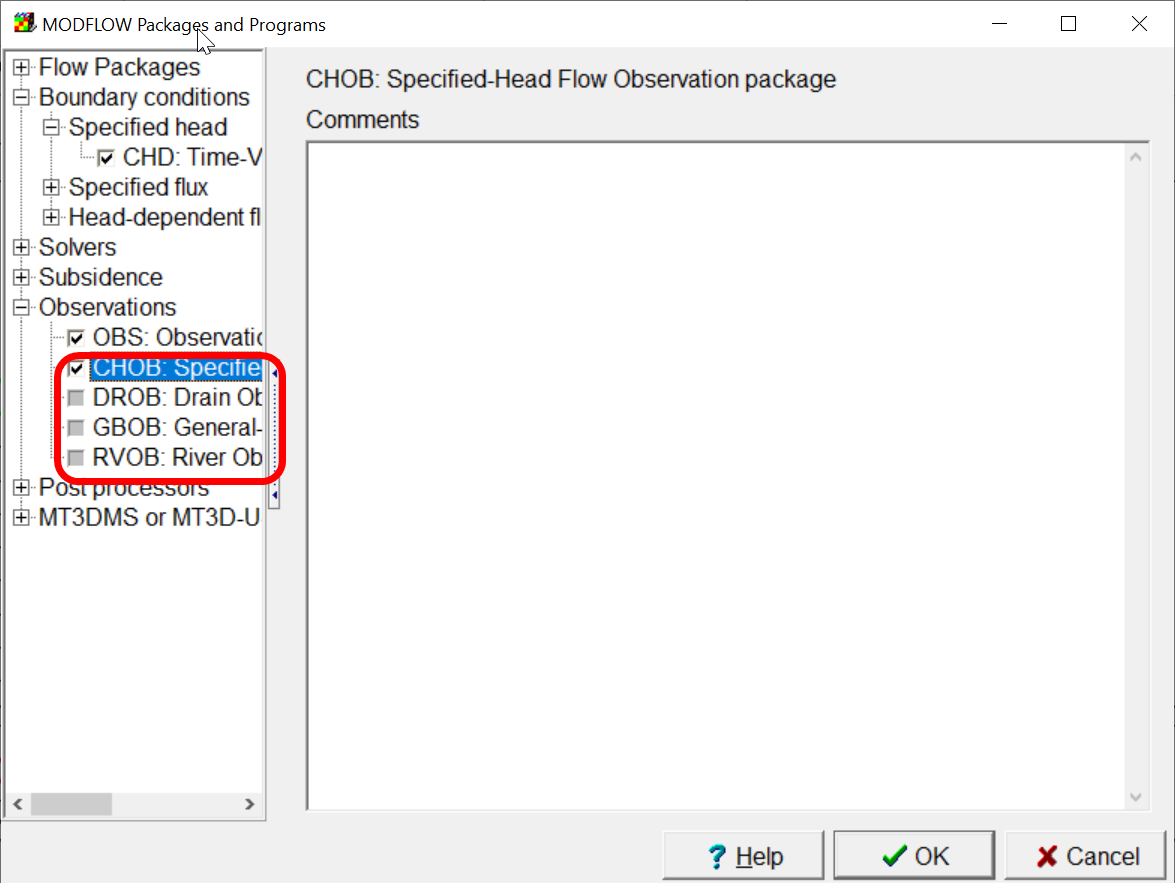
Defining Observations for PEST in ModelMuse Beta 1.

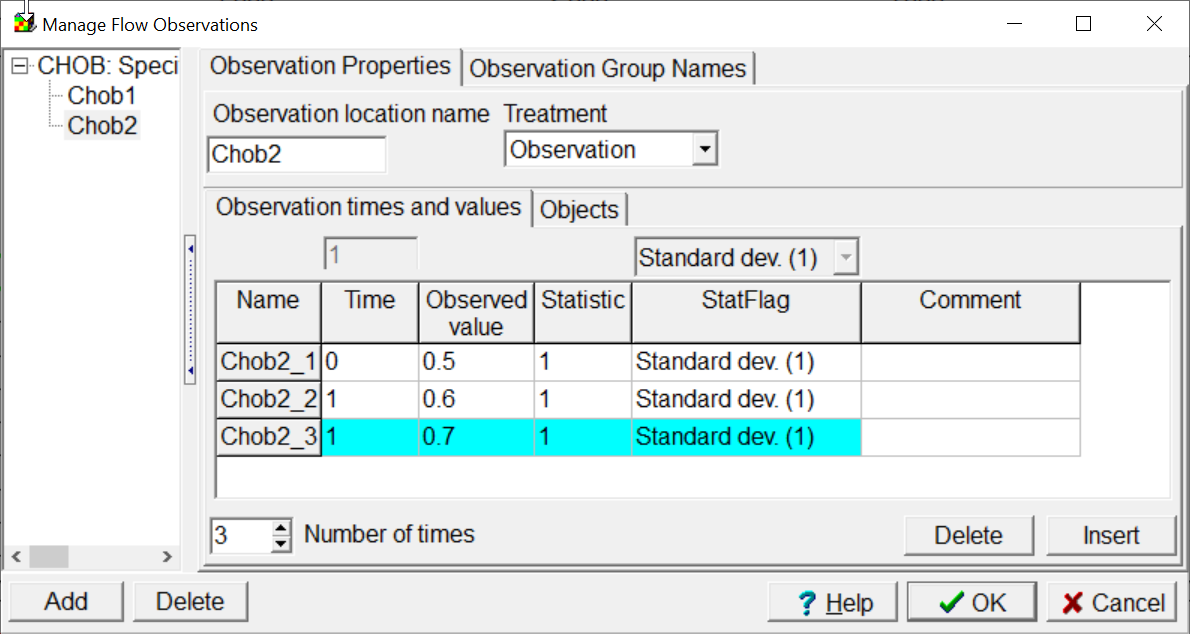
The current beta version of ModelMuse allows the modeler to define observations in ModelMuse. When this is done, ModelMuse will export one or more script files that can be used with utility programs to both extract observations from the model input files and to generate an instruction file for PEST. There are three different utility programs. They are distributed along with this beta version of ModelMuse. Documentation for the utility programs is included in separate documents. The observations can all be defined in similar ways within ModelMuse but a different utility program is required depending on whether the model is a MODFLOW-2005 (or related), MODFLOW 6, or SUTRA model. Examples of ModelMuse model files are include with the beta release to illustrate how the various observation types can be defined and how the utility programs can be used.

# MODFLOW 6

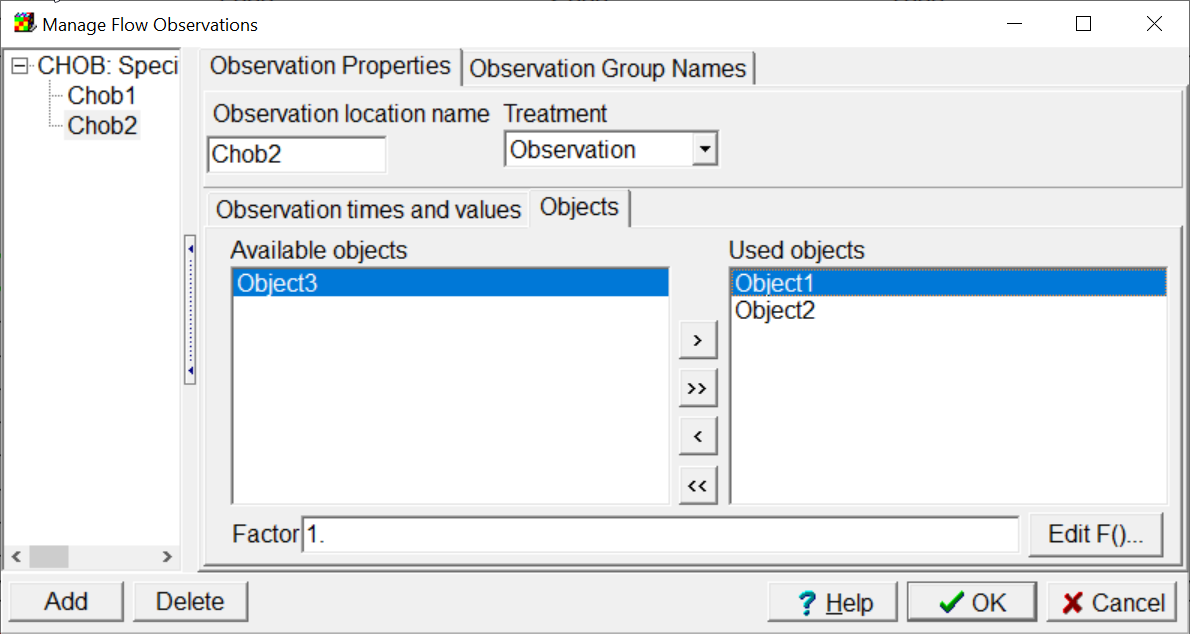
The first change to ModelMuse is in the MODFLOW Packages and Programs dialog box. In the beta version, the checkboxes for the CHOB, DROB, GBOB, RVOB packages are present whereas in the released version of ModelMuse, they are only present in MODFLOW-2005 and related models such as MODFLOW-NWT. The CHOB, DROB, GBOB, RVOB packages are not part of MODFLOW 6 but ModelMuse now allows the modeler to define observations for PEST for the CHD, DRN, GHB, and RIV packages in (nearly) the same way as was done with MODFLOW-2005. The big difference is that behind the scenes, ModelMuse will take care of aggregating flow data from different boundary cells using a separate utility program rather than relying on MODFLOW to do the aggradation.



