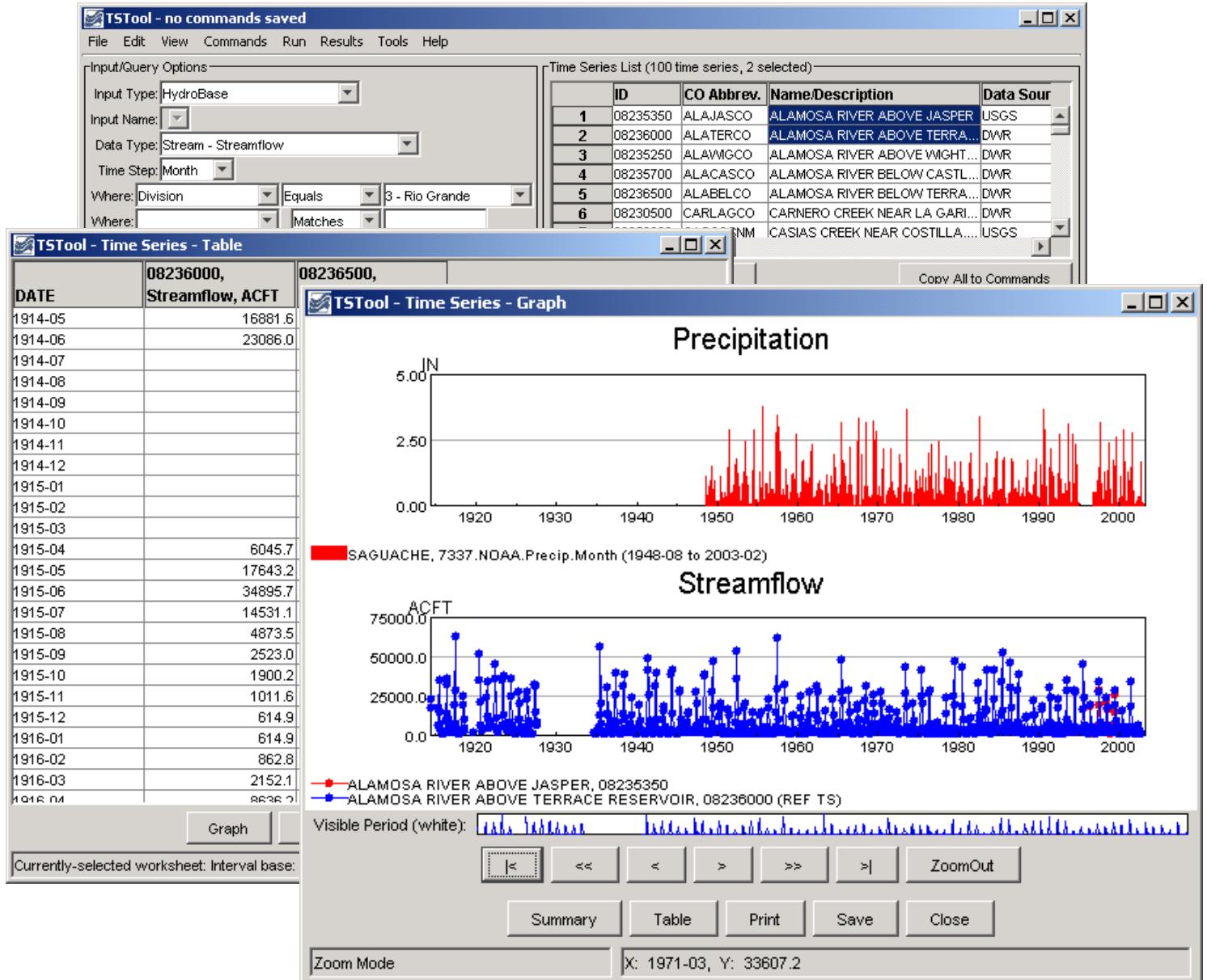


TSTool - Time Series Tool



**Colorado Department of Natural Resources
Colorado Water Conservation Board
Division of Water Resources**

Developed by:



Version 7.00.00, 2006-10-31

This page is intentionally blank.

This document is formatted for double-sided printing.

Table of Contents

Version 06.17.00, 2006-04-28, Color, Acrobat Distiller

1 Acknowledgements

2 Introduction

- 2.1 Time Series Objects and Identifiers
- 2.2 Date/Time Conventions
- 2.3 Time Scale for Time Series Data
- 2.4 Time Series Commands and Processing Sequence
- 2.5 Using Time Series Aliases

3 Getting Started

- 3.1 Starting TSTool
- 3.2 Select HydroBase Dialog
- 3.3 Main Interface
- 3.4 File Menu – Main Input and Output Control
- 3.5 Edit Menu – Editing Commands
- 3.6 View Menu – Display Map Interface
- 3.7 Commands Menu
- 3.8 Run Menu – Running Commands
- 3.9 Results Menu – Display Time Series
- 3.10 Tools Menu
- 3.11 Help Menu

4 Commands

- 4.1 Create Time Series
- 4.2 Converting Time Series Identifier to Read Command
- 4.3 Read Time Series
- 4.4 Fill Time Series Data
- 4.5 Set Time Series Data
- 4.6 Manipulate Time Series
- 4.7 Analyze Time Series
- 4.8 Models
- 4.9 Output Time Series
- 4.10 Commands for Specific Input Types
- 4.11 General Commands – Global Settings

5 Tools

- 5.1 Report Tools

6 Examples of Use

- 6.1 General Examples
- 6.2 Model Data Processing Examples
- 6.3 Time Series Trace Examples
- 6.4 Time Series Product Examples

7 Using the Map

- 7.1 Time Series and Map Layer Relationships
- 7.2 Opening a Map
- 7.3 Using Time Series to Select Locations on the Map
- 7.4 Using Locations on the Map to Select Time Series
- 7.5 Spatial Analysis Commands

8 Troubleshooting

- 8.1 Obsolete Commands

[Command Glossary](#)

[Command Reference](#)

Time Series Identifier (TSID) reference, listed alphabetically by input type

Command reference, listed alphabetically by command name

[Appendix – TSTool Installation and Configuration](#)

[Appendix – Release Notes](#)

[Appendix – Colorado SMS Input Type](#)

[Appendix – DateValue Input Type](#)

[Appendix – HydroBase \(CDSS\) Input Type](#)

[Appendix – RiverWare Input Type](#)

[Appendix – StateCU Input Type](#)

[Appendix – StateMod Input Type](#)

[Appendix – StateModB Input Type](#)

[Appendix – USGSNWIS Input Type](#)

[Appendix – TSView Time Series Viewing Tools](#)

[Appendix – GeoView Mapping Tools](#)

Acknowledgements

Version 06.08.02, 2004-08-05, Color, Acrobat Distiller

TSTool has been developed by Riverside Technology, inc. (RTi). Early development work was funded by the State of Colorado, Water Conservation Board, under the Colorado River Decision Support System (CRDSS). Later development occurred as part of CRDSS maintenance, the State of Colorado's Rio Grande Decision Support System (RGDSS) and the State of Colorado's South Platte Decision Support System (SPDSS), under the umbrella CDSS effort (Colorado's Decision Support Systems).

Enhancements to the CDSS version have also been made by RTi in order to support a wider variety of uses and data formats. Enhancements will continue to be made by RTi while offering backward compatibility as much as possible.

Users are encouraged to provide general feedback to RTi using the email address: support@riverside.com and to provide feedback specific to CDSS functionality (e.g., HydroBase, StateMod, StateModB, and StateCU input types) using the email address: cdss@state.co.us. RTi's web site is <http://www.riverside.com>. The CDSS web site is <http://cdss.state.co.us>.

The MOVE.1 and MOVE.2 procedures description was taken from:

Hirsch, R. M., 1982, A Comparison of Four Streamflow Record Extension Techniques: Water Resources Research, Vol. 18, No. 4, pages 1081-1088.

Additional information can be found in:

Guidelines for Determining Flood Flow Frequency, Bulletin 17B, USGS.

This page is intentionally blank.

Introduction

Version 06.16.01, 2006-03-03, Color, Acrobat Distiller

TSTool displays and analyzes time series data, either interactively or in batch mode. The TSTool GUI (Graphical User Interface) allows time series to be easily viewed, manipulated, and analyzed. Time series can be read from and written to a variety of input types. TSTool can be thought of as a time series calculator. Although a graphical user interface is provided, the heart of TSTool's analytical features is an interpreter that processes a command language. Depending on your needs, you may avoid the command language completely or use it extensively. This flexibility makes TSTool useful for basic data viewing and advanced analysis. The documentation is divided into the following main sections.

Chapter 2 – Introduction provides background information on time series concepts and how TSTool processes time series.

Chapter 3 – Getting Started provides an overview of TSTool interface features.

Chapter 4 – Commands provides a summary of time series processing commands.

Chapter 5 – Tools provides information about analysis tools.

Chapter 6 – Examples of Use provides examples of how TSTool is commonly used.

Chapter 7 – Using the Map provides information about using the map interface to link time series with spatial data.

Chapter 8 – Troubleshooting provides troubleshooting information, including a list of obsolete commands.

The **Commands Reference** provides a complete commands reference, with commands listed in alphabetical order. Because some commands are used in more than one situation, this allows the commands to be fully documented once, and referred to as needed.

The **Installation and Configuration** appendix provides information about installing and configuring TSTool.

The **Release Notes** appendix summarizes TSTool changes over time.

Several appendices provide information about supported input types (appendices are inserted as additional input types are added).

The **TSView Time Series Viewing Tools** appendix provides a general reference for time series viewing features. These features are used throughout TSTool and other software developed by RTi.

The **GeoView Mapping Tools** appendix provides a general reference for the GeoView map interface. The mapping interface is being phased in and is used by other software developed by RTI.

2.1 Time Series Objects and Identifiers

TSTool considers time series as objects that are queried, manipulated, viewed and output. A time series is defined as a series of date/time versus data pairs. Data generally consist of floating point values; however, time series may contain other data (e.g., data quality flags). TSTool treats time series as either regular interval (equal spacing of date/time) or irregular interval (e.g., infrequent measurements). Regular time series lend themselves to simpler storage and faster processing because date/time information only needs to be stored for the endpoints.

TSTool defines each time series as having an interval base and multiplier (e.g., 1Month, 24Hour). In many cases, the multiplier is 1 and is not shown in output (e.g., Month rather than 1Month is shown). In addition to a period of record, interval, and data values, time series have attributes that include:

- units (e.g., CFS)
- data type (e.g., Streamflow)
- data limits (the maximum, minimum, etc.)
- description (generally a station or structure name)
- missing data value (used internally to mark missing data and trigger data filling, often -999 or in some cases NaN [Not a Number])
- comments (often station comments, if available)
- genesis history (a list of comments about how the time series was created and manipulated)

To manage time series, TSTool associates each time series with an identifier that uses the notation:

`Location.Source.Type.Interval.Scenario [Seq]~InputType~InputName`

The first five parts (`Location.Source.Type.Interval.Scenario`) are used to identify time series, similar to a variable name. The optional sequence number `[Seq]` is used in cases where multiple time series traces may be available, with all other identifier information being equal (e.g., for simulations where multiple versions of input are used or for cases when a historical time series is cut into annual traces). Typically the sequence number is a four-digit year corresponding to the data input year. The last two parts (`InputType~InputName`) are used indicate input information so that a time series can be located in a file or database, and also simplify the use of input types where more than one time series can be stored in the input (e.g., multiple time series in a file or database). The input information was introduced starting with TSTool version 05.04.00. A summary of input types that are currently supported or under development is listed in the following table (see the input type appendices for more information about how time series identifiers are formatted for specific input types). Features for these input types may or may not be available, depending on the TSTool configuration (see the **TSTool Installation and Configuration** appendix). The main constraint on whether an input type is considered for TSTool is that the input type should be a standard that is used in a relatively wide audience. By supporting general formats, TSTool can support the largest group of users and provide the most useful general features.

Input Types for TSTool

Input Type	Description
DateValue	General delimited date/value file with extended header information, able to store one or more time series.
DIADvisor	DIADvisor real-time environmental monitoring software, from OneRain, Inc.
HydroBase	State of Colorado database.
MexicoCSMN	Hydrometeorological database for Mexico Coordinación Servicio Meteorológico Nacional (CSMN, similar to US National Weather Service).
MODSIM	Colorado State University MODSIM model.
NWSCard	National Weather Service River Forecast System (NWSRFS) card file format for hourly data.
NWSRFS_ESPTraceEnsemble	NWSRFS Ensemble Streamflow Prediction binary files.
NWSRFS_FS5Files	NWSRFS binary FS5Files preprocessor and processed database.
RiversideDB	Riverside Technology, inc. database used for real-time and historic time series data (e.g., with RiverTrak® software).
RiverWare	University of Colorado Center for Advanced Decision Support for Water and Environmental Systems RiverWare model data format.
SHEF	Standard Hydrologic Exchange Format, a common data format used by United States government agencies.
StateCU	State of Colorado consumptive use model time series and report formats.
StateMod	State of Colorado StateMod model time series file format.
StateModB	State of Colorado StateMod model output binary file.
USGS NWIS	United States Geological Survey National Water Information System format

An example of a time series identifier for a monthly streamflow time series in HydroBase is:

```
09010500.USGS.Streamflow.Month~HydroBase
```

The same time series for a USGS NWIS input source might be identified using:

```
09010500.USGS.Streamflow.Month~USGSNWIS~C:\temp\09010500.txt
```

In this example, the optional scenario (fifth part) and sequence number are not used. This identifier string can be saved in a commands file or time series product description file, which can be processed again later. The identifier string allows TSTool to determine how to re-query the time series. The time series identifier is useful for managing time series. The TSTool GUI typically handles creation of all time series identifiers; however, identifiers can be created with an editor once the format is familiar.

Because time series identifiers are somewhat cumbersome to work with, TSTool allows a time series *alias* to be used instead. For example, the following command illustrates how a HydroBase time series can be read and associated with an alias:

```
TS X = readTimeSeries("09010500.USGS.Streamflow.Month~HydroBase")
```

This allows the time series to be referred to as X during further processing (e.g., when manipulated with commands). Whether full identifiers or aliases are used, the overall identifier must be unique during processing to guarantee that time series commands are processed as desired (duplicate aliases and identifiers can be present but the first one found will be used - see **Section 2.4 Time Series Commands and Processing Sequence** for an example). TSTool ignores case when comparing identifiers, aliases, and other commands, although it is good practice to be consistent.

When editing commands, TSTool does not normally show the input type and input name parts of the identifier because this information is most appropriate for read commands. There are cases where two time series identifiers will be the same except for the input type and name. In these cases, an alias should be assigned when reading the time series and the alias used in later commands. If for some reason an alias cannot be used, the input type and name may need to be manually added because the command editors do not display by default.

2.2 Date/Time Conventions

TSTool uses date/time information in several ways:

1. data values in time series are associated with a date/time and the precision of all date/time information should be consistent within the time series, as discussed below,
2. the data interval indicates the time spacing between data points and is represented as a multiplier (optional if 1) and a base (e.g., Day, 24Hour),
3. the period of a time series is defined by start and end date/time values, using an appropriate precision,
4. an analysis period may be used to indicate when data processing should occur,
5. output is typically formatted for calendar year (January to December), water year (October to November), or irrigation year (November to October) – calendar year is the default but can be changed in some commands and output.

A date/time has a precision. For example, 2002-02 has a monthly precision and 2002-02-01 has a daily precision. Each date/time object knows its precision and “extra” date/time information is set to reasonable defaults (e.g., hour, minute, and second for a monthly precision date/time are set to zero and the day is set to 1). The date/time precision is important because TSTool uses the date/time objects to iterate through data, to compare dates, and to calculate a plotting position for graphs. Specifying date/time information with incorrect precision may cause inconsistent behavior.

The TSTool documentation and user interface typically use ISO 8601 International Standard formats for date/time information. For example, dates are represented using YYYY-MM-DD and times are represented using hh:mm:ss. A combined date/time is represented as YYYY-MM-DD hh:mm:ss. In order to support common use, TSTool also attempts to handle date/time information that uses United States and other date formats. In such cases, the length of the date/time string and the position of special characters are used to make a reasonable estimate of the format. Using ambiguous formats (e.g., two-digit years that may be confused with months) may cause errors in processing. Adhering to the ISO 8601 standard formats will result in the fewest number of errors. However, it is understood that various input types use other date/time formats.

Plotting positions are computed by converting dates to floating point values, where the whole number is the year, and the fraction is the fractional part of the year, considering the precision. The floating-point

date is then interpolated to the screen pixels, as integers. In most cases, the high-precision date/time parts are irrelevant because they default to zero. However, in some cases the precision can impact plots significantly. For example, when plotting daily and monthly data on the same graph, the monthly data will be plotted ignoring the day whereas the daily values correspond days 1 to 31. The ability to plot monthly data mid-month or end-of-month has not been implemented. The **TSView Time Series Viewing Tools** appendix provides examples of plots.

The date/time precision is very important when performing an analysis or converting between time series file formats. For example, a file may contain 6Hour data using a maximum hour of 24 (e.g., 6, 12, 18, 24). When reading this data, TSTool will convert the hour 24 values to hour 0 of the next day. Consequently, the hour and day of the original data will seemingly be shifted, even though the data are actually consistent. This shift may also be perceived when converting from hourly data to daily data because the hour can have a value of 0 to 23, whereas days in the month start with 1. The perceived shift is purely an artifact of time values having a minimum value of zero.

2.3 Time Scale for Time Series Data

The time scale for time series data gives an indication of how the data value were measured or computed. The time scale is generally determined from the data type (or the data type and interval) and can be one of the following (the abbreviations are often used in software choices):

- Instantaneous (INST): The data value represents the data observed at the time associated with the reading (e.g., instantaneous temperature, streamflow, or snow depth). Instantaneous data may be of irregular or regular interval, depending on the data collection system. If irregular, the precision of the date/time associated with the reading may vary (e.g., automated collection systems may have very precise times whereas infrequently recorded field measurements may only be recorded to the nearest day).
- Accumulated (ACCM): The data value represents the accumulation of the observed data over the preceding interval. The date/time associated with the data value corresponds to the end of the interval. For example, precipitation (rain or snow recorded as melt) is often recorded as an accumulation over some interval. Accumulated values are typically available as a regular time series, although this is not a requirement (e.g., precipitation might be accumulated between times that a rain gage is read and emptied).
- Mean (MEAN): The data value represents the mean value of observations during the preceding interval. The date/time associated with the data value corresponds to the end of the interval. The mean includes values after the previous timestamp and including the current timestamp. The computation of mean values may be different depending on whether the original data are irregular or regular. For example, if the original data are regular interval, then equal weight may be given to each value when computing the mean (a simple mean). If the original data are irregular interval, then the weight given to each irregular value may depend on the amount of time that a value was observed (a time-weighted mean, not a simple mean).

Without having specific information about the time scale for data, TSTool assumes that all data are instantaneous for displays. If time series are graphed using bars, an option is given to display the bar to the left, centered on, or to the right of the date/time. If time series are graphed using lines or points, the data values are drawn at the date/time corresponding to data values. This may not be appropriate for the time scale of the data. In most cases, this default is adequate for displays. Graphing data of different time scales together does result in visual inconsistencies. These issues are being evaluated and options may be implemented in future releases of the software. In particular, an effort to automatically determine the time scale from the data type and interval is being evaluated. This can be difficult given that data types

are not consistent between input types and time scale may be difficult to determine when reading time series. Refer to the input type appendices for information about time scale.

The time scale is particularly important when changing the time interval of data. For example, conversion of instantaneous data to mean involves an averaging process. Conversion of instantaneous data to accumulated data involves summing the original data. Commands that change interval either operate only on data of a certain time scale or require that the time scale be specified to control the conversion. Refer to the command documentation for specific requirements.

2.4 Time Series Commands and Processing Sequence

Although TSTool can be run in batch mode (see **Chapter 3 – Getting Started**), you should be able to perform all time series viewing and manipulation within the GUI. Commands are used to read, manipulate, and output time series. Commands are processed sequentially from the first to the last commands using the steps described below. This section describes in detail the processing sequence. See the examples in **Chapter 6 – Examples of Use** for illustrations of the processing sequence.

Note that older versions of TSTool (before version 5.xx.xx) did not allow multi-step manipulation and therefore time series were read and manipulated in one step. This convention had limitations and has been changed to allow multi-step operations on time series, allowing more options for filling and manipulation. Old command files are supported as much as possible but some updates to old command files may be required.

TSTool commands fall into three main categories:

1. time series identifiers and aliases (see section **2.1 – Time Series Objects and Identifiers**), which are equivalent to time series “read” commands (where the identifier input type is used to determine which read command to use),
2. general commands, which are used to set properties like the period for output, and,
3. time series commands, which are used to read and manipulate time series and output results.

Commands are processed sequentially and can be repeated as necessary. A typical user starts learning TSTool by performing simple queries and displaying results while gradually utilizing more commands. The current software uses command syntax as follows:

```
command (param1=value1,param2="value",...)
```

Values that may contain spaces or commas are normally surrounded by double quotes. This notation is useful for the following reasons:

1. The parameter names are included in the command, in order to make the command more readable.
2. Because the parameter name is included, the parameters can generally be in any order. The command editor dialogs will enforce a default order.
3. Parameters that have default values can be omitted from the parameter list, shortening commands.
4. New parameters can be added over time, without requiring parameter order to change in existing commands.

The above notation is being used for new commands and older commands are being updated to the new syntax as time allows. Command editor dialogs will update old commands to the new syntax and the

processing code will recognize old and new command syntax. The **Command Reference** illustrates the current command syntax.

The following sequence occurs when processing commands:

1. **Parse the command.** A time series identifier or command is parsed to determine how to execute the command. Example commands are shown below. If the command is a general command, the action is taken and a new command is read in step 1 (general commands can be specified multiple times to change properties throughout a run). If the command results in reading or creating a time series, steps 2 - 4 are executed. If a command is a time series manipulation command, step 4 is executed.

```
# Example commands
08235350.USGS.Streamflow.Month~HydroBase
08236000.USGS.Streamflow.Month~HydroBase
add(08235350.USGS.Streamflow.Month, IgnoreMissing, 08236000.DWR.Streamflow.Month)
08235350.USGS.Streamflow.Month~HydroBase
```

2. **Read Time Series.** TSTool recognizes that certain commands should read a new time series and will perform the appropriate action. For example, in the above example, the time series identifier 08235350.USGS.Streamflow.Month~HydroBase indicates that the corresponding time series should be read from a HydroBase database (and since the full identifier is specified, no alias is assigned). The identifier input type is used to determine how to read the time series. Unless the `setQueryPeriod()` command has been used, **the entire time series period is read in this step** because data filling steps may require the full period (e.g., to determine regression relationships or long-term monthly average).

Commands that do not cause a time series to be read (but instead to be manipulated) are described in step 4.

If the input type, and if needed, input name, are specified in the identifier, they are only used in the initial read. Additional manipulation commands only use the first five parts of the identifier or the time series alias to identify the time series. If the same time series needs to be read from two input types (e.g., to compare whether a time series was properly loaded into a database from a file), use a different time series alias for each time series to uniquely identify each time series.

At the end of this step, a new time series will exist in memory.

3. **Compute Data Limits.** The time series data limits are computed because they may be needed later for filling. This information includes the long-term monthly averages. These limits are referred to as the original data limits.
4. **Access and Manipulate Time Series.** Commands that manipulate time series (fill, add, etc.) do not read the time series or make another copy. Instead, time series that are in memory are located and manipulated. The following example illustrates how the time series identified by 08235350.USGS.Streamflow.Month has its data values modified by adding the data from the time series identified by 08236000.USGS.Streamflow.Month.

```
# Example commands
08235350.USGS.Streamflow.Month~HydroBase
08236000.USGS.Streamflow.Month~HydroBase
add(08235350.USGS.Streamflow.Month, IgnoreMissing, 08236000.DWR.Streamflow.Month)
08235350.USGS.Streamflow.Month~HydroBase
```

To locate a time series so that it can be modified, TSTool first checks the alias of known time series (those that have been defined in previous commands) against the current time series of interest (08235350.USGS.Streamflow.Month, the first argument of the add() command), assuming that this string is an alias. If the alias is not found, it checks the full identifier of known time series against the current time series of interest. In this example, time series 08235350.USGS.Streamflow.Month was read in the first step and is therefore found as a match for the identifier. Similarly, the second time series in the command (08236000.USGS.Streamflow.Month) is found and is used to process the command, resulting in a modification of the first time series. **Sequential manipulations of the same time series can occur (e.g., fill by one method, then fill by another).**

To locate time series in memory, TSTool looks through the list of time series, searching backwards. In this way, it is possible to use the same identifier more than once in a command file while allowing localized processing of each time series. In the above example, the time series identified by 08235350.USGS.Streamflow.Month~HydroBase is read twice, once to be acted on by the add() command, and once with no manipulation (e.g., to compare the "before" and "after").

During processing, extra time series can accumulate and will be present during output steps. There are two ways to indicate that a time series should only be used temporarily:

- Use the free() command to free time series that are no longer needed. This removes the time series from memory. See also the deselectTimeSeries() and selectTimeSeries() commands.
- Use the TEMPTS notation in commands that support it. For example, the fillRegression() command requires a dependent (to be filled) and independent (data for filling) time series. Because it may not be desired to output the independent time series, it can be treated as a temporary time series. The edit dialog for the fillRegression() command allows you to right click on the independent time series and convert it to a TEMPTS. This keyword tells TSTool to read the time series for the command and then free the time series as soon as it has been used. Not all commands accept the TEMPTS notation - use the command editors to create supported commands.

Note that when editing commands, you typically select identifiers for time series that have been defined to that point in the commands list. If TEMPTS is used, you will need to add the time series before the command and then remove the time series after the TSTool command editor has found the time series for use with the TEMPTS notation.
Alternatively, edit the command manually to bypass the editor's help.

5. After processing the time series, a list of available time series that are in memory are listed in the GUI. One or more of these time series can be selected and viewed using the **Results** menu or analyzed using the **Tools** menu. Time series can also be saved in some of recognized input type formats using the **File...Save...Time Series As** menus.

If running in batch mode using the -commands option, all of the above steps occur in sequence and the GUI interfaces are not displayed. Old command files should be updated to reflect the new processing sequence. For example, old files may use the -ostatemod option. New command files should use the writeStateMod() command, which is executed at the time it is found in the command file. The old

-ostatemod option did not initiate a write when the command was found but caused the write at the end of processing. The newer features allow much greater flexibility in data processing. Processing the example shown in step 5 results in three time series in memory:

1. A time series identified by 08235350.USGS.Streamflow.Month, containing the sum of the two time series.
2. A time series identified by 08236000.USGS.Streamflow.Month, containing the input to the add() command.
3. A time series **also** identified by 08235350.USGS.Streamflow.Month, containing the original data from the time series that is added to. This contains the original data because a time series identifier by itself in a command list will cause the time series to be read.

These time series can be graphed or saved in an output file.

2.5 Using Time Series Aliases

The previous sections discussed time series identifiers and processing time series. The concept of a time series alias was described as a “shortcut” when identifying a time series. Aliases are useful when creating more complicated lists of commands, where using full time series identifiers becomes cumbersome. Aliases are typically assigned when creating new time series using the command syntax:

```
TS X = someCommandThatCreatesATimeSeries()
```

Time series aliases are meant primarily for use during processing, not as a way to identify time series in output. Most supported time series input types do not inherently use aliases (an exception is the DateValue format, which will initialize a time series’ alias if the input DateValue file specifies the information). Instead, time series typically are identified by the location part of the time series identifier (e.g., a station identifier). Although time series can use aliases to simplify processing, the location part of the identifier will generally be used when outputting time series to files or databases.

The time series manipulation features of TSTool facilitate using variations of an input time series for analysis. For example, a time series may be read and manipulated to produce several variants, which are then written for use in a model or analysis. The copy() command could be used to create copies of the original time series; however, this command does not currently reset the location part of the identifier and therefore the time series, when written to output, will likely have the same identifier (an enhancement to the copy() command is being studied that will assign a new time series identifier during the copy). An alternative is to use the TS X = newTimeSeries() command to create a new time series (specifying a location part of the time series identifier that is suitable for output), and then use the setFromTS() command to copy all or part of the original time series into the new time series. These two commands therefore allow one time series to be read and copied into new time series, each of which has a new location in the time series identifier.

If specified, time series aliases are generally output in the legends of graphs and other data products.

The remainder of this document describes TSTool's specific time series processing and viewing features.

This page is intentionally blank.

Getting Started

Version 07.00.00, 2006-11-21, Color, Acrobat Distiller

This chapter provides an overview of the TSTool graphical user interface. The TSTool GUI has three main functions:

Display and analyze time series data independent of full-scale modeling. In this capacity, a graph or summary can be created and then TSTool can be closed.

1. Format long lists of time series for use with applications like the State of Colorado's StateMod and StateCU models or other software. In this capacity, information from the previous item can be incorporated into a commands file and run the application to generate model files.
2. Read time series and produce time series products (e.g., JPEG image files), for use on web sites or to facilitate review of database contents or model output. In this capacity TSTool is used to generate data products in a streamlined fashion.

The remainder of this section provides an overview of the graphical user interface, in the order of the menus on the menu bar.

3.1 Starting TSTool

Within the State of Colorado's CDSS, TSTool can be started using **Start... All Programs... CDSS... TSTool** (or **Start... Programs... CDSS... TSTool**).

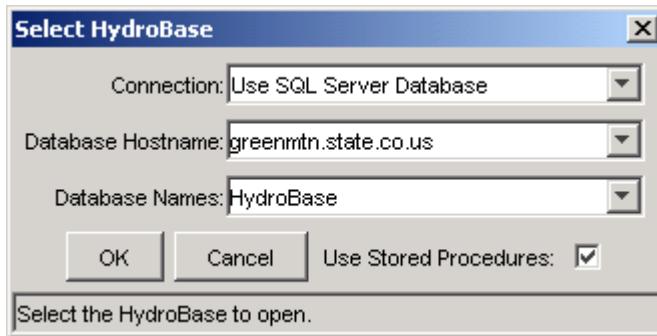
If a commands file has been created, it can be processed in batch mode using the following command line:

```
tstool -commands commands.TSTool
```

It is customary to name commands files with a *.TSTool* file extension.

3.2 Select HydroBase Dialog

If the HydroBase input type is enabled (see the **HydroBase Input Type** appendix), the HydroBase login dialog will be automatically shown when TSTool starts in interactive mode. The dialog is used to select an ODBC data source name (DSN) for the State of Colorado's HydroBase database. A HydroBase database can also be selected from the **File...Open... HydroBase...** menu.



Menu_Open_HydroBase

Select HydroBase Database Dialog

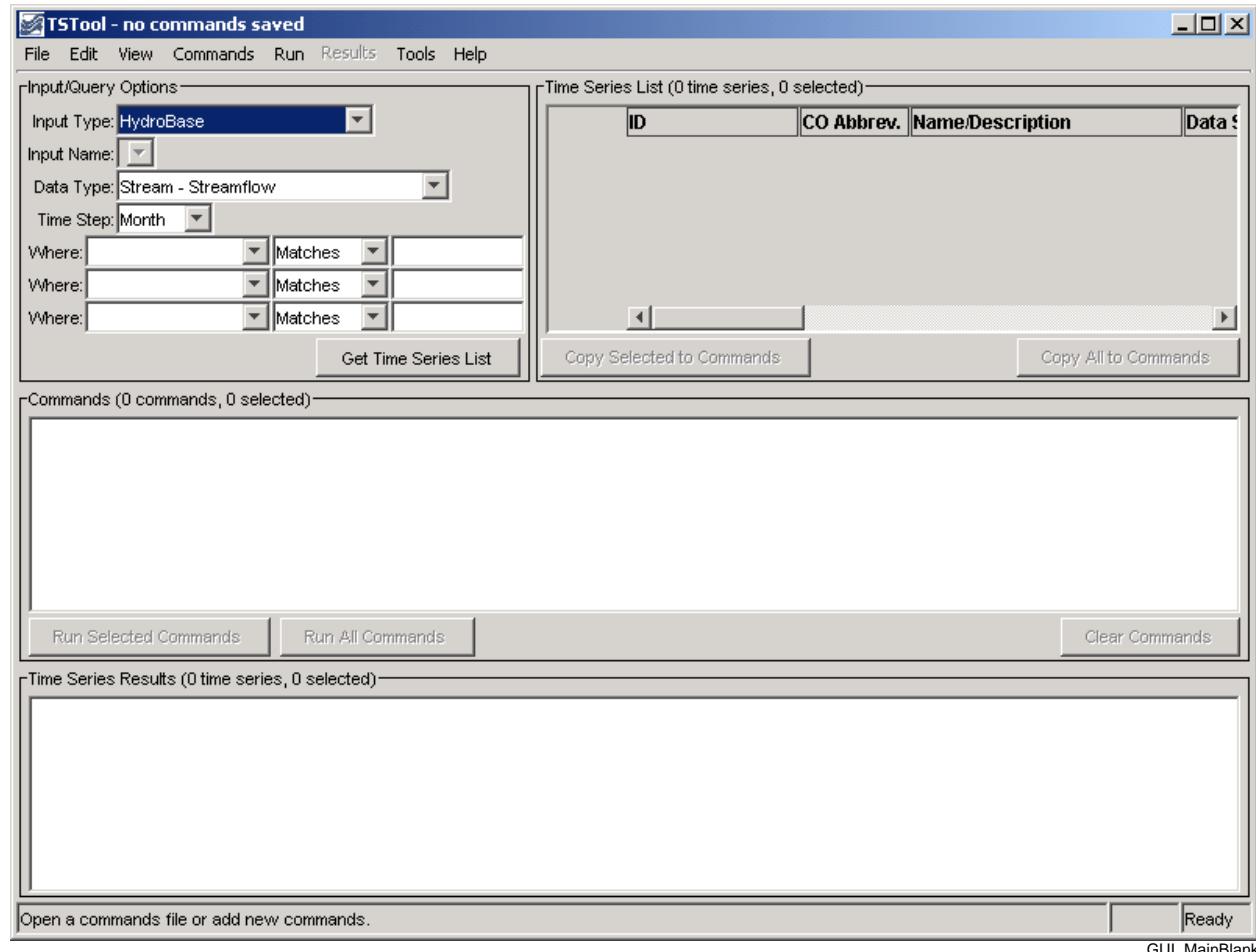
If using the HydroBase database with Microsoft Access, you should have already configured a HydroBase database ODBC DSN. You can select a local database and appropriate ODBC DSN, or, if you have access to a SQL Server HydroBase server, you can select **Use SQL Server Database** (as shown above), and select the database server. You can also cancel the login, in which case HydroBase features will be disabled but you will be able to work with other input types.

3.3 Main Interface

The following figure illustrates the main TSTool interface during a typical session, immediately after starting. The main interface is divided into three main areas:

- **Input/Query Options** (top left) and **Time Series List** area (top right)
- **Commands** (middle)
- **Time Series Results** (bottom)

Status, warning, and debug messages appear in the status message area at the bottom of the main window.



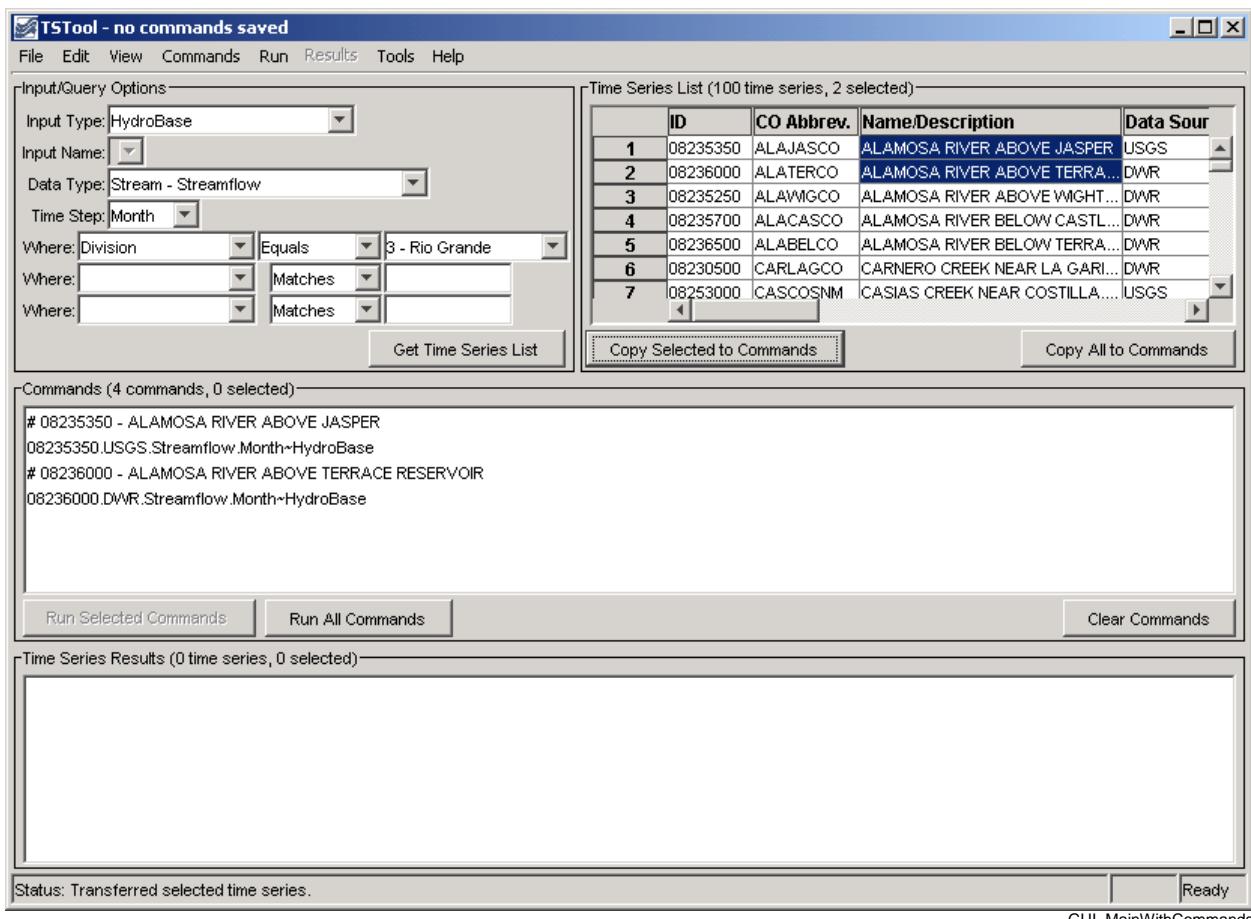
Initial TSTool Interface

3.3.1 Input/Query Options and Time Series List Area

The upper part of the main window contains the **Input/Query Options** and **Time Series List** area. The **Input/Query Options** choices help select time series information from input types. The interactive interface is useful when searching a database or picking a time series from a file that includes multiple time series. An alternative to the following interactive approach is to use the **Commands** menu (see the **Commands** chapter), which is appropriate for more complicated analysis. To select time series, execute the following steps:

1. Select the **Input Type**. Input types define the storage format (e.g., database or file) for time series data. The DateValue input type is the default. More specific input types (e.g., the HydroBase database) may be the default if enabled. See the appropriate appendix for a description about supported input types. Depending on the input type, some of the remaining selection choices may be disabled or limited. In most cases, TSTool will assume that you are correctly associating an input type with the actual input that is selected (e.g., that you will select a file that matches the input type when **Get Time Series List** is pressed – see below). Selecting some input types may prompt for a file, which is then listed in the **Input Name** choices.
2. Select the **Data Type** (if appropriate for the input type). For example, select **Streamflow** or **Diversion** if using a HydroBase input type. For some input types, the data type will be listed as **Auto**, indicating that the data type automatically will be determined from the input itself.
3. Select the **Time Step** (if appropriate for the input type). The time step, also referred to as the data interval, will generally be limited by the input type. For example, if reading from the HydroBase database, the Streamflow data type will result in Day, Month, and Irregular (real-time) time steps being listed. The time step will be shown as **Auto** for some input and data types and will be determined as data are read.
4. Specify the **Where** and **Is** clause(s) for the query (if appropriate for the input type). This information will limit the number of time series that are returned. For some input types, choices will be displayed as choices, whereas for other input types, all time series for the input type will be listed.
5. Press the **Get Time Series List** button in the **Input/Query Options** area, and TSTool will display a list of matching time series in the **Time Series List**. If the input type is a file, you may first be prompted to select the file containing the time series. The **Time Series List** list shows a list of matching time series, including standard time series information. As much as possible, the column headings are consistent between different input types. The results are typically sorted by name or identifier if from a database, or if read from a file are listed according to the order in the file. Right-click on the column headings and select **Sort Ascending** to sort by that column (the other columns will adjust accordingly). The sorts are alphabetical so some numeric fields may not sort as expected due to spaces, etc.
6. Move time series to the **Commands** list in the center of the main interface selecting rows in the Time Series list and then pressing the **Copy Selected to Commands** button. Or, if appropriate, press the **Copy All to Commands** button. The **Commands** and **Time Series Results** areas are discussed in the following section and can contain mixed data types and time steps. Analysis commands may require that the units are compatible, but general viewing tools do not require the same units. To mix data types, make multiple queries using the **Input/Query Options** and **Time Series List** areas and select from the lists as necessary, accumulating time series identifiers in the **Commands** list.

After providing selection information, the main interface might look like the following figure:



TSTool after Pressing *Get Time Series List* and selecting from *Time Series List*

3.3.2 Commands List

The **Commands** list occupies the middle of the main interface and contains:

- time series identifiers corresponding to time series selected from the **Time Series List**, and
- commands selected from the **Commands** menu (see **Chapter 4 – Commands**).

Time series identifiers are added to the **Commands** list by single clicking on items in the **Time Series List** and copying the identifiers to the **Commands** list. Time series identifiers are formatted by transferring information from the appropriate columns in the **Time Series List** list to the **Commands** list.

An alternative to using the **Time Series List** to select time series is to use specific read commands from the **Commands** menu (e.g., use a `readDateValue()` command). Using read commands is useful when performing a more complicated analysis on specific input sources (e.g., specific data files) or when processing large amounts of data. The interactive interface is useful when searching a database or generating a simple graph.

The **Commands** (and the **Time Series Results**) lists behave according to Windows conventions:

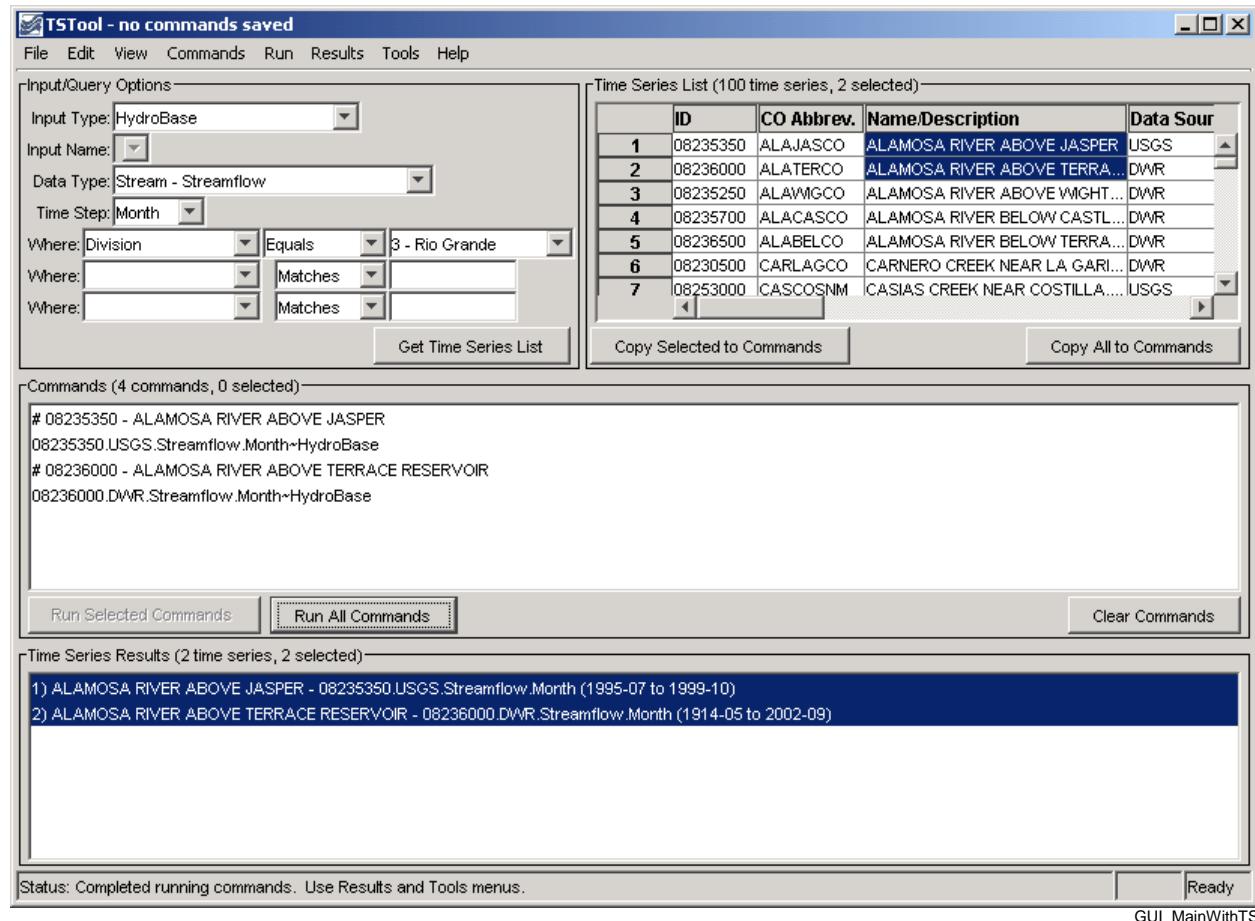
- **Single-click** to select one item.
- **Ctrl-click** to additionally select an item.
- **Shift-click** to select everything between the previous selection and the current selection.

Right-clicking over the **Commands** list displays a pop-up menu with useful command manipulation choices, some of which are further described in following sections (e.g., edit menu choices are discussed in **Section 3.5 - Edit Menu**). A summary of the pop-up menu choices is as follows:

Menu Choice	Description
Edit	Edit the selected command using custom edit dialogs, which provide error checks and format commands.
Edit (no error checks)	Edit the selected command using a generic editor. This is useful if you are familiar with a command's syntax or need to edit a command that would not be understood by an interactive command editor dialog.
Cut	Cut the selected commands for pasting.
Copy	Copy the selected commands for pasting.
Paste	Paste commands that have been cut/copied, pasted after the selected row.
Delete	Delete the selected commands (currently same as Cut).
Find Commands(s)	Find commands in the command list. This displays a dialog. Use the right-click in the found items to go to or select found items.
Select All	Select all the commands.
Deselect All	Deselect all the commands. This is useful because only selected commands are processed (or all if none are selected). It is therefore important not to unknowingly have one or a few commands selected during processing.
Convert Selected Commands to # Comments	Convert selected commands to # comments.
Convert Selected Commands from # Comments	Convert # comments to commands.
Run All Commands (create all output)	Run all commands and create output (e.g., graphs and files).
Run All Commands (ignore output commands)	Run all commands but skip any output commands. This is useful if a batch command file has been read and time series are to be listed in the GUI but output products are not to be generated automatically.
Run Selected Commands (create all output)	Run selected commands and create output (e.g., graphs and files).
Run Selected Commands (ignore output commands)	Run selected commands but skip any output commands. This is useful if a batch command file has been read and time series are to be listed in the GUI but output products are not to be generated automatically.

3.3.3 Time Series Results

The commands in the **Commands** list are processed by pressing the **Run Selected Commands** or **Run All Commands** buttons below the commands list area (or by using the **Run** menu). The time series that result from processing are listed in the bottom of the main interface, as shown in the following figure:



TSTool after Running Commands

The time series listed in the **Time Series Results** list can then be viewed using the **Results** menu, analyzed further using the **Tools** menu, or output using the **File...Save...** menus. Only the selected time series will be output (or all if none are selected).

Right-clicking over the **Time Series Results** list displays a pop-up menu with useful time series viewing choices, including a choice to view the time series properties. The right-click menu choices are summarized below:

Time Series Results List Popup Menu Choices

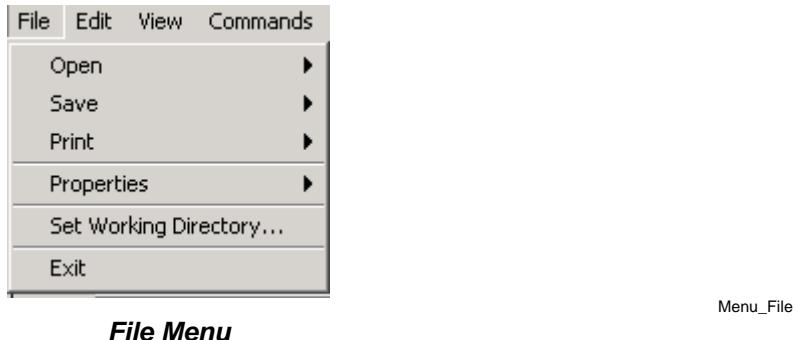
Menu Choice	Description
Graph - Bar (left of date)	Display bar graph for selected time series, drawing bars to the left of the date.
Graph - Bar (center on date)	Display bar graph for selected time series, drawing bars centered on the date.
Graph - Bar (right of date)	Display bar graph for selected time series, drawing bars to the right of the date.
Graph - Duration	Display a duration graph for the selected time series.
Graph - Line	Display a line graph for selected time series.
Graph - Line (log Y-axis)	Display a line graph for the selected time series, using a log10 y-axis.
Graph - Period of Record	Display a period of record graph for the selected time series.
Graph - Point	Display a graph using symbols but no connecting lines.
Graph - Predicted Value	Display a graph of data and the predicted values from regression.
Graph - Predicted Value Residual	Display a graph of data minus the predicted values from regression.
Graph - XY-Scatter	Display an XY-scatter plot for the selected time series.
Table	Display a scrollable table for the selected time series.
Report - Summary	Display a summary for selected time series.
Find Time Series...	Find time series in the time series list. This displays a dialog. Use the right-click in the found items to go to or select found items.
Select All for Output	Select all time series for output.
Deselect All	Deselect all time series for output.
Time Series Properties	Display the time series properties dialog (see the TSView Time Series Viewing Tools appendix for a complete description of the properties interface).

Viewing capabilities are described further in [Section 3.9 - Results Menu](#) and the [TSView Time Series Viewing Tools](#) appendix.

The remainder of this chapter summarizes the TSTool menus.

3.4 File Menu - Main Input and Output Control

The **File** menu provides standard input and output features as described below. Some menus are visible only when certain input types are enabled (see the **Installation and Configuration** appendix). Some menus are only enabled when time series have been processed.



3.4.1 File...Open – Open Commands File or Databases

The **File...Open** menu displays menu items as follows:



The **File...Open...Commands File** menu item displays a dialog to select an existing commands file. After a file is selected, the file contents replace the contents of the **Commands** list. If commands already exist in the **Commands** list and have been modified, you are given the option of saving the existing commands first. Opening a commands file causes the working directory to be set to the directory from which the commands file was read, as if the `setWorkingDir()` command was executed. Consequently, `setWorkingDir()` commands may not be needed.

The **File...Open...HydroBase** menu item displays the **Select HydroBase** dialog discussed in **Section 3.2** (see also the **HydroBase Input Type** appendix).

The **File...Open...RiversideDB** menu item displays a dialog to select a RiverTrak® System configuration file, which specifies the location of a RiversideDB database (see the **RiversideDB Input Type** appendix).

3.4.2 File...Save – Save Commands File, and Time Series

The **File...Save** menu displays the following menu choices:



The **File...Save...Commands** and **File...Save...Commands As** menu items save the contents of the **Commands** list to a file. The name of the current commands file is shown in the TSTool title bar. All commands are saved, even if only a subset is selected. Saving a commands file causes the working directory to be set to the where the commands file was written, as if the `setWorkingDir()` command was executed. Consequently, `setWorkingDir()` commands may not be needed.

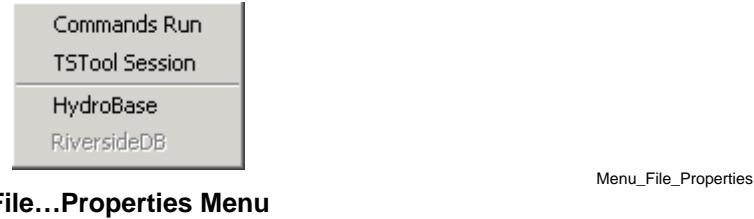
The **File...Save...Time Series As** menu item displays a file chooser dialog for saving time series in the **Time Series Results** list. See the **Input Type** appendices for examples of supported file formats. Only the selected time series in the **Time Series Results** list are saved (or all, if none are selected). Not all formats are supported because in most cases the write commands are used to automate processing of time series.

3.4.3 Print Commands

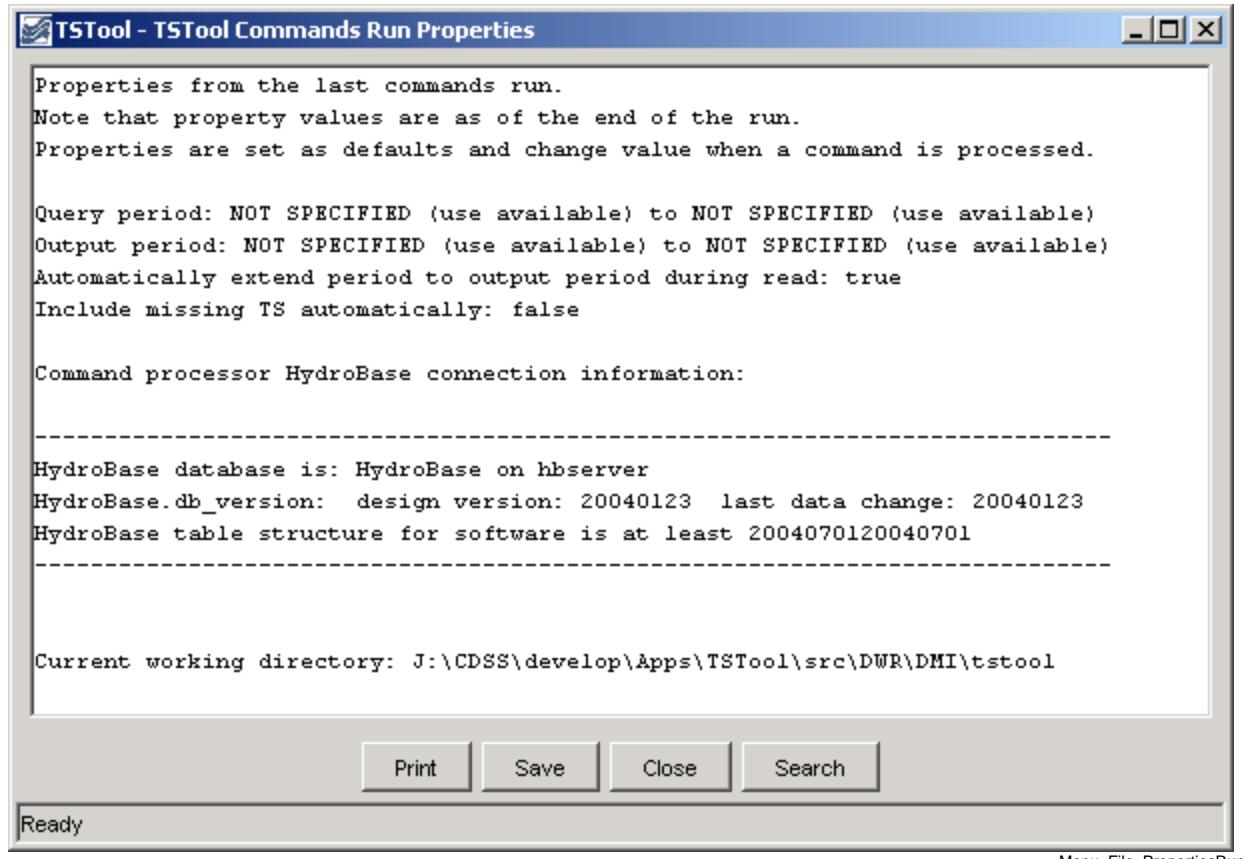
The **File...Print..Commands** menu prints the contents of the **Commands** list. This is useful while editing commands.

3.4.4 Properties for Commands Run, TSTool Session, and Input Types

The **File...Properties** menu displays the following menu items:

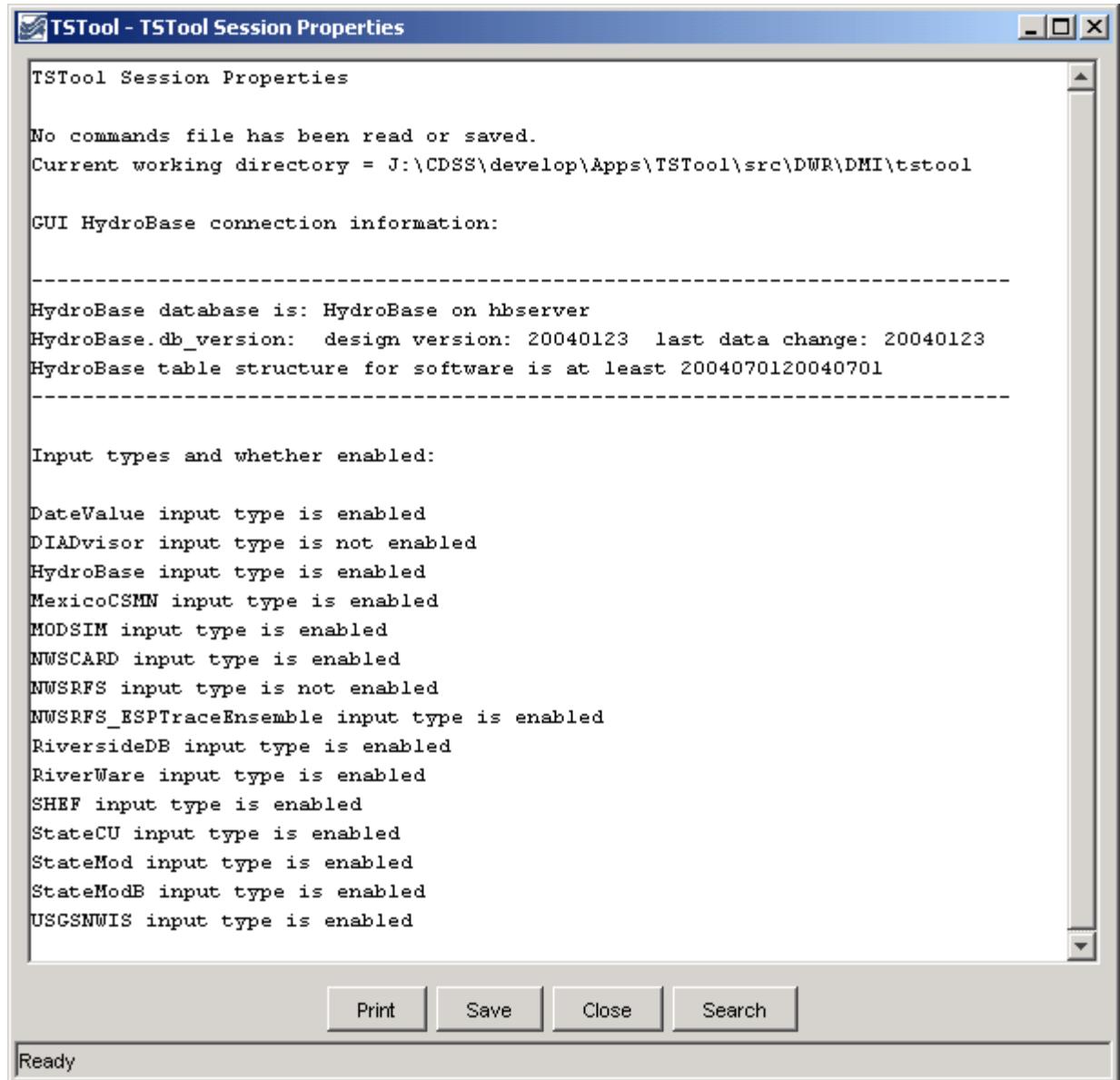


The **File...Properties...Commands Run** menu item displays information from the last time that the commands were run, including global properties that impact results:



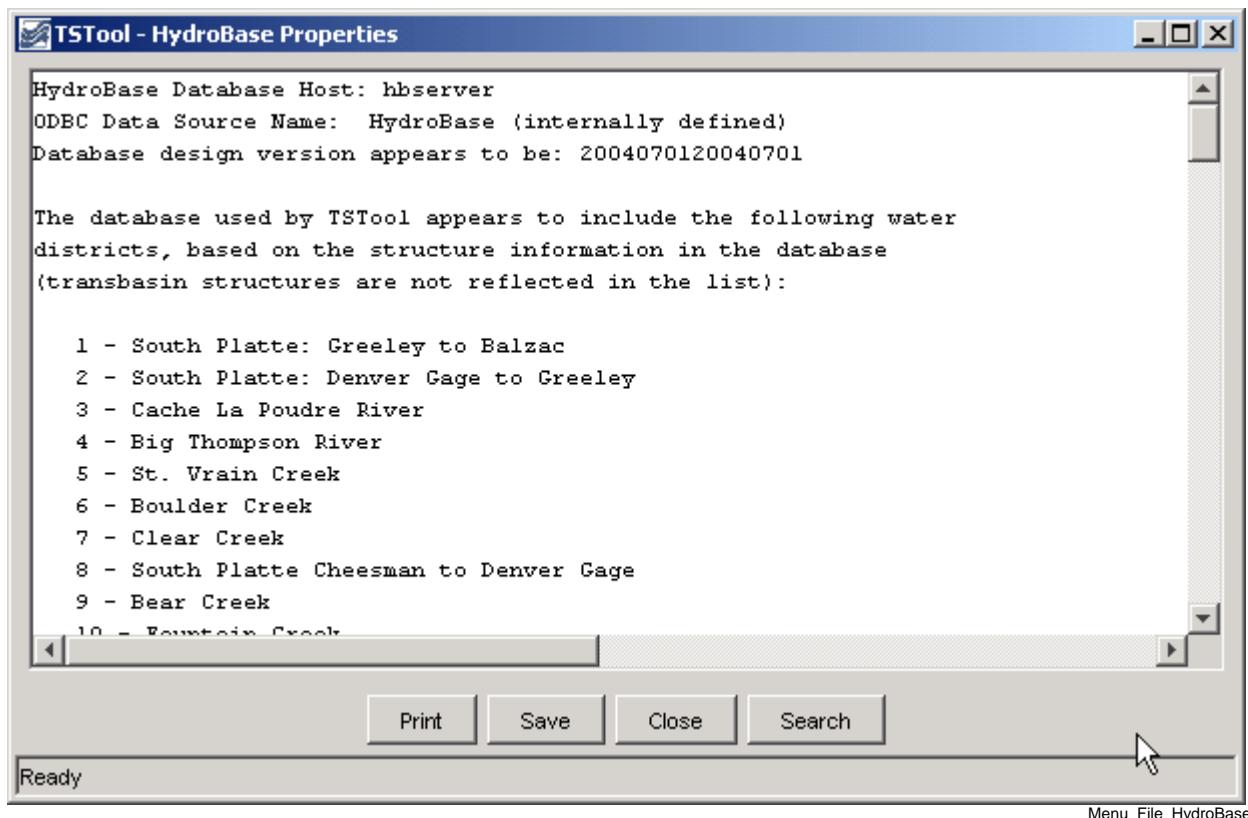
Properties of the Last Commands Run

The **File...Properties...TSTool Session** menu item displays information about the current TSTool session, as follows:



Menu_File_Properties_TSToolSession

The **File...Properties...HydroBase** menu item displays HydroBase properties, including the database that is being used, database version, and the water districts that are in the database being queried. The water districts are determined from the structure table in HydroBase. The information that is shown is consistent with that shown by other State of Colorado tools and is useful for troubleshooting.



HydroBase Properties Dialog

Menu_File_HydroBase

The **File...Properties...RiversideDB** menu item displays RiversideDB properties, if a RiversideDB connection is in place.

3.4.5 Set Working Directory

The **File...Set Working Directory** menu item displays a file chooser dialog that allows you to select the working directory. The working directory, when set properly, can greatly simplify commands files because relative file paths can be used for input and output. The working directory is normally set in one of the following ways, with the current setting being defined by the most recent item that has occurred:

1. the startup directory for the TSTool program,
2. the directory where a commands file was opened,
3. the directory where a commands file was saved,
4. the directory specified by a `setWorkingDir()` command,
5. the directory specified by **File...Set Working Directory**.

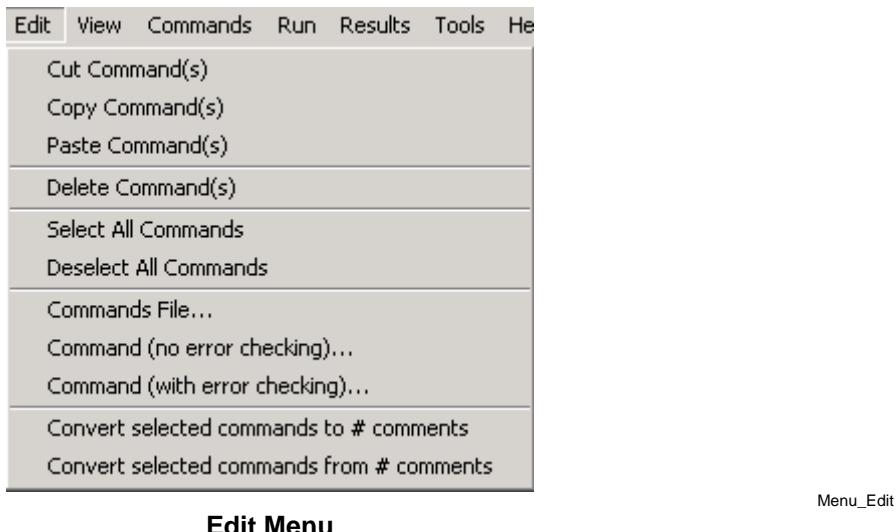
The menu item is provided to allow the working directory to be set before a commands file has been saved (or opened) and it typically eliminates the need for `setWorkingDir()` commands in commands files.

3.4.6 Exit

The ***File...Exit*** menu exits TSTool. You will be prompted to confirm the exit. If commands have been modified, you will be prompted to save before exiting.

3.5 Edit Menu – Editing Commands

The **Edit** menu can be used to edit the **Commands** list. Edit options are enabled and disabled depending on the status of the **Commands** list. Specific edit features are described below. Right clicking over the **Commands** list provides a popup menu with many choices described below.



3.5.1 Cut/Copy/Paste/Delete

The **Edit...Cut** and **Edit...Copy** menu items are enabled if there are items in the **Commands** list. Currently, these features do not allow interaction with other applications. **Cut** deletes the selected item(s) from the **Commands** list and saves its information in memory. **Copy** just saves the information in memory. After **Cut** or **Copy** is executed, select an item in the **Commands** list and use **Paste** (see below).

Paste is enabled if one or more commands from the **Commands** list has been cut or copied. To paste the command(s), select commands in the **Commands** list and press **Edit...Paste**. The commands will be added after the last selected command. To insert at the front of the list, paste after the first command, and then cut and paste the first command to reverse the order.

The **Delete** choice currently works exactly like the **Cut** choice. Additionally, after lines in the **Commands** have been selected, you can press the **Clear Commands** button below the **Commands** list to cut/delete.

The **Clear Commands** button in the **Commands** area deletes the selected commands or all commands if none are selected. You will be prompted to confirm the clear if no commands are specifically selected.

3.5.2 Select All Commands/Deselect All Commands

The **Edit...Select All Commands** and **Edit...Deselect All Commands** menu items are enabled if there are items in the **Commands** list. Use these menus to facilitate editing. Note that when editing commands it is often useful to deselect all commands so that new commands are added at the end of the commands list.

3.5.3 Edit Commands File

The **Edit...Commands File** menu choice can be used to edit a commands file using **NotePad** on Windows or **nedit** on UNIX machines. Currently, there is no way to change the editor. You must re-read the commands file into TSTool after using the editor for TSTool to recognize the time series commands in the commands file. This feature is less useful than in the past because editor dialogs have now been implemented for all commands.

3.5.4 Edit Command

The **Edit... Command (no error checking)** menu can be used to edit an individual command using a one-line text area dialog. This is suitable when you need to quickly change a command (e.g., to change a time series identifier from Month to Day). This edit mode is also useful if you are well-versed in TSTool's commands or need to update an old commands file. This feature is also accessible by right-clicking on the **Commands** list and selecting the **Edit (no error checks)** menu item.

The **Edit... Command (with error checking)** menu can be used to edit an individual command. TSTool will determine the command that is being edited and will display the editor dialog for that command, performing data checks. **Most old commands will be automatically detected and will be converted to new command syntax. Consequently, use the edit feature to systematically update old command files.** This feature is also accessible by right-clicking on the **Commands** list and selecting the **Edit** menu item.

3.5.5 Convert Selected Commands To/From Comments

The **Edit...Convert selected commands to comments** menu can be used to toggle selected commands in the **Commands** list to comments (lines that begin with #). This is useful when temporarily disabling commands, rather than deleting them.

The **Edit...Convert selected commands from comments** menu can be used to toggle selected commands in the **Commands** list from comments back to active commands. This is useful when re-enabling commands that were temporarily disabled.

Note that the multi-line /* */ comment notation can be inserted using the **Commands...General** menu.

3.6 View Menu – Display Map Interface

The **View** menu currently has limited choices:



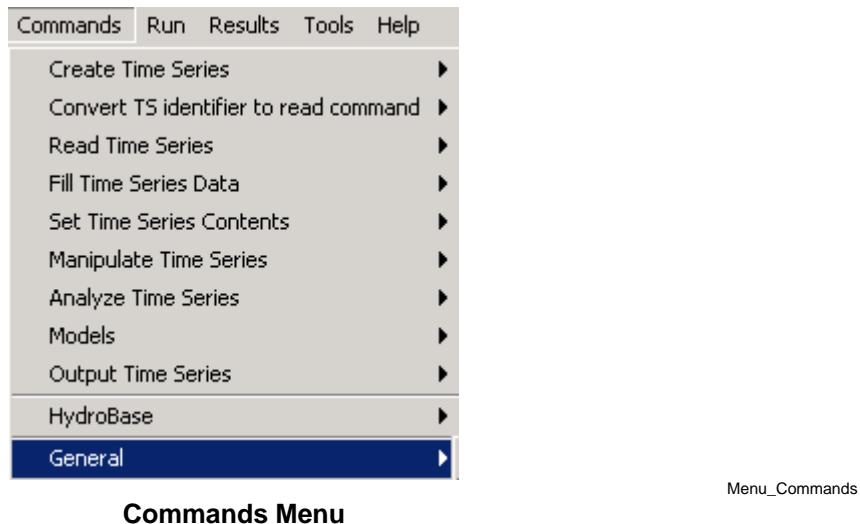
The **View...Map Interface** menu displays a map interface in a separate window. This interface has limited capabilities. See the **GeoView Mapping Tools** appendix for more information.

The **GeoView** component in TSTool will display any GeoView project file (.gvp). The GeoView window has a **File** menu that is separate from the main TSTool interface. Use the **GeoView File...Open Project** menu item to read a .gvp file. Use the **File...Add Layer** menu item to add a layer (e.g., an ESRI Shapefile) to the view. The **File...Add Summary Layer** menu item is useful to display the spatial variability of data.

This section will be expanded as additional integration of spatial displays with time series analysis occurs.

3.7 Commands Menu

The **Commands** menu provides several menus (as shown in the following figure), which allow time series processing commands to be inserted into the **Commands** list.



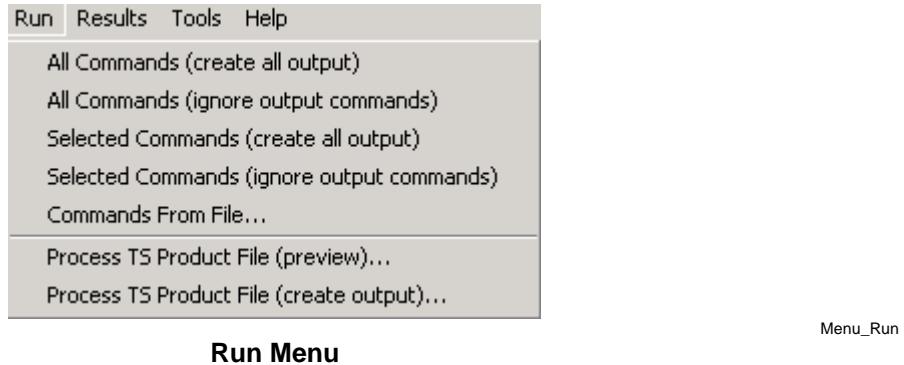
Time series commands are organized into the following categories:

1. **Create Time Series** - create one or more new time series in memory
2. **Convert TS identifier to read command** – convert a time series identifier in the **Commands** list area to a read command
3. **Read Time Series** – read time series from a file or database
4. **Fill Time Series** - fill missing data
5. **Set Time Series** – set time series data or properties
6. **Manipulate Time Series** - manipulate data by transforming the contents of the time series (e.g., scale a time series' data values)
7. **Analyze Time Series** – perform analysis on time series, without modifying the time series
8. **Models** – advanced or specific models that operate on time series data
9. **Output Time Series** - write time series results to a file or graph
10. Commands specific to various input types
11. **General** – general commands (e.g., to set output period)

Chapter 4 – Commands discusses commands in more detail and the **Command Reference** at the back of this documentation provides a reference for each command.

3.8 Run Menu – Run Commands

The **Run** menu processes the **Commands** list to generate the **Time Series Results** for output.



The **Run...All Commands (create all output)** menu will process all the commands in the **Commands** list and create output if appropriate. For example, the `writeStateMod()` command will write the time series that are in memory to a StateMod file.

The **Run...All Commands (ignore output commands)** menu will process the commands in the **Commands** list, ignoring commands that generate output products. With this option, you can process a commands file prepared for batch mode, but only have the time series available for viewing in the GUI rather than generating the output files. For example, `writeStateMod()` commands will not be processed. This increases performance and minimizes creation of files.

The **Run...Selected Commands** menu items are similar to the above, except that only selected commands are run.

The **Run...Commands From File** choice will run a commands file but not generate any time series for viewing in the GUI. This is equivalent to running in batch mode but initiating the run from the TSTool GUI.

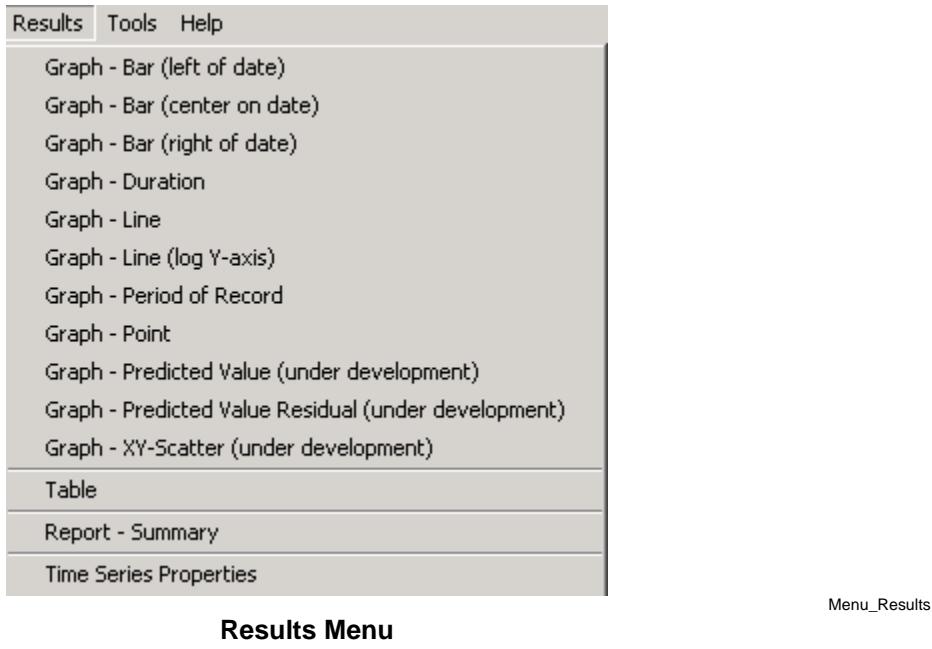
Menu items similar to the above are also available in a popup menu by right-clicking on the **Commands** list.

3.8.1 Process TSProduct

The **Run...Process TS Product File** menu items can be used to create time series products by processing time series product definition files. The **TSView Time Series Viewing Tools** appendix describes the format of these files. Time series product definition files can be saved from graph views using **Save As... Time Series Product**. The `processTSProduct()` command provides equivalent functionality.

3.9 Results Menu – Display Time Series

The **Results** menu displays time series that are listed in the **Time Series Results** list. The time series can be viewed multiple time, using the same time series results.



Graphing time series results in slightly different viewing options being available, depending on the type of graph. In many cases, you will be able to see three views of time series, consisting of a graph, summary, and table. Additionally, you can select the graph properties and choose the colors and symbols to be used for each time series. The **TSView Time Series Viewing Tools** appendix describes in detail the graphing tools.

Most of the main **Results** menu choices are available in a popup menu that is displayed when right-clicking on the **Time Series Results** list.

3.9.1 Graph - Bar

Bar graphs are generated by selecting time series from the **Time Series Results** list and pressing **Results...Graph - Bar** menus. See the **TSView Time Series Viewing Tools** appendix for information about bar graphs. The position of the bars relative to the date/time position depends on whether data are instantaneous, mean, or accumulated.

3.9.2 Graph - Duration

Duration graphs are generated by selecting time series in the **Time Series Results** list and selecting the **Results...Graph - Duration** menu. See the **TSView Time Series Viewing Tools** appendix for information about duration graphs.

3.9.3 Graph - Line

A line graph is generated by selecting time series in the **Time Series Results** list and then selecting the **Results...Graph - Line** menu. See the **TSView Time Series Viewing Tools** appendix for information about line graphs.

3.9.4 Graph - Line (log Y-axis)

Log Y-axis line graphs are generated by selecting time series in the **Time Series Results** list and then selecting the **Results...Graph - Line (log Y-axis)** menu. See the **TSView Time Series Viewing Tools** appendix for information about log Y-axis line graphs.

3.9.5 Graph - Period of Record

The period of record graph is useful to display the availability of data over a period. Horizontal lines are drawn for each time series, with breaks in the line indicating missing data. An alternative to this graph type is to use the **Tools...Data Coverage by Year** report (see **Chapter 5 – Tools**). See the **TSView Time Series Viewing Tools** appendix for information about period of record graphs.

3.9.6 Graph – Point

Point graphs are useful for data that are collected infrequently. For example, the interval of the data may be daily; however, values may only be available once per month, on various days of the months. See the **TSView Time Series Viewing Tools** appendix for information about point graphs.

3.9.7 Graph – Predicted Value

The predicted value graph requires as input two time series. First, a regression analysis is performed, similar to the analysis done for the XY-Scatter plot. The original two time series are then plotted, additionally with the time series that would be generated using the regression results. The predicted time series and the original time series will be the same where their periods overlap, with only the predicted time series shown outside of that period.

3.9.8 Graph – Predicted Value Residual

The predicted value residual graph performs the same analysis as the predicted value graph. Where the original and predicted time series overlap, the difference is computed and plotted as a time series. The resulting bar graph therefore shows the relative goodness of fit of the estimated time series.

3.9.9 Graph - XY-Scatter

An XY-scatter graph is generated by selecting two or more time series from the **Time Series Results** list and then selecting the **Results...Graph - XY-Scatter** menu. See the **TSView Time Series Viewing Tools** appendix for information about XY-Scatter graphs.

3.9.10 Table

A table display is generated by selecting one or more time series from the **Time Series Results** list and then selecting the **Results...Table** menu. See the **TSView Time Series Viewing Tools** appendix for information about table displays.

3.9.11 Report - Summary

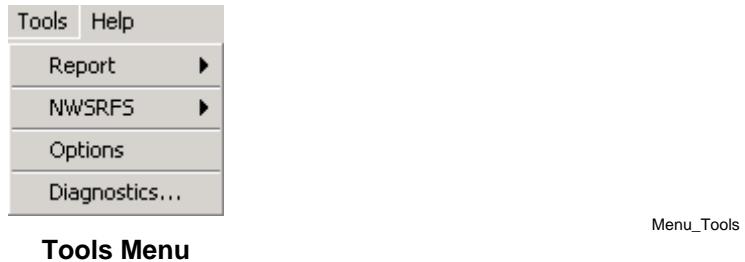
A summary report for time series can be generated by selecting time series in the **Time Series Results** list and then selecting the **Results...Report - Summary** menu. See the **TSView Time Series Viewing Tools** appendix for information about summary displays.

3.9.12 Time Series Properties

Time series properties include all the information other than the data values. For example, properties include the period of record, data units, processing history, etc

3.10 Tools Menu

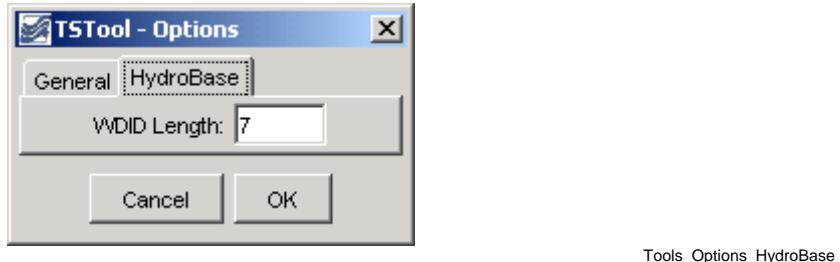
The **Tools** menu lists tools that perform additional analysis on time series that are selected in the **Time Series Results** list. These features are similar to the **Results** features in that a level of additional analysis is performed to produce the data product.



Analysis tools are described in more detail in **Chapter 5 – Tools**. The following sections describe the **Tools...Options** and **Tools...Diagnostics** features.

3.10.1 Options

The **Tools...Options** menu displays program options, in tabbed panels. Currently, only one option is available, to specify the total length of water district identifiers (WDIDs), for use with the HydroBase input type.



The WDIDs are used as the location part of the time series identifier. A water district identifier is comprised of a two-digit zero-padded water district (e.g., 01, 20) and a zero-padded identifier for structures within the water district. For example, ditch headgates are typically numbered 500 or greater, within each water district. For modeling purposes, the WDIDs are typically treated as character strings. To allow for distinct and unambiguous identifiers, WDIDs are typically padded with zeros to have consistent overall lengths. In the past, six characters were used for identifiers in model data sets. However, this length is insufficient to handle identifiers in some water districts and therefore the default in TSTool is seven characters. This menu item can be used to set the length if TSTool is being used to create time series and the default length is not compatible with the needed output. The WDID length is enforced when time series are listed. If necessary, the time series identifiers can be edited manually to add or remove padding zeros.

3.10.1 Diagnostics

The **Tools...Diagnostics** menu displays the diagnostics interface, which is used to set message levels and view messages as TSTool processes data. **The Tools...Diagnostics – View Log File** menu displays the log file viewer. These tools are useful for troubleshooting problems. Refer to **Chapter 5 Tools** for more information.

3.11 Help Menu

The help menu displays the version of TSTool.

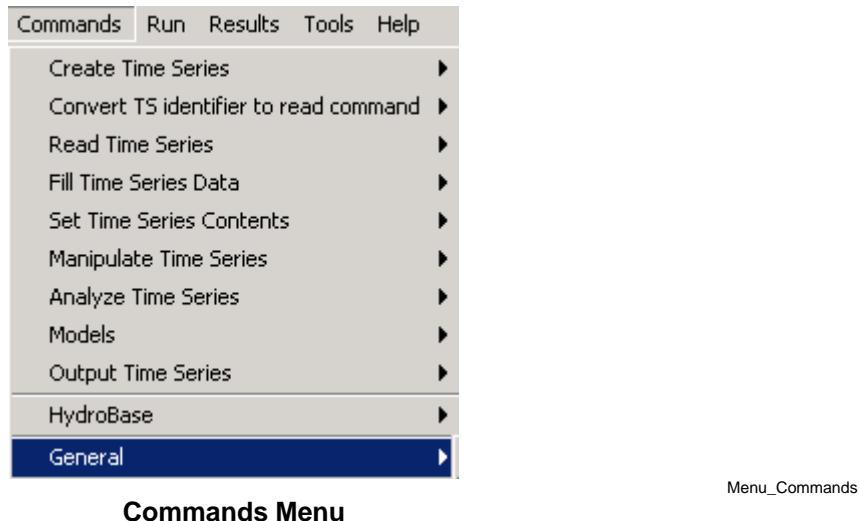


The **Help...About TSTool** menu displays the program version number, for use in troubleshooting and support:

Commands

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The **Commands** menu provides several submenus that allow commands related to time series to be inserted into the **Commands** list.



Commands Menu

Menu_Commands

Time series commands are organized into the following categories:

1. **Create Time Series** - create one or more new time series in memory
2. **Convert TS identifier to read command** – convert a time series identifier in the **Commands** list area to a read command
3. **Read Time Series** – read time series from a file or database
4. **Fill Time Series** - fill missing data
5. **Set Time Series** – set time series data or properties
6. **Manipulate Time Series** - manipulate data by transforming the contents of the time series (e.g., scale a time series' data values)
7. **Analyze Time Series** – perform analysis on time series, without modifying the time series
8. **Models** – advanced or specific models that operate on time series data
9. **Output Time Series** - write time series results to a file or graph
10. Commands specific to various input types
11. **General** – general commands (e.g., to set output period)

Specific commands within each category are discussed in the following sections. The command syntax is described in detail in the **Command Reference** at the end of this documentation. Menu options are enabled/disabled depending on the state of the application.

4.1 Create Time Series

The **Commands...Create Time Series** menu inserts commands for creating new time series in memory.

```
createFromList()... <read 1(+) time series from a list of identifiers>
createTraces()... <convert 1 time series to 1+ annual traces>
TS Alias = changeInterval()... <convert time series to one with a different interval (under development)>
TS Alias = copy()... <copy a time series>
TS Alias = disaggregate()... <disaggregate longer interval to shorter>
TS Alias = newDayTSFromMonthAndDayTS()... <create daily time series from monthly total and daily pattern>
TS Alias = newEndOfMonthTSFromDayTS()... <convert daily data to end of month time series>
TS Alias = newStatisticYearTS()... <create a year time series using a statistic from another time series>
TS Alias = newTimeSeries()... <create and initialize a new time series>
TS Alias = normalize()... <normalize time series to unitless values>
TS Alias = relativeDiff()... <relative difference of time series>
TS Alias = weightTraces()... <weight traces to create a new time series>
```

Menu_Commands_CreateTimeSeries

Commands...Create Time Series Menu

These commands create new time series from external data (see `createFromList()`), by using user-supplied data (see `newTimeSeries()`), or by operating on existing time series. Time series created from existing time series are fundamentally different from the original and cannot take its place with the same identifier. For example, the data interval or identifier is different in the new time series.

One important difference between the commands is whether one or multiple time series are created as the result of a command. For example, the `TS Alias =` commands create a single time series, which can be referenced using the time series alias (`Alias`). However, other commands may create more than one time series, and the identifiers for the time series may not be known until the commands are run and input data are read. Consequently, commands that operate on multiple time series are useful for bulk processing of data but may be more difficult to use for detailed analysis and manipulation.

4.2 Converting Time Series Identifier to Read Command

The **Commands...Convert Time Series Identifier to Read Command** menu converts a selected time series identifier in the **Commands** list to a read command.

```
Convert TS identifier (X.X.X.X.X) to TS Alias = readTimeSeries()
Convert TS identifier (X.X.X.X.X) to TS Alias = readDateValue()
Convert TS identifier (X.X.X.X.X) to TS Alias = readHydroBase()
Convert TS identifier (X.X.X.X.X) to TS Alias = readRiverWare()
Convert TS identifier (X.X.X.X.X) to TS Alias = readStateMod()
Convert TS identifier (X.X.X.X.X) to TS Alias = readStateModB()
Convert TS identifier (X.X.X.X.X) to TS Alias = readUsgsNwis()
```

Menu_Commands_ConvertTSID

Commands...Convert Time Series Identifier to Read Command Menu

Currently, the only enabled feature is to convert a time series identifier to a `TS Alias = readTimeSeries()` command. This is used to assign an alias to the time series, simplifying its use in other commands. The input type from the time series identifier is interpreted by TSTool to determine how to read the time series. Unlike other commands menus, this menu does not insert a new command. It converts a single selected time series identifier.

In the future, it is envisioned that this menu will be enhanced to enable conversion of a time series identifier to a specific read command – this will allow the features of each read command to be used.

4.3 Read Time Series

The **Commands...Read Time Series** menu inserts commands to read time series from databases or files.

```
readDateValue()... <read 1(+) time series from a DateValue file>
readMODSIM()... <read 1(+) time series from a MODSIM output file>
readNwsCard()... <read 1(+) time series from an NWS CARD file>
readNWSRFSESPTraceEnsemble()... <read 1(+) time series from an NWSRFS ESP trace ensemble file>
readStateCU()... <read 1(+) time series from a StateCU file>
readStateMod()... <read 1(+) time series from a StateMod file>
readStateModB()... <read 1(+) time series from a StateMod binary output file>
statemodMax()... <generate 1(+) time series as max() of T5 in two StateMod files>
TS Alias = readDateValue()... <read 1 time series from a DateValue file>
TS Alias = readMODSIM()... <read 1 time series from a MODSIM output file>
TS Alias = readNwsCard()... <read 1 time series from an NWS CARD file>
TS Alias = readNWSRFSF55Files()... <read 1 time series from an NWSRFS F55 Files>
TS Alias = readRiverWare()... <read 1 time series from a RiverWare file>
TS Alias = readStateMod()... <read 1 time series from a StateMod file>
TS Alias = readStateModB()... <read 1 time series from a StateMod binary file>
TS Alias = readUsgsNwis()... <read 1 time series from a USGS NWIS file>
setIncludeMissingTS()... <create empty time series if no data>
setQueryPeriod()... <for reading data>
```

Menu_Commands_ReadTimeSeries

Commands...Read Time Series Menu

Read commands are grouped into commands that process one or more time series in a file (top) and commands that read a specific time series (TS Alias = commands). The TS Alias = commands assign an alias to the time series, which can then be referenced by other commands.

Commands that read multiple time series do not usually have access to the time series while commands are edited; consequently, they are useful for bulk processing of data and may be more difficult to use for detailed analysis and manipulation. It may be useful to read many time series and then use the `selectTimeSeries()` and `deselectTimeSeries()` commands to select a subset of the results (see output commands).

4.4 Fill Time Series Data

The **Commands...Fill Time Series Data** menu inserts commands for filling missing data in time series.

```

fillCarryForward()... <Fill TS by carrying forward - ** see fillRepeat()**>
fillConstant()... <Fill TS with constant>
fillDayTSFrom2MonthTSAAnd1DayTS()... <fill daily time series using D1 = D2*M1/M2>
fillFromTS()... <Fill time series with values from another time series>
fillHistMonthAverage()... <Fill monthly TS using historic average>
fillHistYearAverage()... <Fill yearly TS using historic average>
fillInterpolate()... <Fill TS using interpolation>
fillMixedStation()... <Fill TS using mixed stations (under development)>
fillMOVE2()... <Fill TS using MOVE2 method>
fillPattern()... <Fill TS using WET/DRY/AVG pattern>
fillProrate()... <Fill TS by prorating another time series>
fillRegression()... <Fill TS using regression>
fillRepeat()... <Fill TS by repeating values>
fillUsingDiversionComments()... <use diversion comments as data - HydroBase ONLY>
setAutoExtendPeriod()... <for data filling and manipulation>
setAveragePeriod()... <for data filling>
setIgnoreLEZero()... <ignore values <= 0 in historical averages>
setMissingDataValue()... <for data filling>
setPatternFile()... <for use with fillPattern() >
setRegressionPeriod()... <for fillRegression()>

```

Menu_Commands_FillTimeSeries

Commands...Fill Time Series Data Menu

Fill commands will only change values that are missing, whereas set commands will change values regardless of whether they are missing or non-missing. These commands can be executed in sequence to apply multiple fill techniques to time series. Many commands accept the * wildcard for the time series identifier, allowing all time series to be filled.

Time series may contain missing data due to the following reasons:

1. the data collection system is unavailable because of a failure, maintenance cycle, or hardware that is turned off because of seasonal use
2. in a real-time system the most current data have not yet been received
3. data collection hardware was not in place during a period (e.g., an early period)
4. measured values are suspected of being in error and are changed to missing
5. values in a computed time series cannot be computed (e.g., input data are missing)
6. a data source stores only observed values and non-recorded values are assumed to be missing rather than a specific value (e.g., zero)

Observations that are available are typically either measured values or values that have been estimated by the organization that collects and/or maintains the data. Data flags indicating missing data may or may not be available in the original source data (e.g., an 'm' or 'e' character flag is often used to indicate missing and estimated data).

TSTool handles missing data by internally assigning a special numeric value where data are missing. Different input types may have different missing data values but typically -999 or a similar extreme value is used. If the output period is specified using `setOutputPeriod()`, then extensions to the available time series period are filled with the missing data value. Data flags are supported for some input types. TSTool displays and output products indicate missing data by blanks, showing the missing data value, or a string (e.g., NC).

Filled time series are often required for use in computer models. TSTool provides a number of features to fill time series data. The data filling process consists of analyzing available data and using the results to estimate missing data values. The estimation process can be simple or complex, resulting in varying degrees of error and statistical characteristics of the final time series. The data analysis uses data available at the time that the fill command is encountered. Consequently if values have been changed since the initial read (e.g., because of layered fill commands), the changed values may impact the analysis. Basic statistical properties of the original data are saved after the initial read to allow use in later fill commands. For example, for monthly time series, the historical monthly averages are computed after the initial read to allow use with a `fillHistMonthAverage()` command.

The overall period that is being filled is controlled by the time series period or analysis period that is specified with fill commands. TSTool will not automatically extend the period of a filled time series after the time series is initially read. Use the `setQueryPeriod()` and `setOutputPeriod()` commands to control the time series period.

The following table lists the fill techniques that are supported by TSTool.

TSTool Fill Techniques and Associated Commands

Technique	Command	Typical Use
Carry Forward	<code>fillCarryForward()</code>	See <code>fillRepeat()</code> . This command is being phased out.
Constant	<code>fillConstant()</code>	Use when missing data can be estimated as a constant. For example, if only the early period of a "regulated" (e.g., reservoir) time series is missing, it may be appropriate to set the values to zero.
Monthly total, daily pattern	<code>fillDayTSFrom2MonthTSAnd1DayTS()</code>	Use to estimate a daily time series by applying the pattern of a related daily time series to monthly totals from the related and current time series. For example, use to estimate daily streamflow from monthly total values.
Fill from time series	<code>fillFromTS()</code>	Use non-missing values from a time series to fill missing values in another time series.
Historic Monthly Average	<code>fillHistMonthAverage()</code>	Use with monthly time series to estimate missing monthly values as the average of historic monthly values. For example, if applied to monthly precipitation data, a missing July value would be set to the average of observed July precipitation values (zero is an observation).

TSTool Fill Techniques and Associated Commands (continued)

Technique	Command	Typical Use
Historic Year Average	<code>fillHistYearAverage()</code>	Use with yearly time series to estimate missing data as the average of annual values.
Interpolation	<code>fillInterpolate()</code>	Use to estimate missing data by interpolating between non-missing values. For example, use to estimate reservoir level changes.
Mixed Station	<code>fillMixedStation()</code>	This command tries various combinations of <code>fillRegression()</code> and <code>fillMove2()</code> parameters with time series at different locations, to use the best combination.
Maintenance of Variance	<code>fillMOVE1()</code>	Use to estimate missing data using the Maintenance of Variance Extension (MOVE.1). For example, use to estimate unregulated streamflow from a related gage. This command is currently not enabled.
Maintenance of Variance	<code>fillMOVE2()</code>	Use to estimate missing data using the Maintenance of Variance Extension (MOVE.2). For example, use to estimate unregulated streamflow from a related gage. This approach has been shown to be slightly better than the MOVE.1 approach.
Historic Pattern Averages	<code>fillPattern()</code>	Similar to filling with historic averages with additional complexity of classifying historic months into categories. For example, historic averages for wet, dry, and average periods are computed and used as the historic averages. This command requires that the <code>setPatternFile()</code> control command also be used.
Prorate	<code>fillProrate()</code>	Fill a time series by prorating known values with another time series.
Regression	<code>fillRegression()</code>	Use to estimate missing data by using ordinary least squares regression. For example, use to estimate streamflow from a related gage.
Repeat	<code>fillRepeat()</code>	Use when it can be assumed that the last observed value before a missing period is a good estimate for missing data. For example, use with "forecasted" data where no future value is available for interpolation.
Using diversion comments	<code>fillUsingDiversionComments()</code>	This command is only available with the HydroBase input type and uses diversion comments to set additional diversion amounts to zero.

Fill commands can be layered (e.g., use `fillRegression()`, then `fillInterpolate()`, then `fillConstant()`). However, the analysis that occurs for each command may be impacted by earlier

fill commands. If necessary, use the `setFromTS()` command to piece together the results of independent fill commands into a final time series. The **Results...Graph - XY-Scatter** output provides options for selecting different fill techniques and viewing analysis details.

4.5 Set Time Series Data

The **Commands...Set Time Series Contents** menu inserts commands that set time series. Unlike fill commands, set commands reset values regardless of whether the values were missing in the time series.

```
replaceValue()... <replace value (range) with constant in TS>
setConstant()... <set all values to constant in TS>
setConstantBefore()... <set all values on and before a date to constant in TS>
setDataValue()... <set a single data value in a TS>
setFromTS()... <set time series values from another time series>
setMax()... <set values to maximum of time series>
```

Menu_Commands_SetTimeSeries

Commands...Set Time Series Contents Menu

Additional commands may be added in the future to set time series properties, including description/name, comments, etc.

4.6 Manipulate Time Series

The **Commands...Manipulate Time Series** menus insert commands for manipulating time series.

```
add()... <Add one or more TS to another>
addConstant()... <Add a constant value to a TS>
adjustExtremes()... <adjust extreme values>
ARMA()... <lag/attenuate a time series using ARMA>
blend()... <Blend one TS with another>
convertDataUnits()... <Convert data units>
cumulate()... <Cumulate values over time>
divide()... <Divide one TS by another TS>
free()... <Free TS>
multiply()... <Multiply one TS by another TS>
runningAverage()... <Convert TS to running average>
scale()... <Scale TS>
shiftTimeByInterval()... <Shift TS by an even interval>
subtract()... <Subtract one or more TS from another>
```

Menu_Commands_Manipulate

Commands...Manipulate Time Series Menu

Because the fundamental nature of the time series (e.g., data type, interval) is not changed, these commands do not result in the creation of a new time series. Manipulation commands typically add comments to the time series history, which is displayed in summary output.

4.7 Analyze Time Series

The **Commands...Analyze Time Series** menu inserts commands for analyzing time series, which typically produce a report or other output product:

```
analyzePattern()... <determine pattern(s) for fillPattern() (under development)>
compareTimeSeries()... <find differences>
```

Menu_Commands_AnalyzeTimeSeries

Commands...Analyze Time Series Menu

4.8 Models

The **Commands...Models** menu inserts commands that perform tasks that are more complex than simple data processes.

```
lagK()... <lag and attenuate (route) (under development)>
```

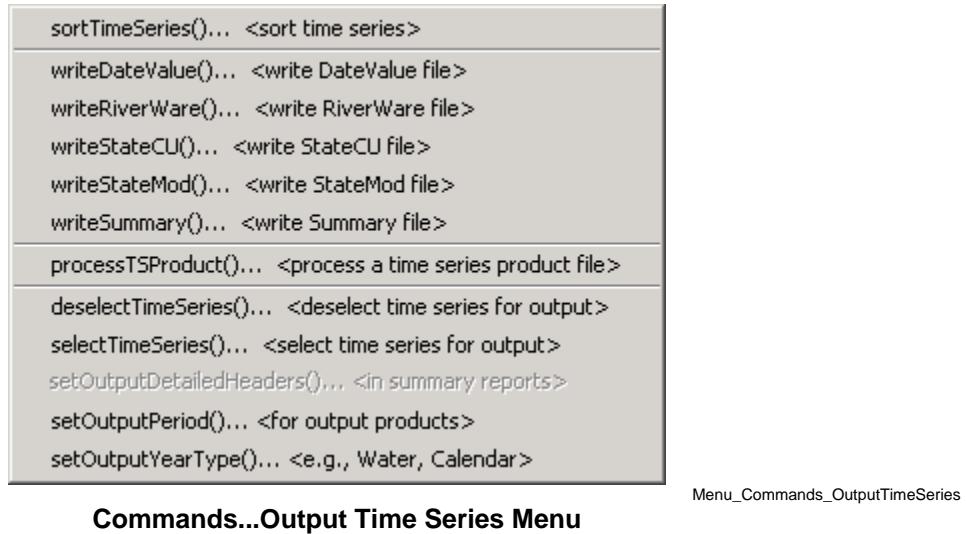
Menu_Commands_Models

Commands...Models Menu

Minimal model features are currently available. However, it is envisioned that additional capabilities will be added in the future.

4.9 Output Time Series

The **Commands...Output Time Series** menu inserts commands for outputting time series.



Commands that operate on time series are listed first and commands that set global configuration values (e.g., output period) are listed at the end of the menu.

Using the output commands allows the results of processing to be saved but increases processing time. If commands are processed repeatedly during analysis or debugging, the following steps may be taken to increase overall efficiency:

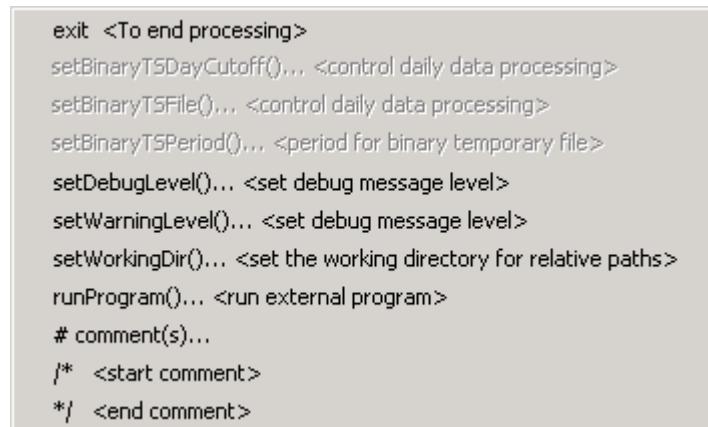
1. Output commands that produce output files are not executed if the **Commands** list is processed with **Run... All Commands (ignore output commands)** or **Run...Selected Commands (ignore output commands)**. Therefore, use this menu choice to ignore the output commands.
2. Only selected commands are processed. Therefore select all but the output commands.
3. Use an exit control command before output commands to skip the output commands. This command can then be deleted or commented out when not needed.
4. Commands can be converted to comments using the **Commands** menu or the popup menu that is displayed when right-clicking on the **Commands** list. Therefore, output commands can be temporarily converted to comments until output needs to be created.

4.11 Commands for Specific Input Types

Depending on the input types that are enabled, additional command menus may be enabled. For example, if the HydroBase input type is enabled, a HydroBase menu will be enabled, and will list commands specific to HydroBase. In particular, commands like `openHydroBase()` are provided to make database connections while processing commands.

4.12 General Commands – Global Settings

The **Commands...General** menu provides choices to insert commands to set parameters that affect the behavior of other commands.



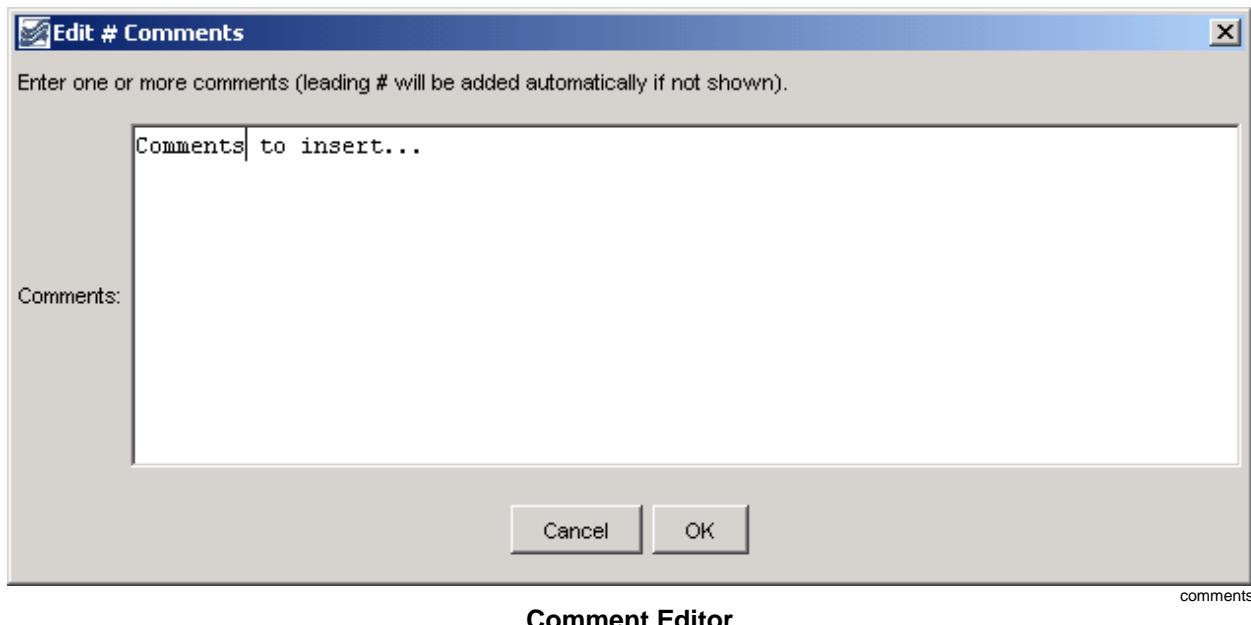
Menu_Commands_General

Commands...General Menu

These commands can be repeated and the effects of a command are in effect after it is encountered until another command changes the setting.

