PTC Supplement

Argus MeshMaker

PTC Supplement

Contents

Overview

This supplement describes the use of Argus MeshMaker Pro as a pre-processor for the PTC (Princeton Transport Code) ground water model. It is assumed that the user is familiar with both Argus MeshMaker and PTC.

Once again, the generic nature of Argus MeshMaker proves its benefits over model specific pre-processors. Using MeshMaker's powerful Export Template Scripting Language, developers and users of PTC created a MeshMaker export template which automatically creates all the necessary data (input) files for PTC.

PTC is operated by executing PTC commands stored in a .run file. These commands enable the user to define the files to be read and echoed, and to specify other model options. MeshMaker's PTC export template also automatically creates this PTC command file. Thus, to run PTC, the user need only type a shell command specifying the command file that was created by MeshMaker.

Creating the MeshMaker export template for PTC did not require any changes in PTC itself thus, any PTC version can be used in conjunction with this template. The template can be used as is, or it can be continuously changed by the user to support future features of PTC and Argus MeshMaker.

In the following pages you will find a description of the template, instructions for using MeshMaker with it, and details about new MeshMaker functions and options that will be included in the next major update and that are already used in the template for PTC.

Creating a MeshMaker Project for PTC

A MeshMaker project for PTC does not differ from any other MeshMaker project file. You create information and maps layers and link them to a mesh layer. You then import or digitize information in these layers to characterize the site. From background maps, parameters distributions and boundary conditions information, stored in the MeshMaker layers you have created, you define the domain's outline (site) in a domain layer and then mesh that domain in a mesh layer. To export the information you just need to execute the Export mesh command.

However, to make MeshMaker export the PTC command file, you have to specify for each MeshMaker parameter that is to be exported to PTC, the appropriate PTC command. For instance if a layer parameter describes the elevation of a geological formation at mesh nodes, that parameter has to be assigned the PTC command `drnelev`, and if the project describes more then one formation, the number of that formation.

You store this additional information in parameters tags. Each MeshMaker layer and parameter can be assigned as many tags as you need.

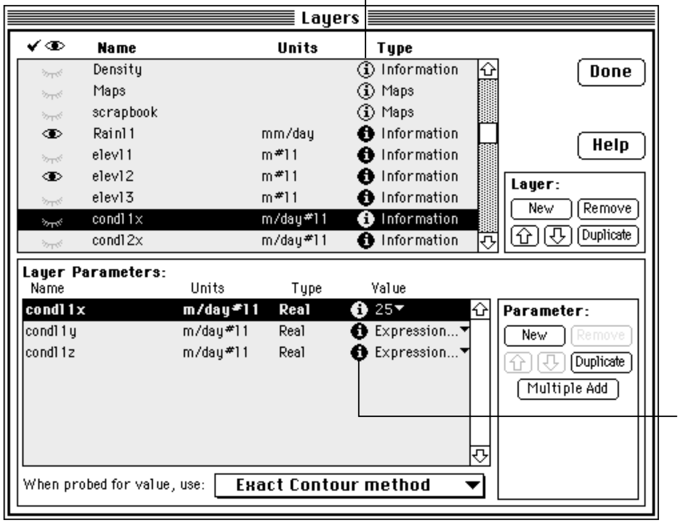
Parameters’ Tags

A parameter tag is defined by its name, the TagName and its value, the TagValue, where both are strings.

Assigning tags to parameters

Parameters tags are assigned in the Parameter Tag dialog which is invoked from the Layers dialog. A layer’s main parameter as well as its other parameters can be assigned tags. Every layer and parameter line in the Layers dialog is marked with an ⓘ icon standing for additional information. Clicking that icon brings up the Parameter Tag dialog.

Click this icon to open the Parameter Tag dialog.



Layers

✓ 👁	Name	Units	Type
	Density		ⓘ Information
	Maps		ⓘ Maps
	scrapbook		ⓘ Maps
👁	Rainl1	mm/day	ℹ Information
	elevl1	m*11	ℹ Information
👁	elevl2	m*11	ℹ Information
	elevl3	m*11	ℹ Information
	condl1x	m/day*11	ℹ Information
	condl2x	m/day*11	ℹ Information

Layer Parameters:

Name	Units	Type	Value
condl1x	m/day*11	Real	ℹ 25
condl1y	m/day*11	Real	ℹ Expression...
condl1z	m/day*11	Real	ℹ Expression...

