Automated Mine Polygons & Production Scheduling Interface

Mohammad Tabesh and Hooman Askari-Nasab
Mining Optimization Laboratory (MOL)
University of Alberta, Edmonton, Canada

Abstract

We have developed a Matlab application to implement our clustering algorithms and mathematical models for long-term open-pit production planning. The application is designed to handle all the steps of importing blocks data from various sources, clustering blocks with two different techniques, setup MILP parameters and interpret and plot results. The application has a graphical user interface for easy use. In this document, we explain the details of programming and guidelines on how to use the application. Moreover, we review the implementations and outcomes by studying a small case study. The case study includes multiple scenarios for clustering the blocks for long-term and short-term planning and using the GUI to complete the tasks.

1. Step 1: Prepare ".dat" Block Model

In order to start working with the clustering application, we need a text file containing the block information. This file is a tab-separated text file with 3 sections: the header, column headers and the block data. It can be exported from any mine planning software and formatted to adhere to the following features. It has to be saved with a ".dat" extension. You can find a sample block model file in "2014_MILP/InputData/Test/Blocks.dat". Open the file in a text editor to have an example of formatting the block model file.

1.1. The Header

The header consists of 17 lines , see Table 1, ("-…" at the end of each line is a comment and is not imported into Matlab).

1.2. The Column Headers

Line number 18 in the file contains the column headers. The number of columns and their headers changes based on the block model properties such as number of elements. The columns should be formatted in the following order (Table 2).

1.3. The Block Model

The block data is listed under the header according the aforementioned columns. Bear in mind that no field is allowed to be left empty. Therefore, you have to use NA and 0 for unavailable string and number fields respectively. The block data is allowed to have multiple rows per block where a block consists of multiple rock types or it can be sent to multiple destinations. It is also allowed to have multiple rows per block when a block is divided between two periods or pushbacks. In this

Pourrahimian, Y., Ben-Awuah, E., and Askari-Nasab, H. (2015), Mining Optimization Laboratory (MOL), Report Six, © MOL, University of Alberta, Edmonton, Canada, Pages 250 , ISBN: 978-1-55195-356-4, pp. 225-240.

case, the import function will add up tonnages from the two rows. However, make sure that multiple rows describing a single block have to have a common block ID.

Table 1. Header Lines

Line #	Description	Example	Notes
1	Number of Destinations	4{tab}-Number of Destinations	
2	Destination Names	MILL{tab}Process{tab}W005{tab}-np-{tab}-Destination Names	The destination names should match the number of destinations in line 1 (e.g. –np- refers to not processed material and defined as a waste destination)
3	Number of Processes	2{tab}-Number of Processes	The first 2 destinations are assumed to be processes
4	Number of Waste Dumps	2{tab}-Number of Waste Dumps	The last 2 destinations are assumed to be waste dumps
5	Number of Stockpiles	0{tab}-Number of Stockpiles	Considered for later extensions
6	Number of Rock Types	7{tab}-Number of Rock Types	Number of rock types in the block model
7	Rock Types	2{tab}3{tab}5{tab}8{tab}101{tab}201{tab}301{tab}-Rock Types	The rock codes should match the number of rock types in line 6
8	Number of Element	3{tab}-Number of Elements	Number of elements in the block model
9	Elements	P{tab}S{tab}MWT{tab}-Elements	The elements should match the number of elements in line 8
10	Number of Elements Processed	1{tab}-Number of Elements Processed	Number of elements processed (not contaminants)
11	Number of Rock Types Processed	3{tab}-Number of Rock Types Processed	Number of rock types processed (mineralized rock types)
12	The origin of Indices	41{tab}68{tab}22{tab}-XI,YI,ZI Origin	Only saved for later references and not used for clustering operations (can be left 0,0,0)
13	The block dimensions	25{tab}25{tab}15{tab}-X,Y,Z Dimension	
14	The origin of coordinates	97525{tab}600200{tab}1440{tab}-X,Y,Z Origin	
15	Number of Blocks	19561{tab}-Number of Blocks	The number of blocks in the block model/The number of unique rows
16	Number of Rows	42117{tab}-Number of Rows	The number of rows in the file excluding the header rows
17	Number of Production Periods	20{tab}-Number of Periods	The number of production periods currently determined using mine planning tools

Table 2. Block Data Columns

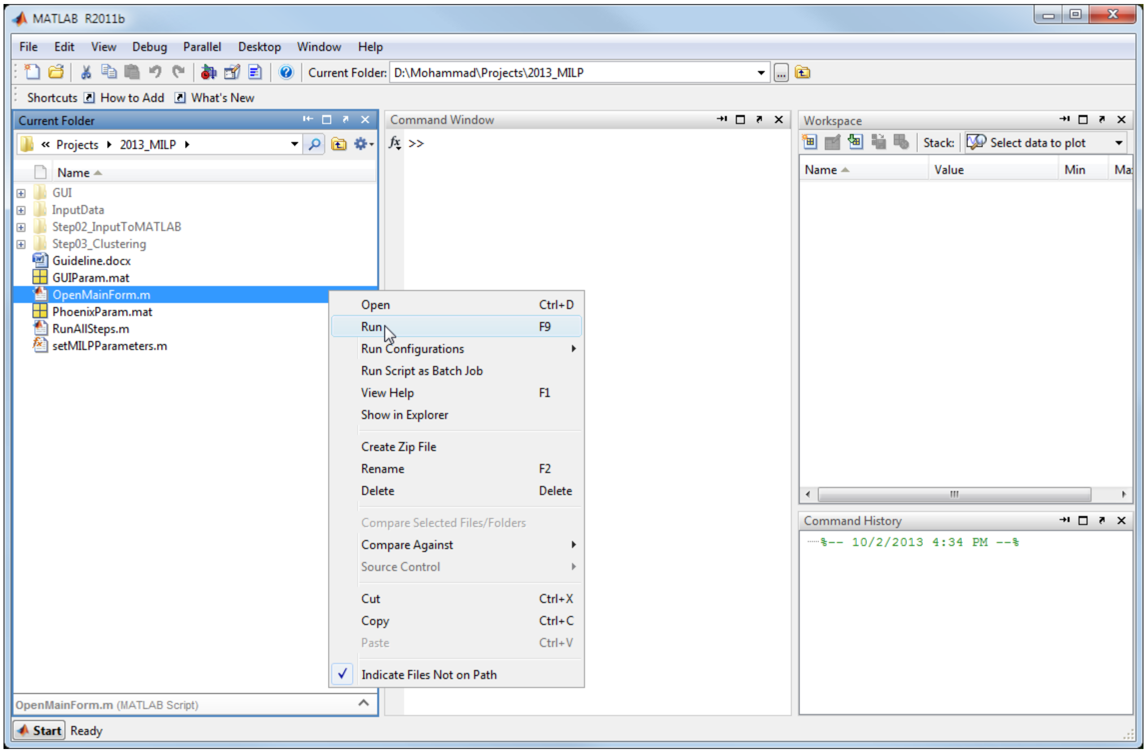
Column #	Header	Type	Description
1	BlockID	Integer	Unique block incremental ID number
2	IX	Integer	Block column index
3	IY	Integer	Block row index
4	IZ	Integer	Block elevation index
5	X	Float	Block X coordinate
6	Y	Float	Block Y coordinate
7	Z	Float	Block Z coordinate
8	PitID	Integer	Pit number (in case there are multiple pits in one model)
9	PhaseID	Integer	Push back number
10	BenchPhaseID	Integer	Bench-phase/panel number
11	numRockTypesInBlock	Integer	Number of rock types in the block
12	RockType	String	The rock type e.g. 101, HYPO
13	RockCode	Integer	The rock type order (defined in the header e.g. 5 for 101)
14	NumDestination	Integer	Number of possible destinations for the block
15	Destination	String	Block destination name e.g. MILL
16	DestinationCode	Integer	Block destination order e.g. 1 for MILL
17	BlockTonnage	Float	Total block tonnage
18	BlockValue	Float	Total block value if sent to this destination
19	Revenue	Float	Total revenue if sent to this destination
20	ProcessingCosts	Float	Total processing cost at this destination
21	MiningCostAndHaulage	Float	Total cost of mining and hauling to this destination
22	BlockReferenceMiningCost	Float	Total cost of mining and hauling to this destination
23	MineralizedTonnage	Float	Total extractable mineralized tonnage from this rock type (considering dilution e.g. 0.95*block tonnage)
24	OreTonnage	Float	Total Ore tonnage (mineralized tonnage above cut-off) from this rock type
25	WasteTonnage	Float	Total waste rock in the block
26	TotalWasteTonnage	Float	Total waste tonnage (waste rock + undefined waste + dilution)
27	ProcessThroughPut	Float	Process throughput at this destination for this block
28	BlockFraction	Float	Fraction of block extracted in this period
29	Period	Integer	Period number
30	{element}_Quantity	Float	Element quantity in the block (tonnage)
31	{element}_Grade	Float	Element grade in the block (%mass)
32	{element}_Recovery	Float	Element recovery at this destination (0-1)
33	{element}_Revenue	Float	Element revenue (in addition to block processing revenue)
34	{element}_ProcessingCost	Float	Element processing cost (in addition to block processing cost)
…	Repeat 30 to 34 for each element		
	DestinationFraction	Float	Fraction of block sent to this destination in this period