4.12.1 Inserting # Comments

Comments are inserted by selecting a line in the **Commands** list and then selecting **Commands...General...# Comment(s)**. The comments will be inserted before the selected commands. Unlike most other command editors, multiple command lines can be selected. The interface will automatically insert the # character. The following dialog is used to edit comments.



Comment Editor

4.12.2 Start /* */ Comments

Multiple commands can be commented using the following notation:

```
/*
commented lines
commented lines
*/
```

This syntax is consistent with a number of programming languages, including C, C++, and Java, and can be used to quickly disable multiple commands. Use the **Commands...General.../* <start comment>** menu to start a comment above the selected command. Matching start and end comments should be inserted. See also the exit control command.

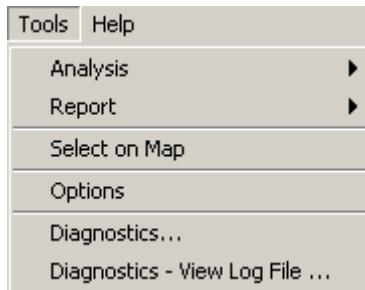
4.12.3 End /* */ Comments

Use the **Commands...General...*/<end comment>** menu to end a multi-line comment in commands.

Tools

Version 06.10.06, 2005-08-05, Color, Acrobat Distiller

This chapter discusses the tools available under the **Tools** menu. The **Tools** menu lists tools that perform additional analysis on time series that are selected in the **Time Series Results** list. These features are similar to the **Results** menu features in that a level of additional analysis is performed to produce the data product. Tools may or may not correspond to commands – often tools internally execute the features available in commands, in order to implement a more complicated analysis.

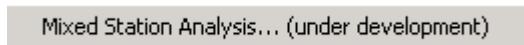


Tools Menu

Menu_Tools

5.1 Analysis Tools

Analysis tools analyze time series and typically produce an output report.



Tools...Analysis Menu

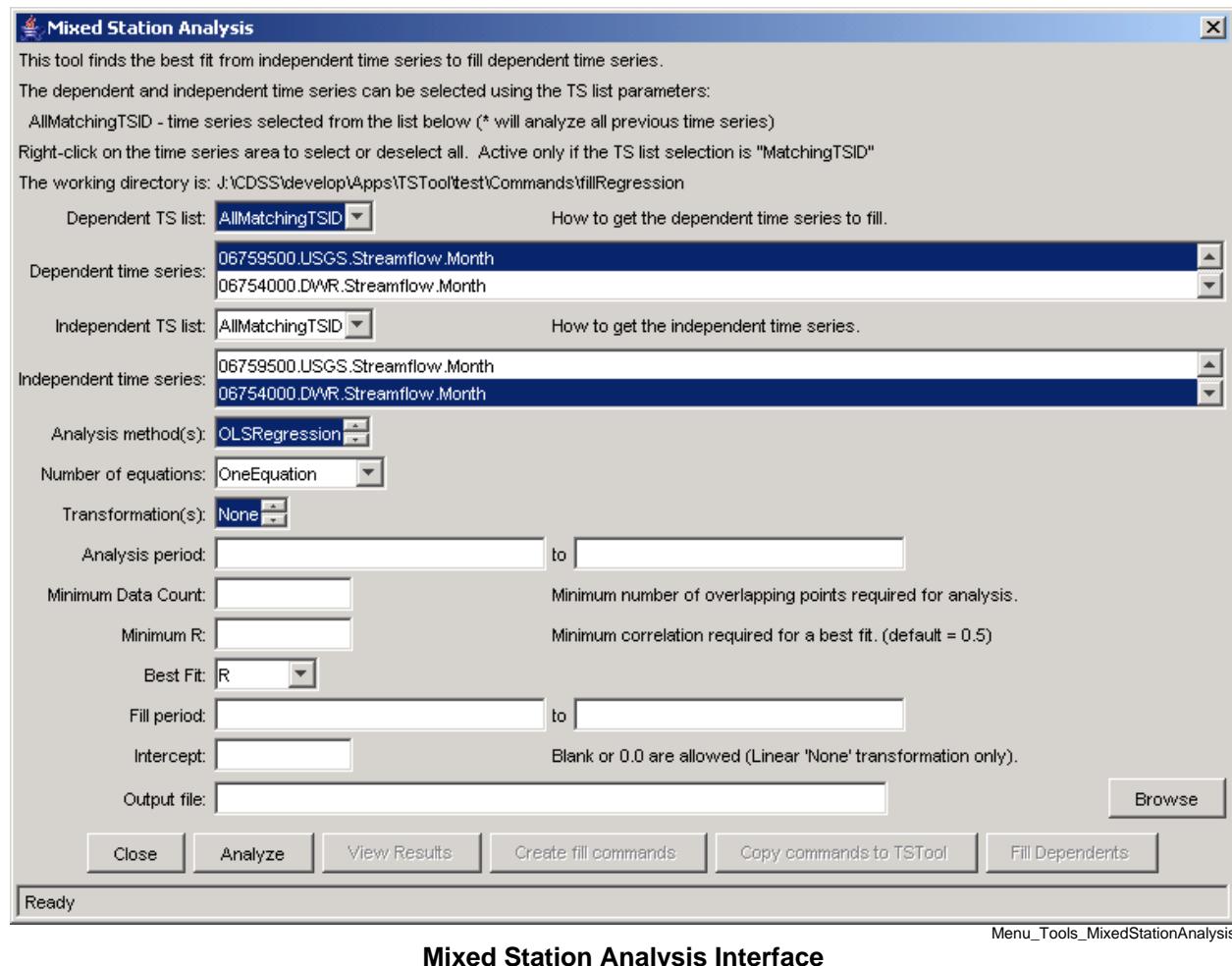
Menu_Tools_Analysis

Currently, only the Mixed Station Analysis analysis tool is available, and it is under development (see the next section).

5.1.1 Mixed Station Analysis

This tool is under development.

The Mixed Station Analysis tool is an interactive tool that tries to find the best combination of time series necessary to fill data using regression or the MOVE2 method. The optimal results can then optionally be used as parameters for the `fillMixedStation()` command. The following figure illustrates the Mixed Station Analysis tool.



Mixed Station Analysis Interface

5.2 Report Tools

Report tools analyze time series, typically creating a summary report.

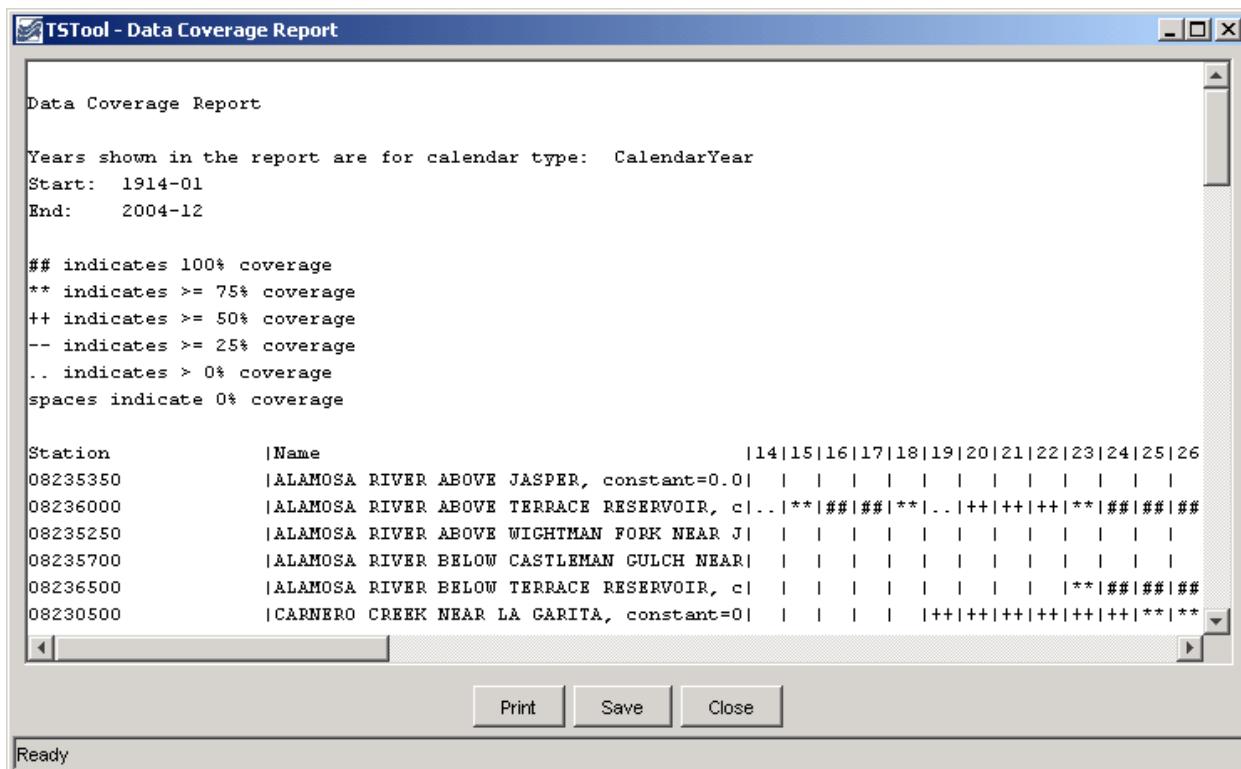


Tools...Report Menu

Menu_Results_Report

5.2.1 Data Coverage by Year Report

The **Tools...Report - Data Coverage by Year** menu processes the time series that have been selected and produces a report similar to the following (abbreviated). This report is useful, in particular, for evaluating data availability for multiple time series over a period. Although effort has been taken to make the report as compact as possible, it will likely need to be printed in landscape format on a large paper size.



Data Coverage by Year Report

Menu_Results_Report_DataCoverage

5.2.2 Data Limits Summary Report

The **Tools...Report - Data Limits Summary** menu processes the time series that have been selected and produces a report similar to the following (abbreviated). The data limits summary for each time series is included. This report is useful, in particular, for evaluating data availability for specific time series. Currently, only monthly time series have detailed summaries. All other data intervals shown overall period summaries. The value -999 is used to indicate missing data.

TSTool - Data Limits Report

DATA LIMITS REPORT

ALAMOSA RIVER ABOVE JASPER
08235350.USGS.Streamflow.Month

Time series: 08235350.USGS.Streamflow.Month (ACFT)
Monthly limits for period 1914-01 to 2004-12 are:

Month	Min	MinDate	Max	MaxDate	Sum	# Miss.	% Miss.	# Not Miss.	% Not Miss.	Mean
Jan	-999.0		-999.0		-999.0	91	100.00	0	0.00	-999.0
Feb	-999.0		-999.0		-999.0	91	100.00	0	0.00	-999.0
Mar	-999.0		-999.0		-999.0	91	100.00	0	0.00	-999.0
Apr	2614.3	1999-04	3011.0	1997-04	5625.2	89	97.80	2	2.20	2812.6
May	14098.7	1999-05	20592.7	1998-05	71489.3	87	95.60	4	4.40	17872.3
Jun	4159.4	1996-06	28433.5	1997-06	74599.4	87	95.60	4	4.40	18649.9
Jul	2806.7	1996-07	8362.4	1999-07	23155.4	87	95.60	4	4.40	5788.8
Aug	1013.6	1996-08	6894.6	1999-08	20691.9	86	94.51	5	5.49	4138.4
Sep	862.8	1996-09	4228.8	1997-09	13011.8	86	94.51	5	5.49	2602.4
Oct	1374.6	1999-10	1422.2	1998-10	2796.7	89	97.80	2	2.20	1398.4
Nov	224.1	1998-11	224.1	1998-11	224.1	90	98.90	1	1.10	224.1
Dec	-999.0		-999.0		-999.0	91	100.00	0	0.00	-999.0
Period	224.1	1998-11	28433.5	1997-06	211593.8	1065	97.53	27	2.47	7836.8

Print Save Close

Ready

Data Limits Summary Report

Menu_Report_DataLimits

5.2.3 Month Summary Reports

The **Tools...Report - Month Summary** menus process the time series that have been selected and produces a report similar to the following (abbreviated). This report is similar to default summary output for monthly time series; however, it is applied to shorter data intervals, including minute, hour, and day interval. Values are first accumulated to daily values (by averaging the values in a day if the **Daily Means** report is chosen or by totaling the values in a day if the **Daily Totals** report is chosen). For example, use total for precipitation and means for average flows or daily temperature. The daily values are then further accumulated to produce monthly values, again using means or totals. The report includes a header for the time series, the report, and footnotes. Values are only shown if full data are available for a month and statistics are computed using only complete months.

TSTool - Monthly Summary Report (Daily Means)

MONTHLY SUMMARY REPORT (Daily Means)

Time Series Identifier	= 08235350.USGS.Streamflow.Day
Description	= ALAMOSA RIVER ABOVE JASPER
Data source	= USGS
Data type	= Streamflow
Data interval	= Day
Data units	= CFS
Period	= 1995-07-01 to 1999-10-31
Orig./Avail. period	= 1995-07-01 to 1999-10-31

Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct
1995	NC	NC	NC	NC	NC	NC	NC	92.45	48.10	NC
1996	NC	NC	NC	NC	290.00	69.90	45.65	16.48	14.50	NC
1997	NC	NC	NC	50.60	308.45	477.83	108.39	78.97	71.07	NC
1998	NC	NC	NC	NC	334.90	278.30	86.55	36.48	24.17	23.1:
1999	NC	NC	NC	43.93	229.29	427.63	136.00	112.13	60.83	22.3:
Min	NC	NC	NC	43.93	229.29	69.90	45.65	16.48	14.50	22.3:
Max	NC	NC	NC	50.60	334.90	477.83	136.00	112.13	71.07	23.1:
Mean	NC	NC	NC	47.27	290.66	313.42	94.15	67.30	43.73	22.7:
SDev	NC	NC	NC	4.71	44.87	183.13	38.14	39.72	23.96	0.5:
MMxD	NC	NC	45.00	125.75	534.75	492.00	261.40	114.80	94.20	38.21
MMnD	NC	NC	20.00	35.50	75.00	183.00	71.60	46.60	26.80	21.21

Notes:

- Years shown are calendar years.
- Annual values and statistics are computed only on non-missing data.
- NC indicates that a value is not computed because of missing data or the data value itself is missing.
- Statistics are for values shown in the main table except:
- MMxD are the means of the maximum daily values in a month.
- MMnD are the means of the minimum daily values in a month.

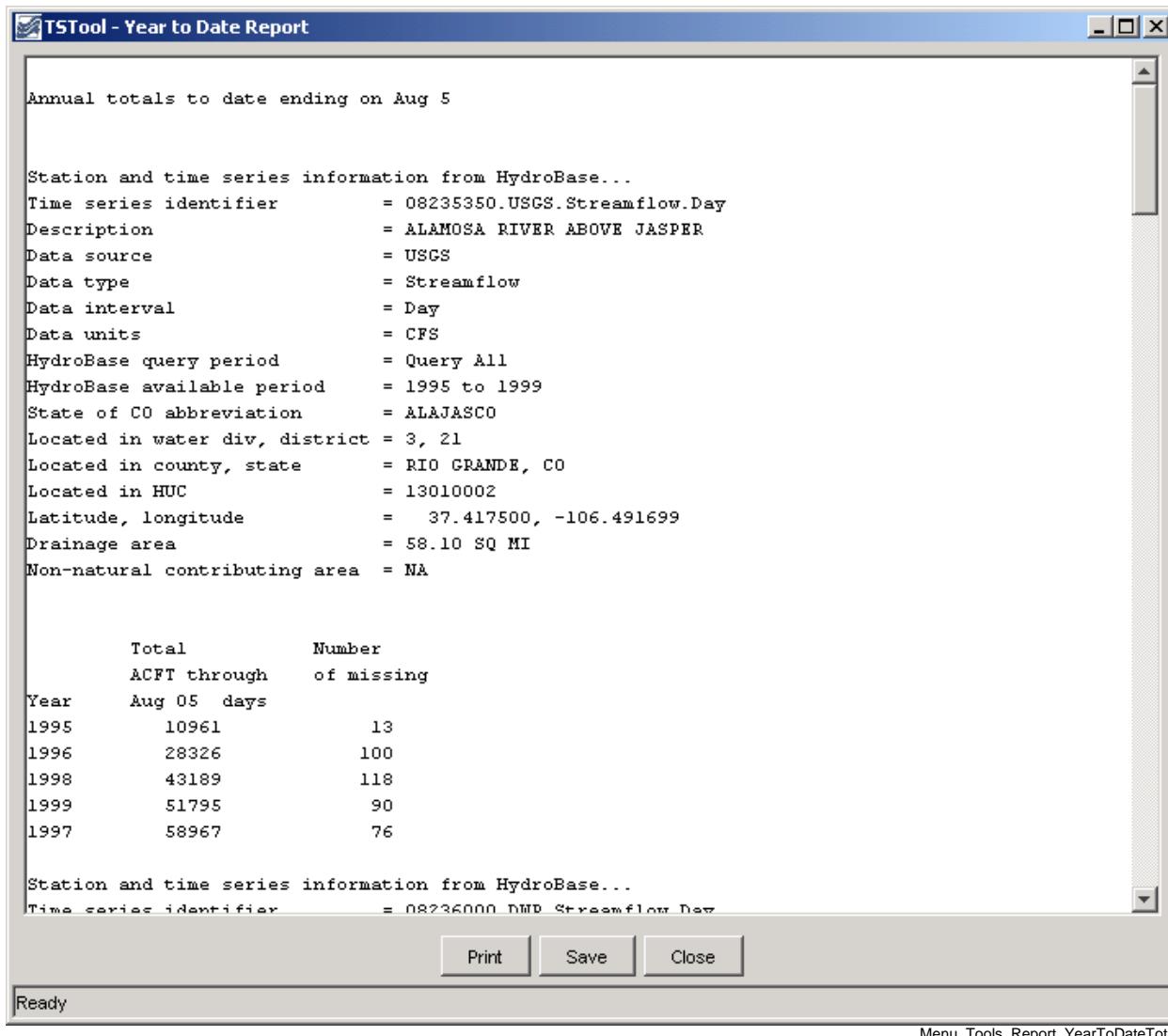
Ready

Monthly Summary (Daily Mean) Report

Menu_Tools_MonthSummaryDailyMean

5.2.4 Year to Date Total Report

The **Tools...Report - Year to Date Total** menu processes the time series that have been selected and produces a report similar to the following (abbreviated). This report is useful, in particular, for comparing on a volumetric basis the different years of a time series over a full period. The year to date volumes are sorted; to find a particular year, use the **Search** button on the report display. The report information can then be used, for example, to select time series traces for analysis and output. Currently, this report can only be used to process daily CFS data. Real-time data can be analyzed by first converting to a daily interval using the `changeInterval()` command. **Warning:** some years may have no data at the beginning of a year and the corresponding year-to-date totals will consequently be zero. Refer to the data coverage and data limit reports for more detail.



Menu_Tools_Report_YearToDateTotal

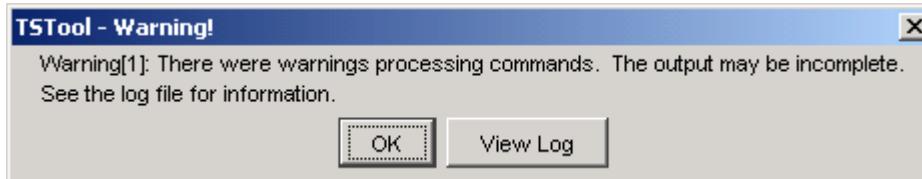
Year to Date Total Report

5.3 Map Tools

The **Tools...Show on Map** button is enabled when a map is displayed (using **View...Map**) and time series are listed in the upper right part of the main window. The locations corresponding to selected or all time series in this list can be displayed on the map. See **Chapter 8 – Using the Map** for more information.

5.4 Diagnostics

Diagnostics features are useful for troubleshooting. When an error occurs, a small warning dialog may be displayed, as shown in the following figure:



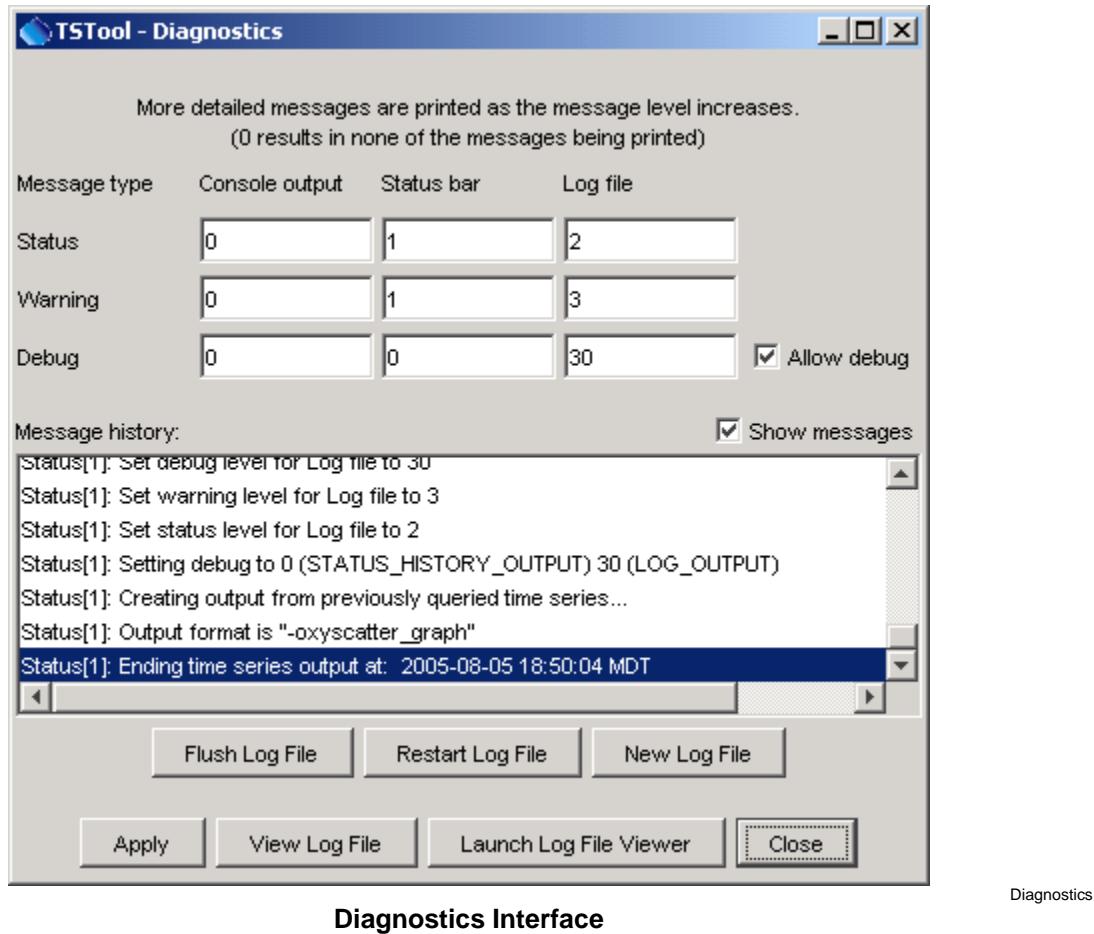
Diagnostics_Warning

Example Warning Message Dialog

If results are not as expected, also review the messages in the status bar at the bottom of the main or secondary windows.

5.4.1 Diagnostics Settings

The **Tools...Diagnostics** menu item displays the **Diagnostics** dialog, which is used to set message levels and view messages as the application runs. The **Diagnostics** dialog (see the following figure) can be used to evaluate a problem.



Diagnostics

The settings at the top of the dialog are used to specify the level of detail for messages printed to the console window, the status area at the bottom of the main window (and the **Diagnostics** dialog), and the log file. The log file contains warning, status, and debug messages, many of which are not normally displayed in the main interface. The log file is created in the *logs* directory under the installation directory. The **Diagnostics** interface features are as follows:

Status, Warning, Debug

Enter integer values, with larger numbers resulting in more output and slower performance. Zero indicates no output. If troubleshooting, a good guideline is to set the debug level to 10 or 30 (and select the **Allow Debug** checkbox). The default settings are often enough for normal troubleshooting and result in good software performance.

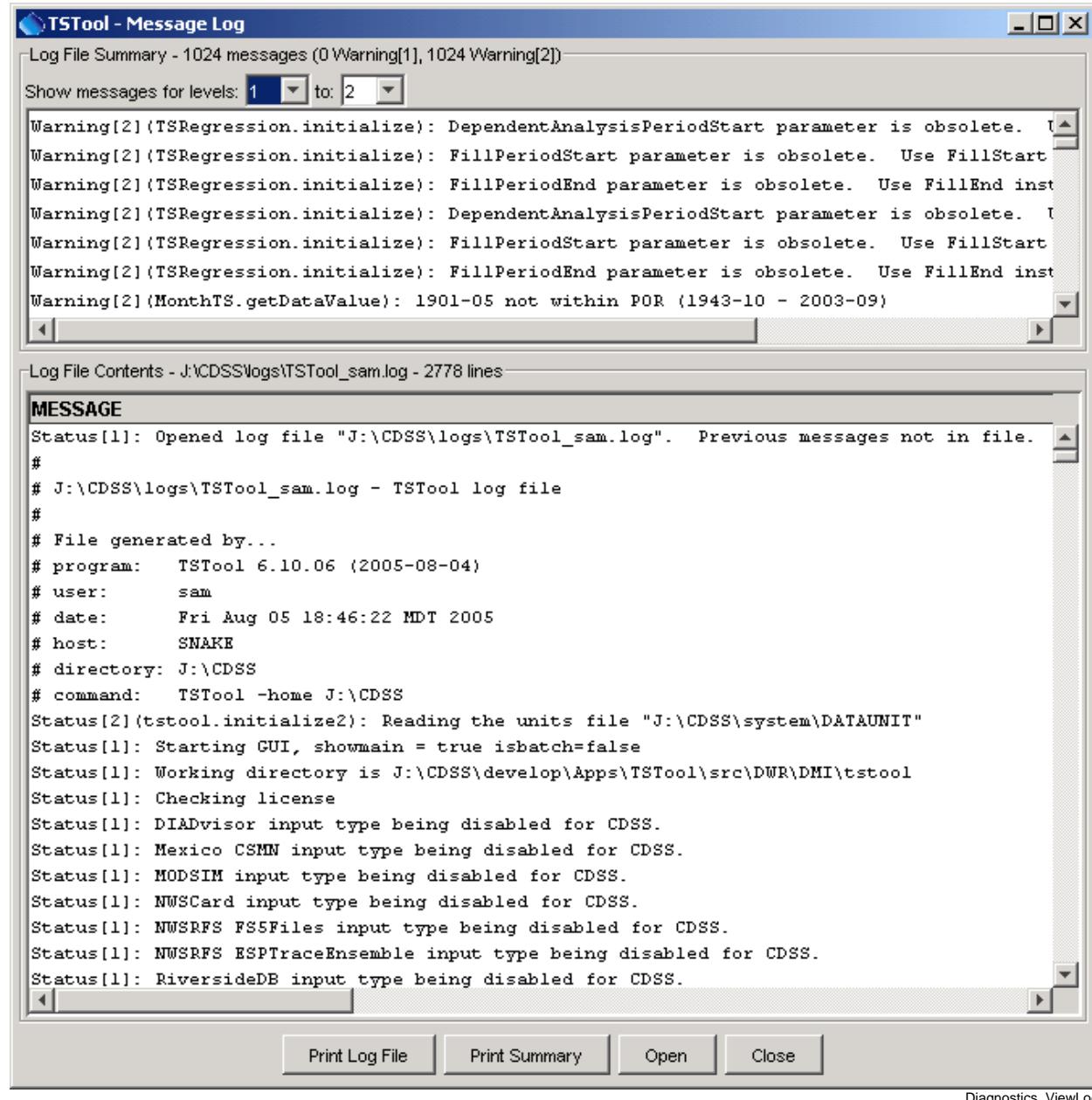
Allow Debug

Select to enable debug messages. Turning on debug messages will significantly slow down the software.

Show Messages	Select to display messages in the Diagnostics window.
Flush Log File	Force messages to be written to the log file. Messages can be buffered in memory and may not otherwise immediately be written to the log file.
Restart Log File	Restart the log file. This is useful if a long session has occurred and troubleshooting will occur on new actions.
New Log File	Open a new log file, with a new name.
Apply	Apply the settings in the Diagnostics dialog.
View Log File	View the log file in an integrated window. The View Log File button will be enabled if the log file has been opened.
Launch Log File Viewer	View the log file using a viewer from the operating system. On Windows computers, Notepad will be used.
Close	Apply the settings in the Diagnostics dialog and close the window.

5.4.2 Diagnostics – View Log File

The **Tools...Diagnostics – View Log File** menu item displays the integrated log file viewer. Selecting this menu item is equivalent to selecting the **View Log File** button in the **Diagnostics** dialog. The log file viewer will be displayed in a window as shown in the following figure:



Log File Viewer Window

Diagnostics_ViewLog

The log file messages can be scrolled. To find a string in the log file, right-click and select the **Find** menu item. The information in the log file can also be copied and pasted into email, when contacting support.

6

Examples of Use

Version 06.10.06, 2005-08-05, Color, Acrobat Distiller

This chapter provides examples for retrieving and manipulating time series data using TSTool. The heading for the section gives an indication of the example purpose. Where appropriate, input and output types are indicated to simplify finding a useful example. General examples are listed first, followed by more complex examples.

6.1 General Examples

6.1.1 General – One-time Time Series Display/Analysis

6.1.2 General – Reproducing an Analysis with a Commands File

6.2 Model Data Processing Examples

6.2.1 Modeling – Preparing Model Files Using a Commands File

6.2.2 Modeling – Processing End of Month Reservoir Data (Input=HydroBase, Output=StateMod)

6.2.3 Modeling – Filling Reservoir Targets with a Pattern File (Input=HydroBase, Output=StateMod)

6.2.4 Modeling – Using a List File to Automate Time Series Processing (Input=HydroBase, Output=StateMod)

6.2.5 Modeling – Processing Frost Dates (Input=HydroBase, Output=StateCU)

6.2.6 Modeling – Filling Streamflow using MOVE2 (Input=HydroBase)

6.3 Time Series Trace Examples

6.3.1 Time Series Traces – Comparing Historical and Current Conditions (Input=HydroBase)

6.4 Time Series Product Examples

6.4.1 Time Series Product – Using TSTool to Display Graphs from Another Software Application

6.4.2 Time Series Product – Automating Graphs for Compare Observed and Simulated Time Series

6.1 General Examples

This section includes examples related to general TSTool use, which may be appropriate for general users.

6.1.1 General – One-time Time Series Display/Analysis

The following example session illustrates how to query time series data for display, analysis, and viewing.

1. Start TSTool. If the HydroBase or other database input types are enabled, select a database (see [Section 3.2- Select HydroBase Dialog](#), for example).
2. To manipulate time series in any way, first select the time series of interest (see [Section 3.3 - Main Interface](#)). Pick appropriate input type, data type, and filter information. Press **Get Time Series List**

to list the available time series. After pressing **Get Time Series List**, a list of time series will be shown in the upper-right corner of the interface.

3. Select one or more time series from the list and transfer to the **Commands** list as time series identifiers. Time series identifiers are explained in **Chapter 2 – Introduction**.
4. Press the **Run All Commands** button to query the time series. They should now be listed in the **Time Series Results** list.
5. Use the **Results** and **Tools** menus to view the time series or the **File...Save...Time Series As** menus to export as files. For example, display a line graph (using **Results...Graph - Line**) and then view the time series as a summary or table.
6. Go back to the **Commands** list and use the **Commands** menu to manipulate time series. For example:
 - insert a `fillInterpolate()` command to fill data
 - insert a `cumulate()` or `runningAverage()` command to transform the data
7. Repeat steps 4 – 5 to process and view time series.

6.1.2 General – Reproducing an Analysis with a Commands File

To reproduce an analysis, save the commands shown in the **Commands** list to a commands file and then reload and run the commands later. For example, assuming that steps similar to the previous section have been executed:

1. Use the **File...Save...Commands As** menu to save the commands. It is recommended that commands files be saved with a file extension *.TSTool*.
2. Exit TSTool and restart (alternatively, clear the commands using the **Clear Commands** button).
3. Use **File...Open...Commands File**. Select the file that you previously saved.
4. Then run the commands by pressing the **Run All Commands** button. Display the results using the **Results** menu.

As the above example shows, reproducing an analysis consists of saving a commands file that can be reused later. The main complications in this approach are that the environment in which the commands are run may change over time. For example if using a HydroBase database, the database version, ODBC data source name, database host, or working directory may be differ between computers. It is recommended that commands use directories relative to a working directory and that the working directory is defined consistently on different computers that will use the commands. Using paths relative to the working directory will consequently allow commands files to be portable. TSTool will internally set the working directory that the directory where a commands file is opened or saved.

6.2 Model Data Processing Examples

Most computer models require data that adhere to a consistent format. TSTool facilitates processing model data files with features that:

- allow a specific period of record to be output
- fill missing data
- produce time series in a specific order

The following examples illustrate how to use TSTool to process model data.

6.2.1 Modeling – Preparing Model Files Using a Commands File

To prepare model files, multiple commands will usually process numerous time series. Modelers often run TSTool in batch mode from a command shell using a command like:

```
tstool -commands commandsfile
```

However, it is recommended that commands files be run using the GUI, if possible, in order to take advantage of additional error-checking and feedback features.

TSTool provides command editor dialogs for every command and helps ensure the integrity of commands by searching for input time series for each command. An effort has been made to make the current TSTool recognize and process old commands. However, there have been some changes that will require updates to commands. It is recommended that old command files be migrated to the new syntax using the following approaches:

1. Be familiar with this documentation and, in particular, the command reference.
2. Run an existing commands file and review the log file for warnings about commands that need to be updated. Then edit the commands in the GUI (see the next step).
3. Open an existing commands file using the **File...Open...Commands File** menu and then edit commands using the editor dialogs. These dialogs will help to convert old commands to the new syntax. Most commands files focus on a particular data type and filling technique. Therefore most updates will generally involve only a few changes.

A number of commands have been added/enhanced to promote reuse of commands files in both batch and GUI run modes. For example, the `openHydroBase()` indicates whether the command is active for batch and GUI run modes. Choosing the correct setting simplifies exchange of command files between users and operating environments.

When querying time series, select a subset of the commands for intermediate work to verify filling or other manipulation. General commands (e.g., `setOutputPeriod()`) may be required even if a subset of time series is being processed.

TSTool by default reads all available data. However, the `setQueryPeriod()` is available to limit the period that is read. The `setOutputPeriod()` is now used only to control the period for output products.

6.2.2 Modeling – Processing End of Month Reservoir Data (Input=HydroBase, Output=StateMod)

The commands file shown below was originally generated by an older version of TSTool (before TSTool 05.xx.xx).

```
10/1974 9/1991
-units DFLT
-wy
-o statemod
-o colouptar
# CBT SHADOW MTN GRAND L
Fillconst(513695.USBR.ResEOM.MONTH.,0)
513695.USBR.ResEOM.MONTH.
# CBT GRANBY RESERVOIR
Fillconst(514620.USBR.ResEOM.MONTH.,0)
514620.USBR.ResEOM.MONTH.
# DILLON RESERVOIR
Fillconst(364512.USBR.ResEOM.MONTH.,0)
364512.USBR.ResEOM.MONTH.
# GREEN MOUNTAIN RESERVOIR
Fillconst(363543.USBR.ResEOM.MONTH.,0)
363543.USBR.ResEOM.MONTH.
# RIFLE GAP RESERVOIR
Fillconst(393508.USBR.ResEOM.MONTH.,0)
393508.USBR.ResEOM.MONTH.
```

The equivalent commands using new commands syntax are shown below. Note that only a few changes are required to update the file. Although the resulting commands file may be longer in some cases, the fill operations are now separate from the initial query, allowing multiple fill commands to be applied sequentially and also allowing the intermediate time series to be queried and viewed independently in the GUI. The new commands structure also allows the same results to be obtained in multiple ways. For example, the zero-filled time series can be created by using `newTimeSeries()`, or by utilizing `setIncludeMissingTS()`, `fillConstant()`, and `setConstant()`. **Note that since the original commands file was created, greater definition has been given to data sources in time series identifiers. The TSTool GUI will always supply a data source from the HydroBase database and may need to be used to determine the proper data source for a time series identifier.**

```
# Reservoir target file commands
# Each reservoir needs a minimum (zero) and maximum time series (from HydroBase)
setOutputPeriod(10/1974,9/1991)
setOutputYearType(Water)
# CBT SHADOW MTN GRAND L
TS_ShadowMtn = newTimeSeries(513695.USBR.ResEOM.MONTH., "CBT SHADOW MTN GRAND L", *, *, AF, 0.0)
513695.USBR.ResEOM.MONTH.
# CBT GRANBY RESERVOIR
TS_Granby = newTimeSeries(514620.USBR.ResEOM.MONTH., "CBT GRANBY RESERVOIR", *, *, AF, 0.0)
514620.USBR.ResEOM.MONTH.
# DILLON RESERVOIR
TS_Dillon = newTimeSeries(364512.DWB.ResEOM.MONTH., "DILLON RESERVOIR", *, *, AF, 0.0)
364512.DWB.ResEOM.MONTH.
# GREEN MOUNTAIN RESERVOIR
TS_GreenMtn = newTimeSeries(363543.USBR.ResEOM.MONTH., "GREEN MOUNTAIN RESERVOIR", *, *, AF, 0.0)
363543.USBR.ResEOM.MONTH.
# RIFLE GAP RESERVOIR
TS_RifleGap = newTimeSeries(393508.USBR.ResEOM.MONTH., "RIFLE GAP RESERVOIR", *, *, AF, 0.0)
393508.USBR.ResEOM.MONTH.
writeStateMod("colouptar", *)
```

6.2.3 Modeling – Filling Reservoir Targets with a Pattern File (Input=HydroBase, Output=StateMod)

The following example illustrates an old command file (before TSTool 05.xx.xx) for creating a StateMod reservoir target file, using pattern filling.

```

# Phase IIIb modifications
#   Fill missing data using water district indicator gages determined in demands runs
#   Fill with historical monthly average if no wetness pattern average available
#   Set start dates for reservoirs in March of year listed in Ray A fax (9/8/98)
#
10/1908 9/1998
-ignorezero
-units DFLT
-wy
-filldata fill.pat
-fillhistave
-ostatemod
-o ../ym2001H.tar
# ELKHEAD RESERVOIR
443902.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(443902.StateMod.RSTO.MONTH.../ts_files/443902.stm,9251000,0,03/1975)
# ALLEN BASIN RESERVOIR
583500.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(583500.StateMod.RSTO.MONTH.../ts_files/583500.stm,9239500,0,03/1909)
# FISH CREEK RESERVOIR
583508.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(583508.StateMod.RSTO.MONTH.../ts_files/583508.stm,9239500,0,03/1956)
# LESTER CREEK RESERVOIR
583521.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(583521.StateMod.RSTO.MONTH.../ts_files/583521.stm,9239500,0,03/1975)
# STILLWATER RESERVOIR 1
583540.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(583540.StateMod.RSTO.MONTH.../ts_files/583540.stm,9239500,0,03/1939)
# LAKE CATAMOUNT
583631.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(583631.StateMod.RSTO.MONTH.../ts_files/583631.stm,9239500,0,03/1974)
# STEAMBOAT LAKE
583787.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern(583787.StateMod.RSTO.MONTH.../ts_files/583787.stm,9239500)
#fillpattern_setconstbefore(583787.StateMod.RSTO.MONTH.../ts_files/583787.stm,9239500,0,03/1974
)
# STAGECOACH RESERVOIR
584213.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(584213.StateMod.RSTO.MONTH.../ts_files/584213.stm,9239500,0,03/1988)
# YAMCOLO RESERVOIR
584240.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
fillpattern_setconstbefore(584240.StateMod.RSTO.MONTH.../ts_files/584240.stm,9239500,0,03/1980)
...some commands omitted...

```

The same results are obtained using new commands with the following commands file.

```

# Phase IIIb modifications
#     Fill missing data using water district indicator gages determined in demands runs
#     Fill with historical monthly average if no wetness pattern average available
#     Set start dates for reservoirs in March of year listed in Ray A fax (9/8/98)
#
#
# Updated by SAM (2001-04-02) during TSTool testing. Made the following
# changes:
#
# Read time series then use fillPattern() (2 steps rather than 1)
# Comment out -o options and replace with writeStateMod() at end
# Change -fillhistave to fillHistMonthAverage(*) at end
# Change -ignorelezero to setIgnoreLEZero(true)
# remove -units
# replace -filldata with setPatternFile()
# change -wy to setOutputYearType(Water)
# change period to setOutputPeriod()
# Add fillConstant(*,0.0) at end
#
setOutputPeriod(10/1908,9/1998)
setIgnoreLEZero(true)
setOutputYearType(Water)
setPatternFile(fill.pat)
# ELKHEAD RESERVOIR
443902.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 443902 = readTimeSeries(443902.StateMod.ResEOM.MONTH.../ts_files/443902.stm)
fillPattern(443902,9251000)
setConstantBefore(443902,0,03/1975)
# ALLEN BASIN RESERVOIR
583500.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 583500 = readTimeSeries(583500.StateMod.ResEOM.MONTH.../ts_files/583500.stm)
fillPattern(583500,9239500)
setConstantBefore(583500,0,03/1909)
# FISH CREEK RESERVOIR
583508.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 583508 = readTimeSeries(583508.StateMod.ResEOM.MONTH.../ts_files/583508.stm)
fillPattern(583508,9239500)
setConstantBefore(583508,0,03/1956)
# LESTER CREEK RESERVOIR
583521.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 583521 = readTimeSeries(583521.StateMod.ResEOM.MONTH.../ts_files/583521.stm)
fillPattern(583521,9239500)
setConstantBefore(583521,0,03/1975)
# STILLWATER RESERVOIR 1
583540.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 583540 = readTimeSeries(583540.StateMod.ResEOM.MONTH.../ts_files/583540.stm)
fillPattern(583540,9239500)
setConstantBefore(583540,0,03/1939)
# LAKE CATAMOUNT
583631.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 583631 = readTimeSeries(583631.StateMod.ResEOM.MONTH.../ts_files/583631.stm)
fillPattern(583631,9239500)
setConstantBefore(583631,0,03/1974)
# STEAMBOAT LAKE
583787.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 583787 = readTimeSeries(583787.StateMod.ResEOM.MONTH.../ts_files/583787.stm)
fillPattern(583787,9239500)
#fillpattern_setconstbefore(583787.StateMod.ResEOM.MONTH.../ts_files/583787.stm,9239500,0,03/1974)
# STAGECOACH RESERVOIR
584213.DelimFile.RSTO.MONTH.../ts_files/zero_dat.del
TS 584213 = readTimeSeries(584213.StateMod.ResEOM.MONTH.../ts_files/584213.stm)
fillPattern(584213,9239500)
setConstantBefore(584213,0,03/1988)
# YAMCOLO RESERVOIR
584240.DelimFile.ResEOM.MONTH.../ts_files/zero_dat.del
TS 584240 = readTimeSeries(584240.StateMod.ResEOM.MONTH.../ts_files/584240.stm)
fillPattern(584240,9239500)
setConstantBefore(584240,0,03/1980)
...some commands omitted...
fillHistMonthAverage(*)
fillConstant(*,0)
writeStateMod(.../ym2001H.tar,*)

```

6.2.4 Modeling – Using a List File to Automate Time Series Processing (Input=HydroBase, Output=StateMod)

It may be desirable to output a StateMod daily time series file given a list of station/structure identifiers. The following example illustrates the commands file to accomplish this task.

```
#  
# Example to illustrate how a list file can be used to generate a StateMod daily  
# time series file with for many structures  
#  
# Read a list file, where the first column is structure identifiers. Try to get  
# daily diversion time series for each structure. If data are not available, create  
# an empty missing time series.  
#  
createFromList(ListFile="structure_list.txt",DataSource=DWR,DataType=DivTotal,  
Interval=Day,InputType=HydroBase,HandleMissingTSHow=DefaultMissingTS)  
#  
# Now write the results to a StateMod file.  
#  
setOutputYearType(Calendar)  
writeStateMod("structure_list.stm",*)
```

6.2.5 Modeling – Processing Frost Dates (Input=HydroBase, Output=StateCU)

Frost dates are special time series consisting of four dates per year. The dates correspond to:

- Last day in spring that the temperature was 28° F
- Last day in spring that the temperature was 32° F
- First day in fall that the temperature was 32° F
- First day in fall that the temperature was 28° F

These specific dates are currently consistent with the HydroBase and StateCU input types. Older versions of TSTool (before version 06.00.00) treated frost dates as a single time series, where the four components were internally manipulated as dates. The add() command and a limited number of features supported manipulating frost date time series. However, other commands could not be used to process the time series. Consequently, display, analysis, and manipulation capabilities were limited.

As of TSTool version 06.00.00, TSTool handles each of the above data items as separate data types and time series, internally treating the values as Julian days from January 1. The StateCU input type, which is used when reading and writing frost date files, converts between Julian days and Month/Year in the file. By handling as four separate numerical time series, all of TSTool's manipulation tools can be used to fill and analyze frost dates. This does require each time series to be specified, whereas before the four were internally handled with a single time series identifier. Because frost dates are internally treated as numerical Julian days, using the generic numerical add() command functionality may result in unexpected output if the time series overlap (Julian days will be added). To avoid this situation, use the fillFromTS(), setFromTS(), or blend() commands when merging multiple time series. The following example illustrates how to process a frost dates file for StateCU:

```

setOutputPeriod(1950,2002)
#
# 0130 - ALAMOSA SAN LUIS VALLEY RGNL
0130.NOAA.FrostDateL28S.Year~HydroBase
0130.NOAA.FrostDateL32S.Year~HydroBase
0130.NOAA.FrostDateF32F.Year~HydroBase
0130.NOAA.FrostDateF28F.Year~HydroBase
# Add Meeker Stations (5484 and 5487)
# then "free" 5487
5484.NOAA.FrostDateL28S.Year~HydroBase
5484.NOAA.FrostDateL32S.Year~HydroBase
5484.NOAA.FrostDateF32F.Year~HydroBase
5484.NOAA.FrostDateF28F.Year~HydroBase
5487.NOAA.FrostDateL28S.Year~HydroBase
5487.NOAA.FrostDateL32S.Year~HydroBase
5487.NOAA.FrostDateF32F.Year~HydroBase
5487.NOAA.FrostDateF28F.Year~HydroBase
fillFromTS(5484.NOAA.FrostDateL28S.Year,5487.NOAA.FrostDateL28S.Year,*,*)
fillFromTS(5484.NOAA.FrostDateL32S.Year,5487.NOAA.FrostDateL32S.Year,*,*)
fillFromTS(5484.NOAA.FrostDateF32F.Year,5487.NOAA.FrostDateF32F.Year,*,*)
fillFromTS(5484.NOAA.FrostDateF28F.Year,5487.NOAA.FrostDateF28F.Year,*,*)
free(TSID="5487*")
#
#
# fillHistYearAverage(*)
#
#
writeStateCU("../\StateCU\Frost2002.stm")

```

6.2.6 Modeling – Filling Streamflow Using MOVE2 (Input=HydroBase)

Data filling is an important activity for modeling. TSTool provides a number of data filling commands, as described in **Section 4.4 – Fill Time Series Data**. Data filling can be accomplished using varying levels of complexity. The approach used for data filling depends on the data type and interval. For example, estimating daily precipitation may be difficult because relationships between daily precipitation time series may not exist. TSTool provides tools for data filling but it does not automatically pick the most appropriate fill methods. Consequently, data filling involves a number of steps:

1. initial review of the data (e.g., using the **Results...Report - Data Coverage by Year** menu, and graphs)
2. review of the spatial proximity of gages using the TSTool **View...Map Interface** capability or GIS software.
3. comparison of candidate time series (e.g., using the **Results...Graph - XY-Scatter** menu)
4. apply data filling commands
5. review final results visually and review time series histories (by right-clicking on a time series in the **Time Series Results** list and selecting **Time Series Properties**)

As indicated above, the data filling approach can be simple or complex. An example of a complex data filling technique is to use the `fillMOVE2()` command on daily streamflow data. In particular, consider the following case:

- Time series 1 (TS1) has a long period of gaged unregulated data (e.g., a headwater): 1900 to 2000
- Time series 2 (TS2) has a shorter period of gaged data with 1920 to 1950 being unregulated and 1950 to 2000 being regulated (e.g., due to the construction of a reservoir)
- The goal is to produce an estimate of unregulated flow for TS2 for the full period 1900 to 2000.

This can be accomplished using the following commands:

```
# Data filling example - assume daily DateValue time series as input
#
# Set the output period so that time series are guaranteed to have a period to fill...
setOutputPeriod(1900-01-01,2000-12-31)
# Read the dependent time series (TS2), which will receive the final result
TS ts2 = readDateValue(InputFile="ts2.txt")
# Read the independent time series (TS1)
TS ts1 = readDateValue(InputFile="ts1.txt")
# Make a copy of the dependent to analyze and fill after 1950...
# Need this because we need to clear out the regulated data
TS ts2_copy = copy(ts2)
# Set the later period to missing in the copy so it will be filled...
replaceValue(ts2_copy,-10000,10000,-999,1950-01-01,*)
# Analyze and fill the early period. Transform the data to log10 and
# use monthly equations. The first pair of dates is the dependent analysis period.
# The second pair of dates is the independent analysis period. The third pair of
# dates is the fill period.
fillMOVE2(ts2,ts1,MonthlyEquations,Log,1920-01-01,1949-12-31,*,*,*,1919-12-31)
# Analyze and fill the later period. Transform the data to log10 and
# use monthly equations...
fillMOVE2(ts2_copy,ts1,MonthlyEquations,Log,1920-01-01,1949-12-31,*,*,1950-01-01,*)
# Now set the later period into the main time series...
setFromTS(ts2,ts2_copy,1950-01-01,*)
```

The above example illustrates a somewhat complicated situation where data filling is facilitated by the features of the `fillMOVE2()` command. If the `fillMOVE2()` command is not appropriate, then the `fillRegression()` or other commands can be applied. In some cases, it may be appropriate to fill different parts of the period using different independent time series. A simpler approach may involve only a single filling step (e.g., fill the entire period using a single `fillRegression()` command).

6.3 Time Series Trace Examples

The general term *time series traces* refers to groups of a time series, often shown in overlapping fashion. Common ways to create traces are:

- Split time series into N-year lengths and shift to overlap.
- Run a model multiple times with different input, in order to generate many possible outcomes.
- Generate synthetic data to use as input to a model.

Several TSTool commands have been implemented to create and process time series traces. TSTool manages trace data by using the same identifier for all traces and internally keeping a trace (sequence) number, which is the same as the starting year of the trace. An extension to the time series identifier convention may be made in the future to allow a trace to be uniquely identified by a time series identifier. Currently, you must use the GUI and its time series selection features to select individual traces.

6.3.1 Time Series Traces – Comparing Historical and Current Conditions (Input=HydroBase)

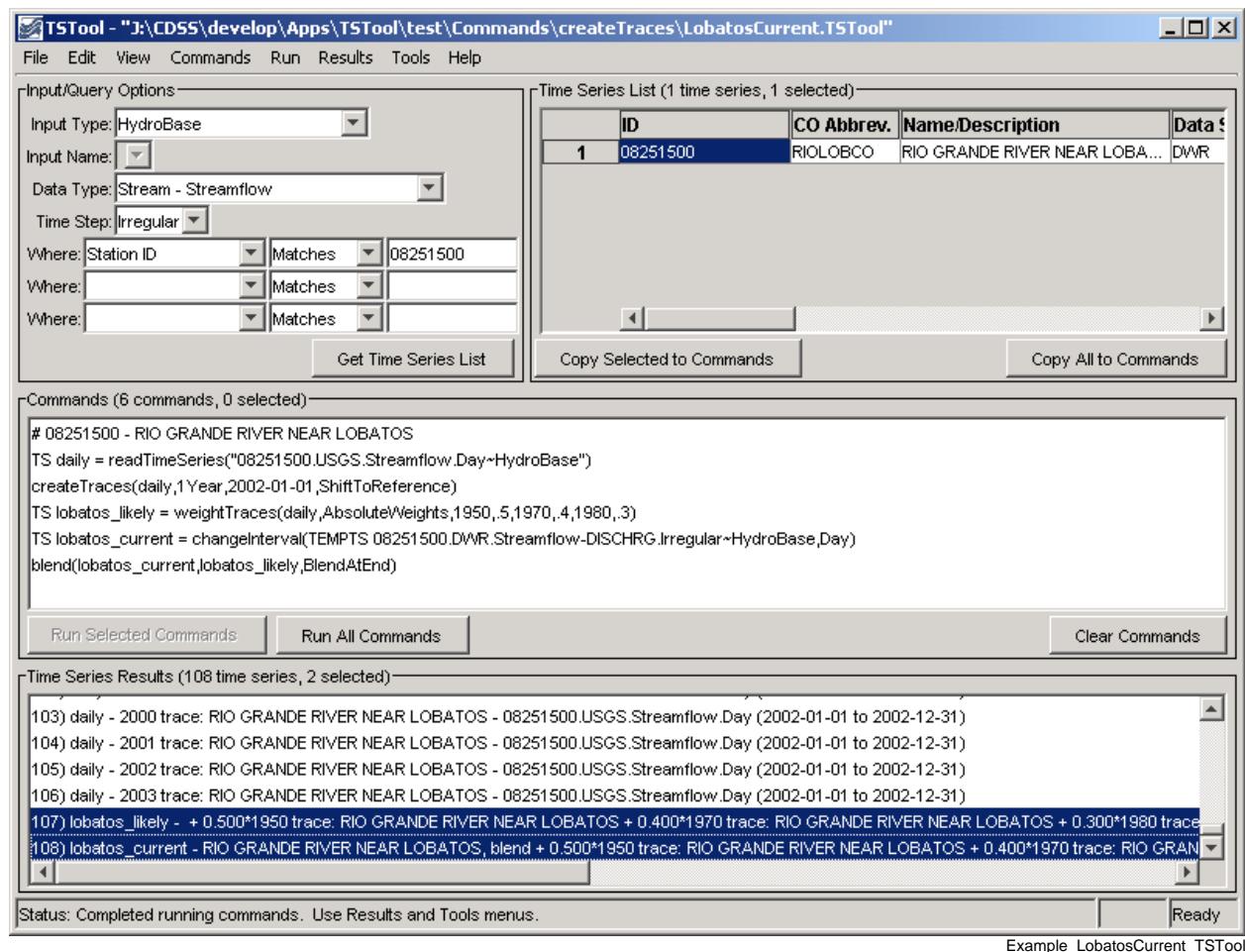
The following commands file illustrates how historic time series traces can be plotted on top of real-time data. The features illustrated in the example were implemented to help determine an estimate of future flow based on current conditions.

```

#
# These commands query historic and real-time data at the lobatos gage and
# compute a weighted "best-guess" for the flows at lobatos for the remainder
# of the current year.
#
# First get the historic daily lobatos gage and convert to traces. Shift each trace to
# 01/01/2001 so that the data can overlay the current values. Internally, each trace
# has the identifier 08251500.DWR.QME.Day (which is used in commands) and a
# hidden sequence number that matches the year of the start of the trace.
# Note that the TEMPTS key word is used in the command to query the daily
# time series so the time series is discarded after conversion to traces.
TS daily = readTimeSeries("08251500.USGS.Streamflow.Day~HydroBase")
createTraces(daily,1Year,01/01/2002,ShiftToReference)
#
# Now weight the traces using representative historic years. This uses absolute
# weights which means that the total of the weights can be different than 1.0. An
# option to use normalized weights may be added to force the weights to scale to 1.0,
# even with missing data.
#
TS lobatos_likely = weightTraces(daily,AbsoluteWeights,1950,.5,1970,.4,1980,.3)
#
# Now query the current (real-time) flows. HydroBase may only hold a few weeks or months
# of data.
#
# Uncomment the following line to see the actual real-time values (slower)
# 08251500.DWR.RT_rate.Real-time.DISCHRG
TS lobatos_current = changeInterval(TEMPTS 08251500.DWR.RT_rate.Real-time.DISCHRG,Day)
#
# After the above commands are executed, time series in memory will include the traces and
# the current time series. You can select only the time series of interest and plot
# OR select many time series and then disable/enable in the plot. You may need to use
# both approaches to find appropriate time series to weight.
blend(lobatos_current,lobatos_likely,BlendAtEnd)

```

The results of processing the above commands in TSTool are a list of the traces, a weighted time series (based on three traces), and the current daily data, all at the same streamflow gage, as shown in the following figure.

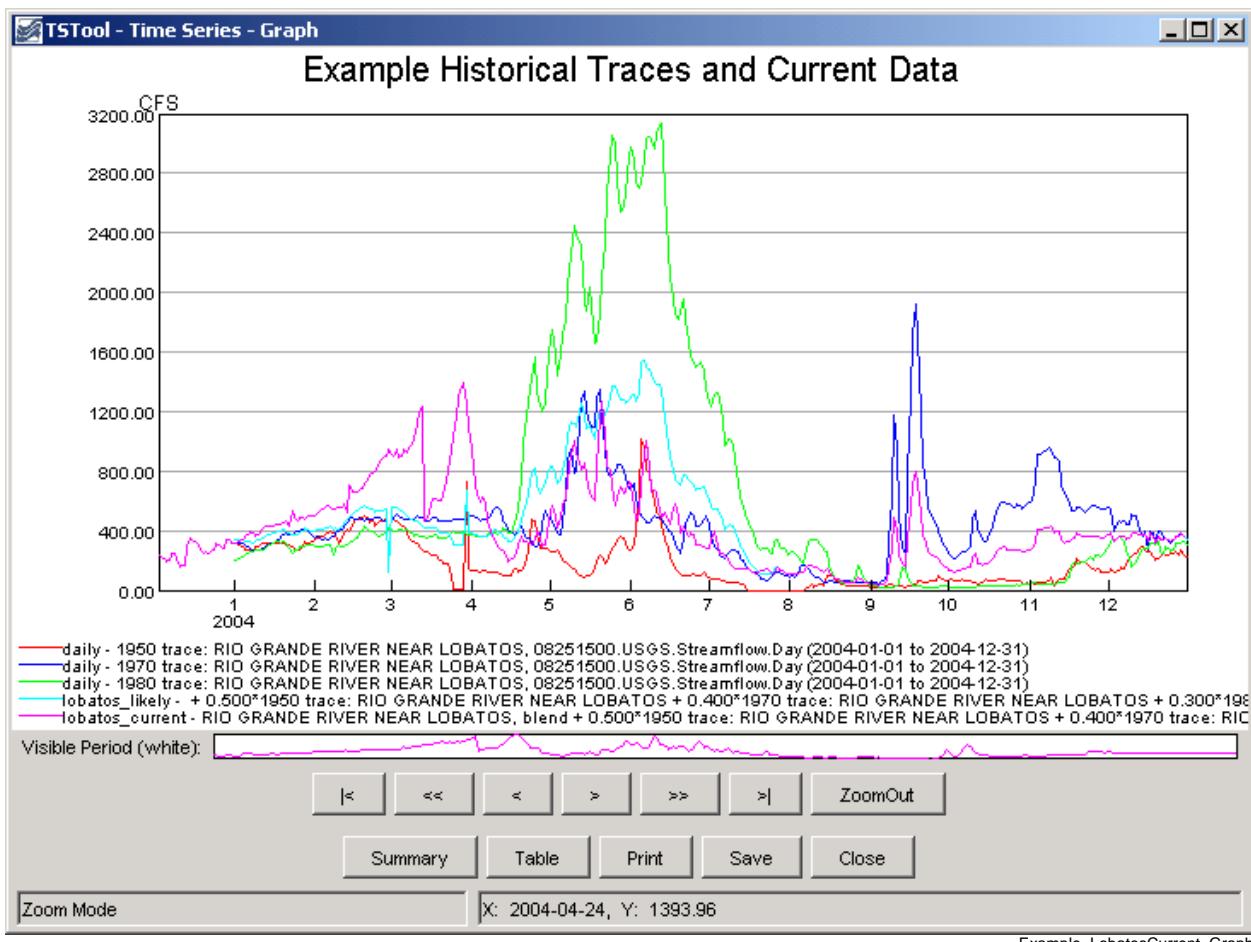


Example_LobatosCurrent_TSTool

Any of the time series can be selected and viewed. The following features are useful for selecting appropriate traces:

1. Convert a time series to traces and then plot all the traces. Use the graph properties to turn traces on/off or use symbols to identify. You can consequently visually select traces that are similar.
2. Use the tools described in **Chapter 5 - Tools** to evaluate time series and traces. For example, the **Year to Date** report can be used to determine how well different years compare volumetrically.

Key traces and output time series can be selected and graphed, as shown below.



6.4 Time Series Product Examples

Time series products are described in the **TSView Time Series Viewing Tools** appendix. In summary, a time series product file can be generated that uses time series identifiers to indicate data to be processed, and includes other properties (e.g., titles) to configure a graph. TSTool can process time series products in a number of ways, as illustrated by the following examples.

6.4.1 Time Series Product - Using TSTool to Display Graphs from an Application

TSTool is primarily used as an interactive tool or to process commands files in batch mode. However, it is also possible to run TSTool with a commands file, no main graphical user interface, and still display only specific graphs. For example, TSTool can be called from an application to display a graph by reading data from a recognized database or file format. This takes advantage of TSTool's features rather than adding additional features to the application. The following example illustrates how to display a graph of precipitation and streamflow data in a single graph, using data from the State of Colorado's HydroBase database. TSTool should be started using a command line similar to:

```
tstool -commands example.tstool -nomaingui
```

Additionally, the HydroBase database to be used should be configured in the TSTool configuration file (see the **Installation and Configuration** appendix). This run mode actually runs the GUI; however, the GUI is never made visible. Instead, it is used to control a batch commands file run, with graphical displays. Because the interactive main interface is disabled, the normal HydroBase login dialog is not shown; therefore, the HydroBase information must be defined in the configuration information.

Although it is possible to display several graphs at the same time, it is currently assumed that only one graph will be shown. Closing the graph will close TSTool. The commands file can be complex but in many cases will be simple because an application is calling TSTool to display a single graph. The following example shows a typical commands file for this run mode:

```
# Example commands file to run a commands file without showing the main GUI
# but showing a plot to the screen. When the plot window closes, TSTool will
# exit without prompting. TSTool should be called using:
#
# TSTool -commands ThisFile -nomaingui
#
# This is useful for displaying plots from applications that don't have features
# to query HydroBase.
#
# SAM, RTi, 2002-06-02
#
# Process a time series product description file and display a plot window
# to the screen.
processTSPProduct("test.tspd", GUIAndBatch, Preview)
```

The `processTSProduct()` command references a time series product file. An example of the file is as follows (see the **TSView Time Series Viewing Tools** appendix for a full description of time series product file properties):

```
[Product]

ProductType = "Graph"
TotalHeight = "400"
TotalWidth = "600"

[SubProduct 1]

GraphType = "Bar"
MainTitleString = "Precipitation"
BarPosition = "CenteredOnDate"

[Data 1.1]

Color = "Blue"
TSID = "7337.NOAA.Precip.Month~HydroBase"

[SubProduct 2]

GraphType = "Line"
MainTitleString = "Streamflow"

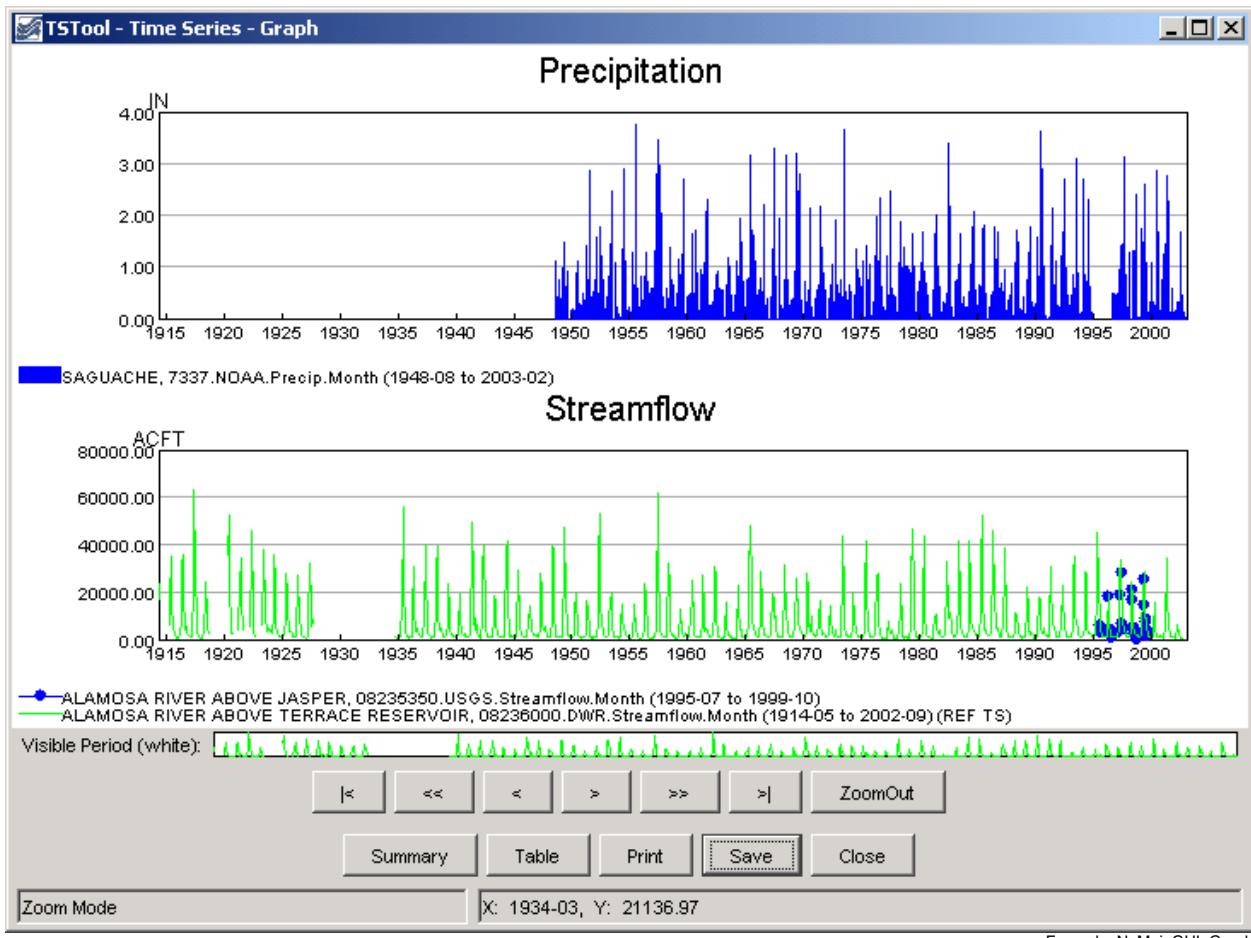
[Data 2.1]

SymbolSize = "7"
SymbolStyle = "Circle-Filled"
TSID = "08235350.USGS.Streamflow.Month~HydroBase"

[Data 2.2]

TSID = "08236000.DWR.Streamflow.Month~HydroBase"
```

The resulting graph is shown in the following figure. Pressing **Close** will exit TSTool.



6.4.2 Automating Graphs to Compare Observed and Simulated Time Series

It is often useful to automate comparison of observed and simulated time series (e.g., during model calibration). For example, after making an adjustment to a model during calibration, many comparisons may be made to evaluate the changes. TSTool can help with comparisons. For example, consider the simulated and observed time series stored in a DateValue file (*results.dv*), as follows (in this case the DateValue file was actually created by reading model input and output, and the **TSID** and **DataType** lines were hand-edited in the file to facilitate this example).

```
# DateValueTS 1.3 file
# File generated by...
# program: TSTool 6.08.02 (2004-07-27) Java
# user: sam
# date: Thu Jul 29 09:17:35 MDT 2004
# host: host unknown
# directory: J:\CDSS\develop\Apps\TSTool\test\Commands\createTraces
# command: TSTool -home C:\CDSS
#-----
#
# Commands used to generate output:
#
# 09152500...MONTH~StateMod~J:\CDSS\DataSets\SWSI_Gunnison\StateMod\gunnv.rih
# 09152500.StateMod.River_Outflow.Month~StateModB~J:\CDSS\DataSets\SWSI_Gunnison\StateMod\gunnrb.b43
#
Delimiter      = " "
NumTS          = 2
TSID           = "09152500..StreamFlow_Observed.MONTH" "09152500..Streamflow_Simulated.Month"
Alias          = " " "
Description    = "09152500" "Gunn R. NR GrandJ _FLO"
DataType        = "Streamflow_Observed" "Streamflow_Simulated"
Units          = "ACFT" "ACFT"
MissingVal     = -999.0000 -999.0000
Start          = 1908-10
End            = 2001-09
#
# Time series comments/histories:
#
#
# Creation history for time series 1 TSID=09152500...MONTH Alias=):
#
# Read StateMod TS for 1908-10 to 2001-09 from "J:\CDSS\DataSets\SWSI_Gunnison\StateMod\gunnv.rih"
#
# Creation history for time series 2 TSID=09152500.StateMod.River_Outflow.Month Alias=):
#
#   Read from "J:\CDSS\DataSets\SWSI_Gunnison\StateMod\gunnrb.b43 for 1908-10 to 2000-09
#
#EndHeader
Date "09152500...MONTH, ACFT" "09152500.StateMod.River_Outflow.Month, ACFT"
1908-10 -999.0000 82035.8828
. . . omitted - periods do not exactly line up . . .
1916-10 61488.5000 95692.6641
1916-11 56529.8000 97254.9297
1916-12 55339.6000 94700.1563
1917-01 52265.2000 47388.2305
1917-02 49984.2000 42303.5938
1917-03 79935.0000 64748.8125
. . . similar to end of file . . .
```

These time series can be read into TSTool, an XY graph produced, and the time series product saved (*results.tsp*), as shown in the following example (note the original TSID properties have been inserted, corresponding to the original data):

```
[Product]

ProductType = "Graph"
TotalWidth = "600"
TotalHeight = "400"
MainTitleString = "Streamflow Gage 09152500"
SubTitleString = "Comparison of Observed and Simulated"

[SubProduct 1]

GraphType = "XY-Scatter"
XYSscatterMethod = "OLSRegression"
LegendFormat = "Auto"
MainTitleString = ""

[Data 1.1]

#TSID = "09152500...MONTH~StateMod~J:\CDSS\DataSetS\SWSI_Gunnison\StateMod\gunnv.rih"
TSID = "09152500..Streamflow_Observed.MONTH~DateValue~results.dv"

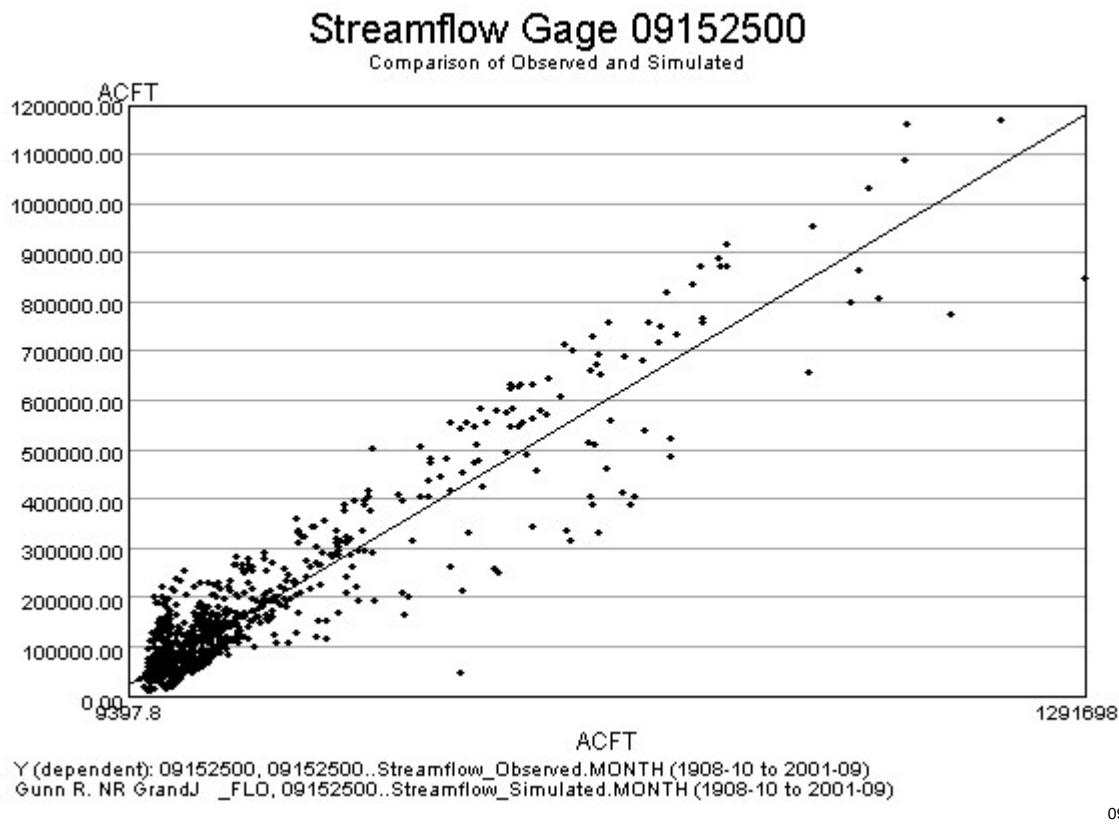
[Data 1.2]

#TSID = "09152500.StateMod.River_Outflow.Month~StateModB~J:\CDSS\DataSetS\SWSI_Gunnison\StateMod\gunnv.b43"
TSID = "09152500..Streamflow_Simulated.MONTH~DateValue~results.dv"
```

Finally, a commands file (*results.TSTool*) can be created that processes the time series and time series product file:

```
processTSPProduct("results.tsp", GUIAndBatch, NoPreview, "09152500.jpg")
```

The above commands can be run from the TSTool GUI or in batch mode to produce the following graph (note in this example that the x-axis data values are so large that the software is having difficulty finding good labels):



Because this approach relies primarily on the time series identifiers to associate the time series data with the time series product, it is important to establish a concise and clean identifier scheme. Once a working example is established, the example can be scaled up to a larger production either by repeating the example (and changing identifiers) or by automatically changing the example to replace strings. The latter is not currently part of TSTool; however, TSTool can call external programs (see the `runProgram()` command), which could supplement the existing TSTool features.

This page is intentionally blank.

Troubleshooting

Version 07.00.00, 2006-11-22, Color, Acrobat Distiller

This chapter discusses how to troubleshoot TSTool problems. **Section 8.1 – Obsolete Commands** lists obsolete commands, which may no longer be supported by current software and should be phased out.

TSTool runs in both graphical user interface (GUI) and batch mode. In both cases a log file contains messages from the program. When the GUI is used, the log file is created in the logs directory under the main installation directory (e.g., *C:\CDSS\logs\tstool.log* for a CDSS installation or *C:\Program Files\RTi\RiverTrak\logs\TSTool.log* for an RTi RiverTrak® system installation). It is recommended that the `startLog()` command be inserted as the first command in each command file, using the name of the commands file in its name.

The most common problems are program configuration (see the **Installation and Configuration** appendix), user input error (see the commands reference for command syntax), and data errors for various input types (see below and see also the input type appendices). Other problems should be reported to the TSTool developers (see **Chapter 1 - Acknowledgements** for support contacts). Code has been implemented to detect common errors, but you may need to refer to the log file to determine the nature of a problem.

In general, when running the TSTool GUI, you will be warned about major problems using pop-up dialogs. When running in batch mode, warnings are only printed to the log file. In either case, the log file can be viewed. The log file viewer can be used to pinpoint the source of warnings. If the run has been successful, you will only see status messages in the log file. Status messages provide useful information (such as regression results).

The following table summarizes common errors and their fixes. **Because database designs change over time, if an error is occurring in a batch run, you may want to interactively query the time series because the interface enforces conventions that are compatible with current database designs and it may be easier to diagnose errors interactively.**

TSTool Errors and Possible Solutions

Error	Possible solutions
TSTool does not run on Windows (error at start-up).	<p>1. If using the TSTool executable and the following is shown:</p>  <p style="text-align: right;">Troubleshooting_LaunchError</p> <p>This error may be shown if running the software from the command line after an installation. Reboot the computer so that installation settings are fully recognized.</p> <p>2. If TSTool is run using the batch file... The batch file (in the <i>bin</i> directory under the main install point) uses a command shell window that may be running out of environment space (in this case you should see a message in the command shell window to that effect). To correct, change the command shell window properties so that the initial memory is 4096 or greater. This may not take effect unless the command shell window was started from the Start menu.</p> <p>Additionally, to help diagnose errors, try running the <i>TSTool.exe</i> or <i>TSTool.bat</i> batch file from a command shell rather than from a desktop shortcut or Windows Explorer. Doing so may print useful messages to the command shell window.</p>
When used with the HydroBase input type for a Microsoft Access database, an error occurs selecting the HydroBase database.	<p>1. TSTool tries to list the ODBC DSN that are available for HydroBase databases. It does so by running the <i>shellcon.exe</i> program, typically installed in <i>\cdss\bin</i> if TSTool is used with CDSS. If this directory is not in the PATH environment variable, the program will not be found and an error will occur. The PATH normally will include the directory after installation, in order to allow TSTool to be run from directories other than the installation directory. The PATH can be checked by opening a command shell and typing path at the command prompt. If the PATH is not properly set, edit it as follows:</p> <ul style="list-style-type: none"> • For Windows NT/2000/XP machines, add <i>\CDSS\bin</i> (or <i>\Program Files\RTi\RiverTrak\bin</i>) to the PATH using the Settings...Control Panel...System...Advanced...Environment Variables settings. You may need to have the administrator perform this step. • For Windows 95/98 machines, add <i>\CDSS\bin</i> (or <i>\Program Files\RTi\RiverTrak\bin</i>) to the PATH in the <i>autoexec.bat</i> file. Reboot to apply for all subsequent windows. You may have done this previously. <p>A work-around is to manually type in the ODBC DSN in the entry field.</p> <p>2. Only user ODBC DSNs are listed when selecting HydroBase. If the DSN was defined as a system DSN, it will not be listed. Redefine the DSN as a user DSN.</p>

TSTool Errors and Possible Solutions (continued)

Error	Possible solutions
Time series (of any data type) are not returned from the HydroBase database.	<ol style="list-style-type: none"> Verify that the database includes the water districts of interest using the File...Properties...HydroBase Information menu. Verify that the structure/station identifier is valid. For example, a USGS gage identifier may have changed. To verify, try querying by its name rather than the identifier. Also, use the CDSS StateView application to view HydroBase data. If a zero-filled time series file cannot be found, check its path or use the <code>TS alias = newTimeSeries()</code> command.
A specific time series is not returned from the HydroBase database.	Time series in HydroBase are tagged with the data source (e.g., USGS). These data source abbreviations or their handling by software may have changed over time and a data source in a time series identifier may not be valid. Current software requires the data source for HydroBase time series, if a data source is used with the data type in HydroBase. Try re-querying the time series to see if the data source has changed.
A data type combination is not available for queries.	TSTool has been implemented to support various input types as much as possible. However, it may not have features to view all time series in an input type. For HydroBase, the CDSS StateView application has additional displays.
Time series (of any data type) have -999 values.	TSTool queries time series by allocating memory for the requested period and then filling in values from the database. The output period (or maximum if not specified) may be such that time series values were not found in the database and were set to the missing data value of -999. You can use fill options to fill the missing data within the requested period.
TSTool fails on large queries.	TSTool may run out of memory on queries (hundreds or thousands of time series, depending on machine memory). More time series can be handled if run in batch mode. In the <code>tstool.bat</code> file, change the <code>-XmxNNm</code> option after the JRE program name to tell Java to allow more memory (increase the number of MB NN as appropriate for the amount of memory available on the machine – use a high number to force using hard disk swap space if desired).
Unexpected failure.	<p>If there was an error in input that was serious, TSTool may quit processing input. See the log file for details. If the log file does not offer insight, contact support. Specific causes of failure may include:</p> <ol style="list-style-type: none"> TSTool has been developed using Java 1.4.2, which is referenced in the startup batch file or command line. Trying to use a different Java version may cause unexpected errors. To determine the Java version that is being used with TSTool, use Tools...Diagnostics and select the Allow debug checkbox. Then use the Help...About TSTool menu item and press Show Software/System Details to display information that includes the Java version that is being used to run TSTool.
Unable to find files correctly.	TSTool uses Java technology, which at this time does not allow the working directory to be changed internally. The working directory is assumed to be the same as the location of the commands file. Verify that the working directory is as expected using File...Properties...TSTool Session .

8.1 Obsolete Commands

TSTool and the commands that it supports have evolved over time. In early versions, many commands used syntax similar to the following:

```
-SomeCommand parameter
```

Later, the function notation was adopted:

```
someCommand(parameter1,parameter2)
```

However, parameters for such commands were required to be in a specific order and enhancements were difficult to implement because the parameter order needed to be maintained. Recent enhancements have added new commands and converted some older commands to a new free-format “named parameter” notation:

```
someCommand(param1=value1,param2=value2)
```

The new notation allows parameters to be omitted when using a default value, and allows new parameters to be added to commands, as necessary, to enhance existing functionality. Old commands will be transitioned to the new syntax until all commands have been updated. Support for the older notation is provided where possible. In most cases, editing an old command with the command editor dialogs will convert from old to new syntax automatically.

The following table lists obsolete commands. Although TSTool tries to run old commands as much as possible, some obsolete commands are not recognized. Old commands should be updated to the new commands, if possible. TSTool will print warnings for commands that are not recognized or are out of date.

The TSID abbreviation, when inside parentheses for a command, is interchangeable with the time series alias.

TSTool Command Summary – Obsolete Commands

Command	Description
add(TSID,TSID1, TSID2,...)	Add the 2 nd + time series to the first time series, retaining the original identifier. This form of the command is obsolete and should be updated to use the new form described that includes a flag for handling missing data.
-archive_dbhost HostName	This option is normally set during installation and is typically not specified in command files. Specify the Internet host name for the remote HydroBase database server. This is configured at installation time and will be either "localpc" (for a local Microsoft Access HydroBase database, indicating that no remote server is used) or a machine name for the Informix database server. To change the defaults from those in the <i>tstool.bat</i> file, specify this option again on the command line or edit the batch file. See also -dbhost. This option is used in addition to the -dbhost information to allow a TSTool user to switch between the local PC and the main database server.
-averageperiod MM/YYYY MM/YYYY	Specify the period to be used to compute averages when the -fillhistave option is specified. This command is obsolete and should be converted to <code>setAveragePeriod()</code>.
-batch	Indicates to run in batch mode. This is automatically set if -commands is specified. This command is no longer needed.
-browser Path	This option is normally set during installation and is typically not specified in command files. Specify the path to the web browser to use for on-line documentation. This command is no longer needed.
-cy	Output in calendar year format. This command is obsolete and should be replaced with <code>setOutputYearType()</code> .
-d#[,#]	Set the debug level. The first number is the debug level for the screen. The second is for the log file. If one level is specified, it is applied to the screen and log file output. This command is obsolete and should be replaced with <code>setDebugLevel()</code> .
-data_interval Interval	Indicate the data interval (e.g., MONTH, DAY) to use with all structures/stations indicated by the -slist option. See the appendices for a list of intervals for different input and data types. This option is only available in batch mode. This command is obsolete. Use the <code>createFromList()</code> command.
-datasource ODBCDataSourceName	Specify an ODBC Data Source Name to use for the HydroBase database. In commands files, this option is obsolete and should be replaced with <code>setDataSource()</code> .
-data_type Type	Indicate the data type (e.g., DivTotal, DQME) to use with all structures/stations indicated by the -slist option. This option is only available in batch mode. This command is obsolete. Use the <code>createFromList()</code> command.

TSTool Command Summary – Obsolete Commands (continued)

Command	Description
day_to_month_reservoir (TSID, ndays, flag)	Read a daily time series and convert to a monthly time series using the reservoir method. This is generally only applied to reservoir storage. This command has been replaced with the TS X = newEndOfMonthTSFromDayTS() command, optionally followed by a fillInterpolate() command.
-dbhost HostName	This option is normally set during installation and is typically not specified in command files. Specify the Internet host name for the primary HydroBase database server. This is configured at installation time and will be either localpc (for a local Microsoft Access database) or a machine name for the Informix database server. To change the defaults from those in the tstool.bat file, specify this option again on the command line or edit the batch file. See also -archive_dbhost. In command files, this command is obsolete and should be replaced with setDatabaseHost().
-detailedheader	Insert time series creation information in output headers. This preserves information from the log file that may otherwise be lost. The default is not to generate detailed headers. This command is obsolete and should be replaced with setOutputDetailedHeaders().
fillconst (TSID, Value)	Fill the time series with a constant value. This command is obsolete and has been replaced by fillConstant().
-fillData File	Specify a StateMod format fill pattern file to be used with the fillpattern() command. This command can be repeated for multiple pattern files. This command is obsolete and should be replaced by setPatternFile().
-fillhistave	Currently only enabled for frost dates and monthly data. Indicates that the time series should be filled with the historical average values from the output period where data are missing (after filling by other methods). See also the -averageperiod option. This command is obsolete and should be replaced by fillHistMonthAverage() and fillHistYearAverage().
Graph g = newGraph(GraphType, Visibility, TimeSeriesToGraph)	Create a new graph window. This command is no longer supported.
-helpindex Path	This option is normally set during installation and is typically not specified in command files. Specify the path to help index file for on-line documentation.
-ignorelezero	Treat data values <= 0 as missing when computing averages but do not replace when filling. This command is obsolete and should be replaced with setIgnoreLEZero().
-include_missing_ts	If a time series cannot be found, include an empty time series. This command is obsolete and should be replaced with setIncludeMissingTS().

TSTool Command Summary – Obsolete Commands (continued)

Command	Description
-informix	Use <code>setDatabaseEngine()</code> .
-missing Value	Use the specified value for missing data values (StateMod only). The default is -999.0. This command is obsolete and should be replaced by <code>setMissingDataValue()</code> .
-fillusingcomments	This option only applies to diversion time series and causes the diversion comments to be evaluated. Comments that indicate no diversion in an irrigation year will result in missing data for that year being replaced with zeros. This command is obsolete and should be replaced with <code>setUseDiversionComments()</code> .
month1/year1 month2/year2	Specifies beginning and ending months for period of record - calculations are still based on the entire period of record (i.e., regression values) but the final output is according to these values, if given. Month 1 is January. Years are 4-digit. This command is obsolete and should be replaced by <code>setOutputPeriod()</code> .
-o outfile	Specify output file name. This is used in conjunction with other -o options. This option is obsolete and should be replaced with <code>writeStateMod()</code> or other output commands.
-odatevalue	Output a DateValue format file. This option is obsolete and should be replaced with <code>writeDateValue()</code> .
-ostatemod	Output a StateMod format file. This option is obsolete and should be replaced with <code>writeStateMod()</code> or other output commands.
-osummary	Output a time series summary. This command is obsolete. There is currently not a replacement other than using the GUI.
-osummarynostats	Output a time series summary without statistics (this is used with the data extension procedure developed by Ayres for CDSS). This command is obsolete. There is currently not a replacement other than using the GUI.
regress(TSID1,TSID2)	Performs a linear regression analysis between the two time series, filling missing data of the first time series. Regression information is printed to the log file. This command is obsolete and should be replaced with <code>fillRegression()</code> .
regress12(TSID1, TSID2) regressMonthly(TSID1,TSID2)	Same as <code>regress()</code> except 12 separate monthly regressions values are calculated. This command is obsolete and should be replaced with <code>fillRegression()</code> .
regresslog(TSID1, TSID2)	Same as <code>regress()</code> except regressions values are calculated logarithmically. This command is obsolete and should be replaced with <code>fillRegression()</code> .
regresslog12(TSID1, TSID2) regressMonthlyLog(TSID1,TSID2)	Same as <code>regresslog()</code> except 12 monthly regressions values are calculated. This command is obsolete and should be replaced with <code>fillRegression()</code> .

TSTool Command Summary – Obsolete Commands (continued)

Command	Description
<code>setconst(TSID,Value)</code>	Set the time series to the given value for all data. If the time series is not in the database, created an empty time series and then set to a constant value. This command is obsolete and should be replaced with <code>setConstant()</code> .
<code>setconstbefore(TSID, Value, Date)</code>	The time series to the given value for all data on and before the specified date (YYYY-MM or MM/YYYY). This command is obsolete and should be replaced with <code>setConstantBefore()</code> .
<code>setQueryPeriod(Start,End)</code>	Set the global period to query databases and read from files. This command has been replaced by the <code>setInputPeriod()</code> command.
<code>-sqlserver</code>	This command is obsolete and should be replaced by <code>setDatabaseEngine()</code> .
<code>-slist File</code>	Create time series from a list file. This command is obsolete and should be replaced by <code>createFromList()</code> .
<code>-units value</code>	Output using the specified units (default is to use database units). This command is not active.
<code>-w#[,#]</code>	Set the warning level. The first number is the warning level for the screen. The second is for the log file. If one level is specified, it is applied to the screen and log file output. This command is obsolete and should be replaced with <code>setWarningLevel()</code> .
<code>-wy</code>	Output in water year format. This command is obsolete and should be replaced with <code>setOutputYearType()</code> .

Command Glossary

Version 06.17.00, 2006-04-30, Acrobat Distiller

The following parameter names and terms are used throughout TSTool commands. A term indicated in **bold** font is a definition. A term indicated in **bold courier** font is a parameter name. Parameters that are infrequently used are listed with the corresponding commands. Common parameters are defined but long lists of corresponding commands are not provided.

a1,... – Used with the ARMA () command.

b1,... – Used with the ARMA () command.

Alias – A (generally) short identifier for a time series, used in place of the TSID, which simplifies commands. The Alias and TSID values are interchangeable when used as parameters to commands and may both be referred to as TSID in command editors. See also TSID.

Alias – A (generally) short identifier for a time series, used in place of the TSID, which simplifies commands. When used to create/read a time series, the syntax of a command is typically similar to: TS Alias = command (...). See also TSID.

AddTSID – Time series identifiers for time series to add. See the add () command.

AddValue – A numerical value to be added to a time series. See the addConstant () command.

AdjustMethod – Indicates the method used when adjusting a time series. See the adjustExtremes () command.

AllowMissingCount – Indicate how many missing data values are allowed in an interval, in order to allow processing. See the changeInterval () and newStatisticYearTS () commands.

AnalysisEnd – A DateTime that indicates the end of an analysis.

AnalysisMonth – One or more months indicating which months should be processed in the analysis. See the fillRegression () command.

AnalysisStart – A DateTime that indicates the start of an analysis.

ARMAInterval – The data interval used in an ARMA analysis. See the ARMA () command.

AutoExtendPeriod – Indicate whether to autoextend the period of all time series to be the output period. See the setAutoExtendPeriod () command.

AverageEnd – A DateTime that indicates the end of an averaging analysis. See the setAveragePeriod () command.

AverageMethod – Indicate the method to use when averaging data. See the runningAverage () command.

AverageStart – A DateTime that indicates the start of an averaging analysis. See the setAveragePeriod () command.

BlendMethod – The method to use when blending time series. See the `blend()` command.

BlendTSID – Time series identifiers for time series to blend into main time series. See the `blend()` command.

Bracket – The number of days to search forward and back for a non-missing value. See the `newEndOfMonthTSFromDayTS()` and `runningAverage()` commands.

CalculateFactorHow – Indicate how to calculate the factor used when prorating values. See the `fillProrate()` command.

CommandLine – The command line for a program to run. See the `runProgram()` command.

ConstantValue – A numerical value used for filling, etc. See the `fillConstant()`, `setConstant()` and `setConstantBefore()` command.

DatabaseName – The name of a database, when making a database connection. See the `openHydroBase()` command.

DatabaseServer – The name of a database server, when making a database connection. See the `openHydroBase()` command.

DataSource – The data source to use when forming a TSID. See the `createFromList()` command.

DataType – The data source to use when forming a TSID. See the `createFromList()` command.

DateTime – A date/time value, typically represented as a string, which indicates a point in time. Date/time strings have a precision that is interpreted by the software. For example, the date/time string 1990 has a precision of year, whereas the string 1990-01-12 has a precision of day.

DateTime – A specific date/time associated with time series data. See the `setDataValue()` command.

DayTSID – Time series identifier for a daily time series. See the `newDayTSFromMonthAndDayTS()` command.

DefaultFlow – Indicate a default flow value to be used if observations or filled values cannot be found. See the `lagK()` command.

Delim – The delimiter character(s) used when processing delimited files. See the `createFromList()` command.

DependentAnalysisEnd – A DateTime that indicates the end of an analysis of dependent time series. See the `fillMOVE2()` command.

DependentAnalysisStart – A DateTime that indicates the start of an analysis of dependent time series. See the `fillMOVE2()` command.

Description – The description (name) for a time series. See the `newTimeSeries()` command.

DeselectAllFirst – Indicate whether to deselect all time series before processing the command. See the `selectTimeSeries()` command.

DiffFlag – A character flag used to indicate when time series values are different. See the `compareTimeSeries()` command.

Divisor – Indicate which time series is the divisor. See the `relativeDiff()` command.

DivisorTSID – Time series identifier for time series to divide another time series. See the `divide()` command.

ExtremeToAdjust – Indicates whether the maximum or minimum value in a time series should be adjusted. See the `adjustExtremes()` command.

ExtremeValue – The threshold value when adjusting extreme values. See the `adjustExtremes()` command.

FillDirection – Indicate which direction (Forward or Backward) filling should occur. See the `fillProrate()` and `fillRepeat()` commands.

FillEnd – A DateTime that indicates the end of a fill process.

FillFlag – A character flag used to indicate when time series values are filled. See the `fillHistMonthAverage()`, `fillHistYearAverage()`, `fillMOVE2()`, `fillProrate()`, and `fillRegression()` commands.

FillNearest – Indicate whether missing data values should be filled with the nearest non-missing value. See the `lagK()` command.

FillStart – A DateTime that indicates the start of fill process.

FillUsingDivComments – Indicate whether missing data values should be filled using diversion comments. See the `readHydroBase()` and `TS Alias = readHydroBase()` commands.

FillUsingDivCommentsFlag – A character flag used to indicate when time series values are filled. See the `readHydroBase()`, and `TS Alias = readHydroBase()` commands.

HandleMissingHow – Indicate how to handle missing data values when processing time series. For example, when adding time series, missing values can be ignored or can result in a missing value in the result. See the `add()`, `cumulate()`, and `subtract()` commands.

HandleMissingTShow – Indicate how to handle missing time series during processing. See the `createFromList()` command.

ID – The identifier to match in a file. See the `createFromList()` command.

IDCol – The column number (or name) to be read from a delimited file. See the `createFromList()` command.

IgnoreLEZero – Indicate whether values less than or equal to zero should be ignored when computing historical averages for time series. See the `setIgnoreLEZero()` command.

IncludeMissingTS – Indicate whether missing time series (e.g., from a query or read) should automatically be included using default information. See the `setIncludeMissingTS()` command.

IndependentTSID – Time series identifier for the independent time series being processed. See the `fillFromTS()`, `fillMOVE2()`, `fillProrate()`, `fillRegression()`, `setFromTS()`, and `setMax()` commands.

InflowStates – The inflow states (initial states) when routing a flow time series. See the `lagK()` command.

InitialValue – Indicate an initial value needed for computations. See the `fillProrate()` and `newTimeSeries()` commands.

InputEnd – A DateTime that indicates the end of a file read or a database query.

InputFile, InputFile1, InputFile2 – The name of an input file to read, used by many commands.

InputName – The input name to use when forming a TSID. See the `createFromList()` command.

InputStart – A DateTime that indicates the start of file read or a database query.

InputType – The input type to use when forming a TSID. See the `createFromList()` command.

Intercept – The intercept to be enforced when determining a line of best fit. See the `fillRegression()` command.

Interval – The data interval to use when forming a TSID. See the `createFromList()`, `readHydroBase()`, and `shiftTimeByInterval()` commands.

K – The attenuation factor used when routing a flow time series. See the `lagK()` command.

Lag – The lag term for routing a flow time series. See the `lagK()` command.

Length – The length of a time series trace. See the `createTraces()` command.

ListFile – The name of an input or output list (delimited) file to be written or read, specified using a relative or absolute path. See the `createFromList()` command.

LogFile – The name of the log file, specified using a relative or absolute path. See the `setLogFile()` command.

LogFileLevel – The level for messages printed to the log file. See the `setDebugLevel()` and `setWarningLevel()` commands.

MatchDataType – Indicate whether the data type part of a TSID should be matched when comparing time series identifiers. See the `compareTimeSeries()` command.

MatchLocation – Indicate whether the location part of a TSID (Alias) should be matched when comparing time series identifiers. See the `compareTimeSeries()` command.

MaxIntervals – The maximum number of intervals to process, typically used to limit a fill or analysis procedure. See the `adjustExtremes()`, `fillInterpolate()`, and `fillRepeat()` commands.

MaxValue – The maximum value in an analysis. See the `normalize()` and `replaceValue()` commands.

Method – A method used when processing data, used to more specifically control how a command functions. See the `analyzePattern()` and `disaggregate()` commands.

MinValue – The minimum value in an analysis. See the `normalize()` and `replaceValue()` commands.

MinValueHow – Indicate how to determine the minimum value in an analysis. See the `normalize()` command.

MissingValue – A numerical value used for missing data in time series. See the `writeStateMod()` command.

MonthTSID – Time series identifier for a monthly time series. See the `newDayTSFromMonthAndDayTS()` command.

MonthValues – Monthly values used for filling, etc. See the `setConstant()` command.

MultiplierTSID – Time series identifier for the time series to multiply the main time series. See the `multiply()` command.

Multiplier – Value(s) to multiply time series value(s) by when processing. See the `shiftTimeByInterval()` command.

NewDataType – The data type for a new time series, typically used where the data type must be explicitly defined and is not determined from a TSID. See also `NewTSID`. See the `changeInterval()` command.

NewInterval – The data interval for a new time series, typically used where the interval must be explicitly defined and is not determined from a TSID. See also `NewTSID`. See the `changeInterval()` command.

NewTimeScale – The new time scale (ACCM for accumulated data, INST for instantaneous data, MEAN for mean data) for a time series. See the `changeInterval()` command.

NewTSID – The new time series identifier for a time series, used with commands that create new time series. See the `copy()` and `newDayTSFromMonthAndDayTS()` commands.

NewUnits – The new data units for a time series. See the `convertDataUnits()`, `TS Alias = readDateValue()`, `TS Alias = readMODSIM()`, `TS Alias = readNWSCard()`, and `TS Alias = readRiverWare()` commands.

NewValue – The new value in an analysis. See the `replaceValue()` and `setDataValue()` commands.

NumEquations – Number of equations to use when analyzing data (typically one or monthly equations). See the `fillMOVE2()` and `fillRegression()` commands.

ObsTSID – The time series identifier for an observed time series. See the `lagK()` command.

OdbcDSN – The Open Database Connectivity (ODBC) Data Source Name (DNS) for a database connection. See the `openHydroBase()` command.

OldTimeScale – The old time scale (ACCM for accumulated data, INST for instantaneous data, MEAN for mean data) for a time series. See the `changeInterval()` command.

OutflowStates – The outflow states (initial states) when routing a flow time series. See the `lagK()` command.

OutputEnd – A DateTime that indicates the end of output.

OutputFile – The name of an output file to be written, specified using a relative or absolute path.

OutputStart – A DateTime that indicates the start of output.

OutputYearType – Indicate the type of year (e.g., calendar year, water year) for output. See the `setOutputYearType()` command.

PatternFile – The file name for a pattern file. See `setPatternFile()` command.

PatternID – An identifier for a pattern (e.g., WET, DRY, AVG). See the `analyzePattern()` and `fillPattern()` commands.

Percentile – Percentile value(s) used when analyzing time series. See the `analyzePattern()` command.

Pos – The position in the time series list. See the `deselectTimeSeries()` and `selectTimeSeries()` commands.

pP – Used with the `ARMA()` command.

Precision – The precision (number of digits after the decimal point) used when comparing values or formatting values for output. See the `compareTimeSeries()`, `writeRiverWare()`, and `writeStateMod()` commands.

QueryEnd – A DateTime that indicates the end of a database query. The `InputEnd` parameter is preferred and is used in new commands.

QueryStart – A DateTime that indicates the start of database query. The `InputStart` parameter is preferred and is used in new commands.

qQ – Used with the `ARMA()` command.

Read24HourAsDay – Indicate that a time series with data interval 24Hour should be automatically read as Day. See the `readNwsCard()` and `TS Alias = readNwsCard()` commands.

ReadEnd – A DateTime that indicates the end of a file read. See the `readNWSCard()` command. The `InputEnd` parameter is preferred.

ReadStart – A DateTime that indicates the start of file read. See the `readNWSCard()` command. The `InputStart` parameter is preferred.

RecalcLimits – Recalculate the data limits for a time series, usually when supplemental raw data are being supplied after an initial read. See the `fillUsingDiversionComments()` command (used with the State of Colorado's HydroBase input type).

ReferenceDate – The starting date for a time series trace. See the `createTraces()` command.

Reset – A DateTime field that indicates when to reset data values in a manipulation. For example, a time series may be set to zero at the start of each year when used with the `cumulate()` command. See the `cumulate()` command.

RunMode – Typically used to indicate whether the command should be processed in batch mode, via the GUI, or both. See the `openHydroBase()`, `processTSPproduct()`, and `setWorkingDir()` commands.

Scale – A scale factor to be applied to data. See the `writeRiverWare()` command.

ScaleValue – A numerical value used for scaling time series. See the `scale()` command.

Scenario – The scenario to use when forming a TSID. See the `createFromList()` command.

ScreenLevel – The level for messages printed to the screen (console). See the `setDebugLevel()` and `setWarningLevel()` commands.

SelectAllFirst – Indicate whether to select all time series before processing the command. See the `deselectTimeSeries()` command.

SearchStart – A DateTime that indicates the search start date/time in an analysis. See the `newStatisticYearTS()` command.

SetEnd – A DateTime that indicates the end of a set process.

Set_scale – See the `writeRiverWare()` command.

SetStart – A DateTime that indicates the start of set process.

Set_units – See the `writeRiverWare()` command.

ShiftDataHow – Indicate how to shift time series traces. See the `createTraces()` command.

SpecifyWeightsHow – Indicate how to specify weights when processing time series. See the `TS Alias = weighTimeSeries()` command.

Statistic – A statistic to evaluate. See the `newStatisticYearTS()` command.

SubtractTSID – Time series identifiers for time series to subtract. See the `subtract()` command.

Suffix – The suffix to be automatically applied to the name of a file. See the `setLogFile()` command.

TestValue – A test value used in an analysis. See the `newStatisticYearTS()` command.

Timeout – The timeout when running an external program, after which processing will continue. See the `runProgram()` command.

Tolerance – A value (or values) used to indicate an allowable error/difference. See the `compareTimeSeries()` command.

TransferHow – Indicate how to transfer data during processing, either according to the date/time or sequentially. The latter can be used when time series do not align on date/time (e.g., due to a shift, leap year, etc.). See the `setFromTS()` command.

Transformation – Indicate whether the time series data should be transformed before processing. See the `fillInterpolate()`, `fillMOVE2()`, and `fillRegression()` commands.

TSID – Time series identifier, which is used to uniquely identify a time series. In full notation, this consists of a string similar to the following:

Location.DataSource.DataType.Interval.Scenario~InputType~InputName. In abbreviated form, the InputType and InputName are often omitted. The InputType and InputName are typically used only by read and write commands. Because a TSID may be long (especially when file names are used for the InputName), an Alias may be assigned to the time series. The TSID parameter is typically used in commands for the time series that is being processed. See also Alias.