If any of the checkboxes for the CHOB, DROB, GBOB, RVOB packages is checked, another dialog box will be displayed, the Manage Flow Observations checkbox. It will be displayed either when the MODFLOW Packages and Programs dialog box is closed or when the user selects “Model|Manage Flow Observations”. This dialog box has been part of ModelMuse for several years; you can read about it in the ModelMuse help. In brief, the modeler defines observation groups by clicking the Add button. Each observation group can be used to define one or more observations. All the observations in an observation group will involve the same boundary cells but each observation will have a different observation time. In the illustration below, two observation groups have been defined named Chob1 and Chob2. In the Chob2 group, there are three observations named Chob2\_1, Chob2\_2, and Chob2\_3. Each of these observations has a different observation time.

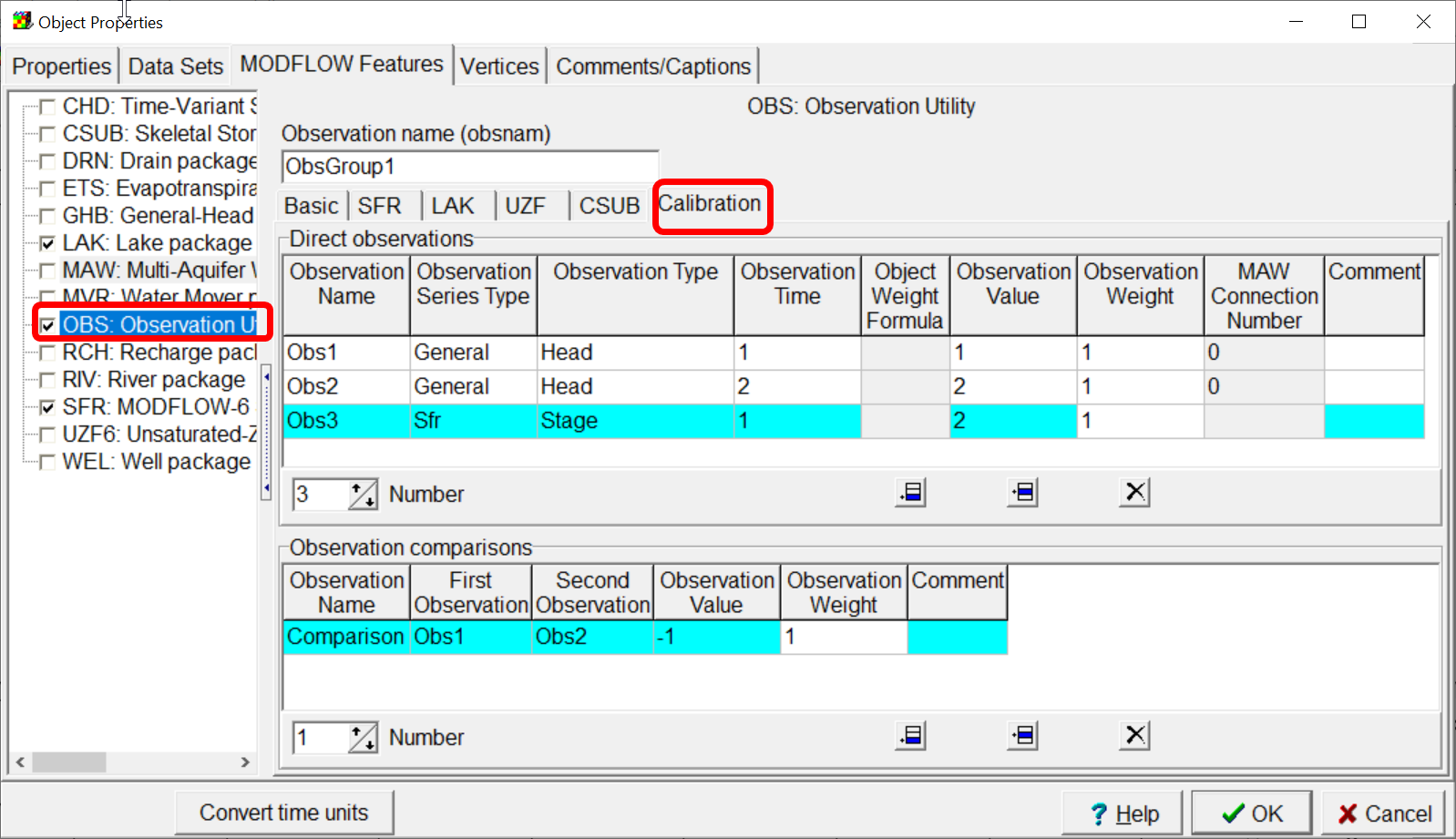


Each flow observation must have one or more boundary cell associated with it. In the case illustrated below, there are three specified head boundaries defined by objects named Object1, Object2, and Object3. The flows through the specified head boundaries defined by Object1 and Object2 are part of the observation while the flow through the specified head boundaries defined by Object3 are not part of the observation. In MODFLOW-2005, there is a “Factor” formula associated with each object. The formula would be evaluated at each boundary cell associated with the object. The formula should evaluate to a number between 0 and 1. MODFLOW-2005 would multiply flow the boundary cell by the factor and sum of all the products would be the simulated value for the observation. This allowed the user to include only part of some flows through a boundary cell in the observation. In practice, however, modelers nearly always left the factor at the default value of 1 so that all of the boundaries defined by an object were fully included in the simulated value. Because of this and because it was simpler for me, **the Factor formula is not used with flow observations in MODFLOW 6**.



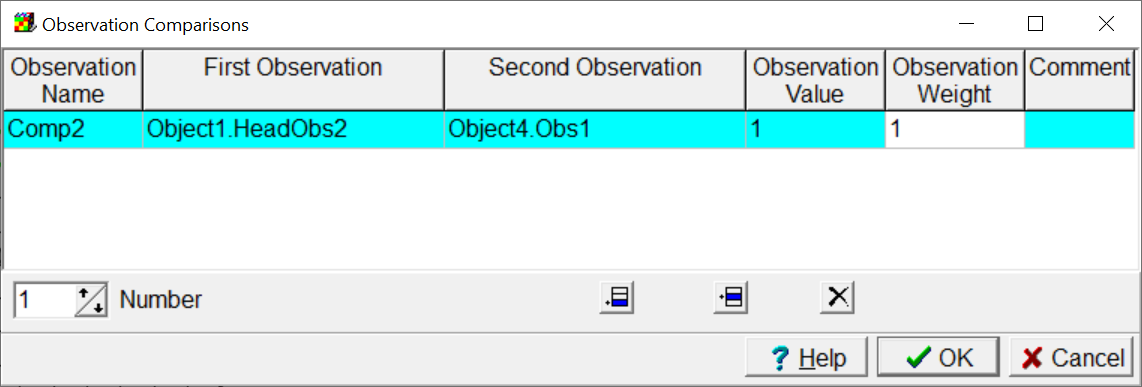
The Observation Utility in MODFLOW 6, allows you to specify observations of various types. These observation types are documented in the file mf6io.pdf distributed with MODFLOW 6. For all these observation types, MODFLOW 6 will produce a time-series of values for cells or groups of cells. Typically, the simulation times will not correspond exactly with the observation time. For some observation types, such as head observations, it may also be desirable to interpolate between the cell centers of the cells around the observation location to the location of the observation. ModelMuse, in conjunction with a utility program will interpolate in time and space to the time and location of the observation. For spatial interpolation, a finite-element basis function will be used to interpolate to the observation location.

