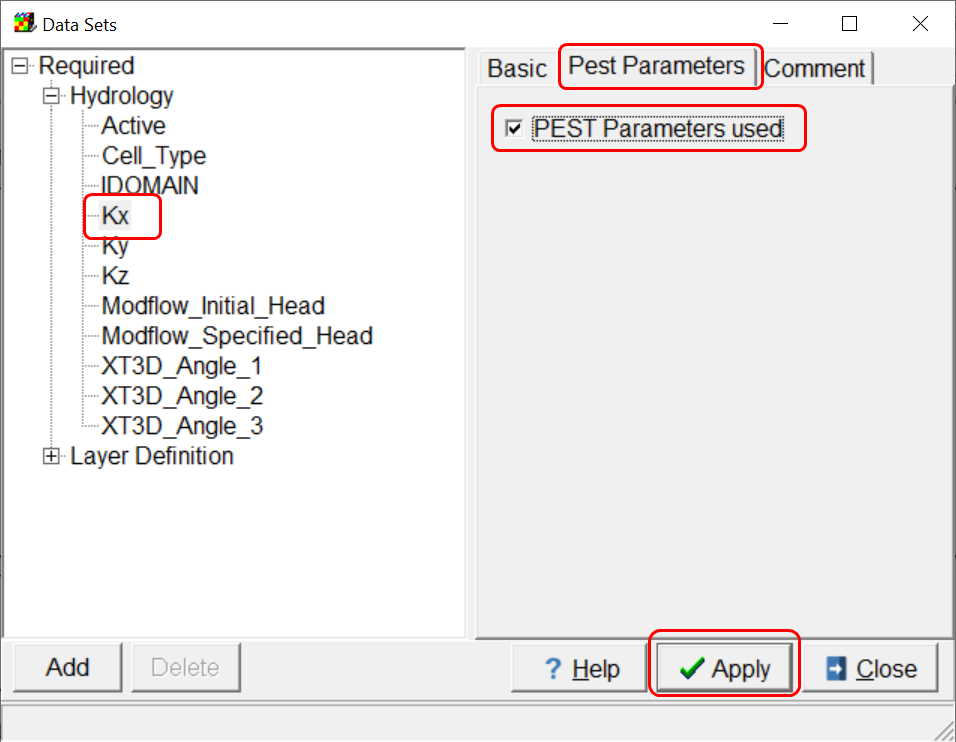
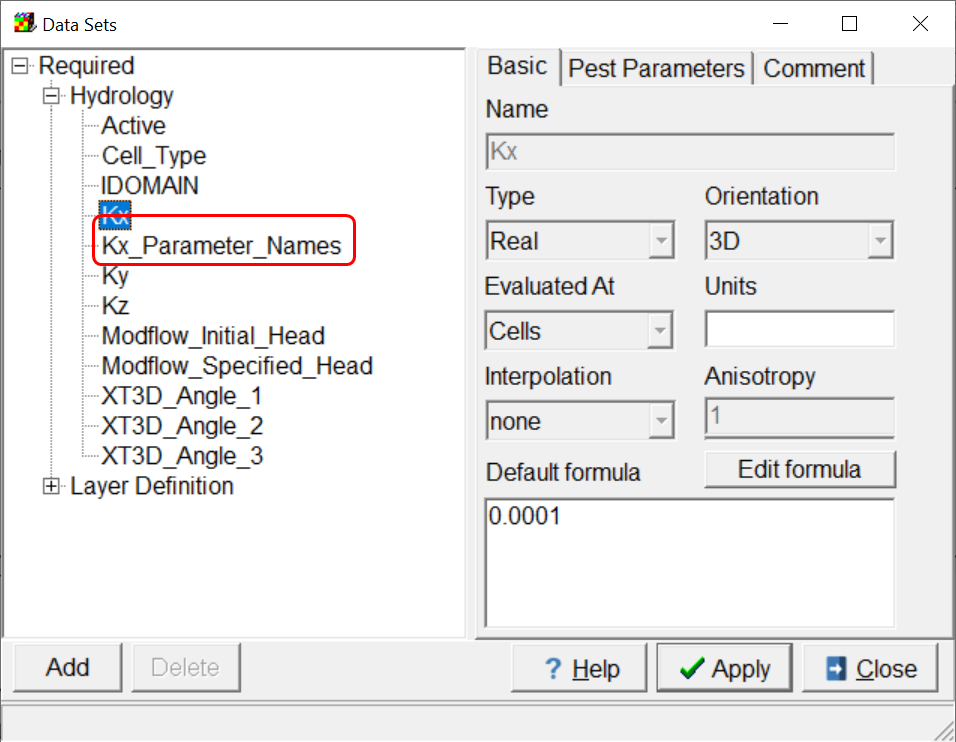
ModelMuse with Support for PEST – Beta 5

This version adds more flexible methods for adding (or deleting) pilot points. It also resolves some of the known issues from Beta 4.