TSID – When used as a command parameter the time series identifier indicates the time series to be processed. The TSID or alias can typically be specified. See also `Alias`.

TSID1 – Time series identifier for the first daily time series in a command. See the `fillDayTSFrom2MonthTSAnd1DayTS()` command.

TSID2 – Time series identifier for the second daily time series in a command. See the `fillDayTSFrom2MonthTSAnd1DayTS()` command.

TSID_D1 – Time series identifier for the first monthly time series in a command. See the `TS Alias = relativeDiff()` command.

TSID_D2 – Time series identifier for the second monthly time series in a command. See the `fillDayTSFrom2MonthTSAnd1DayTS()` command.

TSID_M1 – Time series identifier for the first yearly time series in a command. See the `fillDayTSFrom2MonthTSAnd1DayTS()` command.

TSID_M2 – Time series identifier for the second monthly time series in a command. See the `fillDayTSFrom2MonthTSAnd1DayTS()` command.

TSList – Indicates how the list of time series is determined. Typical values are AllTS (process all time series), AllMatchingTSID (process all time series having identifiers that match the TSID parameter), SelectedTS (process all time series that have been selected with the `selectTimeSeries()` and `deselectTimeSeries()` commands). This parameter is being phased in to allow more flexibility when processing time series.

TSProductFile – The name of a time series product (TSProduct) file. See the `processTSPublic()` command.

Units – The data units for a time series. See the `newTimeSeries()`, `TS Alias = readNWSRFSFS5Files()`, and `writeRiverWare()` commands.

Version – Indicates the file version, to allow the software to handle different data formats. See the `readStateModB()` command.

View – Indicate whether a product should be graphically previewed (as opposed to simply writing an output file). See the `processTSPublic()` command.

UseStoredProcedures – Indicates whether stored procedures should be used (versus straight SQL calls). This is being used to transition HydroBase queries to stored procedures. See the `openHydroBase()` command.

WarnIfDifferent – Indicates whether a warning should be generated if data differences are detected. See the `compareTimeSeries()` and `compareFiles()` commands.

WarnIfSame – Indicates whether a warning should be generated if data differences are NOT detected. See the `compareTimeSeries()` and `compareFiles()` commands.

Weight – Weight(s) used when processing time series. See the `TS Alias = weighTimeSeries()` command.

Where1, Where2 – Input filter information used when reading/querying data. See the `readHydroBase()` command.

Year – Specify year(s) of interest. See the `TS Alias = weighTimeSeries()` command.

This page is intentionally blank.

Command Reference

add()

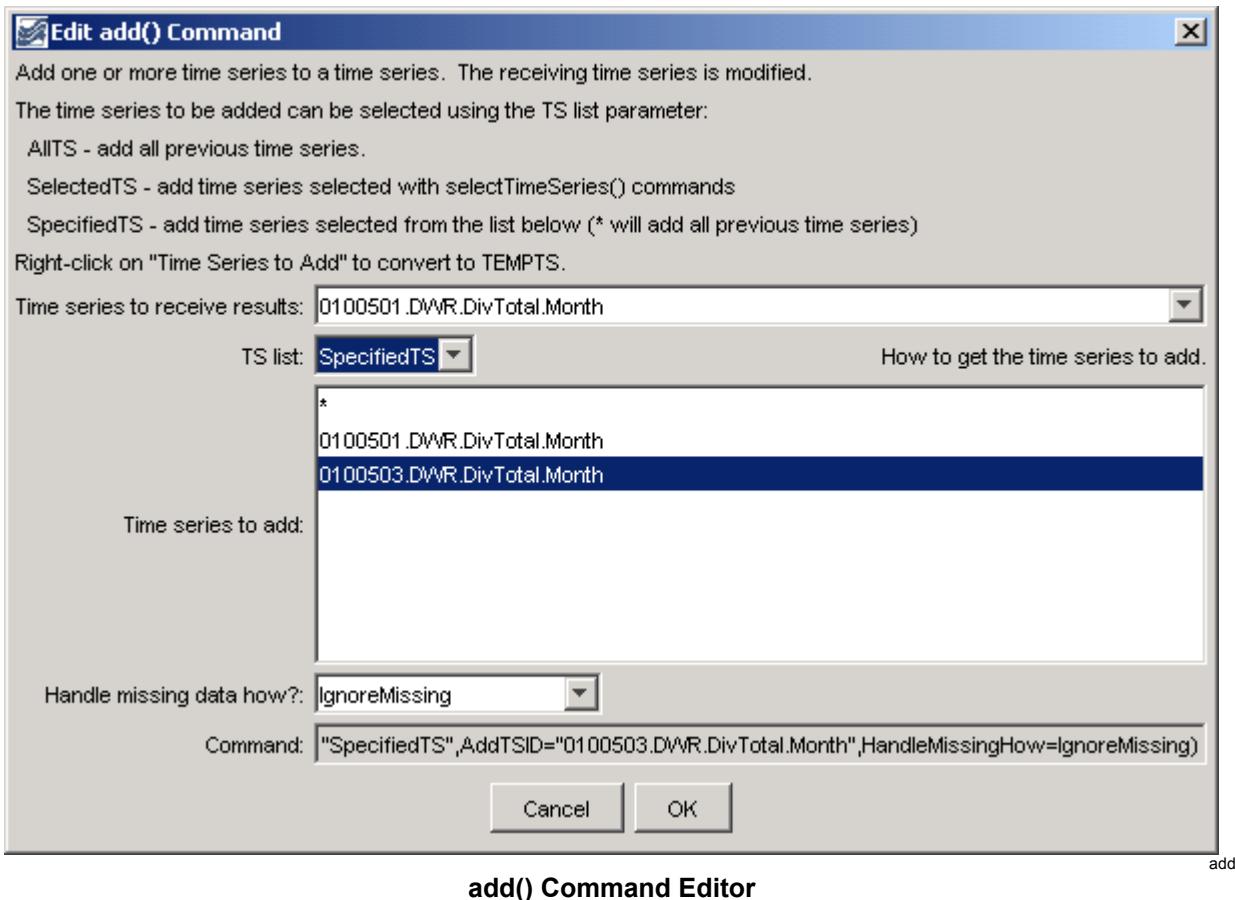
Add One or More Time Series to Another Time Series

Version 06.10.00, 2005-02-23, Color, Acrobat Distiller

The `add()` command adds regular interval time series. The receiving time series will be set to the sum of itself and all indicated time series. See also the `newTimeSeries()` command, which can create an empty time series to receive a sum. This command replaces the older `add()` command, which did not have the flag for how to handle missing data.

This command will generate an error if the time series do not have compatible units. If the units are compatible but are not the same (e.g., IN and FT), then the units of the part will be converted to the units of the sum before addition. Missing data in the parts can be ignored (do not set the sum to missing) or can result in missing values in the sum. The user should consider the implications of ignoring missing data. Time series being added must have the same data interval.

The following dialog is used to edit the command and illustrates the syntax of the command.



The command syntax is as follows:

```
add(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to receive the sum.	None – must be specified.
TSList	Indicates how the list of time series is specified, one of: <ul style="list-style-type: none"> • AllTS – all time series prior to the command. • SelectedTS – the time series are those selected with the selectTimeSeries() command. • SpecifiedTS – the specified list of time series given by the AddTSID parameter. 	AllTS
AddTSID	If the TSList parameter is SpecifiedTS, provide the list of time series identifiers (or alias) to add. Right-clicking on a time series allows toggling of the TEMPTS key word. This will cause the time series to add to be read and then discarded.	Must be specified if TSList=SpecifiedTS, ignored otherwise.
HandleMissingHow	Indicates how to handle missing data in a time series, one of: <ul style="list-style-type: none"> • IgnoreMissing – create a result even if missing data are encountered in one or more time series – this option is not as rigorous as the others • SetMissingIfOtherMissing – set the result missing if any of the other time series values is missing • SetMissingIfAnyMissing – set the result missing if any time series value involved is missing 	IgnoreMissing

A sample commands file is as follows:

```
# 0100501 - EMPIRE DITCH
0100501.DWR.DivTotal.Month~HydroBase
# 0100503 - RIVERSIDE CANAL
0100503.DWR.DivTotal.Month~HydroBase
add(TSID="0100501.DWR.DivTotal.Month",TSList="SpecifiedTS",
AddTSID="0100503.DWR.DivTotal.Month",HandleMissingHow=IgnoreMissing)
```

Command Reference

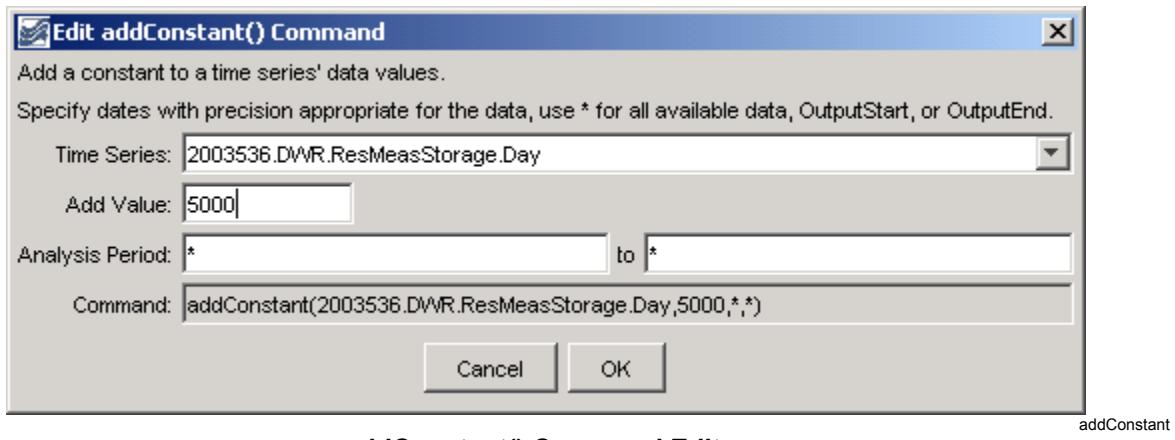
addConstant()

Add a Constant Value to all Data Values in a Time Series

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The addConstant() command adds a constant value to each data value in a time series within the specified period. This command is useful, for example, when a time series needs to be adjusted for a constant bias. Another example is to adjust a reservoir total volume time series by the dead pool storage in order to compute the active storage (or inverse). Missing data values will remain missing in the result.

The following dialog is used to edit the command and illustrates the syntax of the command.



addConstant() Command Editor

addConstant

The command syntax is as follows:

```
addConstant(TSID,AddValue,AnalysisStart,AnalysisEnd)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
AddValue	The data value to add to the time series.	None – must be specified.
AnalysisStart	The date/time to start analyzing data.	Full period if * is specified.
AnalysisEnd	The date/time to end analyzing data.	Full period if * is specified.

A sample commands file is as follows:

```
# 2003536 - CONTINENTAL RES
2003536.DWR.ResMeasStorage.Day~HydroBase
addConstant(2003536.DWR.ResMeasStorage.Day,5000,*,*)
```

This page is intentionally blank.

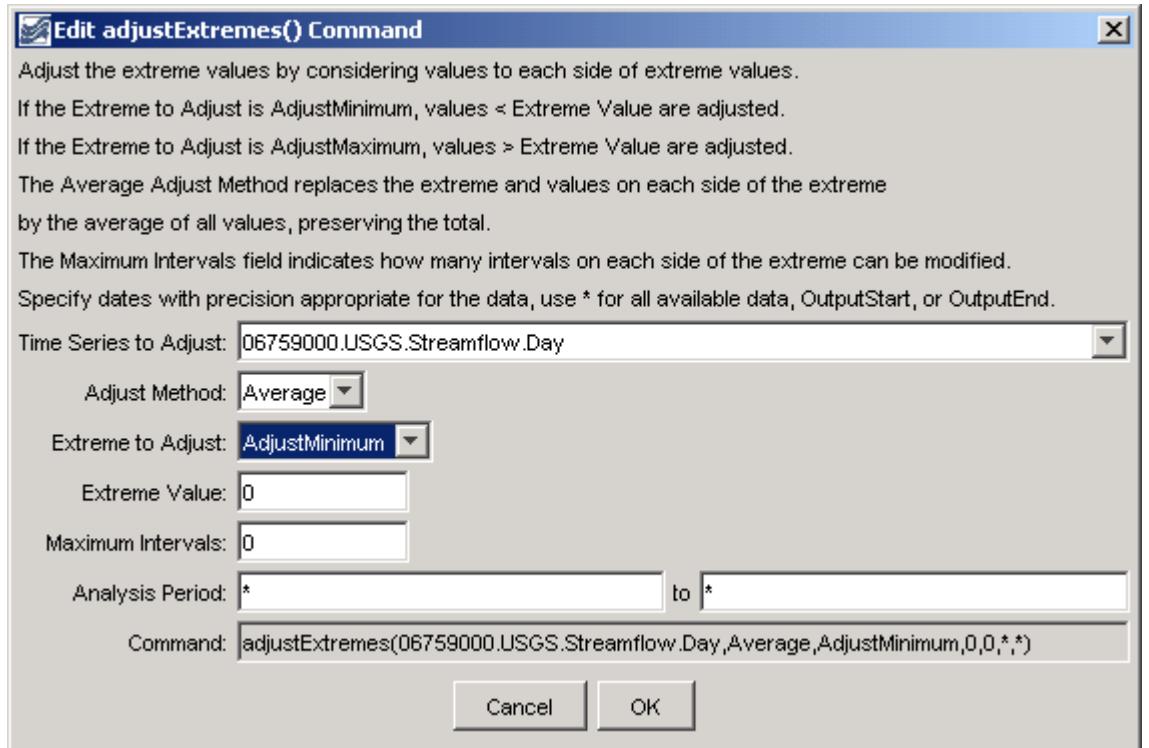
Command Reference

adjustExtremes()

Adjust the Extreme Values in Time Series Data

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `adjustExtremes()` command adjusts extreme values in time series (e.g., to remove negative values from a time series that can only have zero or positive values), while preserving “mass”. The following dialog is used to edit the command and illustrates the syntax of the command.



adjustExtremes() Command Editor

adjustExtremes

The command syntax is as follows:

```
adjustExtremes(TSID,AdjustMethod,ExtremeToAdjust,ExtremeValue,
MaxIntervals,AnalysisStart,AnalysisEnd)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
AdjustMethod	Currently only the Average adjust method is implemented, in which adjusted data values are set to the average over the adjusted period, necessary to maintain the total/mass of the original values. This method adjusts extreme values by considering neighboring values equally on each side of the point in question. When adjusting minimum values, neighboring values are added until the average is above the allowed extreme value, and all values that make up the sum are then set to the average value. Missing values remain missing and therefore this command should only be applied to filled data. If a satisfactory result cannot be reached within this limit, then the original values are not changed. Changed values are listed in the time series history, which is viewed with the time series properties. Applying the command will result in the time series having periods of constant value, with the length of the period being controlled by the magnitude of the extreme value.	None – must be specified.
ExtremeToAdjust	Indicate whether minimum (AdjustMinimum) or maximum (AdjustMaximum) values to be adjusted.	None – must be specified.
ExtremeValue	The extreme value that is the limit of acceptable values.	None – must be specified.
MaxIntervals	Indicates how many values on each side of a point are allowed to be examined. A value of zero indicates no limit on the number of points that can be examined.	
AnalysisStart	The date/time to start analyzing data.	Full period if * is specified.
AnalysisEnd	The date/time to end analyzing data.	Full period if * is specified.

A sample commands file is as follows:

```
# 06759000 - BIJOU CREEK NEAR WIGGINS, CO.
06759000.USGS.Streamflow.Day~HydroBase
adjustExtremes(06759000.USGS.Streamflow.Day,Average,AdjustMinimum,0,0,*,*)
```

Command Reference

DRAFT – THIS COMMAND IS UNDER DEVELOPMENT

analyzePattern()

Determine historical average patterns for monthly time series

Version 06.10.00, 2005-04-28, Color, Acrobat Distiller

The `analyzePattern()` command creates the pattern file for use with the `fillPattern()` command (see also `setPatternFile()`). Each time series to be processed is analyzed as follows:

1. Create a time series to contain the pattern identifiers for each month.
2. For each month, determine the monthly values for the time series (e.g., find all of the January values).
3. Rank the values in ascending order.
4. Evaluate the percentile rank information and assign in the pattern time series an appropriate pattern identifier. For example, if the percentile values are 25 and 75, assign the first pattern identifier to values < 25% of the total count, assign the second pattern identifier to values >= 25% and < 75%, and assign the third identifier to the values > 75%.

The resulting time series list is then written to a file. **This command is enabled for monthly data only.** See below for an example of a fill pattern file. One or more patterns can be included in each pattern file, similar to StateMod time series files (see the **StateMod Input Type** appendix), and multiple pattern files can be used, if appropriate.

```
# Years Shown = Water Years
# Missing monthly data filled by the Mixed Station Method, USGS 1989
# Time series identifier      = 09034500.CRDSS_USGS.QME.MONTH.1
# Description                  = COLORADO RIVER AT HOT SULPHUR SPRINGS, CO.
# -e-----eb-----eb-----eb-----eb-----eb-----eb-----eb-----eb-----eb-----eb-----eb-----e
10/1908 -         9/1996 ACFT WYR
1909 09034500     AVG     AVG     AVG     WET     WET     AVG     AVG     AVG     WET     WET     WET     WET
1910 09034500     WET     WET     WET     WET     WET     AVG     AVG     AVG     AVG     AVG     AVG
1911 09034500     AVG     AVG     WET     AVG     AVG     WET     WET     WET     AVG     AVG     WET
1912 09034500     WET     WET     WET     WET     AVG     AVG     WET     WET     WET     WET     WET
...committed...
```

The following dialog is used to edit the `analyzePattern()` command and illustrates the syntax of the command.

analyzePattern() Command Editor

analyzePattern

The command syntax is as follows:

```
analyzePattern(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSList	Indicate how the list of time series to process should be determined, one of: <ul style="list-style-type: none"> ▪ AllTS – all time series ▪ AllMatchingTSID – all time series that have identifiers matching the given TSID parameter. ▪ SelectedTS – all time series that have been selected with <code>selectTimeSeries()</code> commands. 	None – must be specified.
TSID	The time series identifier or alias for the time series to be analyzed. A pattern containing the * wildcard character can be used to process multiple time series (e.g., * or 29*).	Must be specified if the TSList parameter has a value of AllMatchingTSID.
Method	Method used to determine the patterns. Currently only Percentile is recognized.	Percentile
Percentile	A comma-separated list of percentiles for cutoffs, used when Method=Percentile. Values should be 0 to 100.	None – must be specified.
PatternID	The pattern identifiers to use, corresponding to the percentiles.	None – must be specified.
OutputFile	Output file to write, which will contain the pattern information. Currently only the StateMod pattern file format is supported.	None – must be specified.

A sample commands file is as follows:

```
# Set the output period...
setOutputPeriod(1900-01,2004-12)
# Read the time series to fill...
readStateMod("x.stm")
# Analyze the time series to determine the pattern...
analyzePattern(TSList=AllTS,Method=Percentile,PatternID="DRY,AVG,WET",
Percentile="25,75",OutputFile="basin.pat")
```

Command Reference

ARMA()

Lag and Attenuate a Time Series Using AutoRegressive Moving Average

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The ARMA() command lags and attenuates a time series (e.g., to route a streamflow time series downstream). This approach preserves the “mass” of the data. The general equation for ARMA is:

$$O_t = a_1*O_{t-1} + a_2*O_{t-2} + \dots + a_p*O_{t-p} + b_0*I_t + b_1*I_{t-1} + \dots + b_q*I_{t-q}$$

Where:

t = time step

O_t = output value at time t

I_t = input value at time t

a, b = ARMA coefficients

and the p and q values indicate the degree of the equation: ARMA(p, q).

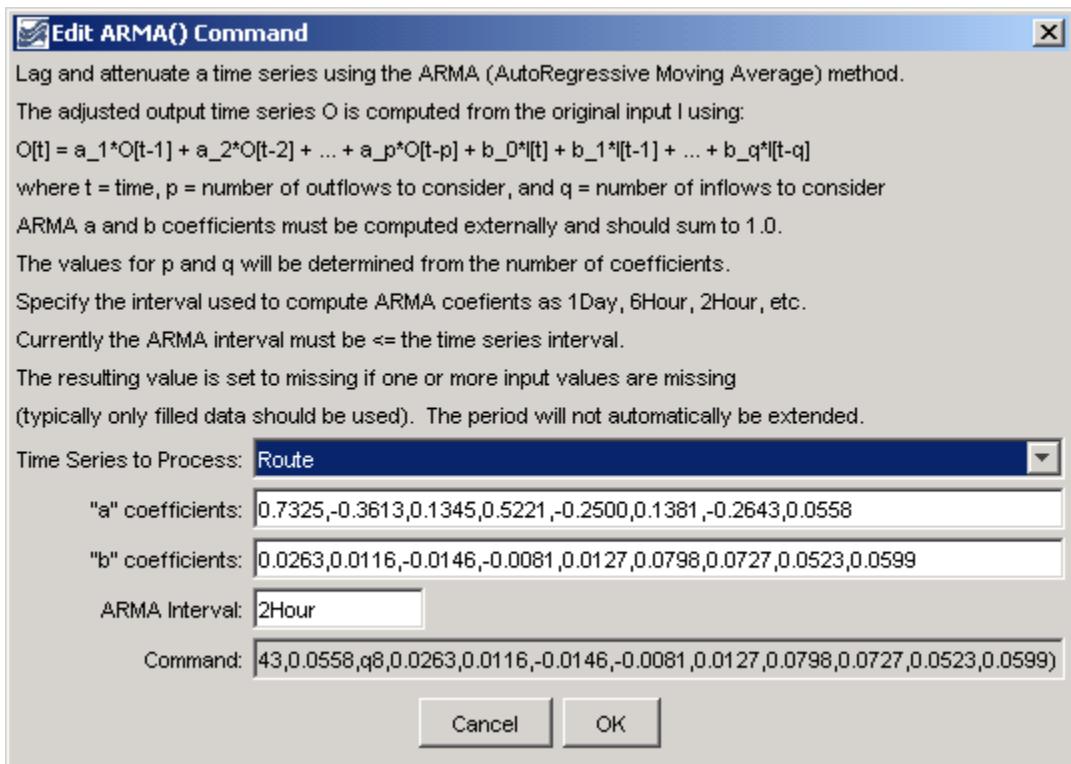
The ARMA coefficients are determined by analyzing historical data and may be developed using a data interval that is different than the data interval of the time series that is being manipulated. The coefficients are typically computed by an external analysis program (TSTool does not perform this function).

The time series to process can have any interval. The a and b coefficients are listed in the dialog from left-most to right-most in the equation. Note that there are p a -coefficients and $(q + 1)$ b -coefficients (because there is a b -coefficient at time t_0). The interval used to compute the ARMA coefficients can be different from the data interval but the data and ARMA intervals must be divisible by a common interval. The ARMA algorithm is executed as follows:

1. The data and ARMA intervals are checked and if they not the same, the data are expanded by duplicating each value into a temporary array. For example, if the data interval is 6Hour and the ARMA interval is 2Hour, each data value is expanded to three data values (2Hour values). If the data interval is 6Hour and the ARMA interval is 10Hour, each data value is expanded to three data values (2Hour values).
2. The ARMA equation is applied at each point in the expanded data array. However, because the ARMA coefficients were developed using a specific interval, only the data values at the ARMA interval are used in the equation. For example, if the expanded data array has 2Hour data and the ARMA interval is 10Hour, then every fifth value will be used (e.g., t corresponds to the “current” value and $t - 1$ corresponds to the fifth value before the current value). Because the ARMA algorithm depends on a number of previous terms in both the input and output, there will be missing terms at the beginning of the data array and in cases where missing data periods are encountered. Ideally ARMA will be applied to filled data and only the initial conditions will be an issue. In this case the output period should ideally be less than the total period so that the initial part of the routed time series can be ignored. In cases where O values are missing, the algorithm first tries to use the I values. If any values needed for the result are missing, the result is set to missing.

3. The final results are converted to a data interval that matches the original input, if necessary. If the original data interval and the ARMA interval are the same, no conversion is necessary. For example, if the original data interval is 6Hour and the ARMA interval is 10Hour, then the expanded data interval will be 2Hour. Consequently, three sequential expanded values are averaged to obtain the final 6Hour time series.

The following dialog is used to edit the command and illustrates the command syntax.



ARMA

ARMA() Command Editor

The command syntax is as follows:

```
ARMA(TSID,ARMAInterval,pP,a1,...,aP,qQ,b0,...,bQ)
```

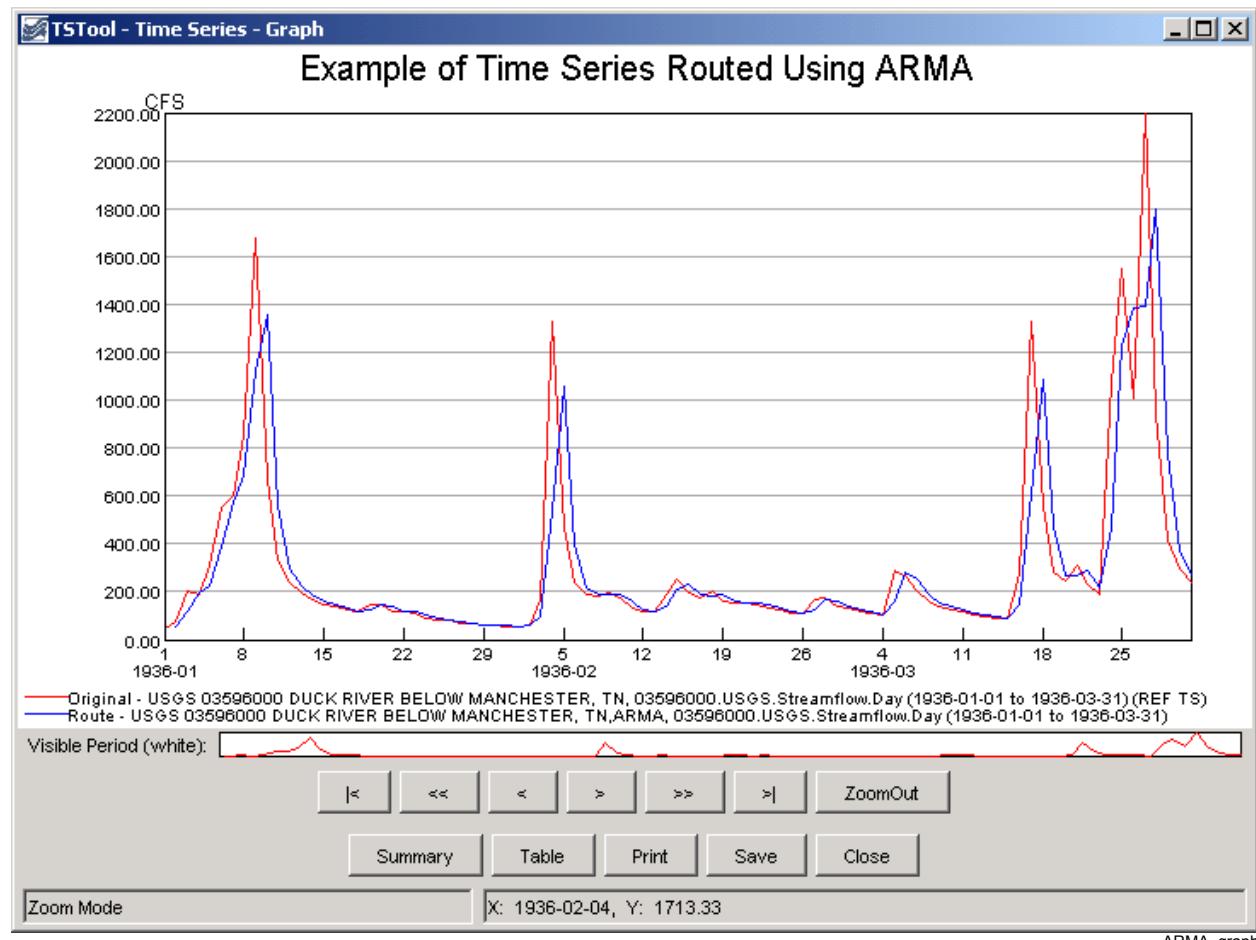
Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
ARMAInterval	The ARMA interval to use in the analysis	None – must be specified.
pP	Indicates the p-degree of the equation (P = number of a coefficients).	None – must be specified.
a1 ,..., aP	a coefficients.	None – must be specified.
qQ	Indicates the q-degree of the equation (Q = number of b coefficients - 1).	None – must be specified.
b0 ,..., bQ	b coefficients.	None – must be specified.

A sample commands file is as follows:

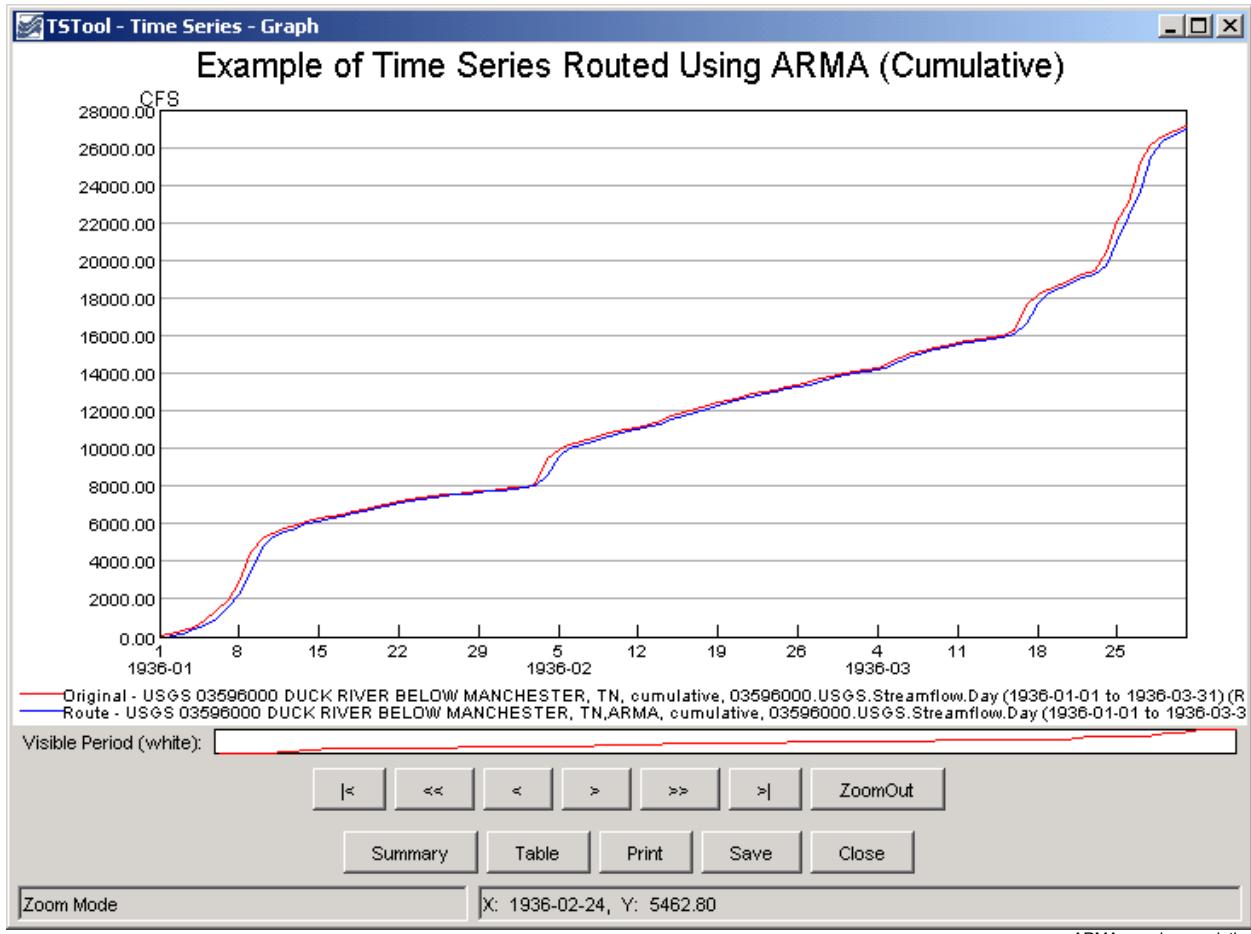
```
setOutputPeriod(1936-01-01,1936-03-31)
TS Original = readUsgsNwis("G03596000.in1",*,*)
TS Route = copy(Original)
# TS Route = readUsgsNwis("G03596000.in1",*,*)
ARMA(Route,2Hour,p8,0.7325,-0.3613,0.1345,0.5221,-0.2500,0.1381,
-0.2643,0.0558,q8,0.0263,0.0116,-0.0146,
-0.0081,0.0127,0.0798,0.0727,0.0523,0.0599)
writeDateValue(OutputFile="TS_ARMA.out")
```

The following figure shows the original and routed time series.



The `cumulate()` command can be used to verify mass balance of the original and routed time series (see the `cumulate()` command discussion below). For example, insert a `cumulate(*,CarryForwardIfMissing)` command near the end of a command file.

The following figure shows the time series from the previous graph, this time as cumulative time series.



Command Reference

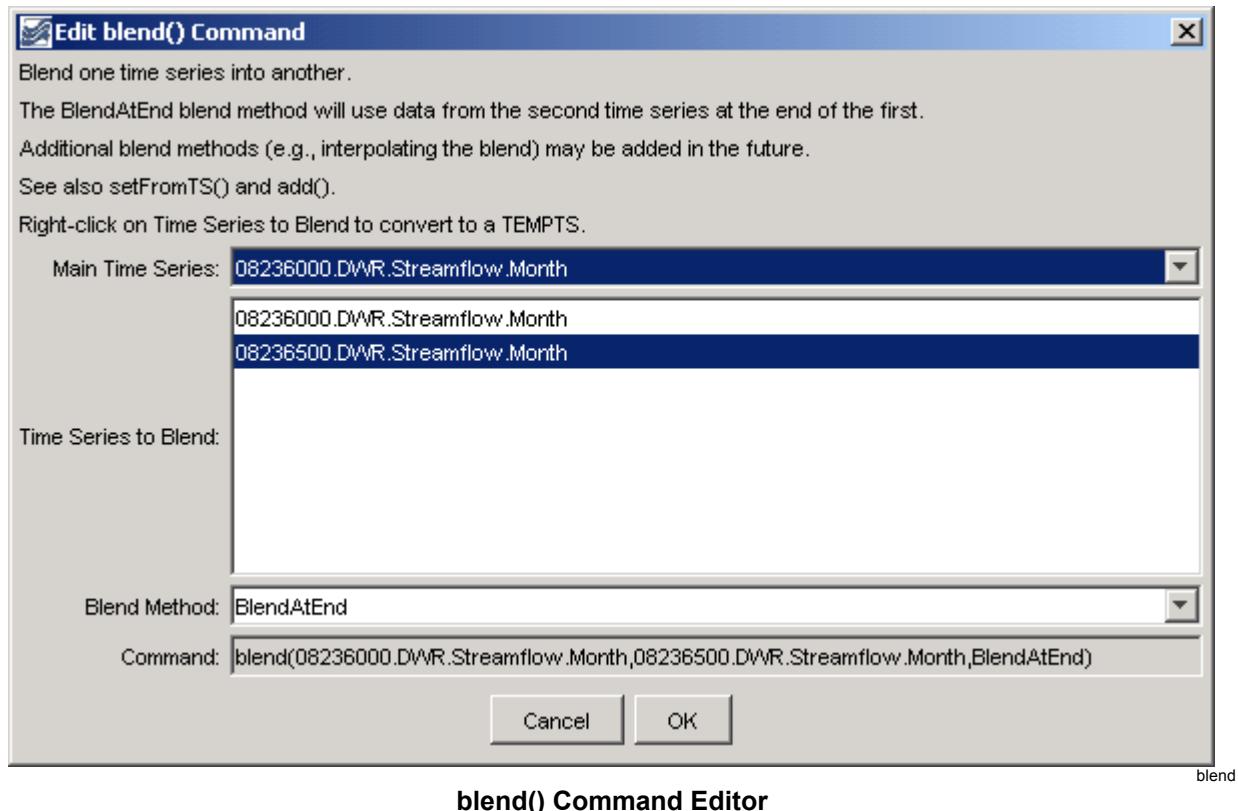
blend()

Append a Time Series to the End of Another Time Series

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `blend()` command blends one time series into another, extending the first time series period if necessary. This is typically used for combining time series for a station that has been renamed or to blend historic and real-time data. See also the `setFromTS()` and `add()` commands. The `blend()` command ensures that single values are used whereas `add()` may add values at the same date/time. The `setFromTS()` does not extend the period.

The following dialog is used to edit the command and illustrates the syntax of the command.



blend() Command Editor

The command syntax is as follows:

```
blend(TSID,TSID2,BlendMethod)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
BlendTSID	The time series identifier or alias for the time series to be blended to the first time series. Right-clicking on a time series to blend allows toggling of the TEMPTS key word. This will cause the time series to add to be read and then discarded, simplifying the input file. However, to insert the time series the first time, the time series must have been listed in the commands file as a time series identifier.	None – must be specified.
BlendMethod	The date/time to start analyzing data. The only blend method currently supported is BlendAtEnd, resulting in the main time series having the other time series attached to the end of its period.	None – must be specified.

A sample commands file is as follows:

```
# 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR
08236000.DWR.Streamflow.Month~HydroBase
# 08236500 - ALAMOSA RIVER BELOW TERRACE RESERVOIR
08236500.DWR.Streamflow.Month~HydroBase
blend(08236000.DWR.Streamflow.Month,08236500.DWR.Streamflow.Month,BlendAtEnd)
```

Command Reference

DRAFT – THIS COMMAND IS UNDER DEVELOPMENT

TS X = changeInterval()

Create a New Time Series by Changing a Time Series Data Interval

Version 06.10.00, 2005-02-18, Color, Acrobat Distiller

A `changeInterval()` command can be inserted to change the data interval of an existing time series, resulting in a new time series. The majority of the original header data (e.g., description, units) are maintained in the new time series. Time series data values have a time scale, which is accumulated (e.g., volume), mean (e.g., flow), or instantaneous (e.g., temperature). Changing the interval can also result in a change in the time scale (e.g., converting instantaneous values to a mean value). Currently, the time scale is NOT determined from the data type and interval and must be specified as ACCM, MEAN, or INST. For regular time series, the data intervals must be aligned so that each larger interval aligns with the endpoints of the corresponding smaller intervals (e.g., the ends of 6-hour intervals align with the daily interval). **IS THIS BEING CHECKED IN THE CODE?** The following conversions are currently supported, with a description of the conversion process: **CODE NEEDS TO THROW EXCEPTIONS IF A CONVERSION IS NOT SUPPORTED.** The period of the time series will include any intervals where there were one or more values in the original data???. In TSTool an overall output period can be set (`setOutputPeriod()`) – we may need to deal with this later.

Small Interval ACCM (Accumulation) to Large Interval ACCM (Accumulation)

Changing the interval for small interval accumulated data to large interval accumulated data involves summing the small interval data values for the period that overlaps the large interval.

Accumulated data have a timestamp corresponding to the interval-end for the accumulation. Conversions involving time intervals that have zero values (e.g., Hour 0, Minute 0) result in a perceived shift in time because the zero occurs on the boundary between larger intervals. The following examples illustrate the accumulation for common cases. In cases where an accumulation jumps over two or more interval categories (e.g., minute to day), the accumulation occurs as if the two intermediate accumulations occurred in succession. In the following examples, the general representation is shown first, followed by an example where appropriate.

NHour to Day (6Hour to Day example, i equals the hour multiplier):

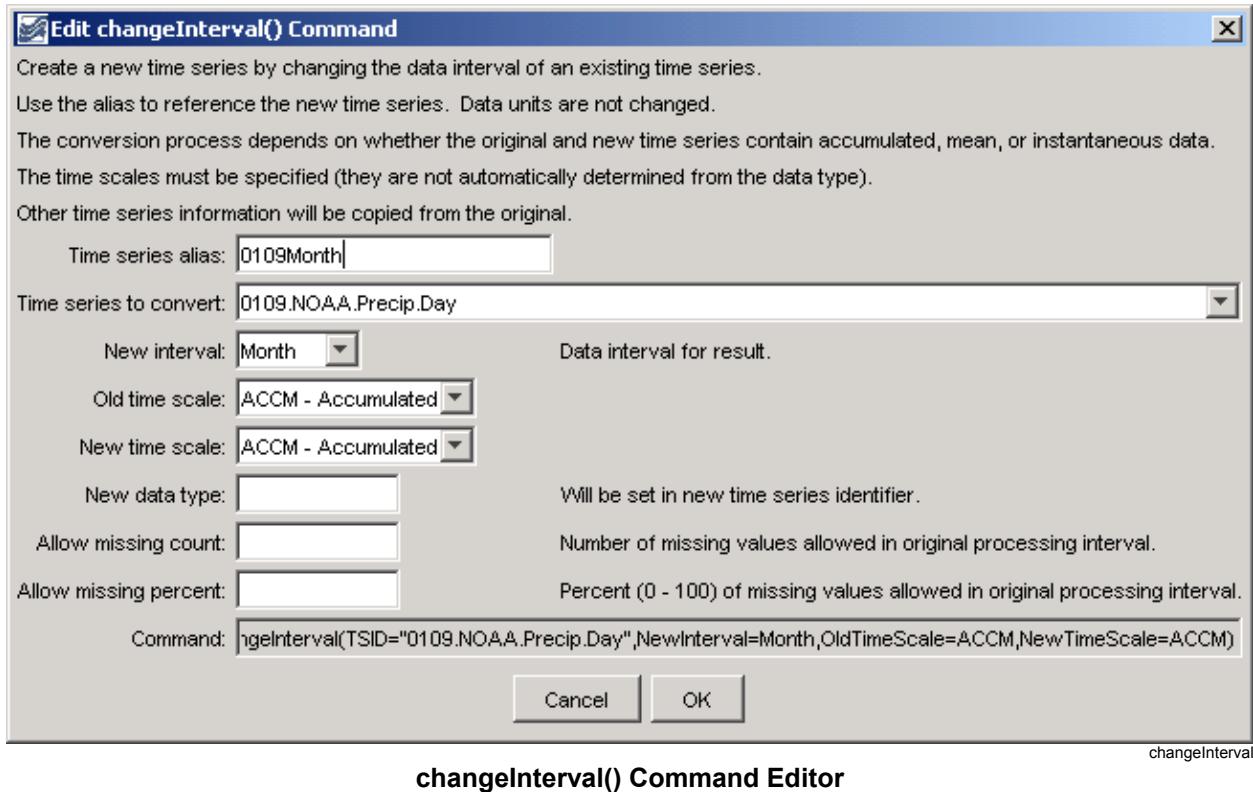
Day 1, Hour 0	Day 1, Hour i	Day 1, Hour $2i$	Day 1, Hour $3i$	Day 2, Hour 0
Day 1, Hour 0	Day 1, Hour 6	Day 1, Hour 12	Day 1, Hour 18	Day 2, Hour 0
Day 1 accumulation				

NDay to Month (example for a month with 30 days):

Month 1, Day 1	Month 1, Day 30
Month 1 accumulation				

Need to insert examples of ALL supported conversions so users have NO QUESTIONS about what is done.

The following dialog is used to edit the command and illustrates the syntax for the command.



changeInterval() Command Editor

The command syntax is as follows:

```
TS X = changeInterval(param=value,...)
```

Command Parameters

Parameter	Description	Default
Alias	The alias to assign to the new time series. The time series identifier for the new time series will be the same as the original time series, with the new interval and optionally a new data type (see the NewDataType parameter).	None – must be specified.
TSID	The time series identifier or alias for the original (old) time series.	None – must be specified.
NewInterval	The data interval for the new time series, from the provided choices. For example: 6Hour, Day, Month, Year.	None – must be specified.
OldTimeScale	The time scale for the original time series, one of: ACCM – accumulated data INST – instantaneous data MEAN – mean data In the future, this parameter may be made optional if the time scale can be determined from the data type.	None – must be specified.
NewTimeScale	The time scale for the new time series (see OldTimeScale for possible values). In the future, this parameter may be made optional if the time scale can be determined from the data type.	None – must be specified.
NewDataType	The data type for the new time series. This will be set in the identifier of the new time series.	Use the data type from the original time series.
AllowMissingCount	The number of missing values allowed in the source interval(s) in order to produce a result. For example, if converting daily data to monthly, a value of 5 would allow <= 5 missing daily values and still compute the result. This capability should be used with care because it may result in data that are not representative of actual conditions.	0 – do not allow any missing data in the source data when computing a result.

A sample commands file is as follows (convert a daily precipitation time series to monthly total):

```
0109.NOAA.Precip.Day~HydroBase
TS 0109Month =
changeInterval(TSID="0109.NOAA.Precip.Day", NewInterval=Month, OldTimeScale=ACCM, NewTimeScale=ACCM)
```

This page is intentionally blank.

Command Reference

compareFiles()

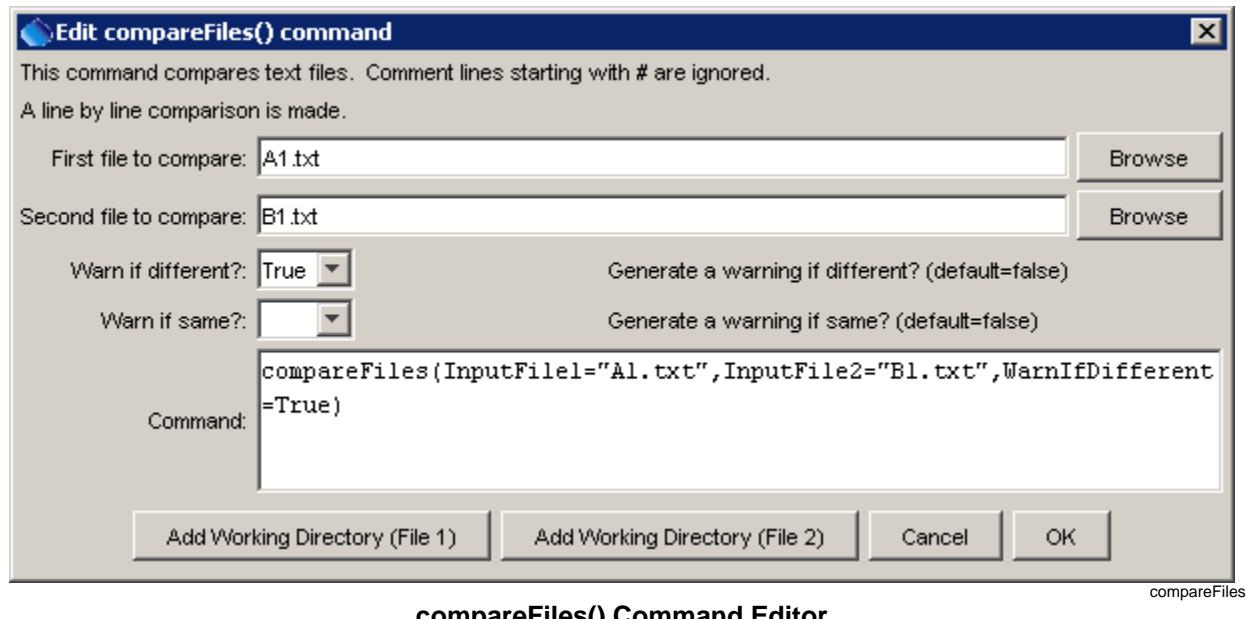
Compare text files to determine whether they are different

Version 06.18.00, 2006-05-03, Color, Acrobat Distiller

The `compareFiles()` command compares text files to determine data differences. For example, the command can be used to compare old and new files produced by a software process.

Each line in the file is compared. Lines beginning with # are treated as comment lines and are ignored. Therefore, only non-comment lines are compared. Differences and simple statistics are printed to the log file. A warning can be generated if a difference is detected or if no differences are detected (see also the `compareTimeSeries()` and `runCommands()` commands).

The following dialog is used to edit the command and illustrates the syntax for the command.



The command syntax is as follows:

```
compareFiles (param=value,...)
```

Command Parameters

Parameter	Description	Default
InputFile1	The name of the first file to read. Enclose the name in double quotes to protect whitespace and special characters.	None – the file name is required.
InputFile2	The name of the second file to read. Enclose the name in double quotes to protect whitespace and special characters.	None – the file name is required.
WarnIfDifferent	If True and at least one difference is detected, a warning will be generated by the command, which will result in software like TSTool displaying a warning. If False, only status messages are written to the log file. The warning is useful if it is critical to detect any difference in the files.	Do not generate a warning if the files are different. Differences are printed to the log file.
WarnIfSame	If True and no differences are detected, a warning will be generated by the command, which will result in software like TSTool displaying a warning. If False, only status messages are written to the log file. The warning is useful if it is critical to detect files that are the same.	Do not generate a warning if the files are the same.

The following example illustrates how two files can be compared. For example, use similar commands to compare results from two model runs or two database queries:

```
compareFiles(InputFile1="A1.txt", InputFile2="B1.txt", WarnIfDifferent=True)
```

Command Reference

compareTimeSeries()

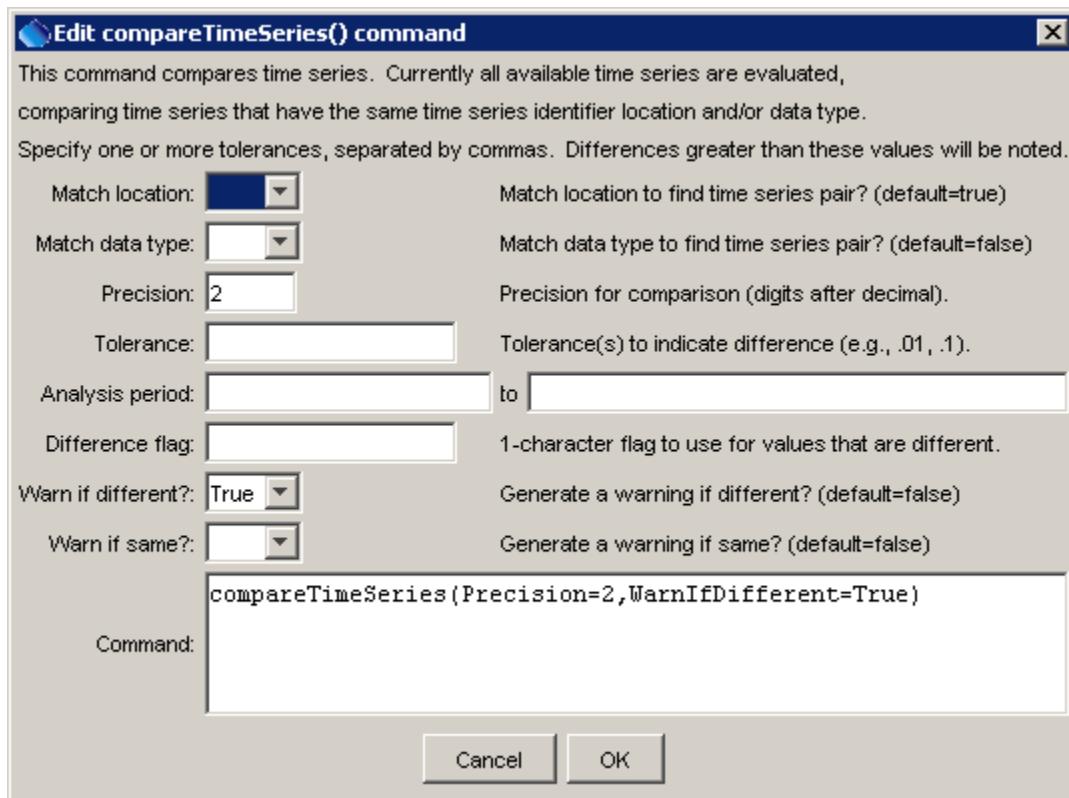
Compare time series to find data value differences

Version 06.18.00, 2006-05-02, Color, Acrobat Distiller

The `compareTimeSeries()` command compares time series to determine data differences. Currently time series header information is NOT compared – only data values are compared. It is designed to process many time series in bulk fashion. For example, read commands can be used to read time series from two different versions of a database, or from two files. Time series to compare are determined by trying to match each available time series with another time series in the list (ignoring itself); consequently, the list of time series should contain only pairs of time series.

Time series that are matched by TSID location and/or data type are compared value by value, with the differences computed as the value from the second time series minus the value from the first time series. The values can be rounded based on a specified precision. It may be important to read each set of time series from files to ensure that final round off is consistent. The checks occur by comparing the difference to one or more specified tolerances. Differences and simple statistics are printed to the log file. Values that are different can optionally be tagged with a character flag, for use with the graphing package. A warning can be generated if a difference is detected, or if no differences are detected (see also the `compareFiles()` and `runCommands()` commands).

The following dialog is used to edit the command and illustrates the syntax for the command.



compareTimeSeries

compareTimeSeries() Command Editor

The command syntax is as follows:

```
compareTimeSeries (param=value, ...)
```

Command Parameters

Parameter	Description	Default
MatchLocation	Match the location part of time series identifiers when matching time series to compare.	True
MatchDataType	Match the data type part of time series identifiers when matching time series to compare.	False
Precision	When comparing data values, round the values to the given precision. For example, a precision of 2 will round to the hundredths place. This can be used to do comparisons on the lowest precision of the available time series.	Compare the available values without rounding.
Tolerance	Specify a comma-separated list of values. The difference in the time series values will be compared to the tolerances and messages printed to the log file.	A tolerance of zero will be used to detect differences.
AnalysisStart	The starting date/time to analyze for differences. Specify a date/time of appropriate precision for the time series or <code>OutputStart</code> to use the output start.	Analyze all available data.
AnalysisEnd	The ending date/time to analyze for differences. Specify a date/time of appropriate precision for the time series or <code>OutputEnd</code> to use the output end.	Analyze all available data.
DiffFlag	Specify as a single character to append a flag to the data flags for the time series. Each value that is different is flagged in both time series that are compared. The flag can be displayed by the graphing package. This is useful for verification processes.	Do not flag data.
WarnIfDifferent	If <code>True</code> and at least one difference is detected, a warning will be generated by the command, which will result in software like TSTool displaying a warning. If <code>False</code> , only status messages are written to the log file. The warning is useful if it is critical to detect any change in the time series.	Do not generate a warning if time series are different. Differences are printed to the log file.
WarnIfSame	If <code>True</code> and no differences are detected, a warning will be generated by the command, which will result in software like TSTool displaying a warning. If <code>False</code> , only status messages are written to the log file. The warning is useful if it is critical to detect that time series are the same.	Do not generate a warning if time series are the same.

The following example illustrates how time series from two files can be compared. For example, use similar commands to compare results from two model runs or two database queries:

```
# Example to compare files. Since they are different, a warning will be generated.  
readDateValue("RawData1.dv")  
readDateValue("RawData1Scaled.dv")  
compareTimeSeries(Precision=2,WarnIfDifferent=True)
```

The following example compares matching time series for the full available period, doing checks for several tolerances:

```
compareTimeSeries(Precision=2,Tolerance="0,.1,.5,1",DiffFlag="x")
```

The following example compares data only within the output period, as specified by the `setOutputPeriod()` command:

```
compareTimeSeries(Precision=2,Tolerance="0,.1,.5,1",  
AnalysisStart="OutputStart",AnalysisEnd="OutputEnd",DiffFlag="x")
```

This page is intentionally blank.

Command Reference

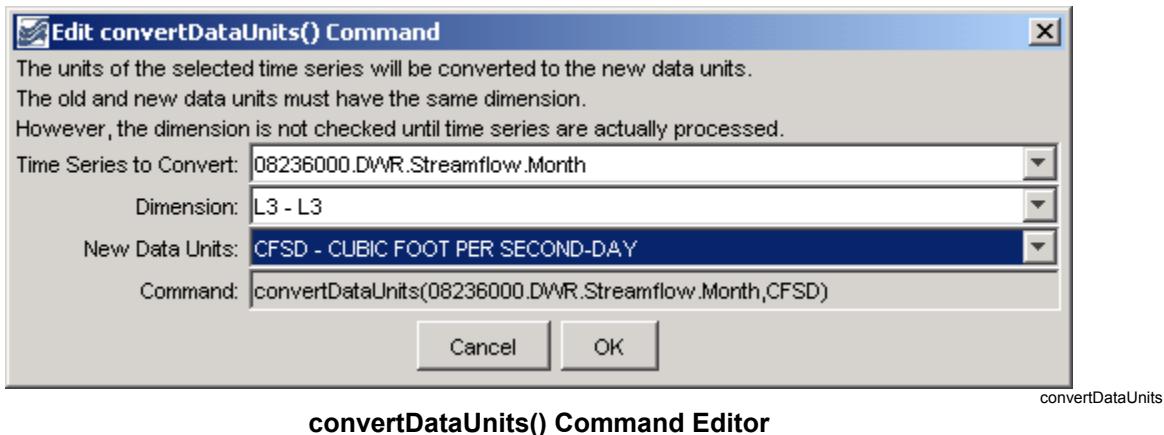
convertDataUnits()

Convert Time Series Data from One Set of Units to Another

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `convertDataUnits()` command converts the data units for a time series (e.g., before output to a file). Note that some read and write commands also allow units to be converted.

The following dialog is used to edit the command and illustrates the syntax of the command.



convertDataUnits() Command Editor

The **Dimension** choice should be selected to narrow the list of available units to the appropriate dimension. Next, select the **New Data Units** for the time series. The list of available data units is taken from the information described in the TSTool *DATAUNITS* file (see the TSTool **Installation and Configuration** appendix for more information). If desired units are not available, contact the TSTool developers to suggest adding units to the *DATAUNITS* file or edit the command manually after initial creation.

Note that the dialog cannot display the current units for the time series because the units are not available until time series are actually processed – commands are edited before processing.

The command syntax is as follows:

```
convertDataUnits(TSID,NewUnits)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A single time series identifier/alias can be selected, or the asterisk (*) can be selected to convert all time series.	None – must be specified.
NewUnits	The new data units.	None – must be specified.

A sample commands file is as follows:

```
# 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR
08236000.DWR.Streamflow.Month~HydroBase
convertDataUnits(08236000.DWR.Streamflow.Month,CFSD)
```

Command Reference

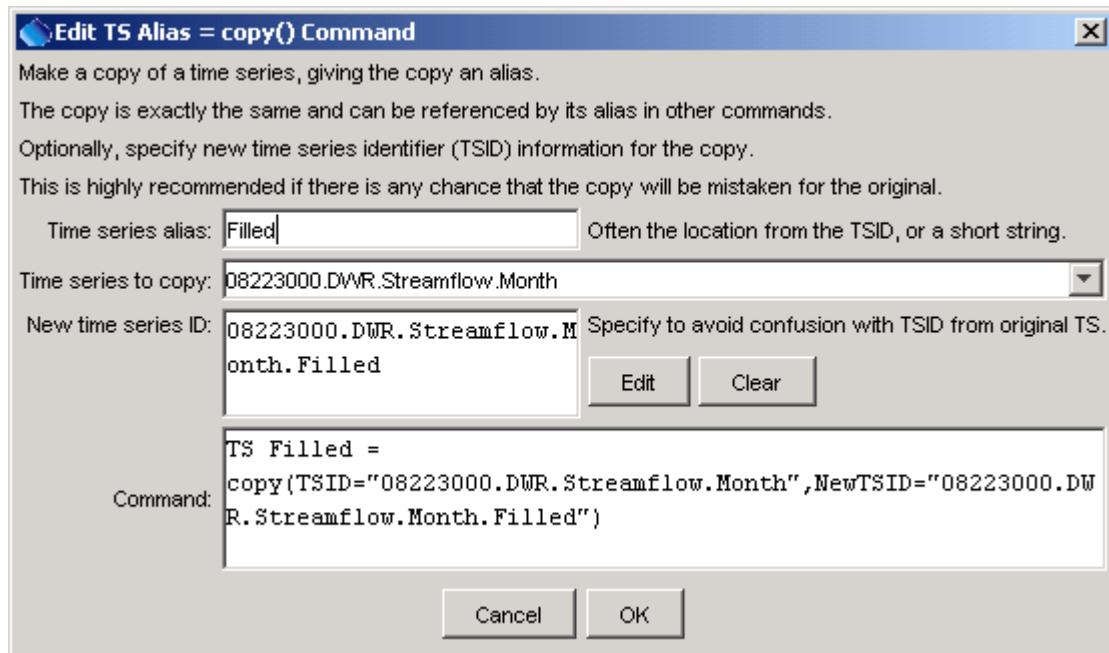
TS Alias = copy()

Create a new time series as a copy of a time series

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The TS Alias = copy() command creates a copy of an existing time series, assigning an alias to the result. The copy is an exact copy except that the alias is different. The alias can then be used for further time series manipulation. A copy of a time series is useful when data filling or other manipulation will occur and the original time series needs to be maintained for graphing or other purpose.

The following dialog is used to edit the command and illustrates the syntax for the command.



TS Alias = copy() Command Editor

copy

The command syntax is as follows:

```
TS Alias = copy(param=value,...)
```

Command Parameters

Parameter	Description	Default
Alias	The alias to assign to the new copy, which can be used instead of the TSID by other commands.	None – must be specified.
TSID	The time series identifier or alias of the time series to copy. The time series will be found by searching backwards from the copy command.	None – must be specified.
NewTSID	A new time series identifier to assign to the copy. This is useful to avoid confusion with the original time series. Use the Edit button to edit the time series identifier parts. The data interval must match that of the original time series.	Copy the original time series TSID.

A sample commands file is as follows:

```
# 08223000 - RIO GRANDE RIVER AT ALAMOSA
08223000.DWR.Streamflow.Month~HydroBase
TS Filled = copy(TSID="08223000.DWR.Streamflow.Month",
NewTSID="08223000.DWR.Streamflow.Month.Filled")
```

Command Reference

createFromList()

Create One or More Time Series from a List File

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

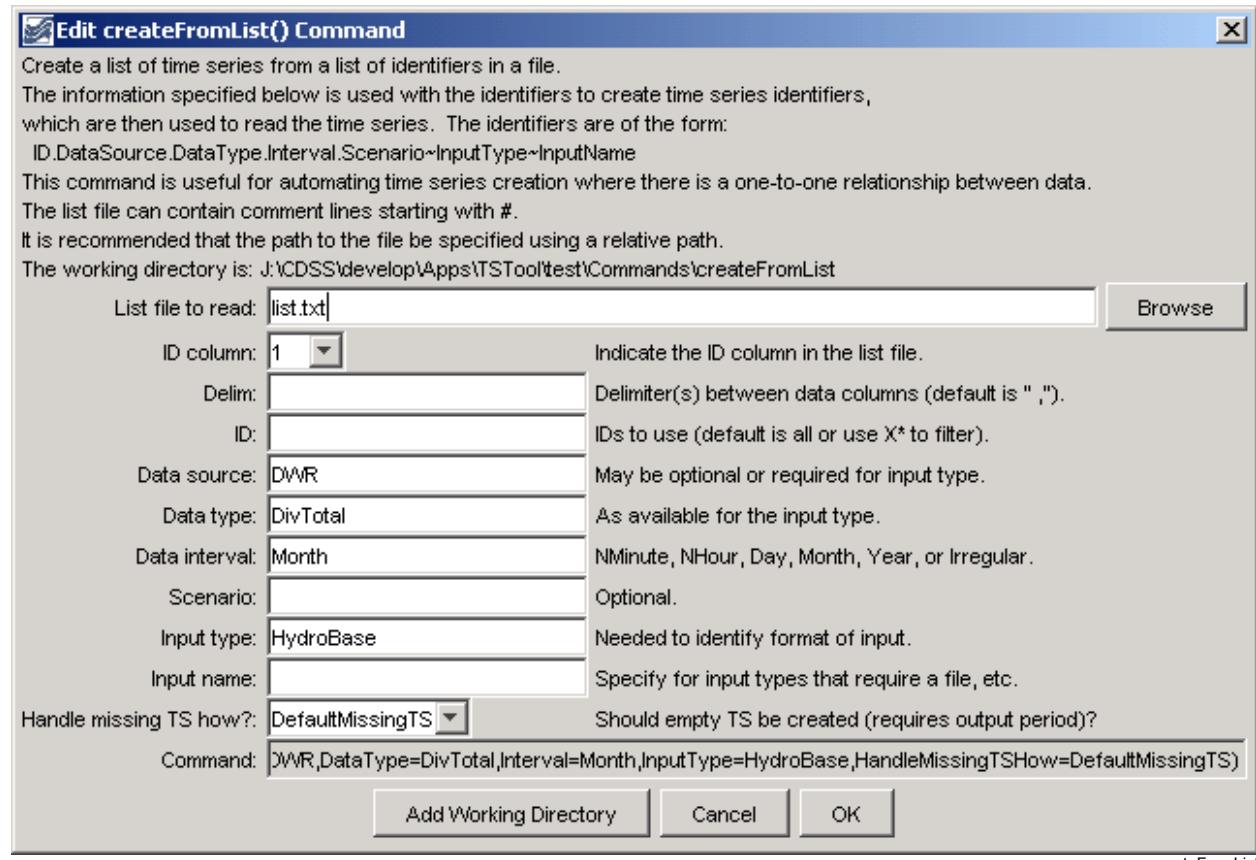
A `createFromList()` command can be inserted to create one or more time series using identifiers in a list file, an example of which is shown below:

```
# Example list file. Comments start with the # character.  
500501  
500502
```

This command can streamline processing in the following situations:

- A list of identifiers may have been generated from a database query and saved to a file.
- A list of identifiers may have been extracted from a model data set..

The following dialog is used to edit the command and illustrates the syntax of the command. TSTool reads the list file and internally creates a list of time series identifiers, using time series identifier information supplied by the `createFromList()` command. TSTool queries each time series, which can then be processed further.



createCommandEditor

The command syntax is as follows:

```
createFromList(param=value,param=value,...)
```

Command Parameters

Parameter	Description	Default
ListFile	The name of the list file to read, surrounded by double quotes.	None – must be specified.
IDCol	The column (1+) in the list file containing the location identifiers to use in time series identifiers.	1
Delim	The delimiter characters that separate columns in the list file.	Comma and space. If only a space is used, surround with another character that is unlikely to be found so that the space is not discarded as white space.
ID	Indicate a pattern to use filter the list of identifiers file.	Process all identifiers.
DataSource	The data source in the time series identifier, appropriate for InputType.	May or may not be required, depending on the input type. Refer to the input type appendices.
DataType	The data type in the time series identifier, as appropriate for InputType.	Usually required for an input type. Refer to the input type appendices.
Interval	Data interval in the time series identifier, using standard values.	None – must be specified.
Scenario	Scenario in the time series identifier.	Usually not required.
InputType	The input type in the time series identifier.	None – must be specified. Refer to the input type appendices.
InputName	The input name in the time series identifier.	Typically only required if the input type requires a file name.
HandleMissingTShow	Indicates how to handle missing time series.	If set to DefaultMissingTS, time series that are not found will be initialized to missing data. If set to IgnoreMissingTS, time series will not be created.

A sample commands file is as follows:

```
# Read monthly diversion total from HydroBase for the structures in the list
# file. The data source is set to DWR because data source is saved in
# HydroBase.

createFromList(ListFile="list.txt",IDCol=1,DataSource=DWR,DataType=DivTotal,
Interval=Month,InputType=HydroBase,HandleMissingTShow=DefaultMissingTS)
```

This command replaces the older -slist, -data_interval, and -data_type commands.

Command Reference

createTraces()

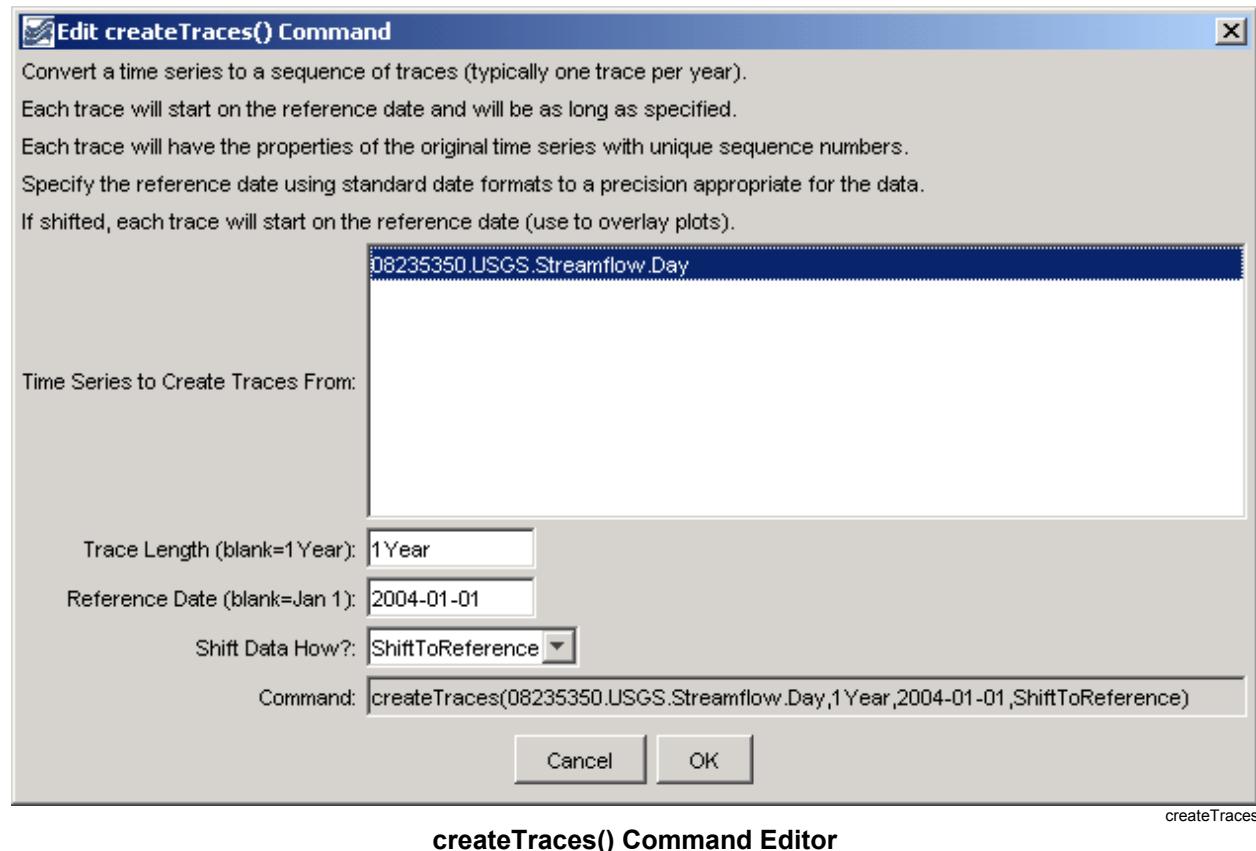
Create One or More Traces from a Time Series

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

A `createTraces()` command can be inserted to convert a time series into a series of traces, which can then be manipulated as independent time series. This command has been implemented on a limited basis will be enhanced in the future. For example, shifts currently ensure that data are sequential; however, there is no option to create traces and make sure that February 29 is always February 29 (currently the reference date controls whether the trace will have leap year or not).

New traces are created using the time series identifier information from the original time series. The original time series is not modified. To avoid confusion, the original time series may need to be discarded after processing or use a temporary time series.

The following dialog is used to edit the command and illustrates the syntax of the command:



createTraces() Command Editor

createTraces

The command syntax is as follows:

```
createTraces(TSID, Length, ReferenceDate, ShiftDataHow)
```

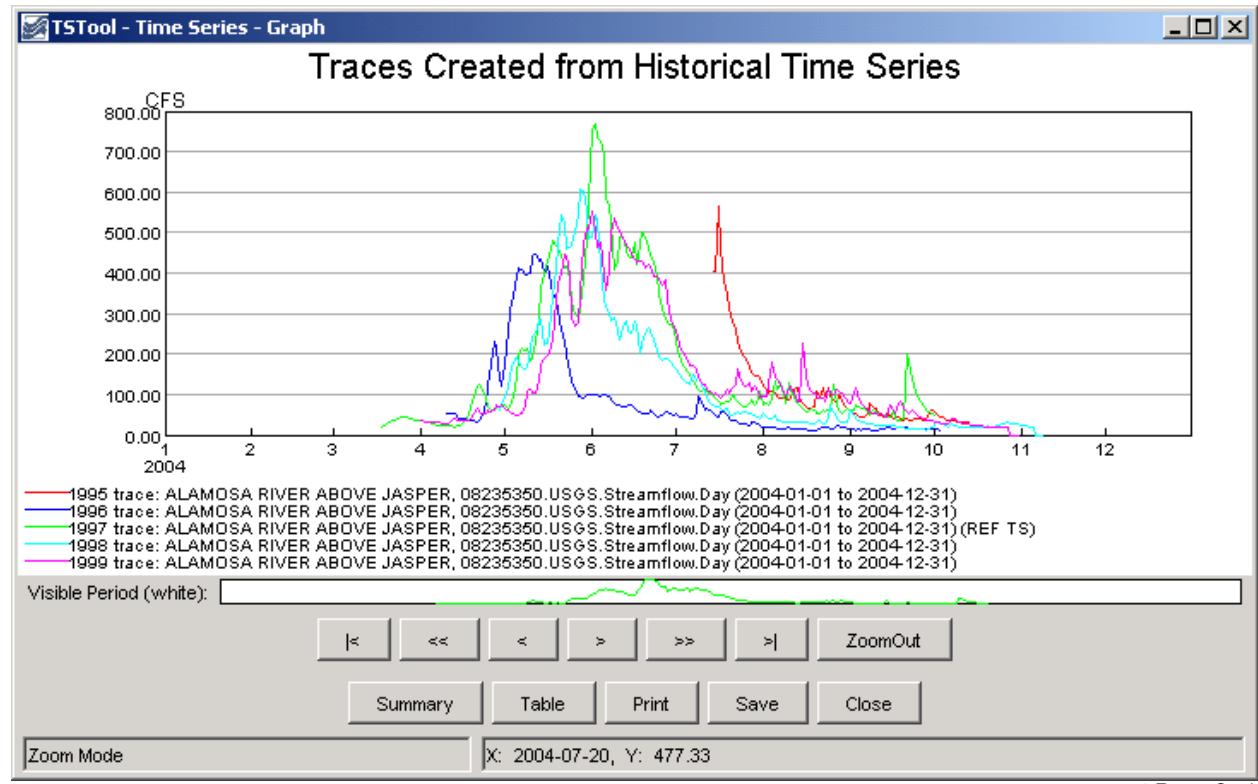
Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to split into traces. The original time series to process can be selected explicitly or can be identified as a TEMPTS.	None – must be specified.
Length	The length of each trace to be created (e.g., 1Year, #Month or, #Day). The starting (reference) date will be incremented by this interval to determine the trace end point.	1Year.
ReferenceDate	The reference date indicates the starting date for each trace and should be left blank (resulting in a default of January 1 of the current year), or set to January 1 of a year of interest (use the format 01/01/YYYY or YYYY-MM-DD). Each trace can optionally be shifted (see ShiftDataHow).	January 1.
ShiftDataHow	Indicates whether the traces should be shifted. Possible values are: <ul style="list-style-type: none"> • ShiftToReference – each trace will be shifted to the reference date, resulting in overlapping time series. • NoShift – plotting the traces will result in a total line that matches the original time series, except that each trace can be manipulated individually. 	NoShift

A sample commands file is as follows:

```
#  
# Create annual traces from a time series shifted to current year  
#  
# (1995-1998) ALAMOSA RIVER ABOVE JASPER, CO  USGS  Streamflow  Day  
08235350.USGS.Streamflow.Day~HydroBase  
createTraces(08235350.USGS.Streamflow.Day,1Year,2004-01-01,ShiftToReference)
```

Results from processing the command are as follows:



Results from the createTraces() Command

createTraces_Graph

This page is intentionally blank.

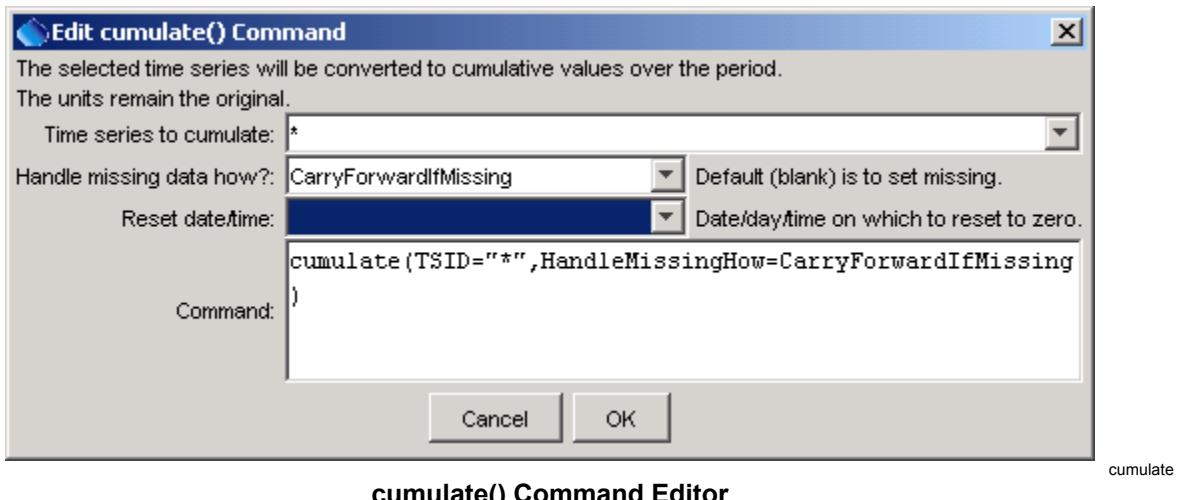
Command Reference

cumulate()

Convert time series data values to cumulative values

Version 06.10.09, 2005-09-30, Color, Acrobat Distiller

The `cumulate()` command converts a time series into cumulative values, which is useful for comparing the cumulative trends of related time series (e.g., nearby gages or precipitation gages) and can serve as a substitute for the double-mass graph, which has difficulty handling missing data. It is also useful to check the mass balance when routing time series (the cumulative values before and after routine will track closely). The following dialog is used to edit the command and illustrates the syntax of the command.



cumulate() Command Editor

The command syntax is as follows:

```
cumulate(param=value,...)
```

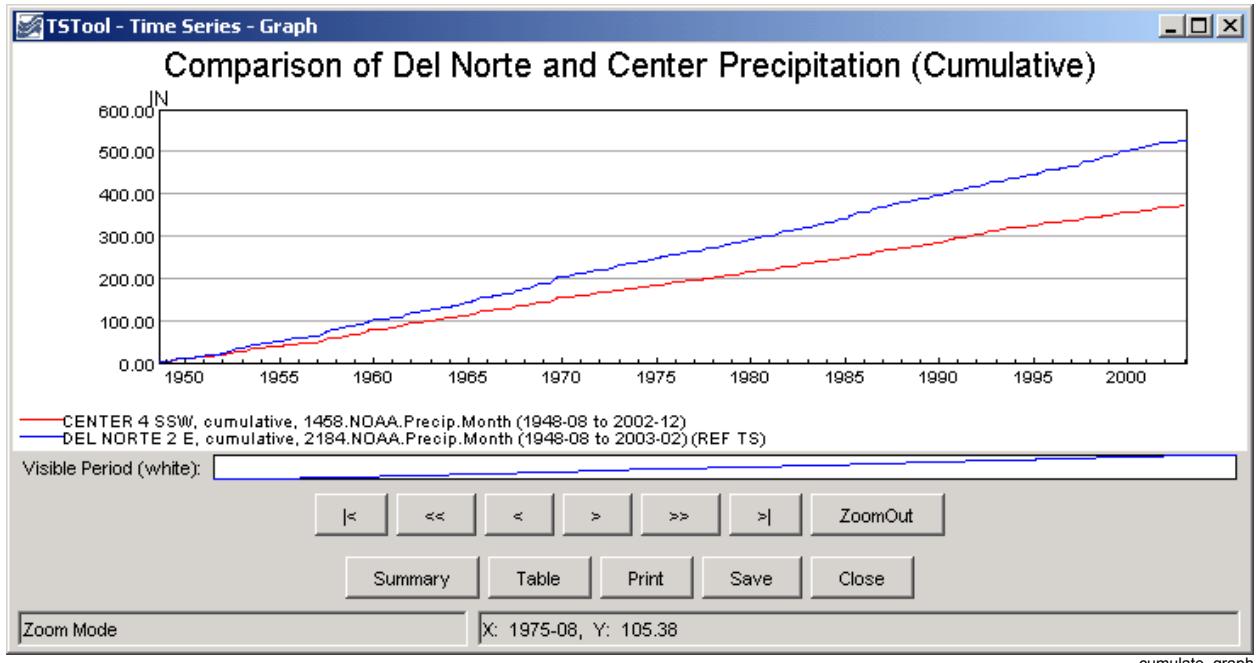
Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A single TSID/alias or all time series (*) can be specified.	None – must be specified.
Handle Missing How	Indicate how to handle missing data, one of: <ul style="list-style-type: none">• <code>CarryForwardIfMissing</code> –carry forward the last non-missing value• <code>SetMissingIfMissing</code> – set the result to missing if the original value is missing. <p>The only difference in output is that the period of missing data will either be blank or a horizontal line in graphs.</p>	<code>SetMissingIfMissing</code>
Reset	A MM-DD date, day (1-31), or month (1-12) indicating when to reset the cumulative value to zero, before beginning to cumulate again. The features of this parameter are under development.	Do not reset.

A sample commands file is as follows:

```
# 1458 - CENTER 4 SSW
1458.NOAA.Precip.Month~HydroBase
# 2184 - DEL NORTE 2 E
2184.NOAA.Precip.Month~HydroBase
cumulate(TSID="*", HandleMissingHow=CarryForwardIfMissing)
```

The following graph illustrates cumulative data for two precipitation gages in the same region, where missing data results in carrying forward the last known value.



Example Graph Showing Results of `cumulate()` Command

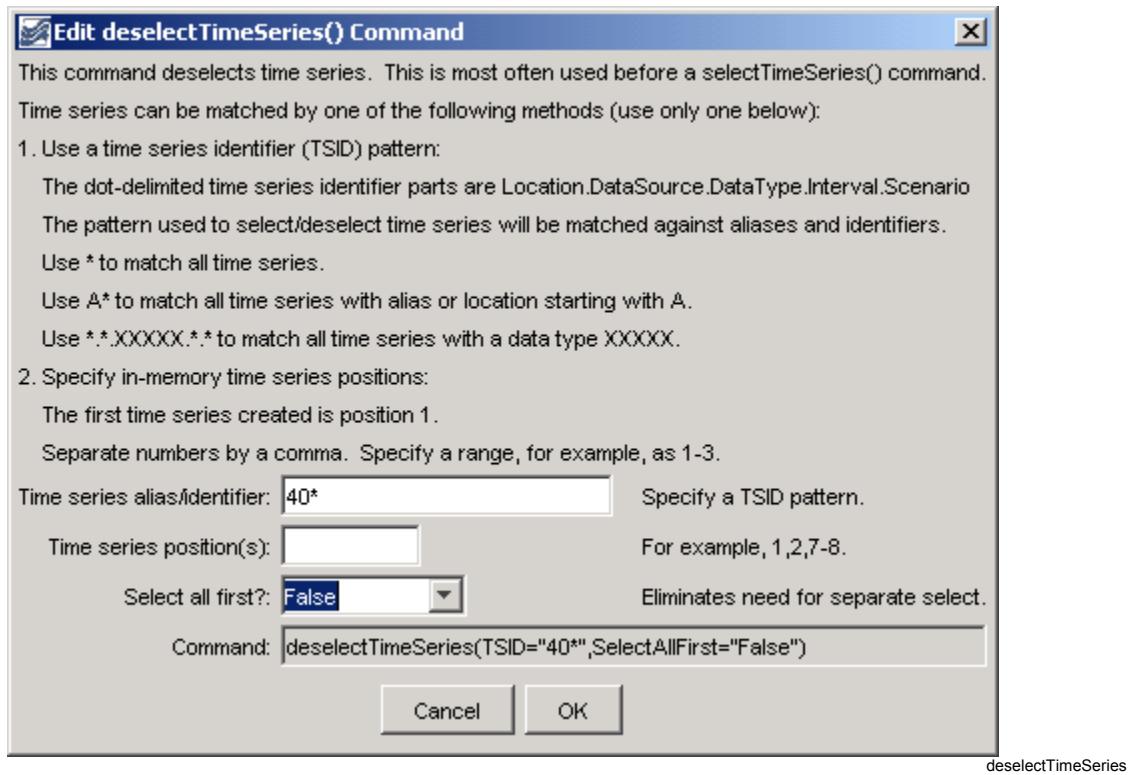
Command Reference

deselectTimeSeries()

Deselect Time Series for Output

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `deselectTimeSeries()` command deselects output time series, as if done interactively. The command minimizes the need for the `free()` command, when used in conjunction with other commands that use a time series list based on selected time series. See also the `selectTimeSeries()` command. The following dialog is used to edit the command and illustrates the command syntax.



deselectTimeSeries

deselectTimeSeries() Command Editor

The command syntax is as follows:

```
deselectTimeSeries(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias pattern, to indicate time series to deselect. See the above figure for examples.	Can be blank if Pos is specified.
Pos	A list of time series positions in output (1+), separated by commas. This parameter is useful if it is known that the first time series is being modified by subsequent commands.	Use the TSID.
SelectAllFirst	Indicates whether all time series should be selected before deselecting the specified time series: True or False.	False

A sample commands file is as follows:

```
deselectTimeSeries(TSID="40*",SelectAllFirst=True)
```

Command Reference

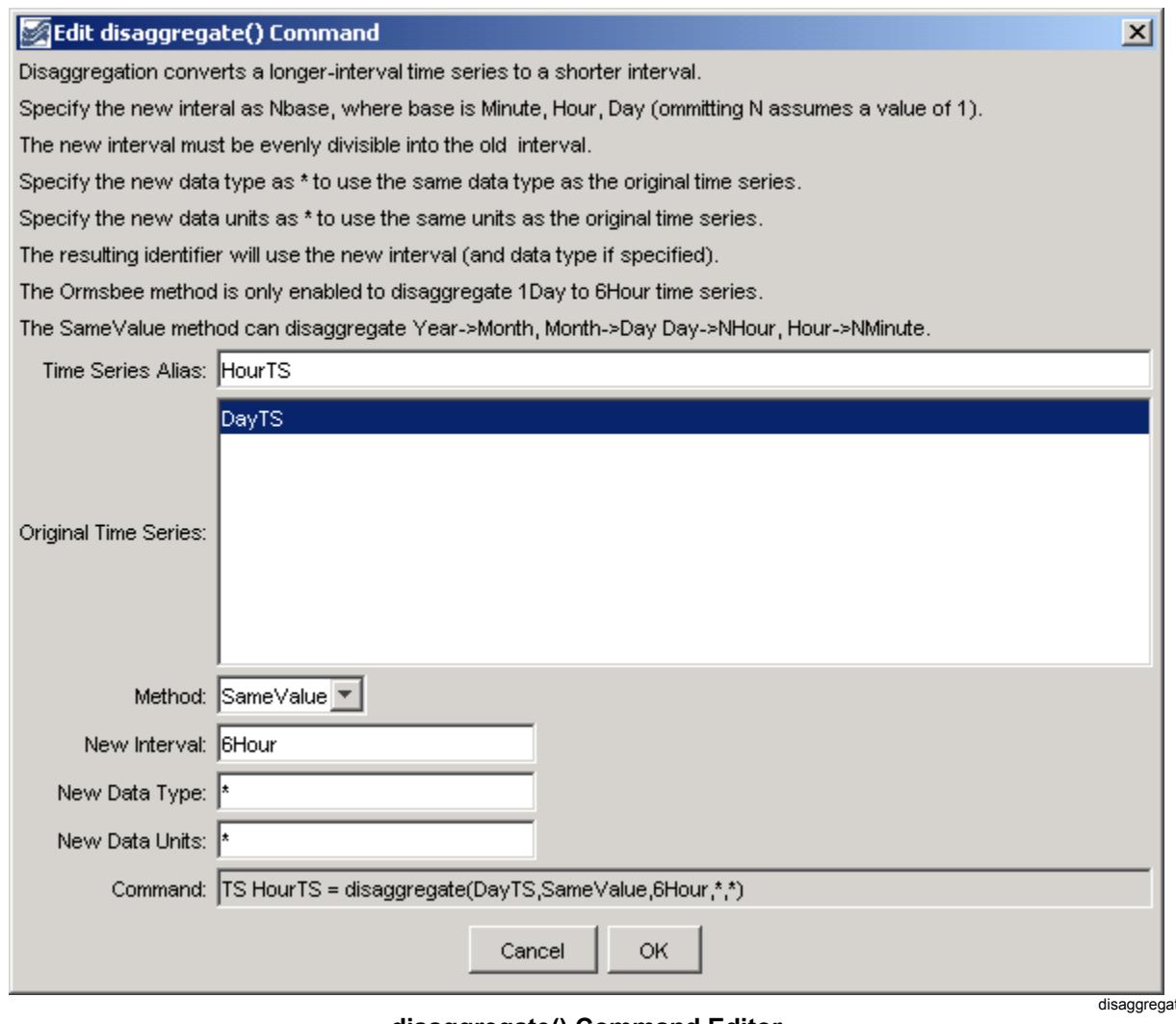
TS X = disaggregate()

Create a New Time Series with Shorter Interval

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `disaggregate()` command creates a new time series by disaggregating a time series with a longer data interval into a time series with a shorter data interval. The resulting time series will have the same header information and identifier as the original time series, with a different data interval.

Converting longer-interval data may cause a perceived shift in the time. For example, 1Day data shifted to 24Hour data will result in the daily values being set at hour zero of the following day. This shift is necessary to generically represent different time precision. Plots will also reflect the shift because hours are not considered when computing plot positions for daily data. It is important to understand how disaggregated data is treated with respect to time when using with other applications. If necessary, use the `shiftTimeByInterval()` command to manipulate the resulting output time series. The following dialog is used to edit the command and illustrates the syntax for the command.



disaggregate() Command Editor

The command syntax is as follows:

```
TS X = disaggregate(TSID,Method,NewInterval,NewDataType,NewUnits)
```

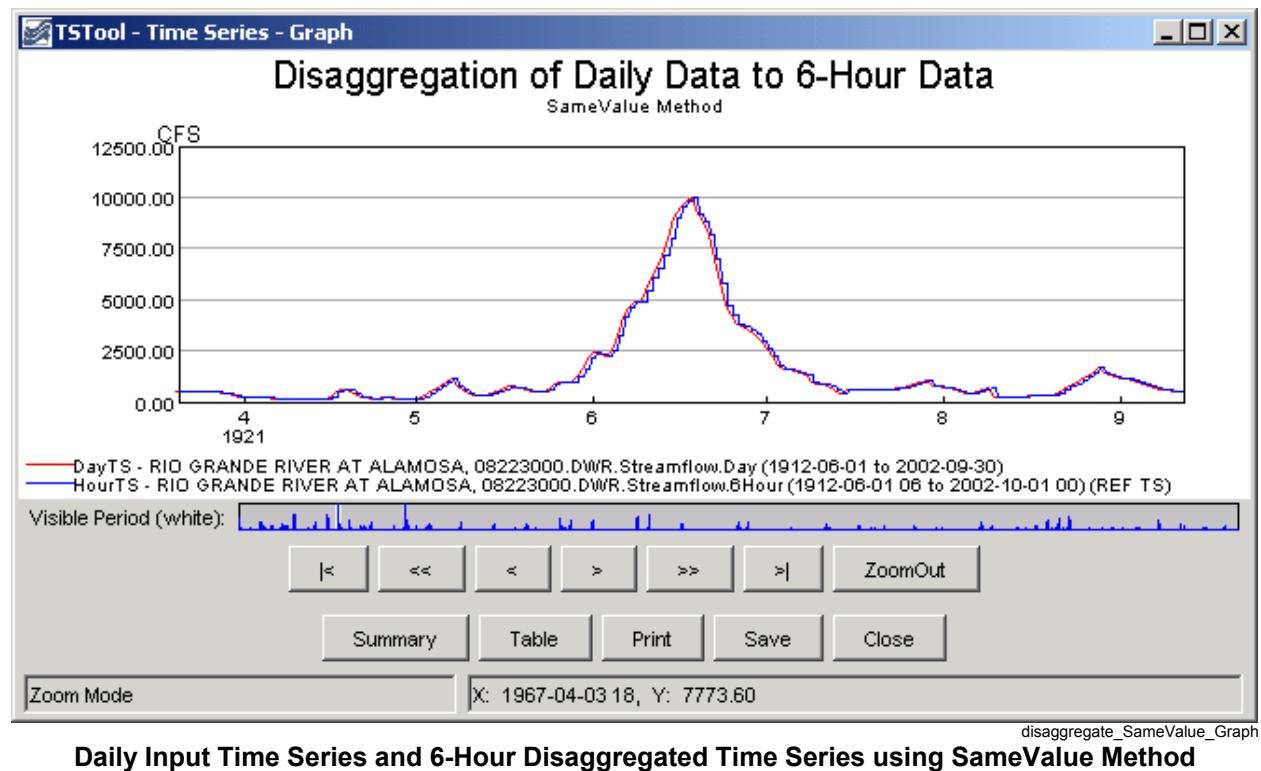
Command Parameters

Parameter	Description	Default
X	Alias for the disaggregated time series. The resulting time series will have the same header information and identifier as the original time series, with a different data interval and alias.	None – must be specified.
TSID	The time series identifier or alias for the time series to be disaggregated.	None – must be specified.
Method	<p>The method used to perform the disaggregation, one of the following:</p> <p>Orsmbee – this method was presented in “Rainfall Disaggregation Model for Continuous Hydrologic Modeling,” Ormsbee, Lindell E., Journal of Hydraulic Engineering, ASCE, April, 1989. Currently the method has only been enabled for disaggregating 1Day (not 24Hour) data to 6Hour data.</p> <p>SameValue – this simple method causes the resulting time series to have the same value as the original. For example, a monthly time series that is disaggregated to a daily time series will result in each daily value being the same as for the corresponding value in the original monthly time series. Currently the following disaggregations are supported:</p> <ul style="list-style-type: none"> • Year to Month • Month to Day • Day to NHour (including 24Hour) • Hour to NMinute (including 60Minute) 	None – must be specified.
NewInterval	The data interval for the disaggregated time series (NHour, NDay, etc.).	None – must be specified.
NewDataType	The data type for the disaggregated time series, if different from the original.	Use * to indicate the same data type as the original time series.
NewUnits	The units for the disaggregated time series, if different from the original.	Use * to indicate the same data type as the original time series.

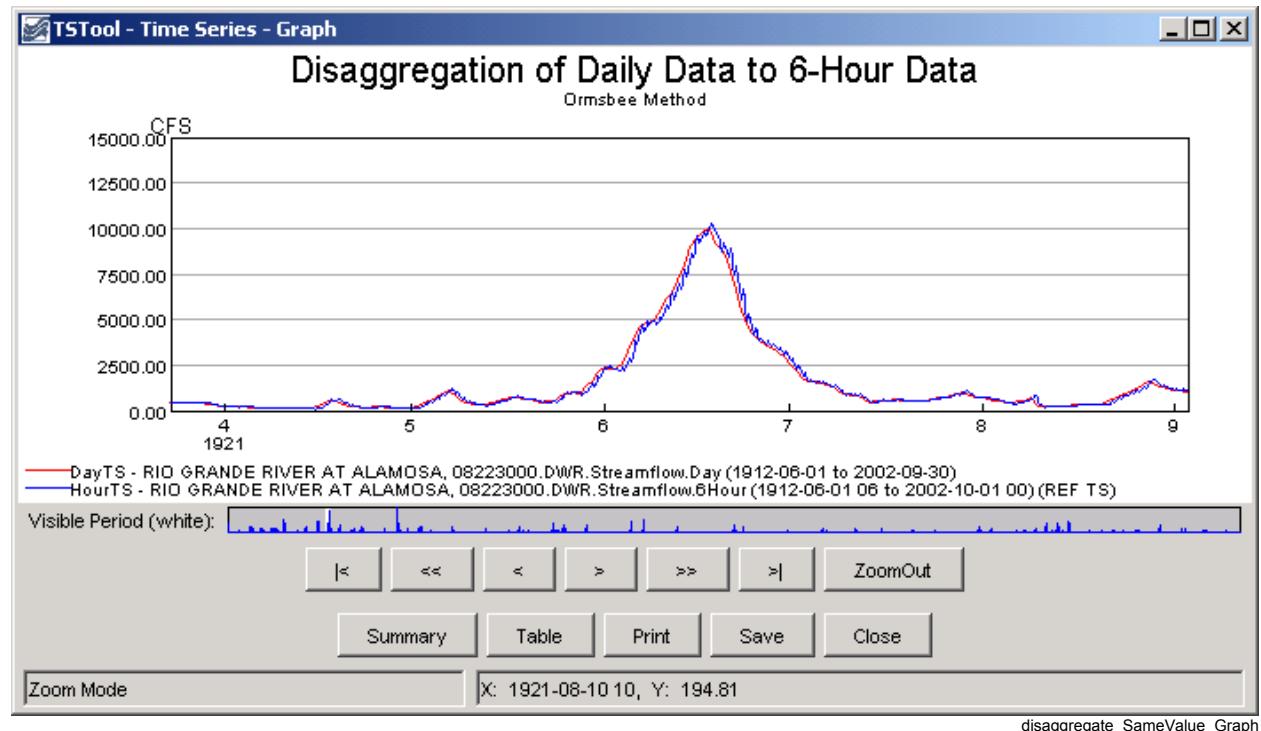
An example commands file is as follows:

```
# 08223000 - RIO GRANDE RIVER AT ALAMOSA
TS DayTS = readTimeSeries("08223000.DWR.Streamflow.Day~HydroBase")
TS HourTS = disaggregate(DayTS,Ormsbee,6Hour,*,*)
```

Examples of graphs for the original and disaggregated data are shown below, for the two disaggregation methods:



Daily Input Time Series and 6-Hour Disaggregated Time Series using SameValue Method



Daily Input Time Series and 6-Hour Disaggregated Time Series using Ormsbee Method

This page is intentionally blank.

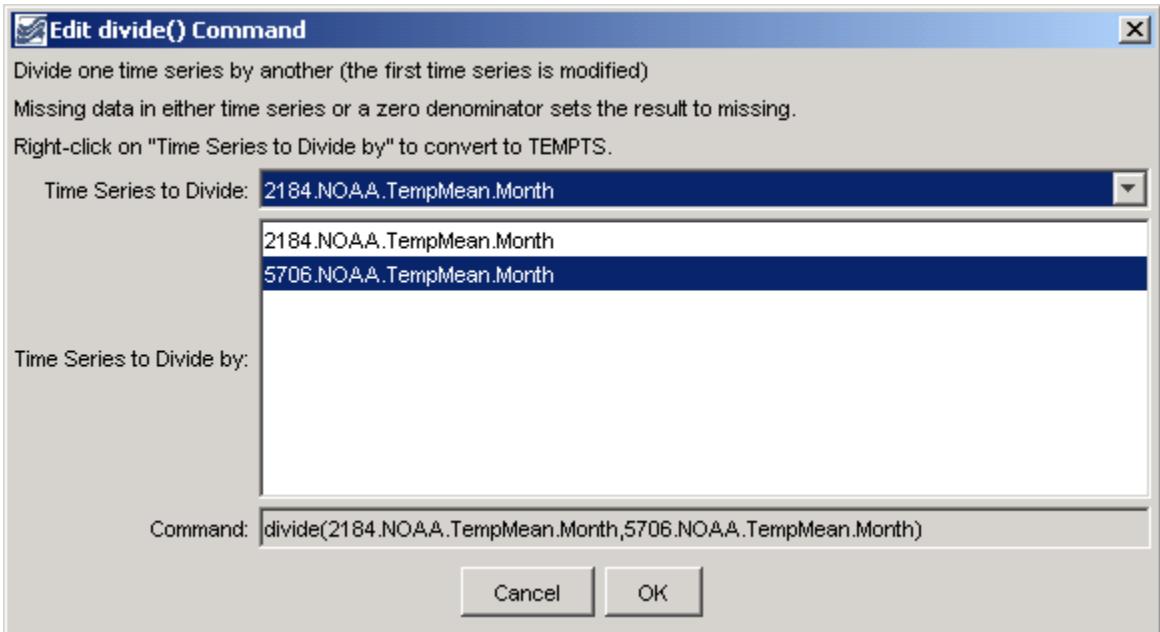
Command Reference

divide()

Divide the Data Values in one Time Series by Data Values in Another Time Series

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `divide()` command divides one time series by another. This is useful for comparing the relative size of time series values (see also `relativeDiff()`). If the divisor is zero or missing, the result is set to missing. The following dialog is used to edit the command and illustrates the syntax of the command.



divide

divide() Command Editor

The command syntax is as follows:

```
divide(TSID,DivisorTSID)
```

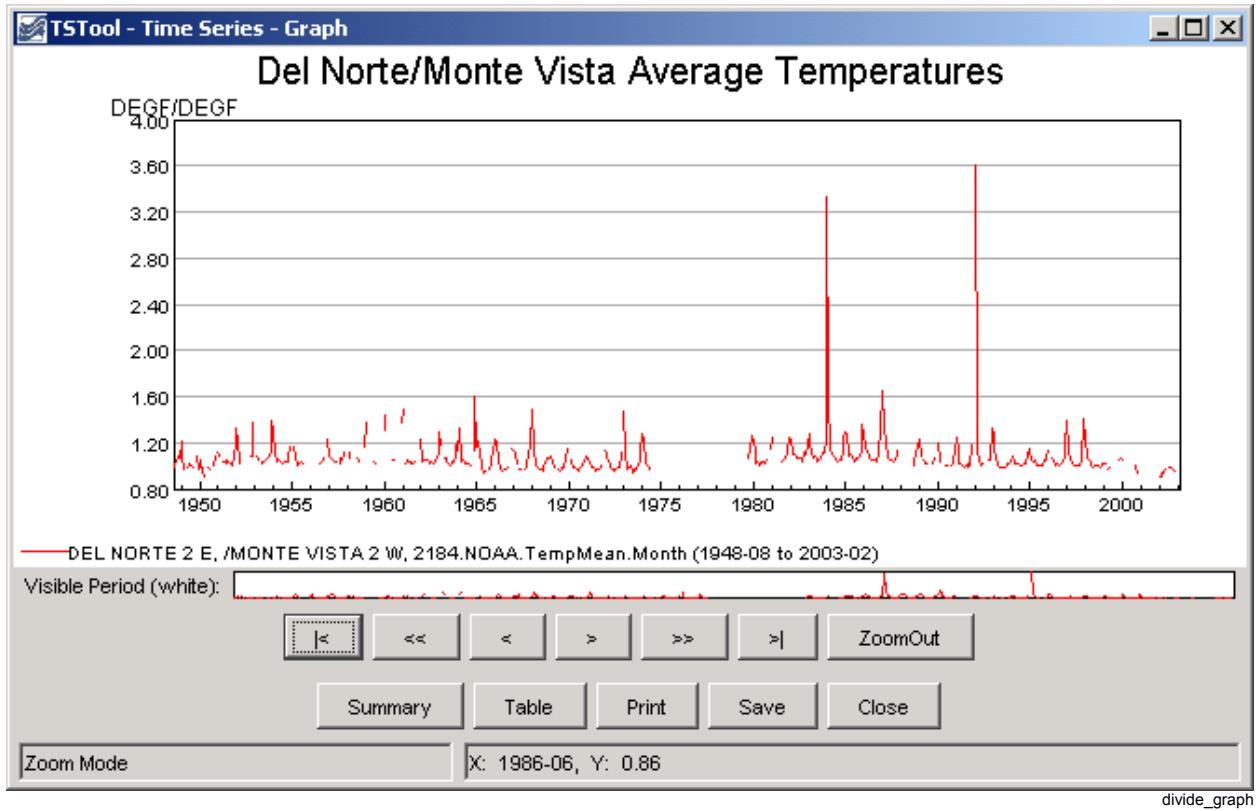
Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. Right-clicking on a time series to divide by allows toggling of the TEMPTS key word. This will cause the time series to divide to be read and then discarded, simplifying the input file. However, to insert the time series the first time, the time series must have been listed in the commands file as a time series identifier.	None – must be specified.
DivisorTSID	The time series identifier or alias for the time series that is the divisor.	None – must be specified.

A sample commands file is as follows:

```
# 2184 - DEL NORTE 2 E
2184.NOAA.TempMean.Month~HydroBase
# 5706 - MONTE VISTA 2 W
5706.NOAA.TempMean.Month~HydroBase
divide(2184.NOAA.TempMean.Month,5706.NOAA.TempMean.Month)
```

The resulting graph is as follows:



Results from divide() Command

Command Reference

exit()

Stop Processing Commands

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `exit` command can be inserted anywhere in a commands file and causes the processing of commands to stop at that line. This is useful for temporarily processing a subset of a long list of commands. Multi-line comments (`/* */`) can also be used to temporarily disable one or more commands. It is also useful to add an `exit` command at the end of the file so that it is easy to insert commands above this line when the end line is selected (rather than having to deselect all commands when editing).

This page is intentionally blank.

Command Reference

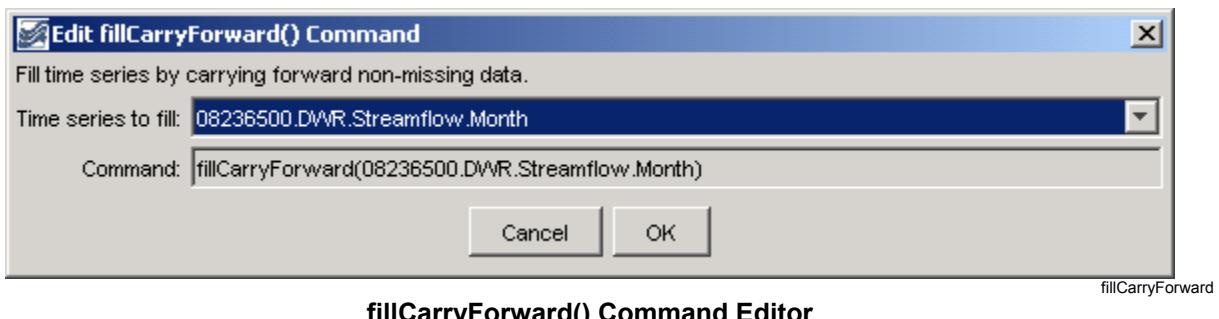
fillCarryForward()

Fill Missing Time Series Data by Carrying Data Forward

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

This command has been replaced by **fillRepeat()**.

The **fillCarryForward()** command fills a time series by carrying forward observations until another observation is found. This fill technique is useful, for example, where time series are likely to be step-wise or nearly constant, such as some reservoir and diversion time series. Currently there is no constraint to limit the number of intervals over which the filling will occur. There is also no way to limit the fill operation to a period (the entire time series is filled). The following dialog is used to edit the command and illustrates the syntax of the command.



fillCarryForward() Command Editor

The command syntax is as follows:

```
fillCarryForward(TSID)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled. Specify as * to fill all time series.	None – must be specified.