For each observation used for PEST parameter estimation, you must create an object. Most often, it will need to be a point object because most observation types are defined at a single point. If it is possible to define an observation for calibration purposes with an object, an extra tab named Calibration will be present on the OBS pane of the Object Properties dialog box (see illustration below). For each observation, you must define the observation name, type, time, value and weight. (You can ignore the “Object Weight Formula”. I plan to eliminate it.) In the lower half of the pane, you can define observation comparisons as observations. The observed value for an observation comparison should be the difference between the observed value of two observations defined in the upper half of the Calibration tab. In this case, the value of the comparison is -1 which is the value of the value of Obs1 minus the value of Obs2. At present, ModelMuse does not actually use the observation value or observation weight. They will be used in the future. If an observation is used in an observation comparison, however, it would normally be appropriate to set the weight of that observation to zero. (That isn’t done in the illustration below.



Observation comparisons defined in the lower half of the Calibration tab would normally be between observations of the same type at different times. However, ModelMuse does not restrict you from defining comparisons between any of the calibration observations defined by the object.

It is also possible to define observation comparisons between observations defined by different objects. This is done in the “Model|Edit Object Comparisons” dialog box as illustrated below. ModelMuse does not restrict such observation comparisons to observations of the same type or observations made at the same time. A typical use of such a comparison observation would be when the user has observed a gradient in head between two locations and has measured the difference in elevations of the well heads but has not surveyed in the elevations of the well head from a known survey marker. Because the true elevations of the well heads are not known, the weights assigned to the heads at those wells would need to be low. However, the difference in the elevation of the well heads is known and hence the weight that could be assigned to the difference in head between the wells could be considerably higher.



ModelMuse uses the observations defined in the model to create two input files for the utility program Mf6ObsExtractor. One of the input files is used to extract simulated values at specific times and locations from the MODFLOW 6 output files. The other input file is used to create an instruction file for PEST. A simplified version of the sequence operations when performing automated parameter calibration is as follows.

1. Run the uncalibrated model once.
2. Generate an instruction file with Mf6ObsExtractor.
3. Start automated parameter calibration.
4. PEST runs MODFLOW 6 using new parameter values.
5. PEST runs Mf6ObsExtractor to extract simulated values.
6. PEST reads the simulated values using the instruction file from step 2.
7. If the PEST stopping criteria have not been met go to step 4.

Mf6ObsExtractor is documented in a separate file. Mf6ObsExtractor has the potential to be used independently of ModelMuse. Mf6ObsExtractor can be compiled to run on Windows, Linux, and MacIntosh operating systems using the open source Free Pascal compiler.

# MODFLOW-2005

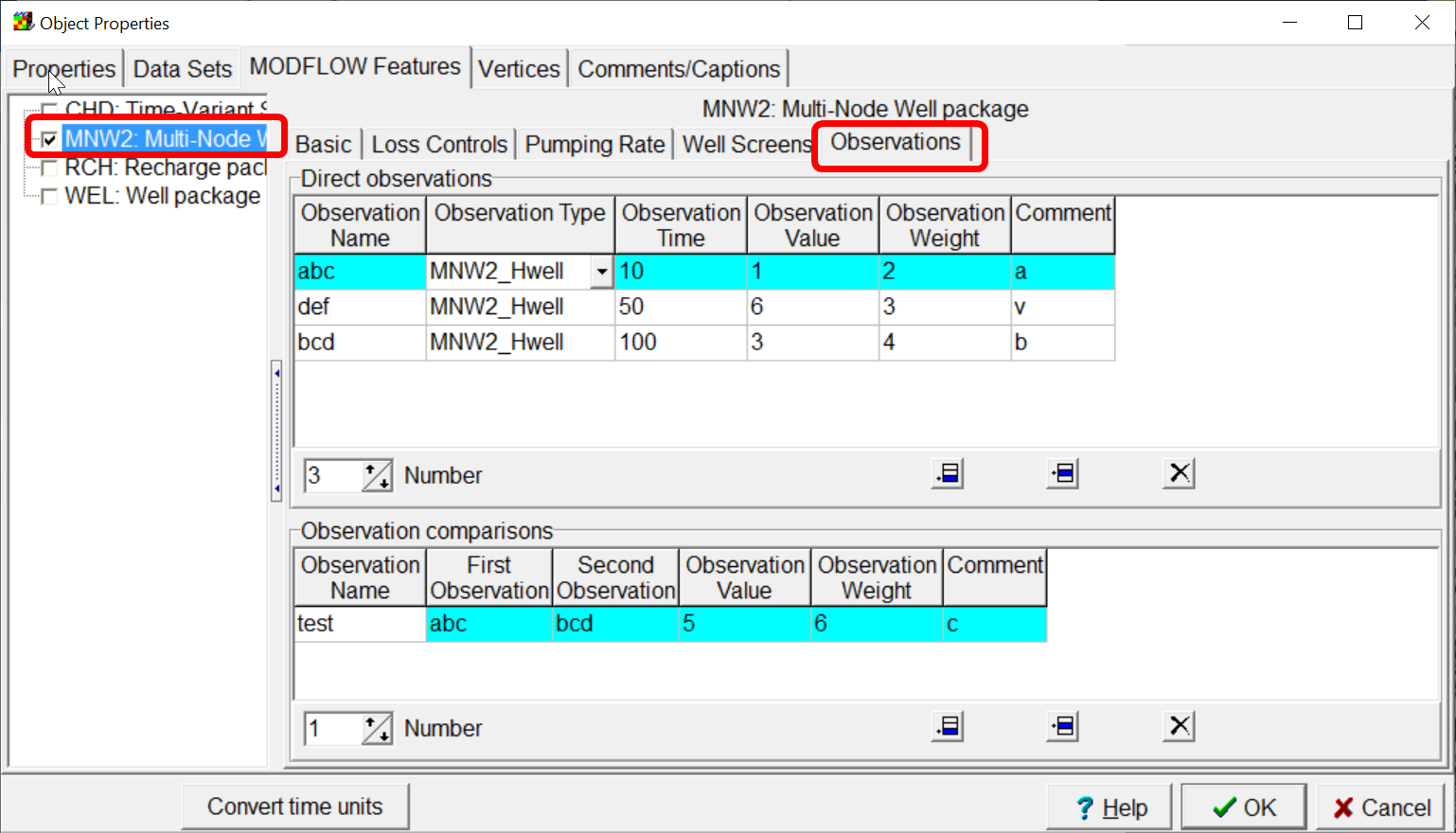
Methods for defining observations of head and flow through some boundary conditions are built into MODFLOW-2005 and other versions of MODFLOW based directly on it such as MODFLOW-NWT. This is done in the MODFLOW-2005 Observation Process. The output from the Observation Process packages consist of files that list the observation name, corresponding simulated values and a few other pieces of data for each simulated value. Instruction files to read these output files are simple to construct.

However, there are several packages in MODFLOW-2005 that produce output that could potentially be compared with observed values for the purposes of calibrating a model, but which do not produce the simple files generated in the Observation Process. These include the GAGE, MNWI, SUB, SWT, and SWI packages.

This beta version of ModelMuse allows the user to define observations at specific locations and times with those packages similarly to how observations are defined in ModelMuse with MODFLOW 6. The utility program ObsSeriesExtractor is used to extract values at specific locations and times from the MODFLOW output files. ObsSeriesExtractor can also create instruction files for PEST to use in reading the extracted values.