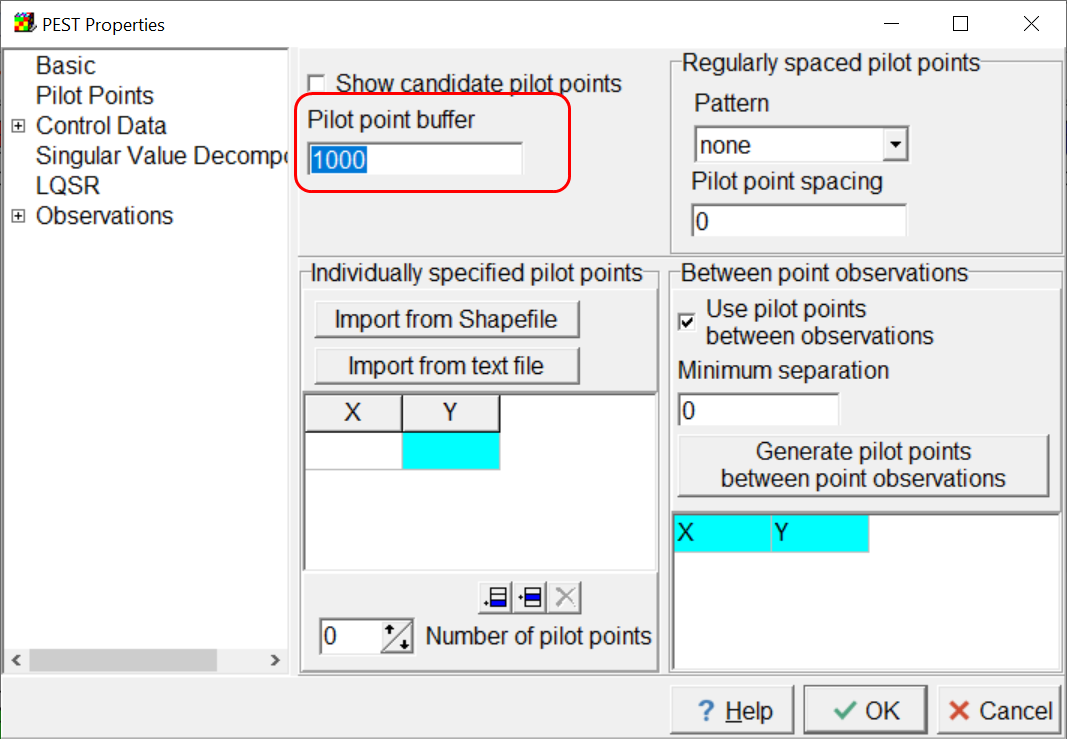
# How Pilot Points are Used to Help Assign Values to Data Sets

If the user wishes PEST to assign values to an array during the calibration process, the user selects the corresponding data set in the **Data|Edit Data Sets** dialog box. If the corresponding data set is eligible for PEST calibration, a **PEST Parameters** tab will appear. If the **PEST Parameters used** check box is checked on that tab, a new data set will be created when the Apply button is clicked. The new data set will be used for specifying the names of the parameters to be applied to different locations in the related data set.

After the user has defined parameters in the **Model|Manage Parameters** dialog box, the user can assign the parameter names to data set using either default formulas for the Parameter\_Names data set or objects. When exporting model input files, ModelMuse will create scripts for the program PLPROC that will substitute parameter values into the array. Each time PEST runs the model, these scripts will be modified to include the current values of the parameters that PEST is testing.

This process is modified if the user checks the **Pilot Points** checkbox for a parameter in the **Model|Manage Parameters** dialog box. If that option is selected, instead of substituting the parameter value, interpolation among pilot points will be used to assign values to the data set. For each parameter that uses pilot points on each layer of the data set, a group of pilot points will be selected. The selected pilot points will be those that either are in a cell for which the corresponding parameter is to be used or are within a user-defined distance of such a cell. The distance is the **Pilot point buffer** defined in the **Model|Pest Properties** dialog box. If no pilot points that meet these criteria are found, the parameter value will be substituted as if the **Pilot Points** option had not been selected.