When probed for value, use: Exact Contour method

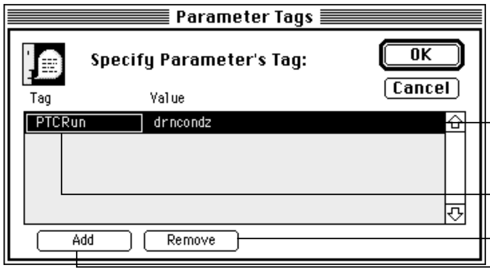
Parameter has no tags

Parameter has tags

Click this icon to open the Parameter Tag dialog and edit tags.

To assign tags to a parameter

1. Select the line of the layer its tags or its parameters’ tags you wish to edit.
2. Click the ⓘ icon. The Parameter Tags dialog box appears.



Parameter Tags

Specify Parameter's Tag:

Tag	Value
PTCRun	drncondz

Type the tag’s Value.

Type the tag’s Name.

Click to remove a tag.

Click to add a tag.

Although both the TagName and TagValue can include any character, you should avoid using special characters. The $ character must not be used in the TagName.

Using Tags for PTC

As mentioned earlier in this supplement, tags are used to create the PTC .run file which contains the PTC commands. The tags names appearing in the following examples are mere suggestions. You can change these names, but you then have to also change them in the MeshMaker export template.

Geological formations numbering

PTC requires that the different geological formations modeled are numbered from bottom to top and that various data associated with each formation are assigned the formation number they belong to. For instance, if the problem at hand is defined by 2 geological formations, each will have elevation, conductivity, boundary conditions, initial conditions and storativity information associated to mesh nodes or elements and marked by the formation number.

To number information layers for PTC

1. Open the Layers dialog.
2. Create an information type layer and name it for instance `elevl1`, standing for "Elevation of 1st layer".
   When you export the information stored in that layer, MeshMaker will create a file named after the layer name with the `.dat` extension. You can use any name you wish, however long names or names containing spaces or dots will work only on Macintosh. If you're using PC you should not use names longer than 8 characters.
   Placing the formation number at the end of the layer name will allow you to easily create new layers for additional formation using the Duplicate button, so that MeshMaker will automatically increase the formation number.
3. Click the layer's main parameter ⓘ icon to create a tag and assign it a value. The Parameter Tags dialog appears.
4. Click the Add button to create a new tag name.
5. In the Tag text edit field type the name `LayerNumber`.
6. In the tag Value field type the number of the geological formation, in this case 1.
7. Close the Parameter Tag dialog.

Assigning PTC commands to layers

PTC interprets the information stored in input files according to PTC commands. These commands describe the type of data in the file and the way it is assigned to the mesh. For instance, `drncondx` means that the `x` conductivity (`cond`) is assigned at nodes (`n`), or `drestor` means that storativity (`stor`) is assigned at elements (`e`).

MeshMaker's template for PTC assumes that each parameter is assigned a tag describing the PTC command to be executed when that file is read by PTC. For instance to assign the PTC command `drnelev` to the parameter `elevl1` that we just created:

To assign a PTC command to a parameter

1. In the Layers dialog select the `elevl1` parameter line.
2. Click the ⓘ icon to open the Parameter Tag dialog.
3. Click the Add button to create a new tag
4. In the TagName field type `PTCRun`
5. In the TagValue field type `drnelev`
6. Click the OK button to close the Parameter Tag dialog.

Duplicating layers

If your problem domain contains a number of geological formations, you will need to create additional information type layers into which you will bring information describing the spatial distribution of these formations. To save yourself the tedious task of creating all these layers manually you can duplicate layers using the Duplicate button.

When you create layers using the Duplicate button, all you have to do is make sure that the various tag values and names are correct. For instance, if you duplicate the layer `elevl1` to create the layer `elevl2`, enter the Parameter Tag dialog and change the `LayerNumber` tag value from `1` to `2`.

Boundary Condition Layers and Tags

MeshMaker allows you to easily combine all your boundary conditions and point sources for each geological formation in one information layer. You can define boundary conditions areas, using close contours, line boundary conditions such as “rivers” or “faults”, using the open contour, and point sources, such as wells using the point object. MeshMaker functions and expressions allow for exact discrimination of mesh nodes that lie above such boundary conditions, and thus allows the mesh nodes to be automatically assigned there boundary condition type and values.