The clustering algorithm works based on a single element in most cases and it is easier to have one element imported into Matlab. However, it is possible to define multiple elements, import them into Matlab and try different combinations when using the clustering procedure. Many of the columns introduced earlier are defined to account for MILP formulations and are not needed for clustering. Therefore, they can be left as NA or 0 based on their data type.

2. Step 2: Import to Matlab

The next step is to import the block model into Matlab with the specified format.

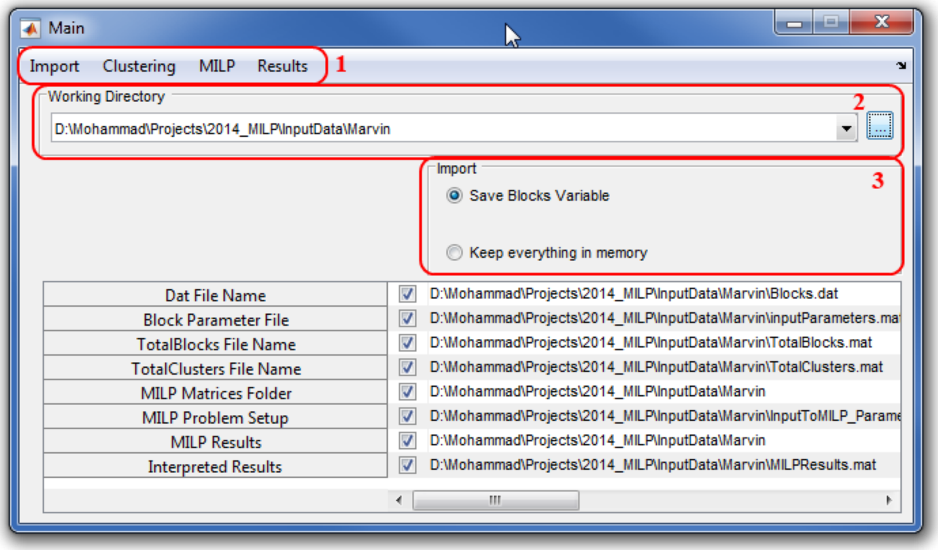
1. Open Matlab
2. Change the active directory to “2014_MILP”
3. Right-click on the script file “OpenMainForm.m” and click on “Run” as in Fig 1. It will open the GUI developed for running the clustering algorithm.



```
%-- 10/2/2013 4:34 PM --%
```

Fig 1. Opening GUI

The main GUI has three sections to be explained separately: the menus, the active directory and memory usage option. The menus are used to navigate through different parts of the application. However, using the application starts by setting the working directory. Results of every step are loaded from and saved into the active directory. There are two options for memory usage: “Save Blocks Variable” and “Keep Everything in memory”. The first option takes more time since it loads blocks variable, performs the operation, saves the results and clears memory every time. This helps the application consume less memory. In contrast, the second option keeps the blocks variable in the memory and the user has to save it if needed. There is also a read-only table on the main GUI that checks which files exist in the active directory.



Working Directory

D:\Mohammad\Projects\2014_MILP\InputData\Marvin

Import

● Save Blocks Variable

○ Keep everything in memory

Dat File Name	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin\Blocks.dat
Block Parameter File	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin\inputParameters.ma
TotalBlocks File Name	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin\TotalBlocks.mat
TotalClusters File Name	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin\TotalClusters.mat
MILP Matrices Folder	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin
MILP Problem Setup	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin\InputToMILP_Parame
MILP Results	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin
Interpreted Results	☑	D:\Mohammad\Projects\2014_MILP\InputData\Marvin\MILPResults.mat

Fig 2. Main GUI

The first step is to create a directory and set it as the active directory.

1. Click on the "…" button in section 2 of the GUI
2. Browse to the desired directory (e.g. 2014_MILP/InputData)
3. Press the "New Folder" button and give it a name

Afterwards, you have to import the block file into Matlab. Choose the "Import DAT File" command from the "Import" menu and browse to the tab-separated block model file. It will create a copy of the file in the active directory and import the block data into Matlab structured variable. It will then save the TotalBlocks variable to hard disk or keep it in memory based on the option chosen. The "Load Existing TotalBlocks" and "Save TotalBlocks" menus can be used to manually load and save the blocks variable when the keep in memory option is chosen.

3. Step 3: Clustering

Two clustering methods are currently available in the software: horizontal hierarchical and k-means. They can be called from the clustering menu on the upper left corner of the main GUI.

- If the "Save Blocks Variable" option is chosen:

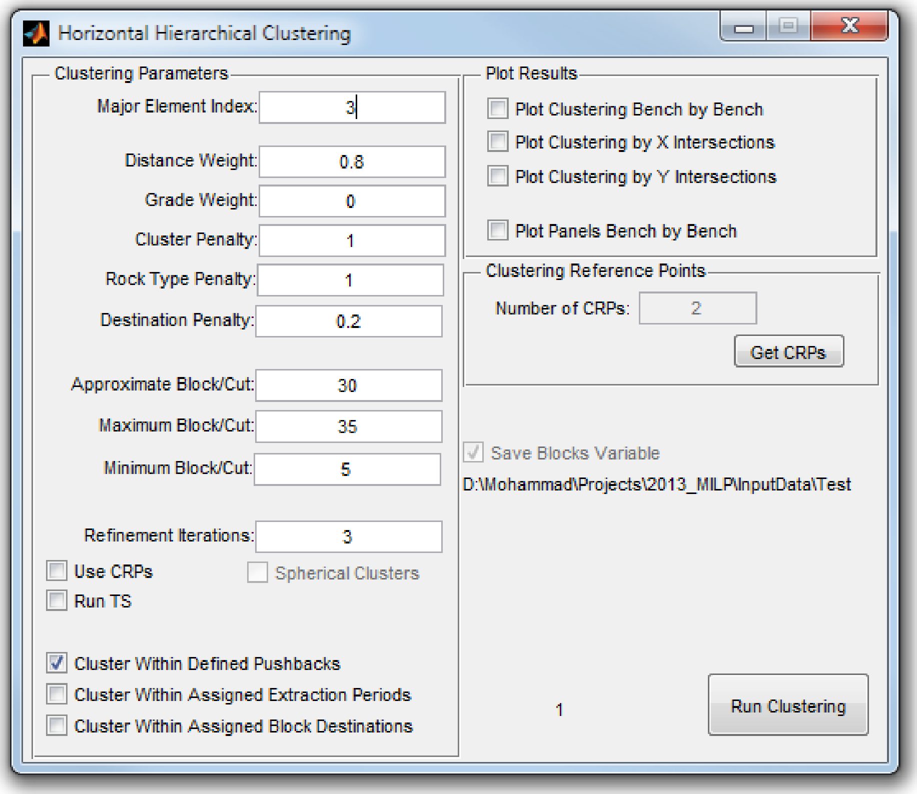
The clustering algorithms automatically load the TotalBlocks variable. However, if the program is not able to find the TotalBlocks.mat file in the active directory the user has to use the "Load Existing TotalBlocks" menu to locate the TotalBlocks variable which will be copied to the active directory then and loaded when the clustering algorithm starts.

- If the "Keep in Memory" option is chosen:

The user has to load the TotalBlocks variable if it is not already loaded in the memory by using "Load Existing TotalBlocks" menu and browsing to the TotalBlocks.mat file.

3.1. Horizontal Hierarchical Clustering

The two algorithms and their control parameters are explained in the following sections. The clustering GUI (Fig 3) contains five textboxes to enter the weight and penalty parameters as explained in Appendix I.



Horizontal Hierarchical Clustering

Clustering Parameters

Major Element Index: 3

Distance Weight: 0.8

Grade Weight: 0

Cluster Penalty: 1

Rock Type Penalty: 1

Destination Penalty: 0.2

Approximate Block/Cut: 30

Maximum Block/Cut: 35

Minimum Block/Cut: 5

Refinement Iterations: 3

□ Use CRPs □ Spherical Clusters

□ Run TS

☑ Cluster Within Defined Pushbacks

□ Cluster Within Assigned Extraction Periods

□ Cluster Within Assigned Block Destinations

Plot Results

□ Plot Clustering Bench by Bench

□ Plot Clustering by X Intersections

□ Plot Clustering by Y Intersections

□ Plot Panels Bench by Bench

Clustering Reference Points

Number of CRPs: 2

☑ Save Blocks Variable

D:\Mohammad\Projects\2013_MILP\InputData\Test

1

Fig 3. Hierarchical Clustering GUI

- Major Element Index: Since the data structure is able to handle multiple elements you need to indicate which element grade to use for calculating the grade difference between the blocks. This textbox is considered to input the order index of the element of interest in the input file (3 represents MWT in our example).

- Distance Weight: The weight put on distance and direction measure in calculating the similarities (W_D)

- Grade Weight: The weight put on the major element grade difference measure in calculating the similarities (W_G)

- Cluster Penalty: The penalty value for the blocks located above different clusters in calculating the similarities (C_{ij})

- Rock Type Penalty: The penalty value for the blocks from different rock types in calculating the similarities (R_{ij})

- Destination Penalty: The penalty value for the blocks determined to be sent to different destinations in calculating the similarities (N_{ij})

- There are also 3 textboxes considered for average, maximum and minimum blocks per cut (cluster) that control the cluster sizes.

- Refinement Iterations: The number of shape refinement iterations to be performed after the clustering

- Use CRPs: If checked the user has to provide CRPs for directional clustering, otherwise the M_{ij} will be set to 1.

- Spherical Clusters: This checkbox will be enabled if “Use CRPs” is checked. If this is checked, the mining direction factor will be calculated.

- Run TS: The Tabu search (TS) procedure will be called, if this checkbox is checked, in order to reduce the number of arcs between formed clusters in each bench and the ones in the bench below. This helps planning stage get to the bottom of the pit faster. However, experiments of various datasets have shown that reducing the precedence arcs by manipulating the cluster shapes decreases the homogeneity of the clusters and the practicality of the production plans.

- There are 3 checkboxes in the lower left side of the GUI for clustering within boundaries. If the phase (pushback) IDs, production periods or destinations are assigned for the blocks in the input file they can be used to apply strict boundaries on the clustering. This means blocks from different regions cannot be merged to form a cluster.

- There are 4 checkboxes in the “Plot Results” panel that can be used to plot the clustering results after the clustering is performed. They will produce plots of clustering bench by bench, X intersections and Y intersection. However, for a real-size dataset there will be many figures generated and it is not recommended to plot them at the same time in this GUI. Another GUI is designed for plotting the results in plan views and cross sections. The last checkbox “Plot Panels Bench by Bench” plots the panels/bench phases to be able to compare against the results when “Clustering within Defined Pushbacks” is desired.

- The number of CRP points is currently fixed to 2 but the textbox is considered for future expansions on the algorithm. The GUI has a button to add the CRPs to each bench. It starts from the lowest bench and continues to the higher benches. The red squares represent blocks on each bench and the user can use the cross hair to specify the start and end point of the mining direction vector. The same way, the user can specify 2 points for what we call the spherical clustering.

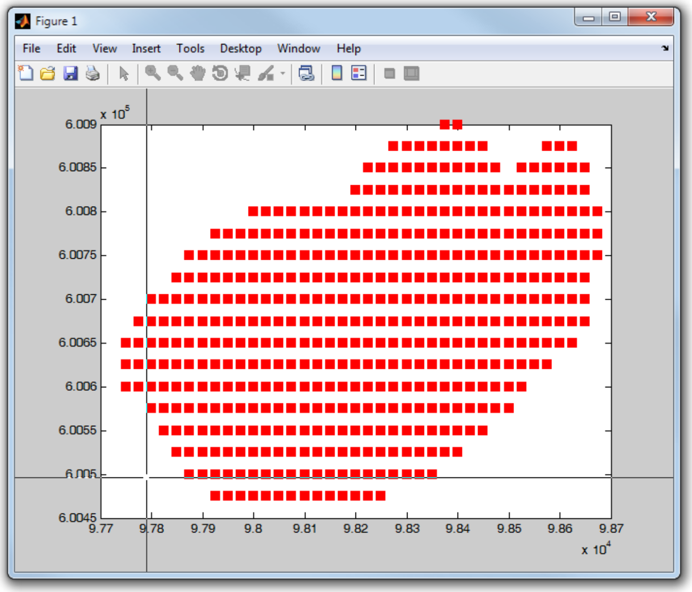
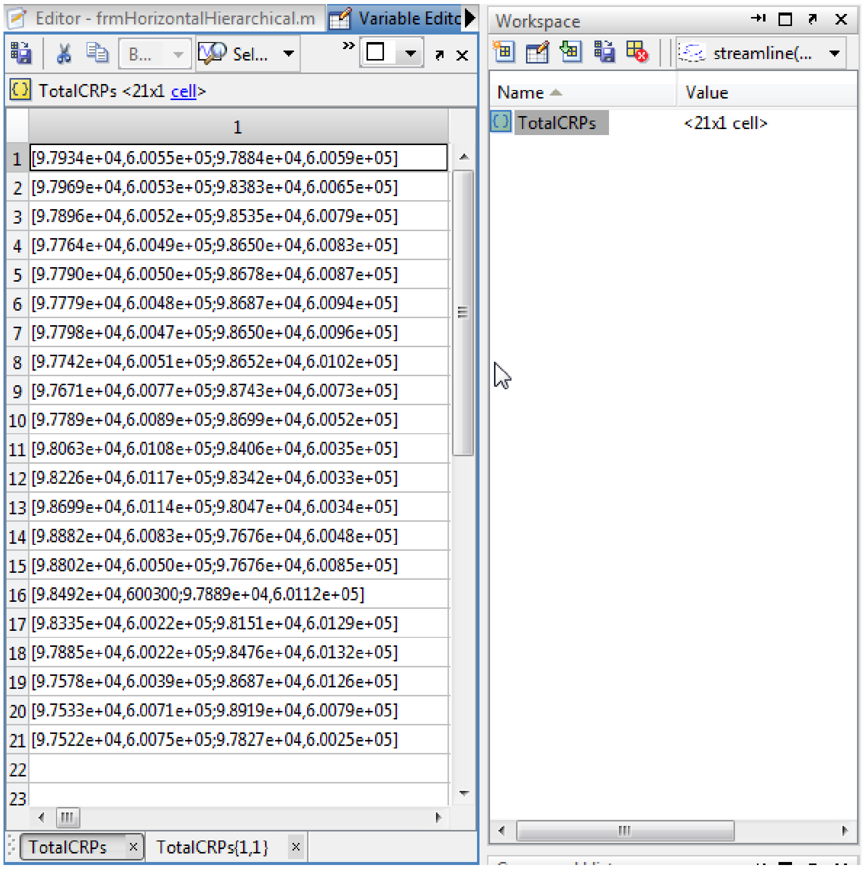