A sample commands file is as follows:

```
# 08236500 - ALAMOSA RIVER BELOW TERRACE RESERVOIR
08236500.DWR.Streamflow.Month~HydroBase
fillCarryForward(08236500.DWR.Streamflow.Month)
```

This page is intentionally blank.

Command Reference

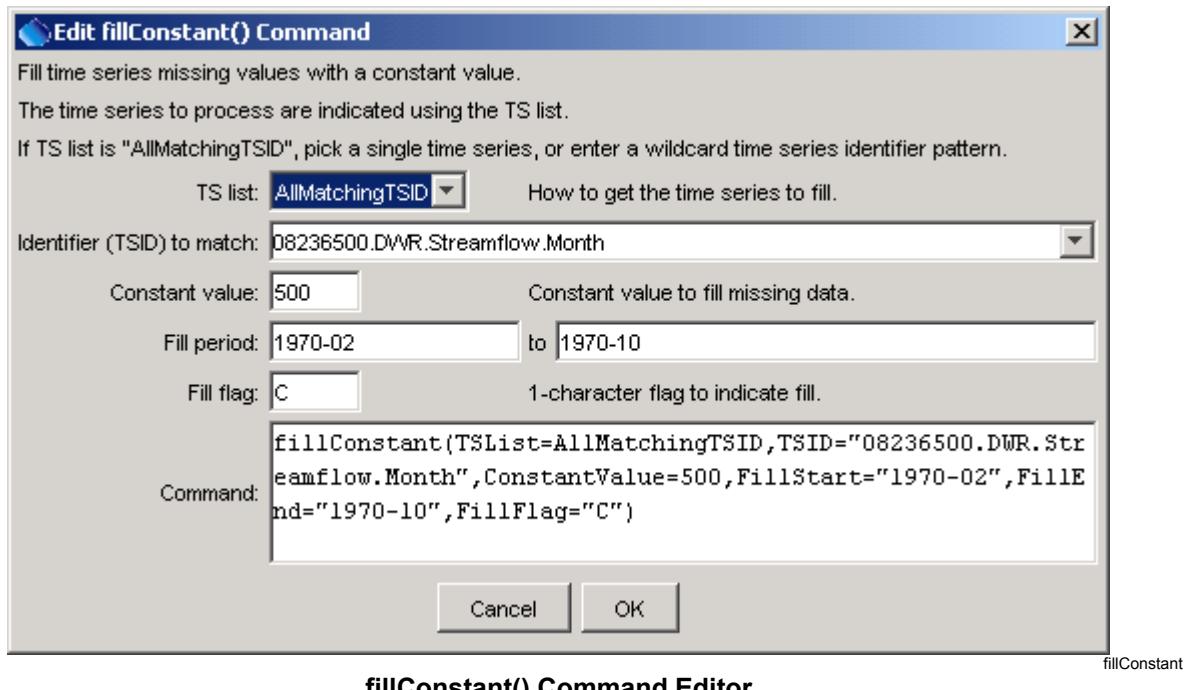
fillConstant()

Fill missing time series data using a constant value

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The `fillConstant()` command fills the missing data in a time series with the specified value. This fill technique is useful for filling missing data with zeros, perhaps as the last step in a sequence of filling commands.

The following dialog is used to edit the command and illustrates the command syntax.



fillConstant() Command Editor

The command syntax is as follows:

```
fillConstant(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSList	Indicate how to determine the list of time series to process, one of: <ul style="list-style-type: none"> ▪ AllMatchingTSID – process time series that have identifiers matching the TSID parameter. ▪ AllTS – process all the time series. ▪ SelectedTS – process the time series that are selected (see <code>selectTimeSeries()</code>). 	None – must be specified.
TSID	Used if <code>TSList=AllMatchingTSID</code> to indicate the time series identifier or alias for the time series to be filled. Specify * to match all time series or use a wildcard for one or more identifier parts.	Required if <code>TSList=AllMatchingTSID</code> .
ConstantValue	Constant value to use when filling missing data.	None – must be specified.
FillStart	Date/time indicating the start of filling, using a precision appropriate for the time series, or <code>OutputStart</code> .	Fill the entire time series.
FillEnd	Date/time indicating the end of filling, using a precision appropriate for the time series, or <code>OutputEnd</code> .	Fill the entire time series.
FillFlag	If specified as a single character, data flags will be enabled for the time series and each filled value will be tagged with the specified character. The flag can then be used later to label graphs, etc. The flag will be appended to existing flags if necessary.	No flag is assigned.

A sample commands file is as follows:

```
# 08236500 - ALAMOSA RIVER BELOW TERRACE RESERVOIR
08236500.DWR.Streamflow.Month~HydroBase
fillConstant(TSList=AllMatchingTSID,TSID="08236500.DWR.Streamflow.Month",
ConstantValue=500,FillStart="1970-02",FillEnd="1970-10",FillFlag="C")
```

Command Reference

fillDayTSFrom2MonthTSAnd1DayTS()

Fill Daily Time Series from Monthly Volumes and Daily Pattern

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `fillDayTSFrom2MonthTSAnd1DayTS()` command fills a daily time series using the following relationship:

$$D1_i = D2_i * (M1_i / M2_i)$$

where:

i = interval

D1 is the daily data at point1

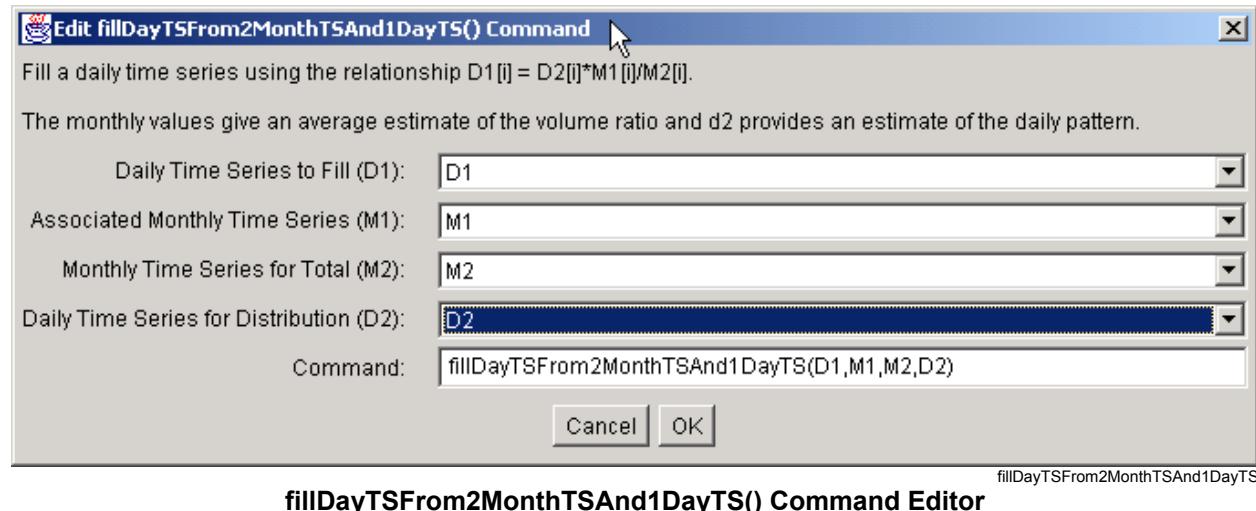
M1 is the monthly data at point1

D2 is the daily data at point2

M2 is the monthly data at point2.

This fill method assumes the monthly time series are filled and reasonably correlated and that the daily pattern D2 can be applied at D1. For example, use this command to fill daily streamflow where filled monthly data are available at nearby locations and filled daily data is available at the independent (D2) station. There is currently no way to limit the fill operation to a period (the entire time series is filled).

The following dialog is used to edit the command and illustrates the syntax of the command.



fillDayTSFrom2MonthTSAnd1DayTS

fillDayTSFrom2MonthTSAnd1DayTS() Command Editor

The command syntax is as follows:

```
fillDayTSFrom2MonthTSAnd1DayTS(TSID_D1,TSID_M1,TSID_M2,TSID_D2)
```

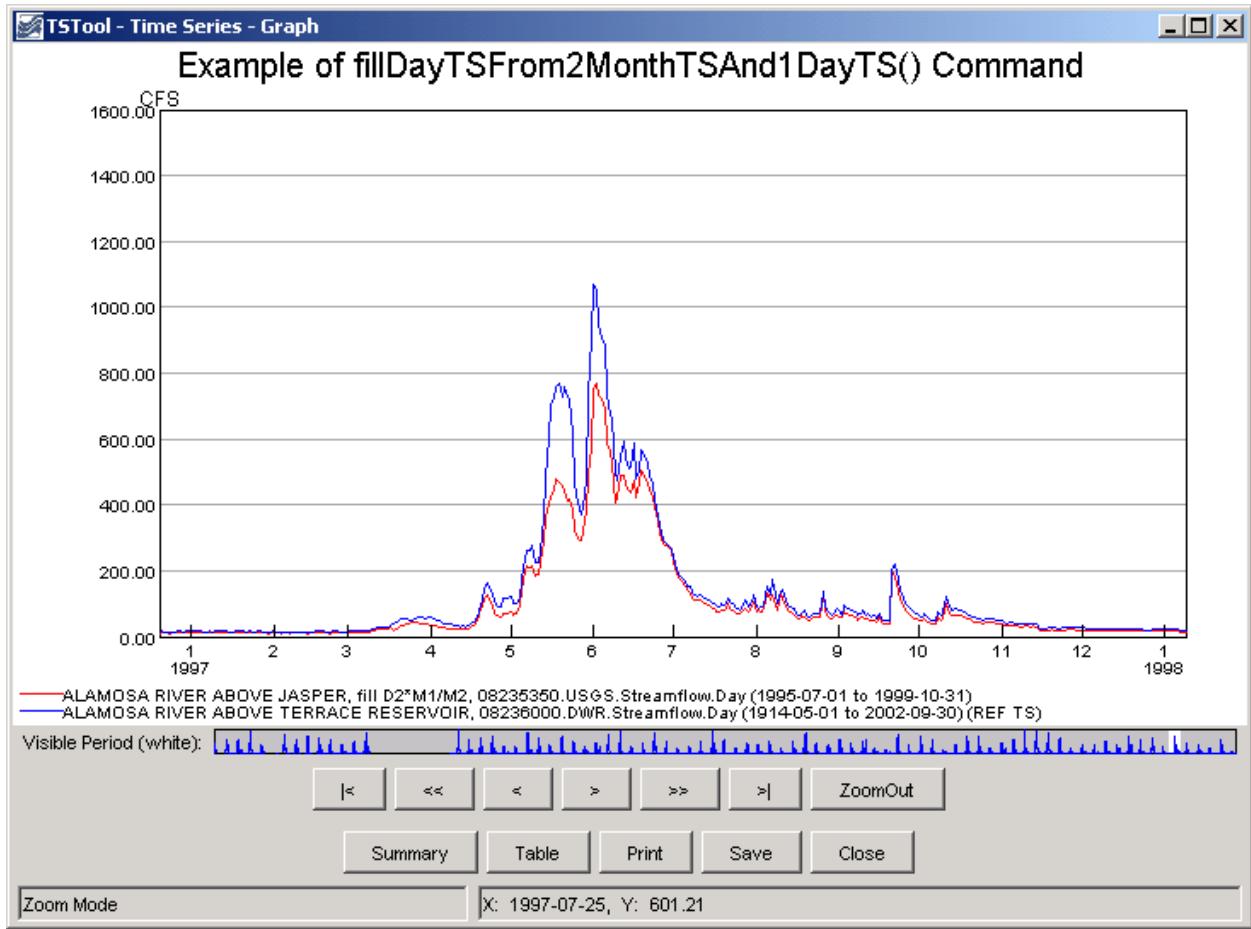
Command Parameters

Parameter	Description	Default
TSID_D1	The time series identifier or alias for the daily time series to be filled.	None – must be specified.
TSID_M1	The time series identifier or alias for the monthly time series, corresponding to TSID_D1, to supply the monthly values to be distributed to daily.	None – must be specified.
TSID_M2	The time series identifier or alias for the independent monthly time series.	None – must be specified.
TSID_D2	The time series identifier or alias for the independent daily time series, corresponding to TSID_M2.	None – must be specified.

An example commands file is shown below with the resulting graph of daily time series.

```
#  
# Test fillDayTSFrom2MonthTSAnd1DayTS: D1 = D2*M1/M2  
#  
# SAM, RTi, 2001-02-25  
#  
# The following is D1:  
# (1995-1998) ALAMOSA RIVER ABOVE JASPER, CO USGS Streamflow  
Daily  
08235350.USGS.Streamflow.Day~HydroBase  
# The following is M1:  
# (1995-1998) ALAMOSA RIVER ABOVE JASPER, CO USGS Streamflow  
Monthly  
08235350.USGS.Streamflow.Month~HydroBase  
# The following is D2:  
# (1914-1998) ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO. DWR  
Streamflow Daily  
08236000.DWR.Streamflow.Day~HydroBase  
# The following is M2:  
# (1914-1998) ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO. DWR  
Streamflow Monthly  
08236000.DWR.Streamflow.Month~HydroBase  
fillRegression(08235350.USGS.Streamflow.Month,08236000.DWR.Stream  
flow.Month,OneEquation,Linear,*,*,*,*)  
fillDayTSFrom2MonthTSAnd1DayTS(08235350.USGS.Streamflow.Day,  
08235350.USGS.Streamflow.Month,08236000.DWR.Streamflow.Month,  
08236000.DWR.Streamflow.Day)
```

The following graph shows the two daily time series used in the command (zoomed in). Note that the shape of the filled time series is similar to the other time series.



Example of Filled Data

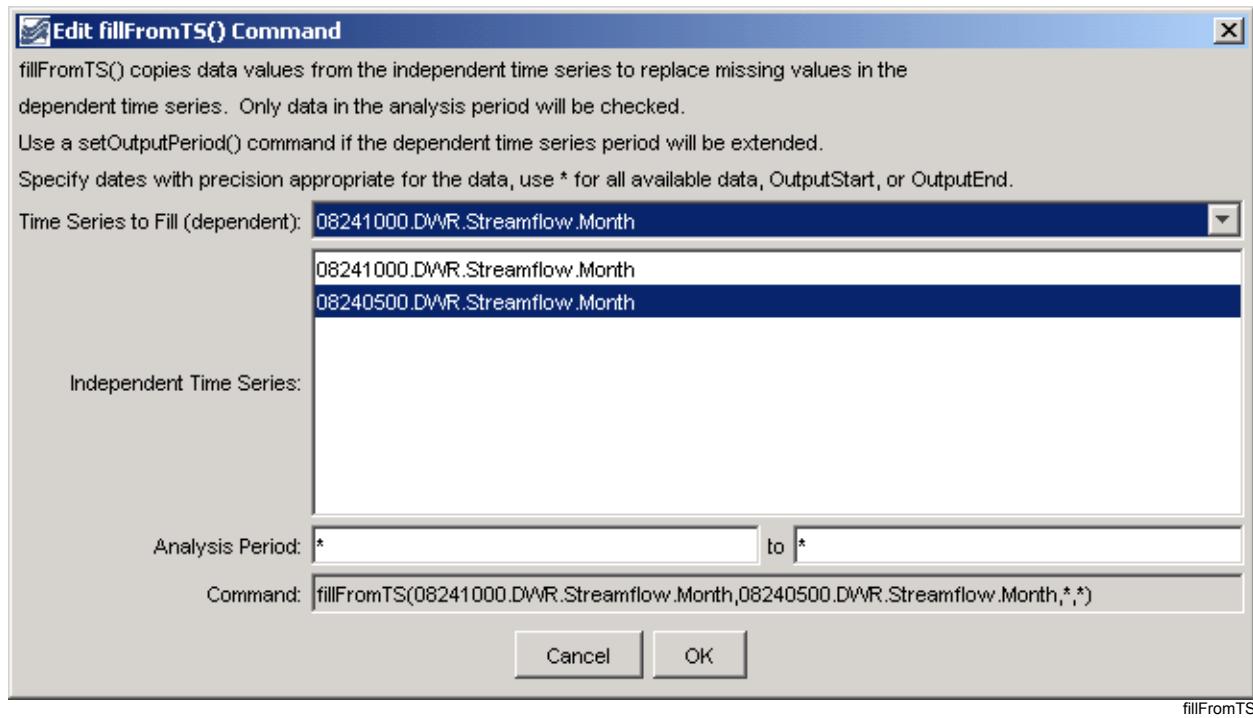
Command Reference

fillFromTS()

Fill Missing Time Series Data Using Data From Another Time Series

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `fillFromTS()` command fills missing data in a time series by transferring non-missing values from another time series. This is useful when two time series typically have very similar values. The filled time series is not automatically extended. An analysis period can be specified to limit the period that is checked for missing data. See also the `setFromTS()` command, which will transfer all values. The following dialog is used to edit the command and illustrates the command syntax.



fillFromTS

fillFromTS() Command Editor

The command syntax is as follows:

```
fillFromTS(TSID,IndependentTSID,FillStart,FillEnd)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled.	None – must be specified.
IndependentTSID	The time series identifier or alias for the independent time series, to supply data.	None – must be specified.
FillStart	The date/time to start filling, if other than the full time series period.	Full period, if * is specified.
FillEnd	The date/time to end filling, if other than the full time series period.	Full period, if * is specified.

A sample commands file is as follows:

```
#  
# 08241000 - TRINCHERA CREEK ABOVE MOUNTAIN HOME RESERVOIR  
08241000.DWR.Streamflow.Month~HydroBase  
# 08240500 - TRINCHERA CREEK ABOVE TURNER'S RANCH  
08240500.DWR.Streamflow.Month~HydroBase  
fillFromTS(08241000.DWR.Streamflow.Month,08240500.DWR.Streamflow.Month,* ,*)
```

Command Reference

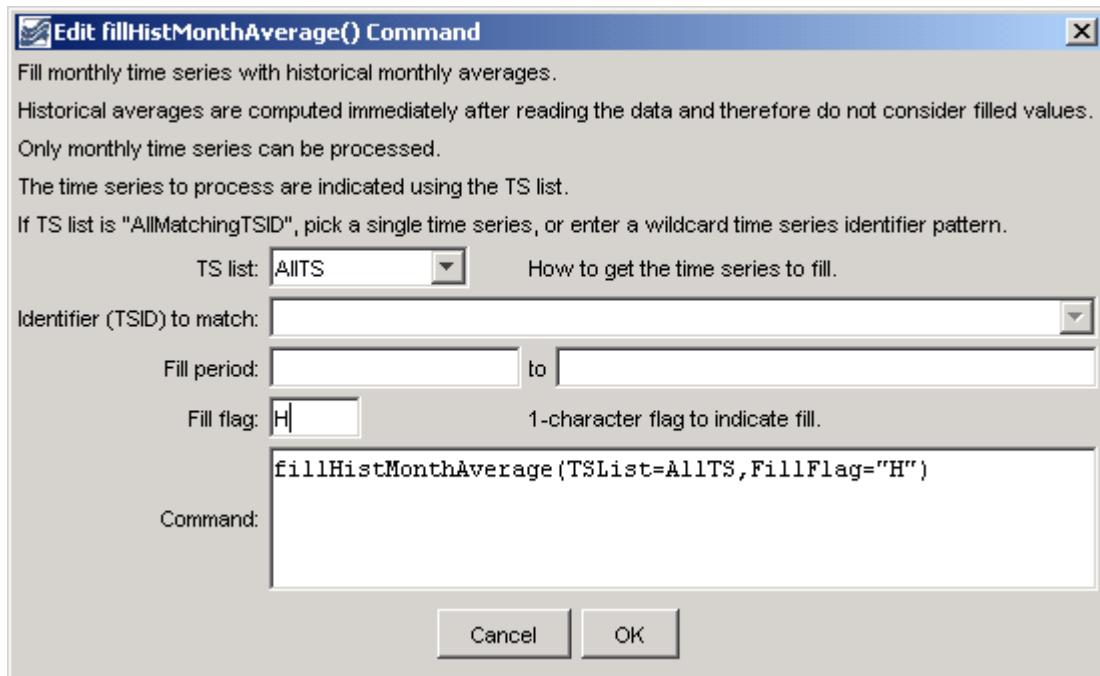
fillHistMonthAverage()

Fill missing time series data using historical monthly average data

Version 06.10.01, 2005-06-08, Color, Acrobat Distiller

The `fillHistMonthAverage()` command fills missing data in monthly time series with the average monthly values. The average values are computed using the available data period (or specified averaging period – see the `setAveragePeriod()` command) immediately after the time series is read and are then applied when this command is encountered.

The following dialog is used to edit the command and illustrates the syntax of the command.



fillHistMonthAverage() Command Editor

The command syntax is as follows:

```
fillHistMonthAverage(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSList	Indicate how to determine the list of time series to process, one of: <ul style="list-style-type: none"> ▪ AllMatchingTSID – process time series that have identifiers matching the TSID parameter. ▪ AllTS – process all the time series. ▪ SelectedTS – process the time series that are selected (see <code>selectTimeSeries()</code>). 	None – must be specified.
TSID	Used if <code>TSList=AllMatchingTSID</code> to indicate the time series identifier or alias for the time series to be filled. Specify * to match all time series or use a wildcard for one or more identifier parts.	Required if <code>TSList=AllMatchingTSID</code> .
FillStart	Date/time indicating the start of filling, using a precision appropriate for the time series, or <code>OutputStart</code> .	Fill the entire time series.
FillEnd	Date/time indicating the end of filling, using a precision appropriate for the time series, or <code>OutputEnd</code> .	Fill the entire time series.
FillFlag	If specified as a single character, data flags will be enabled for the time series and each filled value will be tagged with the specified character. The flag can then be used later to label graphs, etc. The flag will be appended to existing flags if necessary.	No flag is assigned.

Sample commands files are as follows:

```
# 0125 - ALAMOSA
0125.NOAA.Precip.Month~HydroBase
fillHistMonthAverage(TSList=AllMatchingTSID,TSID="0125.NOAA.Precip.Month",
FillFlag="H")
```

```
# read time series...
fillHistMonthAverage(TSList=AllMatchingTSID,TSID="019*",FillFlag="H")
```

Time series data limits for the averages are printed to the log file, similar to the following examples (note that the period for averaging is always shown and may be different than the output period).

Status: Historic averages for time series follow...										
Time series: 0125.NOAA.Precip.Month (IN)										
Monthly limits for period 1948-08 to 1949-12 are:										
Month	Min	MinDate	Max	MaxDate	Sum	# Miss.	% Miss.	# Not Miss.	% Not Miss.	Mean
Jan	0.2	1949-01	0.2	1949-01	0.2	0	0.00	1	100.00	0.2
Feb	0.1	1949-02	0.1	1949-02	0.1	0	0.00	1	100.00	0.1
Mar	0.1	1949-03	0.1	1949-03	0.1	0	0.00	1	100.00	0.1
Apr	-999.0		-999.0		-999.0	1	100.00	0	0.00	-999.0
May	-999.0		-999.0		-999.0	1	100.00	0	0.00	-999.0
Jun	0.7	1949-06	0.7	1949-06	0.7	0	0.00	1	100.00	0.7
Jul	1.5	1949-07	1.5	1949-07	1.5	0	0.00	1	100.00	1.5
Aug	0.7	1949-08	0.8	1948-08	1.5	0	0.00	2	100.00	0.8
Sep	0.1	1948-09	1.1	1949-09	1.2	0	0.00	2	100.00	0.6
Oct	0.1	1949-10	0.5	1948-10	0.7	0	0.00	2	100.00	0.3
Nov	0.0	1949-11	0.8	1948-11	0.8	0	0.00	2	100.00	0.4
Dec	0.0	1949-12	0.2	1948-12	0.2	0	0.00	2	100.00	0.1
Period	0.0	1949-11	1.5	1949-07	6.9	2	11.76	15	88.24	0.5

This page is intentionally blank.

Command Reference

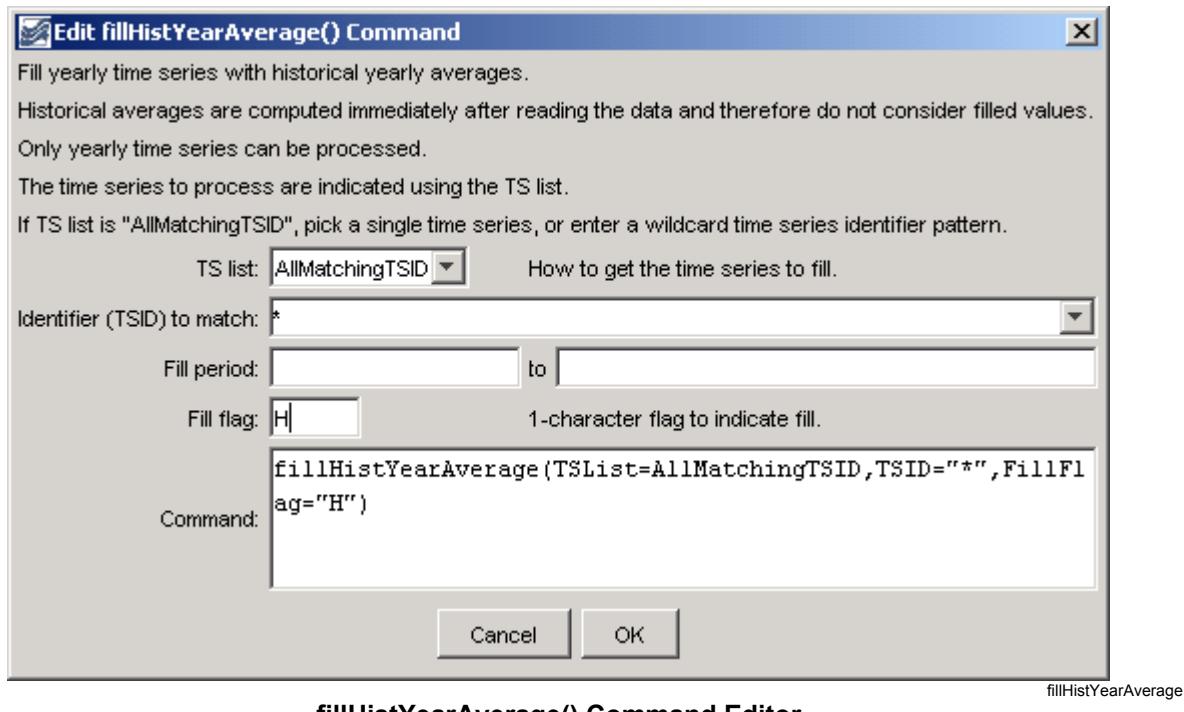
fillHistYearAverage()

Fill missing time series data using historical yearly average data

Version 06.10.01, 2005-06-08, Color, Acrobat Distiller

The `fillHistYearAverage()` command fills missing data in yearly time series with the average annual value. The average values are computed using the available data period (or specified averaging period – see the `setAveragePeriod()` command) immediately after the time series is read and are then applied when this command is encountered.

The following dialog is used to edit the command and illustrates the syntax of the command.



fillHistYearAverage

fillHistYearAverage() Command Editor

The command syntax is as follows:

```
fillHistYearAverage(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSList	Indicate how to determine the list of time series to process, one of: <ul style="list-style-type: none"> ▪ AllMatchingTSID – process time series that have identifiers matching the TSID parameter. ▪ AllTS – process all the time series. ▪ SelectedTS – process the time series that are selected (see <code>selectTimeSeries()</code>). 	None – must be specified.
TSID	Used if <code>TSList=AllMatchingTSID</code> to indicate the time series identifier or alias for the time series to be filled. Specify * to match all time series or use a wildcard for one or more identifier parts.	Required if <code>TSList=AllMatchingTSID</code> .
FillStart	Date/time indicating the start of filling, using a precision appropriate for the time series, or <code>OutputStart</code> .	Fill the entire time series.
FillEnd	Date/time indicating the end of filling, using a precision appropriate for the time series, or <code>OutputEnd</code> .	Fill the entire time series.
FillFlag	If specified as a single character, data flags will be enabled for the time series and each filled value will be tagged with the specified character. The flag can then be used later to label graphs, etc. The flag will be appended to existing flags if necessary.	No flag is assigned.

A sample commands file is as follows:

```
LARIMER.NASS.CropArea-Vegetables, Harvested.Year~HydroBase  
fillHistYearAverage(TSList=AllMatchingTSID,  
                    TSID="LARIMER.NASS.CropArea-Vegetables, Harvested.Year")
```

Time series data limits for the averages are printed to the log file, similar to the following examples (note that the period for averaging is always shown and may be different than the output period).

```
Min:          95.0000 ACRE on 1954  
Max:         2684.0000 ACRE on 1959  
Sum:        11090.0000 ACRE  
Mean:       1008.1818 ACRE  
Number Missing: 42 (79.25%)  
Number Not Missing: 11 (20.75%)  
Total period: 1945 to 1997  
Non-missing data period: 1945 to 1997
```

This page is intentionally blank.

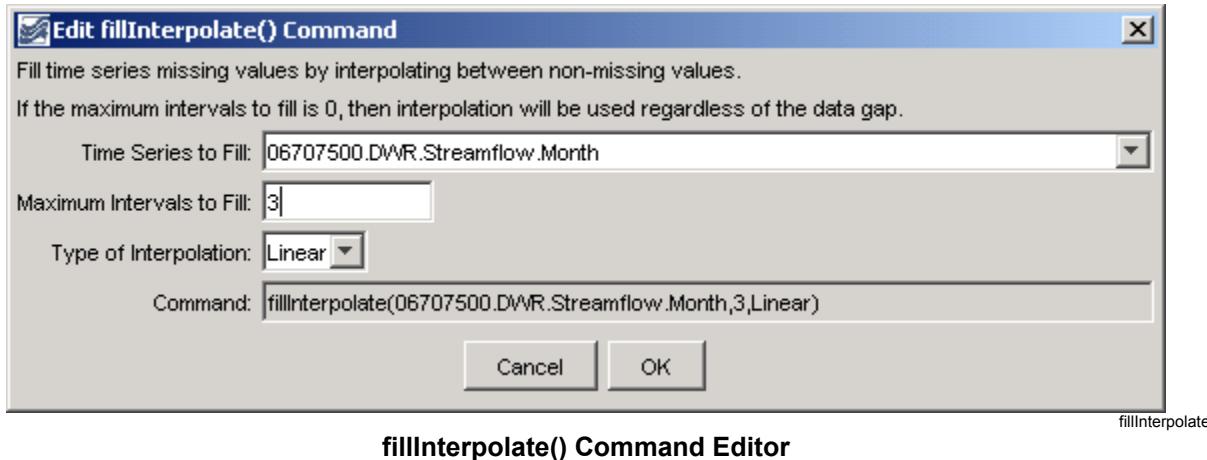
Command Reference

fillInterpolate()

Fill Missing Time Series Data by Interpolating Between Known Values

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `fillInterpolate()` command fills missing data in a time series by interpolating between known values within the same time series. There is currently no way to limit the fill operation to a period (the entire time series is filled). The following dialog is used to edit the command and illustrates the syntax of the command.



fillInterpolate() Command Editor

The command syntax is as follows:

```
fillInterpolate(TSID,MaxIntervals,Transformation)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled. Specify * to fill all time series.	None – must be specified.
MaxIntervals	The maximum number of consecutive intervals to fill (0 indicates no limits on the number of consecutive intervals that can be filled).	None – must be specified.
Transformation	Indicate the data transformation to occur for interpolation. Currently, Linear is the only option.	None – must be specified.

A sample commands file is as follows:

```
# 06707500 - SOUTH PLATTE RIVER AT SOUTH PLATTE
06707500.DWR.Streamflow.Month~HydroBase
fillInterpolate(06707500.DWR.Streamflow.Month,3,Linear)
```

Command Reference

DRAFT - THIS COMMAND IS BEING DEVELOPED

fillMixedStation()

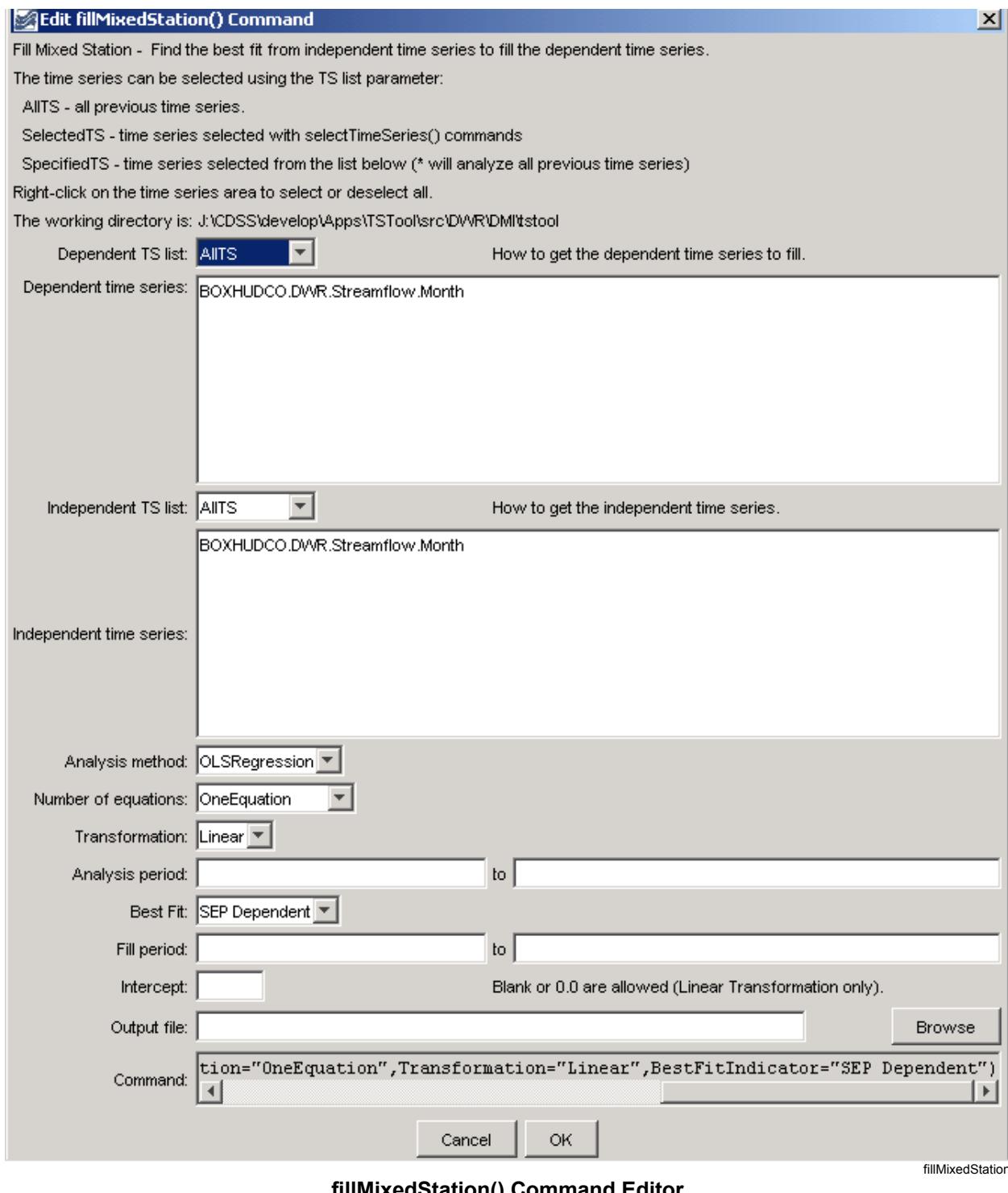
Fill missing time series data using the best fit from OLS regression or MOVE2, multiple independent time series and data transforms, optionally by month

Version 06.10.00, 2005-04-06, Color, Acrobat Distiller

The `fillMixedStation()` command fills missing data in a time series using ordinary least squares (OLS) regression (see the `fillRegression()` command for details) or MOVE2 (see the `fillMOVE2()` command for details). This command first performs an analysis to determine the combination of one or more independent time series, data transformation(s), and analysis method(s), using monthly or a single equation. The resulting best fit, based on standard error of prediction, is then used to fill the time series. Because extensive analysis may be necessary to evaluate all the combinations of parameters, this command will be slower than other commands that specifically indicate how to perform the filling. Therefore, this command is best suited to automatically filling a large list of time series.

The results are printed to a report file.

The following dialog is used to edit the command and illustrates the syntax of the command:

**fillMixedStation() Command Editor**

The command syntax is as follows:

```
fillMixedStation(param=value,...)
```

Command Parameters

Parameter	Description	Default
DependentTSList	<p>Indicates how the list of dependent time series is specified, one of:</p> <ul style="list-style-type: none"> ▪ AllTS – all time series prior to the command. ▪ SelectedTS – the time series are those selected with the <code>selectTimeSeries()</code> command. ▪ MatchingTSID – the specified list of time series given by the <code>DependentTSID</code> parameter. 	None – must be specified.
DependentTSID	<p>The time series identifier or alias for the independent time series to be filled. Specify as a single TSID or a comma-separated list of TSIDs, surrounded by double quotes.</p>	Must be specified if <code>DependentTSList=SpecifiedTS</code> , ignored otherwise.
IndependentTSList	<p>Indicates how the list of independent time series is specified, one of:</p> <ul style="list-style-type: none"> ▪ AllTS – all time series prior to the command. ▪ SelectedTS – the time series are those selected with the <code>selectTimeSeries()</code> command. ▪ MatchingTSID – the specified list of time series given by the <code>DependentTSID</code> parameter. <p>If an independent time series matches the independent time series, the analysis combination will be skipped.</p>	None – must be specified.
IndependentTSID	<p>The time series identifier or alias for the independent time series to be compared. Specify as a single TSID or a comma-separated list of TSIDs, surrounded by double quotes.</p>	Must be specified if <code>IndependentTSList=SpecifiedTS</code> , ignored otherwise.
AnalysisMethod	<p>Specify the method(s) to analyze the data, in order to determine the best fit, including <code>OLSRegression</code> and/or <code>MOVE2</code>. If multiple methods are specified, separate with commas. Include in double quotes.</p>	

Parameter	Description	Default
NumberOfEquations	The number of equations to use for the analysis: OneEquation or MonthlyEquations.	None – must be specified.
Transformation	Indicates how to transform the data before analyzing. Specify as Linear (no transformation) or Log (for \log_{10}). If the Log option is used, zero and negative values are set to .001 (-999 values are typically treated as missing data and are ignored). If multiple values are selected, separate with a comma. Include in double quotes.	No transformation.
AnalysisStart	The date/time to start the analysis, to focus on a period appropriate for analysis. For example, specify the unregulated period for streamflow.	If blank, analyze the full overlapping period.
AnalysisEnd	The date/time to end the analysis.	If blank, analyze the full overlapping period.
MinimumDataCount	The minimum number of overlapping data points that are required for a valid analysis (N1 in fillRegression() and fillMOVE2() documentation). If the minimum count is not met, then the independent time series is ignored for the specific combination of parameters. For example, if monthly equations are used, the independent time series may be ignored for the specific month; however, it may still be analyzed for other months.	None – must be specified.
MinimumR	The minimum correlation coefficient required for a best fit. If the minimum is not met, then the results are not considered in the best fit ranking.	0.5
BestFitIndicator	Specifies the indicator to use when determining the best fit, one of: <ul style="list-style-type: none"> ▪ R (correlation coefficient). ▪ SEP (Standard Error of Prediction), defined as the square root of the sum of differences between the known dependent value, and the value determined from the equation of best fit at the same point. ▪ SEPTotal, when used with one equation, it is the same as SEP. When used with monthly equations, it is the average error considering all months. 	SEP (Standard Error of Prediction).
FillStart	The date/time to start filling, if other than the full time series period.	If blank, fill the full period.

Parameter	Description	Default
FillEnd	The date/time to end filling, if other than the full time series period.	If blank, fill the full period.
Intercept	Specify as 0 to force the intercept of the best-fit line through the origin. This is made available only for OLS regression analysis on untransformed data.	Parameter is optional and if specified the default is to not force the intercept through zero.
OutputFile	Output file for the results, either as a file name to be written to the working directory, or a full path.	If not specified, partial results of the analysis may be available in the log file.

A sample commands file is as follows:

```
Will need an example.
```

Command Reference

fillMOVE1()

Fill Missing Time Series Data Using MOVE1 Procedure

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The **fillMOVE1()** command has not been enabled. This documentation serves as a reference for the MOVE1 procedure. Refer to the **fillMOVE2()** command.

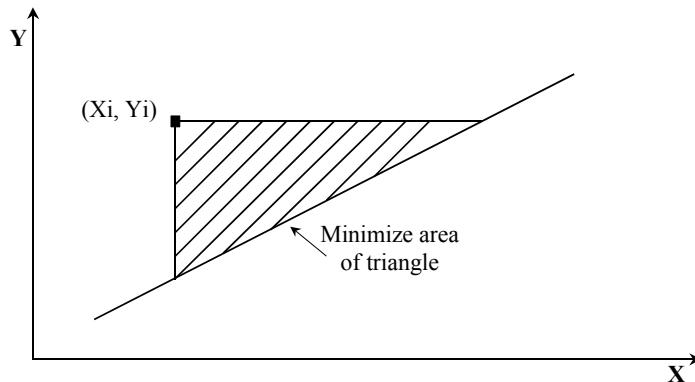
The `fillMOVE1()` command is more sophisticated than the `fillRegression()` command.

Maintenance of variance extension (MOVE) procedures are methods of fitting straight lines to data. The slope and intercept of the MOVE equations are computed differently than in ordinary least squares (OLS) regression (see the `fillRegression()` command for a discussion of OLS regression). As shown below, an area of a triangle is minimized in the MOVE procedures rather than a vertical distance as in OLS regression. The MOVE procedures do not provide the minimum-variance estimate of a single value but an ensemble of points estimated by the MOVE procedures will have the same variability as the true values.

MOVE procedures are useful in extending record at gaging stations where the extended record will be subsequently used in another analysis such as frequency analysis. MOVE procedures will provide about the same estimates as OLS regression near the mean of the data but will provide smaller and larger estimates at the extremes of the data set. The slope of the MOVE relation is steeper than OLS regression. The MOVE procedures are based on only one independent variable and the assumption is that there is a linear relation between the dependent and independent variables. If the untransformed data are not linearly related, then it is common to transform the data using a logarithmic transformation.

The MOVE.1 procedure uses just the data from the N_1 years of concurrent data. The MOVE.2 procedure (see the `fillMOVE2()` command) uses the Two-Station Comparison procedure described in **Appendix 7 of Bulletin 17B, Guidelines for Determining Flood Flow Frequency, USGS**, to compute improved estimates of the mean and variance for the dependent time series and uses all the data at the dependent time series to estimate the mean and variance of the dependent time series. The MOVE.2 procedure has been shown to be marginally better than MOVE.1.

Maintenance of Variance Extension (MOVE)



The MOVE.1 equation is used to estimate values for the dependent time series from the independent time series:

$$Y_i = \bar{Y}_1 + \frac{S_{y1}}{S_{x1}} \left[X_i - \bar{X}_1 \right]$$

or

$$Y_i = a + bX_i$$

where

N_1 = concurrent or overlapping period of record

\bar{X}_1 = mean for independent variable for N_1 years

\bar{Y}_1 = mean for dependent variable for N_1 years

S_{y1} = standard deviation for N_1 years

S_{x1} = standard deviation for N_1 years

$$b = \frac{S_{y1}}{S_{x1}}$$

$$a = \bar{Y}_1 - b\bar{X}_1$$

Note that the slope of the line does not include the correlation coefficient. This is the only difference between OLS regression and MOVE.1.

Command Reference

fillMOVE2()

Fill missing data in time series using the Maintenance of Variance Extension (MOVE.2) procedure

Version 06.16.01, 2006-04-17, Color, Acrobat Distiller

The `fillMOVE2()` command fills missing data in a time series using the MOVE.2 procedure (see the `fillMOVE1()` command for background information). The MOVE.2 procedure uses the Two-Station Comparison procedure described in **Appendix 7 of Bulletin 17B, Guidelines for Determining Flood Flow Frequency, USGS**, to compute improved estimates of the mean and variance at the dependent or short-term station and uses all the data at the dependent time series to estimate the mean and variance of the dependent time series. The MOVE.2 procedure has been shown to be marginally better than MOVE.1. The following MOVE.2 equation is used to estimate values for the dependent time series from the independent time series:

$$Y_i = \bar{Y} + \frac{S_y}{S_x} [X_i - \bar{X}]$$

where

Y_i = discharge for dependent time series

X_i = discharge for independent time series

\bar{X} = mean for independent time series for $N_1 + N_2$ years (N_2 is the additional years in the long-term time series)

S_x = standard deviation for independent time series for $N_1 + N_2$ years

$$\bar{Y} = \bar{Y}_1 + \frac{N_2}{N_1 + N_2} [b(\bar{X}_2 - \bar{X}_1)] \quad (\text{Equation 7-5a for Two-Station Comparison in Appendix 7 of Bulletin 17B})$$

$$S_y^2 = \frac{1}{(N_1 + N_2 - 1)} [(N_1 - 1)S_{y1}^2 + (N_2 - 1)b^2 S_{x2}^2 + \frac{N_2(N_1 - 4)(N_1 - 1)}{(N_1 - 3)(N_1 - 2)} (1 - r^2)S_{y1}^2 + \frac{N_1 N_2}{N_1 + N_2} b^2 (\bar{X}_2 - \bar{X}_1)^2]$$

(Equation 7-10 for Two-Station Comparison in Appendix 7 of Bulletin 17B)

where

$b = r \frac{S_{y1}}{S_{x1}}$, r = correlation coefficient (Note that b is the slope of the ordinary least squares regression line.)

N_1 = concurrent or overlapping period of record

N_2 = additional years available at long-term site

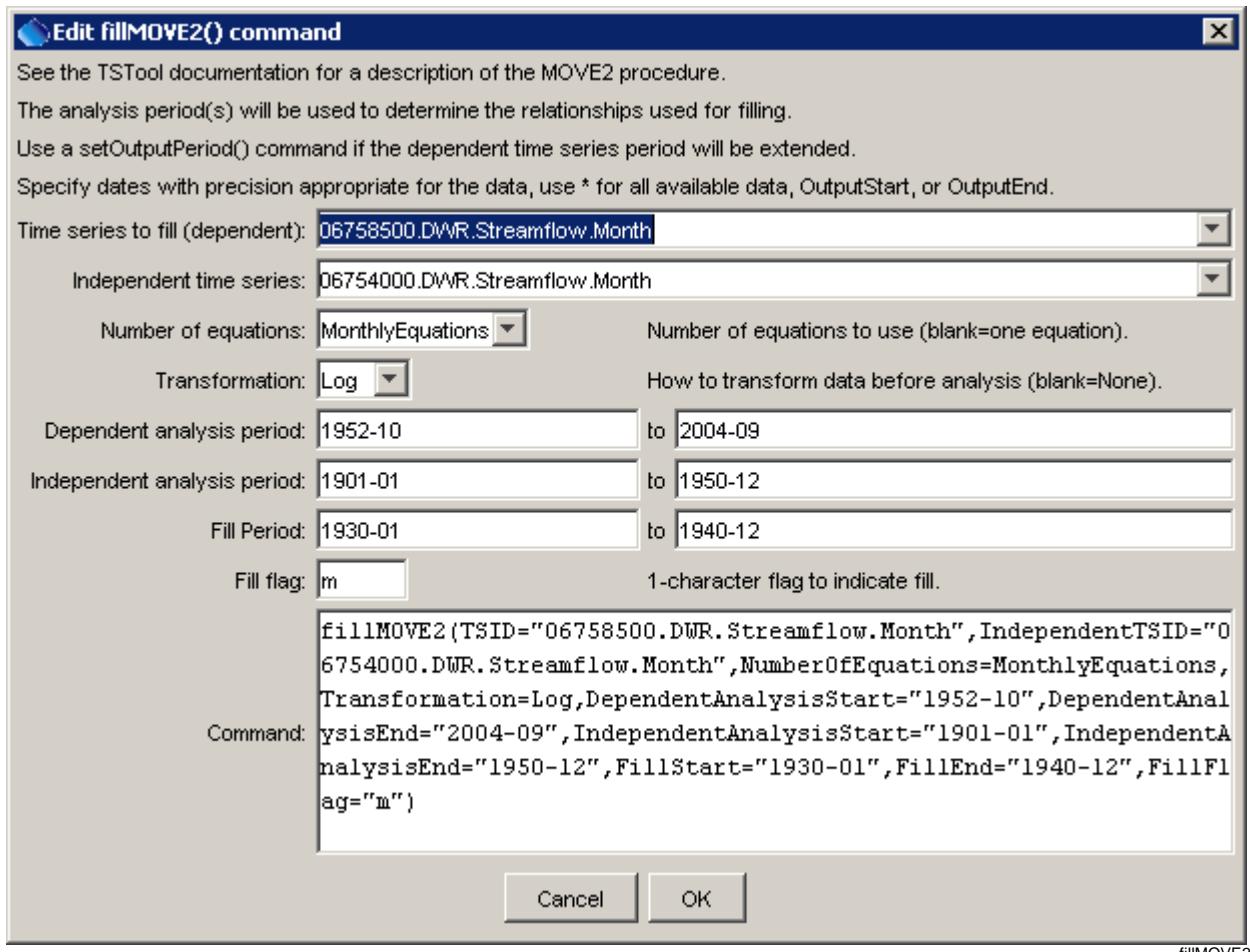
\bar{X}_1 = mean of independent time series for N_1 years

\bar{X}_2 = mean of independent time series for N_2 years

S_{y1} = standard deviation of dependent time series for N_1 years

S_{x1} = standard deviation of independent time series for N_1 years

The following dialog is used to edit the command and illustrates the command syntax.



fillMOVE2() Command Editor

The command syntax is as follows:

```
fillMOVE2 (param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled (dependent time series).	None – must be specified.
IndependentTSID	The time series identifier or alias for the independent time series, to supply data.	None – must be specified.
NumEquations	OneEquation or MonthlyEquations, indicating how many relationships are to be determined.	OneEquation
Transformation	Log or None, indicating the type of data transformation. If the Log option is used, zero and negative values are set to .001 (-999 values are treated as missing data and are ignored), and the data values are transformed using log10.	None
Dependent Analysis Start/End	The period for N ₁ (overlapping data) that is used to analyze the dependent time series. For example, this may be the unregulated period for streamflow data. Typically this is longer than the independent analysis period.	Analyze the full period.
Independent Analysis Start/End	The period for N ₂ (non-overlapping data) that is used to analyze the independent time series. For example, this may be the unregulated period for streamflow data.	Analyze the full period.
FillStart	The date/time to start filling.	Fill the full period.
FillEnd	The date/time to end filling.	Fill the full period.
FillFlag	A single character to be used to flag filled points on graphs and other output.	Do not flag filled data.

A sample commands file is as follows:

```
startLog(LogFile="commands.TSTool.log", Suffix="Date")
setOutputPeriod(1901-01,2004-12)
# 06758500 - SOUTH PLATTE RIVER NEAR WELDONA
06758500.DWR.Streamflow.Month~HydroBase
# 06754000 - SOUTH PLATTE RIVER NEAR KERSEY
06754000.DWR.Streamflow.Month~HydroBase
fillMOVE2(TSID="06758500.DWR.Streamflow.Month",
IndependentTSID="06754000.DWR.Streamflow.Month",
NumberOfEquations=MonthlyEquations, Transformation=Log,
DependentAnalysisStart="1952-10", DependentAnalysisEnd="2004-09",
IndependentAnalysisStart="1901-01", IndependentAnalysisEnd="1950-12",
FillStart="1930-01", FillEnd="1940-12", FillFlag="m")
```

This page is intentionally blank.

Command Reference

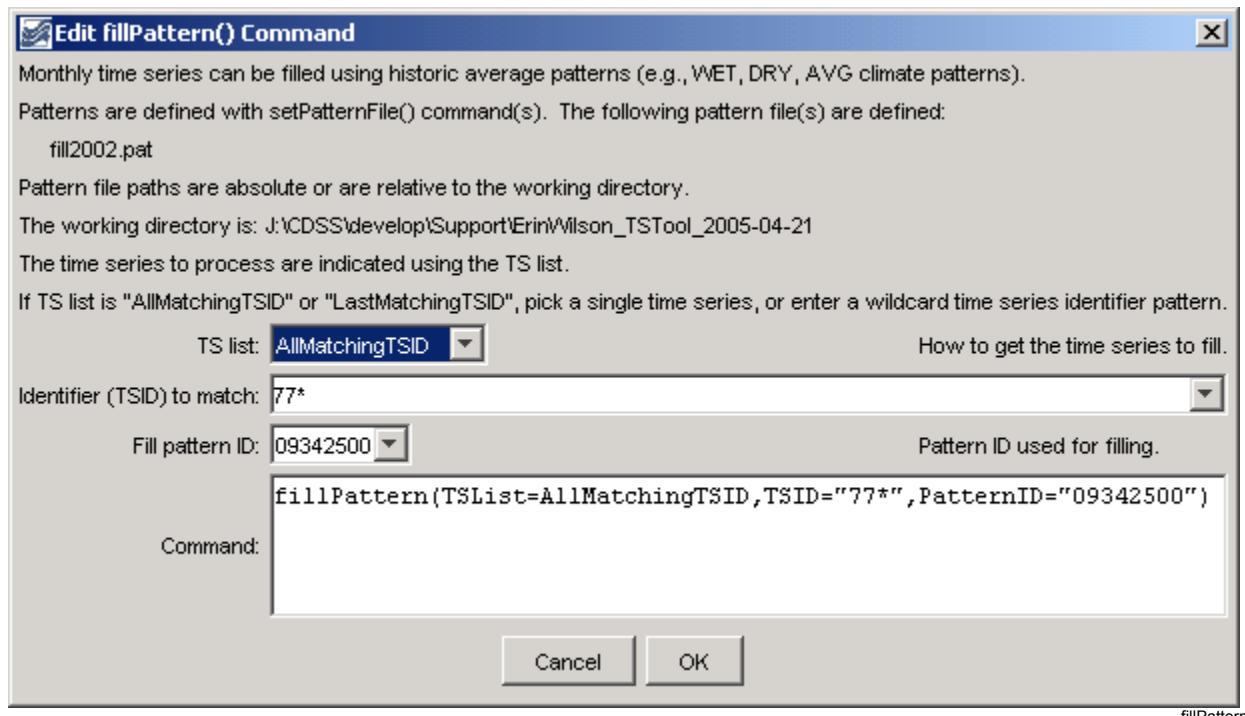
fillPattern()

Fill Missing Time Series Data Using Historical Average Patterns

Version 06.10.00, 2005-04-27, Color, Acrobat Distiller

The `fillPattern()` fills missing data in a time series using historic averages based on a pattern file. For example, if May, 1910 is missing and the pattern indicates that May, 1910 is a WET month, then the average of all WET Mays is used to fill the time series. The pattern file indicates the WET/DRY/AVG patterns and the time series to be filled supplies data to compute averages, for use in filling. **This feature is enabled for monthly data only.** Averages are computed as described for the `fillHistMonthAverage()` command. There is currently no way to limit the fill operation to a period (the entire time series is filled). This command works in conjunction with the `setPatternFile()` command. See below for an example of a fill pattern file. One or more patterns can be included in each pattern file, similar to StateMod time series files (see the **StateMod Input Type** appendix), and multiple pattern files can be used, if appropriate.

The following dialog is used to edit the `fillPattern()` command and illustrates the syntax of the command.



fillPattern() Command Editor

The command syntax is as follows:

```
fillPattern(param=value,...)
```

The obsolete syntax is as follows:

```
fillPattern(TSID,PatternID)
```

Command Parameters

Parameter	Description	Default
TSLIST	Indicate how the list of time series to process should be determined, one of: <ul style="list-style-type: none"> ▪ AllTS – all time series ▪ AllMatchingTSID – all time series that have identifiers matching the given TSID parameter. ▪ LastMatchingTSID – only the last time series matching the given TSID parameter ▪ SelectedTS – all time series that have been selected with <code>selectTimeSeries()</code> commands. 	None – must be specified. For old syntax, <code>AllMatchingTSID</code> will be used if <code>TSID</code> contains wildcards or <code>LastMatchingTSID</code> if <code>TSID</code> is specified without wildcards.
TSID	The time series identifier or alias for the time series to be filled. A pattern containing the * wildcard character can be used to process multiple time series (e.g., * or 29*).	Must be specified if the TSLIST parameter has a value of <code>AllMatchingTSID</code> or <code>LastMatchingTSID</code> .
PatternID	The pattern identifier, matching a pattern read from the file(s) specified with <code>setPatternFile()</code> commands.	None – must be specified.

A sample commands file is as follows:

```
# Read the time series to fill...
readStateMod(..\StateMod\sjm_prelim.ddh)
# Read the file containing pattern data...
setPatternFile("fill.pat")
# Fill time series with identifiers starting with "30", using
# the specified pattern...
fillPattern(TSLIST=AllMatchingTSID,TSID="30*",PatternID="09034500")
# Write the results...
writeStateMod(..\StateMod\sjm.ddh",*)
```

The above example fills all diversion time series with identifier starting with 30, using the pattern 09034500 (a stream gage for the region).

Command Reference

fillProrate()

Fill Missing Time Series Data by Prorating Values in Another Time Series

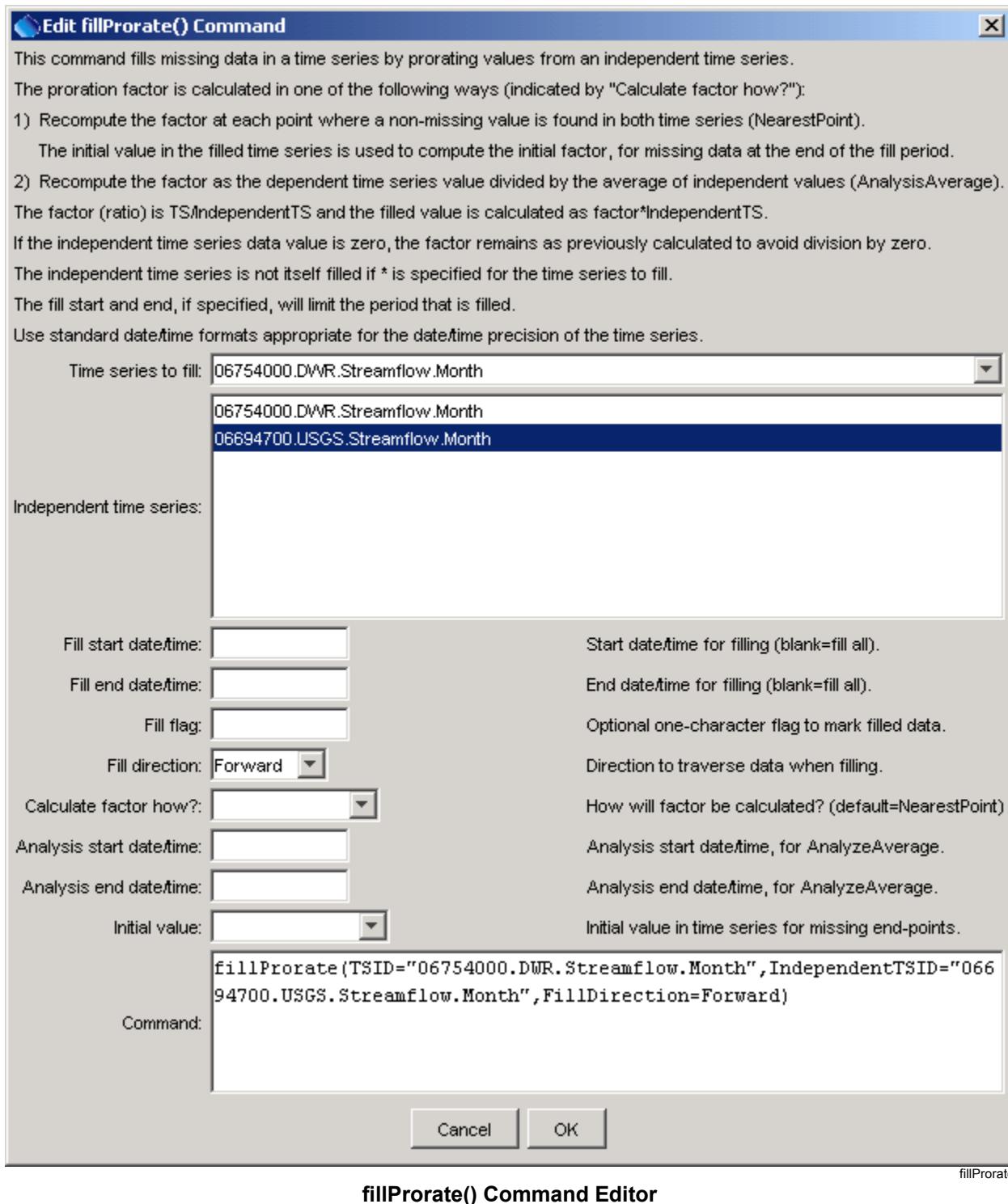
Version 06.10.05, 2005-08-04, Color, Acrobat Distiller

The `fillProrate()` command fills missing data in time series by prorating values from another time series. This fill technique is useful, for example, where two time series are likely to have the same general trend and ratio of data values. The ratio can be computed two ways, as specified by the `ComputeFactorHow` parameter:

- `NearestPoint` – causes the ratio to be recomputed each time that a non-missing value is found in both time series. The ratio is therefore used for filling until another value can be computed.
- `AnalyzeAverage` – computes the ratio as the average ratio of the time series and the independent time series. This was implemented to match an existing fill procedure but can lead to some bias in the results. A different overall average will be obtained depending on whether ratios are computed first and then averaged than if the sum of the numerators are added and divided by the sum of the denominators. In the former, the choice of which time series is in the denominator could impact results.

The initial computation of the ratio may require specifying an initial value due to missing data on the endpoints of the time series (see the `InitialValue` parameter). Alternatively, the time series can be filled in one direction first and then filled in the other direction with a second command.

The following dialog is used to edit the command and illustrates the syntax of the command:



fillProrate

fillProrate() Command Editor

The command syntax is as follows:

```
fillProrate(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled. Specify as * to fill all time series. Currently a pattern other than * is not supported.	None – must be specified.
IndependentTSID	The time series identifier or alias for the independent time series.	None – must be specified.
FillStart	The starting date/time for the fill.	Available period.
FillEnd	The ending date/time for the fill.	Available period.
FillFlag	A one-character flag to tag data values that are filled.	None – do not flag filled data.
FillDirection	Specify the direction of the fill as Forward or Backward.	Forward
CalculateFactorHow	Specify how to calculate the factor to use in proration, one of: <ul style="list-style-type: none"> • AnalyzeAverage – to calculate the factor of the average of the time series divided by the independent time series, using the analysis period. • NearestPoint – calculate the factor at the nearest point where both time series have non-missing values. 	NearestPoint
AnalysisStart	The starting date/time for the analysis, used when CalculateFactorHow=AnalysisAverage.	Analyze the full period.
AnalysisEnd	The ending date/time for the analysis, used when CalculateFactorHow=AnalysisAverage.	Analyze the full period.
InitialValue	The initial value to use for the filled time series, for cases where a value may not be available on the ends of the fill period, one of: <ul style="list-style-type: none"> • NearestBackward – search the time series backward for the nearest non-missing value. • NearestForward – search the time series forward for the nearest non-missing value. • Specify a number to use for the initial value. 	None – filling will not occur at the end.

A sample commands file is as follows:

```
# 06754000 - SOUTH PLATTE RIVER NEAR KERSEY
06754000.DWR.Streamflow.Month~HydroBase
# 06694700 - FOURMILE CREEK NEAR FAIRPLAY, CO.
06694700.USGS.Streamflow.Month~HydroBase
fillProrate(TSID="06754000.DWR.Streamflow.Month",
IndependentTSID="06694700.USGS.Streamflow.Month",
FillDirection=Forward,InitialValue=0)
06754000.DWR.Streamflow.Month~HydroBase
```

This page is intentionally blank.

Command Reference

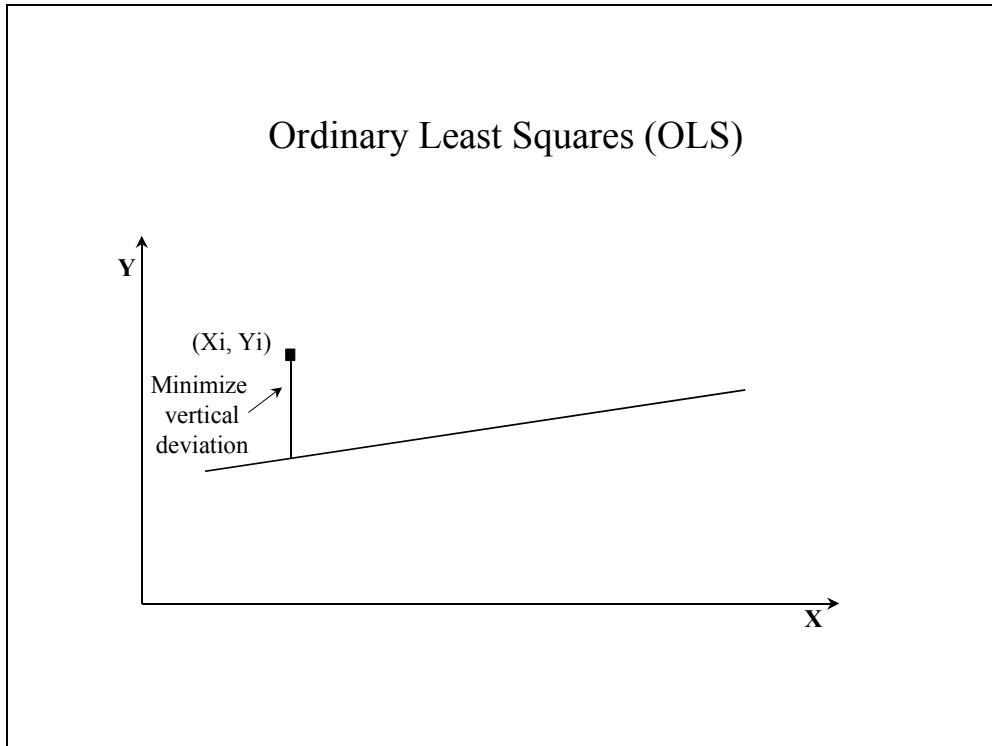
fillRegression()

Fill Missing Time Series Data Using Ordinary Least Squares Regression

Version 06.10.05, 2005-08-05, Color, Acrobat Distiller

The `fillRegression()` command fills missing data in a time series using ordinary least squares (OLS) regression (see also the `fillMOVE2()` command). Prior to TSTool version 05.00.00, regression was performed using `regress*` commands, of which there were several variations. These commands have been replaced with a single `fillRegression()` command, which uses parameters to indicate the type of regression. Regression can be applied only to regular interval time series. The first time series selected (dependent time series) will be filled using the other selected time series (independent time series). The periods of record and output period for the time series should be verified to make sure that the time series periods overlap sufficiently. The **Results...Graph - XY-Scatter** is a useful tool for reviewing data. Regression relationships are developed using the analysis period for the time series and are applied to the fill period. Refer to the log file and time series properties for analysis details.

In OLS regression, the vertical distance from the data point to the regression line is minimized. OLS regression provides the minimum-variance estimate for a single value or observation. However, if an ensemble of points are estimated from OLS regression, the estimated values will have lesser variability than the true values.



The following OLS equation is used to estimate values for the dependent time series from the independent time series:

$$Y_i = \bar{Y}_1 + r \frac{S_{y1}}{S_{x1}} \left[X_i - \bar{X}_1 \right]$$

or

$$Y_i = a + bX_i$$

where

N_1 = concurrent or overlapping period of record

\bar{X}_1 = mean for independent variable for N_1 years

\bar{Y}_1 = mean for dependent variable for N_1 years

S_{y1} = standard deviation for N_1 years

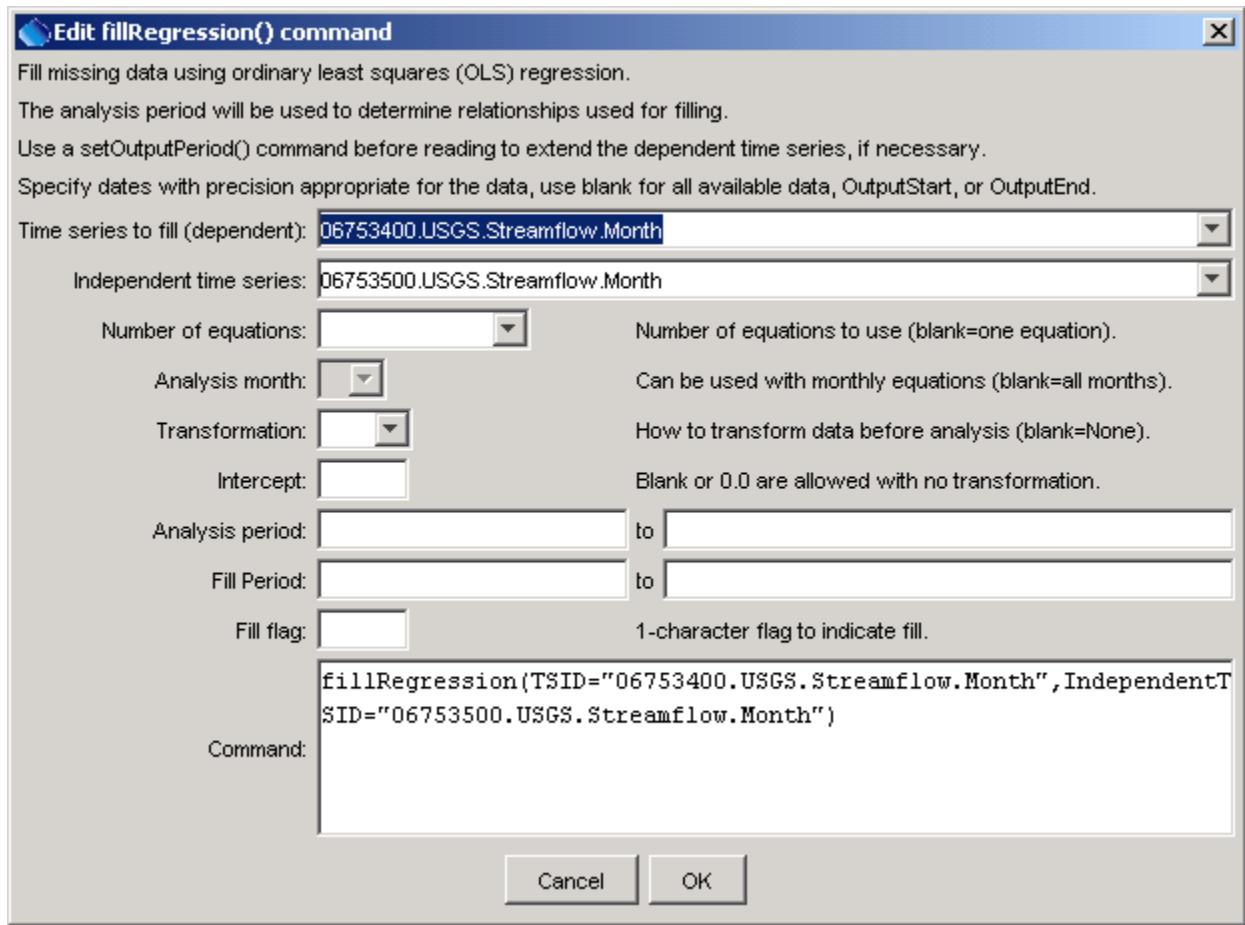
S_{x1} = standard deviation for N_1 years

$$b = r \frac{S_{y1}}{S_{x1}}, r = \text{correlation coefficient}$$

$$a = \bar{Y}_1 - b\bar{X}_1$$

Note that the correlation coefficient, r , is used to compute the slope, b , of the line.

The following dialog is used to edit the command and illustrates the syntax of the command:



fillRegression() Command Editor

fillRegression

The command syntax is as follows:

```
fillRegression(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled.	None – must be specified.
Independent TSID	The time series identifier or alias for the independent time series. Right-clicking on a time series allows it to be converted to a temporary time series (TEMPTS).	None – must be specified.
NumberOf Equations	The number of equations to use for the analysis: OneEquation or MonthlyEquations.	OneEquation
AnalysisMonth	Indicate the month to process when using monthly equations. Currently only a single month can be specified.	Process all months.
Transformation	Indicates how to transform the data before analyzing. Specify as None (previously Linear) or Log (for \log_{10}). If the Log option is used, zero and negative values are set to .001 (missing data values are ignored in the analysis).	None (no transformation).
Intercept	Specify as 0 to force the intercept of the best-fit line through the origin (not available for log transformation).	Parameter is optional and if specified the default is to not force the intercept through zero.
AnalysisStart	The date/time to start the analysis – use to focus on only a period appropriate from analysis. For example specify the unregulated period for streamflow.	Analyze the full period.
AnalysisEnd	The date/time to end the analysis – use to focus on only a period appropriate from analysis.	Analyze the full period.
FillStart	The date/time to start filling, if other than the full time series period.	Fill the full period.
FillEnd	The date/time to end filling, if other than the full time series period.	Fill the full period.
FillFlag	A single character that will be used to flag filled data.	Filled values will not be flagged.

A sample commands file is as follows (the `fillRegression()` command would be on one line in TSTool):

```
# 06753400 - LONETREE CREEK AT CARR, CO.
06753400.USGS.Streamflow.Month~HydroBase
# 06753500 - LONETREE CREEK NEAR NUNN, CO.
06753500.USGS.Streamflow.Month~HydroBase
fillRegression(TSID="06753400.USGS.Streamflow.Month",
IndependentTSID="06753500.USGS.Streamflow.Month")
```

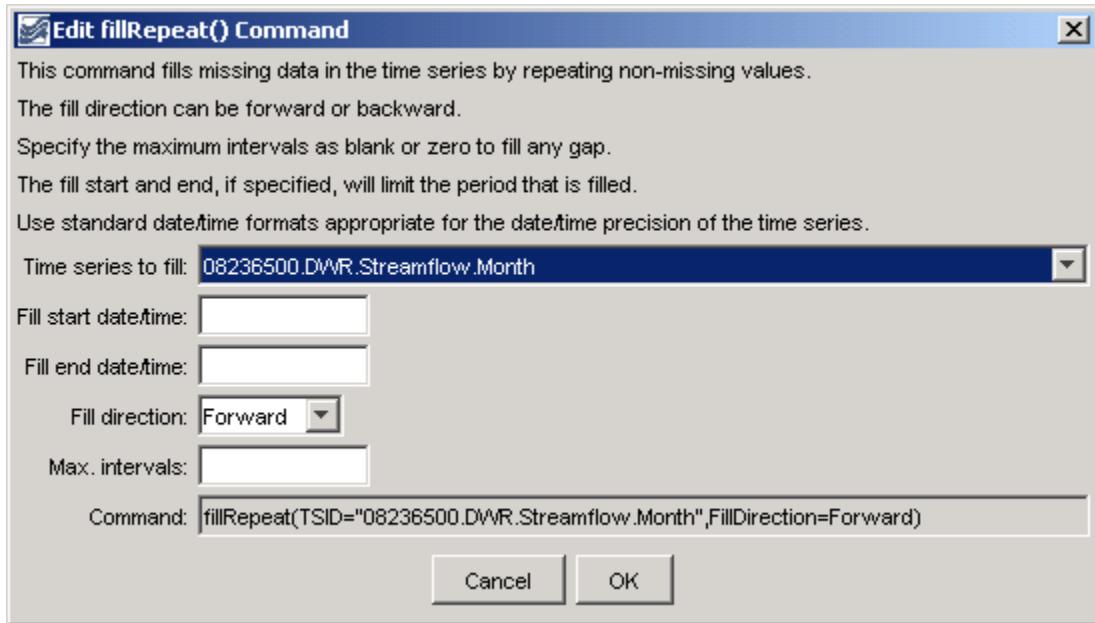
Command Reference

fillRepeat()

Fill Missing Time Series Data by Repeating Known Data Values

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `fillRepeat()` command fills missing data in time series by repeating observations until another observation is found. This fill technique is useful, for example, where time series are likely to be step-wise or nearly constant, such as some reservoir and diversion time series. The following dialog is used to edit the command and illustrates the syntax of the command.



fillRepeat

fillRepeat() Command Editor

The command syntax is as follows:

```
fillRepeat(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled. Specify as * to fill all time series.	None – must be specified.
FillStart	The starting date/time for the fill.	Available period.
FillEnd	The ending date/time for the fill.	Available period.
FillDirection	Specify the direction of the fill as Forward or Backward.	Forward
MaxIntervals	The maximum number of intervals to fill in a data gap.	Fill entire gap.

A sample commands file is as follows:

```
# 08236500 - ALAMOSA RIVER BELOW TERRACE RESERVOIR
08236500.DWR.Streamflow.Month~HydroBase
fillRepeat(TSID="08236500.DWR.Streamflow.Month",FillDirection=Forward)
```

Command Reference

fillUsingDiversionComments()

Fill missing time series data using HydroBase diversion comments

Version 06.19.00, 2006-05-19, Color, Acrobat Distiller

This command is only appropriate for use with diversion (e.g., DivTotal, DivClass data types) and reservoir release (e.g., RelTotal, RelClass data types) time series for the HydroBase input type.

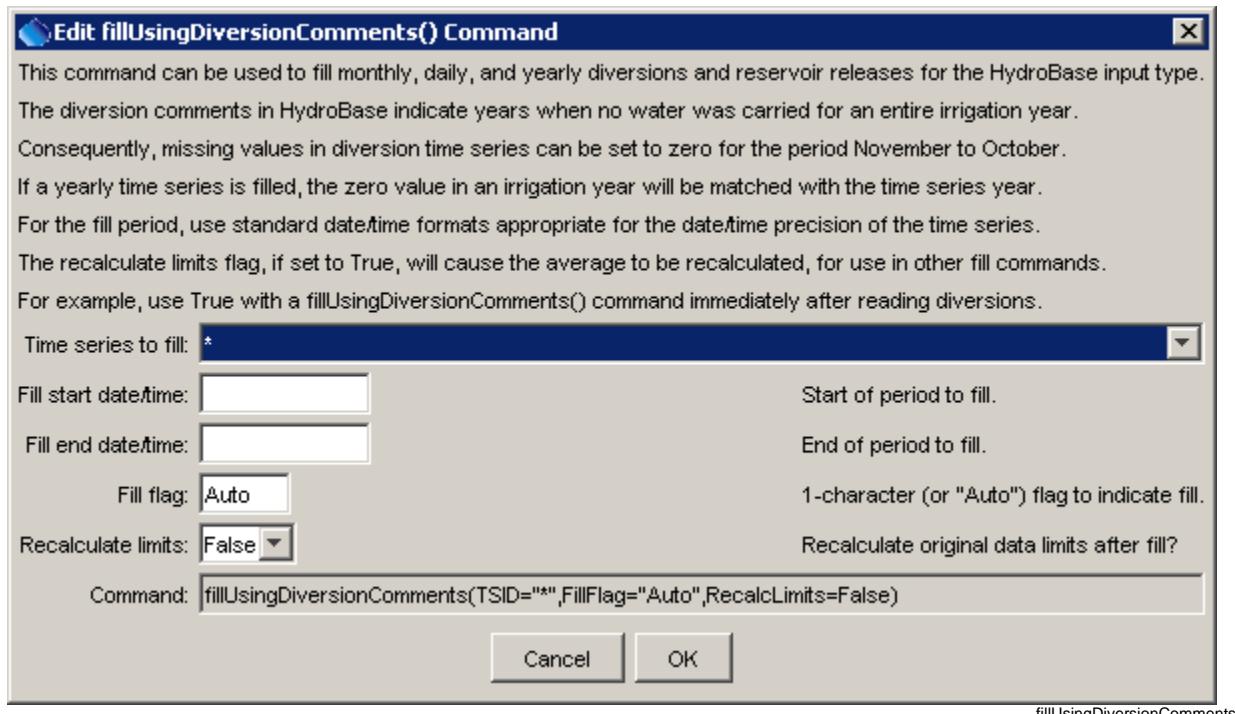
The `fillUsingDiversionComments()` command fills missing data in time series by using diversion comment information in HydroBase. HydroBase contains a *diversion_comment* table with a *not_used* field. If this field has one of the following values, the diversion (or reservoir release) data for the specified irrigation year can be interpreted as zero (see the State of Colorado's **Water Commissioner Manual** for more information):

- A – Structure is not usable
- B – No water is available
- C – Water available, but not taken
- D – Water taken in another structure

If a global output period has been specified (e.g., with the `setOutputPeriod()` command) then the time series will NOT be extended to include diversion comments beyond the output period. If NO output period has been specified, the time series WILL be extended to include the additional diversion comments. After setting additional zero values using this command, the limits of the time series can be recomputed, if appropriate, for use with the `fillHistMonthAverage()` command (see the `RecalcLimits=True` parameter).

See also the `readHydroBase()` and `TS Alias = readHydroBase()` commands, which also allow filling with diversion comments. See also the `readHydroBase()` and `TS Alias = readHydroBase()` commands, which allow filling with diversion comments after reading data. Refer to the **HydroBase Input Type** appendix for more information about diversion time series.

The following dialog is used to edit the command and illustrates the syntax of the command.



fillUsingDiversionComments

fillUsingDiversionComments() Command Editor

The command syntax is as follows:

```
fillUsingDiversionComments (param=value, ...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be filled. Specify as * to fill all time series.	None – must be specified.
FillStart	The starting date/time for the fill.	Available period.
FillEnd	The ending date/time for the fill.	Available period.
FillFlag	If specified as a single character or Auto, data flags will be enabled for the time series and each filled value will be tagged with the specified character. The flag can then be used later to label graphs, etc. The flag will be appended to existing flags if necessary. If Auto is specified, the “not used” value is used for the flag.	No flag is assigned.
RecalcLimits	Indicate whether the original data limits for the time series should be recalculated after the zero values are set.	False (additional zeros are not considered in the original data averages).

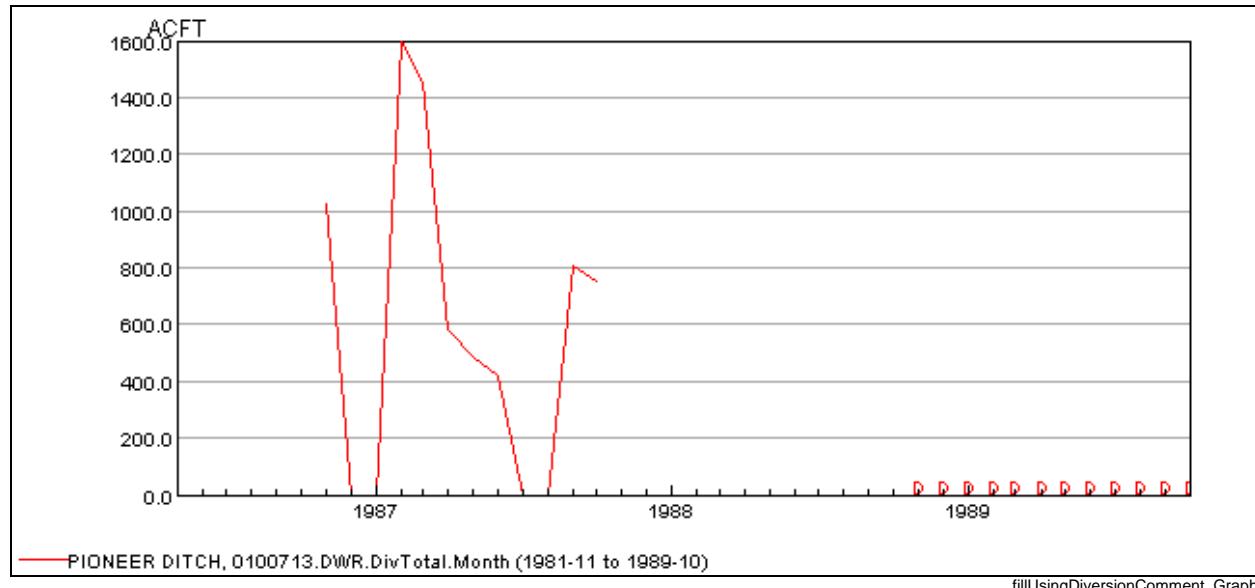
A sample commands file is as follows:

```
# 0100506 - PUTNAM DITCH
0100506.DWR.DivTotal.Month~HydroBase
# 0100503 - RIVERSIDE CANAL
0100503.DWR.DivTotal.Month~HydroBase
# 0100501 - EMPIRE DITCH
0100501.DWR.DivTotal.Month~HydroBase
fillUsingDiversionComments(TSID="*", RecalcLimits=True)
```

The following example fills one time series and labels the values with the flag.

```
# Set the date to cause comments NOT to automatically extend the period.  
# setOutputPeriod(1950-01,1980-12)  
# 0100713 - PIONEER DITCH  
0100713.DWR.DivTotal.Month~HydroBase  
fillUsingDiversionComments(TSID="*",FillFlag="Auto",RecalcLimits=False)
```

The corresponding graph created with data flags as labels is shown below:



This page is intentionally blank.

Command Reference

free()

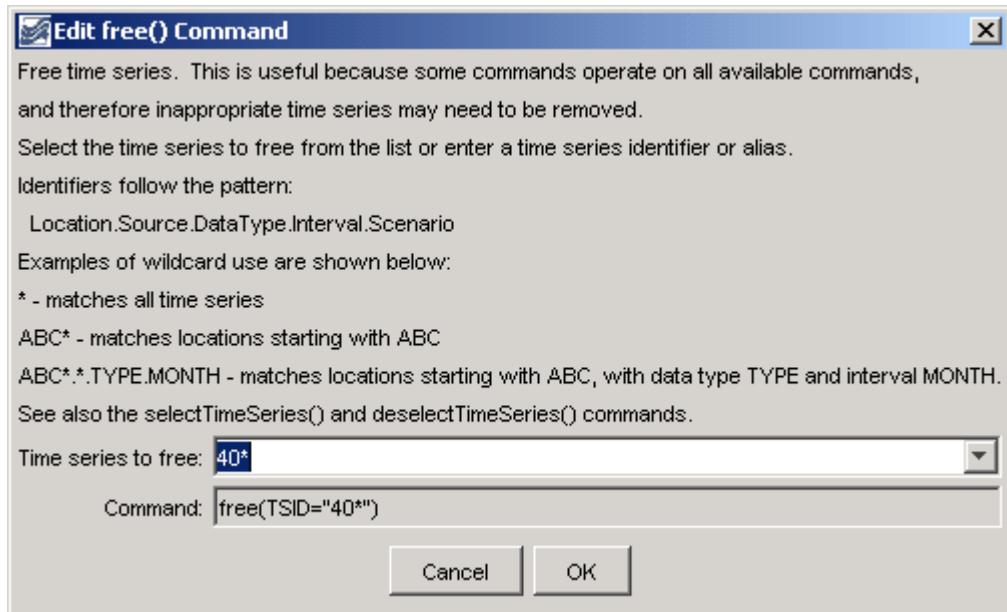
Free Time Series From Memory

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `free()` command frees the memory for the indicated time series. The time series will therefore not be available for use after that line in the commands file. This command is useful for discarding temporary time series needed for data manipulation (e.g., so that they are not written in output).

Rather than freeing time series, it may be more appropriate to use the `selectTimeSeries()` command, which can be used in conjunction with some commands to select time series and then operate on the selected time series. This approach allows selective use of time series and minimized the need for `free()` commands.

The following dialog is used for editing the command and illustrates the command syntax.



free

free() Command Editor

The command syntax is as follows:

```
free(TSID)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be freed. Specific time series can be specified or the identifier can contain wildcards to match patterns in time series identifiers (see figure above).	None – must be specified.

A sample commands file is as follows:

```
free(TSID="40*")
```

Command Reference

lagK()

Lag and attenuate (route) a time series

Version 06.10.04, 2005-07-14, Color, Acrobat Distiller

The `lagK()` command can be used to lag and attenuate an input time series, resulting in a new time series. The command is commonly used to route an instantaneous flow time series through a stretch of river (reach). Lag and K routing is a common routing method that combines the concepts of:

1. lagging the inflow to simulate travel time in a reach and,
2. attenuating the wave to simulate the storage-outflow relationship for the reach (see [Figure 1](#)).

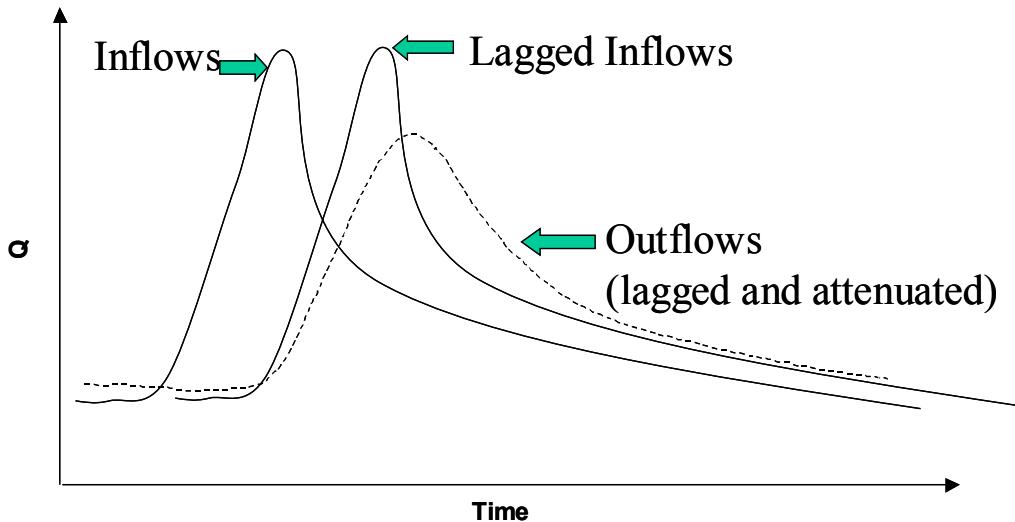


Figure 1: Lag and K Routing

At its fundamental level, the method solves the continuity equation using an approach similar to Muskingum routing (assuming that the Muskingum parameter representing wave storage is negligible). The governing equation for this routing method is given as:

$$Q_{in} - Q_{out} = \frac{\Delta S}{\Delta t}$$

where:

Q_{in} = instantaneous inflow [rate] lagged appropriately,
 Q_{out} = instantaneous outflow [rate] lagged appropriately,
 ΔS = change in storage in the reach [volume],
 Δt = time difference.

The relationship assumes an outflow-storage relationship of the form:

$$S = k \cdot Q_{out},$$

where:

k = attenuation for the outflow [time].

To ensure accurate results, k should be larger or equal to $\Delta t/2$. For discrete time steps these relationships translate into:

$$O_2 = \frac{I_1 + I_2 + \frac{2S_1}{\Delta t} - O_1}{\frac{2k}{\Delta t} + 1}, \quad k \geq \frac{\Delta t}{2}$$

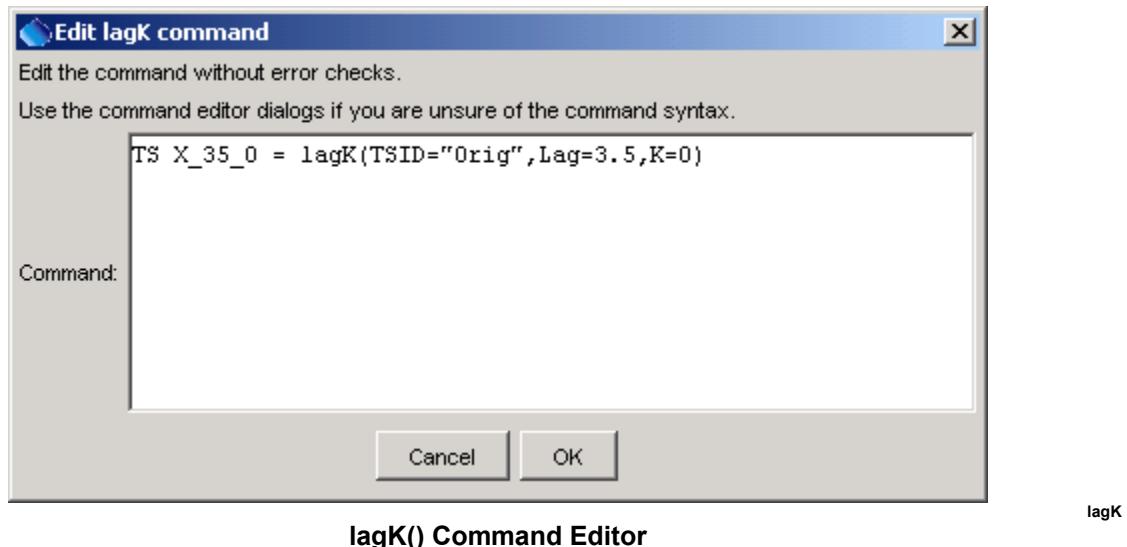
where: I_1 and I_2 are the lagged inflows into the reach at the previous and current time step, respectively,
 O_1 and O_2 are the outflows out of the reach at the previous and current time step, respectively,
 S_1 is the storage within the reach at the previous time step, defined as $S_1 = k \cdot O_1$, and
 Δt is the time difference between the two time steps.

In the case that either I_1 , I_2 or O_1 are missing, these values will be set in the following order:

1. Use data from an observed time series (see ObsTSID parameter below).
2. Use the nearest value in the input time series (see FillNearest parameter below).
3. Use the nearest value in the observed time series (see FillNearest parameter and the ObsTSID parameter below).
4. Use a defined default flow value (see DefaultFlow parameter below).

By default, the identifier of the resulting time series is the same as the original input time series, with the data subtype set to “routed” (e.g., Streamflow becomes Streamflow-routed)

The following dialog is used to edit the command and illustrates the syntax for the command (currently only a text-based editor is available and therefore the parameters must be manually entered):



Values for Lag and K can usually be established by comparing routed flows to downstream observations. Alternatively, the Lag can be estimated using the reach length and wave speed in the reach. Without any other information, K can be set to Lag/2.

The command syntax is as follows:

```
TS routed = lagK(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	Identifier or alias for the time series to be routed. It is assumed that this series describes an instantaneous flow. Due to the lagging, the first data values required for the computation of O_2 are not available within this time series and are therefore set to values set in the InflowStates parameter. See also the ObsTSID time series, and the FillNearest and DefaultFlow parameters.	None – must be specified.
ObsTSID	Identifier or alias for an observed time series. If specified, the missing values in the TSID time series will be taken from the observed time series if non-missing. ObsTSID can be used in conjunction with FillNearest to substitute a missing value in the TSID time series with the nearest non-missing value in ObsTSID.	None

Command Parameters (continued)

FillNearest	If set to True, then when a missing data value is found anywhere in the lagged period, a replacement value will be determined by searching forward and back in time in the input time series to find the nearest non-missing value. The maximum search window depends on the interval of the TSID time series: <ul style="list-style-type: none"> • <= Seconds: 1000 intervals • Minute, Hour: 1 day • Day: 1 Week • > Day: 1 interval only The assumption is that a flow value close in time will be representative of the missing value and will not result in significant errors. This option has lower precedence than specifying the ObsTSID data. It can also find non-missing data in the ObsTSID if ObsTSID is defined (lower precedence). Both options have a higher precedence than DefaultFlow.	False
DefaultFlow	A flow value in the units of the input time series that is substituted for missing values in the input time series. This has the lowest precedence of all missing data substitutions. It will be applied at any time in the lagged period.	0
Lag	Lag time for the modeled reach in the units of the TSID time series base interval. For example, if the input time series is 10 minutes, the units of Lag are assumed to be minute. The Lag value is not required to be evenly divisible by the time step interval; values in the time series between time steps will be linearly interpolated.	Required
K	Attenuation factor to be applied to the wave. The units of K are time, and like the Lag value, it is assumed to have the same units as the input time series.	Required
InflowStates	Comma-delimited list of default inflow values prior to the start of the time series. The order of the values is earliest to latest. The array must specify (Lag/multiplier) + 1 values; i.e., a 10 minute interval with a LAG of 30 must be provided with 30/10 + 1 = 4 inflow carryover values. Note: Specifying values that are not consistent with the Lag and K parameters will result in oscillation!	0 for each value
OutflowStates	Comma-delimited list of default outflow values prior to the start of the time series. See InflowStates for details.	0 for each value

A sample commands file is as follows:

```
TS LKPN6routed = lagK(TSID=LKPN6.USGS.QIN.1HOUR,Lag=3,K=2,FillNearest=true)
```

Command Reference

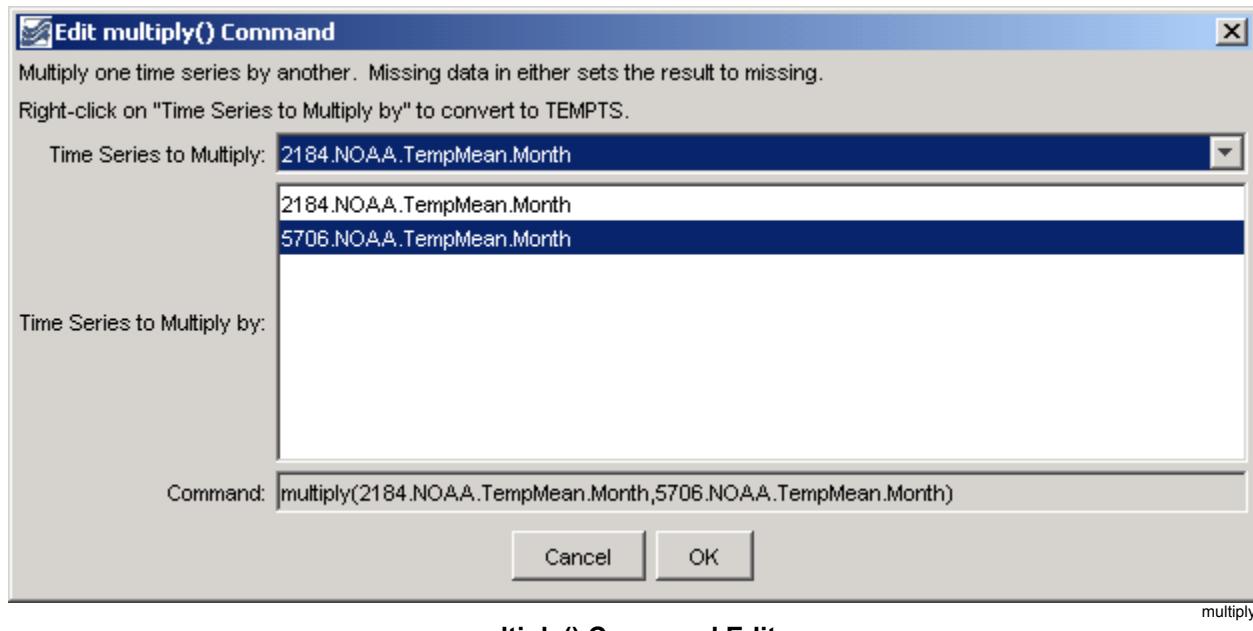
multiply()

Multiply the Data Values in a Time Series by Data Values in Another Time Series

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `multiply()` command multiplies one time series by another. This is similar to `scale()` except the `multiply()` command essentially uses a time series of scale values. Missing data in either time series causes the result to be missing.