## MNWI Package with the MNW2 Package

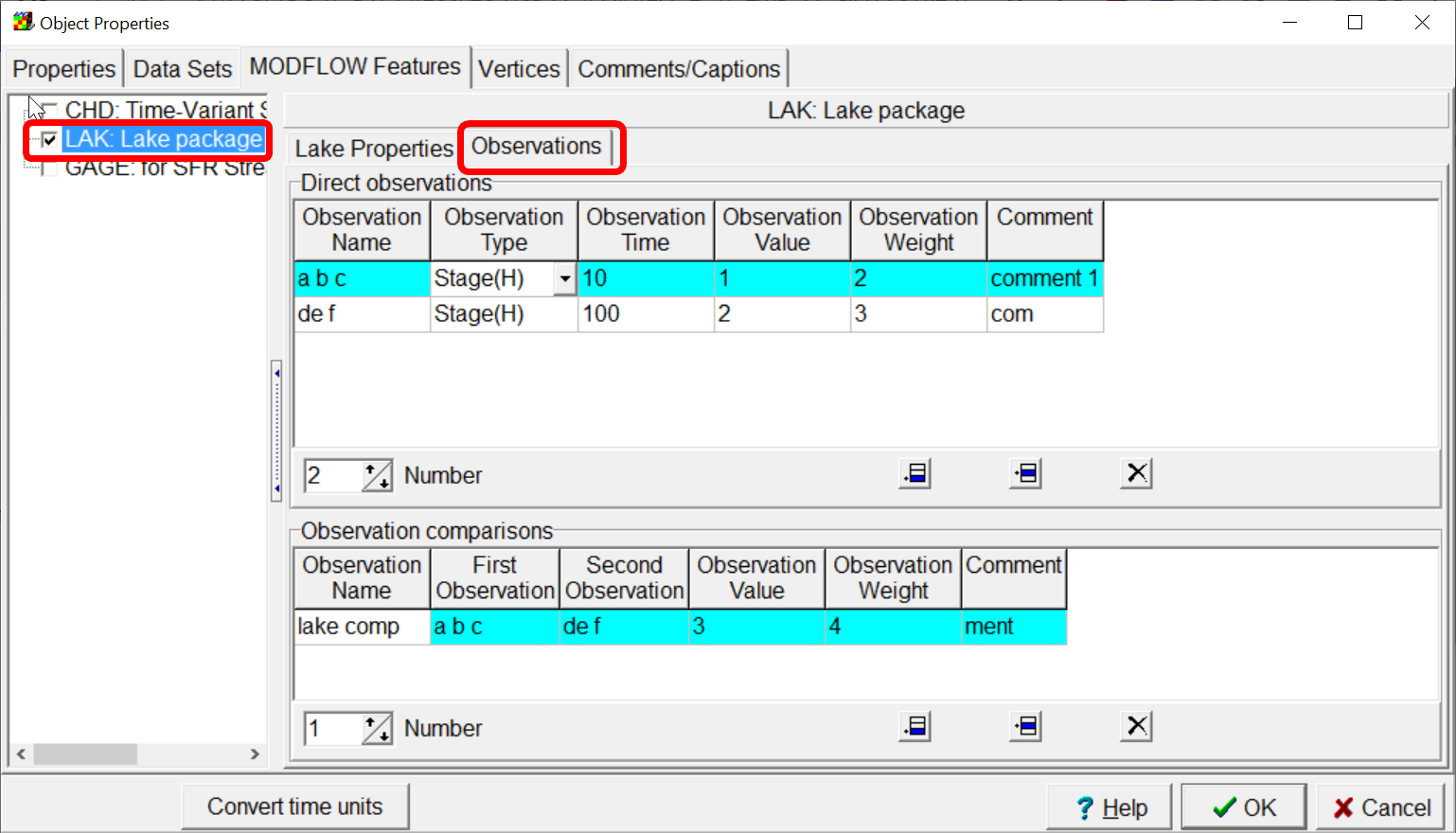
The MNWI package produces time-series data related to the MNW2 package. To define a calibration observation the MNWI package, the user edits an object that defines a well in the MNW2 package and switches to the Observations tab of the MNW2 pane on the Object Properties dialog box. As with observations in MODFLOW 6, the user defines observation names, types, times values and weights. The modeler can also define comparisons between two observations defined for the same MNW2 well. The allowed observation types are the ones that apply to an entire well namely, (1) the flow into the well, (2) the flow out of the well, (3) the net flow to or from the well, (4) the cumulative flow to or from the well and (5) the head in the well.



## GAGE Package with Lakes

The GAGE package produces time-series data for both the LAK and SFR packages. When defining an observation for calibration with the LAK package, the user edits an object that defines a lake and switches to the Observations tab of the Lake package pane. Just as before, the user defines observations names, types, times, values, and weights and can also define comparisons between to observations defined for the same lake. The following types of observations can be defined for lakes.

* Stage
* Lake volume
* Precipitation
* Evaporation
* Runoff
* Groundwater inflow
* Groundwater outflow
* Surface water inflow
* Surface water outflow
* Withdrawals
* Lake inflow
* Total lake conductance
* Change in head in a time step
* Change in lake volume in a time step
* Cumulative change in head
* Cumulative change in lake volume



## GAGE Package with SFR Streams

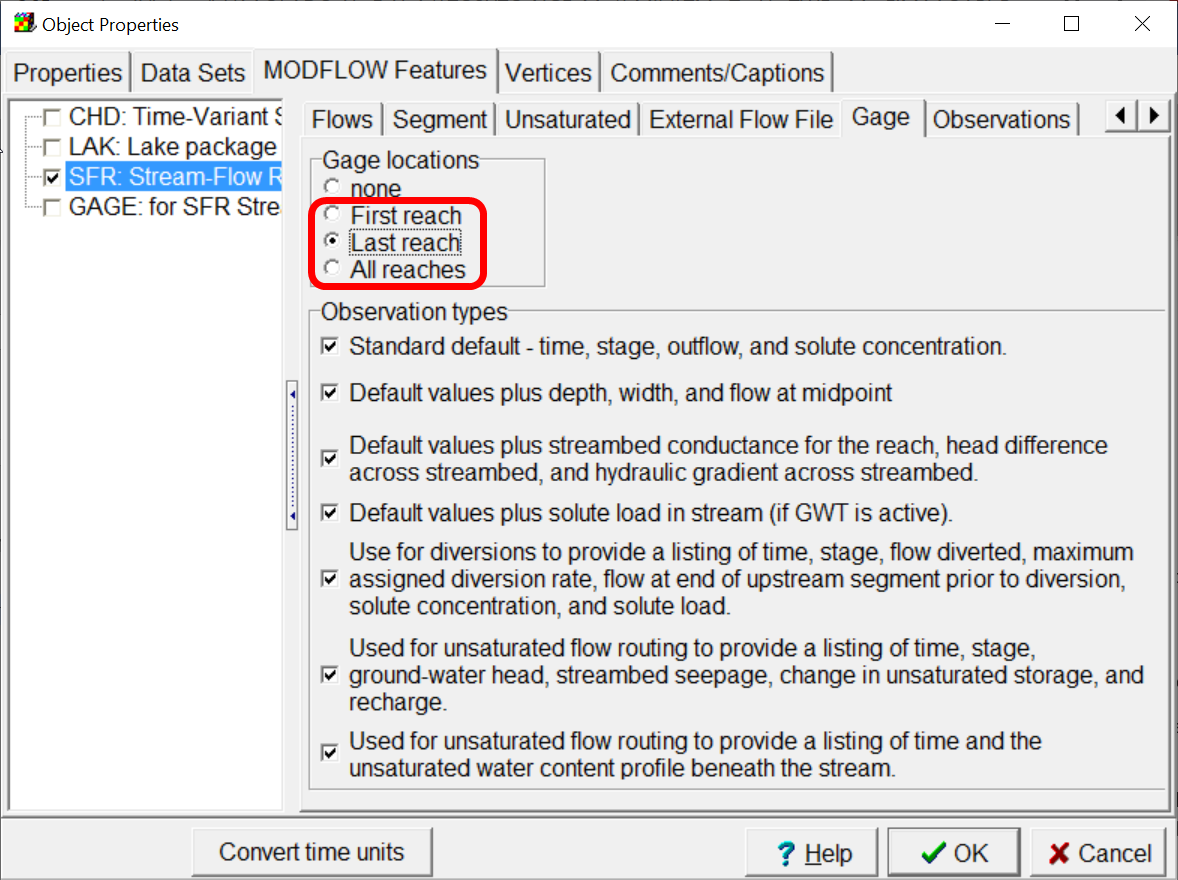
The gage package produces a variety of output for individual stream reaches. When used for calibration purposes, the following observation types can be used.

