273.15; %[K]
ENGINE_DATA.Tgh = (mean(T_0)+mean(T_1))/2 + 273.15; %[K]

ENGINE_DATA.Twk = Bath_Setpoint + 273.15; %[K]
ENGINE_DATA.Twh = Cap_Setpoint + 273.15; %[K]

ENGINE_DATA.Tgc = (mean(T_6)+mean(T_4))/2 + 273.15; %[K]
ENGINE_DATA.Tge = mean(T_0) + 273.15; %[K]

ENGINE_DATA.pmean = pmean; %[Pa]

[ADIABATIC_DATA, ~, ~] = ...
    sea(ENGINE_DATA,crank_inc,Model_Code, Losses_Code,part_traj_on_off);

p_engine_adiabatic = [ADIABATIC_DATA.p]; %(Pa)
```

```
    p_engine_adiabatic = p_engine_adiabatic(1:360);

    W_ind_adiabatic = ADIABATIC_DATA(end).W; %(J)
    Power_adiabatic = ADIABATIC_DATA(end).W*Engine_Hz; %(W)

    % Calculate the ideal adiabatic Model Error in Predicting Indicated
    % Work
    Error_adiabatic = ((W_ind_adiabatic-W_ind_PCB)/W_ind_PCB)*100;

    %% Calculate the Second Order Model Results
    ENGINE_DATA = HTG_ENGINE_DATA;

    % Change operating conditions of the engine data structure to match the
    % current data set
    ENGINE_DATA.Tgk = (((mean(T_2)+mean(T_3))/2)+mean(T_4))/2 + 273.15; %[K]
    ENGINE_DATA.Tgh = (mean(T_0)+mean(T_1))/2 + 273.15; %[K]

%     ENGINE_DATA.Twk = Bath_Setpoint + 273.15; %[K]
%     ENGINE_DATA.Twh = Cap_Setpoint + 273.15; %[K]

    ENGINE_DATA.Twk = ((mean(T_2)+mean(T_3))/2) + 273.15; %[K]
    ENGINE_DATA.Twh = mean(T_1) + 273.15; %[K]

    ENGINE_DATA.Tgc = mean(T_4) + 273.15; %[K]
    ENGINE_DATA.Tge = mean(T_0) + 273.15; %[K]

    ENGINE_DATA.Pbore = 0.044; %[m]
    ENGINE_DATA.Vclp = 1.733e-04 - 0.000132; %[m^3]

    ENGINE_DATA.pmean = pmean; %[Pa]
    ENGINE_DATA.V_buffer_max = 0.0032 + 0.004633333; %[m^3]
    ENGINE_DATA.GSH_config = 4; % All mods

    ENGINE_DATA.freq = Engine_Hz; %[Hz] for power data
%     ENGINE_DATA.freq = Filename_Hz; %[Hz] for efficiency data

    [SECOND_ORDER_DATA,~,LOSSES_DATA] = HTG_2nd_Order(ENGINE_DATA);

    Conduction_Loss_Model = LOSSES_DATA.Q_cond; %[W]

    pdrop_regen_model = LOSSES_DATA.pdropr; %[Pa]
    pdrop_regen_model = pdrop_regen_model(1:360)';

    %% Calculate Experimental Forced Work according to Validynes
    P_engine_exp = P2_avg_Pa; %[Pa]
    P_buffer_exp = P3_avg_Pa; %[Pa]

    % Call forced work subfunction
    [FW_W_ind, Exp_FW, FW_W_shaft] =
FW_Subfunction_v3(theta_rounded,P_engine_exp,P_buffer_exp,Vtotal_rounded,0.7,
0);

    % Temperature Drop Between Heating Cap and Expansion Space Gas
    Tdrop_Cap_to_Expansion_Space = Cap_Setpoint - Tge; %[degC]
```

```
    %% Store the data in a structure
    DATA_STRUCTURE(counter).filename = log_files_info(i).name;
    DATA_STRUCTURE(counter).Filename_Hz = Filename_Hz;
    DATA_STRUCTURE(counter).Filename_P_shaft = Filename_P_shaft;
    DATA_STRUCTURE(counter).time_TC = time_TC;
    DATA_STRUCTURE(counter).time_RTD = time_RTD;
    DATA_STRUCTURE(counter).time_Volt = time_Volt;
    DATA_STRUCTURE(counter).theta_deg = theta_deg;
    DATA_STRUCTURE(counter).T_0 = T_0;
    DATA_STRUCTURE(counter).T_1 = T_1;
    DATA_STRUCTURE(counter).T_2 = T_2;
    DATA_STRUCTURE(counter).T_3 = T_3;
    DATA_STRUCTURE(counter).T_4 = T_4;
    DATA_STRUCTURE(counter).T_5 = T_5;
    DATA_STRUCTURE(counter).T_6 = T_6;
    DATA_STRUCTURE(counter).RTD_0 = RTD_0;
    DATA_STRUCTURE(counter).RTD_1 = RTD_1;
    DATA_STRUCTURE(counter).RTD_2 = RTD_2;
    DATA_STRUCTURE(counter).RTD_3 = RTD_3;
    DATA_STRUCTURE(counter).P1_Pa = P1_Pa;
    DATA_STRUCTURE(counter).P2_Pa = P2_Pa;
    DATA_STRUCTURE(counter).P3_Pa = P3_Pa;
    DATA_STRUCTURE(counter).P4_Pa = P4_Pa;
    DATA_STRUCTURE(counter).P5_Pa = P5_Pa;
    DATA_STRUCTURE(counter).A = A;
    DATA_STRUCTURE(counter).Z = Z;
    DATA_STRUCTURE(counter).TOR = TOR.*2;
    DATA_STRUCTURE(counter).pmean = pmean;
    DATA_STRUCTURE(counter).Cap_Setpoint = Cap_Setpoint;
    DATA_STRUCTURE(counter).Bath_Setpoint = Bath_Setpoint;
    DATA_STRUCTURE(counter).Tge = Tge;
    DATA_STRUCTURE(counter).Tgc = Tgc;
    DATA_STRUCTURE(counter).Tgk = (((mean(T_2)+mean(T_3))/2)+mean(T_4))/2;
    DATA_STRUCTURE(counter).Tgh = (mean(T_0)+mean(T_1))/2;
    DATA_STRUCTURE(counter).T_cooler_inlet = T_cooler_inlet;
    DATA_STRUCTURE(counter).T_cooler_outlet = T_cooler_outlet;
    DATA_STRUCTURE(counter).Torque_avg = Torque_avg;
    DATA_STRUCTURE(counter).Engine_RPM = Engine_RPM;
    DATA_STRUCTURE(counter).Engine_Hz = Engine_Hz;
    DATA_STRUCTURE(counter).P_shaft = P_shaft;
    DATA_STRUCTURE(counter).P_ind_Validyne = P_ind_Validyne;
    DATA_STRUCTURE(counter).P_ind_PCB = P_ind_PCB;
    DATA_STRUCTURE(counter).N_Beale = N_Beale;
    DATA_STRUCTURE(counter).N_West = N_West;
    DATA_STRUCTURE(counter).N_T = N_T;
    DATA_STRUCTURE(counter).Qdot_Cooler = Qdot_Cooler;
    DATA_STRUCTURE(counter).Qdot_ConPipePowCyl = Qdot_ConPipePowCyl;
    DATA_STRUCTURE(counter).Error_isothermal = Error_isothermal;
    DATA_STRUCTURE(counter).Loss_Factor_isothermal = Loss_Factor_isothermal;
    DATA_STRUCTURE(counter).Error_adiabatic = Error_adiabatic;
    DATA_STRUCTURE(counter).Power_adiabatic = Power_adiabatic;
    DATA_STRUCTURE(counter).P_GSH_Loss_Validyne = P_GSH_Loss_Validyne;
    DATA_STRUCTURE(counter).P_GSH_Loss_PCB = P_GSH_Loss_PCB;
    DATA_STRUCTURE(counter).Exp_FW = Exp_FW;
    DATA_STRUCTURE(counter).Tdrop_Cap_to_Expansion_Space =
Tdrop_Cap_to_Expansion_Space;
```

```
    DATA_STRUCTURE(counter).P_2nd_order = SECOND_ORDER_DATA.W_dot;
    DATA_STRUCTURE(counter).Q_dot_in_2nd_order = SECOND_ORDER_DATA.Qh_dot;
    DATA_STRUCTURE(counter).Q_dot_rej_2nd_order = SECOND_ORDER_DATA.Qk_dot;
    DATA_STRUCTURE(counter).eff_thermal_2nd_order =
SECOND_ORDER_DATA.eff_thermal;
    DATA_STRUCTURE(counter).Vtotal_rounded = Vtotal_rounded;
    DATA_STRUCTURE(counter).Vtotal = Vtotal;
    DATA_STRUCTURE(counter).p_engine_experimental = P4_avg_Pa;
    DATA_STRUCTURE(counter).p_engine_isothermal = p_engine_isothermal';
    DATA_STRUCTURE(counter).p_engine_adiabatic = p_engine_adiabatic';
    DATA_STRUCTURE(counter).p_buffer_experimental = P5_avg_Pa;
    DATA_STRUCTURE(counter).p_buffer_isothermal = p_buffer_isothermal;
    DATA_STRUCTURE(counter).p_buffer_adiabatic = p_buffer_adiabatic;
    DATA_STRUCTURE(counter).pdrop_regen_experimental = P1_avg_Pa;
    DATA_STRUCTURE(counter).pdrop_regen_model = pdrop_regen_model;
    DATA_STRUCTURE(counter).T_crankcase = mean(T_5);
    DATA_STRUCTURE(counter).Q_dot_in = Q_dot_in;
    DATA_STRUCTURE(counter).eff_thermal = eff_thermal;
    DATA_STRUCTURE(counter).Conduction_Loss_Model = Conduction_Loss_Model;
    DATA_STRUCTURE(counter).P_lost_regen_FF = P_lost_regen_FF;
    DATA_STRUCTURE(counter).P_lost_regen_FF_model = LOSSES_DATA.P_flow_r;
%[W]
    counter = counter + 1;

    % Update Wait Bar
    waitbar(counter / counter_max)

end

close(WaitBar)
```

importfile_P.m

```
function [A1,Z1,TOR1,P1,P2,P3,P4,P5] = importfile_P(filename, startRow,
endRow)
%IMPORTFILE Import numeric data from a text file as column vectors.
%   [A1,Z1,TOR1,P6,P7,P8,P9,P10] = IMPORTFILE(FILENAME) Reads data from
%   text file FILENAME for the default selection.
%
%   [A1,Z1,TOR1,P6,P7,P8,P9,P10] = IMPORTFILE(FILENAME, STARTROW, ENDROW)
%   Reads data from rows STARTROW through ENDROW of text file FILENAME.
%
% Example:
%   [A1,Z1,TOR1,P6,P7,P8,P9,P10] =
importfile('TH_250_TC_21_P_60_44mm_0_Volt.log',6, 300005);
%
%    See also TEXTSCAN.

% Auto-generated by MATLAB on 2017/10/24 14:52:10

%% Initialize variables.
delimiter = '\t';
if nargin<=2
    startRow = 6;
```

```
    endRow = inf;
end

%% Format for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
%   column4: double (%f)
%   column5: double (%f)
%   column6: double (%f)
%   column7: double (%f)
%   column8: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%f%f%f%f%f%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to the format.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow(1)-startRow(1)+1,
'Delimiter', delimiter, 'TextType', 'string', 'EmptyValue', NaN,
'HeaderLines', startRow(1)-1, 'ReturnOnError', false, 'EndOfLine', '\r\n');
for block=2:length(startRow)
    frewind(fileID);
    dataArrayBlock = textscan(fileID, formatSpec, endRow(block)-
startRow(block)+1, 'Delimiter', delimiter, 'TextType', 'string',
'EmptyValue', NaN, 'HeaderLines', startRow(block)-1, 'ReturnOnError', false,
'EndOfLine', '\r\n');
    for col=1:length(dataArray)
        dataArray{col} = [dataArray{col};dataArrayBlock{col}];
    end
end

%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
% script.

%% Allocate imported array to column variable names
A1 = dataArray{:, 1};
Z1 = dataArray{:, 2};
TOR1 = dataArray{:, 3};
P1 = dataArray{:, 4};
P2 = dataArray{:, 5};
P3 = dataArray{:, 6};
P4 = dataArray{:, 7};
P5 = dataArray{:, 8};
```

importfile_RTD.m

```
function [RTD_0,RTD_1,RTD_2,RTD_3] = importfile_RTD(filename, startRow,
endRow)
%IMPORTFILE Import numeric data from a text file as column vectors.
%   [RTD_0,RTD_1,RTD_2,RTD_3] = IMPORTFILE(FILENAME) Reads data from text
%   file FILENAME for the default selection.
%
%   [RTD_0,RTD_1,RTD_2,RTD_3] = IMPORTFILE(FILENAME, STARTROW, ENDROW)
%   Reads data from rows STARTROW through ENDROW of text file FILENAME.
%
% Example:
%   [RTD_0,RTD_1,RTD_2,RTD_3] =
importfile('TH_250_TC_21_P_60_44mm_0_RTD.log',6, 1005);
%
%    See also TEXTSCAN.

% Auto-generated by MATLAB on 2017/10/24 14:46:26

%% Initialize variables.
delimiter = '\t';
if nargin<=2
    startRow = 6;
    endRow = inf;
end

%% Format for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
%   column4: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%f%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to the format.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow(1)-startRow(1)+1,
'Delimiter', delimiter, 'TextType', 'string', 'HeaderLines', startRow(1)-1,
'ReturnOnError', false, 'EndOfLine', '\r\n');
for block=2:length(startRow)
    frewind(fileID);
    dataArrayBlock = textscan(fileID, formatSpec, endRow(block)-
startRow(block)+1, 'Delimiter', delimiter, 'TextType', 'string',
'HeaderLines', startRow(block)-1, 'ReturnOnError', false, 'EndOfLine',
'\r\n');
    for col=1:length(dataArray)
        dataArray{col} = [dataArray{col};dataArrayBlock{col}];
    end
end
```

```
%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
% script.

%% Allocate imported array to column variable names
RTD_0 = dataArray{:, 1};
RTD_1 = dataArray{:, 2};
RTD_2 = dataArray{:, 3};
RTD_3 = dataArray{:, 4};
```

importfile_TC.m

```
function [T_0,T_1,T_2,T_3,T_4,T_5,T_6,T_7,T_8,T_9,T_10,T_11,T_12,T_13] =
importfile_TC(filename, startRow, endRow)
%IMPORTFILE Import numeric data from a text file as column vectors.
%   [T_0,T_1,T_2,T_3,T_4,T_5,T_6,T_7,T_8,T_9,T_10,T_11,T_12,T_13] =
%   IMPORTFILE(FILENAME) Reads data from text file FILENAME for the default
%   selection.
%
%   [T_0,T_1,T_2,T_3,T_4,T_5,T_6,T_7,T_8,T_9,T_10,T_11,T_12,T_13] =
%   IMPORTFILE(FILENAME, STARTROW, ENDROW) Reads data from rows STARTROW
%   through ENDROW of text file FILENAME.
%
% Example:
%   [T_0,T_1,T_2,T_3,T_4,T_5,T_6,T_7,T_8,T_9,T_10,T_11,T_12,T_13] =
importfile('TH_250_TC_21_P_60_44mm_0_Temp.log',6, 50005);
%
%    See also TEXTSCAN.

% Auto-generated by MATLAB on 2017/10/24 14:51:24

%% Initialize variables.
delimiter = '\t';
if nargin<=2
    startRow = 6;
    endRow = inf;
end

%% Format for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
%   column4: double (%f)
%   column5: double (%f)
%   column6: double (%f)
%   column7: double (%f)
%   column8: double (%f)
%   column9: double (%f)
%   column10: double (%f)
```

```
%   column11: double (%f)
%   column12: double (%f)
%   column13: double (%f)
%   column14: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%f%f%f%f%f%f%f%f%f%f%f%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to the format.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow(1)-startRow(1)+1,
'Delimiter', delimiter, 'TextType', 'string', 'EmptyValue', NaN,
'HeaderLines', startRow(1)-1, 'ReturnOnError', false, 'EndOfLine', '\r\n');
for block=2:length(startRow)
    frewind(fileID);
    dataArrayBlock = textscan(fileID, formatSpec, endRow(block)-
startRow(block)+1, 'Delimiter', delimiter, 'TextType', 'string',
'EmptyValue', NaN, 'HeaderLines', startRow(block)-1, 'ReturnOnError', false,
'EndOfLine', '\r\n');
    for col=1:length(dataArray)
        dataArray{col} = [dataArray{col};dataArrayBlock{col}];
    end
end

%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
% script.

%% Allocate imported array to column variable names
T_0 = dataArray{:, 1};
T_1 = dataArray{:, 2};
T_2 = dataArray{:, 3};
T_3 = dataArray{:, 4};
T_4 = dataArray{:, 5};
T_5 = dataArray{:, 6};
T_6 = dataArray{:, 7};
T_7 = dataArray{:, 8};
T_8 = dataArray{:, 9};
T_9 = dataArray{:, 10};
T_10 = dataArray{:, 11};
T_11 = dataArray{:, 12};
T_12 = dataArray{:, 13};
T_13 = dataArray{:, 14};
```

Encoder_2_Angle.m

```
function [theta_deg] = Encoder_2_Angle(A,Z)

% Written by Connor Speer, October 2017. Subfunction to convert rotary
% encoder voltage data into crank angles.

%% Inputs:
% A --> column vector of A-output voltages from the rotary encoder
% Z --> column vector of Z-output voltages from the rotary encoder
% *** A and Z must be the same length

%% Outputs
% theta_deg --> column vector of angles corresponding to rotary encoder
% outputs in (deg)
% freq --> frequency of rotary encoder in (Hz)
% RPM --> speed of rotary encoder in (RPM)

%% Find reference pulse in Z output
threshold = 3;
ref_pulse = find(Z > threshold, 1);

%% Work backwards from reference pulse to find pulse counts
if A(ref_pulse) > 2.5
    pulse_flag = 1;
else
    pulse_flag = 0;
end

A_count = zeros(size(A));

bwd_counter = 500;
row = ref_pulse - 1;

while row ~= 0
    if pulse_flag == 0 && A(row) < 2.5
        bwd_counter = bwd_counter - 1;
    end

    if bwd_counter == -1
    bwd_counter = 500;
    end

    A_count(row) = bwd_counter;

    if A(row) < 2.5
        pulse_flag = 1;
    else
        pulse_flag = 0;
    end
    row = row - 1;
end

%% Work forwards from reference pulse to find pulse counts
if A(ref_pulse) > 2.5
    pulse_flag = 1;
```

```
else
    pulse_flag = 0;
end

fwd_counter = 0;

for row = ref_pulse:length(A)
    if pulse_flag == 0 && A(row) > 2.5
        fwd_counter = fwd_counter + 1;
    end

    if fwd_counter == 501
    fwd_counter = 0;
    end

    A_count(row) = fwd_counter;

    if A(row) > 2.5
        pulse_flag = 1;
    else
        pulse_flag = 0;
    end
end

%% Use pulse count to calculate crank angles
theta_deg = A_count*(360/500); % [deg]

%% Calculate Crankshaft Speed
refs = find(A < 0.5);

% Initialize variables
pulse_count = 0;

for inc = 1:(length(refs)-1)
    if refs(inc+1) ~= (refs(inc) + 1)
        pulse_count = pulse_count + 1;
    end
end
```

PV_data_avg.m

```
function [avg_pressures] = PV_data_avg(angles,pressures)
% Written by Connor Speer, March 2017

% Inputs:
% angles --> angles corresponding to pressures in [radians].

% Round angles to the nearest whole number and convert to degrees
rounded_angles = round(angles*180/pi);

% Average all pressures which share the same angle

% Initialize vectors
```

```
avg_pressures = zeros(360,1);

for current_angle = 0:1:359
indices = rounded_angles == current_angle;
avg_press = mean(pressures(indices));
avg_pressures(current_angle+1) = avg_press;
end
```

Calibration Functions

Calibration_Function_Call.m

```
% Calibration_Function_Call.m

% Written by Connor Speer, November 2017

clear,clc,close all;

%% Plot Set-Up
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Thermocouples
TC_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Calibration Codes\Thermocouple Calibration
Oct. 22';
[TC_FIT_COEFFICIENTS]...
    = TC_Correction_Terms(TC_Cal_Folder, 0);

% Collect all the log file names from the test data folder
log_files_info = dir(fullfile(TC_Cal_Folder, '*.log'));

counter = 1;
for i = 1:length(log_files_info)
    filename_TC = strcat(TC_Cal_Folder,'\',log_files_info(i).name);
    [TC_0,TC_1,TC_2,TC_3,TC_4,TC_5,TC_6,T_7,T_8,T_9,T_10,T_11,T_12,T_13] =
...
    importfile_TC(filename_TC);
%     T_0 --> Expansion Space
```

```
%     T_1 --> HH-Regen
%     T_2 --> HH Flange - Bypass Side
%     T_3 --> HH Flange - Power Piston Side
%     T_4 --> Displacer Mount
%     T_5 --> Crankcase
%     T_6 --> Power Piston

TC_true(:,counter) = mean([mean(TC_0) mean(TC_1) mean(TC_2) mean(TC_3)
mean(TC_4) mean(TC_5) mean(TC_6)]); %(deg C)

% TC_0_raw(:,counter) = TC_0; %(deg C)
% TC_1_raw(:,counter) = TC_1; %(deg C)
TC_2_raw(:,counter) = TC_2; %(deg C)
TC_3_raw(:,counter) = TC_3; %(deg C)
TC_4_raw(:,counter) = TC_4; %(deg C)
TC_5_raw(:,counter) = TC_5; %(deg C)
TC_6_raw(:,counter) = TC_6; %(deg C)

% TC_0_calibrated(:,counter) =
calibrate_TC(0,TC_0_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)
% TC_1_calibrated(:,counter) =
calibrate_TC(1,TC_1_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)
TC_2_calibrated(:,counter) =
calibrate_TC(2,TC_2_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)
TC_3_calibrated(:,counter) =
calibrate_TC(3,TC_3_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)
TC_4_calibrated(:,counter) =
calibrate_TC(4,TC_4_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)
TC_5_calibrated(:,counter) =
calibrate_TC(5,TC_5_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)
TC_6_calibrated(:,counter) =
calibrate_TC(6,TC_6_raw(:,counter),TC_FIT_COEFFICIENTS); %(deg C)

counter = counter + 1;
end

% Plot 5a - Measured Temperature vs. "True" Temperature before adjustment
% figure('Position', [x y 326 275])
% hold on
% % plot(mean(TC_0_raw),TC_true,'*k')
% % plot(mean(TC_1_raw),TC_true,'*g')
% plot(mean(TC_2_raw),TC_true,'*b')
% plot(mean(TC_3_raw),TC_true,'*m')
% plot(mean(TC_4_raw),TC_true,'*c')
% plot(mean(TC_5_raw),TC_true,'*r')
% plot(mean(TC_6_raw),TC_true,'*y')
% plot(TC_true,TC_true,'-r')
% hold off
% xlabel('"True" Temperature (\circC)')
% ylabel('Measured Temperature (\circC)')
% legend('TC_2','TC_3','TC_4','TC_5','TC_6','"True"
Temperature','Location','NorthWest')

% Plot 5b - Measured Temperature Error before adjustment
figure('Position', [x y 326 275])
hold on
```

```
% plot(TC_true,mean(TC_0_raw)-TC_true,'*k')
% plot(TC_true,mean(TC_1_raw)-TC_true,'*g')
plot(TC_true,mean(TC_2_raw)-TC_true,'*b')
plot(TC_true,mean(TC_3_raw)-TC_true,'*m')
plot(TC_true,mean(TC_4_raw)-TC_true,'*c')
plot(TC_true,mean(TC_5_raw)-TC_true,'*r')
plot(TC_true,mean(TC_6_raw)-TC_true,'*y')
hold off
xlabel('"True" Temperature (\circC)')
ylabel('Measured Temperature Error (\circC)')

% Plot 6a - Measured Temperature vs. "True" Temperature before adjustment
% figure('Position', [x y 326 275])
% hold on
% % plot(mean(TC_0_raw),TC_true,'*k')
% % plot(mean(TC_1_raw),TC_true,'*g')
% plot(mean(TC_2_raw),TC_true,'*b')
% plot(mean(TC_3_raw),TC_true,'*m')
% plot(mean(TC_4_raw),TC_true,'*c')
% plot(mean(TC_5_raw),TC_true,'*r')
% plot(mean(TC_6_raw),TC_true,'*y')
% plot(TC_true,TC_true,'-r')
% hold off
% xlabel('"True" Temperature (\circC)')
% ylabel('Measured Temperature (\circC)')
% legend('TC_2','TC_3','TC_4','TC_5','TC_6','"True"
Temperature','Location','NorthWest')

% Plot 6b - Measured Temperature Error after adjustment
figure('Position', [x y 326 275])
hold on
% plot(TC_true,mean(TC_0_calibrated)-TC_true,'*k')
% plot(TC_true,mean(TC_1_calibrated)-TC_true,'*g')
plot(TC_true,mean(TC_2_calibrated)-TC_true,'*b')
plot(TC_true,mean(TC_3_calibrated)-TC_true,'*m')
plot(TC_true,mean(TC_4_calibrated)-TC_true,'*c')
plot(TC_true,mean(TC_5_calibrated)-TC_true,'*r')
plot(TC_true,mean(TC_6_calibrated)-TC_true,'*y')
hold off
xlabel('"True" Temperature (\circC)')
ylabel('Measured Temperature Error (\circC)')



%% Validyne Absolute Pressure
Abs_Val_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Calibration Codes\Validyne
Calibration Oct. 22';
[ABS_VAL_FIT_COEFFICIENTS]...
    = ABS_VAL_Correction_Terms(Abs_Val_Cal_Folder, 0);

% Collect all the log file names from the test data folder
log_files_info = dir(fullfile(Abs_Val_Cal_Folder, '*.log'));

counter = 1;
```

```
for i = 1:length(log_files_info)
filename_Volt = strcat(Abs_Val_Cal_Folder,'\',log_files_info(i).name);
[A,Z,TOR,P1,P2,P3,P4,P5] = importfile_P(filename_Volt);
%     P1 --> Validyne Regenerator Differential Pressure
%     P2 --> Validyne Power Cylinder
%     P3 --> Validyne Crankcase

P_cal_machine(:,counter) = str2double(extractBetween(log_files_info(i).name,
'E_','PSI')); %(PSI)
P_cal_machine(:,counter) = P_cal_machine(:,counter)*6894.76; %(Pa)

P2_Pa_raw(:,counter) = P2*20*6894.76; %(Pa)
P3_Pa_raw(:,counter) = P3*20*6894.76; %(Pa)

P2_Pa_calibrated(:,counter) =
calibrate_abs_VAL(2,P2_Pa_raw(:,counter),ABS_VAL_FIT_COEFFICIENTS); %(Pa)
P3_Pa_calibrated(:,counter) =
calibrate_abs_VAL(3,P3_Pa_raw(:,counter),ABS_VAL_FIT_COEFFICIENTS); %(Pa)
counter = counter + 1;
end

% % Plot 5a - Measured Pressure vs. "True" Pressure before adjustment
% figure('Position', [x y 326 275])
% hold on
% plot(mean(P2_Pa_raw)./1000,P_cal_machine./1000,'*r')
% plot(mean(P3_Pa_raw)./1000,P_cal_machine./1000,'ok')
% plot(P_cal_machine./1000,P_cal_machine./1000,'-r')
% hold off
% xlabel('Calibration Machine Pressure (kPa)');
% ylabel('Measured Pressure (kPa)')
% legend('Power Cylinder','Crankcase','Cal Machine','Location','NorthWest')

% Plot 5b - Measured Pressure Error before adjustment
figure('Position', [x y 326 275])
hold on
plot(P_cal_machine./1000,mean(P2_Pa_raw)./1000-P_cal_machine./1000,'*r')
plot(P_cal_machine./1000,mean(P3_Pa_raw)./1000-P_cal_machine./1000,'ok')
hold off
xlabel('Calibration Machine Pressure (kPa)');
ylabel('Measured Pressure Error (kPa)')
legend('Power Cylinder','Crankcase','Location','NorthWest')

% % Plot 6a - Measured Pressure vs. "True" Pressure before adjustment
% figure('Position', [x y 326 275])
% hold on
% plot(mean(P2_Pa_calibrated)./1000,P_cal_machine./1000,'*r')
% plot(mean(P3_Pa_calibrated)./1000,P_cal_machine./1000,'ok')
% plot(P_cal_machine./1000,P_cal_machine./1000,'-r')
% hold off
% xlabel('Calibration Machine Pressure (kPa)');
% ylabel('Measured Pressure (kPa)')
% legend('Power Cylinder','Crankcase','Cal Machine','Location','NorthWest')

% Plot 6b - Measured Pressure Error before adjustment
figure('Position', [x y 326 275])
```

```
hold on
plot(P_cal_machine./1000,mean(P2_Pa_calibrated)./1000-
P_cal_machine./1000,'*r')
plot(P_cal_machine./1000,mean(P3_Pa_calibrated)./1000-
P_cal_machine./1000,'ok')
hold off
xlabel('Calibration Machine Pressure (kPa)');
ylabel('Measured Pressure Error (kPa)')
legend('Power Cylinder','Crankcase','Location','NorthWest')

%% Validyne Differential Pressure
Diff_Val_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Calibration Codes\Differential
Validyne Calibration Oct. 22';
[DIFF_VAL_FIT_COEFFICIENTS]...
    = DIFF_VAL_Correction_Terms(Diff_Val_Cal_Folder, 0);

% Collect all the log file names from the test data folder
log_files_info = dir(fullfile(Diff_Val_Cal_Folder, '*.log'));

counter = 1;
for i = 1:length(log_files_info)
filename_Volt = strcat(Diff_Val_Cal_Folder,'\',log_files_info(i).name);
[A,Z,TOR,P1,P2,P3,P4,P5] = importfile_P(filename_Volt);
%     P1 --> Validyne Regenerator Differential Pressure
%     P2 --> Validyne Power Cylinder
%     P3 --> Validyne Crankcase

P_cal_machine(:,counter) = str2double(extractBetween(log_files_info(i).name,
'E_','PSI')); %(PSI)
P_cal_machine(:,counter) = P_cal_machine(:,counter)*6894.76; %(Pa)

P1_Pa_raw(:,counter) = P1*0.5*6894.76; %(Pa)

P1_Pa_calibrated(:,counter) =
calibrate_diff_VAL(1,P1_Pa_raw(:,counter),DIFF_VAL_FIT_COEFFICIENTS); %(Pa)
counter = counter + 1;
end

% Plot 5a - Measured Pressure vs. "True" Pressure before adjustment
% figure('Position', [x y 326 275])
% hold on
% plot(mean(P1_Pa_raw)./1000,P_cal_machine./1000,'*k')
% plot(P_cal_machine./1000,P_cal_machine./1000,'-r')
% hold off
% xlabel('Calibration Machine Pressure (kPa)')
% ylabel('Measured Pressure (kPa)')
% legend('Regenerator Differential','Cal Machine','Location','NorthWest')

% Plot 5b - Measured Pressure Error before adjustment
figure('Position', [x y 326 275])
plot(P_cal_machine./1000,mean(P1_Pa_raw)./1000-P_cal_machine./1000,'*k')
xlabel('Calibration Machine Pressure (kPa)')
ylabel('Measured Pressure Error (kPa)')
```

```
% Plot 6a - Measured Pressure vs. "True" Pressure before adjustment
% figure('Position', [x y 326 275])
% hold on
% plot(mean(P1_Pa_calibrated)./1000,P_cal_machine./1000,'*k')
% plot(P_cal_machine./1000,P_cal_machine./1000,'-r')
% hold off
% xlabel('Calibration Machine Pressure (kPa)')
% ylabel('Measured Pressure (kPa)')
% legend('Regenerator Differential','Cal Machine','Location','NorthWest')

% Plot 6b - Measured Pressure Error before adjustment
figure('Position', [x y 326 275])
plot(P_cal_machine./1000,mean(P1_Pa_calibrated)./1000-
P_cal_machine./1000,'*k')
xlabel('Calibration Machine Pressure (kPa)')
ylabel('Measured Pressure Error (kPa)')

%% RTDs
RTD_Cal_Folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Calibration Codes\RTD Calibration Oct.
22';
[RTD_FIT_COEFFICIENTS]...
    = RTD_Correction_Terms(RTD_Cal_Folder, 0);

% Collect all the log file names from the test data folder
log_files_info = dir(fullfile(RTD_Cal_Folder, '*.log'));

counter = 1;
for i = 1:length(log_files_info)
    filename_RTD = strcat(RTD_Cal_Folder,'\',log_files_info(i).name);
    [RTD_0,RTD_1,RTD_2,RTD_3] = importfile_RTD(filename_RTD);
%     RTD_0 --> Cooler water inlet
%     RTD_1 --> Cooler water outlet
%     RTD_2 --> Con Pipe/Pow Cyl inlet
%     RTD_3 --> Con Pipe/Pow Cyl outlet

RTD_true(:,counter) = mean([mean(RTD_0) mean(RTD_1) mean(RTD_2)
mean(RTD_3)]); %(deg C)

RTD_0_raw(:,counter) = RTD_0; %(deg C)
RTD_1_raw(:,counter) = RTD_1; %(deg C)
RTD_2_raw(:,counter) = RTD_2; %(deg C)
RTD_3_raw(:,counter) = RTD_3; %(deg C)

RTD_0_calibrated(:,counter) =
calibrate_RTD(0,RTD_0_raw(:,counter),RTD_FIT_COEFFICIENTS); %(deg C)
RTD_1_calibrated(:,counter) =
calibrate_RTD(1,RTD_1_raw(:,counter),RTD_FIT_COEFFICIENTS); %(deg C)
RTD_2_calibrated(:,counter) =
calibrate_RTD(2,RTD_2_raw(:,counter),RTD_FIT_COEFFICIENTS); %(deg C)
RTD_3_calibrated(:,counter) =
calibrate_RTD(3,RTD_3_raw(:,counter),RTD_FIT_COEFFICIENTS); %(deg C)

counter = counter + 1;
end
```

```
% Plot 5a - Measured Temperature vs. "True" Temperature before adjustment
% figure('Position', [x y 326 275])
% hold on
% plot(mean(RTD_0_raw),RTD_true,'*k')
% plot(mean(RTD_1_raw),RTD_true,'*g')
% plot(mean(RTD_2_raw),RTD_true,'*b')
% plot(mean(RTD_3_raw),RTD_true,'*m')
% plot(RTD_true,RTD_true,'-r')
% hold off
% xlabel('"True" Temperature (\circC)')
% ylabel('Measured Temperature (\circC)')
% legend('RTD_0','RTD_1','RTD_2','RTD_3','"True"
Temperature','Location','NorthWest')

% Plot 5b - Measured Temperature Error before adjustment
figure('Position', [x y 326 275])
hold on
plot(RTD_true,mean(RTD_0_raw)-RTD_true,'*k')
plot(RTD_true,mean(RTD_1_raw)-RTD_true,'*g')
plot(RTD_true,mean(RTD_2_raw)-RTD_true,'*b')
plot(RTD_true,mean(RTD_3_raw)-RTD_true,'*m')
hold off
xlabel('"True" Temperature (\circC)')
ylabel('Measured Temperature Error (\circC)')

% Plot 6a - Measured Temperature vs. "True" Temperature before adjustment
% figure('Position', [x y 326 275])
% hold on
% plot(mean(RTD_0_raw),RTD_true,'*k')
% plot(mean(RTD_1_raw),RTD_true,'*g')
% plot(mean(RTD_2_raw),RTD_true,'*b')
% plot(mean(RTD_3_raw),RTD_true,'*m')
% plot(RTD_true,RTD_true,'-r')
% hold off
% xlabel('"True" Temperature (\circC)')
% ylabel('Measured Temperature (\circC)')
% legend('RTD_0','RTD_1','RTD_2','RTD_3','"True"
Temperature','Location','NorthWest')

% Plot 6b - Measured Temperature Error after adjustment
figure('Position', [x y 326 275])
hold on
plot(RTD_true,mean(RTD_0_calibrated)-RTD_true,'*k')
plot(RTD_true,mean(RTD_1_calibrated)-RTD_true,'*g')
plot(RTD_true,mean(RTD_2_calibrated)-RTD_true,'*b')
plot(RTD_true,mean(RTD_3_calibrated)-RTD_true,'*m')
hold off
xlabel('"True" Temperature (\circC)')
ylabel('Measured Temperature Error (\circC)')
```

ABS_VAL_Correction_Terms.m

```
function [ABS_VAL_FIT_COEFFICIENTS]...
```

```
    = ABS_VAL_Correction_Terms(P_Cal_Folder, Cal_Stats_On_Off)

% Written by Connor Speer - October '17
% Modified by Shahzeb Mirza Oct 2017
% Modified by Connor Speer - December '17
% The goal of this function is to collect calibration data from a specified
% folder and determine the correction terms and raw data statistics.
% Calibration correction factors are then used to adjust the raw data.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Collect all the log file names from the thermocouple test data folder
P_log_files_info = dir(fullfile(P_Cal_Folder, '*.log'));

% Preallocate space for the structure array
P_DATA(length(P_log_files_info)).P_2_raw = [];
P_DATA(length(P_log_files_info)).P_2_avg = [];
P_DATA(length(P_log_files_info)).P_2_stdev = [];
P_DATA(length(P_log_files_info)).P_2_corr = [];

P_DATA(length(P_log_files_info)).P_3_raw = [];
P_DATA(length(P_log_files_info)).P_3_avg = [];
P_DATA(length(P_log_files_info)).P_3_stdev = [];
P_DATA(length(P_log_files_info)).P_3_corr = [];

P_DATA(length(P_log_files_info)).P_4_raw = [];
P_DATA(length(P_log_files_info)).P_4_avg = [];
P_DATA(length(P_log_files_info)).P_4_stdev = [];
P_DATA(length(P_log_files_info)).P_4_corr = [];

P_DATA(length(P_log_files_info)).P_overall_avg = [];
P_DATA(length(P_log_files_info)).P_single = [];

% Initialize counter variable
counter = 1;
counter_max = 0.5*length(P_log_files_info);

WaitBar = waitbar(0,'Analyzing calibration data...');

% Open Calibration Log Files
for i = 1:1:length(P_log_files_info)
    filename_Volt = strcat(P_Cal_Folder,'\',P_log_files_info(i).name);
    [A,Z,TOR,P_1,P_2,P_3,P_4,P_5] = importfile_P(filename_Volt);
%     A --> 500 PPR Rotary Encoder Output
%     Z --> 1 PPR Rotary Encoder Output
%     TOR --> Torque Transducer Output
%     P1 --> Validyne Regenerator Differential Pressure
%     P2 --> Validyne Power Cylinder
%     P3 --> Validyne Crankcase
%     P4 --> PCB Power Cylinder
%     P5 --> PCB Crankcase

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate Data from the file
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate the average reading for each pressure transducer
```

```
    P_2_avg = mean(P_2)*20*6894.76; %(Pa)
    P_3_avg = mean(P_3)*20*6894.76; %(Pa)
    P_4_avg = mean(P_4)*20*6894.76; %(Pa)

    % Collect the "true" pressure from the name of the log file
    try
    P_cal_machine = str2double(extractBetween(P_log_files_info(i).name,
'E_','PSI')); %(PSI)
    catch

    end

    P_cal_machine = P_cal_machine*6894.76; %(Pa)

    % Calculate the correction term for each pressure transducer
    P_2_corr = P_cal_machine - P_2_avg; %(Pa)
    P_3_corr = P_cal_machine - P_3_avg; %(Pa)
    P_4_corr = P_cal_machine - P_4_avg; %(Pa)

    % Calculate the sample standard deviation for each pressure transducer
    P_2_stdev = std(P_2);
    P_3_stdev = std(P_3);
    P_4_stdev = std(P_4);

    % Find the maximum standard deviation of all the thermocouples
    stdev_max = max([P_2_stdev P_3_stdev P_4_stdev]);

    % Calculate the worst case random uncertainty for a single measurement
    P_single = 2*stdev_max; % t = 2, See example 7.3 in Wheeler and Ganji

    % Store the values from the log file in a structure
    P_DATA(counter).P_2_raw = P_2;
    P_DATA(counter).P_2_avg = P_2_avg;
    P_DATA(counter).P_2_stdev = P_2_stdev;
    P_DATA(counter).P_2_corr = P_2_corr;

    P_DATA(counter).P_3_raw = P_3;
    P_DATA(counter).P_3_avg = P_3_avg;
    P_DATA(counter).P_3_stdev = P_3_stdev;
    P_DATA(counter).P_3_corr = P_3_corr;

    P_DATA(counter).P_4_raw = P_4;
    P_DATA(counter).P_4_avg = P_4_avg;
    P_DATA(counter).P_4_stdev = P_4_stdev;
    P_DATA(counter).P_4_corr = P_4_corr;

    P_DATA(counter).P_cal_machine = P_cal_machine;
    P_DATA(counter).P_single = P_single;

    % Increment the counter variable
    counter = counter + 1;

    % Update Wait Bar
    waitbar(counter / counter_max)
```

```
end
close(WaitBar);

%% Fit Curves to Correction Terms
% Fit a curve to the correction terms. Each thermocouple gets its own
% curve. Could potentially use the R^2 value to quantify the systematic
% error.
P_2_fit = fit([P_DATA.P_2_avg]',[P_DATA.P_2_corr]','poly3');
P_3_fit = fit([P_DATA.P_3_avg]',[P_DATA.P_3_corr]','poly3');
P_4_fit = fit([P_DATA.P_4_avg]',[P_DATA.P_4_corr]','poly3');

% Store the coefficients of the fitted curve equations in a structure
ABS_VAL_FIT_COEFFICIENTS.P_2 = coeffvalues(P_2_fit);
ABS_VAL_FIT_COEFFICIENTS.P_3 = coeffvalues(P_3_fit);
ABS_VAL_FIT_COEFFICIENTS.P_4 = coeffvalues(P_4_fit);


% Display the worst case random uncertainty for a single measurement
D1 = ['Worst case random uncertainty for a single
measurement:',num2str(P_single),' V'];
disp(D1);
```

calibrate_abs_VAL.m

```
function P_calibrated =
calibrate_abs_VAL(Ch_num,P_raw,ABS_VAL_FIT_COEFFICIENTS)

% Written by Connor Speer, December 2017
% This function adds correction terms, calculated via the fit coefficients,
% to a vector of input raw data and returns a vector of calibrated data.

%% Inputs
% Ch_num --> Channel number of the transducer. Must be the same as in the
% calibration data.
% P_raw --> raw pressure data in Pa.
% ABS_VAL_FIT_COEFFICIENTS --> 3rd order polynomial fit coefficients for
% correction terms

%% Outputs
% P_calibrated --> Calibrated pressure in Pa.

%% Calculate correction terms
if Ch_num == 2
    p1 = ABS_VAL_FIT_COEFFICIENTS.P_2(1);
    p2 = ABS_VAL_FIT_COEFFICIENTS.P_2(2);
    p3 = ABS_VAL_FIT_COEFFICIENTS.P_2(3);
    p4 = ABS_VAL_FIT_COEFFICIENTS.P_2(4);
elseif Ch_num == 3
    p1 = ABS_VAL_FIT_COEFFICIENTS.P_3(1);
    p2 = ABS_VAL_FIT_COEFFICIENTS.P_3(2);
    p3 = ABS_VAL_FIT_COEFFICIENTS.P_3(3);
    p4 = ABS_VAL_FIT_COEFFICIENTS.P_3(4);
```

```
end

correction_terms = p1.*P_raw.^3 + p2.*P_raw.^2 + p3.*P_raw + p4;

%% Apply correction terms
P_calibrated = P_raw + correction_terms;
```

DIFF_VAL_Correction_Terms.m

```
function [DIFF_VAL_FIT_COEFFICIENTS]...
    = DIFF_VAL_Correction_Terms(P_Cal_Folder, Cal_Stats_On_Off)

% Written by Connor Speer - October '17
% Modified by Shahzeb Mirza Oct 2017
% Modified by Connor Speer - December '17
% The goal of this function is to collect calibration data from a specified
% folder and determine the correction terms and raw data statistics.
% Calibration correction factors are then used to adjust the raw data.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Collect all the log file names from the thermocouple test data folder
P_log_files_info = dir(fullfile(P_Cal_Folder, '*.log'));

% Preallocate space for the structure array
P_DATA(length(P_log_files_info)).P_1_raw = [];
P_DATA(length(P_log_files_info)).P_1_avg = [];
P_DATA(length(P_log_files_info)).P_1_stdev = [];
P_DATA(length(P_log_files_info)).P_1_corr = [];

P_DATA(length(P_log_files_info)).P_overall_avg = [];
P_DATA(length(P_log_files_info)).P_single = [];

% Initialize counter variable
counter = 1;
counter_max = 0.5*length(P_log_files_info);

WaitBar = waitbar(0,'Analyzing calibration data...');

% Open Calibration Log Files
for i = 1:1:length(P_log_files_info)
    filename_Volt = strcat(P_Cal_Folder,'\',P_log_files_info(i).name);
    [A,Z,TOR,P_1,P_2,P_3,P_4,P_5] = importfile_P(filename_Volt);
%     A --> 500 PPR Rotary Encoder Output
%     Z --> 1 PPR Rotary Encoder Output
%     TOR --> Torque Transducer Output
%     P1 --> Validyne Regenerator Differential Pressure
%     P2 --> Validyne Power Cylinder
%     P3 --> Validyne Crankcase
%     P4 --> PCB Power Cylinder
%     P5 --> PCB Crankcase

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate Data from the file
```

```
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate the average reading for each pressure transducer
    P_1_avg = mean(P_1)*0.5*6894.76; %(Pa)

    % Collect the "true" pressure from the name of the log file
    try
    P_cal_machine = str2double(extractBetween(P_log_files_info(i).name,
'E_','PSI')); %(PSI)
    catch

    end

    P_cal_machine = P_cal_machine*6894.76; %(Pa)

    % Calculate the correction term for each pressure transducer
    P_1_corr = P_cal_machine - P_1_avg; %(Pa)

    % Calculate the sample standard deviation for each pressure transducer
    P_1_stdev = std(P_1);

    % Find the maximum standard deviation of all the thermocouples
    stdev_max = P_1_stdev;

    % Calculate the worst case random uncertainty for a single measurement
    P_single = 2*stdev_max; % t = 2, See example 7.3 in Wheeler and Ganji

    % Store the values from the log file in a structure
    P_DATA(counter).P_1_raw = P_1;
    P_DATA(counter).P_1_avg = P_1_avg;
    P_DATA(counter).P_1_stdev = P_1_stdev;
    P_DATA(counter).P_1_corr = P_1_corr;

    P_DATA(counter).P_cal_machine = P_cal_machine;
    P_DATA(counter).P_single = P_single;

    % Increment the counter variable
    counter = counter + 1;

    % Update Wait Bar
    waitbar(counter / counter_max)

end
close(WaitBar);


%% Fit Curves to Correction Terms
% Fit a curve to the correction terms. Each thermocouple gets its own
% curve. Could potentially use the R^2 value to quantify the systematic
% error.
P_1_fit = fit([P_DATA.P_1_avg]',[P_DATA.P_1_corr]','poly3');

% Store the coefficients of the fitted curve equations in a structure
DIFF_VAL_FIT_COEFFICIENTS.P_1 = coeffvalues(P_1_fit);
```

```
% Display the worst case random uncertainty for a single measurement
D1 = ['Worst case random uncertainty for a single
measurement:',num2str(P_single),' V'];
disp(D1);
```

calibrate_diff_VAL.m

```
function P_calibrated =
calibrate_diff_VAL(Ch_num,P_raw,ABS_VAL_FIT_COEFFICIENTS)

% Written by Connor Speer, December 2017
% This function adds correction terms, calculated via the fit coefficients,
% to a vector of input raw data and returns a vector of calibrated data.

%% Inputs
% Ch_num --> Channel number of the transducer. Must be the same as in the
% calibration data.
% P_raw --> raw pressure data in Pa.
% ABS_VAL_FIT_COEFFICIENTS --> 3rd order polynomial fit coefficients for
% correction terms

%% Outputs
% P_calibrated --> Calibrated pressure in Pa.

%% Calculate correction terms
if Ch_num == 1
    p1 = ABS_VAL_FIT_COEFFICIENTS.P_1(1);
    p2 = ABS_VAL_FIT_COEFFICIENTS.P_1(2);
    p3 = ABS_VAL_FIT_COEFFICIENTS.P_1(3);
    p4 = ABS_VAL_FIT_COEFFICIENTS.P_1(4);
end

correction_terms = p1.*P_raw.^3 + p2.*P_raw.^2 + p3.*P_raw + p4;

%% Apply correction terms
P_calibrated = P_raw + correction_terms;
```

RTD_Correction_Terms.m

```
function [RTD_FIT_COEFFICIENTS]...
    = RTD_Correction_Terms(RTD_Cal_Folder, Cal_Stats_On_Off)

% Written by Connor Speer - October '17
% Modified by Shahzeb Mirza Oct 2017
% Modified by Connor Speer - December '17
% The goal of this function is to collect calibration data from a specified
% folder and determine the correction terms and raw data statistics.
% Calibration correction factors are then used to adjust the raw data.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Collect all the log file names from the thermocouple test data folder
RTD_log_files_info = dir(fullfile(RTD_Cal_Folder, '*.log'));
```

```
% Preallocate space for the structure array
RTD_DATA(length(RTD_log_files_info)).RTD_0_raw = [];
RTD_DATA(length(RTD_log_files_info)).RTD_0_avg = [];
RTD_DATA(length(RTD_log_files_info)).RTD_0_stdev = [];
RTD_DATA(length(RTD_log_files_info)).RTD_0_corr = [];

RTD_DATA(length(RTD_log_files_info)).RTD_1_raw = [];
RTD_DATA(length(RTD_log_files_info)).RTD_1_avg = [];
RTD_DATA(length(RTD_log_files_info)).RTD_1_stdev = [];
RTD_DATA(length(RTD_log_files_info)).RTD_1_corr = [];

RTD_DATA(length(RTD_log_files_info)).RTD_2_raw = [];
RTD_DATA(length(RTD_log_files_info)).RTD_2_avg = [];
RTD_DATA(length(RTD_log_files_info)).RTD_2_stdev = [];
RTD_DATA(length(RTD_log_files_info)).RTD_2_corr = [];

RTD_DATA(length(RTD_log_files_info)).RTD_3_raw = [];
RTD_DATA(length(RTD_log_files_info)).RTD_3_avg = [];
RTD_DATA(length(RTD_log_files_info)).RTD_3_stdev = [];
RTD_DATA(length(RTD_log_files_info)).RTD_3_corr = [];

RTD_DATA(length(RTD_log_files_info)).RTD_true = [];
RTD_DATA(length(RTD_log_files_info)).P_single = [];

% Initialize counter variable
counter = 1;
counter_max = 0.5*length(RTD_log_files_info);

WaitBar = waitbar(0,'Analyzing calibration data...');

% Open Calibration Log Files
for i = 1:1:length(RTD_log_files_info)
    filename_RTD = strcat(RTD_Cal_Folder,'\',RTD_log_files_info(i).name);
    [RTD_0,RTD_1,RTD_2,RTD_3] = importfile_RTD(filename_RTD);
%     RTD_0 --> Cooler water inlet
%     RTD_1 --> Cooler water outlet
%     RTD_2 --> Con Pipe/Pow Cyl inlet
%     RTD_3 --> Con Pipe/Pow Cyl outlet

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate Data from the file
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate the average reading for each RTD
    RTD_0_avg = mean(RTD_0); %(deg C)
    RTD_1_avg = mean(RTD_1); %(deg C)
    RTD_2_avg = mean(RTD_2); %(deg C)
    RTD_3_avg = mean(RTD_3); %(deg C)

    % Calculate the "true" temperature as the average of all RTDs
    RTD_true = mean([RTD_0_avg RTD_1_avg RTD_2_avg RTD_3_avg]);

    % Calculate the correction term for each RTD
    RTD_0_corr = RTD_true - RTD_0_avg; %(deg C)
    RTD_1_corr = RTD_true - RTD_1_avg; %(deg C)
```

```
    RTD_2_corr = RTD_true - RTD_2_avg; %(deg C)
    RTD_3_corr = RTD_true - RTD_3_avg; %(deg C)

    % Calculate the sample standard deviation for each RTD
    RTD_0_stdev = std(RTD_0);
    RTD_1_stdev = std(RTD_1);
    RTD_2_stdev = std(RTD_2);
    RTD_3_stdev = std(RTD_3);

    % Find the maximum standard deviation of all the thermocouples
    stdev_max = max([RTD_0_stdev RTD_1_stdev RTD_2_stdev RTD_3_stdev]);

    % Calculate the worst case random uncertainty for a single measurement
    P_single = 2*stdev_max; % t = 2, See example 7.3 in Wheeler and Ganji

    % Store the values from the log file in a structure
    RTD_DATA(counter).RTD_0_raw = RTD_0;
    RTD_DATA(counter).RTD_0_avg = RTD_0_avg;
    RTD_DATA(counter).RTD_0_stdev = RTD_0_stdev;
    RTD_DATA(counter).RTD_0_corr = RTD_0_corr;

    RTD_DATA(counter).RTD_1_raw = RTD_1;
    RTD_DATA(counter).RTD_1_avg = RTD_1_avg;
    RTD_DATA(counter).RTD_1_stdev = RTD_1_stdev;
    RTD_DATA(counter).RTD_1_corr = RTD_1_corr;

    RTD_DATA(counter).RTD_2_raw = RTD_2;
    RTD_DATA(counter).RTD_2_avg = RTD_2_avg;
    RTD_DATA(counter).RTD_2_stdev = RTD_2_stdev;
    RTD_DATA(counter).RTD_2_corr = RTD_2_corr;

    RTD_DATA(counter).RTD_3_raw = RTD_3;
    RTD_DATA(counter).RTD_3_avg = RTD_3_avg;
    RTD_DATA(counter).RTD_3_stdev = RTD_3_stdev;
    RTD_DATA(counter).RTD_3_corr = RTD_3_corr;

    RTD_DATA(counter).RTD_true = RTD_true;
    RTD_DATA(counter).P_single = P_single;

    % Increment the counter variable
    counter = counter + 1;

    % Update Wait Bar
    waitbar(counter / counter_max)

end
close(WaitBar);

%% Fit Curves to Correction Terms
% Fit a curve to the correction terms. Each thermocouple gets its own
% curve. Could potentially use the R^2 value to quantify the systematic
% error.
RTD_0_fit = fit([RTD_DATA.RTD_0_avg]',[RTD_DATA.RTD_0_corr]','poly3');
RTD_1_fit = fit([RTD_DATA.RTD_1_avg]',[RTD_DATA.RTD_1_corr]','poly3');
```

```
RTD_2_fit = fit([RTD_DATA.RTD_2_avg]',[RTD_DATA.RTD_2_corr]','poly3');
RTD_3_fit = fit([RTD_DATA.RTD_3_avg]',[RTD_DATA.RTD_3_corr]','poly3');

% Store the coefficients of the fitted curve equations in a structure
RTD_FIT_COEFFICIENTS.RTD_0 = coeffvalues(RTD_0_fit);
RTD_FIT_COEFFICIENTS.RTD_1 = coeffvalues(RTD_1_fit);
RTD_FIT_COEFFICIENTS.RTD_2 = coeffvalues(RTD_2_fit);
RTD_FIT_COEFFICIENTS.RTD_3 = coeffvalues(RTD_3_fit);

% Display the worst case random uncertainty for a single measurement
D1 = ['Worst case random uncertainty for a single
measurement:',num2str(P_single),' \circC'];
disp(D1);
```

calibrate_RTD.m

```
function RTD_calibrated = calibrate_RTD(Ch_num,RTD_raw,RTD_FIT_COEFFICIENTS)

% Written by Connor Speer, December 2017
% This function adds correction terms, calculated via the fit coefficients,
% to a vector of input raw data and returns a vector of calibrated data.

%% Inputs
% Ch_num --> Channel number of the transducer. Must be the same as in the
% calibration data.
% P_raw --> raw pressure data in Pa.
% ABS_VAL_FIT_COEFFICIENTS --> 3rd order polynomial fit coefficients for
% correction terms

%% Outputs
% P_calibrated --> Calibrated pressure in Pa.

%% Calculate correction terms
if Ch_num == 0
    p1 = RTD_FIT_COEFFICIENTS.RTD_0(1);
    p2 = RTD_FIT_COEFFICIENTS.RTD_0(2);
    p3 = RTD_FIT_COEFFICIENTS.RTD_0(3);
    p4 = RTD_FIT_COEFFICIENTS.RTD_0(4);
elseif Ch_num == 1
    p1 = RTD_FIT_COEFFICIENTS.RTD_1(1);
    p2 = RTD_FIT_COEFFICIENTS.RTD_1(2);
    p3 = RTD_FIT_COEFFICIENTS.RTD_1(3);
    p4 = RTD_FIT_COEFFICIENTS.RTD_1(4);
elseif Ch_num == 2
    p1 = RTD_FIT_COEFFICIENTS.RTD_2(1);
    p2 = RTD_FIT_COEFFICIENTS.RTD_2(2);
    p3 = RTD_FIT_COEFFICIENTS.RTD_2(3);
    p4 = RTD_FIT_COEFFICIENTS.RTD_2(4);
elseif Ch_num == 3
    p1 = RTD_FIT_COEFFICIENTS.RTD_3(1);
    p2 = RTD_FIT_COEFFICIENTS.RTD_3(2);
    p3 = RTD_FIT_COEFFICIENTS.RTD_3(3);
    p4 = RTD_FIT_COEFFICIENTS.RTD_3(4);
end
```

```
correction_terms = p1.*RTD_raw.^3 + p2.*RTD_raw.^2 + p3.*RTD_raw + p4;

%% Apply correction terms
RTD_calibrated = RTD_raw + correction_terms;
```

TC_Correction_Terms.m

```
function [TC_FIT_COEFFICIENTS]...
    = TC_Correction_Terms(TC_Cal_Folder, Cal_Stats_On_Off)

% Written by Connor Speer - October '17
% Modified by Shahzeb Mirza Oct 2017
% Modified by Connor Speer - December '17
% The goal of this function is to collect calibration data from a specified
% folder and determine the correction terms and raw data statistics.
% Calibration correction factors are then used to adjust the raw data.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Collect all the log file names from the thermocouple test data folder
TC_log_files_info = dir(fullfile(TC_Cal_Folder, '*.log'));

% Preallocate space for the structure array
TC_DATA(length(TC_log_files_info)).TC_0_raw = [];
TC_DATA(length(TC_log_files_info)).TC_0_avg = [];
TC_DATA(length(TC_log_files_info)).TC_0_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_0_corr = [];

TC_DATA(length(TC_log_files_info)).TC_1_raw = [];
TC_DATA(length(TC_log_files_info)).TC_1_avg = [];
TC_DATA(length(TC_log_files_info)).TC_1_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_1_corr = [];

TC_DATA(length(TC_log_files_info)).TC_2_raw = [];
TC_DATA(length(TC_log_files_info)).TC_2_avg = [];
TC_DATA(length(TC_log_files_info)).TC_2_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_2_corr = [];

TC_DATA(length(TC_log_files_info)).TC_3_raw = [];
TC_DATA(length(TC_log_files_info)).TC_3_avg = [];
TC_DATA(length(TC_log_files_info)).TC_3_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_3_corr = [];

TC_DATA(length(TC_log_files_info)).TC_4_raw = [];
TC_DATA(length(TC_log_files_info)).TC_4_avg = [];
TC_DATA(length(TC_log_files_info)).TC_4_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_4_corr = [];

TC_DATA(length(TC_log_files_info)).TC_5_raw = [];
TC_DATA(length(TC_log_files_info)).TC_5_avg = [];
TC_DATA(length(TC_log_files_info)).TC_5_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_5_corr = [];
```

```
TC_DATA(length(TC_log_files_info)).TC_6_raw = [];
TC_DATA(length(TC_log_files_info)).TC_6_avg = [];
TC_DATA(length(TC_log_files_info)).TC_6_stdev = [];
TC_DATA(length(TC_log_files_info)).TC_6_corr = [];

TC_DATA(length(TC_log_files_info)).TC_true = [];
TC_DATA(length(TC_log_files_info)).P_single = [];

% Initialize counter variable
counter = 1;
counter_max = 0.5*length(TC_log_files_info);

WaitBar = waitbar(0,'Analyzing calibration data...');

% Open Calibration Log Files
for i = 1:1:length(TC_log_files_info)
    filename_TC = strcat(TC_Cal_Folder,'\',TC_log_files_info(i).name);

[TC_0,TC_1,TC_2,TC_3,TC_4,TC_5,TC_6,TC_7,TC_8,TC_9,TC_10,TC_11,TC_12,TC_13] =
...
    importfile_TC(filename_TC);
%     T_0 --> Expansion Space
%     T_1 --> HH-Regen
%     T_2 --> HH Flange - Bypass Side
%     T_3 --> HH Flange - Power Piston Side
%     T_4 --> Displacer Mount
%     T_5 --> Crankcase
%     T_6 --> Power Piston

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate Data from the file
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculate the average reading for each RTD
    TC_0_avg = mean(TC_0); %(deg C)
    TC_1_avg = mean(TC_1); %(deg C)
    TC_2_avg = mean(TC_2); %(deg C)
    TC_3_avg = mean(TC_3); %(deg C)
    TC_4_avg = mean(TC_4); %(deg C)
    TC_5_avg = mean(TC_5); %(deg C)
    TC_6_avg = mean(TC_6); %(deg C)

    % Calculate the "true" temperature as the average of all RTDs
    TC_true = mean([TC_0_avg TC_1_avg TC_2_avg TC_3_avg TC_4_avg TC_5_avg
TC_6_avg]);

    % Calculate the correction term for each RTD
    TC_0_corr = TC_true - TC_0_avg; %(deg C)
    TC_1_corr = TC_true - TC_1_avg; %(deg C)
    TC_2_corr = TC_true - TC_2_avg; %(deg C)
    TC_3_corr = TC_true - TC_3_avg; %(deg C)
    TC_4_corr = TC_true - TC_4_avg; %(deg C)
    TC_5_corr = TC_true - TC_5_avg; %(deg C)
    TC_6_corr = TC_true - TC_6_avg; %(deg C)

    % Calculate the sample standard deviation for each RTD
```

```
    TC_0_stdev = std(TC_0);
    TC_1_stdev = std(TC_1);
    TC_2_stdev = std(TC_2);
    TC_3_stdev = std(TC_3);
    TC_4_stdev = std(TC_4);
    TC_5_stdev = std(TC_5);
    TC_6_stdev = std(TC_6);

    % Find the maximum standard deviation of all the thermocouples
    stdev_max = max([TC_0_stdev TC_1_stdev TC_2_stdev TC_3_stdev TC_4_stdev
TC_5_stdev TC_6_stdev]);

    % Calculate the worst case random uncertainty for a single measurement
    P_single = 2*stdev_max; % t = 2, See example 7.3 in Wheeler and Ganji

    % Store the values from the log file in a structure
    TC_DATA(counter).TC_0_raw = TC_0;
    TC_DATA(counter).TC_0_avg = TC_0_avg;
    TC_DATA(counter).TC_0_stdev = TC_0_stdev;
    TC_DATA(counter).TC_0_corr = TC_0_corr;

    TC_DATA(counter).TC_1_raw = TC_1;
    TC_DATA(counter).TC_1_avg = TC_1_avg;
    TC_DATA(counter).TC_1_stdev = TC_1_stdev;
    TC_DATA(counter).TC_1_corr = TC_1_corr;

    TC_DATA(counter).TC_2_raw = TC_2;
    TC_DATA(counter).TC_2_avg = TC_2_avg;
    TC_DATA(counter).TC_2_stdev = TC_2_stdev;
    TC_DATA(counter).TC_2_corr = TC_2_corr;

    TC_DATA(counter).TC_3_raw = TC_3;
    TC_DATA(counter).TC_3_avg = TC_3_avg;
    TC_DATA(counter).TC_3_stdev = TC_3_stdev;
    TC_DATA(counter).TC_3_corr = TC_3_corr;

    TC_DATA(counter).TC_4_raw = TC_4;
    TC_DATA(counter).TC_4_avg = TC_4_avg;
    TC_DATA(counter).TC_4_stdev = TC_4_stdev;
    TC_DATA(counter).TC_4_corr = TC_4_corr;

    TC_DATA(counter).TC_5_raw = TC_5;
    TC_DATA(counter).TC_5_avg = TC_5_avg;
    TC_DATA(counter).TC_5_stdev = TC_5_stdev;
    TC_DATA(counter).TC_5_corr = TC_5_corr;

    TC_DATA(counter).TC_6_raw = TC_6;
    TC_DATA(counter).TC_6_avg = TC_6_avg;
    TC_DATA(counter).TC_6_stdev = TC_6_stdev;
    TC_DATA(counter).TC_6_corr = TC_6_corr;

    TC_DATA(counter).TC_true = TC_true;
    TC_DATA(counter).P_single = P_single;

    % Increment the counter variable
```

```
    counter = counter + 1;

    % Update Wait Bar
    waitbar(counter / counter_max)

end
close(WaitBar);

%% Fit Curves to Correction Terms
% Fit a curve to the correction terms. Each thermocouple gets its own
% curve. Could potentially use the R^2 value to quantify the systematic
% error.
TC_0_fit = fit([TC_DATA.TC_0_avg]',[TC_DATA.TC_0_corr]','poly3');
TC_1_fit = fit([TC_DATA.TC_1_avg]',[TC_DATA.TC_1_corr]','poly3');
TC_2_fit = fit([TC_DATA.TC_2_avg]',[TC_DATA.TC_2_corr]','poly3');
TC_3_fit = fit([TC_DATA.TC_3_avg]',[TC_DATA.TC_3_corr]','poly3');
TC_4_fit = fit([TC_DATA.TC_4_avg]',[TC_DATA.TC_4_corr]','poly3');
TC_5_fit = fit([TC_DATA.TC_5_avg]',[TC_DATA.TC_5_corr]','poly3');
TC_6_fit = fit([TC_DATA.TC_6_avg]',[TC_DATA.TC_6_corr]','poly3');

% Store the coefficients of the fitted curve equations in a structure
TC_FIT_COEFFICIENTS.TC_0 = coeffvalues(TC_0_fit);
TC_FIT_COEFFICIENTS.TC_1 = coeffvalues(TC_1_fit);
TC_FIT_COEFFICIENTS.TC_2 = coeffvalues(TC_2_fit);
TC_FIT_COEFFICIENTS.TC_3 = coeffvalues(TC_3_fit);
TC_FIT_COEFFICIENTS.TC_4 = coeffvalues(TC_4_fit);
TC_FIT_COEFFICIENTS.TC_5 = coeffvalues(TC_5_fit);
TC_FIT_COEFFICIENTS.TC_6 = coeffvalues(TC_6_fit);

% Display the worst case random uncertainty for a single measurement
D1 = ['Worst case random uncertainty for a single
measurement:',num2str(P_single),' \circC'];
disp(D1);
```

calibrate_TC.m

```
function TC_calibrated = calibrate_TC(Ch_num,TC_raw,TC_FIT_COEFFICIENTS)

% Written by Connor Speer, December 2017
% This function adds correction terms, calculated via the fit coefficients,
% to a vector of input raw data and returns a vector of calibrated data.

%% Inputs
% Ch_num --> Channel number of the transducer. Must be the same as in the
% calibration data.
% P_raw --> raw temperature data in deg C.
% ABS_VAL_FIT_COEFFICIENTS --> 3rd order polynomial fit coefficients for
% correction terms

%% Outputs
% P_calibrated --> Calibrated temperature in deg C.

%% Calculate correction terms
```

```
if Ch_num == 0
    p1 = TC_FIT_COEFFICIENTS.TC_0(1);
    p2 = TC_FIT_COEFFICIENTS.TC_0(2);
    p3 = TC_FIT_COEFFICIENTS.TC_0(3);
    p4 = TC_FIT_COEFFICIENTS.TC_0(4);
elseif Ch_num == 1
    p1 = TC_FIT_COEFFICIENTS.TC_1(1);
    p2 = TC_FIT_COEFFICIENTS.TC_1(2);
    p3 = TC_FIT_COEFFICIENTS.TC_1(3);
    p4 = TC_FIT_COEFFICIENTS.TC_1(4);
elseif Ch_num == 2
    p1 = TC_FIT_COEFFICIENTS.TC_2(1);
    p2 = TC_FIT_COEFFICIENTS.TC_2(2);
    p3 = TC_FIT_COEFFICIENTS.TC_2(3);
    p4 = TC_FIT_COEFFICIENTS.TC_2(4);
elseif Ch_num == 3
    p1 = TC_FIT_COEFFICIENTS.TC_3(1);
    p2 = TC_FIT_COEFFICIENTS.TC_3(2);
    p3 = TC_FIT_COEFFICIENTS.TC_3(3);
    p4 = TC_FIT_COEFFICIENTS.TC_3(4);
elseif Ch_num == 4
    p1 = TC_FIT_COEFFICIENTS.TC_4(1);
    p2 = TC_FIT_COEFFICIENTS.TC_4(2);
    p3 = TC_FIT_COEFFICIENTS.TC_4(3);
    p4 = TC_FIT_COEFFICIENTS.TC_4(4);
elseif Ch_num == 5
    p1 = TC_FIT_COEFFICIENTS.TC_5(1);
    p2 = TC_FIT_COEFFICIENTS.TC_5(2);
    p3 = TC_FIT_COEFFICIENTS.TC_5(3);
    p4 = TC_FIT_COEFFICIENTS.TC_5(4);
elseif Ch_num == 6
    p1 = TC_FIT_COEFFICIENTS.TC_6(1);
    p2 = TC_FIT_COEFFICIENTS.TC_6(2);
    p3 = TC_FIT_COEFFICIENTS.TC_6(3);
    p4 = TC_FIT_COEFFICIENTS.TC_6(4);
end

correction_terms = p1.*TC_raw.^3 + p2.*TC_raw.^2 + p3.*TC_raw + p4;

%% Apply correction terms
TC_calibrated = TC_raw + correction_terms;
```

Futek_Torque_Calibration.m

```
function [Torque_Nm]...
    = Futek_Torque_Calibration(Torque_Voltage)

% Written by Connor Speer - October '17
% The Futek torque transducer was shipped with calibration documents from
% the factory. This function corrects raw data using the calibration
% information provided by Futek.

%% Inputs:
% Torque_Voltage --> Column vector of voltages collected from the torque
```

```
% transducer in (V)

%% Outputs:
% Torque_Nm --> Column vector of torques in (Nm) corresponding to the
% input voltages from the torque transducer.

%% Convert from voltage to torque
% Test data from the calibration certificate:
Cal_Input_Nm = [0.000; 1.695; 3.389; 5.084; 6.779; 8.474; 9.999; 0.000;
0.000; -1.695; -3.389; -5.084; -6.779; -8.474; -9.999; 0.000];
Cal_Output_V = [0.000; 0.846; 1.698; 2.543; 3.391; 4.238; 5.001; 0.013;
0.000; -0.846; -1.691; -2.536; -3.383; -4.231; -4.994; 0.001];

% Fit a linear polynomial to the calibration data.
Cal_Data_Fit = fit(Cal_Output_V,Cal_Input_Nm,'poly1');

% Extract the coefficients of the linear fit
Cal_Data_Fit_coeffs = coeffvalues(Cal_Data_Fit);

% Apply equation of fitted polynomial to calculate torque in (Nm)
Torque_Nm = Torque_Voltage*Cal_Data_Fit_coeffs(1) + Cal_Data_Fit_coeffs(2);
%(Nm)
```

PCB_Pressure_Calibration.m

```
function [Pa_LW35041, Pa_LW35042]...
    = PCB_Pressure_Calibration(Voltage_LW35041, Voltage_LW35042)

% Written by Connor Speer - October '17
% The PCB pressure transducers were shipped with calibration documents from
% the factory. This function corrects raw data using the calibration
% information provided by PCB.

%% Inputs:
% Voltage_LW35041 --> Column vector of voltages collected from the
% transducer with LW35041 serial number in (V)

% Voltage_LW35042 --> Column vector of voltages collected from the
% transducer with LW35042 serial number in (V)

%% Outputs:
% Pa_LW35041 --> Column vector of pressures in (Pa) corresponding to the
% input voltages from the LW35041 transducer.

% Pa_LW35042 --> Column vector of pressures in (Pa) corresponding to the
% input voltages from the LW35042 transducer.

% Use test data provided by PCB to convert raw voltages into pressure in
% (Pa).