To create a boundary conditions layer

1. In the Layers dialog create an Information type layer.
2. From the popup menu at the bottom of the dialog choose the Exact Contour interpretation method.
3. Name the layer `BCTypL1` for instance, were `BCTyp` stands for Boundary Condition Type, and `L1` stands for the first geological formation. Since the export template uses layer names to create file names, layer names must not exceed 8 characters on PC. If you use a Macintosh or Unix machine you can use longer and more meaningful names.
4. Set the parameter's default value to `$n/a`. You can also set it to any other value, but PTC template is set so that only nodes having boundary condition type and value different from $n/a are exported to file. This ensures that the boundary condition file contains only boundary condition nodes.
5. Click the ⓘ icon in the layer line and add a tag named `LayerNumber`. Set its value to 1 (representing the number of the geological formation)
6. Click the ⓘ icon in the parameter line and add a tag named `PTCRun`. Set its value to `drnbcflo`
7. Create another tag named `bc` and set its value to `BCValL1`. Where `BCValL1` stands for the Boundary Condition Value and is the name of the next parameter you will create for this layer.
8. Create another layer parameter and name it `BCValL1`.
9. Set this parameter default value to `$n/a`

Layers Describing Heterogenous Properties Distribution

In some instances you need to describe the spatial distribution of a heterogenous property. For instance, the conductivity of a geological formation might be different in the X, Y and Z directions. Both MeshMaker and PTC support this feature. In PTC you denote the conductivity in each of the directions using the PTC commands `drncondx`, `drncondy` and `drncondz`, where in MeshMaker you create two additional layer parameters for the layer describing the conductivity.

To create a layer describing the distribution of conductivity

1. In the Layers dialog create an information type layer.
2. From the popup menu at the bottom of the dialog choose the Exact Contour interpretation method.
3. Name the layer `CondXL1` for instance, were `CondX` stands for the X conductivity and `L1` stands for the first geological formation.
4. Set the parameter's default value to the default X conductivity.
5. Click the ⓘ icon in the layer line and add a tag named `LayerNumber`. Set its value to 1 (representing the number of the geological formation)
6. Click the ⓘ icon in the parameter line and add a tag named `PTCRun`. Set its value to `drncondx`

7. Create a second layer parameter, either by duplicating the main parameter or by creating a new one and name it `CondYL1`.
8. Click the ⓘ icon in the parameter line and change the tag value from `drncondx` to `drncondy`
9. Repeat step 7 to create a parameter `CondZL1` for the Z conductivity.
10. Repeat step 8 and change `drncondy` to `drncondz`

If the Y and Z conductivities are related to the X conductivity, you can define that relation using the expression dialog. For instance if the Y conductivity is 10% of the X conductivity, assign the `CondYL1` parameter the expression `CondYL1`*0.1.

A Tag for the Total Number of Geological Formations

PTC requires that the total number of geological formations be defined in the PTC command file. To accommodate this requirement create a tag named `NumberOfLayers` in the mesh layer you will be exporting, and assign its value the total number of geological formations being modeled. Using this technique you can insert other model or code related information required by PTC and avoid having to manually change them in the PTC export template.

Linking Layer Parameters to a Mesh Layer

After you finished creating the skeleton of the project, you need to link the data from the various information layers onto a mesh. You can link information to mesh nodes and mesh elements. When you will create a mesh, each node and element will be automatically assigned the appropriate information from the layers linked to it.

To link information layers and parameters to nodes

1. In the Layers dialog, select the mesh layer to which you want to link information.
2. In the parameter control panel click the Node Params radio button.
3. Click the Multiple Add button.
   The Add Multiple Parameters dialog appears.
4. Select all parameters you want to link to nodes.
5. Click the OK button.

MeshMaker automatically creates the required node parameters. Each node parameter also contains the tag names and values you have created earlier.

To link information layers and parameters to elements

1. In the Layers dialog, select the mesh layer to which you want to link information.
2. In the parameter control panel click the Element Params radio button.

3. Click the Multiple Add button.
   The Add Multiple Parameters dialog appears.
4. Select all parameters you want to link to elements.
5. Click the OK button.

MeshMaker automatically creates the required element parameters. Each element parameter also contains the tag names and values you have created earlier.

Defining the Domain Outline

After you finish describing the various parameters in the information layers, either using contour tools, or by importing information, you can define the domain outline from boundary conditions, geological information and other relevant information. **The PTC template assumes that the name of the domain outline layer is “Domain Outline”.** You can, however, change the template to reflect the name of the domain layer associated with the mesh layer you export, or create the boundary conditions parameter in another way so that your PTC template is not associated with specific layer names.