The following dialog is used to edit the command and illustrates the syntax of the command. Right-clicking on a time series to divide by allows toggling of the TEMPTS key word. This will cause the time series to divide to be read and then discarded, simplifying the input file. However, to insert the time series the first time, the time series must have been listed in the commands file as a time series identifier.



multiply() Command Editor

The command syntax is as follows:

```
multiply(TSID,MultiplierTSID)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. Right-clicking on a time series to divide by allows toggling of the TEMPTS key word. This will cause the time series to divide to be read and then discarded, simplifying the input file. However, to insert the time series the first time, the time series must have been listed in the commands file as a time series identifier.	None – must be specified.
MultiplierTSID	The time series identifier or alias for the time series that is the multiplier.	None – must be specified.

A sample commands file is as follows (this example does not necessarily make sense – the `multiply()` command is used for numerical calculations in an analysis):

```
# 2184 - DEL NORTE 2 E
2184.NOAA.TempMean.Month~HydroBase
# 5706 - MONTE VISTA 2 W
5706.NOAA.TempMean.Month~HydroBase
multiply(2184.NOAA.TempMean.Month,5706.NOAA.TempMean.Month)
```

Command Reference

TS Alias = newDayTSFromMonthAndDayTS()

Create a new daily time series from monthly total and daily pattern

Version 06.20.00, 2006-10-08, Color, Acrobat Distiller

The newDayTSFromMonthAndDayTS () command creates a new daily time series by distributing a monthly time series according to the pattern of the independent daily time series. This command currently only handles processing monthly ACFT and daily CFS time series. This command is useful where a monthly flow time series is known at a location, and a daily pattern is known at a related gage. The new time series is assigned the given identifier and alias. The following calculations are performed:

$$DayTS2_i = MonthTS2 \frac{ACFT}{NDAYS} * \left(\frac{1DAY}{86400s} \right) \left(\frac{43560FT^2}{1AC} \right) * \left(\frac{\sum_{i=1}^{Nday \sin Month} DayTS1_i}{\sum_{i=1}^{NDAYS} DayTS1_i} \right)$$

where, for days in a month:

$DayTS2_i$ = the daily value being estimated in daily time series 2

$MonthTS2$ = the monthly value being used for volumes for time series 2, shown in units of ACFT/NDAYS (equivalent to ACFT/Month)

$NDAYS$ = the number of days in the month

$DayTS1_i$ = the daily value for indicator daily time series 1

$\sum DayTS1_i$ = the sum of the daily values for indicator time series for the a month

In summary, the monthly volume in ACFT/NDAYS is first converted to an average monthly CFS rate by multiplying by 43560/86400 (or 1/1.9835), and finally the average CFS value is prorated by the ratio of the indicator daily time series daily value divided by the total daily flows for the month, to give a daily CFS value for each day of the month. In this case, the last term is simply a ratio (converting daily average CFS to daily ACFT and calculating the ratio would result in the same value).

Days with missing data are excluded from the summation and the estimated values.

For example, consider May a may total for MonthTS2 = 1001.7 ACFT and daily values (CFS) as follows:

Day 1 = 14

14

13

13

14

14

15

15

15

16

17

17

16

18

18

17

18

18

18

17

17

17

17

16

16

17

18

18

17

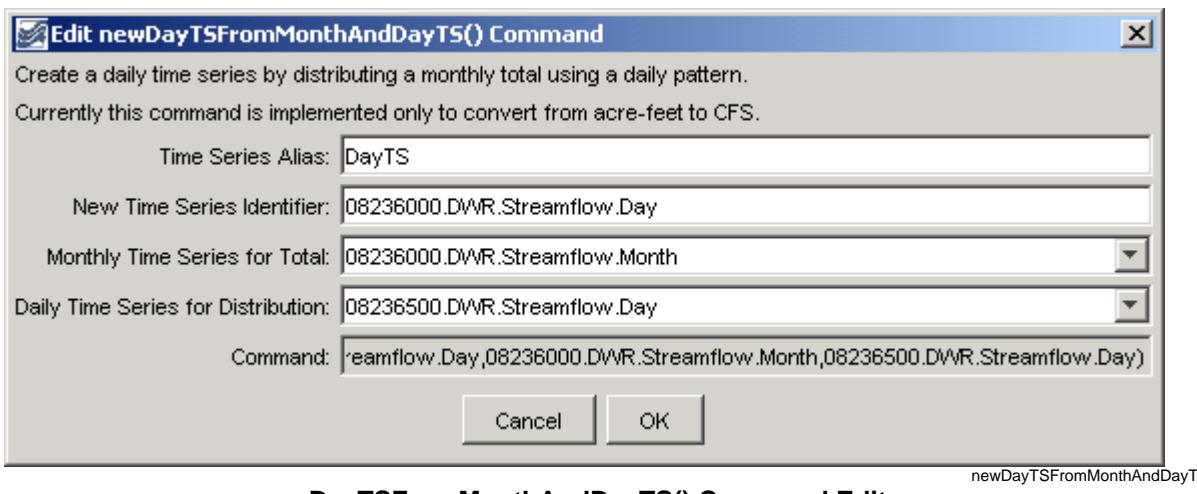
Day 31 = 17

The total is 505 CFS. The estimated value for day 1 of the second daily time series would then be:

$$1001.7 * (1/1.9835) * (14/505) = 14 \text{ CFS}$$

In this case, the indicator time series was the same as the time series being estimated and therefore the estimated value should be the same as the indicator.

The following dialog is used to edit the command and illustrates the syntax for the command.



newDayTSFromMonthAndDayTS

newDayTSFromMonthAndDayTS() Command Editor

The command syntax is as follows:

```
TS Alias = newDayTSFromMonthAndDayTS(NewTSID, MonthTSID, DayTSID)
```

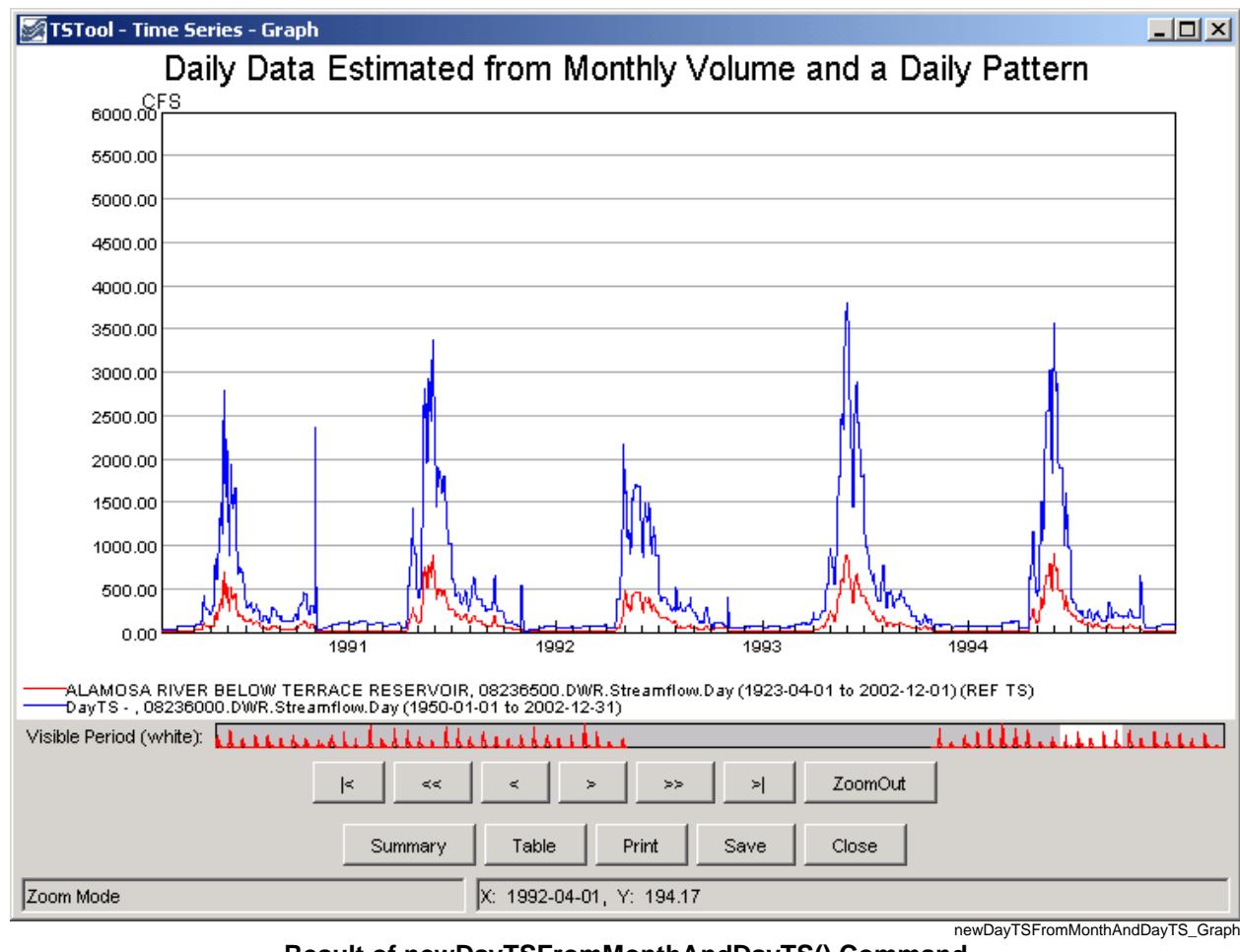
Command Parameters

Parameter	Description	Default
X	The alias of the new time series.	None – must be specified.
NewTSID	The time series identifier of the new time series. The interval must be Day.	None – must be specified.
MonthTSID	The time series identifier or alias for a monthly time series supplying monthly ACFT values.	None – must be specified.
DayTSID	The time series identifier or alias for a daily time series supplying daily flow values (only the pattern is used).	None – must be specified.

A sample commands file is as follows:

```
setOutputPeriod(1950-01,2002-12)
# 08236500 - ALAMOSA RIVER BELOW TERRACE RESERVOIR
08236500.DWR.Streamflow.Day~HydroBase
# 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR
08236000.DWR.Streamflow.Month~HydroBase
TS DayTS = newDayTSFromMonthAndDayTS(08236000.DWR.Streamflow.Day,
08236000.DWR.Streamflow.Month,08236500.DWR.Streamflow.Day)
```

A graph of data resulting from this command may look similar to the following. Note that the each time series have a similar pattern, but at different levels.



Result of newDayTSFromMonthAndDayTS() Command

Command Reference

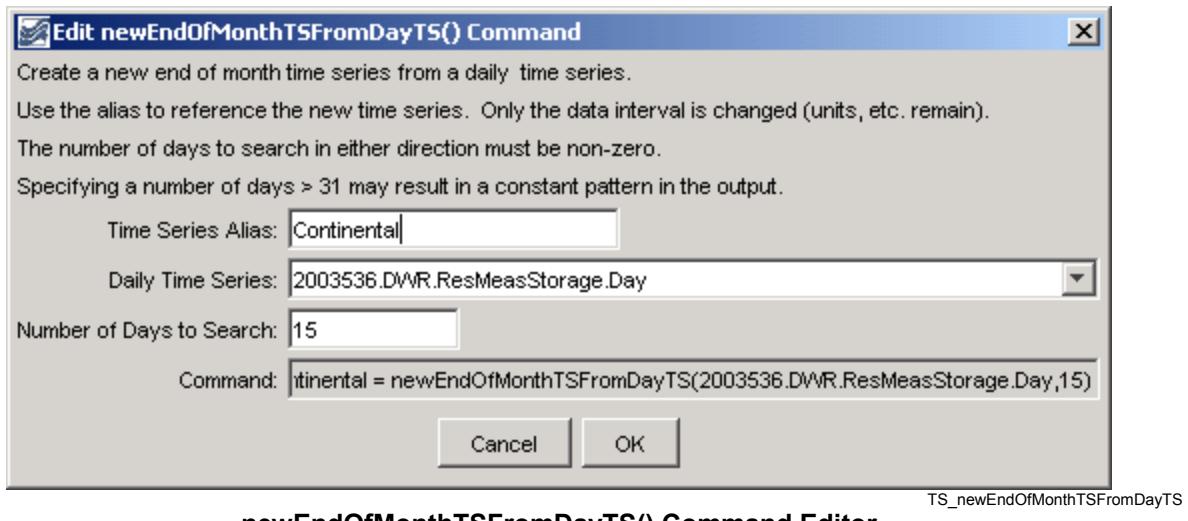
newEndOfMonthTSFromDayTS()

Use a Daily Time Series to Create an End of Month Time Series

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The newEndOfMonthTSFromDayTS() command replaces the older and more specific day_to_month_reservoir() command, and can be used to convert a daily reservoir storage time series to an end of month reservoir storage time series. The command can be applied to other data types (e.g., measured well levels).

Changing from a daily to an end of month monthly time series is accomplished by starting on the month-ending day and searching in both directions for a daily measurement. It is possible that no value will be found for a particular month, with the given restraints. In this case, other fill commands (e.g., fillInterpolate()) can be applied to estimate the remaining missing data. The following dialog is used to edit the command and illustrates the syntax of the command.



newEndOfMonthTSFromDayTS() Command Editor

The command syntax is as follows:

```
TS X = newEndOfMonthTSFromDayTS(TSID, Bracket)
```

Command Parameters

Parameter	Description	Default
X	The alias for the new time series.	None – must be specified.
TSID	The time series identifier or alias of the daily time series to be searched for data.	None – must be specified.
Bracket	The number of days to search from the end of the month, in order to find a daily value to transfer to the end of the month.	None – must be specified.

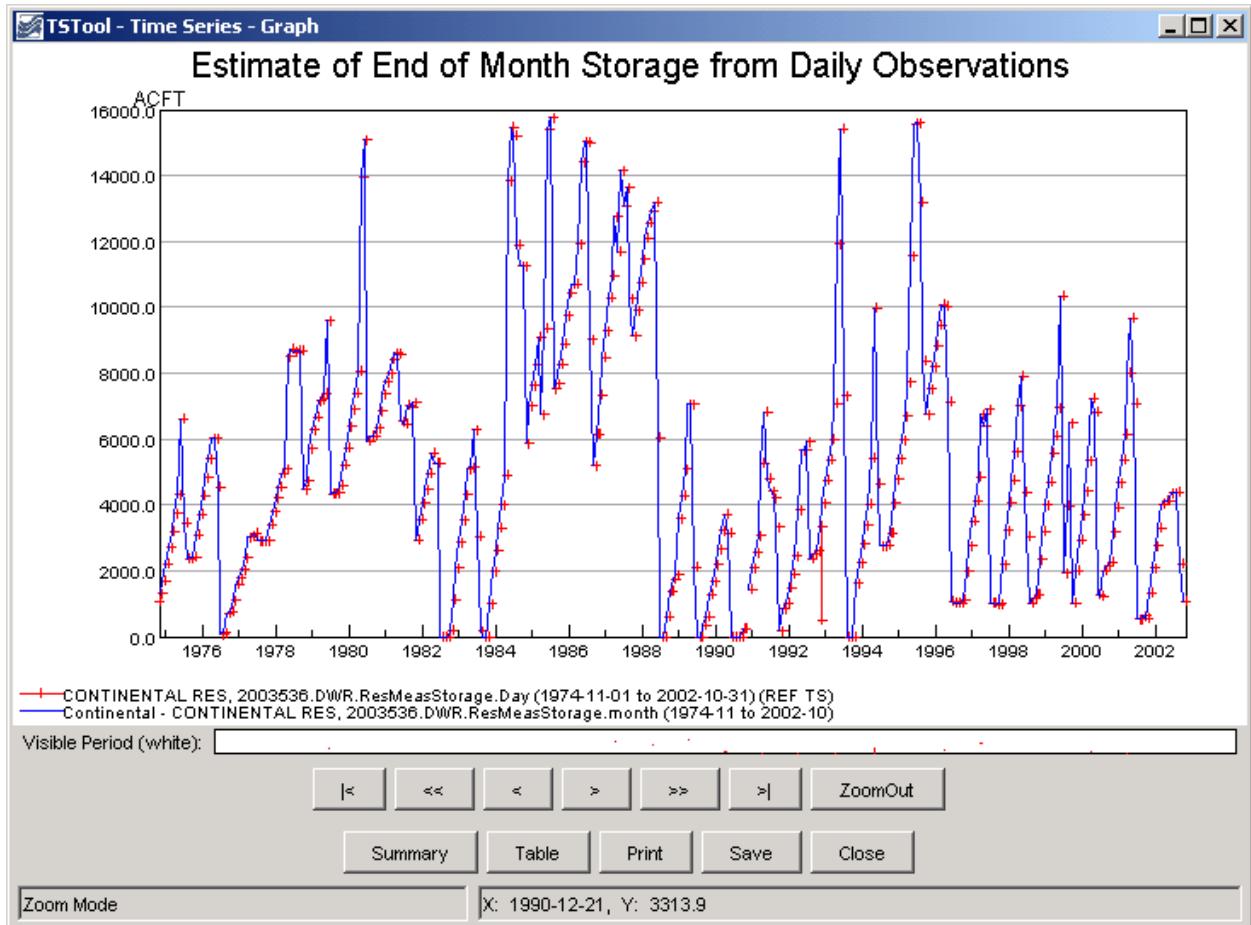
A sample commands file for estimating reservoir contents is:

```
# 2003536 - CONTINENTAL RES
2003536.DWR.ResMeasStorage.Day~HydroBase
TS Continental = newEndOfMonthTSFromDayTS(2003536.DWR.ResMeasStorage.Day, 15)
```

A sample commands file for estimating well levels is:

```
# 2006000 - 80CW100 WELL FWS23-20A
2006000.USGS.WellLevel.Day~HydroBase
TS WellMonth = newEndOfMonthTSFromDayTS(2006000.USGS.WellLevel.Day, 30)
fillInterpolate(WellMonth, 0, Linear)
```

To evaluate the results of this command, it is useful to graph both the input and results, changing the graph properties to add symbols to see the individual measurements, as shown in the following figure.



Results of newEndOfMonthTSFromDayTS() Command

This page is intentionally blank.

Command Reference

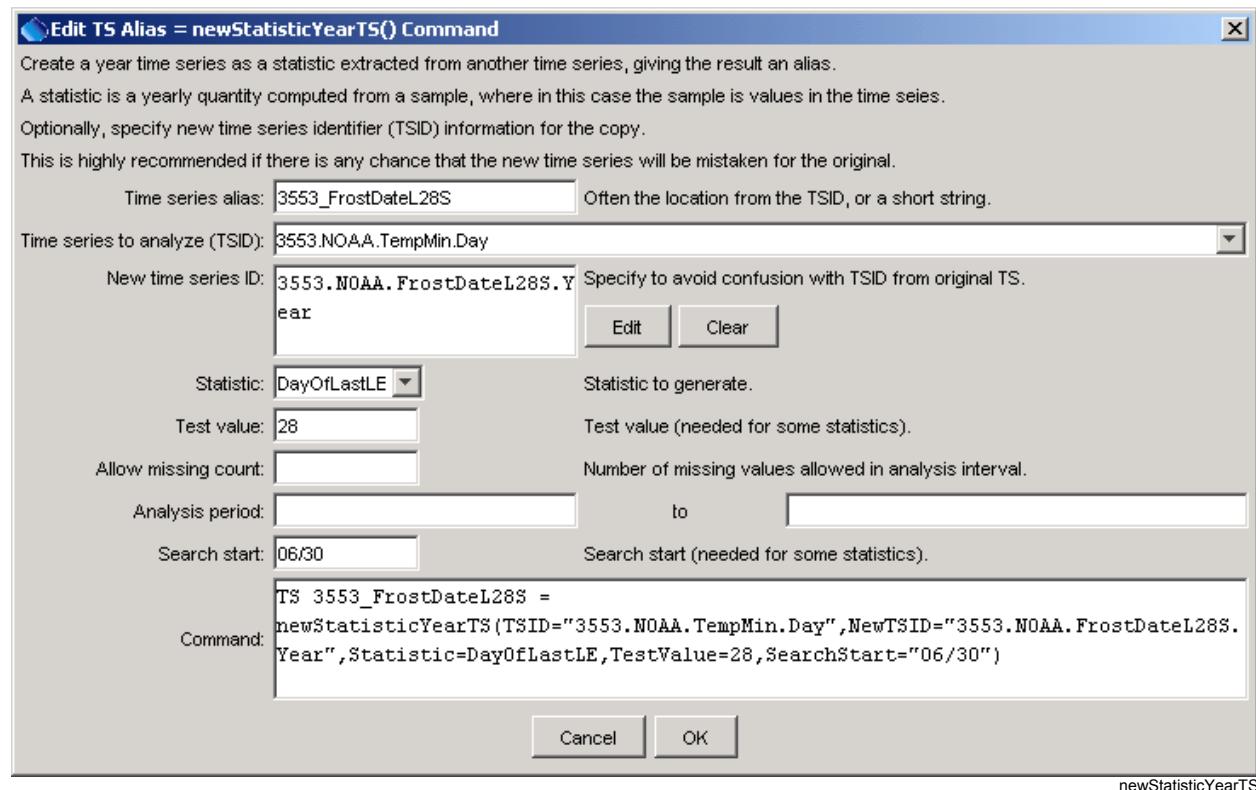
TS Alias = newStatisticYearTS()

Create a new yearly time series containing a statistic determined from another time series

Version 06.10.09, 2005-09-30, Color, Acrobat Distiller

The TS Alias = newStatisticYearTS() command processes a time series and creates a new yearly time series in memory, assigning the result an alias so that it can be more easily manipulated. Each yearly value in the resulting time series contains a statistic determined from the sample of points from the corresponding year in the original time series. For example, if the original time series has a daily time step, then the sample that is analyzed will be 365 or 366 values (depending on leap year). This command is useful because it operates on the raw time series data. Other commands (e.g., changeInterval()) can produce a similar result, for example converting a daily time series to monthly average). The newStatisticYearTS() command is being phased in, offering statistics that have been requested to meet a need. It is envisioned the list of statistics will increase over time and that additional optional parameters may be added (e.g., to indicate how to handle missing data in the sample).

The following dialog is used to edit the command and illustrates the syntax for the command.



TS Alias = newStatisticYearTS() Command Editor

newStatisticYearTS

The command syntax is as follows:

```
TS Alias = newStatisticYearTS(param=value,...)
```

Command Parameters

Parameter	Description	Default
Alias	The alias of the new time series, which can be used instead of the TSID in other commands.	None – must be specified.
TSID	The time series identifier (or alias) of the time series to analyze.	None – must be specified.
NewTSID	The time series identifier to be assigned to the new time series, which is useful to avoid confusion with the original time series.	Use the same identifier as the original time series, with an interval of Year and a data type matching the statistic.
Statistic	See the Available Statistics table below.	None – must be specified.
TestValue	A test value used when analyzing the statistic.	This parameter is required for some statistics and not used for others.
Allow Missing Count	The number of missing values allowed in the source interval(s) in order to produce a result. This capability should be used with care because it may result in data that are not representative of actual conditions.	0 – do not allow any missing data in the source data when computing a result.
TestStart	Specify the start in a year to apply the test, for example to limit the test to a season.	This parameter is planned for implementation in the future.
TestEnd	Specify the end in a year to apply the test, for example to limit the test to a season.	This parameter is planned for implementation in the future.
AnalysisStart	The date/time for the analysis start, using a precision that matches the original time series.	Analyze the full period.
AnalysisEnd	The date/time for the analysis start, using a precision that matches the original time series.	Analyze the full period.
SearchStart	The date/time to begin a data search when processing the statistic. For example, for the DayOfLastLE statistic applied to daily data, specify the MM/DD or MM-DD of the day to begin searching backwards in the year.	This parameter is optional for the DayOf statistics. By default, searches start on Jan 1 for forward searches and Dec 31 for backward searches.

Available Statistics

Statistic	Description	Limitations
CountOfGE	Count of values in a year \geq TestValue.	Analysis is limited to daily data.
CountOfGT	Count of values in a year $>$ TestValue.	Analysis is limited to daily data.
CountOfLE	Count of values in a year \leq TestValue.	Analysis is limited to daily data.
CountOfLT	Count of values in a year $<$ TestValue.	Analysis is limited to daily data.
DayOfFirstGE	Julian day of the year (1-366) for the first data value \geq TestValue. Searches start at the start of the year and move forward (SearchStart will reset the search start).	Analysis is limited to daily data.
DayOfFirstGT	Similar to DayOfFirstGE, for values $>$ TestValue.	Analysis is limited to daily data.
DayOfFirstLE	Similar to DayOfFirstGE, for values \leq TestValue.	Analysis is limited to daily data.
DayOfFirstLT	Similar to DayOfFirstGE, for values $<$ TestValue.	Analysis is limited to daily data.
DayOfLastGE	Julian day of the year (1-366) for the last data value \geq TestValue. Searches start at the end of the year and move backward (SearchStart will reset the search start).	Analysis is limited to daily data.
DayOfLastGT	Similar to DayOfLastGE, for values $>$ TestValue.	Analysis is limited to daily data.
DayOfLastLE	Similar to DayOfLastGE, for values \leq TestValue.	Analysis is limited to daily data.
DayOfLastLT	Similar to DayOfLastGE, for values $<$ TestValue.	Analysis is limited to daily data.
DayOfMax	Julian day of the year (1-366) for the maximum value in the time series.	Analysis is limited to daily data.
DayOfMin	Julian day of the year (1-366) for the minimum value in the time series.	Analysis is limited to daily data.
Max	Maximum value in a year.	Analysis is limited to daily data.
Min	Minimum value in a year.	Analysis is limited to daily data.

Handling of Missing Data at the End of the Period for DayOfXXXXxx Statistics with SearchStart

Day of year statistics that start the search on a given day in the year handle missing data at the start or end of the period as discussed in the following examples. For discussion purposes, SearchStart is June 30 for backward searches (to find the last occurrence in a year) and July 1 for forward searches (to find the first occurrence in a year).

Searching Forward with Gap at Start

	Jun 30		Jul 1		Aug 1	Dec 31	Jan 1		Jun 30		Jul 1		Dec 31
--	--------	--	-------	--	-------	--------	-------	--	--------	--	-------	--	--------

In the above case the period July 1 to July 31 would be treated as missing and the statistic would only be computed if AllowMissingCount is greater than the number of missing values in this initial period.

Searching Backward with Gap at Start

	Jun 30		Jul 1		Aug 1	Dec 31	Jan 1		Jun 30		Jul 1	Dec 31
--	--------	--	-------	--	-------	--------	-------	--	--------	--	-------	--------

In the above case the period on and before June 30 would be treated as missing and the statistic would not be computed in any case.

Searching Forward with Gap at End

Jul 1	Dec 31		Jan 1	Mar 15	Jun 30		Jul 1	
-------	--------	--	-------	--------	--------	--	-------	--

In the above case the period on and after July 1 would be treated as missing and the statistic would not be computed in any case.

Searching Backward with Gap at End

Jul 1	Dec 31		Jan 1	Mar 15	Jun 30		Jul 1	
-------	--------	--	-------	--------	--------	--	-------	--

In the above case the period between March 15 and June 30 would be treated as missing and the statistic would only be computed if `AllowMissingCount` is greater than the number of missing values in this initial period.

Examples

The following example commands file computes the last spring frost date for 28 degrees and 32 degrees, searching backwards from June 30 each year, and the first fall frost date for 32 and 28 degrees, searching forwards from July 1 each year:

```
startLog(LogFile="FrostDates_HydroBase.log")
setOutputPeriod(1950-01,2004-12)
# 3553 - GREELEY UNC
3553.NOAA.TempMin.Day~HydroBase
TS 3553_FrostDateL28S = newStatisticYearTS(TSID="3553.NOAA.TempMin.Day",
NewTSID="3553.NOAA.FrostDateL28S.Year",Statistic=DayOfLastLE,TestValue=28,
SearchStart="06/30")
TS 3553_FrostDateL32S = newStatisticYearTS(TSID="3553.NOAA.TempMin.Day",
NewTSID="3553.NOAA.FrostDateL32S.Year",Statistic=DayOfLastLE,TestValue=32,
SearchStart="06/30")
TS 3553_FrostDateF32F = newStatisticYearTS(TSID="3553.NOAA.TempMin.Day",
NewTSID="3553.NOAA.FrostDateF32F.Year",Statistic=DayOfFirstLE,TestValue=32,
SearchStart="07/01")
TS 3553_FrostDateF28F = newStatisticYearTS(TSID="3553.NOAA.TempMin.Day",
NewTSID="3553.NOAA.FrostDateF28F.Year",Statistic=DayOfFirstLE,TestValue=28,
SearchStart="07/01")
free(TSID="*.*.TempMin.*")
writeStateCU("Test.FrostDates")
```

Command Reference

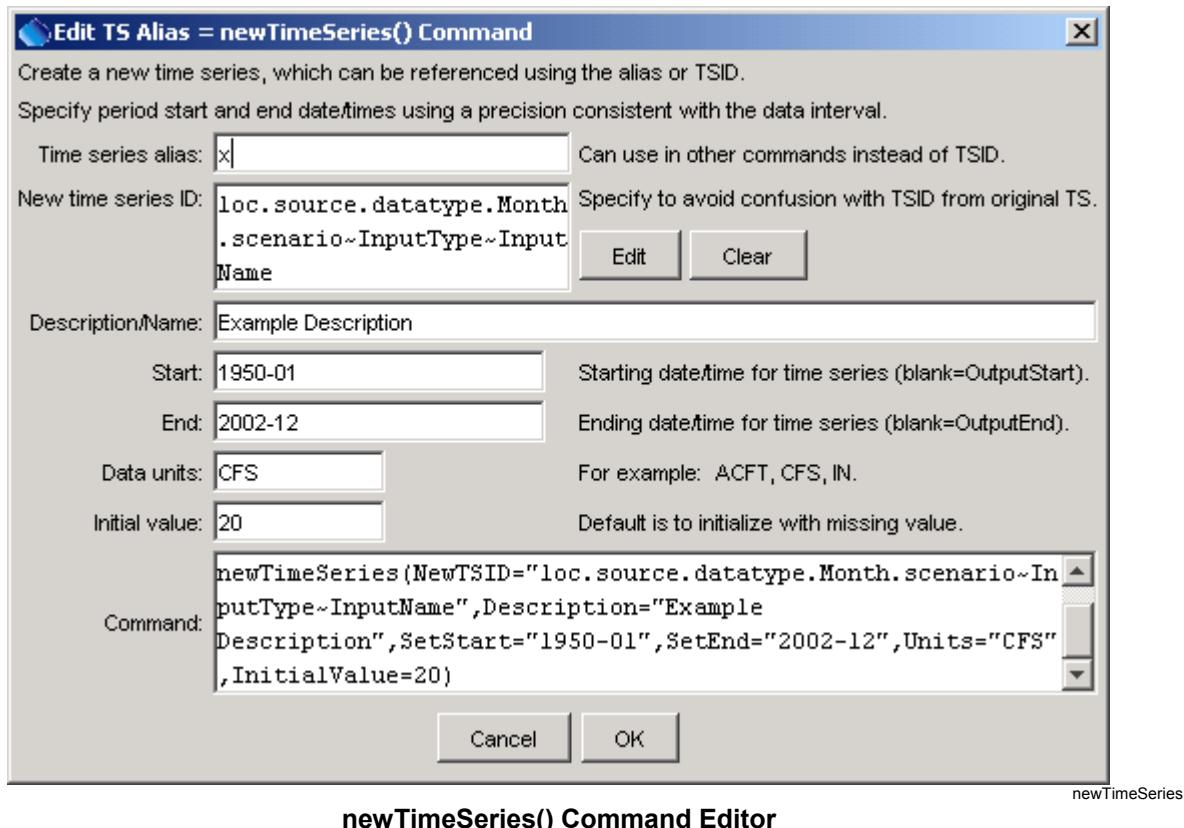
TS Alias = newTimeSeries()

Create a new time series

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The newTimeSeries() command creates a new time series in memory and assigns it an alias. This time series can then be manipulated (e.g., added to, filled). This command is useful, for example, to create a new time series to receive the results of a series of manipulations, rather than having the results accumulate in the first time series.

The following dialog is used to edit the command and illustrates the syntax for the command. The new time series identifier can be edited by pressing the **Edit** button.



newTimeSeries() Command Editor

newTimeSeries

The command syntax is as follows:

```
TS Alias = newTimeSeries(param=value,...)
```

Command Parameters

Parameter	Description	Default
Alias	The alias of the new time series, which can be used in place of the TSID in other commands.	None – must be specified.
NewTSID	The time series identifier of the new time series. The editor dialog formats the identifier from its parts.	None – must be specified with at least minimal information (location, data type).
Description	The description for the time series, used in output.	Blank.
SetStart	The start of the time series data period, or blank to use the output period defined with the <code>setOutputPeriod()</code> command.	Use the start from <code>setOutputPeriod()</code> .
SetEnd	The end of the time series data period, or blank to use the output period defined with the <code>setOutputPeriod()</code> command.	Use the end from <code>setOutputPeriod()</code> .
Units	Data units for the time series.	Blank.
InitialValue	The initial value to populate the time series.	Initialize the time series to missing data.

The example commands file shown below creates a new time series and initializes it to a constant of 20 CFS. Uncommenting the first command would allow the SetStart and SetEnd parameters to be removed from the `newTimeSeries()` command.

```
# setOutputPeriod(1950-01,2002-12)
TS x = newTimeSeries(
    NewTSID="loc.source.datatype.Month.scenario~InputType~InputName",
    Description="Example Description", SetStart="1950-01",
    SetEnd="2002-12", Units="CFS", InitialValue=20)
```

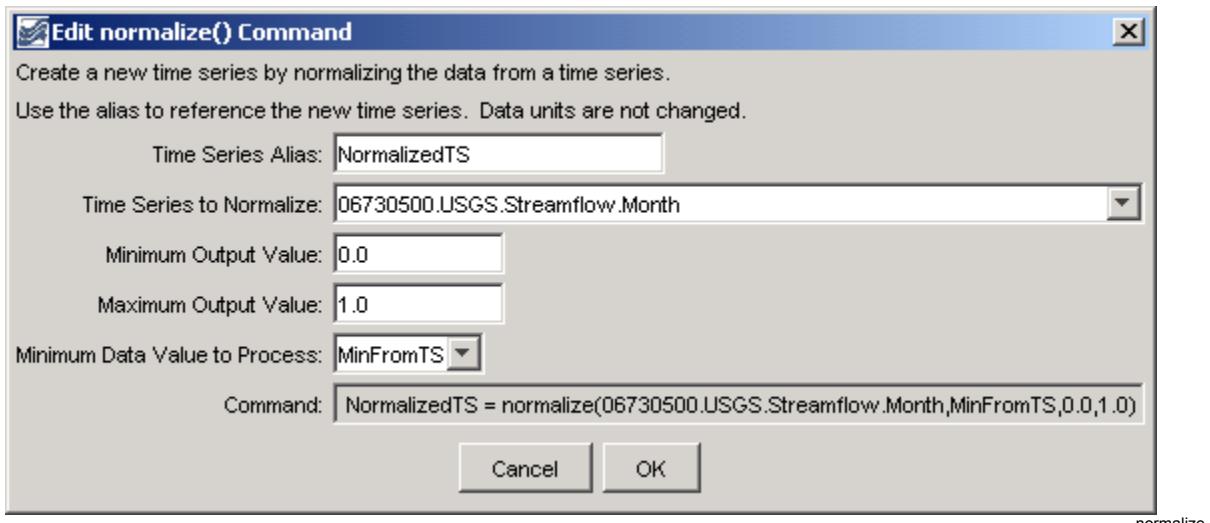
Command Reference

TS Alias = normalize()

Create a normalized time series

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

A `normalize()` command can be inserted to create a new normalized time series from an existing time series, assigning an alias to the result. Normalized time series are useful for analyzing trends and relationships and for allowing time series with different units to be plotted or analyzed together. The alias that is assigned to the time series can be referenced by other commands. The following dialog is used to edit the command and illustrates the syntax of the command.



normalize() Command Editor

The command syntax is as follows:

```
TS Alias = normalize(TSID,MinValue,MaxValue,MinValueHow)
```

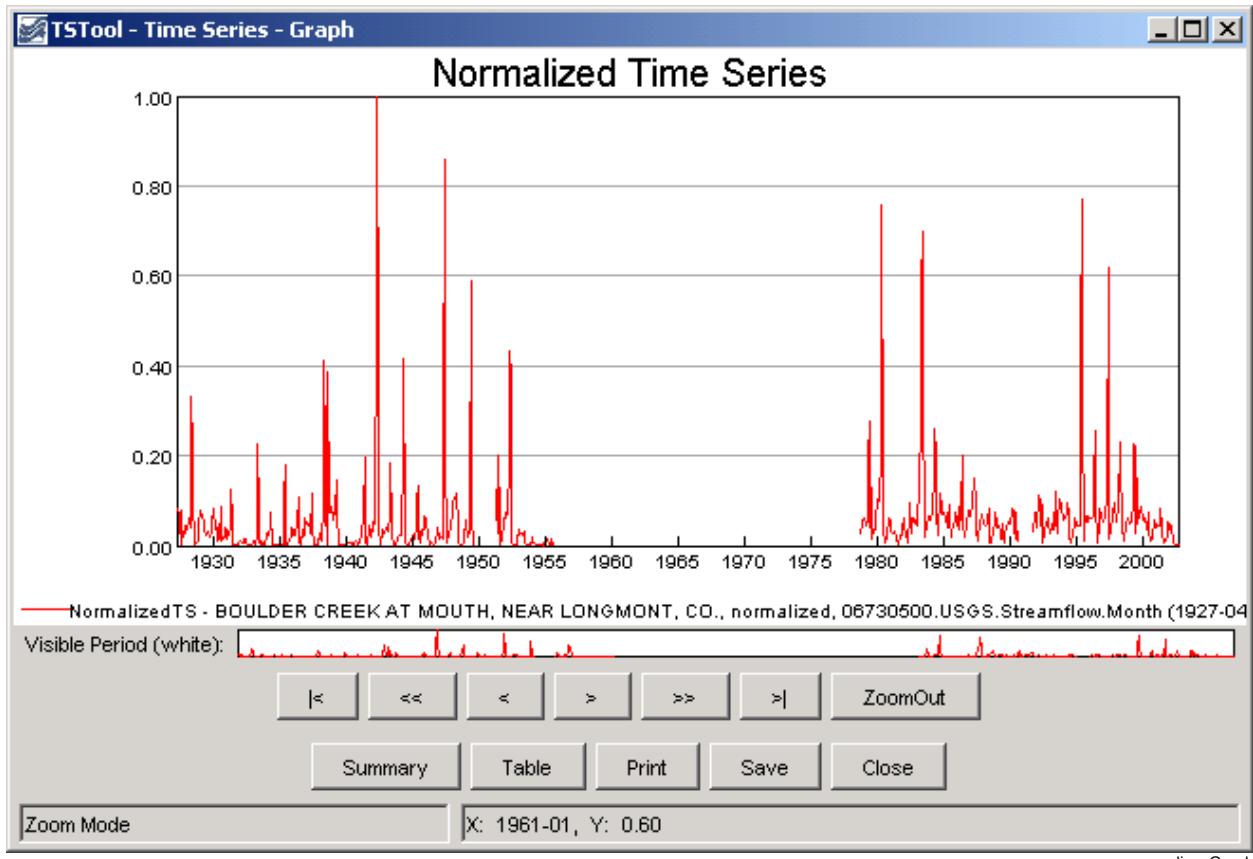
Command Parameters

Parameter	Description	Default
X	The alias for the new time series.	None – must be specified.
TSID	The time series identifier or alias for the time series to be normalized.	None – must be specified.
MinValue	The minimum normalized value.	None – must be specified.
MaxValue	The maximum normalized value.	None – must be specified.
MinValueHow	Indicates how to determine the minimum data value to process: <ul style="list-style-type: none">• MinFromTS – get the minimum value from the time series (typical)• MinZero – use zero (e.g., if negative values are to be ignored)	None – must be specified.

A sample commands file is as follows:

```
# 06730500 - BOULDER CREEK AT MOUTH, NEAR LONGMONT, CO.  
06730500.USGS.Streamflow.Month~HydroBase  
TS NormalizedTS = normalize(06730500.USGS.Streamflow.Month,MinFromTS,0.0,1.0)
```

The results are as follows:



Results of normalize() Command

Command Reference

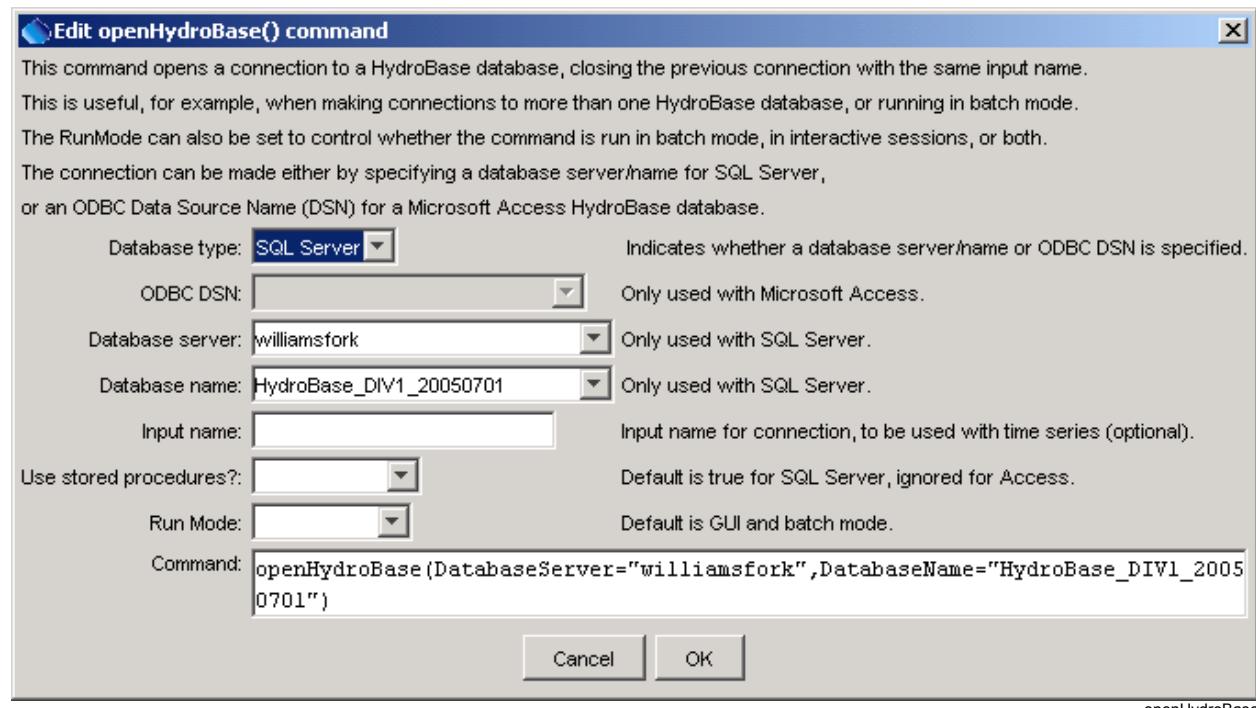
openHydroBase()

Open a connection to a HydroBase database

Version 06.14.00, 2005-12-16, Color, Acrobat Distiller

The `openHydroBase()` command opens a connection to a HydroBase database, allowing data to be read from the database (e.g., with `readHydroBase()` commands and time series identifiers that have `~HydroBase` input types). This command is not typically used for interactive sessions but may be inserted to run in batch only mode to allow a specific database and commands files to be distributed. It may also be used in cases where time series are read from different HydroBase databases, perhaps to compare the contents of the databases – in this case two `openHydroBase()` commands would be used. When connecting to a SQL Server database, a connection will be tried for MSDE and full SQL Server. If both fail, a warning will be shown.

The following dialog is used to edit this command and illustrates the command syntax. The **Database type** is used to control settings for parameters and is not itself a parameter.



openHydroBase() Command Editor

openHydroBase

The command syntax is as follows:

```
openHydroBase(param=value,...)
```

Command Parameters

Parameter	Description	Default
OdbcDsn	The ODBC DSN to use for the connection, used only when working with a Microsoft Access database.	Required if a Microsoft Access database is used.
DatabaseServer	Used with a SQL Server HydroBase. Specify the SQL Server database machine name. A list of choices will be shown, corresponding to properties in the <i>CDSS.cfg</i> file.	Required if a SQL Server database is used, and accepts the generic value DatabaseServer=local, which will automatically be translated to the name of the local computer.
DatabaseName	Used with a SQL Server HydroBase. The name of the database typically follows a pattern similar to: HydroBase_DIVn_YYYYMMDD or HydroBase_CO_East_YYYYMMDD. A list of choices will be shown, corresponding to properties in the <i>CDSS.cfg</i> file.	HydroBase
InputName	The input name corresponding to the ~InputType~InputName information in time series identifiers. This is used when more than one HydroBase connection is used in the same commands file.	Blank (no input name).
UseStoredProcedures	Used with SQL Server, indicating whether stored procedures are used. Stored procedures are being phased in and will be the only option at some point.	True (used stored procedures).
RunMode	Indicates when the command should be run, one of: BatchOnly – run the command only in batch mode. GUIOnly – run the command only in GUI mode. GUIAndBatch – run the command in batch and GUI mode.	GUIAndBatch

The following example commands file illustrates how to connect to a SQL Server database running on a machine named “hbserver”:

```
openHydroBase(DatabaseServer="hbserver",DatabaseName="HydroBase_CO_20050501")
```

The following example commands file illustrates how to connect to a SQL Server database running on the local machine.

```
openHydroBase(DatabaseServer="local",DatabaseName="HydroBase_CO_20050501")
```

The following example commands file illustrates how to make two HydroBase database connections, in this case to test whether the stored procedure and SQL queries return the same results (the `InputName` parameter is used to tell TSTool which connection to use when reading data based on time series identifiers):

```
openHydroBase(DatabaseServer="hbserver",InputName="SP",  
UseStoredProcedures=True,RunMode=GUIAndBatch)  
openHydroBase(DatabaseServer="hbserver",RunMode=GUIAndBatch,  
UseStoredProcedures=False,InputName="NoSP")  
TS ts_sp =  
readHydroBase(TSID="BOXHUCO.DWR.Streamflow.Month~HydroBase~SP")  
TS ts_nosp =  
readHydroBase(TSID="BOXHUCO.DWR.Streamflow.Month~HydroBase~NoSP")
```

The following example commands file illustrates how to connect to a Microsoft Access database:

```
openHydroBase(RunMode=GUIAndBatch,OdbcDsn="HydroBase_DIV1_20030701")
```

This page is intentionally blank.

Command Reference

processTSPProduct()

Process a time series product file to produce output

Version 06.11.00, 2005-10-25, Color, Acrobat Distiller

The processTSPProduct() command is used to streamline creation of time series data products. These products are described in time series product description (*.tsp) files, which are typically created by using the **Save... Time Series Product** choice in graph windows (future enhancements may allow creation of text products from summary or table views). See the **TSView Time Series Viewing Tools** appendix for more information about time series products. For example, the following sequence of actions can be used to define and use time series product description files:

1. Use TSTool and interactively select time series using the main window. The time series identifiers and/or aliases will be referenced in the time series product.
2. Interactively view a graph (e.g., **Results...Graph - Line**) and edit its properties by right-clicking on the graph and selecting the **Properties** choice (e.g., set titles and legend properties).
3. Save the graph as a time series product from the graph window using the **Save... Time Series Product** choice. Typically the product is saved in a location close to the commands file. An example time series product file is as follows:

```
[Product]

ProductType = "Graph"

[SubProduct 1]

GraphType = "Line"
MainTitleString = "Streamflow (Monthly Total)"

[Data 1.1]

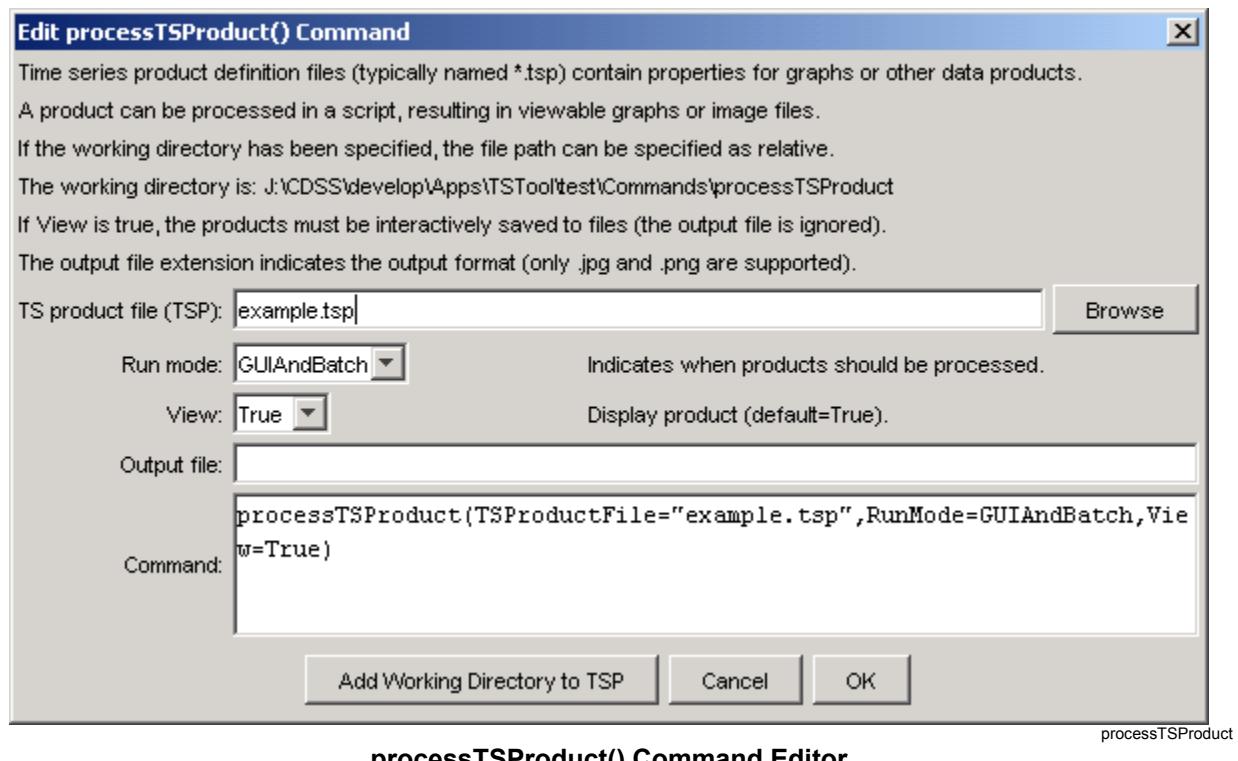
TSID = "08223000.DWR.Streamflow.Month~HydroBase"

[Data 1.2]

TSID = "08220500.DWR.Streamflow.Month~HydroBase"
```

4. Add a processTSPProduct() command to the original commands to allow the product to be created automatically. Select the time series product file created in the previous step.
5. Save the commands in a file (e.g., named *stations.TSTool*) so that they can be run again. The commands file and time series product definition files must be used consistently (e.g., the time series identifiers and directory paths must be consistent).

The following dialog is used to edit the processTSPProduct() command and illustrates the command syntax. The path to the file can be absolute or relative to the working directory. The **Browse** button can be used to select the time series product description file (if a relative path is desired, delete the leading path after the select or use the **Remove Working Directory from TSP** button).



processTSPProduct

processTSPProduct() Command Editor

The command syntax is as follows:

```
processTSProduct(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSProductFile	The time series product file to process. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to write (if a relative path is desired, delete the leading path after the select).	None – must be specified.
RunMode	Indicate the run mode to process the product, one of: <ul style="list-style-type: none"> • BatchOnly – indicates that the product should only be processed in batch mode. • GUIOnly – indicates that the product should only be processed when the TSTool GUI is used (useful when Preview is set to Preview). • GUIAndBatch – indicates that the product should be processed in batch and GUI mode. 	None – must be specified.
View	Indicates whether the output should be previewed interactively, one of: <ul style="list-style-type: none"> • True – display the graph. • False – do not display the graph (specify the output file instead to automate image creation). 	None – must be specified.
OutputFile	The absolute or relative path to an output file. Use this parameter with View=False to automate image processing. If the filename ends in “jpg”, a JPEG image file will be produced. If the filename ends in “png”, a PNG file will be produced.	If specified, an image file will be created.

A sample commands file is as follows:

```
processTSProduct(TSProductFile="example.tsp", RunMode=GUIAndBatch, View=True)
```

After using the above dialog to edit the command, the time series product can be processed from TSTool as follows:

1. Open the commands file, in this case containing the above commands file.

2. Process the commands using ***Run All Commands***. The graph will be displayed for review.

Alternatively, run TSTool in batch mode by specifying an output file (and optionally changing the RunMode parameter to BatchOnly) using:

```
tstool -commands commands.TSTool
```

The working directory will be set to the directory for the commands file and output will be relative to that directory.

Command Reference

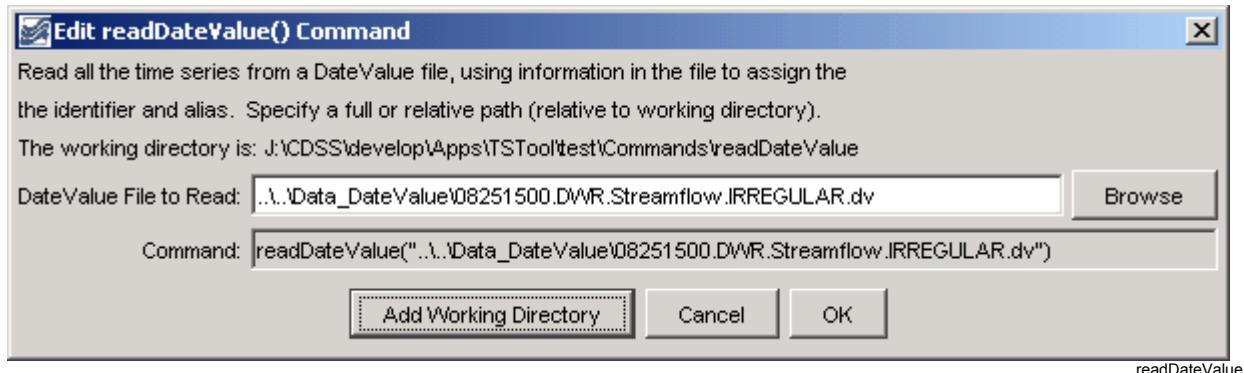
readDateValue()

Read all time series from a DateValue File

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The `readDateValue()` command reads all the time series in a DateValue file into memory (see the **DateValue Input Type** appendix). The actual reading occurs as the commands are being processed. Therefore, if any other commands reference the DateValue time series, time series identifiers will need to be specified manually (they are not available to list in dialogs like other time series specified with identifiers) or use wildcards in identifiers.

The following dialog is used to edit the command and illustrates the syntax for the command. The path to the file can be absolute or relative to the working directory. The **Browse** button can be used to select the file to read (if a relative path is desired, delete the leading path after the select). Use the `TS Alias = readDateValue()` command to read a single time series from a DateValue file.



readDateValue() Command Editor

The command syntax is as follows:

```
readDateValue( inputFile )
```

Command Parameters

Parameter	Description	Default
InputFile	The name of the DateValue input file to read, surrounded by double quotes.	None – must be specified.

A sample commands file is as follows:

```
readDateValue("...\\Data_DateValue\\08251500.DWR.Streamflow.IRREGULAR.dv")
```

This page is intentionally blank.

Command Reference

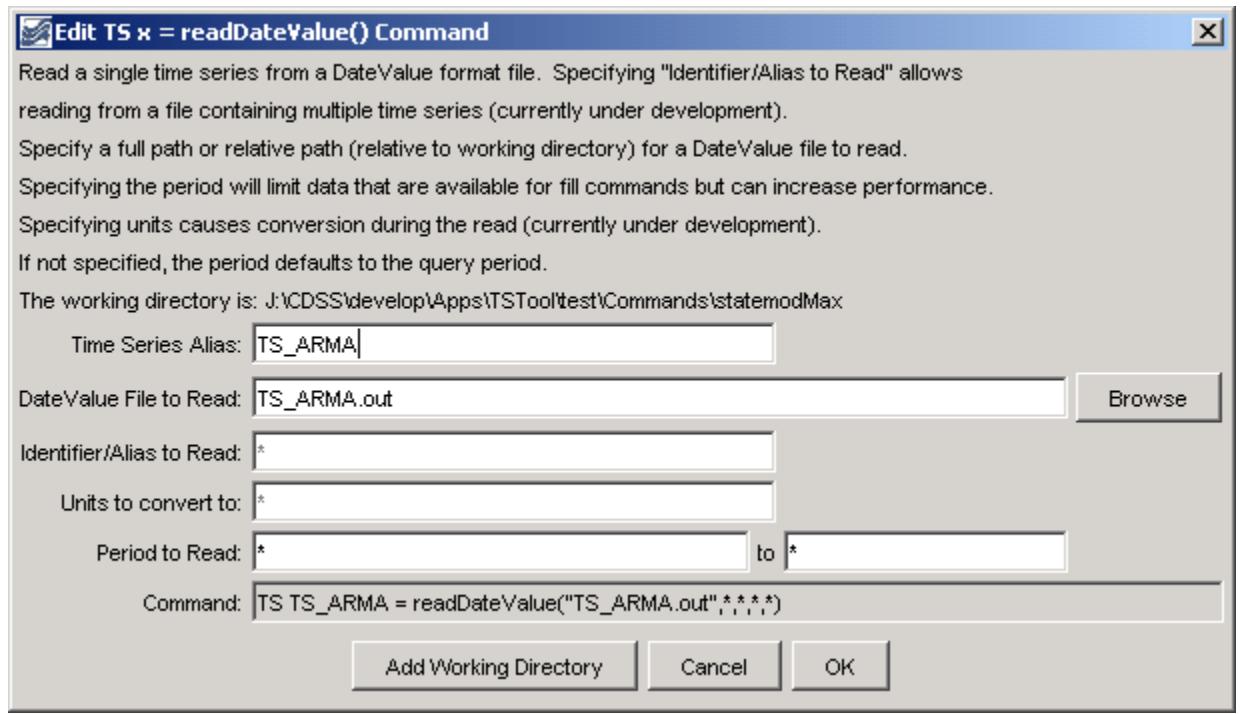
TS X = readDateValue()

Read a single time series from a DateValue File

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The TS Alias = readDateValue() command reads a single time series from a DateValue file (see the **DateValue Input Type** appendix) and assigns an alias to the result. This command should not be confused with the readDateValue() command that does not use the alias, which reads all time series in a DateValue file. Currently the file being read **must contain only one time series**.

The following dialog is used to edit the command and illustrates the syntax.



TS_readDateValue

TS Alias = readDateValue() Command Editor

The command syntax is as follows:

```
TS X = readDateValue(InputFile,TSID,NewUnits,InputStart,InputEnd)
```

Command Parameters

Parameter	Description	Default
Alias	Alias for the new time series that is read from the file, which can be used instead of the TSID in other commands.	None – must be specified.
InputFile	The name of the DateValue input file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
TSID	A time series identifier pattern to filter the read – this parameter is currently not used. Therefore the DateValue file should contain only one time series.	Currently ignored.
NewUnits	The new units for the time series. The data values will be converted to these units.	Use the units read from the file.
InputStart	The start of the period to read data – specify if the period should be different from the global query period.	Use the global query period.
InputEnd	The end of the period to read data – specify if the period should be different from the global query period.	Use the global query period.

A sample commands file is as follows:

```
TS TS_ARMA = readDateValue("TS_ARMA.out",*,*,*,*)
```

Command Reference

readHydroBase()

Read time series from a HydroBase database

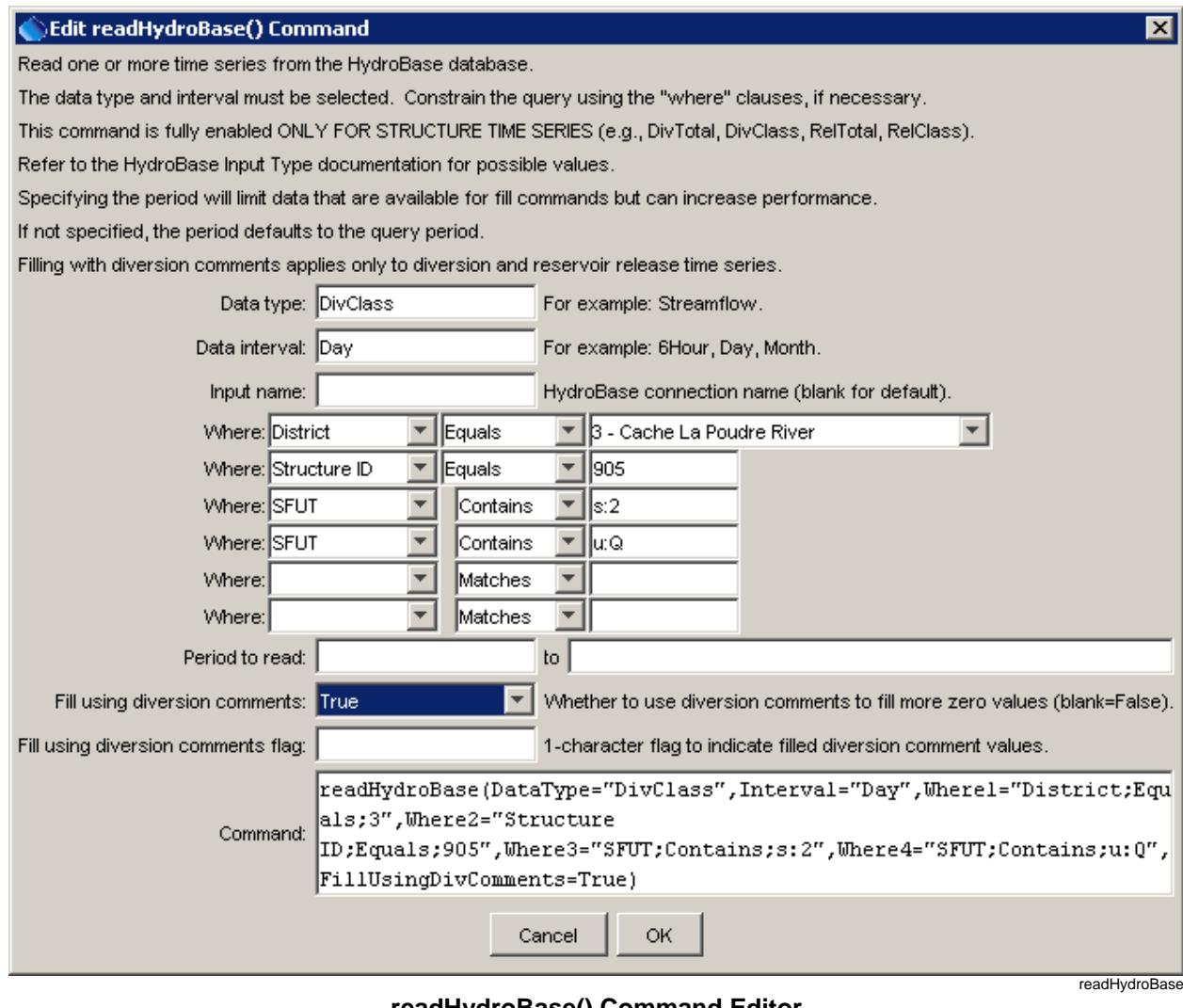
Version 06.17.00, 2006-04-28, Color, Acrobat Distiller

The `readHydroBase()` command reads one or more time series from the HydroBase database (see the **HydroBase Input Type** appendix). The actual reading occurs as the commands are being processed. Therefore, if any other commands reference the HydroBase time series, the time series identifiers must be specified manually or use wildcards in identifiers (identifiers are not available to list in dialogs like other time series specified explicitly with time series identifiers).

The following special actions occur, depending on data type:

1. Daily diversion (`DivTotal` and `DivClass`) and reservoir release (`RelTotal` and `RelClass`) time series have their values automatically carried forward to fill data within irrigation years (Nov to Oct). HydroBase only stores full months of data when non-zero observations or non-zero filled values occur in a month. Therefore, this filling action should only provide additional zero values. Irrigation years with no observations remain as missing after the read. See the `fillHistMonthAverage()` command, which is often used to fill completely missing years.
2. Daily, monthly, and yearly diversion and reservoir release time series can optionally be filled using diversion comments, which indicate when irrigation years should be treated as missing. See the `FillUsingDivComments` parameter below. Note that diversion comments should not conflict with more detailed records but and provide additional information. The older `fillUsingDivComments()` command is also available for filling.

The following dialog is used to edit the command and illustrates the syntax for the command.



readHydroBase() Command Editor

The **Data type**, **Data interval**, and **Where** input fields are similar to those from the main TSTool interface. However, whereas the interactive interface first requires a query to find the matching time series list and then an interactive select for specific time series identifiers, the `readHydroBase()` command reads in one step the time series list and the corresponding data for the time series. This can greatly shorten commands files and simplify command logic, especially when processing large amounts of data.

Currently the **Data type** and **Data interval** must be entered manually (drop-down choices are not available), according to the **HydroBase Input Type** appendix. Currently, only the structure data types (in particular diversions) are supported in the above dialog and have been tested. Support for other data types is under development.

The command syntax is as follows:

```
readHydroBase (param=value,...)
```

Command Parameters

Parameter	Description	Default
DataType	<p>The data type to be queried, as documented in the HydroBase Input Type appendix. The following conditions apply:</p> <ul style="list-style-type: none"> ▪ For diversions, use DivClass without the SFUT sub-type. The SFUT sub-type will be added after data are queried. ▪ For reservoir releases, use RelClass without the SFUT sub-type. The SFUT sub-type will be added after data are queried. 	None – must be specified.
Interval	The data interval for the time series, as documented in the HydroBase Input Type appendix (e.g., Day, Month, Year).	None – must be specified.
InputName	The HydroBase database connection input name to use for the connection, as initialized in <code>openHydroBase ()</code> , which allows reading from more than one HydroBase in the same commands file.	Use the default HydroBase connection.
WhereN	<p>The “where” clauses to be applied when querying data, matching the values in the Where fields in the command editor dialog and the TSTool main interface. The parameters should be named Where1, Where2, etc., with a gap resulting in the remaining items being ignored. The format of each value is:</p> <p>“Item;Operator;Value”</p> <p>Where Item indicates a data field to be filtered on, Operator is the type of constraint, and Value is the value to be checked when querying.</p> <p>Warning: Currently the <code>>=</code> and <code><=</code> operators will produce errors – this issue is being evaluated. Work around by using the Less Than and Greater Than operators with appropriate Value.</p>	If not specified, the query will not be limited and very large numbers of time series may be queried.

Command Parameters (continued)

Parameter	Description	Default
InputStart	Start of the period to query, specified as a date/time with a precision that matches the requested data interval.	Read all available data.
InputEnd	End of the period to query, specified as a date/time with a precision that matches the requested data interval.	Read all available data.
FillUsingDivComments	Indicate whether to fill diversion and reservoir release time series using diversion comments.	False
FillUsingDivCommentsFlag	If specified as a single character, data flags will be enabled for the time series and each filled value will be tagged with the specified character. The flag can then be used later to label graphs, etc. The flag will be appended to existing flags if necessary.	No flag is assigned.

A sample commands file is as follows (read all reservoir releases to structure 0300905):

```
readHydroBase(DataType="DivClass", Interval="Day",
Where1="District; Equals; 3",
Where2="Structure ID; Equals; 905", Where3="SFUT; Contains; s:2" )
```

Command Reference

TS Alias = readHydroBase()

Read a single time series from a HydroBase Database

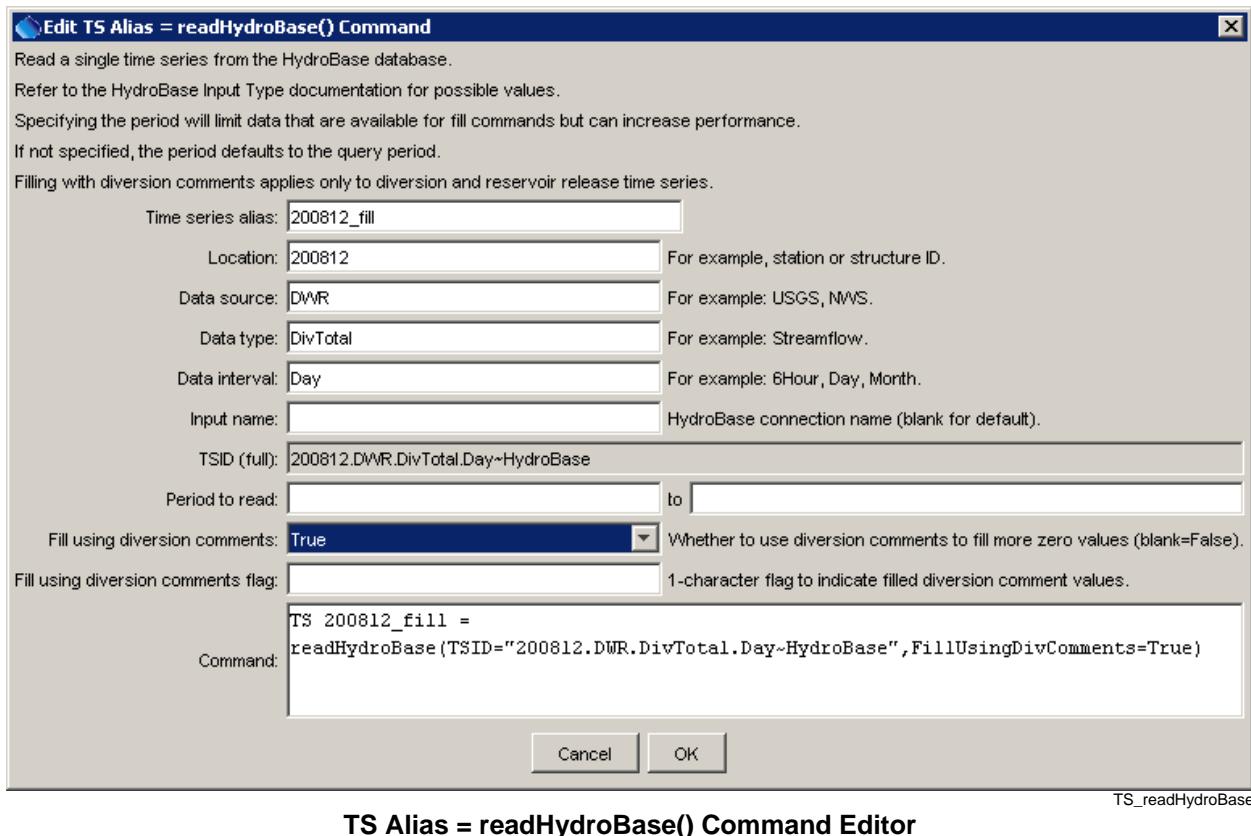
Version 06.17.00, 2006-04-28, Color, Acrobat Distiller

The `TS Alias = readHydroBase()` command reads a single time series from a HydroBase database (see the **HydroBase Input Type** appendix) and assigns an alias to the result. This command should not be confused with the `readHydroBase()` command that does not use the alias, which reads one or more matching time series from a HydroBase database.

The following special actions occur, depending on data type:

1. Daily diversion (`DivTotal` and `DivClass`) and reservoir release (`RelTotal` and `RelClass`) time series have their values automatically carried forward to fill data within irrigation years (Nov to Oct). HydroBase only stores full months of data when non-zero observations or non-zero filled values occur in a month. Therefore, this filling action should only provide additional zero values. Irrigation years with no observations remain as missing after the read. See the `fillHistMonthAverage()` command, which is often used to fill completely missing years.
2. Daily, monthly, and yearly diversion and reservoir release time series can optionally be filled using diversion comments, which indicate when irrigation years should be treated as missing. See the `FillUsingDivComments` parameter below. Note that diversion comments should not conflict with more detailed records but and provide additional information. The older `fillUsingDivComments()` command is also available for filling.

The following dialog is used to edit the command and illustrates the syntax.



TS_readHydroBase

TS Alias = readHydroBase() Command Editor

The command syntax is as follows:

```
TS Alias = readHydroBase(param=value...)
```

Command Parameters

Parameter	Description	Default
Alias	Alias for the new time series that is read from the file.	None – must be specified.
TSID	A time series identifier to read – see the HydroBase Input Type appendix.	Currently ignored.
InputName	The HydroBase database connection input name to use for the connection, as initialized in <code>openHydroBase()</code> , which allows reading from more than one HydroBase in the same commands file.	Use the default HydroBase connection.
InputStart	The start of the period to read data – specify if the period should be different from the global input period.	Use the global input period.
InputEnd	The end of the period to read data – specify if the period should be different from the global input period.	Use the global input period.
FillUsingDivComments	Indicate whether to fill diversion and reservoir release time series using diversion comments.	False
FillUsingDivCommentsFlag	If specified as a single character, data flags will be enabled for the time series and each filled value will be tagged with the specified character. The flag can then be used later to label graphs, etc. The flag will be appended to existing flags if necessary.	No flag is assigned.

A sample commands file is as follows:

```
TS NorthPoudreDiv = readHydroBase(TSID="0300905.DWR.DivTotal.Day~HydroBase")
```

This page is intentionally blank.

Command Reference

readNwsCard()

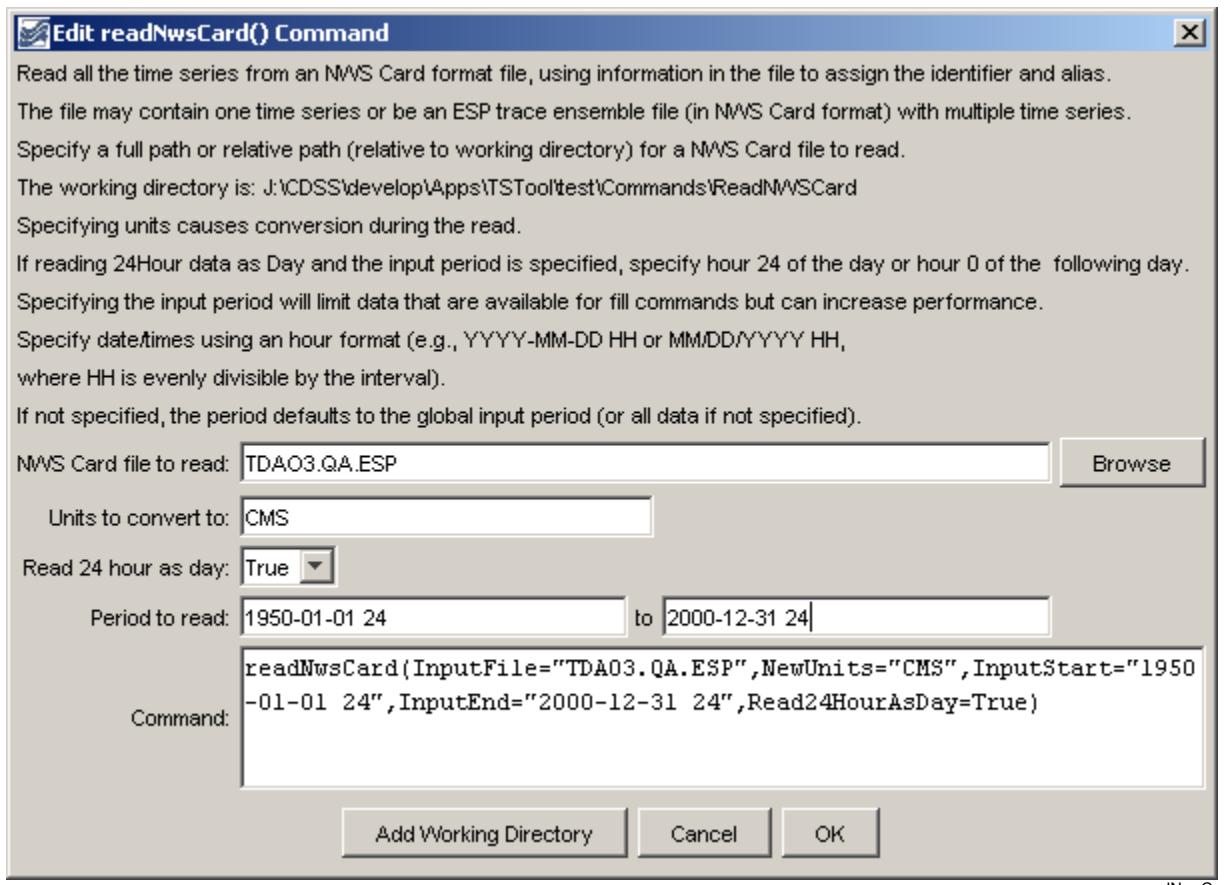
Read all time series from an NWS CARD file

Version 06.14.00, 2006-01-05, Color, Acrobat Distiller

The `readNwsCard()` command reads all the time series in an NWS CARD file into memory (see the **NWSCard Input Type** appendix). The actual reading occurs as the commands are being processed. Therefore, if any other commands reference the NWS Card time series, time series identifiers will need to be specified manually (they are not available to list in dialogs like other time series specified with identifiers) or use wildcards in identifiers. This command can be used to read the single time series format or trace format file.

If a trace file is read, each trace will be identified by the historical year for the start of the trace, and will be available as a time series for other commands.

The following dialog is used to edit the command and illustrates the syntax for the command. The path to the file can be absolute or relative to the working directory. The **Browse** button can be used to select the file to read (if a relative path is desired, delete the leading path after the select). Use the `TS Alias = readNwsCard()` command to read a single time series from an NWS Card file.



readNwsCard() Command Editor

readNwsCard

The command syntax is as follows:

```
readNwsCard(param=value,...)
```

Command Parameters

Parameter	Description	Default
InputFile	The name of the NWS Card input file to read, surrounded by double quotes.	None – must be specified.
NewUnits	The units to convert to after the read.	Do not convert the units.
Read24HourAsDay	If True, read 24Hour time series as if the data were Day interval. Because NWS Card format uses hours 1 to 24, treating as 24Hour results in values being saved at hour zero of the next day. Reading as Day interval causes the values to be stored without the shift.	False – read as hourly and shift data at hour 24 to zero of the next day.
InputStart	The start of the period to read – specify if the read period should be different from the global query period. If Read24HourAsDay=True, specify the period using either hour 24 of the start day, or hour 0 of the next day. This parameter must be specified to hour precision with hour's aligning with the file's data.	Use the global input period or if not specified read all the data in the file.
InputEnd	The end of the period to read – specify if the read period should be different from the global query period. If Read24HourAsDay=True, specify the period using either hour 24 of the start day, or hour 0 of the next day. This parameter must be specified to hour precision	Use the global input period or if not specified read all the data in the file.

The following example command reads a file for the specified period, converts the data (from CFS) to CMS, and reads 24Hour data as Day:

```
readNwsCard(InputFile="TDAO3.QA.ESP", NewUnits="CMS", InputStart="1950-01-01 24", InputEnd="2000-12-31 24", Read24HourAsDay=True)
```

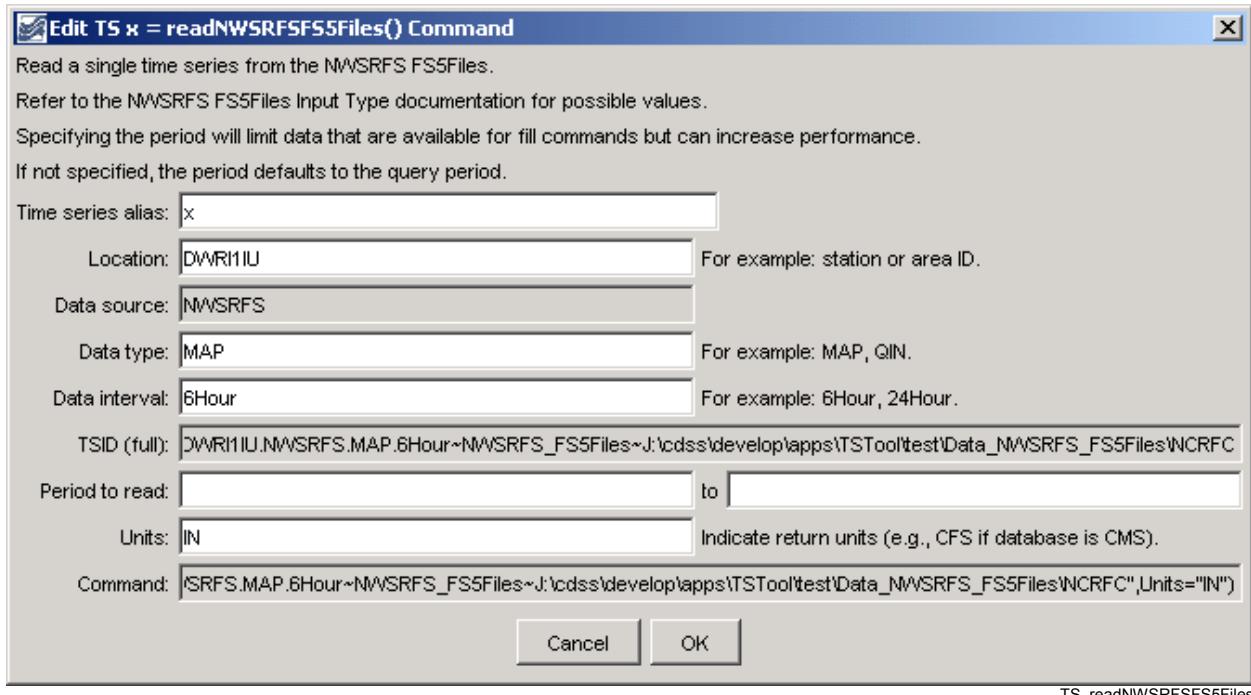
Command Reference

TS X = readNWSRFSFS5Files() Read a single time series from an NWSRFS FS5Files database

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The TS Alias = readNWSRFSFS5Files() command reads a single time series from an NWSRFS FS5Files database (see the **NWSRFS FS5Files Input Type** appendix) and assigns an alias to the result.

The following dialog is used to edit the command and illustrates the syntax.



TS_readNWSRFSFS5Files

TS Alias = readNWSRFSFS5Files() Command Editor

The command syntax is as follows:

```
TS Alias = readNWSRFSFS5Files (param=value...)
```

Command Parameters

Parameter	Description	Default
Alias	Alias for the new time series that is read from the file, which can be used instead of the TSID in other commands.	None – must be specified.
TSID	A time series identifier to read – see the NWSRFSFS5Files Input Type appendix. The input name for the time series identifier will agree with how the NWSRFS FS5Files were specified. For example if Apps Defaults are used, the input name will be omitted. If a directory is specified, it can be an absolute path or can be relative to the working directory.	Currently ignored.
InputStart	The start of the period to read data – specify if the period should be different from the global query period.	Use the global query period.
InputEnd	The end of the period to read data – specify if the period should be different from the global query period.	Use the global query period.
Units	The data units to be returned. The units must be specified compatible with the database units to allow for the conversion.	Use the database units, which are typically SI units.

A sample commands file is as follows:

```
TS x = readNWSRFSFS5Files(TSID="DWRI1IU.NWSRFS.MAP.6Hour~NWSRFS_FS5Files~J:\CDSS\develop\Apps\TSTool\test\Data_NWSRFS_FS5Files\BPA",Units="IN")
```

Command Reference

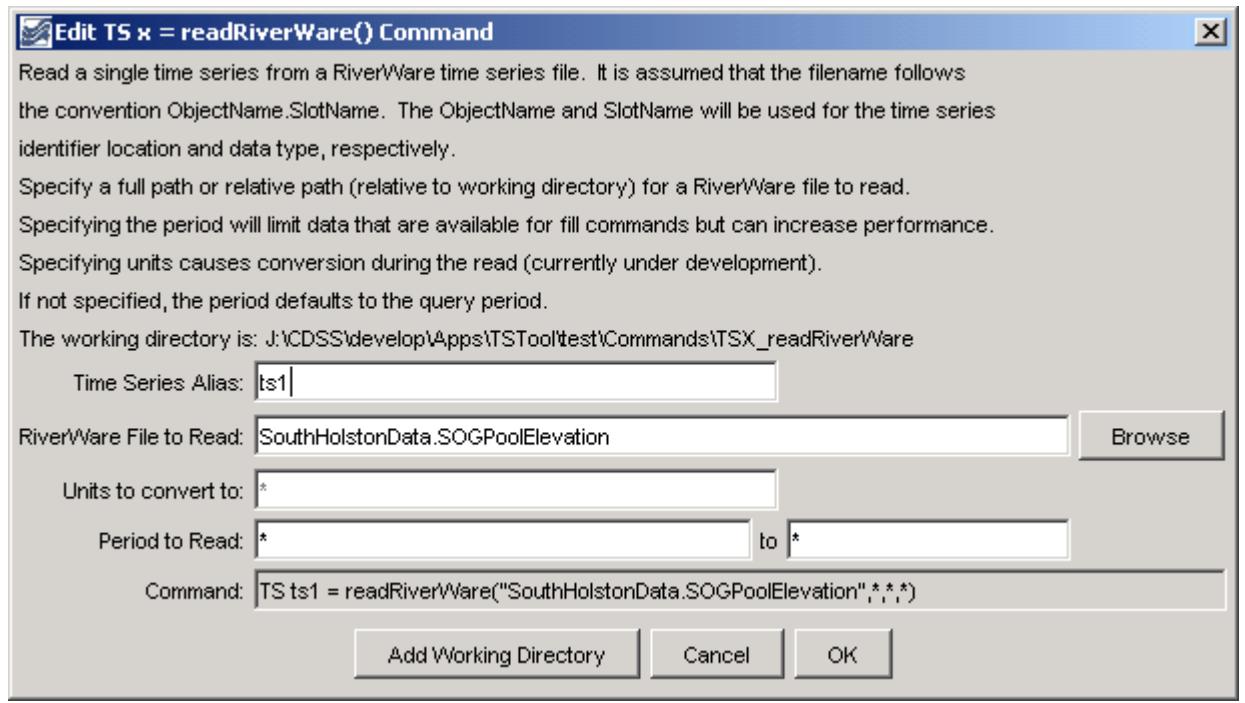
TS Alias = readRiverWare()

Read a single time series from a RiverWare file

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The TS Alias = readRiverWare() command reads a single time series from a RiverWare file (see the **RiverWare Input Type** appendix) and assigns an alias to the result.

The following dialog is used to edit the command and illustrates the command syntax.



TS Alias = readRiverWare() Command Editor

TS_readRiverWare

The command syntax is as follows:

```
TS Alias = readRiverWare(InputFile,NewUnits,InputStart,InputEnd)
```

Command Parameters

Parameter	Description	Default
Alias	Alias for the new time series that is read from the file, which can be used instead of the TSID in other commands.	None – must be specified.
InputFile	The name of the RiverWare file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
NewUnits	The new units for the time series. The data values will be converted to these units.	Use the units read from the file.
InputStart	The start of the period to read data – specify if the period should be different from the global query period.	Use the global query period.
InputEnd	The end of the period to read data – specify if the period should be different from the global query period.	Use the global query period.

A sample commands file is as follows:

TS ts1 = readRiverWare("SouthHolstonData.SOGPoolElevation", "QIN.txt", *, *, *)

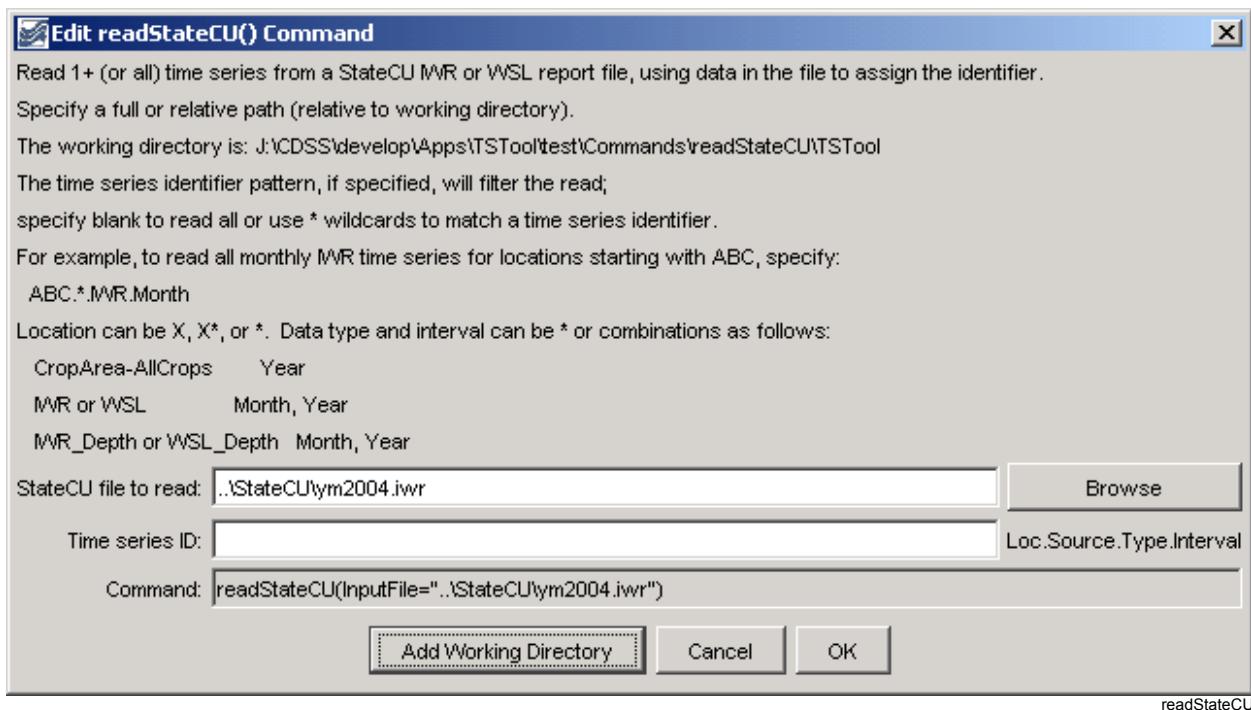
Command Reference

readStateCU()

Read all the Time Series from a StateCU Time Series or Report File

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `readStateCU()` command reads all the time series in a StateCU time series file (e.g., frost dates) or report file (e.g., IWR, WSL) into memory (see the **StateCU Input Type** appendix). The actual reading occurs as the commands are being processed. Therefore, if any other commands reference the StateCU time series, the time series identifiers must be specified manually or use wildcards in identifiers (identifiers are not available to list in dialogs like other time series specified with identifiers). The following dialog is used to edit the command and illustrates the syntax for the command.



readStateCU() Command Editor

The command syntax is as follows:

```
readStateCU(param=value,...)
```

Command Parameters

Parameter	Description	Default
InputFile	The name of the StateCU time series or report file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
TSID	A time series identifier pattern that will be used to filter the list of time series that are read. See the figure above for examples.	Read all time series.

A sample commands file is as follows:

```
readStateCU(..\StateCU\ym2004.iwr)
```

Command Reference

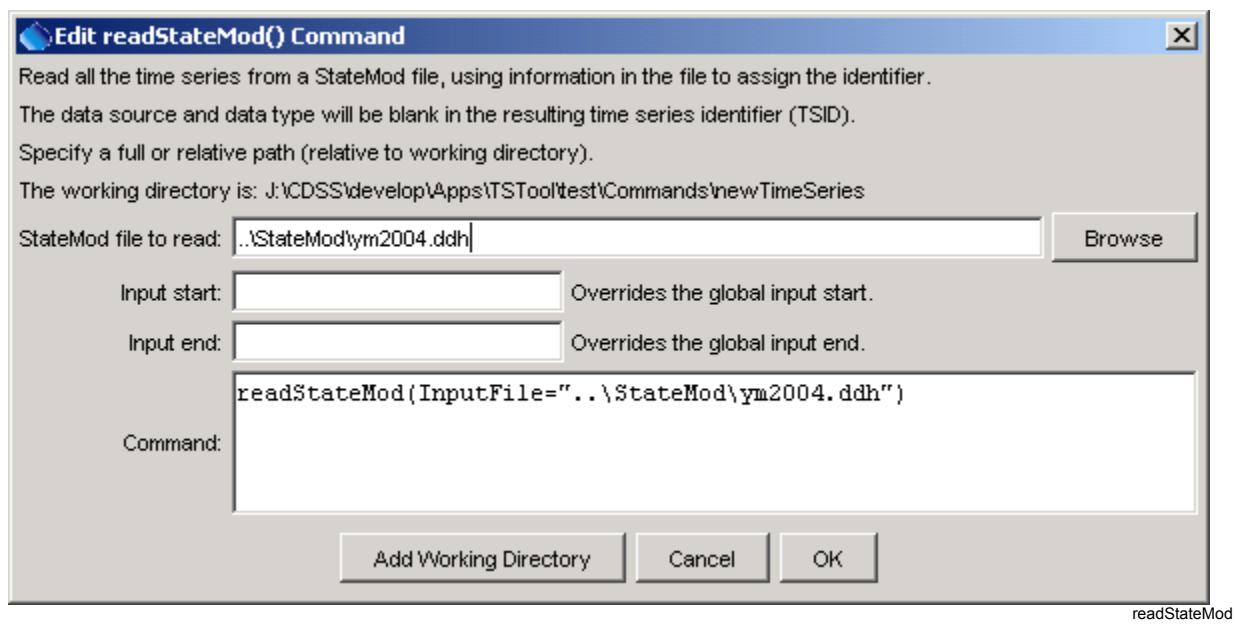
readStateMod()

Read all the time series from a StateMod time series file

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The `readStateMod()` command reads all the time series in a StateMod time series file into memory (see the **StateMod Input Type** appendix). The actual reading occurs as the commands are being processed. Therefore, if any other commands reference the StateMod time series, the time series identifiers must be specified manually or use wildcards in identifiers (identifiers are not available to list in dialogs like other time series specified with identifiers). Time series that are read are added to the list of time series (they do not replace time series with matching identifiers).

The following dialog is used to edit the command and illustrates the syntax for the command.



The command syntax is as follows:

```
readStateMod(param=value,...)
```

Command Parameters

Parameter	Description	Default
InputFile	The name of the StateMod time series file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
InputStart	The start of the period to read data – specify if the period should be different from the global query period.	Use the global query period.
InputEnd	The end of the period to read data – specify if the period should be different from the global query period.	Use the global query period.

A sample commands file is as follows:

```
readStateMod(InputFile=".\\StateMod\\ym2004.ddh")
```

Command Reference

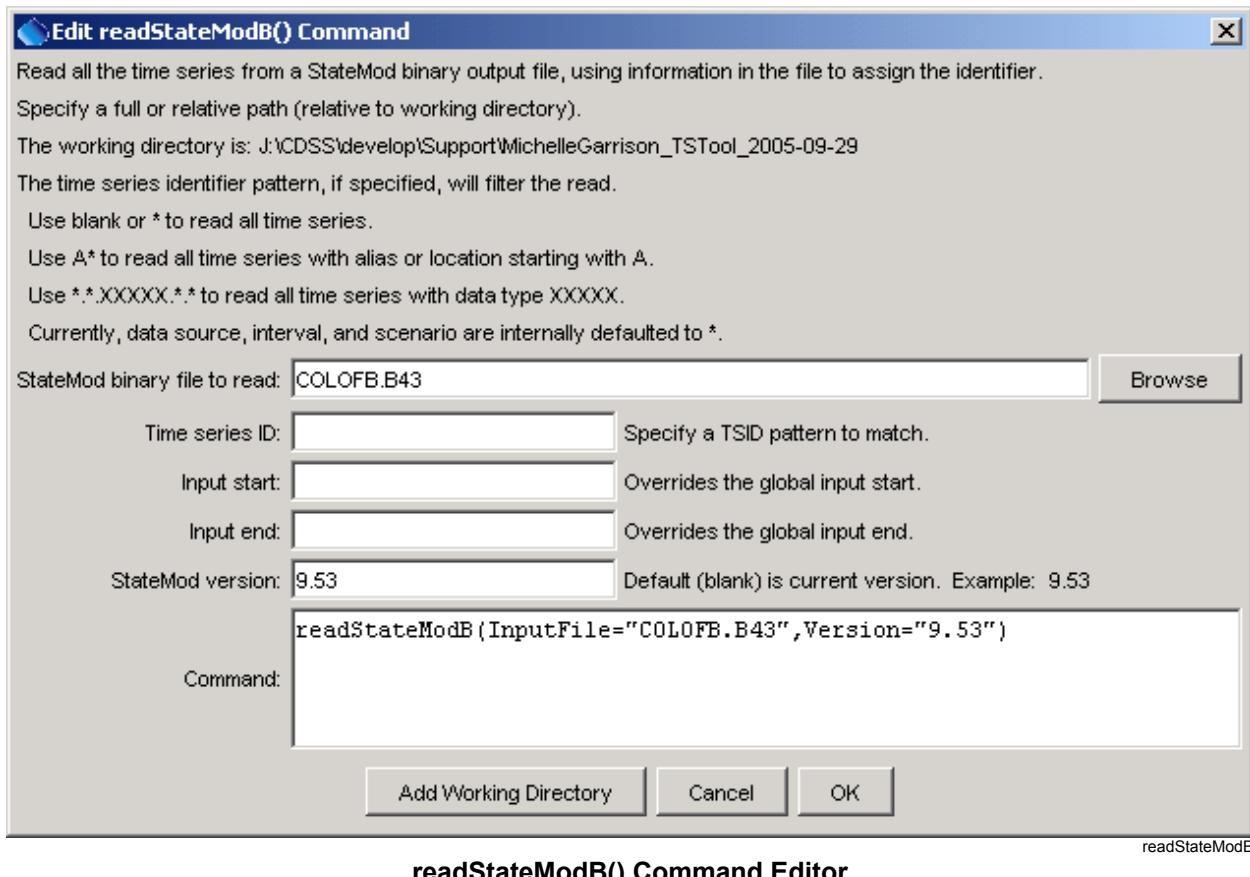
readStateModB()

Read time series from a StateMod binary output time series file

Version 06.10.09, 2005-09-30, Color, Acrobat Distiller

The `readStateModB()` command reads time series from a StateMod binary output time series file into memory (see the **StateModB Input Type** appendix). The actual reading occurs as the commands are being processed. Therefore, if any other commands reference the StateMod time series, the time series identifiers must be specified manually or use wildcards in identifiers (identifiers are not available to list in dialogs like other time series specified with identifiers).

The following dialog is used to edit the command and illustrates the syntax for the command.



readStateModB() Command Editor

readStateModB

The command syntax is as follows:

```
readStateModB(param=value,...)
```

Command Parameters

Parameter	Description	Default
InputFile	The name of the StateMod binary time series file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
TSID	Time series identifier pattern to filter the read.	Read all time series.
Version	StateMod version number corresponding to the file. The version is not included in the file and must be specified as a parameter to for older files. Changes in format occurred with version 9.01 and 9.69, mainly to add new data types.	Assume that the file format matches the current version.

The following example commands file illustrates how to read all Available_Flow time series for identifiers starting with 44 (e.g., to extract all such time series for a water district):

```
readStateModB(InputFile=".\\StateMod\\ym2002b.b43",TSID="44*.*.Available_Flow.*")
```

The following example illustrates how to read all time series from a binary file that was created with StateMod version 9.53 (e.g., to convert to a different format). As shown in the example, debug can be turned on for the log file to evaluate issues with the file format.

```
startLog(LogFile="commands.TSTool.log")
setDebugLevel(0,1)
readStateModB(InputFile="COLOFB.B43",Version="9.53")
```

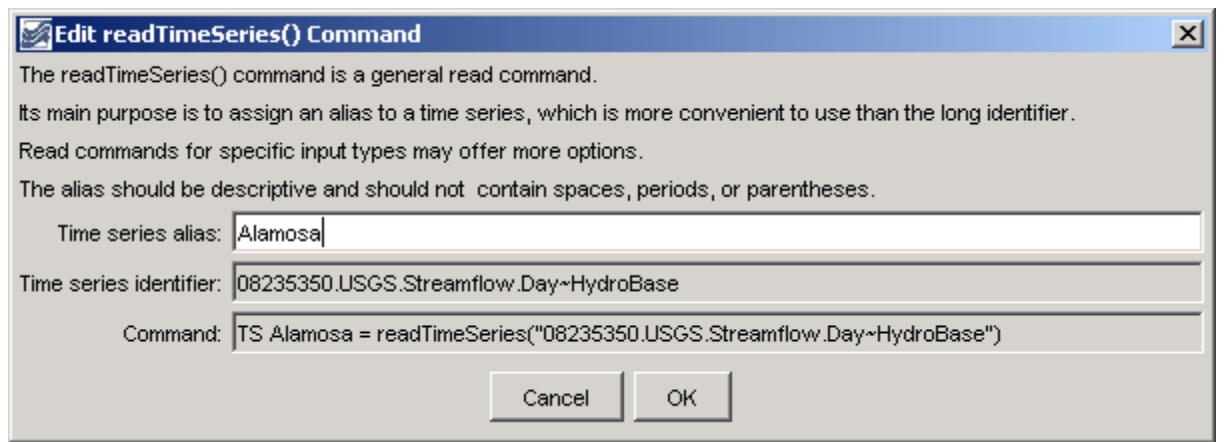
Command Reference

TS X = readTimeSeries()

Read a Single Time Series from an Input Type

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The TS X = readTimeSeries() command is used to read a single time series from any input type. This generalized command is useful for converting time series identifiers from the TSTool interface into read commands that assign an alias to a time series. Because the command is generic, it does not offer specific parameters that may be found in read commands for specific input types. Use the specific read commands where available. The following dialog is used to edit the command and illustrates the syntax of the command.



readTimeSeries() Command Editor

TS_readTimeSeies

The command syntax is as follows:

```
TS X = readTimeSeries("TSID")
```

Command Parameters

Parameter	Description	Default
X	The alias assigned to the time series.	None – must be specified.
TSID	The time series identifier of the time series to read. The identifier should include the input type (and input name) if required. See the input type appendices for examples of time series identifiers for various input types.	None – must be specified.

A sample commands file is as follows:

```
TS Alamosa = readTimeSeries("08235350.USGS.Streamflow.Day~HydroBase")
```

This page is intentionally blank.

Command Reference

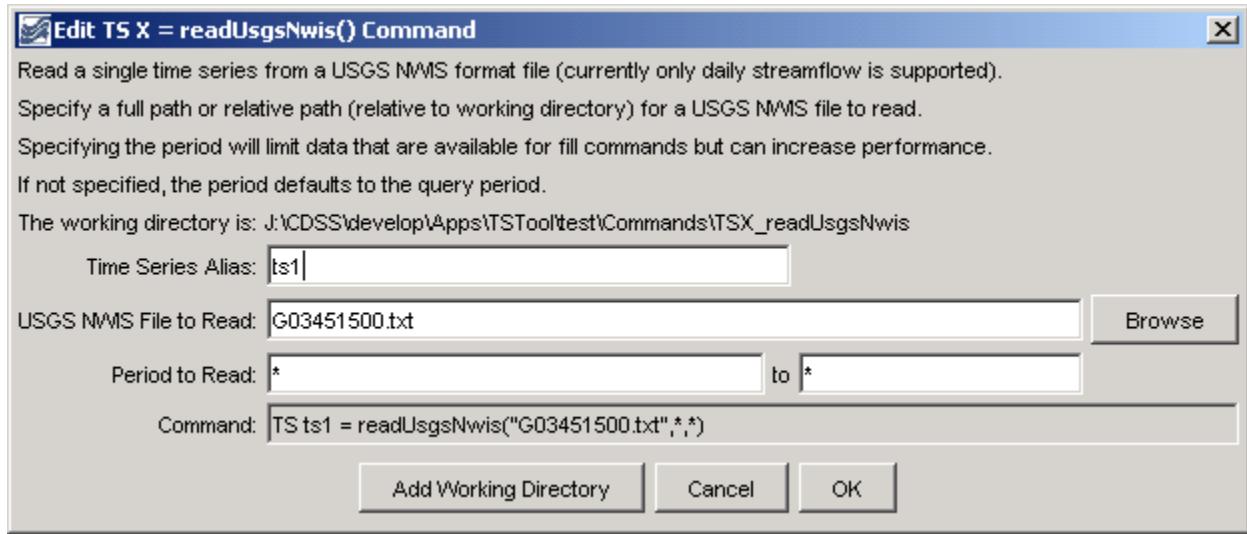
TS Alias = readUsgsNwis()

Read a single time series from a USGS NWIS file

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The TS Alias = readUsgsNwis() command reads a single time series from a USGS NWIS file (see the **USGSNWIS Input Type** appendix) and assigns an alias to the result. Currently only the daily streamflow format is supported and the file being read **must contain only one time series**.

The following dialog is used to edit the command and illustrates the syntax.



TS Alias = readUsgsNwis() Command Editor

TS_readUsgsNwis

The command syntax is as follows:

```
TS Alias =
readUsgsNwis(InputFile,TSID,NewUnits,InputStart,InputEnd)
```

Command Parameters

Parameter	Description	Default
Alias	Alias for the new time series that is read from the file, which can be used instead of the TSID in other commands.	None – must be specified.
InputFile	The name of the USGS NWIS file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
InputStart	The start of the period to read data – specify if the period should be different from the global query period.	Use the global query period.
InputEnd	The end of the period to read data – specify if the period should be different from the global query period.	Use the global query period.

A sample commands file is as follows:

TS ts1 = readUsgsNwis("G03451500.txt",*,*)
--

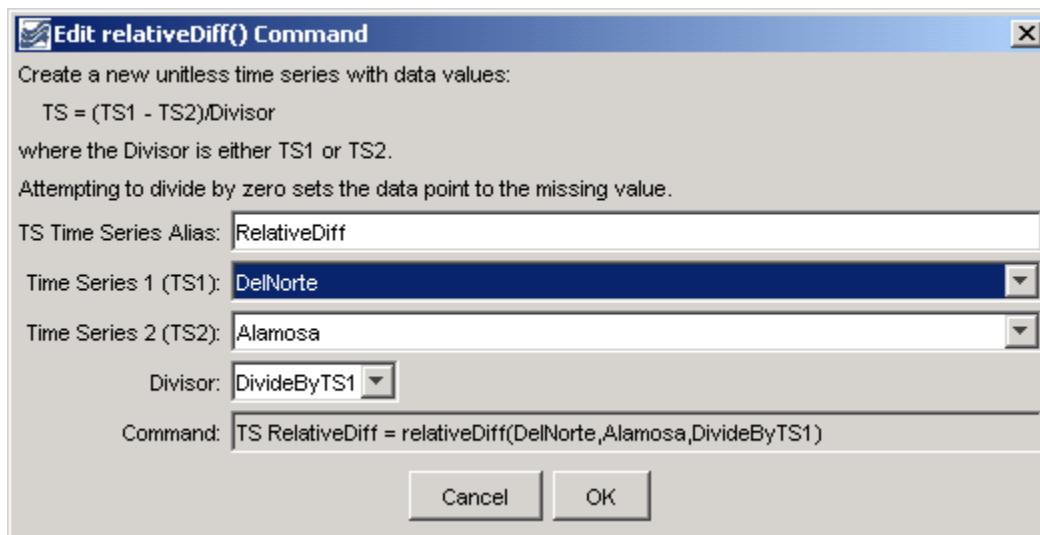
Command Reference

TS X = relativeDiff()

Create a Relative Difference Time Series

Version 06.09.00, 2004-09-14, Color, Acrobat Distiller

A `relativeDiff()` command can be inserted to create a new relative difference time series, computed by subtracting the time series and then dividing by one of the time series. This is useful when analyzing the relative magnitudes of two time series over time. Most of the properties for the new time series are the same as the first time series. The alias for the result can be referenced by other commands. The following dialog is used to edit the command and illustrates its syntax. The divisor can be either of the time series. The result is set to missing if either time series value is missing or the divisor is zero.



relativeDiff

relativeDiff() Command Editor

The command syntax is as follows:

```
TS X = relativeDiff(TSID1,TSID2,Divisor)
```

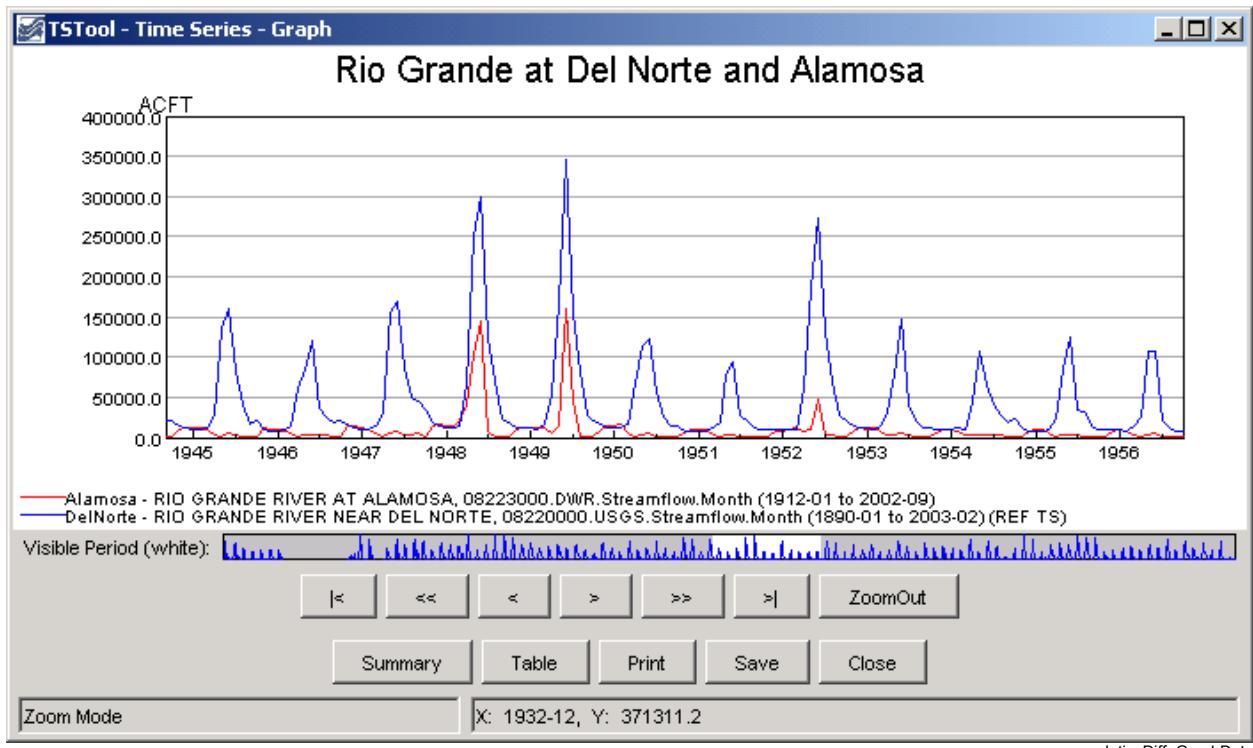
Command Parameters

Parameter	Description	Default
X	The alias for the new time series.	None – must be specified.
TSID1	The time series identifier or alias for the first time series.	None – must be specified.
TSID2	The time series identifier or alias for the second time series (subtracted from the first).	None – must be specified.
Divisor	Indicates whether the first time series is the divisor (<code>DivideByTS1</code>) or the second time series is the divisor (<code>DivideByTS2</code>).	None – must be specified.