Fig 4. Get CRPs Plot

There are instances that the user prefers to enter exact coordinates for CRPs; for example, ramp entrance coordinates can be used as CRPs for spherical clusters. The CRPs are saved in a variable called "TotalCRPs.mat" in the active directory. If the user needs to change the values or input them manually, the file can be opened in Matlab. The TotalCRPs variable is a cell array with a cell per bench starting from the lowest bench. Each cell contains a 2 × 2 matrix with the coordinates of the two reference points as in Fig 5 and Fig 6.



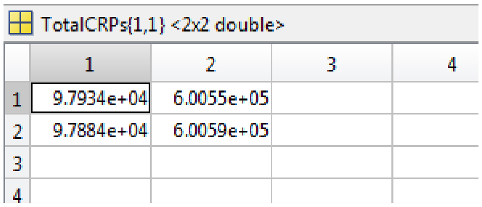
TotalCRPs <21x1 cell>

	1
1	[9.7934e+04,6.0055e+05;9.7884e+04,6.0059e+05]
2	[9.7969e+04,6.0053e+05;9.8383e+04,6.0065e+05]
3	[9.7896e+04,6.0052e+05;9.8535e+04,6.0079e+05]
4	[9.7764e+04,6.0049e+05;9.8650e+04,6.0083e+05]
5	[9.7790e+04,6.0050e+05;9.8678e+04,6.0087e+05]
6	[9.7779e+04,6.0048e+05;9.8687e+04,6.0094e+05]
7	[9.7798e+04,6.0047e+05;9.8650e+04,6.0096e+05]
8	[9.7742e+04,6.0051e+05;9.8652e+04,6.0102e+05]
9	[9.7671e+04,6.0077e+05;9.8743e+04,6.0073e+05]
10	[9.7789e+04,6.0089e+05;9.8699e+04,6.0052e+05]
11	[9.8063e+04,6.0108e+05;9.8406e+04,6.0035e+05]
12	[9.8226e+04,6.0117e+05;9.8342e+04,6.0033e+05]
13	[9.8699e+04,6.0114e+05;9.8047e+04,6.0034e+05]
14	[9.8882e+04,6.0083e+05;9.7676e+04,6.0048e+05]
15	[9.8802e+04,6.0050e+05;9.7676e+04,6.0085e+05]
16	[9.8492e+04,600300;9.7889e+04,6.0112e+05]
17	[9.8335e+04,6.0022e+05;9.8151e+04,6.0129e+05]
18	[9.7885e+04,6.0022e+05;9.8476e+04,6.0132e+05]
19	[9.7578e+04,6.0039e+05;9.8687e+04,6.0126e+05]
20	[9.7533e+04,6.0071e+05;9.8919e+04,6.0079e+05]
21	[9.7522e+04,6.0075e+05;9.7827e+04,6.0025e+05]
22	
23	

TotalCRPs × TotalCRPs{1,1} ×

Name	Value
TotalCRPs	<21x1 cell>

Fig 5. TotalCRPs Variable with 21 Benches



TotalCRPs{1,1} <2x2 double>

	1	2	3	4
1	9.7934e+04	6.0055e+05		
2	9.7884e+04	6.0059e+05		
3				
4				

Fig 6. CRP Coordinates for the First Bench (X,Y)

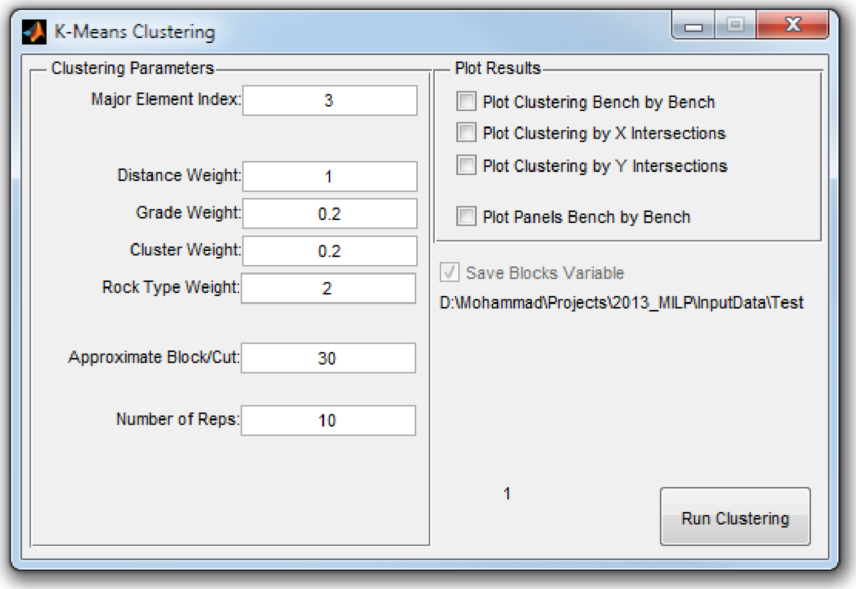
After running the clustering algorithm the user may want to "Save the Clustering Results" if working with "Keep in Memory" option and then use the "Plot and Export" GUI to plot the results or export them into a text file.

3.2. K-Means Clustering

A famous example of partitional algorithms is the k-means, which attempts to find cluster means and assign data points to the closest mean. K-means is a partitioning technique which tries to find a good partitioning scheme by iteratively modifying the partitions. One extension to k-means clustering which is relevant to this project is the kernel k-means which is developed to be able to partition data points which are not linearly separable by mapping them into a kernel space. The clustering technique used is an implementation of the kernel k-means algorithm based on gradient descent search. In this approach, K initial cluster centers are randomly selected at each replication. Then the objects are assigned to the nearest center. Afterwards, based on the gradient descent search technique, the centers are manipulated in such a way that the summation of distances between the objects and the means is locally minimized. Another replication is then started with a new random set of means and the process continues for a limited number of replications.