To force mesh nodes at “well” locations

If the domain problem contains point sources and sinks such as wells, you might want MeshMaker to create nodes above these “well” locations.

1. Bring the boundary conditions layer to front and select all the point objects.
2. From the Edit menu, choose Copy.
3. Bring the domain layer to front and paste these “wells”
4. Repeat steps 1 to 3 for each of the boundary condition layers associated with the various geological formations.
   If some “wells” already exist in the domain layer, MeshMaker will notify you and paste only those not already present.

To force elements’ sides along “rivers/faults” lines

If the domain problem contains open contours representing line entities such as rivers or faults, you might want MeshMaker to create the elements so that element sides do not cross these lines.

1. Bring the boundary conditions layer to front and select all open contours.
2. From the Edit menu, choose Copy.
3. Bring the domain layer to front and paste the open contours.
4. Repeat steps 1 to 3 for each of the boundary condition layers associated with the various geological formations.
   If some “river/faults” already exist in the domain layer, or if they cross existing contours, MeshMaker will notify you and paste only those valid.

Meshing the Domain

In the mesh layer click the magic wand above the domain outline. Examine the resulting mesh, and if you need to change it, use the density layer to refine the mesh, or refine it using manual refinement tools.

Exporting Your Data

All you have to do now to export your data is execute the Export By Template command.

To Export your data

1. Activate the mesh layer you wish to export.
2. Make sure that the export template loaded is the PTC template.
3. From the File menu choose Export By Template.
4. Set the General Export Arguments.
5. In the Save As dialog box type the name of the project.
   Do not use extensions. If you work on a PC do not use more than 8 characters.

New MeshMaker Functions and Export Macros

Two new functions and three new export macros are used in the following examples. Also, the File script command is enhanced to allow calling it from within other commands and nesting. These functions and export macros will be fully documented in the next version of Argus MeshMaker.

New Functions

The isna(expression) function

The isna(expression) function is a Logical function. It tests an expression and returns 1 (true) if the expression evaluates to $n/a and 0 (false) if it does not.

Name & Syntax	*Arguments*	*Description & Return Value*
isna*(expression)*	***expression***	Returns 1 (true) if the expression evaluates to $n/a and 0 (false) if it does not.

The NodeAboveCntr function

This is a node related function.

Name & Syntax	*Arguments*	*Description & Return Value*
NodeAboveCntr*(layer_name)*	***layer_name***	The name of the layer in which you need to find if the node lies above or inside a contour. Returns 0 (false) if not above or within any contour, 1 if above a point object. 2 if above an open contour. 3 if above a close contour. 4 if inside a close contour.

Example: The function `NodeAboveCntr(Domain Layer)` tests the node's location to find if it lies above or within a contour in the layer "Domain Layer" and returns:
0— if it does not lie above a contour, or if it is not enclosed by a close contour.
1 — if it lies above a point object (a one point contour).
2— if it lies above an open contour.
3— if it lies above a close contour.
4— if it is enclosed by a close contour.

New Export Macros

The $HasTag(TagName)$ Macro

This macro returns 1 (true) if it finds a tag having TagName, and 0 (false) if it does not. In an export loop over parameters each parameter is tested for TagName.

The $Tag(TagName)$ Macro

This macro returns the TagValue of TagName. In an export loop over parameters each parameter is tested for TagName. If the TagName is found, TagValue, is returned. If TagName is not found the macro is replaced with its string representation (unresolved) to notify you that a problem occurred.

Example: A Tag name `PTCRun` has a TagValue `drnelev` in a certain parameter. The macro `$Tag(PTCRun)$` returns `drnelev` for that parameter. If TagName `PTCRun` is not found the string `$Tag(PTCRun)$` is returned.

The $BaseNameNoExt$ Macro

Resolving $Basename$

While exporting a mesh layer using the following template, the `$BaseName$` macro is replaced by the file name supplied in the Save As... dialog.

```
Redirect output to: $Basename$.nod
    Node Command
    Node Command
End File
Redirect output to: $Basename$.elm
    Element Command
    Element Command
End File
```

Macintosh: For instance, if you supply the name MESHEXP, the two occurrences of the argument `$Basename$` will be replaced by MESHEXP, to create two files named MESHEXP.nod and MESHEXP.elm.