A sample commands file is as follows:

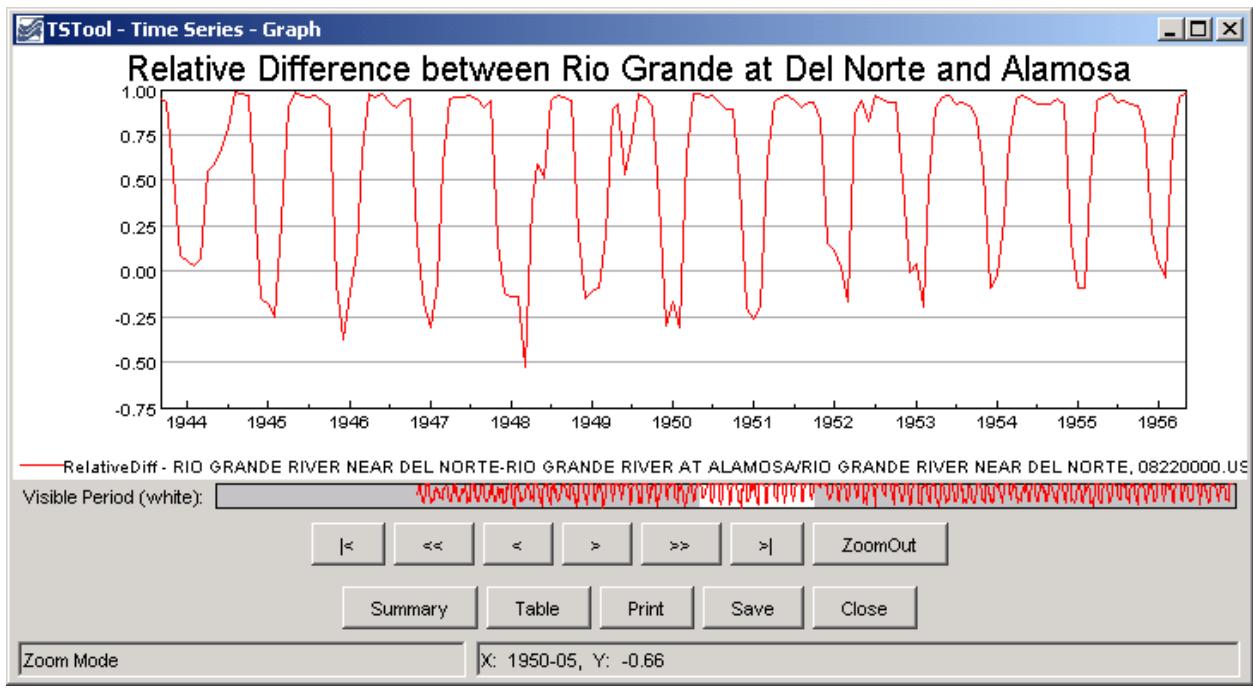
```
setOutputPeriod(01/1912,12/1998)
# (1912-1998) RIO GRANDE AT ALAMOSA, CO. DWR Streamflow Monthly
TS Alamosa = readTimeSeries("08223000.DWR.Streamflow.Month~HydroBase")
# (1890-1998) RIO GRANDE NEAR DEL NORTE, CO. DWR Streamflow Monthly
TS DelNorte = readTimeSeries("08220000.USGS.Streamflow.Month~HydroBase")
TS RelativeDiff = relativeDiff(DelNorte,Alamosa,DivideByTS1)
```

The input time series for the command are shown in the following figure:



Data for the relativeDiff() Command

The results of processing the commands are shown in the following figure:



Results of the relativeDiff() Command

This page is intentionally blank.

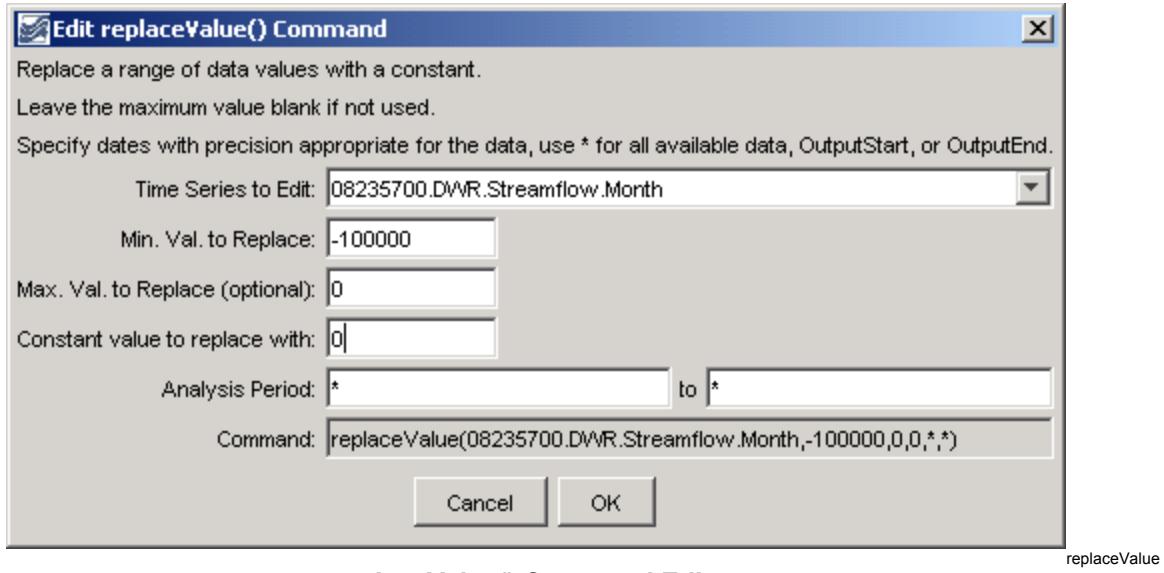
Command Reference

replaceValue()

Replace a Range of Data Values with a Constant Value

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `replaceValue()` command replaces a range of values in a time series with a constant value. The following dialog is used to edit the command and illustrates the syntax of the command:



replaceValue() Command Editor

The command syntax is as follows:

```
replaceValue(TSID,MinValue,MaxValue,NewValue,
AnalysisStart,AnalysisEnd)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A single time series or all (*) can be specified.	None – must be specified.
MinValue	The minimum value to replace.	None – must be specified.
MaxValue	The maximum value to replace.	If not specified, only data values that exactly match the minimum value will be replaced.
NewValue	The new data value.	None – must be specified.
AnalysisStart	The date/time to start filling, if other than the full time series period. Specify as * to process the full period.	None – must be specified.
AnalysisEnd	The date/time to end filling, if other than the full time series period. Specify as * to process the full period.	None – must be specified.

A sample commands file is as follows:

```
# 08235700 - ALAMOSA RIVER BELOW CASTLEMAN GULCH NEAR JASPER
08235700.DWR.Streamflow.Month~HydroBase
replaceValue(08235700.DWR.Streamflow.Month,-100000,0,0,*,*)
```

Command Reference

runCommands()

Run a commands file

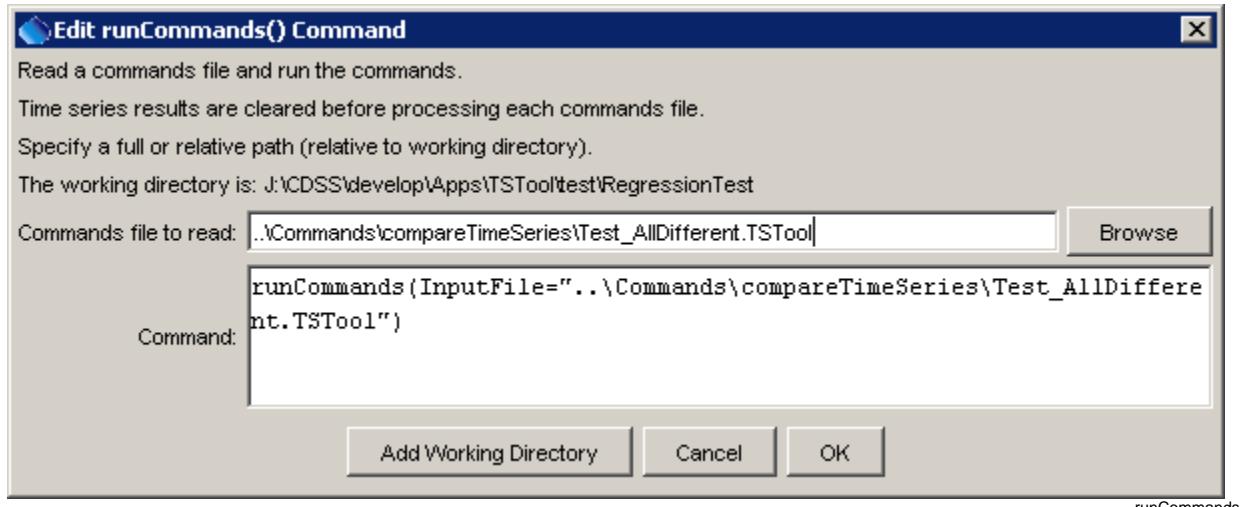
Version 06.18.00, 2006-05-09, Color, Acrobat Distiller

The runCommands () command runs a commands file. This command can be used to manage workflow where multiple commands files are run, and is also useful for testing.

Commands files that are run CANNOT themselves include runCommands () commands. There is currently no special handling of log files; consequently, if the main commands file opens a log file and then a commands file is run that opens a new log file, the main log file will be closed. This behavior is being evaluated.

The working directory is reset to that of the commands file being run. This allows a master commands file to reside in a different location than the individual commands files that are being run.

The following dialog is used to edit the command and illustrates the syntax for the command.



runCommands() Command Editor

The command syntax is as follows:

```
runCommands (param=value ,...)
```

Command Parameters

Parameter	Description	Default
InputFile	The name of the commands file to be run, enclosed in double quotes if the file contains spaces or other special characters. A path relative to the master commands file can be specified.	None – must be specified.
AppendResults	Indicate whether time series results from each commands file should be appended to the overall time series results. This parameter currently always defaults to False, but support for True may be implemented in the future. Consequently, only the time series results from the last commands file that is run will be displayed in TSTool.	Currently always False

The following example illustrates how the `runCommands()` command can be used to test TSTool software. First, individual commands files are implemented to test specific functionality, which will result in warnings if a test fails:

```
# Commands file to make sure that warnings are generated for different data.
readDateValue("RawData1.dv")
readDateValue("RawData1Scaled.dv")
# Generate an error if the files are the same.
# Since the files are different this would indicate a coding error.
compareTimeSeries(WarnIfSame=True)
```

Next, use the `runCommands()` command to run one or more tests:

```
# startLog(LogFile="Regression.log", Suffix="Date")
runCommands(InputFile=".\\Commands\\compareTimeSeries\\Test_AllDifferent.TSTool")
runCommands(InputFile=".\\Commands\\compareTimeSeries\\Test_AllSame.TSTool")
runCommands(InputFile=".\\Commands\\readHydroBase\\2000812.TSTool")
runCommands(InputFile=".\\Commands\\readHydroBase\\2002029.TSTool")
```

Each of the above commands files should produce expected time series results, without warnings. If any commands file unexpectedly produces a warning, a warning will also be visible in TSTool. The issue can then be evaluated to determine whether a software or configuration change is necessary.

Command Reference

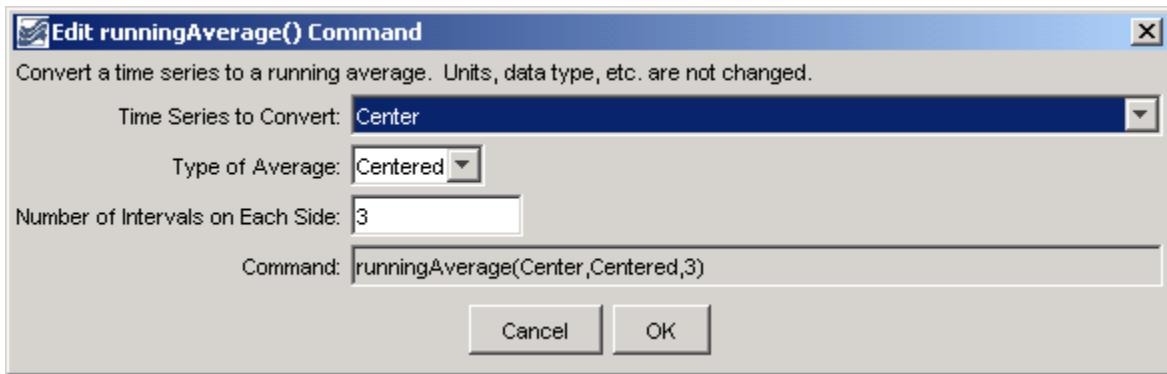
runningAverage()

Convert Time Series Data to Running Average Values

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

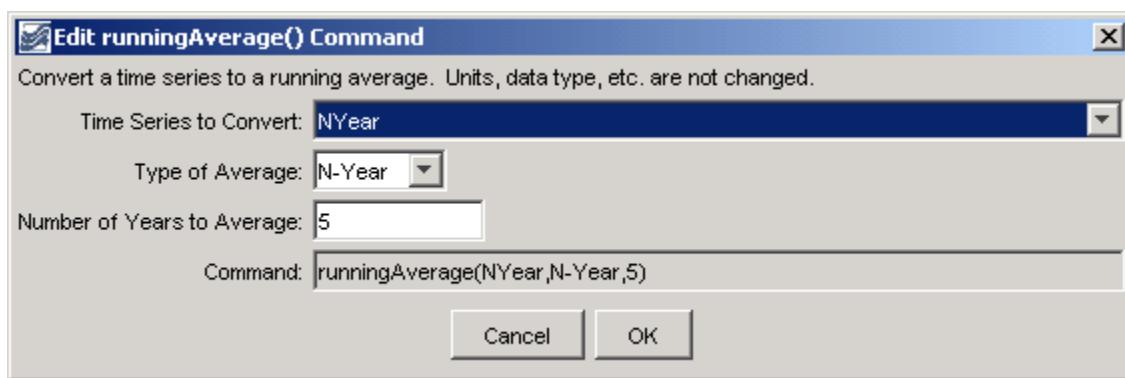
The `runningAverage()` command converts a time series' raw data values to a running average, resulting in data that are smoothed. There are two versions of the command. The centered running average requires that the number intervals on each side of a point be specified (e.g., specifying 1 will average 3 values at each point). The N-year running average is computed by averaging the current year and N - 1 values on previous years, for a specific date. An average value is produced only if the needed number of non-missing values is available.

The following dialog is used to edit the command and illustrates the centered running average command syntax.



runningAverage() Command Editor for Centered Running Average

The following dialog illustrates the N-year running average command syntax.



runningAverage() Command Editor for N-Year Running Average

The command syntax is as follows:

```
runningAverage(TSID, AverageMethod, Bracket)
```

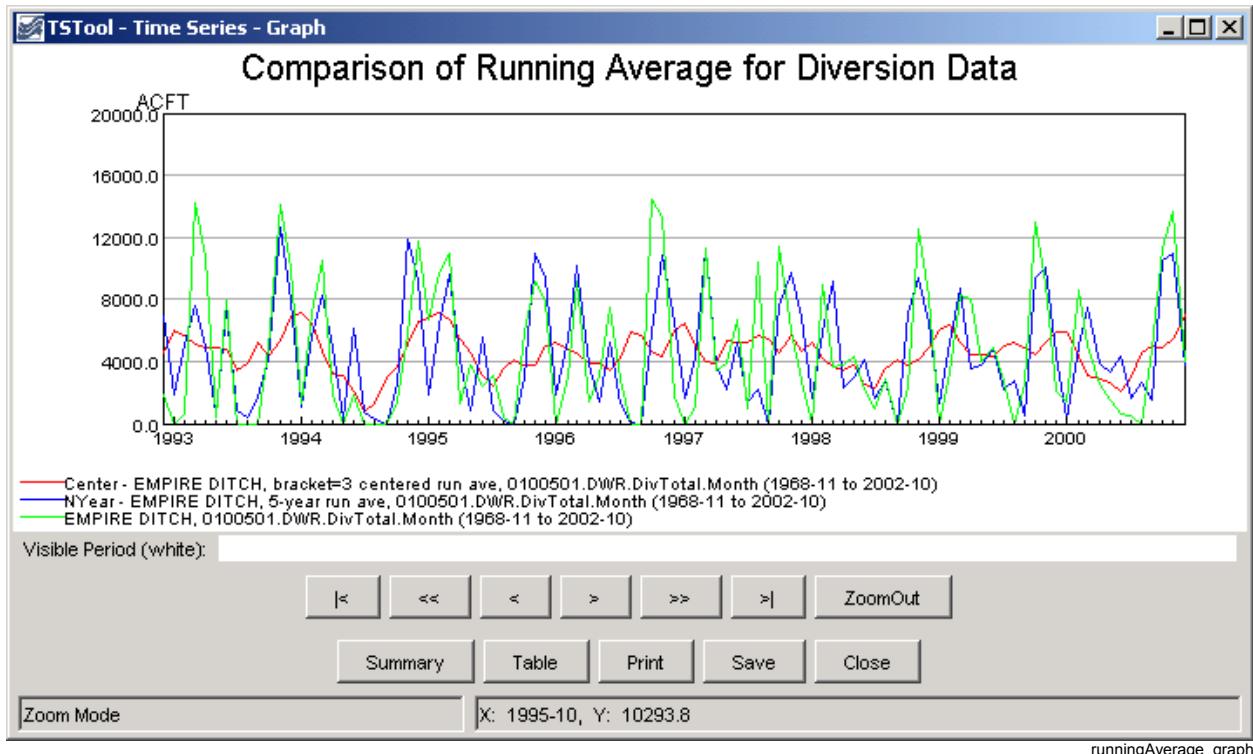
Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A specific time series or all time series (*) can be converted to running averages.	None – must be specified.
AverageMethod	The method used to create the running average, one of: <ul style="list-style-type: none">• Centered – values on each side of a date/time are averaged.• N-Year – values for the current year and (N – 1) preceding years, for the same date/time, are averaged.	None – must be specified.
Bracket	For centered running average, the bracket is the number of points on each side of the current point (therefore a value of 1 will average 3 data values). For N-year running average, the bracket is the total number of years to average, including the current year.	None – must be specified.

A sample commands file is as follows:

```
# 0100501 - EMPIRE DITCH
TS Center = readTimeSeries("0100501.DWR.DivTotal.Month~HydroBase")
runningAverage(Center,Centred,3)
TS NYear = readTimeSeries("0100501.DWR.DivTotal.Month~HydroBase")
runningAverage(NYear,N-Year,5)
0100501.DWR.DivTotal.Month~HydroBase
```

The resulting graph is as follows:



Results from runningAverage() Commands

runningAverage_graph

This page is intentionally blank.

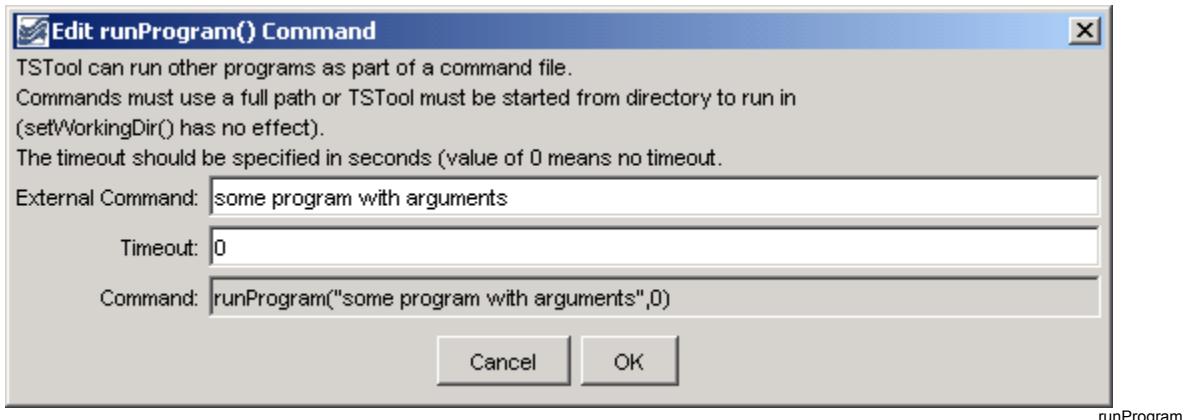
Command Reference

runProgram()

Run an External Program

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

To run an external program, insert a `runProgram()` command. This command will run the given command line and wait until the program is finished before processing additional commands. It is therefore possible to call an external program that reads and/or writes recognized time series formats and can perform some analysis that TSTool cannot. Another use of this command is to create a calibration environment where a model is run and then the results are read and displayed using TSTool. The following dialog is used to edit the command and illustrates the command syntax.



runProgram() Command Editor

runProgram

The command syntax is as follows:

```
runProgram(CommandLine,Timeout)
```

Command Parameters

Parameter	Description	Default
CommandLine	<p>The program command line, with arguments. If the program executable is found in the PATH, then only the program name needs to be specified. Otherwise, specify an absolute path to the program or run TSTool from the same directory.</p> <p>TSTool has no understanding of the external program's command line options. Therefore, it does not try to modify these options to reflect a working directory. It may therefore be necessary to run TSTool from a command prompt (rather from the Start menu) in order for the external program to run correctly.</p>	None – must be specified.
Timeout	The timeout in seconds – if the program has not yet returned, the process will be ended. Zero indicates no timeout.	None – must be specified.

Command Reference

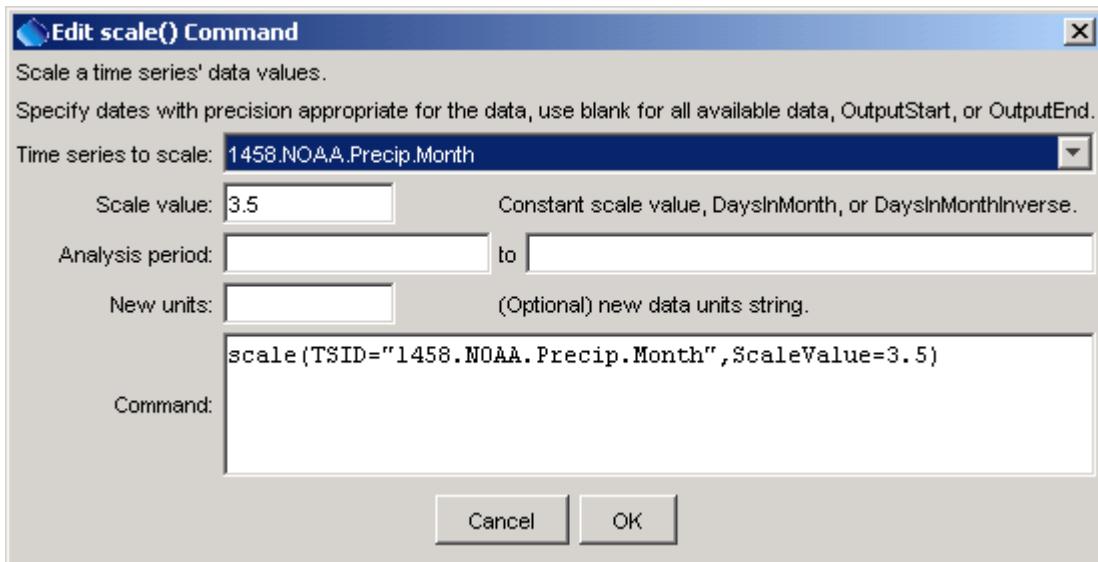
scale()

Scale time series data values by a constant value

Version 06.13.00, 2005-11-13, Color, Acrobat Distiller

The `scale()` command scales each non-missing value in the specified time series.

The following dialog is used to edit the command and illustrates the command syntax.



scale

scale() Command Editor

The command syntax is as follows:

```
scale(param=value, ...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A single time series or all time series (*) can be scaled.	None – must be specified.
ScaleValue	One of the following: <ul style="list-style-type: none"> The numerical value to scale to the time series. DaysInMonth to indicate a scale of the number of days in the month. DaysInMonthInverse to indicate a scale of the inverse of the number of days in the month. 	None – must be specified.
AnalysisStart	The date/time to start analyzing data.	Full period is analyzed.
AnalysisEnd	The date/time to end analyzing data.	Full period is analyzed.
NewUnits	New data units for the resulting time series.	Do not change the units.

The following example scales a precipitation time series by a factor of 3.5:

```
# 1458 - CENTER 4 SSW
1458.NOAA.Precip.Month~HydroBase
scale(TSID="1458.NOAA.Precip.Month", ScaleValue=3.5)
```

The following example scales a monthly streamflow time series with units of ACFT (volume per month) in order to convert the data to average CFS flow values (note that two scale commands are required because the DaysInMonthInverse value cannot currently be combined with a numerical value in one command):

```
# 06754000 - SOUTH PLATTE RIVER NEAR KERSEY
06754000.DWR.Streamflow.Month~HydroBase
scale(TSID="06754000.DWR.Streamflow.Month", ScaleValue=.5042)
scale(TSID="06754000.DWR.Streamflow.Month",
     ScaleValue=DaysInMonthInverse, NewUnits="CFS")
```

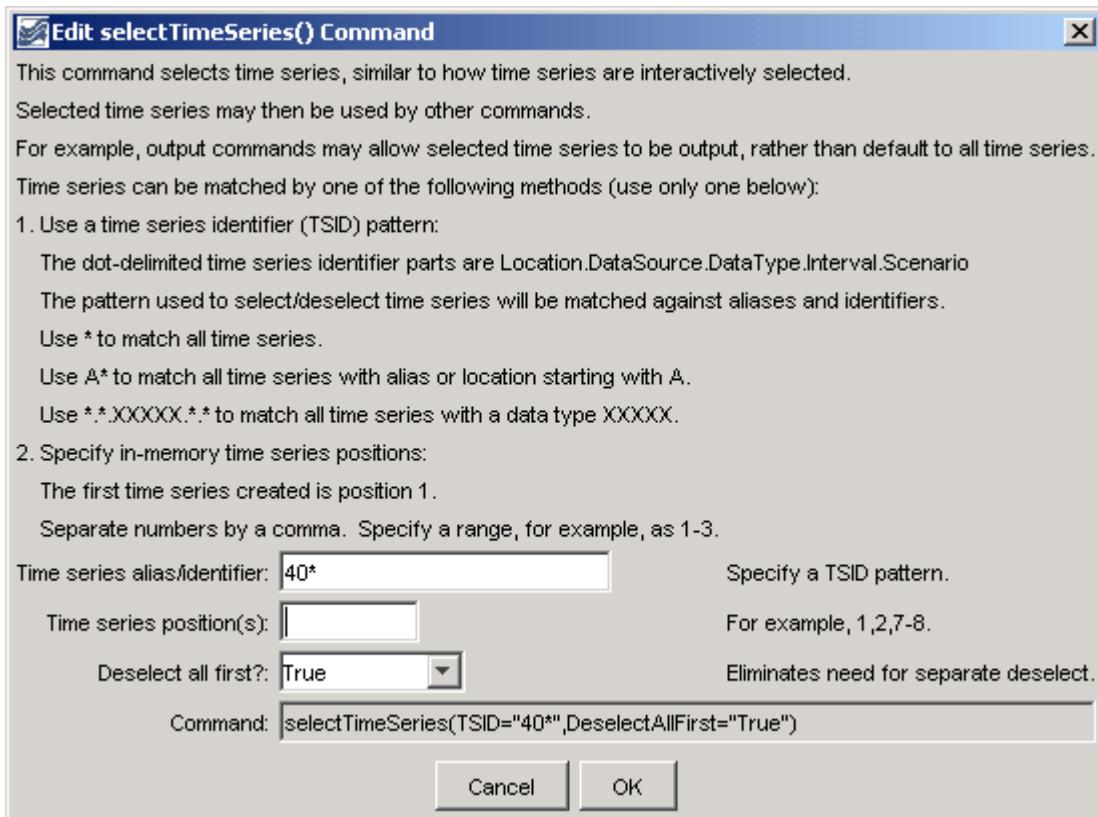
Command Reference

selectTimeSeries()

Select Time Series for Output

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `selectTimeSeries()` command selects output time series, as if done interactively. The command minimizes the need for the `free()` command, when used in conjunction with other commands that use a time series list based on selected time series. See also the `deselectTimeSeries()` command. The following dialog is used to edit the command and illustrates the command syntax.



selectTimeSeries

selectTimeSeries() Command Editor

The command syntax is as follows:

```
selectTimeSeries(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias pattern, to indicate time series to select. See the above figure for examples.	Can be blank if Pos is specified.
Pos	A list of time series positions in output (1+), separated by commas. This parameter is useful if it is known that the first time series is being modified by subsequent commands.	Use the TSID.
DeselectAllFirst	Indicates whether all time series should be deselected before selecting the specified time series: True or False.	False

A sample commands file is as follows:

```
selectTimeSeries(TSID="40*", DeselectAllFirst="True")
```

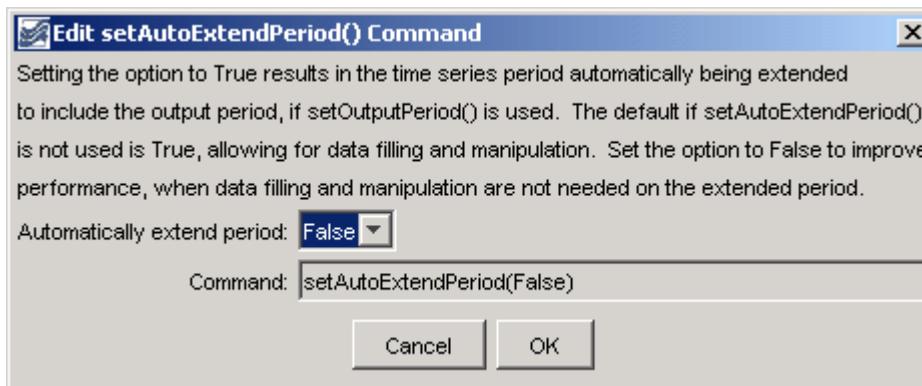
Command Reference

setAutoExtendPeriod()

Set Whether Time Series Periods Should Automatically be Extended to the Output Period

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

By default, the time series period is extended to include the output period, if specified. The `setAutoExtendPeriod()` command can be used to change this setting if it is not desirable (e.g., for performance reasons). The following dialog is used to edit the command and illustrates the command syntax.



setAutoExtendPeriod

setAutoExtendPeriod() Command Editor

The command syntax is as follows:

```
setAutoExtendPeriod(AutoExtendPeriod)
```

Command Parameters

Parameter	Description	Default
AutoExtendPeriod	Indicate whether the period of time series should be automatically extended to the output period when time series are read, <code>True</code> or <code>False</code> .	None – must be specified. The default is <code>True</code> if this command is not used.

A sample commands file is as follows:

```
setAutoExtendPeriod(False)
```

This page is intentionally blank.

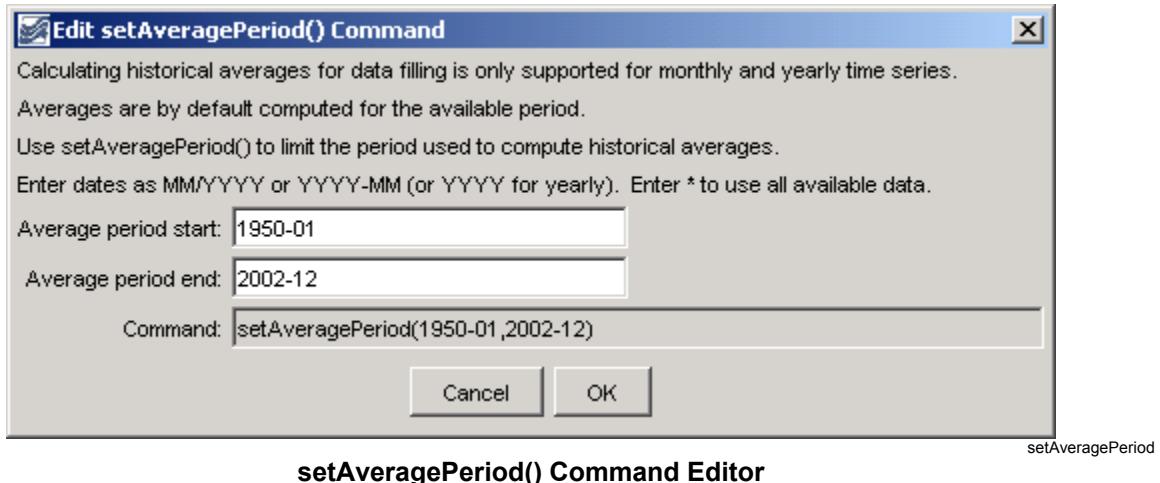
Command Reference

setAveragePeriod()

Set Period Used to Compute Historical Averages

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `setAveragePeriod()` command sets the period that is used to compute historic averages used with the `fillHistMonthAverage()` and `fillHistYearAverage()` commands. If the averaging period is not specified, the available period is used. Use a `setAveragePeriod()` command if a subset of the data should be used to compute averages. The following dialog is used to edit this command and illustrates the command syntax.



setAveragePeriod() Command Editor

The command syntax is as follows:

```
setAveragePeriod(AverageStart,AverageEnd)
```

Command Parameters

Parameter	Description	Default
AverageStart	The date for the start of the averaging period. The precision of the date should agree with that of time series to be processed.	None – must be specified.
AverageEnd	The date for the end of the averaging period. The precision of the date should agree with that of time series to be processed.	None – must be specified.

A sample commands file is as follows:

```
setAveragePeriod(1950-01,2002-12)
```

This page is intentionally blank.

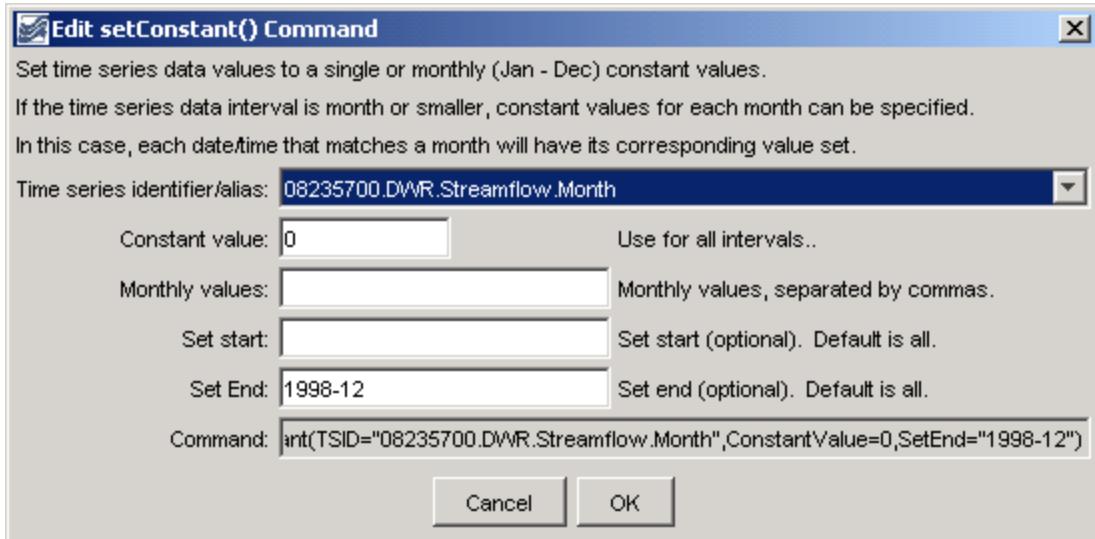
Command Reference

setConstant()

Set Time Series Data to a single or monthly constant values

Version 06.08.02, 2004-08-12, Color, Acrobat Distiller

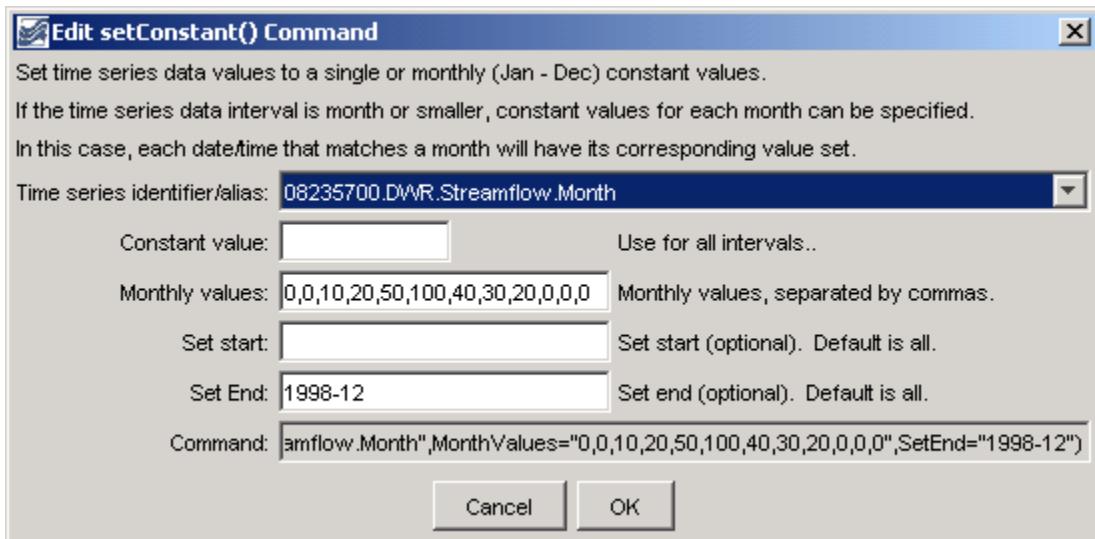
The setConstant() command sets the values of a time series to a single or monthly constant values. The following dialog is used to edit the command and illustrates the command syntax:



setConstant

setConstant() Command Editor

An example for setting only the June and July values is shown in the following figure:



setConstant_month

setConstant() Command Editor, for Monthly Values

The command syntax is as follows:

```
setConstant(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A single time series or all (*) can be specified.	None – must be specified.
ConstantValue	The constant value to use as the data value.	None – must be specified, or specify monthly values.
MonthValues	Monthly values to use as the data values. Twelve values can be specified, separated by commas. If the time series data interval is less than monthly, each date/time will be set for a specific month.	None – must be specified, or specify a constant value.
SetStart	The starting date/time for the data set.	Set data for the full period.
SetEnd	The ending date/time for the data set.	Set data for the full period.

A sample commands file is as follows:

```
# 08235700 - ALAMOSA RIVER BELOW CASTLEMAN GULCH NEAR JASPER
08235700.DWR.Streamflow.Month~HydroBase
setConstant(TSID="08235700.DWR.Streamflow.Month", ConstantValue=0, SetEnd="1998-12" )
```

Command Reference

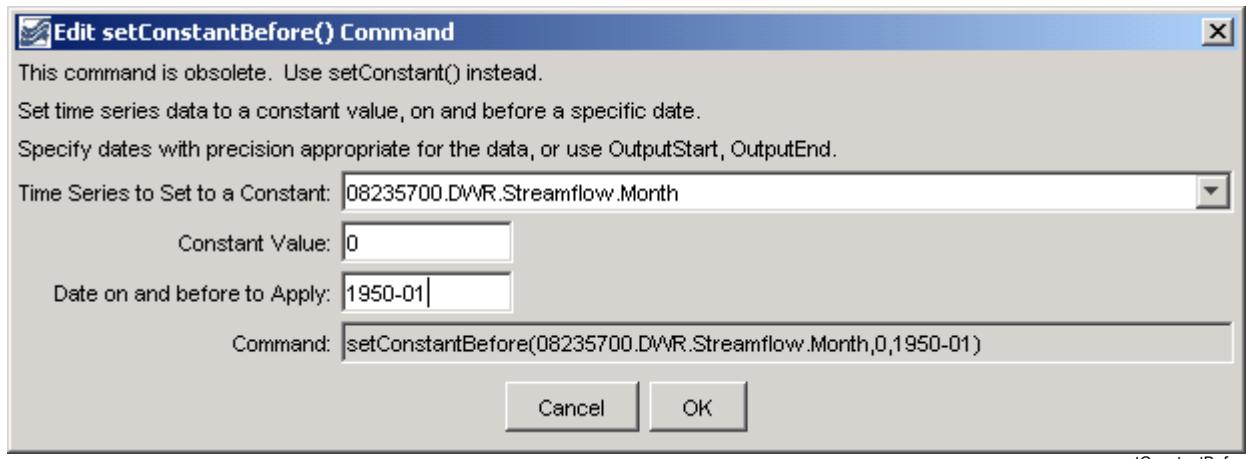
setConstantBefore()

Set time series data to a constant value on and before a specified date/time

Version 06.08.02, 2004-08-12, Color, Acrobat Distiller

The `setConstantBefore()` command sets the values of a time series to a constant value, on and before a specified date/time. This is useful, for example, to remove the effects of a reservoir from a system, before it was in place. **This command is obsolete. For equivalent functionality, use `setConstant()` and specify only the end date.**

The following dialog is used to edit the command and illustrates the command syntax.



setConstantBefore() Command Editor

The command syntax is as follows:

```
setConstantBefore(TSID, ConstantValue, ConstantEnd)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified. A single time series or all (*) can be specified.	None – must be specified.
ConstantValue	The constant value to use as the data value.	None – must be specified.
SetEnd	The last date/time on which to apply the constant value.	None – must be specified.

A sample commands file is as follows:

```
# 08235700 - ALAMOSA RIVER BELOW CASTLEMAN GULCH NEAR JASPER
08235700.DWR.Streamflow.Month~HydroBase
setConstantBefore(08235700.DWR.Streamflow.Month,0,1950-01)
```

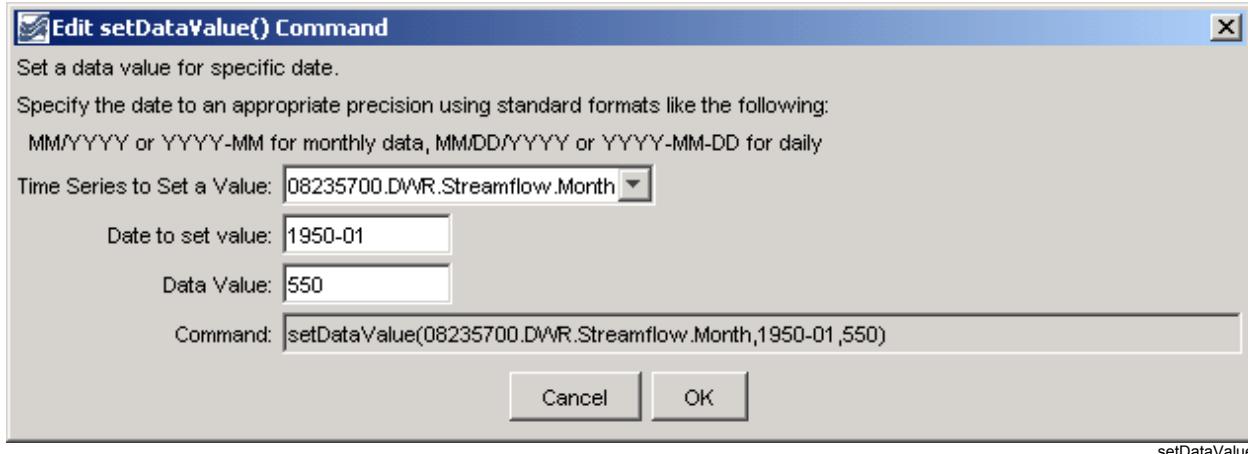
Command Reference

setDataSource()

Set a Specific Time Series Data Value at a Date/Time

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `setDataSource()` command sets a single data value in a time series. Consequently, it can be used to condition filling or to "hard-code" data. The following dialog is used to edit the command and illustrates the syntax of the command. **It is good practice to insert comments when editing data to explain the edits.**



setDataSource

setDataSource() Command Editor

The command syntax is as follows:

```
setDataSource(TSID,DateTime,NewValue)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
DateTime	The date/time at which the data value should be set. Specify the date/time precision according to the time series that is being manipulated.	None – must be specified.
NewValue	The new data value.	None – must be specified.

A sample commands file is as follows:

```
# 08235700 - ALAMOSA RIVER BELOW CASTLEMAN GULCH NEAR JASPER
08235700.DWR.Streamflow.Month~HydroBase
setDataSource(08235700.DWR.Streamflow.Month,1950-01,550)
```

This page is intentionally blank.

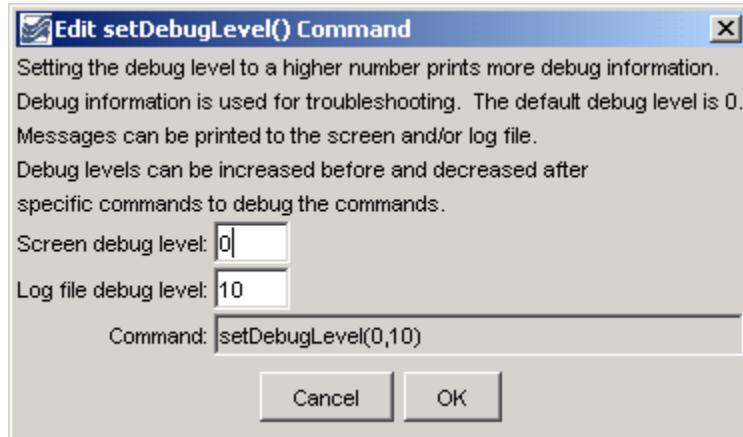
Command Reference

setDebugLevel()

Set Level for Debug Messages

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `setDebugLevel()` command sets the debug levels for screen and log file diagnostic messages. This command can be used multiple times with different debug level (e.g., to isolate a problem). The following dialog is used to edit this command and illustrates the command syntax.



setDebugLevel

setDebugLevel() Command Editor

The command syntax is as follows:

```
setDebugLevel(ScreenLevel,LogFileLevel)
```

Command Parameters

Parameter	Description	Default
ScreenLevel	The debug level for the screen (0+).	None – must be specified.
LogFileLevel	The debug level for the log file (0+).	None – must be specified.

A sample commands file is as follows:

```
setDebugLevel(0,10)
```

This page is intentionally blank.

Command Reference

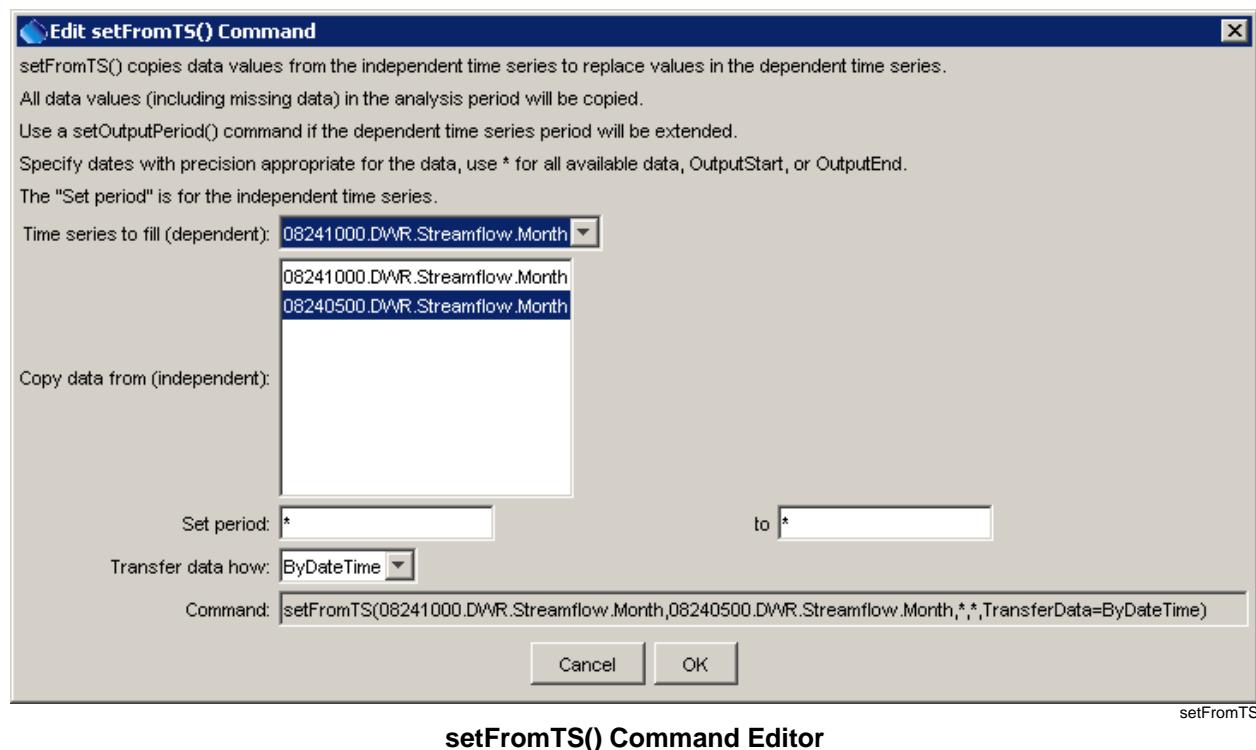
setFromTS()

Set Time Series Data Using Another Time Series

Version 06.17.00, 2006-04-28, Color, Acrobat Distiller

The `setFromTS()` command sets data in a time series by transferring values from another time series. An analysis period can be specified to limit the period that is processed. See also the `fillFromTS()` command, which will transfer values only when the dependent time series has missing data. Only data values are transferred – time series header information (e.g., data type, alias) will not be modified.

The following dialog is used to edit the command and illustrates the command syntax.



setFromTS() Command Editor

The command syntax is as follows:

```
setFromTS (TSID, IndependentTSID, SetStart, SetEnd, TransferHow)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
IndependentTSID	The time series identifier or alias for the independent time series, to supply data.	None – must be specified.
SetStart	The date/time to start setting data, if other than the full time series period.	Full period if * is specified.
SetEnd	The date/time to end setting data, if other than the full time series period.	Full period if * is specified.
TransferHow	Indicates how to transfer data: <ul style="list-style-type: none"> • ByDateTime – a date/time in one time series will be lined up with the other time series. • Sequentially – data from the independent will be transferred sequentially, even if the date/time does not align (used when transferring continuous data over Feb 28/29, without gaps). 	None – must be specified.

A sample commands file is as follows:

```
#  
# 08241000 - TRINCHERA CREEK ABOVE MOUNTAIN HOME RESERVOIR  
08241000.DWR.Streamflow.Month~HydroBase  
# 08240500 - TRINCHERA CREEK ABOVE TURNER'S RANCH  
08240500.DWR.Streamflow.Month~HydroBase  
setFromTS(08241000.DWR.Streamflow.Month, 08240500.DWR.Streamflow.Month, *, *,  
TransferData=ByDateTime)
```

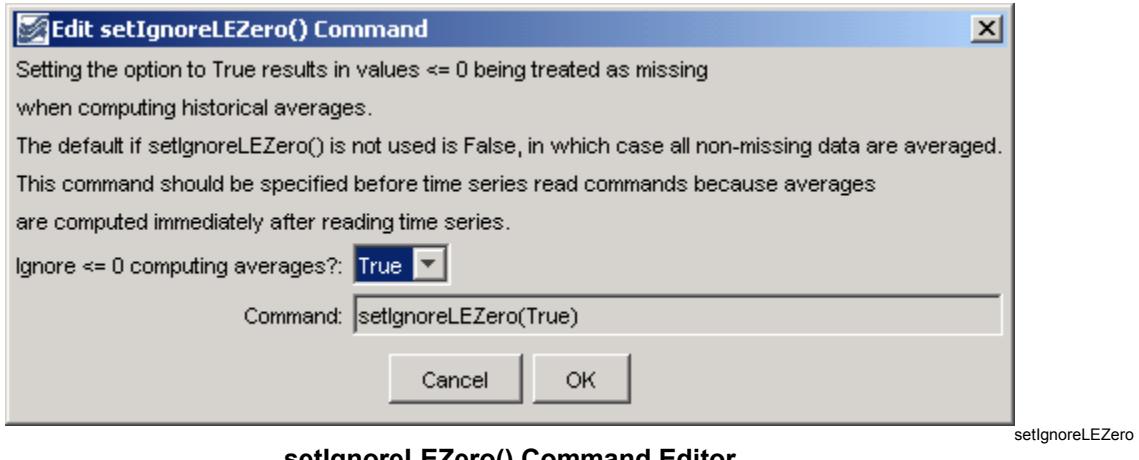
Command Reference

setIgnoreLEZero()

Indicate Whether Time Series Data Values <= Zero Should be Ignored in Historical Averages

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `setIgnoreLEZero()` command sets the global property that indicates whether the computation of historical averages for time series should ignore values less than or equal to zero. By default, all values (other than the missing data placeholder) are used to compute averages. This command is useful when it is appropriate to ignore zero and negative values in averages. The following dialog is used to edit this command and illustrates the syntax of the command.



setIgnoreLEZero() Command Editor

The command syntax is as follows:

```
setIgnoreLEZero( IgnoreLEZero )
```

Command Parameters

Parameter	Description	Default
IgnoreLEZero	Indicates whether the computation of historical averages should ignore values less than or equal to zero, True or False.	If this command is not used, the default is False.

A sample commands file is as follows:

```
setIgnoreLEZero( True )
```

This page is intentionally blank.

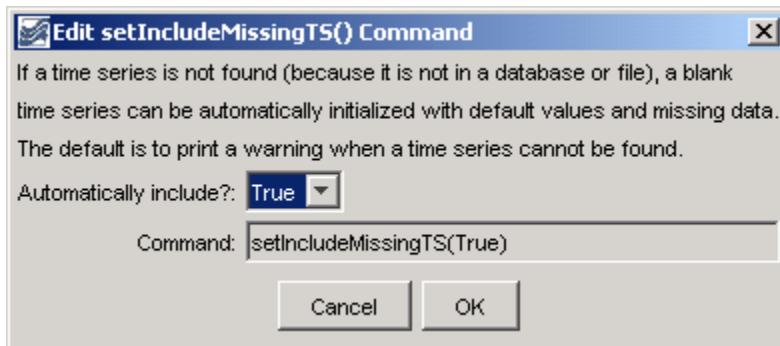
Command Reference

setIncludeMissingTS()

Indicate Whether Missing Time Series Should be Added

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `setIncludeMissingTS()` command sets the global property that indicates whether time series that cannot be found should be added with missing data. By default, time series that cannot be found generate a warning. This command is useful when processing large amounts of data with wildcards, to guarantee a placeholder time series even if time series are not found. The following dialog is used to edit this command and illustrates the syntax of the command.



setIncludeMissingTS

setIncludeMissingTS() Command Editor

The command syntax is as follows:

```
setIncludeMissingTS(IncludeMissingTS)
```

Command Parameters

Parameter	Description	Default
IncludeMissingTS	Indicates whether time series that are not found with read commands should automatically be added with missing data.	If this command is not used, the default is false.

A sample commands file is as follows:

```
setIncludeMissingTS(True)
```

This page is intentionally blank.

Command Reference

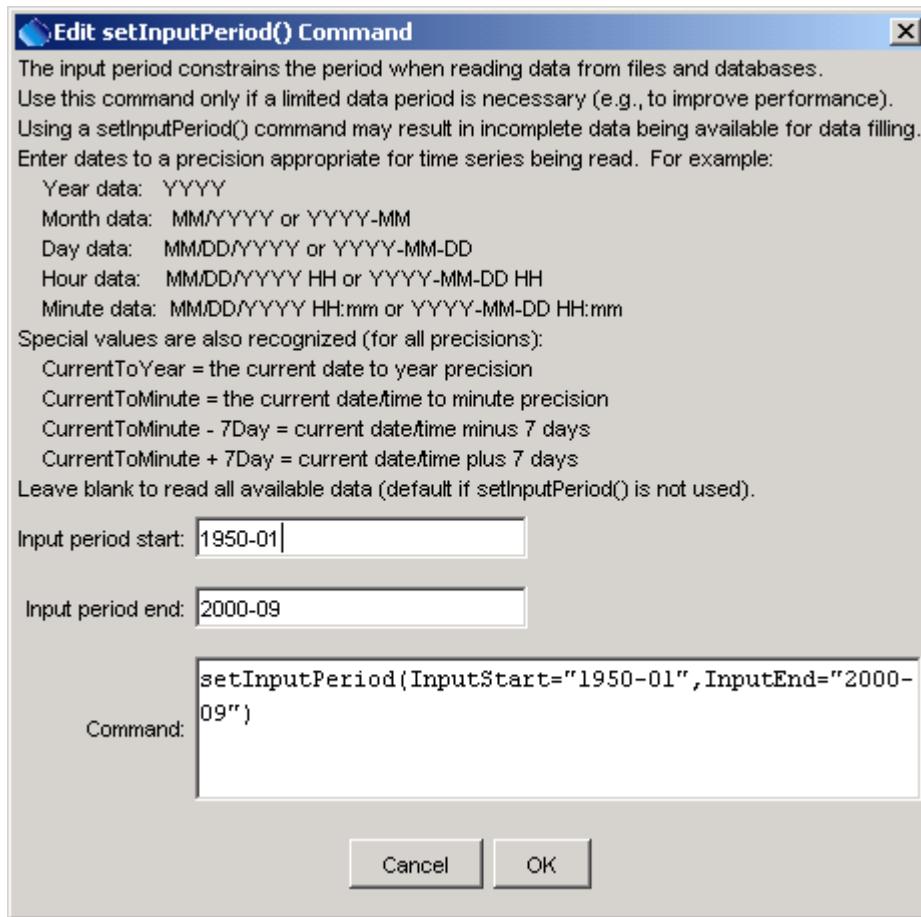
setInputPeriod()

Set the period for reading time series from files and querying from databases

Version 06.14.00, 2005-12-14, Color, Acrobat Distiller

The `setInputPeriod()` command sets the period used for reading time series data from files and querying data from databases. The default is to read/query all available data so that all data are available for analysis and data filling. However, a shorter period may be desirable to increase performance (e.g., when processing real-time data) or to force matching a historical period. This command replaces the `setQueryPeriod()` command. See also the `setOutputPeriod()` command.

The following dialog is used to edit the command and illustrates the command syntax.



setInputPeriod() Command Editor

setInputPeriod

The command syntax is as follows:

```
setInputPeriod(param=value,...)
```

Command Parameters

Parameter	Description	Default
InputStart	<p>The date/time to start reading/querying time series data, one of:</p> <ul style="list-style-type: none"> • A date/time string (see dialog above for examples). • CurrentToYear, CurrentToMonth, CurrentToDay, CurrentToHour, CurrentToMinute, indicating the current date/time to the specified precision. • A Current* value +- an interval, for example: CurrentToMinute - 7Day 	None – must be specified.
InputEnd	<p>The date/time to end reading/querying time series data, one of:</p> <ul style="list-style-type: none"> • A date/time string (see dialog above for examples). • CurrentToYear, CurrentToMonth, CurrentToDay, CurrentToHour, CurrentToMinute, indicating the current date/time to the specified precision. • A Current* value +- an interval, for example: CurrentToMinute - 7Day • An expression involving InputStart, used similar to the Current* values. 	None – must be specified.

A sample commands file for historical data is as follows:

```
setInputPeriod(InputStart="1950-01",InputEnd="2000-09")
# 06754000 - SOUTH PLATTE RIVER NEAR KERSEY
06754000.DWR.Streamflow.Month~HydroBase
```

A sample commands file for real-time data is as follows:

```
setInputPeriod(InputStart="CurrentToMinute - 14Day",InputEnd="CurrentToMinute + 1Hour")
# 06754000 - SOUTH PLATTE RIVER NEAR KERSEY
06754000.DWR.Streamflow-DISCHRG.Irregular~HydroBase
```

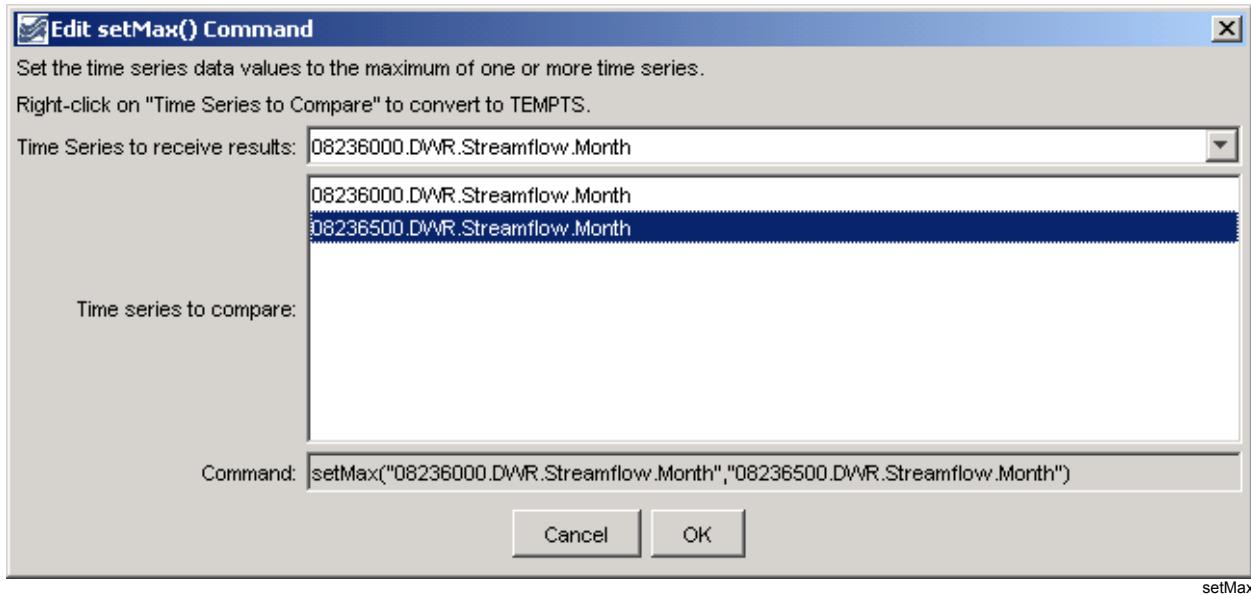
Command Reference

setMax()

Set Data Values to the Maximum of Values from One or More Time Series

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `setMax()` command sets a time series to contain, for each time step, the maximum of its own values and those of one or more additional time series. The following dialog is used to edit the command and illustrates the command syntax.



setMax() Command Editor

The command syntax is as follows:

```
setMax(TSID, IndependentTSID, IndependentTSID)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
IndependentTSID	The time series identifier or alias for the independent time series, to supply data. Repeat for each time series to be compared.	None – at least one time series identifier or alias must be specified.

A sample commands file is as follows:

```
# 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR
08236000.DWR.Streamflow.Month~HydroBase
# 08236500 - ALAMOSA RIVER BELOW TERRACE RESERVOIR
08236500.DWR.Streamflow.Month~HydroBase
setMax("08236000.DWR.Streamflow.Month", "08236500.DWR.Streamflow.Month")
```

Command Reference

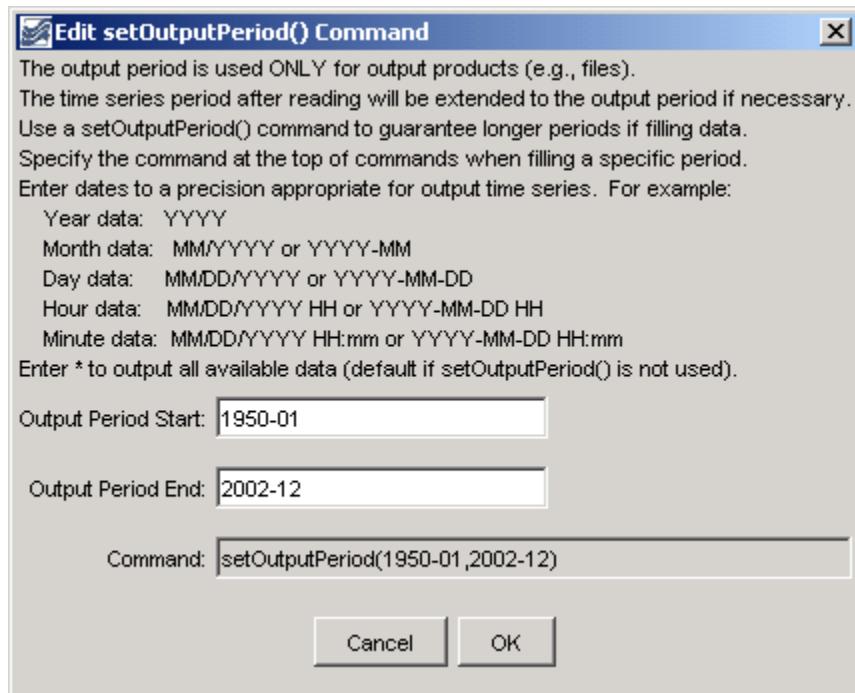
setOutputPeriod() Set the Output Period for Time Series

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `setOutputPeriod()` command sets the output period for time series. See also the `setQueryPeriod()` command. The period for a time series when read or created will be set to the maximum of the following periods, in order to satisfy output and data filling requirements:

- available,
- output (if specified),
- and query (if specified).

Specifying the output period is necessary when creating model files or filling an extended period (time series will not automatically be extended by fill commands). This command replaces the older `MM/YYYY` `MM/YYYY` syntax. The following dialog is used to edit this command and illustrates the syntax of the command. Note that the output period should always use calendar month and year, even if other than calendar year are used for output (see `setOutputYearType()`).



setOutputPeriod() Command Editor

setOutputPeriod

The command syntax is as follows:

```
setOutputPeriod(OutputStart,OutputEnd)
```

Command Parameters

Parameter	Description	Default
OutputStart	The date/time to start output.	None – must be specified.
OutputEnd	The date/time to end output.	None – must be specified.

A sample commands file is as follows:

```
setOutputPeriod(1950-01,2002-12)
```

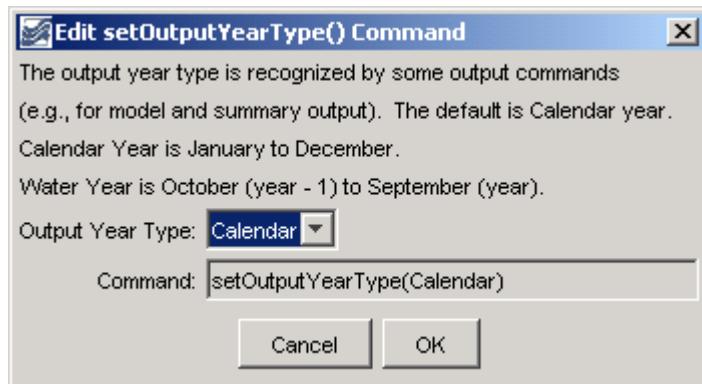
Command Reference

setOutputYearType()

Set the Output Year Type for Time Series

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `setOutputYearType()` command is used to define the global output year type for output reports and files. This command replaces the older `-wy` and `-cy` options. The output year type is recognized by some common tools, including the time series summary and `writeStateMod()` command. Write commands are being updated to allow the global year type to be reset for the specific command. The following dialog is used to edit the command and illustrates the command syntax.



setOutputYearType

setOutputYearType() Command Editor

The command syntax is as follows:

```
setOutputYearType(OutputYearType)
```

Command Parameters

Parameter	Description	Default
OutputYearType	The output year type, one of: <ul style="list-style-type: none">• Calendar – January through December.• Water – October through November. In the future, more generic types like JanToDec may be implemented.	If this command is not specified, Calendar is the default.

A sample commands file is as follows:

```
setOutputYearType(Calendar)
```

This page is intentionally blank.

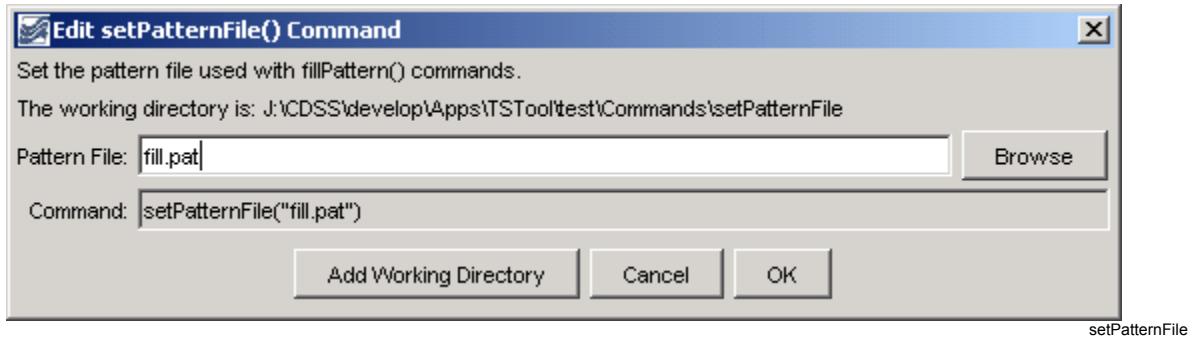
Command Reference

setPatternFile()

Set the Pattern File to be Used With fillPattern() Commands

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `setPatternFile()` command specifies a pattern file to be used with `fillPattern()` commands (see the `fillPattern()` command for more information). This command replaces the older `-filldata` option. The following dialog is used to edit the command and illustrates the command syntax.



The command syntax is as follows:

```
setPatternFile(PatternFile)
```

Command Parameters

Parameter	Description	Default
PatternFile	The path to the pattern file, which can be absolute or relative to the working directory. The Browse button can be used to select the pattern file (if a relative path is desired, remove the leading path after the select).	None – must be specified.

A sample commands file is as follows:

```
setPatternFile("fill.pat")
```

This page is intentionally blank.

Command Reference

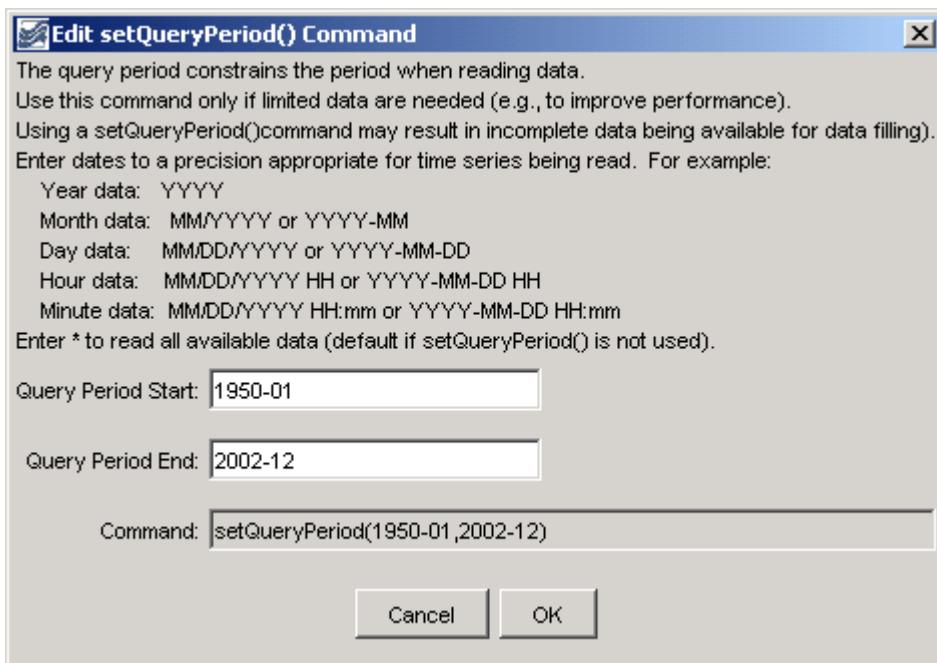
setQueryPeriod()

Set the Period for Queries

Version 06.14.00, 2005-12-14, Color, Acrobat Distiller

This command has been replaced by `setInputPeriod()`. Editing the command with the command editor dialog will convert `setQueryPeriod()` commands to `setInputPeriod()`.

The `setQueryPeriod()` command sets the period used for querying and reading time series data, from databases and files. The default is to query all available data so that all data are available for analysis and data filling. However, a shorter period may be desirable to increase performance (e.g., when processing real-time data). See also the `setOutputPeriod()` command. The following dialog is used to edit the command and illustrates the command syntax.



setQueryPeriod

setQueryPeriod() Command Editor

A sample commands file is as follows:

```
setQueryPeriod(1950-01,2002-12)
```

This page is intentionally blank.

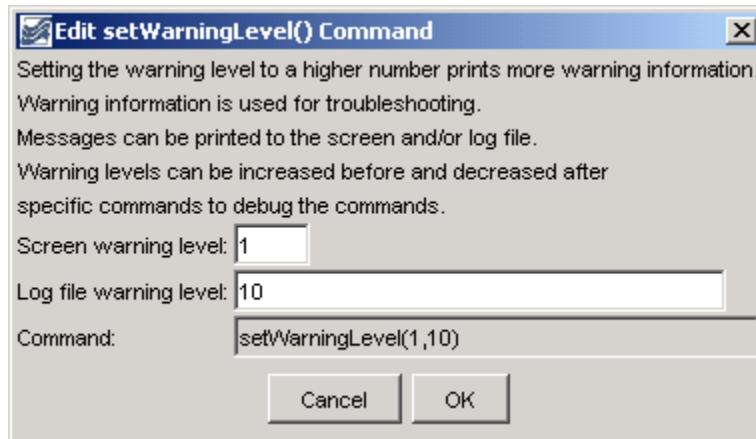
Command Reference

setWarningLevel()

Set Level for Warning Messages

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `setWarningLevel()` command sets the warning levels for the screen and log file. This command can be used multiple times, for example to isolate a problem. The following dialog is used to edit this command and illustrates the command syntax.



setWarningLevel() Command Editor

The command syntax is as follows:

```
setWarningLevel(ScreenLevel,LogFileLevel)
```

Command Parameters

Parameter	Description	Default
ScreenLevel	The warning level for the screen (0+).	None – must be specified. The default is warning level 1 to the screen and 2 to the log file.
LogFileLevel	The warning level for the log file (0+).	None – must be specified.

A sample commands file is as follows:

```
setWarningLevel(1,10)
```

This page is intentionally blank.

Command Reference

setWorkingDir()

Set Working Directory

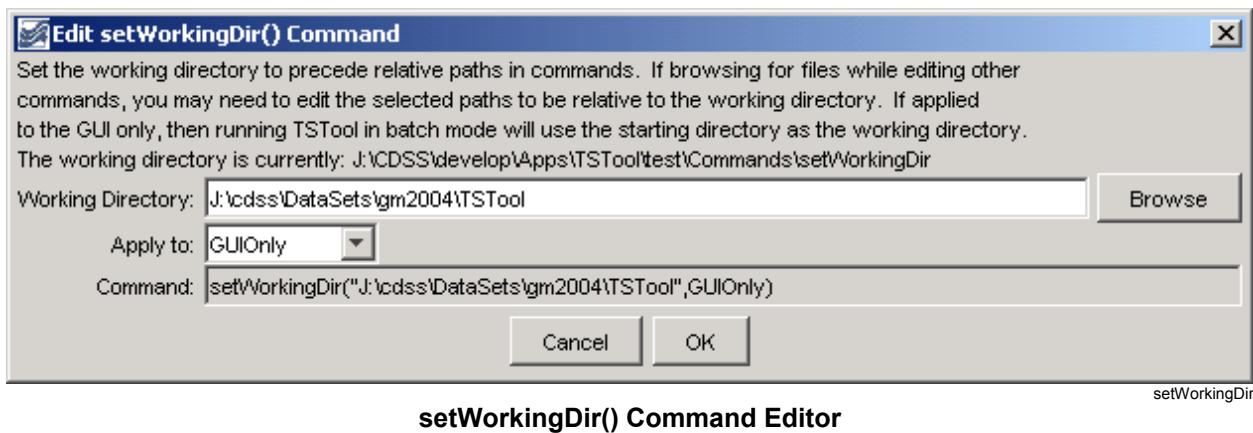
Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

This command should in most cases not be used.

The `setWorkingDir()` command is used to define the working directory for a set of commands. The working directory, when set properly, can greatly simplify commands files because relative file paths can be used for input and output. The working directory is normally set in one of the following ways, with the current setting being defined by the most recent item that has occurred:

1. the startup directory for the TSTool program,
2. the directory where a commands file was opened,
3. the directory where a commands file was saved,
4. the directory specified by a `setWorkingDir()` command,
5. the directory specified by ***File...Set Working Directory***.

In most cases, a `setWorkingDir()` command is not needed and should be avoided because it complicates commands. However, for complicated commands files, it may be necessary to change the working directory from one directory to another during processing. Setting the working directory to an absolute path causes all relative paths for input and output files to be appended to the working directory. The following dialog is used to edit the command and illustrates the syntax of the command.



setWorkingDir() Command Editor

The command syntax is as follows:

```
setWorkingDir(WorkingDir, SetWhen)
```

Command Parameters

Parameter	Description	Default
WorkingDir	<p>The working directory that should be used. The Browse button allows you to select the working directory (it is limited to selecting a file in the directory, but only the directory is used in the command).</p> <p>Currently the working directory cannot be a relative path like “..”, but this capability may be added in the future.</p>	None – must be specified.
RunMode	<p>The values of the Apply To field can be GUIOnly (the command applies only to the GUI) or GUIAndBatch (the command applies to GUI and batch runs). The former is useful because a commands file can be constructed that uses all relative paths and the <code>setWorkingDir()</code> command will tell the GUI where to find files but the batch run will be relative to the directory where it is run.</p>	None – must be specified.

A sample commands file is as follows:

```
setWorkingDir("J:\cdss\DataSet\gm2004\TSTool", GUIOnly)
```

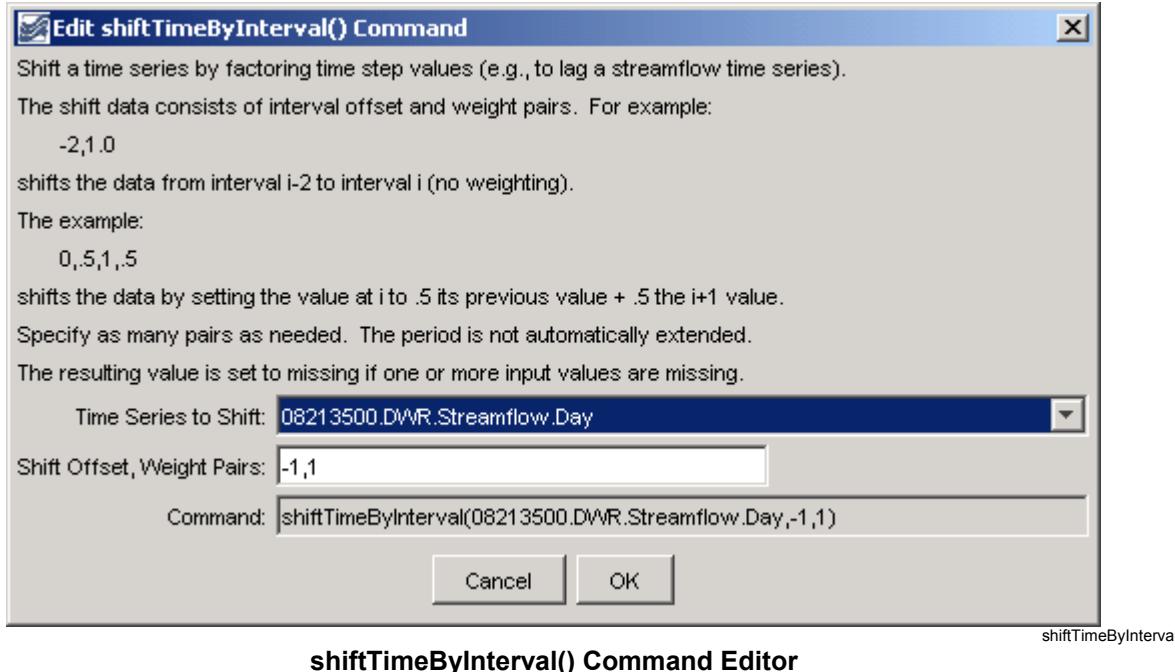
Command Reference

shiftTimeByInterval()

Shift Time Series Data by One or More Time Intervals

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `shiftTimeByInterval()` command shifts a time series in time. This command can be used to perform a simple shift (e.g., to shift hourly data because the `disaggregate()` command did not result in data being set at the desired hours) and to perform simple routing. The following dialog is used to edit the command and illustrates the command syntax.



shiftTimeByInterval() Command Editor

The command syntax is as follows:

```
shiftTimeByInterval(TSID, Interval, Multiplier, ...)
```

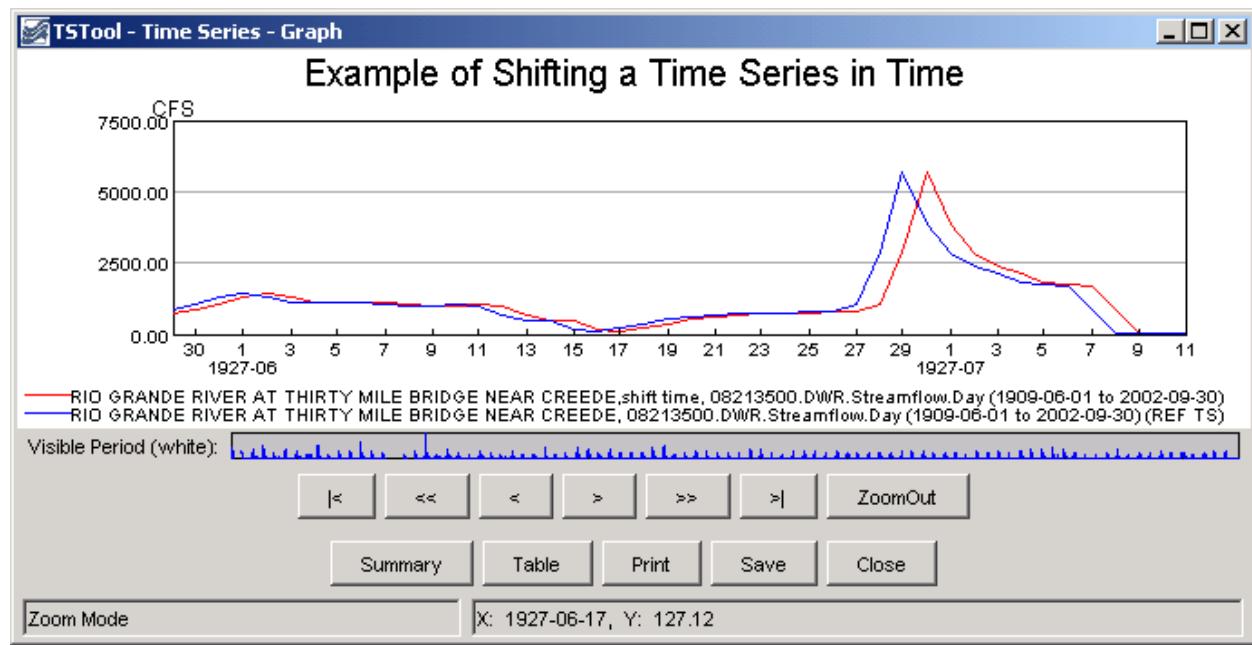
Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to be modified.	None – must be specified.
Interval	The interval to shift. For example, -1 indicates that the previous time step should be shifted to the current time step. Multiple Interval/Multiplier pairs can be specified.	None – at least 1 value must be specified.
Multiplier	The multiplier to apply to the value being shifted. For example, if the Interval is -1 and the Multiplier is 1, the previous time step is shifted to the current and multiplied by 1, effectively shifting the time series by one interval.	None – at least 1 value must be specified.

A sample commands file is as follows:

```
# 08213500 - RIO GRANDE RIVER AT THIRTY MILE BRIDGE NEAR CREEDE
08213500.DWR.Streamflow.Day~HydroBase
shiftTimeByInterval(08213500.DWR.Streamflow.Day,-1,1)
08213500.DWR.Streamflow.Day~HydroBase
```

The resulting graph is as follows:



Results from shiftTimeByInterval() Command

shiftTimeByInterval_graph

Command Reference

sortTimeSeries()

Sort the time series by their identifiers

Version 06.10.00, 2005-05-22, Color, Acrobat Distiller

The `sortTimeSeries()` command sorts the time series alphabetically using the time series identifier. This command is useful for ordering time series before writing output, for example to facilitate comparison with another version of the output or to be consistent with other data files.

The following dialog is used to edit the command and illustrates the syntax for the command.



The command syntax is as follows:

```
sortTimeSeries(param=value,...)
```

Command Parameters

Parameter	Description	Default
	Currently no parameters are available for this command.	

A sample commands file is as follows:

```
sortTimeSeries()
```

This page is intentionally blank.

Command Reference

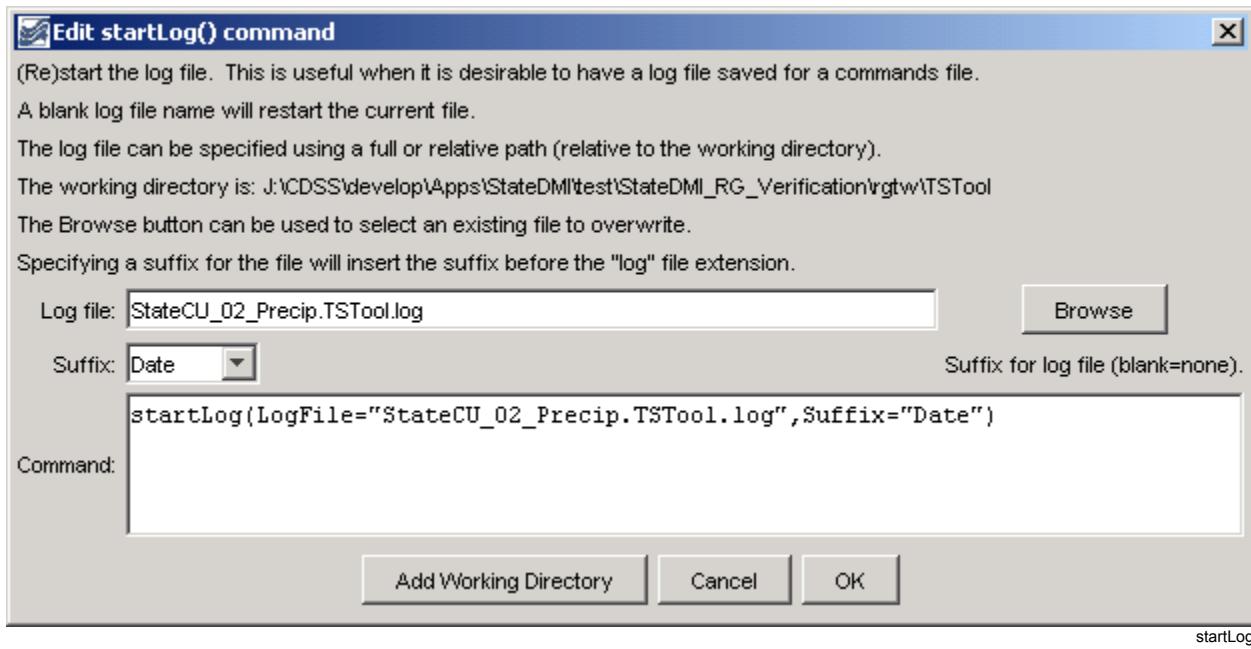
startLog()

(Re)start the log file

Version 06.10.00, 2005-05-23, Color, Acrobat Distiller

The `startLog()` command (re)starts the log file. It is useful to insert this command as the first command in a log file, in order to persistently record the results of processing. A useful standard is to name the log file the same as the commands file, with an additional `.log` extension. A date or date/time can optionally be added to the log file name.

The following dialog is used to edit the command and illustrates the syntax for the command.



startLog() Command Editor

The command syntax is as follows:

```
startLog(param=value,...)
```

Command Parameters

Parameter	Description	Default
LogFile	The name of the log file to write, surrounded by double quotes. The extension of <i>.log</i> will automatically be added, if not specified.	If not specified, the existing file will be restarted.
Suffix	Indicates that a suffix will be added before the <i>.log</i> extension, one of: <ul style="list-style-type: none">▪ Date – add a date suffix of the form YYYYMMDD.▪ DateTime – add a date/time suffix of the form YYYYMMDD_HHMMSS. This is useful for automatically archiving logs corresponding to commands files, to allow checking the output at a later time.	Do not add the suffix.

A sample commands file is as follows:

```
startLog(LogFile="StateCU_02_Precip.TSTool.log",Suffix="Date")
```

Command Reference

statemodMax()

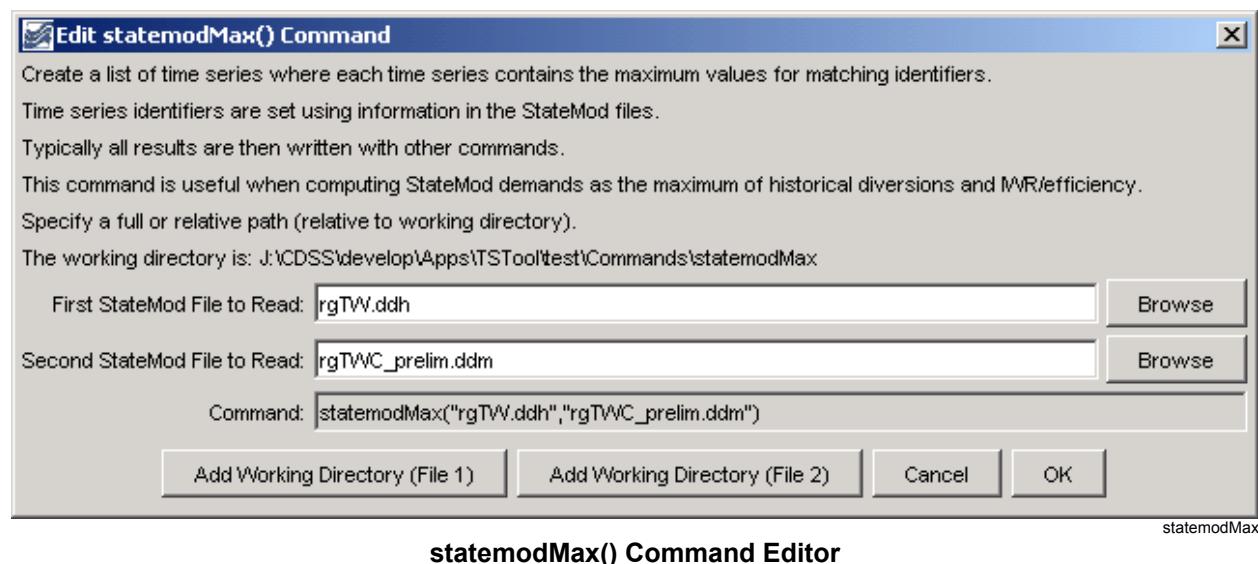
Compute the Maximum of Time Series in Two StateMod Files

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

A statemodMax() command performs the following actions:

- read all time series from one StateMod time series file,
- read all time series from a second StateMod time series file,
- generate a list of time series in memory that contains the maximum values comparing matching time series (using the location identifier).

This command is useful, for example, when creating a demand time series file that is to be the maximum of historical diversions and IWR divided by an average efficiency. It is assumed that the specified time series have matching identifiers (the first file is used as the master list) and have consistent units and data intervals. After the time series have been processed, they can be viewed or written out as a new StateMod file (see the writeStateMod() command). The following dialog is used to edit the command and illustrates the syntax for the command.



statemodMax() Command Editor

The command syntax is as follows:

```
statemodMax( InputFile1, InputFile2 )
```

Command Parameters

Parameter	Description	Default
InputFile1	The name of the first StateMod time series file to read, surrounded by double quotes. The path to the file can be absolute or relative to the working directory. The Browse buttons can be used to select the file to read (if a relative path is desired, remove the leading path after the select).	None – must be specified.
InputFile2	The name of the second StateMod time series file to read, which must have the same data interval and units as the first file.	None – must be specified.

A sample commands file is as follows:

```
statemodMax( "rgTW.ddh" , "rgTWC_prelim.ddm" )
writeStateMod( "rgTW.ddm" , * )
```

Command Reference

subtract()

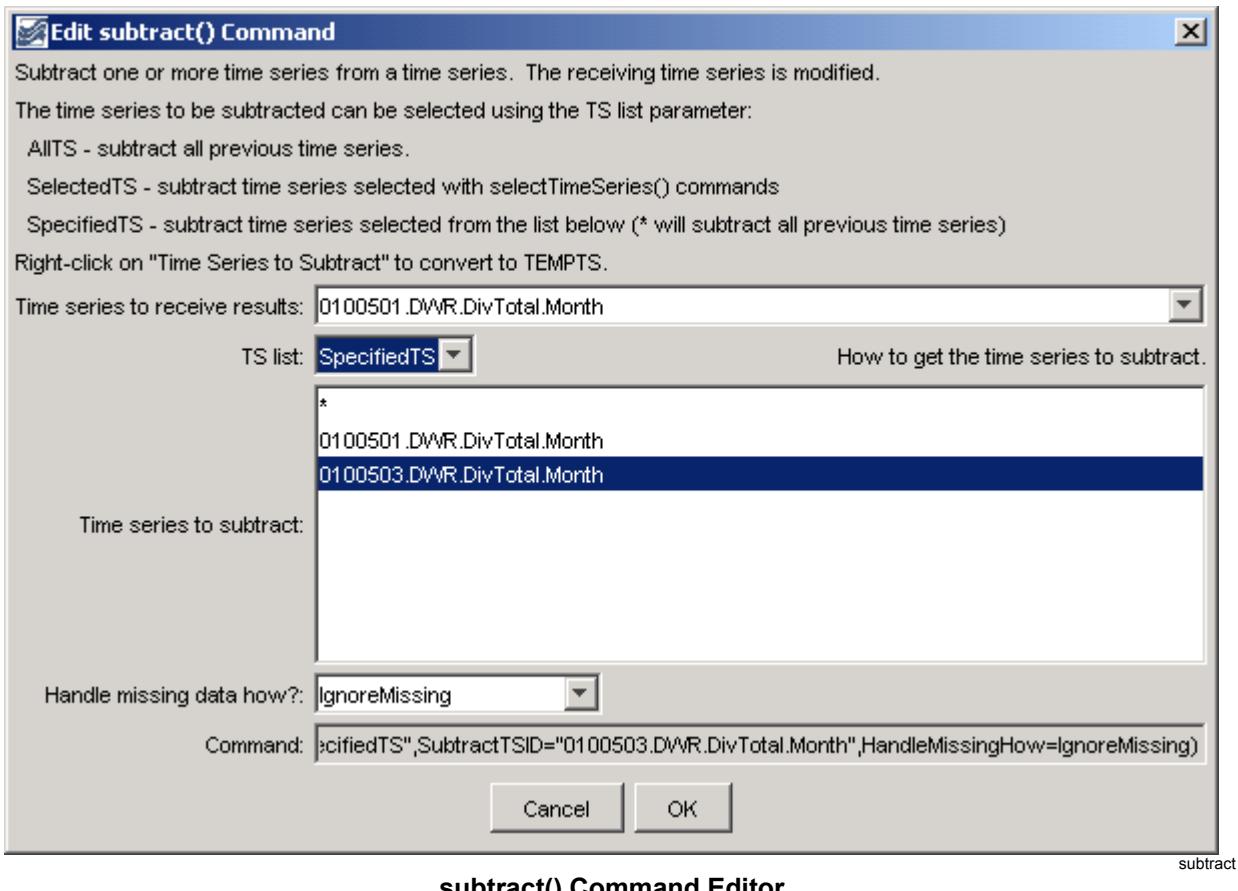
Subtract One or More Time Series From Another Time Series

Version 06.10.00, 2005-02-23, Color, Acrobat Distiller

The `subtract()` command subtracts time series of the same interval. The receiving time series will have data values set to its original values minus the data values in the indicated time series. This command replaces the older `subtract()` command, which did not have the flag indicating how to handle missing data.

This command will generate an error if the time series do not have compatible units. If the units are compatible but are not the same (e.g., IN and FT), then the units of the part will be converted to the units of the result before subtraction. Missing data in the parts can be ignored (do not set the result to missing) or can set missing values in the result. The user should consider the implications of ignoring missing data. Time series being subtracted must have the same data interval.