The first step for this algorithm is to form the feature matrix, which holds all the important properties of all objects. To be consistent with the hierarchical clustering technique, the same parameters are used with the same weighting approach. Then the matrix has to be kernelized in order to get better results. When objects are not linearly separable in their initial space, kernel functions are used to map data points from the initial space to the kernelized space and do the clustering in there. Then the same map is used in returning to the initial space with all the objects labeled as belonging to various clusters. Having tested various kernel functions and parameters, a polynomial kernel function with $d = 1$ is used in this implementation. Afterwards, K initial cluster centers are randomly selected in the kernelized space and objects are assigned to the closest mean. Then the objective function, which is a summation of Euclidean distances between all objects and cluster means, is calculated. Cluster means are then manipulated in an iterative manner based on gradient descent until a local minimum is found. This is stored as a solution to the clustering problem and a new replication starts with another random definition of cluster means. Finally, all of the replications are compared, and the one with the lowest objective function is selected as the solution to the clustering problem on that bench [3]. The problem with the k-means approach is that there is no size control. In addition, it is possible to generate fragmented clusters that are problematic if used in later planning stages.

The k-means algorithm GUI (Fig 7) can be called from the GUI, clustering menu. Instead of penalty values, there are only weights in k-means GUI. The X, Y, grade and rock type are put in a matrix and each are powered to their weights. User can also input the average blocks per cut that determines the number of center points in the k-means algorithm. The last parameter is the number of replications. More replications usually results in better results but by talking more time. The rest of the k-means GUI works the same as the horizontal hierarchical GUI.



Major Element Index: 3

Distance Weight: 1

Grade Weight: 0.2

Cluster Weight: 0.2

Rock Type Weight: 2

Approximate Block/Cut: 30

Number of Reps: 10

□ Plot Clustering Bench by Bench

□ Plot Clustering by X Intersections

□ Plot Clustering by Y Intersections

□ Plot Panels Bench by Bench

☑ Save Blocks Variable

D:\Mohammad\Projects\2013_MILP\InputData\Test

1

Fig 7. K-Means Clustering GUI

3.3. Plot and Export

After performing the clustering with hierarchical or k-means algorithms, the user may need to evaluate the clustering results by looking at plan views or cross sections, or export the results to a text file to import into other applications. A GUI, which can be called from the clustering menu, is developed for this purpose (Fig 8). The GUI consists of three main panels: Clustering Stats, Plot and Export.

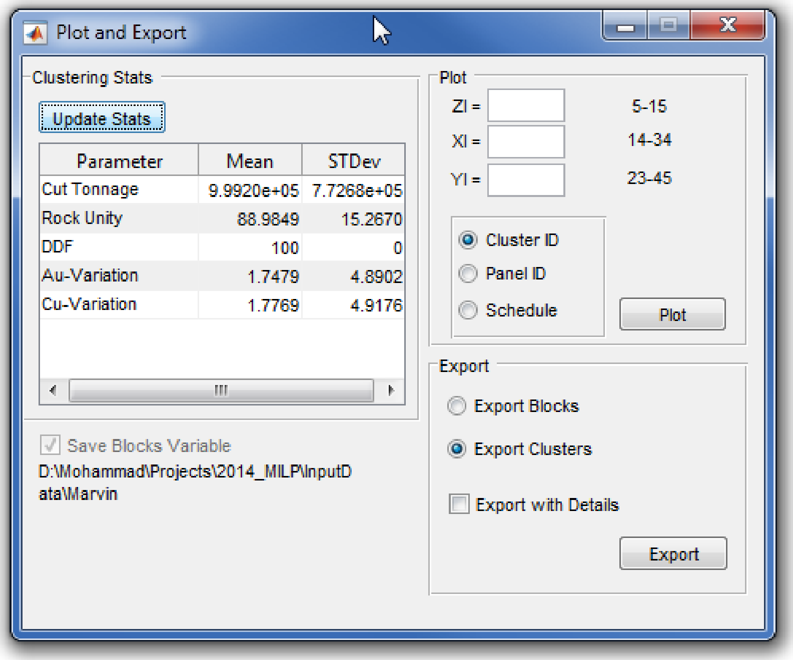
The first panel is designed to provide a summary of the clustering measures to be able to quickly evaluate clustering scheme and change the parameters to get better results. The clustering statistics table is updated by pressing the “Update Stat” button on the upper left corner. Once updated, it will provide the user with the following information:

- Cut Tonnage: Average and the standard deviation of the tonnage of material in the cuts is the first evaluation criterion in the table. This can be used to check if the differences in the tonnage of material in the generated cuts are reasonable.

- Rock Unity: Based on the structure of the problem and existing criteria for categorical variables, a new index is defined as the percentage of rocks in a mining-cut belonging to the most dominant rock type in that mining-cut. This is called the rock unity and is depicted in second row of the table.

- DDF: Destination Dilution Factor is defined in the same way as the rock unity but by considering the predetermined block destination as the homogeneity factor.

- Element Variation: The coefficient of variation (CV) is defined as the standard deviation of a variable divided by its mean. The average and standard deviation of the CV values for each element is presented in the following rows. Average CV can represent the variations in the grade values among the blocks grouped together. This can be helpful when creating mining polygons for estimating the head grade to the processing plant.



Parameter	Mean	STDev
Cut Tonnage	9.9920e+05	7.7268e+05
Rock Unity	88.9849	15.2670
DDF	100	0
Au-Variation	1.7479	4.8902
Cu-Variation	1.7769	4.9176

☑ Save Blocks Variable
D:\Mohammad\Projects\2014_MILP\InputData\Marvin

ZI =
XI =
YI =

● Cluster ID
○ Panel ID
○ Schedule

○ Export Blocks
● Export Clusters
□ Export with Details

Fig 8. Plot and Export GUI

The "Plot" panel can be used to plot the clustering results in plan views, cross sections and in 3D. The minimum and maximum index values for the blocks are presented in front of each textbox. The user can choose to filter the blocks to be plotted by entering the index value in each textbox. The application will create a 3D plot of the dataset if all textboxes are left empty. Otherwise, a plan view (if ZI is specified) or cross section (if XI or YI is specified) will be created. Note that the user cannot enter a range in the filter textboxes but only integer numbers. There are three options to plot clusters, panels and the schedule from the input file.

The "Export" panel is developed to export the clustering results into tab-delimited text files to be used as input to other software. The "Export Blocks" option creates a file with a row for each block with indices, coordinates, cluster IDs etc. The columns in the text file are presented in Table 3 The "Export Clusters" option provides a file with the same format but with one row for each cluster. The indices and coordinates presented for clusters are the average values of the blocks in that cluster.

Table 3. Blocks Brief Output File

Column Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Value	XI	YI	ZI	X	Y	Z	Panel ID	Cluster ID	Cluster ID in Bench	Period	Destination

There is also a checkbox labeled "Export with Details" that adds extra columns to the original format and can be mostly useful for cluster exports. This option can be used to get detailed information about the quality of the generated clusters such as their rock unity, destination dilution factor and element grade variations. It will also provide the precedence arcs between the generated clusters in different benches. This can be especially useful when evaluating the Tabu Search results. The extra columns are presented in Table 4 . After column 20, there are 2 columns for each element in the dataset: mean and variance of grades.