PC and UNIX: For instance, if you supply the name MESHEXP, the two occurrences of the argument `$Basename$` will be replaced by MESHEXP, to create two files named MESHEXP.nod.exp and MESHEXP.elm.exp. Since this is not allowed on PC and usually not needed you could use the `$BaseNameNoExt$` macro instead. In this case the two occurrences of the argument `$BasenameNoExt$` will be replaced by MESHEXP, to create two files named MESHEXP.nod and MESHEXP.elm.

Enhanced File command

Name	*Syntax*	*Arguments*
File	**Redirect output to:** **End File**	***Filename*** **or** ***$Basename$***

- This command can now be nested within any other script command, including the File command.
- This command is a block command.

MeshMaker’s Export Template for PTC

The following pages list MeshMaker’s template for PTC. The template is made up of two main parts. The first part creates the PTC run file, while the second creates the data files for PTC. Some PTC variables are set in the template. However, if you wish them to be automatically set by MeshMaker, create parameter tags, assign them values and change the PTC template to export these tags’ values. (see example of exporting the total number of geological formations).

```
# insert the name of the PTC run file
Redirect output to: $BaseNamenoext$.run
  Start a new line
    Export expression:
           "**********************************************************************\n*\n*
           data set for sample problem
           #1\n*\n**********************************************************************\n
           **********************************************************************\n*\n*
           items included in this example:\n*  - pumping and injecting wells\n*  - several
           types of boundary conditions: constant head, specified\n*    flux, leakage into
           aquifer, leakage out of aquifer, constant\n*    concentration, and specified
           mass flux of contaminant\n*  - 3 layer system with a middle clay layer (2) with
           a high\n*    permeability window\n*  - water table calculations\n*  -
           rainfall\n*\n*  solution procedure:\n*    flow is calculated first, and then
           transport is calculated using\n*    the resultant flow field\n*\n*  units:\n*
           length is in feet and time is in days, unless otherwise noted\n*\n*  note:\n*
           this problem contains all the features of sample problem #1a,\n*    but uses the
           simplest and easiest to understand data structure.\n*    generation of data and
           use of the dfile only occurs for the drnrain\n*    command.\n*    there are no
           graphics commands in this
           example.\n*\n******************************************************************
           *****\n*  begin input for
           cfile\n***********************************************************************\
           n*\n*"
  End line
  Start a new line
    Export expression: "open                     *open a file with unit number of 3\n 3
           'ptc.out'\ncecho                    *echo commands to unit
           3\n3\nefile                    *echo input errors to unit
           3\n3\nopen                     *open an output file for flow mass balance\n11
           'ptc.mbf'\nopen                     *open an output file for transport mass
           balance\n12 'ptc.mbm'\nmwrit                    *direct flow and trans. mass bal.
           to 11 & 12 resp.\n11 12\nrwrit                    *output flow and mass solutions
           to unit 3 & 3 resp.\n3 3\nowrit                    *output echo requests to unit
           3\n3"
  End line
  #
  # in the line below enter the project name
  #
  Start a new line
  Export expression: "rdtitle                 *input title\n 'MeshMaker Problem'"
End line
Start a new line
  Export expression: "rddims                  *specify number of nodes, elements, layers"
End line
Start a new line
  Export expression: NumNodes(); [I5]
  Export expression: NumElements(); [I5]
  Export expression: "$Tag(NumberOfLayers)$"
End line
Start a new line
  Export expression: "rddifusn                *molecular diffusion"
End line
Start a new line
  # insert here the molecular diffusion
  Export expression: 0.0001 [E20.6]
End line
Start a new line
  Export expression: "rdweight                *upstream weight (default = 1.)"
End line
Start a new line
  # insert here the upstream weight
  Export expression: 1. [F8.2]
End line
#
# Start of commands for incidences
#
Start a new line
  Export expression: "indexon\ndfile\n9 '*' 1\nopen\n9 ";
  Export expression: "'$Basenamenoext$.inc'"
End line
Start a new line
  Export expression: "drdcoors\nrdincid\nclose\n9\nindexoff"
End line
#
# end of commands for incidences
#
# PTC commands for exporting parameters
#
Start a new line
  Export expression: "indexon"
End line
Loop for: Element Parameters
  Start a new line
    Export expression: "dfile\n 9 '*' 1\nopen                *open the file with element
         $parameter$\n 9";
    Export expression: "'$parameter$.dat'"
  End line
  Start a new line
    Export expression: "$Tag(PTCRun)$"
  End line
  Start a new line
    Export expression: "$Tag(LayerNumber)$\nclose\n9"
    End line
  End loop
  Loop for: Node Parameters
    If: $HasTag(PTCRun)$
      Start a new line
        Export expression: "dfile\n 9 '*' 1\nopen                 *open the file with node
           $parameter$\n 9";
        Export expression: "'$parameter$.dat'"
      End line
      Start a new line
        Export expression: "$Tag(PTCRun)$"
      End line
      Start a new line
        Export expression: "$Tag(LayerNumber)$\nclose\n9"
      End line
    End if
  End loop
  Start a new line
    Export expression: "indexoff"
  End line
#
  Start a new line
    Export expression: "*  output commands\n*"
  End line
  Start a new line
    Export expression: "eccoors             *echo nodal coordinates  "
  End line
  Start a new line
    Export expression: "ecincid             *echo incidence list"
  End line
  Start a new line
    Export expression: "ecelev              *echo interface elevations "
  End line
  Start a new line
    Export expression: "eclayer             *echo material properties at all elements"
  End line
  Start a new line
    Export expression: "ecbc                *echo boundary conditions "
  End line
  Start a new line
    Export expression: "ecrain              *echo infiltration flux"
  End line
  Start a new line
    Export expression: "ecinith             *echo initial head at all nodes "
  End line
  Start a new line
    Export expression: "rdprtkey            *specification of solution output intervals"
  End line
  Start a new line
    Export expression: " 50 50 50 50        *kftbeg kftinc kmtbeg kmtinc   "
  End line
  Start a new line
    Export expression: "rdvopkey            *specification of velocity output intervals"
  End line
  Start a new line
    Export expression: " 50 50              *kvtbeg kvtinc"
  End line
  Start a new line
    Export expression: "*rdqopkey            *calculated flux at flow Dirichlet nodes "
  End line
  Start a new line
    Export expression: "* 10 10              *kqtbeg kqtinc        "
  End line
  Start a new line
    Export expression: "*rddopkey            *calculated dispersive flux at trans. Dir.
           nodes"
  End line
  Start a new line
    Export expression: "* 10 10              *kdtbeg kdtinc "
  End line
  Start a new line
    Export expression: "*\n* flow simulation\n*"
  End line
  Start a new line
    Export expression: "*rdwtable"
  End line
  Start a new line
    Export expression: "* 0 50   .001       *ytable niter epsilon (default is wtable off)
           "
  End line
  Start a new line
    Export expression: "rdcontrl"
  End line
  Start a new line
    Export expression: " 1 0 0 1 1          *yflow yvel yconc yincor ymbal "
  End line
  Start a new line
    Export expression: "gntime"
  End line
  Start a new line
    Export expression: " 20 10 100 2 1.5 175200 *itmax itchng itchmx nmspf chng tleng"
  End line
  Start a new line
    Export expression: " sim\n end"
  End line
End file
#
# End of PTC Run file
```