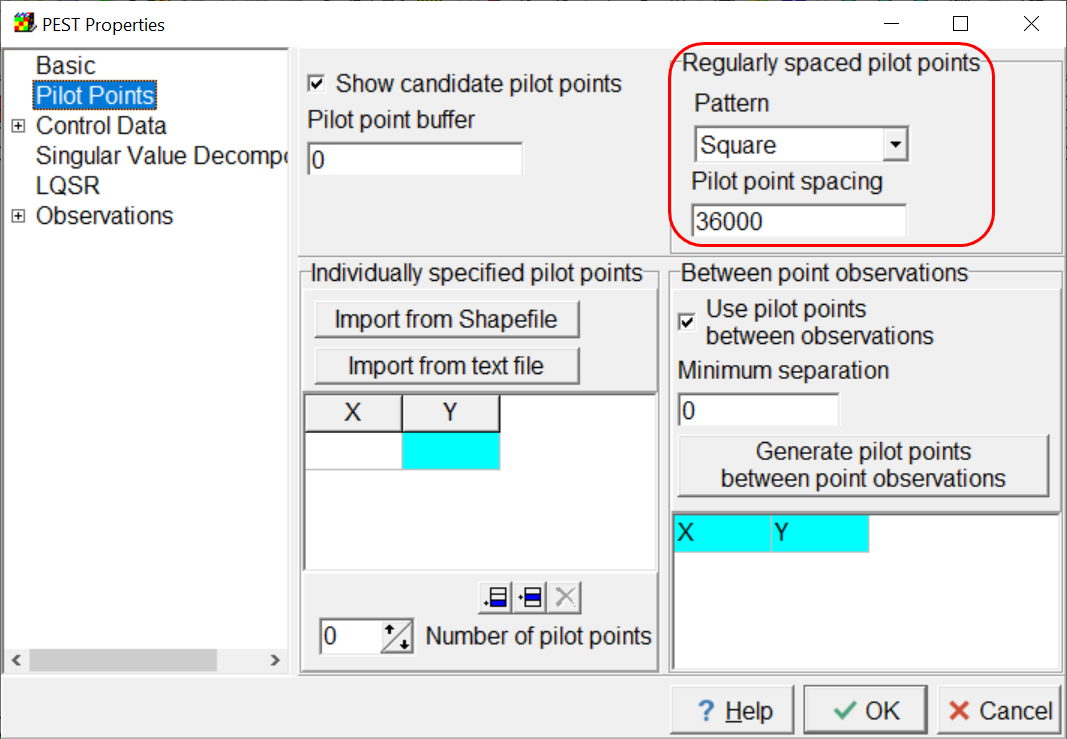
A value is defined for each selected pilot point, and these values are used to interpolate to the cells for which the corresponding parameter is used in the model. The initial value is supplied by ModelMuse and is either the value in the cell where the pilot point is located or the value of the closest cell to the pilot point for which the parameter is to be used. During the calibration process, PEST will modify the pilot point values so that different values are assigned to the model cells.

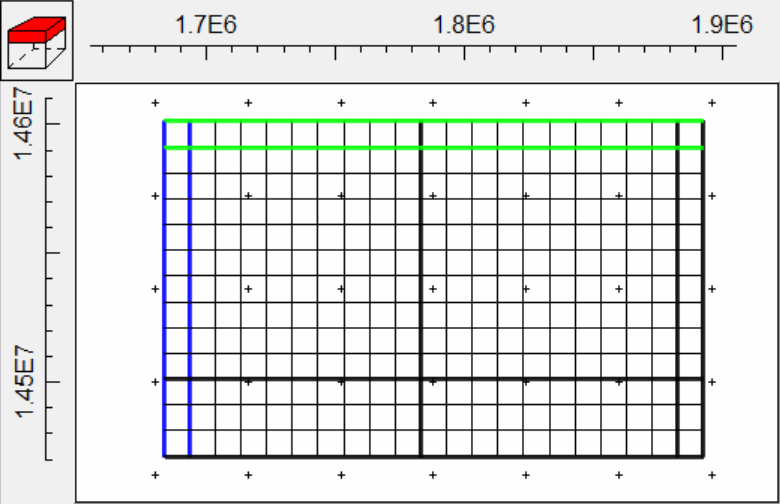
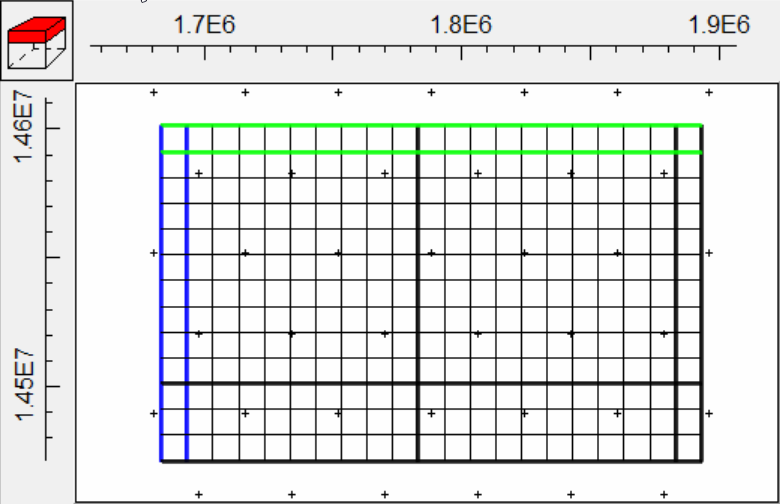
# Defining Pilot Point Locations

