

Multiple Tutorial

Overview7
What is Index?
Fundamental of Multiple
Index, Indices, Partition, and Region
Function of Index
Initialization: Block Mesh
 Relationship between node, element, and index14

Concept of Projection			
	Pro	ojection	20
W	ind	OWS	
	Ma	ain Window	24
	Co	mputation Window	25
		View menu	
		Misc menu	32
		Selections	34

□ Start Using BlockM

Features Overview	
Create	
Blank/Del	52
Move Pts	57
Rotate Pts	66

□ Start Using BlockM

Project	76
Distribute	101
Parameter	114
Read/Write	117



BlockM: Multiple

□ Overview

- Although there are lots of mathematics which go into a mesh generator, meshing in general is an art form. This is especially true for blockM. With blockM, a user starts out with a block of elements. Then he would delete, move, and project nodes and elements in the block to form a desired shape. This is analogous to a sculptor starting out with a block of clay, then needling, cutting, and shaping it into a beautiful work of art.
- BlockM is a powerful mesh generator for hexagonal elements. The basic principals are derived from LS-INGRID. Like LS-INGRID, this is still a purely "manual" mesh generating method. However, the underlying core and user's interface are not same. The user's interface is designed to provide a highly user friendly environment, at the same time allowing the user to generate complex hex meshes efficiently.

For users of LS-INGRID and even TRUEGRID, BlockM has virtually no learning curve. For some novice users, however, blockM might be a challenge. The challenge arises from 2 relatively abstract principles known as "index spacing" and "projection". With some practices, any user can become highly proficient at generating mesh with the BlockM.

• Once a user mastered BlockM, there is virtually nothing that he cannot not mesh.

BlockM: Multiple

- □ What is Index?
 - Index or indices are divisions or cuts in the "**block**" in the I, J, and K directions. These are synonymous with the X, Y, and Z directions respectively in the Euclidean or Cartesian coordinates. In blockM, the I, J, and K will always refer to directions in index space and they are not be confused with the Euclidean space.

- □ Index, Partition, and Region
 - Example (1.0): 1D block in Index Space



- From the example above, the edges are considered index points. It also shows that the minimum number of indices for any block in each of the index direction are 2.
- The space between the index points is called partition or region.
- Partition refers to a single unit of space between 2 indices.
- Region refers to one or more partitions.

- □ Index, Partition, and Region (continued)
 - Example (2.0): 2D block in Index Space



Note the order of I and J indices. The first indices always correspond to the lowest positions of X and Y in the Euclidean space. Likewise, the second indices correspond to the higher positions in the Euclidean space.

- □ Index, Partition, and Region (continued)
 - Example (3.0): 3D block in Index Space



• Note: there is only 1 partition in all 3 index directions.

□ Index, Partition, and Region (continued)

- The intersection of 2 orthogonal indices form edges or lines.
- The intersection of 3 orthogonal indices form a point or vertex.
- A face is comprised 4 connecting edges.



Function of Index

- To achieve a desired shape, the user will have to either move or project nodes and elements to surfaces. In BlockM and LS-INGRID alike, the users do not have direct control of the nodes and elements in the block. The user will have to select the appropriate points, edges, and indices to move or project. This concept will become clearer with examples and further discussions later on.
- The bottom line:
 - Indices, points, edges, faces, and regions are used for controlling the shape of the mesh.

□ Block Mesh

- Relationship between node, element, and index
 - Nodes and elements are the basis of any finite element mesh. Now that the concept of index is touched upon, the user would want to know how this is related to nodes and elements.
 - \Box Example (4.0): Let's look at the 2D block example again.



Node #	X-coord.	Y-coord.
1	0	0
2	1	0
3	1	1
4	0	1

- For a single shell or a 2D element (e1), it is comprised of nodes 1, 2, 3, and 4, represented by n1, n2, n3 and n4.
- To generate this element, note where the indices are: I-1 at x=0, I-2 at x=1, J-1 at y=0, and J-2 at y=1. This is just an example, the initial position of the indices could be anywhere. They could be the at the same position; let's say I-1 and I-2 could both be at x=0 or J-1 and J-2 at y=1,2,3... etc. However, it is good practice to have the initial index position to be as close the desire shape as practically possible.
- As seen above, the indices form regions, which in turn contains the element or group of elements.

- Relationship between node, element, and index (continued)
 - Example (4.1): Let's expand the previous example and add one more element in the I-direction.



Node #	X-coord.	Y-coord.
1	0	0
2	1	0
3	1	1
4	0	1
5	2	0
6	2	1

- Here the 2nd I-index is moved to x=2. This is for visual demonstration purpose. To show that as more elements are added to a partition, they are automatically divided evenly. The 2nd I-index does not necessarily have to be at x=2, but could be at x=1 instead. In which case, n2 and n3 would be at x=.5.
- In this case, x-length of e1 is equaled to e2, which is equaled to 1.
- If we have 4 elements instead of 2, x-length of e1 is equaled to those of e2, e3, and e4, which all equaled to .5.
- It is important to note that the partitions bounded by the I and J indices contain 2 elements

Relationship between node, element, and index (continued)

Example (4.2): Let's continue with previous example and add 1 more partition in the I-direction.



Node #	X-coord.	Y-coord.
1	0	0
2	1	0
3	1	1
4	0	1
5	2	0
6	2	1

- When adding another partition, notice the indices lie right at the boundaries of the elements. Also, notice there are 3 indices in the I-direction and only 2 indices in the J-direction. The order of the indices in the I-direction is for left to right and the bottom to top in the J-direction.
- In this example, the 2nd I-index is set to x=1. Thus, it is no different from the previous example in regards to node and elements. It could well be set anywhere between 0 and 2.
- Because there are 2 elements in I-direction and 1 in the J-direction, it becomes apparent that the number of partition cannot be greater than the number of element for any direction. Also, the number indices is equaled to the number of partition plus 1.

- Relationship between node, element, and index (continued)
 - Example (4.3): Let's continue with previous example. This time we set the 2^{nd} I-index to x=.5



Node #	X-coord.	Y-coord.
1	0	0
2	.5	0
3	.5	1
4	0	1
5	2	0
6	2	1

- This example demonstrates how indices are used to control the size and position of nodes and elements.
- Because the 2^{nd} I-index is set at x=.5, element # 1 is 1/3 the size of element # 2.
- Notice nodes 2 and 3, they follow where ever the 2nd I-index goes. Likewise, nodes 5 and 6 will follow the 3rd I-index.

Relationship between node, element, and index (continued)





- As stated before, the number of partition cannot be more than the number of element in that direction. Since the J-direction has only 1 element, we need to add another element in order to insert a partition.
- Here, we set the 2nd J-index at y=.5, and the 3rd J-index takes the place of the 2nd J-index in the previous example.

Relationship between node, element, and index (continued)
 Example (5.0): Let's look at a 3D example.



- Going from 2-D to 3-D is quite trivial, all we have to do is add an element and a partition in the K-direction.
- Again, it is important to note the order of the indices for each principal direction. The I-direction start from left to right. This means the I-1 index is at a lower x-position than I-2 index. The J-direction goes from bottom to top because the J-1 index is at a lower y-position than the J-2 index. Similarly, because the K-1 index is at a lower z-position than the K-2 index, the direction for the K is from the page and going into the page.

BlockM: Concept of Projection

□ Projection

- The concept of projection in blockM could be view similarly to a video projector projecting images onto a screen or wall.
- The basic idea here is to place the selected nodes onto a surface or set of surfaces much like how light rays are projected onto a screen. Unlike light, the nodes will not bounce off the surface.
- Another way to view projection is surfaces form the mold of a desired shape and the block is soft clay that sets to the mold.
- Simple Example (6.0):
 - □ We want to place a node onto the blue surface by projection



BlockM: Concept of Projection

□ **Projection** (Continued)

- By default, the projected node takes the shortest path to the surface. However, there is an option for the user to specify a vector for the projection.
- Simple Example (7.0):
 - □ Projection with a vector (red)



BlockM: Concept of Projection

□ **Projection** (Continued)

- Simple Example (7.1):
 - □ Same example as before, but without vector
 - As we could see, the node is moved to the closest position on the surface.
- Details of how the projection algorithm will not be discussed here. However, more examples of projection will be provided to further demonstrate how it works.



- In BlockM, there are 2 "major" features we need to be familiar with before any meshing could begin.
- □ Main or Physical Window
 - The Main windows is what we are most familiar with, it's displays all the nodes, elements, curves, and surfaces.

□ Main Window

- Everything is displayed in Euclidean space.
- This serves to display detailed information about the mesh.
 - □ Nodes
 - Elements
 - □ Surfaces
 - □ Curves
 - □ Other parts
- It also allow for user to
 - □ Zoom
 - □ Rotate
 - □ Pan
 - □ Selection



□ Computation Window

- This window pops up as soon as the "create" button is clicked in the main window.
- The main purpose of the computation window is to provide user-friendly environment for selecting nodes and elements.
- As mentioned earlier, to select nodes and elements, the user needs to do this via the indices. Hence, the Computation window displays everything in index space.
- All partitions are displayed as perfect squares or cubes.
- Translation, rotation, and projection of nodes and elements do not affect the block in this window.
- Only deletions of partitions or regions will be affected.