End of Script for creating PTC run file

```
#
```

Begining of Script for creating PTC data files

```
Redirect output to: inc.dat
  Loop for: Nodes
    Start a new line
      Export expression: NodeNumber(); [I5]
      Export expression: PositionX(); [F12.3]
      Export expression: PositionY() [F12.3]
    End line
  End loop
  Loop for: Elements
    Start a new line
      Export expression: ElementNumber(); [I5]
      Export expression: NthNodeNum(1); [I5]
      Export expression: NthNodeNum(2); [I5]
      Export expression: NthNodeNum(3); [I5]
      # the next line tells PTC that these are triangular elements.
      Export expression: 0 [I5]
    End line
  End loop
End file
#
# start output to files for each node related layer
#
Loop for: Node Parameters
  If: $HasTag(PTCRun)$
    If: !$HasTag(bc)$
      Redirect output to: $parameter$.dat
        Loop for: Nodes
          Start a new line
            Export expression: NodeNumber(); [I5]
            Export expression: $parameter$ [E20.6]
          End line
        End loop
      End file
    End if
    If: $HasTag(bc)$
      Redirect output to: $parameter$.dat
        Loop for: Nodes
          If: !isna(if(NodeOnBoundary()|NodeAboveCntr(Domain Outline)=1,$parameter$,$n/
          a))
            Start a new line
              Export expression: NodeNumber(); [I5]
              Export expression: $parameter$; [I5]
              Export expression: 0 [I5]
              Export expression: $Tag(bc)$ [E20.6]
            End line
          End if
        End loop
        Start a new line
          Export expression: "0/"
        End line
      End file
    End if
  End if
End loop
# start output to files for each element related layer
Loop for: Element Parameters
  Redirect output to: $parameter$.dat
    Loop for: Elements
```

```
      Start a new line
        Export expression: ElementNumber(); [I5]
        Export expression: $parameter$ [E20.6]
      End line
    End loop
  End file
End loop
```