Model Muse provides several ways to define pilot point locations. These methods can each be used separately or together with other methods.

## Regularly Spaced

Regularly spaced pilot points arranged either in squares or equilateral triangles can be defined by specifying the desired pattern and pilot point spacing in the **Model|Pest Properties** dialog box. These pilot points will be displayed as small plus symbols. By design, some of these pilot points may lie outside the model grid or mesh. The user controls their position is through the selected pattern and spacing. It is not possible to delete such pilot points individually. If the spacing is set to zero or the pattern is set to “none”, pilot points will not be defined by this method. The spacing is typically at least several times larger than the cell size.

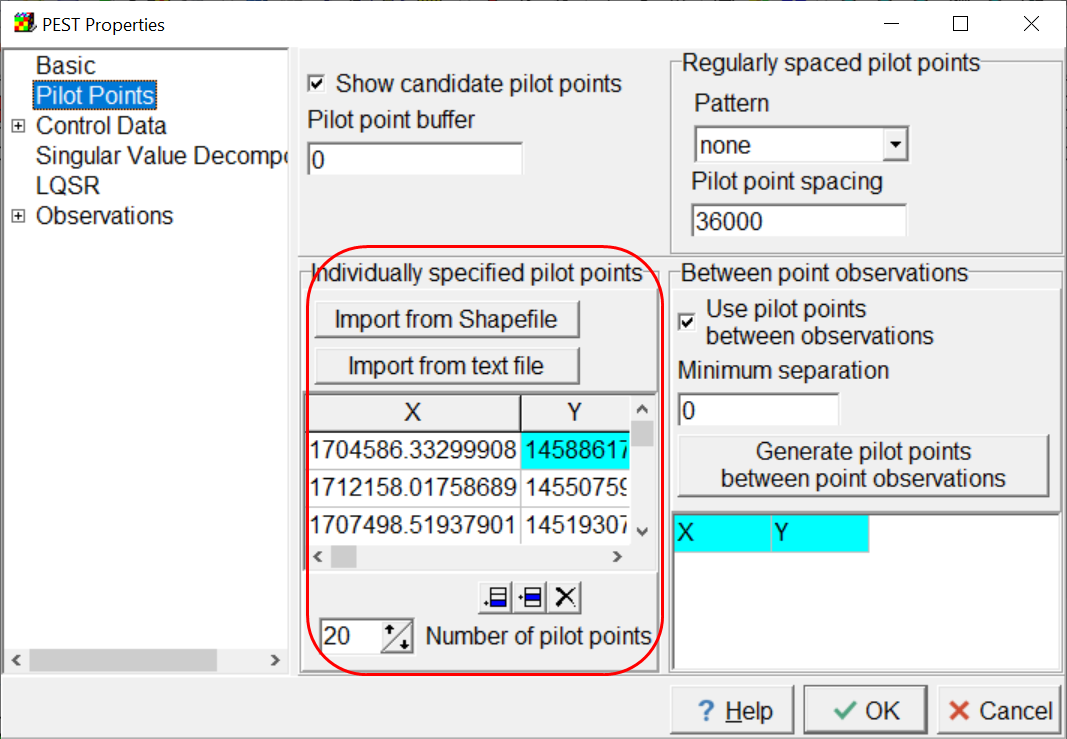


Regularly space pilot points are often used where there is little information available about the distribution of the properties to be estimated by pilot point interpolation.