%% LW35041 Transducer
% Preallocate space for the pressure column vector
Pa_LW35041 = zeros(length(Voltage_LW35041),1);
```

```
for i = 1:1:length(Voltage_LW35041)
    % See what range of voltage it's in and use the corresponding
    % sensitivity to convert to pressure in (Pa).
    if Voltage_LW35041(i) < 0.9
        Pa_LW35041(i) = Voltage_LW35041(i)*(1000000/3.602); %(Pa)
    elseif Voltage_LW35041(i) >= 0.90001
        Pa_LW35041(i) = Voltage_LW35041(i)*(1000000/3.589); %(Pa)
    end
end

%% LW35042 Transducer
% Preallocate space for the pressure column vector
Pa_LW35042 = zeros(length(Voltage_LW35042),1);
for i = 1:1:length(Voltage_LW35042)
    % See what range of voltage it's in and use the corresponding
    % sensitivity to convert to pressure in (Pa).
    if Voltage_LW35042(i) < 0.9
        Pa_LW35042(i) = Voltage_LW35042(i)*(1000000/3.519); %(Pa)
    elseif Voltage_LW35042(i) >= 0.90001
        Pa_LW35042(i) = Voltage_LW35042(i)*(1000000/3.512); %(Pa)
    end
end
```

Chapter 2

Chapter_2_Plots.m

```
% Chapter_2_Plots.m - Written by Connor Speer - October 2017

% Plot examples of raw data.
clear,clc,close all;


%% Cooling System Input Parameters
V_dot_Cooler = 0.800*1.66667e-5; % Cooler water volume flow rate [m^3/s]
V_dot_ConPipePowCyl = 0.200*1.66667e-5; % Connecting Pipe and Power Cylinder
water volume flow rate [m^3/s]
c_water = 4184; % Specific heat capacity of water in [J/kgK]
dens_water = 1000; % Water density [kg/m^3]

% Calculations
m_dot_Cooler = V_dot_Cooler*dens_water; % Cooler water mass flow rate [kg/s]
m_dot_ConPipePowCyl = V_dot_ConPipePowCyl*dens_water; % Connecting Pipe water
mass flow rate [kg/s]


%% Process Experimental Data
% 44 mm piston mod only
example_data_folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 2 Plots\44mmCCDV Power
Tests';
EXAMPLE_DATA = Process_Data(example_data_folder);
```

```
%% Plot Set-Up
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 326;
% height = 275;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% % For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% Pro Tips %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% To change line width for plots, say "'LineWidth',2".

% To include math symbols with their own font size and font type in the
% axes labels, do this:
% xlabel({'\fontsize{11} Engine Speed \fontsize{11} \fontname{Cambria Math}
\omega \fontsize{11} \fontname{Times New Roman} [RPM]'});

% Here is the degree symbol in case you need it °C. Alternatively, use
% \circ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%% Preliminary Data Plots
% Gas Temperatures vs. Time
figure('Position', [x y width height])
hold on
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_0],'.')
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_1],'.')
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_2],'.')
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_3],'.')
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_4],'.')
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_5],'.')
plot([EXAMPLE_DATA(1).time_TC],[EXAMPLE_DATA(1).T_6],'.')
xlabel('Time (s)','FontName',font,'FontSize',font_size);
ylabel('Gas Temperatures (\circC)','FontName',font,'FontSize',font_size);
legend('Expansion Space','Regenerator Hot','Regenerator Cold Left',...
    'Regenerator Cold Right','Displacer Mount','Crankcase','Power
Cylinder','Location','East')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Expansion Space Gas Temperature vs. Crank Angle
figure('Position', [x y width height])
hold on
theta_deg_TC = EXAMPLE_DATA(1).theta_deg(6:6:end);
plot(theta_deg_TC,[EXAMPLE_DATA(1).T_0],'.')
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size);
ylabel('Expansion Space Gas Temperature
(\circC)','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

% Water Temperatures vs. Time
figure('Position', [x y width height])
hold on
plot([EXAMPLE_DATA(1).time_RTD],[EXAMPLE_DATA(1).RTD_0],'r.')
plot([EXAMPLE_DATA(1).time_RTD],[EXAMPLE_DATA(1).RTD_1],'b.')
plot([EXAMPLE_DATA(1).time_RTD],[EXAMPLE_DATA(1).RTD_2],'k.')
plot([EXAMPLE_DATA(1).time_RTD],[EXAMPLE_DATA(1).RTD_3],'g.')
xlabel('Time (s)','FontName',font,'FontSize',font_size);
ylabel('Water Temperatures (\circC)','FontName',font,'FontSize',font_size);
legend('Cooler Inlet','Cooler Outlet','Con Pipe/Pow Cyl Inlet',...
    'Con Pipe/Pow Cyl Outlet','Location','East')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Water Temperatures vs. Crank Angle
figure('Position', [x y width height])
theta_deg_RTD = EXAMPLE_DATA(1).theta_deg(300:300:end);
hold on
plot(theta_deg_RTD,[EXAMPLE_DATA(1).RTD_0],'r.')
plot(theta_deg_RTD,[EXAMPLE_DATA(1).RTD_1],'b.')
plot(theta_deg_RTD,[EXAMPLE_DATA(1).RTD_2],'k.')
plot(theta_deg_RTD,[EXAMPLE_DATA(1).RTD_3],'g.')
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size);
```

```
ylabel('Water Temperatures (\circC)','FontName',font,'FontSize',font_size);
legend('Cooler Inlet','Cooler Outlet','Con Pipe/Pow Cyl Inlet',...
    'Con Pipe/Pow Cyl Outlet','Location','East')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Rotary Encoder Outputs vs. Time
figure('Position', [x y width height])
hold on
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).A],'r')
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).Z],'b')
xlabel('Time (s)','FontName',font,'FontSize',font_size);
ylabel('Voltage (V)','FontName',font,'FontSize',font_size);
xlim([0.23 0.26])
legend('A-Output','Z-Output')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Pressure vs. Time for PCB and Validynes
figure('Position', [x y width height])
hold on
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).P2_Pa],'r.')
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).P3_Pa],'g.')
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).P4_Pa],'b.')
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).P5_Pa],'k.')
xlabel('Time (s)','FontName',font,'FontSize',font_size);
ylabel('Pressure (Pa)','FontName',font,'FontSize',font_size);
legend('Validyne Power Cylinder','Validyne Crankcase','PCB Power
Cylinder',...
    'PCB Crankcase','Location','East')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off
xlim([0 1])

% Pressure vs. Crank Angle for PCB and Validynes
figure('Position', [x y width height])
hold on
plot([EXAMPLE_DATA(1).theta_deg],[EXAMPLE_DATA(1).P2_Pa],'r.')
plot([EXAMPLE_DATA(1).theta_deg],[EXAMPLE_DATA(1).P3_Pa],'b.')
plot([EXAMPLE_DATA(1).theta_deg],[EXAMPLE_DATA(1).P4_Pa],'g.')
plot([EXAMPLE_DATA(1).theta_deg],[EXAMPLE_DATA(1).P5_Pa],'k.')
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size);
ylabel('Pressure (Pa)','FontName',font,'FontSize',font_size);
legend('Validyne Power Cylinder','Validyne Crankcase','PCB Power
Cylinder',...
    'PCB Crankcase','Location','East')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Pressure vs. Crank Angle for Regenerator Differential Validyne
figure('Position', [x y width height])
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).P1_Pa],'k.')
```

```
xlabel('Time (s)','FontName',font,'FontSize',font_size);
ylabel('Regenerator Differential Pressure
(Pa)','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

% Heater Control Signal vs. Time

% Torque vs. Time
figure('Position', [x y width height])
plot([EXAMPLE_DATA(1).time_Volt],[EXAMPLE_DATA(1).TOR],'.')
xlabel('Time (s)','FontName',font,'FontSize',font_size);
ylabel('Torque (Nm)','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
xlim([0 2])

% Torque vs. Crank Angle
figure('Position', [x y width height])
plot([EXAMPLE_DATA(1).theta_deg],[EXAMPLE_DATA(1).TOR],'r.')
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size);
ylabel('Torque (Nm)','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
```

TC_Time_Constant.m

```
% TC_Time_Constant.m - Written by Connor Speer, November 2016
% Calculations based on example 4-1 in Cengel Heat and Mass Transfer pg 222

clear, clc, close all

%% Input Parameters
% Working gas properties (for air at 100 degC and 1 atm)
rho_air = 0.9458; %(kg/m^3)
k_air = 0.03095; %(W/mK)
Pr_air = 0.7111; % No units.
mu_inf = 2.181e-05; %(kg/ms)

mu_s = 1.729e-05; %(kg/ms)

% Type K Thermocouple junction properties
% Chromel: http://www.matweb.com/search/datasheet.aspx?matguid=e14ca9c83c5
% f45dd945e655fa4edff8b
rho_chromel = 8730; %(kg/m^3)
k_chromel = 19.2; %(W/mK)
c_chromel = 448; %(J/kgK)

% Alumel: http://www.matweb.com/search/DataSheet.aspx?MatGUID=c8c0b158524a
% 446bb953b781dee28efb
rho_alumel = 8610; %(kg/m^3)
k_alumel = 29.7; %(W/mK)
c_alumel = 523; %(J/kgK)
```

```
% Diameter of thermocouple junction
D = 0.00077; %(m)

% Average working gas velocity
v = 10:0.1:100; %(m/s)


%% Calculate Average Thermocouple Junction Properties
rho_TC = (rho_chromel+rho_alumel)/2; %(kg/m^3)
k_TC = (k_chromel+k_alumel)/2; %(W/mK)
c_TC = (c_chromel+c_alumel)/2; %(J/kgK)


%% Calculate Reynolds Number
Re = rho_air.*v.*D./mu_inf;


%% Calculate Nusselt Number
% Whitaker correlation from page 413.
% Valid for 3.5 < Re < 80 000 and 0.7 < Pr < 380.
Nu = 2 + (0.4.*(Re.^0.5) +
0.06.*(Re.^(2/3)))*(Pr_air^0.4)*((mu_inf/mu_s)^0.25);


%% Calculate Convective Heat Transfer Coefficient
h = Nu.*k_air./D; %(W/m^2K)


%% Calculate The Characteristic Length
Lc = (1/6)*D; %(m)


%% Calculate Biot Number
Bi = (h.*Lc)./k_TC; % No units


%% Calculate The Exponent b
b = h./(rho_TC*c_TC*Lc); %(s^-1)


%% Calculate Time Constant
% This is the time for the thermocouple junction to reach 99 % of the
% initial temperature difference for a step change in temperature.
t = -log(0.01)./b; %(s)


% % Display the time constant
% D1 = ['Thermocouple time constant for step change in temperature:
',num2str(t),' s'];
% disp(D1);

%% Plots
```

```
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 326;
% height = 275;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% % For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% Pro Tips %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% To change line width for plots, say "'LineWidth',2".

% To include math symbols with their own font size and font type in the
% axes labels, do this:
% xlabel({'\fontsize{11} Engine Speed \fontsize{11} \fontname{Cambria Math}
\omega \fontsize{11} \fontname{Times New Roman} [RPM]'});

% Here is the degree symbol in case you need it °C. Alternatively, use
% \circ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
figure('Position', [x y width height])
hold on
plot(v,t,'LineWidth',2)
plot(v,repelem(0.6,length(t)),'LineWidth',2);
xlabel('Working Gas Velocity,
(m/s)','FontName',font,'FontSize',font_size)
ylabel('Time,     (s)','FontName',font,'FontSize',font_size)
legend('Thermocouple Response Time','Cycle time at 100 RPM')
hold off
```

Uncertainty_Propagation.m

```
% Uncertainty_Propagation.m - Written by Connor Speer, October 2017

% Calculate uncertainty propagation for calculated quantities.

clear, clc, close all;

%% Input Parameters
w_TOR = 0.1; % Uncertainty in torque (Nm)
TOR = 2; % Maximum measured value of torque (Nm)

w_theta = 5*(pi/180); % Uncertainty in crank position (rad)
w_t = 0.001; % Uncertainty in time elapsed (sec)
t_pow = 10; % Maximum time elapsed for power measurement (s)


W_dot_shaft = 10; % Maximum measured shaft power in (W)
ang_freq = 31.4; % Maximum measured angular frequency (rad/s)

w_V_dot = 1.666667e-8; % Uncertainty in water volume flow rate (m^3/s)
V_dot = 1.3333e-5; % Maximum coolant volume flow rate in (m^3/s)
w_rho_water = 5; % Uncertainty in water density (kg/m^3)
rho_water = 1000; % Maximum water density (kg/m^3)
w_c_water = 5; % Uncertainty in water specific heat capacity (J/kgK)
c_water = 4184; % Maximum water specific heat capacity (J/kgK)
Tdrop_water = 5; % Maximum water temperature drop (K)
Q_dot_in = 500; % Maximum measured heat input rate (W)

t_eff = 1800; % Maximum time elapsed for power measurement (s)
Q_dot_rej = 400; % Maximum measured heat rejection rate (W)

w_I_heater = 0.4; % Uncertainty in heater current (Amps)
I_heater = 16.5; % Maximum measured heater current (Amps)
w_R_heater = 1; % Uncertainty in heater resistance (Ohms)
R_heater = 12.088; % Uncertainty in heater resistance (Ohms)
t_on = 280; % Maximum amount of time heater is on (s)
P_heater = 3250; % Maximum power consumption of heater (W)
OnFrac = 0.156; % Maximum fraction of time heater is on (unitless)

w_TC = 0.01; % Uncertainty in gas temperatures, differential measurement (K)
w_RTD = 0.01; % Uncertainty in water temperatures, differential measurement
(K)
```

```
w_p_Val = 270; % Uncertainty in Validyne absolute pressure (Pa)
w_p_Diff = 170; % Uncertainty in Validyne differential pressure (Pa)
w_p_PCB = 5010; % Uncertainty in PCB absolute pressure (Pa)

% w_delta_theta = 5*(pi/180); % Uncertainty in crank angle difference (rad)


%% Relative Uncertainties in Calculated Quantities
% Gas Temperature Drop
w_Tdrop_gas = sqrt(2)*w_TC; %(deg C)

% Water Temperature Drop
w_Tdrop_water = sqrt(2)*w_RTD; %(deg C)

% Engine Frequency
w_ang_freq = sqrt((w_theta/t_pow)^2 + (w_t/(t_pow^2))^2); %(rad/s)

% Shaft Power
w_W_dot_shaft = W_dot_shaft*sqrt((w_TOR/TOR)^2 + (w_ang_freq/ang_freq)^2);
%(W)

% Heat Rejection Rate
w_Q_dot_rej = Q_dot_rej*sqrt((w_V_dot/V_dot)^2 + (w_rho_water/rho_water)^2 +
...
    (w_c_water/c_water)^2 + (w_Tdrop_water/Tdrop_water)^2); %(unitless)

% Heater Power Consumption
w_Pheater = sqrt((w_I_heater*2*I_heater*R_heater)^2 +
(w_R_heater*(I_heater^2))^2); %(W)

% Fraction of time heater is on
w_OnFrac = sqrt((w_t/t_eff)^2 + (w_t*(t_on/(t_eff^2)))^2); %(unitless)

% Heat Input Rate
w_Q_dot_in = Q_dot_in*sqrt((w_Pheater/P_heater)^2 + (w_OnFrac/OnFrac)^2);
%(W)

% Efficiency
w_eff_thermal = sqrt((w_W_dot_shaft/Q_dot_in)^2 + ...
    (w_Q_dot_in*(W_dot_shaft/(Q_dot_in^2)))^2); %(unitless)

% Gas Temperature Drop
Unc_Tdrop_gas = sqrt(2)*w_TC; %(deg C)

% Water Temperature Drop
Unc_Tdrop_water = sqrt(2)*w_RTD; %(deg C)

% Engine frequency (Hz)
RelUnc_freq = sqrt((w_delta_theta*(1/(360*t_pow)))^2 + ...
    ((w_t*delta_theta)/(360*t_pow^2))^2); %(unitless)
```

Chapter 4

Chapter_4_Plots.m

```
% Chapter_4_Plots.m - Written by Connor Speer - October 2017

% Compare the effect of the 3 modifications on the engine performance.
clear,clc,close all;


%% Cooling System Input Parameters
V_dot_Cooler = 0.800*1.66667e-5; % Cooler water volume flow rate [m^3/s]
V_dot_ConPipePowCyl = 0.200*1.66667e-5; % Connecting Pipe and Power Cylinder
water volume flow rate [m^3/s]
c_water = 4184; % Specific heat capacity of water in [J/kgK]
dens_water = 1000; % Water density [kg/m^3]

% Calculations
m_dot_Cooler = V_dot_Cooler*dens_water; % Cooler water mass flow rate [kg/s]
m_dot_ConPipePowCyl = V_dot_ConPipePowCyl*dens_water; % Connecting Pipe water
mass flow rate [kg/s]


%% Process Experimental Data
% 44 mm piston mod only
prelim_log_file_folder_44mm = 'X:\01_Current_Students\Connor Speer\00_Thesis
and Journal Papers\00_Thesis\00_Data and Plots\Chapter 4 Plots (Mod
Comparison)\Power Tests\Prelim Mod Comparison\44mm';
PRELIM_DATA_44mm = Process_Data(prelim_log_file_folder_44mm);

% 44 mm piston mod and crankcase extension
prelim_log_file_folder_44mmCC = 'X:\01_Current_Students\Connor
Speer\00_Thesis and Journal Papers\00_Thesis\00_Data and Plots\Chapter 4
Plots (Mod Comparison)\Power Tests\Prelim Mod Comparison\44mmCC';
PRELIM_DATA_44mmCC = Process_Data(prelim_log_file_folder_44mmCC);

% 44 mm piston mod, crankcase extension, and dead volume reduction
prelim_log_file_folder_44mmCCDV = 'X:\01_Current_Students\Connor
Speer\00_Thesis and Journal Papers\00_Thesis\00_Data and Plots\Chapter 4
Plots (Mod Comparison)\Power Tests\Prelim Mod Comparison\44mmCCDV';
PRELIM_DATA_44mmCCDV = Process_Data(prelim_log_file_folder_44mmCCDV);

% All three mods
log_file_folder_44mmCCDV = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 4 Plots (Mod
Comparison)\Power Tests\44mmCCDV Power Tests';
DATA_44mmCCDV = Process_Data(log_file_folder_44mmCCDV);

% No dead volume mod
log_file_folder_44mmCC = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 4 Plots (Mod
Comparison)\Power Tests\44mmCC Power Tests';
DATA_44mmCC = Process_Data(log_file_folder_44mmCC);
```

```
% Loaded cool down, all mods
loaded_cool_folder_44mmCCDV = 'X:\01_Current_Students\Connor Speer\00_Thesis
and Journal Papers\00_Thesis\00_Data and Plots\Chapter 4 Plots (Mod
Comparison)\Cool Down Tests\Cool Down with Load (All Mods)';
LOADED_COOL_44mmCCDV = Process_Data(loaded_cool_folder_44mmCCDV);


%% Plot Set-Up
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 326;
% height = 275;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% % For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% Pro Tips %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% To change line width for plots, say "'LineWidth',2".
```

```
% To include math symbols with their own font size and font type in the
% axes labels, do this:
% xlabel({'\fontsize{11} Engine Speed \fontsize{11} \fontname{Cambria Math}
\omega \fontsize{11} \fontname{Times New Roman} [RPM]'});

% Here is the degree symbol in case you need it °C. Alternatively, use
% \circ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

save Calibrated_Data_for_Chapter_4_Plots.mat

%% As Built Data Plots
% % Import data from text file.
% % Script for importing data from the following text file:
% %
% %    X:\04_Stirling_Engine\04_Experiments\03_High Temp Gamma\004_HTG Test
Data April 3 - 2017\019_P150_TH500_TL5_0_Temp.log
% %
% % To extend the code to different selected data or a different text file,
% % generate a function instead of a script.
%
% % Auto-generated by MATLAB on 2017/04/04 17:05:54
%
% % Initialize variables.
% filename = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 4 Plots (Mod Comparison)\As-Built
Cool Down Run\P60_THCOOL_TL21_14_Temp.log';
% delimiter = '\t';
% startRow = 6;
%
% % Format for each line of text:
% %   column1: double (%f)
% % column2: double (%f)
% %   column3: double (%f)
% % column4: double (%f)
% %   column5: double (%f)
% % column6: double (%f)
% %   column7: double (%f)
% % column8: double (%f)
% %   column9: double (%f)
% % column10: double (%f)
% %   column11: double (%f)
% % column12: double (%f)
% %   column13: double (%f)
% % column14: double (%f)
% % For more information, see the TEXTSCAN documentation.
% formatSpec = '%f%f%f%f%f%f%f%f%f%f%f%f%f%f%[^\n\r]';
%
% % Open the text file.
% fileID = fopen(filename,'r');
%
% % Read columns of data according to the format.
% % This call is based on the structure of the file used to generate this
% % code. If an error occurs for a different file, try regenerating the code
% % from the Import Tool.
```

```
% dataArray = textscan(fileID, formatSpec, 'Delimiter', delimiter,
'TextType', 'string', 'EmptyValue', NaN, 'HeaderLines' ,startRow-1,
'ReturnOnError', false, 'EndOfLine', '\r\n');
%
% % Close the text file.
% fclose(fileID);
%
% % Post processing for unimportable data.
% % No unimportable data rules were applied during the import, so no post
% % processing code is included. To generate code which works for
% % unimportable data, select unimportable cells in a file and regenerate the
% % script.
%
% % Allocate imported array to column variable names
% T_0 = dataArray{:, 1};
% T_1 = dataArray{:, 2};
% T_2 = dataArray{:, 3};
% T_3 = dataArray{:, 4};
% T_4 = dataArray{:, 5};
% T_5 = dataArray{:, 6};
% T_6 = dataArray{:, 7};
% T_7 = dataArray{:, 8};
% T_8 = dataArray{:, 9};
% T_9 = dataArray{:, 10};
% T_10 = dataArray{:, 11};
% T_11 = dataArray{:, 12};
% T_12 = dataArray{:, 13};
% T_13 = dataArray{:, 14};
%
%
% % Clear temporary variables
% clearvars filename delimiter startRow formatSpec fileID dataArray ans;
%
% % Import data from text file.
% % Script for importing data from the following text file:
% %
% %    X:\04_Stirling_Engine\04_Experiments\03_High Temp Gamma\004_HTG Test
Data April 3 - 2017\020_P150_TH500_TL5_0_Volt.log
% %
% % To extend the code to different selected data or a different text file,
% % generate a function instead of a script.
%
% % Auto-generated by MATLAB on 2017/04/04 17:06:42
%
% % Initialize variables.
% filenameP = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 4 Plots (Mod Comparison)\As-Built
Cool Down Run\P60_THCOOL_TL21_14_Volt.log';
% delimiter = '\t';
% startRow = 6;
%
% % Format for each line of text:
% %   column1: double (%f)
% % column2: double (%f)
% %   column3: double (%f)
% % column4: double (%f)
% %   column5: double (%f)
```

```
% % column6: double (%f)
% %   column7: double (%f)
% % column8: double (%f)
% % For more information, see the TEXTSCAN documentation.
% formatSpec = '%f%f%f%f%f%f%f%f%[^\n\r]';
%
% % Open the text file.
% fileID = fopen(filenameP,'r');
%
% % Read columns of data according to the format.
% % This call is based on the structure of the file used to generate this
% % code. If an error occurs for a different file, try regenerating the code
% % from the Import Tool.
% dataArray = textscan(fileID, formatSpec, 'Delimiter', delimiter,
'TextType', 'string', 'EmptyValue', NaN, 'HeaderLines' ,startRow-1,
'ReturnOnError', false, 'EndOfLine', '\r\n');
%
% % Close the text file.
% fclose(fileID);
%
% % Post processing for unimportable data.
% % No unimportable data rules were applied during the import, so no post
% % processing code is included. To generate code which works for
% % unimportable data, select unimportable cells in a file and regenerate the
% % script.
%
% % Allocate imported array to column variable names
% A = dataArray{:, 1};
% Z = dataArray{:, 2};
% TOR = dataArray{:, 3};
% P1 = dataArray{:, 4}; % Regenerator Differential Pressure
% P2 = dataArray{:, 5}; % Displacer Mount
% P3 = dataArray{:, 6}; % Power Piston
% P4 = dataArray{:, 7}; % Crankcase
% P5 = dataArray{:, 8}; % Expansion Space
%
% %  Collect the Start and End Times of the log file
% volt_log_file = fileread(filenameP);
% start_index = regexp(volt_log_file,'Start');
% end_index = regexp(volt_log_file,'End');
% start_line = volt_log_file(start_index:end_index-1);
% end_line = volt_log_file(end_index:end_index+14);
% [token_start,remain_start] = strtok(start_line);
% [token_end,remain_end] = strtok(end_line);
% start_time = str2double(strip(remain_start));
% end_time = str2double(strip(remain_end));
%
% % Clear temporary variables
% clearvars filenameP delimiter startRow formatSpec fileID dataArray ans;
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Find reference pulse in Z output
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% threshold = 3;
% ref_pulse = find(Z > threshold, 1);
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% % Work backwards from reference pulse to find pulse counts
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% if A(ref_pulse) > 2.5
%     pulse_flag = 1;
% else
%     pulse_flag = 0;
% end
%
% A_count = zeros(size(A));
%
% bwd_counter = 500;
% row = ref_pulse - 1;
%
% while row ~= 0
%     if pulse_flag == 0 && A(row) < 2.5
%         bwd_counter = bwd_counter - 1;
%     end
%
%     if bwd_counter == -1
%     bwd_counter = 500;
%     end
%
%     A_count(row) = bwd_counter;
%
%     if A(row) < 2.5
%         pulse_flag = 1;
%     else
%         pulse_flag = 0;
%     end
%     row = row - 1;
% end
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Work forwards from reference pulse to find pulse counts
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% if A(ref_pulse) > 2.5
%     pulse_flag = 1;
% else
%     pulse_flag = 0;
% end
%
% fwd_counter = 0;
%
% for row = ref_pulse:length(A)
%     if pulse_flag == 0 && A(row) > 2.5
%         fwd_counter = fwd_counter + 1;
%     end
%
%     if fwd_counter == 501
%     fwd_counter = 0;
%     end
%
%     A_count(row) = fwd_counter;
%
%     if A(row) > 2.5
%         pulse_flag = 1;
%     else
```

```
%         pulse_flag = 0;
%     end
% end
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Use pulse count to calculate crank angles
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% crank_angle = A_count*(360/500); % [deg]
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Calculate Crankshaft Speed
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% refs = find(A < 0.5);
%
% % Initialize variables
% pulse_count = 0;
%
% for inc = 1:(length(refs)-1)
%     if refs(inc+1) ~= (refs(inc) + 1)
%         pulse_count = pulse_count + 1;
%     end
% end
%
% Engine_Freq = ((pulse_count/(end_time-start_time))/500)*(2*pi); % [rad/sec]
% Engine_RPM = Engine_Freq*(60/(2*pi)); % [rev/min]
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Average the Crank Angles and Pressures
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% [avg_P1] = PV_data_avg(crank_angle*(pi/180),P1);
% [avg_P2] = PV_data_avg(crank_angle*(pi/180),P2);
% [avg_P3] = PV_data_avg(crank_angle*(pi/180),P3);
% [avg_P4] = PV_data_avg(crank_angle*(pi/180),P4);
% [avg_P5] = PV_data_avg(crank_angle*(pi/180),P5);
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Convert pressures and average pressures into units of bar
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% p_scale = 20*0.0689476;
% P1_bar = P1.*p_scale;
% P2_bar = P2.*p_scale;
% P3_bar = P3.*p_scale;
% P4_bar = P4.*p_scale;
% P5_bar = P5.*p_scale;
%
% P1_avg_bar = avg_P1.*p_scale;
% P2_avg_bar = avg_P2.*p_scale;
% P3_avg_bar = avg_P3.*p_scale;
% P4_avg_bar = avg_P4.*p_scale;
% P5_avg_bar = avg_P5.*p_scale;
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Use crank angles to calculate volumes
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Gamma Engine Input Parameters
% % Dr1 --> Displacer desaxe offset in [m]
```

```
% % Dr2 --> Displacer crank length in [m]
% % Dr3 --> Displacer connecting rod length in [m]
% % Dbore --> Displacer bore in [m]
% % Pr1 --> Piston desaxe offset in [m]
% % Pr2 --> Piston crank length in [m]
% % Pr3 --> Piston connecting rod length in [m]
% % Pbore --> Piston bore in [m]
% % Vdead --> Total dead volume in [m^3]
% % beta_deg --> Phase angle advance of displacer motion over piston motion
[degrees]
%
% % Input Parameters for HTG Engine
% Dr1 = 0; %[m]
% Dr2 = 0.0375; %[m]
% Dr3 = 0.130; %[m]
% Dbore = 0.096; %[m]
% Pr1 = 0; %[m]
% Pr2  = 0.0375; %[m]
% Pr3 = 0.156; %[m]
% Pbore = 0.085; %[m]
% Vdead = 876.79*1e-6; %[m^3]
% V_crankcase_max = 0.0032; %[m^3]
% beta_deg = 90; %[degrees]
%
% % We use the equations from Cleghorn's "Mechanics of Machines" to determine
% % the volume variations here. Cleghorn defines his angle differently, so
% % here we switch to theta_Cleghorn
% theta_Cleghorn = pi - crank_angle*(pi/180); % [rad]
%
% % Calculate Volumes
% beta = beta_deg*(pi/180);
%
% Dtheta3 = pi - asin((-Dr1+(Dr2*sin(theta_Cleghorn+beta)))/Dr3);
% Dr4 = Dr2*cos(theta_Cleghorn+beta) - Dr3*cos(Dtheta3);
% Dr4max = sqrt(((Dr2+Dr3)^2)-(Dr1^2));
% Ve = ((pi/4)*(Dbore^2))*(Dr4max-Dr4);
% DVc = (((pi/4)*(Dbore^2))*2*Dr2) - Ve;
%
% Ptheta3 = pi - asin((-Pr1+(Pr2*sin(theta_Cleghorn)))/Pr3);
% Pr4 = Pr2*cos(theta_Cleghorn) - Pr3*cos(Ptheta3);
% Pr4max = sqrt(((Pr2+Pr3)^2)-(Pr1^2));
% PVc = (((pi/4)*(Pbore^2))*(Pr4max-Pr4));
% Vc = DVc + PVc;
% Pr4min = sqrt(((Pr3-Pr2)^2)-(Pr1^2));
% Pstroke = Pr4max - Pr4min;
%
% Vtotal = Vdead + (((pi/4)*(Dbore^2))*2*Dr2) + PVc;
%
% % Crankcase Volume Variations
% V_crankcase = V_crankcase_max - ((Pr4max-Pr4)*(((pi/4)*(Pbore^2))));
%
% % Calculate volumes at rounded crank angles to go with average pressures
% crank_angle_rounded_deg = 0:1:359; %[deg]
% theta_Cleghorn_rounded = pi - crank_angle_rounded_deg.*(pi/180); %[rad]
%
% Dtheta3_rounded = pi - asin((-
Dr1+(Dr2*sin(theta_Cleghorn_rounded+beta)))/Dr3);
```

```
% Dr4_rounded = Dr2*cos(theta_Cleghorn_rounded+beta) -
Dr3*cos(Dtheta3_rounded);
% Ve_rounded = ((pi/4)*(Dbore^2))*(Dr4max-Dr4_rounded);
% DVc_rounded = (((pi/4)*(Dbore^2))*2*Dr2) - Ve_rounded;
%
% Ptheta3_rounded = pi - asin((-Pr1+(Pr2*sin(theta_Cleghorn_rounded)))/Pr3);
% Pr4_rounded = Pr2*cos(theta_Cleghorn_rounded) - Pr3*cos(Ptheta3_rounded);
% PVc_rounded = (((pi/4)*(Pbore^2))*(Pr4max-Pr4_rounded));
% Vc_rounded = DVc_rounded + PVc_rounded;
%
% Vtotal_rounded = Vdead + (((pi/4)*(Dbore^2))*2*Dr2) + PVc_rounded;
% Vtotal_rounded = Vtotal_rounded(:);
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Calculate Experimental Indicated Work and Power
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% W_ind = polyarea(Vtotal_rounded,P3_avg_bar*100000);
% P_ind = W_ind*(Engine_RPM/60);
%
%
% % Make Experimental Forced Work Plot
% [W_ind, FW, W_shaft] = ...
% FW_Subfunction_v3(1,P3_avg_bar.*100000, ...
% flipud(P4_avg_bar.*100000),Vtotal_rounded,0.7, 1);

%% Modeling of Modifications
% % Forced Work Plot Comparison for Reduced Piston Diameter
% % 85 mm piston
% ENGINE_DATA = HTG_ENGINE_DATA;
% ENGINE_DATA.Pbore = 0.085;
% ENGINE_DATA.V_buffer_max = 0.0032;
% ENGINE_DATA.Vclp = 1.733e-04;
% ENGINE_DATA.GSH_config = 1;
% ENGINE_DATA.Tsource = 273 + 200;
% ENGINE_DATA.Tge = 273 + 200;
% ENGINE_DATA.Tgh = 273 + 200;
% ENGINE_DATA.Twh = 273 + 200;
%
% [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
%         = HTG_2nd_Order(ENGINE_DATA);
%
% % 44 mm piston
% ENGINE_DATA = HTG_ENGINE_DATA;
% ENGINE_DATA.Pbore = 0.044;
% ENGINE_DATA.V_buffer_max = 0.0032;
% ENGINE_DATA.Vclp = 1.733e-04;
% ENGINE_DATA.GSH_config = 1;
% ENGINE_DATA.Tsource = 273 + 200;
% ENGINE_DATA.Tge = 273 + 200;
% ENGINE_DATA.Tgh = 273 + 200;
% ENGINE_DATA.Twh = 273 + 200;
%
% [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
%         = HTG_2nd_Order(ENGINE_DATA);
%
% % Effect of DV Reduction Components
% % 44 mm piston, extended crankcase, no DV reduction
```

```
% ENGINE_DATA = HTG_ENGINE_DATA;
% ENGINE_DATA.Pbore = 0.044;
% ENGINE_DATA.V_buffer_max = 0.0032 + 0.004633333;
% ENGINE_DATA.Vclp = 1.733e-04;
% ENGINE_DATA.GSH_config = 1;
% ENGINE_DATA.Tsource = 273 + 200;
% ENGINE_DATA.Tge = 273 + 200;
% ENGINE_DATA.Tgh = 273 + 200;
% ENGINE_DATA.Twh = 273 + 200;
%
% [SECOND_ORDER_DATA,REF_CYCLE_DATA,LOSSES_DATA] ...
%         = HTG_2nd_Order(ENGINE_DATA);
%
% Pswing_NoDV = max([REF_CYCLE_DATA.p]) - min([REF_CYCLE_DATA.p]);
% TswingExp_NoDV = max([REF_CYCLE_DATA.Tge]) - min([REF_CYCLE_DATA.Tge]);
% deltaTeffective_NoDV = mean([REF_CYCLE_DATA.Tge]) -
mean([REF_CYCLE_DATA.Tgc]);
%
%
% % 44 mm piston, extended crankcase, DV reduction
% ENGINE_DATA = HTG_ENGINE_DATA;
% ENGINE_DATA.Pbore = 0.044;
% ENGINE_DATA.V_buffer_max = 0.0032 + 0.004633333;
% ENGINE_DATA.Vclp = 1.733e-04 - 0.000132;
% ENGINE_DATA.GSH_config = 1;
% ENGINE_DATA.Tsource = 273 + 200;
% ENGINE_DATA.Tge = 273 + 200;
% ENGINE_DATA.Tgh = 273 + 200;
% ENGINE_DATA.Twh = 273 + 200;
%
% [SECOND_ORDER_DATA_DV,REF_CYCLE_DATA_DV,LOSSES_DATA_DV] ...
%         = HTG_2nd_Order(ENGINE_DATA);
%
% Pswing_DV = max([REF_CYCLE_DATA_DV.p]) - min([REF_CYCLE_DATA_DV.p]);
% TswingExp_DV = max([REF_CYCLE_DATA_DV.Tge]) - min([REF_CYCLE_DATA_DV.Tge]);
% deltaTeffective_DV = mean([REF_CYCLE_DATA_DV.Tge]) -
mean([REF_CYCLE_DATA_DV.Tgc]);

%% Preliminary Modification Data Plots
% Torque vs. Frequency for Several Mean Pressures
figure('Position', [x y width height])
hold on
pointsize = 30;
scatter([PRELIM_DATA_44mm.Engine_Hz],[PRELIM_DATA_44mm.Torque_avg],
pointsize, [PRELIM_DATA_44mm.pmean]./1000,'o')
scatter([PRELIM_DATA_44mmCC.Engine_Hz],[PRELIM_DATA_44mmCC.Torque_avg],
pointsize, [PRELIM_DATA_44mmCC.pmean]./1000,'*')
scatter([PRELIM_DATA_44mmCCDV.Engine_Hz],[PRELIM_DATA_44mmCCDV.Torque_avg],
pointsize, [PRELIM_DATA_44mmCCDV.pmean]./1000,'x')
ylim([0 inf])
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
ylabel('Torque (Nm)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('44 mm Piston','Crankcase Extension','Dead Volume Reduction')
set(gca,'fontsize',font_size);
```

```
set(gca,'FontName',font)
hold off

% Power vs. Frequency for Several Mean Pressures
figure('Position', [x y width height])
hold on
pointsize = 30;
scatter([PRELIM_DATA_44mm.Engine_Hz],[PRELIM_DATA_44mm.P_shaft], pointsize,
[PRELIM_DATA_44mm.pmean]./1000,'o')
scatter([PRELIM_DATA_44mmCC.Engine_Hz],[PRELIM_DATA_44mmCC.P_shaft],
pointsize, [PRELIM_DATA_44mmCC.pmean]./1000,'*')
scatter([PRELIM_DATA_44mmCCDV.Engine_Hz],[PRELIM_DATA_44mmCCDV.P_shaft],
pointsize, [PRELIM_DATA_44mmCCDV.pmean]./1000,'x')
ylim([0 3.5])
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
ylabel('Shaft Power (W)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('44 mm Piston','Crankcase Extension','Dead Volume Reduction')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Speed vs. Pressure
figure('Position', [x y width height])
hold on
plot([PRELIM_DATA_44mm.pmean]./1000,[PRELIM_DATA_44mm.Engine_Hz],'o')
plot([PRELIM_DATA_44mmCC.pmean]./1000,[PRELIM_DATA_44mmCC.Engine_Hz],'*')
plot([PRELIM_DATA_44mmCCDV.pmean]./1000,[PRELIM_DATA_44mmCCDV.Engine_Hz],'x')
xlabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
ylabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
legend('44 mm Piston','Crankcase Extension','Dead Volume
Reduction','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Indicated Power vs. Speed for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
hold on
scatter([PRELIM_DATA_44mm.Engine_Hz],[PRELIM_DATA_44mm.P_ind_PCB], pointsize,
[PRELIM_DATA_44mm.pmean]./1000,'o')
scatter([PRELIM_DATA_44mmCC.Engine_Hz],[PRELIM_DATA_44mmCC.P_ind_PCB],
pointsize, [PRELIM_DATA_44mmCC.pmean]./1000,'*')
scatter([PRELIM_DATA_44mmCCDV.Engine_Hz],[PRELIM_DATA_44mmCCDV.P_ind_PCB],
pointsize, [PRELIM_DATA_44mmCCDV.pmean]./1000,'x')
ylim([0 inf])
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Indicated Power (W)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('44 mm Piston','Crankcase Extension','Dead Volume
Reduction','Location','SouthEast')
```

```
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Crankcase Gas Spring Hysteresis Loss vs. Speed for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
hold on
scatter([PRELIM_DATA_44mm.Engine_Hz],[PRELIM_DATA_44mm.P_GSH_Loss_PCB],
pointsize, [PRELIM_DATA_44mm.pmean]./1000,'o')
scatter([PRELIM_DATA_44mmCC.Engine_Hz],[PRELIM_DATA_44mmCC.P_GSH_Loss_PCB],
pointsize, [PRELIM_DATA_44mmCC.pmean]./1000,'*')
scatter([PRELIM_DATA_44mmCCDV.Engine_Hz],[PRELIM_DATA_44mmCCDV.P_GSH_Loss_PCB
], pointsize, [PRELIM_DATA_44mmCCDV.pmean]./1000,'x')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Crankcase Gas Spring Hysteresis Loss
(W)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('44 mm Piston','Crankcase Extension','Dead Volume
Reduction','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Experimental Forced Work vs. Speed for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
hold on
scatter([PRELIM_DATA_44mm.Engine_Hz],[PRELIM_DATA_44mm.Exp_FW], pointsize,
[PRELIM_DATA_44mm.pmean]./1000,'o')
scatter([PRELIM_DATA_44mmCC.Engine_Hz],[PRELIM_DATA_44mmCC.Exp_FW],
pointsize, [PRELIM_DATA_44mmCC.pmean]./1000,'*')
scatter([PRELIM_DATA_44mmCCDV.Engine_Hz],[PRELIM_DATA_44mmCCDV.Exp_FW],
pointsize, [PRELIM_DATA_44mmCCDV.pmean]./1000,'x')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Experimental Forced Work (J)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('44 mm Piston','Crankcase Extension','Dead Volume
Reduction','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Indicator Diagrams Before and After Crankcase Extension
figure('Position', [x y width height])
hold on
plot([PRELIM_DATA_44mm(33).Vtotal_rounded],[PRELIM_DATA_44mm(33).p_engine_exp
erimental]./1000,'.r')
plot([PRELIM_DATA_44mmCC(41).Vtotal_rounded],[PRELIM_DATA_44mmCC(41).p_engine
_experimental]./1000,'.k')
plot([PRELIM_DATA_44mm(33).Vtotal_rounded],[PRELIM_DATA_44mm(33).p_buffer_exp
erimental]./1000,'.r')
```

```
plot([PRELIM_DATA_44mmCC(41).Vtotal_rounded],[PRELIM_DATA_44mmCC(41).p_buffer
_experimental]./1000,'.k')
xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size);
legend('44 mm Piston','Crankcase Extension','Location','NorthEast')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

%% Further Testing on Dead Volume Reduction Parts Plots
% Power vs. Frequency
figure('Position', [x y 326 275])
hold on
pointsize = 30;
scatter([DATA_44mmCC.Engine_Hz],[DATA_44mmCC.P_shaft], pointsize,'b*')
scatter([DATA_44mmCCDV.Engine_Hz],[DATA_44mmCCDV.P_shaft], pointsize,'rx')
ylim([0 inf])
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
ylabel('Shaft Power (W)','FontName',font,'FontSize',font_size);
legend('No DV Reduction','DV Reduction','Location','SouthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Indicated Power vs. Speed for Several Mean Pressures
figure('Position', [x y 326 275])
pointsize = 30;
hold on
scatter([DATA_44mmCC.Engine_Hz],[DATA_44mmCC.P_ind_PCB], pointsize,'b*')
scatter([DATA_44mmCCDV.Engine_Hz],[DATA_44mmCCDV.P_ind_PCB], pointsize,'rx')
ylim([0 inf])
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Indicated Power (W)','FontName',font,'FontSize',font_size);
legend('No DV Reduction','DV Reduction','Location','SouthEast')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Cooler Heat Removal vs. Speed for Several Mean Pressures
figure('Position', [x y 326 275])
pointsize = 30;
hold on
Q_dot_rej_44mmCC = [DATA_44mmCC.Qdot_Cooler];
Q_dot_rej_44mmCCDV = [DATA_44mmCCDV.Qdot_Cooler];
scatter([DATA_44mmCC.Engine_Hz],Q_dot_rej_44mmCC, pointsize,
[DATA_44mmCC.pmean]./1000,'b*')
scatter([DATA_44mmCCDV.Engine_Hz],Q_dot_rej_44mmCCDV, pointsize,
[DATA_44mmCCDV.pmean]./1000,'rx')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Heat Rejection Rate (W)','FontName',font,'FontSize',font_size);
legend('No DV Reduction','DV Reduction','Location','SouthEast')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Con Pipe/Pow Cyl Heat Removal vs. Speed for Several Mean Pressures
```

```
figure('Position', [x y 326 275])
pointsize = 30;
hold on
Q_dot_rej_44mmCC = [DATA_44mmCC.Qdot_ConPipePowCyl];
Q_dot_rej_44mmCCDV = [DATA_44mmCCDV.Qdot_ConPipePowCyl];
scatter([DATA_44mmCC.Engine_Hz],Q_dot_rej_44mmCC, pointsize, 'b*')
scatter([DATA_44mmCCDV.Engine_Hz],Q_dot_rej_44mmCCDV, pointsize,'rx')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Heat Rejection Rate (W)','FontName',font,'FontSize',font_size);
legend('No DV Reduction','DV Reduction','Location','SouthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

%% Loaded Cool Down Plots for All Mods
% Torque vs. Expansion Space Temperature
figure('Position', [x y 326 275])
pointsize = 30;
hold on
scatter([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.Torque_avg],
pointsize,[LOADED_COOL_44mmCCDV.Engine_Hz],'o')
ylim([0.1 inf])
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size);
ylabel('Torque (Nm)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
colormap(jet)
hold off

% Power vs. Expansion Space Temperature
figure('Position', [x y 326 275])
pointsize = 30;
hold on
scatter([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.P_shaft], pointsize,
[LOADED_COOL_44mmCCDV.Engine_Hz],'o')
ylim([0 inf])
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size);
ylabel('Shaft Power (W)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
colormap(jet)
hold off

% Shaft Work vs. Expansion Space Temperature
figure('Position', [x y 326 275])
pointsize = 30;
hold on
scatter([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.P_shaft]./[LOADED_CO
OL_44mmCCDV.Engine_Hz], pointsize, [LOADED_COOL_44mmCCDV.Engine_Hz],'o')
% ylim([0 inf])
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size);
ylabel('Shaft Work (J)','FontName',font,'FontSize',font_size);
c = colorbar;
```

```
ylabel(c,'Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
colormap(jet)
hold off

% Speed vs. Expansion Space Temperature
figure('Position', [x y width height])
hold on
plot([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.Engine_Hz],'ko')
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off


% Indicated Work vs. Expansion Space Temperature
figure('Position', [x y width height])
hold on
plot([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.P_ind_PCB]./[LOADED_COO
L_44mmCCDV.Engine_Hz],'o')
ylim([0 inf])
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Indicated Work (J)','FontName',font,'FontSize',font_size);
hold off

% Series of Indicator Diagrams for Dead Volume Reduction
figure('Position', [x y width height])
pointsize = 1;
hold on

for i = 1:length(LOADED_COOL_44mmCCDV)
    if LOADED_COOL_44mmCCDV(i).Tge > 180

scatter([LOADED_COOL_44mmCCDV(i).Vtotal_rounded],[LOADED_COOL_44mmCCDV(i).p_e
ngine_experimental]./1000,pointsize,repelem([LOADED_COOL_44mmCCDV(i).Tge],360
))
    end
end
xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
ylabel({'Engine Pressure (kPa)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
colormap(jet)

% Series of Buffer Pressure Diagrams for Dead Volume Reduction
figure('Position', [x y width height])
pointsize = 1;
hold on
for i = 1:length(LOADED_COOL_44mmCCDV)
    if LOADED_COOL_44mmCCDV(i).Tge > 180

scatter([LOADED_COOL_44mmCCDV(i).Vtotal_rounded],[LOADED_COOL_44mmCCDV(i).p_b
```

```
uffer_experimental]./1000,pointsize,repelem([LOADED_COOL_44mmCCDV(i).Tge],360
))
    end
end
xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
ylabel({'Buffer Pressure (kPa)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
colormap(jet)

% Crankcase and Power Cylinder Mean Pressures vs. Expansion space
% temperature for Dead Volume Reduction
figure('Position', [x y width height])
hold on
for i = 1:length(LOADED_COOL_44mmCCDV)
plot([LOADED_COOL_44mmCCDV(i).Tge],mean([LOADED_COOL_44mmCCDV(i).p_engine_exp
erimental]./1000),'or')
plot([LOADED_COOL_44mmCCDV(i).Tge],mean([LOADED_COOL_44mmCCDV(i).p_buffer_exp
erimental]./1000),'*k')
end
hold off
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size)
legend('Mean Power Cylinder Pressure','Mean Crankcase Pressure')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)

% Experimental Forced Work vs. Expansion Space Temperature, All Mods
figure('Position', [x y 326 275])
hold on
plot([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.Exp_FW],'r*')
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Forced Work (J)','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Experimental Indicated Work vs. Expansion Space Temperature, All Mods
figure('Position', [x y 326 275])
hold on
plot([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.P_ind_PCB]./[LOADED_COO
L_44mmCCDV.Engine_Hz],'ko')
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Indicated Work (J)','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% Forced Work Ratio vs. Expansion Space Temperature
figure('Position', [x y width height])
hold on
```

```
plot([LOADED_COOL_44mmCCDV.Tge],[LOADED_COOL_44mmCCDV.Exp_FW]./([LOADED_COOL_
44mmCCDV.P_ind_PCB]./[LOADED_COOL_44mmCCDV.Engine_Hz]),'ko')
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Forced Work Ratio','FontName',font,'FontSize',font_size);
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off

% GSH Loss vs. Expansion Space Temperature
figure('Position', [x y width height])
pointsize = 30;
scatter([LOADED_COOL_44mmCCDV.Tge],([LOADED_COOL_44mmCCDV.P_GSH_Loss_PCB]./[L
OADED_COOL_44mmCCDV.Engine_Hz]),
pointsize,[LOADED_COOL_44mmCCDV.Engine_Hz],'o')
xlabel('Expansion Space Temperature
(\circC)','FontName',font,'FontSize',font_size)
ylabel('Gas Spring Hysteresis (J)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
colormap(jet)
```

Chapter 5

Chapter_5_Plots.m

```
% Chapter_5_Plots.m - Written by Connor Speer - October 2017

% Compare model to experimental data.

clear,clc,close all;


%% Cooling System Input Parameters
V_dot_Cooler = 0.800*1.66667e-5; % Cooler water volume flow rate [m^3/s]
V_dot_ConPipePowCyl = 0.200*1.66667e-5; % Connecting Pipe and Power Cylinder
water volume flow rate [m^3/s]
c_water = 4184; % Specific heat capacity of water in [J/kgK]
dens_water = 1000; % Water density [kg/m^3]

% Calculations
m_dot_Cooler = V_dot_Cooler*dens_water; % Cooler water mass flow rate [kg/s]
m_dot_ConPipePowCyl = V_dot_ConPipePowCyl*dens_water; % Connecting Pipe water
mass flow rate [kg/s]


%% Process Experimental Data
% % Efficiency data, TH = 300, TC = 21, P = 60
% efficiency_file_folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Efficiency';
% EFFICIENCY_DATA = Process_Data(efficiency_file_folder);
%
```

```
% % Conduction Loss Data
% conduction_loss_file_folder = 'X:\01_Current_Students\Connor
Speer\00_Thesis and Journal Papers\00_Thesis\00_Data and Plots\Chapter 5
Plots (Model Validation)\Conduction Loss';
% COND_LOSS_DATA = Process_Data(conduction_loss_file_folder);

% Fully Modified Tests
full_mods_200deg_file_folder = 'X:\01_Current_Students\Connor Speer\00_Thesis
and Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Fully Modified Tests\TH 200';
TH200_DATA = Process_Data(full_mods_200deg_file_folder);

% Indicator Diagram Plots
full_mods_PV_file_folder = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Fully Modified Tests\P 60 TH 200 and 300';
PV_DATA = Process_Data(full_mods_PV_file_folder);



%% Plot Set-Up
set(0,'defaultfigurecolor',[1 1 1])

% For Connor's Thesis Paper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Location of Figures
x = 500;
y = 500;

% Size of Figures
width = 550;
height = 400;

% Font For Figures
font = 'Arial';
font_size = 11;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% For Paper with 2 Columns (Save as enhanced metafile) %%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
%
% % Size of Figures
% width = 326;
% height = 275;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% % For Presentation (edit --> copy figure) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Location of Figures
% x = 500;
% y = 500;
```

```
%
% % Size of Figures
% width = 550;
% height = 400;
%
% % Font For Figures
% font = 'Times New Roman';
% font_size = 16;

% Pro Tips %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% To change line width for plots, say "'LineWidth',2".

% To include math symbols with their own font size and font type in the
% axes labels, do this:
% xlabel({'\fontsize{11} Engine Speed \fontsize{11} \fontname{Cambria Math}
\omega \fontsize{11} \fontname{Times New Roman} [RPM]'});

% Here is the degree symbol in case you need it °C. Alternatively, use
% \circ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Efficiency Plots
% % Thermal Efficiency vs. Speed for Several Mean Pressures
% figure('Position', [x y width height])
% Sorted_efficiencies = [EFFICIENCY_DATA(5:7).eff_thermal
EFFICIENCY_DATA(1:4).eff_thermal];
% Sorted_frequencies = [EFFICIENCY_DATA(5:7).Filename_Hz
EFFICIENCY_DATA(1:4).Filename_Hz];
% plot(Sorted_frequencies,Sorted_efficiencies.*100,'--o','LineWidth',2)
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
% ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size);
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
%
% % Thermal Efficiency vs. Speed for Several Mean Pressures with Overlayed
2nd Order Model
% figure('Position', [x y width height])
% hold on
% plot([EFFICIENCY_DATA.Filename_Hz],[EFFICIENCY_DATA.eff_thermal].*100,'r*')
%
plot([EFFICIENCY_DATA.Filename_Hz],[EFFICIENCY_DATA.eff_thermal_2nd_order].*1
00,'bo')
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
% ylabel('Thermal Efficiency (%)','FontName',font,'FontSize',font_size);
% legend('Experimental Data','2nd Order Model','Location','NorthWest')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
% hold off
%
% % Heat Input with Overlaid Model
% figure('Position', [x y width height])
% hold on
% plot([EFFICIENCY_DATA.Filename_Hz],[EFFICIENCY_DATA.Q_dot_in],'r*')
%
plot([EFFICIENCY_DATA.Filename_Hz],[EFFICIENCY_DATA.Q_dot_in_2nd_order],'bo')
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
```

```
% ylabel('Heat Input Rate (W)','FontName',font,'FontSize',font_size);
% legend('Experimental Data','2nd Order Model','Location','NorthWest')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
% hold off
%
% % Shaft Power with Overlaid Model
% figure('Position', [x y width height])
% pointsize = 30;
% hold on
% scatter([EFFICIENCY_DATA.Filename_Hz],[EFFICIENCY_DATA.Filename_P_shaft],
pointsize, [EFFICIENCY_DATA.pmean],'*')
% scatter([EFFICIENCY_DATA.Filename_Hz],[EFFICIENCY_DATA.P_2nd_order],
pointsize, [EFFICIENCY_DATA.pmean],'o')
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% ylabel('Shaft Power (W)','FontName',font,'FontSize',font_size);
% c = colorbar;
% ylabel(c,'Mean Pressure (Pa)','FontName',font,'FontSize',font_size)
% colormap(jet)
% legend('Experimental Data','2nd Order Model')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
% hold off


%% Conduction Loss Plots
% % Heat Rejection Rates vs. Heating Cap Setpoint with Overlaid Conduction
% % Loss Model
% figure('Position', [x y 326 275])
% Q_dot_rej = [COND_LOSS_DATA.Qdot_Cooler] +
[COND_LOSS_DATA.Qdot_ConPipePowCyl];
% hold on
% plot([COND_LOSS_DATA.Cap_Setpoint],Q_dot_rej,'--ob','LineWidth',2)
%
plot([COND_LOSS_DATA.Cap_Setpoint],[COND_LOSS_DATA.Conduction_Loss_Model],'-
k','LineWidth',2)
% xlabel('Heating Cap Setpoint (°C)','FontName',font,'FontSize',font_size)
% ylabel('Conduction Loss (W)','FontName',font,'FontSize',font_size);
% legend('Measured Heat Rejection Rate','Conduction Loss
Model','Location','NorthWest')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
% hold off
%
% % Measured Conduction Loss as a Fraction of the Measured Heat Input Rate
% figure('Position', [x y 326 275])
% Q_dot_rej_percent = (Q_dot_rej./[COND_LOSS_DATA.Q_dot_in]).*100;
% hold on
% plot([COND_LOSS_DATA.Cap_Setpoint],Q_dot_rej_percent,'--ob','LineWidth',2)
% xlabel('Heating Cap Setpoint (°C)','FontName',font,'FontSize',font_size)
% ylabel({'Conduction Loss as a';'Fraction of Heat Input
(%)'},'FontName',font,'FontSize',font_size);
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
% hold off
%
```

```
% % Heat Loss Through Insulation
% figure('Position', [x y 326 275])
% Q_dot_rej = [COND_LOSS_DATA.Qdot_Cooler] +
[COND_LOSS_DATA.Qdot_ConPipePowCyl];
% Q_loss_insulation = [COND_LOSS_DATA.Q_dot_in] - Q_dot_rej;
% plot([COND_LOSS_DATA.Cap_Setpoint],Q_loss_insulation,'--ok','LineWidth',2)
% xlabel('Heating Cap Setpoint (°C)','FontName',font,'FontSize',font_size)
% ylabel('Heat Lost Through Insulation
(W)','FontName',font,'FontSize',font_size);
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
%
% % Fraction of Heat Input Lost Through the Insulation
% figure('Position', [x y 326 275])
% Q_loss_insulation_percent =
(Q_loss_insulation./[COND_LOSS_DATA.Q_dot_in]).*100;
% plot([COND_LOSS_DATA.Cap_Setpoint],Q_loss_insulation_percent,'--
ok','LineWidth',2)
% xlabel('Heating Cap Setpoint (°C)','FontName',font,'FontSize',font_size)
% ylabel({'Heat Lost Through Insulation';'as a Fraction of Heat Input
(%)'},'FontName',font,'FontSize',font_size);
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)


%% Gas Spring Hysteresis Plots
% Crankcase Gas Spring Hysteresis Loss vs. Speed for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_GSH_Loss_PCB]./[TH200_DATA.Engin
e_Hz], pointsize, [TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Crankcase Gas Spring';'Hysteresis Loss Per Cycle
(J)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Crankcase Gas Spring Hysteresis Loss vs. Speed for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_GSH_Loss_PCB], pointsize,
[TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Crankcase Gas Spring Hysteresis Loss
(W)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% 3D Plot of Crankcase Gas Spring Hysteresis Loss vs. Speed and Mean Pressure
figure('Position', [x y width height])
```

```
X = [TH200_DATA.Engine_Hz]';
Y = [TH200_DATA.pmean]./1000;
Y = Y';
Z = [TH200_DATA.P_GSH_Loss_PCB]';
[sf,gof] = fit([X,Y],Z,'poly22');
plot(sf,[X,Y],Z)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
zlabel({'Crankcase Gas Spring';'Hysteresis Loss
(W)'},'FontName',font,'FontSize',font_size)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Crankcase Gas Spring Hysteresis Loss vs. Speed for Several Mean Pressures
% with Overlayed Empirical Model
figure('Position', [x y width height])
hold on
plot([TH200_DATA.Engine_Hz],[TH200_DATA.P_GSH_Loss_PCB],'ok')
P_hys_Empirical =
HTG_GSH_Empirical([TH200_DATA.Engine_Hz],[TH200_DATA.pmean],4);
plot([TH200_DATA.Engine_Hz],P_hys_Empirical,'*r')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Crankcase Gas Spring';' Hysteresis Loss
(W)'},'FontName',font,'FontSize',font_size)
legend('Experimental Data','Empirical Model','Location','northwest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off


%% Indicated Work Plots
% Ideal Isothermal Model and Ideal Adiabatic Model Error in Indicated
% Work vs. Speed for Two Heating Cap Temperatures
figure('Position', [x y width height])
pointsize = 30;
hold on
scatter([PV_DATA.Engine_Hz],[PV_DATA.Error_isothermal], pointsize,
[PV_DATA.Cap_Setpoint],'o')
scatter([PV_DATA.Engine_Hz],[PV_DATA.Error_adiabatic], pointsize,
[PV_DATA.Cap_Setpoint],'*')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Indicated Work Error (%)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Heating Cap Temperature
(\circC)','FontName',font,'FontSize',font_size)
mycolors = [0 0 1; 1 0 0];
colormap(mycolors)
c.Ticks = [200 300];
legend('Ideal Isothermal Model','Ideal Adiabatic
Model','Location','NorthWest')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)
hold off

% Experimental Indicator Diagram with Overlaid isothermal and adiabatic
% models
```

```
% for i = 1:length([PV_DATA.filename])-1
%     figure('Position', [x y width height])
%     hold on
%
plot([PV_DATA(i).Vtotal_rounded],[PV_DATA(i).p_engine_experimental]./1000,'*k
')
%
plot([PV_DATA(i).Vtotal_rounded],[PV_DATA(i).p_engine_isothermal]./1000,'r','
LineWidth',2)
%
plot([PV_DATA(i).Vtotal_rounded],[PV_DATA(i).p_engine_adiabatic]./1000,'b','L
ineWidth',2)
%     hold off
%     xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
%     ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size)
%     legend('Experimental Data','Ideal Isothermal Model','Ideal Adiabatic
Model')
%     title(PV_DATA(i).filename)
%     set(gca,'fontsize',font_size)
%     set(gca,'FontName',font)
% end

% Example Indicator Diagram
figure('Position', [x y 326 275])
hold on
plot([PV_DATA(28).Vtotal_rounded],[PV_DATA(28).p_engine_experimental]./1000,'
ok','MarkerSize',2)
plot([PV_DATA(28).Vtotal_rounded],[PV_DATA(28).p_engine_isothermal]./1000,'r'
,'LineWidth',2)
plot([PV_DATA(28).Vtotal_rounded],[PV_DATA(28).p_engine_adiabatic]./1000,'b',
'LineWidth',2)
hold off
xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size)
xlim([0.00134 0.00148])
ylim([300 550])
legend('Exp Data','Isothermal','Adiabatic')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

%% Teknic Spin Test Plots
% % Friction_Plots.m - Written by Connor Speer - July 2017
%
% % Calls the overall plots subfunction for several data folders and produces
% % plots.
%
% % Fully Assembled
% log_file_folder_1 = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Tecknic Spin Tests\01_Fully Assembled';
% DATA_STRUCTURE_1 =
Subfunction_Overall_Plots_Tecknic_Spin(log_file_folder_1);
%
% % Crankcase Cover Removed
```

```
% log_file_folder_2 = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Tecknic Spin Tests\02_Crankcase Cover Removed';
% DATA_STRUCTURE_2 =
Subfunction_Overall_Plots_Tecknic_Spin(log_file_folder_2);
%
% % Crankcase Cover and Heater Head Removed
% log_file_folder_3 = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Tecknic Spin Tests\03_Crankcase Cover & Heater Head Removed';
% DATA_STRUCTURE_3 =
Subfunction_Overall_Plots_Tecknic_Spin(log_file_folder_3);
%
% % CC Cover, HH, and Power Cylinder Head Removed
% log_file_folder_4 = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Tecknic Spin Tests\04_CC Cover & HH & Power Cyl Head Removed';
% DATA_STRUCTURE_4 =
Subfunction_Overall_Plots_Tecknic_Spin(log_file_folder_4);
%
% % CC Cover, HH, P Cyl Head, and Cooler Removed
% log_file_folder_5 = 'X:\01_Current_Students\Connor Speer\00_Thesis and
Journal Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model
Validation)\Tecknic Spin Tests\05_CC Cover & HH & Power Cyl Head & Cooler
Removed';
% DATA_STRUCTURE_5 =
Subfunction_Overall_Plots_Tecknic_Spin(log_file_folder_5);
%
% % Plot 1 - Total Friction vs. Speed Showing Relative Contributions
% figure('Position', [x y width height])
% hold on
% speed = [50 100 200 300 400 500 600];
% frequency = speed./60;
%
% plot(frequency,-[DATA_STRUCTURE_1.P_shaft],'ob')
% plot(frequency,-[DATA_STRUCTURE_2.P_shaft],'*r')
% plot(frequency,-[DATA_STRUCTURE_3.P_shaft],'^k')
% plot(frequency,-[DATA_STRUCTURE_4.P_shaft],'xm')
% plot(frequency,-[DATA_STRUCTURE_5.P_shaft],'.g')
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% ylabel('Power Consumption (W)','FontName',font,'FontSize',font_size)
% legend('Fully Assembled','No CC Cover','No CC Cover or HH','No CC Cover,
HH, or P Cyl Head','No CC Cover, HH, P Cyl Head, or
Cooler','Location','NorthWest')
% hold off
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)

% % Plot 2 - Indicated Power Consumption, Measured Shaft Power Consumption,
% % Mechanism Effectiveness
% figure('Position', [x y width height])
% hold on
%
% yyaxis left
% plot(frequency,[DATA_STRUCTURE_1.P_ind],'ob')
% plot(frequency,-[DATA_STRUCTURE_1.P_shaft],'*b')
```

```
% ylabel('Indicated Power Consumption
(W)','FontName',font,'FontSize',font_size)
% set(gca,'ycolor','b')
%
% yyaxis right
% plot(frequency,-[DATA_STRUCTURE_1.P_ind]./[DATA_STRUCTURE_1.P_shaft],'or')
% ylabel('Mechanism Effectiveness','FontName',font,'FontSize',font_size)
% set(gca,'ycolor','r')
%
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% xlim([0 7])
% legend('Indicated Power Consumption','Power Supplied by Electric
Motor','Mechanism Effectiveness','Location','North')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)


%% Buffer Pressure, Forced Work, and Mechanism Effectiveness Plots
% Series of Buffer Pressure Indicator Diagrams with Overlaid Isothermal and
% Adiabatic Buffer Pressure Models
% for i = 1:length([TH200_DATA.filename])-1
%     figure('Position', [x y width height])
%     hold on
%
plot([TH200_DATA(i).Vtotal_rounded],[TH200_DATA(i).p_buffer_experimental]./10
00,'ok')
%
plot([TH200_DATA(i).Vtotal_rounded],[TH200_DATA(i).p_buffer_isothermal]./1000
,'r','LineWidth',2)
%
plot([TH200_DATA(i).Vtotal_rounded],[TH200_DATA(i).p_buffer_adiabatic]./1000,
'b','LineWidth',2)
%     hold off
%     xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
%     ylabel('Pressure (kPa)','FontName',font,'FontSize',font_size)
%     legend('Experimental Data','Isothermal Model','Adiabatic
Model','Location','NorthWest')
%     title(TH200_DATA(i).filename)
%     set(gca,'fontsize',font_size)
%     set(gca,'FontName',font)
% end

% Example buffer pressure diagram with overlaid isothermal and adiabatic
% models
figure('Position', [x y width height])
hold on
plot([TH200_DATA(44).Vtotal_rounded],[TH200_DATA(44).p_buffer_experimental]./
1000,'ok')
plot([TH200_DATA(44).Vtotal_rounded],[TH200_DATA(44).p_buffer_isothermal]./10
00,'r','LineWidth',2)
plot([TH200_DATA(44).Vtotal_rounded],[TH200_DATA(44).p_buffer_adiabatic]./100
0,'b','LineWidth',2)
hold off
xlabel('Engine Volume (m^3)','FontName',font,'FontSize',font_size)
ylabel('Crankcase Pressure (kPa)','FontName',font,'FontSize',font_size)
```

```
legend('Experimental Data','Isothermal Assumption','Adiabatic
Assumption','Location','NorthWest')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

%% Overall Torque and Power Plots
% Torque vs. Frequency for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
scatter([TH200_DATA.Engine_Hz],[TH200_DATA.Torque_avg], pointsize,
[TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
ylabel('Torque (Nm)','FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
ylim([0 inf])
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Power vs. Frequency for Several Mean Pressures
figure('Position', [x y width height])
pointsize = 30;
scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_shaft], pointsize,
[TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
ylabel('Shaft Power (W)','FontName',font,'FontSize',font_size);
ylim([0 inf])
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Power vs. Speed for Several Mean Pressures with Overlayed 2nd Order Model
figure('Position', [x y width height])
hold on
pointsize = 30;
scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_shaft], pointsize,
[TH200_DATA.pmean]./1000,'*')
scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_2nd_order], pointsize,
[TH200_DATA.pmean]./1000,'o')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size);
ylabel('Shaft Power (W)','FontName',font,'FontSize',font_size);
% ylim([0 inf])
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('Experimental Data','2nd Order Model','Location','NorthWest')
set(gca,'fontsize',font_size);
set(gca,'FontName',font)
hold off


%% Relative Cooling Zone Plots
% Heat Rejection rate with overlaid model
```

```
figure('Position', [x y width height])
pointsize = 30;
hold on
scatter([TH200_DATA.Engine_Hz],
[TH200_DATA.Qdot_Cooler]+[TH200_DATA.Qdot_ConPipePowCyl], pointsize,
[TH200_DATA.pmean]./1000,'*')
scatter([TH200_DATA.Engine_Hz], [TH200_DATA.Q_dot_rej_2nd_order], pointsize,
[TH200_DATA.pmean]./1000,'o')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Heat Rejection Rate (W)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('Experimental Data','2nd Order Model')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Heat Rejected in both zones
figure('Position', [x y width height])
pointsize = 30;
hold on
scatter([TH200_DATA.Engine_Hz], [TH200_DATA.Qdot_Cooler], pointsize,
[TH200_DATA.pmean]./1000,'o')
scatter([TH200_DATA.Engine_Hz], [TH200_DATA.Qdot_ConPipePowCyl], pointsize,
[TH200_DATA.pmean]./1000,'*')
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Heat Rejection Rate (W)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
legend('Cooler','Con Pipe and Pow Cyl')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Relative Heat Rejection of Con Pipe/Pow Cyl Zone
figure('Position', [x y width height])
pointsize = 30;
RelQrejCPPC = ([TH200_DATA.Qdot_ConPipePowCyl]./ ...
    ([TH200_DATA.Qdot_ConPipePowCyl] + [TH200_DATA.Qdot_Cooler]))*100;
scatter([TH200_DATA.Engine_Hz], RelQrejCPPC, pointsize,
[TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Portion of Heat Rejected in'; 'Connecting Pipe and Power Cylinder
(%)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)


%% Heat Exchanger and Regenerator Performance Plots
% Temperature Drop Between Cap Setpoint and Gas in Heater vs. Speed for
Several Mean Pressures
figure('Position', [x y 326 275])
pointsize = 30;
```

```
HeaterTdrop = [TH200_DATA.Cap_Setpoint] - [TH200_DATA.Tgh];
scatter([TH200_DATA.Engine_Hz],HeaterTdrop, pointsize,
[TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Heater Temperature Drop
(˚C)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Temperature Drop Between Gas in Cooler and Bath Setpoint for Several Mean
Pressures
figure('Position', [x y 326 275])
pointsize = 30;
CoolerTdrop = [TH200_DATA.Tgk] - [TH200_DATA.Bath_Setpoint];
scatter([TH200_DATA.Engine_Hz], CoolerTdrop, pointsize,
[TH200_DATA.pmean]./1000)
xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
ylabel({'Cooler Temperature Drop
(˚C)'},'FontName',font,'FontSize',font_size);
c = colorbar;
ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
colormap(jet)
set(gca,'fontsize',font_size)
set(gca,'FontName',font)


% Regenerator Pressure Drop vs. Crank Angle for Several Mean Pressures
% Overlaid Model
figure('Position', [x y  326 275])
theta_deg = 0:1:359;
hold on
plot(theta_deg,[TH200_DATA(97).pdrop_regen_experimental]./1000,'--
k','LineWidth',2)
plot(theta_deg,[TH200_DATA(100).pdrop_regen_experimental]./1000,'--
g','LineWidth',2)
plot(theta_deg,[TH200_DATA(109).pdrop_regen_experimental]./1000,'--
r','LineWidth',2)
plot(theta_deg,[TH200_DATA(97).pdrop_regen_model]./1000,'k','LineWidth',2)
plot(theta_deg,[TH200_DATA(100).pdrop_regen_model]./1000,'g','LineWidth',2)
plot(theta_deg,[TH200_DATA(109).pdrop_regen_model]./1000,'r','LineWidth',2)
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
ylabel('Regenerator Pressure Drop
(kPa)','FontName',font,'FontSize',font_size)
xlim([0 360])
legend('freq = 2.3 Hz','freq = 0.85 Hz','freq = 1.7 Hz','Location','South')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% Regenerator Pressure Drop vs. Crank Angle for Several Rotational Speeds
% Overlaid Model
figure('Position', [x y  326 275])
theta_deg = 0:1:359;
hold on
```

```
plot(theta_deg,[TH200_DATA(5).pdrop_regen_experimental]./1000,'--
k','LineWidth',2)
plot(theta_deg,[TH200_DATA(73).pdrop_regen_experimental]./1000,'--
g','LineWidth',2)
plot(theta_deg,[TH200_DATA(87).pdrop_regen_experimental]./1000,'--
r','LineWidth',2)
plot(theta_deg,[TH200_DATA(5).pdrop_regen_model]./1000,'k','LineWidth',2)
plot(theta_deg,[TH200_DATA(73).pdrop_regen_model]./1000,'g','LineWidth',2)
plot(theta_deg,[TH200_DATA(87).pdrop_regen_model]./1000,'r','LineWidth',2)
xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
ylabel('Regenerator Pressure Drop
(kPa)','FontName',font,'FontSize',font_size)
xlim([0 360])
legend('pmean = 707 kPa','pmean = 293 kPa','pmean = 421
kPa','Location','South')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)