□ Computation Window (continued)

- Let's look at the different aspects within the computation window.
 - View menu contains different display modes for the block.
 - Perspective
 - Hide*
 - AutoCen
 - Trimetric View
 - Triad*



*Default display setting

□ Computation Window: View menu (continued)

- Perspective
 - □ When activated, this provides the block with a perception of dept.



Example of **Perspective** display **on**.



Example of **Perspective** display **off**.

□ Computation Window: View menu (continued)

- Hide
 - □ When activated, this shows only the outer edges of the block. This is set on by default.



Example of **Hide** display **on**.



Example of **Hide** display **off**.

□ Computation Window: View menu (continued)

- AutoCen
 - This works much like the Acen feature in the Main window; it automatically centers the block within the window. This is does not "lock" the block in view. Hence, the user will need to activated every time it is needed.



Example of AutoCen display on.



Example of AutoCen display off.

□ Computation Window: View menu (continued)

- Trimetric View
 - □ This set block in the computation window to trimetric view, such as shown below. Much like the AutoCen feature, this does not hold the view.



Example of **Trimetric View** display **on**.

□ Computation Window: View menu (continued)

- Triad
 - □ This is a toggle to turn on and off the display of the triad. It is turned on by default.



Example of **Triad** display **on**.



Example of AutoCen display off.

□ Computation Window

- Misc menu contains additional user's preference settings for the computation window
 - □ Sync with MainWin*
 - This toggle setting maintains the same orientation for block in the main and computation windows.
 - For example, if the block is rotated by 30 degree about the J-axis, the block in the main window will rotate by the same amount about the Y-axis. Vise versa, if the block in the main window is rotated by 30 degree, the block in the computation window will rotate by the same amount.
 - The Sync is only applicable for rotation. It does not apply to pan and zoom.
 - □ 1.5x Win
 - Increases the computation window by 1.5 times the default size.
 - \square 2.0x Win
 - Increases the computation window by 2 times the default size.



*Default display setting

□ Computation Window: selection

- The most important part of the computation window is the selection of points, edges, faces, and regions.
- There are 2 ways to do selection.
 - 1. Using the mouse to pick a point on the block or drag over the region of interest.
 - 2. Using the mouse to select the "index slider bars" at the top, right, and both of the computation window.
- It is important to notice there are points on the slider bars and each point has a number associated with it. These points are index number for each direction.
- To the left, the block has 3 partitions and 4 indices for each direction. Hence, we see 4 points for each slider bar.
- To clear selection: as long as the Computation window is active, press the "d" key. This might be change to "ESC" key.



Slider bar for I-index

□ Computation Window: selections (continued)

Point selection

- $\Box \quad Example (8.0)$
 - Lets select a **point** at
 - I-index: 3
 - □ J-index: 4
 - □ K-index: 4
 - We could simply click on the point on the block or click at 3 locations on the slider bars.
 - Notice how the selected point is highlighted in **red**.
 - Also, the corresponding indices also turn red on the slider bars.



Selected point

□ Computation Window: selection (continued)

Edge selection

- $\Box \quad Example (8.1)$
 - Lets select an **edge** at
 - I-index: 1 3
 - J-index: 4
 - □ K-index: 4
 - We simply drag-select along the edge.
 - Or we could click and slide along the I-index slider bar from 1 to 3 and click at point 4 on both the J and K index slider bar.
 - Notice the selected edge is highlighted in green.
 - The corresponding partitions turn red in the I index slider bar.

Selected edge



□ Computation Window: selection (continued)

Face selection

 $\Box \quad Example (8.2)$

- Lets select a face at
 - □ I-index: 1 4
 - □ J-index: 2 4
 - K-index: 4
- Again, we could drag-select over the area of interest.
- Or we could click and slide along the I-index slider bar from 1 to 4, slide along the J bar from 2 to 4, and click on point 4 on the K bar.
- Notice the selected face is highlighted in **yellow**.



Selected face
BlockM: Windows

□ Computation Window: selection (continued)

Region selection

- $\Box \quad Example (8.3)$
 - Lets select an **region** at
 - I-index: 2 4
 - J-index: 24
 - □ K-index: 3 4
 - Again, we could drag-select over the area of interest.
 - Or we could click and slide along the I-index slider bar from 2 to 4, slide along the J bar from 2 to 4, and slide along the K bar from 3 to 4.

Also, notice the selected region is highlighted in **green**.



Selected region

Features Overview

- To the right is the BlockM interface and it is the first thing we see when we click on the BlockM button. The Create radio button is selected by default.
- Current tools and features
 - □ Create
 - □ Blank/Del
 - $\square \quad Move Pts.$
 - □ Rotate Pts.
 - □ Parameter
 - □ Rd/Write
 - □ Distribute
 - □ Project

1	2	3	4	5	6	7	D	
- Blo	ckm I	nterf	ace -					
•	Crea	te		C F	arar	nete	r -	
0	Blan	k/De	1	C F	td/W	rite		
C Move Pts. C Distribute								
C	Rota	te P	ts.	O F	roje	ct		
TVD	e: []	1ulti	ole F	Block	<u>د</u>		-	
). 				-			
	uex	LISC						
-								
JIn	Idex	List:					_	
	1.5100							
KI	Idex	List	8					
I								
XP	ositio	on Li	st:					
YP	ositio	on Li	st:					
ZP	ositio	on Li	st:					
					i.			
			Cre	ate				
Par	t ID:	1				PLis	t	
R	eject	1	Acc	ept	1	Don	e:	

Create

- □ In the earlier sections, we discussed the concept of index and partition. We also discussed how they are related to nodes and elements. It is here that we apply these concepts to generate the block. BlockM could create both SOLID and SHELL elements as separate parts or mixed together in one part.
- As we see to the right, there are 6 fields. We need to fill in the fields with appropriate values.
- □ The top 3 fields:
 - I Index List
 - J Index List
 - K Index List
 - are where we define the number of elements and partitions in the I, J, and K directions. Again, these correspond to X, Y, and Z directions in the Euclidean space, respectively.
- **The bottom 3 fields:**
 - X Position List
 - Y Position List
 - Z Position List
 - are where we specify the positions of the indices define in the top 3 fields.

1 2	3 4	5	6	7	D
- Blockm In	terface e	C P	aran	nete	r
C Blank	/Del	CR	d/W	rite	
C Rotat	e Pts.	C P	roje	oute ct	
Type: M	ultiple	Block	5		•
I Index L	ist:				
J Index L	.ist:				
K Index I	list:				
X Positio	n List:				
Y Positio	n List:				
Z Positio	n List:				
	Cr	eate			
Part ID:	1			PLis	t
	1				

- Create (continued)
 - □ Create button
 - Once all the field are fill in appropriately, click on this button to bring up the Computation window.
 - □ Part ID field
 - This sets the part number for the block.
 - □ **PList** button
 - We could enter a number in the Part ID field or click on this button and a select a part from a popup list.
 - □ Reject button
 - Once a block is create and for whatever reason we want to start over, just click on this button.
 - □ Accept button
 - Similarly to the Reject button above, we click this button if everything is acceptable.
 - □ Done
 - To exit out of BlockM.

			-	-		_	_
1	2	3	4	5	6	7	D
- Blo	ockm	Interf	ace -		, 31	2 D	-
	Crea	ite		C P	aran	nete	r
0	Blan	k/De	el	CR	d/W	rite	
0	Mov	e Pt	5.	00	istri	bute	
C	Rota	ite P	ts.	() P	roje	ct	
Тур	e: [1	1ulti	ple E	lock	s		•
II	Idex	List:					
							_
11	vehr	List					
	IUCA	LISU					_
-		12216					
KI	ndex	List	8				_
XP	ositi	on Li	st:				_
YP	ositi	on Li	st:				
ZP	ositi	on Li	st:				
					40		
			Cre	ate			
Pai	t ID:	1				PLis	t
R	ejecl		Acc	ept	1	Done	
	_						_

Create (continued) 2 3 1 Blockm Interface Syntax for the fields • Create We could use spaces or commas between values. Blank/Del Move Pts. Examples (9.0): **SOLID** element Rotate Pts. C Project Let's create a very simple 3D block, the same one Type: Multiple Blocks given in earlier section. I Index List: 12 To the right is how we enter the indices and their J Index List: positions. 12 Note, the first index in any direction must begin K Index List: with 1 or higher, but less than the 2 index. 12 X Position List: 01 Y Position List: 2nd K-01 index 2nd J-index 1st K-Z Position List: index 01 K Part ID: 11 1st J-index Reject 1st I-index 2nd I-index

5

6

C Parameter

C Distribute

Rd/Write

+

PList

Done

Accept

4

- Create (continued)
 - \Box Examples (9.0)
 - Below is what the computation window show when we click on the create button for the specified block.