* Stage
* Stream flow
* Depth
* Width
* Midpoint flow
* Precipitation
* Evapotranspiration
* Runoff
* Conductance
* The difference in head between the stream and its connected groundwater model cell
* The hydraulic gradient across the streambed
* Groundwater flow

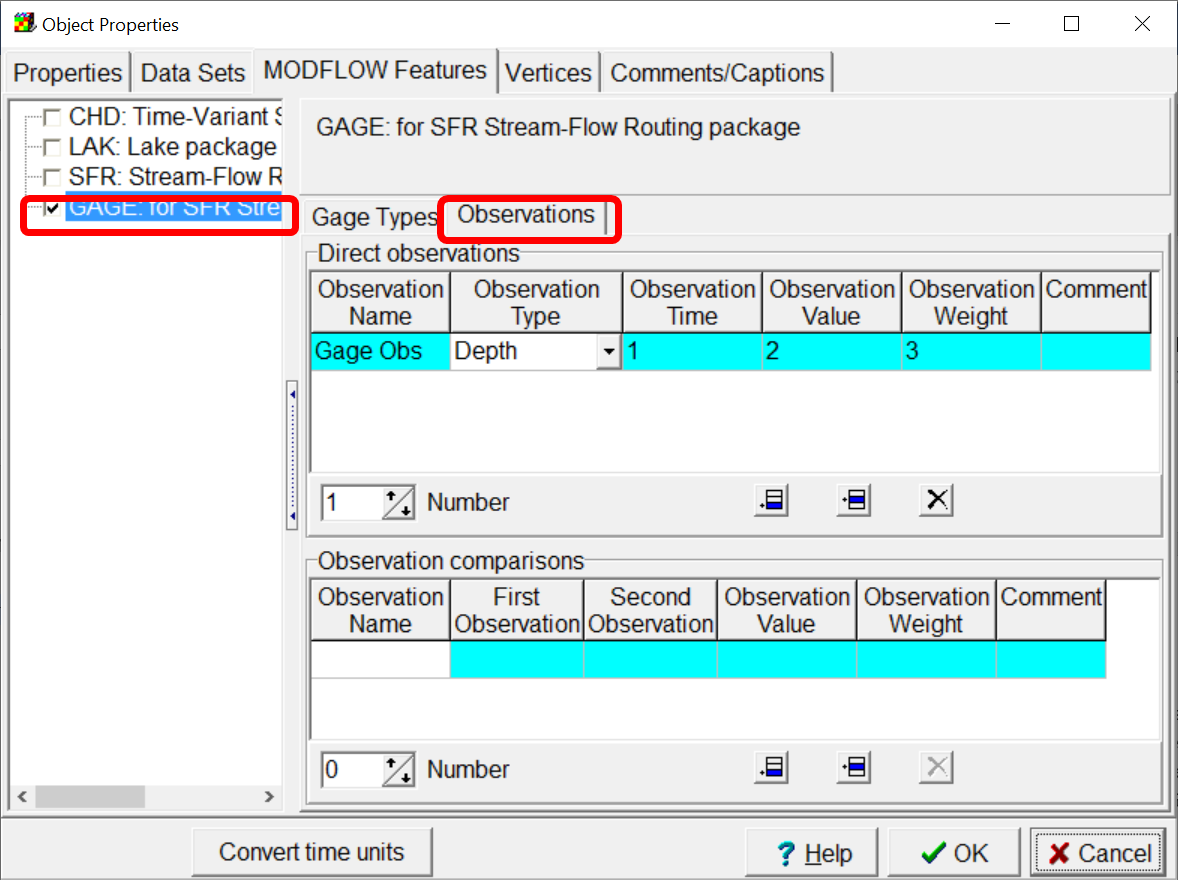
Of these, only groundwater flow makes sense when applied to all the reaches in a segment.

When defining an observation for a stream the user can either edit an object that defines a stream or use a point object in the same cell as a stream reach.

To define an observation for calibration with an object that defines a stream segment, the user edits the object in the Object Properties dialog box and selects either First reach, Last reach, or All reaches on the Gage tab of the SFR pane. Doing so will cause the Observations tab to appear.



On the observation tab, the user defines observations and observation comparisons in the same way as before. If “All reaches” is selected on the Gage tab, only “Groundwater flow” will be available as an observation type and the computed value of groundwater flow will be the sum of the groundwater flow through all the reaches in the segment. If first reach or last reach is selected, all the observation types will be available, but they will only apply to the first or last reach in the segment. To specify a calibration observation for a single reach that is neither the first not last reach in a segment, the user creates a point object in the same cell as the reach of interest and define the observation on the Observations tab of the Gage pane on the Object Properties dialog box.



## SUB Package

The Subsidence package has options to save arrays of data of various sorts at user-specified time steps. ModelMuse in conjunction with ObsSeriesExtractor allows the user to specify observed values at specific locations and times and extract simulated values for those locations and times. ModelMuse ensures that the appropriate arrays are saved for the time steps enclosing the observation times. ObsSeriesExtractor interpolates in time to the observation time for the cells surrounding the observation location. It uses a finite-element basis function for interpolating spatially to the observation location from the surrounding cell locations.

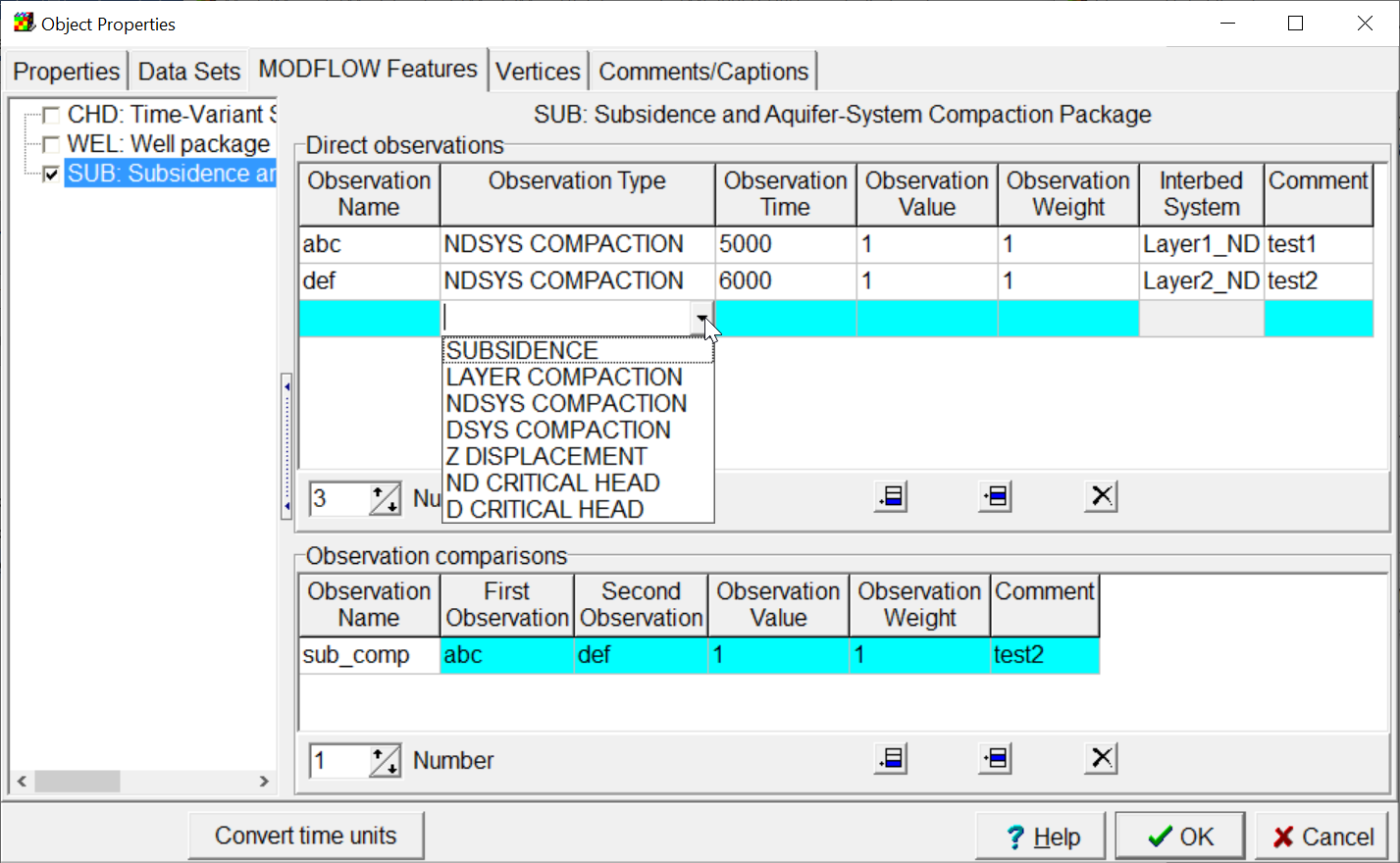
Subsidence observations are defined using point objects that intersect only a single layer. As with previous observation types, the user must specify an observation name, type, time, value, and weight. With certain types of observations, an interbed system must also be specified.