% % Regenerator Pressure Drop vs. Crank Angle for Several Mean Pressures
% % Overlaid "Validated" Model
% figure('Position', [x y  326 275])
% theta_deg = 0:1:359;
% hold on
% plot(theta_deg,[TH200_DATA(97).pdrop_regen_experimental]./1000,'--
k','LineWidth',2)
% plot(theta_deg,[TH200_DATA(100).pdrop_regen_experimental]./1000,'--
g','LineWidth',2)
% plot(theta_deg,[TH200_DATA(109).pdrop_regen_experimental]./1000,'--
r','LineWidth',2)
%
plot(theta_deg,[TH200_DATA(97).pdrop_regen_model]./1000.*8,'k','LineWidth',2)
%
plot(theta_deg,[TH200_DATA(100).pdrop_regen_model]./1000.*8,'g','LineWidth',2
)
%
plot(theta_deg,[TH200_DATA(109).pdrop_regen_model]./1000.*8,'r','LineWidth',2
)
% xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
% ylabel('Regenerator Pressure Drop
(kPa)','FontName',font,'FontSize',font_size)
% xlim([0 360])
% legend('freq = 2.3 Hz','freq = 0.85 Hz','freq = 1.7 Hz','Location','South')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
%
% % Regenerator Pressure Drop vs. Crank Angle for Several Rotational Speeds
% % Overlaid "Validated" Model
% figure('Position', [x y  326 275])
% theta_deg = 0:1:359;
% hold on
% plot(theta_deg,[TH200_DATA(5).pdrop_regen_experimental]./1000,'--
k','LineWidth',2)
% plot(theta_deg,[TH200_DATA(73).pdrop_regen_experimental]./1000,'--
g','LineWidth',2)
% plot(theta_deg,[TH200_DATA(87).pdrop_regen_experimental]./1000,'--
r','LineWidth',2)
```

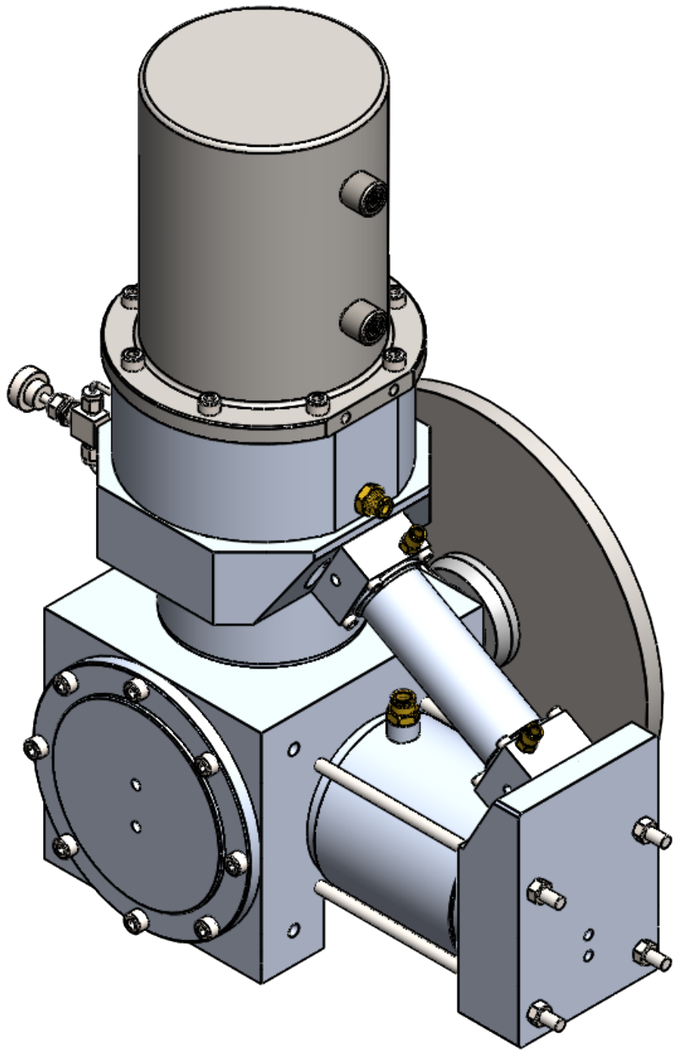
```
%
plot(theta_deg,[TH200_DATA(5).pdrop_regen_model]./1000.*8,'k','LineWidth',2)
%
plot(theta_deg,[TH200_DATA(73).pdrop_regen_model]./1000.*8,'g','LineWidth',2)
%
plot(theta_deg,[TH200_DATA(87).pdrop_regen_model]./1000.*8,'r','LineWidth',2)
% xlabel('Crank Angle (\circ)','FontName',font,'FontSize',font_size)
% ylabel('Regenerator Pressure Drop
(kPa)','FontName',font,'FontSize',font_size)
% xlim([0 360])
% legend('pmean = 707 kPa','pmean = 293 kPa','pmean = 421
kPa','Location','South')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)

% % Regenerator Pressure Drop Amplitude vs. Speed and Pressure
% figure('Position', [x y width height])
% pointsize = 30;
% maxes = max([TH200_DATA.P1_Pa]./1000);
% mins = min([TH200_DATA.P1_Pa]./1000);
% scatter([TH200_DATA.Engine_Hz],(maxes-mins), pointsize,
[TH200_DATA.pmean]./1000)
% ylabel('Regenerator Pressure Drop Amplitude
(kPa)','FontName',font,'FontSize',font_size);
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% c = colorbar;
% ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
% colormap(jet)
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
%
% % Power Lost to Regenerator Flow Friction (Power consumed by displacer)
% figure('Position', [x y width height])
% pointsize = 30;
% scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_lost_regen_FF], pointsize,
[TH200_DATA.pmean]./1000)
% ylabel('Power Lost to Regenerator Flow Friction
(W)','FontName',font,'FontSize',font_size);
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% c = colorbar;
% ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
% colormap(jet)
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
%
% % Power Lost to Regenerator Flow Friction with overlaid model
% figure('Position', [x y width height])
% pointsize = 30;
% hold on
% scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_lost_regen_FF], pointsize,
[TH200_DATA.pmean]./1000,'*')
% scatter([TH200_DATA.Engine_Hz],[TH200_DATA.P_lost_regen_FF_model],
pointsize, [TH200_DATA.pmean]./1000,'o')
% ylabel('Power Lost to Regenerator Flow Friction
(W)','FontName',font,'FontSize',font_size);
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% c = colorbar;
```

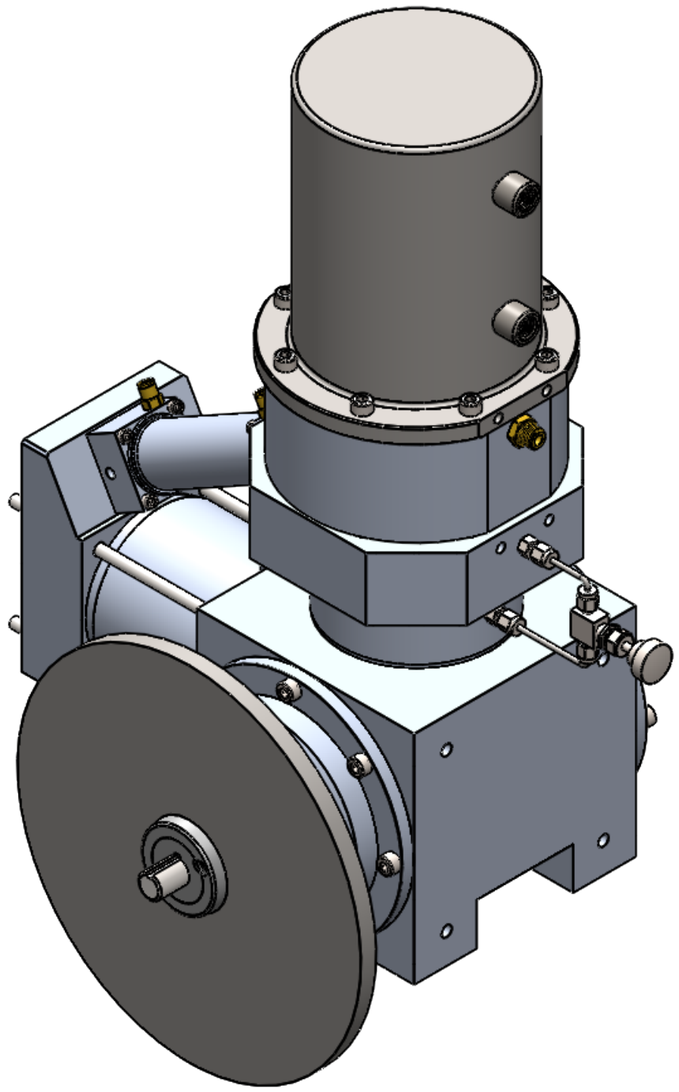
```
% ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
% colormap(jet)
% legend('Experimental Data','Quasi-Steady Model','Location','NorthWest')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)
%
% % Error in Regenerator Pressure Drop Amplitude vs. Speed and Pressure
% figure('Position', [x y width height])
% pointsize = 30;
% exp_maxes = max([TH200_DATA.pdrop_regen_experimental]./1000);
% exp_mins = min([TH200_DATA.pdrop_regen_experimental]./1000);
% exp_amps = (exp_maxes-exp_mins);
% model_maxes = max([TH200_DATA.pdrop_regen_model]./1000);
% model_mins = min([TH200_DATA.pdrop_regen_model]./1000);
% model_amps = (model_maxes-model_mins);
% val_model_maxes = max([TH200_DATA.pdrop_regen_model]./1000.*8);
% val_model_mins = min([TH200_DATA.pdrop_regen_model]./1000.*8);
% val_model_amps = (val_model_maxes-val_model_mins);
% pdropr_error = ((model_amps-exp_amps)./exp_amps).*100;
% val_pdropr_error = ((val_model_amps-exp_amps)./exp_amps).*100;
% hold on
% scatter([TH200_DATA.Engine_Hz],pdropr_error, pointsize,
[TH200_DATA.pmean]./1000,'o')
% scatter([TH200_DATA.Engine_Hz],val_pdropr_error, pointsize,
[TH200_DATA.pmean]./1000,'*')
% ylabel({'Regenerator Pressure Drop';'Amplitude Error
(%)'},'FontName',font,'FontSize',font_size);
% ylim([-91 25])
% xlabel('Engine Frequency (Hz)','FontName',font,'FontSize',font_size)
% c = colorbar;
% ylabel(c,'Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
% colormap(jet)
% legend('Original Model','Model x8')
% set(gca,'fontsize',font_size)
% set(gca,'FontName',font)

% Preferential Flow Evidence
% Data for Plot 3 (60 PSI, 600 RPM)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
filenameT_0 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_0_Temp.log';
filenameP_0 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_0_Volt.log';
[theta_rounded_deg_0,P5_avg_bar_0] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_0,filenameP_0);

filenameT_1 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_1_Temp.log';
filenameP_1 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_1_Volt.log';
```

```
[theta_rounded_deg_1,P5_avg_bar_1] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_1,filenameP_1);

filenameT_2 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_2_Temp.log';
filenameP_2 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_2_Volt.log';
[theta_rounded_deg_2,P5_avg_bar_2] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_2,filenameP_2);

filenameT_3 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_3_Temp.log';
filenameP_3 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_3_Volt.log';
[theta_rounded_deg_3,P5_avg_bar_3] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_3,filenameP_3);

filenameT_4 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_4_Temp.log';
filenameP_4 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_4_Volt.log';
[theta_rounded_deg_4,P5_avg_bar_4] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_4,filenameP_4);

filenameT_5 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_5_Temp.log';
filenameP_5 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_5_Volt.log';
[theta_rounded_deg_5,P5_avg_bar_5] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_5,filenameP_5);

filenameT_6 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_6_Temp.log';
filenameP_6 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_6_Volt.log';
[theta_rounded_deg_6,P5_avg_bar_6] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_6,filenameP_6);

filenameT_7 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_7_Temp.log';
filenameP_7 = 'X:\01_Current_Students\Connor Speer\00_Thesis and Journal
Papers\00_Thesis\00_Data and Plots\Chapter 5 Plots (Model Validation)\Tecknic
Spin Tests\06_HH Rotation Tests\P_60_RPM_600_7_Volt.log';
[theta_rounded_deg_7,P5_avg_bar_7] =
Averaged_Data_Plots_Tecknic_Spin(filenameT_7,filenameP_7);
```

```
% Polar Plot of Heater Pressure Drop Amplitude
figure('Position', [x y width height])
amp_0deg = (max(P5_avg_bar_0) - min(P5_avg_bar_0))*100;
amp_45deg = (max(P5_avg_bar_1) - min(P5_avg_bar_1))*100;
amp_90deg = (max(P5_avg_bar_2) - min(P5_avg_bar_2))*100;
amp_135deg = (max(P5_avg_bar_3) - min(P5_avg_bar_3))*100;
amp_180deg = (max(P5_avg_bar_4) - min(P5_avg_bar_4))*100;
amp_225deg = (max(P5_avg_bar_5) - min(P5_avg_bar_5))*100;
amp_270deg = (max(P5_avg_bar_6) - min(P5_avg_bar_6))*100;
amp_315deg = (max(P5_avg_bar_7) - min(P5_avg_bar_7))*100;
HHangles = [0 45 90 135 180 225 270 315 0];
HHangles = HHangles*(pi/180);
amps = [amp_0deg amp_45deg amp_90deg amp_135deg amp_180deg amp_225deg
amp_270deg amp_315deg amp_0deg];
polarplot(HHangles,amps,'-o','LineWidth',2)
ax = gca;
ax.ThetaTickLabel = {'Power Cylinder',' ',' ',' ','Bypass Line',' ',' ',' '};
ax.ThetaTick = [0 45 90 135 180 225 270 315];