Block in Index space

- Create (continued)
 - \Box Examples (9.0)
 - The same block is shown in the main window in Euclidean space. This block has only 1 element.





Block in Euclidean space

- Create (continued)
 - $\Box \quad Examples (9.0)$
 - Let's add more elements to the previous example.
 - Notice the block in the computation window. It looks the same as the previous example. However, the block in the Main window appears slightly different. It has more elements than the previous example.





Computation window



44

- Create (continued)
 - $\Box \quad Examples (9.0)$
 - Notice the block in the main window has 4 elements in the X and Y, and 2 in the Z direction.
 - Let's see how this works. In the fields to the right, we see the only differences between this set of inputs and that of the previous examples are the top 3 fields.
 - □ I-index: 1 5 -----> 4 elements
 - □ **J-index**: 1 5 ----> 4 elements
 - **K-index:1** 3 ----> 2 elements
 - There is a simple relationship between the number of elements and indices for any partition.
 - N element = higher index # lower index #
 - Higher index # = lower index # + N element
 - It is not necessary that the 1st index has to be 1. It just has to be less than the 2nd and 3rd index.
 - There are countless number of ways to specify the same block. We could initialize the same block in another way.
 - □ I-index: 4 8
 - □ J-index: 10 14
 - **K-index:** 5 7

1 2 3	3 4	5	6	7	D
- Blockm Inte	erface -		e R	2 () 	
• Create	2	C P	aran	nete	2
C Blank/	Del	C R	d/W	rite	
C Move	Pts.	CD	istri	bute	
C Rotate	e Pts.	C P	roje	ct	
					_
Type: Mu	ltiple E	Block	5		•
I Index Lis	st:				
15					_
J Index Li	st:				
15	5.72				_
 K Teday Li	100				
K Index Li	SC:				
13					
X Position	List:				
01					
Y Position	List:				
01					
7 Position	List:				
01					_
1					
	Cre	ate			
Part ID:	1			PLis	t
Reject	Acc	ept		Done	91

- Create (continued)
 - \square Examples (9.1)
 - Let's add 1 partition to each of the index direction for the block in example 8.0. We set the 2nd index right in the middle. Each partition has 2 elements, except for the ones in the K-direction.







Computation window

Main window

- Create (continued)
 - $\Box \quad Examples (9.1)$
 - Let's see how the relationship stated above works.
 - **I-index:** 1 + (2 el.) = 3 + (2 el.) = 5
 - **J-index:** 1 + (2 el.) = 3 + (2 el.) = 5
 - K-index: 1 + (1 el.) = 2 + (1 el.) = 3
 - The relationship works out quite nicely.
 - Notice the bottom 3 fields:
 - **X-pos** : 0 .5 1
 - □ **Y-pos**: 0.51
 - **Z**-pos : 0 .5 1
 - Because we have 3 indices in each of the index fields, we need to have 3 values for each of the position field as well.

1 1				-		
1 2	3	4	5	6	7	D
- Blockm Ir	nterf	ace -				
• Creat	te		C P	arar	nete	•
© Blank	c/De	el	CR	d/W	rite	
C Move	e Pte	5.	CD	istri	bute	
C Rota	te P	ts.	C P	roje	ct	
Type: M	lulti	ple B	lock	s		•
I Index I	ist:					
135						
J Index	List:					
135						
K Index	List					
123						
X Positio	n Li	st:				
0.51						_
Y Positio	n Li	st:				
0.51						
Z Positio	n Li	st:				
0.51				_		_
				a l		
		Cre	ate			
Part ID:	1				PLis	t
Reject	1	Acc	ept		Done	

- Create (continued)
 - $\Box \quad \text{Examples (9.1)}$
 - The table below illustrates the order on the indices and their corresponding spatial initial positions.

	1 st Index	2 nd Index	3 rd Index
I-index	1	3	5
X-position	0	.5	1
J-index	1	3	5
Y-position	0	.5	1
K-index	1	2	3
Z-position	0	.5	1

1	2	3	4	5	6	7	D
 Blockm Interface 							
• Create C Parameter							
C	Blan	k/De	el 🛛	C R	d/W	rite	
C	Mov	e Pt	5.	C D	istri	bute	
C	Rota	nte P	ts.	C P	roje	ct	
					-		_
Тур	e: ī	1ulti	ple B	lock	5		•
I In	dex	List:					
13	5						
JI	ndex	List:					
13	5						
KI	ndex	List					
12	23						
XP	ositi	on Li	st:				
0.	51						
YP	ositi	on Li	st:				
0.	51						
ZP	ositi	on Li	st:				
0.	51						
					r.		
			Cre	ate			
Par	t ID:	1				PLis	t
R	ejecl	:	Acc	ept		Done	•
	_		_	_		_	_



Conceptual Block

Create (continued) 3 1 2 4 5 Blockm Interface • Create C Parameter Examples (9.2): SHELL element Blank/Del C Rd/Write Move Pts. Notice the -1 in the K Index List field. C Rotate Pts. C Project Below is the shell block shown in Computation Type: Multiple Blocks I Index List: window. 123 J Index List: 123 BlockM Computation Window K Index List: View Misc. -1 к 1 X Position List: 2 0.52 Y Position List: 0.51 Z Position List: 0 Part ID: 1 Reject Accept З

6

Distribute

*

PList

Done

Block in Index space

- Create (continued)
 - □ Examples (9.2): SHELL element
 - Below is the shell block shown in Main window.







Blank/Del

- This feature allows for both visually disabling (blanking) and for deleting elements and nodes.
- To blank or delete, we need to first select a region. Blank and delete are not applicable to point, edge, and face.
- □ There are 2 ways to select a region. We could either specify the region with indices in the index list fields, as seen on the right, or select from the computation window.
- □ If selection is made from computation window, the index list fields will be automatically filled in accordingly.
- Once a region is selected, click on the blank or delete button to the right.



Blank/Del (continued)

- □ (Un)Blank button
 - Blanking is for disabling the display of nodes and elements. Click this button again will unblank and enable the display of nodes and elements again.
 - This is useful for visually checking interior elements.
 - Example (10.0)
 - Let's use the same block as that in example 8. We want to blank out the region:
 - □ I-index: 2 3
 - □ J-index: 2 3
 - **K-index:** 2 3

	2	3	4	5	6] 7	1
- Blo	ckm	Interf	ace		0 3		100
C .	Crea	te		O F	araı	nete	r
•	Blan	k/De		CF	v\bs	/rite	
CI	Mov	e Pts	5.	0)istri	bute	
с.,	Rota	ite P	ts.	C.F	roje	ct	
I In	dex	List:					
23							_
J In	dex	List:	6				
23	8						
K In	dex	List					
23							
23							
23		((Un)	Blan	k		
23		Re	(Un) vers	Blan se Bla	k ank		
23		(Re	(Un) ver: Disp	Blani se Bla lay A	k ank II		
23		(Re	(Un) ver Disp De	IBlani se Bla lay A lete	k ank II		

- Blank/Del (continued)
 - □ (Un)Blank button
 - Example (10.0)
 - Notice the blank applies to both the main and computation window.



Main window



Computation window



- Blank/Del (continued)
 - □ Reverse Blank button
 - This blanks the region that is currently display and unblank the visually disabled region.
 - **Example** (10.1)
 - Note the fields to the right do not update to reflect the corresponding displayed region.





Computation window



Main window

- Blank/Del (continued)
 - **Display** All button
 - This enables the display of all blank regions.
 - **Delete** button
 - This feature is similar to the blank. Visually, the blank and delete features appear to be the same. However, the delete actually remove nodes and elements from the block once the "accept" button is clicked.
 - Unlike the blank feature, unblanking the missing region will not bring it back.
 - □ Undelete button
 - This feature brings back the last deleted region.

1	2	3	4	5	6	7
- Blo	ckm I	nterf	ace -		0 3	
0	Crea	te		O F	arar	nete
•	Blan	k/De	el	C F	kd/W	rite
CI	Mov	e Pts	5.	្ព)istri	bute
01	Rota	te P	ts.	C F	roje	ct
I In	dex	List:	ł			
23						
J In	dex	List:				
23	3					
K In	dex	List:				
23						
						1
			(Un)	Blan	¢	
		(Re	(Un) vers	Blani ie Bla	k Ink	
		(Re	(Un) vers Displ	Blani ie Bla lay A	k Ink	
		(Re ^r	(Un) vers Displ	Blani e Bla lay A lete	k Ink II	

Move Pts.

- □ This feature lets the user move points, edges, faces, and regions.
- □ There are a number of options to choose from depending on the intend of the translations.
- □ To move:
 - 1. Select the point, edge, face or region using the computation window as explained previously.
 - 2. Select one of the constrained options to the right.
 - 3. Enter the distance to translate. This does not apply to any screen constrained translation.
 - 4. Click "Trans-" or "Trans+" button. A less precise method is to use the mouse to drag in the Main window.