## Individually Specified Pilot Point Locations.

Another way to specify pilot points is to specify them individually This can be done in several ways. One way is to import them from a text file or a Shapefile. The **Import from Shapefile** and **Import from text file** buttons on the **Model|Pest Properties** dialog box can be used for that purpose. For text files, each line must define a pilot point location. The line must start with two numbers separated by a comma and/or one or more spaces. Any additional text on the line will be ignored. For Shapefiles, each unique point in the Shapefile will define a pilot point location. The pilot point locations will be displayed in a table in the **Model|Pest Properties** dialog box. The user can also type pilot point locations in that table or copy locations from a spreadsheet program and paste them in the table. Finally, on the ModelMuse main form, the user can select **Edit|Add Pilot Point** or click the **Add pilot point** button  and then click on the top view of the model to add a pilot point at the location where the mouse button was released.

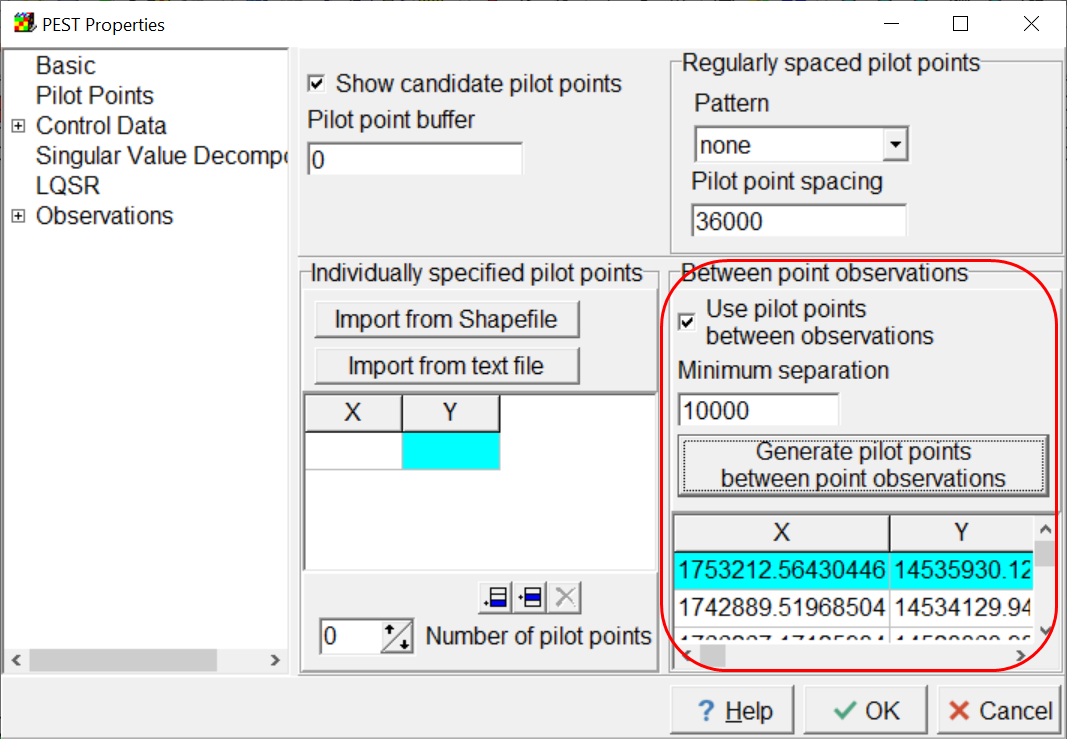


Individually specified pilot points are displayed with a small x symbol instead of a small plus symbol. They can be deleted by the user.

Individually specified pilot points can be useful in helping to express the user’s expert knowledge about a property. For instance, suppose we believe that hydraulic conductivity of a unit is higher near rivers than along ridges. In such a case we might place the pilot points along the river and on the ridge lines. This will help guide the interpolation process to a hydraulic conductivity distribution that is in accordance with our expert knowledge.

## Generate Pilot Points Locations Between Observation Locations

Pairs of point observations can give information about the properties of the material between them. For example, a large head difference between to observations might mean that the material between them has a low transmissivity whereas a small head difference might mean that the material between them has a low transmissivity. ModelMuse provides a way of automatically generating pilot point locations between point observations. To do this, the user clicks the **Generate pilot points between point observations** button on the **Model|Pest Properties** dialog box. ModelMuse will identify all the point objects that define calibration observations and create a triangulation of them. It will then create pilot points at the midpoint of the edges of each triangle.