figure('Position', [x y width height])
hold on
plot([TH200_DATA.pmean]./1000,mean([TH200_DATA.T_2]),'ob')
plot([TH200_DATA.pmean]./1000,mean([TH200_DATA.T_3]),'*r')
ylabel({'Temperature (\circC)'},'FontName',font,'FontSize',font_size);
xlabel('Mean Pressure (kPa)','FontName',font,'FontSize',font_size)
legend('Regenerator Cold Left (Bypass Side)','Regenerator Cold Right (Power
Cylinder Side)','Location','NorthWest')
set(gca,'fontsize',font_size)
set(gca,'FontName',font)
hold off
```

Appendix E: Drawing Package for Modified Stirling Engine





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

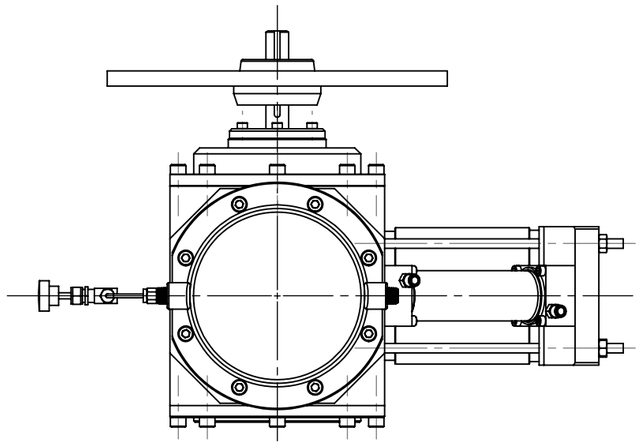
GSE - 1 Overall Assembly

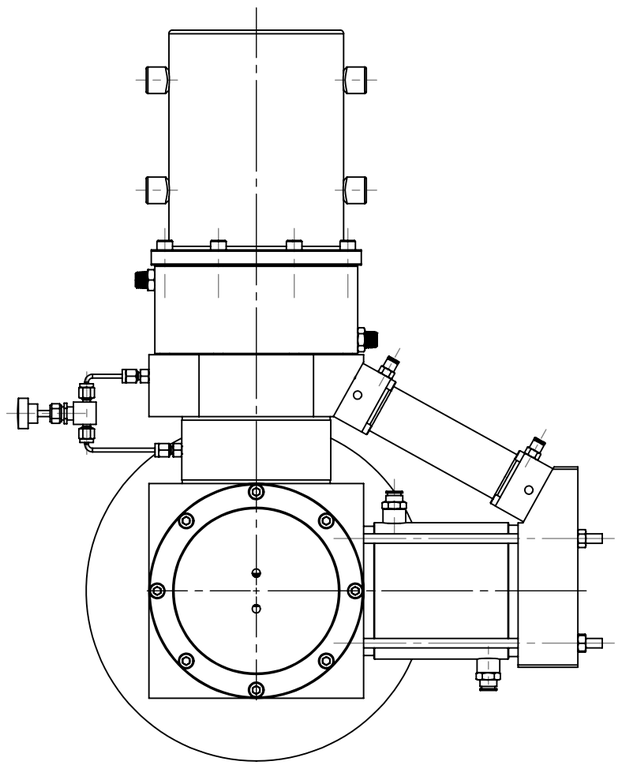
Material: N/A

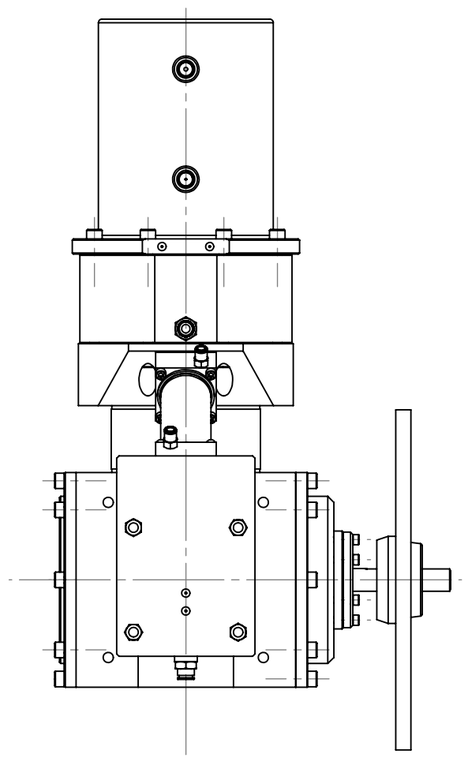
DWG NO. STIRLING_ENGINE_RESEARCH_VERSION

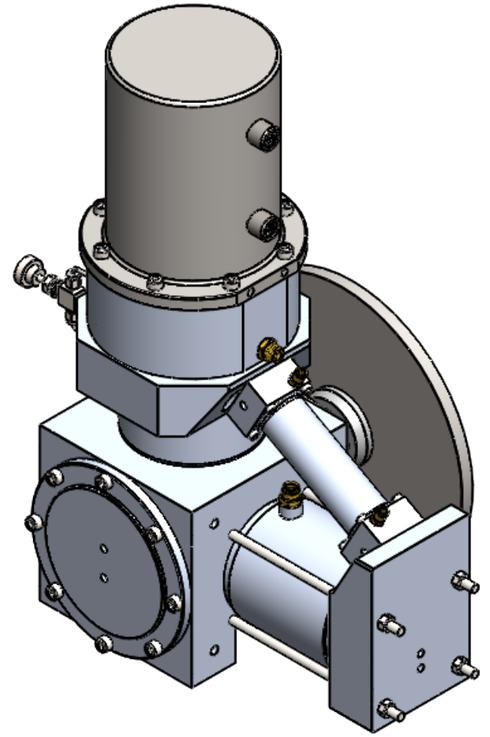
REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: GSE-1 | Mass: | SCALE:1:3.5 | SHEET 1 OF 8









UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

GSE - 1 Overall Assembly

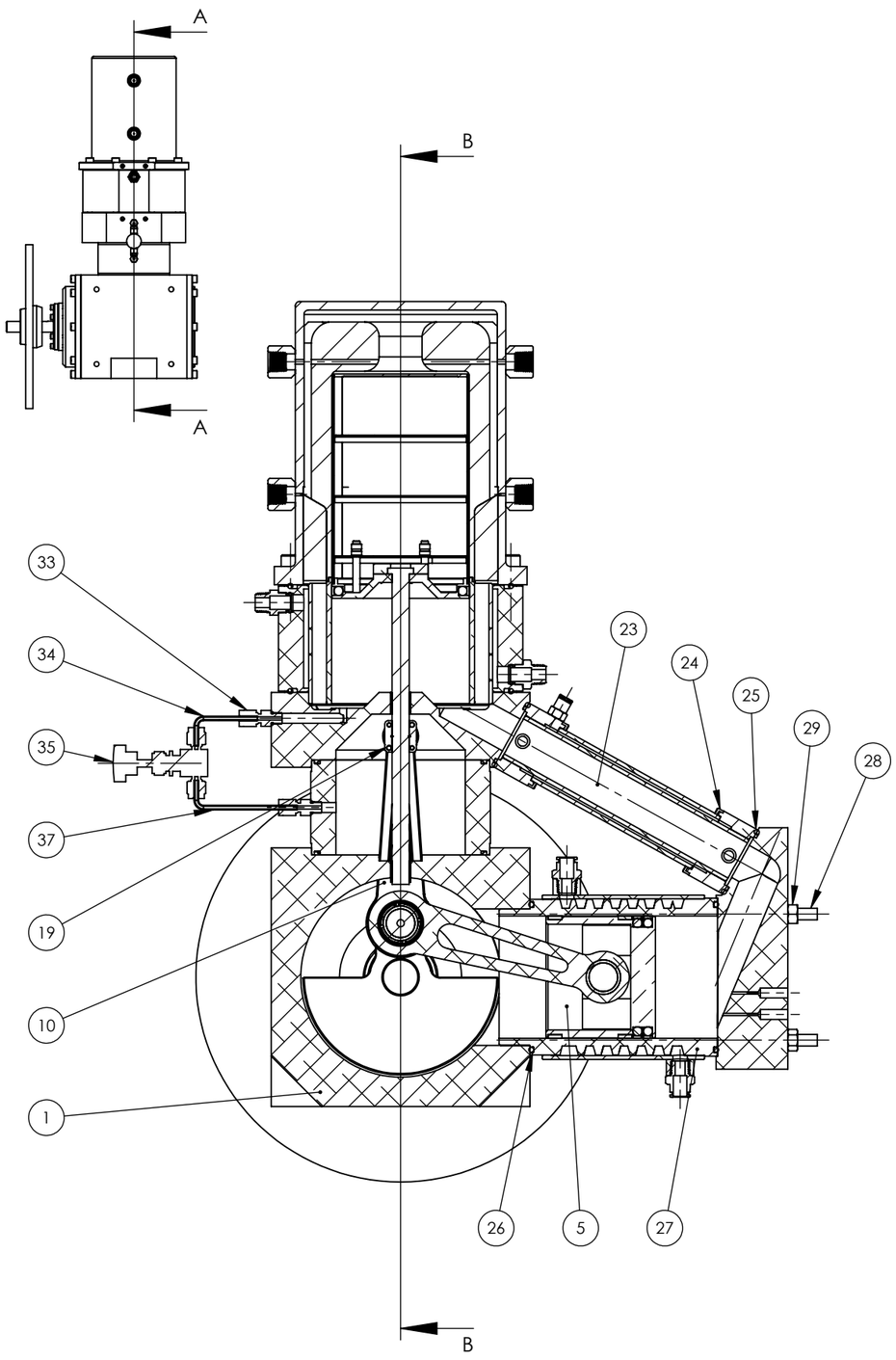
Material: N/A

DWG NO. STIRLING_ENGINE_RESEARCH_VERSION

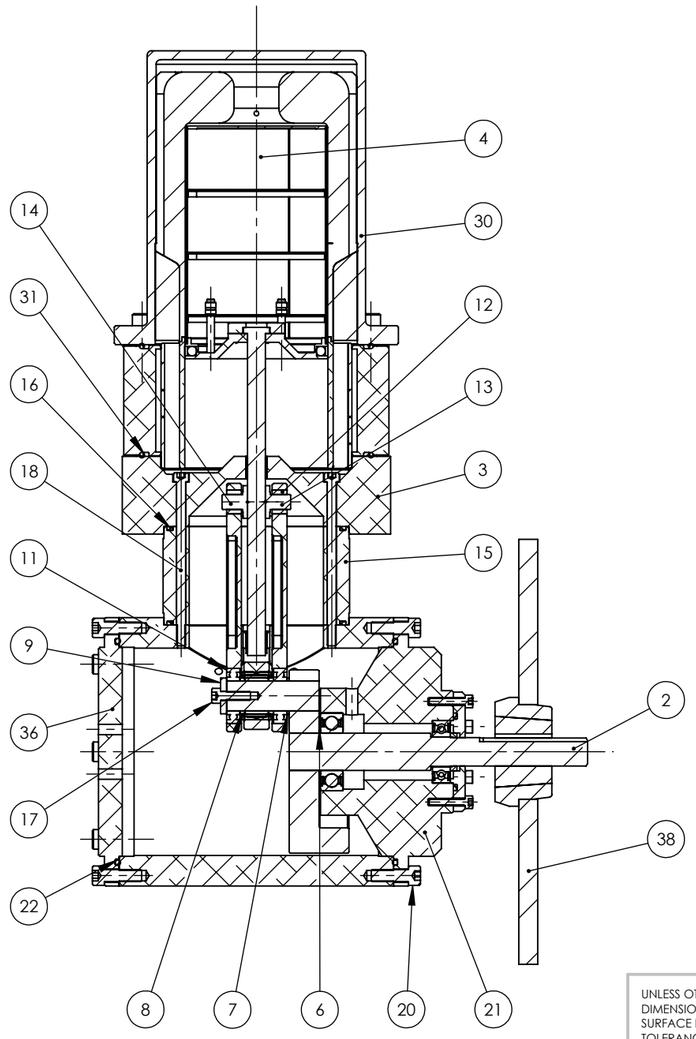
REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: **GSE-1** | Mass: | SCALE:1:5 | SHEET 2 OF 8

No. 32 is not shown on this drawing.
Please refer to ASSEMBLY STEP 4



SECTION A-A
SCALE 1 : 4



SECTION B-B
SCALE 1 : 4

ITEM NO.	PART NUMBER	QTY.
1	C-CC-Z-00-CRANKCASE_ASM	1
2	C-CS-Z-00-CRANKSHAFT_ASM	1
3	A-DM-Z-00-DISP_MOUNT_ASM	1
4	A-DP-F-00-DISP_FULL_ASM	1
5	B-PP-Z-00-POWER_PISTON_ASM	1
6	C-ZZ-Z-02-SPACER_115	1
7	C-ZZ-Z-03-SPACER_116	1
8	C-ZZ-Z-04-SPACER_117	2
9	C-ZZ-Z-05-SPACER_118	1
10	A-ZZ-Z-04-DISP_PISTON_ROD	2
11	AFBMA 20.1 - 19-20 - 14,SI,NC,14_68	2
12	A-ZZ-Z-05-BRONZE_BUSHING_10MM	2
13	A-ZZ-Z-02-R_CROSSHEAD	1
14	A-ZZ-Z-03-L_CROSSHEAD	1
15	A-ZZ-Z-01-FOOT	1
16	O-ring 112x1.8-A-ISO 3601-1	2
17	B18.3.1M - 6 x 1.0 x 20 Hex SHCS -- 20NHX	1
18	B18.3.1M - 6 x 1.0 x 110 Hex SHCS -- 24NHX	6
19	B18.3.1M - 3 x 0.5 x 12 Hex SHCS -- 12NHX	4
20	B18.3.1M - 8 x 1.25 x 25 Hex SHCS -- 25NHX	16
21	C-BC-Z-00-BEARING_CARTRIDGE_ASM	1
22	O-ring DIN 3771 - 145x3.55	2
23	B-CP-Z-00-CNCT_PIPE_ASM	1
24	B18.3.1M - 5 x 0.8 x 35 Hex SHCS -- 22NHX	8
25	O-ring DIN 3771 - 43.7x3.55	2
26	O-ring DIN 3771 - 97.5x3.55	1
27	B-PC-Z-00-POWER_CYL_ASM	1
28	C-ZZ-Z-06-THREADED_ROD	4
29	B18.2.4.1M - Hex nut, Style 1, M8 x 1.25 --D-N	4
30	A-DC-Z-00-DISP-CYL_ASM	1
31	O-ring 150x3.55-A-ISO 3601-1	1
32	B18.3.1M - 8 x 1.25 x 30 Hex SHCS -- 30NHX	8
33	A-ZZ-Z-07-0.125IN_ST_SWAGELOK	2
34	A-ZZ-Z-10-BYPASS_TUBE_2	1
35	A-ZZ-Z-06-0.125IN_VALVE	1
36	C-ZZ-Z-01-COVER	1
37	A-ZZ-Z-09-BYPASS_TUBE_1	1
38	C-ZZ-F-00-FLYWHEEL_ASM	1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

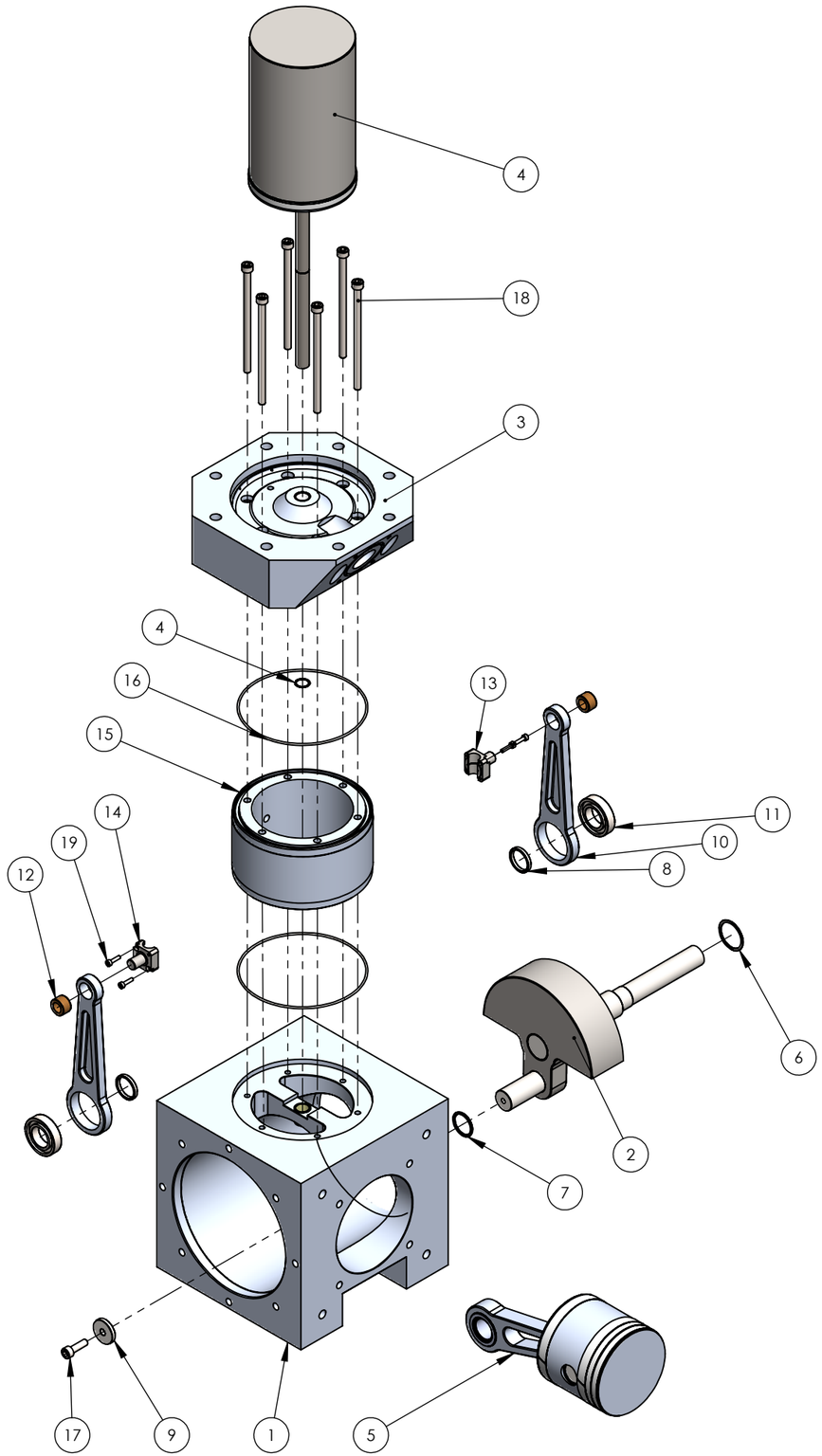
GSE - 1 Overall Assembly

Material:

DWG NO. **STIRLING_ENGINE_RESEARCH_VERSION**

REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: **GSE-1** | SCALE:1:5 | SHEET 3 OF 8



ITEM NO.	DRW NUMBER	Assembly Step 1/QTY.
1	C-CC-Z-00-CRANKCASE_ASM	1
2	C-CS-Z-00-CRANKSHAFT_ASM	1
3	A-DM-Z-00-DISP_MOUNT_ASM	1
4	A-DP-F-00-DISP_FULL_ASM	1
5	B-PP-Z-00-POWER_PISTON_ASM	1
6	C-ZZ-Z-02-SPACER_115	1
7	C-ZZ-Z-03-SPACER_116	1
8	C-ZZ-Z-04-SPACER_117	2
9	C-ZZ-Z-05-SPACER_118	1
10	A-ZZ-Z-04-DISP_PISTON_ROD	2
11	AFBMA 20.1 - 19-20 - 14,SI,NC,14_68	2
12	A-ZZ-Z-05-BRONZE_BUSHING_10MM	2
13	A-ZZ-Z-02-R_CROSSHEAD	1
14	A-ZZ-Z-03-L_CROSSHEAD	1
15	A-ZZ-Z-01-FOOT	1
16	O-ring 112x1.8-A-ISO 3601-1	2
17	B18.3.1M - 6 x 1.0 x 20 Hex SHCS -- 20NHX	1
18	B18.3.1M - 6 x 1.0 x 110 Hex SHCS -- 24NHX	6
19	B18.3.1M - 3 x 0.5 x 12 Hex SHCS -- 12NHX	4

ASSEMBLY STEP 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

GSE - 1 Overall Assembly

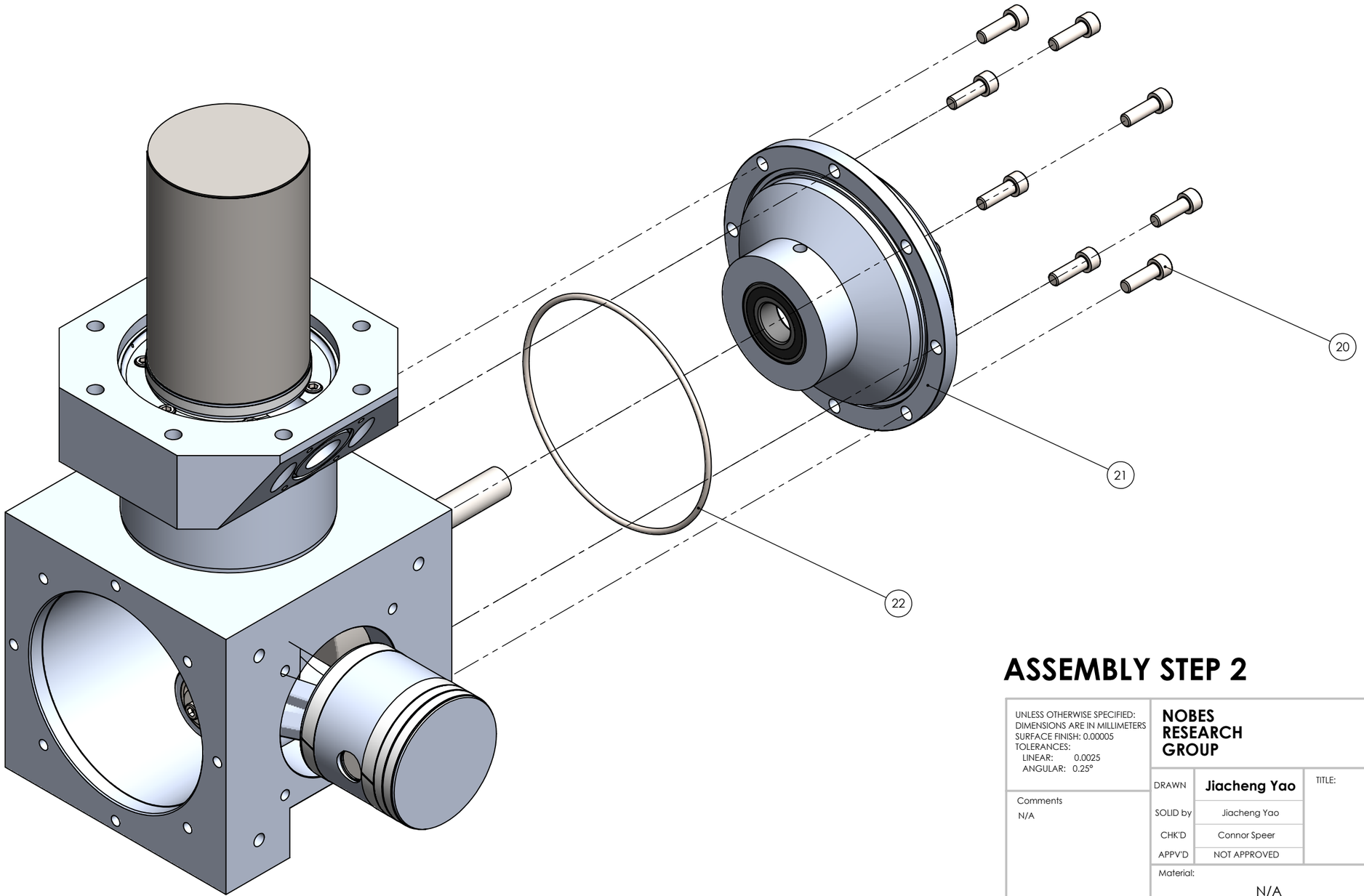
Material: N/A

DWG NO. STIRLING_ENGINE_RESEARCH_VERSION

REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: **GSE-1** | Mass: | SCALE:1:5 | SHEET 4 OF 8

ITEM NO.	DRW NUMBER	Assembly Step 2/QTY.
20	B18.3.1M - 8 x 1.25 x 25 Hex SHCS -- 25NHX	8
21	C-BC-Z-00-BEARING_CARTRIDGE_ASM	1
22	O-ring DIN 3771 - 145x3.55	1



ASSEMBLY STEP 2

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

GSE - 1 Overall Assembly

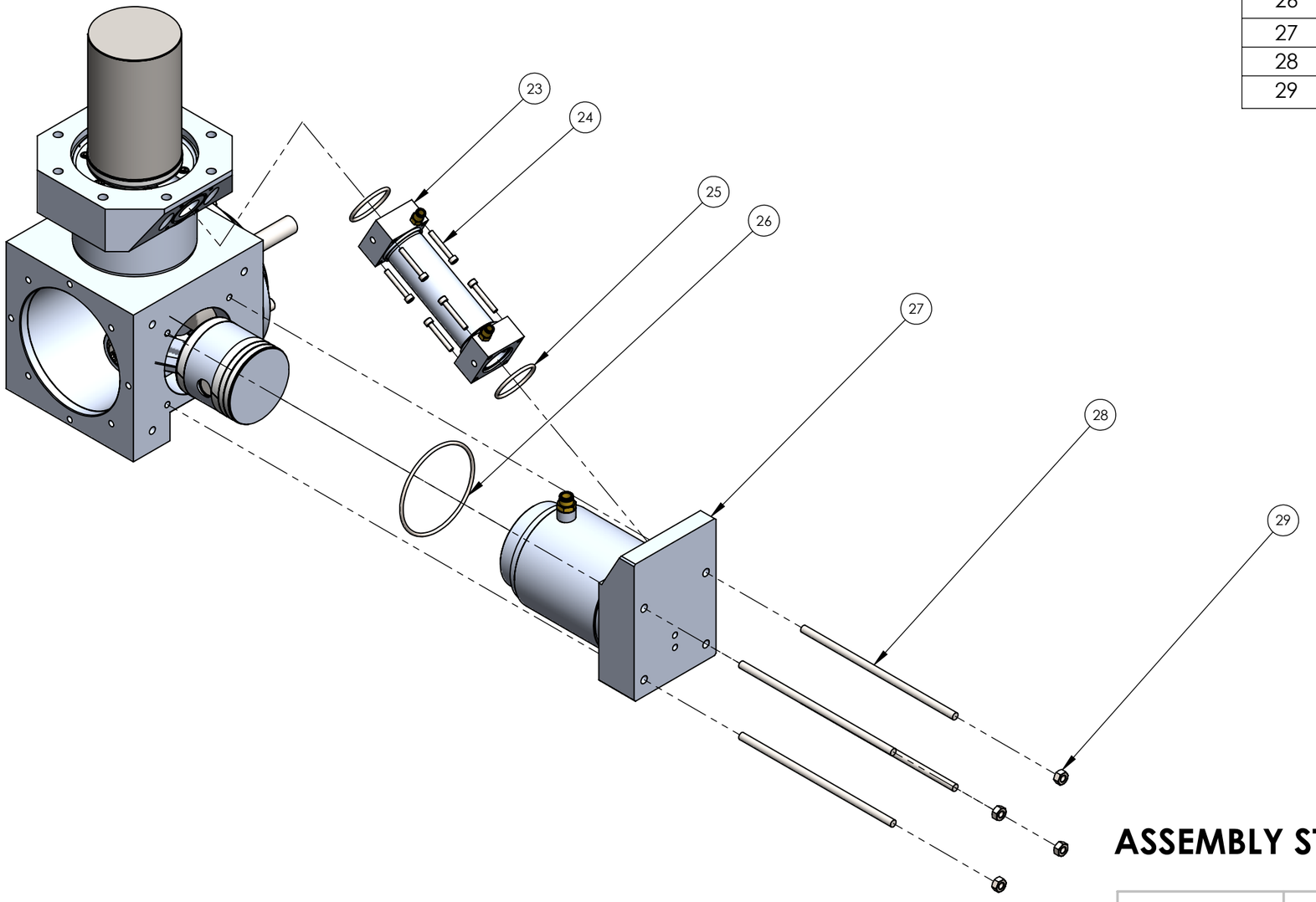
Material: N/A

DWG NO. **STIRLING_ENGINE_RESEARCH_VERSION**

REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: **GSE-1** | Mass: | SCALE:2:5 | SHEET 5 OF 8

ITEM NO.	DRW NUMBER	Assembly Step 3/QTY.
23	B-CP-Z-00-CNCT_PIPE_ASM	1
24	B18.3.1M - 5 x 0.8 x 35 Hex SHCS -- 22NHX	8
25	O-ring DIN 3771 - 43.7x3.55	2
26	O-ring DIN 3771 - 97.5x3.55	1
27	B-PC-Z-00-POWER_CYL_ASM	1
28	C-ZZ-Z-06-THREADED_ROD	4
29	B18.2.4.1M - Hex nut, Style 1, M8 x 1.25 --D-N	4



ASSEMBLY STEP 3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

GSE - 1 Overall Assembly

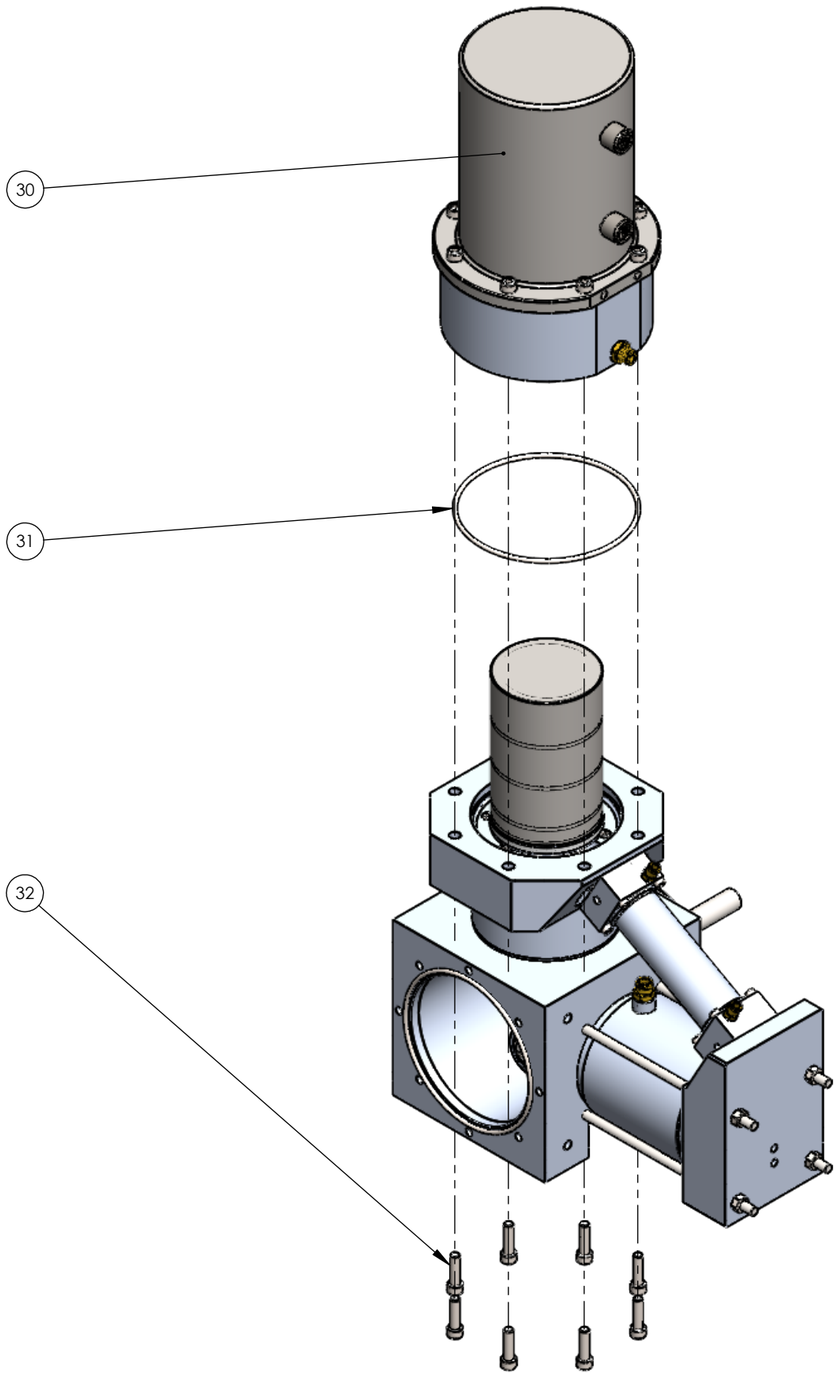
Material: N/A

DWG NO. STIRLING_ENGINE_RESEARCH_VERSION

REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: GSE-1 | Mass: | SCALE:1:5 | SHEET 6 OF 8

ITEM NO.	PART NUMBER	Assembly Step 4/QTY.
30	A-DC-Z-00-DISP-CYL_ASM	1
31	O-ring 150x3.55-A-ISO 3601-1	1
32	B18.3.1M - 8 x 1.25 x 30 Hex SHCS -- 30NHX	8



ASSEMBLY STEP 4

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

GSE - 1 Overall Assembly

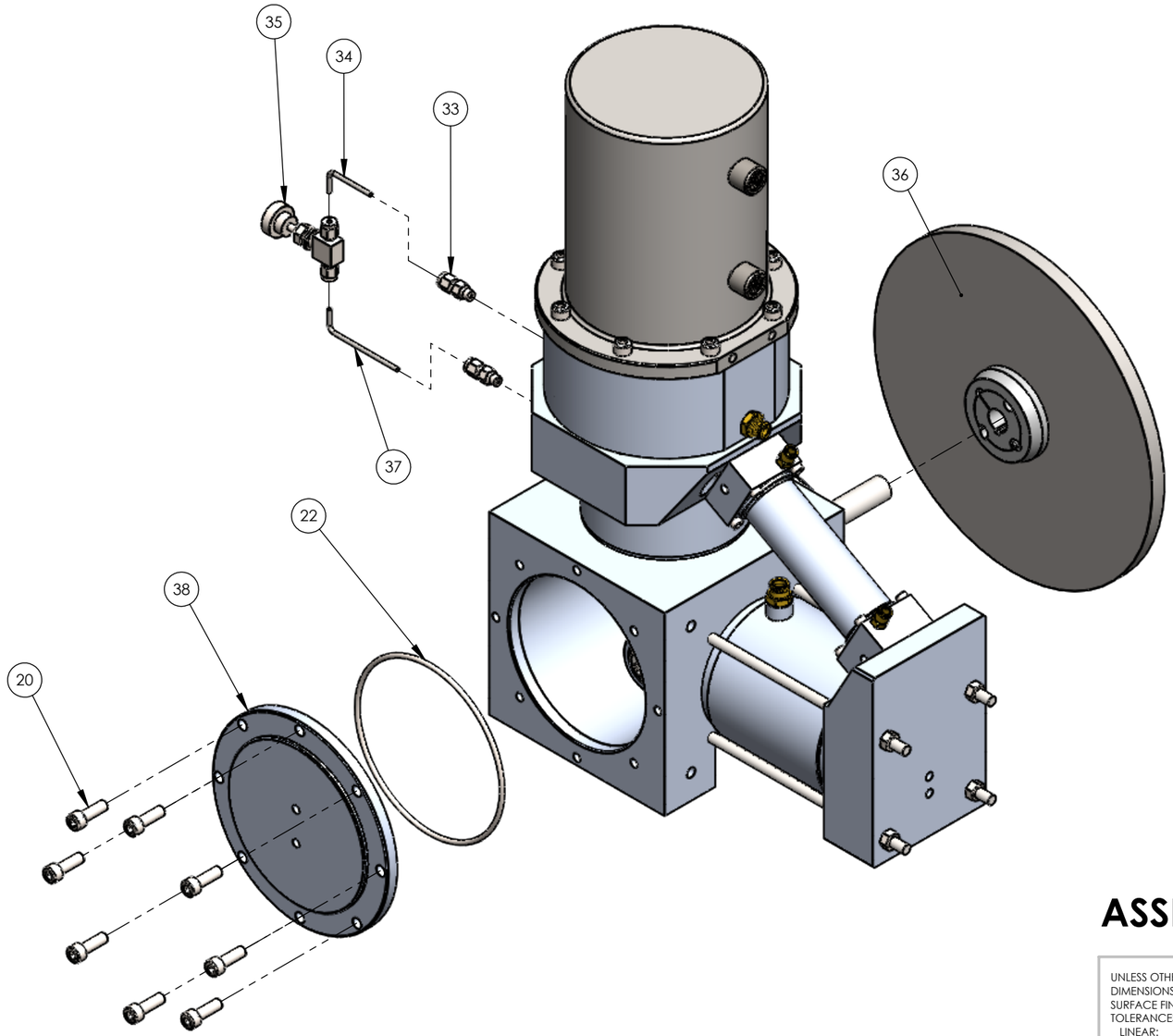
Material: N/A

DWG NO. **STIRLING_ENGINE_RESEARCH_VERSION**

REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: **GSE-1** | Mass: | SCALE:1:5 | SHEET 7 OF 8

ITEM NO.	PART NUMBER	Assembly Step 5/QTY.
20	B18.3.1M - 8 x 1.25 x 25 Hex SHCS -- 25NHX	16
22	O-ring DIN 3771 - 145x3.55	2
33	A-ZZ-Z-07-0.125IN_ST_SWAGELOK	2
34	A-ZZ-Z-10-BYPASS_TUBE_2	1
35	A-ZZ-Z-06-0.125IN_VALVE	1
36	C-ZZ-F-00-FLYWHEEL_ASM	1
37	A-ZZ-Z-09-BYPASS_TUBE_1	1
38	C-ZZ-Z-01-COVER	1



ASSEMBLY STEP 5

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
N/A

Monday, April 18, 2016 9:48:16 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

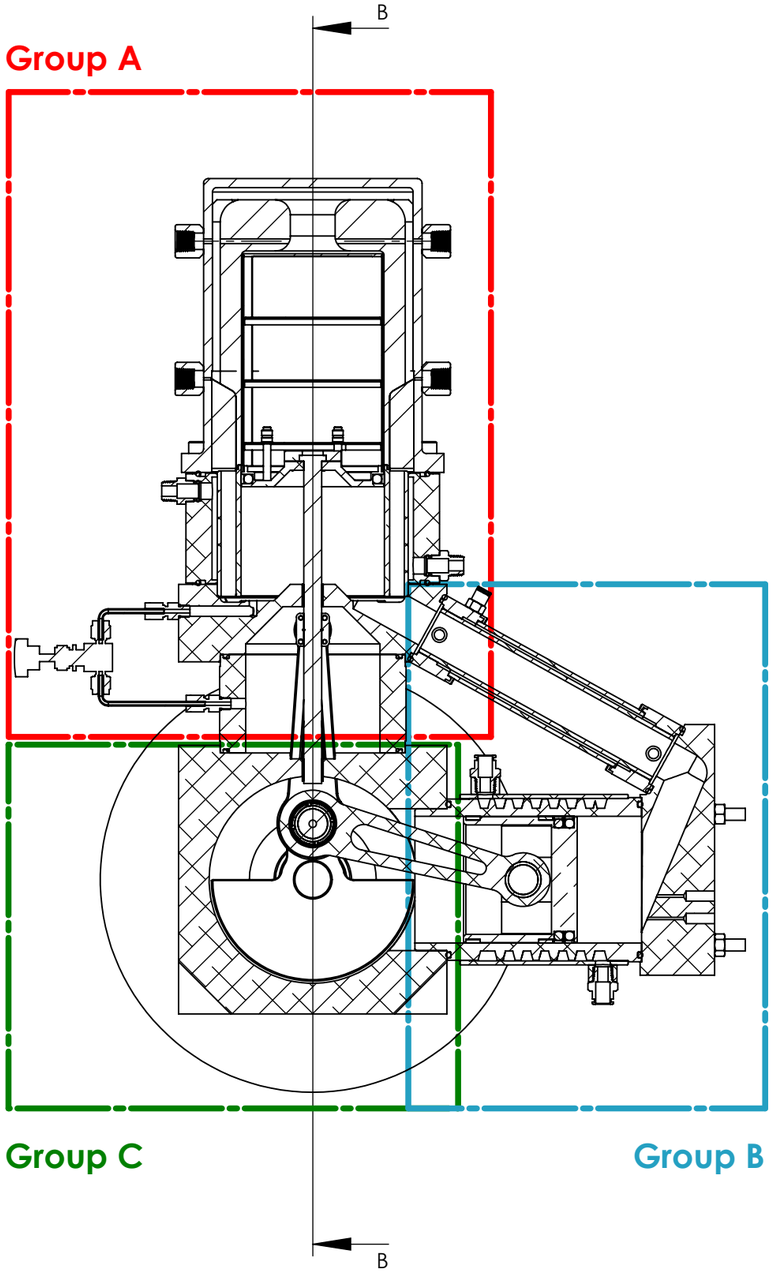
GSE - 1 Overall Assembly

Material: N/A

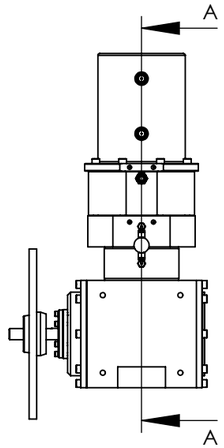
DWG NO. **STIRLING_ENGINE_RESEARCH_VERSION**

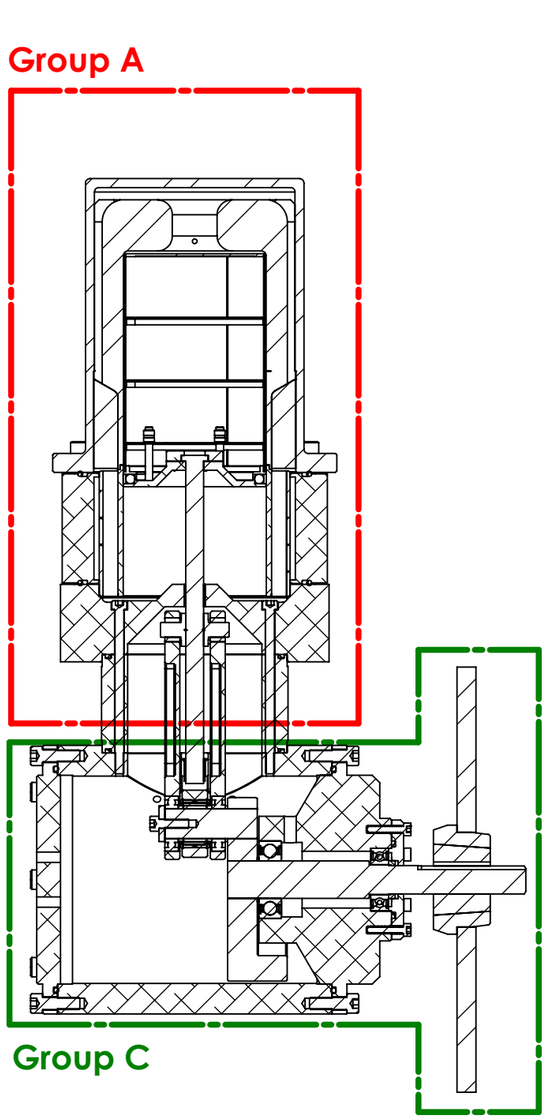
REVISION A

DRW File: 000_STIRLING_ENGINE_RESEARCH_VERSION | Project: **GSE-1** | Mass: | SCALE:1:4 | SHEET 8 OF 8

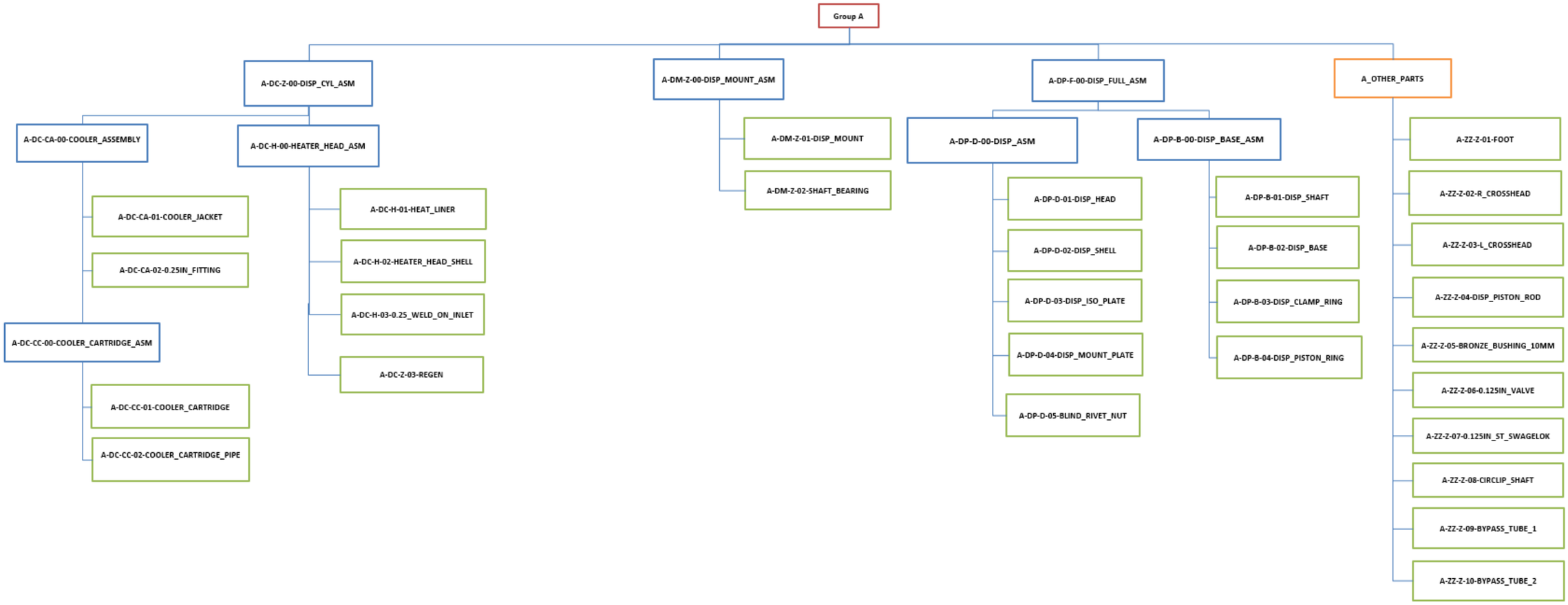


SECTION A-A
SCALE 1 : 4

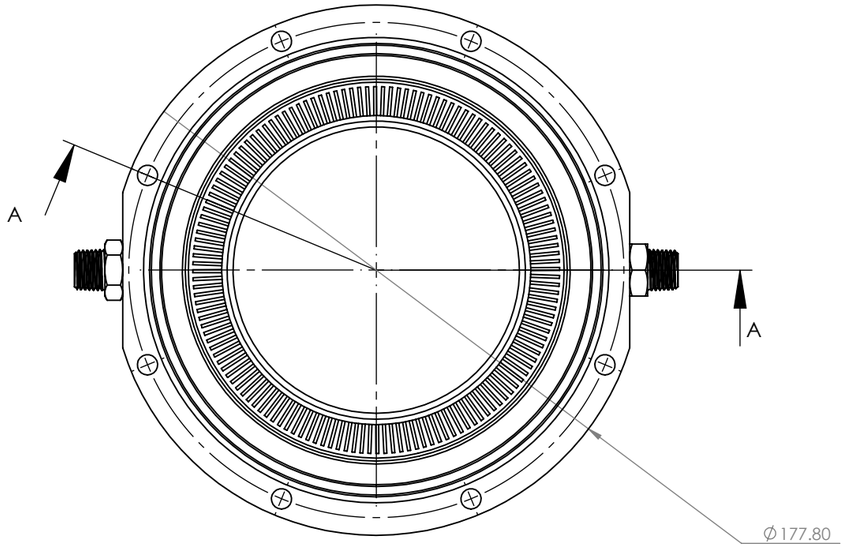


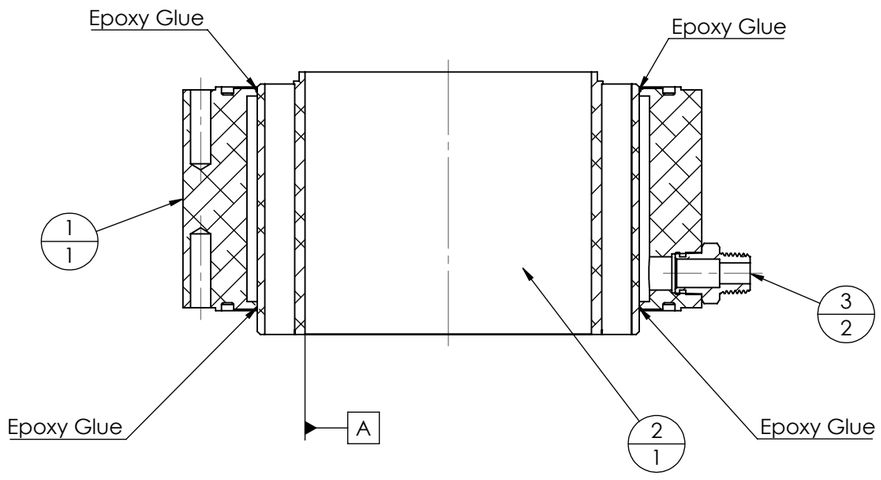


SECTION B-B
SCALE 1 : 4

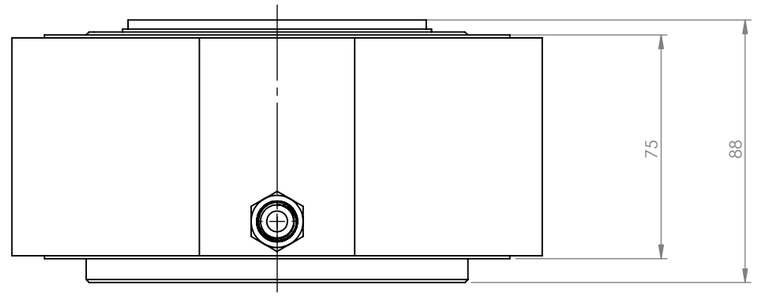

Group A
A-DC-Z-00-DISP_CYL_ASM
A-DC-CA-00-COOLER_ASSEMBLY
A-DC-CA-01-COOLER_JACKET
A-DC-CA-02-0.25IN_FITTING
A-DC-CC-00-COOLER_CARTRIDGE_ASM
A-DC-CC-01-COOLER_CARTRIDGE
A-DC-CC-02-COOLER_CARTRIDGE_PIPE
A-DC-H-00-HEATER_HEAD_ASM
A-DC-H-01-HEAT_LINER
A-DC-H-02-HEATER_HEAD_SHELL
A-DC-H-03-0.25_WELD_ON_INLET
A-DC-Z-03-REGEN
A-DM-Z-00-DISP_MOUNT_ASM
A-DM-Z-01-DISP_MOUNT
A-DM-Z-02-SHAFT_BEARING
A-DP-F-00-DISP_FULL_ASM
A-DP-D-00-DISP_ASM
A-DP-D-01-DISP_HEAD
A-DP-D-02-DISP_SHELL
A-DP-D-03-DISP_ISO_PLATE
A-DP-D-04-DISP_MOUNT_PLATE
A-DP-D-05-BLIND_RIVET_NUT
A-DP-B-00-DISP_BASE_ASM
A-DP-B-01-DISP_SHAFT
A-DP-B-02-DISP_BASE
A-DP-B-03-DISP_CLAMP_RING
A-DP-B-04-DISP_PISTON_RING
A_OTHER_PARTS
A-ZZ-Z-01-FOOT
A-ZZ-Z-02-R_CROSSHEAD
A-ZZ-Z-03-L_CROSSHEAD
A-ZZ-Z-04-DISP_PISTON_ROD
A-ZZ-Z-05-BRONZE_BUSHING_10MM
A-ZZ-Z-06-0.125IN_VALVE
A-ZZ-Z-07-0.125IN_ST_SWAGELOK
A-ZZ-Z-08-CIRCLIP_SHAFT
A-ZZ-Z-09-BYPASS_TUBE_1
A-ZZ-Z-10-BYPASS_TUBE_2

ITEM NO.	DRW NUMBER	QTY.
1	A-DC-CA-01-COOLER_JACKET	1
2	A-DC-CC-00-COOLER_CARTRIDGE_ASM	1
3	B-PC-Z-02-0.25IN_FITTING	2





SECTION A-A



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 18, 2016 9:33:36 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Cooler Assembly

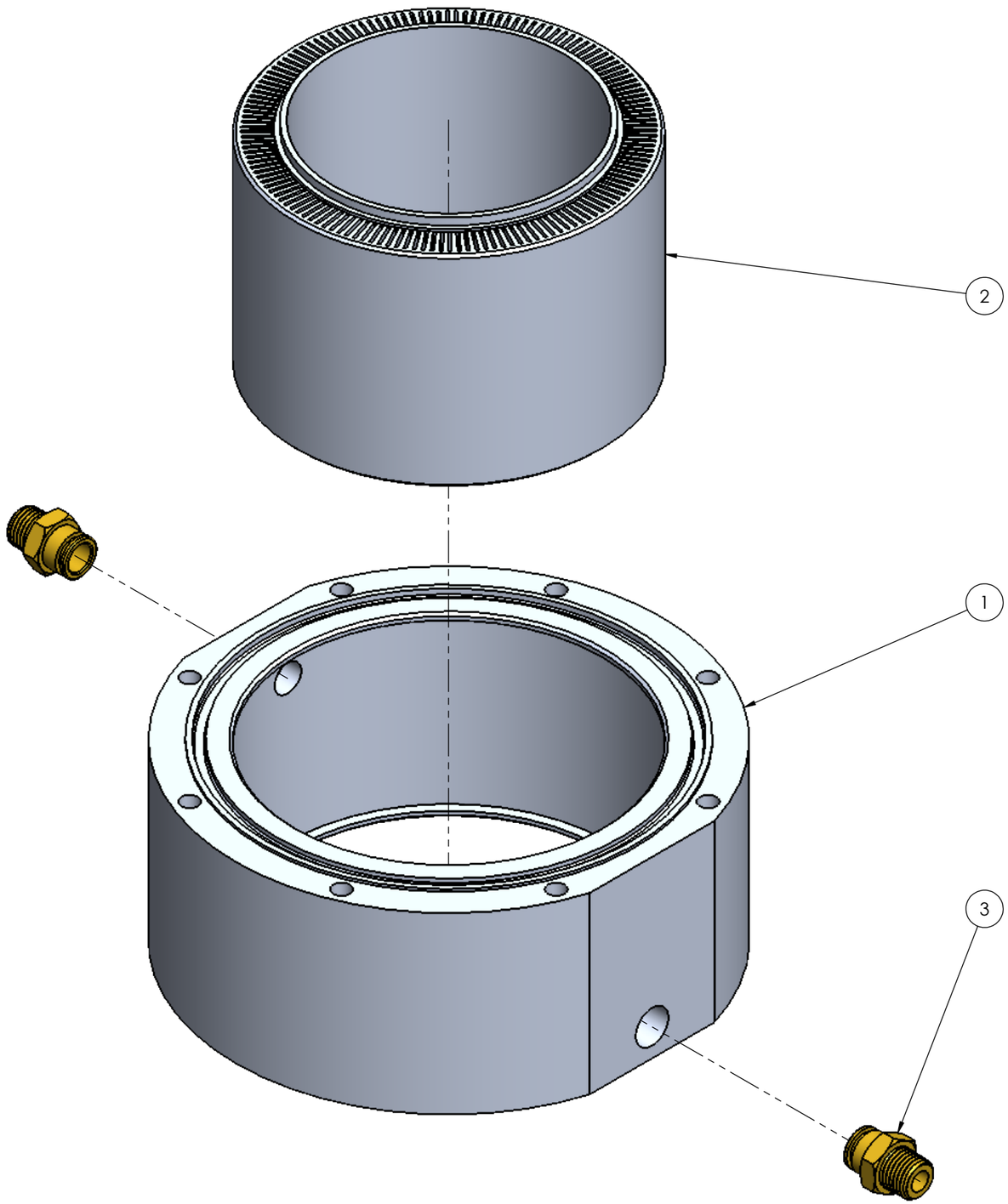
Material: N/A

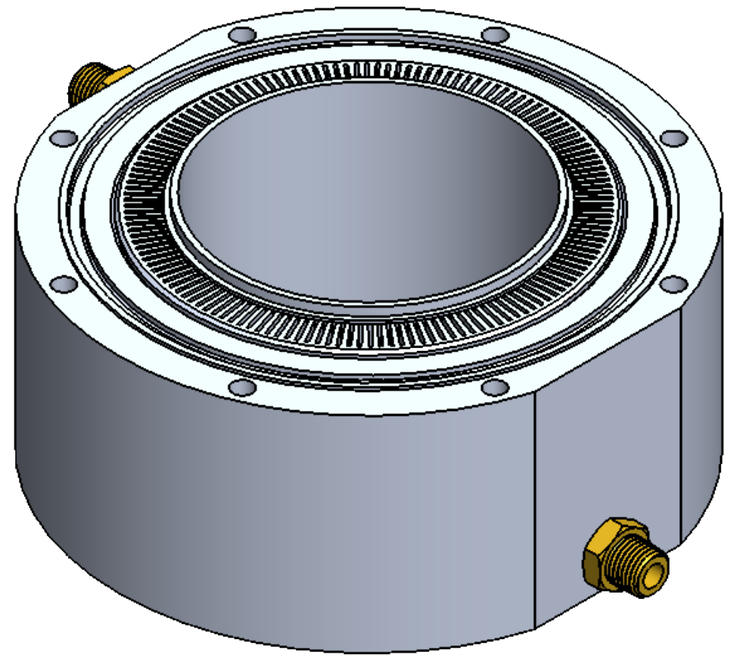
DWG NO. **A-DC-CA-00-COOLER_ASSEMBLY**

REVISION A

DRW File: 008_A-DC-CA-00-COOLER_ASSEMBLY | Project: **GSE-1** | Mass: | SCALE:1:2 | SHEET 1 OF 2

ITEM NO.	DRW NUMBER	QTY.
1	A-DC-CA-01-COOLER_JACKET	1
2	A-DC-CC-00-COOLER_CARTRIDGE_ASM	1
3	B-PC-Z-02-0.25IN_FITTING	2





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 18, 2016 9:33:36 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Cooler Assembly

Material: N/A

DWG NO. **A-DC-CA-00-COOLER_ASSEMBLY**

REVISION A

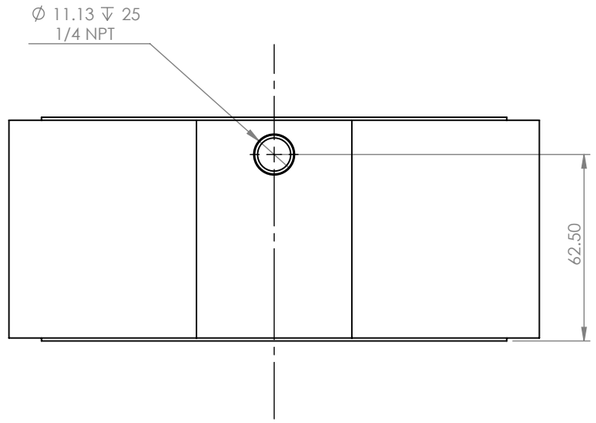
DRW File: 008_A-DC-CA-00-COOLER_ASSEMBLY

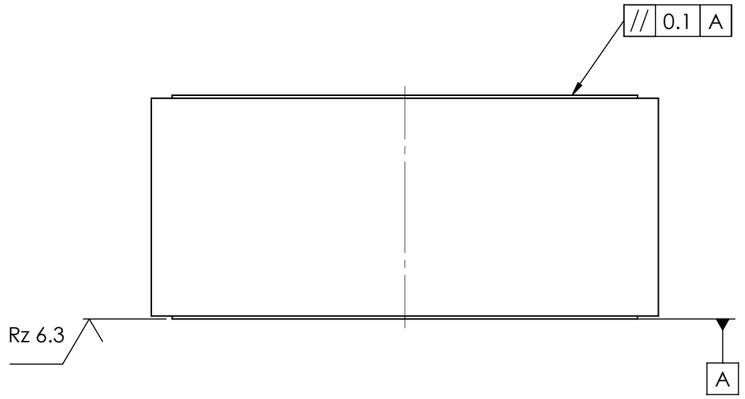
Project: **GSE-1**

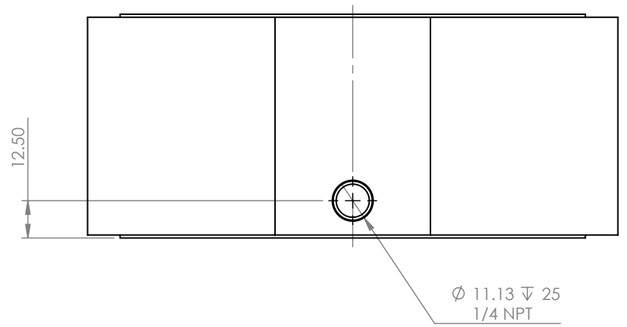
Mass:

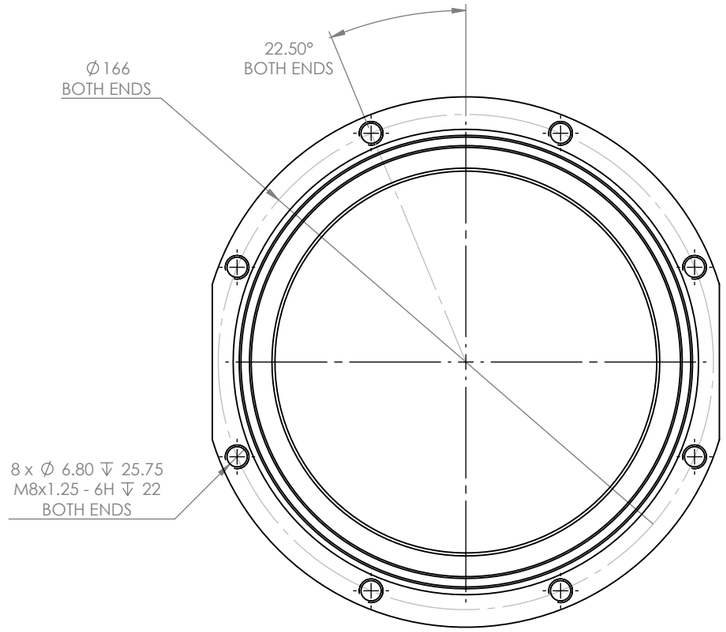
SCALE:2:3

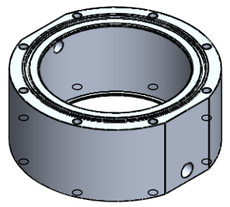
SHEET 2 OF 2











SCALE 1 : 5

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Material Supplier

McMaster-Carr
7392T46

Quantity: 1

Friday, April 15, 2016 1:39:42 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

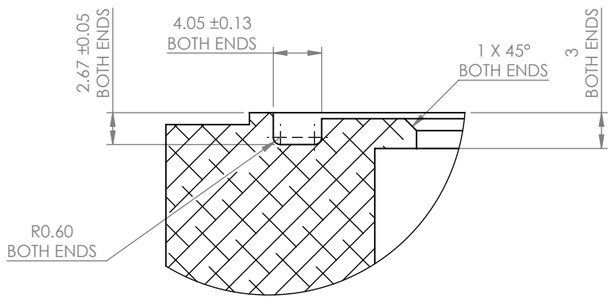
Cooling Jacket

Material: 6061-T6 (SS)

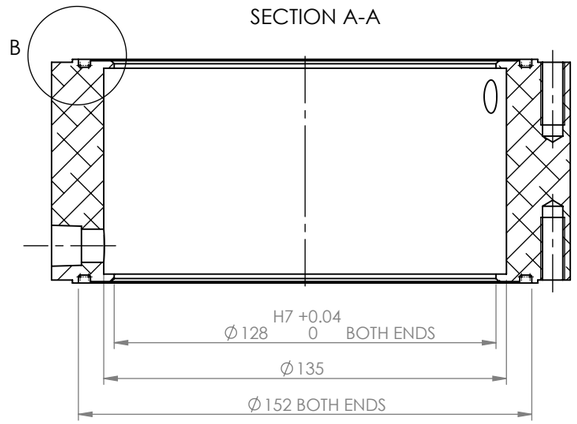
DWG NO. A-DC-CA-01-COOLER_JACKET

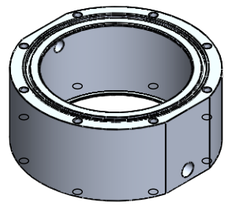
REVISION A

DRW File: 008_A-DC-CA-01-COOLER_JACKET | Project: GSE-1 | Mass: 1979.66301 | SCALE:1:2 | SHEET 1 OF 2



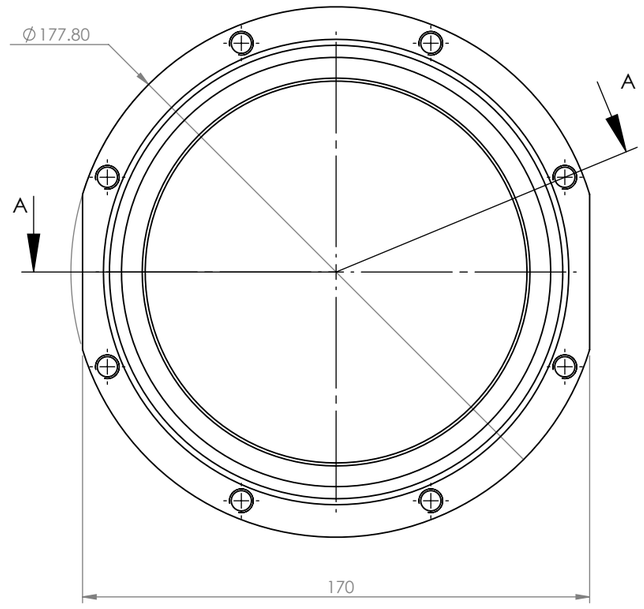
DETAIL B
SCALE 2 : 1

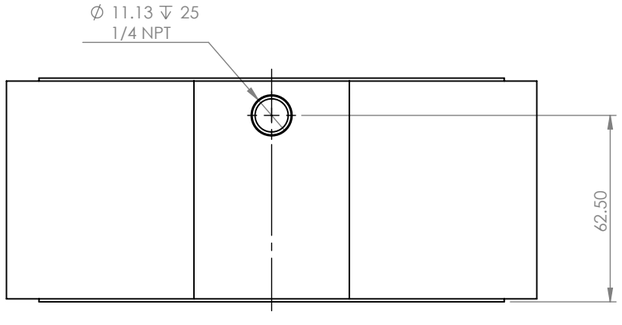


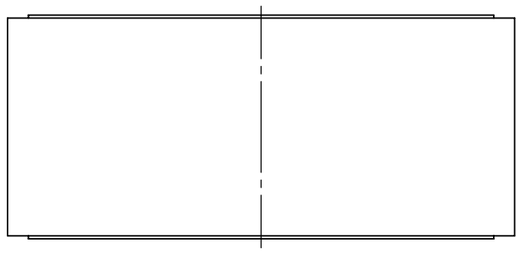


SCALE 1 : 5

O-Ring Information
Size: 145 x 3.5mm
Supplier: Hi-Tech Seals
Part #: 35145
Material: 1, Viton and 1, Nitrile







Please fillet/chamfer all edges to R0.3

General surface finish: Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Material Supplier

McMaster-Carr
7392T46

Quantity: 1

Friday, April 15, 2016 1:39:42 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE: **Cooling Jacket**

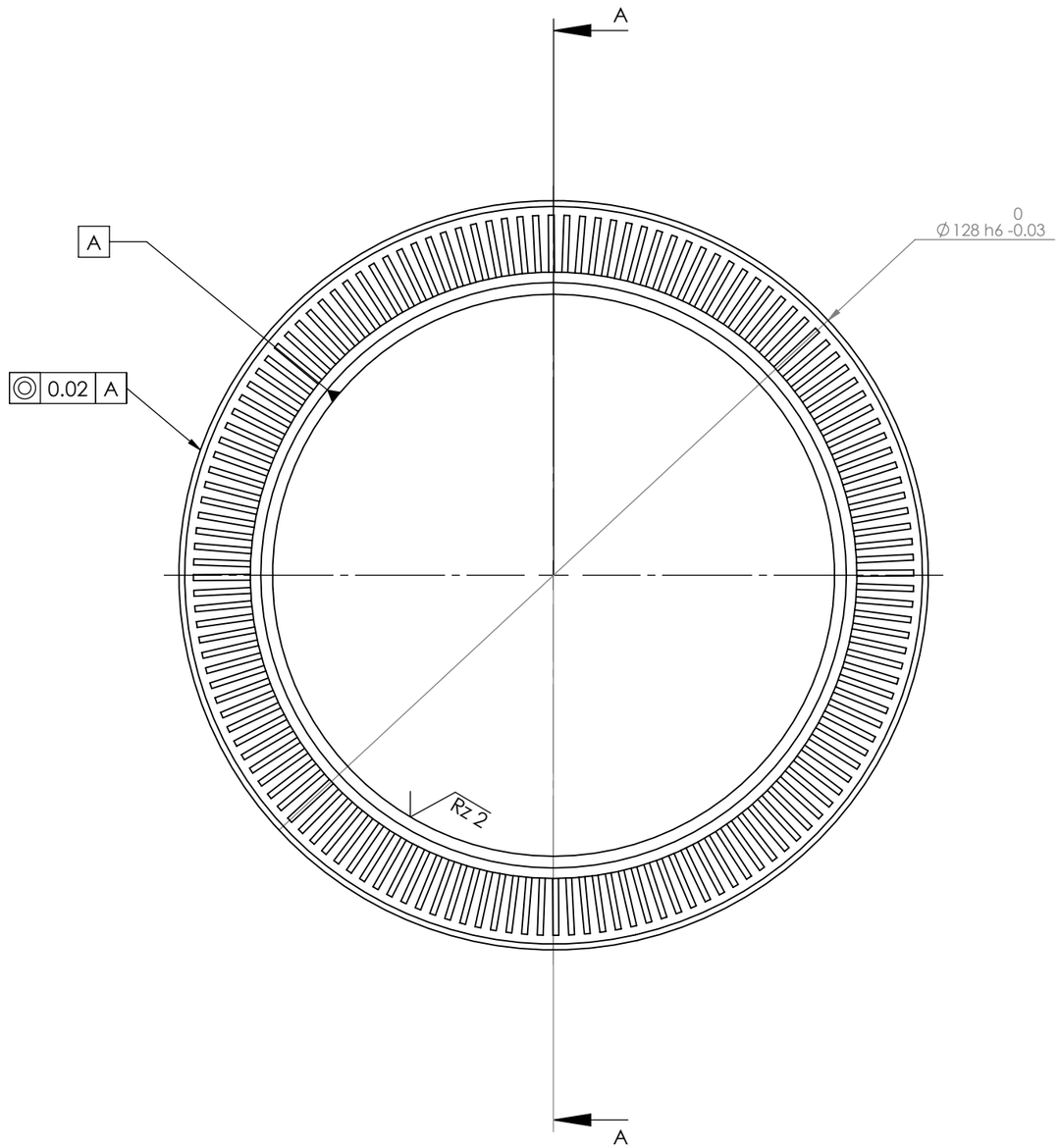
Material: 6061-T6 (SS)

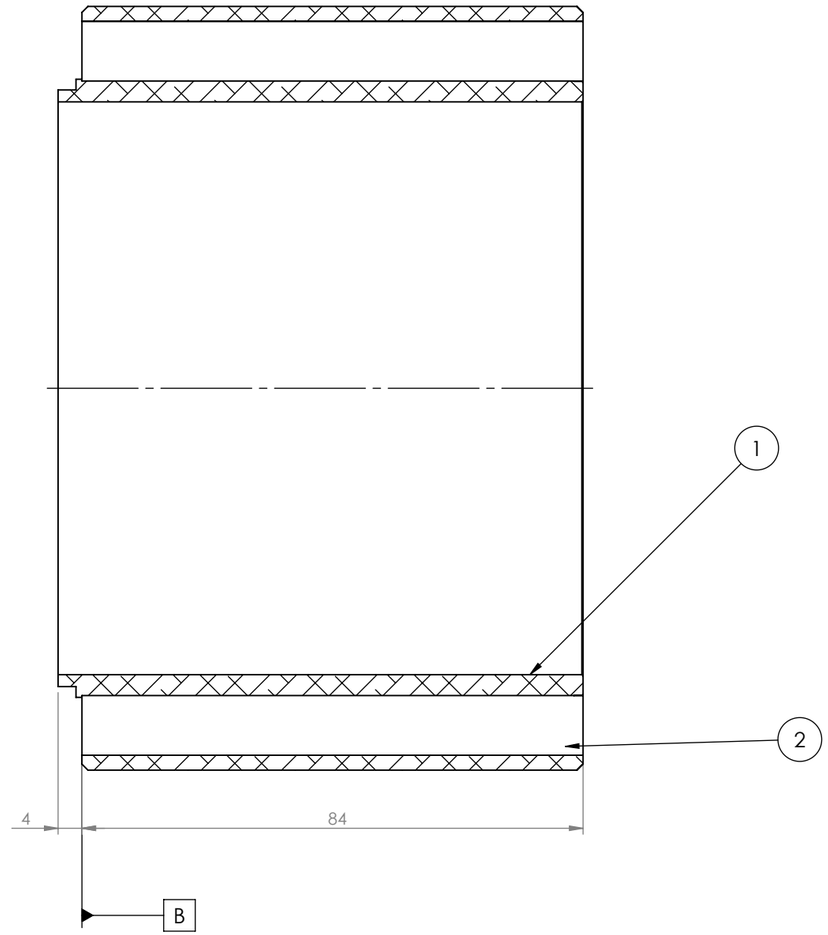
DWG NO. **A-DC-CA-01-COOLER_JACKET**

REVISION A

DRW File: 008_A-DC-CA-01-COOLER_JACKET | Project: **GSE-1** | Mass: 1979.66301 | SCALE:1:2 | SHEET 2 OF 2

ITEM NO.	DRW NUMBER	Material	QTY.
1	A-DC-CC-02-COOLER_CARTRIDGE_PIPE	6061-T6 (SS)	1
2	A-DC-CC-01-COOLER_CARTRIDGE	6061 Alloy	1





SECTION A-A

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Sunday, April 10, 2016 5:27:08 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

Cooler Cartridge Assembly

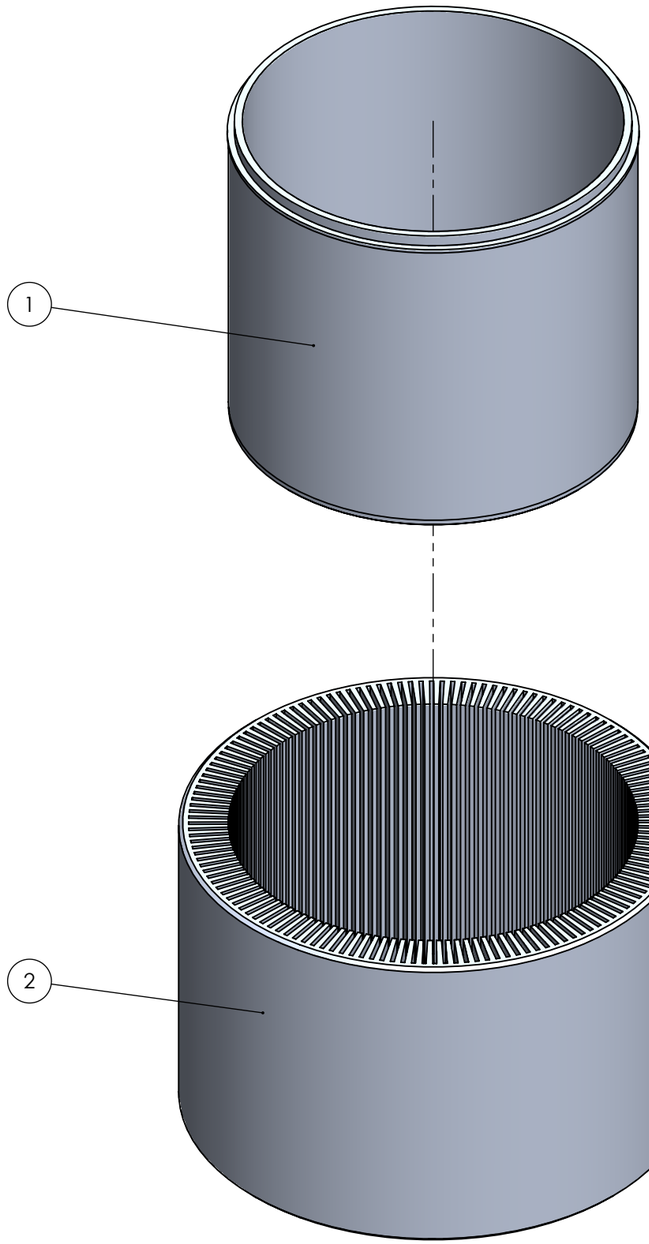
Material: N/A

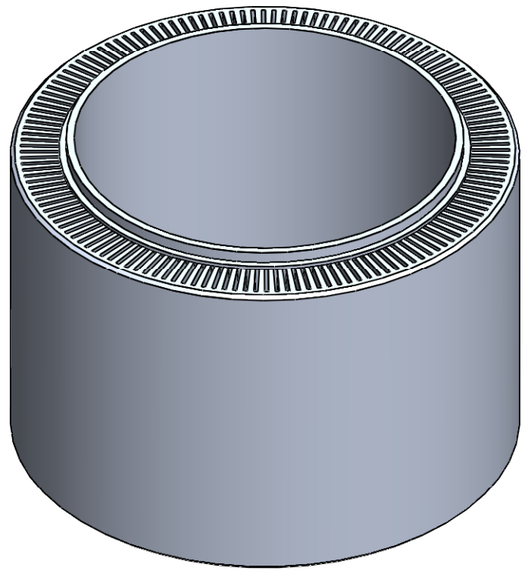
DWG NO. **A-DC-CC-00-COOLER_CARTRIDGE_ASM**

REVISION A

DRW File: 008_A-DC-CC-00-COOLER_CARTRIDGE_ASM | Project: **GSE-1** | Mass: | SCALE:1:1 | SHEET 1 OF 2

ITEM NO.	DRW NUMBER	Material	QTY.
1	A-DC-CC-02-COOLER_CARTRIDGE_PIPE	6061-T6 (SS)	1
2	A-DC-CC-01-COOLER_CARTRIDGE	6061 Alloy	1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Sunday, April 10, 2016 5:27:08 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

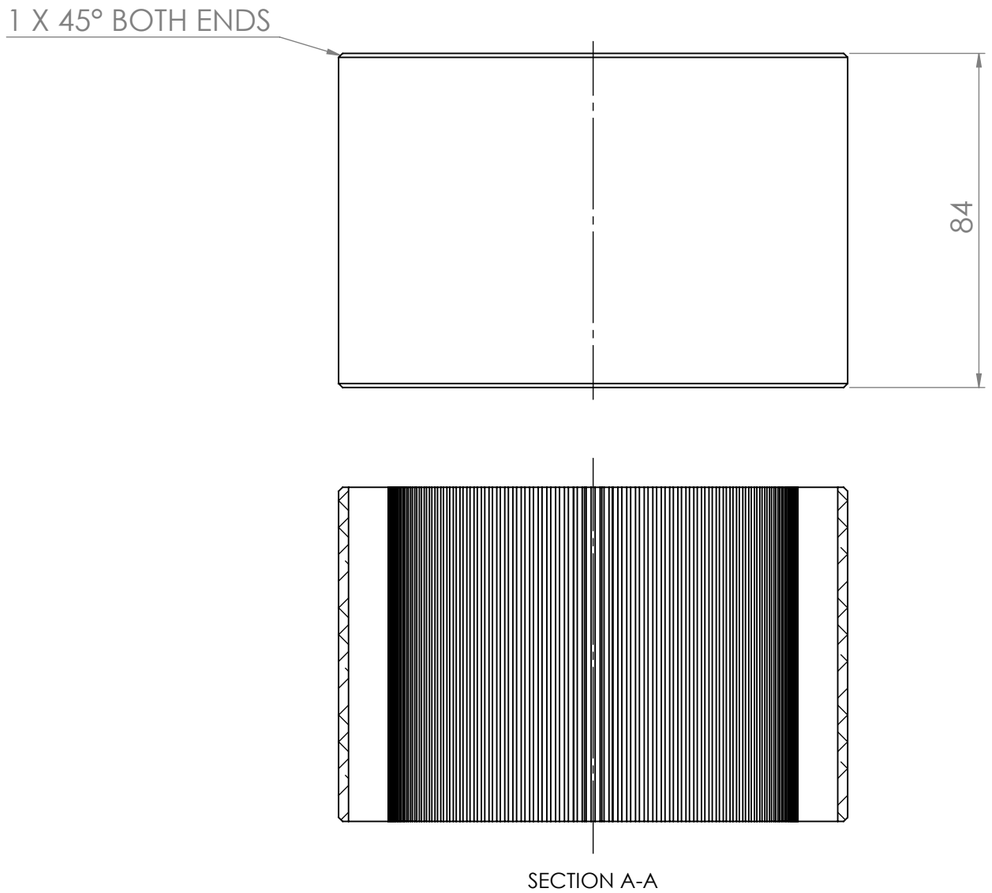
Cooler Cartridge Assembly

Material: N/A

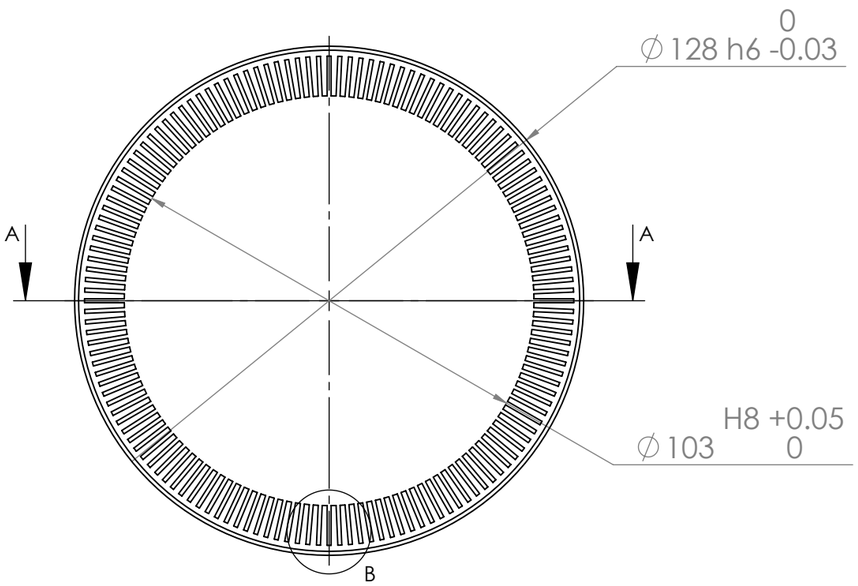
DWG NO. **A-DC-CC-00-COOLER_CARTRIDGE_ASM**

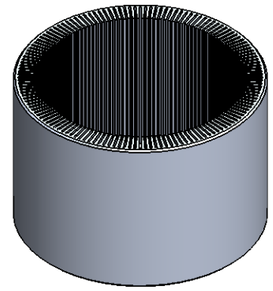
REVISION A

DRW File: 008_A-DC-CC-00-COOLER_CARTRIDGE_ASM | Project: **GSE-1** | Mass: | SCALE:1:1 | SHEET 2 OF 2



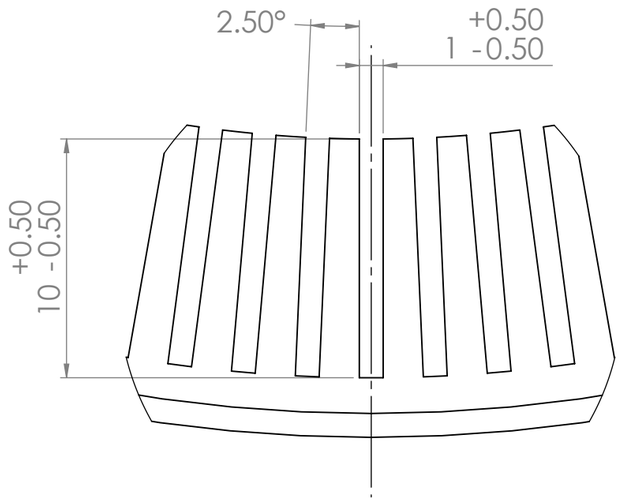
SECTION A-A





SCALE 1 : 3

Note: There are 144 slots in total.
Slots are evenly spaced.



DETAIL B
SCALE 4 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:38:32 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

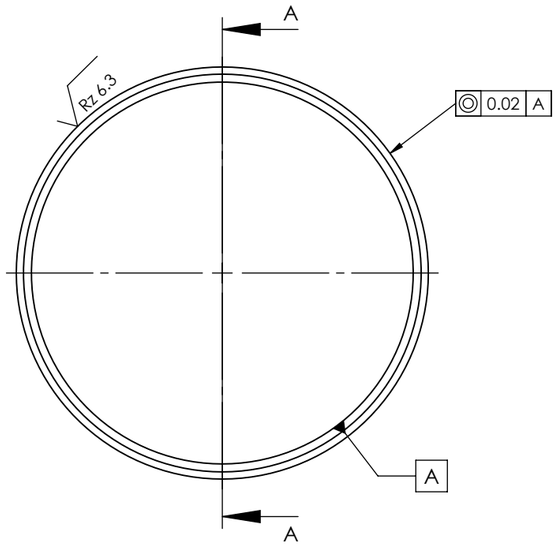
Cooler Cartridge

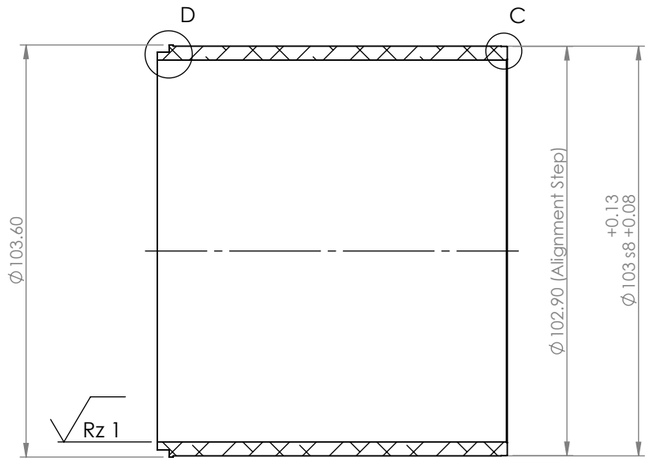