- Move Pts. (continued)
 - **Constrain To:**
 - NULL*
 - This is a default setting. It means no constraint. We could move the nodes and elements in any direction. Hence, there are 3 fields for the 3 global directions (X, Y, and Z).
 - X-Axis
 - Nodes and elements could only be translated along the X-axis.
 - Y-Axis
 - Likewise, nodes and elements could only be translated along the Y-axis.
 - Z-Axis
 - Similarly, nodes and elements could only be translated along the Z-axis.
 - Screen X Axis
 - Nodes and element could only be translated along the screen's X-axis or horizontal direction. This not the same as the global X-axis.

1	2	3	4	5	6	7	C
- Blo	ckm I	nterf	ace -	~ •			
0	Crea Blan	te k/n	a	CP	arar d/w	nete Irito	r. S
	Mov	P Pt		C n	listri	hute	
C	Rota	te P	ts.	CP	roie	ct	
100		252	075 	0.503	1.140		_
Con	trair	1 To:	1				
	6	e Ni	ULL				
	0	X	Axis				
	0	Y	Axis				
	0	Z	Axis				
	9	S	cree	n X A	xis		
	-	5	cree	n Y A	xis		
	9) X	Y Pla	ne			
	2	X	Z Pla	ne			
	2	Y	Z Pla	ne			
	5	S	cree	n Plai	ne		
Dist	ance	e:					
т	rans	- 1	Tra	ns+			
					- 12		
	R	edo	1		Und	0	1
1			_	-			1

- Move Pts. (continued)
 - **Constrain To:**
 - Screen Y Axis
 - Nodes and element could only be translated along the screen's Y-axis or vertical direction. This not the same as the global Y-axis.
 - XY Plane
 - Nodes and element could only be translated on the global XY plane.
 - XZ Plane
 - Nodes and element could only be translated on the global XZ plane.
 - YZ Plane
 - Nodes and elements could only be translated on the global YZ plane.
 - Screen Plane
 - Nodes and elements could only be translated along both the screen's X and Y axis.

- Bk		3 nterf	4 ace -	5	0	<u>, ,</u>	-
C	Crea	te		C P	arar	nete	r
C	Blan	k/De	el	CR	d/w	/rite	
(F	Mov	e Pt	5.	CD	istri	bute	
0	Rota	te P	ts.	C P	roje	ct	
Cor	ntrair	1 To:	t.		_		-
	6	F NI	JLL				
	0	×	Axis				
	0	Y	Axis				
	(z	Axis				
	(5	ree	XA	xis		
	¢	5	ree	NY A	xis		
	(٦ XI	r Pla	ne			
	0	٦ xz	Z Pla	ne			
	0	Ϋ́	Z Pla	ne			
	(5	ree	n Plai	ne		
Dis	tance	e:					
Γ					ГГ		
1	Frans	-	Tra	ns+			
	R	edo	1		Und	10	1
1			_	-			

Move Pts. (continued)	1 2 3 4 5 6 7 D Blockm Interface
 Examples (11.0): Constrain to X-Axis Using the same block as in example 10.0, let's move a point along the X-axis a distance of 1.0. 	C Create C Parameter C Blank/Del C Rd/Write Move Pts. C Distribute C Rotate Pts. C Project
Point to move	Contrain To: C NULL C X Axis
File Msc. Toggle Badground Applications Settings Help Image: Toggle Badground Applications Help Image: Toggle Bad	Y Axis Z Axis Screen X Axis Screen Y Axis XY Plane XZ Plane YZ Plane Screen Plane Distance: 1 Trans- Redo Undo

Before



After

Move Pts. (continued)

□ Examples (11.1): Constrain to Screen's X-Axis

Using the same block as in example 10.0, let's move an edge along the screen's X-axis by some distance. As mentioned before, we need to drag in the Main window. The block is rotated 90 degree so that the global X is pointing up.



2 3 4 5 6 7 1 Blockm Interface C Create Parameter C Rd/Write Blank/Del Move Pts. C Distribute Rotate Pts. C Project Contrain To: C NULL C X Axis C Y Axis C Z Axis Screen X Axis Screen Y Axis C XY Plane C XZ Plane C YZ Plane Screen Plane Distance: -1.#IND00



- Move Pts. (continued)
 - □ Examples (11.1): Constrain to Screen's X-Axis
 - Notice that the screen X-axis is not the same as that on the global X-axis. In this example, they are almost orthogonal to each other.



- Blo	ckm Ir	nterf	ace -	-	-	
C	Creat	te		CP	araı	nete
0	Blank	c/De	el	C R	d/W	/rite
•	Move	Pt	s.	CD	istri	ibute
C	Rota	te P	ts.	C P	roje	ct
Con	train	To				
	0	N	JLL			
	C	x	Axis			
	0	Y	Axis			
	0	Z	Axis			
	6	5	cree	N X A	kis	
	C	5	cree	NY A	kis	
	0	X	Y Pla	ne		
	0	X	Z Pla	ne		
	0	Y.	Z Pla	ne		
	ç	5	cree	n Plar	1e	
Dis	tance	:				
0.0	1153	8	0.44	9360	0	.0852
T	rans	- 1	Tra	105+	1	
	Tdus		100	0157		
	Re	do	1		Und	lo



Before



After

Rotate Pts.

- □ This feature lets the user rotate points, edges, faces, and regions about a point.
- Similar to the Move Pts. feature, there are a number of options to choose from depending on the intend of the rotations.

Blockm Interface C Create C Blank/Del Move Pts. Rotate Pts. C Distribute Rotate Pts. C Project Contrain To: X Axis X Axis X Axis Screen X Axis Screen Y Axis	er	mete		1	1.1		1
Create C Paramete Blank/Del C Rd/Write Move Pts. C Distribute Rotate Pts. C Project Contrain To: X Axis X Axis X Axis X Axis Screen X Axis Screen Y Axis	er	mete		-	rface	ockm inte	Ble
Contrain To:	er o ol		Dava	C	- acc	Croate	C
Contrain To: NULL X Axis X Axis		Muito		~	100	Plank/	~
Rotate Pts. C Distribution Rotate Pts. C Project Contrain To:		ibuto	Dicto	~	te te	Movo D	~
Contrain To: NULL X Axis Y Axis Z Axis Screen X Axis Screen Y Axis		act	Dist	-	Dte	Detate	a
Contrain To: NULL X Axis X Axis X Axis X Axis Screen X Axis Screen Y Axis		cu	rioje	2	F LS.	NULALC	1
 NULL X Axis Y Axis Z Axis Screen X Axis Screen Y Axis 					0:	ntrain T	Cor
C X Axis C Y Axis C Z Axis C Screen X Axis C Screen Y Axis					ULL	•	
C Y Axis C Z Axis C Screen X Axis C Screen Y Axis				5	(Axis	<u> </u>	
C Z Axis C Screen X Axis C Screen Y Axis				5	Y Axis	C .	
Screen X Axis				5	Z Axis	C :	
Screen Y Axis			X Axis	n X	Scree	0	
0			Y Axis	n Y	Scree	0	
Screen Z Axis			Z Axis	n Z	Scree	0.1	
Rotate Axis Base Point:			Point:	e P	s Bas	tate Axi	Rot
Position						osition	Po
0.0 0.0 0.0		.0	0.	_	0.0	D	0.0
Rotation Axis Direction:			ction:	irea	xis D	tation A	Rot
Direction						rection	Di
			-		0.0	0	1.0
10.0 0.0			Ju		10.0		11.0
and the second sec	_	00	0.0000	0	gle:	tate An	Rot
Rotate Angle: 0.000000				-	Ro	otate+	n
Rotate Angle: 0.000000 Rotate+ Rotate-			te-	tat	1 100	order of	R
Rotate Angle: 0.000000 Rotate+ Rotate-	4		te-	tat	1	otote i	R

- Rotate Pts. (continued)
 - **To rotate:**
 - 1. Select the point, edge, face or region using the computation window as explained previously.
 - 2. Select one of the constrained options to the right.
 - 3. Enter the "Rotation Axis Base Point". This point could be entered manually by its x, y, and z coordinates in the fields right below the "Position" button. Or the point could be picked from the Main window, by clicking on the Position button.
 - 4. Enter the "Rotation Axis Direction". This is the vector that points along the axis of rotation. Or the "Direction" button could be used to pick the vector graphically. If a constraint is chosen, there is no need to enter the direction.
 - 5. Enter the angle of rotation in degree in the "Rotate Angle" field.
 - 6. Click on either "Rotate+" or "Rotate-" to rotate counter-clockwise or clockwise direction.