A Closer Look

The following illustration explains the if-statements used in the part of the template which creates the output files for node related parameters. The most inner if statement is used to export to file only nodes lying on boundaries (internal and external) and that are assigned a boundary condition different then $n/a.

```
# start output to files for each node related layer
#
Loop for: Node Parameters
  If: $HasTag(PTCRun)$
    If: !$HasTag(bc)$
      Redirect output to: $parameter$.dat
        Loop for: Nodes
          Start a new line
            Export expression: NodeNumber(); [I5]
            Export expression: $parameter$ [E20.6]
          End line
        End loop
      End file
    End if
    If: $HasTag(bc)$
      Redirect output to: $parameter$.dat
        Loop for: Nodes
          If: !isna(if(NodeOnBoundary()|NodeAboveCntr(Domain
          Outline)=1,$parameter$,$n/a))
            Start a new line
              Export expression: NodeNumber(); [I5]
              Export expression: $parameter$; [I5]
              Export expression: 0 [I5]
              Export expression: $Tag(bc)$ [E20.6]
            End line
          End if
        End loop
        Start a new line
          Export expression: "0/"
        End line
      End file
    End if
  End if
End loop
```

Export only node parameters which are assigned a PTCRun tag

Export parameters other then Boundary Conditions.

Export Boundary Conditions parameters

Export only nodes lying on boundaries or on internal point sources (wells).

An Example of a PTC .run File Created by MeshMaker

The following pages list the PTC .run file that was automatically created by Argus MeshMaker for an example problem. The example problem is described below in the PTC .run file. The MeshMaker project file defining the example problem is also enclosed.

```
**********************************************************************
*
* data set for sample problem #1
*
**********************************************************************
**********************************************************************
*
*  items included in this example:
*  - pumping and injecting wells
*  - several types of boundary conditions: constant head, specified
*    flux, leakage into aquifer, leakage out of aquifer, constant
*    concentration, and specified mass flux of contaminant
*  - 3 layer system with a middle clay layer (2) with a high
*    permeability window
*  - water table calculations
*  - rainfall
*
*  solution procedure:
*    flow is calculated first, and then transport is calculated using
*    the resultant flow field
*
*  units:
*    length is in feet and time is in days, unless otherwise noted
*
*  note:
*    this problem contains all the features of sample problem #1a,
*    but uses the simplest and easiest to understand data structure.
*    generation of data and use of the dfile only occurs for the drnrain
*    command.
*    there are no graphics commands in this example.
*
***********************************************************************
*  begin input for cfile
***********************************************************************
*
*
open                    *open a file with unit number of 3
 3 'ptc.out'
cecho                   *echo commands to unit 3
3
efile                   *echo input errors to unit 3
3
```

```
open                    *open an output file for flow mass balance
11 'ptc.mbf'
open                    *open an output file for transport mass balance
12 'ptc.mbm'
mwrit                   *direct flow and trans. mass bal. to 11 & 12 resp.
11 12
rwrit                   *output flow and mass solutions to unit 3 & 3 resp.
3 3
owrit                   *output echo requests to unit 3
3
rdtitle                 *input title
 'MeshMaker Problem'
rddims                  *specify number of nodes, elements, layers
  259   459 2
rddifusn                *molecular diffusion
       1.000000e-004
rdweight                *upstream weight (default = 1.)
    1.00
indexon
dfile
9 '*' 1
open
9  'inc.dat'
drdcoors
rdincid
close
9
indexoff
indexon
dfile
 9 '*' 1
open                *open the file with element Storl1
 9 'Storl1.dat'
drestor
1
close
9
dfile
 9 '*' 1
open                *open the file with element Storl2
 9 'Storl2.dat'
drestor
2
close
9
dfile
 9 '*' 1
open                *open the file with node Rainl1
 9 'Rainl1.dat'
drnrain
1
close
9
```