Material: 6061 Alloy

DWG NO. A-DC-CC-01-COOLER_CARTRIDGE

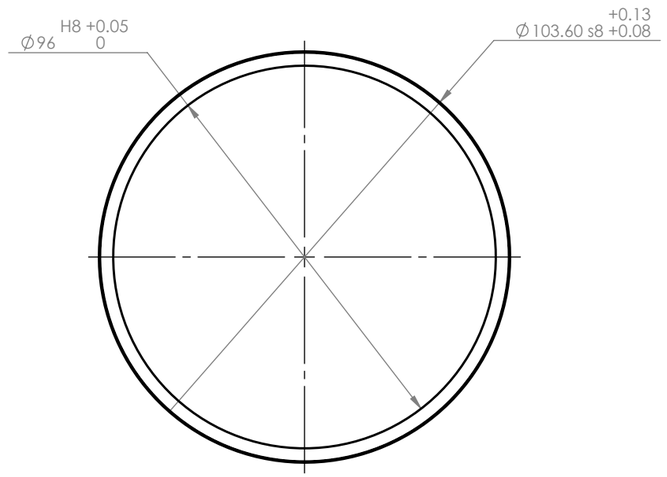
REVISION A

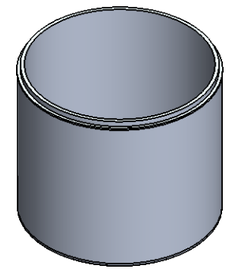
DRW File: 008_A-DC-CC-01-COOLER_CARTRIDGE | Project: GSE-1 | Mass: 700.91318 | SCALE:2:3 | SHEET 1 OF 1



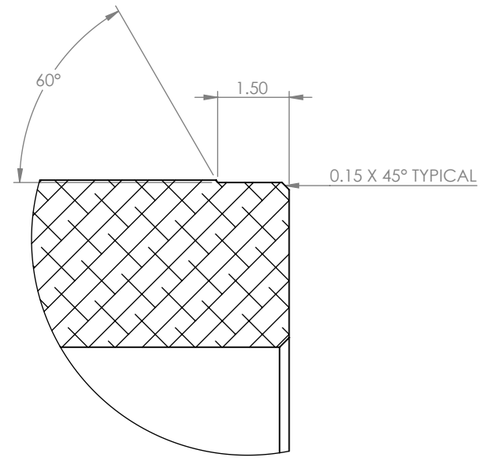


SECTION A-A

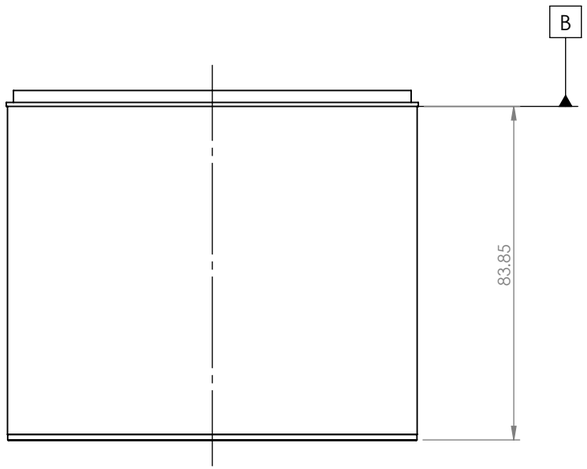


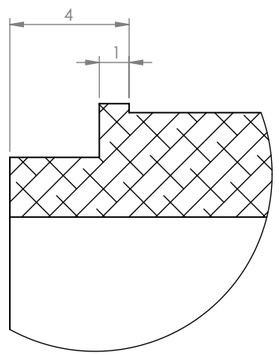


SCALE 1 : 3



DETAIL C
SCALE 8 : 1





DETAIL D
SCALE 5 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:20:46 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Cooler Cartridge Pipe

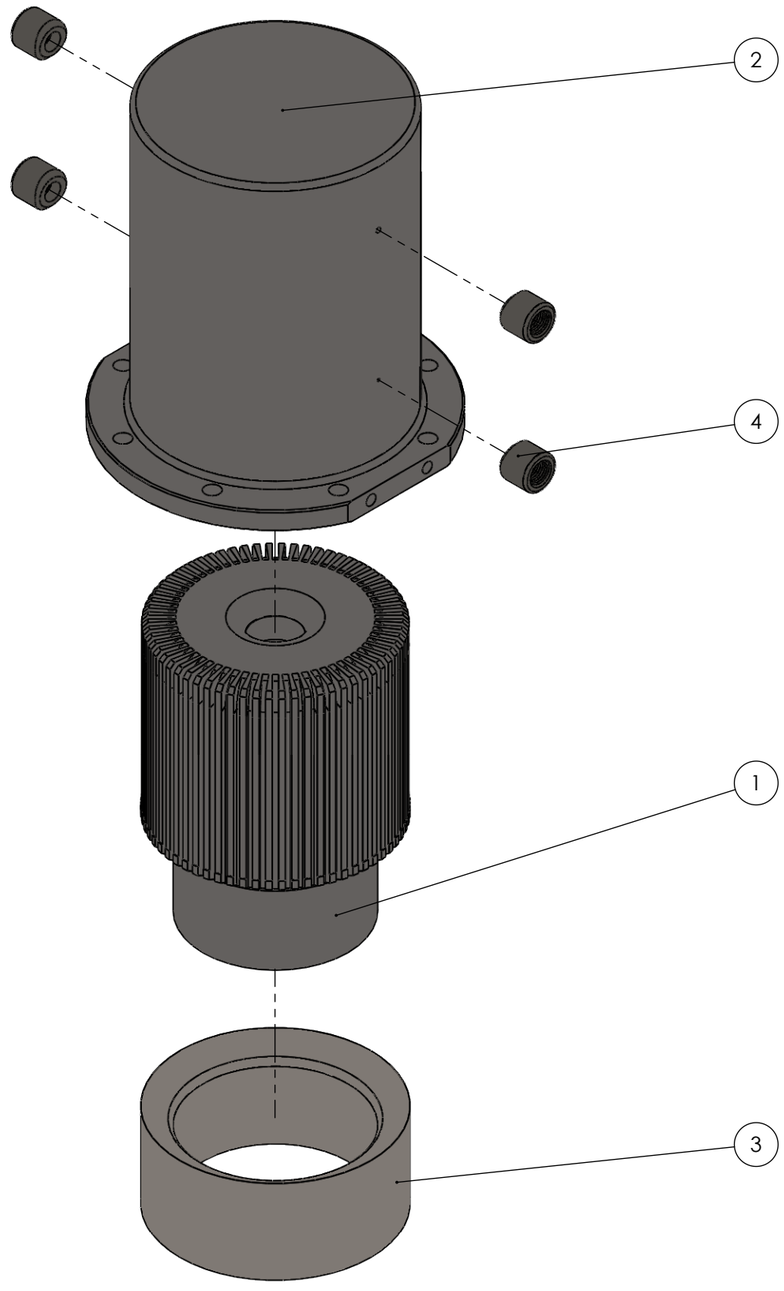
Material: 6061-T6 (SS)

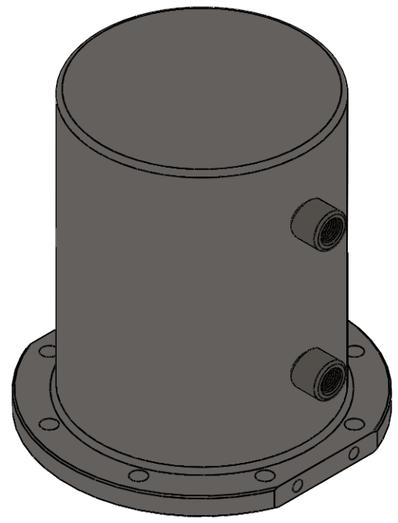
DWG NO. **A-DC-CC-02-COOLER_CARTRIDGE_PIPE**

REVISION A

DRW File: 008_A-DC-CC-02-COOLER_CARTRIDGE_PIPE | Project: **GSE-1** | Mass: 256.24495 | SCALE:2:3 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	A-DC-H-01-HEAT_LINER	1
2	A-DC-H-02-HEATER_HEAD_SHELL	1
3	A-DC-Z-03-REGEN	1
4	A-DC-H-03-HEATER_HEAD_INLET	4





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:44:58 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Displacer Piston Shell

Material: N/A

DWG NO. **A-DP-D-02-DISP_SHELL**

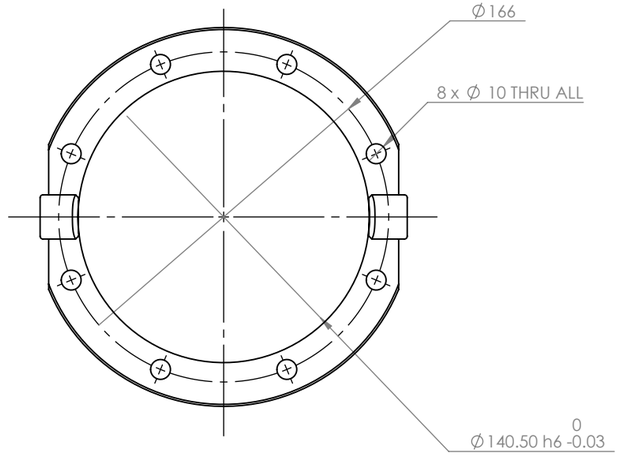
REVISION A

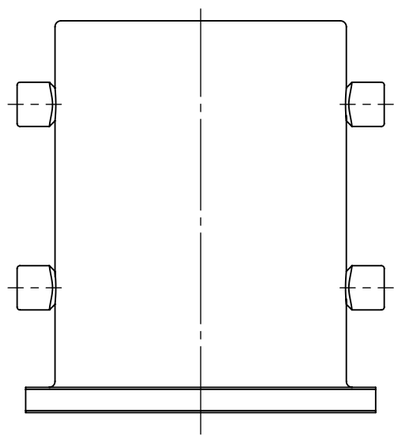
DRW File: 010_A-DC-H-00-HEATER_HEAD_ASM

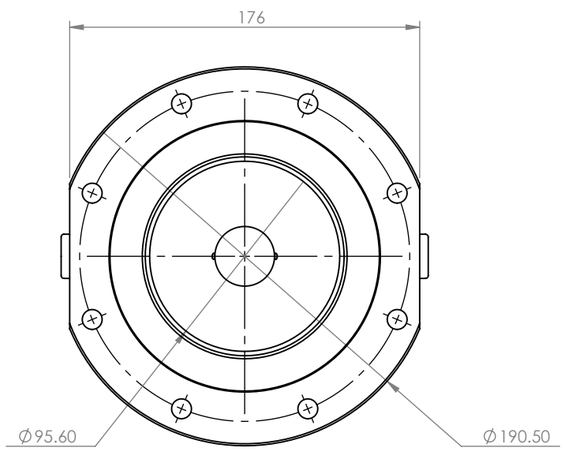
Project: **GSE-1**

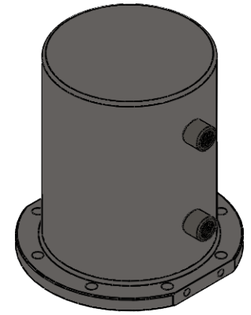
Mass:

SCALE:1:3

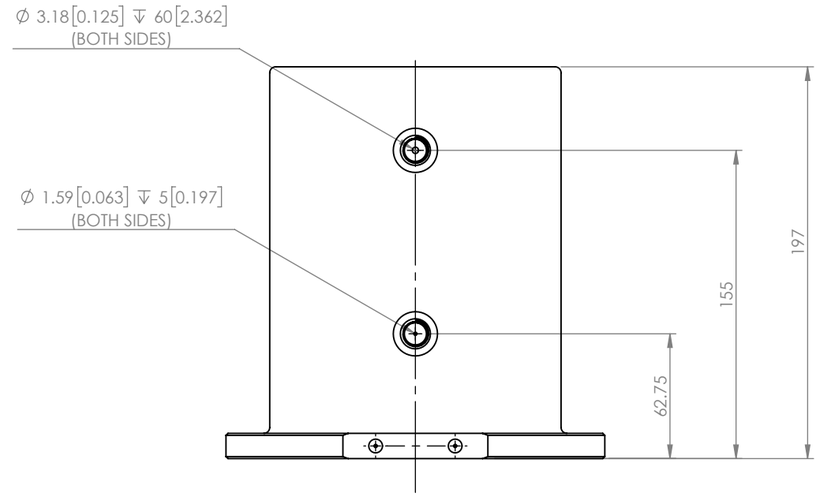








SCALE 1 : 5



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:44:58 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

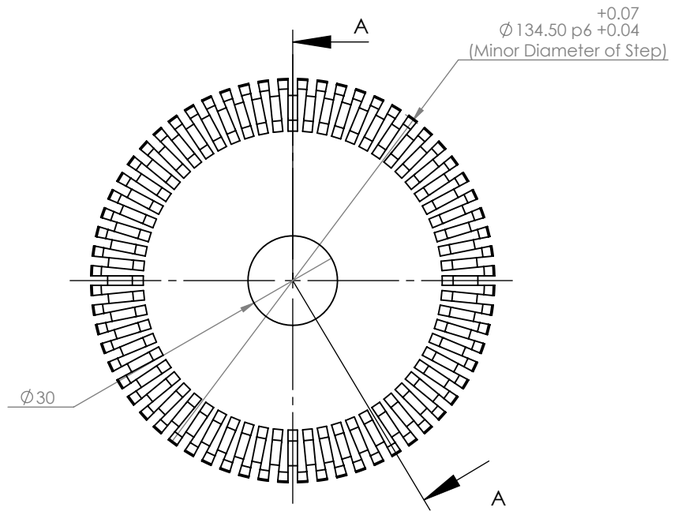
Displacer Piston Shell

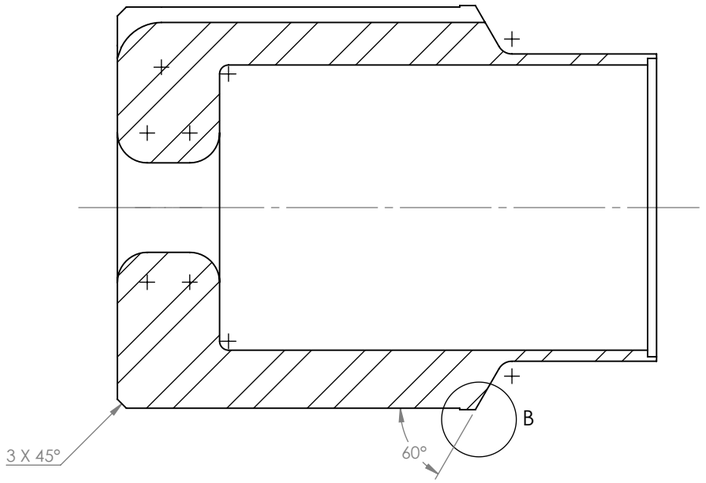
Material: N/A

DWG NO. **A-DP-D-02-DISP_SHELL**

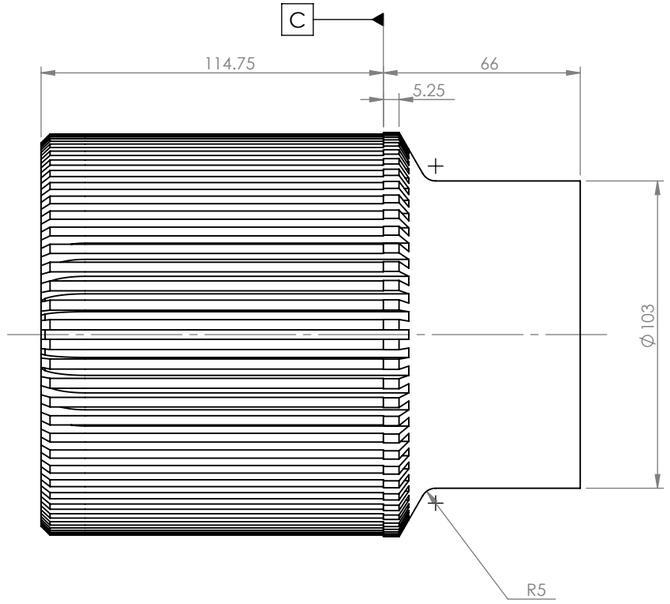
REVISION A

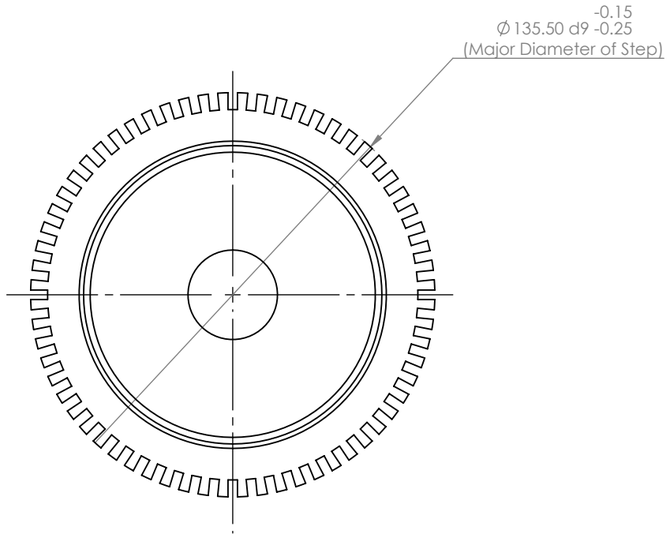
DRW File: 010_A-DC-H-00-HEATER_HEAD_ASM | Project: **GSE-1** | Mass: | SCALE:1:3 | SHEET 2 OF 2

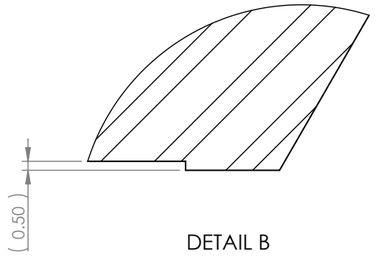




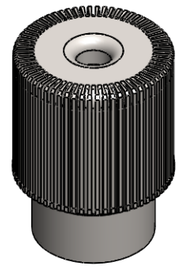
SECTION A-A







DETAIL B
SCALE 3 : 1



SCALE 1 : 5

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:42:50 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

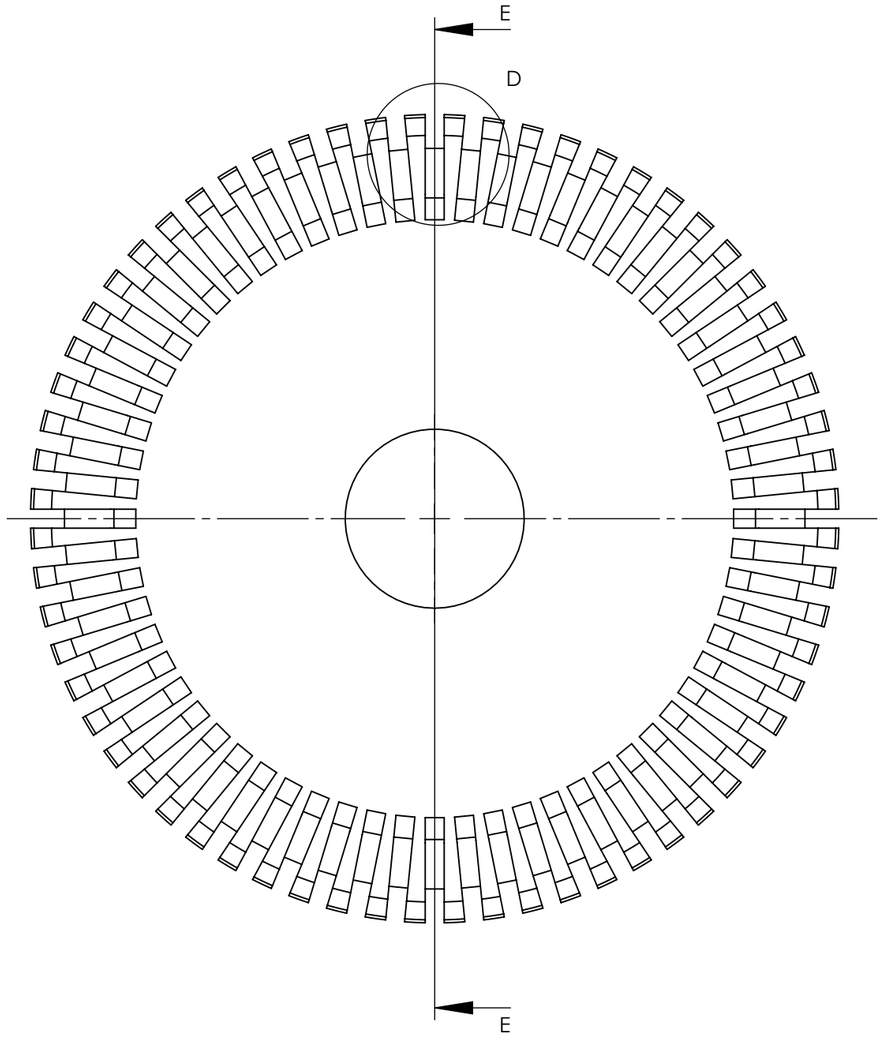
DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

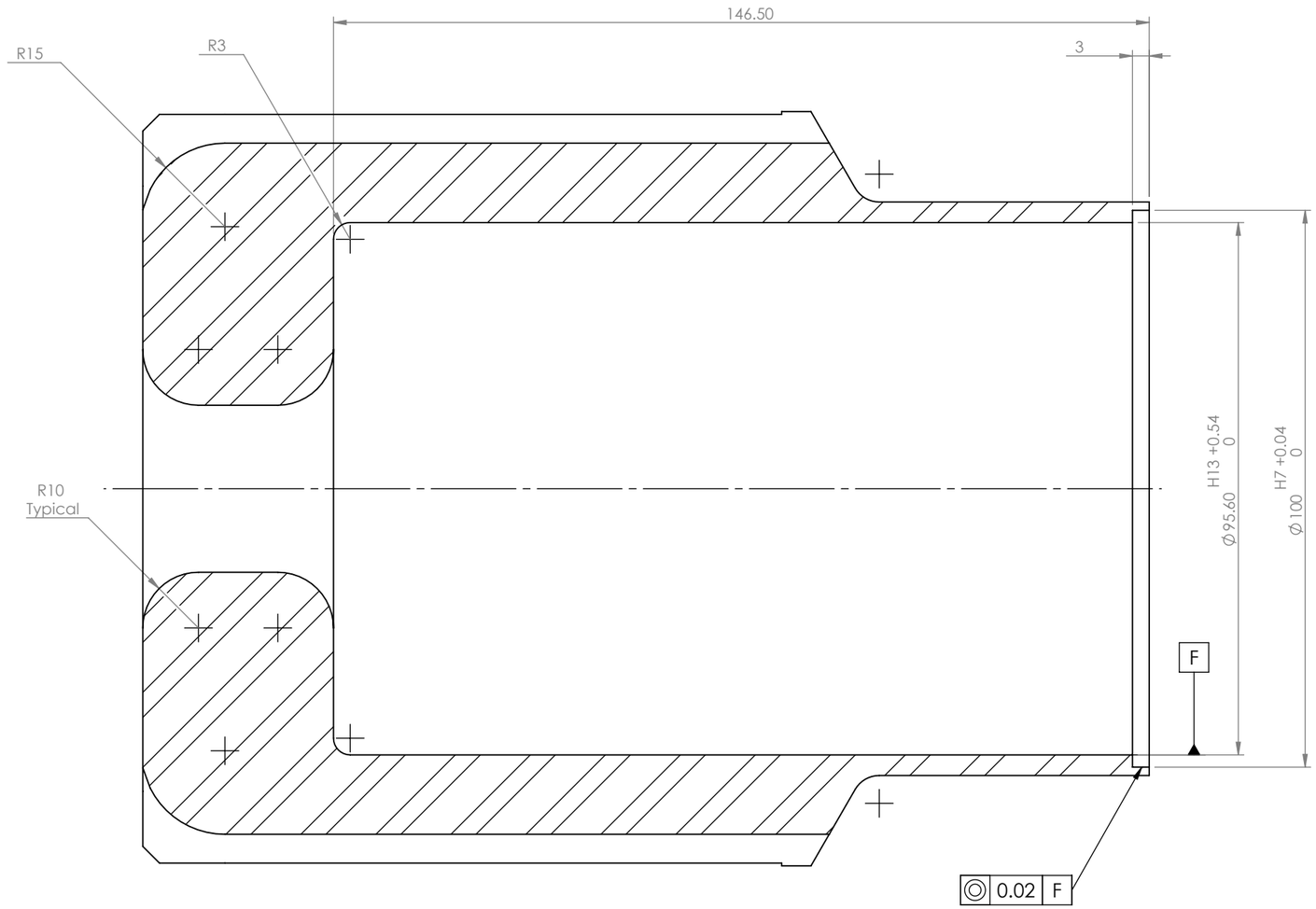
TITLE:

Heater Head Cylinder Liner

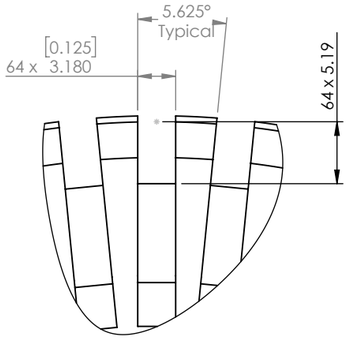
Material:	DWG NO.	REVISION
AISI 1020	**A-DC-H-01-HEAT_LINER**	A

DRW File: 010_A-DC-H-01-HEAT_LINER | Project: **GSE-1** | Mass: 8049.17873 | SCALE:1:2 | SHEET 1 OF 2





SECTION E-E



DETAIL D
SCALE 2 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:42:50 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

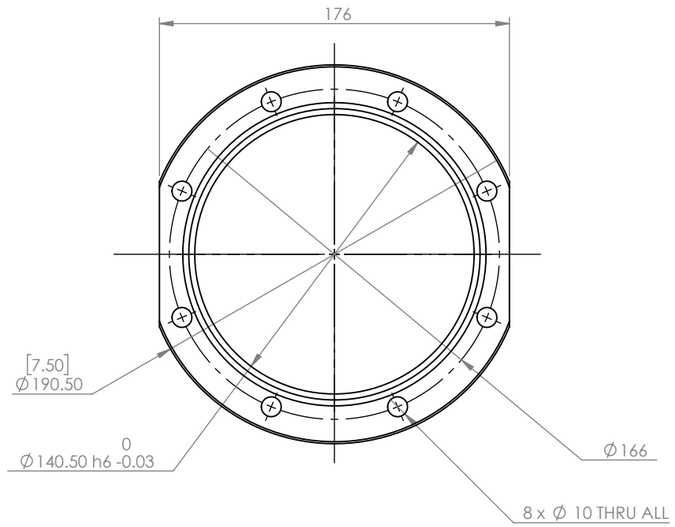
Heater Head Cylinder Liner

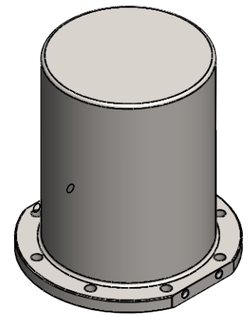
Material: AISI 1020

DWG NO. **A-DC-H-01-HEAT_LINER**

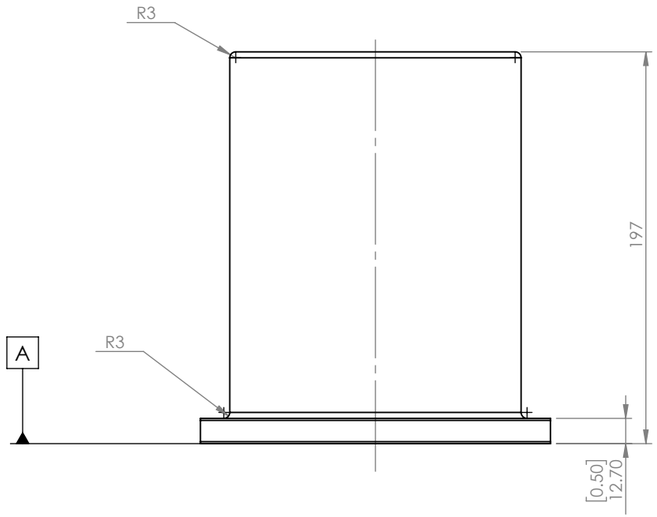
REVISION A

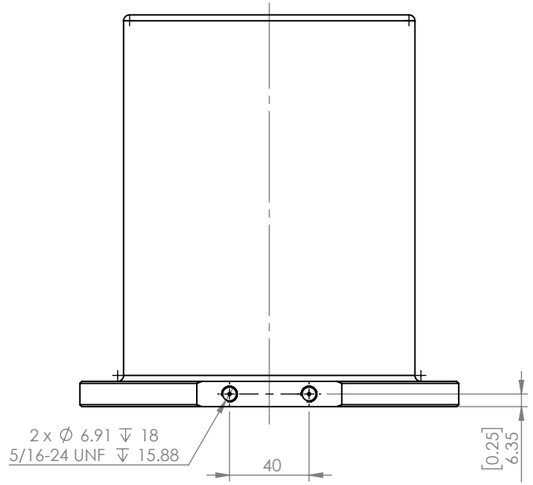
DRW File: 010_A-DC-H-01-HEAT_LINER | Project: **GSE-1** | Mass: 8049.17873 | SCALE:1:1 | SHEET 2 OF 2





SCALE 1 : 5





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:47:19 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

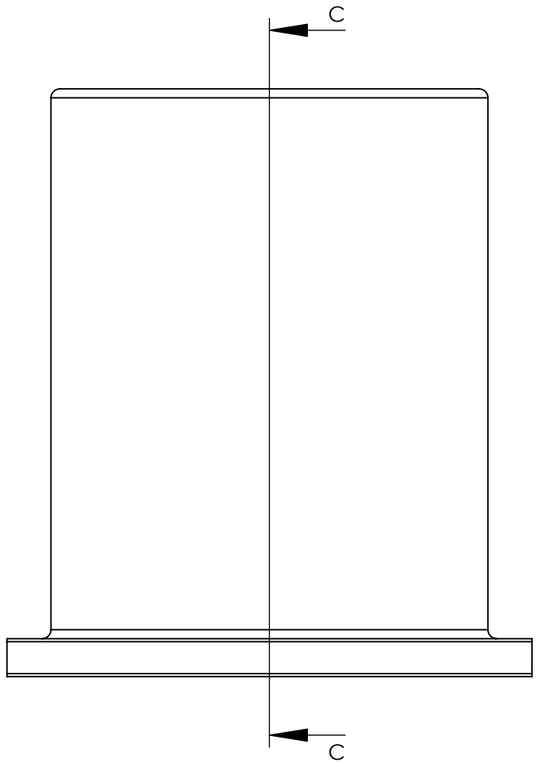
Heater Head Outer Shell

Material: AISI 1020

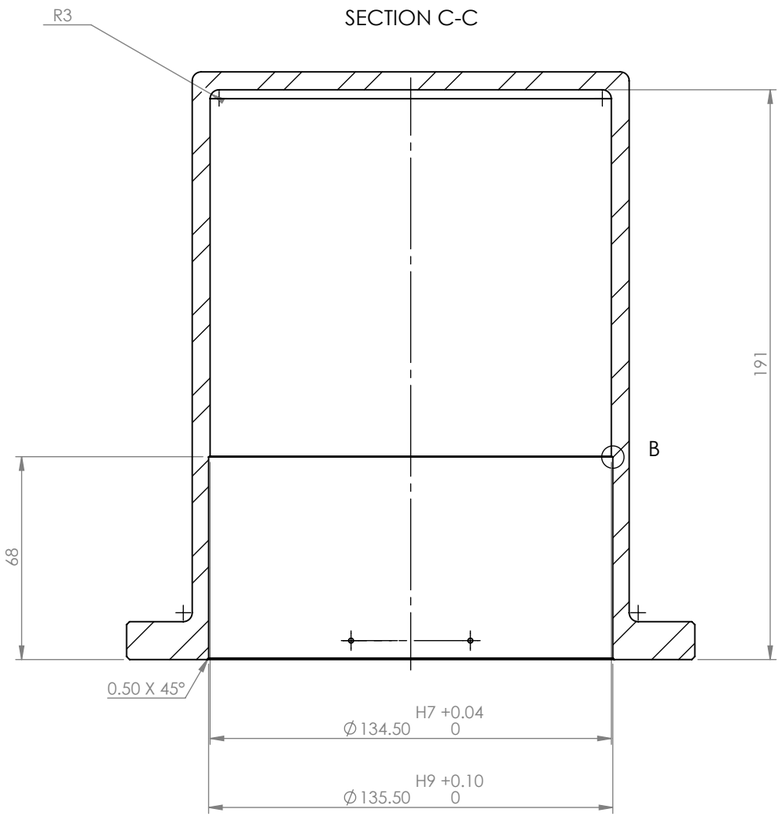
DWG NO. A-DC-H-02-HEATER_HEAD_SHELL

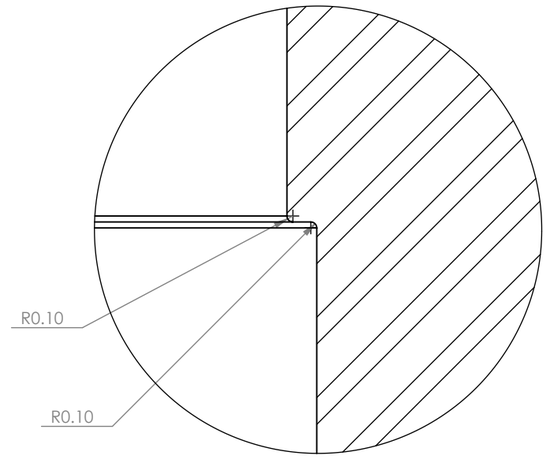
REVISION A

DRW File: 010_A-DC-H-02-HEATER_HEAD_SHELL | Project: GSE-1 | Mass: 5695.49312 | SCALE:1:3 | SHEET 1 OF 2



SECTION C-C





DETAIL B
SCALE 10 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:47:19 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

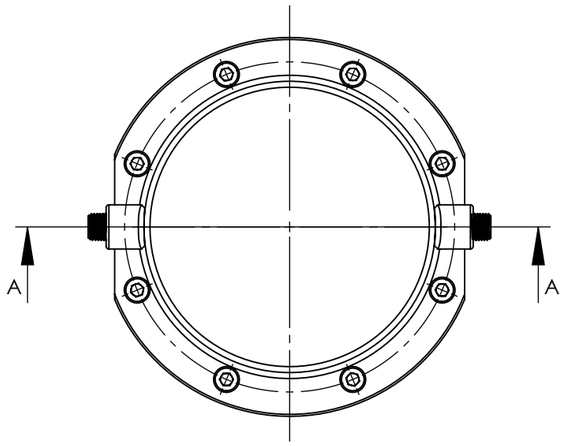
Heater Head Outer Shell

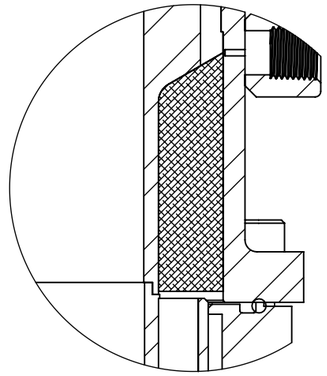
Material: AISI 1020

DWG NO. A-DC-H-02-HEATER_HEAD_SHELL

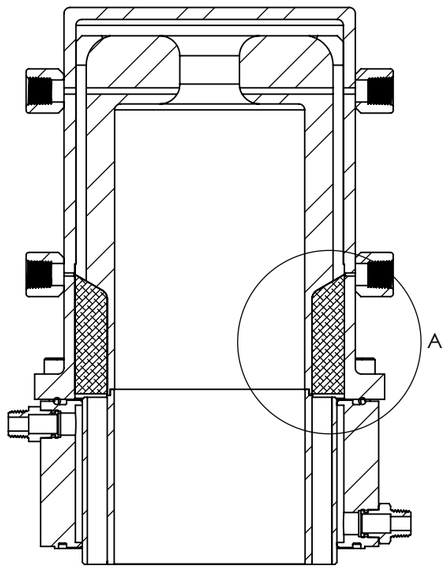
REVISION A

DRW File: 010_A-DC-H-02-HEATER_HEAD_SHELL | Project: GSE-1 | Mass: 5695.49312 | SCALE:1:2 | SHEET 2 OF 2





DETAIL A
SCALE 2 : 3



SECTION A-A

Stuff Regenerator space with 250 g of Knitted Stainless Steel Wire Mesh dia. 0.05 - 0.06 mm. This will yield a filling factor of 0.1.

Before stuffing, remove oil residues from the wire by heating it to 330 deg C. Then, blow off any particles using compressed air.

Take care to stuff the mesh equally into the regenerator space and to keep the Heater Head free from impurities and/or wire particles.

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 1:48:32 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

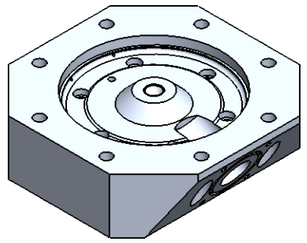
UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

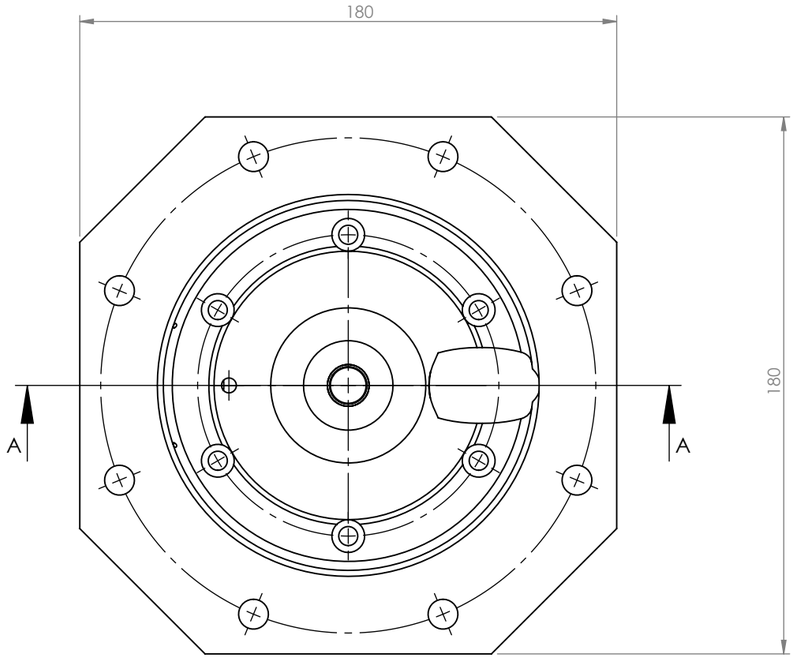
DRAWN	**Jiacheng Yao**	TITLE: **Regenerator**
SOLID by	Jiacheng Yao	
CHK'D	Connor Speer	
APPV'D	NOT APPROVED	

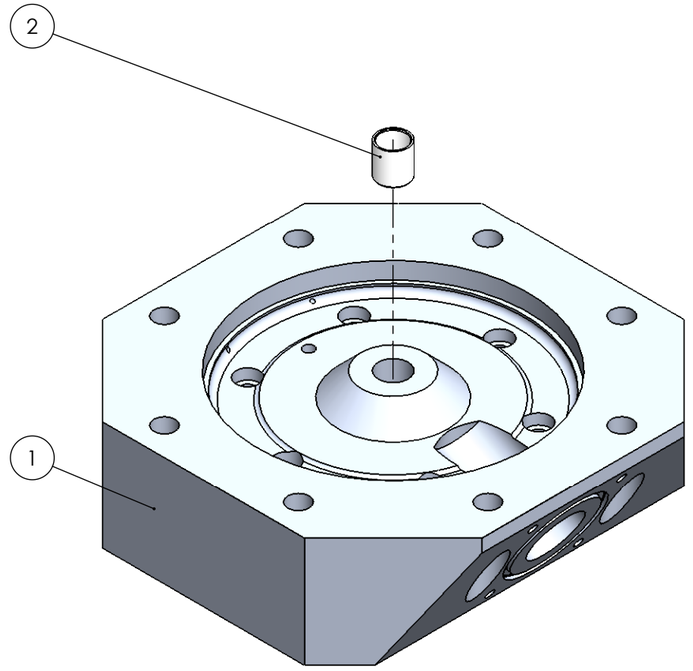
Material:	DWG NO.	REVISION
SS Wire Mesh	**A-DC-Z-00-DISP_CYL_ASM**	A

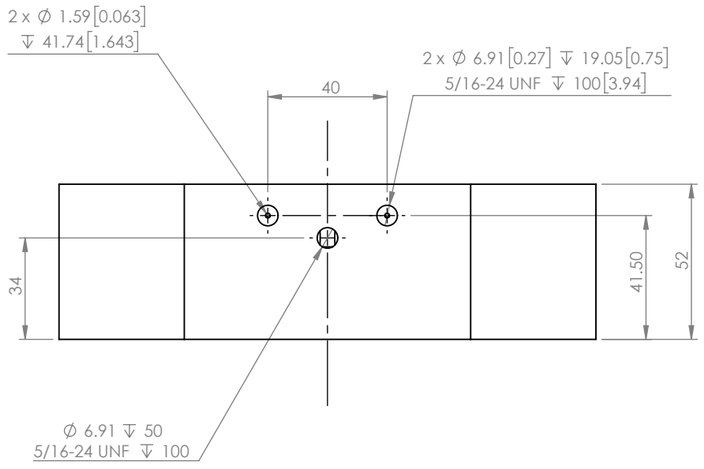
DRW File: 012_A-DC-Z-03-REGEN	Project: **GSE-1**	Mass:	SCALE:1:3	SHEET 1 OF 1

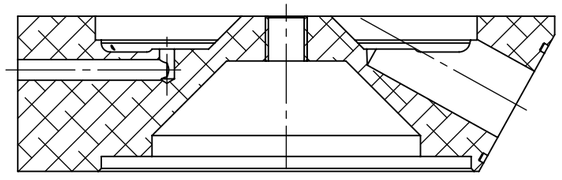
ITEM NO.	PART NUMBER	QTY.
1	A-DM-Z-01-DISP_MOUNT	1
2	A-DM-Z-02-SHAFT_BEARING	1











SECTION A-A

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 10:35:24 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

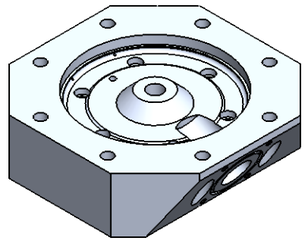
Displacer Mount Assembly

Material: N/A

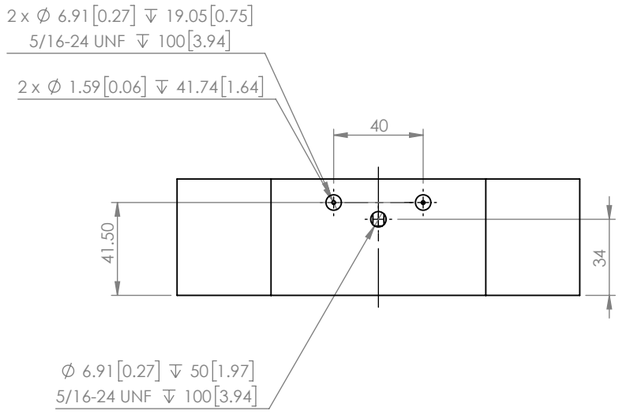
DWG NO. **A-DM-Z-00-DISP_MOUNT_ASM**

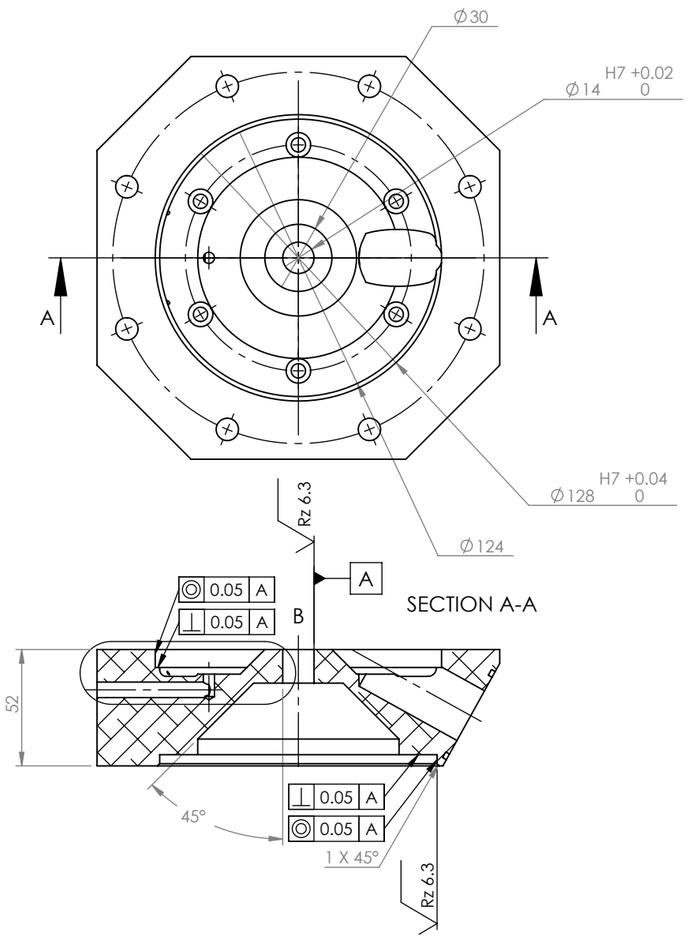
REVISION A

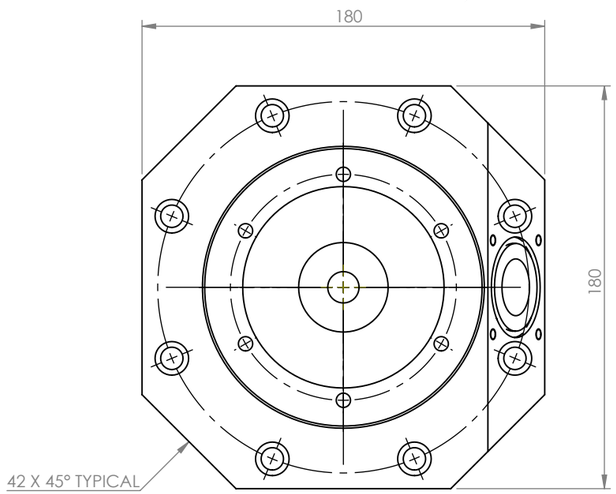
DRW File: 014_A-DM-Z-00-DISP_MOUNT_ASM | Project: **GSE-1** | Mass: | SCALE:1:2 | SHEET 1 OF 1

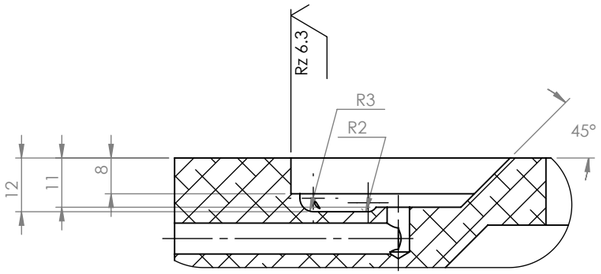


SCALE 1 : 4









DETAIL B
SCALE 6 : 8

Please fillet/chamfer all edges to R0.3

General surface finish: √ Rz 6.3

O-ring groove: √ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier:

McMaster-Carr
9246K22

Quantity: 1

Friday, April 15, 2016 2:08:41 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

DISPLACER CYLINDER MOUNT

Material: 6061-T6 (SS)

DWG NO. A-DM-Z-01-DISP_MOUNT

REVISION A

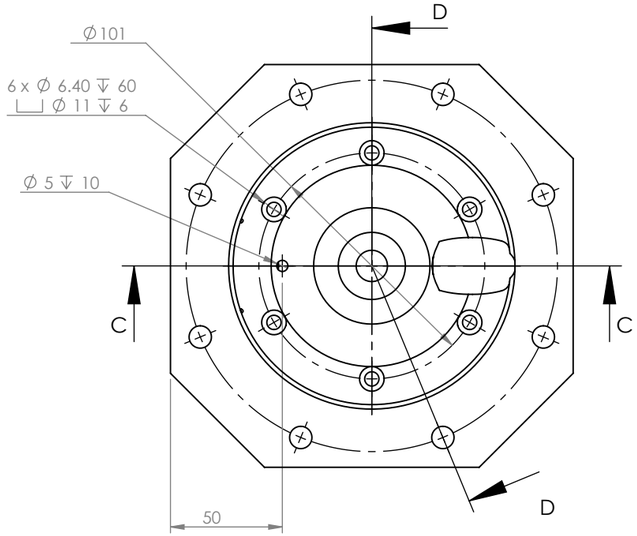
DRW File: 014_A-DM-Z-01-DISP_MOUNT

Project: GSE-1

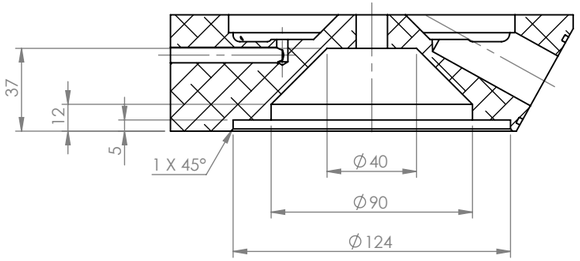
Mass: 2762.50256

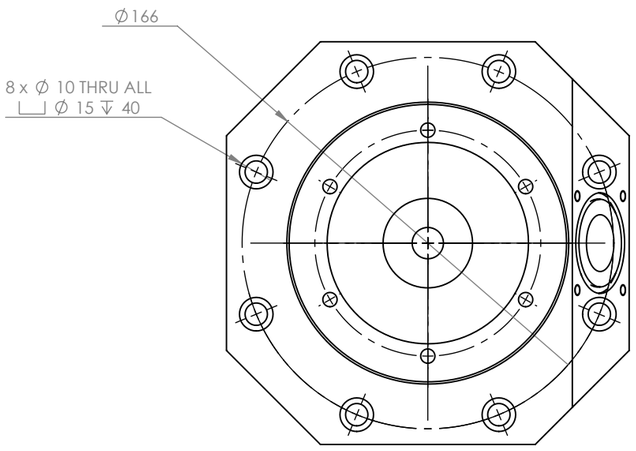
SCALE:3:8

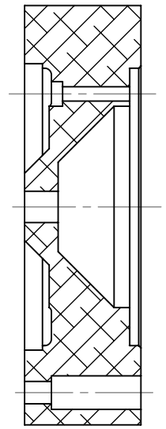
SHEET 1 OF 3



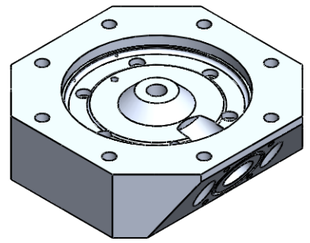
SECTION C-C







SECTION D-D



SCALE 1 : 4

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier:

McMaster-Carr
9246K22

Quantity: 1

Friday, April 15, 2016 2:08:41 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

DISPLACER CYLINDER MOUNT

Material: 6061-T6 (SS)

DWG NO. A-DM-Z-01-DISP_MOUNT

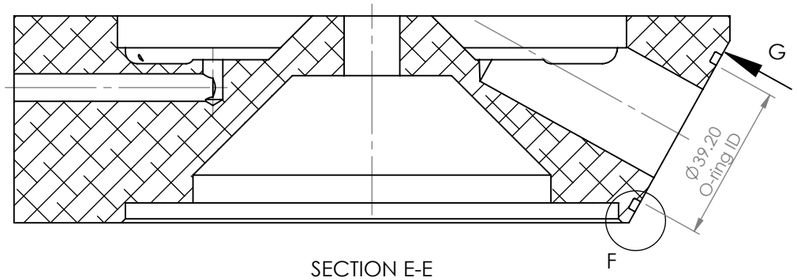
REVISION A

DRW File: 014_A-DM-Z-01-DISP_MOUNT

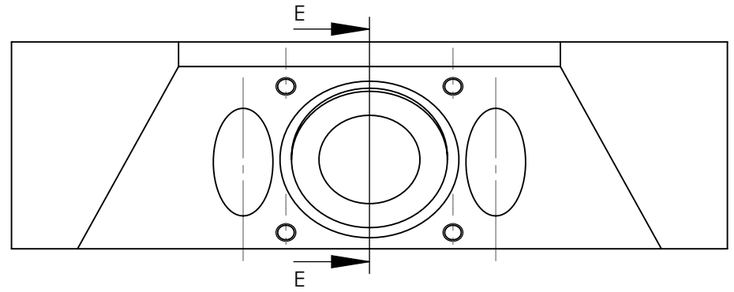
Project: GSE-1

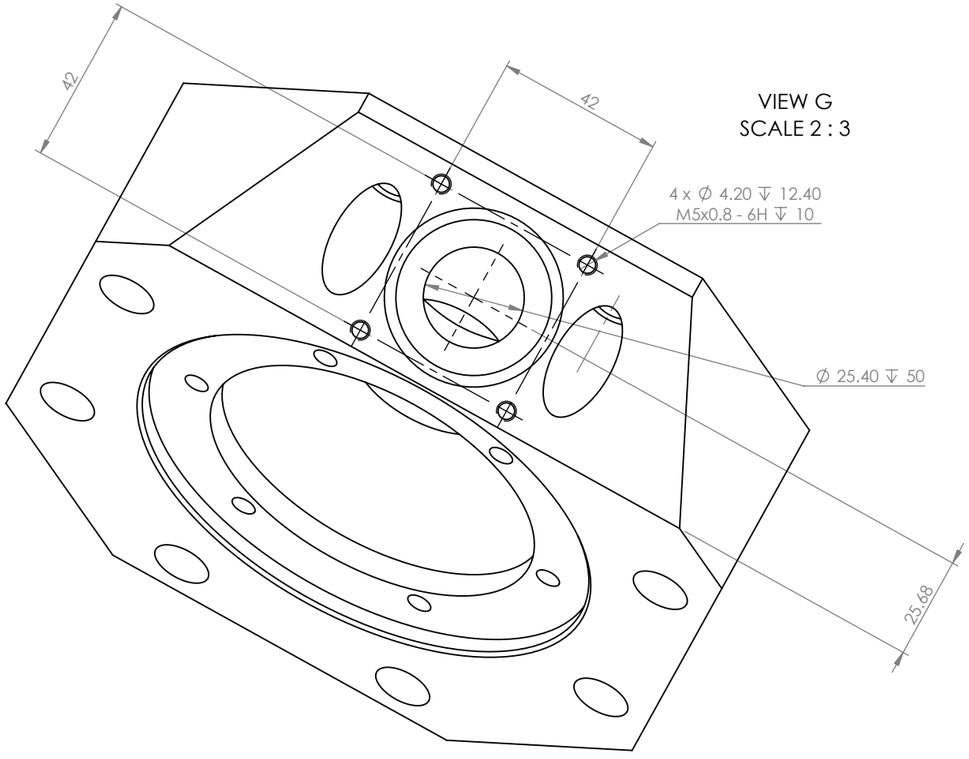
Mass: 2762.50256

SCALE:3:8



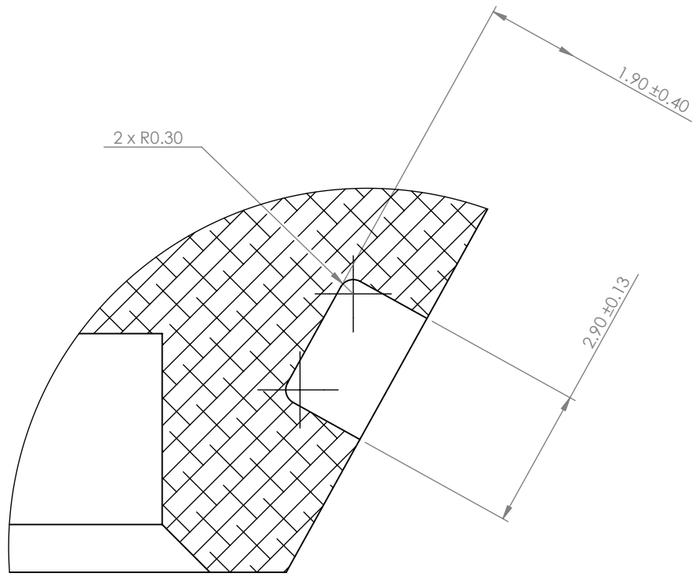
SECTION E-E
SCALE 2 : 3





VIEW G
SCALE 2 : 3

O-Ring Information:
Size: 40 x 2.5mm
Material: Nitrile
Supplier: Hi-Tech Seals
Part #: MN7 25040



DETAIL F
SCALE 8 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier:

McMaster-Carr
9246K22

Quantity: 1

Friday, April 15, 2016 2:08:41 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

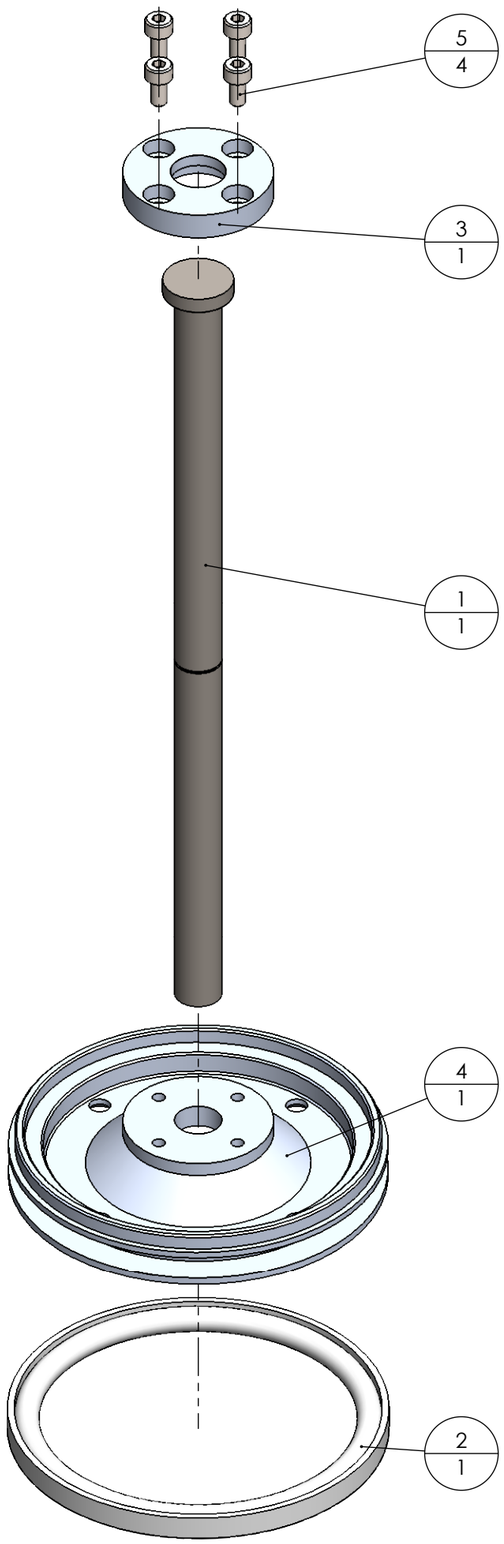
DISPLACER CYLINDER MOUNT

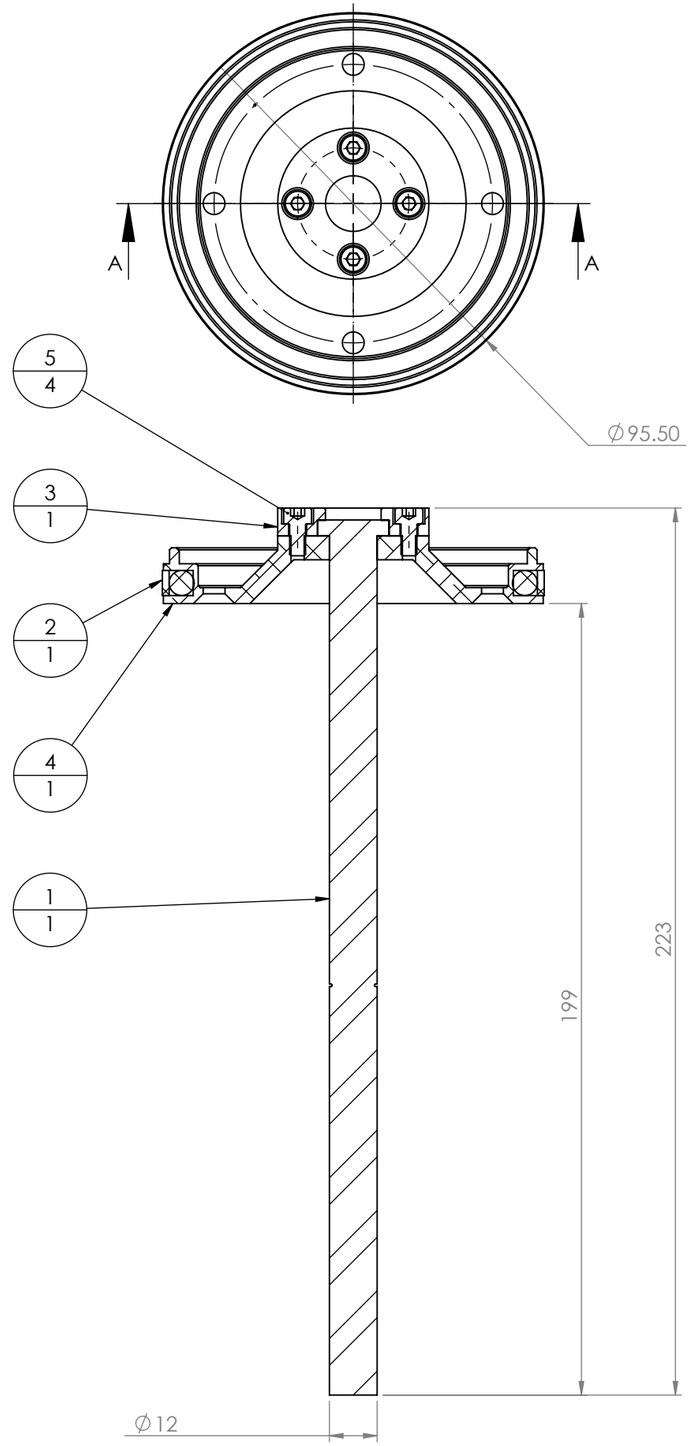
Material: 6061-T6 (SS)

DWG NO. A-DM-Z-01-DISP_MOUNT

REVISION A

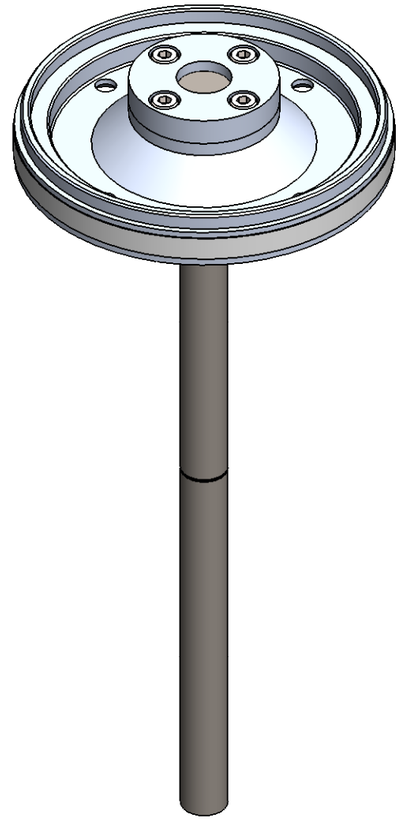
DRW File: 014_A-DM-Z-01-DISP_MOUNT | Project: GSE-1 | Mass: 2762.50256 | SCALE:3:8 | SHEET 3 OF 3





SECTION A-A

ITEM NO.	DRW NUMBER	QTY.
1	A-DP-B-01-DISP_SHAFT	1
2	A-DP-B-04-DISP_RING	1
3	A-DP-B-03-DISP_CLAMP_RING	1
4	A-DP-B-02-DISP_BASE	1
5	B18.3.1M - 4 x 0.7 x 8 Hex SHCS -- 8NHX	4



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH: 0.00005
TOLERANCES:
LINEAR: 0.0025
ANGULAR: 0.25°

Comments
Quantity: 1

Friday, April 15, 2016 1:50:16 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

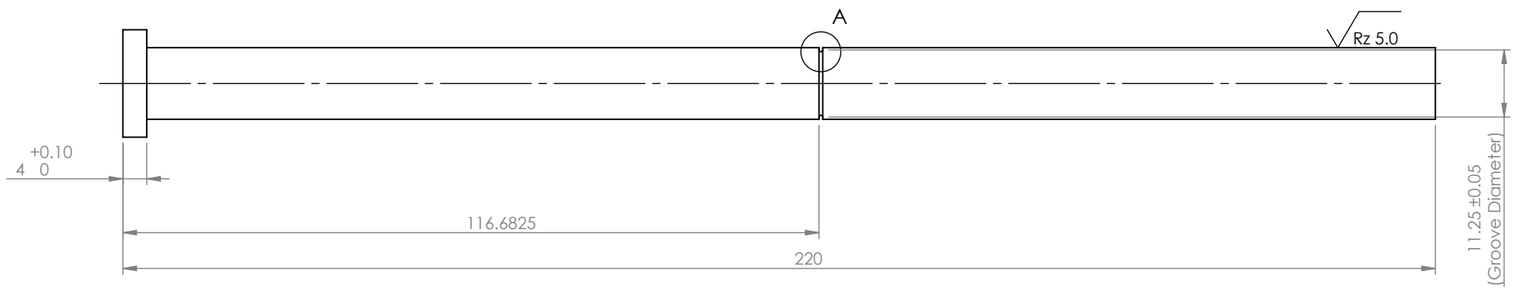
Displacer Piston Base

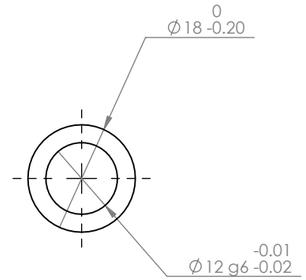
Material: N/A

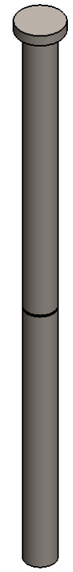
DWG NO. **A-DP-B-00-DISP_BASE_ASM**

REVISION A

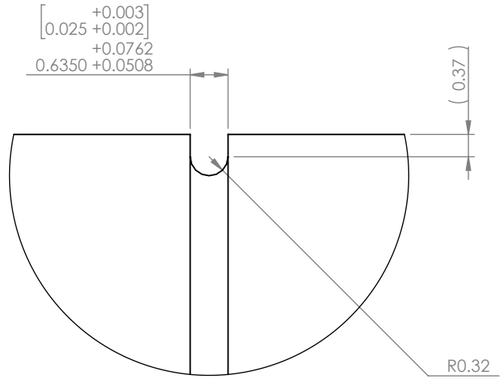
DRW File: 016_A-DP-B-00-DISP_BASE_ASM | Project: **ST05G** | Mass: | SCALE:2:3 | SHEET 1 OF 1







SCALE 1 : 2



DETAIL A
SCALE 10 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Can be machined from standard 12 mm ejector pin

Monday, April 11, 2016 11:11:03 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

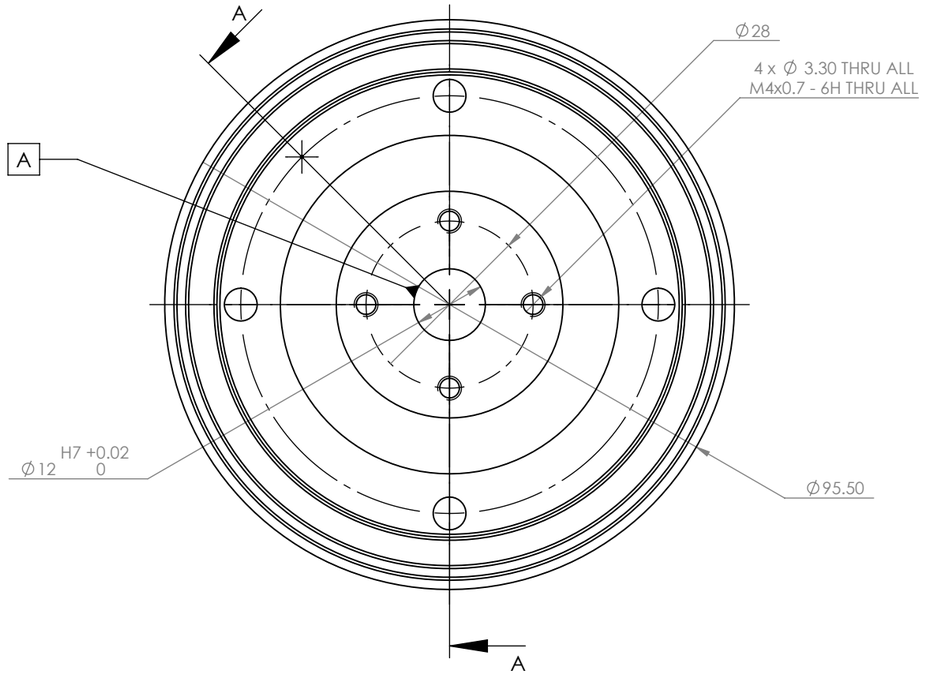
Displacer Piston Shaft

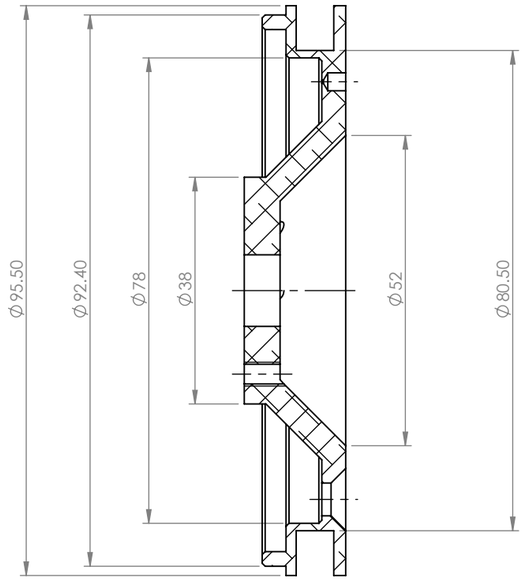
Material: ASTM A36 Steel

DWG NO. A-DP-B-01-DISP_SHAFT

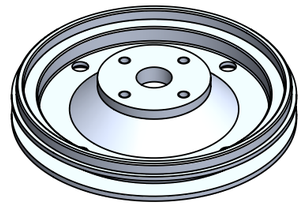
REVISION A

DRW File: 016_A-DP-B-01-DISP_SHAFT | Project: GSE-1 | Mass: 199.64721 | SCALE:1:1 | SHEET 1 OF 1

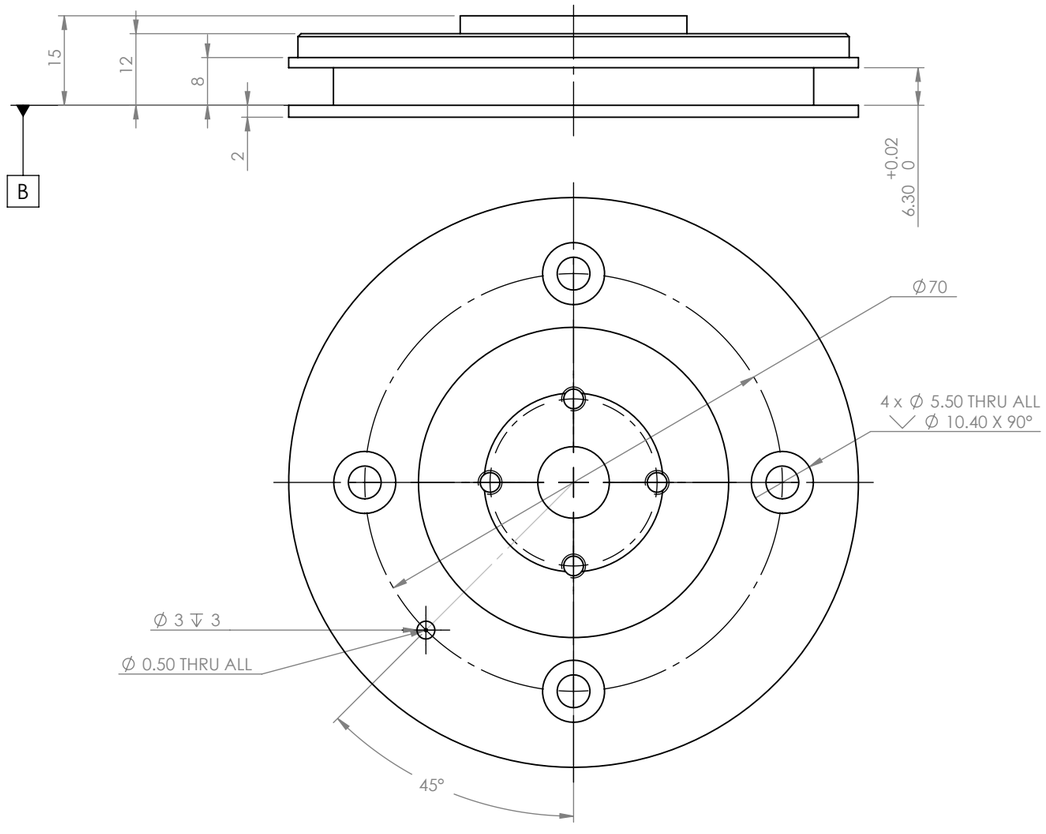




SECTION A-A



SCALE 1 : 2



Please chamfer all interior edges to 0.5 x 45 deg

Surface finish for piston ring groove : √Rz 2.5

General surface finish: √Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 8975K233

Quantity: 1

Friday, April 15, 2016 1:52:32 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Displacer Base

Material: 6061-T6 (SS)

DWG NO. A-DP-Z-02-DISP_BASE_NEW

REVISION A

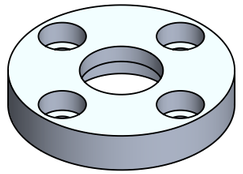
DRW File: 016_A-DP-B-02-DISP_BASE

Project: GSE-1

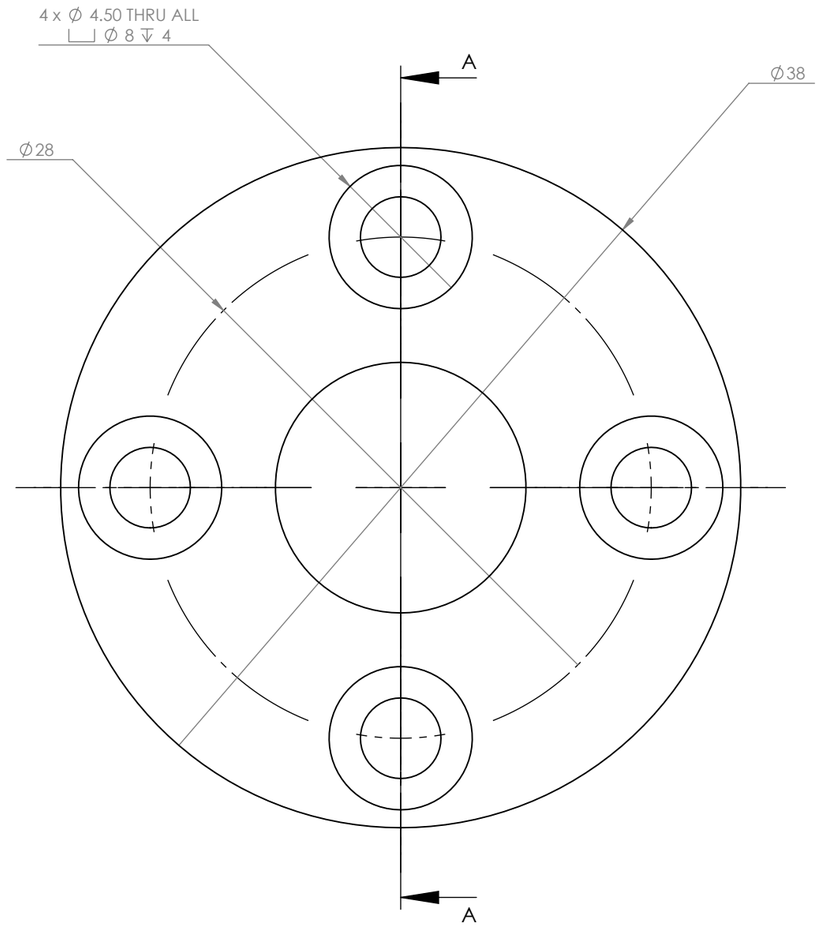
Mass: 99.91083

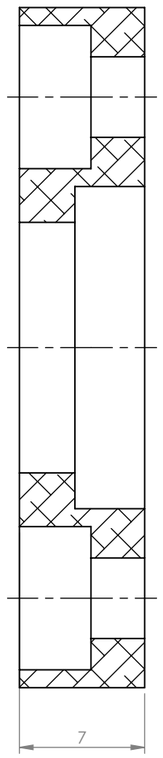
SCALE:1:1

SHEET 1 OF 1

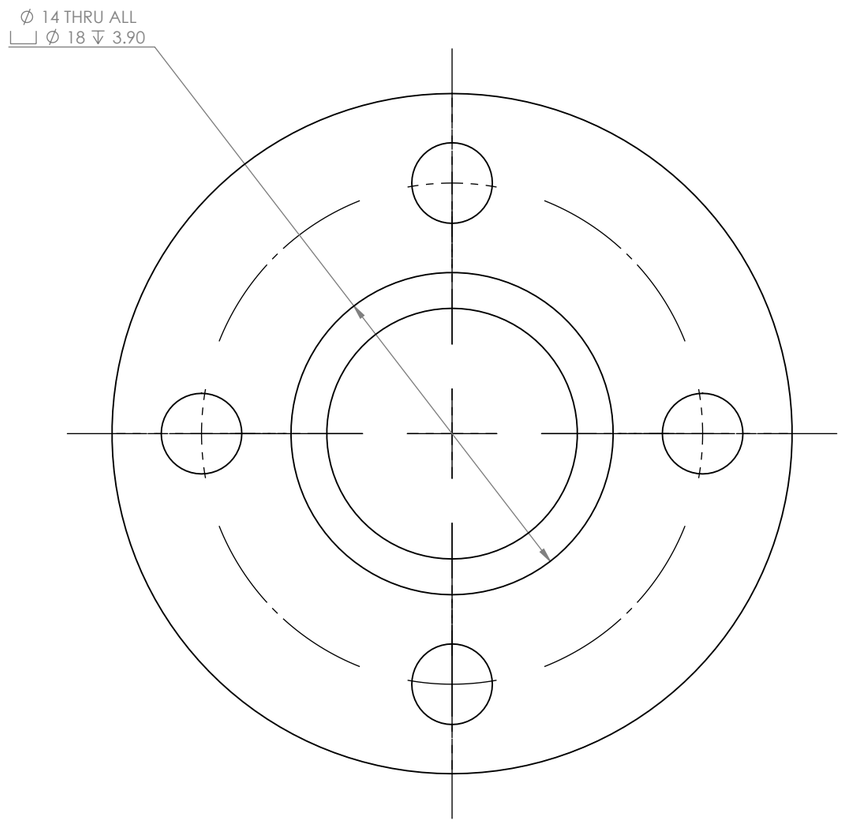


SCALE 1 : 1





SECTION A-A
SCALE 3 : 1



Please fillet/chamfer all edges to R0.3

General surface finish $\sqrt{}$ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 8975K74

Quantity: 1

Monday, April 11, 2016 12:12:34 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE: **Clamp Ring**

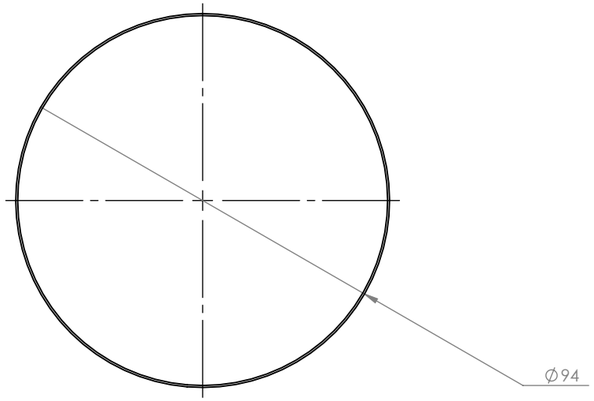
Material: 6061-T6 (SS)

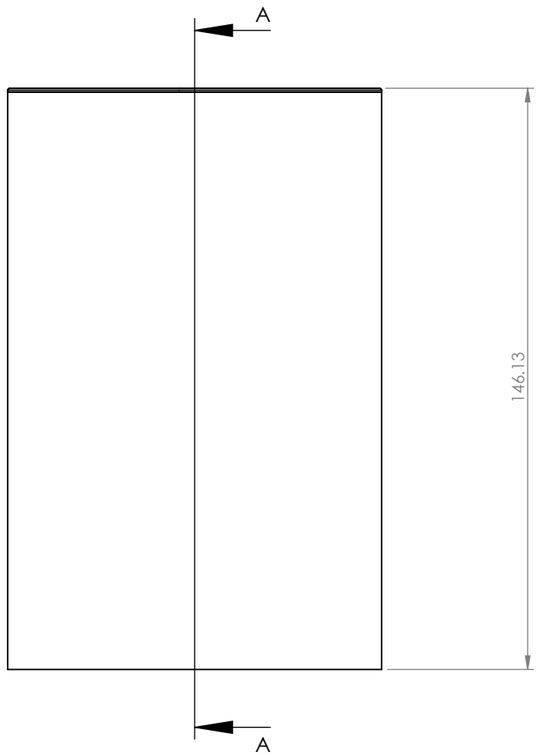
DWG NO. **A-DP-B-03-DISP_CLAMP_RING**

REVISION A

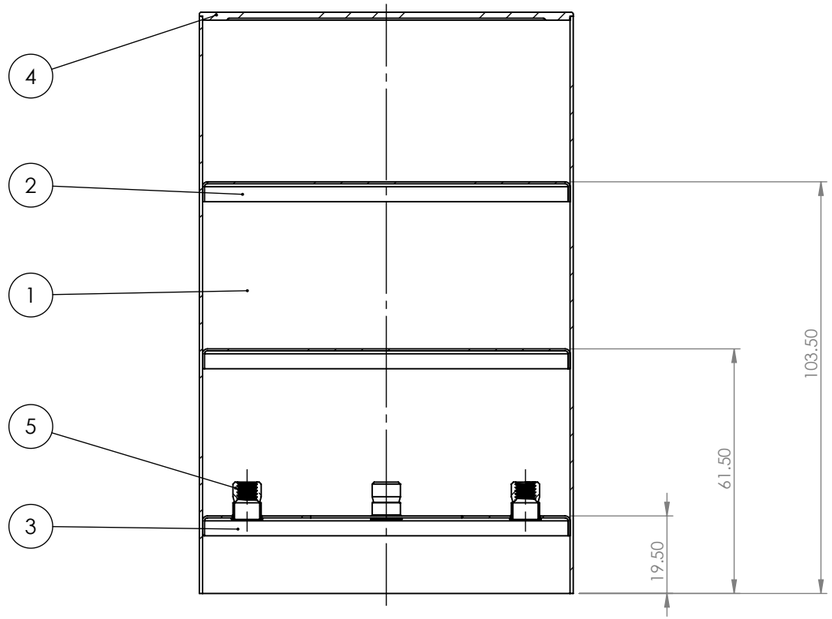
DRW File: 016_A-DP-B-03-DISP_CLAMP_RING | Project: **GSE-1** | Mass: 14.77998 | SCALE:1:10 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	A-DP-D-02-DISP_SHELL	1
2	A-DP-D-03-DISP_ISO_PLATE	2
3	A-DP-D-04-DISP_MOUNT_PLATE	1
4	A-DP-D-01-DISP_HEAD	1
5	A-DP-D-05-BLIND_RIVET_NUT	4





SECTION A-A



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 12:08:48 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Displacer Piston Assembly

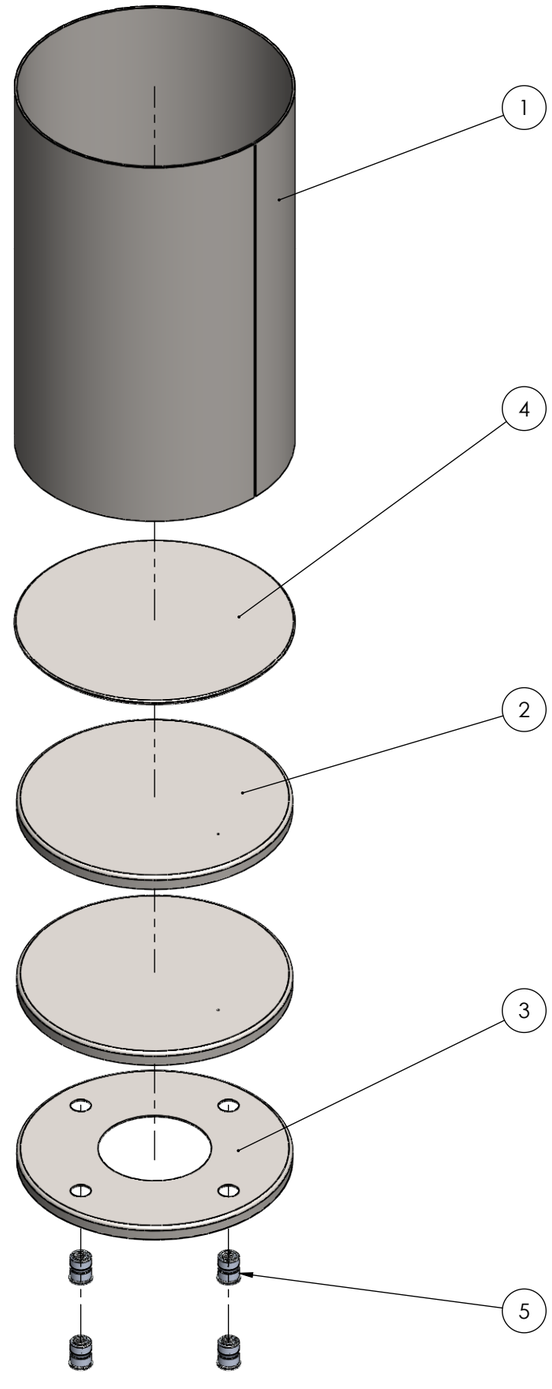
Material: N/A

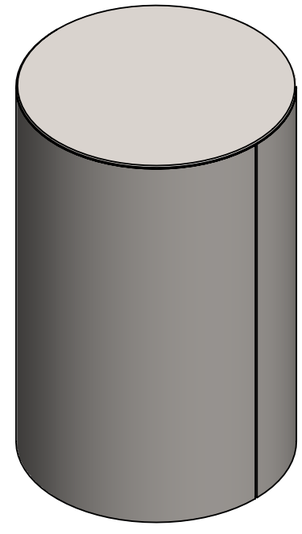
DWG NO. **A-DP-D-00-DISP_ASM**

REVISION A

DRW File: 016_A-DP-D-00-DISP_ASM | Project: **GSE-1** | Mass: | SCALE:1:1.5 | SHEET 1 OF 2

ITEM NO.	DRW NUMBER	QTY.
1	A-DP-D-02-DISP_SHELL	1
2	A-DP-D-03-DISP_ISO_PLATE	2
3	A-DP-D-04-DISP_MOUNT_PLATE	1
4	A-DP-D-01-DISP_HEAD	1
5	A-DP-D-05-BLIND_RIVET_NUT	4





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 12:08:48 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

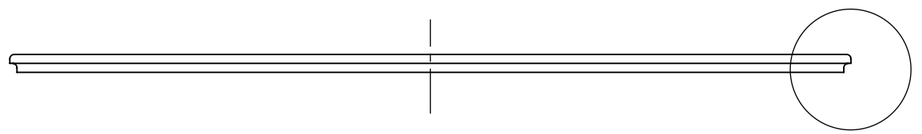
Displacer Piston Assembly

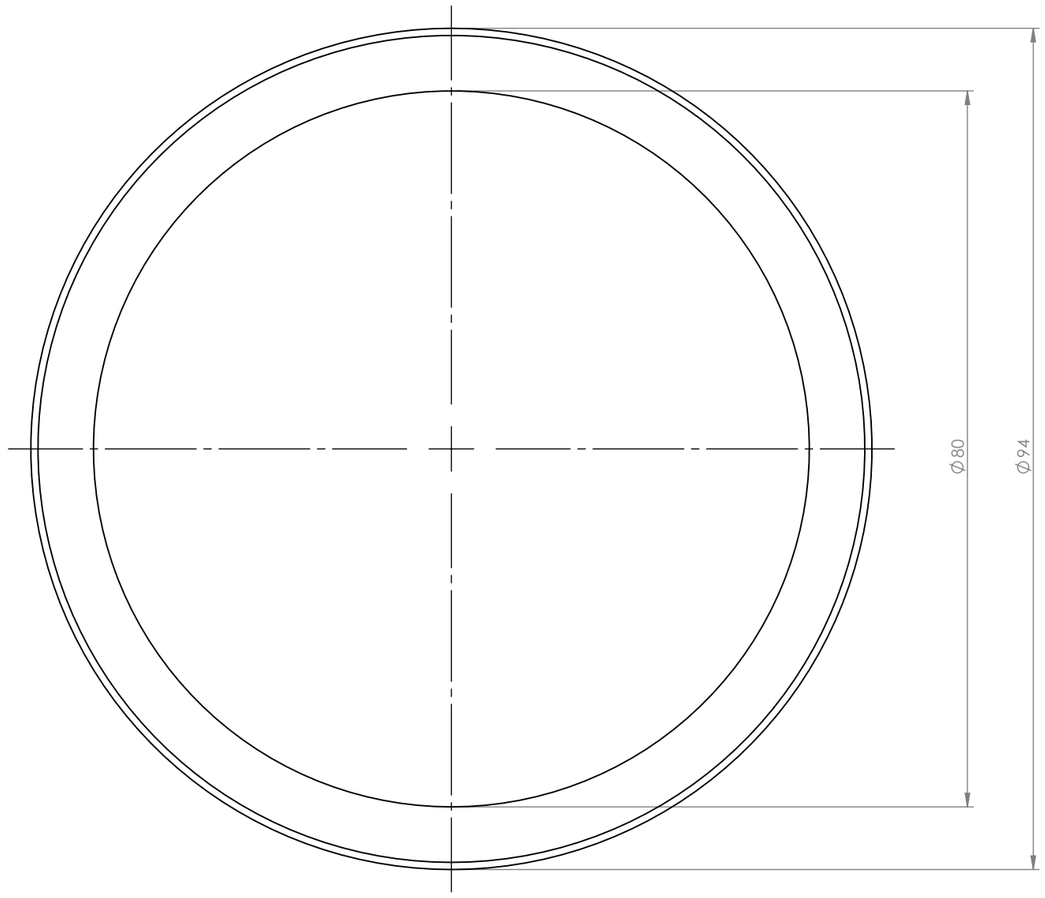
Material: N/A

DWG NO. **A-DP-D-00-DISP_ASM**

REVISION A