- Blockm	Interface -	3	-	-	-
C Crea	ite	C P	aran	iete	r
C Blan	k/Del	CR	d/W	rite	
C Mov	e Pts.	CD	istrit	oute	į.
• Rota	ate Pts.	C P	rojec	:t	
Contrai	n To:				
(• NULL				
(X Axis				
(Y Axis				
(Z Axis				
	-				
	Scree	n X A	xis		
(Scree	n X A n Y A	xis xis		
0	Scree Scree Scree	n X A n Y A n Z A	xis xis xis		
(((Rotate	Screet Screet Screet Axis Bas	n X A n Y A n Z A e Poir	xis xis xis nt:		
(((Rotate Positio	Screen Screen Screen Axis Base	n X A n Y A n Z A e Poir	xis xis xis nt:		
Rotate Positio	Screen Screen Screen Axis Base	n X A n Y A n Z A e Poir	xis xis xis nt:		
Rotate Positio	Screet Screet Axis Base 01 0.0	n X A n Y A n Z A e Poir	xis xis xis nt: 0.0		
Rotate Positio	Screet Screet Axis Base 0 0.0 n Axis Di 0.0	n X A n Y A n Z A e Poir	xis xis xis nt: 0.0 on:	<u>-</u>	
Rotate Positic 0.0 Rotatio Directic	Screet Screet Axis Base 0 0.0 n Axis Di 0.0	n X A n Y A n Z A e Poir	xis xis nt: 0.0 on:	0	
Rotate Positic 0.0 Rotatio Directio	Screen Screen Axis Base 0.0 n Axis Di 0.0	n X A n Y A n Z A e Poir	xis xis nt: 0.0 on:	D	
Rotate Positio 0.0 Rotatio Directii 1.0 Rotate	Screet Screet Axis Base 0 0.0 0.0 0.0 0.0 0.0 0.0	n X A n Y A n Z A e Poir rectio	xis xis nt: 0.0 on: 0.0	0	
Rotate Positio 0.0 Rotatio Directio 1.0 Rotate Rotate	Screet Screet Axis Base 0.0 Axis Di 0.0 Angle: e+ Roi	n X A n Y A n Z A e Poir rection 0.0 tate-	xis xis nt: 0.0 0000	0	
Rotate Positio 0.0 Rotatio Directio 1.0 Rotate	Screet Screet Axis Base 0 0.0 n Axis Di 0.0 Angle: e+ Rol	n X A n Y A n Z A e Poir rection 0.0 tate-	xis xis nt: 0.0 0000	D 0	

- Rotate Pts. (continued)
 - **Constrain** To:
 - NULL*
 - Nodes and elements are free to rotate about any axis.
 - X Axis
 - Nodes and elements could only be rotated about the global X-axis.
 - Y Axis
 - Nodes and elements could only be rotated about the global Y-axis.
 - $\blacksquare ZAxis$
 - Nodes and elements could only be rotated about the global Z-axis.

- Blockn C Cre C Bla C Mo	3 4 In Interface - eate nk/Del ve Pts.	5 () P () R () D	6 7 arameter d/Write istribute
Contra	in To: NULL X Axis Y Axis Z Axis Screen Screen Screen Axis Base	n X Au n Y Au n Z Au e Poir	kis kis kis
Positi	ion		
Positi	ion 0.0		0.0
Rotati Direct	on Axis Dir	rectio	0.0 on:
Rotate Positi 0.0 Rotati Direct	on Axis Dir tion	rectio	0.0 on:
Rotate Positi 0.0 Rotate Direct 1.0 Rotate Rotate	on Axis Dir on Axis Dir tion 0.0 e Angle: te+ Rot	o.ou	0.0 on: 0.0

- Rotate Pts. (continued)
 - **Constrain To:**
 - Screen X Axis
 - Nodes and elements could only be rotated about the screen's X-axis.
 - Screen Y Axis
 - Nodes and elements could only be rotated about the screen's Y-axis.
 - Screen Z Axis
 - Nodes and elements could only be rotated about the screen's Z-axis. The screen's Zaxis is always coming out of the screen or normal to the screen.

Blockm	Interface.	1.5	-	
C Crez	te	CP	aram	eter
C Blan	k/Del	CR	d/w	rite
C Mov	e Pts	CI	istrib	ute
• Rota	ate Pts.	C P	rojec	:t
Contrai	n To:			
1	• NULL			
1	X Axis			
1	Y Axis			
1	C Z Axis			
1	Scree	n X A	xis	
	Scree	n Y A	xis	
1	Scree	n Y A n Z A	xis xis	
(Rotate	Scree Scree	n Y A n Z A e Poi	xis xis	
Rotate Positic	Scree Scree Axis Bas	n Y A n Z A e Poi	xis xis nt:	
Rotate Positic	Scree Scree Axis Bas	n Y A n Z A e Poi	xis xis nt: 0.0	
Rotate Positic	Scree Scree Axis Bas	n Y A n Z A e Poi	xis xis nt: 0.0	
Rotate Positic 0.0 Rotatio	Scree Scree Axis Bas 0 0 0.0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	n Y A n Z A e Poi recti	xis xis nt: 0.0 on:	
Rotate Positic 0.0 Rotatio Directi	Scree Scree Axis Bas 01 0.0 n Axis Di 01	n Y A n Z A e Poi recti	xis xis nt: 0.0 on:	
Rotate Positic 0.0 Rotatio Directi 1.0	Scree Scree Axis Bas 0.0 n Axis Di 0.0	n Y A n Z A e Poir	xis xis 0.0 on: 0.0)
Rotate Positic 0.0 Rotatio Directi 1.0 Rotate	Scree Scree Axis Bas 0 0.0 n Axis Di 0.0 Angle:	n Y A n Z A e Poir rection 0.0	xis xis 0.0 000 00000)
Rotate Positic 0.0 Rotatio Directi 1.0 Rotate Rotate	Scree Scree Axis Bas n 0.0 n Axis Di 0.0 Angle: e+ Ro	n Y A n Z A e Poir rection 0.0 tate-	xis xis nt: 0.0 on: 0.0 00000	0
Rotate Positic 0.0 Rotatio Directi 1.0 Rotate Rotate	Scree Scree Axis Bas 00 0.0 n Axis Di 0.0 Angle: e+ Ro	n Y A n Z A e Poir rection 0.0 tate-	xis xis nt: 0.0 on: 0.0000) 0



Before



Original orientation

Normal to rotation orientation






Original orientation

Normal to rotation orientation





Shaded view

Wire frame view



- **Rotate Pts.** (continued)
 - \Box Example (12.2): Constrain to Z Axis

Face rotated



10 March 10 March 10	4	3	4	5	6	7	D
- Blo	ckm I	nterf	ace -				
C	Crea	te		CP	arar	nete	r
CI	Blan	k/De	el 🛛	CF	d/W	rite	
CI	Mov	e Pts	5.	CD	istri	bute	
•	Rota	te P	ts.	C P	roje	ct	
Con	trair	1 To:	E.				_
	(N	JLL				
	0	X.	Axis				
	0	Y.	Axis				
	0	Z.	Axis				
	0	50	ree	n X A	xis		
	1	50	ree	NY A	xis		
	6	F 50	ree	n Z A	xis		
Rot	(ate /	i So Axis	ree Bas	n Z A e Poi	xis nt:		
Rot Po	ate /	Sc Axis n	ree Bas	n Z A e Poi	xis nt:		
Rot Po 0.0	ate /	F So Axis n	Bas	n Z A e Poi	xis nt: 1.0)	
Rot Po 0.0	ate /	Sc Axis n	Basi	n Z A e Poi	xis nt: 1.0)	
Rot Po 0.0 Rot	ate /	Axis n 0 1 Axis	Basi D.0 is Di	n Z A e Poi recti	xis nt: 1.0 on:)	
Rot Po 0.0 Rot Dir	ate /	Axis n n n n Axis n	Bas 0.0 is Di	n Z A e Poi recti	xis nt: 1.0 on:)	
Rot Po 0.0 Rot Dir	ate / sitio ation ectio	Axis n n n Axis n (n Axis	D.0 is Di	n Z A e Poi recti	xis nt: 1.0 on:	,165	
Rot 0.0 Rot Dir Rot	ate / sitio ation ectio 981	Axis n n n Axis n (n Angl	D.0 is Di e:	n Z A e Poi recti 06 30	xis nt: 1.0 on:	,165	
Rot Dir Rot Rot Rot	ate / sitio ation ectio 981 ate / otate	Axis n n n Axis n (n Axis n (n Angl + +	D.0 is Di -0.1 e: Roi	n Z A e Poi recti 06 30 tate-	xis nt: 0n:	.165	
Rot Po 0.0 Rot Dir -0.3 Rot Rot	ate / sitio ation ectic 981 ate /	Axis Axis n (Axis n (Angle +	cree Bas D.0 is Di -0.1 e: Ro	n Z A e Poi recti 06 30 tate-	xis nt: 0n: 0	,165	
Rot Po 0.0 Rot Dir -0.3 Rot	ate / sitio ation ectic 981 ate / otate	Sc Axis n (1 Axis n (1 Axis n (1 Axis n (1 Axis n (1 Axis n) (1 Axis n) () () () () () () () () ()	cree Bas D.0 is Di -0.1 e: Ro	n Z A e Poi recti 06 30 tate-	xis nt: 1.0 on: 0) .165	

After

Project

- Early on, the concept of projection was introduced. It is here that this concept is implemented.
- □ There are 7 different types of "target". More targets will be added in the future.
 - Point
 - Line
 - Surface
 - Shell Elem
 - Plane
 - Cylinder
 - Sphere
- □ Individual components of the projection could be specified with X, Y, and Z radio buttons at the bottom of the menu.
- Point, Line, Surface, and Shell Elem all requires external source. These must be generated outside of BlockM.
- □ Plane, Cylinder, and Sphere are generated within BlockM.