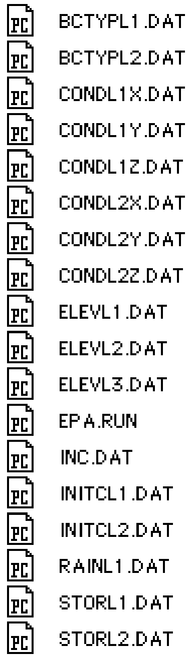
```
dfile
 9 '*' 1
open                *open the file with node elevl1
 9 'elevl1.dat'
drnelev
1
close
9
dfile
 9 '*' 1
open                *open the file with node elevl2
 9 'elevl2.dat'
drnelev
2
close
9
dfile
 9 '*' 1
open                *open the file with node elevl3
 9 'elevl3.dat'
drnelev
2
close
9
dfile
 9 '*' 1
open                *open the file with node condl1x
 9 'condl1x.dat'
drncondx
1
close
9
dfile
 9 '*' 1
open                *open the file with node condl1y
 9 'condl1y.dat'
drncondy
1
close
9
dfile
 9 '*' 1
open                *open the file with node condl1z
 9 'condl1z.dat'
drncondz
1
close
9
dfile
 9 '*' 1
open                *open the file with node condl2x
 9 'condl2x.dat'
drncondx
```

```
2
close
9
dfile
 9 '*' 1
open                *open the file with node condl2y
 9 'condl2y.dat'
drncondy
2
close
9
dfile
 9 '*' 1
open                *open the file with node condl2z
 9 'condl2z.dat'
drncondz
2
close
9
dfile
 9 '*' 1
open                *open the file with node InitCL1
 9 'InitCL1.dat'
drninith
1
close
9
dfile
 9 '*' 1
open                *open the file with node InitCL2
 9 'InitCL2.dat'
drninith
2
close
9
dfile
 9 '*' 1
open                *open the file with node BCTypL1
 9 'BCTypL1.dat'
drnbcflo
1
close
9
dfile
 9 '*' 1
open                *open the file with node BCTypL2
 9 'BCTypL2.dat'
drnbcflo
2
close
9
indexoff
*  output commands
```

```
*
eccoors             *echo nodal coordinates
ecincid             *echo incidence list
ecelev              *echo interface elevations
eclayer             *echo material properties at all elements
ecbc                *echo boundary conditions
ecrain              *echo infiltration flux
ecinith             *echo initial head at all nodes
rdprtkey            *specification of solution output intervals
 50 50 50 50        *kftbeg kftinc kmtbeg kmtinc
rdvopkey            *specification of velocity output intervals
 50 50              *kvtbeg kvtinc
*rdqopkey            *calculated flux at flow Dirichlet nodes
* 10 10              *kqtbeg kqtinc
*rddopkey            *calculated dispersive flux at trans. Dir. nodes
* 10 10              *kdtbeg kdtinc
*
* flow simulation
*
*rdwtable
* 0 50   .001        *ytable niter epsilon (default is wtable off)
rdcontrl
 1 0 0 1 1          *yflow yvel yconc yincor ymbal
gntime
 20 10 100 2 1.5 175200 *itmax itchng itchmx nmspf chng tleng
sim
 gwrit
9
 grfname
 'srfp' 1
grsolfmt
'(i5,t1,(i4,t1,1pe15.7,1pe15.7,1pe15.7:))'
*
* transport simulation
*
rdcontrl
 1 1 1 1 1          *yflow yvel yconc yincor ymbal
 rdtime
  1 5 50 20 175200 1.5 *itmax itchng itchmx nmspf delt chng
 sim
 end
```

List of data files Automatically Created by MeshMaker for PTC

In addition to creating the PTC .run file, MeshMaker also automatically creates al the necessary data files for running the problem with PTC. The following picture lists the .run file and data files created by MeshMaker for the example problem.



BCTYPL1.DAT
BCTYPL2.DAT
CONDL1X.DAT
CONDL1Y.DAT
CONDL1Z.DAT
CONDL2X.DAT
CONDL2Y.DAT
CONDL2Z.DAT
ELEVL1.DAT
ELEVL2.DAT
ELEVL3.DAT
EPA.RUN
INC.DAT
INITCL1.DAT
INITCL2.DAT
RAINL1.DAT
STORL1.DAT
STORL2.DAT