The following dialog is used to edit the command and illustrates the syntax of the command.



subtract() Command Editor

The command syntax is as follows:

```
subtract(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSID	The time series identifier or alias for the time series to receive the result.	None – must be specified.
TSList	Indicates how the list of time series is specified, one of: <ul style="list-style-type: none"> • AllTS – all time series prior to the command. • SelectedTS – the time series are those selected with the selectTimeSeries() command. • SpecifiedTS – the specified list of time series given by the SubtractTSID parameter. 	AllTS
SubtractTSID	If the TSList parameter is SpecifiedTS, provide the list of time series identifiers (or alias) to subtract. Right-clicking on a time series allows toggling of the TEMPTS key word. This will cause the time series to subtract to be read and then discarded.	Must be specified if TSList=SpecifiedTS, ignored otherwise.
HandleMissingHow	Indicates how to handle missing data in a time series, one of: <ul style="list-style-type: none"> • IgnoreMissing – create a result even if missing data are encountered in one or more time series – this option is not as rigorous as the others • SetMissingIfOtherMissing – set the result missing if any of the other time series values is missing • SetMissingIfAnyMissing – set the result missing if any time series value involved is missing 	IgnoreMissing

A sample commands file is as follows:

```
# 0100501 - EMPIRE DITCH
0100501.DWR.DivTotal.Month~HydroBase
# 0100503 - RIVERSIDE CANAL
0100503.DWR.DivTotal.Month~HydroBase
subtract(TSID="0100501.DWR.DivTotal.Month",TSList="SpecifiedTS",
SubtractTSID="0100503.DWR.DivTotal.Month",HandleMissingHow=IgnoreMissing)
```

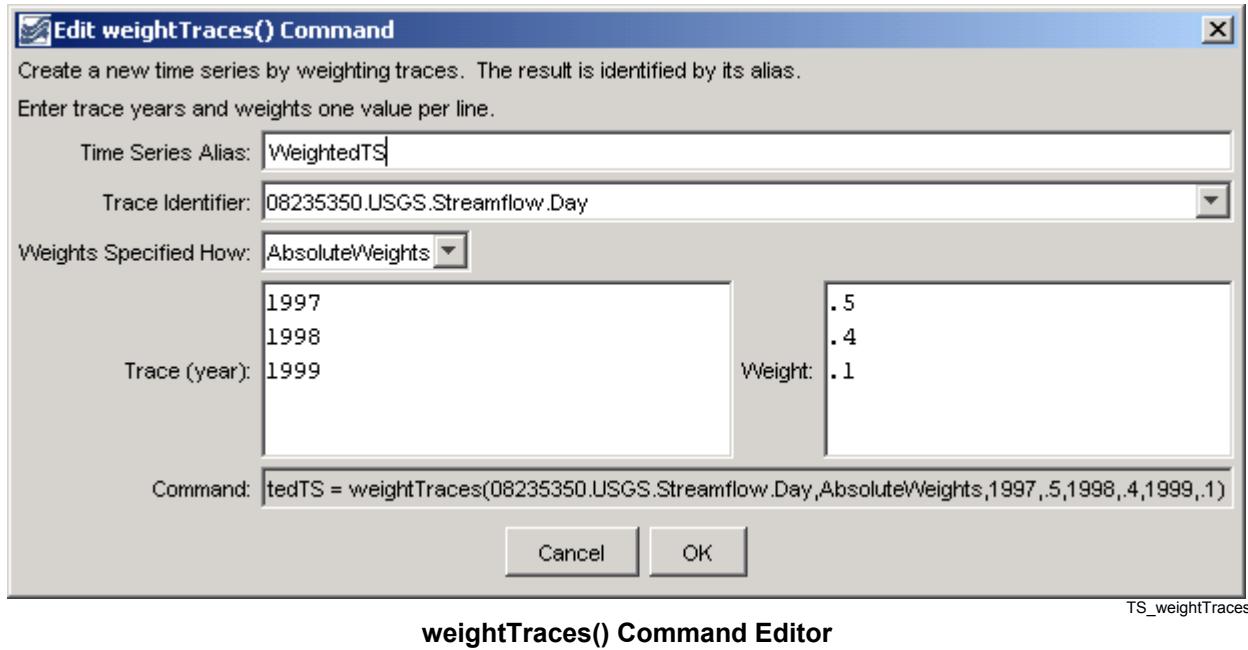
Command Reference

TS X = weightTraces()

Create a Time Series by Weighting Data from Time Series Traces

Version 06.08.02, 2004-07-29, Color, Acrobat Distiller

The `weightTraces()` command creates a new time series as a weighted sum of time series traces, which were produced by a `createTraces()` command. The following dialog is used to edit the command and illustrates the syntax of the command. Most often when using this command you will have interactively reviewed available traces to decide which traces should be weighted.



weightTraces() Command Editor

The command syntax is as follows:

```
TS X = weightTraces(TSID,SpecifWeightsHow,Year,Weight,...)
```

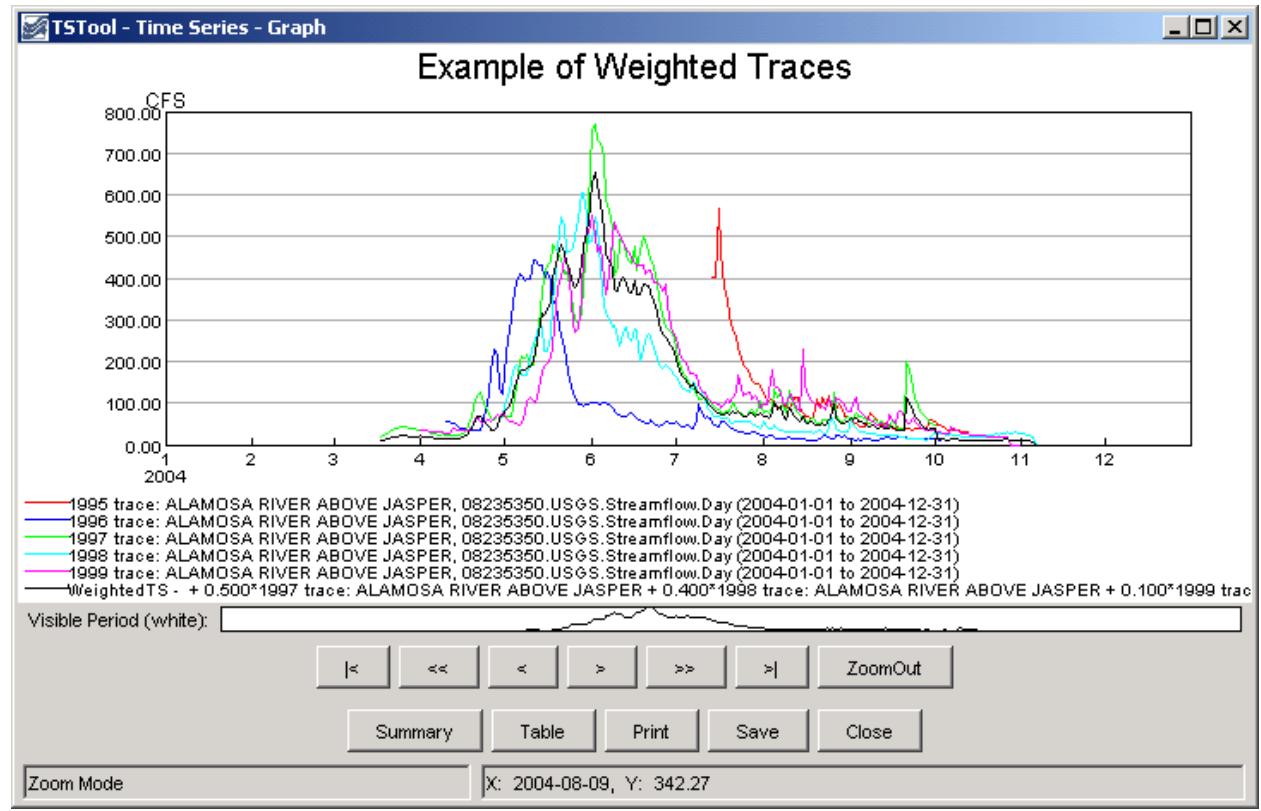
Command Parameters

Parameter	Description	Default
X	The alias of the new time series.	None – must be specified.
TSID	The time series identifier or alias for a time series that has been converted into traces (e.g., from a <code>createTraces()</code> command). Time series matching the identifier will be checked.	None – must be specified.
SpecifyWeightsHow	Weights are currently only applied as <code>AbsoluteWeights</code> (in the future an option may be added to normalized weights to 1.0 accounting for missing data in the traces).	Must be <code>AbsoluteWeights</code> .
Year,Weight	Specify pairs of trace year and weights (0-1.0), used to create the new time series. Trace years must be manually entered because at the time that the command is edited, time series have not yet been queried.	None – must be specified.

A sample commands file is as follows:

```
# Create annual traces from a time series shifted to the current year
#
# (1995-1998) ALAMOSA RIVER ABOVE JASPER, CO  USGS  Streamflow  Day
08235350.USGS.Streamflow.Day~HydroBase
createTraces(08235350.USGS.Streamflow.Day,1Year,2004-01-01,ShiftToReference)
TS WeightedTS =
  weightTraces(08235350.USGS.Streamflow.Day,AbsoluteWeights,1997,.5,1998,.4,1999,.1)
```

The results from the commands are shown in the following graph:



Results of the weightTraces() Command

This page is intentionally blank.

Command Reference

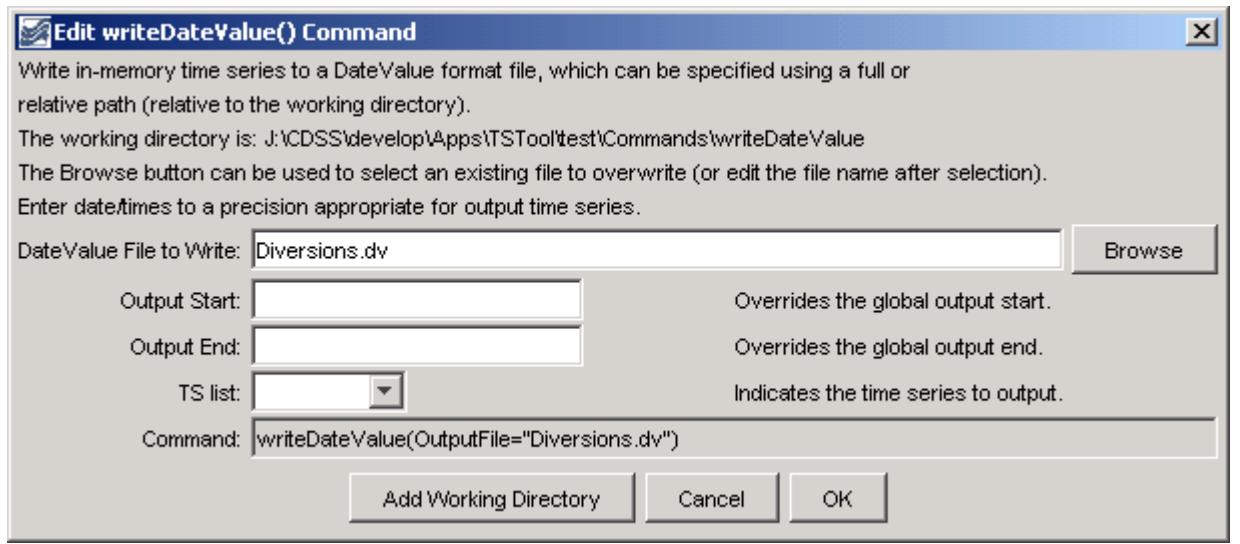
writeDateValue()

Write Time Series to a DateValue Format File

Version 06.08.02, 2004-08-02, Color, Acrobat Distiller

The `writeDateValue()` command will write time series in memory to the specified DateValue format file. See the **DateValue Input Type** appendix for more information about the file format. The time series should have the same data interval.

The following dialog is used to edit the command and illustrates the syntax of the command.



writeDateValue() Command Editor

The command syntax is as follows:

```
writeDateValue(param=value,...)
```

Command Parameters

Parameter	Description	Default
OutputFile	The DateValue output file. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to write (if a relative path is desired, delete the leading path after the select).	None – must be specified.
OutputStart	The date/time for the start of the output.	Use the global output period.
OutputEnd	The date/time for the end of the output.	Use the global output period.
TSList	Indicates the time series to be output, one of: <ul style="list-style-type: none"> • AllTS – all available time series will be output. • SelectedTS – time series that have been selected with the <code>selectTimeSeries()</code> command will be output. 	AllTS

A sample commands file is as follows:

```
# 0100501 - EMPIRE DITCH
0100501.DWR.DivTotal.Month~HydroBase
# 0100503 - RIVERSIDE CANAL
0100503.DWR.DivTotal.Month~HydroBase
writeDateValue(OutputFile="Diversions.dv")
```

Command Reference

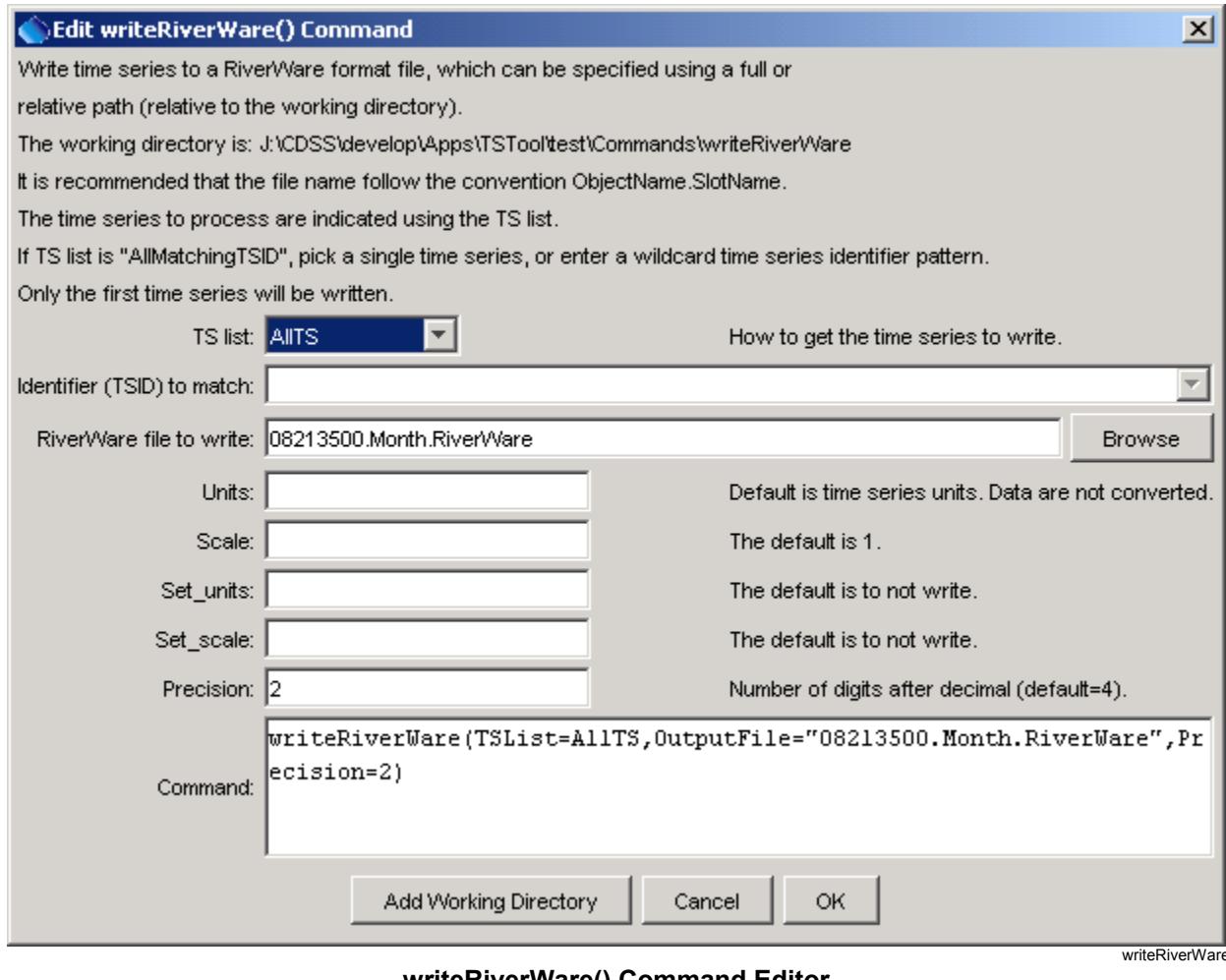
writeRiverWare()

Write Time Series to a RiverWare Format File

Version 06.10.06, 2005-08-04, Color, Acrobat Distiller

The `writeRiverWare()` command writes the first time series in memory to the specified RiverWare format file. See the **RiverWare Input Type** appendix for more information about the file format.

The following dialog is used to edit the command and illustrates the syntax of the command.



writeRiverWare() Command Editor

writeRiverWare

The command syntax is as follows:

```
writeRiverWare(param=value, ...)
```

Command Parameters

Parameter	Description	Default
TSList	Indicate how to determine the list of time series to process, one of: <ul style="list-style-type: none"> ▪ AllMatchingTSID – process time series that have identifiers matching the TSID parameter. ▪ AllTS – process all the time series. ▪ SelectedTS – process the time series that are selected (see <code>selectTimeSeries()</code>). 	None – must be specified.
TSID	Used if <code>TSList=AllMatchingTSID</code> to indicate the time series identifier or alias for the time series to be filled. Specify * to match all time series or use a wildcard for one or more identifier parts.	Required if <code>TSList=AllMatchingTSID</code> .
OutputFile	The RiverWare file to write. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to write (if a relative path is desired, delete the leading path after the select).	None – must be specified.
Units	The data units to be output. Specify units that are recognized by RiverWare – the units are not actually converted but the new units string is used in the output file.	Use the units in the time series.
Scale	See the RiverWare Input Type appendix.	1
Set_units	See the RiverWare Input Type appendix.	Set_units are not output.
Set_scale	See the RiverWare Input Type appendix.	Set_scale are not output.
Precision	The number of digits after the decimal to write.	4

A sample commands file is as follows:

```
# 08213500 - RIO GRANDE RIVER AT THIRTY MILE BRIDGE NEAR CREEDE
08213500.DWR.Streamflow.Month~HydroBase
writeRiverWare(TSList=AllTS,OutputFile="08213500.Month.RiverWare",Precision=2)
```

Command Reference

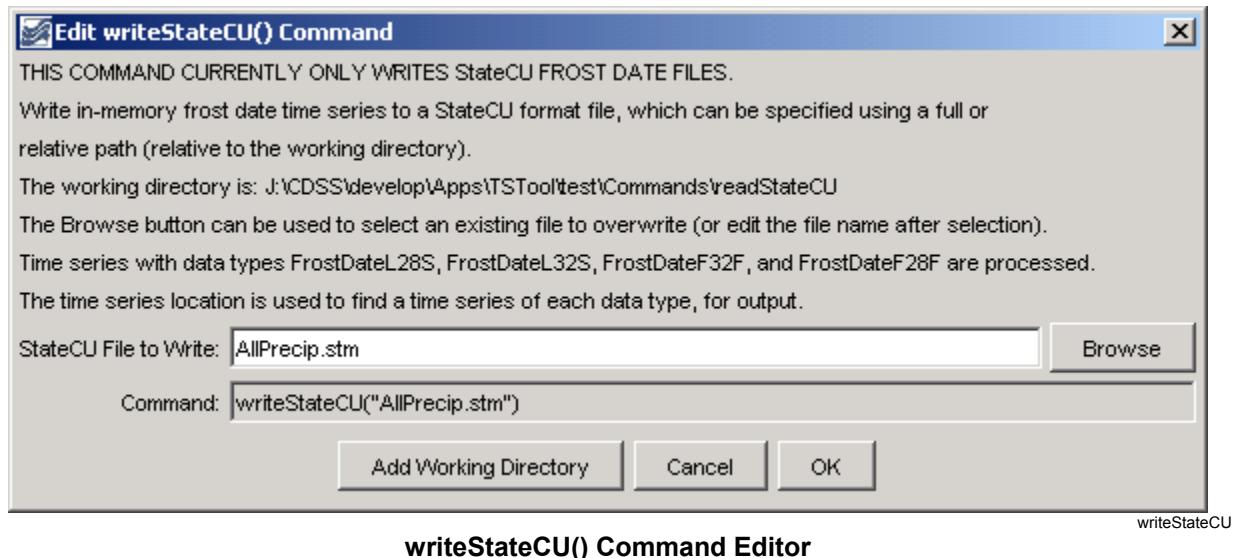
writeStateCU()

Write time series to a StateMod format file

Version 06.10.08, 2005-09-22, Color, Acrobat Distiller

The `writeStateCU()` command writes the time series in memory to the specified StateCU frost dates format file. **Currently only the frost dates file can be written with this command.** See the `writeStateMod()` command to write StateMod format files (e.g., for the precipitation, temperature, and diversion time series files used with the StateCU model). See the **StateCU Input Type** appendix for more information about the file format.

The following dialog is used to edit the command and illustrates the syntax of the command.



writeStateCU() Command Editor

The command syntax is as follows:

```
writeStateCU(OutputFile)
```

Command Parameters

Parameter	Description	Default
OutputFile	The StateCU frost dates output file. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to write (if a relative path is desired, delete the leading path after the select).	None – must be specified.

A sample commands file is as follows:

```
# 0109 - AKRON 4 E
0109.NOAA.FrostDateL28S.Year~HydroBase
0109.NOAA.FrostDateL32S.Year~HydroBase
0109.NOAA.FrostDateF32F.Year~HydroBase
0109.NOAA.FrostDateF28F.Year~HydroBase
writeStateCU( "test.stm" )
```

Command Reference

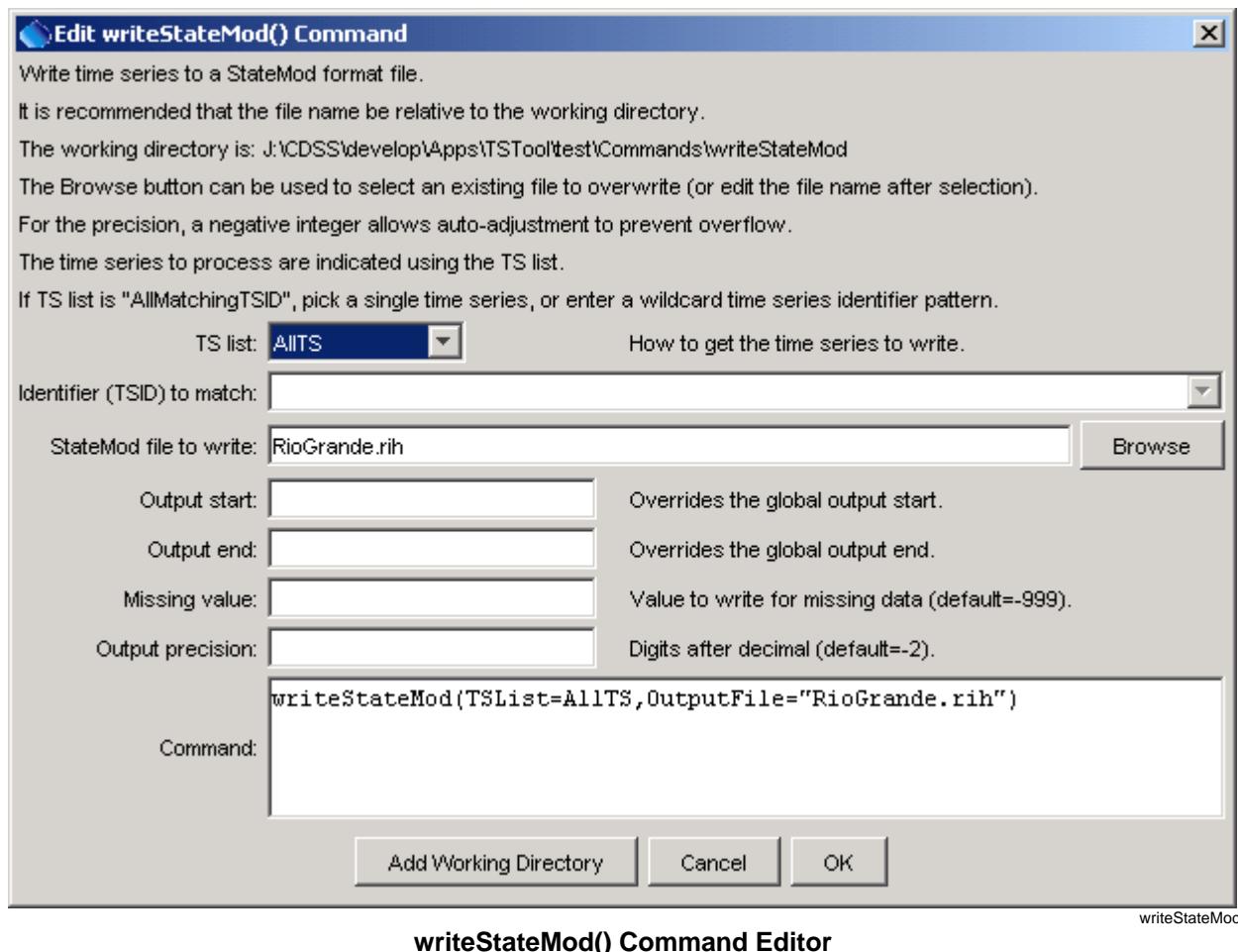
writeStateMod()

Write time series to a StateMod format file

Version 06.17.00, 2006-05-01, Color, Acrobat Distiller

The `writeStateMod()` command writes the time series in memory to the specified StateMod format file. See the **StateMod Input Type** appendix for more information about the file format. It is expected that the time series have the same interval. The time series identifier location part is written as the identifier, even if an alias is assigned to a time series.

The following dialog is used to edit the command and illustrates the syntax of the command.



writeStateMod() Command Editor

writeStateMod

The command syntax is as follows:

```
writeStateMod(param=value,...)
```

Command Parameters

Parameter	Description	Default
TSList	Indicate how to determine the list of time series to process, one of: <ul style="list-style-type: none"> ▪ AllMatchingTSID – process time series that have identifiers matching the TSID parameter. ▪ AllTS – process all the time series. ▪ SelectedTS – process the time series that are selected (see <code>selectTimeSeries()</code>). 	None – must be specified.
TSID	Used if TSList=AllMatchingTSID to indicate the time series identifier or alias for the time series to be filled. Specify * to match all time series or use a wildcard for one or more identifier parts.	Required if TSList=AllMatchingTSID.
OutputFile	The StateMod file to write. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to write (if a relative path is desired, delete the leading path after the select).	None – must be specified.
OutputStart	The date/time for the start of the output.	Use the global output period.
OutputEnd	The date/time for the end of the output.	Use the global output period.
MissingValue	The value to write for missing data.	-999
Precision	The number of digits to use after the decimal point, for data values. A negative number indicates that if the formatted number is larger than the allowed output width, adjust the format accordingly by truncating fractional digits. A special value of -2001 is equivalent to -2 and additionally NO decimal point will be printed for large values.	The default output precision if not specified is -2, which is then reset based on the data units (see the <i>system\DATAUNIT</i> file).

A sample commands file is as follows:

```
setOutputPeriod(1950-01,2002-12)
# 08213500 - RIO GRANDE RIVER AT THIRTY MILE BRIDGE NEAR CREEDE
08213500.DWR.Streamflow.Month~HydroBase
# 08217000 - RIO GRANDE AT WASON, BELOW CREEDE, CO.
08217000.USGS.Streamflow.Month~HydroBase
writeStateMod(TSList=AllTS,OutputFile="RioGrande.rih")
```

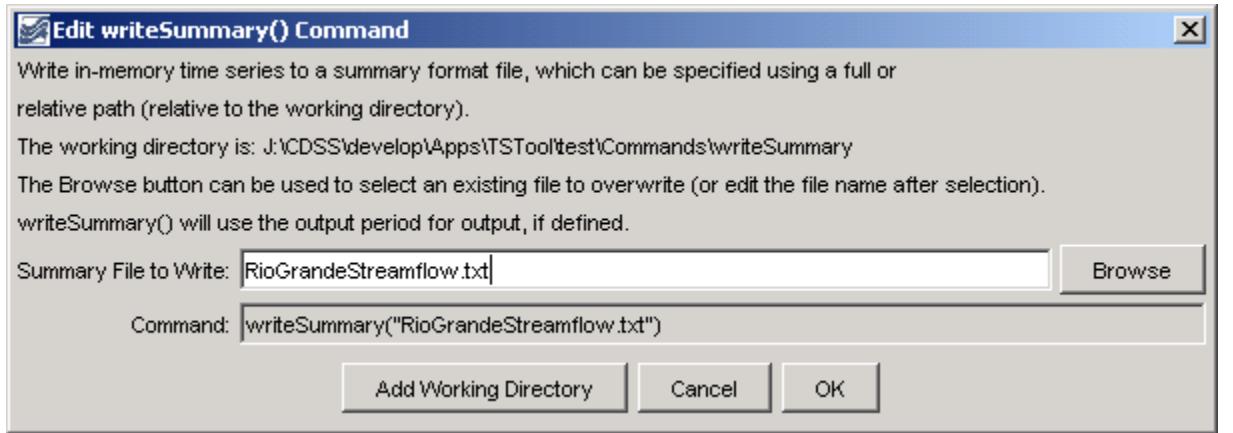
Command Reference

writeSummary()

Write Time Series to a Summary Format File

Version 06.08.02, 2004-08-03, Color, Acrobat Distiller

The `writeSummary()` command writes the time series in memory to a summary report file. The format of the file is a default for the data interval. The following dialog is used to edit the command and illustrates the syntax of the command.



writeSummary() Command Editor

The command syntax is as follows:

```
writeSummary(OutputFile)
```

Command Parameters

Parameter	Description	Default
OutputFile	The summary file. The path to the file can be absolute or relative to the working directory. The Browse button can be used to select the file to write (if a relative path is desired, delete the leading path after the select).	None – must be specified.

A sample commands file is as follows:

```
setOutputPeriod(1950-01,2002-12)
# 08213500 - RIO GRANDE RIVER AT THIRTY MILE BRIDGE NEAR CREEDE
08213500.DWR.Streamflow.Month~HydroBase
# 08217000 - RIO GRANDE AT WASON, BELOW CREEDE, CO.
08217000.USGS.Streamflow.Month~HydroBase
writeSummary("RioGrandeStreamflow.txt")
```

This page is intentionally blank.

Appendix

TSTool Installation and Configuration for CDSS

CDSS Version, 07.00.00, 2006-12-10, Acrobat Distiller

1. Overview

This appendix describes how to install TSTool in the CDSS (Colorado's Decision Support Systems) environment. The State of Colorado's CDSS consists of the HydroBase database, modeling, and data viewing/editing software. TSTool can be used within this system to process time series from the HydroBase database, CDSS model files, and other databases and files.

2. File Locations

Locations of TSTool software files within the CDSS main folder are as shown below. The _XXX notation indicates the JRE [Java Runtime Environment] version number (e.g., _142), which may change as upgrades to the system occur.

\CDSS\	Top-level CDSS install directory.
bin\	Directory for <i>TSTool.exe</i> , <i>TSTool.bat</i> file and Java JAR files.
<i>Blowfish_XXX.jar</i>	Used for encryption/security.
<i>HydroBaseDMI_XXX.jar</i>	State of Colorado HydroBase database interface package.
<i>msbase.jar</i> , <i>mssqlserver.jar</i> ,	Microsoft SQL Server packages (see special installation instructions below).
<i>msutil.jar</i>	
<i>NWSRFS_DMI_XXX.jar</i>	National Weather Service River Forecast System (NWSRFS) package.
<i>RiversideDB_DMI_XXX.jar</i>	Riverside Technology, inc., RiversideDB database package to support RiverTrak® systems.
<i>RTi_Common_XXX.jar</i>	Riverside Technology, inc. supporting packages.
<i>SatmonSysDMI_XXX.jar</i>	State of Colorado's Satellite Monitoring System package.
<i>shellcon.exe</i>	Executable program used to read from the Windows registry (e.g., to determine the default web browser and list available ODBC data source names).
<i>StateCU_XXX.jar</i>	State of Colorado's StateCU model package.
<i>StateMod_XXX.jar</i>	State of Colorado's StateMod model package.
<i>TSTool.bat</i>	Batch file to run TSTool using the JRE software. You may need to edit this if the installation is not standard.
<i>TSTool_XXX.jar</i>	TSTool main application package.
\CDSS\doc\TSTool\UserManual\	Main documentation directory for TSTool.
<i>TSTool.pdf</i>	TSTool documentation as PDF.
\CDSS\JRE_XXX\	Java Runtime Environment.
\CDSS\logs\	Directory for TSTool log files (should be writeable).
\CDSS\system\	Directory for system files.
<i>CDSS.cfg</i>	Optional configuration file for CDSS, impacting the HydroBase login defaults (see below).
<i>DATAUNIT</i>	Data units file (see below).

TSTool.cfg

Configuration file to modify TSTool defaults (see below).

3. Installing TSTool

TSTool can be installed either as part of the HydroBase Tools CD/DVD installation, or as a separate installation. In both cases, it is recommended that the CDSS file structure be used.

3.1 Installing TSTool from the “HydroBase data set Analysis Query Tools CD/DVD”

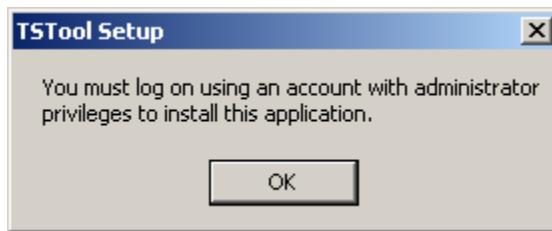
If you have purchased a HydroBase CD/DVD, TSTool will be installed during the CD install process. Refer to the installation instructions for that distribution.

3.2 Installing TSTool from the TSTool Setup File

Use the following instructions to install TSTool using the *TSTool_Setup.exe* installer program, for example if TSTool software was downloaded from the CDSS web site (<http://cdss.state.co.us>):

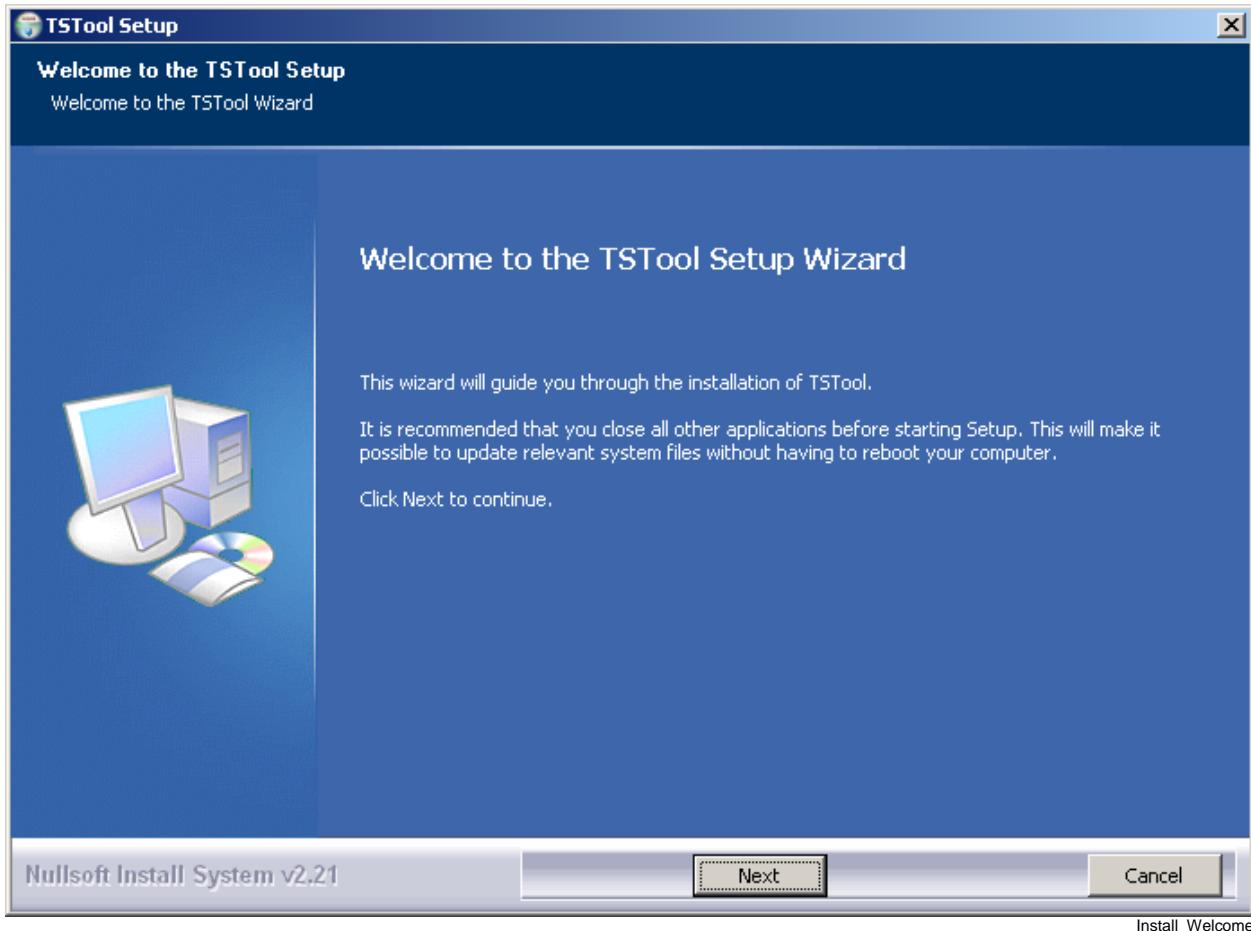
1. Run the *TSTool_Setup.exe* file by selecting from Windows Explorer, the **Start... Run...** menu, or from a command shell.

You must be logged into the computer using an account with administrator privileges. Otherwise, the following warning will be displayed:

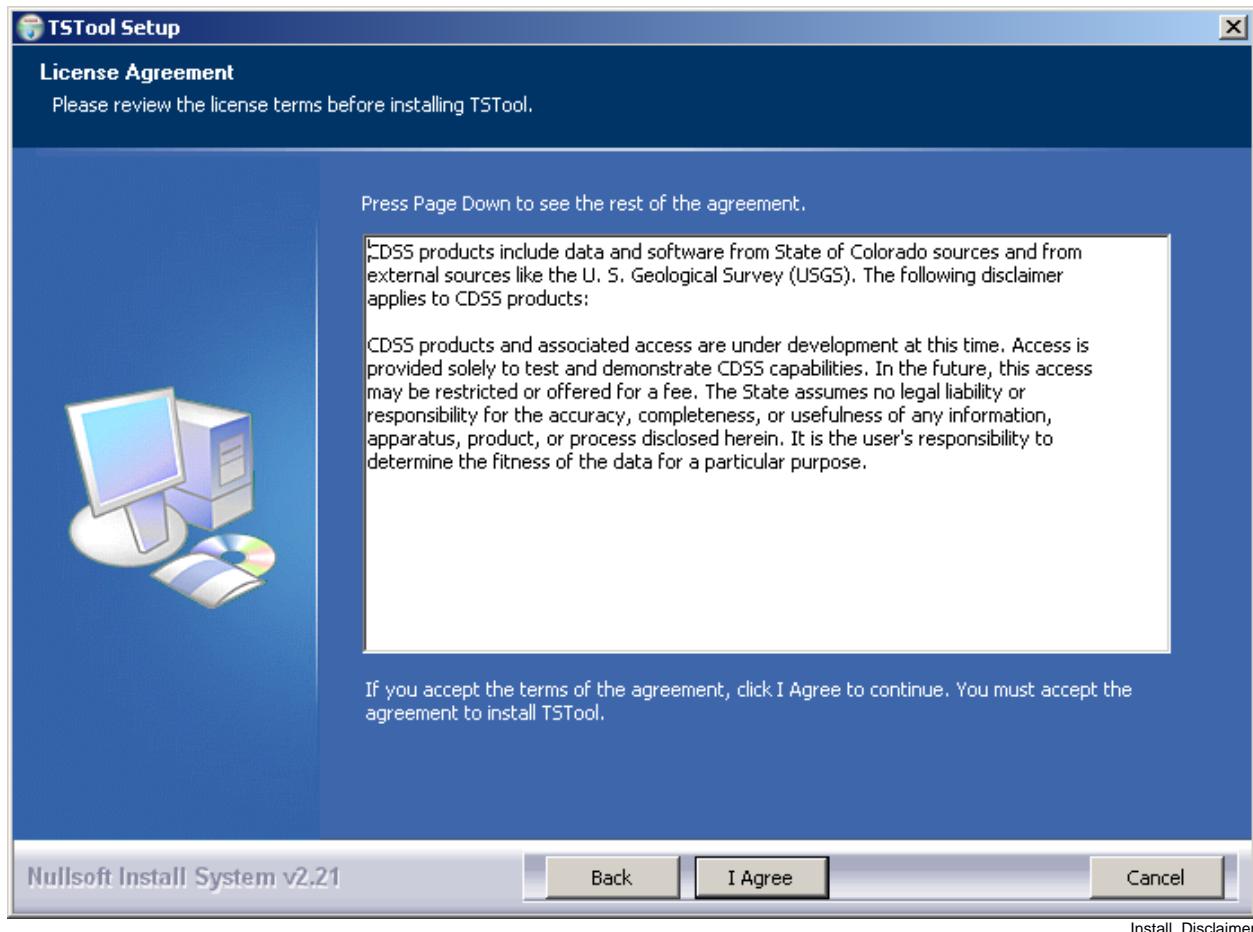


Install_AdministratorWarning

If you have administrative privileges, the following welcome will be displayed, and the installation can continue:

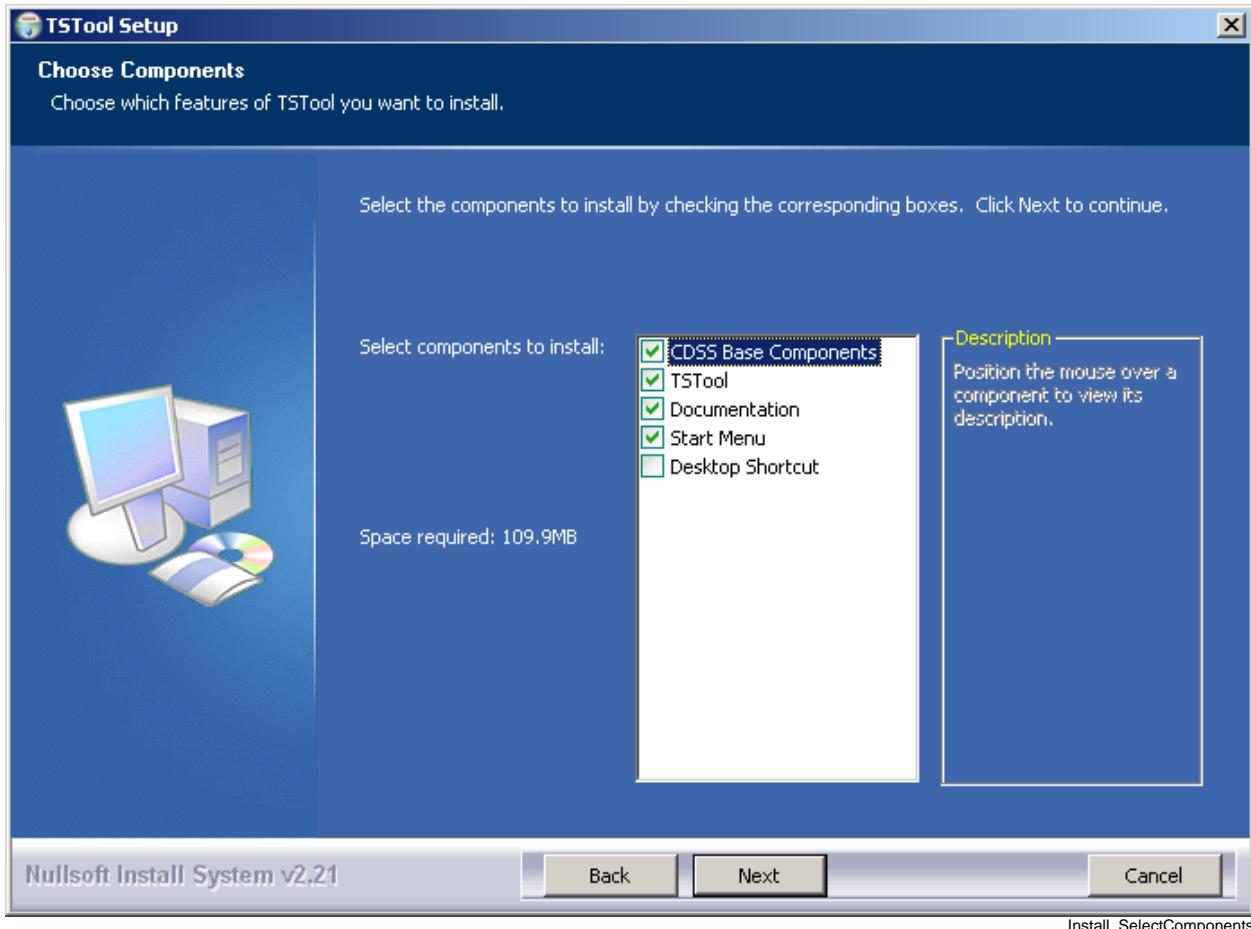


Press **Next** to continue with the installation.



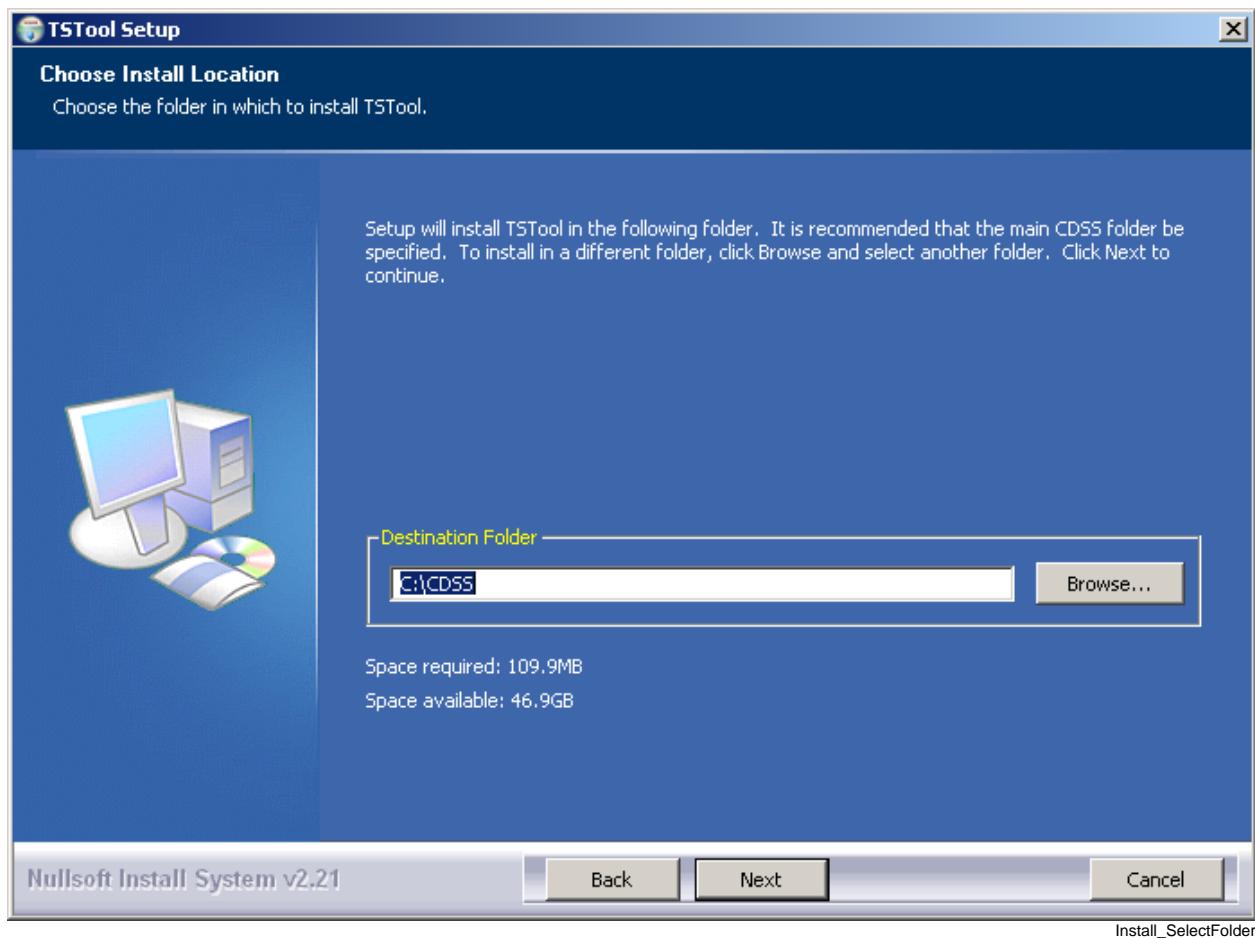
TSTool is distributed with CDSS with no license restrictions. However the disclaimer must be acknowledged. Press **I Agree** to continue with the installation.

2. Several components can be selected for the install as shown in the following dialog. Position the mouse over a component to see its description.



Select the components to install and press **Next**.

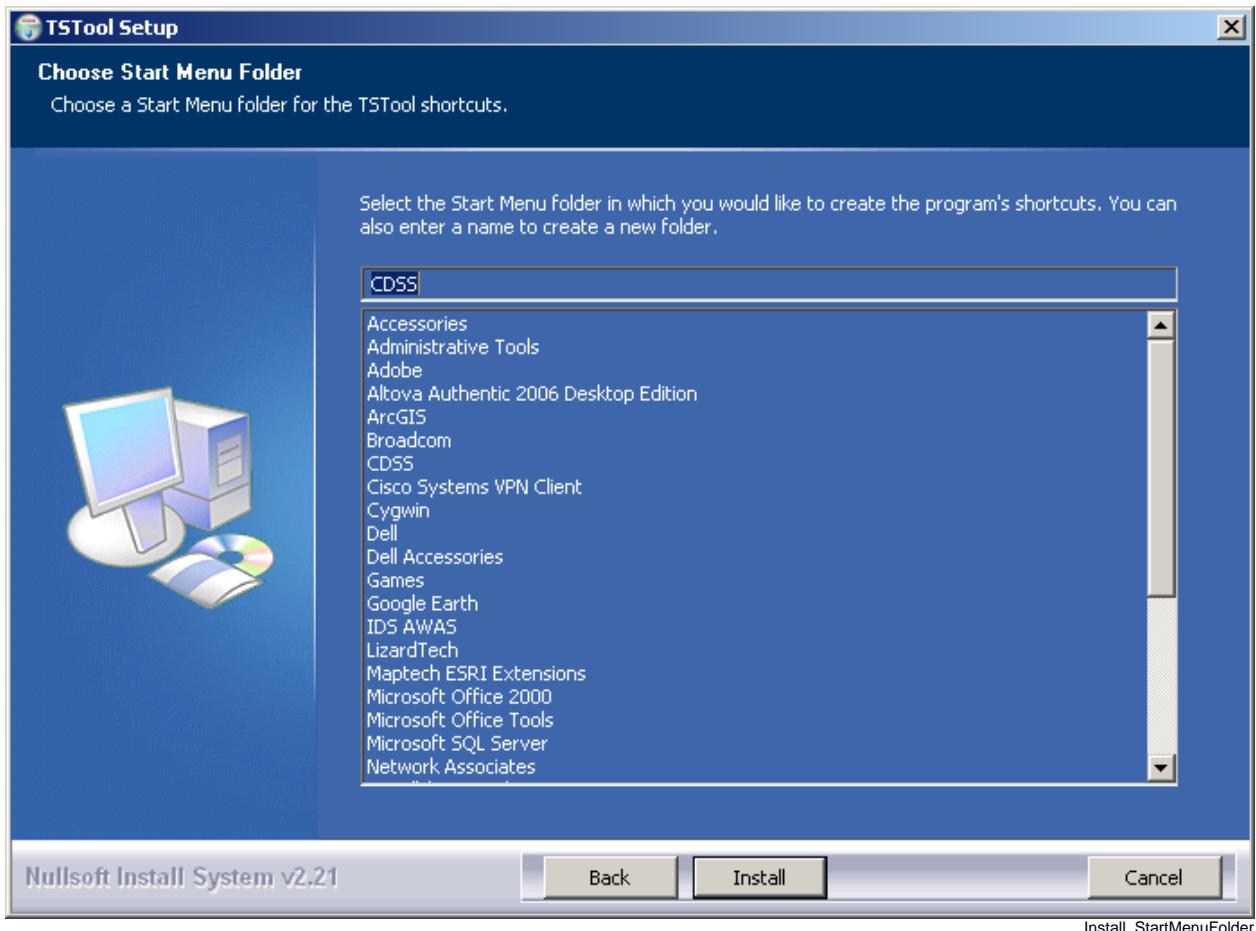
3. The following dialog is then shown and is used to select the installation location for TSTool. To be consistent with other CDSS components, select the main CDSS folder. The following dialog will display the CDSS install location if the CDSS Base component has been previously installed:



After selecting the install location, press **Next**.

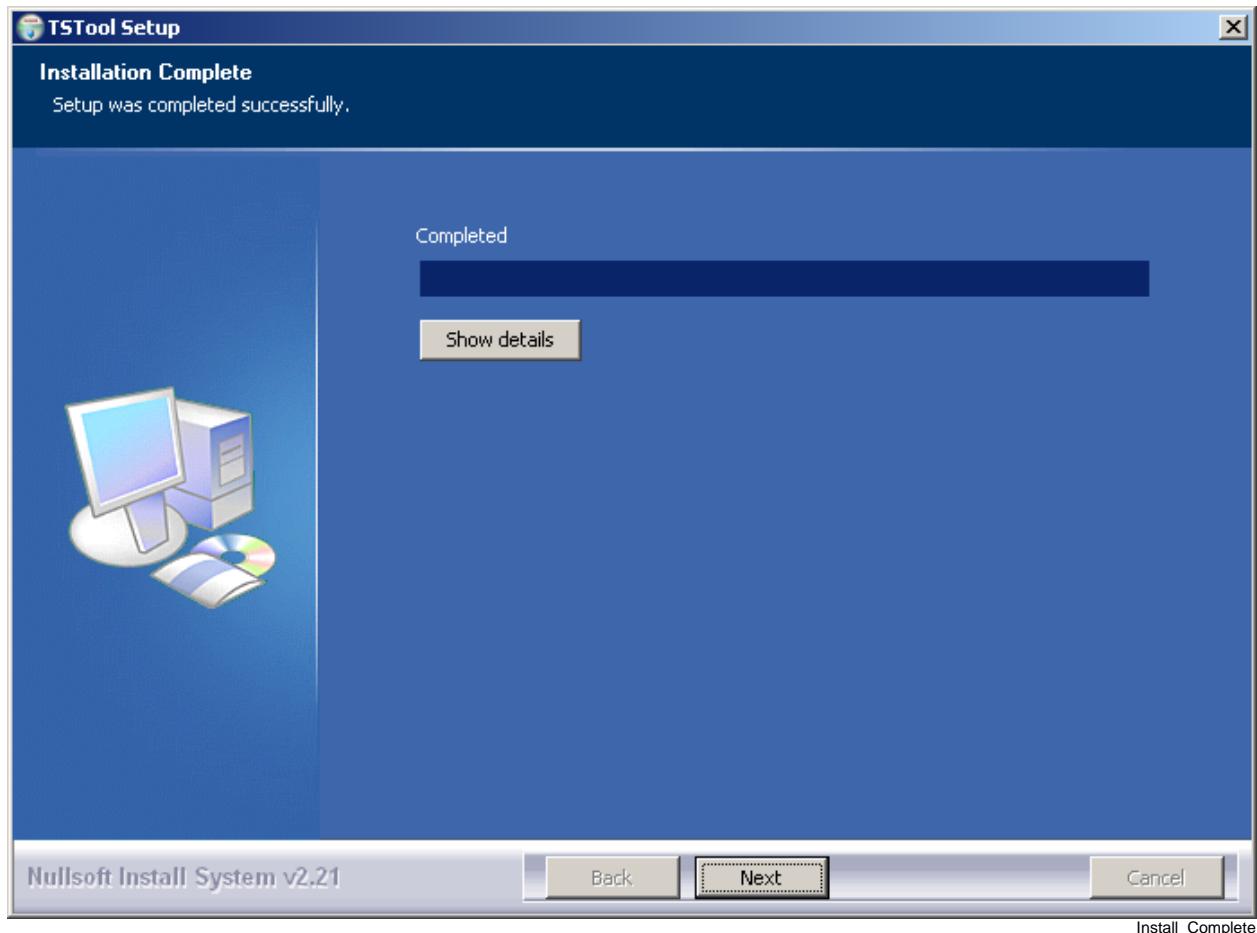
Note that this location will be saved as a Windows registry setting (HKEY_LOCAL_MACHINE\Software\State of Colorado\TSTool\Path) to allow future updates to check for and default to the same install location, and to allow the standard software uninstall procedure to work correctly.

4. The following dialog will be shown to select the menu for the software:



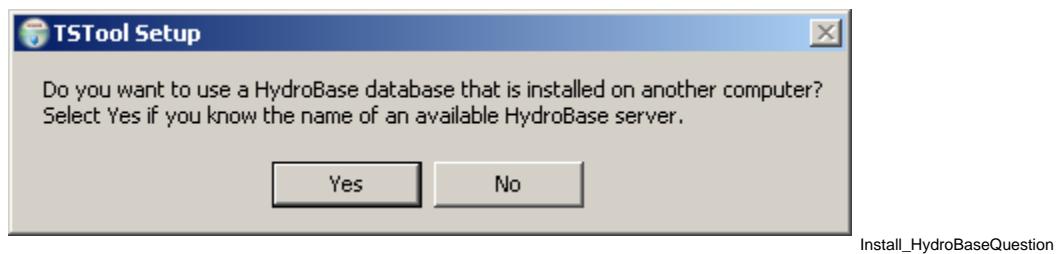
After selecting the folder, press **Install**.

5. The following dialog will show the progress of the installation:



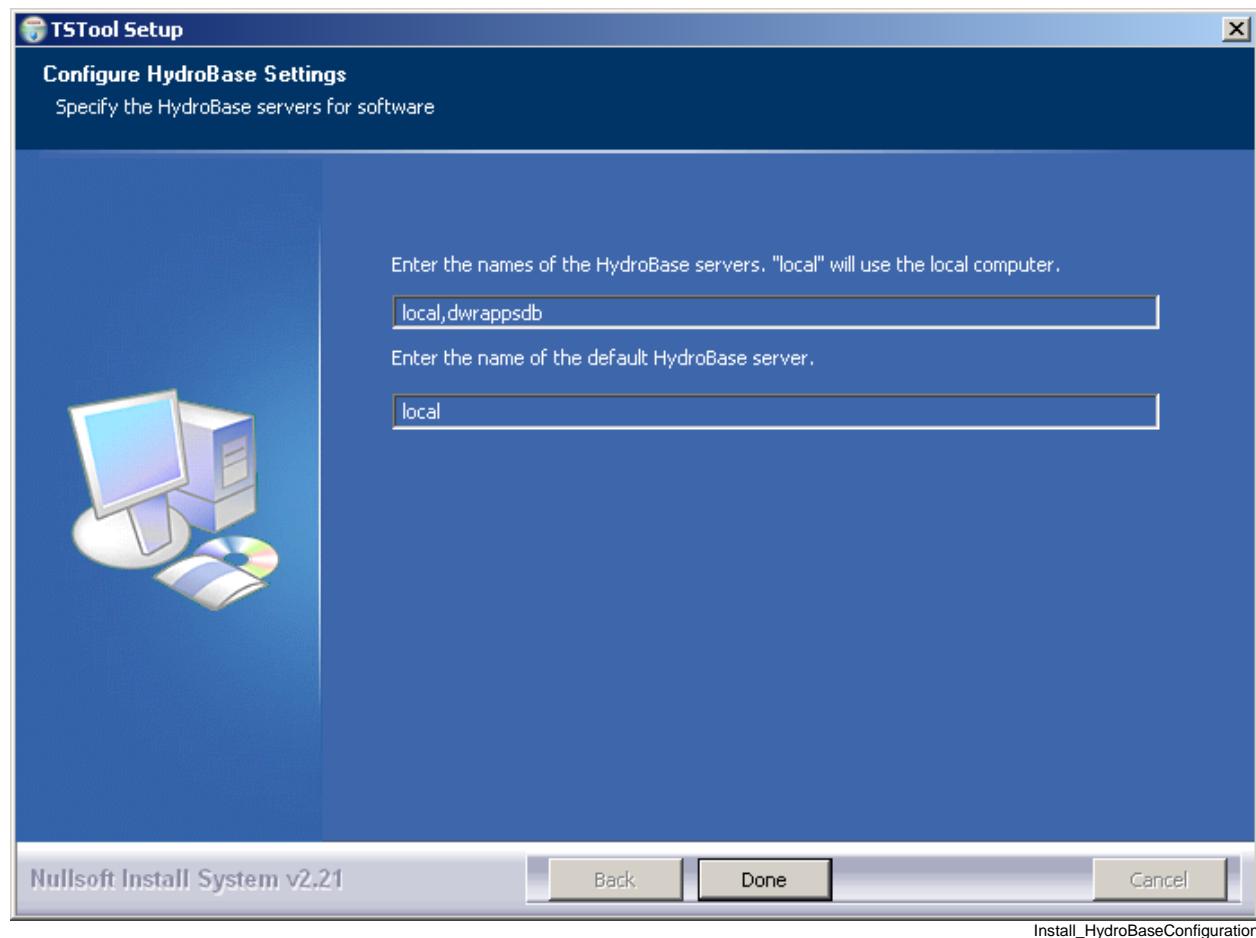
Press **Show details** to see the files that were installed or press **Next** to continue.

6. If the CDSS Base Components were selected for install, the following dialog will be displayed:



TSTool and other CDSS software can utilize HydroBase running on the local computer as well as other computers. Press **Yes** if HydroBase has been installed on another computer in the network environment and may be used by the software (then continue to the next step). Otherwise, press **No** (skip to step 8).

7. The following dialog allows additional HydroBase servers to be specified for use by CDSS software (the example below configures CDSS software to list the *dwrappsdb* HydroBase server in choices and defaults to HydroBase on the local computer):



Install_HydroBaseConfiguration

After entering the name of a HydroBase server and the default server to use, press **Done**.

8. The following dialog will then be shown asking whether the TSTool software should be run:



Install_RunTSToolQuestion

Press **Yes** to run the software or **No** to exit the installation procedure.

9. A reboot is not required to use the software from the **Start** menu. However, a reboot may be required on some computers to run TSTool from the command line.

3.3 Installing TSTool on a File Server

TSTool can be installed on a file server, which allows software updates to be made in one location, thereby eliminating the need to install software on individual machines. For this type of installation, all computers that access the software must have similar configuration, including network configuration. The standard installer described in this documentation focuses on individual installs on user computers. To make TSTool software installed on a server available to other computers, perform the following (this is typically performed by system administrators):

1. Run the *TSTool_Setup.exe* installer as described above. During installation specify the CDSS home using a drive letter and path for the server or specify a Universal Naming Convention (UNC) path (e.g., <\\CDSSServer\CDSS>). All computers that will use the software will need to have access to the server in a consistent way because the TSTool software will expect the CDSS installation home at runtime to be that specified during the installation.
2. The menus and shortcuts will only be configured for the computer from which the installation was run. Therefore, menus and shortcuts for other computers will need to be manually configured.
3. If appropriate, edit the *CDSSInstallHome\system\CDSS.cfg* file to include additional information (e.g., Satellite Monitoring System database configuration – see **Section 6** below). The installer described in this documentation focuses on general users who in most cases will not have access to advanced features available only within the State of Colorado's server environment.

If TSTool has been installed on a local computer and it is also available on the network, the network version can be run by running the software in the *NetworkCDSSInstallHome\bin* folder. The software will expect that file locations use the same drives as when the software was installed.

3.4 Special Installation Instructions for Microsoft SQL Server Support

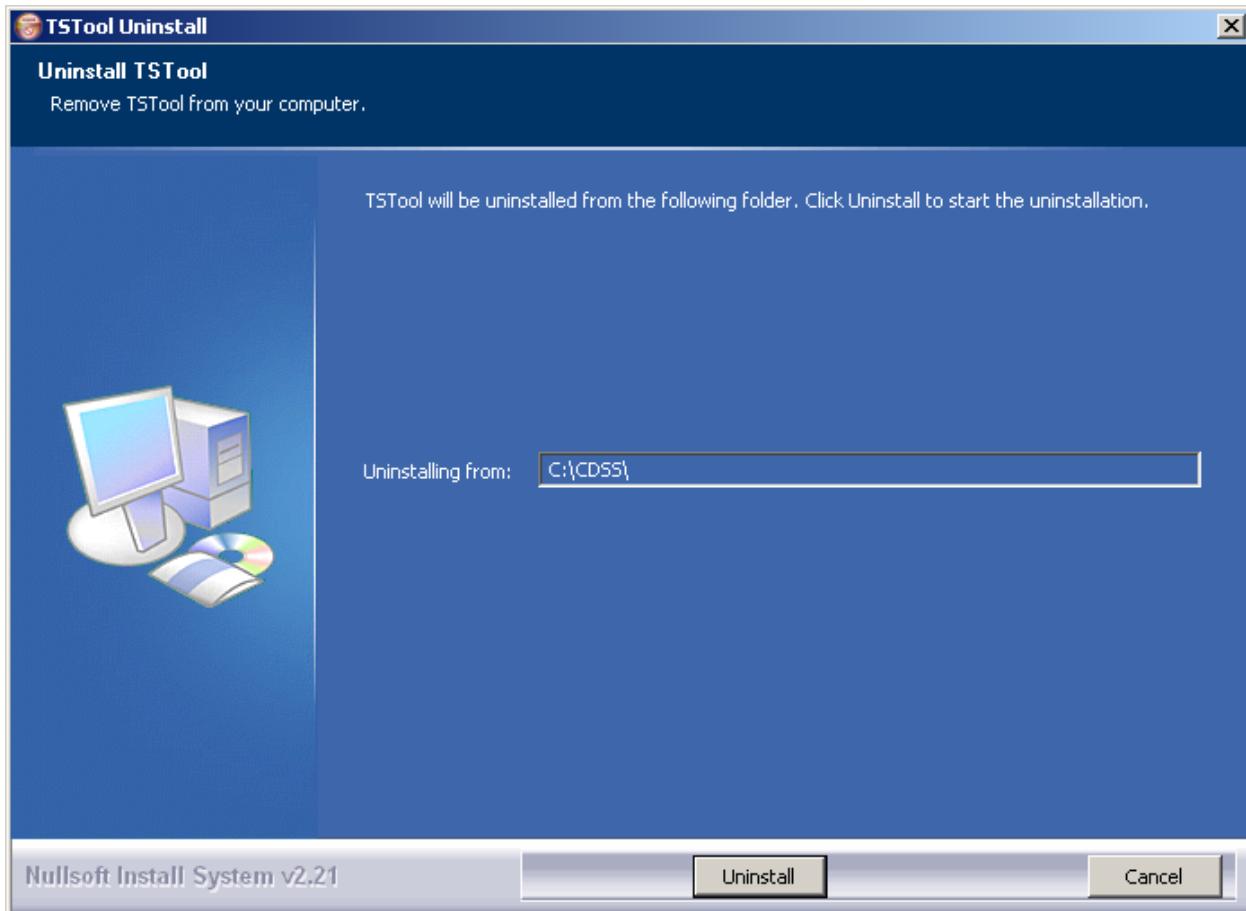
CDSS software uses the free SQL Server driver from Microsoft. The CDSS software requires only the *msbase.jar*, *mssqlserver.jar*, and *msutil.jar* files from the Microsoft installation. However, Microsoft requires that you do a full installation of its driver in order to acknowledge the End User License Agreement (EULA). Because only three files listed previously are needed for CDSS Java software, it is recommended that the Microsoft install be completed once within an organization (to complete the EULA recognition), but then use the three files distributed with CDSS software as is (ignore the files from the full Microsoft installation). If installing the HydroBase CD, the Microsoft files will be installed automatically and this EULA will be displayed.

To manually perform the Microsoft installation, do one of the following:

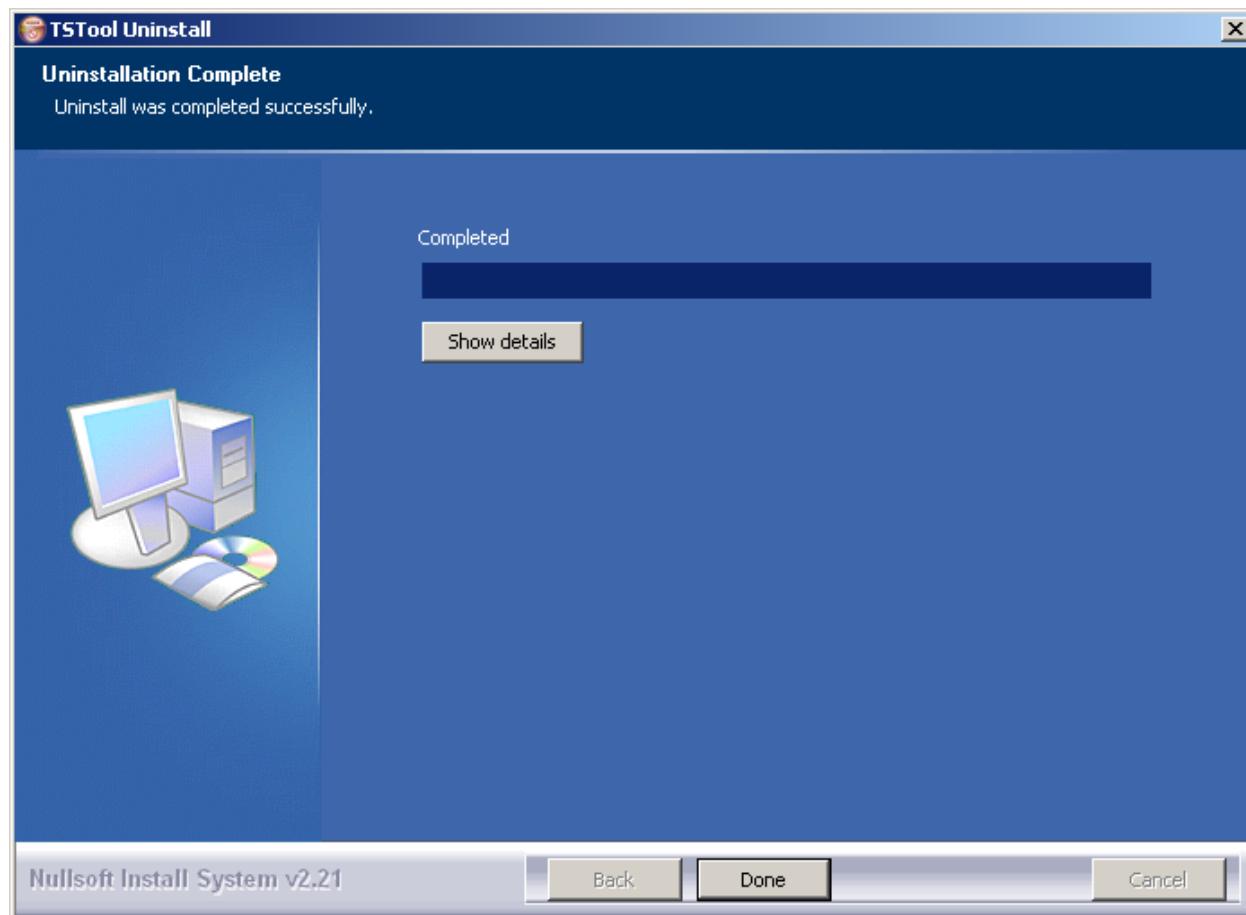
1. Contact the supplier of the CDSS software (e.g., the State of Colorado IT staff) to provide the Microsoft driver. For software installations at the State, this step needs be done only once by the State of Colorado IT staff. All other CDSS users at the State can then use the software using the three software files described above.
2. Download and install the Microsoft JDBC driver from <http://www.microsoft.com/sql/downloads> and install according to its instructions. Do this if you have purchased SQL Server and will install the State's HydroBase using SQL Server at your organization.

4. Uninstalling TSTool Software

To uninstall TSTool software, select the **CDSS...Uninstall...TSTool** from the **Start** menu and confirm the uninstall. CDSS components that are used by other software (e.g., CDSS Base component software) as well as user data will remain installed.



Press **Uninstall** to uninstall the software.



Press **Show details** to see the list of files that were removed. Press **Done** to exit the uninstall.

5. Running TSTool

TSTool can be started in several ways as described below.

5.1 CDSS Menu

The **Start...All Programs...TSTool** (or **Start... Programs... TSTool**) menu can be used to start the software. This runs the *CDSSInstallHome\bin\TSTool.exe* software.

5.2 Command Line Executable

The installation process adds the *CDSSInstallHome\bin* folder to the path, allowing the TSTool software to be started anywhere by running TSTool. Running TSTool from any drive will result in the software being run in the installation location. Specifying a commands file on the command line or interactively will reset the working directory to that of the commands file.

5.3 TSTool Batch File

TSTool was previously started with and can still be run with the *CDSSInstallHome\bin\TSTool.bat* file, for example to support troubleshooting. In this case, the file name *tstool.bat* must be fully specified because running *tstool* will result in the executable program being run (see previous section).

6. TSTool Configuration

TSTool requires minimal configuration after installation. This section describes TSTool configuration files that can be customized for a system. Configuration is specified for each TSTool installation. Installations on a server use one configuration for all users.

6.1 TSTool Configuration File

The *CDSSInstallHome\system\TSTool.cfg* file under the main installation directory contains top-level configuration information for TSTool. The format of the file is as follows:

```
#  
# Configuration file for TSTool  
  
[TSTool]  
  
ColoradoSMSEnabled = true  
DateValueEnabled = true  
HydroBaseEnabled = true  
RiverWareEnabled = true  
StateCUEnabled = true  
StateModEnabled = true  
  
MapLayerLookupFile = "\cdss\gis\co\TimeSeriesMapLookup.csv"  
  
LicenseOwner = "CDSS"  
LicenseType = CDSS  
LicenseCount = NoLimit  
LicenseExpires = Never  
LicenseKey = 00-77960bdfb1dde707-1dd052fe0327a332-a07266ee645e8845-7560192d374235c5-  
1dd052fe0327a332
```

Example TSTool Configuration File

The example illustrates the format of the file. The *Enabled properties can be used to enable/disable input types. Common formats are enabled by default and more specialized formats are disabled by default, if not specified in the file. For example, use HydroBaseEnabled = false to disable the automatic HydroBase login that occurs with the HydroBase input type (e.g., if HydroBase is unavailable for some reason). The license properties are assigned by RTi and should not normally be changed by TSTool users. Each input type can have additional properties, although only a few currently do, as described below.

The optional MapLayerLookupFile property indicates the name of the time series to map layer lookup file. See the **Map Configuration** section below.

6.2.1 HydroBase Configuration

The following properties can be defined in the *TSTool.cfg* file in a [HydroBase] section to control how TSTool interacts with HydroBase. See also the **CDSS Configuration File** section below.

TSTool HydroBase Configuration Properties

Property	Description	Default
WDIDLength	Indicates the length of water district identifiers (WDIDs) constructed from separate WD and ID data, when creating time series identifiers. Because time series identifier strings are compared literally, it is important that the WDIDs are consistent within a commands file.	7

6.2 Data Units File

The *CDSSInstallHome\system\DATAUNIT* file under the main installation directory contains data unit information that defines conversions and output precision. In most cases the default file can be used but additional units may need to be added for a user's needs (in this case please notify the developers so the units can be added to the default file distributed with installations). Currently, the *DATAUNIT* file is the only source for units information – in the future units may be determined from the various input sources.

6.3 CDSS Configuration File

By default, TSTool will automatically look for HydroBase databases on the current (local) machine and the State servers. State server databases are typically only accessible to State of Colorado computers. If SQL Server or MSDE HydroBase versions have been installed on a different machine, the *CDSSInstallHome\system\CDSS.cfg* file can be used to indicate the database servers. An example of the configuration file is as follows:

```
[HydroBase]

ServerNames="ServerName,local"
DefaultServerName="ServerName"
DefaultDatabaseName="HydroBase_CO_East_20050730"

[ColoradoSMS]

ServerNames="ServerName,local"
DefaultServerName="ServerName"
DefaultDatabaseName="RealtimeStreamflow"
UserLogin="UserLogin"
```

The ColoradoSMS input type is being used to support annotation of real-time data graphs with alert information. These features are under development and should be configured only on State of Colorado computers.

Properties can be specified on the TSTool command line using the notation "Property=Value" and will in some cases override the values in the configuration file. These features are under development as necessary.

The CDSS configuration properties are described in the following tables:

CDSS HydroBase Database Configuration Properties

Property	Description	Default
ServerNames	A comma-separated list of server names to list in the HydroBase login dialog.	The state server is listed.
Default ServerName	The default HydroBase server name to use. This allows the HydroBase login dialog to preselect a default that applies to most users in the system. If TSTool is run in batch mode and the HydroBase input type is enabled, use this property to make a default connection to HydroBase, for use with other commands in the batch run.	greenmtn.state.co.us
Default DatabaseName	The default HydroBase database name to use. This allows the HydroBase login dialog to preselect a default that applies to most users in the system. If TSTool is run in batch mode and the HydroBase input type is enabled, use this property to make a default connection to HydroBase, for use with other commands in the batch run.	
Database Engine	Reserved for internal use.	
DatabaseName	The database name to use for the initial connection. This overrides the default server.	
Database Server	The server name to use for the initial connection. This overrides the default server.	
SystemLogin	Reserved for internal use.	
SystemPassword	Reserved for internal use.	
UserLogin	Reserved for internal use.	

CDSS Satellite Monitoring System (ColoradoSMS) Database Configuration Properties

Property	Description	Default
ServerNames	A comma-separated list of server names to list in the SMS login dialog.	The state server is listed.
Default ServerName	The default SMS database server name to use. This allows the SMS login dialog to preselect a default that applies to most users in the system. If TSTool is run in batch mode and the ColoradoSMS input type is enabled, use this property to make a default connection to the SMS database, for use with other commands in the batch run.	greenmtn.state.co.us
Default DatabaseName	The default SMS database name to use. This allows the SMS login dialog to preselect a default that applies to most users in the system. If TSTool is run in batch mode and the ColoradoSMS input type is enabled, use this property to make a default connection to the SMS database, for use with other commands in the batch run.	
Database Engine	Reserved for internal use.	
DatabaseName	The database name to use for the initial connection. This overrides the default server.	
Database Server	The server name to use for the initial connection. This overrides the default server.	
SystemLogin	Reserved for internal use.	
SystemPassword	Reserved for internal use.	
UserLogin	The user login, for use with TSTool batch runs. The ColoradoSMS.UserLogin parameter can be specified on the command line and will be used when making the initial SMS database connection.	

The SMS database cannot currently be opened with a login dialog. Therefore, correct information must be specified in the CDSS configuration file and the TSTool command line.

6.4 Map Configuration

TSTool can display maps configured as GeoView project files. See the **GeoView Mapping Tools** appendix for more information about these files. To allow a link between time series and map layers, use the `TimeSeriesMapLayerLook` property in the `TSTool.cfg` file to specify a time series to map layer lookup file (see the **TSTool Configuration File** section above). The following example file illustrates the contents of the lookup file:

```
# This file allows time series in TSTool to be linked to stations in spatial
# data layers. The columns are used as appropriate, depending on the direction
# of the select (from time series list or from the map).
#
# This file has been tested with the \CDSS\GIS\CO\co_TSTool.gvp file. Not all
# possible combinations of time series and map layers have been defined - only
# enough to illustrate the configuration.
# Additional attributes need to be added to the point files to allow more
# extensive functionality. For example, if attributes for data interval (time
# step) and data source are added to the attributes, then a definition query
# can be defined on the layer for displays to use the same data file. The
# configuration below can then use the different names to configure the link
# to time series.
#
# TS_InputType - the time series input type, as used in TSTool
# TS_DataType - the data type shown in TSTool, specific to an input type
#                 For example, TSTool uses "Streamflow" for HydroBase, whereas
#                 for other input types a different data type string may be used.
# TS_Interval - time series interval of interest (e.g.,"Month", "Day", "1Hour"
#                 "Irregular")
# Layer_Name - the layer name used in the map layer list
# Layer_Location - the attribute that is used to identify a location, to be
#                   matched against the time series data location
# Layer_DataType - the attribute that is used to indicate the data type for a
#                   station's time series (CURRENTLY NOT USED - UNDER EVALUATION)
# Layer_Interval - the attribute that is used to indicate the interval for a
#                   station's time series
# Layer_DataSource - the attribute that is used to indicate the data source for
#                   a station's time series.
#
# When matching time series in the TSTool time series query list with features
# on the map, the TS_* values are matched with the time series identifier
# values and the Layer_* attributes are matched against specific time series.
#
# Data layers are listed from largest interval to smallest.
"TS_InputType","TS_DataType","TS_Interval","Layer_Name","Layer_Location","Layer_DataSource"
HydroBase,DivTotal,Day,"Diversions",id_label_7, ""
HydroBase,DivTotal,Month,"Diversions",id_label_7, ""
HydroBase,EvapPan,Day,"Evaporation Stations",station_id, ""
HydroBase,EvapPan,Month,"Evaporation Stations",station_id, ""
HydroBase,Precip,Irregular,"Precipitation Stations",station_id, ""
HydroBase,Precip,Day,"Precipitation Stations",station_id, ""
HydroBase,Precip,Month,"Precipitation Stations",station_id, ""
HydroBase,RelTotal,Day,"Reservoirs",id_label_7, ""
HydroBase,RelTotal,Month,"Reservoirs",id_label_7, ""
HydroBase,Streamflow-DISCHRG,Irregular,"Streamflow Gages - Real-time",station_id, ""
HydroBase,Streamflow,Day,"Streamflow Gages - Historical",station_id, ""
HydroBase,Streamflow,Month,"Streamflow Gages - Historical",station_id, ""
```

Example Time Series Map Layer Lookup File

The columns in the lookup file indicate how information in the time series input/query list can be matched against time series in map layers. In particular, the `TS*` columns define values that are seen in the TSTool interface and the `Layer*` columns define the layer and attribute names for map layers. The `Layer_Interval` and `Layer_DataSource` are optional but if specified result in more specific links between time series and map layers.

This page is intentionally blank.

Appendix

TSTool Release Notes

Version, 7.00.00, 2006-03-03, Acrobat Distiller

This appendix provides information about changes that have occurred in TSTool versions.

TSTool Version History

The following table summarizes the TSTool release history. See the following sections for more detailed information about each version. Only recent versions are documented.

TSTool Version History Summary (most current at top)

TSTool Version	Version Information	Release Date
7.00.00	Begin distributing software using a new installer. Add CASS livestock data and human population data.	2006-10-31
6.19.00	Update to extend period when filling with diversion comments.	2006-05-19
6.18.00	Add the runCommands () command to facilitate data processing.	2006-05-02
6.17.00	Add the compareFiles() command to facilitate testing.	2006-04-17
6.16.02	Begin adding commands to test data, for alarms.	2006-04-17
6.16.01	Time series to map link is enabled. Improve UNC support. Improve startup performance in batch mode.	2006-02-16
6.16.00	Begin adding support for NDFD (National Digital Forecast Database) input type, and maintenance.	2006-01-31
6.15.00	Begin adding time series to map link.	2006-01-16
6.14.00	Update some commands to named parameter notation, and maintenance.	2005-12-14
6.13.00	Internal release.	2005-11-13
6.12.00	Improve error handling when running in batch mode with graphs.	2005-10-05
6.11.00	Enable the ColoradoSMS input type for hydrograph annotations and update batch mode features to better utilize the CDSS configuration file.	2005-10-05
6.10.09	Maintenance release – convert some commands to use named parameters.	2005-09-28
6.10.08	Maintenance release – convert some commands to use named parameters. Add the newStatisticYearTS () command.	2005-09-22
6.10.07	Maintenance release – convert some commands to use named parameters.	2005-08-24
6.10.06	Release corresponding to the CDSS CD release.	2005-08-04

TSTool Version History Summary, continued (most current at top)

TSTool Version	Version Information	Release Date
6.10.05	Respond to CDSS testing feedback.	2005-08-01
6.10.04	Respond to CDSS testing feedback. Add additional query filters for HydroBase stations and structures.	2005-07-20
6.10.03 BETA	Begin phasing in saving time series products to HydroBase and RiversideDB.	2005-07-08
6.10.02 BETA	Update the <code>openHydroBase()</code> command to use free-format parameters.	2005-06-28
6.10.01 BETA	Begin enabling data flags for time series to support enhancements to fill commands.	2005-06-03
6.10.00 BETA	Initial release supporting HydroBase stored procedures with initial prototypes of Mixed Station Analysis and related features. Implement new message log viewer and commands to simplify comparison of time series.	2005-06-01
6.09.03	Maintenance release.	2004-12-21
6.09.02	Maintenance release.	2004-10-05
6.09.01	Add NWSRFS FS5Files input type.	2004-09-01
6.09.00	Add <code>readHydroBase()</code> commands.	2004-08-27
6.08.02	Documentation made current to include all version 6 changes.	2004-07-27
6.08.01	Allow HydroBase connection to be made at startup.	2004-07-20
6.08.00	Allow wildcards in commands that read from StateCU and StateModB input types.	2004-07-11
	Initial Java version.	1997-10-23

Changes in Version 7.00.00

- Begin using the Nullsoft Scriptable Install System (NSIS) to build software installers.
- Begin distributing TSTool as an executable file `TSTool.exe`, which starts up the Java Runtime Environment. This allows for simpler configuration of the **Start** menu and gives users a more traditional executable to run.
- The software organization is slightly different from the previous releases in order to recognize clearer boundaries between components. Several new Jar files are provided, rather than being merged with other Jar files. The **Installation and Configuration** appendix lists the files.
- Add support for Colorado Agricultural Livestock Statistics and human population time series in HydroBase.

Changes in Version 6.19.00

- Update `fillUsingDiversionComments()` to extend time series with diversion comments available outside the normal diversion records period, if no query or output period has been specified.

Changes in Version 6.18.00

- Add `runCommands()` to allow a controlling commands file to run other commands files.

Changes in Version 6.17.00

- Add `compareFiles()` to help with regression testing, to verify current and expected results.

Changes in Version 6.16.02

- Begin adding data test commands in development mode – these commands will evaluate time series for critical conditions.
- Reenabled `fillMove2()`, which was unintentionally disabled in a previous release.

Changes in Version 6.16.01

- First version that includes operational features to support link between time series and map interface.
- Increase performance at startup when no main GUI is shown, for cases when TSTool is being used to provide graphs for other software.
- Add support for Universal Naming Convention (UNC) for software home in startup files.
- Change ***View...Map Interface*** to ***View...Map***.

Changes in Version 6.16.00

- Implement hooks for the NDFD input type.
- Improve handling of NWS Card file extensions in commands and ***File...Save*** menu choices.
- Add map interaction features. See the **Installation and Configuration** appendix for more information about configuring links with maps.

Changes in Version 6.15.00

- Begin implementing link between time series and map interface.
- Reorder general command menus to be more consistent with other software.
- Add warning if time series cannot be retrieved from the RiversideDB input type.

Changes in Version 6.14.00

- Change the `setQueryPeriod()` command to `setInputPeriod()` to be consistent with other software nomenclature. The old command is still supported.
- The `readNwsCard()` and `TS Alias = readNwsCard()` commands both now use the named-parameter notation and have the new `Read24HourAsDay` parameter.
- Blank lines in commands files now display properly.
- Fix bug where time series table sometimes showed half-drawn rows.
- Fix bugs where StateMod binary and StateCU input type file chooser prompt would not allow a cancel of the file select to occur. Cancel now results in the previous file that was selected being displayed.

Changes in Version 6.12.00

- Improve error handling for processing time series products. In particular, TSTool now returns a non-zero exit status if there is an error processing a product. This can be detected by external software that is running TSTool.

Changes in Version 6.11.00

- Enable the ColoradoSMS input type and begin adding alert annotations for streamflow graphs.
- Fix bug so that if a commands file is specified using a relative path, the working directory is interpreted correctly to determine the full path to the commands file.
- Add the ability to accept Parameter=Value command line parameters. This will allow override of configuration file information.
- Convert `processTSPproduct()` to use named parameters and ensure that output can be viewed even if in batch mode with no main GUI.
- Update so that for batch runs, the `CDSS.cfg` file information for HydroBase is used to make the initial connection. Phase out the HydroBase database properties in the `TSTool.cfg` file.

Changes in Version 6.10.09

- Convert `cumulate()` to use named parameters and begin development of a new `Reset` parameter.
- Convert `readStateModB()` to use named parameters and add the `Version` parameter to allow reading of old files. The features associated with the `Version` parameter are under development.
- Update the `newStatisticYearTS()` to support calculation of maximum and minimum values in a year and count of values in a year above/below a test value. Also update the command to better handle incomplete data at the end of the analysis period.
- Update the `openHydroBase()` command to check the `CDSS.cfg` information and provide database server and database name choices to the user, to minimize errors in use.

Changes in Version 6.10.08

- Convert `fillConstant()` to use named parameters.
- Convert `newTimeSeries()` to use named parameters.
- Add the `newStatisticYearTS()` command, in particular to support calculation of frost date time series.
- Update `openHydroBase()` to accept the database name parameter.
- Double-clicking on a command will now cause the editor for the command to be displayed.
- Add a **Command Glossary** to the documentation and begin to standardize command parameter names to be consistent.

Changes in Version 6.10.07

- Convert `scale()` to use named parameters.
- Change `TS X = ...` to `TS Alias = ...` in menus. Start to change notation in documentation and command dialogs.
- Convert `copy()` to use named parameters and add the ability to assign a new TSID to the copy.
- Convert `writeStateMod()` to use named parameters and add ability to select time series to write.
- Convert `readStateMod()` to use named parameters and add parameters for the input period..

Changes in Version 6.10.06

- Official release to support stored procedures.
- Documentation made current to reflect changes since the last documentation issue.
- Respond to feedback from previous 6.10.x incremental releases.
- Fix bug where XY-Scatter graph was not working due to changes in the 6.10.00 BETA release.

Changes in Version 6.10.05

- Add the `lagK()` command.
- Update the `fillProrate()` command `InitialValue` parameter to support `NearestForward` and `NearestBackward`.

Changes in Version 6.10.04

- Add additional input filter choices for HydroBase structures and stations, consistent with the StateView software.
- Update the `fillProrate()` command to include the `ComputeFactorHow` parameter to allow computing the proration factor based on an average of ratios. Update the command to support free-format parameters.
- Update the `selectTimeSeries()` command to allow combinations of selection filters, to allow more flexibility.
- Add the ability to query HydroBase infrequent diversion and reservoir release time series.

Changes in Version 6.10.03 BETA

- Input filters for HydroBase well structures and stations are now handled properly.
- Add initial support for saving time series products to HydroBase and RiversideDB.

Changes in Version 6.10.02 BETA

- Update the `openHydroBase()` command to use free-format parameters.

Changes in Version 6.10.01 BETA

- Enable ability to have data flags for daily and monthly data.
- Update the `writeRiverWare()` command to handle time steps other than hourly.

Changes in Version 6.10.00 BETA

- Begin releasing support for HydroBase stored procedures.
- Begin development of generic `changeInterval()` command and update to free-format parameters.
- Begin work on the Mixed Station Analysis tool and `fillMixedStation()` command.
- Update the `fillRegression()` command to support free-format parameters.
- Begin work on the `analyzePattern()` command.
- Add the **Commands...Analyze Time Series** menu for analysis commands.
- Add the **Commands...Models** menu for more complicated modeling commands.
- Add the **Tools...Analysis** menu for analysis tools.

- Begin implementing the generic log file viewer, which allows links between commands and log messages.
- Change defaults to NOT display messages to the console, to improve performance.
- Add the point graph type.
- Add the predicted value graph type.
- Add the predicted value residual graph type.
- Add the `sortTimeSeries()` command.
- Add the ability for the `readNWSCard()` command to read 1+ time series.
- Add the `startLog()` command.
- Add the `compareTimeSeries()` command.
- Update the `fillHistMonthAverage()` and `fillHistYearAverage()` commands to have fill flag and free-format parameters.
- Add a warning in the `add()` command when frost date time series are added and indicate more appropriate commands.

Changes in Version 6.09.03

- Fix bug where initial directory with spaces in name was causing errors.

Changes in Version 6.09.02

- Added release notes to documentation.
- Fix bug in NWSRFS FS5Files input type where identifiers with underscores were not being handled.
- StateModB input type reservoir data types (and some well types) had ? for data groups – this has been resolved using StateMod 10.27 HTML documentation.

Changes in Version 6.09.01

- Added NWSRFS FS5Files input type support, for use with the National Weather Service River Forecast System (NWSRFS).
- Fix summary reports (daily totals and means) to handle minute data.

Changes in Version 6.09.00

- Add the `readHydroBase()` command to read one or more HydroBase time series while filtering based on location, ID, etc.

Changes in Version 6.08.02

- Documentation updated to reflect all version 6 changes.
- Minor corrections to interface based on documentation review.

Changes in Version 6.08.01

- For the HydroBase input type, allow the ODBC DSN to be specified in the TSTool configuration file, to allow a HydroBase connection to be made at startup without prompting. This supports the CDSS CD distribution.

Changes in Version 6.08.00

- Allow StateCU input type time series read commands to allow wildcards.
- Allow StateMod input type time series read commands to allow wildcards.

This page is intentionally blank.

Appendix

DateValue Input Type

2004-08-05, Acrobat Distiller

Overview

The DateValue time series file format can be used to store one or more time series of consistent time interval. The format has been developed by Riverside Technology, inc. The example below shows the format of the file. Important comments about the file format are:

- The file is divided into a header section (top) and data section (bottom). Comments can occur anywhere in the file and are lines that start with #.
- If not specified, many of the header properties will be set to reasonable defaults as data are read by software such as TSTool. However, as much information as possible should be specified to allow complete time series handling. Header information is displayed by applications like TSTool to allow selection of time series before the data section is read.
- Properties are checked in a case-independent fashion.
- The TSID, Start, End, and Units properties are important for basic time series handling.
- The interval part of the TSID is used to determine how memory should be allocated for data.
- The Start and End values are used to allocate memory for regular interval time series. Dates associated with data values are used to allocate memory for irregular interval time series.
- For regular interval time series, if data lines between the start and end dates are omitted, the unspecified values are set to the missing data value for the time series (default is -999).

```
# DateValueTS 1.1 file
#
# This is a sample of a typical DateValue minute time series. This format
# was developed by Riverside Technology, inc. to store time series data. An
# example file is as follows and conforms to the following guidelines:
#
# * Comments are lines that start with #.
# * Applications often add a comments section at the top indicating how the
#   file was created
# * Any line that starts with a number is assumed to be a data line.
# * Date hours should be in the range 0 to 23 (an hour of 24 will be
#   converted to hour 0 of the next day).
# * If a time is necessary, the date/time may be separated by a space, T, :, or
#   @. If a space is used, use a date and time column headings,
#   if headings are used.
# * The same general format is used for year, month, day, hour, and minute
#   data, except the format of the date is adjusted accordingly.
# * If multiple time series are written, header variables are delimited with
#   space or tab characters. Data are delimited by tab or space (or use the
#   Delimiter property to set the delimiters used for data lines)
# * Internally, the time series identifier may initially be set using the file
#   name. For example, a file name of XXX.USGS.Streamflow.MONTH will result
#   in the location being set to "XXX", the data source to "USGS", the data
#   type to "Streamflow", and the interval to 1 month. The identifier
#   information is reset if individual properties are specified in the file.
# * This format is free-format and additional information may be added in
```

DateValue Input Type

```
#   future (e.g., data quality strings).
# * For portability, data in a DateValue file should have compatible intervals.
# * Header variables and column headers can be enclosed in double quotes if
#   the data contain spaces.
# * Missing data can either be coded as the missing data value or no value
# * Missing records will result in missing data being used when read.
#
# The following header variables are recognized. This information can be
# used by software.
Version = 1.1                      # Optional. File format version
                                      # (to handle format changes)
Delimiter = " "                     # Optional. Delimiter for data lines
                                      # (default is space and tab)
NumTS = 2                           # Optional. Number of time series in file
                                      # (default is 1)
TSID = "XXX.USGS.Streamflow.15MINUTE" "YYY.USGS.Streamflow.15Minute"
                                      # Required.
                                      # List of time series identifiers in file
                                      # Location.Source.DataType.Interval.Scenario
                                      # Do not include input type and name in identifier
SequenceNum = 1950 1951             # Optional - used with time series traces.
                                      # Indicates the year for the trace
Description = "Flow at XXX" "Flow at Y"
                                      # Optional. Description for each time series.
DataFlags = true,1 false            # Optional. Indicates whether data flags (e.g.,
                                      # character data quality) are provided. If true,
                                      # specify the maximum number of characters that
                                      # will be used in any flag (the default is 2 if
                                      # not specified). The data value column for each
                                      # time series with data flags is followed by a
                                      # column for the data flag. Surround the flags by
                                      # "" if a flag is not specified or is a space.
DataType = Streamflow Streamflow
                                      # Optional. Data types for each time series
                                      # (consistent with TSID if specified).
                                      # The default is to use the data type in the TSID
                                      # Supplied to simplify use by other programs.
Units = CFS CFS                   # Optional. Units for each time series
                                      # (default is no units).
MissingVal = -999 -999             # Optional. Missing data value for each
                                      # time series (default is -999).
IncludeCount = true                # Optional. If true, column after date/time
                                      # is record count (1...) (default is false).
IncludeTotalTime = true            # Optional. If true, column after date
                                      # is cumulative time (0...) (default is false).
# Both of above can be true, and both columns will be added after the date
Start = 1996-10-18:00:00           # Required. Start date for time series
End = 1997-06-14:00:00             # Required. End date for time series
                                      # Period dates should be of a precision consistent
                                      # with the dates used in the data section below.
# Optional. The following line can be read into a spreadsheet or database for
# headers. The lines above this line can be ignored in a spreadsheet import.
# The number of headings should agree with the number of columns.
Date "Time" "Count" "TotalTime" "Description 1" "DataFlag1" "Description 2"
1996-10-18 00:00 1 0 110.74 "m" 14.2
1996-10-18 00:15 2 15 113.24 ""
...
...
```

DateValue Files and Standard Time Series Properties

The standard time series identifier for DateValue files is of the form:

Location.DataSource.DataType.Interval.Scenario~DateValue~PathToFile

Because DateValue time series files are a persistent storage format for in-memory time series objects that have been developed by RTi, the properties stored in the file closely match the standard time series properties of the objects. In particular, the time series data type, units, and missing data value are consistent with time series header information. The TSID property in a DateValue file is directly applied to time series objects read from the file, allowing explicit identification of the time series in the file, regardless of the name of the file. This allows multiple time series to be saved in a single file. The data source typically agrees with that determined from a data-supplying agency or model that generates the data.

Limitations

DateValue files have the following limitations:

- The header information in DateValue files may be too technical for some general tools. However, simple delimited files cannot be handled as rigorously by some applications, like TSTool. Spreadsheets can import DateValue files easily by ignoring the header lines.
- Because date/time values are included on every data line, processing DateValue time series files requires more disk space and processing time. However, using the dates on each line also allows gaps in data to be omitted from the file. Inclusion of the date/times for each data point is considered a reasonable trade-off to ensure data quality and readability. Many other time series file formats also include the date/time on each line.

DateValue Input Type

This page is intentionally blank.

Appendix

HydroBase Input Type

2006-11-06, Acrobat Distiller

Overview

The State of Colorado's HydroBase database stores a variety of time series data. The time series conventions described here, in particular for time series identifiers, are consistent for major CDSS software components including TSTool, StateView/CWRAT, StateDMI, and StateMod GUI. This allows for consistent features and sharing of data between software tools.

The current database design splits time series into three main categories:

1. Data related to structures or administrative data maintained by the State of Colorado (e.g., diversions, reservoirs). Structure locations are typically identified using a water district identifier (WDID), consisting of a two digit State of Colorado water district number and a trailing structure identifier (which in the past was four digits but has been increased to five or more digits to support longer identifiers). Although a single WDID identifier is used when identifying time series, the separate WD and ID fields are generally needed to find information in HydroBase.
2. Data for stations, consisting mainly of location information and time series (e.g., NOAA precipitation data, USGS streamflow). Station locations are typically identified using a station identifier from the data source. For example, stations can use a USGS identifier, a State of Colorado Satellite Monitoring System abbreviation, or other identifier.
3. Data recorded at locations that are not stations or structures. For example, Water Information Sheet (WIS) are daily spreadsheets used to administer water. Although WIS contain data values for structures and stations, the time series are extracted from database tables that are not directly associated with structure or station database tables. Other examples include Colorado and national agricultural crop statistics.

A structure or station may have more than one identifiers depending on the number of agencies involved with data collection, etc. For example, a reservoir may have a State of Colorado WDID because it has water rights, a Bureau of Reclamation identifier, a US Geological Survey identifier, and a second State of Colorado identifier because real-time data are collected. HydroBase collects data from many sources; however, the State has not attempted in all cases to cross-references the identifiers. For example, a streamflow station may have a partial time series record with a "USGS" data source and identifier and a partial time series record with a "DWR" (Division of Water Resources) data source and identifier – the user must recognize that this may be the same station, under different management at different times.

HydroBase is updated for release to the public approximately once per year, although internal updates may occur year-round. Time series are used with CDSS (Colorado's Decision Support Systems) applications and follow basic time series standards when used by TSTool and other software.

HydroBase and Standard Time Series Properties

The standard time series identifier format for HydroBase time series is of the form:

Location.DataSource.DataType.Interval~HydroBase

Due to the variety of data types, sources, and formats in HydroBase, time series properties can be set a number of ways. General guidelines are as follows:

- The location part of the time series identifier is set to a station or structure identifier, which is typically the identifier used by the managing agency. For example, USGS stream gages will use the 8-digit USGS identifier and State of Colorado diversions will use a structure WDID.
- The source part of the time series identifier corresponds to the current source of the data. For example, if the current provider for a time series is the USGS, then the data source will be USGS. If the State of Colorado has at some point taken over maintenance of a station from the USGS, then the data source will be DWR. Individual data records may indicate a variety of data sources. The convention in HydroBase is to store the data records under the current data source, rather than force the user to query more than one time series and merge the time series. If, however, a station has moved, then separate time series will typically be stored, likely under different identifiers.
- The data type part of the time series identifier as much as possible uses the “measurement type” information in HydroBase or a readable and reasonable data type phrase. For example “Precip” is a measurement type for station data and “DivTotal” (diversion total) is a measurement type for diversion data. In some cases, especially with real-time data, the data type may not exactly match HydroBase. For example, HydroBase uses a measurement type “RT_Rate” for multiple stream related data types. TSTool uses a data type of “Streamflow”. In the past, TSTool and other software used data types that did not as closely match the measurement types in HydroBase. For example, daily streamflow was identified as QME (a National Weather Service notation) because that is how it was defined in CRDSS modeling efforts. The table at the bottom of this appendix describes all available HydroBase data types and provides guidance for upgrading from old data types.
- Data intervals are set based on the tables that are being queried. In most cases, a regular interval like DAY or MONTH is used. IRREGULAR is used for real-time data because there is currently no way to know without doubt what the regular data interval is (e.g., 15MIN). Data that are measured infrequently (e.g., reservoir field measurements) are typically stored as a regular interval time series with interval DAY. This allows more flexibility in data processing and filling.
- In older versions of TSTool, the scenario part of the identifier was sometimes used to supplement the data type information. For example, real-time flow data in the database has a number of attributes (Streamflow, RT_Rate, DISCHRG) that cannot easily fit into the standard time series identifier. The current version of TSTool uses datatype-subdatatype where necessary and generally does not use the scenario for normal time series identifiers (WIS time series are an exception) and this field is being reserved to possibly indicate historical data, filled data, etc.
- Units are set based on the database table definitions.
- Period of record is set based on the available database contents. Periods are typically not determined by checking the data because this would require querying large amounts of data. When listing time series, periods are normally determined from summary information available in the database. In some cases, the period of record information is not saved at a precision sufficient to accurately represent the true period (e.g., the database may indicate data for years but not months). Therefore, the true period will only be available when data are actually queried.
- Missing data are typically set to -999 in time series but are typically stored as nulls in the database.
- The input type of the time series identifier may not be used for older applications. The new convention is being phased in and uses an input type of HydroBase (e.g.,

12345678.USGS.QME.DAY~HydroBase). If multiple HydroBase connections are needed, the input name may also be added (e.g., 12345678.USGS.QME.DAY~HydroBase~ServerName), although this capability is only in the evaluation stage.

- The time scale for data (whether accumulated [ACCM], instantaneous [INST], or mean [MEAN]) is not automatically determined from the data type and interval.

The following tables present a summary of time series identifier fields for the HydroBase data types. Data sources may be added and/or removed with data updates. Data types are listed by major group and are alphabetized by the data type description within the group. The time scale is provided to facilitate data use, in particular when changing the time interval.

HydroBase Time Series Types and Standard Time Series Identifier Fields
Agricultural Crop and Livestock Data

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Agricultural/ CASS	Colorado Agricultural Statistics Service crop area harvested	County Name	CASS	CropAreaHarvested-Commodity_Practice Commodity and practice are from available values in HydroBase.	Year INST	See NASS data for orchards, pasture, and vegetables. Perennial crops usually have only harvested value.
	CASS area planted	County Name	CASS	CropAreaPlanted-Commodity_Practice Commodity and practice are from available values in HydroBase.	Year INST	Annual crops should have planted value but use maximum of planted and harvested if necessary.
	CASS livestock head	County Name	CASS	LivestockHead-Commodity_Type Commodity and type are from available values in HydroBase.	Year INST	For each commodity (e.g., sheep), multiple types (e.g. sheep at various maturity levels).
Agricultural/ GIS	CDSS irrigated lands assessment result. See also Diversion Comments below.	WDID	CDSSGIS	CropAreaAllIrrigation-CropType CropAreaDrip-CropType CropAreaFlood-CropType CropAreaFurrow-CropType CropAreaSprinkler-CropType CropType is taken from available values in HydroBase.	Year INST	Data are only available for years where DSS projects or data refreshes have occurred. Partial data for intermediate years may be available in spatial data layer attributes but not HydroBase. Data are available for lands served by surface water structures, listed by crop/year/irrigation type.
Agricultural/ NASS	CropArea	County Name	NASS	CropArea-Commodity Commodity is taken from available values in HydroBase.	Year INST	See CASS data where available. NASS does not distinguish between planted and harvested. NASS data are useful for orchards, pasture, and vegetables, which may not be reported in CASS.

HydroBase Time Series Types and Standard Time Series Identifier Fields (Climate Data)
Climate Group Table 1 of 2

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Climate	Evaporation (Pan)	Station ID	NOAA	EvapPan Old (obsolete) data type was EPAN.	Day ACCM, Month ACCM	
	Frost Dates (derived from temperatures)	Station ID	COAGM, NOAA	FrostDateL28S, FrostDateL32S, FrostDateF28F, FrostDateF32F Old (obsolete) data type was FrostDate or FrostDates.	Year INST	Time series in software are the Julian day of the year (1-366) to allow graphing, filling, and manipulation.
	Precipitation	Station ID	COAGM, NOAA	Precip Old (obsolete) data type was PTPX.	Day ACCM, Month ACCM, Irregular ACCM	Irregular data are real-time increments.
	Snow (accumulation on ground during interval).	Station ID	NOAA	Snow Old (obsolete) data type was SNOG.	Day ACCM, Month ACCM	
	Snow course depth and snow water equivalent	Station ID	SCS	SnowCourseDepth, SnowCourseSWE Old (obsolete) data type was SnowCrse, SNWE.	Day INST	Values are recorded on a day, with one or more times a month.
	Solar radiation	Station ID	COAGM	Solar Old (obsolete) data type was RADS.	Day ACCM	
	Temperature (instantaneous)	Station ID	various	Temp	Irregular INST	
	Temperature (maximum)	Station ID	COAGM, NOAA	TempMax Old (obsolete) data type was MaxTemp, TAMN.	Day INST	
	Temperature (mean of maximum daily values)	Station ID	COAGM, NOAA	TempMeanMax Old (obsolete) data type was MaxTemp, TAMX with monthly interval.	Month MEAN	
	Temperature (mean)	Station ID	COAGM, NOAA	TempMean Old (obsolete) data type was MeanTemp, TAVG.	Month MEAN	
	Temperature (minimum)	Station ID	COAGM, NOAA	TempMin Old (obsolete) data type was MinTemp, TAMN.	Day INST	
	Temperature (mean of minimum daily values)	Station ID	COAGM, NOAA	TempMeanMin Old (obsolete) data type was MinTemp, TAMN with monthly interval.	Month MEAN	

HydroBase Time Series Types and Standard Time Series Identifier Fields (Climate Data)
Climate Group Table 2 of 2

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Climate	Vapor pressure (mean daily)	StationID	COAGM	VaporPressure Old (obsolete) data type was VP, MVP.	Day MEAN	
	Wind run	Station ID	AGRO, COAGM	Wind Old (obsolete) data type was UDIS.	Day ACCM	

HydroBase Time Series Types and Standard Time Series Identifier Fields (Demographic Data)

Demographic data are related to human population. See the Agricultural Data above for livestock population.

Data Group	Data Type Description	Location	Data Source	Data Type(s)	Available Intervals and Time Scale	Comments
Demographics	Human population (persons)	Area_type-Area_name The type indicates whether a county, municipality, state, etc. The name agrees with the type. The combination defines a unique location.	(blank) This could be assumed from the Pop_type part of the data type; however, the data source is not readily available in HydroBase.	HumanPopulation-Pop_type The population type is Census, Estimated, etc.	Year INST	See CDSS documents for information on how population estimates are determined.

HydroBase Time Series Types and Standard Time Series Identifier Fields (Diversion Data)

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Diversion May include records for reservoir and well structures, as per State of Colorado administration practices. See also reservoir data.	Diversion Class (showing water color)	WDID	DWR	DivClass-SFUT Old (obsolete) data type was DQME, Div, or Diversion.	Day MEAN, Month INST or ACCM, Year INST or ACCM	SFUT is encoded as: S:s F:f U:u T:t s = source f = from u = use t = type Annual values are for irrigation year (Nov-Oct).
	Diversion Comment (the acreage for a diversion and string data flag indicating whether a structure irrigated in a year)	WDID	DWR	DivComment	Year INST or ACCM	The numerical time series value is set to the acreage for the year. The data quality flag is set to the HydroBase <i>diversion_comment_not_used</i> flag. Therefore, this time series can be used to extract total acreage for a structure and determine if diversions should be zero for a year. Annual values are for irrigation year (Nov-Oct).
	Diversion Total (sum of all DivClass records for a structure).	WDID	DWR	DivTotal Old (obsolete) data type was DQME, Div, or Diversion.	Day MEAN, Month INST or ACCM, Year INST or ACCM	Annual values are for irrigation (Nov-Oct) year.

The above table summarizes how diversion records are available as time series. However, to determine a complete diversion time series, it is necessary to understand the various ways that diversion records can be stored. See also the **State of Colorado's Water Commissioner Manual**.

Raw data observations for a diversion structure are stored as one or more of the following forms in HydroBase:

- Daily water class time series. These data are recorded using irrigation year (November to October). If one or more values have been entered in a month, then HydroBase will include a full month of data. Days at the beginning of the irrigation year that have no observed values at the start of the year should be considered to be zero, regardless of values found in previous irrigation years. Once an observation occurs, then days within the month where an observation was not recorded are set to the last observed value. Therefore, if an irrigation year contains at least one value, that irrigation year will have at least one month of values (with no missing in the month). To preserve space in HydroBase, months with no observations are not included in the daily data in the database. If a year has no observation, then no data are available in HydroBase for the year and a determination of whether the data values should be zero or other must be determined using other data (see below) or engineering judgment. **TSTool and StateView by default implement the carry-forward procedure within irrigation years.**
- Diversion comments. Diversion comments may be included for an irrigation year. The `not_used` flag indicates if a diversion was not used in a year. If this is the case, then daily diversion records should not be available and a zero value can be assumed for the water year. **TSTool and StateView DO NOT by default use diversion comments when providing daily or monthly time series.**
- Infrequent water class. Infrequent water class values can be entered as an annual value for the irrigation year, or as a monthly value. The data can be accessed as time series in TSTool, although no specific capabilities have been implemented to supplement the daily or monthly time series.

Processed (derived) data records are created as follows:

- Daily total diversion. Daily water class values are accumulated to daily total records. Similar to the daily water class, any month that has at least one value will result in a month with no missing data. To preserve space in HydroBase, only months that include an observation are included in HydroBase. Other months in the same irrigation year should be carried forward. Irrigation years with no observation have no records in HydroBase and a determination of whether the data values should be zero or other must be made using other data (see below) or engineering judgment. **TSTool and StateView by default implement the carry-forward procedure within irrigation years.**
- Monthly water class. Monthly water class is computed by converting the daily water class values (average CFS) to ACFT for each day of the month, and adding the values. Because of the way that daily data are treated, a month will either have all daily values or none. A month with no data will have its value set to missing in the database. Full irrigation years with no observation will result in a full year of missing values, and a determination of whether the data values should be zero or other must be determined using other data (see below) or engineering judgment. Unlike daily data, monthly diversion records are included in HydroBase for the full data period. Full years of missing values may be included in the database.
- Monthly total diversion. This is derived using the same procedure as monthly water class; however, the daily total diversion is used as input.
- Infrequent data are not considered when producing the monthly total time series.

Therefore, to determine a complete time series, the following must be performed, using TSTool or other software:

Daily time series:

1. Read the daily time series from HydroBase. The default in TSTool and StateView is now to carry forward daily diversion time series within the irrigation year.
2. Utilize the diversion comments to set additional years of data to zero. Using diversion comments is an option with TSTool and StateDMI time series read commands.
3. For years with no data, use an appropriate fill technique. If it is known that the ditch did not operate, then zeros should be used. If it is known that the ditch did operate, use historical averages or some other method to fill the data.
4. HydroBase infrequent diversions could be used to supplement the data, but currently there is no software to help users with this process.

Monthly time series:

1. Read the monthly time series from HydroBase. Any irrigation year with at least one daily observation results in 12 monthly time series values.
2. Utilize the diversion comments to set additional years of data to zero. Using diversion comments is an option with TSTool and StateDMI time series read commands.
3. For years with no data, use an appropriate fill technique. If it is known that the ditch did not operate, then zeros should be used. If it is known that the ditch did operate, use historical averages or some other method to fill the data.
4. HydroBase infrequent diversions could be used to supplement the data, but currently there is no software to help users with this process.

Yearly time series:

1. Infrequent time series can be read by TSTool and can supplement the above data. However, currently there is no software to help users with this process. General TSTool commands must be used as appropriate.

HydroBase Time Series Types and Standard Time Series Identifier Fields (Hardware Data)

Data Group	Data Type Description	Location	Data Source	Data Type(s)	Available Intervals and Time Scale	Comments
Hardware	Battery voltage	Station ID	DWR	Battery	Irregular INST	Limited data are available. This data type allows remote system maintenance checks.

Hardware data types are not commonly available have been implemented as a test and to allow for greater future use.

HydroBase Time Series Types and Standard Time Series Identifier Fields (Reservoir Data)
Reservoir Group Table 1 of 2

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Reservoir	Field Measurements	WDID	DWR, other	ResMeasElev, ResMeasEvap, ResMeasFill, ResMeas Release, ResMeas Storage Old (obsolete) data type was RSTO.	Day INST, Day ACCM, Day ACCM, Day ACCM, Day ACCM	Reservoir measurements are often recorded at the beginning or end of the month.
	Pool Elevation	Station ID or State of CO Abbrev.	DWR, other	PoolElev	Irregular INST	Real-time data for reservoirs are recorded using a station abbreviation that does not match a WDID.
	Release Class (showing water color)	WDID	DWR	RelClass - SFUT	Day MEAN, Month INST or ACCM, Year INST or ACCM	SFUT is encoded as: S:s F:f U:u T:t s = source f = from u = use t = type Annual values are for irrigation year (Nov-Oct).
	Release Comment (the acreage for a release and string data flag)	WDID	DWR	RelComment	Year INST or ACCM	See DivComment comments. Sometimes acreage is associated with reservoirs. Annual values are for irrigation year (Nov-Oct).
	Release Total (sum of all RelClass records for a structure).	WDID	DWR	RelTotal	Day MEAN, Month INST or ACCM, Year INST or ACCM	Annual values are for irrigation year (Nov – Oct).

HydroBase Time Series Types and Standard Time Series Identifier Fields (Reservoir Data)
Reservoir Group Table 1 of 2

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Reservoir	Release (instantaneous)	Station ID	DWR, other	Release	Irregular INST	Real-time data for reservoirs are recorded using a station abbreviation that does not match a WDID.
	Reservoir Storage (end of month).	WDID	USBR, DWR, other	ResEOM Old (obsolete) data type was RSTO.	Month INST	Few time series are available.
	Reservoir Storage (end of year).	WDID	USBR, DWR, other	ResEOY	Year INST	From <i>annual_res</i> table. Annual value is for irrigation year (Nov-Oct).
	Storage (instantaneous)	Station ID or State of CO Abbrev.	DWR, other	Storage	Irregular INST	Real-time data for reservoirs are recorded using a station abbreviation that does not match a WDID.

HydroBase Time Series Types and Standard Time Series Identifier Fields (Stream Data)

Data Group	Data Type Description	Location	Data Source	Data Type	Available Intervals and Time Scale	Comments
Stream	Natural Flow	Station ID	USBR	NaturalFlow Old (obsolete) data type was Nat_flow, NQME	Month INST or ACCM	
	Stage	Station ID	DWR, other	Stage	Irregular INST	Real-time data.
	Streamflow	DWR abbreviation or USGS station ID	DWR, USGS, other	Streamflow Old (obsolete) daily, monthly data type was QME. Old real-time data type used RT_rate and scenario DISCHRG or other VAXfield to indicate channel.	Day MEAN, Month INST or ACCM, Irregular INST	Real-time data use Irregular time interval.
	Streamflow (maximum of daily mean)	Station ID	DWR, USGS	StreamflowMax Old (obsolete) data type was Maxq, Maxflow.	Month INST	
	Streamflow (minimum of daily mean)	Station ID	DWR, USGS	StreamflowMin Old (obsolete) data type was Minq, Minflow.	Month INST	
	Water temperature (instantaneous)	Station ID, State of CO abbreviation	DWR, other	WatTemp	Irregular INST	Real-time data, using identifier that does not match USGS or other identifier for historical data.

HydroBase Time Series Types and Standard Time Series Identifier Fields
(Water Information Sheet Data)

Data Group	Data Type Description	Location	Data Source	Data Type	Available Interval and Time Scale	Comments
WIS	Water Information Sheet (WIS) cell values, over time	WIS row identifier. For example, structures have an identifier wdid>NNNN NNN, where the leading “wdid:” is a literal string and the following information is the actual WDID. Similarly, stations start with “stat:”, followed by a station ID; confluences with “conf:”, followed by the HydroBase <i>wd_water</i> numbers for the tributary and the larger stream; other row types with “othr:”, followed by a sequential number in the WIS.	DWR	Data types match the WIS columns, as follows: WISPointFlow, WISNaturalFlow, WISDeliveryFlow, WISGainLoss, WISRelease, WISPriorityDiversion, WISDeliveryDiversion, WISTribNaturalFlow, WISTribDeliveryFlow, WISDryRiver (not currently implemented – may be implemented as a data flag in the future).	Day MEAN	The scenario part of the time series identifier is set to the sheet name. Over time, WIS with a particular sheet name may be modified in format. The combination of sheet name and row identifier can be used to find data. The time series description is set to the row label. Data values are as stored for the WIS, which reflect the gain method used when the sheet was stored.

**HydroBase Time Series Types and Standard Time Series Identifier Fields
(Well Data)**

Data Group	Data Type Description	Location	Data Source*	Data Type	Available Intervals and Time Scale	Comments
Well	Well level (elevation)	Location identifier, based on the current data source. For example, if the data source is USGS, the location identifier will be the USGS identifier.	BJORKLUND CH2MHILL CSU CWSD DWR FOX HALAPASKA HILLIER MCCONAGHY NELSON ROBSON ROBSONBANT SCHNEIDER SEO SMITH SOUTHMETRO SPDSS USGS USGS_NAWQA WILSON *as of 2005-06-16	WellLevel	Day INST, Irregular INST	Daily data are historical measurements, often at the ends of a month. A well may have multiple identifiers. However, the identifier presented in TSTool is that corresponding to the current data source. Use StateView to see alternate identifiers for the location, to cross-reference with data outside of HydroBase. Irregular data are real-time using state station abbreviations, which do not match the identifier for historical data.

Limitations

HydroBase has the following limitations related to time series storage:

- The station and structure measurement types and time series tables defined in HydroBase do not always allow information to be determined from database records. Instead, some time series properties must be hard-coded based on the table design. For example, the *meas_type* table has a MeanTemp, MaxTemp, MinTemp types defined, but these refer primarily to the separate daily tables for such data. The *monthly_temp* table includes *avg_max_t*, *avg_min_t*, and *mean_t* fields that do not correspond one-to-one with *meas_type* values. Therefore, applications like TSTool use data types that are not specifically defined as strings in HydroBase, which have consequently been hard-coded. This is an issue with station and structure time series.
- Real-time data types in HydroBase do not directly translate to time series data types used in TSTool. An effort has been made to be as consistent as possible while using data types that can be understood by users.
- Data units are not defined consistently in tables. Some tables have a units string and others do not and the units abbreviations are not always consistent (units of “A” are often used for acre-feet and “C” for CFS). A master units table is not used in HydroBase to enforce data units consistency throughout the database.
- The time scale for time series (whether accumulated, instantaneous, or mean) is not automatically determined from the data type and interval. Users must understand how to interpret the data, in particular when changing the data interval.

Appendix

RiverWare Input Type

2004-07-27, Acrobat Distiller

Overview

RiverWare is a river and reservoir model developed by the Center for Advanced Decision Support for Water and Environmental Systems (CADSWES) at the University of Colorado. RiverWare uses data management interfaces (DMIs) to read time series data from various formats at run-time. The format described in this appendix is a standard time series format that is imported into the RiverWare data sets and can be output during runs. The example below shows the format of a file. Refer to the **RiverWare Data Management Interface** documentation for more information. Important comments about the file format are:

- The file is divided into a header section (top) and data section (bottom). Comments can occur anywhere in the file and are lines starting with #.
- The data period is defined by the `start_date` and `end_date` keywords. Date/times must include hours and minutes regardless of the date/time precision (the more precise information is ignored if not needed).
- The data interval is defined by the `timestep` keyword and consists of an integer multiplier and a base interval string, separated by a space. Recognized intervals are HOUR, DAY, WEEK, MONTH, and YEAR.
- The data units are specified using the `units` keyword and are the units after the scale is applied. The `scale` keyword indicates a value that should be applied to the data values to result in the specified units. For example, a data value of 1.5 with units of cfs and a scale of 1000 will result of a value of 1500 cfs in memory.
- Optional `set_units` and `set_scale` keywords may be used similar to `units` and `scale` to indicate the units and scale to be converted to when data are read. These properties can be written by TSTool's `writeRiverWare()` command but currently are not evaluated by TSTool when reading data.

The following example illustrates the format of a RiverWare file.