1 2 Blocker	3	4	5	6	7
C Crea	ate		P	ara	mete
C Blar	ik/De		C F	ld/V	/rite ibut/
C Rot	ate Pt	s.	۰Ē	roje	ct
@ Po	oint	(^ P	lane	1
C Li	ne	(n c	ylin	der
C 50	irface	(° 5	phe	re
C Sh	ell Ele	em			
Snap of	f Picki	ng Po	oint +		
Snap of	f Picki Pick Near End Mid	ng Poin Poin Point Point	bint t Poin t	ıt	
Snap of	f Picki Pick Near End Mid	ng Poin Poin Point Point	pint t Poin t	it V	z
Snap of	f Picki Pick Near End Mid	ng Poin rest I Point Point	pint t Poin t	it I	z
Snap ol c c c c c c c c c c c c c c c c c c c	f Picki Pick Near End Mid	ng Poin rest I Point Point 7 Y ocks	t Point t	it V	z

- Project (continued)
 - □ Point
 - In order to project to a point, we need to generate a point. To do this, we need an "external" surface or curve such as igest surface and curve.
 - 2. To select a target point, simply click on a desired location on a surface or curve in the Main window.
 - It is good practice to project a point to a point, rather than an edge or face to a point.
 - 3. Select a "source" point from the Computation window by one the processes described above.
 - 4. Click on the Project button.

 Create Blank/Del Move Pts. Rotate Pts. 	 Paramete Rd/Write Distribute Project
Point	C Plane
CLine	C Cylinder
C Shall Floor	spnere
Snap of Picking	Point
Snap of Picking Pick Po Neares End Po Mid Po	Point bint st Point int int
Snap of Picking Pick Po Neares End Po Mid Po	Point bint st Point int int
Snap of Picking Pick Po Neares End Po Mid Po X V V	Point bint st Point int int Y Z
Snap of Picking Pick Po Neares End Po Mid Po X V 1	Point bint st Point int int Y IZ ks

Project (continued)

□ Point

- There are 4 snapping options for point selection. This simplifies the point selection process, and it is more applicable to picking a point on a curve or line.
 - Pick Point This is the default setting. The point selected is right where the cursor is positioned.
 - Nearest Point This option allows the selection of point nearest to the cursor.
 - End Point This options allows picking of the nearest end point on a curve or line.
 - Mid Point Similar to the previous options, this allows for the selection of a mid-point of a line.

C Create	C Paramete
C Move Pts	. C Distribute
C Rotate Pt	s. 📀 Project
Point	C Plane
C Line	C Cylinder
C Surface	C Sphere
C Shell Ele	em
Snap of Picki	ng Point
Snap of Picki Pick Near End Mid	ng Point Point rest Point Point Point
Snap of Picki	ng Point Point Point Point Point
Snap of Picki Pick Near End Mid	ng Point Point Point Point Ž Y 🔽 Z
Snap of Picki C Pick C Near C End C Mid	ng Point Point Point Point Z Y IZ Z ocks

- Project (continued)
 - □ Point
 - Example (13.0) : Projecting point to a point on a surface



1 2 3 4 5 6 7 D

C Parameter

C Rd/Write

C Distribute

Blockm Interface

C Blank/Del

C Move Pts.

Point selection panel – contains options for picking and unpicking points .

Before

- Project (continued)
 - □ Point
 - Example (13.0) : Projecting point to a point on a surface





After

- Project (continued)
 - □ Line
 - 1. Much like that for the Point, we need an external curve or line such as igest curve. A curve could be generated in the **Curves** interface on pages 2 and 7.
 - 2. To select a target curve, simply click on a desired curve in the **Main** window.
 - It is good practice to project a point or an edge to a line or curve, rather than a face or a region to a line.
 - 3. Select a source edge from the Computation window by one the processes described above.
 - 4. Click on the Project button.

1	2	3	4	1	5	6	7
- BIC	Crea	Inter	tace -	C	P	aran	nete
C	Blan	k/D	el	C	R	d/W	rite
C	Mov	e Pt	s.	C	D	istri	bute
C	Rota	ate I	Pts.	œ	P	roje	ct
C	° Po	int		C	P	lane	
6	Lin	ie		C	C	ylind	ler
C	° Su	rfac	e	C	s	pher	e
A	x		ν	2		L	z
Г	Bla	ink I	Block	s			

- **Project** (continued)
 - Line
 - Example (13.1): Projecting an edge to a curve



1 2 3 4 5

Blockm Interface

Blank/Del

C Create

C

6 7

C Parameter

C Rd/Write

D

Line selection panel – contains options for picking and unpicking lines and curves.

Before

- Project (continued)
 - □ Line
 - Example (13.1) : Projecting an edge to a curve





After

Project (continued)

□ Surface

- 1. Like the previous 2 projection options, we need an external surface such as igest surface. A surface could be generated from curves in the **Surface** interface on page 7.
- 2. To select a target surface or surfaces, simply click on a desired ones in the **Main** window.
- It is good practice to project a point, edge, or face to a surface. It is not a good idea to project a region to a surface.
- 3. Select a "source" point, edge or face from the **Computation** window by one the processes described above.
- 4. Click on the **Project** button.



- Project (continued)
 - □ Surface
 - Example (13.2) : Projecting 2 faces to a cylindrical surface.
 - □ Use Closest 3D proj dir.



Surface selection panel – contains options for picking and unpicking surface .

Before

C Create C Parameter Blank/Del C Rd/Write C Move Pts. C Distribute Rotate Pts. . Project C Point C Plane C Line C Cylinder G Surface C Sphere C Shell Elem Closest 3D proj dir. C Vector projection V X V Y VZ Blank Blocks Project Redo

3 4

5 6 7

2

Blockm Interface

1

- Project (continued)
 - □ Surface
 - Example (13.2) : Projecting faces to a surface





After

Project (continued)

□ Shell Elem

- 1. This option requires some sort of shell elements to be generated first.
- 2. To select target shell elements, simply click on a desired ones in the **Main** window. Shell elements could be selected with either propagation, area, or part option.
- It is good practice to project a point, edge, or face to shell elements. It is not a good idea to project a region to shell elements, though.
- 3. Select a source point, edge or face from the **Computation** window by one the processes described above.
- 4. Click on the Project button.

1 Pla	2	3	4	5		6	J.	/
C 010	Crea	ite	ace -	C	P	ara	me	ete
C	Blan	k/De	el	C	R	d/V	Vri	te
C	Mov	e Pt	5.	C	D	istr	ibu	ıte
C	Rota	ate P	ts.	e	P	roje	ect	
C	Po	int		C	P	an	e	
0	Lin	e		C	C	ylin	de	r
C	Su	rface	e	C	5	phe	re	
6	Sh	ell El	em					
	Clo	sest	t 3D	pro	j c	lir.		
6 C D	Cla Ve	sest	t 3D proj ⊽ v	pro	j (io	lir. n		
ح ح ح	Cla Ve	osest ctor	t3D proj ⊽γ	pro	j c io	lir. n	ĨZ	<u>.</u>
	' Cla ' Ve X Bla	osest ctor nnk B	: 3D proj ▼ Y lock	pro ect	j (io	lir. n	ź	5

- **Project** (continued)
 - Surface
 - Example (13.3) : Projecting a face to shell elements.
 - Use Closest 3D proj dir



3 4

Rotate Pts.
 Project

V Y

5 6

C Parameter

C Rd/Write

C Distribute

C Plane

C Cylinder

VZ

Redo

C Sphere

7

2

C Create

Blockm Interface

Blank/Del

C Move Pts.

1

Element selection panel - contains options for picking and unpicking shell elements.