If the point observations are closely spaced together, especially if there are several in the same cell, the pilot points may be too tightly clustered together. In such cases, it can be desirable to delete some of the closely spaced pilot points. To do this, the user can specify a **minimum separation** when generating the pilot points between observations. ModelMuse will use this separation to delete some of the pilot points so that the remaining ones are not too closely spaced.

Pilot points generated between point observations are displayed with a small x symbol instead of a small plus symbol. They can be deleted by the user.

# Deleting Pilot Points

Pilot points can be deleted in several ways. First, for individually specified pilot points, the user can delete the row in the table containing them in the **Model|Pest Properties** dialog box. The user can also select **Edit|Delete Pilot point(s)** or click on a **Delete pilot point(s)** button  and then click on a pilot point to delete it. This only applies to individually specified pilot points or pilot points generated between point observations. It does not apply to the regularly spaced pilot points. To distinguish between these two types of pilot points, the regularly spaced pilot points are drawn using a plus symbol and the others are drawn with a x symbol. To delete multiple pilot points at once, select **Edit|Delete Pilot point(s)** or click on a **Delete pilot point(s)** button  and then click down on a location on the top view of the model. This location defines one corner of a rectangle. Then drag the mouse while holding the mouse button down. The location where the mouse button is released defines the opposite corner of a rectangle. Pilot points in the rectangle will be deleted.

# Example of a PLPROC Script

#Script for PLPROC

#Read pilot point data

PilotPoints1 = read\_list\_file(skiplines=0,dimensions=2, &

plist='Hk2\_1';column=5, &

id\_type='indexed',file='PestPilotPointTest.Kx.Hk2.1.pp')

#Read MODFLOW 6 grid information file

cl\_Discretization = read\_mf6\_grid\_specs(file='PestPilotPointTest.dis.grb', &

dimensions=2, &

slist\_layer\_idomain=id1; layer=1, &

plist\_layer\_bottom =bot1; layer=1, &

plist\_top = top)

#Read data to modify

read\_list\_file(reference\_clist='cl\_Discretization',skiplines=1, &

slist=s\_PIndex1;column=2, &

plist=p\_Value1;column=3, &

file='PestPilotPointTest.Kx.PstValues')

#Read parameter values

Hk1 = 1.000000000000000D-004

# Pilot points are not used with Hk1.

Hk2 = 1.000000000000000D-002

# Pilot points are used with Hk2.

# Modfify data values

temp=new\_plist(reference\_clist=cl\_Discretization,value=0.0)

# Setting values for layer 1

# Setting values for parameter Hk1

# Substituting parameter values in zones

p\_Value1(select=(s\_PIndex1 == 1)) = p\_Value1 \* Hk1

# Setting values for parameter Hk2

# Substituting interpolated values

# Get interpolated values

temp=Hk2\_1.krige\_using\_file(file='PestPilotPointTest.Kx.Factors1';form='formatted', &

transform='log')

# Write interpolated values in zones

p\_Value1(select=(s\_PIndex1 == 2)) = temp

#Write new data values

write\_column\_data\_file(header='no', &

file='arrays\PestPilotPointTest.npf.Kx\_1.txt';delim="space", &

plist=p\_Value1)

# Other changes

* The RunModel.bat file now contains commands to delete some model output files and some model input files. This means that if the model fails to run properly PEST will not read output from another MODFLOW run.
* ModelMuse can not import SUTRA input files generated by PEST for the purposes of visualizing the model input. The command to do this is “File|Import|SUTRA Files.”
* ModelMuse can not import PVAL files containing parameters intended only for PEST.
* The Utility programs used by ModelMuse in conjunction with PEST no longer need to be in the model directory. They can also be either in the PEST directory or the directory containing ModelMuse.

# Known Issues

There are several issues that are yet to be resolved. Here are the most prominent ones of which you should be aware.