Table 4. Block Detailed Output File

Column Number	**1**	**…**	**13**	**14**	**15**	**16**
Value	ID	…	Phase ID	Rock Unity	Destination Dilution Factor	Total Tonnage
Column Number	17	18	19	20	…	
Value	Downward Relation Count	Upward Relation Count	ID in Bench	Bench ID	Average Grade	Grade Variance

4. Prepare Input to MILP

We have to prepare the matrices for the MILP formulation based on four different resolutions: cluster-panel, cluster-cluster (CC), block-cluster (BC) and block-block (BB). This can be done through the menu items under MILP menu as shown in Fig 9.

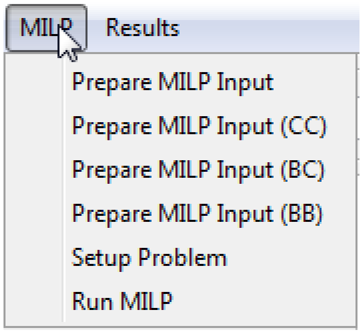


Fig 9. MILP Menu

5. Setup MILP Parameters

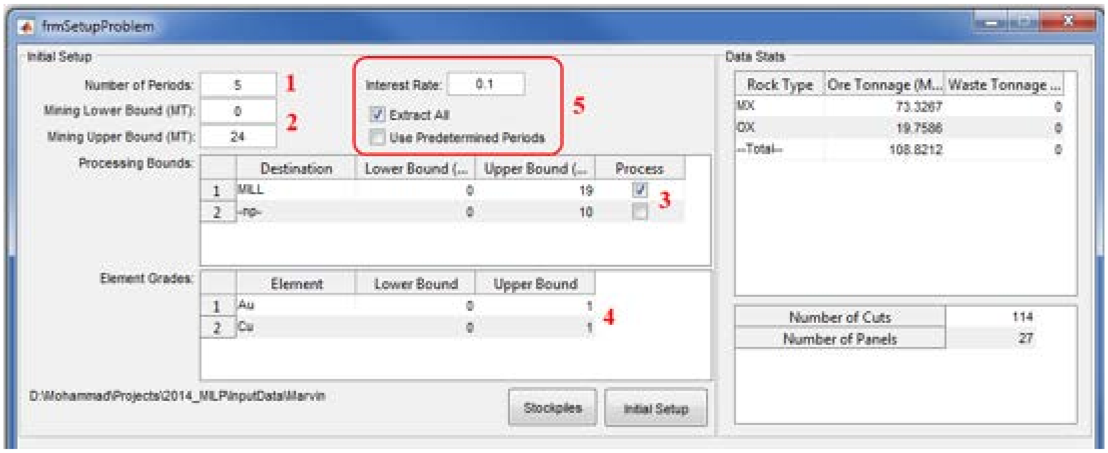
We have developed a GUI for setting up the MILP by changing the parameters in two user-friendly forms. The first form is called the setup form and includes any parameters related to the MILP except than the stockpiles. The stockpile settings are in a separate form that can be opened from the main setup form. The setup form can be opened by the "Setup Problem" item under the MILP menu.

5.1. Initial Setup

The first step for setting up the MILP parameters is to define the number of periods and the default values for mining and processing constraints. The initial setup panel on the top-left corner of the form (Fig 10) is where the initial settings have to be defined. We start by defining the number of periods in the textbox marked with number 1 in Fig 10. Two textboxes, market with number 2 in Fig 10, are designed to input the default values for lower and upper limits on the mining capacity. The table marked with number 3 in Fig 10 presents the list of destinations defined in the input file. We can check the Process checkbox for processing destinations (the ones that have a limited capacity) and assign lower and upper bounds for them. Note that the lower and upper bound values are in millions. The table marked with number 4 in Fig 10 presents the elements in the block model and provides the option to assign default lower and upper values for average grade constraints for each element. Capacities and average grade bounds can be edited for each period in the next step. The interest rate is a fixed number for the mine life and can be provided in the textbox market with number 5. If the checkbox "Extract All" is checked the model will be forced to extract everything within the final pit instead of determining what to mine and what to leave in ground based on NPV. The checkbox "Use Predetermined Periods" will limit the extraction variables to predetermined periods for each panel and cut in cases where a multi-step solution is required. This will be explained in more details later. The "Data Stats" panel on the top-right corner is the summary of the block model to be used for determining the proper capacities.

5.2. Period by Period Setup

Pressing the "Initial Setup" button will replicate the default values for lower and upper limits for the number of periods and make the corresponding tables visible. Now we can change the constraints for different periods in the three tables marked with 1 to 3 in Fig 11. The numbers in these three tables are not in millions in contrast to the default values.



Initial Setup

Number of Periods: 5

Mining Lower Bound (MT): 0

Mining Upper Bound (MT): 24

Interest Rate: 0.1

☑ Extract All

☐ Use Predetermined Periods

Processing Bounds:

	Destination	Lower Bound (...	Upper Bound (...	Process
1	MILL	0	19	☑
2	-np-	0	10	☐

Element Grades:

	Element	Lower Bound	Upper Bound
1	Au	0	1
2	Cu	0	1

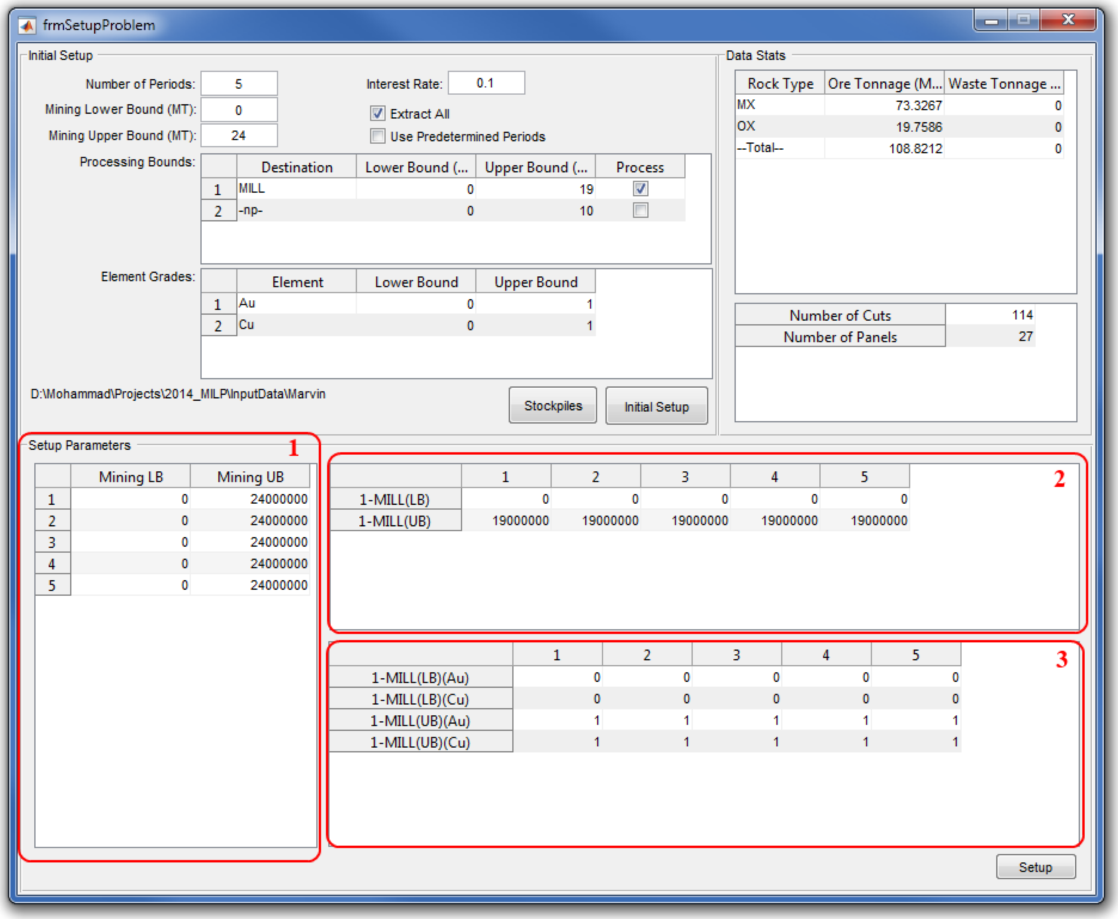
D:\Mohammad\Projects\2014_MILP\InputData\Marvin