Before

- Project (continued)
 - □ Surface
 - Example (13.3) : Projecting a face to shell elements.
 - Use Closest 3D proj dir





After

- Project (continued)
 - □ Plane
 - Unlike the previous 4 options, the Plane is generated in BlockM. By default, the plane is at x=0, y=0, z=0 and the normal vector is along the z-axis.
 - 1. Position the plane by either entering the XYZ coordinates in the fields to the right or click on the Position button to select with a mouse.
 - 2. Specify the orientation of the plane by either entering the vector components in the direction fields, or click on the Direction button to visually select.
 - Plane is an infinite plane, although visually the plane is "clipped".
 - 3. Select a source point, edge or face from the **Computation** window by one the processes described above.
 - 4. Click on the Project button.

and the second se	2	3	4	5	6	7	D
- Blo	ckm I	nterfa	ice -	1025-00			
0	Crea	te		O P	aran	nete	r)
0	Blan	k/De		C R	d/W	rite	
~	Piov	to Di		GR	Istri	ouce et	
	RULA	LE FI	э.	1	roje		
C	Poi	nt		(P	lane		
C	Lin	e		C C	ylind	er	
C	Su	face		C S	pher	e	
C	She	ell Ele	m				
Po 0 Dir	sitio			_	0		
0	ccu)		1		
0							
•	Clo Vec	sest ctor p	3D p proje	oroj (ectio	dir. n		
	Clo Vec	sest ctor (3D j proje 7 Y	oroj (ectio	dir. on	z	
	Clo Vec X	sest ctor p R nk Bl	3D (proje 7 Y ocks	oroj (ectio	dir. n	z	



- Project (continued)
 - □ Plane
 - Example (13.4) : Projecting a face to a plane.
 - Use Closest 3D proj dir



Blockm Int	3 4	5	6	7
C Creat	e	C.	Para	mete
C Blank	/Del	С	Rd/	Write
Move	Pts.	C	Dist	ribute
C Rotat	e Pts.	•	Proj	ect
C Poin	t	(F	Plan	e
C Line		\mathbf{C}	Cylir	nder
C Surf	ace	C	Sphe	ere
	1			
Position 0 Directior	.75		0	(
Position 0 Directior 0	.75 1		0 	0
Position 0 Direction 0 Close C Vect	.75 1 1 est 3D cor pro	proj jecti	j dir.	0
Position 0 Direction 0 Close C Vect	.75 1 1 est 3D for pro	proj jecti	j dir. j dir. jon	o ž z
Position Direction O Close C Vect X Blan	I I I I I I I I I I I I I I I I I I I	proj jecti (j o j dir. j on	o Z z

- Project (continued)
 - □ Cylinder
 - Similar to the Plane option above, the cylinder is generated in BlockM. By default, the cylinder is at x=0, y=0, z=0, the length is along the z-axis, and radius=5.
 - 1. Specify the cylinder's radius in the Radius field.
 - 2. Position the cylinder by either entering the XYZ coordinates in the fields to the right or click on the Position button to select with a mouse.
 - 3. Specify the orientation of the cylinder's axial direction by either entering the vector components in the direction fields, or click on the Direction button to visually select.
 - A cylinder has infinite length. It is visually clipped.
 - 4. Select a source point, edge or face from the **Computation** window by one the processes described above.
 - 5. Click on the Project button.

1 2	3 4	4 5	6	7	D
- Blockm	Interface				
C Blar	ate 1k/Del	C	aran Rd/W	rite	
C Mov	e Pts.	CI	Distri	bute	
C Rot	ate Pts.	•	Proje	ct	
C Po	oint	C F	lane		
C Li	ie		ylind	ler	
C Su	Inface	C S	pher	e	
C Sh	ell Elen	1			
Radius	5				
Positi	on				
0	0		0		
Directi	ion				
0	0		1		_
← Cle ← Ve	osest 31 ector pr	D proj ojectio	dir. on		
∀ x	•	Y	•	z	
∏ Bla	ank Bloo	ks			
Proje	ct 🗐	Jndo	f	Redo	



Cylinder is positioned at x=-1.5, y=0, z=0, radius=1.0, and axial direction is 45-degree on the YZ plane.

position, and orientation

- Project (continued)
 - □ Cylinder
 - Example (13.5) : Projecting a face to a cylindrical surface.
 - □ Use Closest 3D proj dir



1 2	3 4	1 5	6	7	1
- Blockm I	nterface		1.000		
C Crea	te	CI	Parar	nete	•
C Blan	k/Del	CI	Rd/W	/rite	
C Mov	e Pts.	CI	Distri	bute	
C Rota	te Pts.	• 1	Proje	ct	
C Poi	int	CI	Plane		
CLin	e		ylind	ler	
C Su	face	C s	Sphe	re	
C Sh	ell Elen	n.			
Radius	1				-
Positio	n				
-1.5	0		0		
Directio	on				
0	1		1	3	
€ Clo © Vec	sest 31 ctor pro) proj ojectio	dir. on		
∀ x	•	Y	~	z	
∏ Bla	nk Bloc	:ks			
Projec	t l I	Indo	1	Redo	

- Project (continued)
 - □ Sphere
 - Similar to the Plane and Cylinder options above, the sphere is also generated in BlockM. By default, the sphere is positioned at x=0, y=0, z=0, and a radius=5.
 - 1. Specify the cylinder's radius in the Radius field.
 - 2. Position the cylinder by either entering the XYZ coordinates in the fields to the right or click on the Position button to select with a mouse.
 - 4. Select a source point, edge or face from the **Computation** window by one the processes described above.
 - 5. Click on the Project button.

Blockm I	3 4 nterface	5	6	1.
C Crea	te	C F	aran	nete
C Blan	k/Del	C F	kd/W	rite
C Mov	e Pts.	C)istri	bute
C Rota	te Pts.	• F	roje	ct
C Poi	int	C P	lane	
CLin	e	00	ylind	ler
C Su	rface	(• s	pher	e
C Sh	ell Elem	R		
Radius	5			
Positio	n			
0	0		0	
	sest 30) proj	dir.	
⊙ Veo	ctor pro	jectio	n	
€ Clo C Veo	ctor pro	ojectio Y	on V	z
© Clo © Vec ⊽ X □ Bla	ctor pro	pjectio Y ks	n V	z





Project (continued)

- Sphere
 - Example (13.7) : Projecting a to a spherical surface.





1

Blockm Interface

2 3 4 5 6 7 D



- Project (continued)
 - □ Sphere
 - Example (13.8) : Projecting a to a spherical surface.

1.

2.







1 2 3 4 5 6 7 D

After

Distribute

- This feature allows for added mesh refinement by redistributing nodes with minimum user's inputs.
 Currently, the distribution feature is limited to edges only. However, nodal distribution for faces and regions will be added shortly.
- □ There are 3 options are available:
 - Edge along line
 - This projects an edge to a curve and then distributes the nodes evenly along it.
 - 1. Select an edge using the Computation window
 - 2. Click on the "Along a whole line" radio button
 - 3. Select a curve from the Main window.
 - 4. To apply, click on the "Edge along line" button.

1	2	3 4	5	6] 7	
- Blo	ockm li	nterface				
C	Crea	te	CI	Para	mete	er
C	Blanl	k/Del	0	Rd/W	/rite	
C	Move	e Pts.	•	Distri	ibute	•
C	Rota	te Pts.	CI	Proje	ct	
Г	Alon	g a wh	ole lin	e		
÷			Edge	along) line	£
-	Vect	ex inte	erpola	tion		
Г	Vect Edge	ex Inter	erpola polatio	on Lir	iear	8
70	Vect Edge	ex Inter Inter Ratio:	polatio	n Dn Lir	iear	8
Zor	Vect Edge ning F Both	ex Inter Inter Inter Inter Inter	polatio	Lin	iear	2
Zor	Vect Edge ning F Both	ex Inter Inter Latio: Ends	polatio	Lin	ning	



Before

- Distribute (continued)
 - □ Edge along line
 - Example (14.0) :
 - Note how the nodes are evenly distributed a long the entire curve.



C Create C Parameter Blank/Del C Rd/Write Move Pts. Distribute Rotate Pts. C Project Along a whole line Edge along line Vectex Interpolation Edge Interpolation Linear Zoning Ratio: 1.0 Both Ends Zoning Undo

3

Blockm Interface

4 5 6 7

1 2

Distribute (continued)

Linear

- This feature distributes nodes evenly along a selected edge. This option is quite straight forward. Vectex Interpolation is selected by default. Edge Interpolation option could be selected as well.
- 1. Select the edge or edges to linear distribute.
- 2. Click the "Linear" button to apply.



- Distribute (continued)
 - □ Linear
 - Example (14.1) : Distribute nodes on 2 edges.





Before

- Distribute (continued)
 - □ Linear
 - Example (14.1) : Distribute nodes on 2 edges.





After

Distribute (continued)

Zoning

- This option distributes nodes along an edge such that the adjacent segment length divided by the current segment length equals the Zoning Ratio. By default, the Zoning Ration is equaled to 1.0.
- 1. Select the edge or edges to be distributed.
- 2. Set the Zoning Ratio. This number cannot be negative.
- 3. If both ends of the edge are to be the same than click the Both Ends radio button.
- 4. To apply, click on the Zoning button.