The observation types that are supported are identified by the same identifiers used in the Subsidence output files namely

* SUBSIDENCE
* LAYER COMPACTION
* NDSYS COMPACTON
* DSYS COMPACTON
* Z DISPLACEMENT
* ND CRITICAL HEAD
* D CRITICAL HEAD

For observations of NDSYS COMPACTION, the user must specify a no-delay interbed system. For observations of DSYS COMPACTION, the user must specify a delay interbed system.

The user can also specify comparisons between observations defined by the same object.

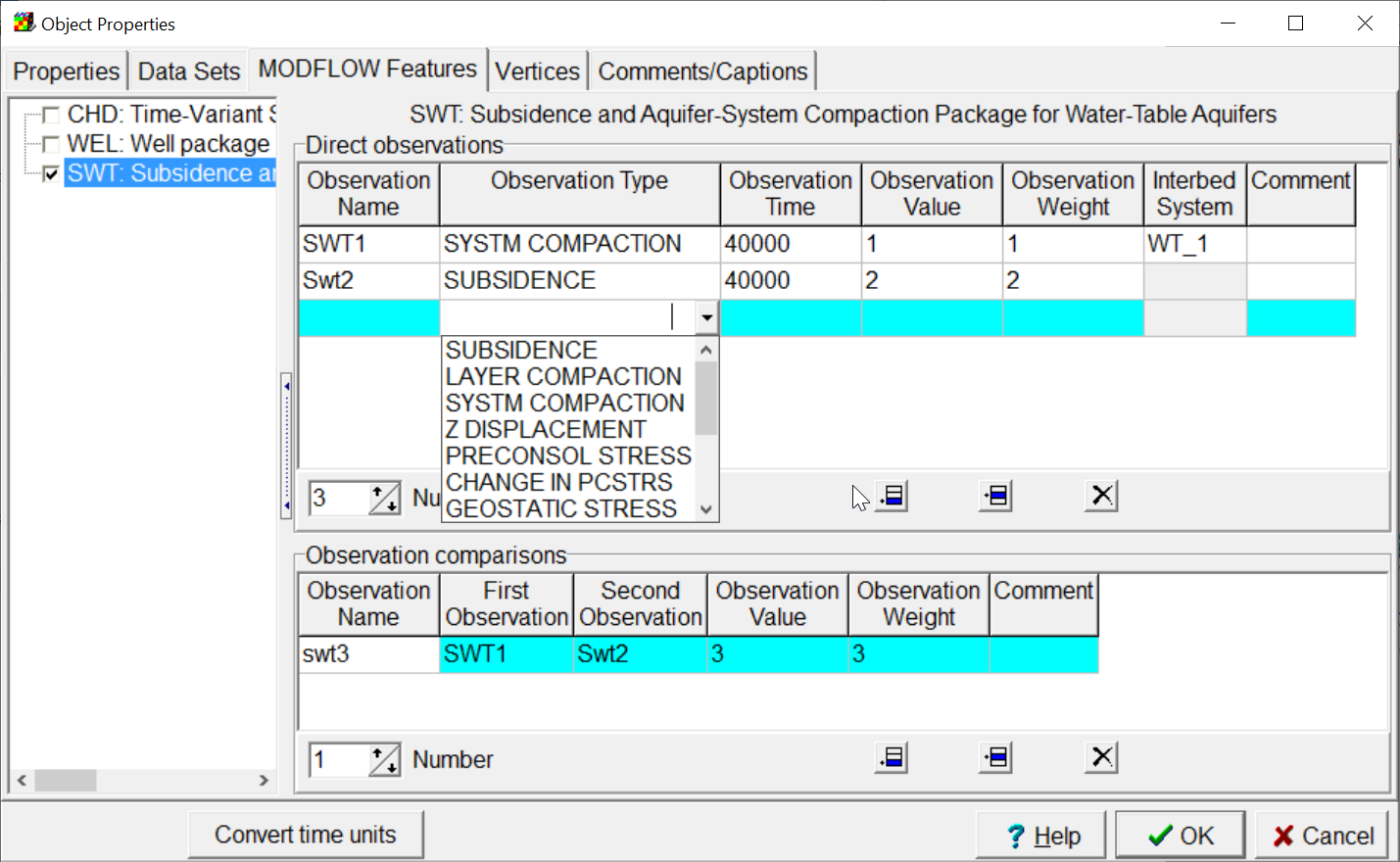


## SWT Package

With respect to how observations for calibration are defined in ModelMuse, the SWT package is similar to resembles the SUB package. The main difference is that the observation types differ from those in the SUB package. The observation types are

* SUBSIDENCE
* LAYER COMPACTION
* SYSTM COMPACTION
* Z DISPLACEMENT
* PRECONSOL STRESS
* CHANGE IN PCSTRS
* GEOSTATIC STRESS
* CHANGE IN G-STRS
* EFFECTIVE STRESS
* CHANGE IN E-STRS
* VOID RATIO
* THICKNESS
* CENTER ELEVATION

If the observation type is SYSTM COMPACTION, VOID RATIO, or THICKNESS, an interbed system must be specified.