Data Stats

Rock Type	Ore Tonnage (M...	Waste Tonnage ...
MX	73.3267	0
OX	19.7586	0
--Total--	108.8212	0

Number of Cuts	114
Number of Panels	27

Fig 10. Initial Setup Panel



Initial Setup

Number of Periods: 5

Mining Lower Bound (MT): 0

Mining Upper Bound (MT): 24

Interest Rate: 0.1

☑ Extract All

☐ Use Predetermined Periods

Processing Bounds:

	Destination	Lower Bound (...	Upper Bound (...	Process
1	MILL	0	19	☑
2	-np-	0	10	☐

Element Grades:

	Element	Lower Bound	Upper Bound
1	Au	0	1
2	Cu	0	1

D:\Mohammad\Projects\2014_MILP\InputData\Marvin

Data Stats

Rock Type	Ore Tonnage (M...	Waste Tonnage ...
MX	73.3267	0
OX	19.7586	0
--Total--	108.8212	0

Number of Cuts	114
Number of Panels	27

Setup Parameters

	Mining LB	Mining UB
1	0	24000000
2	0	24000000
3	0	24000000
4	0	24000000
5	0	24000000

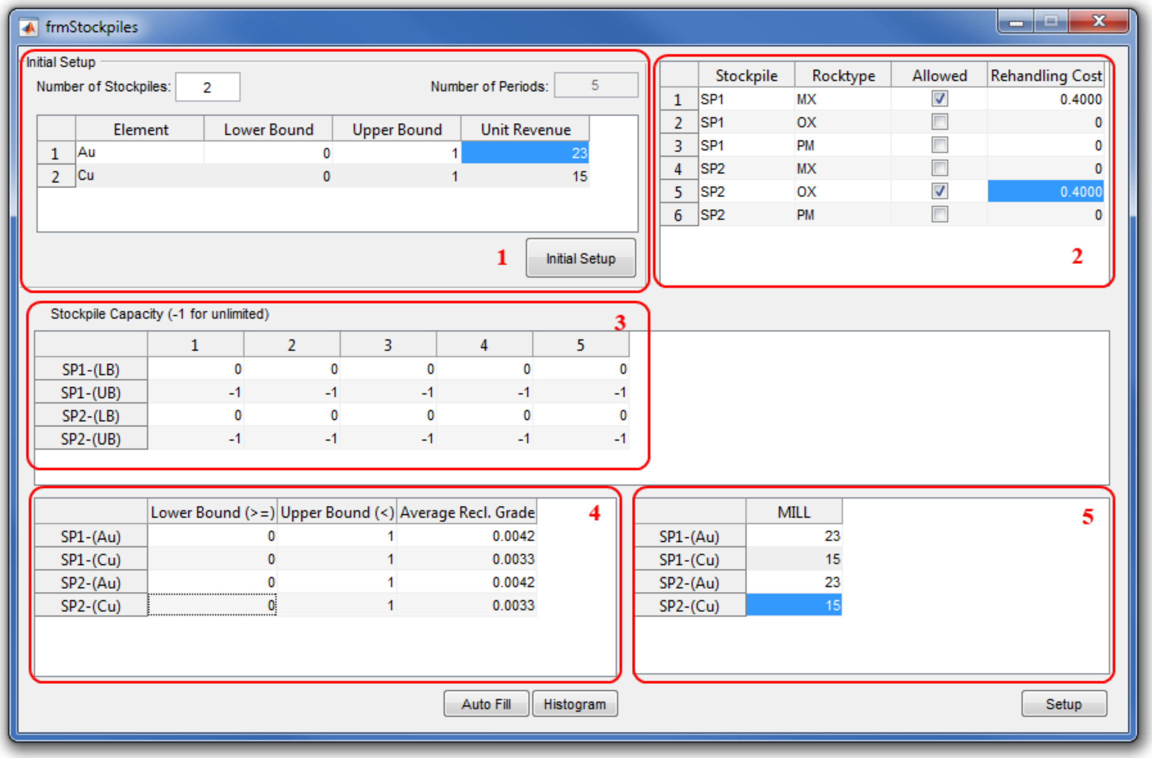
	1	2	3	4	5
1-MILL(LB)	0	0	0	0	0
1-MILL(UB)	19000000	19000000	19000000	19000000	19000000

	1	2	3	4	5
1-MILL(LB)(Au)	0	0	0	0	0
1-MILL(LB)(Cu)	0	0	0	0	0
1-MILL(UB)(Au)	1	1	1	1	1
1-MILL(UB)(Cu)	1	1	1	1	1

Fig 11. Period by Period Setup

5.3. Stockpile Setup

Following the same design as the main setup form, the stockpile setup requires an initial setup with the number of stockpiles and default values for lower and upper bounds on average grades as well as the reclamation revenues (Fig 12). After setting the number of stockpiles and pressing the initial setup, we can limit stockpiles to rock types and assign re-handling cost per ton of material reclaimed in the top-right table market with number 2. Moreover, we can limit the tonnage sent to the stockpile in each period through the table marked with number 3. The lower and upper bounds on the average grade of material sent to the stockpile and the reclamation grade for each stockpile and element can be set in table 4. The revenue made from processing elements in the stockpile can be set in table 5.



Number of Stockpiles: 2

Number of Periods: 5

	Element	Lower Bound	Upper Bound	Unit Revenue
1	Au	0	1	23
2	Cu	0	1	15

	Stockpile	Rocktype	Allowed	Rehandling Cost
1	SP1	MX	☑	0.4000
2	SP1	OX	☐	0
3	SP1	PM	☐	0
4	SP2	MX	☐	0
5	SP2	OX	☑	0.4000
6	SP2	PM	☐	0

Stockpile Capacity (-1 for unlimited)

	1	2	3	4	5
SP1-(LB)	0	0	0	0	0
SP1-(UB)	-1	-1	-1	-1	-1
SP2-(LB)	0	0	0	0	0
SP2-(UB)	-1	-1	-1	-1	-1

	Lower Bound (>=)	Upper Bound (<)	Average Recl. Grade
SP1-(Au)	0	1	0.0042
SP1-(Cu)	0	1	0.0033
SP2-(Au)	0	1	0.0042
SP2-(Cu)	0	1	0.0033