1 2 3 4 Blockm Interface	5 6 7
C Create	C Parameter
C Blank/Del	C Rd/Write
C Move Pts.	Oistribute
C Rotate Pts.	C Project
🗆 Along a who	ole line
	Edge along line
✓ Vectex Inter ☐ Edge Interp	erpolation polation
	Linear
Zoning Ratio:	1.0
1	Zoning
Redo	Undo

- Distribute (continued)
 - □ Zoning
 - Example (14.2) : Distribute nodes on 3 edges with Zoning ration = 1.5.





Before
- Distribute (continued)
 - □ Zoning
 - Example (14.2) :
 - Zoning ration = 1.5, note how the element sizes are changing.





After

- Distribute (continued)
 - □ Zoning
 - Example (14.3) : Distribute nodes on 3 edges with Zoning ration = .5.





Before

- **Distribute** (continued)
 - Zoning
 - Example (14.3) :
 - Zoning ration = .5, note how the element sizes are reversed.

2 3

Blockm Interface

Blank/Del

C Move Pts.

C Create

1

5

4

6 7 D

C Parameter

C Rd/Write

Distribute
 Distribute

Edge along line

.5

Linear

Zonina

Undo





- Distribute (continued)
 - □ Zoning
 - Example (14.4) : Distribute nodes on 3 edges with Zoning ration = .5 and Both Ends.



Blank/Del C Rd/Write Move Pts. Oistribute Rotate Pts. C Project Along a whole line Edge along line Vectex Interpolation Edge Interpolation Linear .5 Zoning Ratio: Both Ends Zoning Redo Undo

2 3

Blockm Interface

C Create

1

4 5

6 7 D

C Parameter

Before

- Distribute (continued)
 - □ Zoning
 - Example (14.4) :
 - Zoning ration = .5 and Both Ends, note how the smaller elements are concentrated at the center.







- Parameter
 - □ This feature allows for parameterized mesh generations. What this means is that a mesh could be changed and updated with the change of a parameter's value.
 - □ Functions and expressions could be added here and applied directly to a point, edge, face, and or region.
 - **To implement Parameter:**
 - 1. Enter parameters or expression in the 1st field
 - 2. Click Create button or press the Enter key. This puts the parameters or expressions in the list below.
 - 3. To Delete: select an expression or parameter from the list and click the Delete button.
 - 4. To apply to the block directly: enter the parameters in the corresponding X, Y, and Z Coordinate fields.
 - 5. Select a point, edge, face and region using the Computation window.
 - 6. Click the Apply button.

1	2	3	4	5	6	7
- Blo	ockm li	nterf	ace -	-		
0	Crea	te		O P	arar	neter
2	Blank	k/De	1	CR	td/W	rite
0	Move	e Pts		0.0	listri	Dute
0	Rota	te P	ts.	(C) P	roje	ct
Par	amet	er o	r Exp	oress	sion:	2
e.g.	A=1	50 o	r R=	T1*9	sin(T	2)
	Cre	ate		1	Del	
						ere
		utt			Dei	ete
00	rdina	ite X	•		Dei	ete
200	rdina	ite X	•		Der	
C00	rdina rdina	ite X ite Y	:		Der	ete
	rdina rdina	ite X ite Y ite Z	:			ete
Coo Coo	rdina rdina	te X te Y	:			ete

- Parameter (continued)
 - □ Example (15.0): Let's create a parametric block.
 - Set the following parameters:
 - □ width= 2.0
 - $\square \quad hite = 3.0$
 - $\square \qquad len = width + hite$
 - \Box xseg = 8
 - \Box yseg = 6
 - \Box zseg = 4

	-		-		- 1		
1	2	3	4	5	6	7	D
- 51	ockm I	nterfac	ж —	-	00000		
0	Crea	te		Pa	ram	ietei	r
0	Blan	k/Del	2	Rd	/Wi	rite	
0	Mov	e Pts.	2	DIS	strit	ute	
0	Rota	te Pts	. 0	Pro	ojec	t	
Par	amet	ter or	Expre	essio	on:		
e.g.	A=1	50 or	R=TI	L*sii	n(T2	2)	
	Cre	ate		1	Dele	te	
wii hit ler xs ys zs	dth=: n=wic eg=8 eg=6 eg=4	2.0 0 Ith+hi	ite				
Coc	ordina	te X:					
Coc	ordina	ite Y:					
Coc	ordina	te Z:					
1							
		- 1		_			
. 1	Apply		Unde	0		Redo))

- Parameter (continued)
 - **Example** (15.0):

Below is what a parametric block looks like.



1	2	3	4	5	6	7	D
- Blo	ockm I	nterf	ace -				
•	Crea	te		CP	aran	nete	r -
C	Blan	k/De	el	CF	kd/W	rite	
C	Mov	e Pt	5.	C)istri	bute	
C	Rota	te P	ts.	O P	roje	ct	
тур	e: h	1ulti	ple B	lock	5		•
II	ndex	List:					
1 (1+xs	seg)					
JI	ndex	List					
1 (1+ys	seg)					
KI	ndex	List					
1(1+zs	eg)					
X P	ositio	on Li	st:				
0	len						
YP	ositio	on Li	st:				
0 1	vidth	8					
ZP	ositio	on Li	st:				
0	hite			_	_		_
-							
			Cre	ate			
Pai	rt ID:	1				PLis	t
	aject	11	Acc	ont	1	Done	

Rd/Write

- Within BlockM there is also a way to save the current work to a file. This is convenient to save just features and information related to BlockM. The file written out is in ASCII format. The user could modify the file with a text editor such as Notepad, WordPad under Windows and Vi under Linux/Unix.
- □ To save or Write:
 - 1. Click the Write button.
 - 2. Enter the file name and path in the pop-up window. By default, the file as the .cfile extension.
 - 3. Click on the Save button.

Blo	ockm	Interf	ace -	3	0	
C	Crea	ite k/De	-	(P	aran d/W	nete rite
C	Mov	e Pt	5.	CD	istri	bute
C	Rota	nte P	ts.	C P	roje	ct
	I		Em	pty		
			Em Re	pty ad		

	Rd/Write (continued)
--	----------------------

3.

4.

- □ To open an existing **BlockM** file:
 - 1. Simply click on the Read button.
 - 2. Select the file in the pop-up window.
 - Click the Open button.
 - A CFile Dialog window pop-up such as the one to the left. Click the Start button for the file to be processed.
 - 5. Click the Done button at the bottom.

- Blockm Interface C Create C Parameter C Blank/Del G Rd/Write C Move Pts. C Distribute C Rotate Pts. C Project Empty Read Write	1	2	3	4	5	6	7	C
C Create C Parameter Blank/Del C Rd/Write Move Pts. Distribute Rotate Pts. Project Empty Read Write	- BI	ockm	Interf	face -				
C Blank/Del (• Rd/Write C Move Pts. C Distribute C Rotate Pts. C Project	0	Crea	ite		C F	araı	nete	r
Move Pts. Distribute Rotate Pts. Project Empty Read Write Vrite	0	Blan	k/D	el	9	kd/W	/rite	
Empty Read Write	2	Mov	e Pt	s.	01)istri	bute	
Empty Read Write	5	ROL	ite P	τs.		roje	CT	
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Empty Read Write								
Read				Em	nhi		1	
Read		-		LIII	pry		_	
Write		r		_			-1	
Write		L		Re	ad			
Write							1	
				100	at a			

卐ī CF	ile Dialog	X
0.00	Pause in Sec	
) 14	oop Continuously	
	Open	
	Start	
	Step	
	Rewind	
	Close	
	D	1
	Done	

Rd/Write (continued)

- □ There is a special button here, Empty.
 - The purpose is to clear all the BlockM commands from the buffer.
 - Use this button with caution because all works could be loss if saves to the same file that is currently loaded.
 - To avoid possibly losing works, it is a good idea to save to a different file.

1.00	2	3	4	5	6	7
BI	ockm	Interf	ace			
C	Crea	ite		C	Para	mete
C	Blan	k/De	el	•	Rd/V	Vrite
C	Mov	e Pt	s.	0	Distr	ibute
C	Rota	ate P	ts.	C	Proje	ect
			Em	ipty		1
			En Re	npty ead		

Rd/Write (continued)

- \square Example (16.0): Saved block file from example 15.0
 - As seen below, the BlockM input file is composed of a series of commands. As a user become familiar with BlockM's commands, he could "program" a part without using the BlockM's visual interface. This could be handy because it is a lot faster to generate and modify meshes.

bmparam create "width=2.0"
bmparam create "hite=3.0"
bmparam create "len=width+hite"
bmparam create "xseg=8"
bmparam create "yseg=6"
bmparam create "zseg=4"
bmcreate multiple i 1 (1+xseg) j 1 (1+yseg) k 1
(1+zseg) x 0 len y 0 width z 0 hite

000	Create Blank/	e /Del	000	Paran Rd/Wi	ieter rite
c	Rotate	e Pts.	Ċ	Projec	t
		En	tu		ï
		Er	npty]
		En	npty ead		