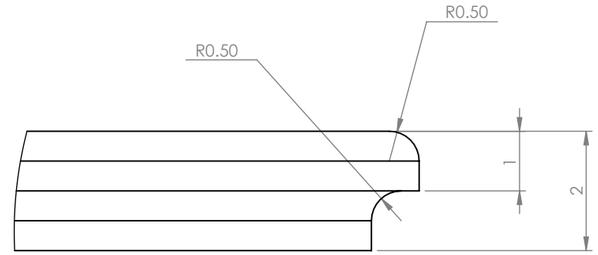
DRW File: 016_A-DP-D-00-DISP_ASM | Project: **GSE-1** | Mass: | SCALE:1:2 | SHEET 2 OF 2







SCALE 1 : 2



DETAIL A
SCALE 10 : 1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 11:55:56 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Displacer Head

Material: 1.4401 (X5CrNiMo17-12-2)

DWG NO. **A-DP-D-01-DISP_HEAD**

REVISION A

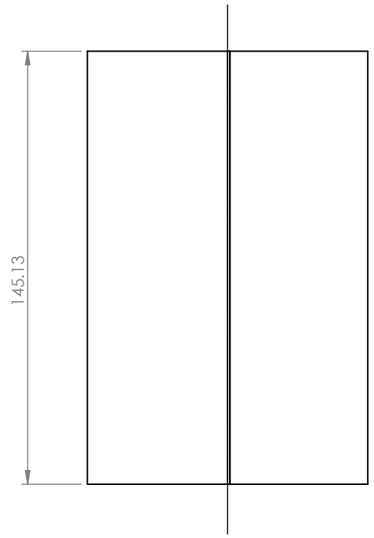
DRW File: 016_A-DP-D-01-DISP_HEAD

Project: **GSE-1**

Mass: 89.16238

SCALE:1.5:1

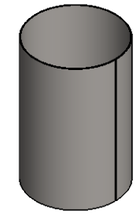
SHEET 1 OF 1



As Bent



Flat Pattern



SCALE 1 : 5

Sheet metal thickness: 0.8 mm
Bend sheet metal and TIG-weld joints.

Alternatively, scarf joint 3 mm from both ends and spot weld without gaps.

Please fillet all edges to R0.1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 11:50:58 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

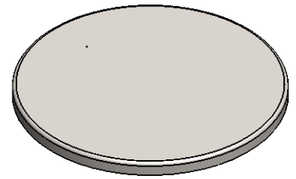
DRAWN	**Jiacheng Yao**	TITLE: **Displacer Piston Shell**
SOLID by	Jiacheng Yao	
CHK'D	Connor Speer	
APPV'D	NOT APPROVED	

Material: 1.4571 (X6CrNiMoTi17-12-2)

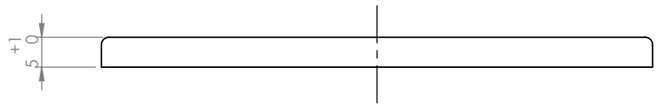
DWG NO. **A-DP-D-02-DISP_SHELL**

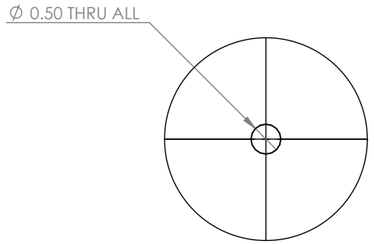
REVISION A

DRW File: 016_A-DP-D-02-DISP_SHELL | Project: **GSE-1** | Mass: 271.20326 | SCALE:1:2 | SHEET 1 OF 1

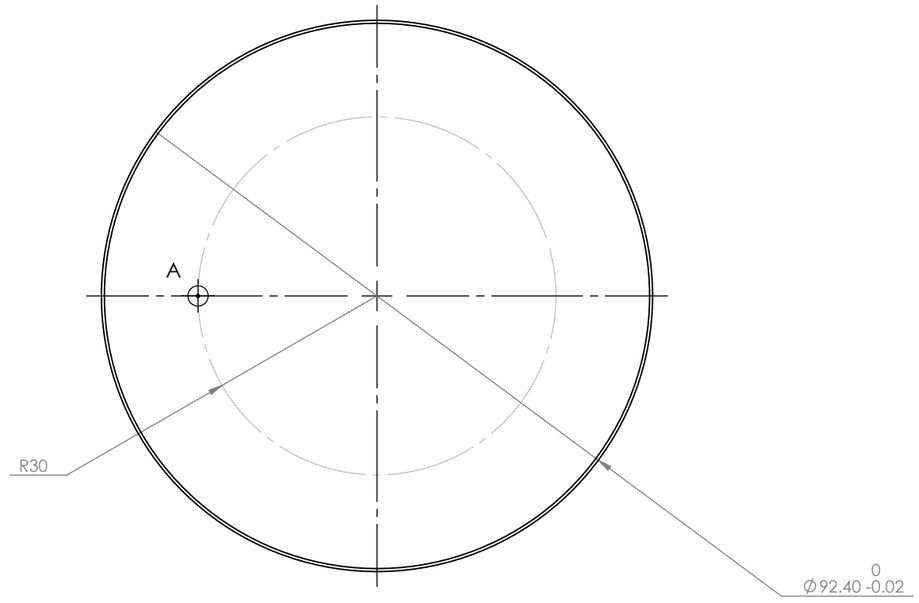


SCALE 1 : 2

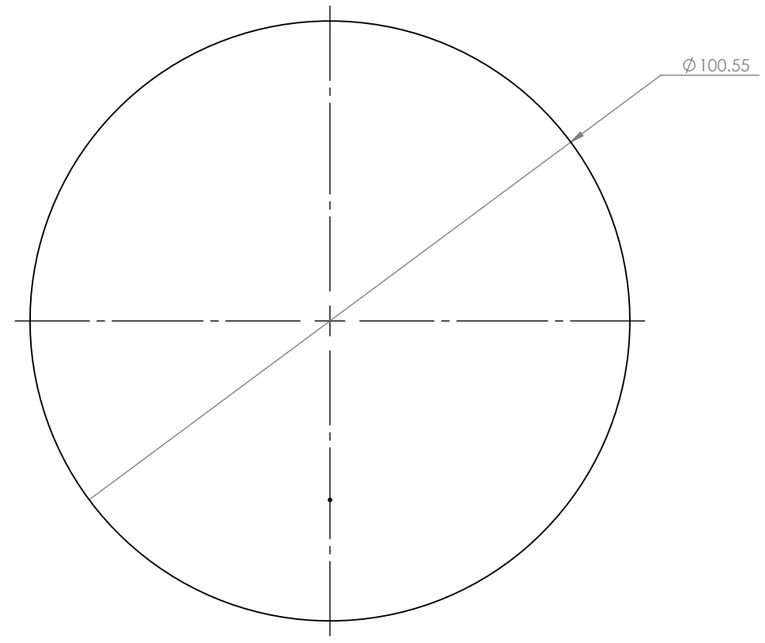




DETAIL A
SCALE 10 : 1



As Bent



Flat Pattern

Roll in over a 91.9 mm diameter core.
Sheet metal thickness is 0.5 mm.
Please fillet all edges to R0.1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 11:45:29 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

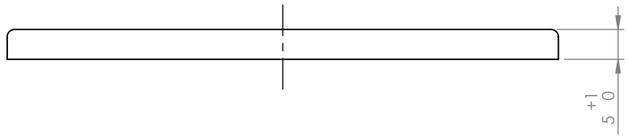
Displacer Isolating Plate

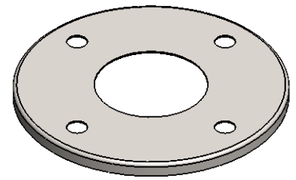
Material: 1.4401 (X5CrNiMo17-12-2)

DWG NO. A-DP-D-03-DISP_ISO_PLATE

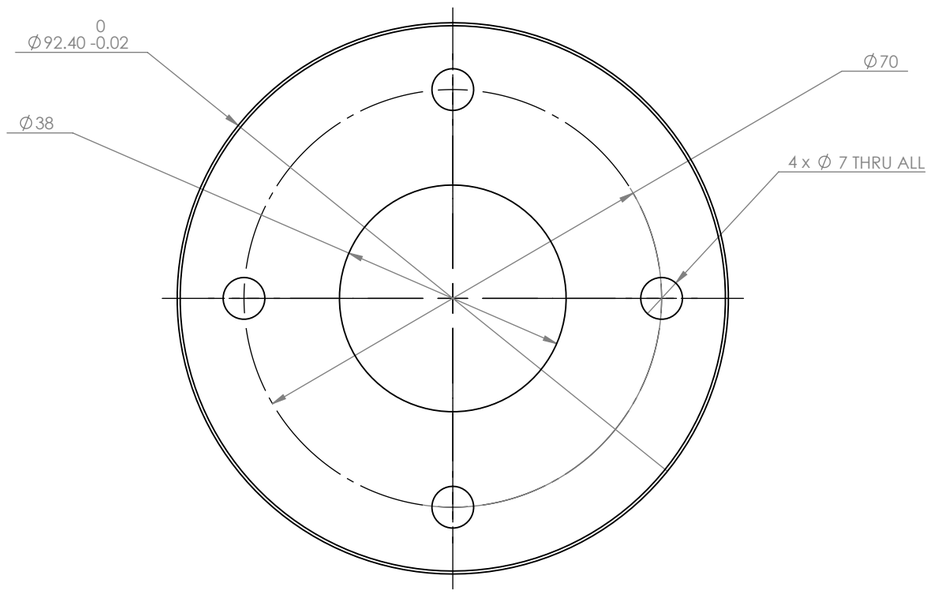
REVISION A

DRW File: 016_A-DP-D-03-DISP_ISO_PLATE | Project: GSE-1 | Mass: 31.52718 | SCALE:1:1 | SHEET 1 OF 1

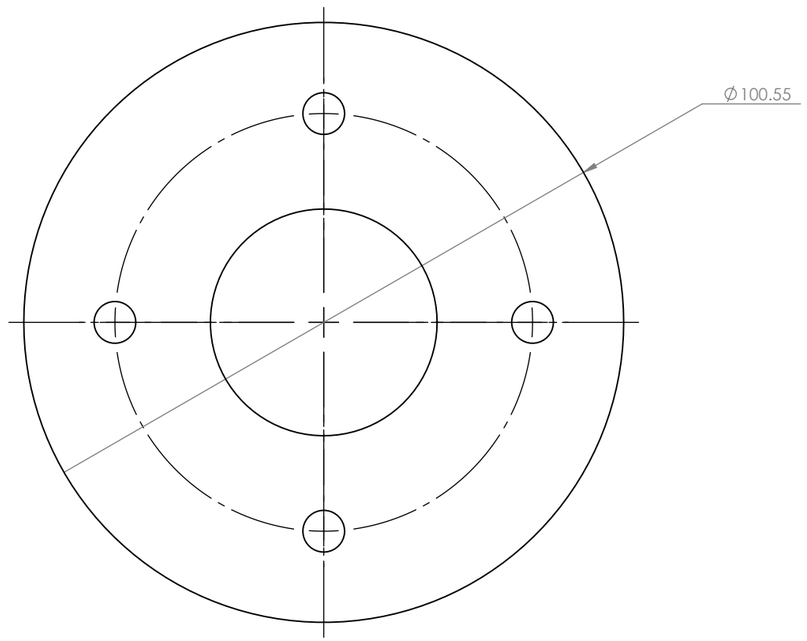




SCALE 1 : 2



As Bent



Flat Pattern

Sheet metal thickness: 0.8 mm
Roll in over a 90.8 mm diameter core.
Please fillet all edges to R0.1

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 11:40:11 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Displacer Mounting Plate

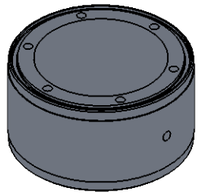
Material: 1.4571 (X6CrNiMoTi17-12-2)

DWG NO. **A-DP-D-04-DISP_MOUNT_PLATE**

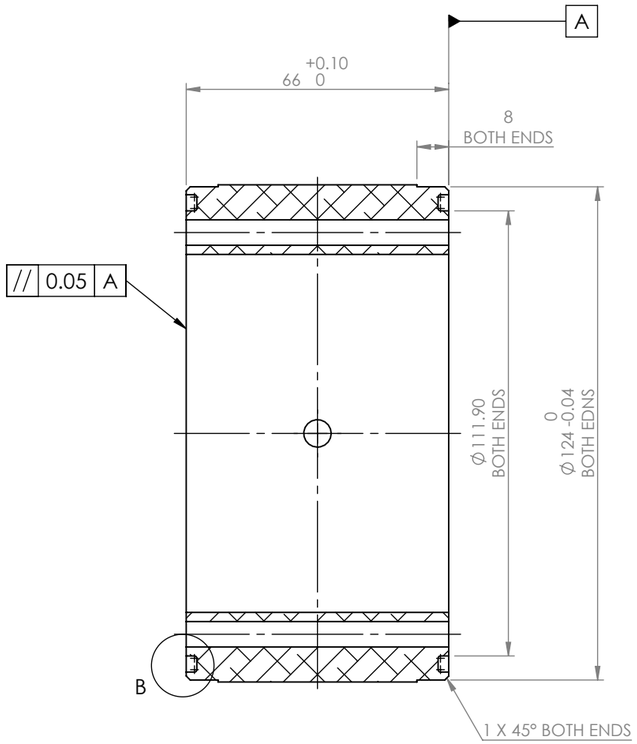
REVISION A

DRW File: 016_A-DP-D-04-DISP_MOUNT_PLATE | Project: **GSE-1** | Mass: 26.37576 | SCALE:1:1 | SHEET 1 OF 1

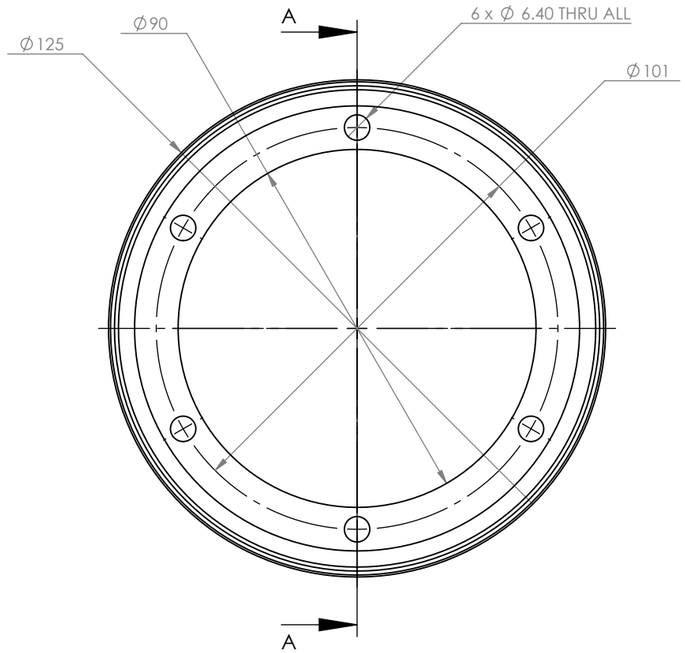
A Other Parts

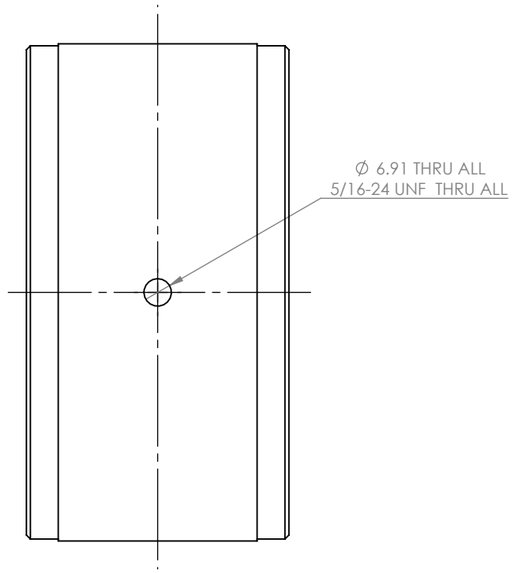


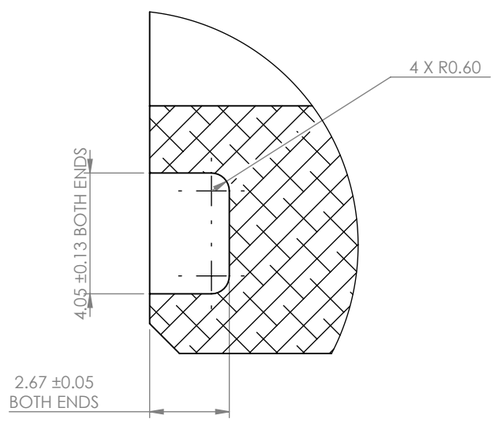
SCALE 1 : 4



SECTION A-A







DETAIL B
SCALE 5 : 1

O-Ring Information
Size: 3.5X113mm
Hi-Tech Seals Part#: 35113
Material: Nitrile

General surface finish: Rz 10

O-ring groove surface finish Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Wednesday, April 13, 2016 2:26:39 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

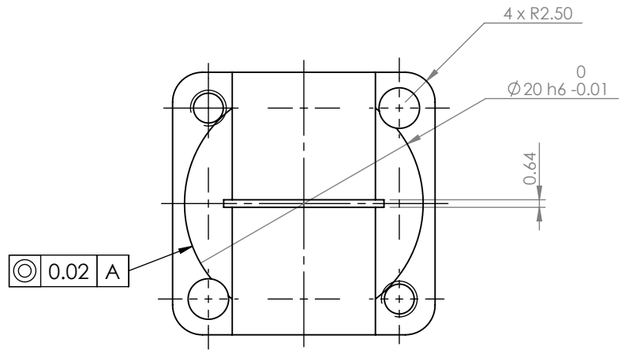
Displacer Cylinder Foot

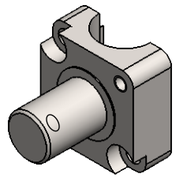
Material: 6061-T6 (SS)

DWG NO. A-ZZ-Z-01-FOOT

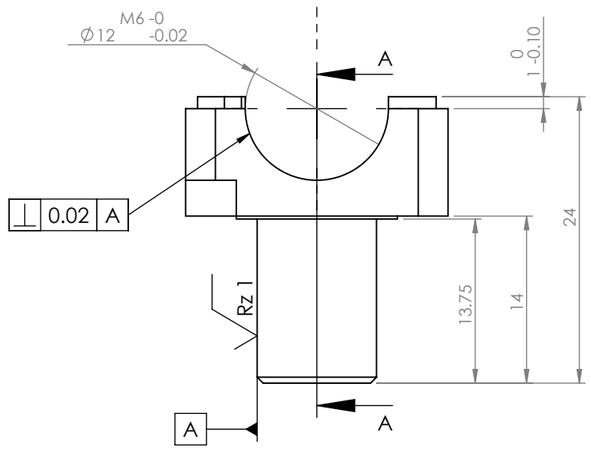
REVISION A

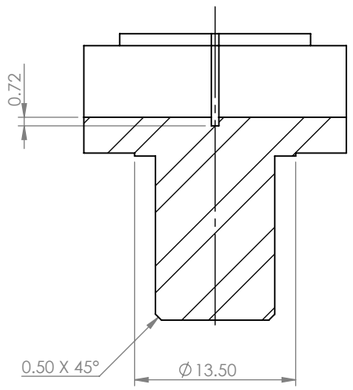
DRW File: 018_A-ZZ-Z-01-FOOT | Project: GSE-1 | Mass: 986.55418 | SCALE:2:3 | SHEET 1 OF 1



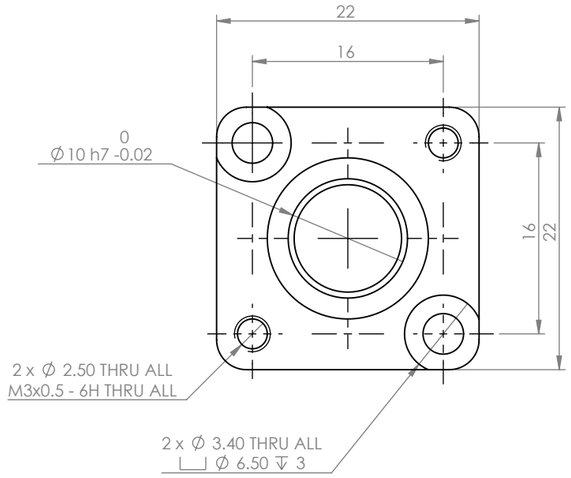


SCALE 1 : 1





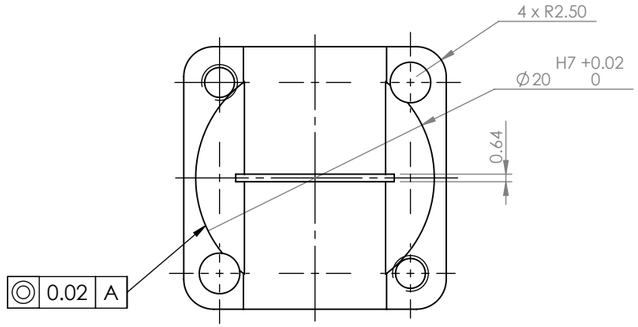
SECTION A-A

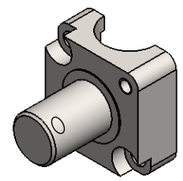


Please fillet/chamfer all edges to R0.1

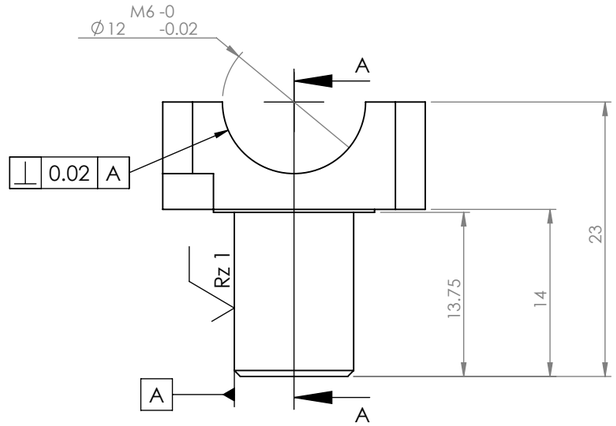
General surface finish: √ Rz 10

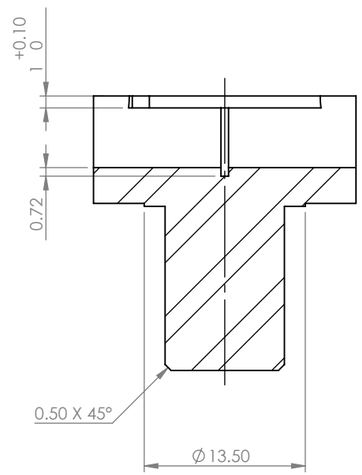
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS TOLERANCES: LINEAR: 0.1 ANGULAR: 1.0°	**NOBES RESEARCH GROUP**		UASolve TEC Edmonton Department of Mechanical Engineering UNIVERSITY OF ALBERTA	
Comments Quantity: 1	DRAWN	**Jiacheng Yao**	TITLE:	
	SOLID by	Jiacheng Yao	**Crosshead Right**	
	CHK'D	NOT CHECKED		
	APPV'D	NOT APPROVED		
Friday, April 15, 2016 10:14:50 AM	Material: AISI 304		DWG NO. **A-ZZ-Z-02-R_CROSSHEAD**	REVISION A
DO NOT SCALE DRAWING	DRW File: 018_A-ZZ-Z-02-R_CROSSHEAD_	Project: **GSE-1**	Mass: 30.91090	SCALE:2:1 / SHEET 1 OF 1



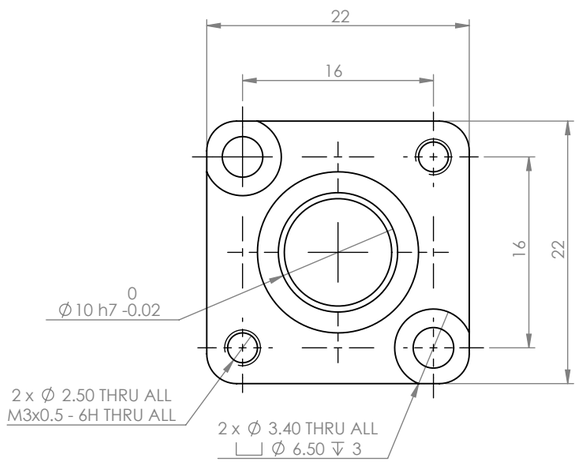


SCALE 1 : 1





SECTION A-A



Please fillet/chamfer All Edges R0.1mm

General surface finish: √Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 10:27:41 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

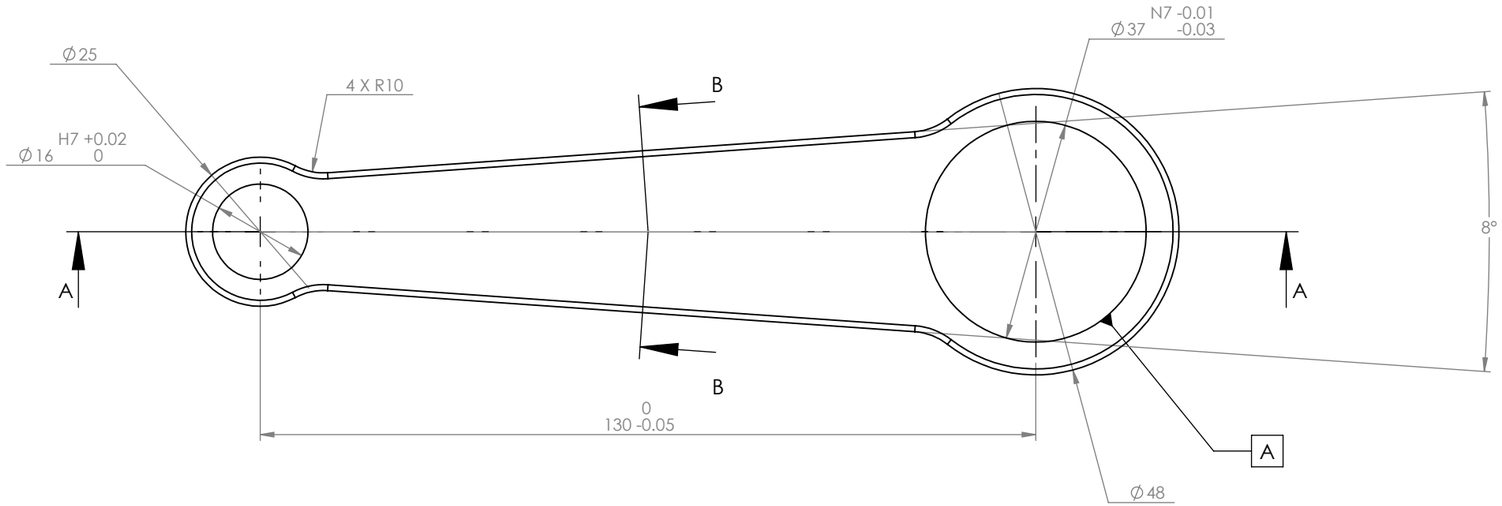
DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

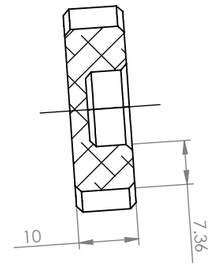
TITLE:

Crosshead Left

Material:	DWG NO.	REVISION
AISI 304	**A-ZZ-Z-03-L_CROSSHEAD**	A

DRW File: 018_A-ZZ-Z-03-L_CROSSHEAD | Project: **GSE-1** | Mass: 29.50406 | SCALE:2:1 | SHEET 1 OF 1

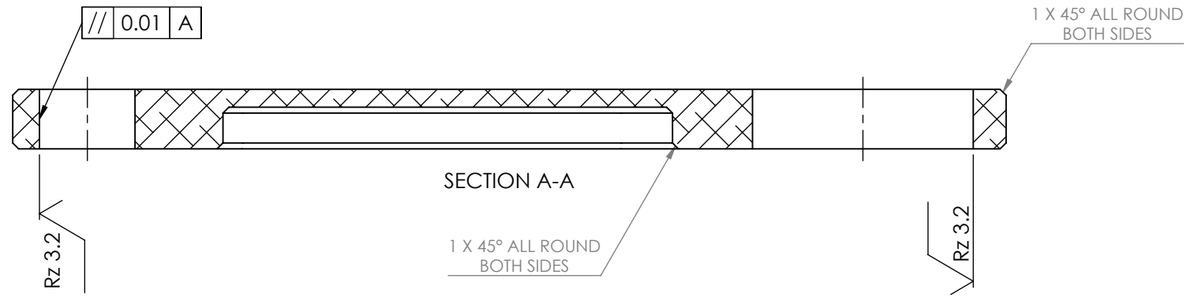


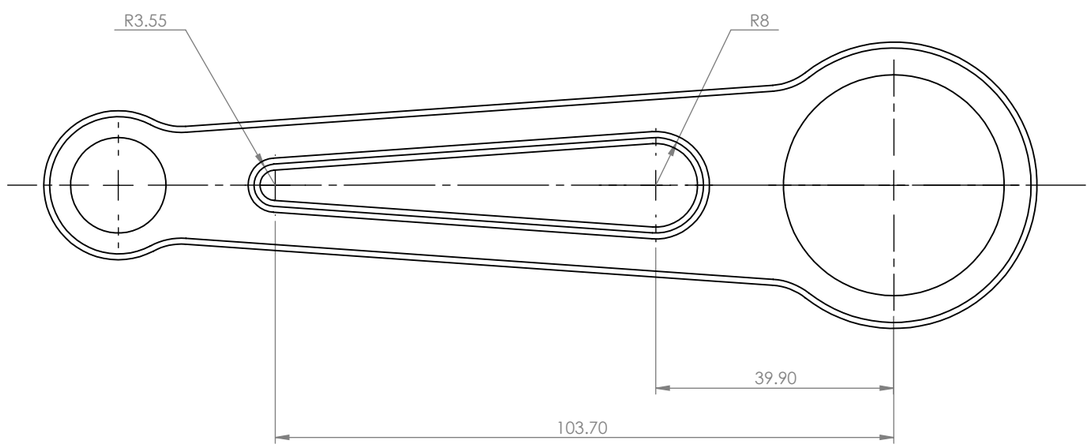


SECTION B-B



SCALE 1 : 3





Please fillet/chamfer all edges to R1

General surface finish: Rz 10

Maximum deviation between pair is 0.01 mm.

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, January 15, 2018 10:20:44 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

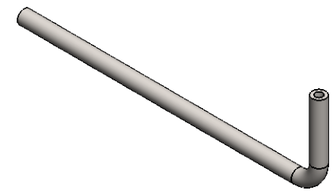
DISPLACER PISTON CONNECTING ROD

Material: 7075-T6, Plate (SS)

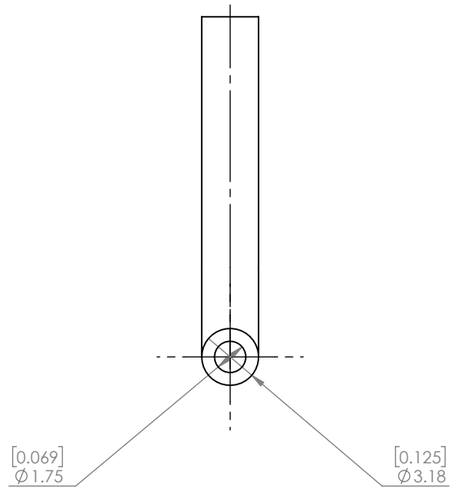
DWG NO. A-ZZ-Z-04-DISP_PISTON_ROD

REVISION A

DRW File: 018_A-ZZ-Z-04-DISP_PISTON_ROD | Project: GSE-1 | Mass: 83.28454 | SCALE:1:1 | SHEET 1 OF 1



SCALE 1 : 1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 12:25:58 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

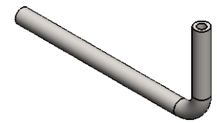
Working Gas Bypass Tube 1

Material: AISI 304

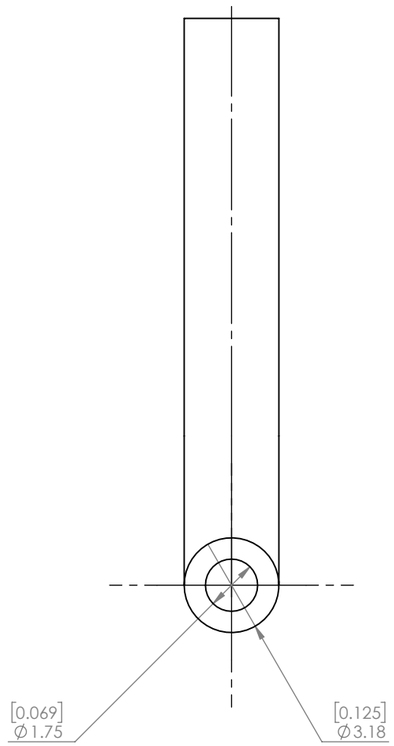
DWG NO. **A-ZZ-Z-09-BYPASS_TUBE_1**

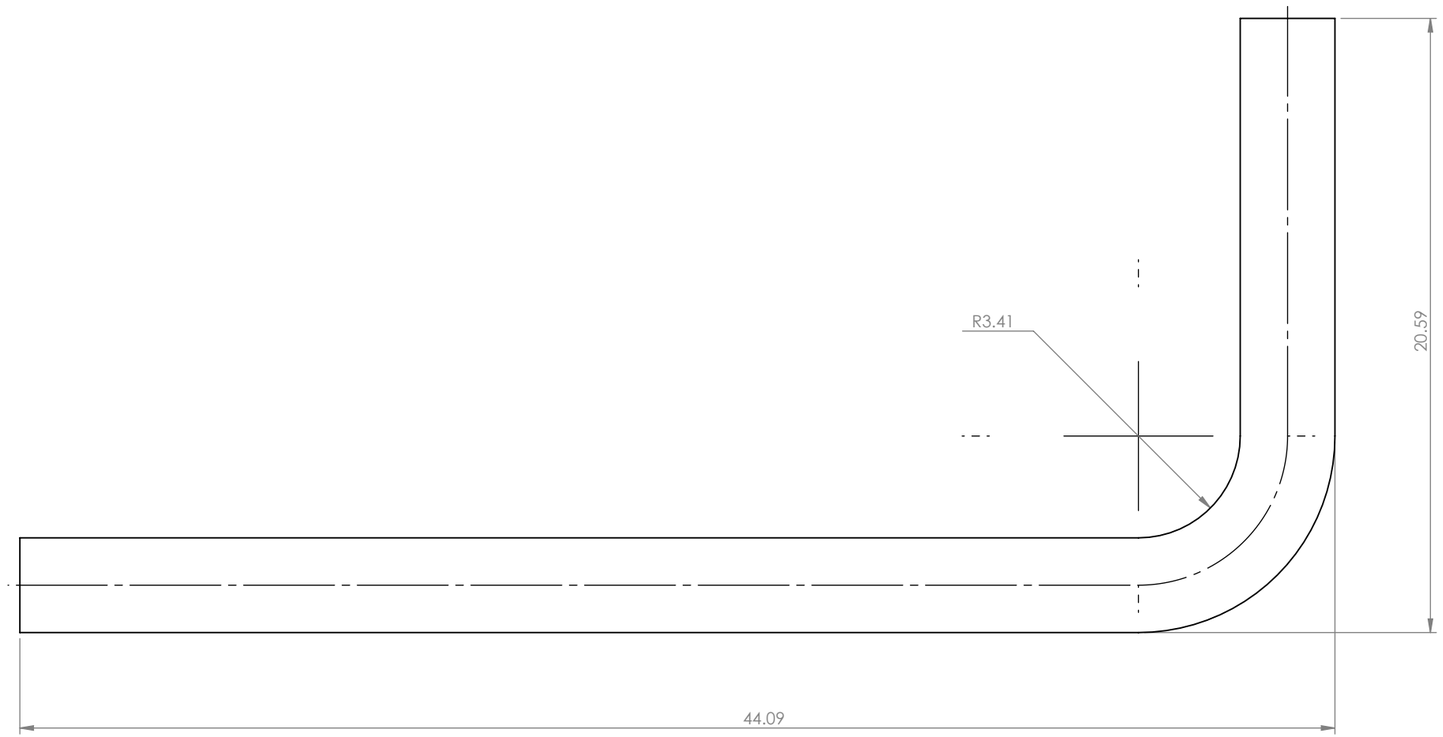
REVISION A

DRW File: 018_A-ZZ-Z-09-BYPASS_TUBE_1 | Project: **GSE-1** | Mass: 3.82993 | SCALE:3:1 | SHEET 1 OF 1



SCALE 1 : 1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Monday, April 11, 2016 12:22:22 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

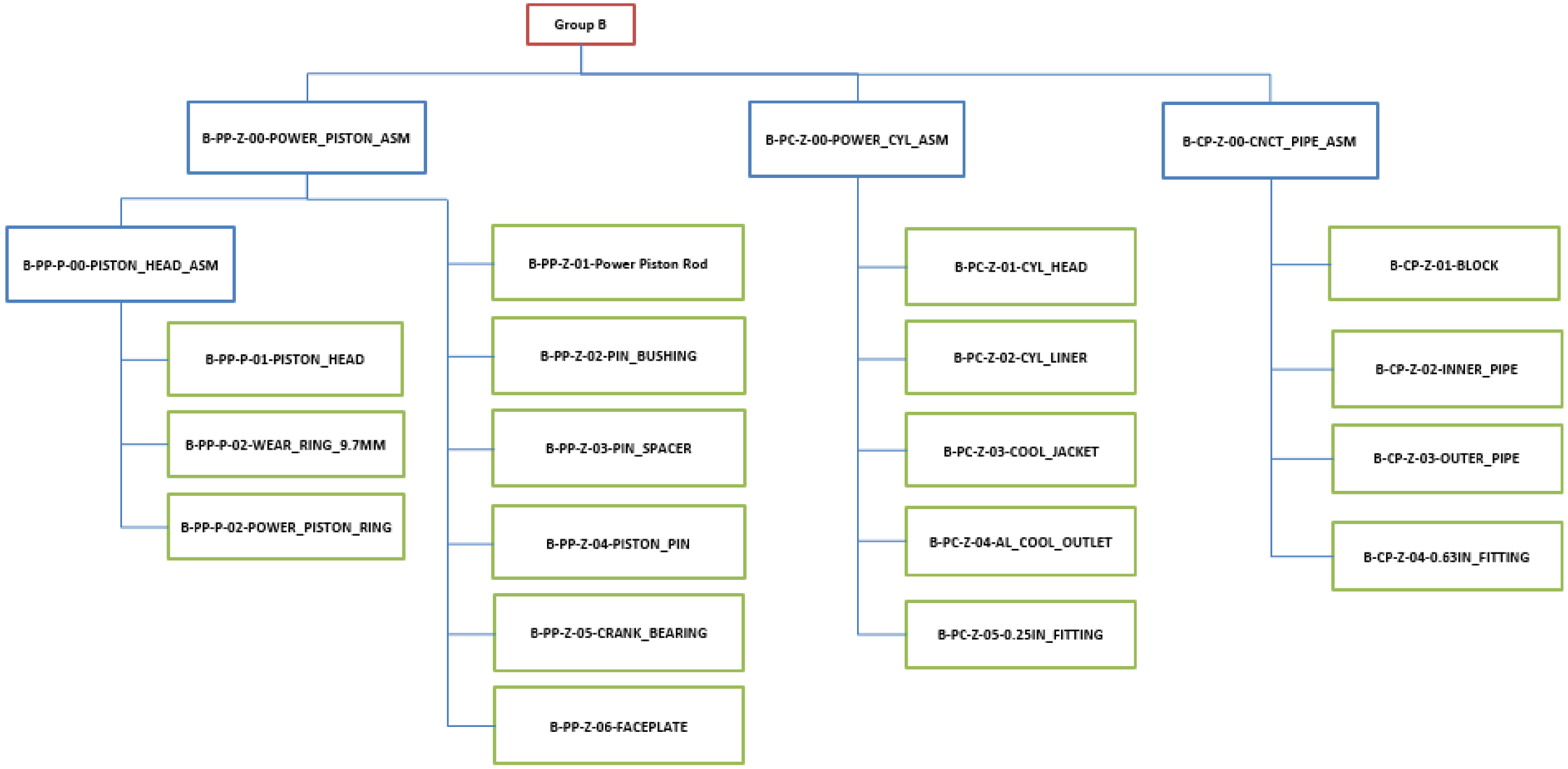
Working Gas Bypass Tube 2

Material: AISI 304

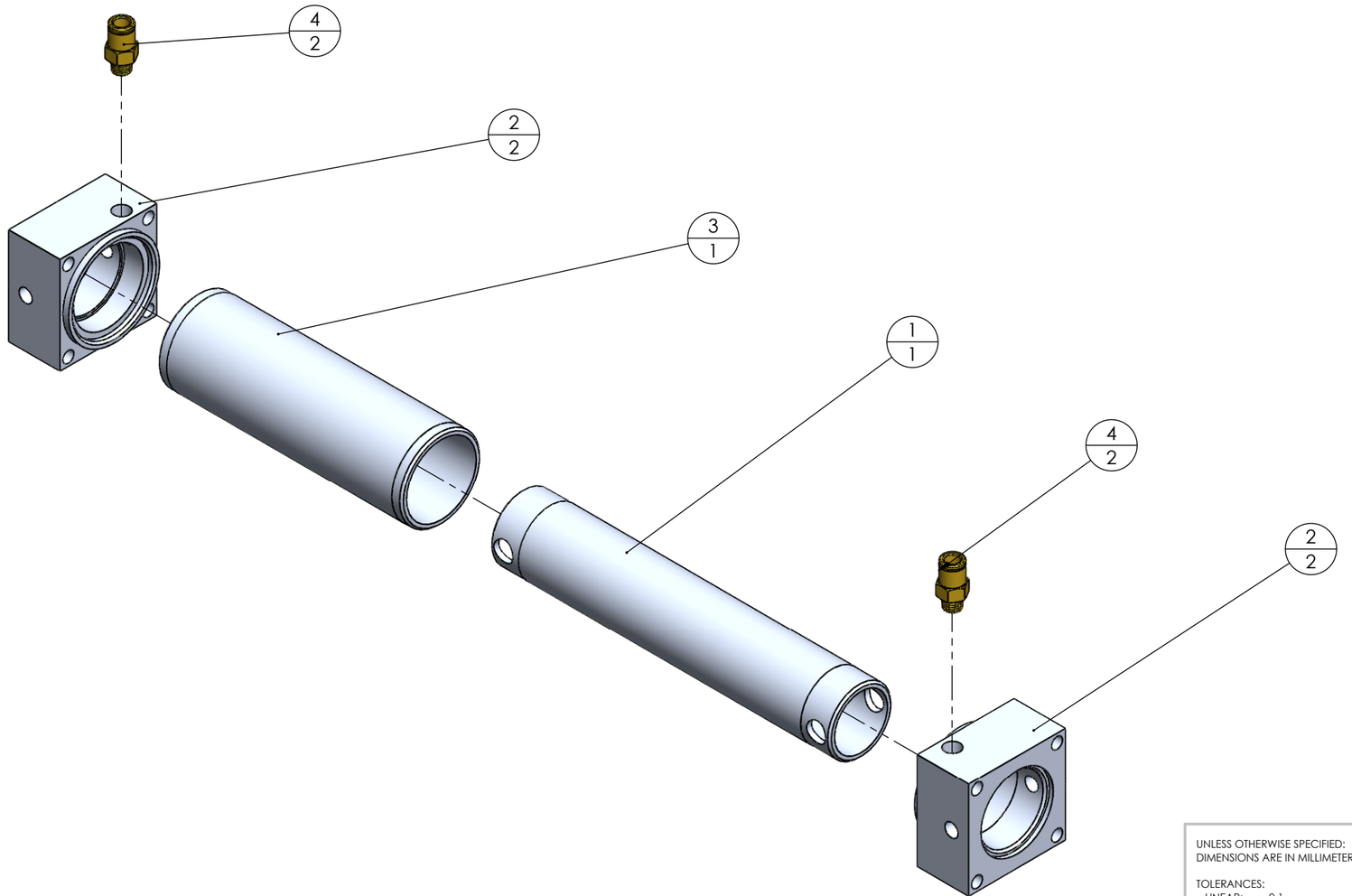
DWG NO. A-ZZ-Z-10-BYPASS_TUBE_2

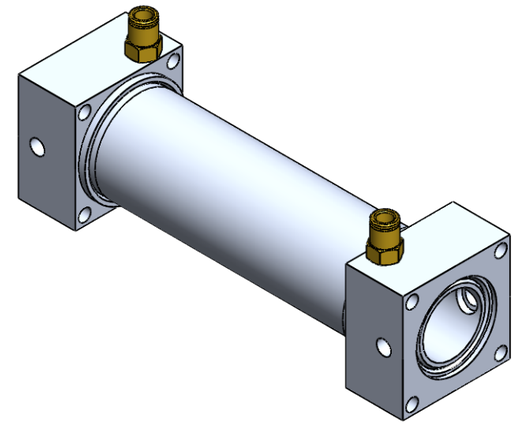
REVISION A

DRW File: 018_A-ZZ-Z-10-BYPASS_TUBE_2 | Project: GSE-1 | Mass: 2.61728 | SCALE:5:1 | SHEET 1 OF 1


Group B
B-PP-Z-00-POWER_PISTON_ASM
B-PP-P-00-PISTON_HEAD_ASM
B-PP-P-01-PISTON_HEAD
B-PP-P-02-WEAR_RING_9.7MM
B-PP-P-02-POWER_PISTON_RING
B-PP-Z-01-Power Piston Rod
B-PP-Z-02-PIN_BUSHING
B-PP-Z-03-PIN_SPACER
B-PP-Z-04-PISTON_PIN
B-PP-Z-05-CRANK_BEARING
B-PP-Z-06-FACEPLATE
B-PC-Z-00-POWER_CYL_ASM
B-PC-Z-01-CYL_HEAD
B-PC-Z-02-CYL_LINER
B-PC-Z-03-COOL_JACKET
B-PC-Z-04-AL_COOL_OUTLET
B-PC-Z-05-0.25IN_FITTING
B-CP-Z-00-CNCT_PIPE_ASM
B-CP-Z-01-BLOCK
B-CP-Z-02-INNER_PIPE
B-CP-Z-03-OUTER_PIPE
B-CP-Z-04-0.63IN_FITTING

ITEM NO.	DRW NUMBER	Material	QTY.
1	B-CP-Z-02-INNER_PIPE	6061-O (SS)	1
2	B-CP-Z-01-BLOCK	6061-O (SS)	2
3	B-CP-Z-03-OUTER_PIPE	6061-O (SS)	1
4	B-CP-Z-04-0.63IN_FITTING		2





Assembly Notes: Leave extra material on sealing faces. Machine sealing faces to final tolerance during engine assembly.
Epoxy glue on outside edge of cooler pipe and inner surface of connecting pipe blocks.

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:37:34 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

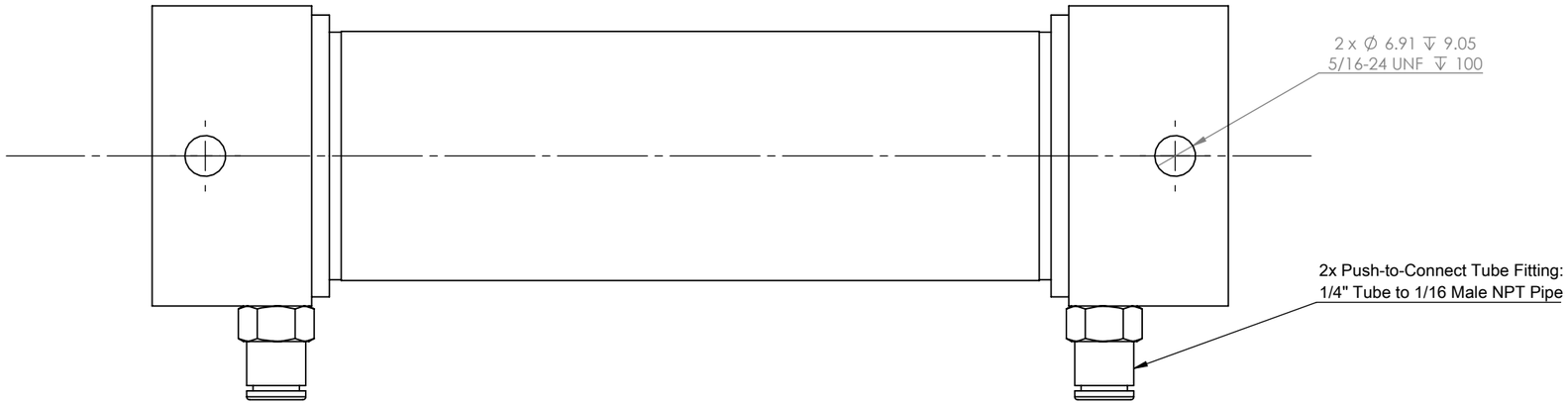
Connecting Pipe Assembly

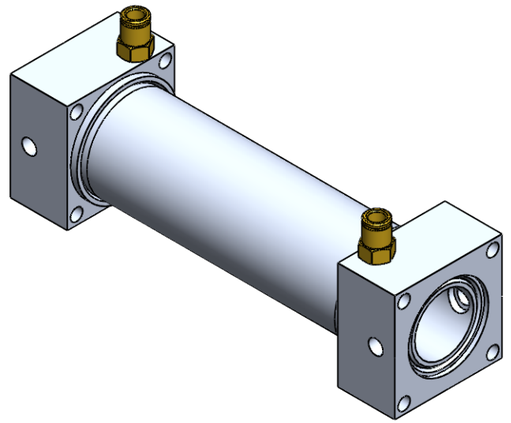
Material:

DWG NO. B-CP-Z-00-CNCT_PIPE_ASM

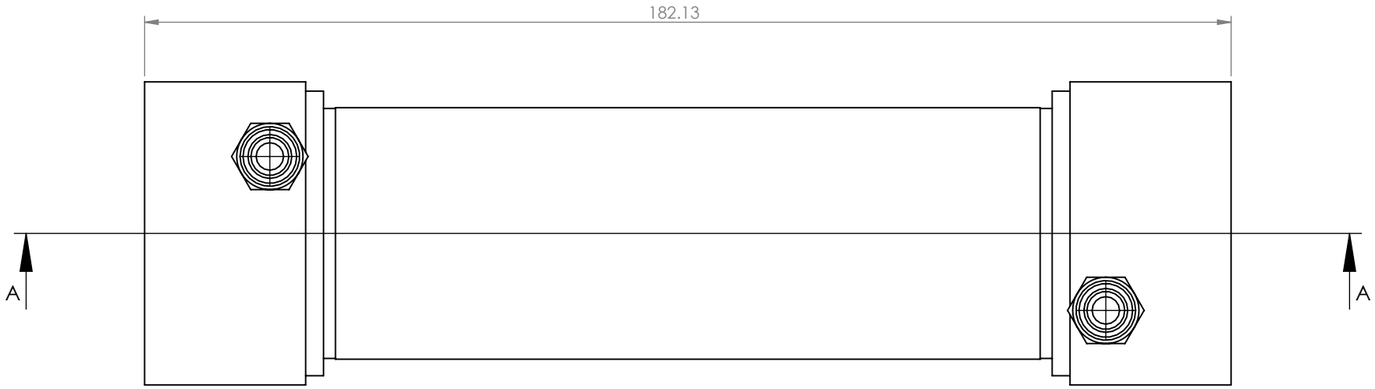
REVISION A

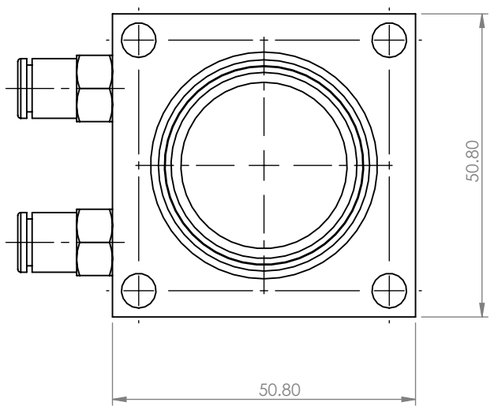
DRW File: 020_B-CP-Z-00-CNCT_PIPE_ASM | Project: GSE-1 | Mass: | SCALE:1:2 | SHEET 1 OF 2

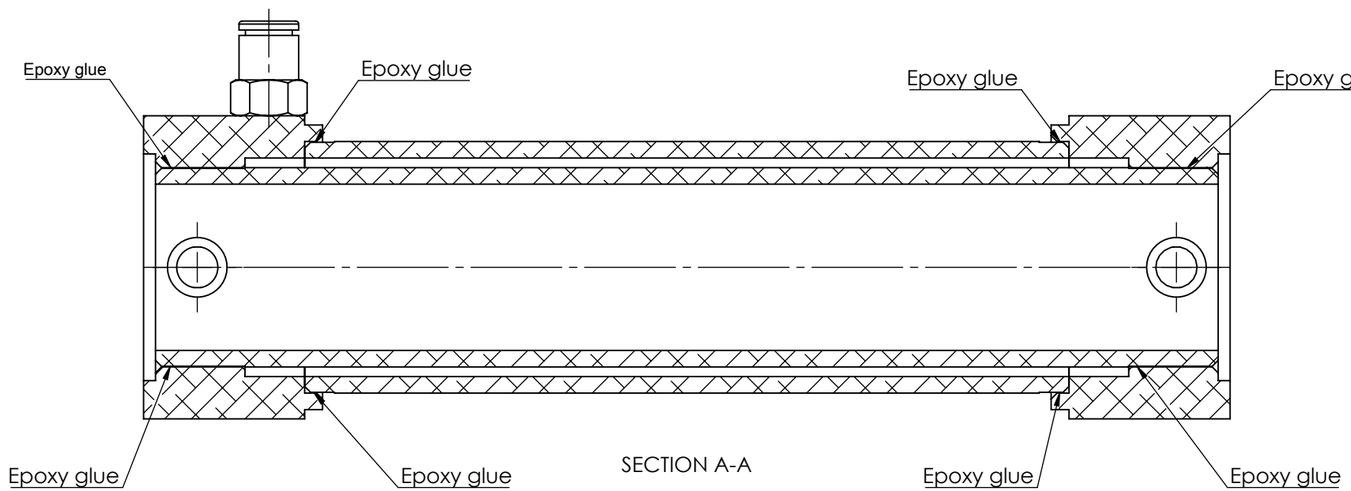




SCALE 1 : 2







SECTION A-A

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:37:34 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

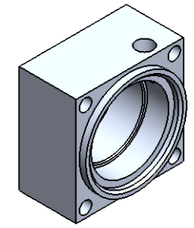
Connecting Pipe Assembly

Material:

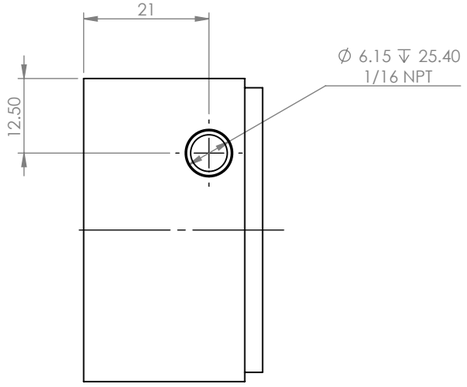
DWG NO. **B-CP-Z-00-CNCT_PIPE_ASM**

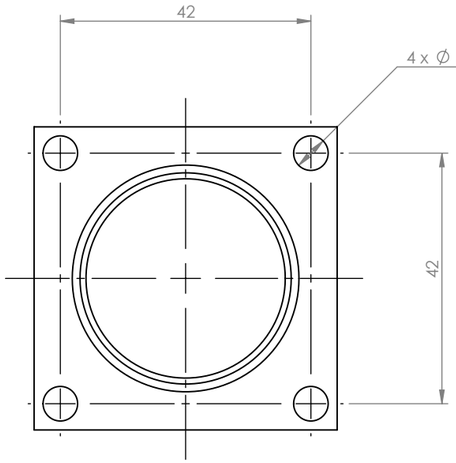
REVISION A

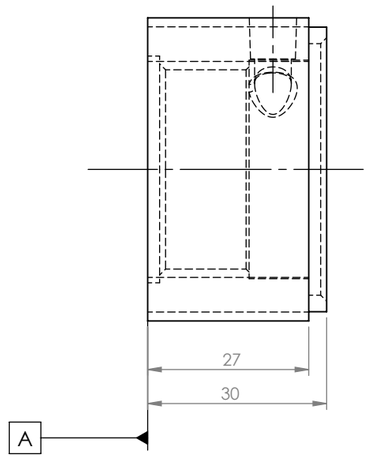
DRW File: 020_B-CP-Z-00-CNCT_PIPE_ASM | Project: **GSE-1** | Mass: | SCALE:1:1 | SHEET 2 OF 2

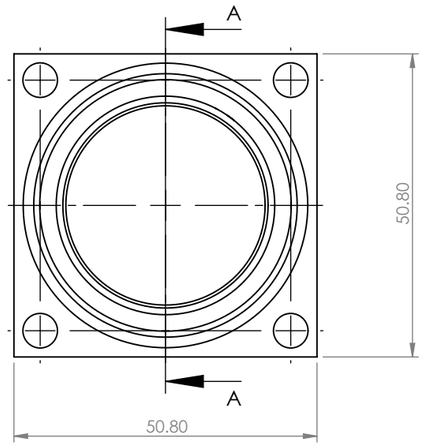


SCALE 1 : 2

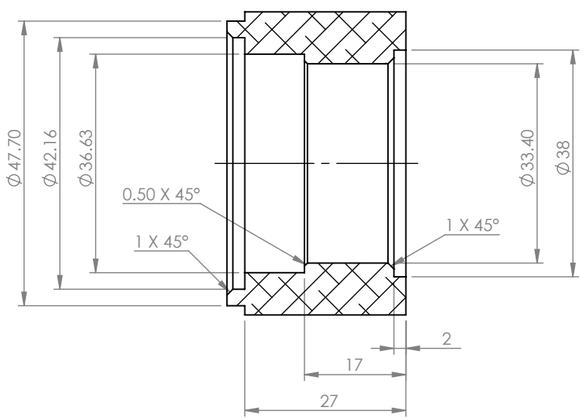








SECTION A-A



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 9008K53

Quantity: 2

Wednesday, April 13, 2016 2:46:12 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

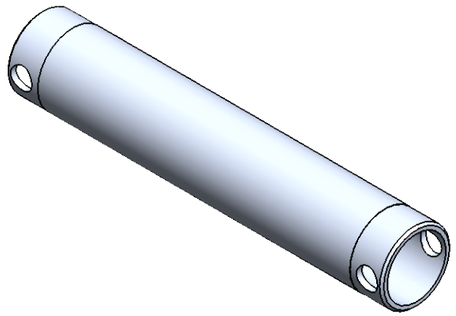
DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

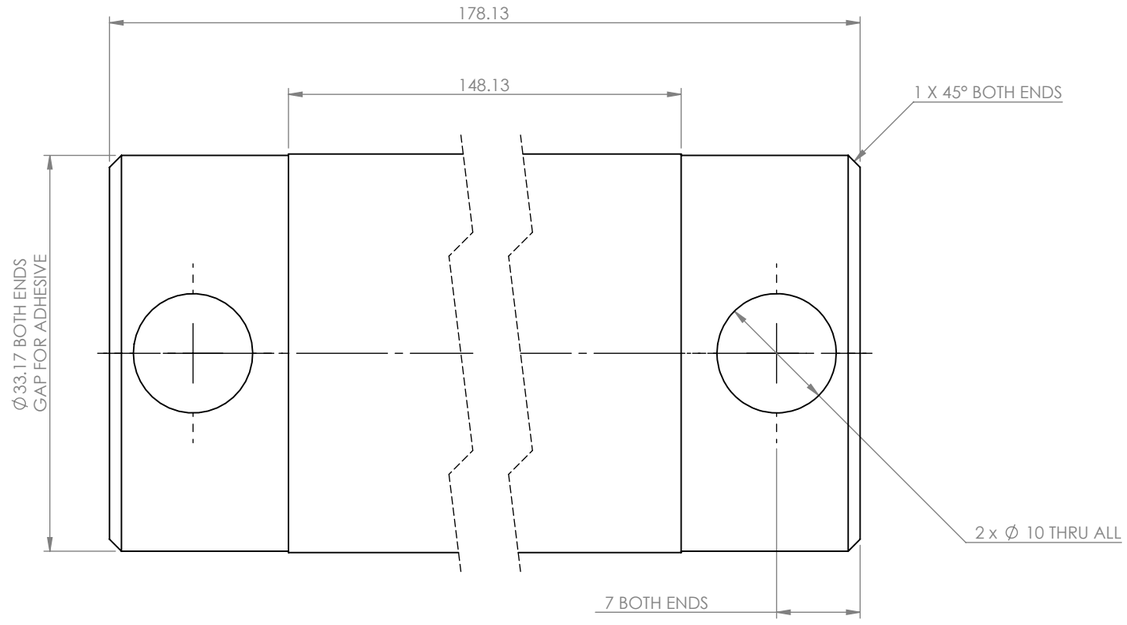
Connecting Pipe Block

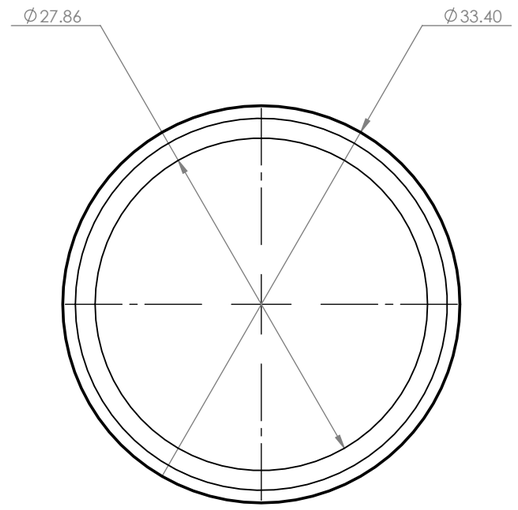
Material:	DWG NO.	REVISION
6061-O (SS)	**B-CP-Z-01-BLOCK**	A

DRW File: 020_B-CP-Z-01-BLOCK	Project: **GSE-1**	Mass: 111.82140	SCALE:1:1	SHEET 1 OF 1



SCALE 1 : 2





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
1" SCH 10 Pipe
Possible Supplier: McMaster-Carr 4561T311

Quantity: 1

Wednesday, April 13, 2016 2:33:47 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

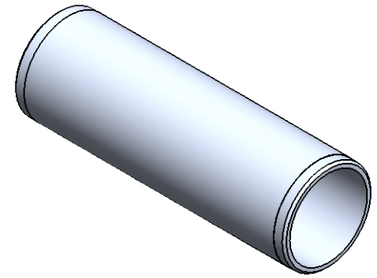
Inner Connecting Pipe

Material: 6061-O (SS)

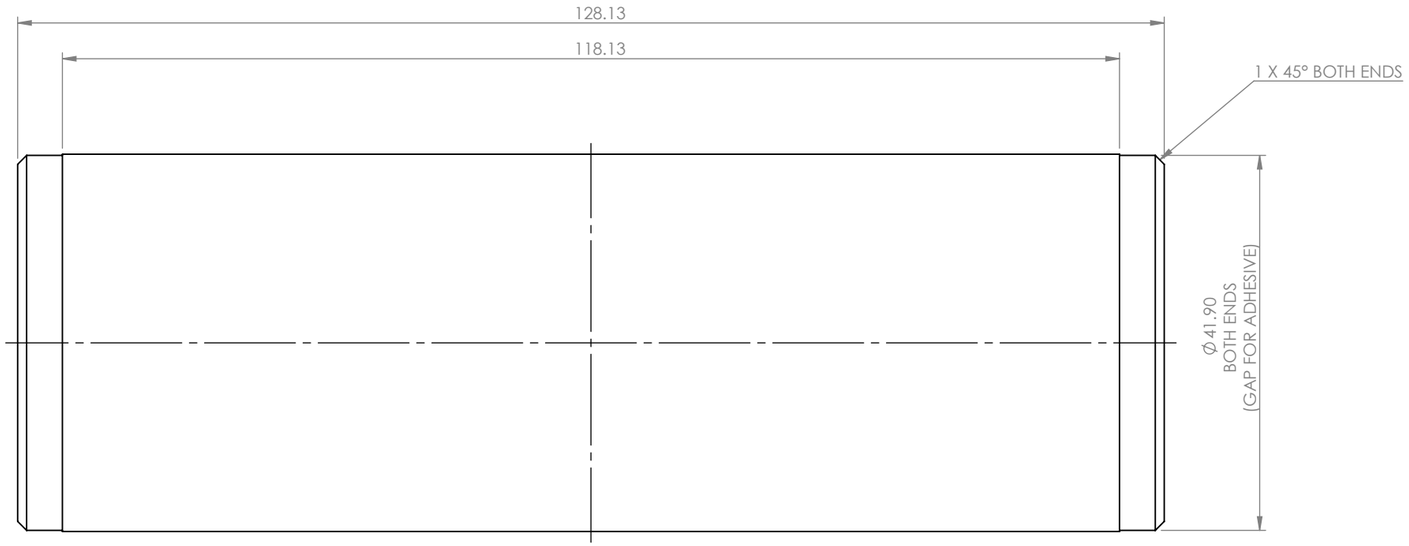
DWG NO. **B-CP-Z-02-INNER_PIPE**

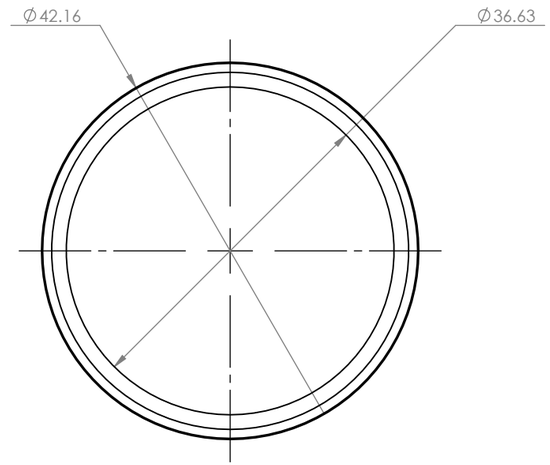
REVISION A

DRW File: 020_B-CP-Z-02-INNER_PIPE | Project: **GSE-1** | Mass: 124.65968 | SCALE:2:1 | SHEET 1 OF 1



SCALE 1 : 2





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
1 1/4" SCH 10 pipe
Possible Supplier: McMaster-Carr 4561T411

Quantity: 1

Wednesday, April 13, 2016 2:30:30 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Outer Connecting Pipe

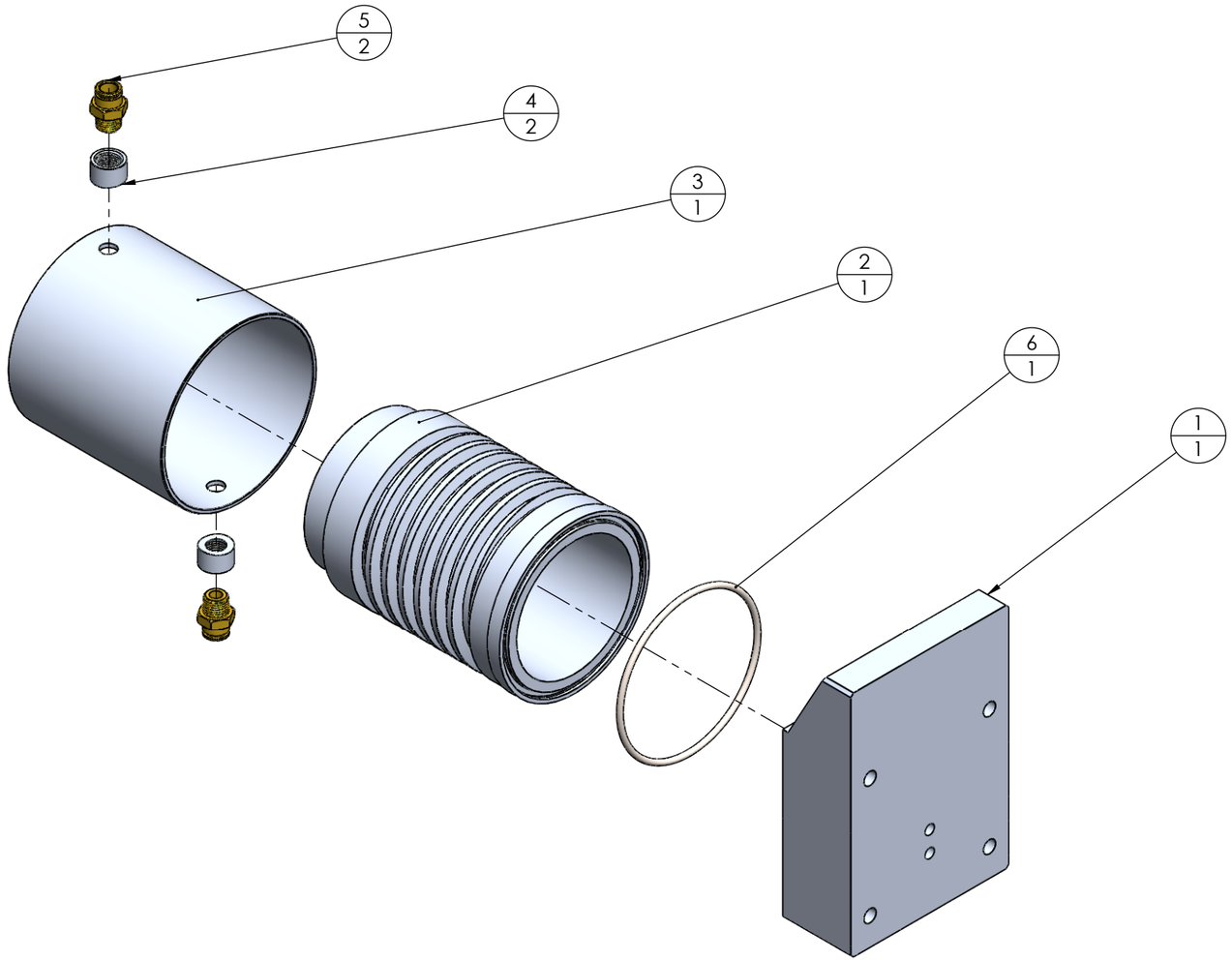
Material: 6061-O (SS)

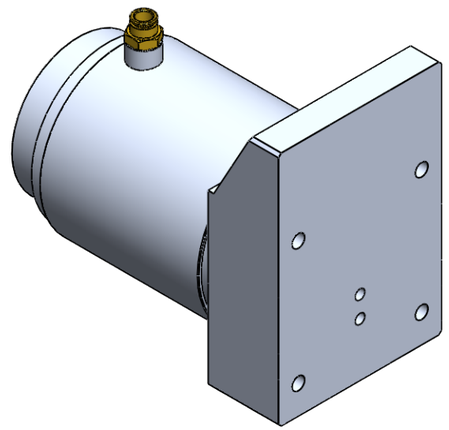
DWG NO. **B-CP-Z-03-OUTER_PIPE**

REVISION A

DRW File: 020_B-CP-Z-03-OUTER_PIPE | Project: **GSE-1** | Mass: 117.57288 | SCALE:1.5:1 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	B-PC-Z-01-CYI_HEAD	1
2	B-PC-Z-02-CYI_LINER	1
3	B-PC-Z-03-COOL_JACKET	1
4	B-PC-Z-04-AL_COOL_OUTLET	2
5	B-PC-Z-05-3_8ths_PUSH_TO_CONNECT_FITTING2	2
6	O-ring 97.5x3.55-A-ISO 3601-1	1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:29:25 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

Power Cylinder Assembly

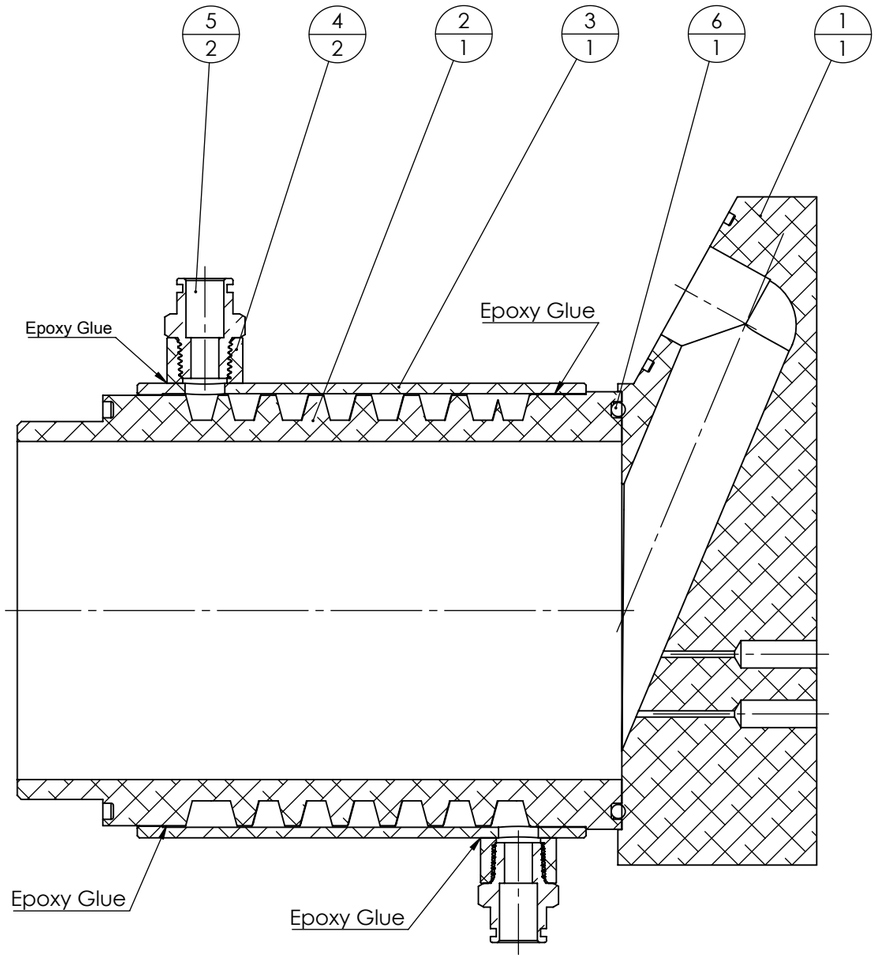
Material:

DWG NO. **B-PC-Z-00-POWER_CYL_ASM**

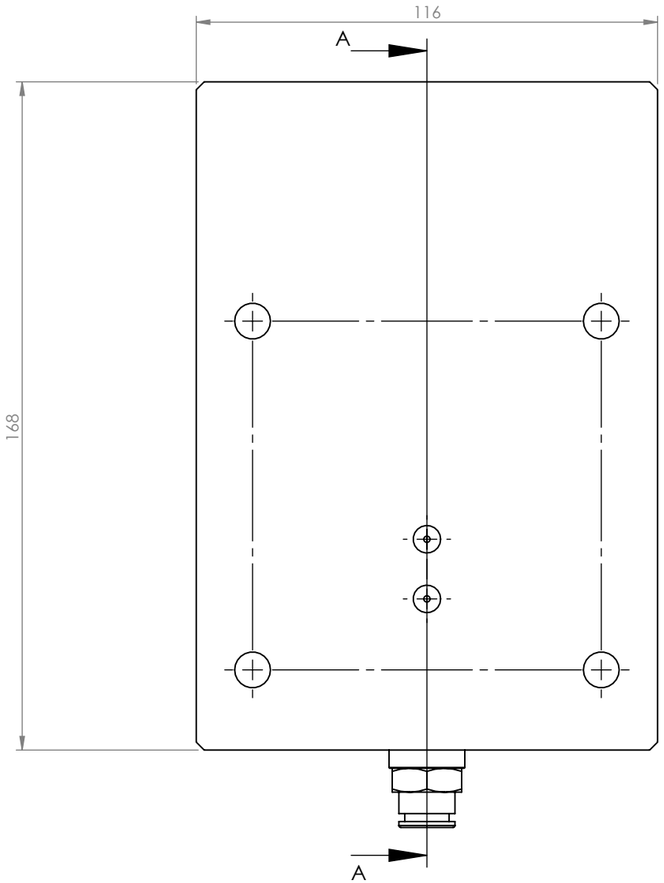
REVISION A

DRW File: 022_B-PC-Z-00-POWER_CYL_ASM | Project: **GSE-1** | Mass: | SCALE:1:3 | SHEET 1 OF 2

ITEM NO.	DRW NUMBER	QTY.
1	B-PC-Z-01-CYI_HEAD	1
2	B-PC-Z-02-CYI_LINER	1
3	B-PC-Z-03-COOL_JACKET	1
4	B-PC-Z-04-AL_COOL_OUTLET	2
5	B-PC-Z-05-3_8ths_PUSH_TO_CONNECT_FITTING2	2
6	O-ring 97.5x3.55-A-ISO 3601-1	1



SECTION A-A



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:29:25 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

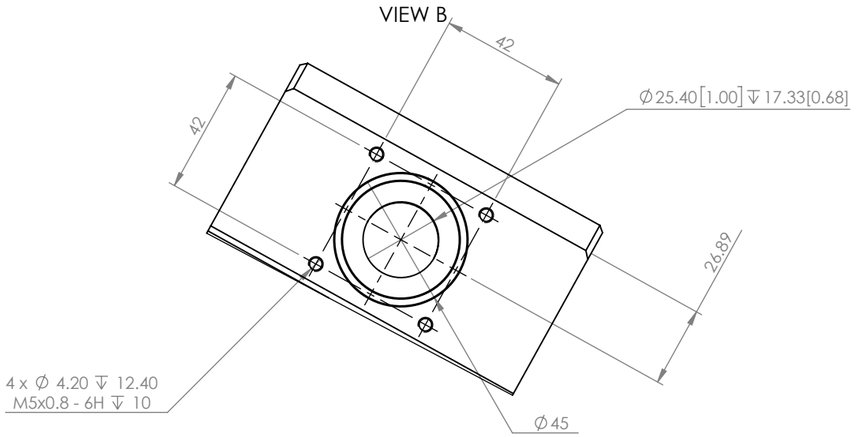
Power Cylinder Assembly

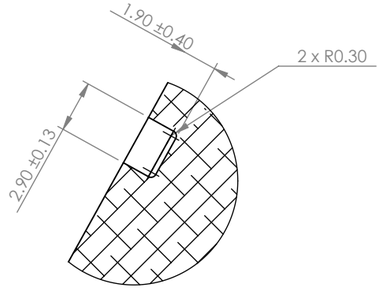
Material:

DWG NO. **B-PC-Z-00-POWER_CYL_ASM**

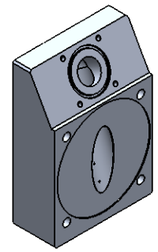
REVISION A

DRW File: 022_B-PC-Z-00-POWER_CYL_ASM | Project: **GSE-1** | Mass: | SCALE:2:3 | SHEET 2 OF 2

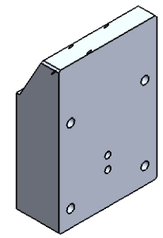