	MILL
SP1-(Au)	23
SP1-(Cu)	15
SP2-(Au)	23
SP2-(Cu)	15

Fig 12. Stockpile Initial Setup

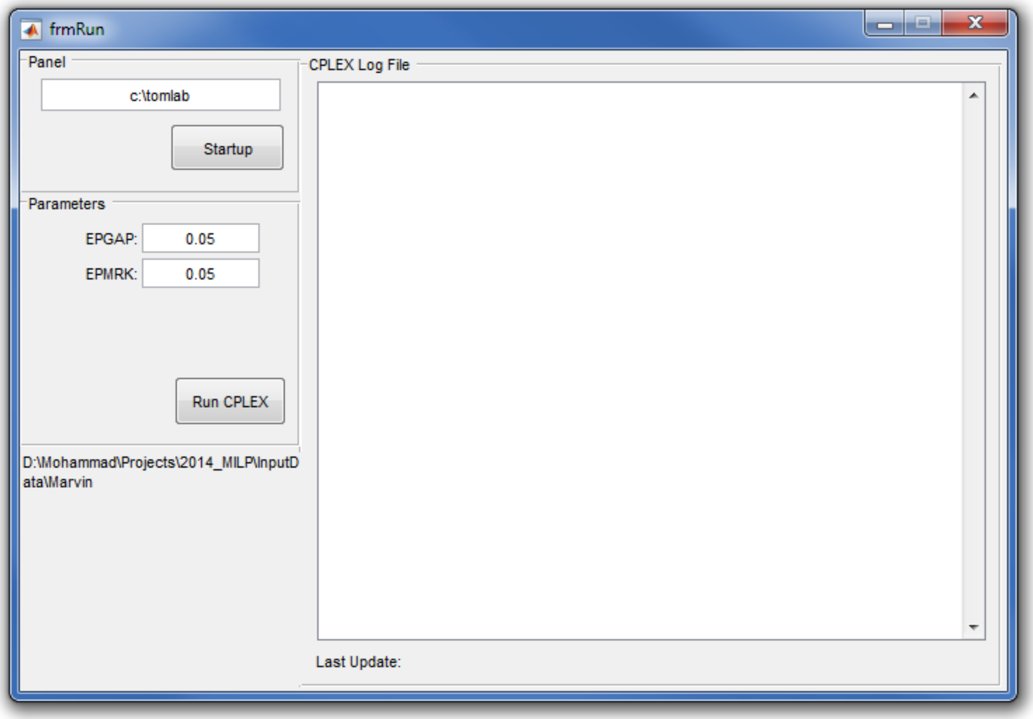
We have added two buttons to help find out the proper reclamation grades for stockpiles. The first button is the “Auto Fill” that will calculate the average grade of element by filtering clusters based on the lower and upper bounds provided. The “Histogram” button plots the histograms of the grades based on the provided lower and upper bounds.

6. Run MILP Solver

After setting up the MILP parameters, we run the solver to the determined gaps. We developed a GUI for starting up Tomlab, setting the gaps and calling the solver. The GUI is presented in Fig 13.

7. Interpret Results

The next step, after solving the MILP, is to interpret the results (Fig 14). For this step, we have developed a GUI that provides various options for the user to interpret, plot and export the results in different formats. There are four options for the different resolutions that were used in creating the MILP matrices as well as two options for updating cluster schedules based on panel schedules. Next, we have the plot functions available for initial evaluation of the output. We usually use the saved variables to create the plots with better looks in Excel. The first four plots are stripping ratio, destination (production), stockpiling (re-handling) and stockpile inventory. For the rest of the plots, the user has to choose an element from the drop-down list before calling the plot function to plot the head grade, cut-off grade (lowest grade of each element sent to process in each period), real stockpile grade and head grade with stockpile.



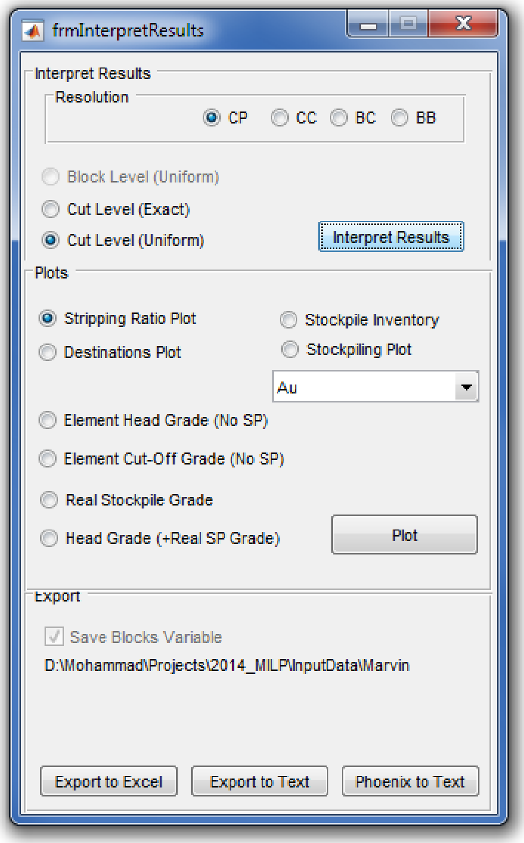
c:\tomlab

EPGAP: 0.05

EPMRK: 0.05

D:\Mohammad\Projects\2014_MILP\InputData\Marvin

Fig 13. Run MILP



● CP ○ CC ○ BC ○ BB

○ Block Level (Uniform)

○ Cut Level (Exact)

● Cut Level (Uniform)

● Stripping Ratio Plot ○ Stockpile Inventory

○ Destinations Plot ○ Stockpiling Plot

Au

○ Element Head Grade (No SP)

○ Element Cut-Off Grade (No SP)

○ Real Stockpile Grade

○ Head Grade (+Real SP Grade)

☑ Save Blocks Variable

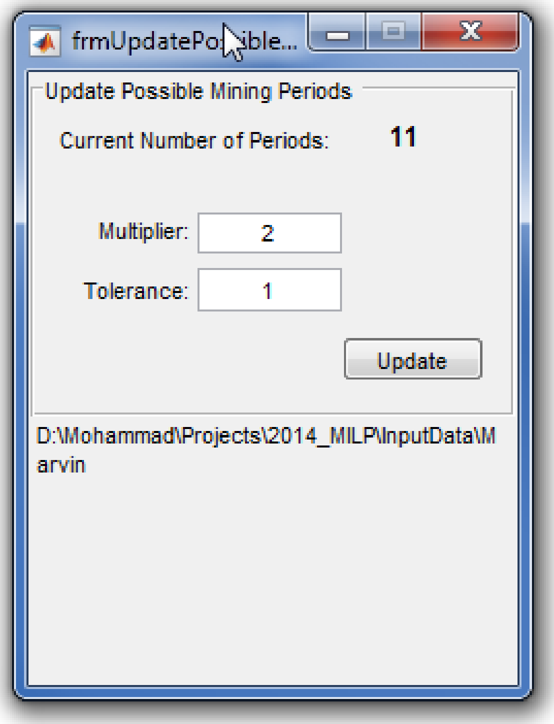
D:\Mohammad\Projects\2014_MILP\InputData\Marvin

Fig 14. Interpret Result GUI

The export to Excel button exports the matrices into an Excel file where the Export to Text feature creates a text file from all the blocks with their assigned extraction period and destination.

8. Update Possible Periods

As mentioned we sometimes solve the MILP model and use the solution to limit the decision variables and resolve the model. For example, if the mine life is 15 years, we can initially solve the model for 5 periods of three years. Then, by using a multiplier of 3 and a tolerance of 1, a panel that is initially scheduled to be extracted in period 2 will have extraction variables for years 3 to 7. We call these the possible periods of that panel and save it as an attribute for the each panel. We have developed a simple GUI to update the possible periods as shown in Fig 15. After updating the possible periods, the user has to go back to the MILP setup, set the number of periods to 15 and check the "Use Possible Periods" checkbox to use this information to solve the model in shorter time.



Current Number of Periods: 11

Multiplier: 2

Tolerance: 1

D:\Mohammad\Projects\2014_MILP\InputData\M
arvin

Fig 15. Update Possible Period