## No Prior Information for Pilot Points

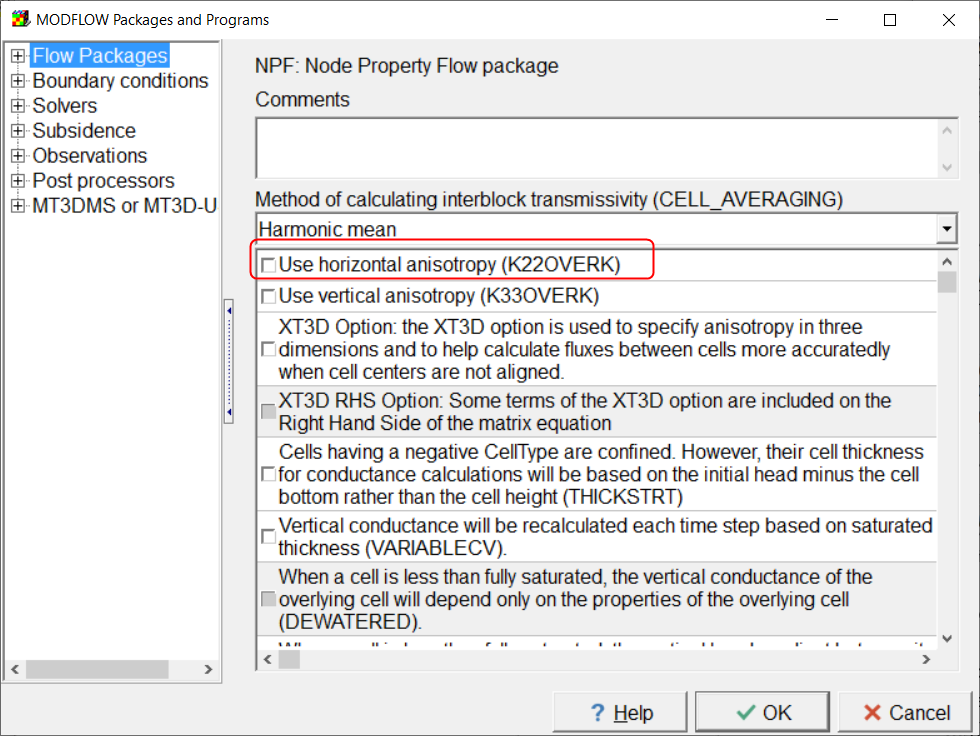
ModelMuse does not include any prior information for Pilot Points as would typically be required. You may wish to use GENREG in the PEST groundwater utilities. To add such information. There are other utilities in the PEST Utility Support Software that you might also consider.

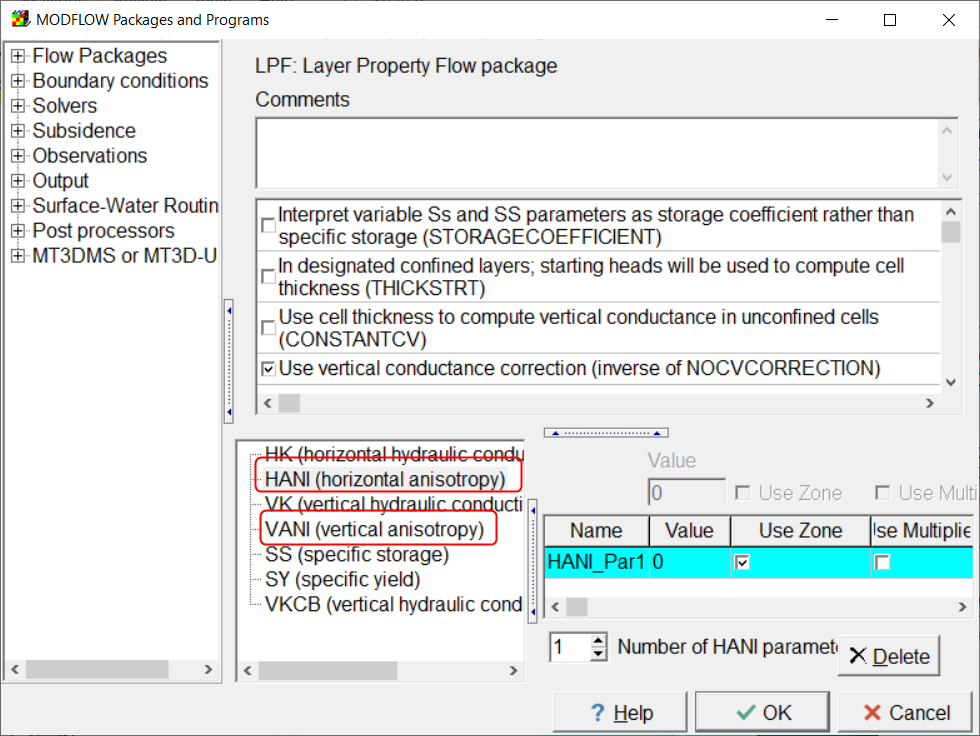
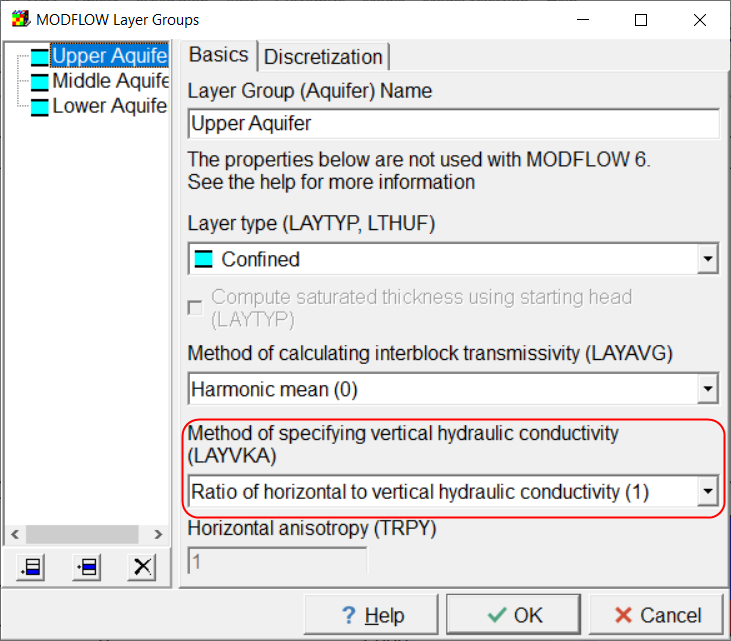
## Linked Data Sets and Anisotropy.