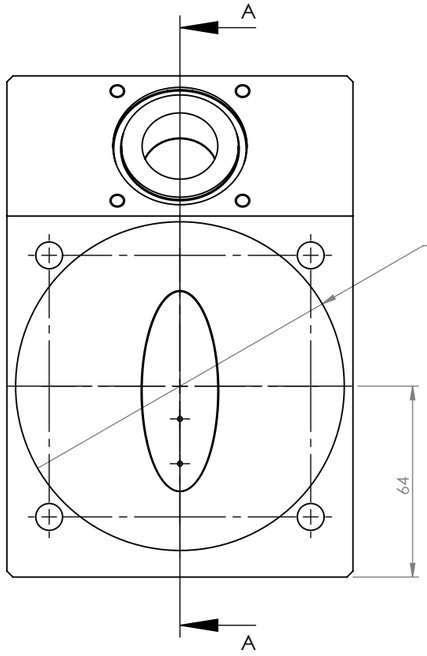
DETAIL C
SCALE 3 : 1

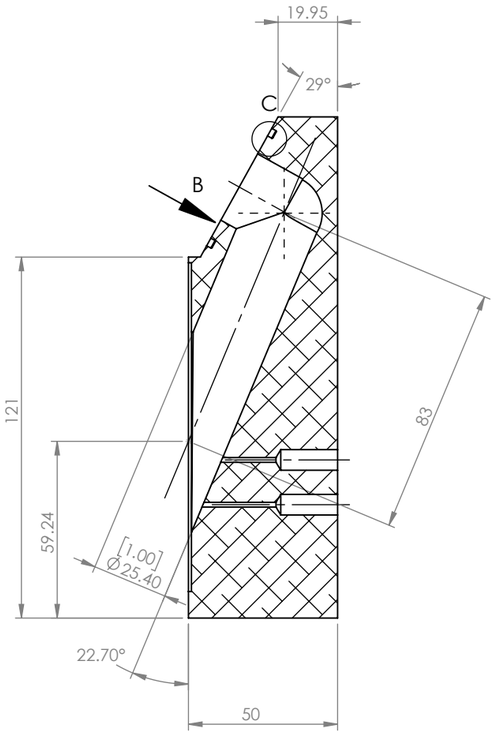


SCALE 1 : 5

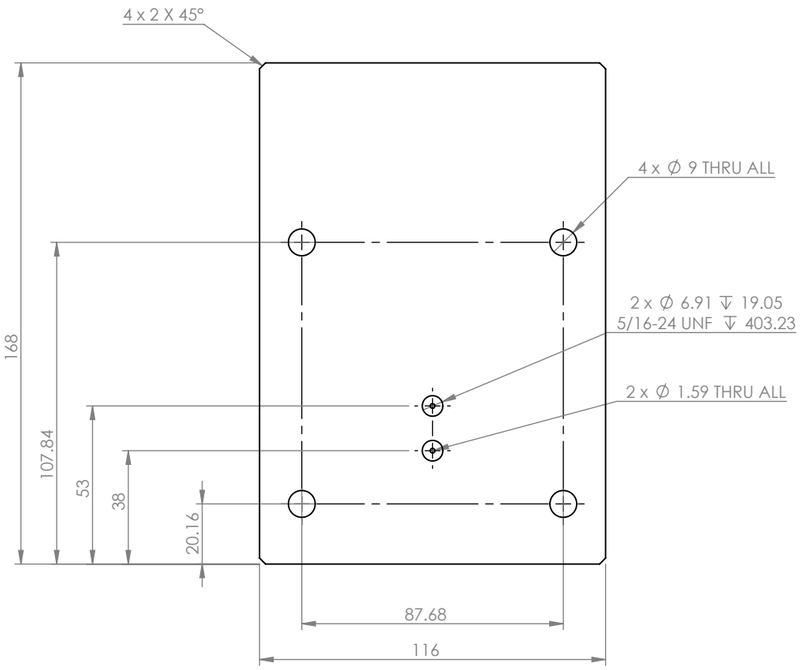


SCALE 1 : 5





SECTION A-A



Please fillet/chamfer all edges to R0.3

General surface finish: √ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:21:44 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

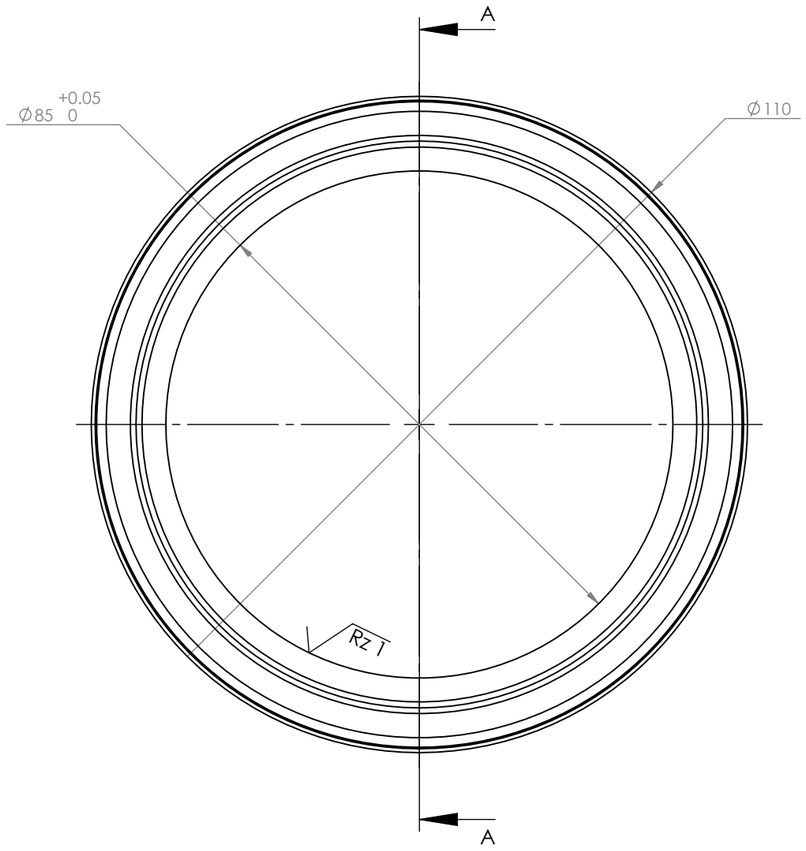
DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

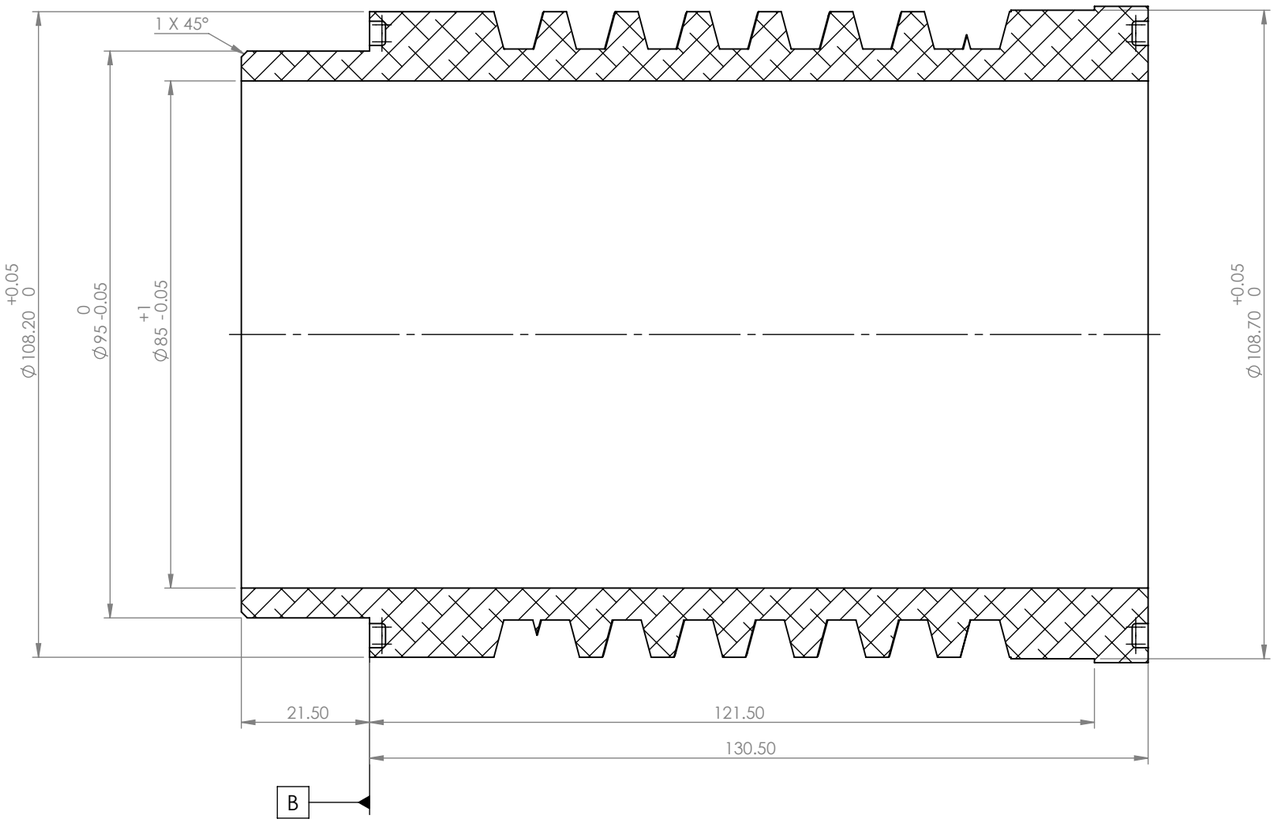
Displacer Piston Shell

Material:	DWG NO.	REVISION
6061-T6 (SS)	**A-DP-D-02-DISP_SHELL**	A

DRW File: 022_B-PC-Z-01-CYL_HEAD | Project: **GSE-1** | Mass: 2178.42960 | SCALE:1:2 | SHEET 1 OF 1



SECTION A-A

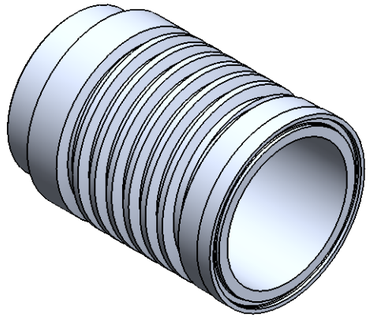


Please fillet/chamfer all edges to R0.3

General surface finish: √ Rz 10

O-ring groove surface finish: √ Rz 6.3

Bore surface finish: √ Rz 1



SCALE 1 : 3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 9056K49

Quantity: 1

Wednesday, April 13, 2016 3:27:56 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

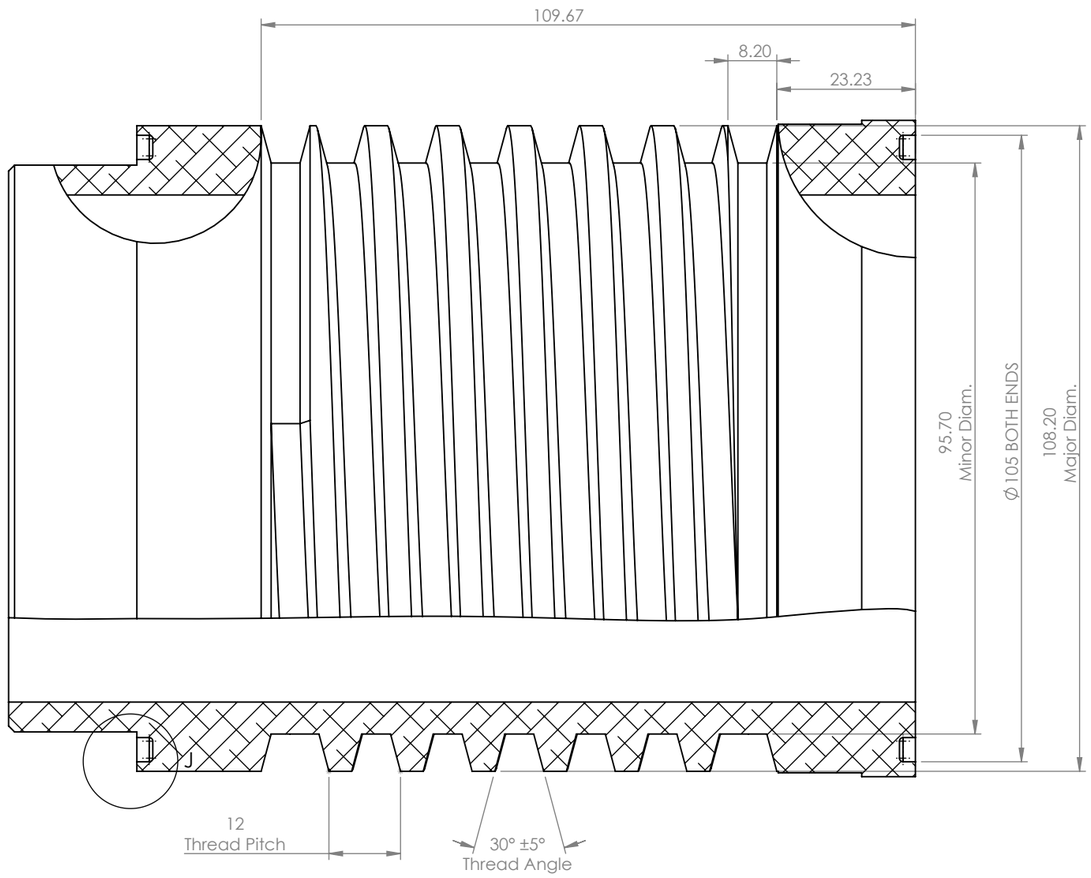
TITLE: **Cylinder Liner**

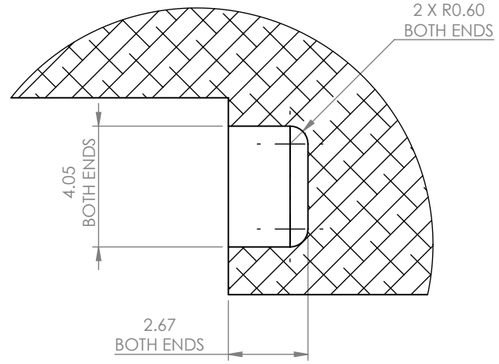
Material: 6061-T6 (SS)

DWG NO. **B-PC-Z-02-CYL_LINER**

REVISION A

DRW File: 022_B-PC-Z-02-CYL_LINER | Project: **GSE-1** | Mass: 1032.26952 | SCALE:1:1 | SHEET 1 OF 2





DETAIL J
SCALE 5 : 1

O-Ring Information
Material: Nitrile
Size: 98 x 3.5mm
Supplier: Hi-Tech Seals
Part #: MN7 35098

Please fillet/chamfer all edges to R0.3

General surface finish: $\sqrt{}$ Rz 10

O-ring groove surface finish: $\sqrt{}$ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 9056K49

Quantity: 1

Wednesday, April 13, 2016 3:27:56 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

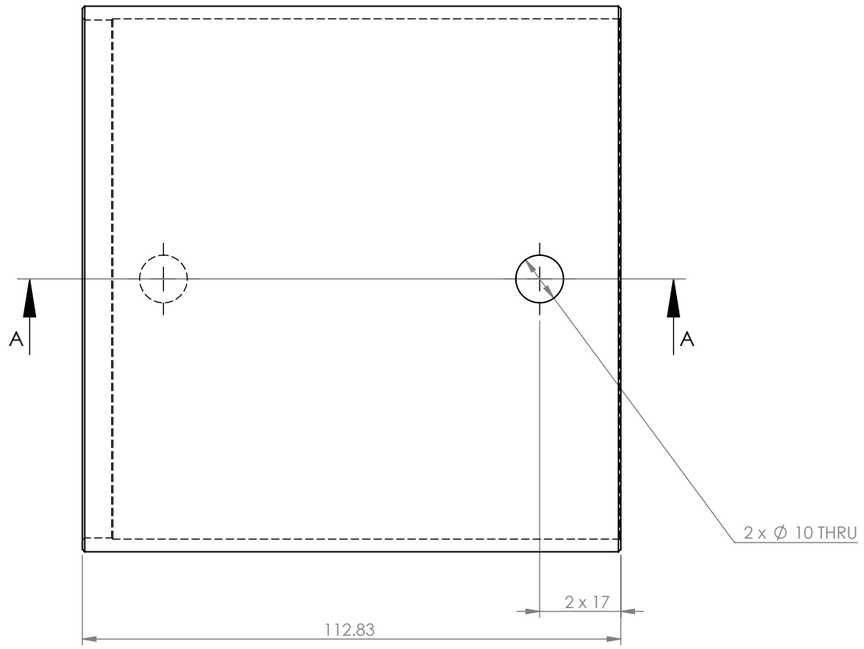
TITLE: **Cylinder Liner**

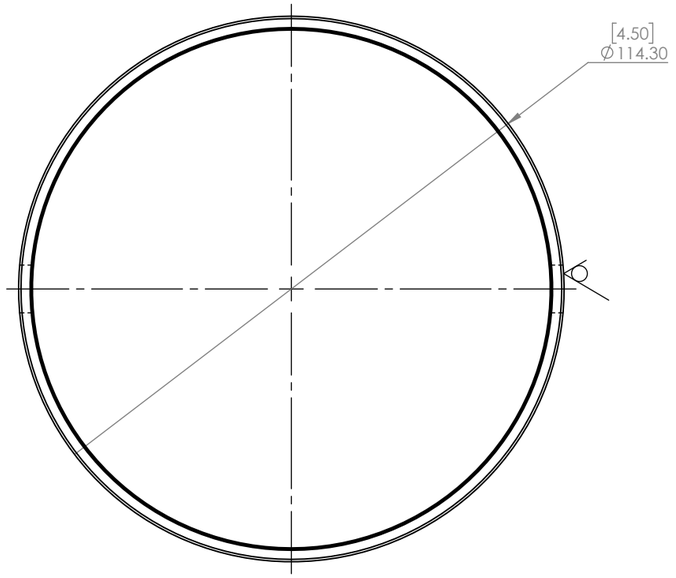
Material: 6061-T6 (SS)

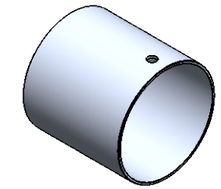
DWG NO. **B-PC-Z-02-CYL_LINER**

REVISION A

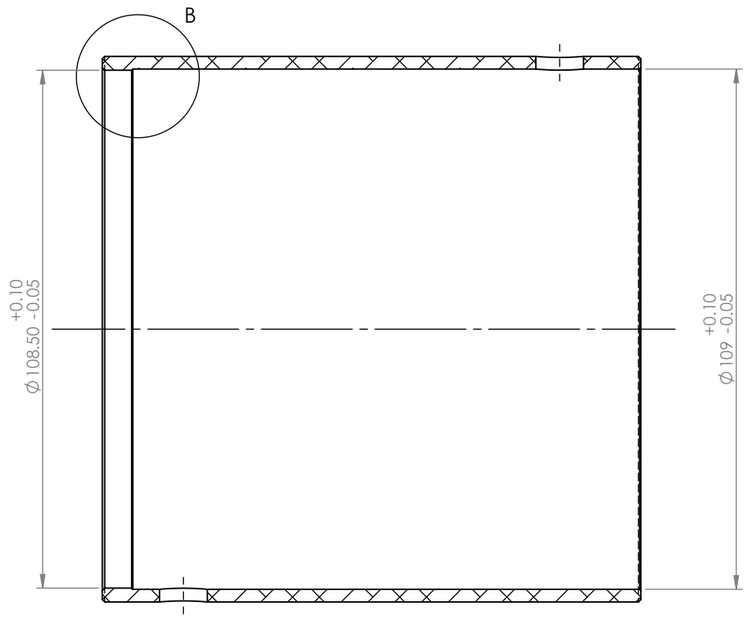
DRW File: 022_B-PC-Z-02-CYL_LINER | Project: **GSE-1** | Mass: 1032.26952 | SCALE:1:1 | SHEET 2 OF 2



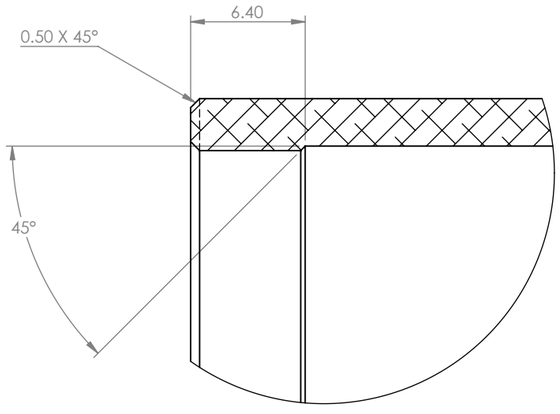




SCALE 1 : 5



SECTION A-A



DETAIL B
SCALE 3 : 1

Please fillet/chamfer all edges to R0.3

General Surface Finish $\sqrt{}$ Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Wednesday, April 13, 2016 3:18:28 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

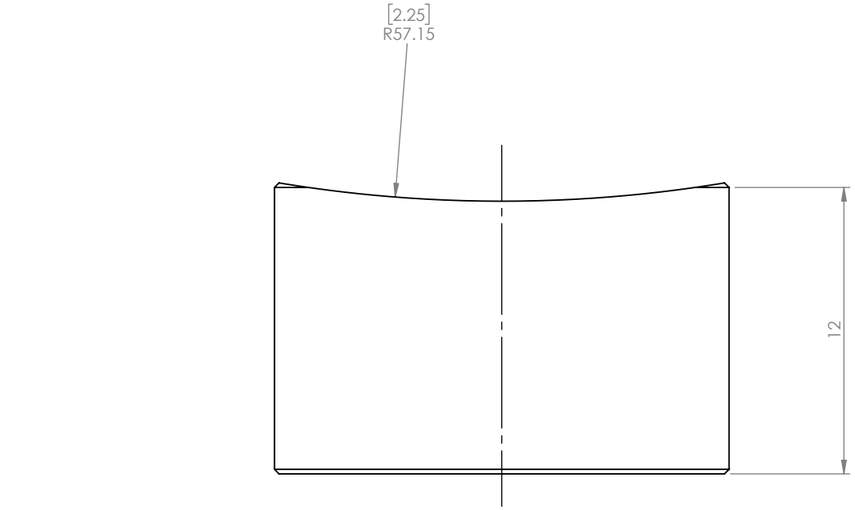
Power Piston Outer Cooling Jacket

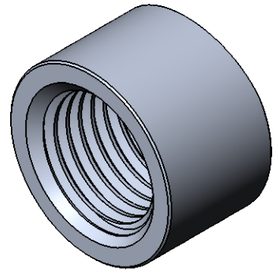
Material: 6061-T6 (SS)

DWG NO. **B-PC-Z-03-COOL_JACKET**

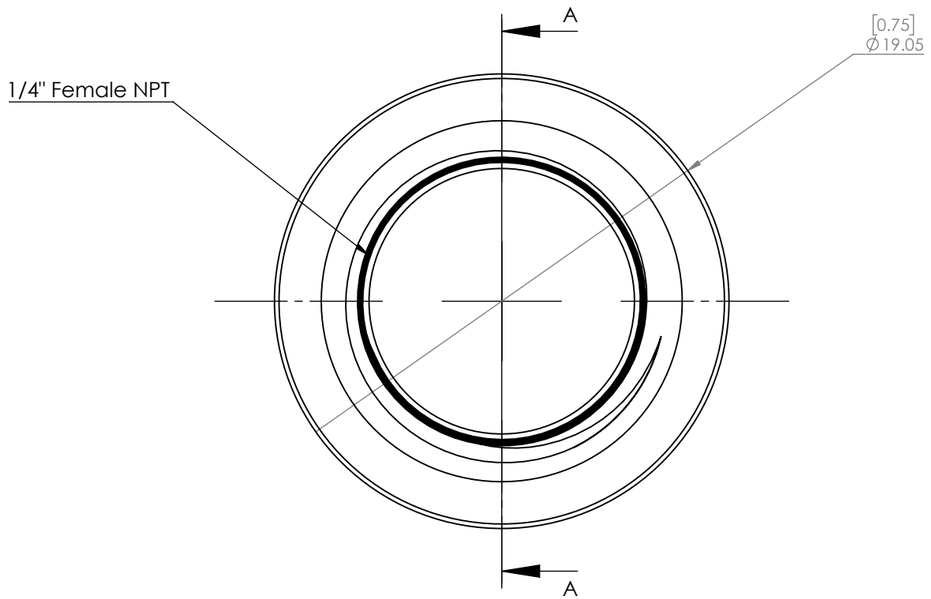
REVISION A

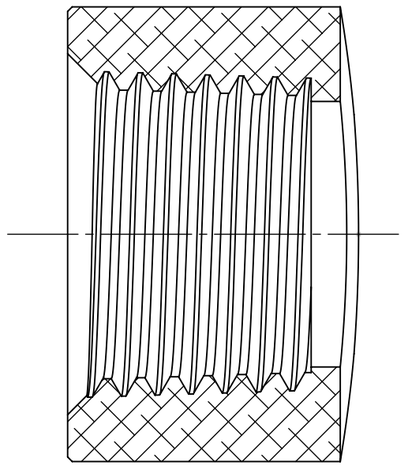
DRW File: 022_B-PC-Z-03-COOL_JACKET | Project: **GSE-1** | Mass: 283.10333 | SCALE:4:5 | SHEET 1 OF 1





SCALE 2 : 1





SECTION A-A

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Machine from aluminum coupling.
Possible Supplier: McMaster-Carr 44705K317

Quantity: 2

Wednesday, April 13, 2016 3:00:44 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	McMaster-Carr
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Epoxy Glue-on Aluminum Coolant

Material:

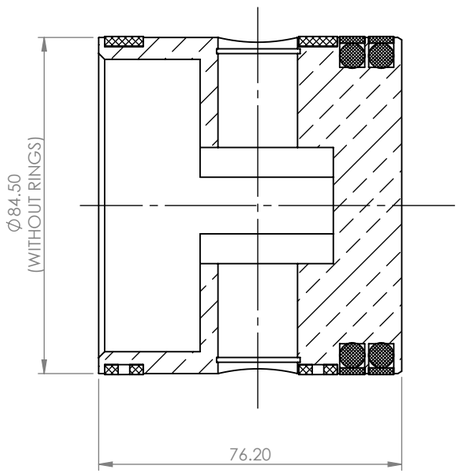
DWG NO. B-PC-Z-04-AL_COOL_OUTLET

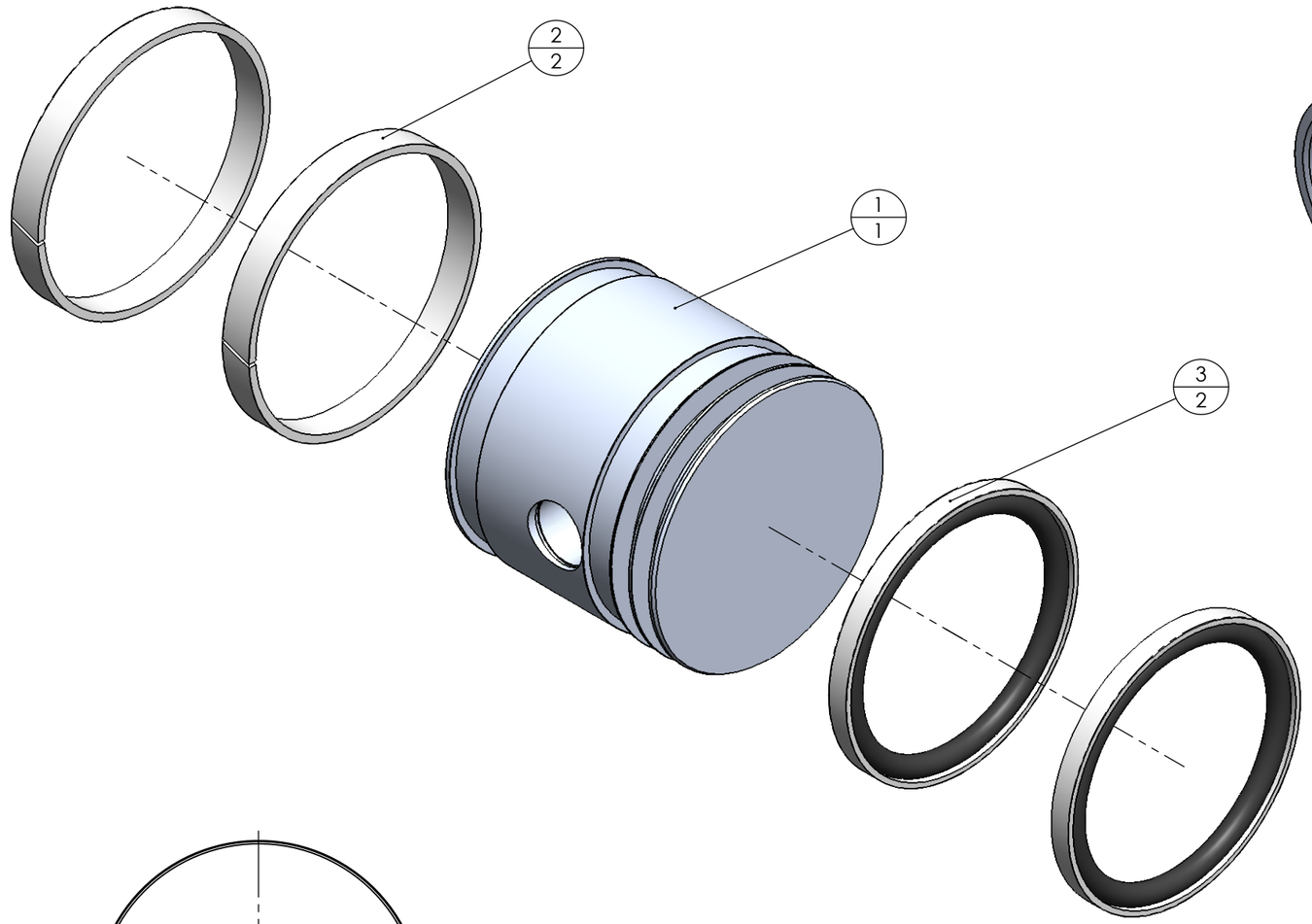
REVISION A

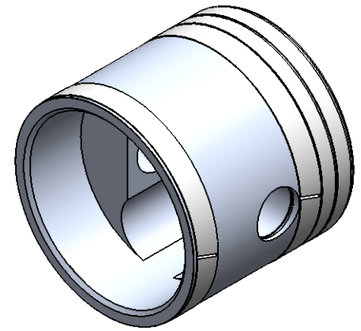
DRW File: 022_B-PC-Z-04-COOL_OUTLET | Project: GSE-1 | Mass: | SCALE:4:1 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	B-PP-P-01-PISTON_HEAD	1
2	B-PP-P-02-WEAR_RING_9.7MM	2
3	B-PP-P-03-POWER_PISTON_RING	2

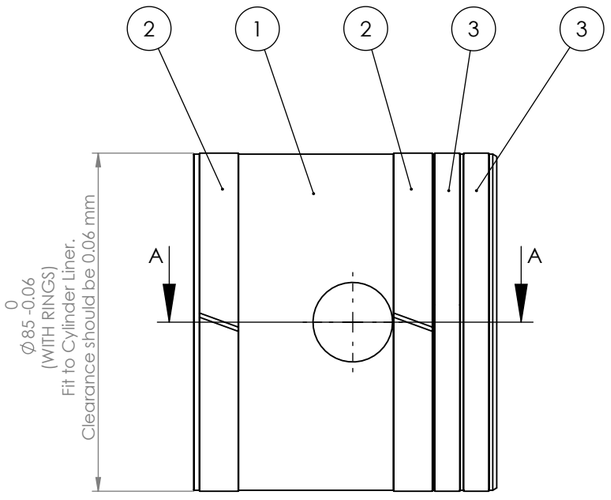
SECTION A-A

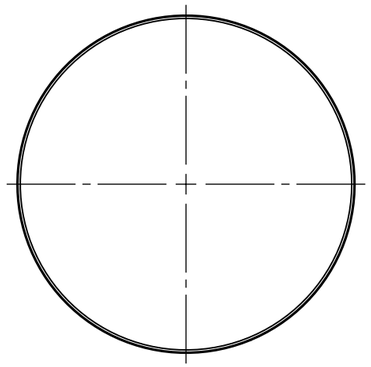






SCALE 1 : 2





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Wednesday, April 13, 2016 3:37:41 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

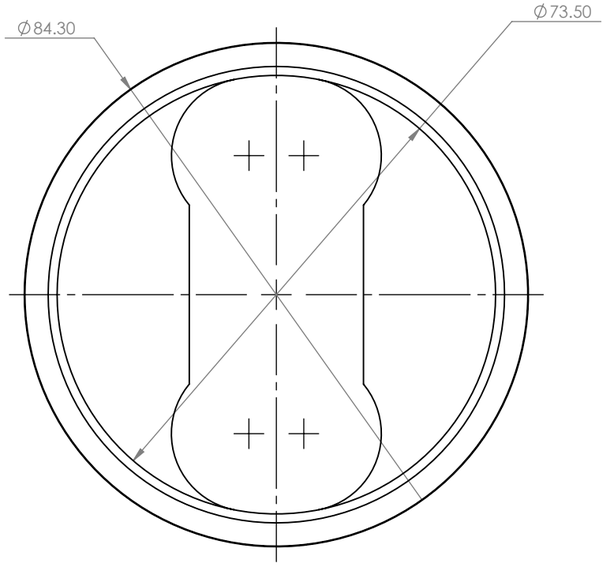
Power Piston Assembly

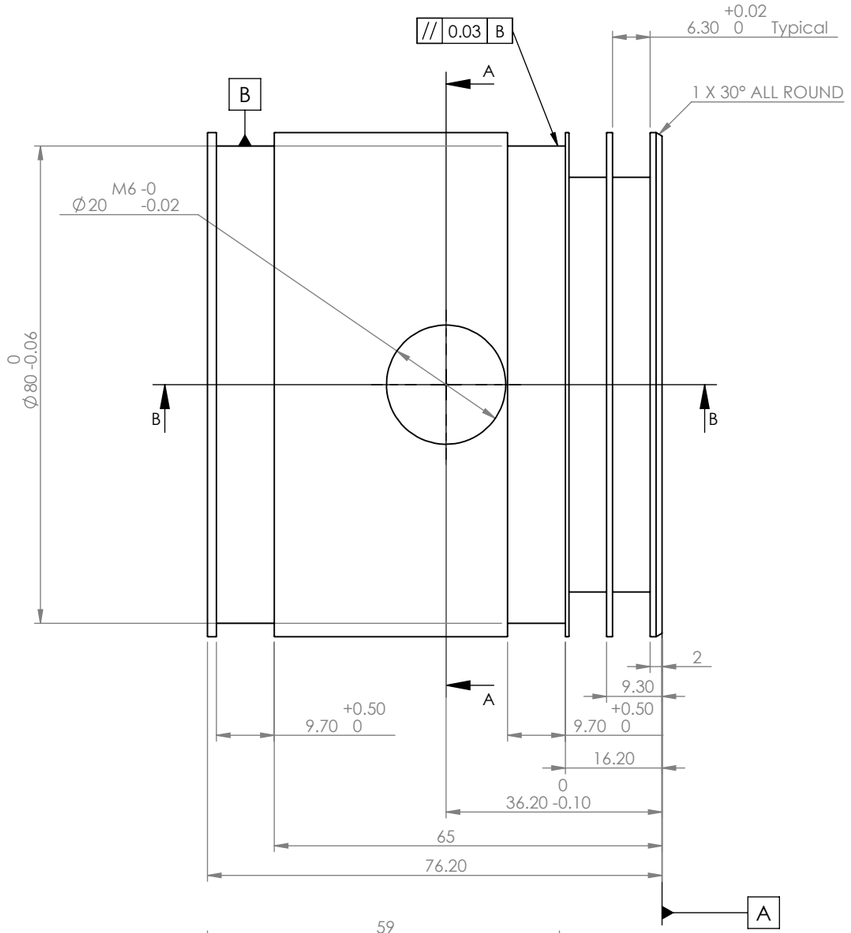
Material: N/A

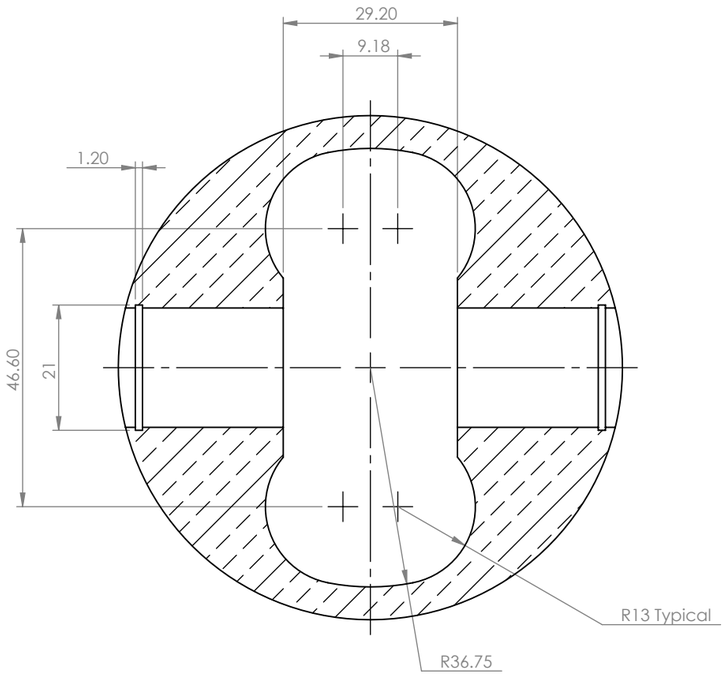
DWG NO. **B-PP-P-00-PISTON_HEAD_ASM**

REVISION A

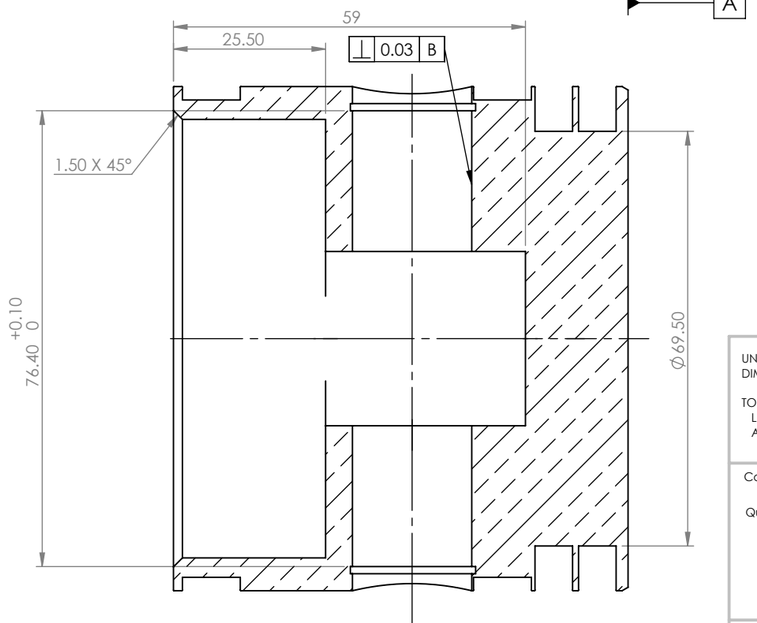
DRW File: 024_B-PP-P-00-PISTON_HEAD_ASM | Project: **GSE-1** | Mass: | SCALE:2:3 | SHEET 1 OF 1



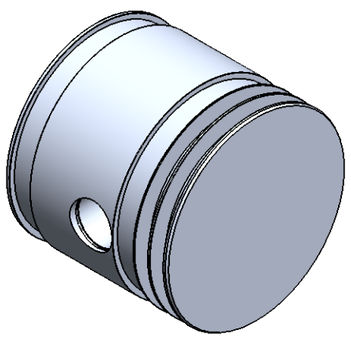




SECTION A-A



SECTION B-B



SCALE 1 : 2

Please fillet/chamfer all edges to R0.1

General surface finish: Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 12:53:39 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE: **Power Piston**

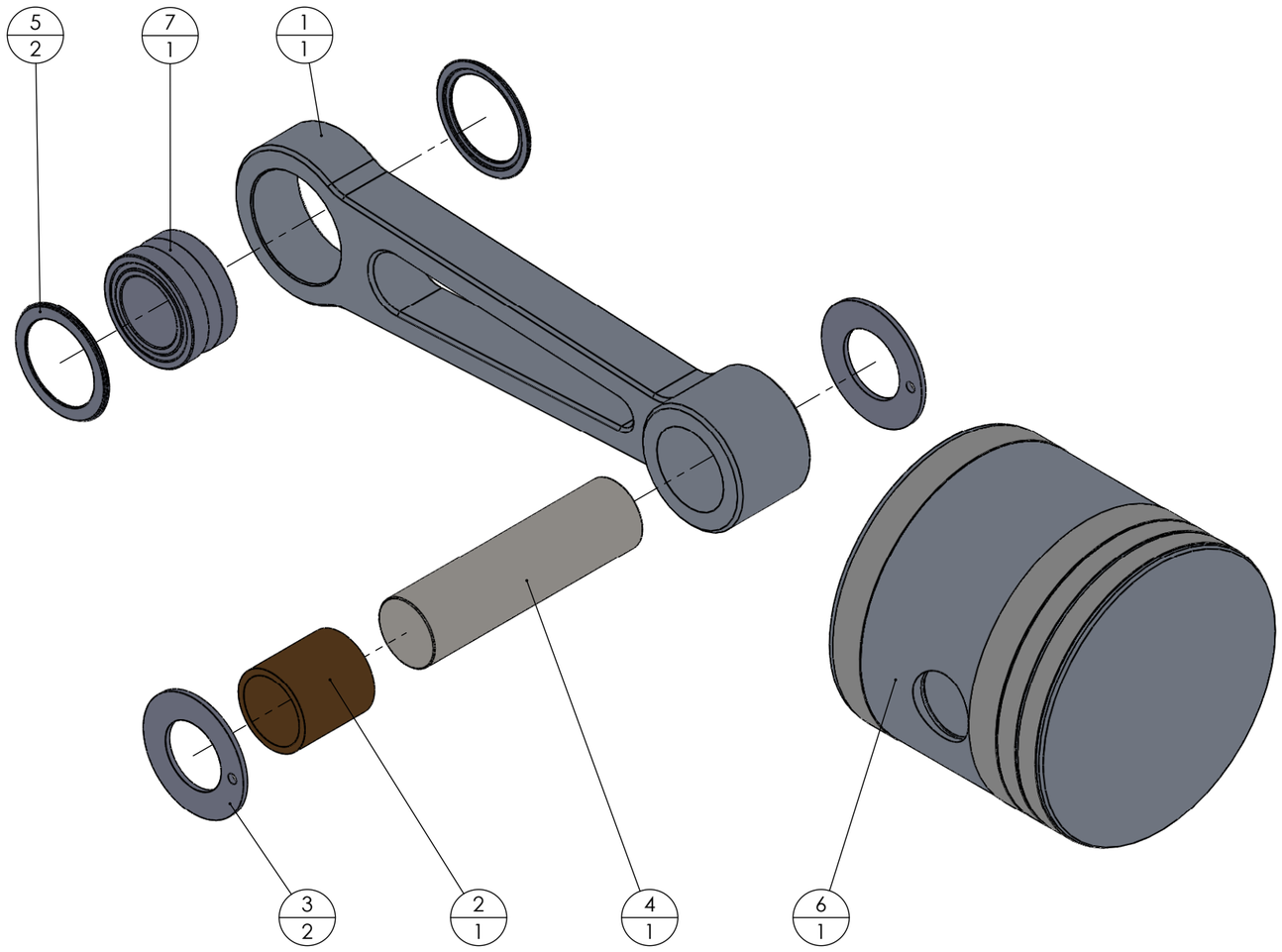
Material: 3.3547 (EN-AW 5083)

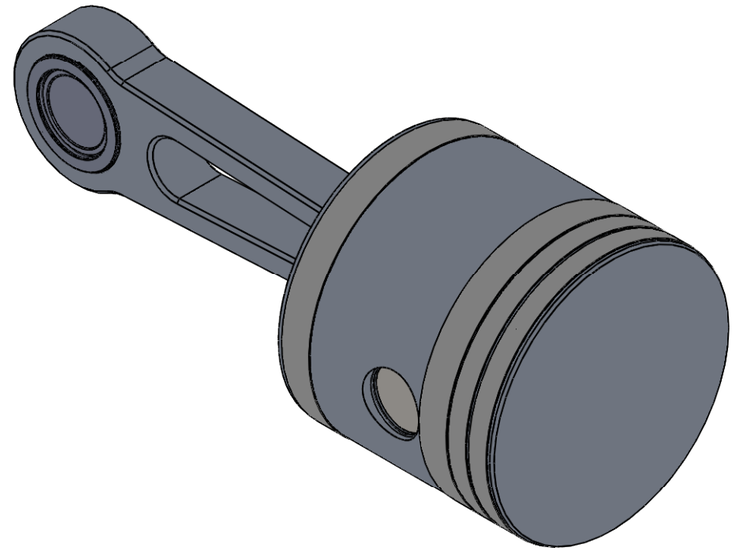
DWG NO. **B-PP-P-01-PISTON_HEAD**

REVISION A

DRW File: 024_B-PP-P-01-PISTON_HEAD | Project: **GSE-1** | Mass: 514.69082 | SCALE:1:1 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	B-PP-Z-01-POWER_PISTON_ROD	1
2	B-PP-Z-02-PIN_BUSHING	1
3	B-PP-Z-03-PIN_SPACER	2
4	B-PP-Z-04-PISTON_PIN	1
5	B-PP-Z-06-FACEPLATE	2
6	B-PP-P-00-PISTON_HEAD_ASM	1
7	B-PP-Z-05-CRANK_BEARING	1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:53:36 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

Power Piston Assembly

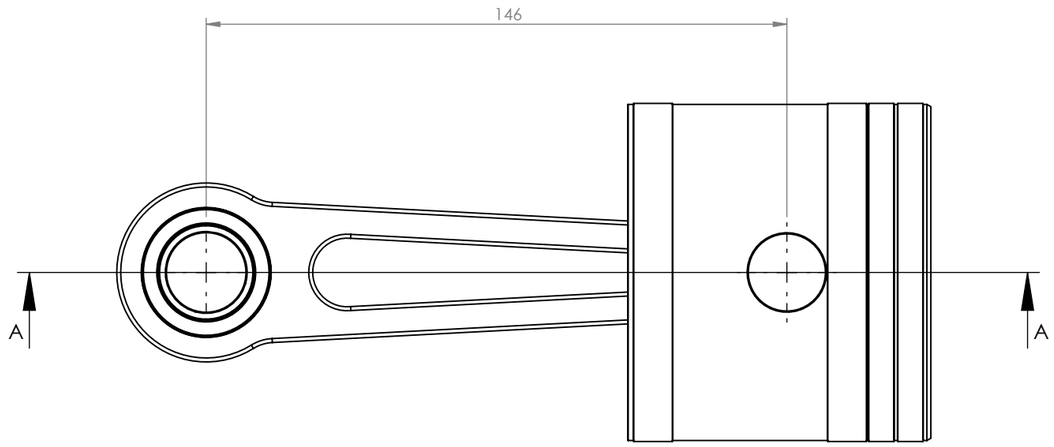
Material: N/A

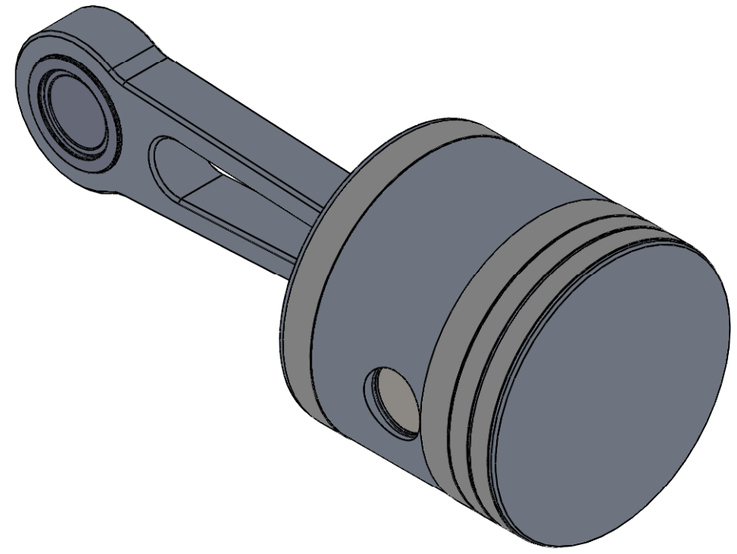
DWG NO. **B-PP-Z-00-POWER_PISTON_ASM**

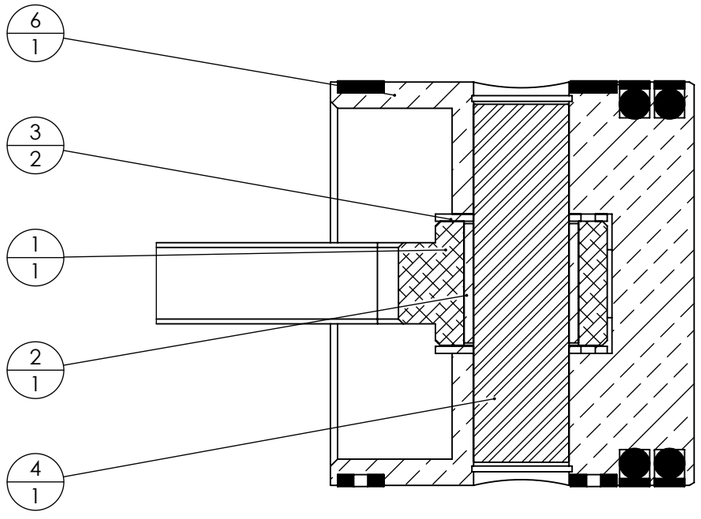
REVISION A

DRW File: 024_B-PP-Z-00-POWER_PISTON_ASM | Project: **GSE-1** | Mass: | SCALE:2:3 | SHEET 1 OF 2

ITEM NO.	DRW NUMBER	QTY.
1	B-PP-Z-01-POWER_PISTON_ROD	1
2	B-PP-Z-02-PIN_BUSHING	1
3	B-PP-Z-03-PIN_SPACER	2
4	B-PP-Z-04-PISTON_PIN	1
5	B-PP-Z-06-FACEPLATE	2
6	B-PP-P-00-PISTON_HEAD_ASM	1
7	B-PP-Z-05-CRANK_BEARING	1







SECTION A-A
SCALE 4 : 5

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:53:36 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

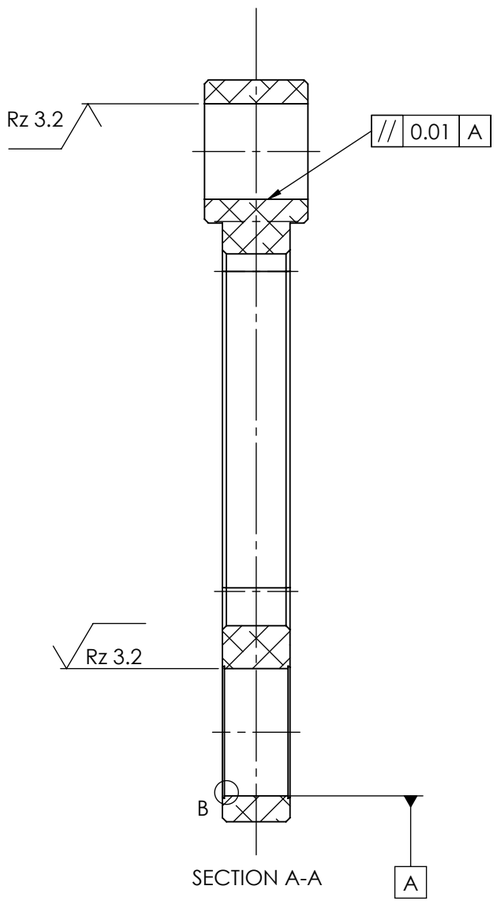
Power Piston Assembly

Material: N/A

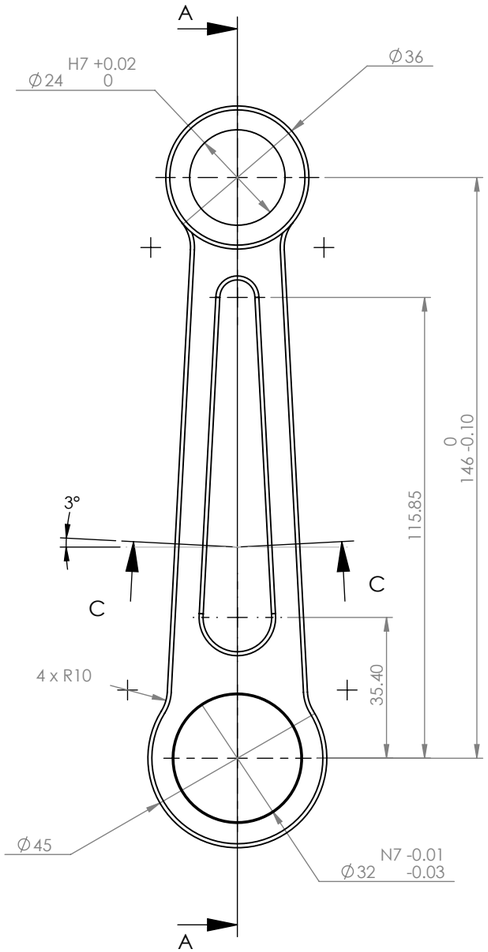
DWG NO. **B-PP-Z-00-POWER_PISTON_ASM**

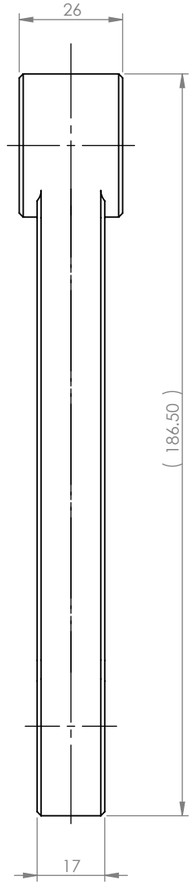
REVISION A

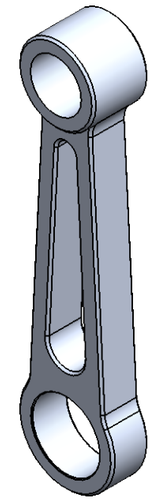
DRW File: 024_B-PP-Z-00-POWER_PISTON_ASM | Project: **GSE-1** | Mass: | SCALE:2:3 | SHEET 2 OF 2



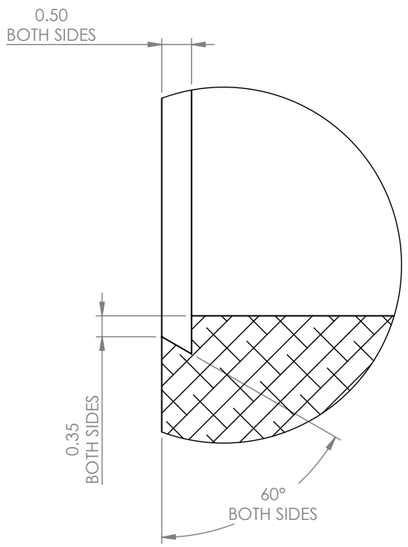
SECTION A-A



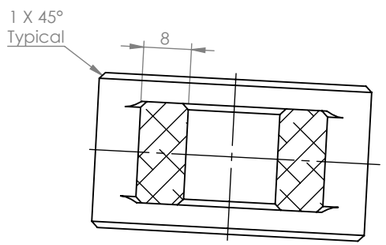




SCALE 1 : 2



DETAIL B
SCALE 10 : 1



SECTION C-C
SCALE 1 : 1

Please fillet/chamfer all edges to R0.5

General surface finish: $\sqrt{Rz\ 10}$

0.50
BOTH SIDES

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 11:55:58 AM

DO NOT SCALE DRAWING

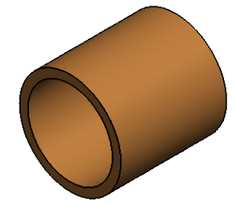
NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

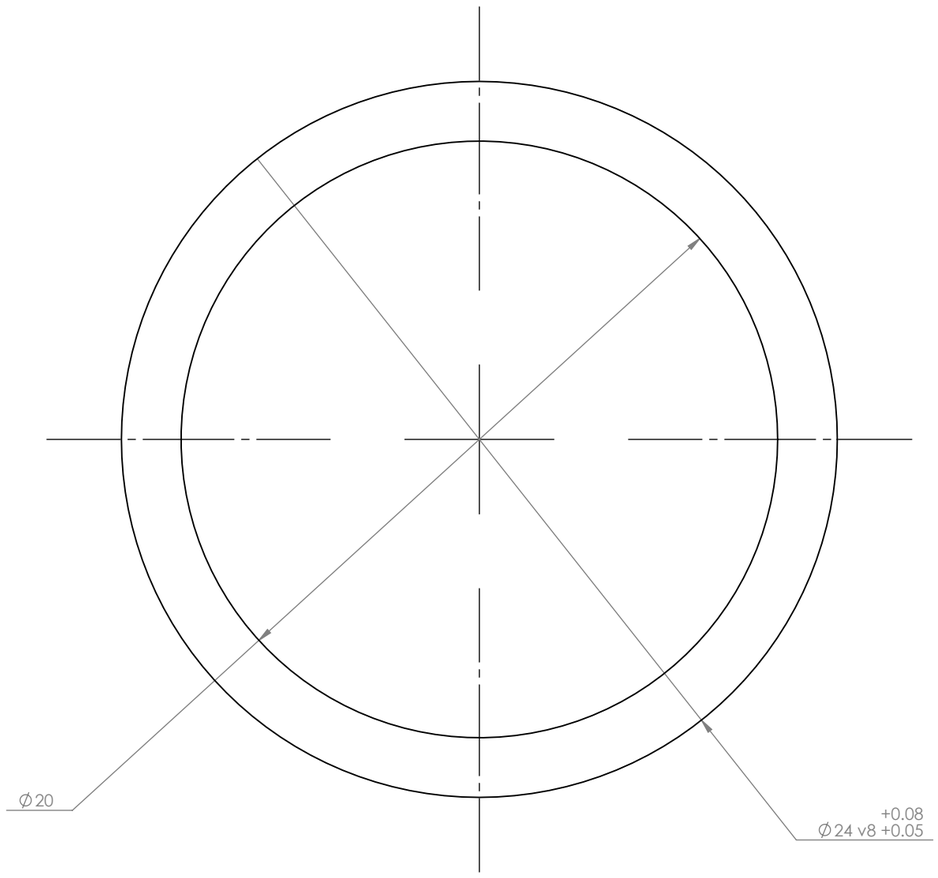
DRAWN	Jiacheng Yao	TITLE:
SOLID by	Jiacheng Yao	**Power Piston Connecting Rod**
CHK'D	NOT CHECKED	
APPV'D	NOT APPROVED	

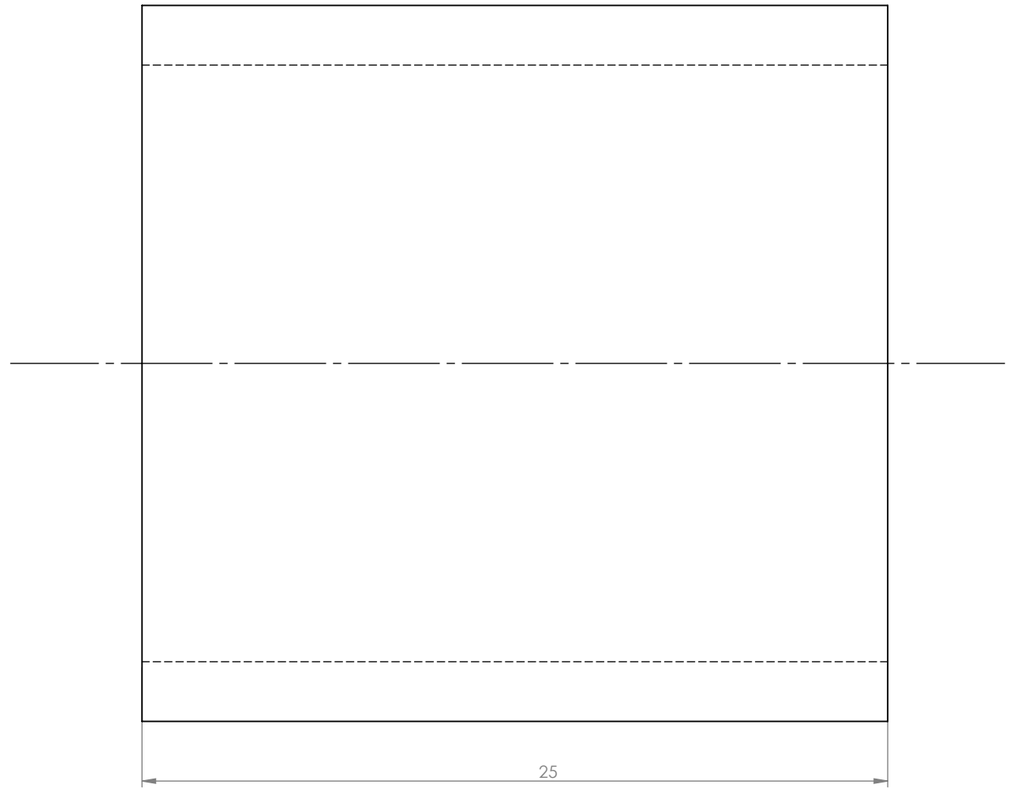
Material:	DWG NO.	REVISION
7075-T6, Plate (SS)	**B-PP-Z-01-POWER_PISTON_ROD**	A

DRW File: 024_B-PP-Z-01-POWER_PISTON_ROD	Project: **GSE-1**	Mass: 172.45184	SCALE:2:3	SHEET 1 OF 1



SCALE 1 : 1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Machined from stock brass bearing.

Supplier: Bunting Bearings LLC #CBM020025025

Quantity: 1

Wednesday, April 13, 2016 4:05:01 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

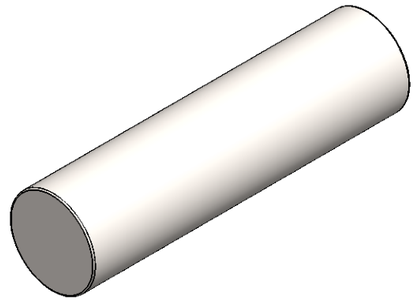
Power Piston Pin Bushing

Material: Tin Bearing Bronze

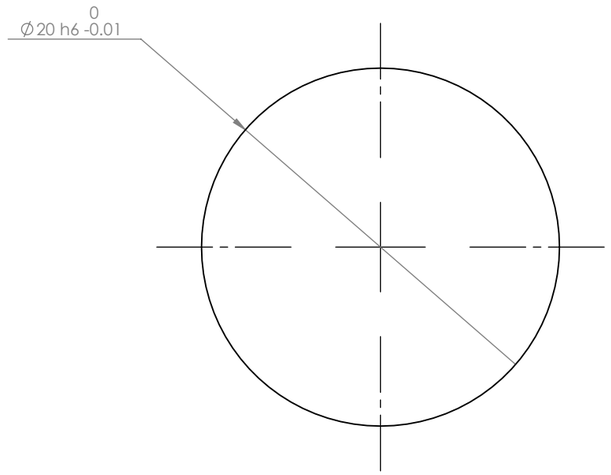
DWG NO. **B-PP-Z-02-PIN_BUSHING**

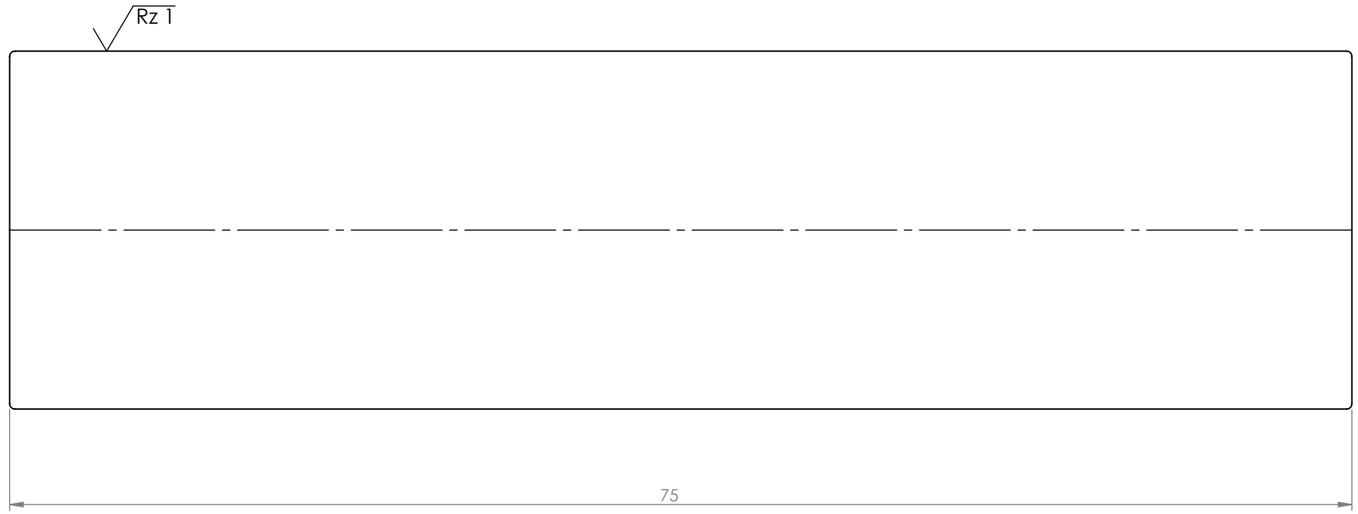
REVISION A

DRW File: 024_B-PP-Z-02-PIN_BUSHING | Project: **GSE-1** | Mass: 28.68274 | SCALE:5:1 | SHEET 1 OF 1



SCALE 1 : 1





Please fillet/chamfer all edges to R0.3

General surface finish: Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 1482K42

Quantity: 1

Hardened to RHC 57-62
Thermex Metal Treating Ltd.

Wednesday, April 13, 2016 3:59:17 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

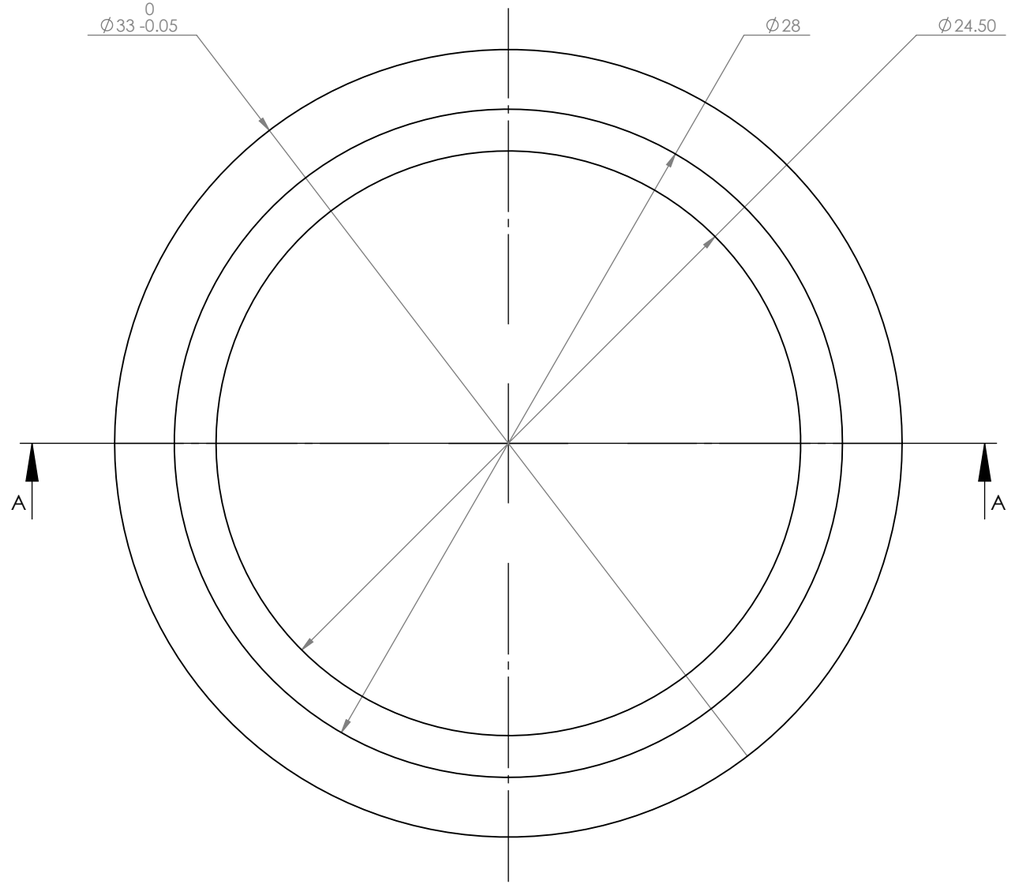
DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

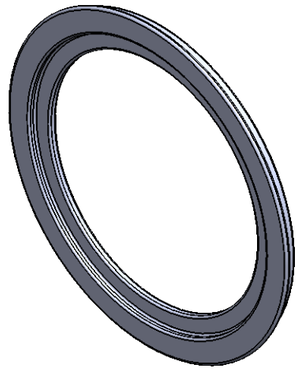
TITLE:

Power Piston Pin

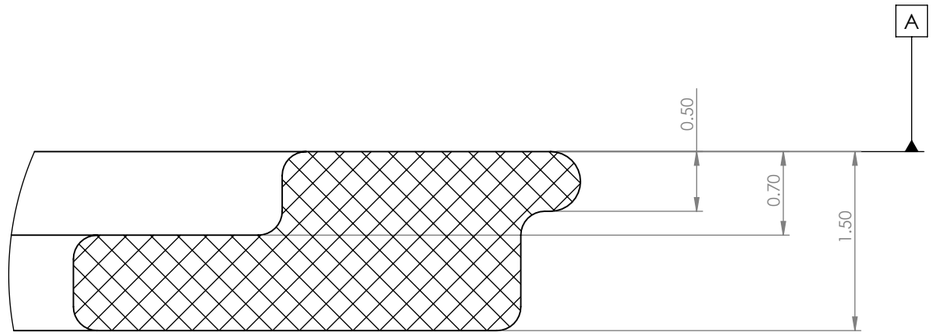
Material:	DWG NO.	REVISION
Plain Carbon Steel	**B-PP-Z-04-PISTON_PIN**	A

DRW File: 024_B-PP-Z-04-PISTON_PIN	Project: **GSE-1**	Mass: 183.76437	SCALE:3:1	SHEET 1 OF 1

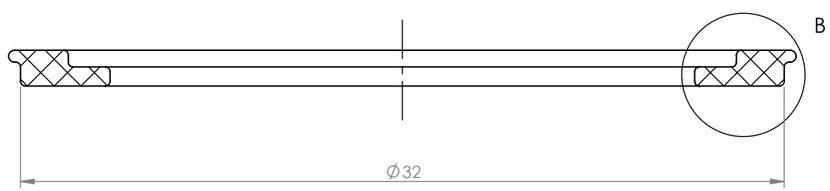




SCALE 2 : 1



DETAIL B
SCALE 20 : 1



SECTION A-A

Please fillet/chamfer all edges to R0.2

General surface finish: $\sqrt{}$ Rz 6.3

Quantity: 2

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 8578K411

Quantity: 2

Wednesday, April 13, 2016 3:48:14 PM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Faceplate

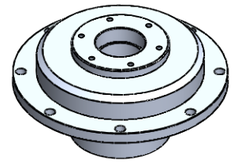
Material:
Delrin 2700 NC010, Low Viscosity Acetal Copolymer (SS)

DWG NO.
B-PP-Z-06-FACEPLATE

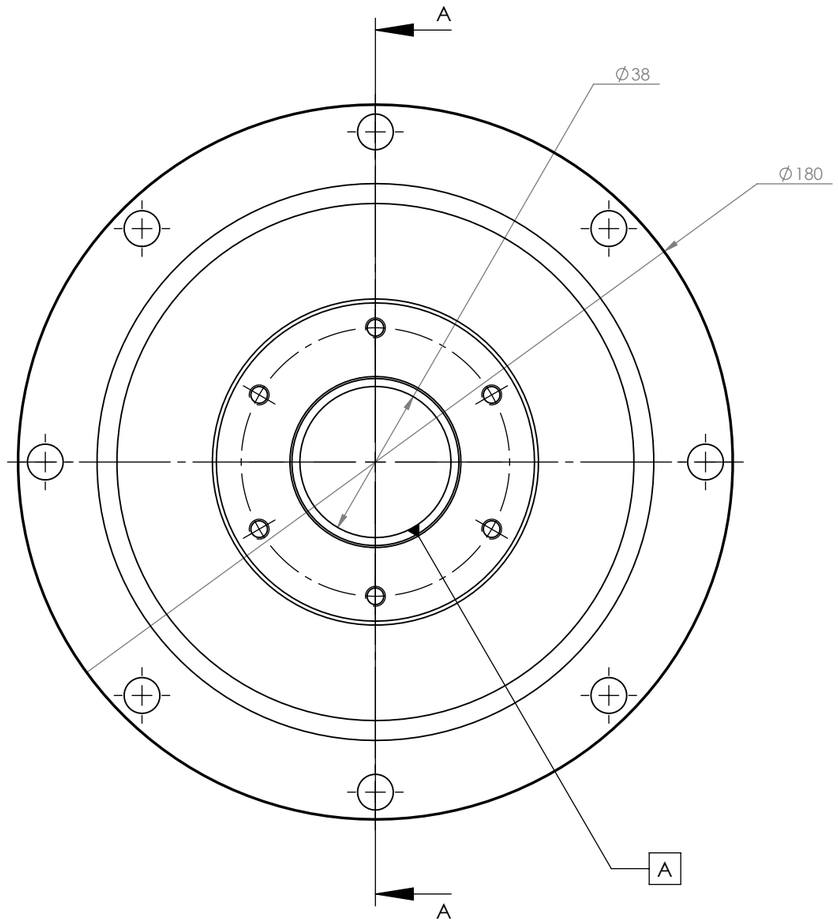
REVISION
A

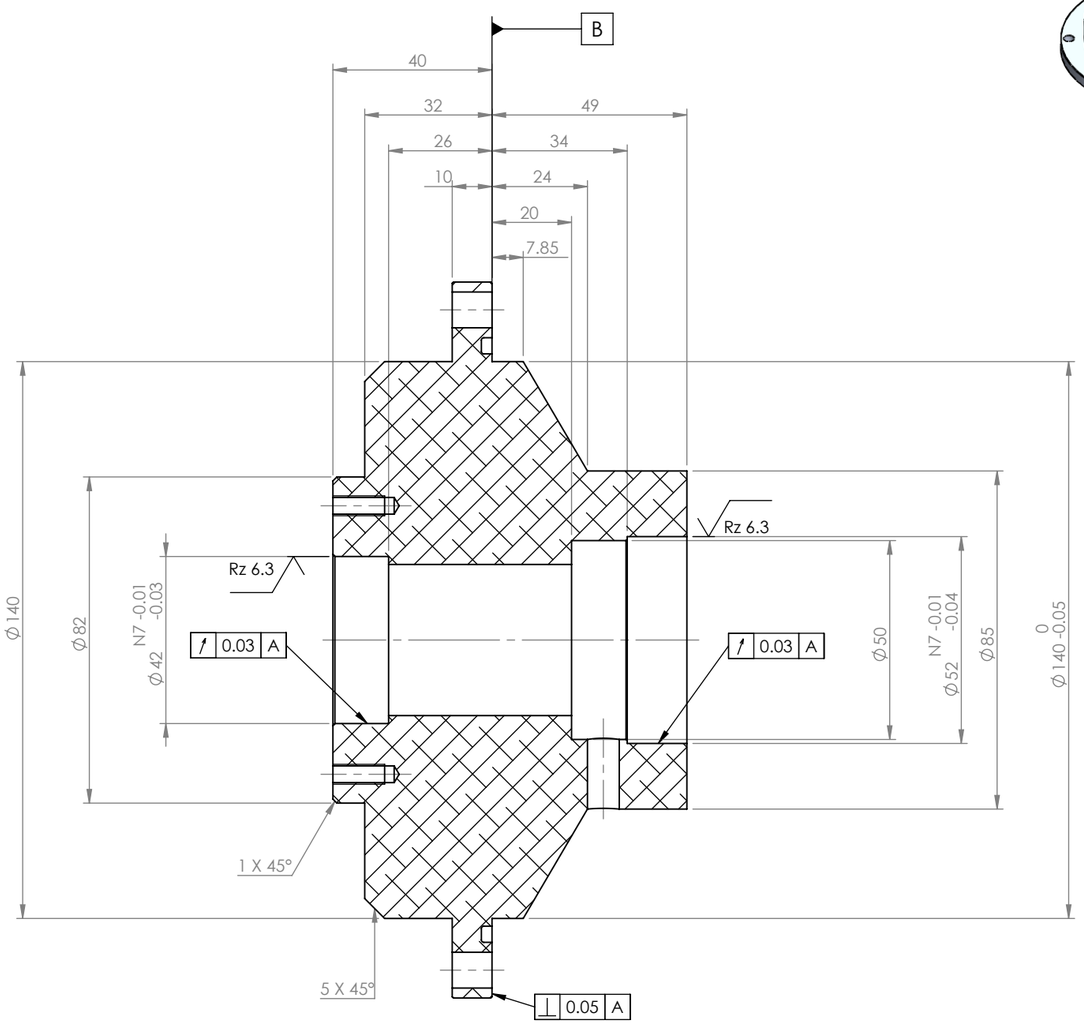
DRW File: 024_B-PP-Z-06-FACEPLATE | Project: **GSE-1** | Mass: 0.59167 | SCALE:4:1 | SHEET 1 OF 1

- Group C
  - C-BC-Z-00-BEARING_CARTRIDGE_ASM
    - C-BC-Z-01-BEARING_CARTRIDGE
    - C-BC-Z-02-ROTARY_SEAL_HOLDER
    - C-BC-Z-03-ROTARY_SEAL
    - C-BC-Z-04-CRANK_BEARING_1
    - C-BC-Z-05-CRANK_BEARING_2
    - C-BC-Z-06-CARTRIDGE_SCREW
  - C-CC-Z-00-CRANKCASE_ASM
    - C-CC-Z-01-CRANKCASE
    - C-CC-Z-02-SHAFT_BEARING
  - C-CS-Z-00-CRANKSHAFT_ASM
    - C-CS-Z-01-SHAFT
    - C-CS-Z-02-PIVOT
    - C-CS-Z-03-CRANKWEB
  - C Other Parts
    - C-ZZ-Z-01-COVER
    - C-ZZ-Z-02-FLEX_COUPLER_A
    - C-ZZ-Z-03-SPACER_115
    - C-ZZ-Z-04-SPACER_116
    - C-ZZ-Z-05-SPACER_117
    - C-ZZ-Z-06-SPACER_118
    - C-ZZ-Z-07-THREADED_ROD
    - C-ZZ-F-00-FLYWHEEL_ASM
      - C-ZZ-F-01-DISC
      - C-ZZ-F-02-TAPER_LOCK_BUSHING
      - C-ZZ-F-03-WELD_ON_BUSHING_BORE_HUB



SCALE 1 : 5





SECTION A-A

Please fillet/chamfer all edges to R0.3

General surface finish: √ Rz 10

O-ring groove surface finish: √ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 8974K9

Quantity: 1

Thursday, April 14, 2016 10:19:15 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

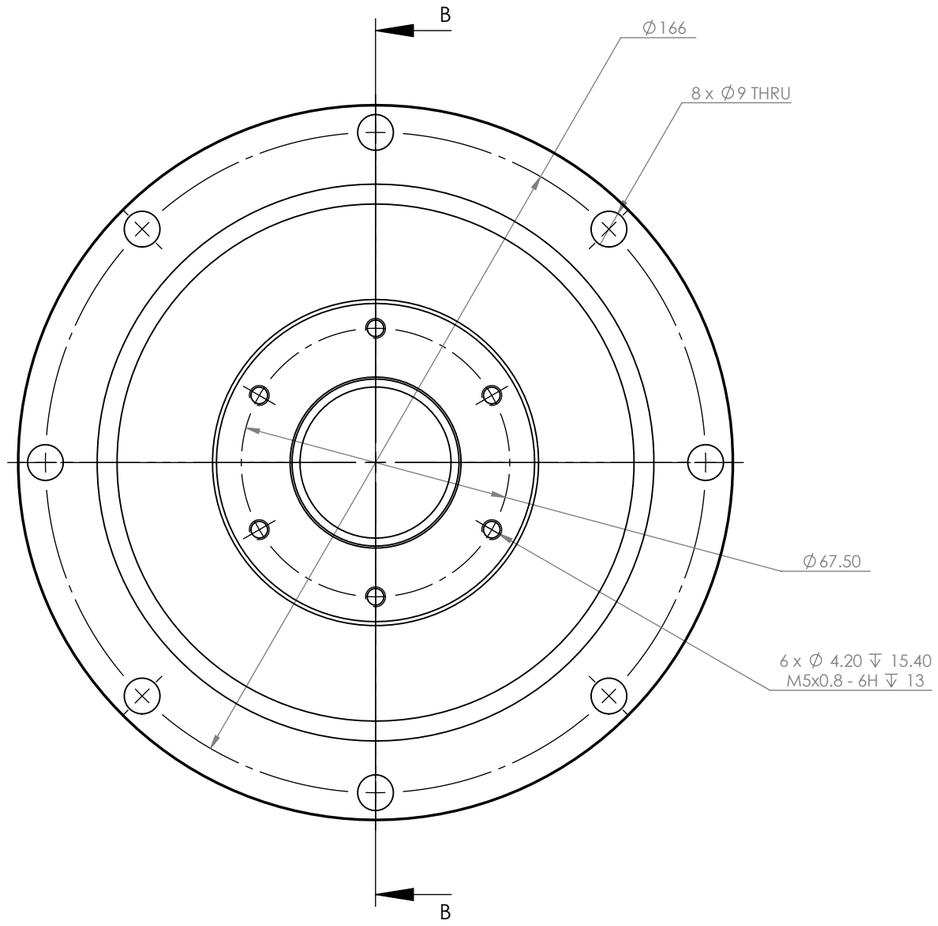
Crankshaft Bearing Cartrige

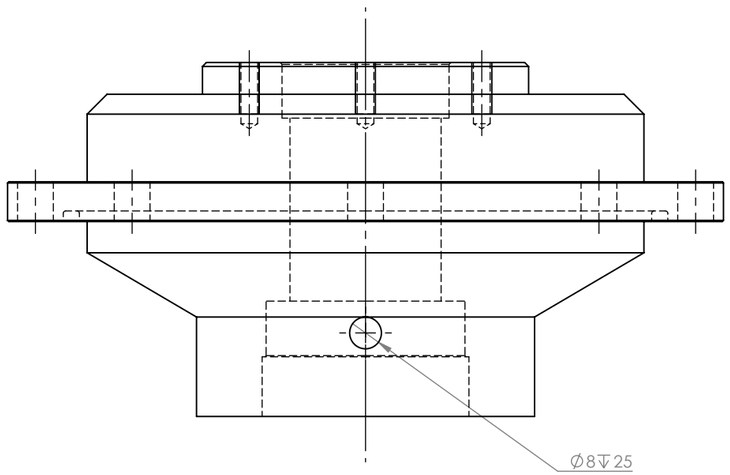
Material: 6061-T6 (SS)

DWG NO. **C-BC-Z-01-BEARING_CARTRIDGE**

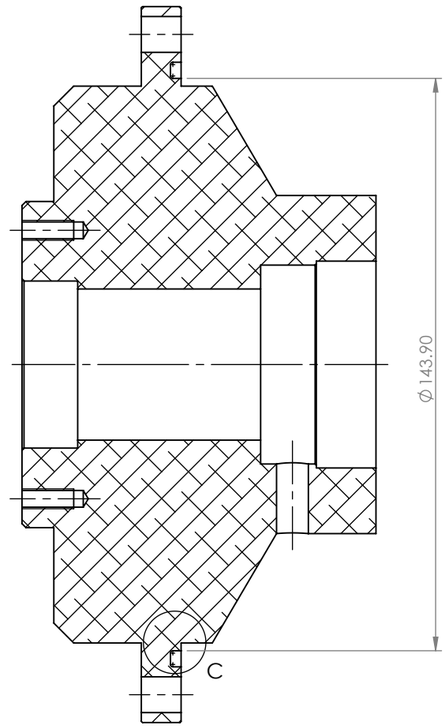
REVISION A

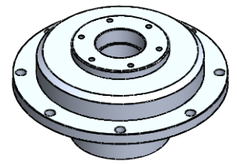
DRW File: 026_C-BC-Z-01-BEARING_CARTRIDGE | Project: **GSE-1** | Mass: 2465.36205 | SCALE:2:3 | SHEET 1 OF 2





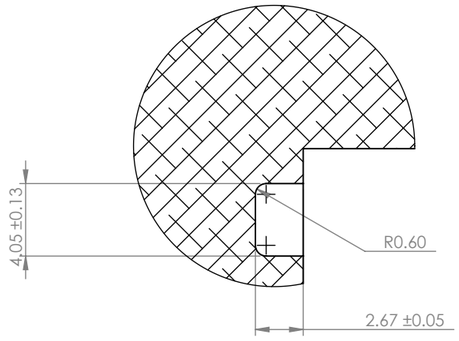
SECTION B-B





SCALE 1 : 5

O-Ring Information:
Size: 3.5X145mm
HiTechSeals Part #: 35145
Material: Nitrile



DETAIL C
SCALE 3 : 1

Please fillet/chamfer all edges to R0.3

General surface finish: √ Rz 10

O-ring groove surface finish: √ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 8974K9

Quantity: 1

Thursday, April 14, 2016 10:19:15 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

Crankshaft Bearing Cartrige

Material: 6061-T6 (SS)

DWG NO. C-BC-Z-01-BEARING_CARTRIDGE

REVISION A

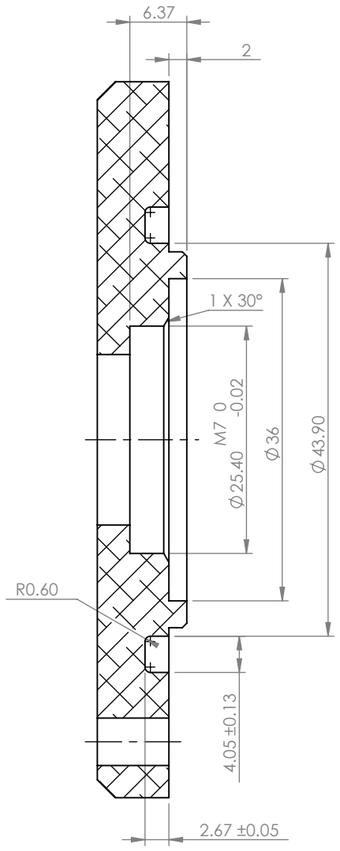
DRW File: 026_C-BC-Z-01-BEARING_CARTRIDGE

Project: GSE-1

Mass: 2465.36205

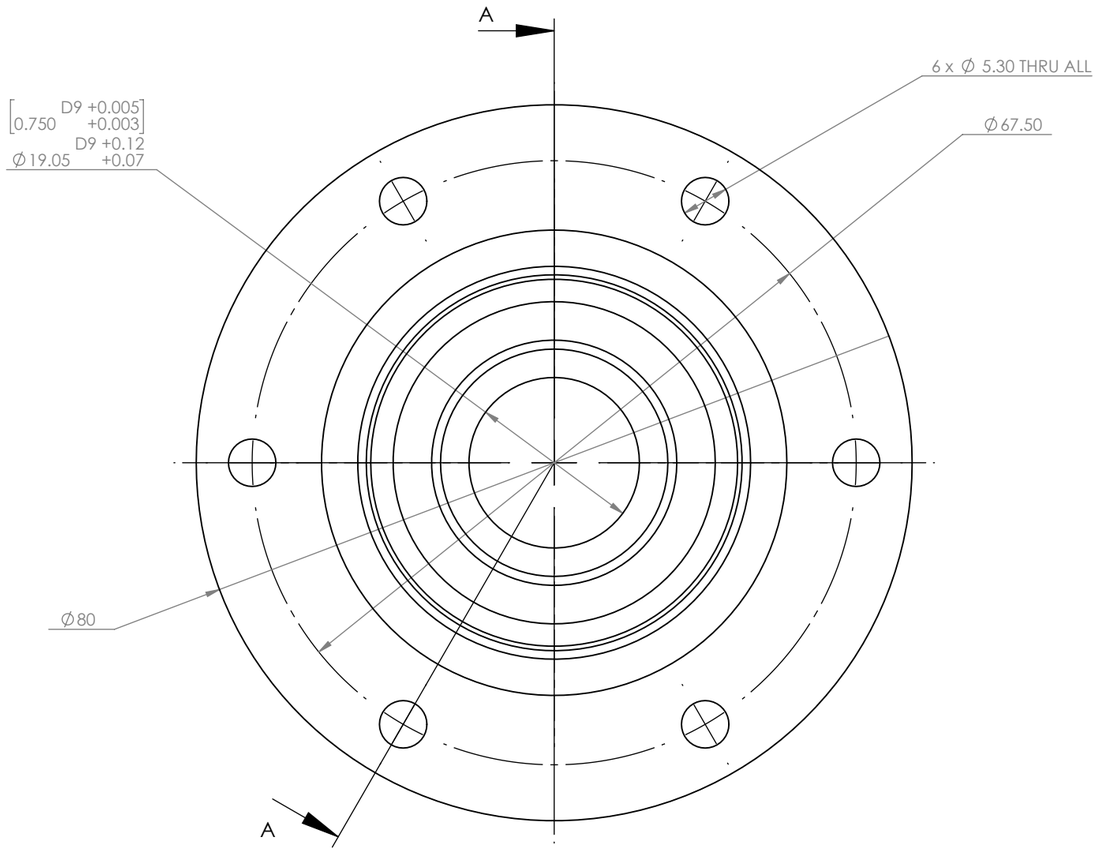
SCALE:2:3

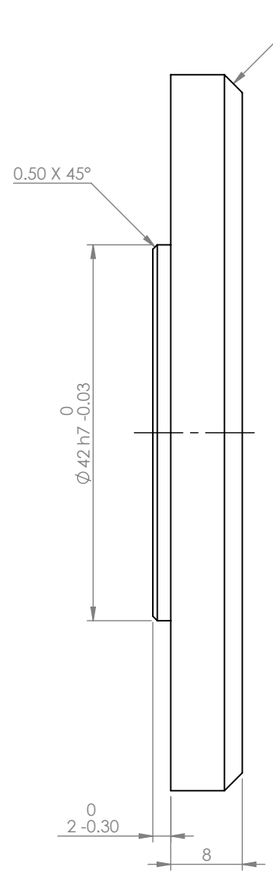
SHEET 2 OF 2

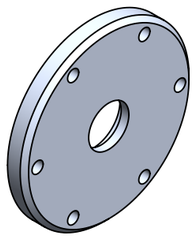


SECTION A-A

O-ring Information
Size: 3.5x45mm
Supplier: Hi Tech Seals
Part#: 35045
Material: Nitrile







SCALE 1 : 2

Please fillet/chamfer all edges to R0.1

General Surface finish Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 1610T27