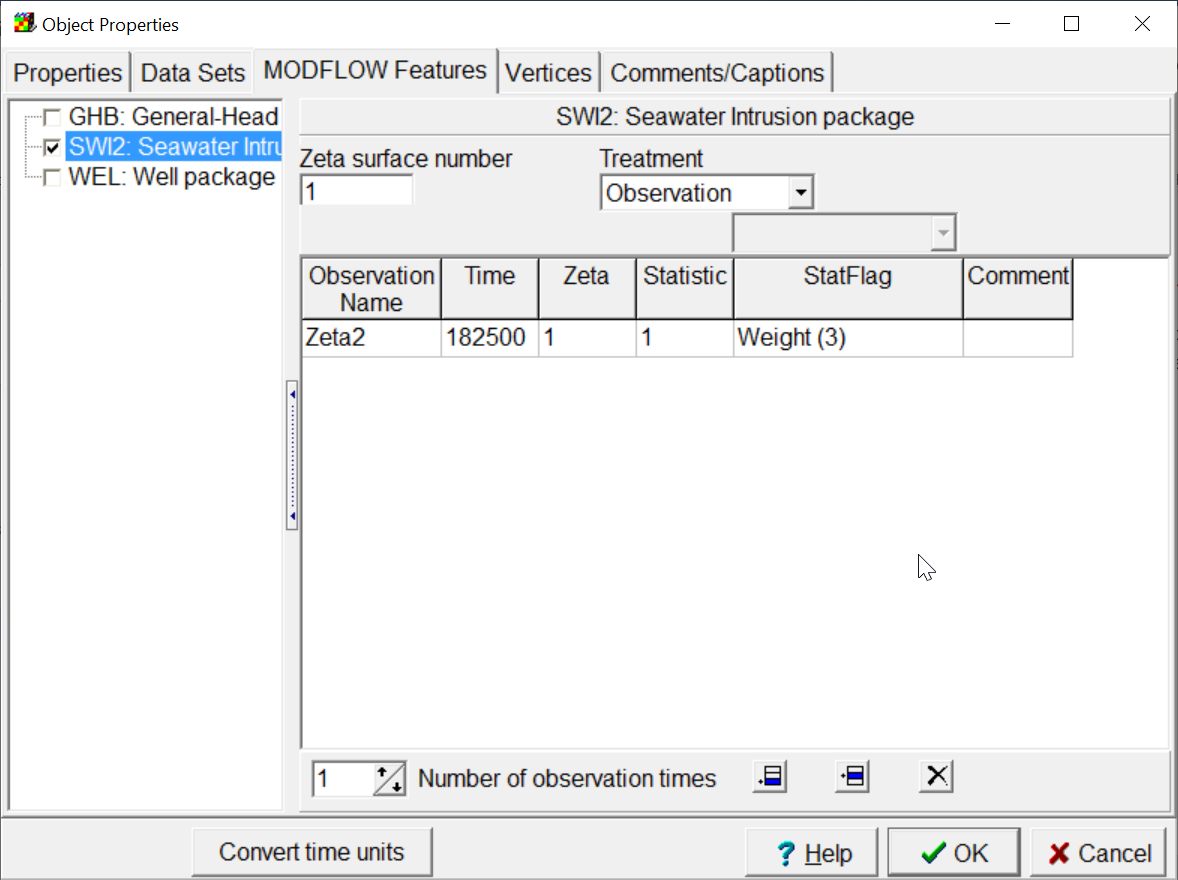


## SWI Package

The Seawater Intrusion (SWI) package has an option to save a time-series of ZETA values at specific cells where Zeta is the elevation of the active surface. ModelMuse in conjunction with ObsSeriesExtractor allows the user to specify observed values at specific locations and times and extract simulated values for those locations and times. The user defines SWI ZETA observations similarly to how head observations are defined. Because there is only one observation type, the user does not need to specify the observation type. The user specifies an observation name, time, observed ZETA value, Statistic, and StatFlag. StatFlag, determines whether the Statistic is a variation, standard deviation, coefficient of variation, weight, or square root of the weight. (All these different statistic types can be converted to weights for use in PEST.) The user must also specify a Zeta surface number.

ModelMuse will ensure that the SWI package generates a time series file for the cells surrounding the observation location. ObsSeriesExtractor will interpolate all of them in time to the observation time and will interpolate to the observation location using a finite-element basis function.

The user also specifies a treatment which determines whether the observations will be used for model calibration or for estimating error in model predictions.



## Observation Comparisons

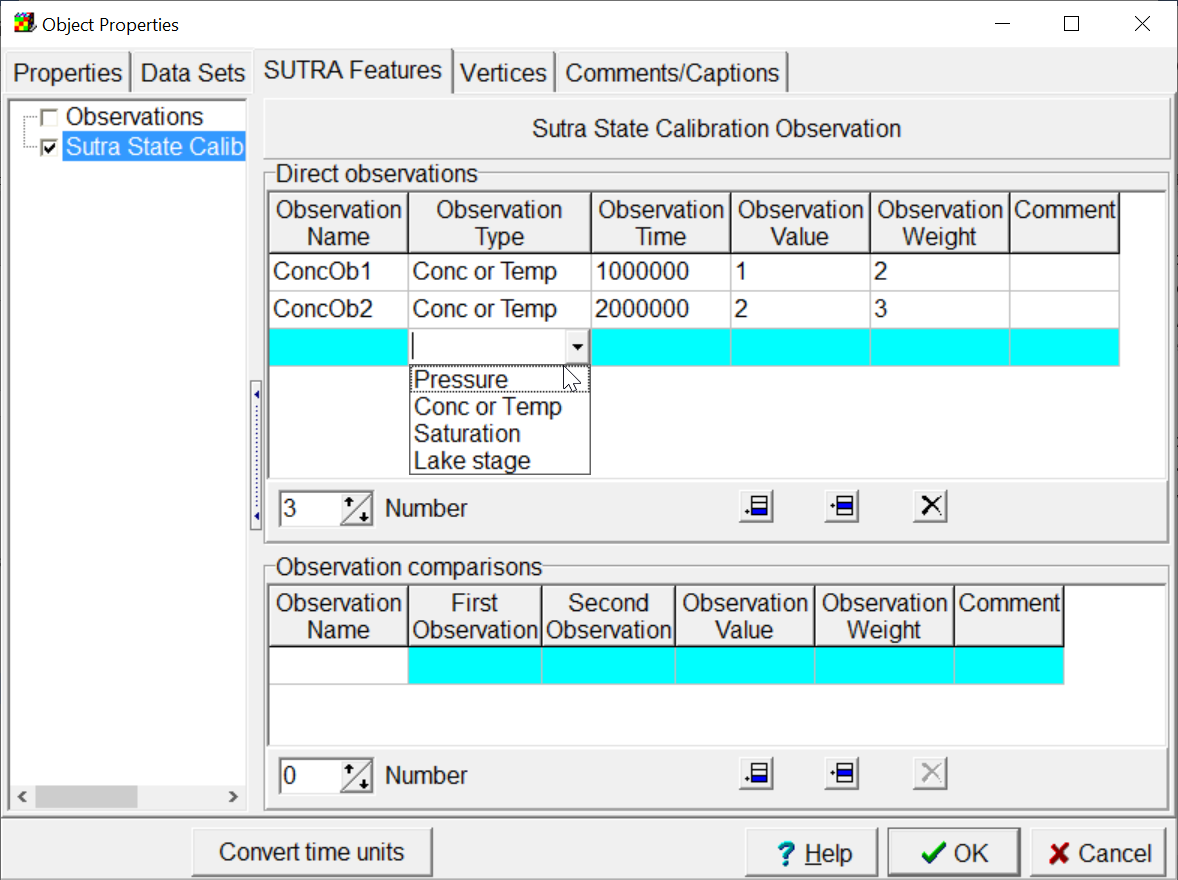
Observation comparisons can be defined for observations defined using different objects in the Observations Comparisons dialog box. This dialog box was described previously in the section on MODFLOW 6. With MODFLOW-2005, some observation types are not currently supported for observation comparisons. The unsupported types are head observations, flow observations, and SWI ZETA observations. Comparisons of these observation types may be supported in the future.

# SUTRA

Both observations of state and observations of flux through boundaries in SUTRA models can be defined using ModelMuse in conjunction with the utility program SutraObsExtractor. Four types of state observation can be defined: pressure, U (concentration or temperature), saturation, and lake stage. Lake stage observations can only be defined in SUTRA models in which lakes are active.