In ModelMuse, there are a number of data sets whose default formulas link them with other data sets. The most prominent of these are Kx, Ky, and Kz. The default formulas for Ky is Kx and the default formula for Kz is Kx/10. Now consider the case where you want Ky to have the same value as Kx but you also want to calibrate Kx. One option would be to define one or more parameters for Kx but do nothing with Ky. PEST will then modify Kx but the value of Ky was set by ModelMuse and doesn’t instruct PEST to modify it so nothing happens to Ky so you don’t achieve your goal of having Ky be the same as Kx. Another option would be to have both Kx and Ky be estimated and to use the same parameters for both and in the same locations. This doesn’t work either so long as the formula for Ky is set to Kx. Suppose the parameter value was 1E-4 m/s. Kx then gets a value of 1E-4 times whatever value was assigned to Kx by the default formula or objects. Let’s assume that the default formula for Kx is 1 so the final value for Kx is 1E-4. The value of Ky set by the default formula is 1E-4. This is multiplied by the parameter value to get a final value of 1E-8. That is very different from your goal of having Kx equal to Ky.

The best way to handle this to meet the goal is to specify horizontal anisotropy as the model input rather than specifying Ky directly. In MODFLOW 6, this is done using an option in the NPF package. There is a similar option for vertical anisotropy. In MODFLOW-2005, horizontal anisotropy is part of the model input by default and you can also have vertical anisotropy be part of the model input. You can also have horizontal anisotropy and vertical anisotropy parameters. The parameters are specified in the “Model|MODFLOW Packages and Programs” dialog box. The vertical anisotropy option is specified in the “Model|MODFLOW Layers” dialog box.

There are no similar options for SUTRA. Your best option is probably to use tied parameters for Ky and Kz. However, this isn’t supported right now if pilot points are used.



## Tied Parameters and Pilot Points.

At present, if Pilot Points is selected for a parameter in the “Model|Manage Parameters” dialog box, the parameter is replaced by a series of pilot points. Therefore you can not have such a parameter involved in tied parameters either by being tied to another parameter or by having another parameter being tied to it. However, ModelMuse doesn’t prevent you from tying such parameters in either direction. There might be some way of handling this at least in some cases but, at present, ModelMuse will just create a defective PEST control file.

## SUTRA Boundary Condition Parameters

There isn’t yet a way a specifying boundary condition parameters for SUTRA.

## Pilot Points for Boundary Conditions

ModelMuse does not yet provide a way to utilize pilot points for boundary conditions.

## Bugs in SUTRA

There are some bugs in the released version of SUTRA that inhibit it from being used with PEST. Alden Provost has provided a fixed version of SUTRA but we are still awaiting the official release of a fixed version.

## Pilot Points are not displayed in the Export Image dialog box.