Quantity: 1

Thursday, April 14, 2016 9:48:36 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

ROTARY SEAL HOLDER

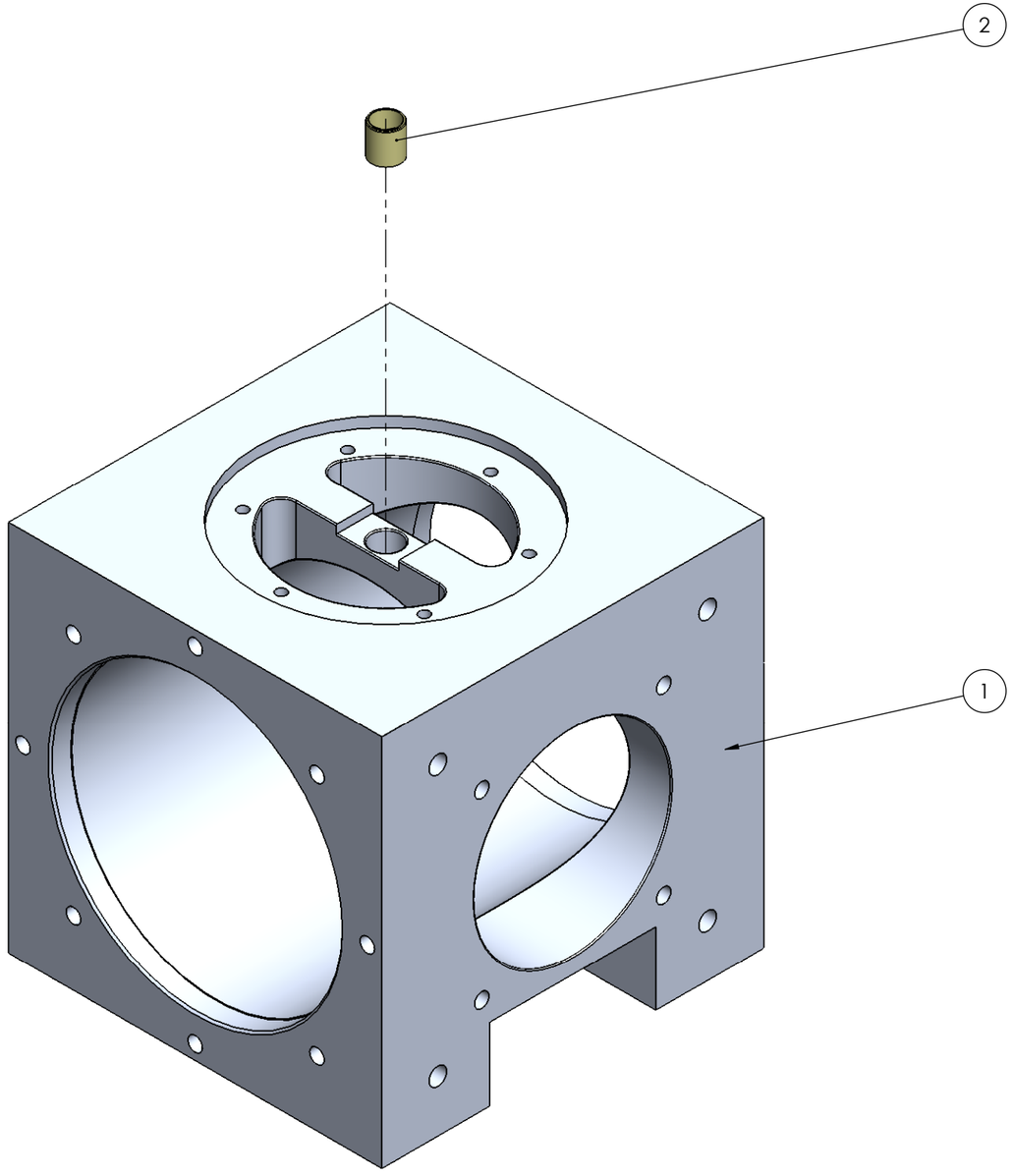
Material: 6061-T6 (SS)

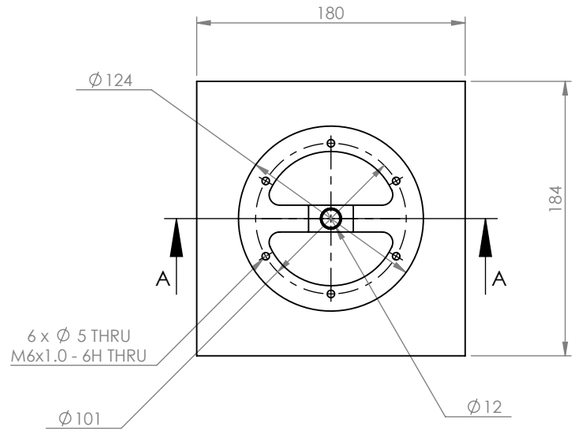
DWG NO. C-BC-Z-02-ROTARY_SEAL_HOLDER

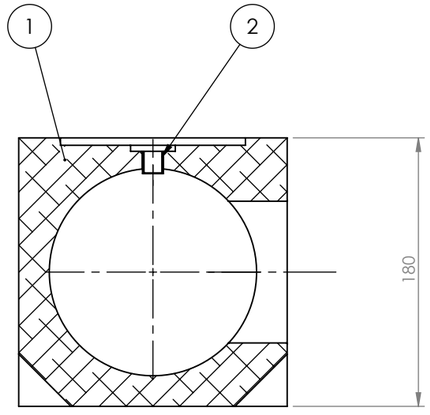
REVISION A

DRW File: 026_C-BC-Z-02-ROTARY_SEAL_HOLDER | Project: GSE-1 | Mass: 93.15171 | SCALE:3:2 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	C-CC-Z-01-CRANKCASE	1
2	C-CC-Z-02-SHAFT_BEARING	1







SECTION A-A
SCALE 1 : 4

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Thursday, April 14, 2016 10:30:24 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

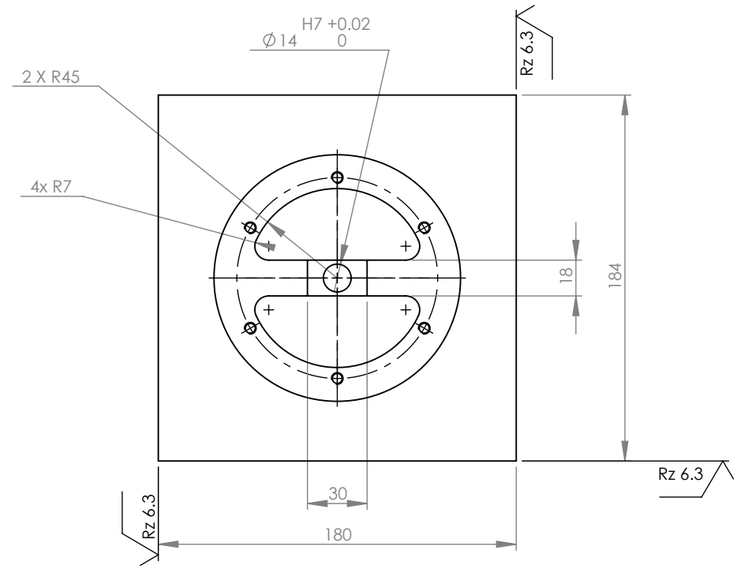
Crankcase Assembly

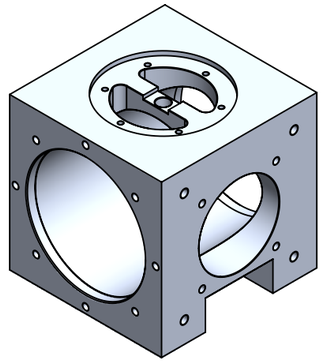
Material: N/A

DWG NO. **C-CC-Z-00-CRANKCASE_ASM**

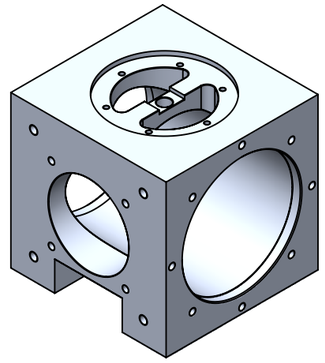
REVISION A

DRW File: 028_C-CC-Z-00-CRANKCASE_ASM | Project: **GSE-1** | Mass: | SCALE:1:2 | SHEET 1 OF 1

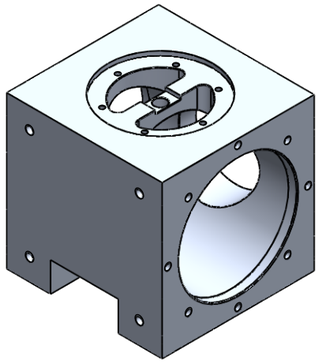




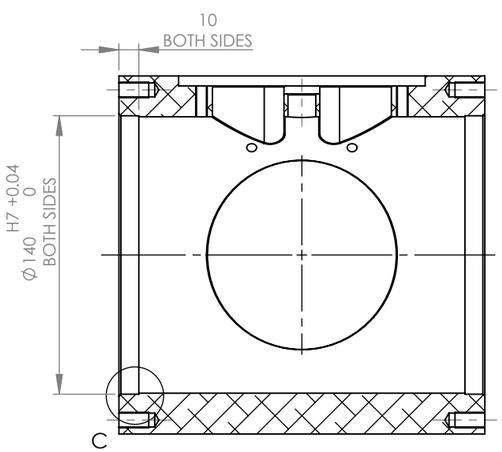
SCALE 1 : 5



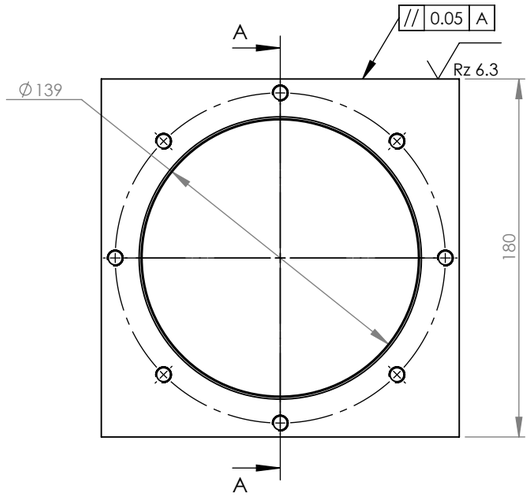
SCALE 1 : 5

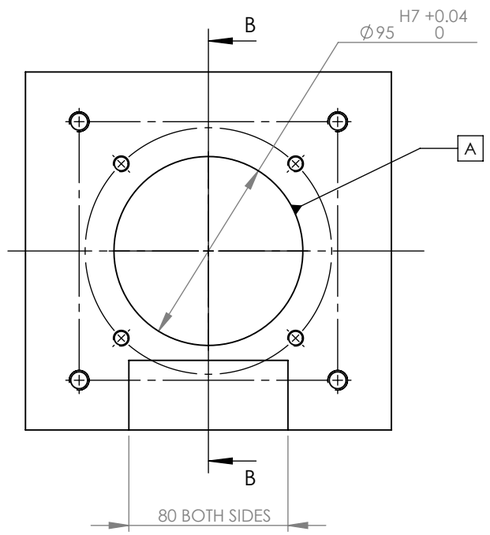


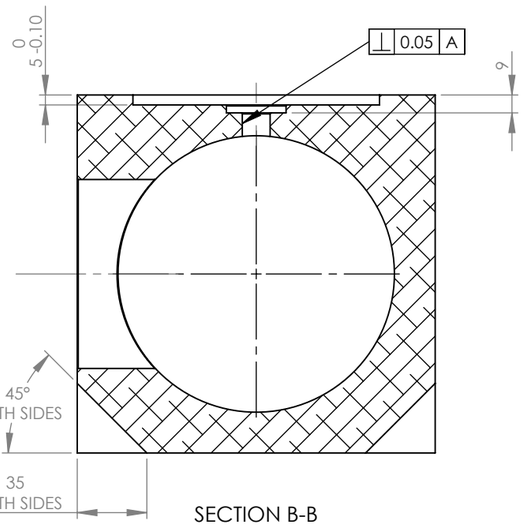
SCALE 1 : 5



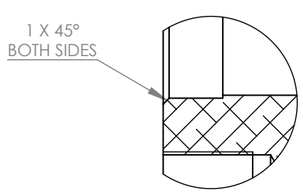
SECTION A-A







SECTION B-B



DETAIL C
SCALE 1 : 1

Please fillet/chamfer all edges to R0.5

General surface finish Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Quantity: 1

Friday, April 15, 2016 10:00:04 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**David S. Nobes**
SOLID by	Jiacheng Yao
CHK'D	GM/AB
APPV'D	DSN

TITLE:

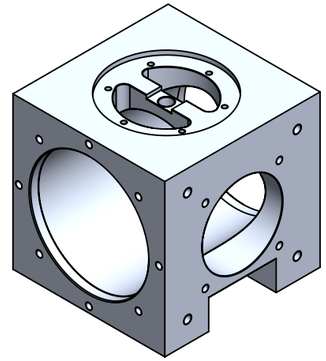
CRANKCASE

Material: 6061-T6 (SS)

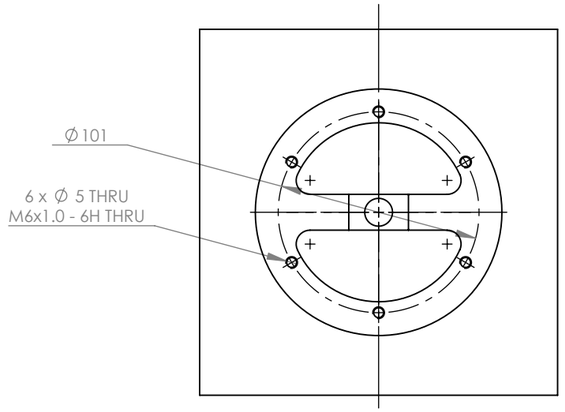
DWG NO. **C-CC-Z-01-CRANKCASE**

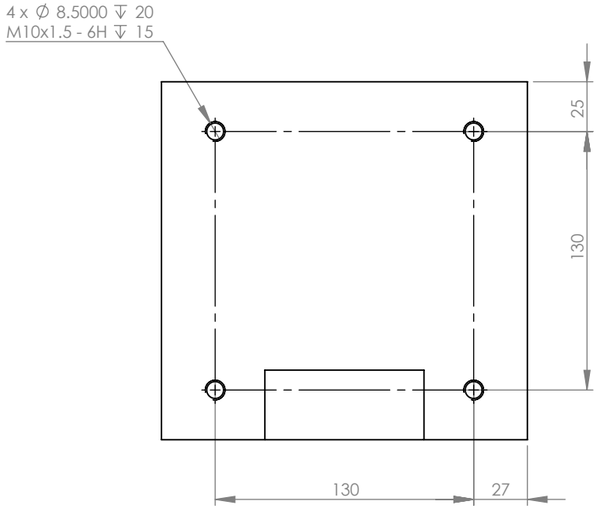
REVISION A

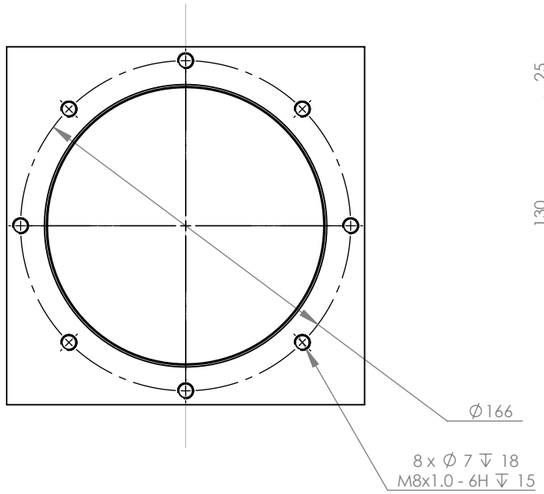
DRW File: 028_C-CC-Z-01-CRANKCASE | Project: **GSE-1** | Mass: 7309.56029 | SCALE:1:3 | SHEET 1 OF 2

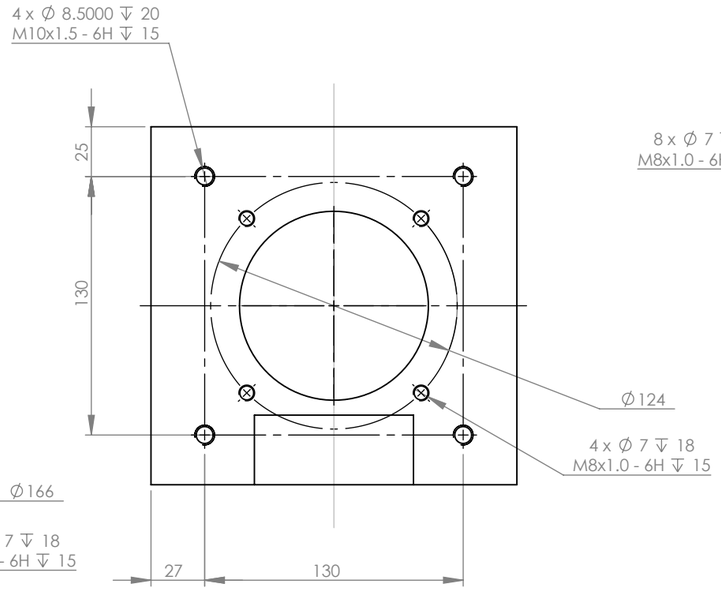


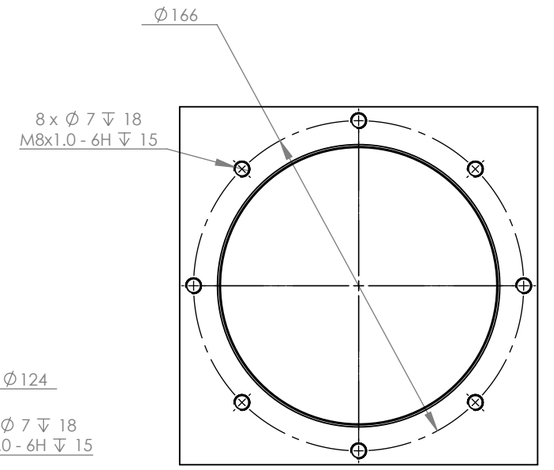
SCALE 1 : 5

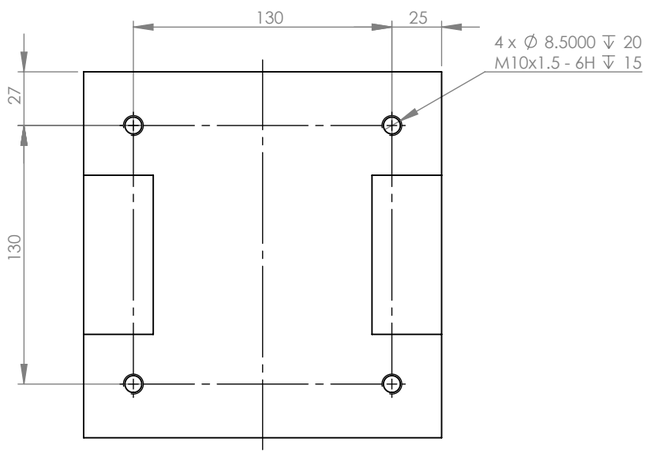












Please fillet/chamfer all edges to R0.5

General surface finish √ Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Quantity: 1

Friday, April 15, 2016 10:00:04 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	David S. Nobes
SOLID by	Jiacheng Yao
CHK'D	GM/AB
APPV'D	DSN

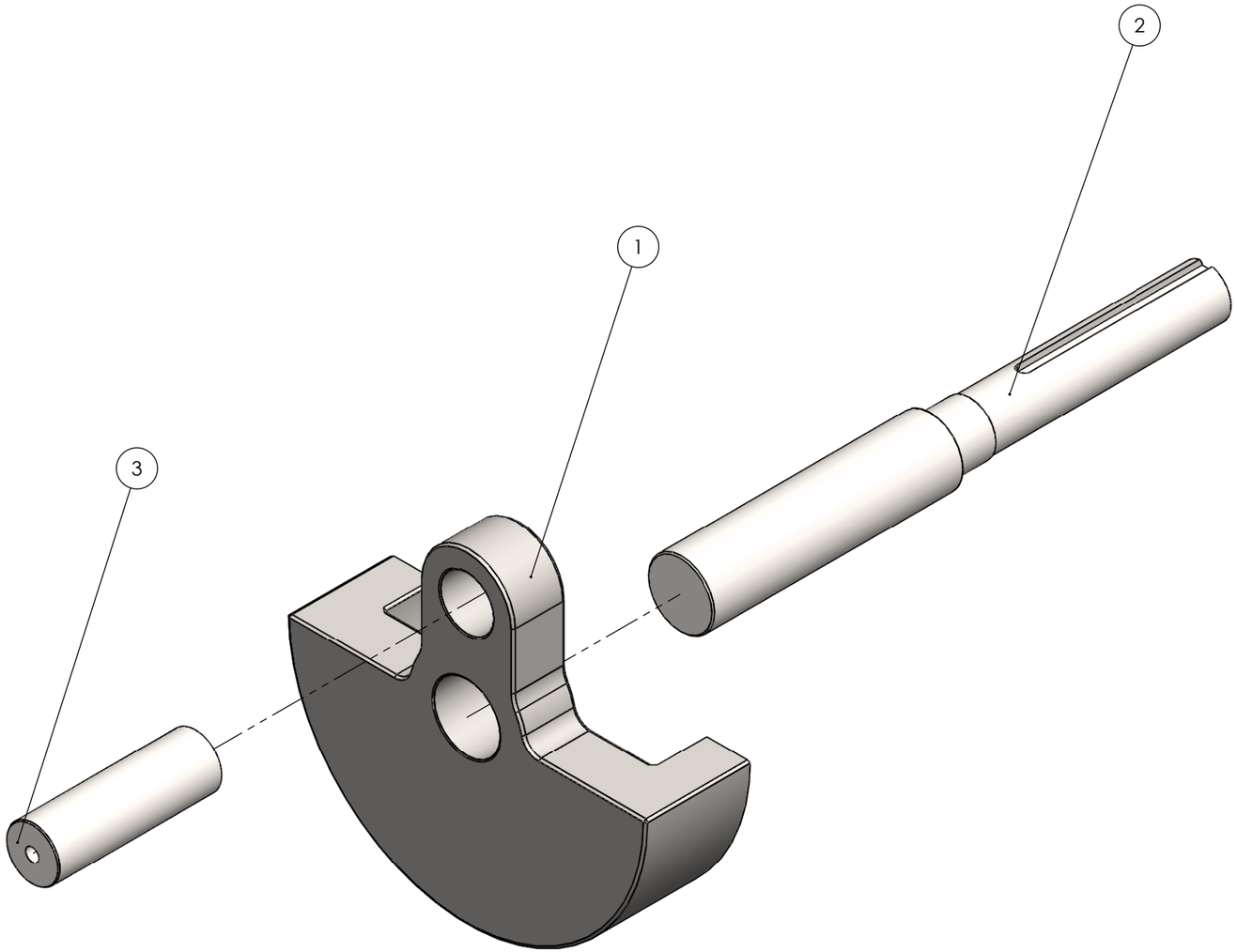
TITLE:

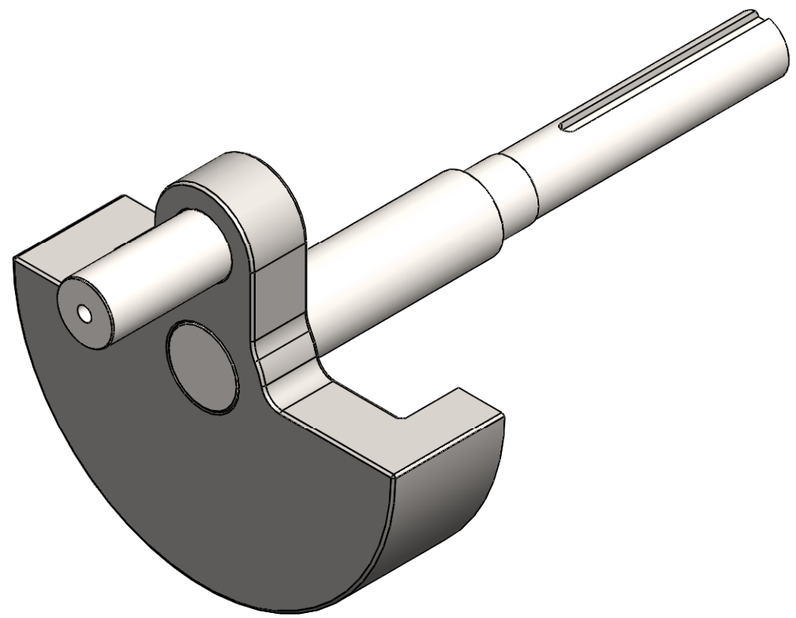
CRANKCASE

Material:	DWG NO.	REVISION
6061-T6 (SS)	C-CC-Z-01-CRANKCASE	A

DRW File: 028_C-CC-Z-01-CRANKCASE | Project: GSE-1 | Mass: 7309.56029 | SCALE:1:3 | SHEET 2 OF 2

ITEM NO.	DRW NUMBER	QTY.
1	C-CS-Z-03-CRANKWEB	1
2	C-CS-Z-01-SHAFT	1
3	C-CS-Z-02-PIVOT	1





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Thursday, April 14, 2016 11:00:04 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

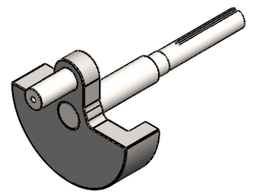
Crankshaft Assembly

Material: N/A

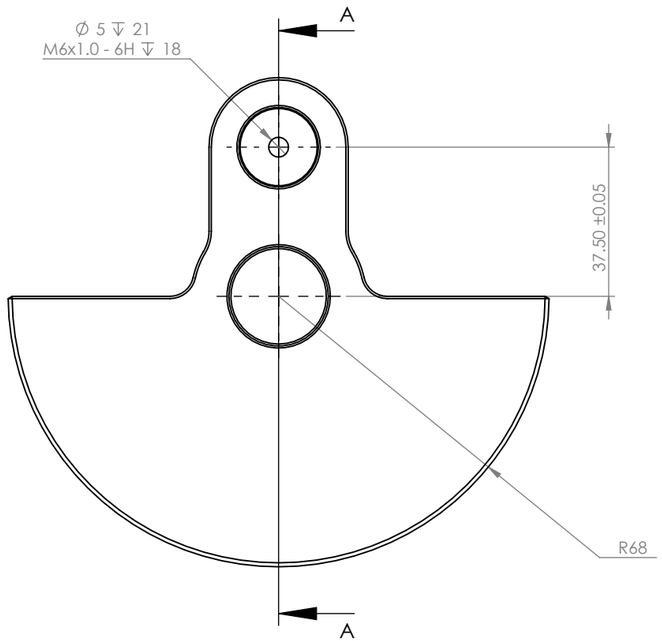
DWG NO. **C-CS-Z-00-CRANKSHAFT_ASM**

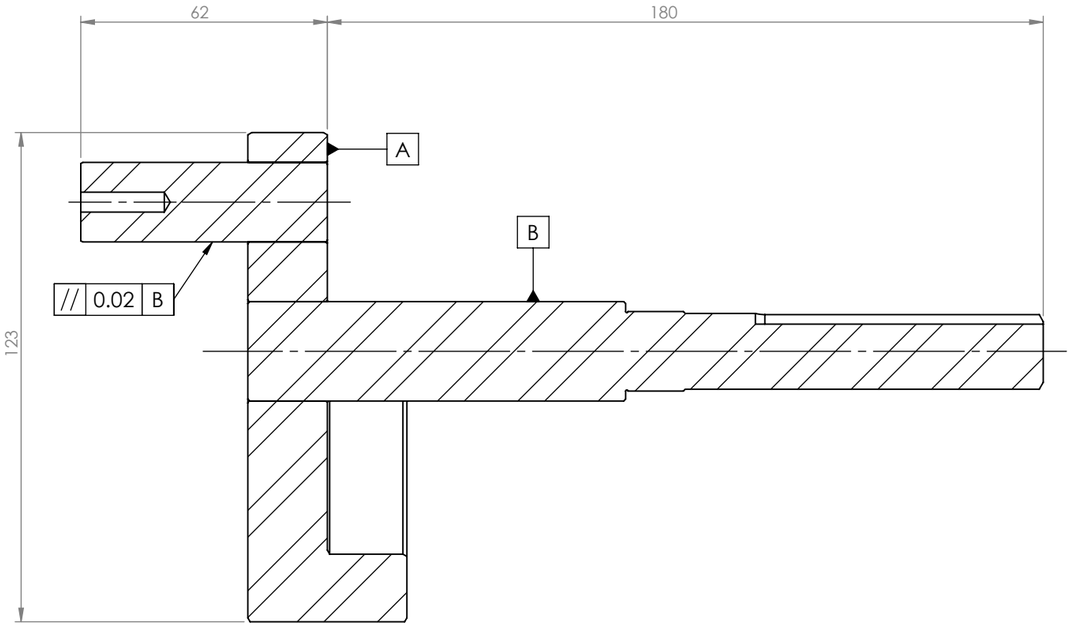
REVISION A

DRW File: 030_C-CS-Z-00-CRANKSHAFT_ASM | Project: **GSE-1** | Mass: | SCALE:2:3 | SHEET 1 OF 2



SCALE 1 : 5





SECTION A-A

Interference fit: allowance 1/350 of diameter.
Use deer tallow as a pressing aid.

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Thursday, April 14, 2016 11:00:04 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

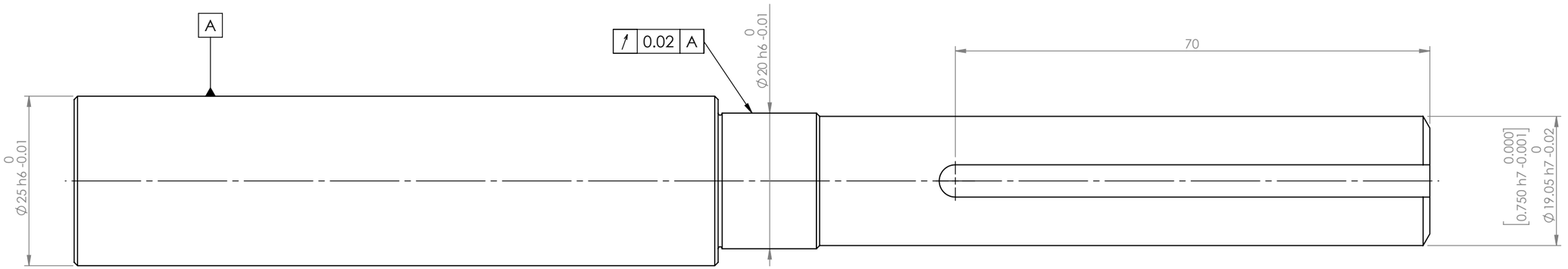
Crankshaft Assembly

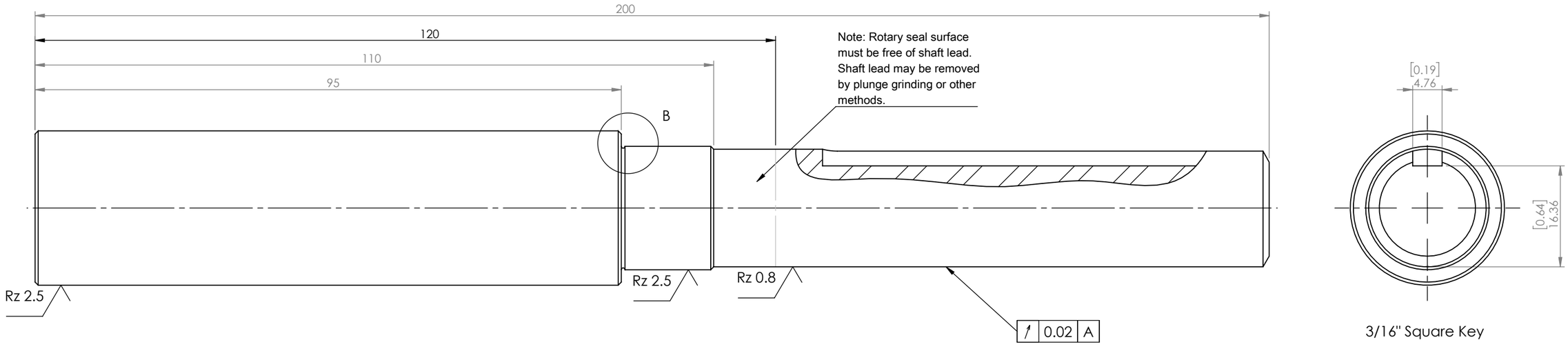
Material: N/A

DWG NO. **C-CS-Z-00-CRANKSHAFT_ASM**

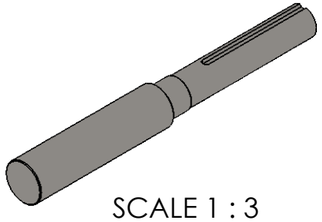
REVISION A

DRW File: 030_C-CS-Z-00-CRANKSHAFT_ASM | Project: **GSE-1** | Mass: | SCALE:2:3 | SHEET 2 OF 2

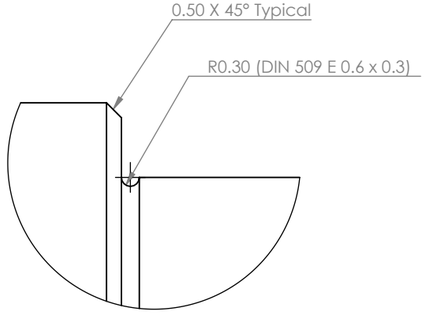




3/16" Square Key



SCALE 1 : 3



DETAIL B
SCALE 5 : 1

Please fillet/chamfer all edges to R0.5

General surface finish: √ Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 1482K49

Quantity: 1

Hardened to RHC 57 - 62
Thermex Metal Treating Ltd.

Thursday, April 14, 2016 11:20:31 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Jiacheng Yao
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

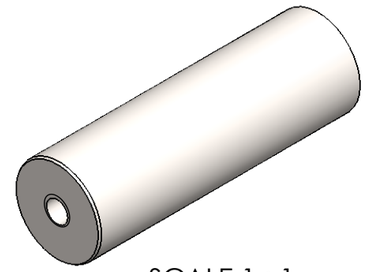
TITLE: **Crankshaft**

Material: Plain Carbon Steel

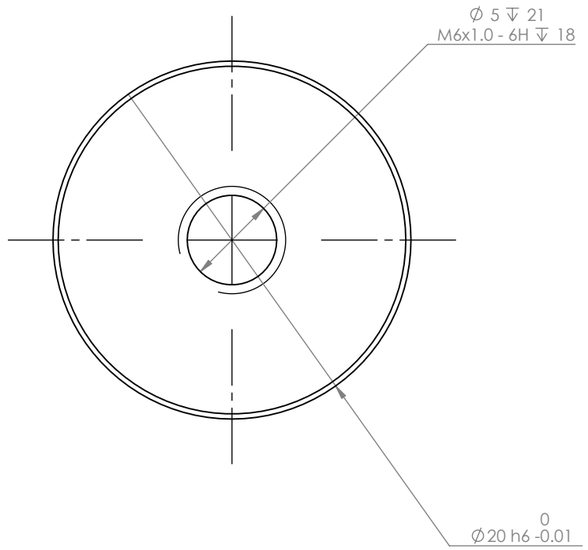
DWG NO. C-CS-Z-01-SHAFT

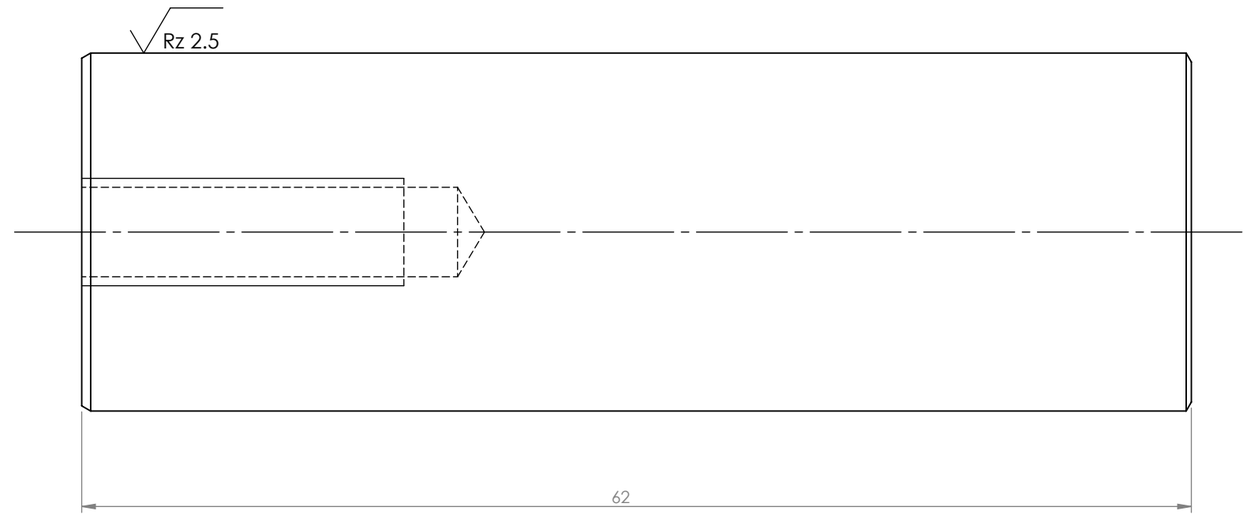
REVISION A

DRW File: 030_C-CS-Z-01-SHAFT | Project: GSE-1 | Mass: 593.03321 | SCALE:3:2 | SHEET 1 OF 1



SCALE 1 : 1





Please fillet/chamfer all edges to R0.5

General surface finish: √ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 1482K42

Quantity: 1

Hardened to RHC 57-62
Thermex Metal Treating Ltd.

Thursday, April 14, 2016 11:23:17 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

TITLE:

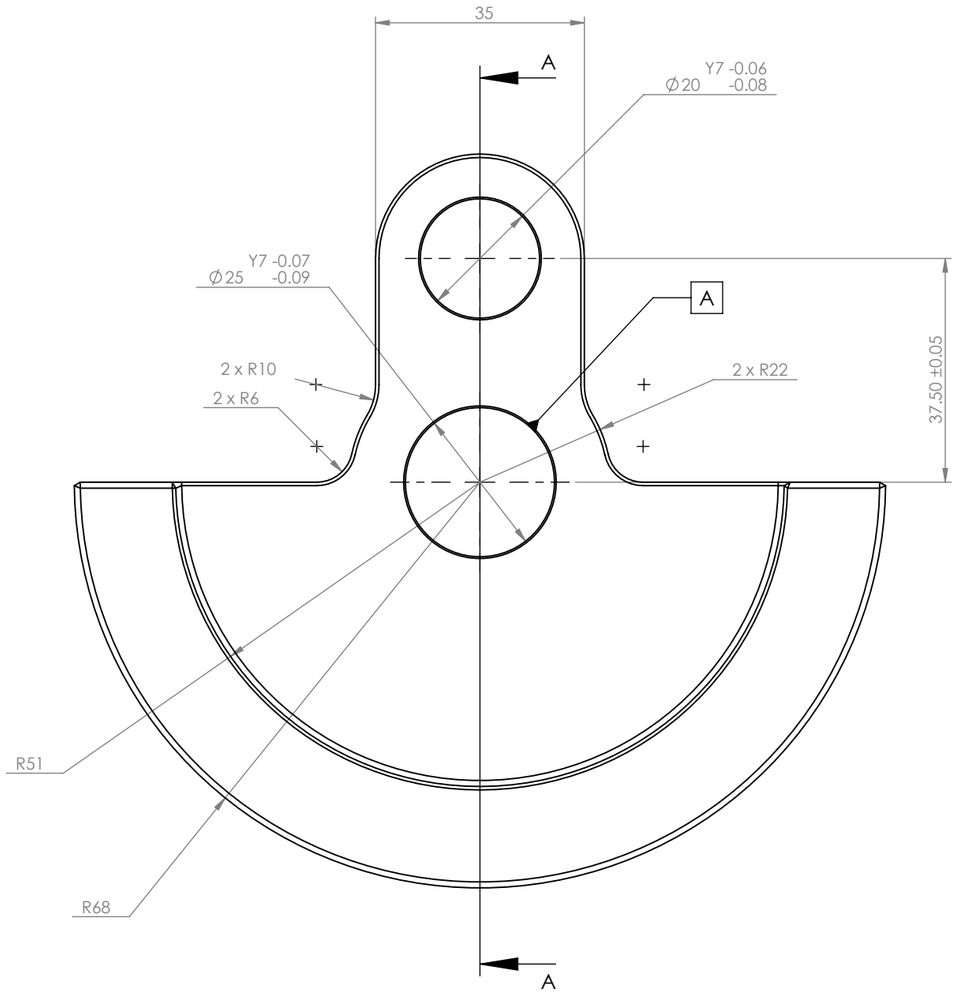
Connecting Rod Pivot

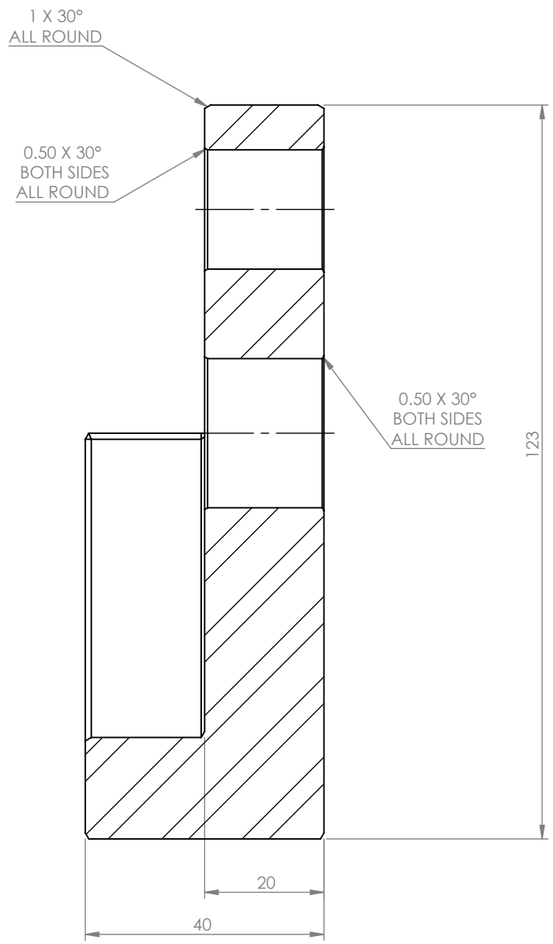
Material: Plain Carbon Steel

DWG NO. **C-CS-Z-02-PIVOT**

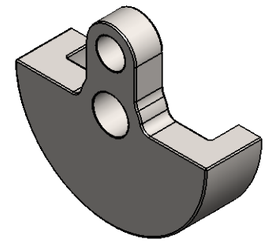
REVISION A

DRW File: 030_C-CS-Z-02-PIVOT | Project: **GSE-1** | Mass: 148.56472 | SCALE:3:1 | SHEET 1 OF 1

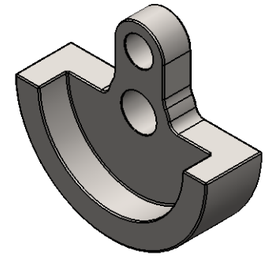




SECTION A-A



SCALE 1 : 3



SCALE 1 : 3

Please fillet/chamfer all edges to R1

General surface finish: $\sqrt{}$ Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 8910K102

Quantity: 1

Thursday, April 14, 2016 11:30:46 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Jiacheng Yao**
SOLID by	Jiacheng Yao
CHK'D	Connor Speer
APPV'D	NOT APPROVED

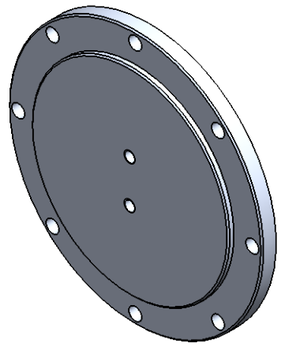
TITLE:

Crank Web

Material:	DWG NO.	REVISION
AISI 1020	**C-CS-Z-03-CRANKWEB**	A

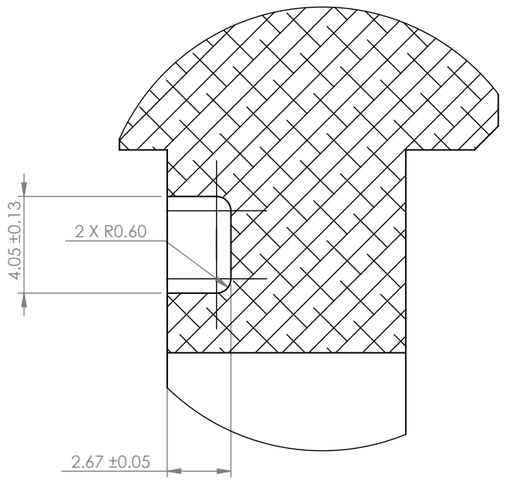
DRW File: 030_C-CS-Z-03-CRANKWEB | Project: **GSE-1** | Mass: 1818.39287 | SCALE:1:1 | SHEET 1 OF 1

C Other Parts

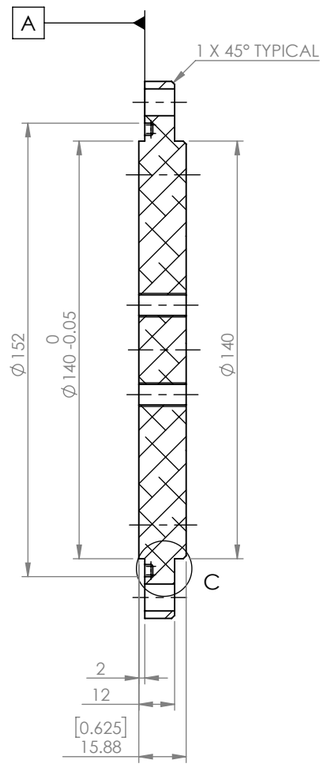


Scale 1:3

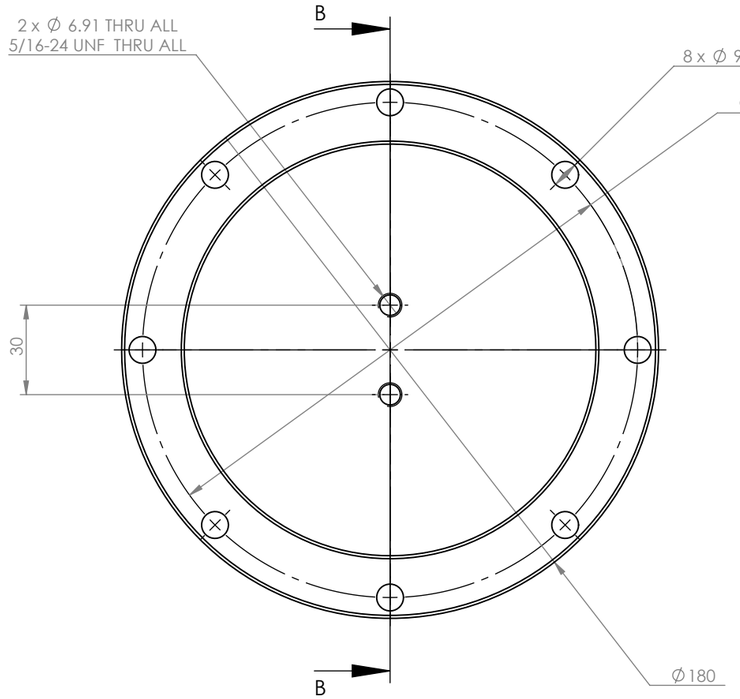
O-Ring Information
Size: 3.5x145mm
Hi-Tech Seals Part #: 35145
Material: Nitrile



DETAIL C
SCALE 4 : 1



SECTION B-B



Please fillet/chamfer all edges to R1

General Surface finish: √ Rz 10

O-ring groove surface finish: √ Rz 6.3

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Quantity: 1

5/8" Plate
Possible Supplier: McMaster-Carr 9246K41

Friday, April 15, 2016 9:54:06 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Connor Speer
SOLID by	Jiacheng Yao
CHK'D	Jiacheng Yao
APPV'D	NOT APPROVED

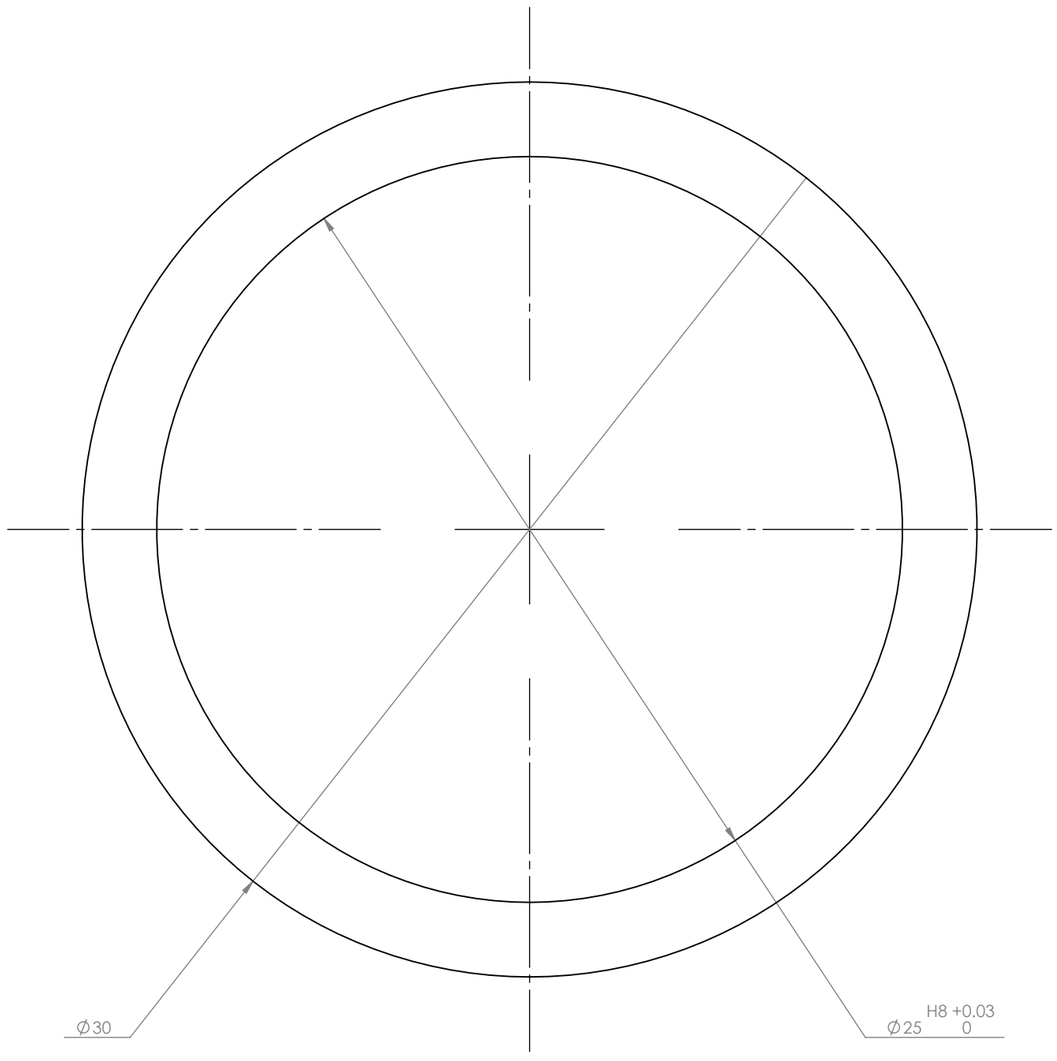
TITLE: **Crankcase Cover**

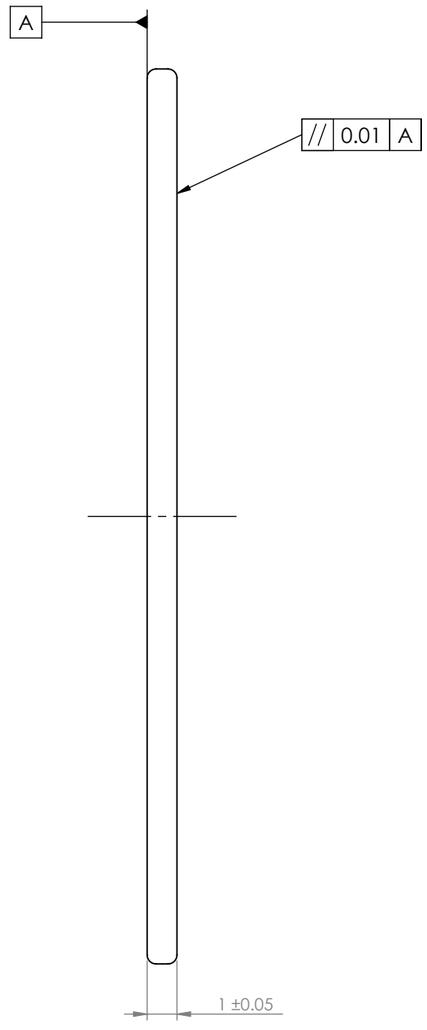
Material: 6061-T6 (SS)

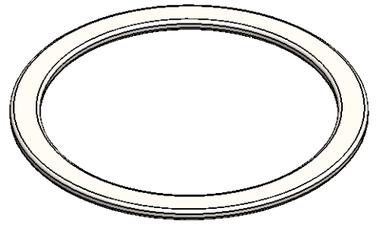
DWG NO. **A-DP-D-02-DISP_SHELL**

REVISION A

DRW File: 032_C-ZZ-Z-01-COVER | Project: **GSE-1** | Mass: 899.56745 | SCALE:1:2 | SHEET 1 OF 1







Scale 2:1

Please fillet/chamfer all edges to R0.3

General surface finish: √Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Quantity: 1

Possible Supplier: McMaster-Carr 4416T23

Friday, April 15, 2016 9:47:29 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

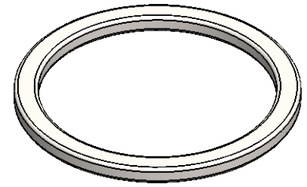
DRAWN	**Connor Speer**
SOLID by	Jiacheng Yao
CHK'D	Jiacheng Yao
APPV'D	NOT APPROVED

TITLE:

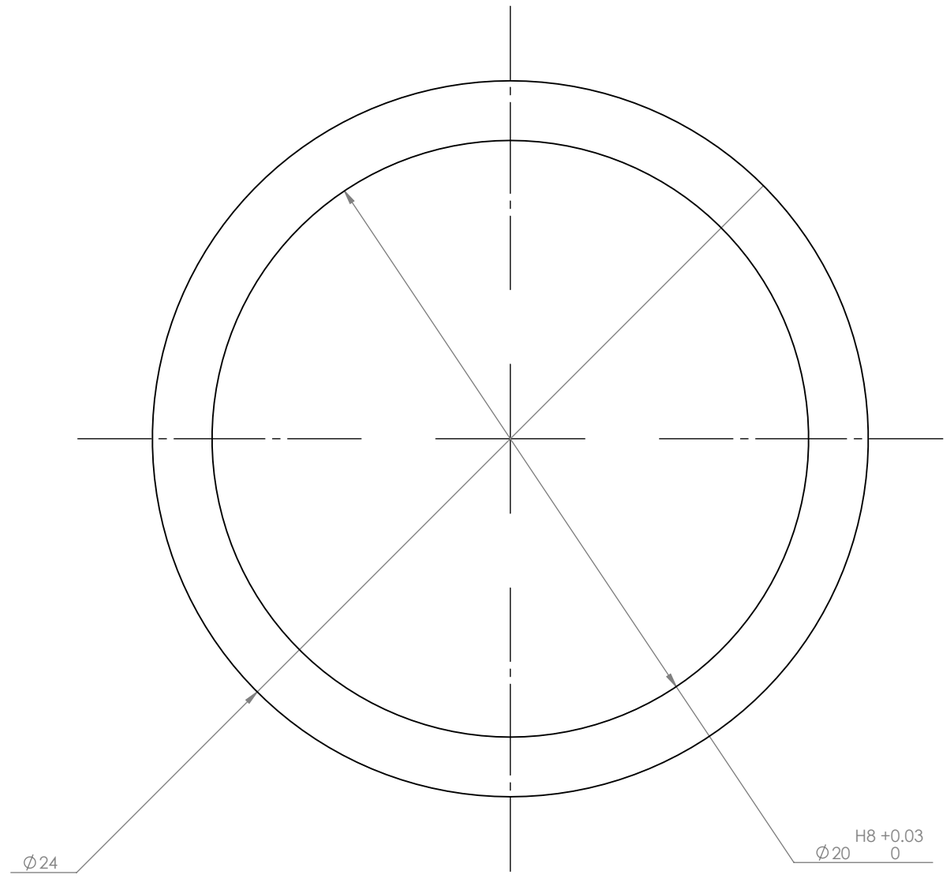
Crankshaft Spacer 115

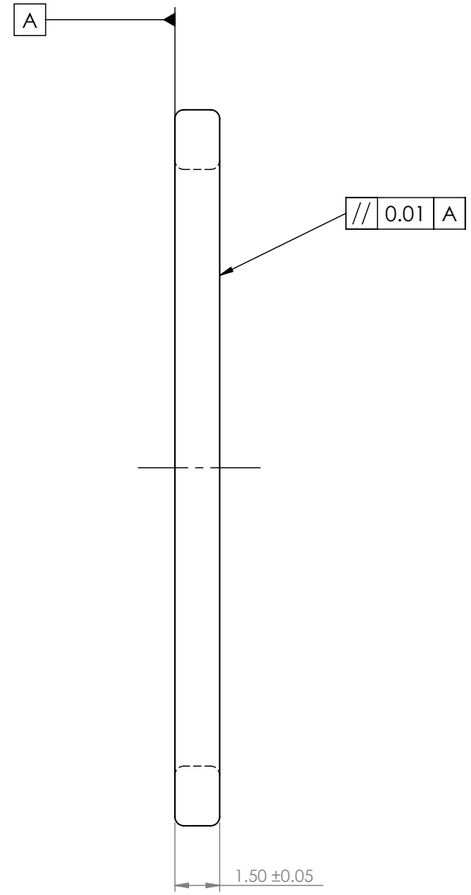
Material:	DWG NO.	REVISION
1.0715 (11SMn30)	**C-ZZ-Z-02-SPACER_115**	A

DRW File: 032_C-ZZ-Z-02-SPACER_115 | Project: **GSE-1** | Mass: 1.63262 | SCALE:5:1 | SHEET 1 OF 1



Scale 2:1





Please fillet/chamfer all edges to R0.3

General surface finish: Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 4416T23

Quantity: 1

Friday, April 15, 2016 9:49:47 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Connor Speer
SOLID by	Jiacheng Yao
CHK'D	Jiacheng Yao
APPV'D	NOT APPROVED

TITLE:

Crankshaft Spacer 116

Material: 1.0715 (11SMn30)

DWG NO. C-ZZ-Z-03-SPACER_116

REVISION A

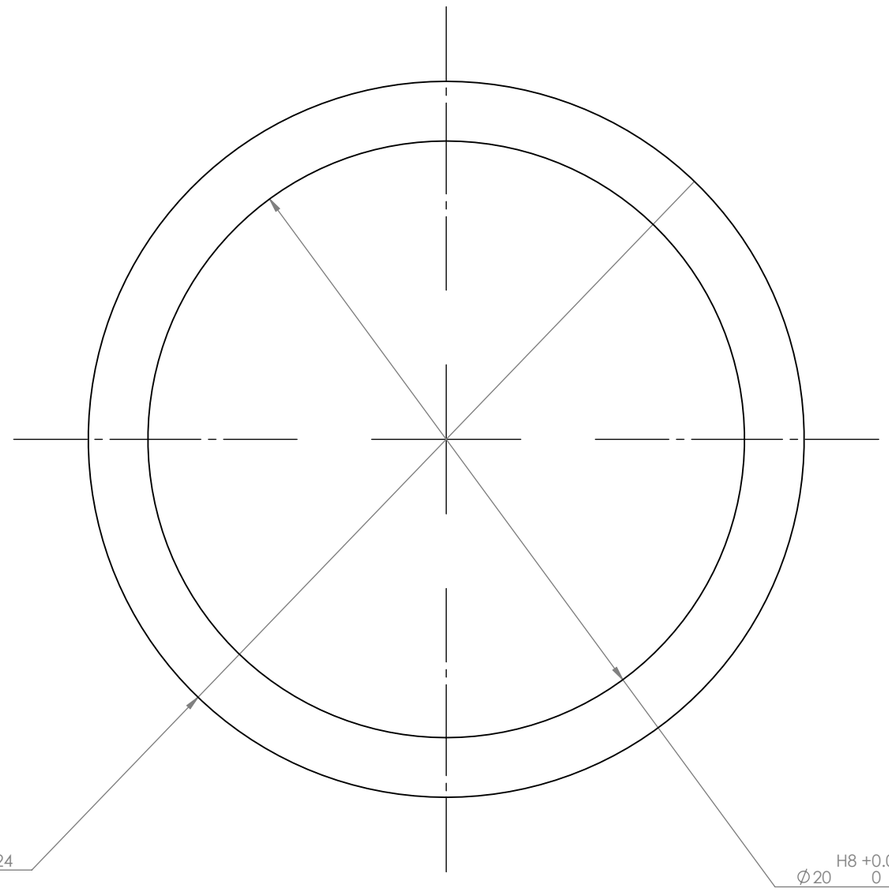
DRW File: 032_C-ZZ-Z-03-SPACER_116

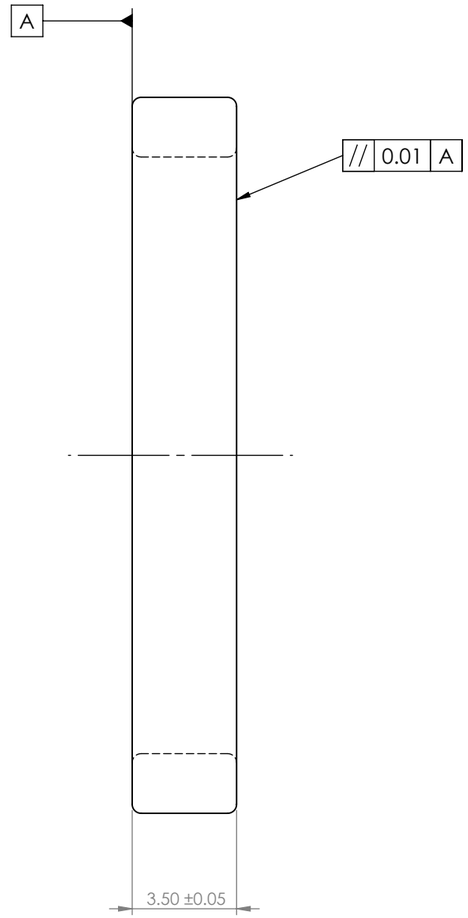
Project: GSE-1

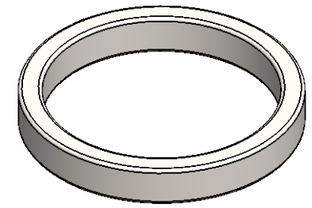
Mass: 1.57564

SCALE:5:1

SHEET 1 OF 1







Scale 2:1

Please fillet/chamfer all edges to R0.3

General surface finish: $\sqrt{}$ Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Quantity: 1

Possible Supplier: McMaster-Carr 4416T23

Friday, April 15, 2016 9:54:46 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

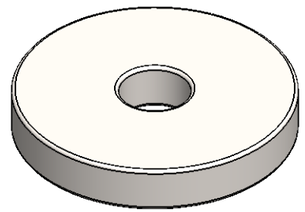
DRAWN	**Connor Speer**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

TITLE:

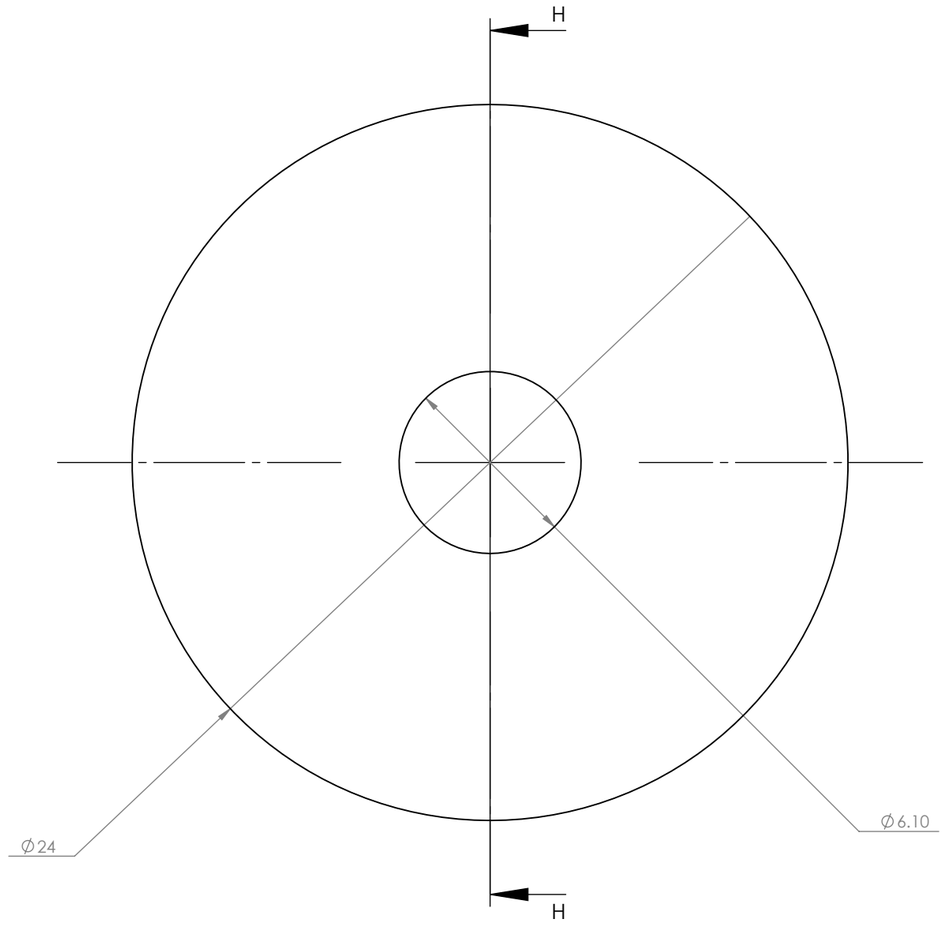
Crankshaft Spacer 117

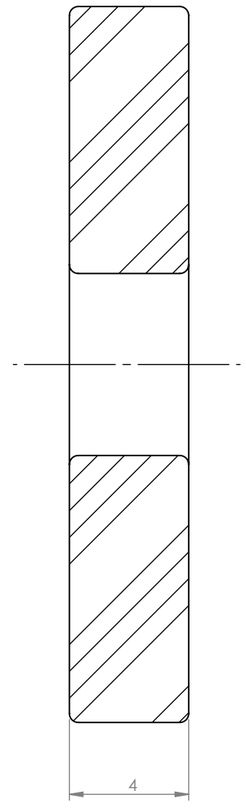
Material:	DWG NO.	REVISION
1.0715 (11SMn30)	**C-ZZ-Z-04-SPACER_117**	A

DRW File: 032_C-ZZ-Z-04-SPACER_117	Project: **GSE-1**	Mass: 3.73203	SCALE:5:1	SHEET 1 OF 1



Scale 2:1





SECTION H-H

Please fillet/chamfer all edges to R0.3

General surface finish: $\sqrt{}$ Rz 10

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Quantity: 1

Possible Supplier: McMaster-Carr 4416T23

Friday, April 15, 2016 9:51:17 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Connor Speer**
SOLID by	Jiacheng Yao
CHK'D	NOT CHECKED
APPV'D	NOT APPROVED

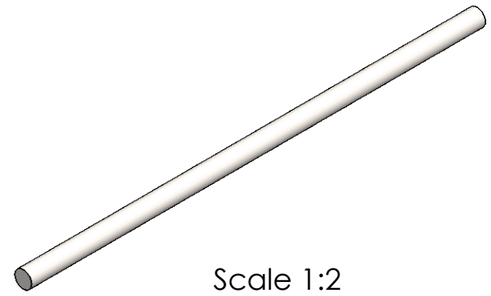
TITLE: **Spacer 118**

Material: 1.0715 (11SMn30)

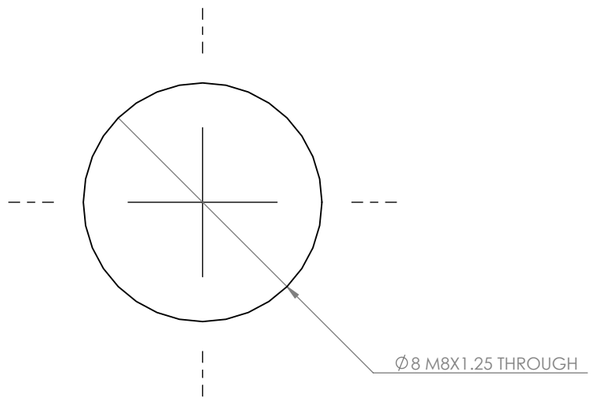
DWG NO. **C-ZZ-05-SPACER_118**

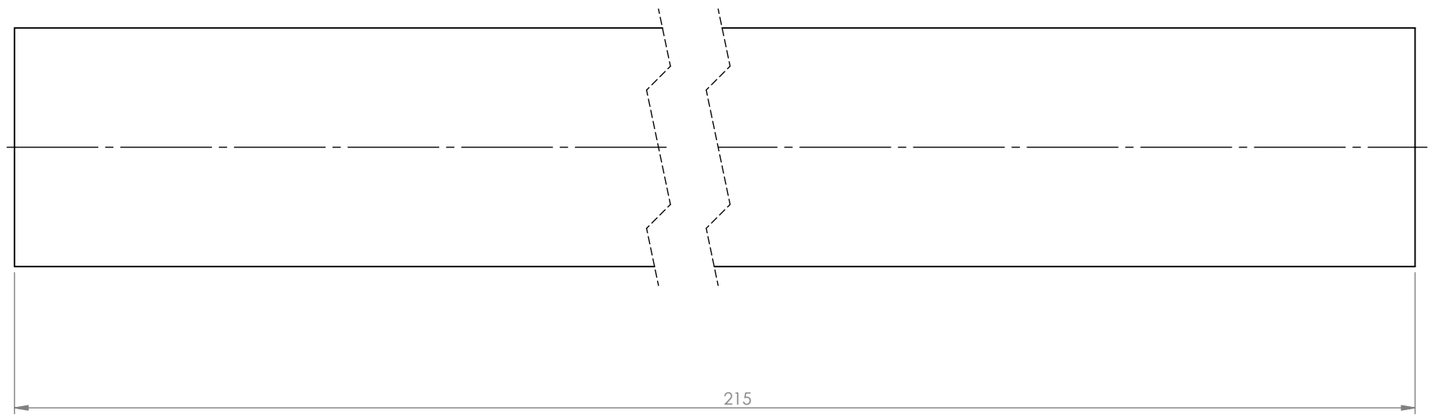
REVISION A

DRW File: 032_C-ZZ-Z-05-SPACER_118 | Project: **GSE-1** | Mass: 13.17425 | SCALE:5:1 | SHEET 1 OF 1



Scale 1:2





Quantity: 4 Threaded Rods

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments
Possible Supplier: McMaster-Carr 99055A125

1 meter rod cut to lengths
Grade 8.8, Uncoated
Threaded Steel Rod

Friday, April 15, 2016 9:53:25 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Connor Speer**
SOLID by	Jiacheng Yao
CHK'D	Jiacheng Yao
APPV'D	NOT APPROVED

TITLE:

Threaded Rod

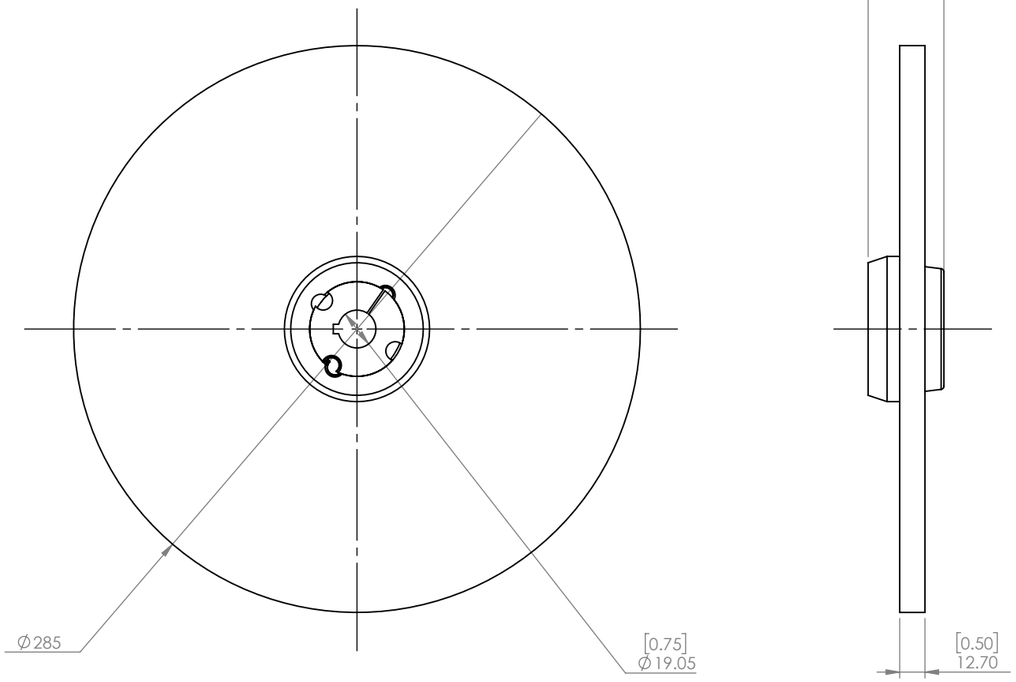
Material: Plain Carbon Steel

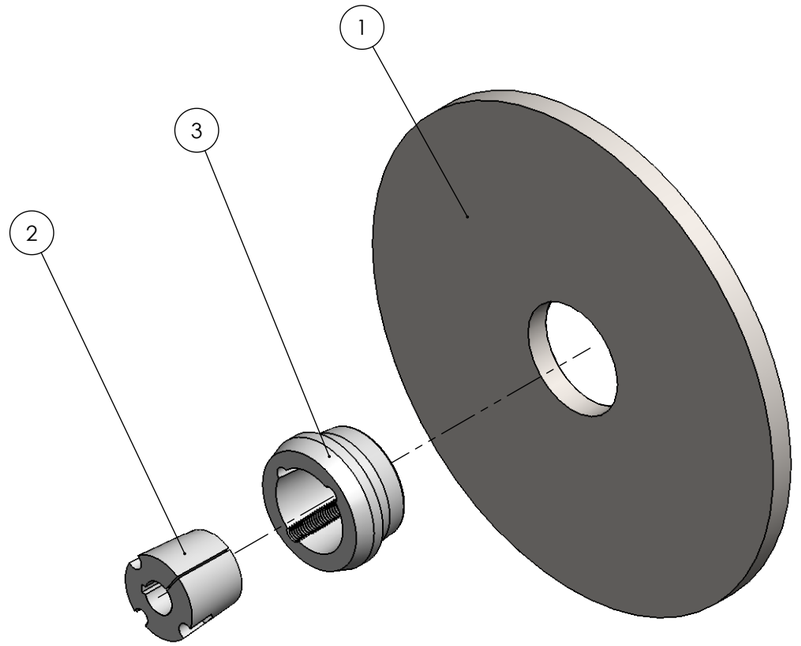
DWG NO. **C-ZZ-Z-06-THREADED ROD**

REVISION A

DRW File: 032_C-ZZ-Z-06-THREADED_ROD | Project: **GSE-1** | Mass: 84.29521 | SCALE:5:1 | SHEET 1 OF 1

ITEM NO.	DRW NUMBER	QTY.
1	C-ZZ-F-01-DISC	1
2	C-ZZ-F-02-TAPER_LOCK_BUSHING	1
3	C-ZZ-F-03-WELD_ON_BUSHING_BORE_HUB	1





Note: Dynamic balancing after assembly is recommended.

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 9:38:44 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	Connor Speer
SOLID by	Jiacheng Yao
CHK'D	Jiacheng Yao
APPV'D	NOT APPROVED

TITLE:

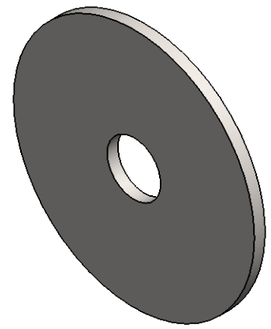
Flywheel Assembly

Material: N/A

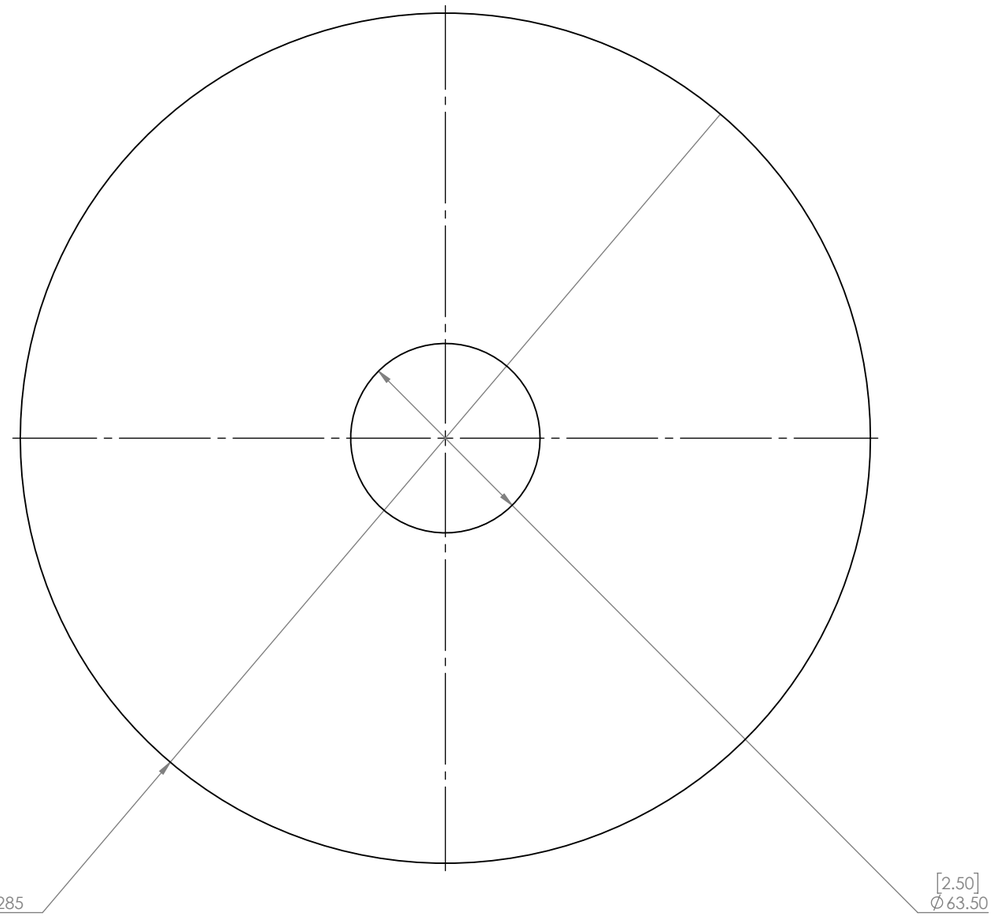
DWG NO. C-ZZ-F-00-FLYWHEEL_ASM

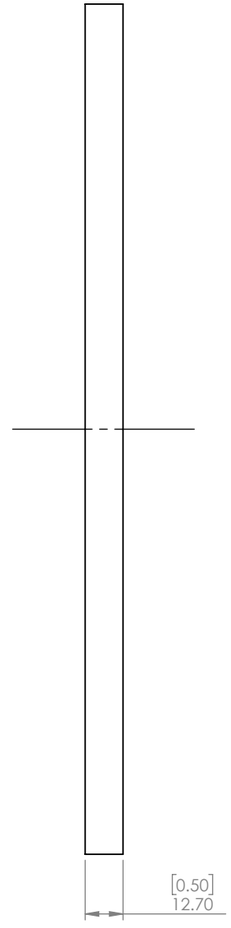
REVISION A

DRW File: 034_C-ZZ-F-00-FLYWHEEL_ASM | Project: GSE-1 | Mass: | SCALE:1:3 | SHEET 1 OF 1



Scale 1:5





UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

TOLERANCES:
LINEAR: 0.1
ANGULAR: 1.0°

Comments

Quantity: 1

Friday, April 15, 2016 9:33:52 AM

DO NOT SCALE DRAWING

NOBES RESEARCH GROUP

UASolve
TEC Edmonton
Department of Mechanical Engineering
UNIVERSITY OF ALBERTA

DRAWN	**Connor Speer**
SOLID by	Jiacheng Yao
CHK'D	Jiacheng Yao
APPV'D	NOT APPROVED

TITLE:

Flywheel Disk

Material:	DWG NO.	REVISION
AISI 1020	**C-ZZ-F-01-DISC**	A

DRW File: 034_C-ZZ-F-01-DISC	Project: **GSE-1**	Mass: 6082.71133	SCALE:1:2	SHEET 1 OF 1