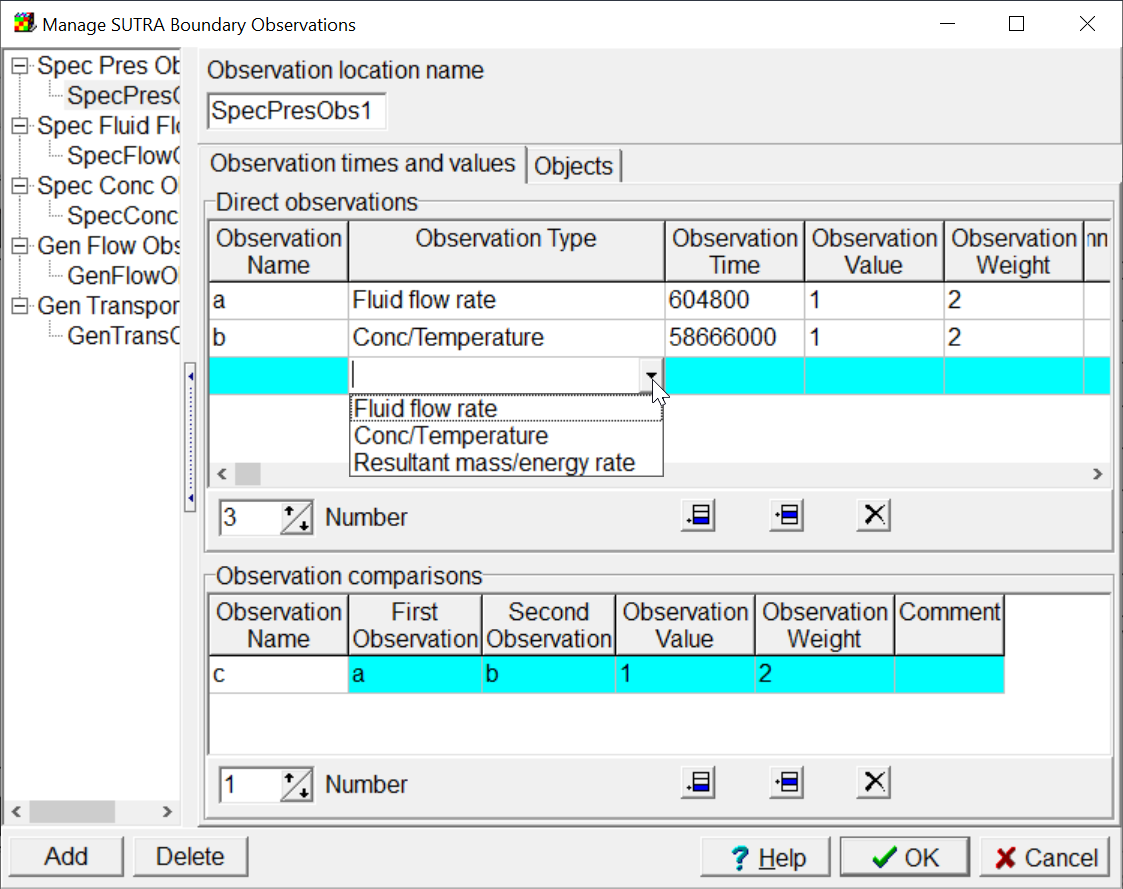
## SUTRA State Observations

Sutra state observations are defined using point objects. Users specify an observation name, type, time, value and weight. Observation comparisons can also be defined for observations defined using the same object as described previously in the section on MODFLOW 6.

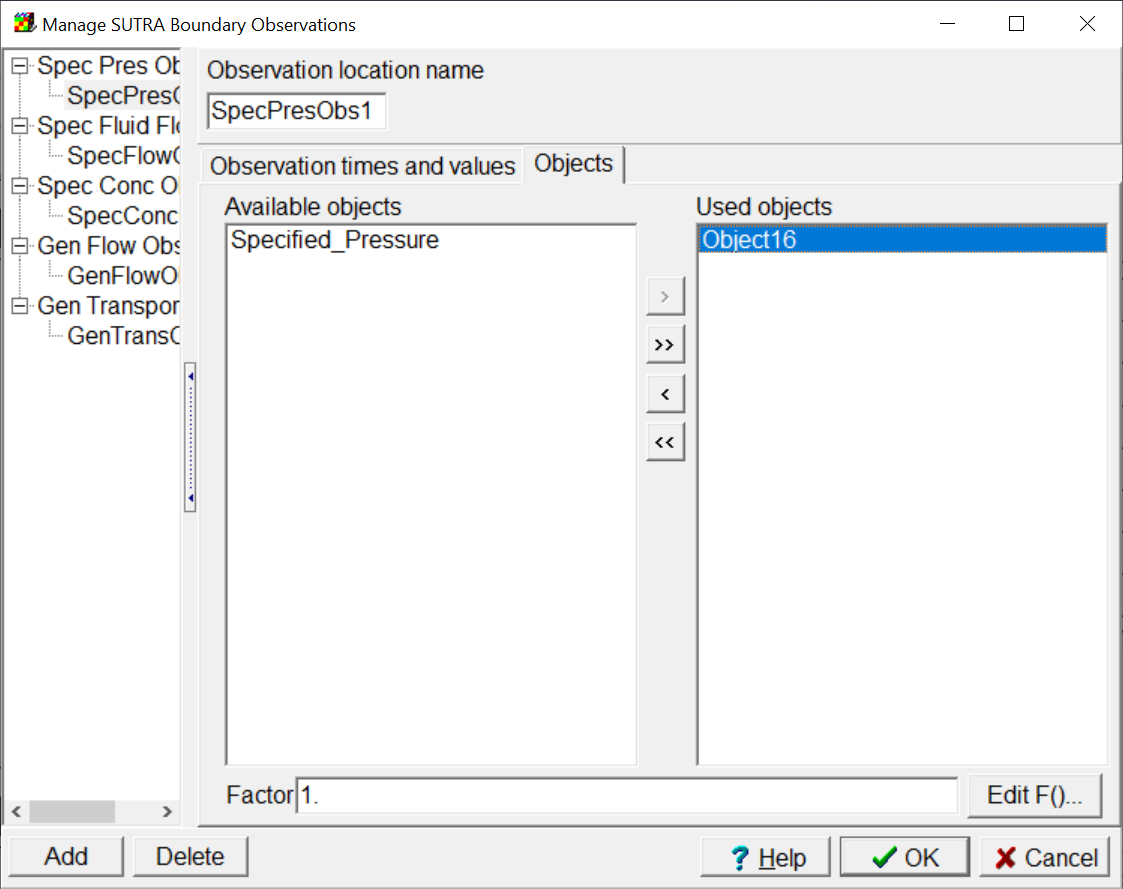


## SUTRA Flux Observations

Observations of flux through SUTRA boundaries are defined in the “Model|Manage SUTRA Boundary Observations” dialog box. This is similar in design to the “Manage Flow Observations” dialog box for MODFLOW-2005 models. On the left, there is a list of the boundary types for which flux observations can be defined. The user can define a new observation group by selecting a boundary type and clicking the Add button. Each observation group is a associated with one or more objects that defined boundaries of the selected type. For each boundary type, different observations types can be defined. The user specifies and observation name, type, time, value, and weight. Observation comparisons can be defined between observations within the same observation group.



On the Objects tab, the user selects one or more objects from the list of available objects and transfers it to the list of used objects. The used objects must define SUTRA boundaries of the appropriate type. The values derived from all the SUTRA boundary nodes defined by the selected object will be included as part of the simulated value. If it is desirable to exclude some of nodes from the simulated value, the Factor formula associated with an object can be used for that purpose. The formula will be evaluated at each node that is defined by the object. The formula should result in a real number between zero and one inclusive. That value will be multiplied by the value from the boundary node when calculating the final simulated value. Thus, if the formula evaluates to zero for a node, the value from that node will have no effect on the final simulated value.



## Observation Comparisons

Comparisons between SUTRA observations can be used for calibration as described previously in the section on MODFLOW 6.

# Parameter Substitution

In addition to the utility programs for extracting simulated values from the model output files, there is one additional utility program included in the beta version named EnhancedTemplateProcessor. This program performs operations like what PEST does when processing a template file to generate model input files. The main difference is that instead of only replacing a parameter name with its value, EnhancedTemplateProcessor can perform mathematical operations on the formulas defined within the template.

I had originally planned to use time-series files with MODFLOW 6 models and make the SFAC variable within those files be a parameter used for calibration. It turns out, this is impractical. Instead, ModelMuse will create templates for the boundary condition files in which a formula will be defined for replacing values for each boundary with the parameter value times a multiplier. (Other ways of using the capability are also possible.) For PEST to generate the model input files, the following sequence of operations will occur. PEST will process on ordinary template file to produce a PVAL file. Then EnhancedTemplateProcessor will use the PVAL file generated by PEST along with another template file to produce the model input file. Usage of EnhancedTemplateProcessor is described in a separate document. EnhancedTemplateProcessor does not limit the length of lines in the template or the complexity of the mathematical expression.

This Beta version of ModelMuse does not currently generate input for EnhancedTemplateProcessor. My plan for the next beta release is to support parameter substitution and EnhancedTemplateProcessor will be used in that release.