```
# Comments
start_date: 1903-01-01 06:00
end_date: 2001-12-31 24:00
timestep: 6 HOUR
scale: 1
set_scale: 1
units: ft
set_units: ft
1356.00
1356.00
1356.00
1356.00
1356.00
NaN
NaN
...
...
```

RiverWare Files and Standard Time Series Properties

The standard time series identifier for RiverWare time series files is of the form:

```
Location..DataType.Interval~RiverWare~PathToFile
```

RiverWare time series files contain limited information to assign to standard time series properties. The following assignments are made:

- The location part of the identifier is taken from the first part of the file name. It is assumed that the file name is of the form *ObjectName.SlotName*.
- The data source part of the time identifier is left blank.
- The data type is taken from the second part of the file name. It is assumed that the file name is of the form *ObjectName.SlotName*.
- The data interval is determined from the timestep property in the file.
- The data units are determined from the units property in the file. Currently the set_units property is not evaluated when reading data.
- The missing data value is assigned to NaN (not a number).
- The description is set to the location, a comma, followed by the data type.

Limitations

RiverWare files have the following limitations:

- RiverWare time series files require that units be spelled exactly as required by RiverWare, including upper/lower case. TSTool currently does not know about RiverWare units and therefore commands like `writeRiverWare()` must be used to verify that the units are correct for RiverWare.
- Only one time series can be saved in a file (other RiverWare files support multiple time series and may be supported in the future).
- RiverWare files do not store the data type or location information for the time series. These values are assigned from the file name, as described above. Relying on a file name convention may cause errors if the convention is not followed.
- Data lines do not contain the date. Therefore it is difficult to use the files in other applications without first assigning dates for all the values.

Appendix

StateCU Input Type

2004-05-27, Acrobat Distiller

Overview

The StateCU time series input type corresponds to the file formats used by the State of Colorado's StateCU consumptive use model, including:

- crop pattern time series file, yearly (*.cds)
- irrigation water requirement, formatted for StateMod (*.ddc)
- historical direct diversions, monthly (*.ddh) and daily (*.ddd) (in StateMod format but treated as StateCU input when used with a StateCU data set)
- irrigation practice time series, yearly (*.ipy)
- irrigation water requirement (IWR, *.iwr) and water supply limited (WSL, *.wsl) output report files, including monthly and yearly values
- frost dates, yearly
- precipitation, monthly (in StateMod format)
- temperature, monthly (in StateMod format)

See also the StateMod input type, which corresponds to StateMod input files, and the StateModB input type, which corresponds to StateMod binary output files.

See the **StateCU Documentation** for a complete description of StateCU input files. Refer to the **StateMod Documentation** for files that use the StateMod file format.

Important comments about the StateCU file formats are:

- Input files are divided into a header section (top) and data section (bottom). Comments can occur only at the top and are lines that begin with #.
- One or more time series can be stored in a file.
- Consistency in the order and number of the stations is required for each year of data, within the file.
- Other than comments, input files are fixed-format, compatible with FORTRAN applications. See the **StateCU Documentation** and **StateMod Documentation** for field specifications.
- Input file formats are optimized to allow a full year of data to be read for the entire data set. Reading a time series for a single location for the full period requires reading through the entire file.
- The precision of data values may be controlled by software, resulting in more or fewer fractional digits. This may lead to round-off differences when comparing raw data values output by the software.

StateCU Files and Standard Time Series Properties

The standard time series identifier for StateCU files is of the form:

```
Location.StateCU..Month~StateCU~PathToDDCFile (for DDC file)
Location.StateCU.CropArea-AllCrops.Year~StateCU~PathToIWRReport (for IWR report acreage)
Location.StateCU.IWR.Month~StateCU~PathToIWRReport (for IWR report monthly IWR)
Location.StateCU.IWR.Year~StateCU~PathToIWRReport (for IWR report yearly IWR)
Location.StateCU.IWR_Depth.Year~StateCU~PathToIWRReport (for IWR report IWR depth)
Location.StateCU.CropArea-AllCrops.Year~StateCU~PathToWSLReport (for WSL report acreage)
Location.StateCU.WSL.Month~StateCU~PathToWSLReport (for WSL report monthly WSL)
Location.StateCU.WSL.Year~StateCU~PathToWSLReport (for WSL report yearly WSL)
Location.StateCU.WSL_Depth.Year~StateCU~PathToWSLReport (for WSL report WSL depth)
Location.StateCU.DataType.Interval~StateCU~PathToFile (historical time series data files)
```

StateCU files contain limited header information (e.g., period of record but no data type). Time series properties are set using the following guidelines:

- For input files, the location part of the time series identifier is taken from the identifier field in the data records (from the first year of data). A change in the year indicates that all time series have been identified. For output files, the location is identified by lines that start with an underscore (this allows the StateCU interface to search for identifiers in output).
- The source part of the time series identifier is set to `StateCU` or blank.
- The data type may not be assigned because it is not defined in the file (e.g., temperature and precipitation time series). Currently no interpretation of the file name extension occurs. Some specific applications may set the data type, based on reading a StateCU data set response file (and therefore knowing the specific contents of the file).
- The data interval is assigned as `Day`, `Month`, or `Year` based on the file format (determined automatically).
- The scenario is typically not assigned.
- The input type part of the time series identifier is set to `StateCU`, indicating the file format. Software will use the interval and/or examine the file contents to verify whether the data are in daily or monthly format.
- The input name part of the time series identifier is set to the file name, either as the full path or a relative path to the working directory.
- The units are assigned to those indicated in the file header or based on the file type.
- The missing data value is assigned to `-999.0`.
- The description is set to the same value as the location. A verbose description can typically be determined by cross-referencing the identifier with another StateCU data file (e.g., CU Locations, `*.str`).
- The period is set based on the header information.

Limitations

StateCU files have the following limitations:

- The formats of the files do not facilitate extracting one time series from the file. Software has been optimized to perform this within current constraints.
- Some time series properties are not explicitly included in StateCU files (e.g., data type). Therefore, general software like TSTool may not be able to provide default information. For example, a graph may show multiple time series with nearly the same legend text because more detailed information cannot be defaulted. The following has not been implemented but may be in the future: DDC file data type = IWR.
- Although the complete output report files contain all values needed to evaluate water balance, these values are not available in files that can be easily read as time series. Currently the verbose reports are not available for reading as part of the StateCU input type.

This page is intentionally blank.

Appendix

StateMod Input Type

2004-07-27, Acrobat Distiller

Overview

The StateMod time series input type corresponds to the file format used by the State of Colorado's StateMod model, including standard daily, monthly, average monthly (referred to as annual in the StateMod documentation) file formats. See also the StateModB input type, which corresponds to StateMod binary output files and the StateCU input type, which corresponds to the State of Colorado's StateCU consumptive use model.

The following example illustrates the format of the three main file formats. See the **StateMod Documentation** for a complete description of StateMod input files. Important comments about the file format are:

- The file is divided into a header section (top) and data section (bottom). Comments can occur only at the top and are lines that begin with #.
- One or more time series can be stored in a file.
- Consistency in the order and number of the stations is required for each year of data, within the file.
- Other than comments, the file is fixed-format, compatible with FORTRAN applications. See the **StateMod Documentation** for field specifications.
- The format is optimized to allow a full year of data to be read for the entire data set. Reading a time series for a single location for the full period requires reading through the entire file.
- In addition to the required values, a total/average value is accepted as the far-right value on each data line. This value may be ignored by applications (it can be computed from the data values on the line if necessary).
- The precision of data values may be controlled by software, resulting in more or fewer fractional digits. This may lead to round-off differences when comparing raw data values with the total/average in the optional end column.

```
# StateMod time series files can have 3 main forms (monthly, average monthly, daily) as
# described below. The order of time series is important for
# some files (e.g., order of diversion time series should match order of
# diversion stations in .dds file); however, StateMod is being updated over
# time to remove this requirement). Different StateMod input files have
# slight variations on the general format (e.g., the reservoir target file
# has two time series for each reservoir for minimum and maximum targets).
# Missing data are typically indicated by -999.
# The generic extension for StateMod time series files is .stm, although specific
# extensions are used in a StateMod data set.
#
# 1) This is an example of a StateMod monthly time series for water year data:
#
# Comments are lines at the top of the file starting with the # character.
# The header may contain software-generated comments about the time series.
# The remainder of the file is fixed format, with the first non-comment
# line being a header with the following elements (i5,1x,i4,5x,i5,1x,i4,a5,a5):
#
# Beginning month (1=Jan)
# Beginning year (4-digit)
# Ending month
# Ending year
```

StateMod Input Type

```
# Data units (AF/M, ACFT, CFS or ""), where rates are for diversions and
# flow, and volume is for reservoir contents. Units are not used for
# dimensionless data (like weight or percent).
# Year type (CYR=calendar, WYR=water, IYR=irrigation)
#
# Data lines then follow with:
# Year Station 12-monthly-values year-total/average (i4, 1x, a12, 12f8, f10)
# The year value is optional and is generally not read as input but is
# computed for output. The year in data lines corresponds to the calendar type.
# An example follows:
  10/1926 - 9/1998 ACFT  WYR
1927 08236000    1229.8   892.6   922.3    737.9    555.4    922.3   7049.4  32263.6
31000.1 14541.0  5662.9   8326.7  104104.0
1927 08235250    -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -
999.0   -999.0   -999.0   -999.0     0.0
1927 08235700    -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -
999.0   -999.0   -999.0   -999.0     0.0
1927 08236500    1047.3    595.1    614.9    614.9    555.4    1900.2   6769.7  31226.2
20338.8 14777.1  9465.3   4476.8   92381.5
...
#
# 2) This is an example of a StateMod average monthly time series for water year data:
#
# The average monthly time series is a pattern of twelve monthly values
# that are applied for each year in the period.
# The format is exactly the same as a monthly time series; however, the
# years in the header should be set to zero and year and month are ignored in data rows
# and can therefore be blank.
#
# An example follows:
  10/ 0 - 9/ 0 ACFT  WYR
  08236000    1229.8   892.6   922.3    737.9    555.4    922.3   7049.4  32263.6
31000.1 14541.0  5662.9   8326.7  104104.0
  08235250    -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -
999.0   -999.0   -999.0   -999.0     0.0
  08235700    -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -999.0   -
999.0   -999.0   -999.0   -999.0     0.0
  08236500    1047.3    595.1    614.9    614.9    555.4    1900.2   6769.7  31226.2
20338.8 14777.1  9465.3   4476.8   92381.5
...
#
# 3) This is an example of a StateMod daily time series for water year data:
#
# The daily time series is similar to the monthly time series except that
# a year and month are included on the data lines and 28, 30, or 31 daily
# data values can occur on each line (end values ignored, depending on month).
# The data format is (i4, i4, 1x, a12, 31f8, f8). The month total/average
# is optional and is generally read as input but is computed for output.
# Regardless of the calendar type in the header, the year and month in data records use
# calendar year (month 1 = January).
#
# An example follows:
  10/1926 - 9/1998 ACFT  WYR
1926 10 08236000    -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -
999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -
999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -
0.00      0.00
...
1927 4 08236000     38.00    42.00    42.00    67.00    90.00    90.00   100.00   118.00
93.00    80.00    93.00    80.00    80.00    80.00    80.00    68.00    80.00    68.00
68.00    80.00    80.00   106.00   136.00   170.00   229.00   250.00   296.00   322.00   348.00
0.00    114.65
1927 4 08235250    -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -
999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -
999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00  -999.00
0.00      0.00
...
```

StateMod Files and Standard Time Series Properties

The standard time series identifier for StateMod files is of the format:

Location...Interval~StateMod~PathToFile

StateMod files contain limited header information. Time series properties are set using the following guidelines:

- The location part of the time series identifier is taken from the identifier field in the data records (from the first year of data). A change in the year indicates that all time series have been identified.
- The data source part of the time series identifier is set to StateMod or blank. In the past this information was used to indicate the input type (file format) in the time series identifier; however, the new input type notation has a specific field for the input type and therefore data source can be used more appropriately. In the future, it may be possible to pass along the original input source but this information cannot currently be saved in the StateMod file format.
- The data type is often not assigned because it is not defined in the file. Currently no interpretation of the file name extension occurs. Some specific applications (e.g., the StateMod GUI) may set the data type, based on reading a StateMod data set response file (and therefore knowing the specific contents of the file).
- The data interval is assigned as Day or Month based on the file format (determined automatically).
- The scenario is typically not assigned. Older software may use the scenario to store the file name; however, the new time series identifier notation stores the file name as the input name field (see below).
- The input type part of the time series identifier is set to StateMod, indicating the file format. Software will use the interval and/or examine the file contents to verify whether the data are in daily or monthly format.
- The input name part of the time series identifier is set to the file name, either as the full path or a relative path to the working directory.
- The units are assigned to those indicated in the file header.
- The missing data value is assigned to -999 . 0.
- The description is set to the same value as the location. A verbose description can typically be determined by cross-referencing the identifier with another StateMod data file (e.g., diversion stations).
- The period is set based on the header information.

Limitations

StateMod files have the following limitations:

- The format of the does not facilitate extracting one time series from the file. Software has been optimized to perform this within current constraints.
- Some time series properties are not explicitly included in StateMod files (e.g., data type). Therefore, general software like TSTool may not be able to provide default information. For example, a graph may show multiple time series with nearly the same legend text because more detailed information cannot be defaulted.
- If two time series for the same station are stored in the same file (e.g., reservoir maximum and minimum targets), there is no way to uniquely identify the two time series. The application or user must understand the file type and data organization. Some specific software (e.g., StateMod GUI) may be able to recognize the specific format.

This page is intentionally blank.

Appendix

StateModB Input Type (StateMod Binary Output Files)

2004-07-27, Acrobat Distiller

Overview

The StateModB time series input type corresponds to the file format used by the State of Colorado's StateMod model, in particular the binary FORTRAN direct access output files. These files contain important water balance information for every node in the model network. The following table summarizes the contents of the binary files and corresponding text report files (all files can be large for large data sets):

Node Type	Monthly Binary File	Monthly Report File	Daily Binary File	Daily Report File
Diversion	*.b43	*.xdd	*.b49	*.xdy
Instream flow	*.b43	*.xdd	*.b49	*.xdy
Reservoir	*.b44	*.xre	*.b50	*.xry
Stream gage and Stream estimate	*.b43	*.xdd	*.b49	*.xdy
Well	*.b42	*.xwe	*.b65	*.xwy

The following documentation describes the format of the B43 binary file. Other files are similar. See the **StateMod Documentation** for a complete description of StateMod output files. Important comments about the file format are:

- The file is generated by StateMod as a direct access binary file with fixed-length records. The record length is 140 bytes.
- The file is divided into a header section (top) and data section (bottom).
- The format is optimized to allow a full year of data to be read for the entire data set. Efficiently reading a time series for a single location for the full period requires reading appropriate lines of the file using direct access. Because the file is binary and consistent for a given data set, file reads can be optimized.
- The data period and the calendar year type are consistent with the StateMod control file.
- All character strings are left justified and are padded with spaces. Therefore, software that reads the file should trim trailing spaces after reading the strings.
- River node identifiers in record 5 are included for all nodes in the network and data records (record 11) follow this order. Subsequent lists for various node types are a subset of the list in record 5 and have data items to reference the position in the river node list. Time series are queried using the identifiers in records 6+. However, the river node position is actually used to retrieve data in the file.

The B43 binary file contains the following records:

Record	Field	StateMod Variable	Type	Description
1	1	iystr0	integer	Beginning year of simulation, for year type in StateMod control file.
	2	iyend0	integer	Ending year of simulation, for year type in StateMod control file.
2	1	numsta	integer	Number of river nodes.
	2	numdiv	integer	Number of direct diversions.
	3	numifr	integer	Number of instream flows.
	4	numres	integer	Number of reservoirs.
	5	numown	integer	Number of reservoir owners.
	6	nrsact	integer	Number of active reservoirs.
	7	numrun	integer	Number of base flow nodes.
	8	numdivw	integer	Number of diversion structures with wells.
	9	numdxw	integer	Number of well only structures.
3	1	xmonam(14)	Each is char(4).	Month names corresponding to the calendar type for the simulation. This information is provided as a convenience for data processing. For example, if the year type is WYR (water year), xmonam(1) is 'OCT'. The 13 th value is 'TOT' and the 14 th value is 'AVE'.
4	1	mthday(12)	Each is integer.	Number of days per month, corresponding to the calendar type for the simulation. This information is provided as a convenience for data processing and to convert daily data values to monthly. For example, if the year type is WYR (water year), mthday(1) is 31 for October. The number of days in February is typically 28 and is used for all data processing, regardless of whether a year is a leap year.
5 Repeat record for numsta	1	j	integer	Counter for record type 5.
	2	cstaid(j)	char(12)	River node identifiers.
	3	stanam(j)	real(6)	River node names.
6 Repeat record for numdiv	1	j	integer	Counter for record type 6.
	2	cdivid(j)	char(12)	Diversion identifier.
	3	divnam(j)	real(6)	Diversion name.
	4	idvsta(j)	integer	River node position (1+) to allow cross-reference with river nodes.
7 Repeat record for numifr	1	j	integer	Counter for record type 7.
	2	cifrid(j)	char(12)	Instream flow identifier.
	3	xfrnam(j)	real(6)	Instream flow name.
	4	ifrst(j)	integer	River node position (1+) to allow cross-reference with river nodes.
8 Repeat record for numres	1	j	integer	Counter for record type 8.
	2	cresid(j)	char(12)	Reservoir identifier.
	3	resnam(j)	real(6)	Reservoir name.
	4	irssta	integer	River node position (1+) to allow cross-reference with river nodes.

9 Repeat record for numrun	1	j	integer	Counter for record type 9.
	2	crunid(j)	char(12)	Base flow node identifier.
	3	runnam(j)	real(6)	Base flow node name.
	4	irusta(j)	integer	River node position (1+) to allow cross-reference with river nodes.
10 Repeat record for numdivw	1	j	integer	Counter for record type 10.
	2	cdividw(j)	char(12)	Well identifier.
	3	divnamw(j)	real(6)	Well name.
	4	idvstw(j)	integer	River node position (1+) to allow cross-reference with river nodes.
11 Repeat record for every river node numsta, for every month of the simulation. See the StateMod documentation for a full description of parameters. Parameters are grouped as shown in the *.xdd file.	1	dat(1)	real	Demand
	2	dat(2)	real	Demand
	3	dat(3)	real	Water Supply
	4	dat(4)	real	Water Supply
	5	dat(5)	real	Water Supply
	6	dat(6)	real	Water Supply
	7	dat(7)	real	Water Supply
	8	dat(8)	real	Water Supply
	9	dat(9)	real	Water Supply
	10	dat(10)	real	Water Supply
	11	dat(11)	real	Water Supply
	12	dat(12)	real	Shortage
	13	dat(13)	real	Shortage
	14	dat(14)	real	Water Use
	15	dat(15)	real	Water Use
	16	dat(16)	real	Water Use
	17	dat(17)	real	Water Use
	18	dat(18)	real	Station In/Out
	19	dat(19)	real	Station In/Out
	20	dat(20)	real	Station In/Out
	21	dat(21)	real	Station In/Out
	22	dat(22)	real	Station In/Out
	23	dat(23)	real	Station Balance
	24	dat(24)	real	Station Balance
	25	dat(25)	real	Station Balance
	26	dat(26)	real	Station Balance
	27	dat(27)	real	Available Flow
	28	dat(28)	real	Structure type (Na): <ul style="list-style-type: none"> • < 0 = Baseflow node (e.g., -10001 indicates a diversion that is a baseflow node). • 0 = Well only. • 1-5000 = Diversion • 5001 – 7500 = Instream flow • 7501 – 10000 = Reservoir
	29	dat(29)	real	Number of structures at this node (typically 1).

StateMod B43 Files and Standard Time Series Properties

The standard time series identifier for StateMod binary time series is of the form:

Location.StateMod.DataType.Interval~StateModB~PathToFile

Time series properties are set using the following guidelines:

- The location part of the time series identifier is taken from the identifier field in the data. The identifier for the specific node type (e.g., diversion) is used, not the river node identifier. The river node identifier is often the same as for the specific node type, but this is not a requirement within StateMod.
- The data source part of the time series identifier is set to StateMod, because StateMod has created the output time series.
- The data type is assigned as the parameter name (see record 11 above, without using the group).
- The data interval is assigned as Month or Day, depending on the file extension.
- The scenario is set to blank.
- The input type is set to StateModB.
- The input name is set to the name of the file.
- The units for daily data are assigned as CFS. The units for monthly data in the files are average CFS for the month and are converted to ACFT, assuming a constant number of days per month, as read from record 4. February normally has 28 days per month in the header and therefore leap years have one fewer days than actual.
- The missing data value is assigned to -999.0.
- The description is set to the node name.
- The period is set based on the header information in record 1 (for the year) and record 3 (to determine the start and end months, based on the calendar type).

Limitations

StateMod binary files have the following limitations:

- The file does not contain a format version; therefore, it is difficult for software to handle changes in the file format.
- The file does not contain header information indicating the source of the file (e.g., the creation date, user, directory, StateMod response file, command line). Therefore, it is difficult to know with certainty how a file was created.
- Leap years are not explicitly handled with 29 days.
- Baseflow nodes in record 9 may have the same identifier as other nodes because any node can be a baseflow node. This can be confusing since software may list the node in more than one list. The software that reads the file filters out duplicate time series identifiers to try to resolve this problem.
- This documentation is limited in that it presents the file format only for the *.b43 file. Additional documentation may be added in the future.

Appendix

USGSNWIS Input Type

2004-07-27, Acrobat Distiller

Overview

The USGSNWIS time series input type corresponds to the United States Geological Survey (USGS) National Water Information System (NWIS) format. A number of formats are available but currently only the surface water daily format is supported. Data files can be created by saving USGS web site data to a text file. The example below shows the format of a daily surface water file. Important comments about the file format are:

- The file is divided into a header section (top) and data section (bottom). Comments can occur only at the top and are lines that begin with #.
- Optional data flags are saved with the data values, if available (e.g., e indicates estimated data). Applications like TSTool may include features to use the data flags.
- HTML remnants may be present at the end of the file. These lines are stripped out during time series processing.

The following example illustrates the format of a USGS NWIS file.

```
#  
# U.S. Geological Survey  
# National Water Information System  
# Retrieved: 2002-01-28 13:35:25 EST  
#  
# This file contains published daily mean streamflow data.  
#  
# This information includes the following fields:  
#  
# agency_cd    Agency Code  
# site_no      USGS station number  
# dv_dt        date of daily mean streamflow  
# dv_va        daily mean streamflow value, in cubic-feet per-second  
# dv_cd        daily mean streamflow value qualification code  
#  
# Sites in this file include:  
# USGS 03451500 FRENCH BROAD RIVER AT ASHEVILLE, NC  
#  
#  
agency_cd      site_no dv_dt      dv_va      dv_cd  
5s            15s     10d      12n       3s  
USGS          03451500           1895-10-01      740  
USGS          03451500           1895-10-02      740  
...  
USGS          03451500           1985-01-20      1100      e  
USGS          03451500           1985-01-21      1100      e  
USGS          03451500           1985-01-22      1100      e  
...  
USGS          03451500           2000-09-28      675  
USGS          03451500           2000-09-29      597  
USGS          03451500           2000-09-30      550  
<font face="Arial" size=2>  
<p>Microsoft VBScript runtime </font> <font face="Arial" size=2>error '800a01a8'</font>  
<p>  
<font face="Arial" size=2>Object required: 'db'</font>  
<p>  
<font face="Arial" size=2>/ctp_workgroup/cgi-bin/includes/Inc_htm_utils.asp</font>  
<font face="Arial" size=2>, line 217</font> <font face="Arial" size=2>  
<p>Microsoft VBScript runtime </font> <font face="Arial" size=2>error '800a01a8'</font>  
<p><font face="Arial" size=2>Object
```

USGSNWIS Files and Standard Time Series Properties

The standard time series identifier for USGS NWIS time series is of the form:

```
Location.DataSource.DataType.Interval~USGSNWIS~PathToFile
```

It is difficult to automatically assign standard time series properties from a USGS NWIS file. The limited support of this file format assumes the following:

- The location part of the time series identifier is taken from the second field (`site_no`) in the data records.
- The source part of the time series identifier is taken from the first field (`agency_cd`) in the data records.
- The data type is assigned as `Streamflow` (interpretation of the verbose `dv_va` field in the header is not implemented).
- The data interval is assigned as `1Day` (interpretation of the verbose `dv_va` field in the header is not implemented).
- The input type is set to `USGSNWIS`, indicating the format of input.
- The input name is set to the absolute or relative path to the file.
- The Units are assigned as `CFS`.
- The missing data value is assigned to `-999.0` (gaps in data records will result in this value).
- The description is set to the information after the `Sites in this file include:` line. It is assumed that only one time series per file is used.

Limitations

USGSNWIS files have the following limitations:

- Riverside Technology, inc. is working to support the standard USGS file format(s). Limited information is available for the file specifications. Currently only the daily surface water format has been tested.
- Additional specific limitation will be listed when file format specifications are fully determined.
- The period for the data is not available in the file header. Therefore the period is determined from the first and last dates in the data records. This introduces a slight performance penalty.
- Although data flags are read in for use by applications, no standard flag values are enforced (the end user will need to know the meaning of the flags to use them properly).

Appendix

TSView - Time Series Viewing Tools

Color, 2005-08-05, Original Maintained with TSTool, Acrobat Distiller

Overview

Time Series Terminology

Time Series Properties Interface

- Time Series Properties – General
- Time Series Properties – Comments
- Time Series Properties – Period
- Time Series Properties – Limits
- Time Series Properties – History
- Time Series Properties – Data Flags

Time Series Traces

Time Series Views

Time Series Graph View

- Line Graph
- Line Graph – Log Y Axis
- Bar Graph
- Double Mass Curve
- Duration Graph
- Period of Record Graph
- XY-Scatter Graph

Time Series Product Properties

- Product Properties – General
- Product Properties – Titles
- Product Properties – Layout
- Graph Properties – General
- Graph Properties – Graph Type
- Graph Properties – Titles
- Graph Properties – X Axis
- Graph Properties – Y Axis
- Graph Properties – Label
- Graph Properties – Legend
- Graph Properties – Zoom
- Graph Properties – Analysis
- Graph Properties – Annotations
- Time Series Properties – General
- Time Series Properties – Graph Type
- Time Series Properties – Axes
- Time Series Properties – Symbol
- Time Series Properties – Label
- Time Series Properties – Legend
- Time Series Properties - Analysis

Changing a Graph Page Layout

Time Series Summary View

Time Series Table View

Time Series Product Reference

Overview

The TSView package contains integrated software components that can be used with software applications to enable time series viewing capabilities. The main purpose of the TSView package is to provide simple, consistent, and flexible displays that can be used in a variety of applications with little or no reconfiguration. TSView also provides features to configure and process time series products (e.g., graphs), where the time series data are stored separately from the configuration information.

The TSView package has been developed by Riverside Technology, inc., using Java technology. TSView interfaces can be embedded in Java applications and can be used in web pages either as embedded applets or stand-alone windows. TSView tools operate similarly on Microsoft Windows and UNIX operating systems.

This appendix describes general TSView features and can be used as a reference for how to configure and use TSView components. Software program documentation may include specific information about using TSView features.

Time Series Terminology

The TSView package treats time series as objects that can be read, manipulated, and output in various formats. A time series is defined as having header information (attributes) and data, which usually consists of a series of date/time versus data pairs. Internally, time series are considered to have either regular interval (equal spacing of date/time) or irregular interval (e.g., occasional observations). Regular time series lend themselves to simpler storage and faster processing because date/time information can be stored only for the endpoints. The following basic attributes are stored for each time series:

- data interval as an interval base (e.g., Month, Hour) and multiplier (e.g., 1 for month, or 24 for hour) - in many cases, the multiplier is 1 and is not shown in output (e.g., Month rather than 1Month),
- data type (e.g., Streamflow), which ideally can be checked to determine if a time series contains mean, instantaneous, or accumulated values,
- units (e.g., CFS), which ideally can be used to make units conversions and look up precision for output,
- period of record, using dates that are of an appropriate precision for the interval,
- data limits (the maximum, minimum, etc.),
- description (generally a station, structure, or sensor name),
- missing data value (used internally to mark missing data and trigger data filling, often -999),
- comments (often station comments, if available),
- genesis history (a list of comments about how the time series was created).

In order to uniquely and consistently identify time series, a multi-part *time series identifier* is employed, having the following parts:

- location (or location-sublocation)
- data source
- data type (or datatype-subdatatype)
- data interval (time step)
- scenario

and optionally:

- sequence number (currently being evaluated)
- input type
- input name

These time series attributes are typically concatenated into a time series identifier string. The following example illustrates how the basic identifier parts can be used (without input type and name):

```
12345678.USGS.Streamflow.DAY.HIST
```

The above example identifies a USGS streamflow gage identified as location 12345678, at which historic average daily flow data are available. If possible, data types appropriate for the input type should be used to avoid confusion; however, time series file input types often do not contain a simple data type abbreviation (see the input type appendices in the TSTool documentation for more information). The above example illustrates that the scenario can be used to qualify the data (in this case as historic data, HIST). The scenario is often omitted. When the scenario is used, it often indicates some specific condition (e.g., FLOOD, DROUGHT, HIST, FILLED)

The optional input type and input name are used to specify the time series input format and storage location, especially in cases where the identifier is saved in a file and the input type is needed for later processing. For example:

```
12345678.USGS.Streamflow.DAY.HIST~USGSNWIS~C:\data\12345678.txt  
12345678.USGS.Streamflow.DAY~HydroBase
```

The first example illustrates a time series identifier for a USGS National Water Information System data file. The second example illustrates the identifier for the same time series, in the HydroBase database. Using the input parts of the identifier allows software to transparently locate the data, and for the above examples, would allow the time series to be read from each input source and compared.

The use of the input type and name is being phased into TSView and related components. Input types that have been added to software more recently (e.g., as of version 05.04.00 of the TSTool application) use the new convention and older input types are being updated accordingly. The TSTool appendices that describe each input type identify issues with compatibility.

Using the above time series identifier convention omits use of time series attributes like the period of record and the units, even though these attributes could conceivably be used to distinguish between time series that are otherwise the same. Instead, it is assumed that the period of record and units can be determined from the input and do not need to be part of the identifier. If necessary, different input files can be used to further differentiate time series.

The TSView components use the time series identifiers extensively to locate and manage time series. For example, graph properties for each time series are cross-referenced to time series by using the identifiers. Perhaps most importantly, the time series identifiers as simple strings can be stored in files and can be used by a variety of software to consistently and reliably locate data for processing.

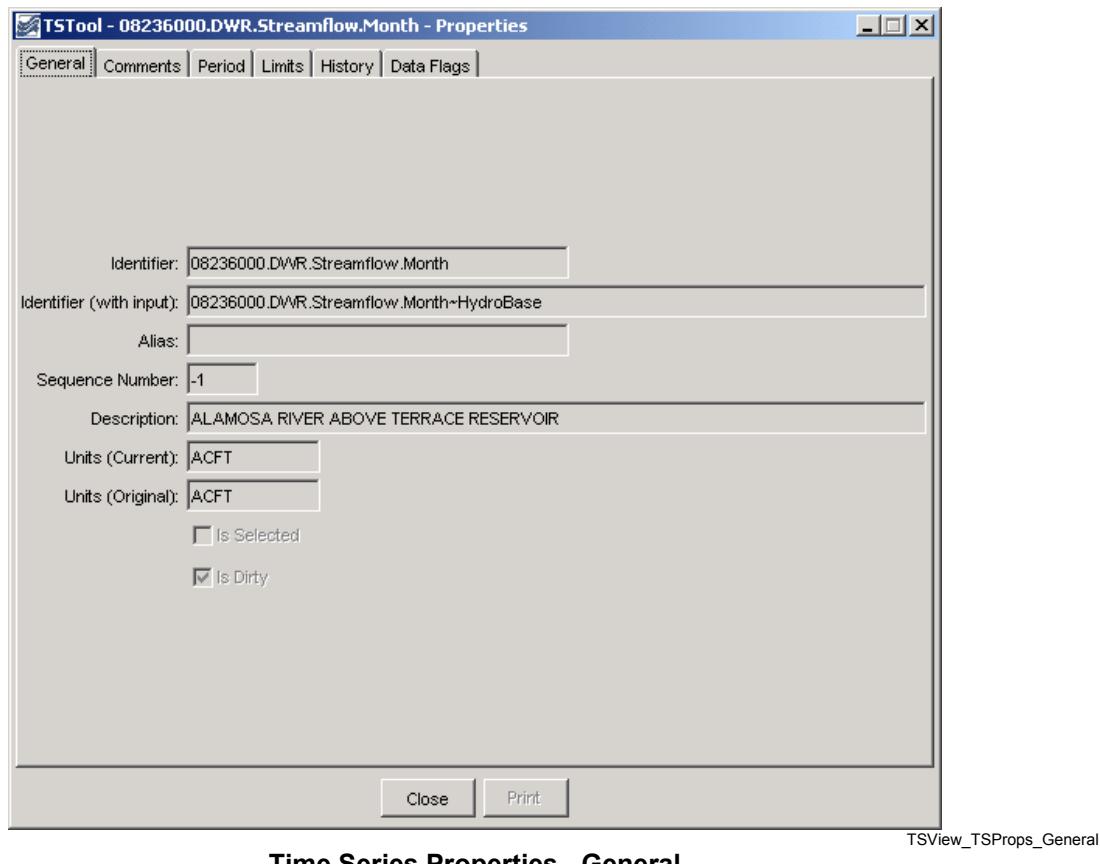
The following table summarizes important time series terminology.

Time Series Terminology (listed alphabetically)

Term	Description
<i>Data Interval</i>	Time interval between time series data values. If a <i>regular</i> time series, the interval is constant. If an <i>irregular</i> time series, the interval can vary. Intervals are represented as an optional multiplier followed by a base interval string (e.g., 1MONTH, 24HOUR) or IRREGULAR for irregular time series.
<i>Data Source</i>	A string abbreviation for a data source, which is part of the time series identifier and typically indicates the origin of the data (e.g., an agency abbreviation, or a model name if the result of a simulation).
<i>Data Type</i>	A string abbreviation for a data type, which is part of the time series identifier (e.g., Streamflow).
<i>Date/Time Precision</i>	Date/time objects used with time series have a precision that corresponds to the time series data interval. The precision is typically handled transparently but it is important that the precision is consistent (e.g., monthly data should not use date/time objects with daily precision). Displaying time series with various precision usually results in the smallest time unit being used for labels.
<i>Input Name</i>	A string input name corresponding to an input type, which is part of a time series identifier. For database input types, the name may be omitted or may be the name of the database connection (e.g., ARCHIVE). For input files, the name is typically the name of the file.
<i>Input Type</i>	A string abbreviation that indicates the input type (persistent format) for a time series, and is part of a time series identifier. This is often the name of a database (e.g., HydroBase, RiversideDB) or a standard data file format type (e.g., StateMod, MODSIM, RiverWare).
<i>Location</i>	A string identifier that is part of a time series identifier and typically identifies a time series as being associated with a location (e.g., a stream gage or sensor identifier). The location may be used with certain input types to determine additional information (e.g., station characteristics may be requested from a database table using the location).
<i>Scenario</i>	A string label that is part of a time series identifier, and serves as a modifier for the identifier (e.g., HIST for historical).
<i>Sequence Number</i>	A number indicating the sequence position of a time series in a series. For example, possible time series traces may be identified with a sequence number matching the historical year for the data. The use of sequence numbers with traces is being evaluated.
<i>Time Series Product</i>	A graph or report that can be defined and reproduced. See the Time Series Product Reference section.
<i>Time Step</i>	See Data Interval.

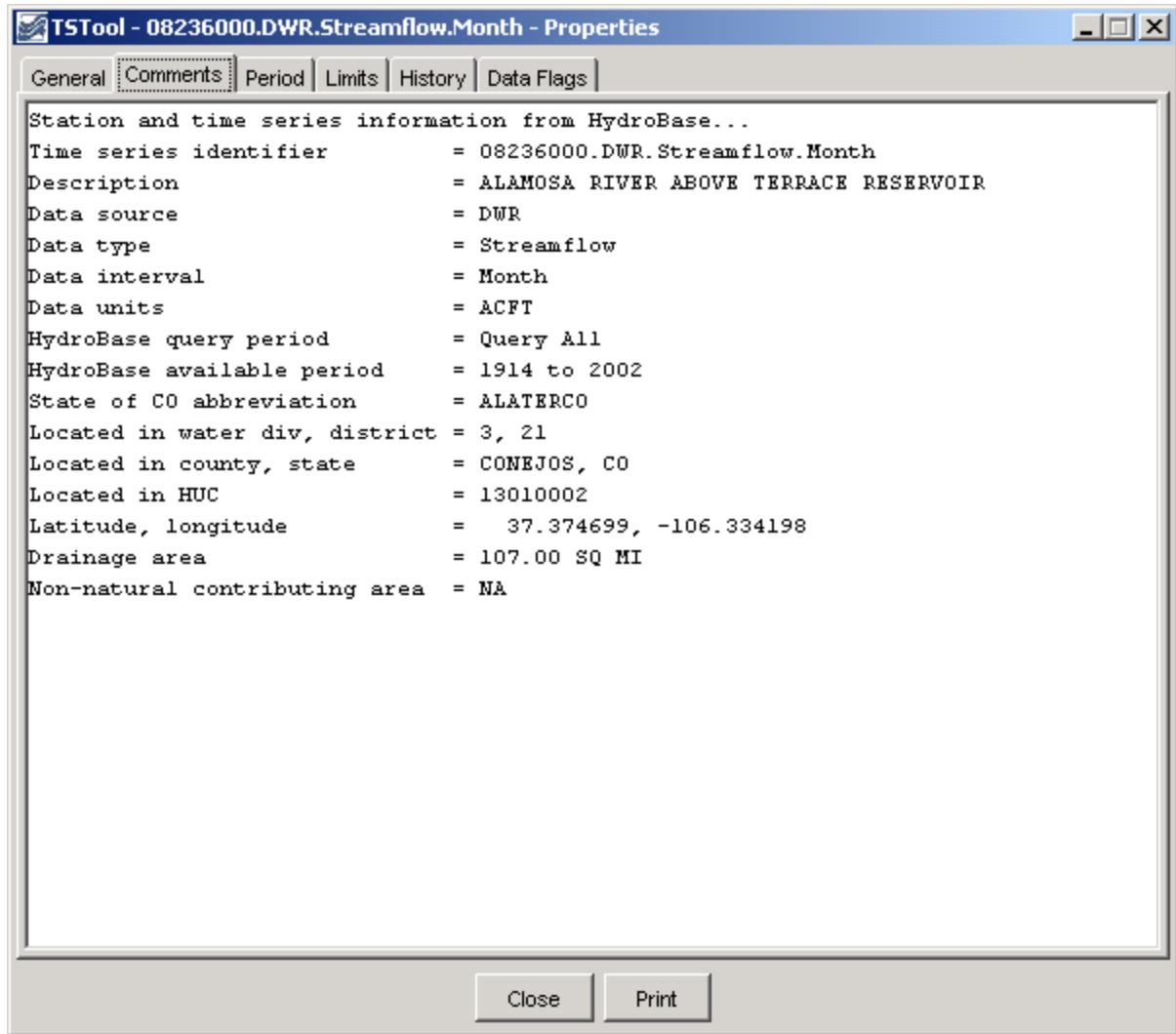
Time Series Properties Interface

Time series properties are displayed in a tabbed panel as appropriate in applications (e.g., the TSTool application can display the properties after time series are read and listed in the TSTool interface). Differences between time series input types may result in variations in the properties (e.g., some input types do not have descriptions for time series). The following figures describe the properties tabs. The size of each tabbed panel is set to the size of the largest tab; therefore, some tabbed panels are not completely filled.

Time Series Properties - General**Time Series Properties - General**

General time series properties are as follows:

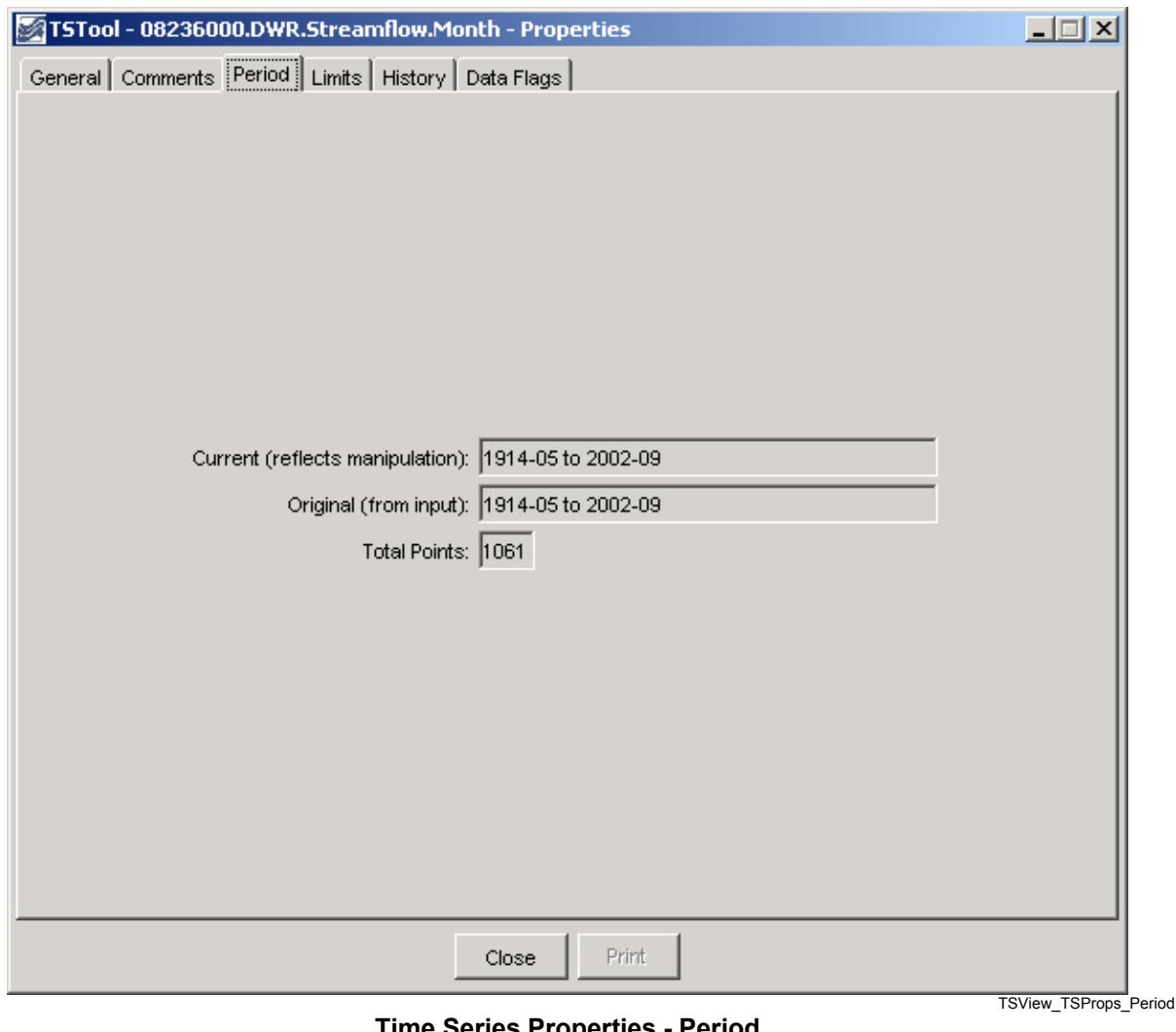
Identifier	The five-part time series identifier without the input type and name. This identifier is often used internally in applications to manage time series. See the Time Series Terminology section for a complete explanation of time series identifiers.
Identifier (with input)	The full identifier, including the input type and name (if available). The input type and name indicate the format and storage of the data.
Alias	A time series may be assigned an alias to facilitate processing (e.g., the alias is used by the TSTool application in time series commands).
Sequence Number	If the time series is part of a series of traces, the sequence number is used to identify the trace. Often it is the year for the start of the trace.
Description	The description is a mid-length phrase (i.e., longer than the location but shorter than comments) describing the time series (e.g., XYZ RIVER AT ABC).
Units (Current)	The units that are currently used for data. The units may have been converted from the original.
Units (Original)	The units in the original data source.

Time Series Properties – Comments**Time Series Properties - Comments**

TSView_TSProps_Comments

Comments for time series can be created a number of ways and may be formatted specifically for an application. Common ways of creating comments are:

- read comments from the original data source - this is ideal; however, electronic comments are often not available (e.g., the USGS previously published comments for data stations in hard copy water reports; however, comments may no longer available electronically),
- format comments from existing data pieces (e.g., the figure illustrates a standard set of comments for State of Colorado data, using the HydroBase input type).

Time Series Properties – Period

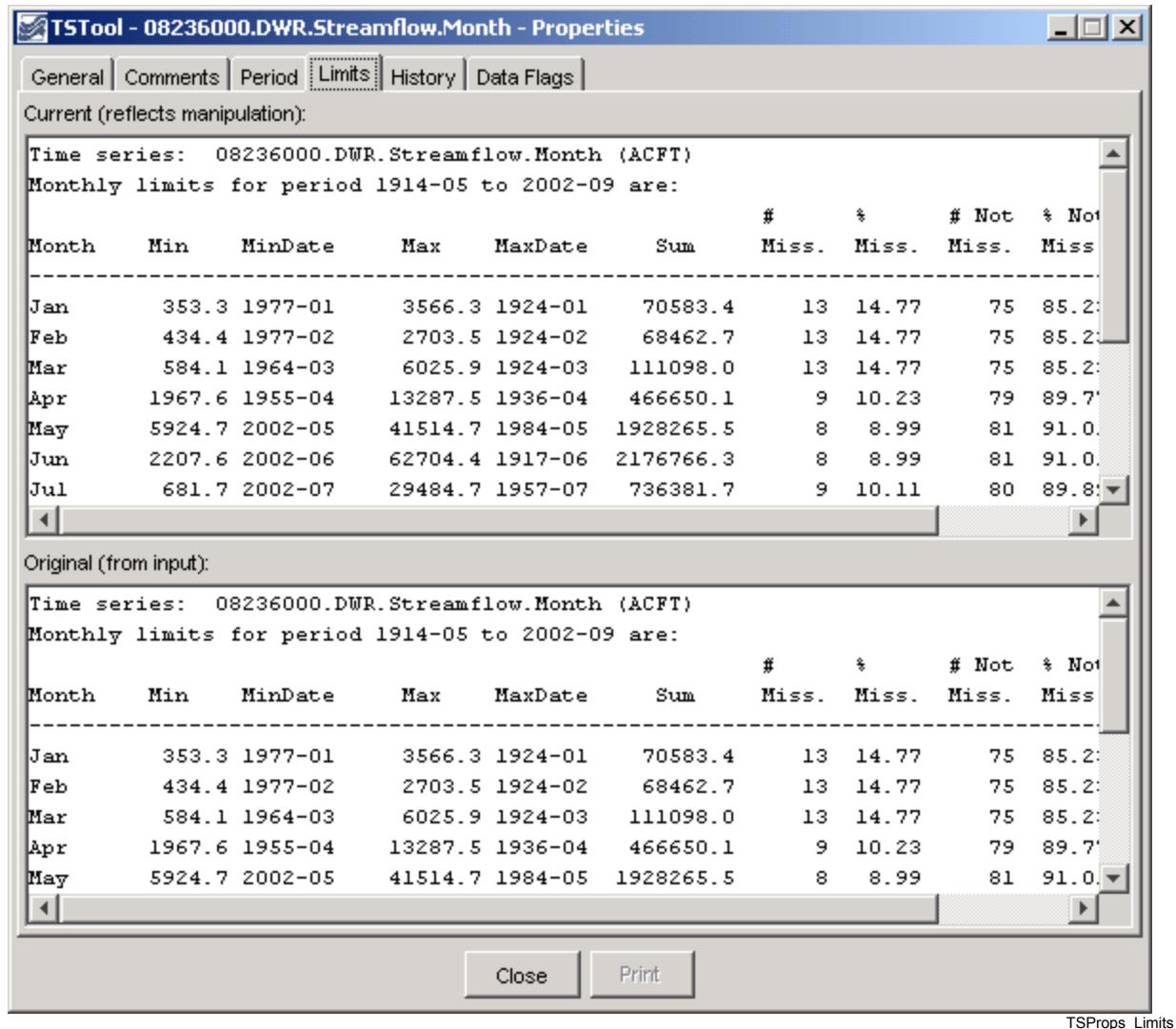
TSView_TSProps_Period

Time Series Properties - Period

Properties related to the period are as follows:

- | | |
|--|--|
| Current
(reflects
manipulation) | The current period is used to allocate computer memory for the time series data. This period may be set by an application (e.g., when creating model input files a specific period may be used). The precision of the date/time objects should generally be consistent with the time series data interval. |
| Original (from
input) | The original period can be used to indicate the full period available from a database. Setting the original period can sometimes be complicated by how missing data are handled (e.g., a database or file may indicate a certain period but a much shorter period is actually available). |
| Total Points | Total number of points in a time series. If a regular time series, this can be computed from the period. If an irregular time series, the number of points is determined from a count of all data values. The data points may include missing data – see the data limits for additional information. |

Time Series Properties – Limits

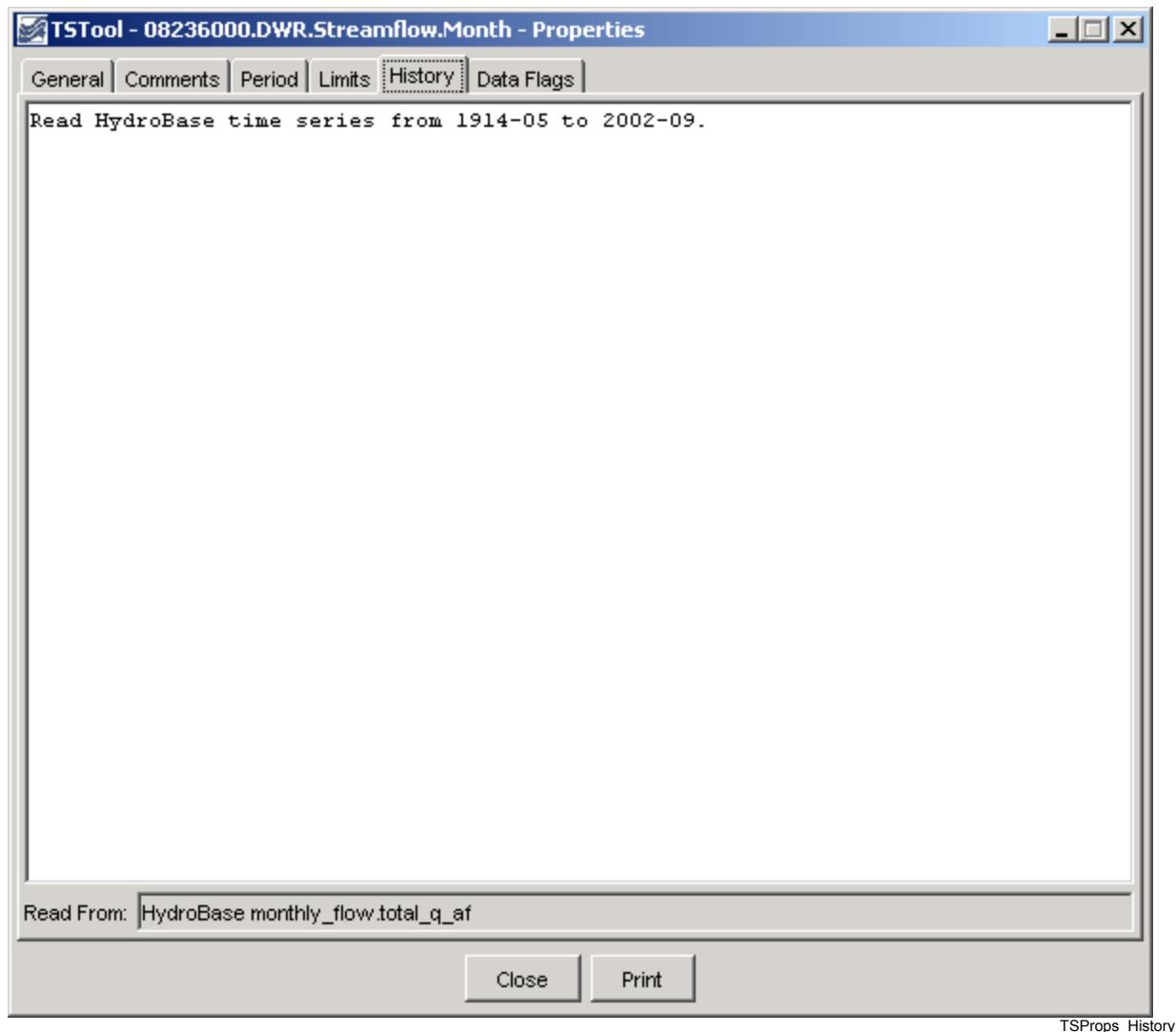


Time Series Properties - Limits

Time series limits are determined for both the current data (top in figure) and the original data (bottom in figure). This is useful because the original data may contain missing data, which are later filled. The data limits are displayed consistent with the data interval. In the example shown, limits are computed for each month. For other time series having other intervals, only overall data limits may be computed.

Theoretically, it is possible that a daily time series could have day limits (e.g., max/min values for each day of the year), month limits (e.g., computed as an average of the daily values by month), and year limits (e.g., computed as an average of all daily values in a year). However, automatically including this level of detail decreases performance and it is difficult to automatically make the right decisions (e.g., about whether to average or total values). Consequently, the limits are currently computed in a basic fashion on the raw data (no interval changes).

Time Series Properties – History

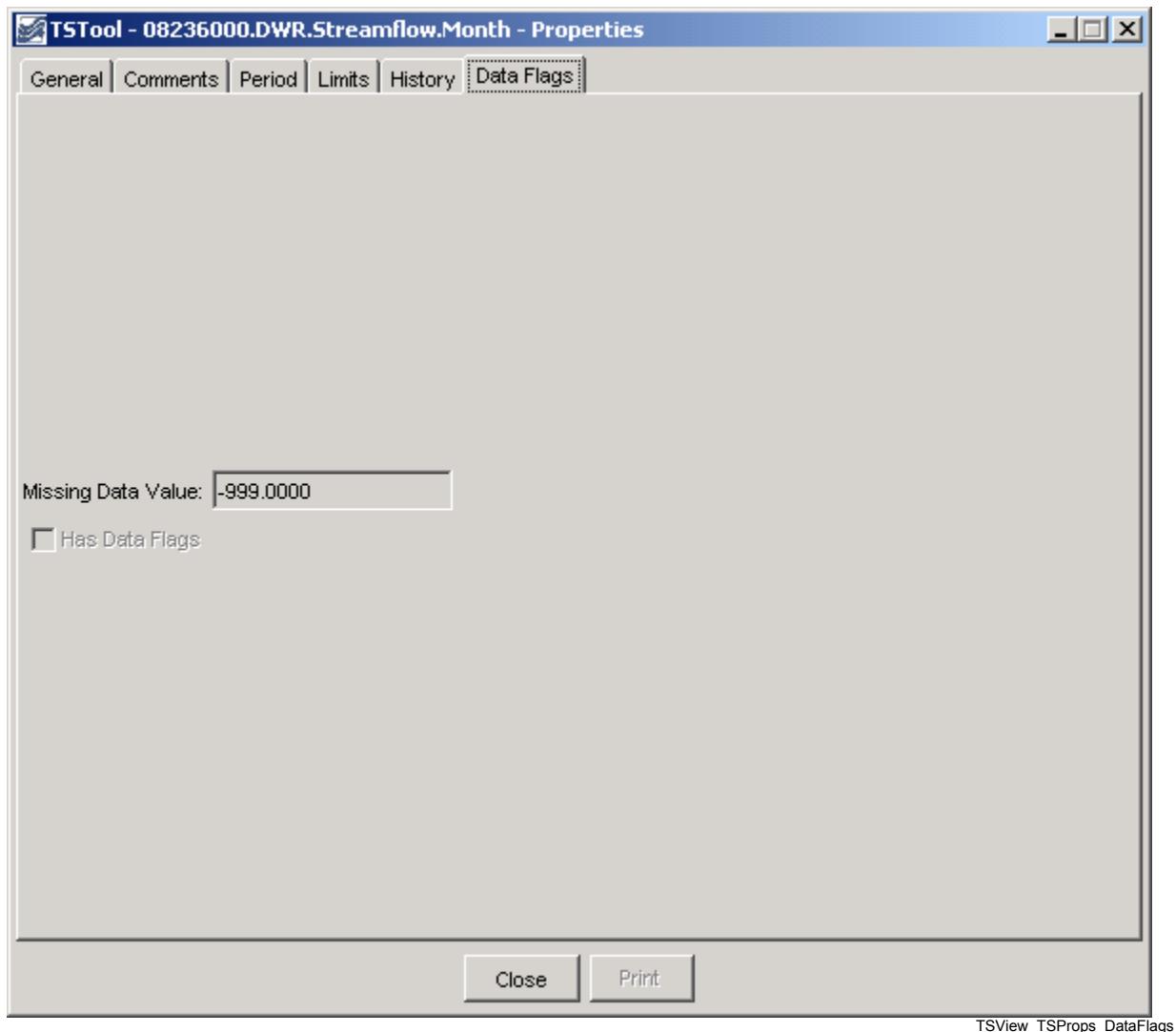


Time Series Properties - History

The time series history (sometimes called the *genesis history*) is a list of comments indicating how the time series has been processed. The completeness of this history is totally dependent on the time series input/output and manipulation software. Although efforts have been made to add appropriate comments as time series are processed, enhancements to the history comments are always being considered.

At the bottom of the history list (see **Read From**) is the input name that was actually used to read the data. This input name may or may not be exactly the same as the input name in the time series identifier. For example, if reading from a HydroBase database, the time series identifier may specify an input type of HydroBase and no input name (because the software knows from the other parts of the time series identifier which database tables to read). However, it is also useful to know the actual table that is read in order to help users and developers understand the data flow. If reading from a file input type, the **Read From** information will show the full path to the file; however, the input name in the time series identifier may only include a relative path.

Time Series Properties – Data Flags



TSView_TSProps_DataFlags

Time series data flags contain information that describe the quality of a data point. The missing data value indicates a special number that is used to indicate that a data value is missing at a point. Currently only floating point values are recognized; however the NaN (not a number) value is generally supported for input types that use the convention. All time series are typically assigned a missing data value.

The **Has Data Flags** checkbox indicates whether the time series has data flags. Full support for data flags is being phased in, based on whether an input type supports data flags. The USGS NWIS file format is an example of an input type that supports data flags (e.g., e is used to indicate estimated data).

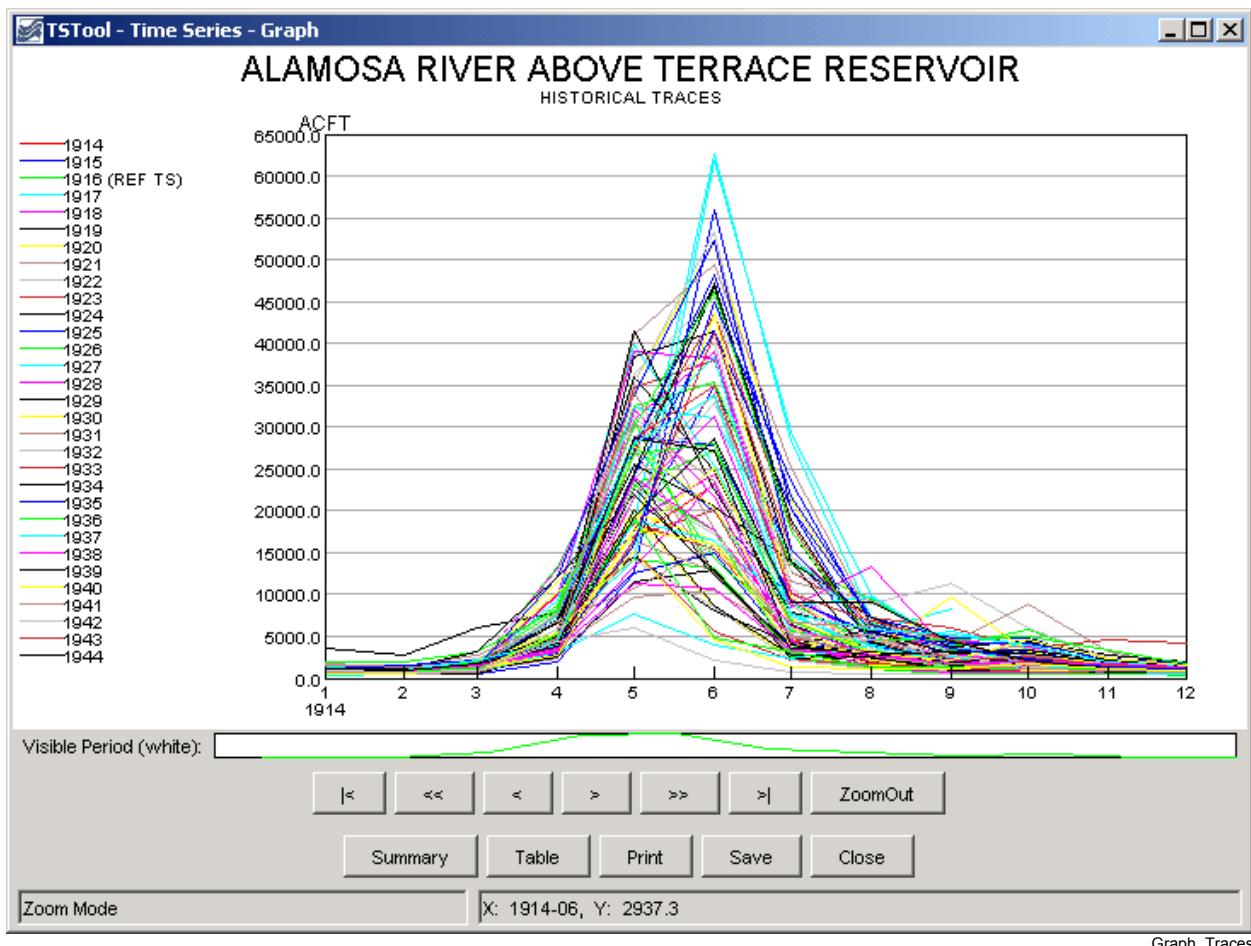
One of the issues with fully supporting data flags is that different input types (and even different data within an input type) treat data flags inconsistently. Therefore, it is easier to add data flags to time series visualization tools (e.g., label points on a graph with the flag) than to integrate data flags in data filling and analysis features. Features related to data flags will continue to be enhanced.

Time Series Traces

The term *time series traces* in general refers to a group of time series, often shown in overlapping fashion. Common uses of time series traces are:

- separate a full time series into annual traces and plot them on top of each other, shifted so that they all start at the same date/time,
- run a model or analytical tool multiple times, with input being a series of input traces, and generating a series of output traces, in order to produce probabilistic simulations.

The power of using traces is that a large amount of data can be used to visualize and study statistical qualities of the data, as shown in the following figure.



Example Graph for Time Series Traces

The TSView package supports time series traces at various levels. Time series properties include a sequence number that can be used to identify a time series as being in a group of traces. However, for data management and viewing, time series identifiers often do not indicate whether a time series is in a group of traces (the sequence number is managed internally). Full support of time series traces is being phased in.

Currently, applications like TSTool include features to create time series traces and TSView tools can be used to view the time series as if they were separate time series. Additional visualization features are being enabled as time allows.

The following sections describe the different time series views that are available in TSView. Although most illustrations using simple time series, most features are also available for use with traces.

Time Series Views

The main components of the TSView package are configured to provide multiple views for time series data. The three main views that are available are:

1. **Graph** - line, bar, or other graph
2. **Summary** - text report suitable for the data type and interval
3. **Table** - spreadsheet-like table with scrolling, suitable for export to other tools

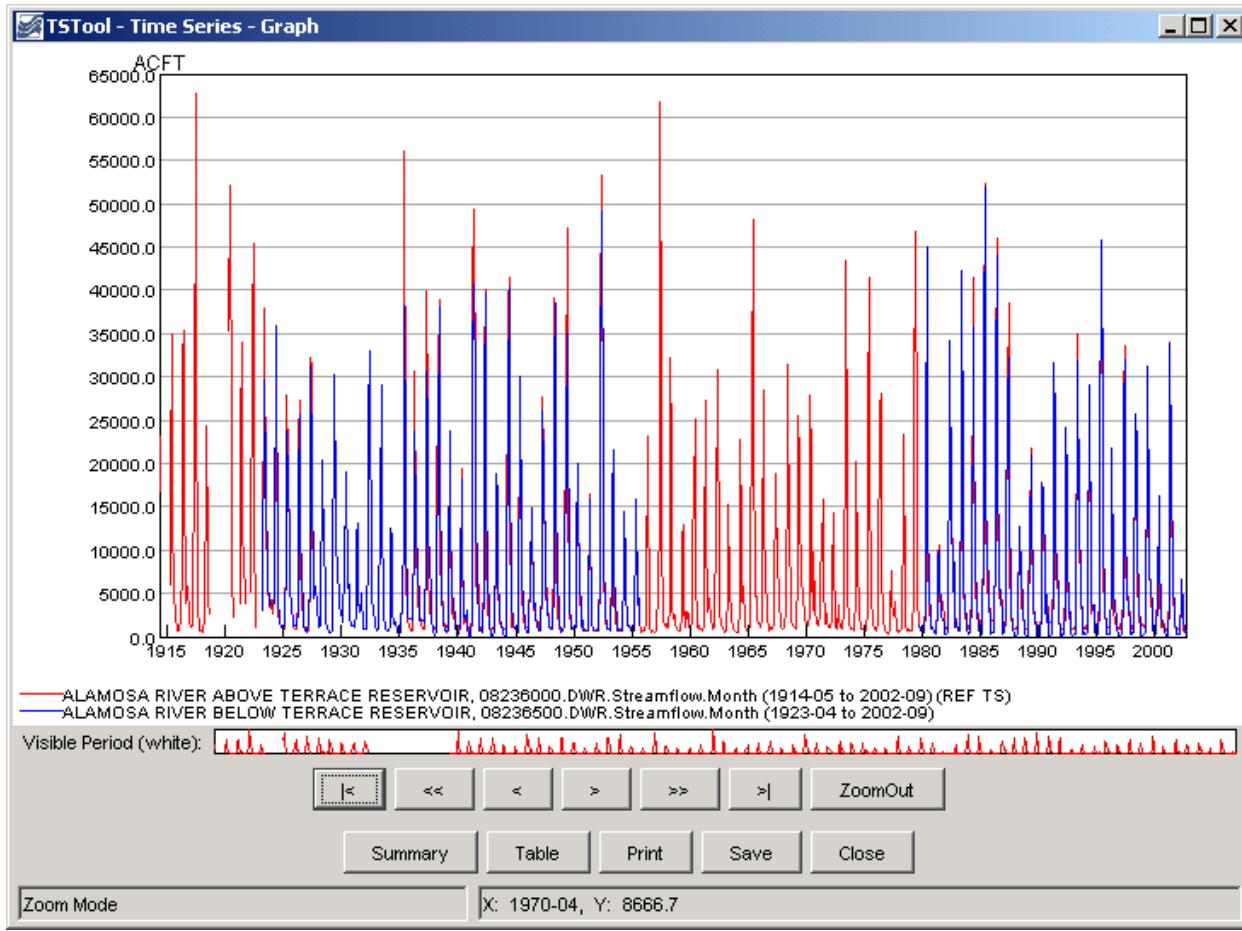
The initial view for a time series list is typically determined from the actions of the software user. For example, a **Graph** button may be displayed on a screen, which when pressed will result in a graph being displayed. The time series that are displayed in the view can typically contain one or more time series (some graph types may have a restriction on the number of graph types). To increase performance and capacity, the TSView package as much as possible uses a single copy of the time series data for visualization. For example, to generate graphs, the data for the time series objects are used directly rather than being copied into a graphing tool's data space. This also allows TSView to more easily display different data intervals on the same view because the data do not need to be forced into a consistent grid data structure.

The following sections describe the three time series views. The graph view type requires more extensive explanation due to the variety of graph properties.

Time Series Graph View

The graph view for time series supports a variety of graph types. The features of the various graph types will be discussed in detail in the following sections, starting with basic graph types, followed by more specific types.

Typically, the graph type is selected in the application (e.g., menus are available in TSTool for selecting the graph type for a list of time series). In many applications, the graph type often defaults to a line graph. The following figure illustrates a line graph for two monthly streamflow time series.



Example Line Graph for Monthly Streamflow

The graph view is divided into the following main areas:

Graph Canvas

The graph canvas is the area where the graph and legend are drawn. This area is used to interact with the graph (e.g., zoom). More than one graph can be drawn in the canvas (see the **Time Series Product Reference** section for additional details). If zooming is supported for the graph, a box can be drawn with the mouse to zoom in to a shorter period. Right-click over a graph of interest to show the popup menu for graph properties and analysis details (e.g., regression results). The canvas area is essentially a preview of a printed graph.

Reference Graph The reference graph below the main graph canvas shows the current view extent (the white area in reference graph in the figure above). The reference graph is only shown for graph types that support zooming. If shown, it can be used for zooming, similar to the main graph. The time series with the longest period of record is drawn in the reference graph to illustrate variations in the data over time (this time series is noted in the main graph legend with REF TS – this label is not shown in printed output).

Scroll/Zoom Buttons Under the graph areas is a layer of buttons used for zooming. The **Zoom Out** button will zoom to the full extent of the data.

The other buttons facilitate scrolling through data as described below. For all scrolling operations, the visible graph extent (or *page*) is maintained during the scroll. Scrolling can use the buttons or keys described below. To use the keyboard, first click in the main graph canvas to shift focus to that area.

<	Home	Scroll the visible window to the start of the period.
<<	Page Down	Scroll the visible window one full page to the left (earlier in time).
<	Left Arrow	Scroll the visible window 1/2 page to the left.
>	Right Arrow	Scroll the visible window 1/2 page to the right (later in time).
>>	Page Up	Scroll the visible window a full page to the right.
>	End	Scroll the visible window to the end of the period.

Main Buttons The bottom row of buttons provides features for displaying other views, printing, and exporting:

Summary	Display the summary view for the time series (see the Time Series Summary View section).
Table	Display the table view for the time series (see the Time Series Table View section).
Print	Print the graph. Because the physical extents of the printed page are different from the visible window, the printed graph may not exactly match the viewed version (e.g., more or less axis labels may be used).

Save	Save the graph as a Portable Network Graphic (PNG) or JPEG graphic, a DateValue file (a useful time series format), or a <i>Time Series Product</i> file (see the Time Series Product Reference section) by selecting from the choices. Depending on the main application, saving to a database as a time series product may also be enabled.
Close	Close the graph window. If related summary or table windows are still visible, the graph view can be quickly re-displayed by pressing the Graph button on the other view windows. If the graph properties have been changed but have not been saved, a warning will be displayed.
Status Message Area	The lower-left status message area is used to provide general user instructions and feedback.
Mouse Tracker Area	The lower-right status message area is used to indicate the position of the mouse on a graph, in data units. The coordinates are typically shown using an appropriate precision as determined from the time series date/time precision and data units.

Within each graph canvas it is possible to draw more than one graph, each with its own titles, legend, etc. The **Time Series Product Properties** section (below) provides an example and discusses how to edit graph properties. The **Time Series Product Reference** section describes in detail the format of *Time Series Product* files. These files, when saved from the graph view, can be used to recreate a graph interactively or in batch mode, at a later time.

Because TSView is a general tool, a number of rules are in place when viewing time series in graphs (see the **Time Series Product Properties** section for information on changing specific graph properties to override the defaults):

1. Time series plotted on the same graph should generally have the same units or have units that can be readily converted. If the units are not consistent, you will be warned and the units will be displayed in the legend rather than on the axis. (A future enhancement may allow multiple axes, each with different units.)
2. Time series can have different data intervals (e.g., daily data can be plotted with monthly data). However, other output options, such as reports, may not allow the same flexibility. It is important to understand the data type characteristics. For example, some data are instantaneous (e.g., real-time streamflow) whereas other data are accumulated (e.g., precipitation) or mean (e.g., mean temperature). Therefore, the representation of the data may need to be selected with care to ensure consistency. For example, some data intervals and types may be better represented as bars and others as lines or points.
3. Data values are plotted at exactly the point that they are recorded. The plot positions are determined using the year as the whole number and months, days, etc. to determine the fractional part of the plot position. The end-user does not typically see these computed positions because labeling uses data units, including dates. The plot positions are determined from the dates associated with data and no

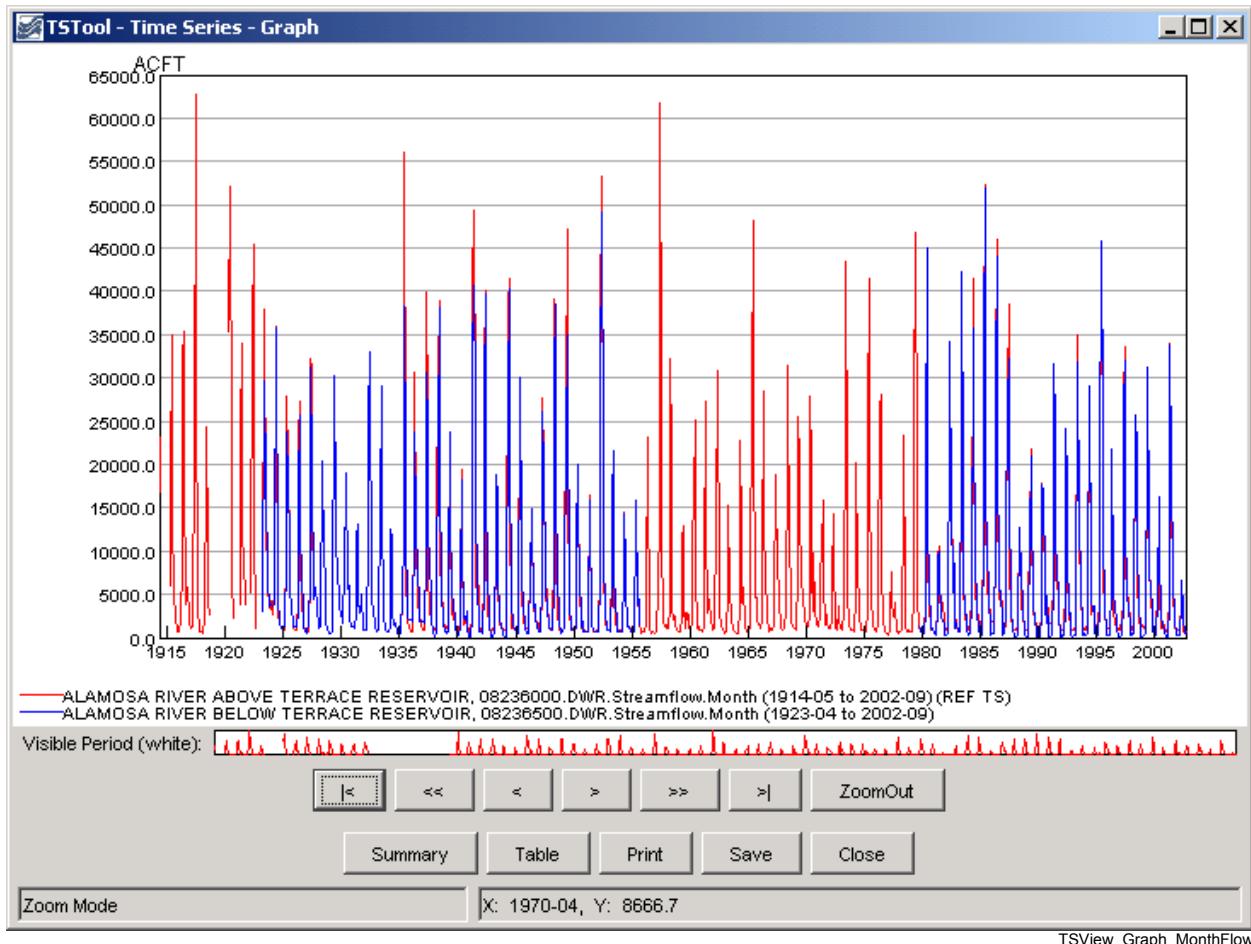
adjustments are made to plot in the middle an interval. For example, monthly data are plotted at the first day of the month (not day 15). Properties to override this convention are being evaluated. Bar graphs allow you to select whether the bars are drawn to the left or right of the date, or centered on the date. This allows flexibility to show a period over which a value was recorded, if appropriate.

4. The mouse coordinates that are displayed by the mouse tracker are computed by interpolating screen pixels back to data coordinates (which involves a conversion of the plot position to date/time notation). Consequently, the values shown may be rounded off (depending on the zoom extent and data precision). The mouse coordinates are displayed based on the precision of the time series data. When moved, the mouse will display the same date until the date changes within the given precision. For example, for monthly data, moving the mouse left to right, the mouse coordinate will display as 1999-01 as soon as the date changes from 1998-12 to 1999-01. The label will remain at 1999-01 until 1999-02 is encountered. Because data values are drawn at points, you should therefore always position the mouse slightly to the right of the point to see the date corresponding to the value. This is very important for bar graphs because the bar may extend over several dates. If specific values need to be determined, use the summary or table views.
5. Labels for axes are determined automatically in most cases based on the font requirements, available display space, and data range. Major and minor tic marks are drawn to help determine the data coordinates. Labels are redrawn as the visible period is changed.
6. Graphs that can be zoomed do not allow the vertical axis to be re-scaled on the fly. This capability is being evaluated.
7. Currently, graph types cannot be mixed for time series on a graph. In other words, a graph cannot contain a bar graph for one time series and a line graph for another time series. This ability may be added in the future. A work-around is to use multiple graphs on a page (see the **Time Series Product Properties** section for an example).
8. The precision used to format graph labels is determined from data unit information provided by the application. This generally produces acceptable graphs. However, in some cases, the range of values being plotted results in inappropriate labels where label values are truncated and/or repeated.
9. Graph types can be changed after the initial display, with limitations. Graphs can be switched between simple types like line and bar graphs; however, simple graphs cannot be changed to more complex types.

The following sections describe various graph types supported by the TSView package. Graph properties are mentioned in some sections. The discussion of how to change graph properties is included in the **Time Series Product Properties** section after the graph type descriptions.

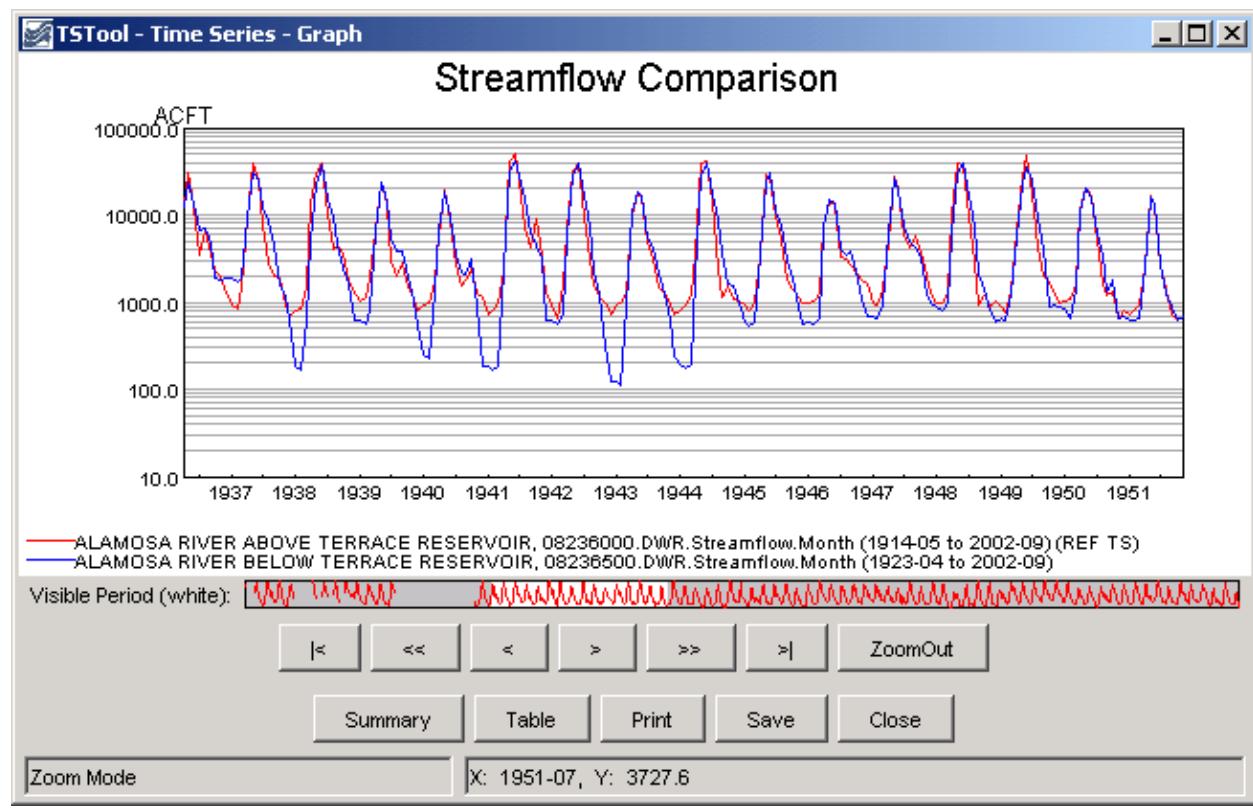
Line Graph

Line graph features have been illustrated in previous discussion. The line graph type is also used to generate graphs with only points by setting the line style to None (for example, software that displays daily data where gaps are expected may default to using symbols and no line).



Line Graph - Log Y Axis

Log-axis line graphs are similar to simple line graphs. The following figure illustrates a typical graph.



Example Log Y Axis Graph showing Monthly Streamflow

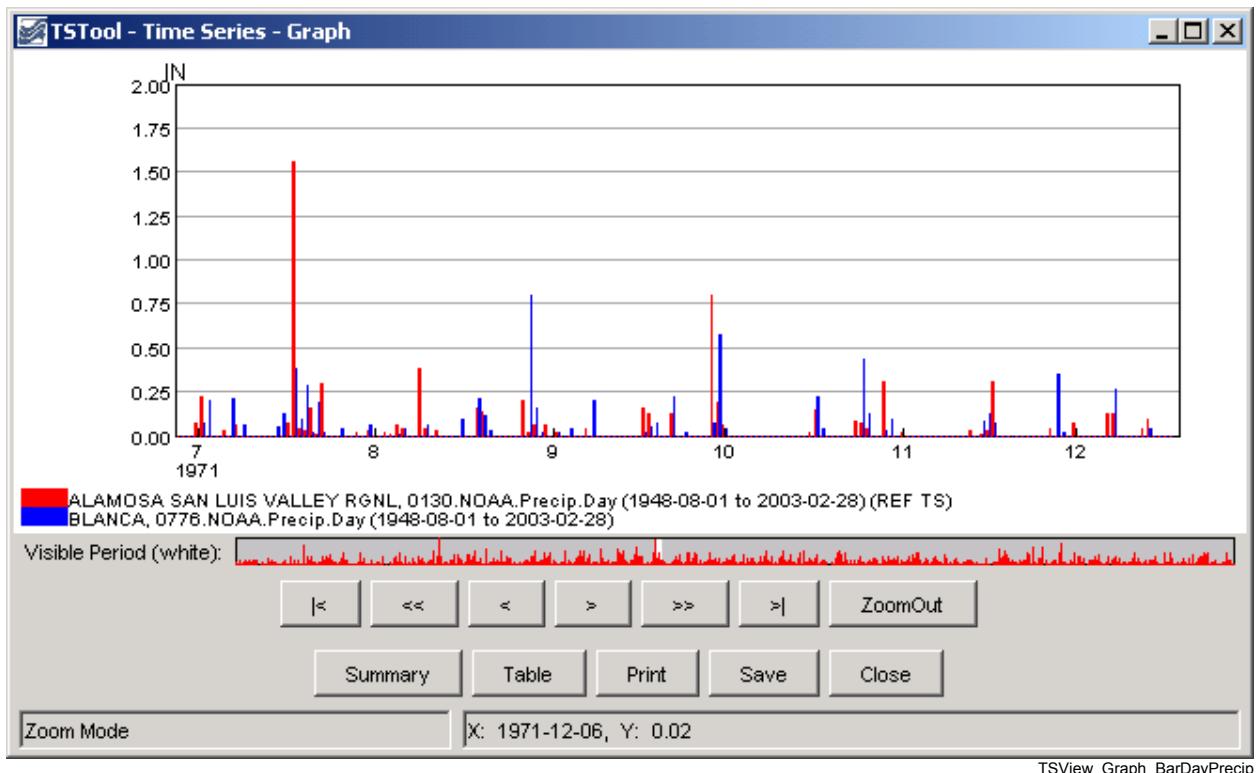
TSView_Graph_LogMonthFlow

Characteristics of the log plot are:

- If the minimum data value is ≤ 0.0 , then .001 is used for the minimum plotting value.

Bar Graph

The bar graph type produces a graph with parallel vertical bars, as shown in the following example:



Example Bar Graph showing Daily Precipitation

The above example illustrates that at the given zoom extent (which is a small part of the full period - see the white area in the reference graph), labels are drawn for months. Zooming in more would display the day in the labels. The mouse tracker in all cases shows days since that is the precision of the data. Characteristics of the bar graph are as follows:

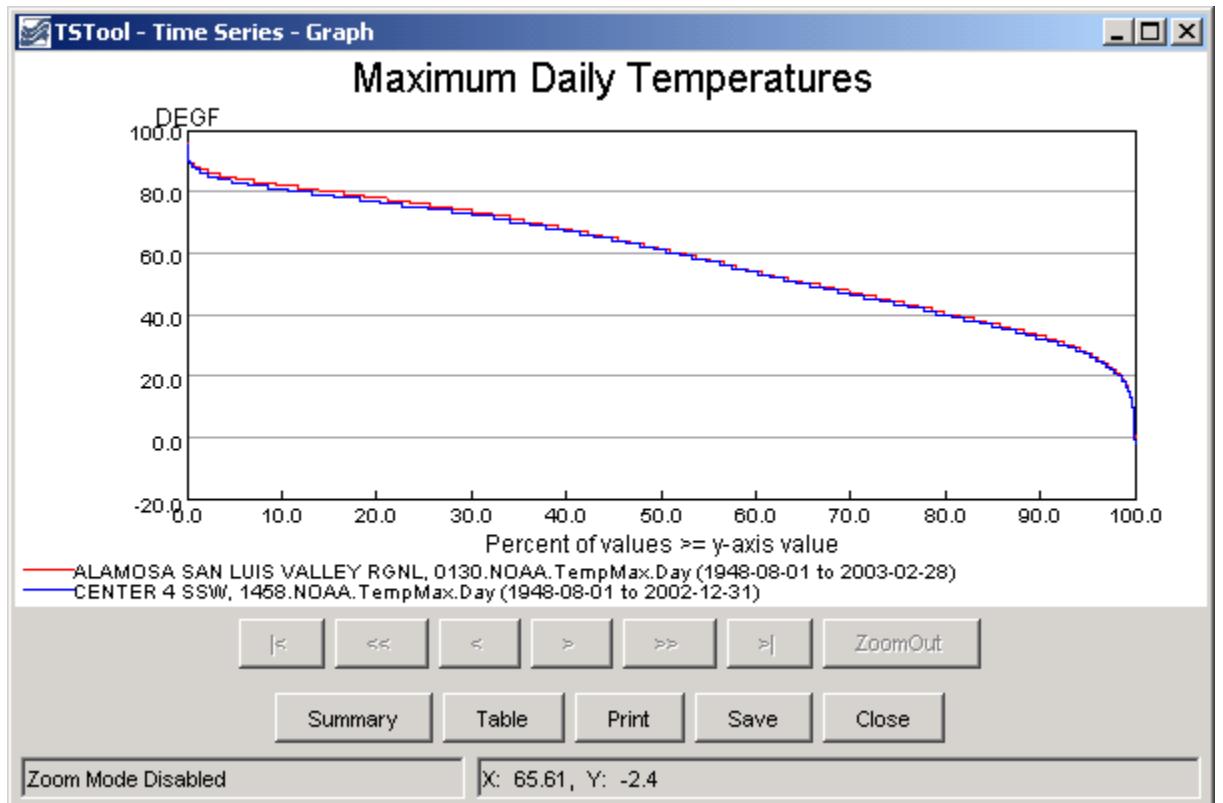
- Bars can be plotted centered on, to the left of, or to the right of the dates. If multiple time series are plotted, the overall total width of the bars will correspond to one data interval. If drawn to the left of the date, the bars for all graphed time series are drawn to the left of the date. If drawn to the right of the date, the bars for all graphed time series are drawn to the right of the date. If centered on the date, half the bars are drawn to the left of the date, and half to the right
- Bar widths are determined based on the number of time series being plotted. Monthly time series use a slightly narrower bar (larger gap between bars) because the number of days in a month varies. To make bars stand out better, a white line may be drawn to separate adjacent bars. If bars are very narrow the line is not drawn. Bars will always be drawn at least one pixel wide, even if this obscures neighboring bars (zoom in to see more detail). Round-off in drawing bars may result in some bars being slightly wider or narrower than other bars.
- Bars always end at the zero value on the Y axis. In other words, bars extend up or down from zero.
- The mouse cursor display dates relative to the axis and does not determine the data value relative to edges of the bars. For example, if bars are plotted centered on dates, 1/2 of the bar will actually be in the previous date, according to the mouse tracker.

Double Mass Curve

Double mass curves are currently disabled. An alternative is to use the TSTool application and generate cumulative time series, which can be viewed in a line graph.

Duration Graph

A duration graph indicates the range of values in a time series and how often they occur, as shown in the following example:



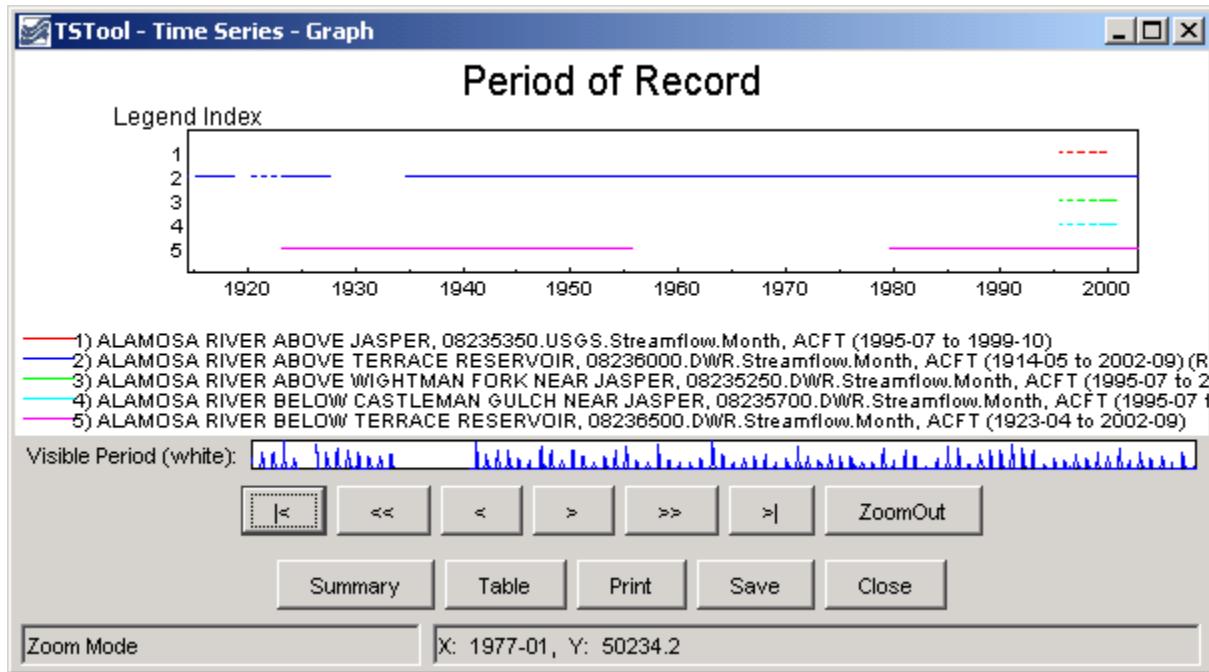
Example Duration Graph showing Maximum Monthly Temperatures

The algorithm for calculating and graphing a duration curve was taken from the book **Handbook of Applied Hydrology** (edited by Ven Te Chow): “*When the values of a hydrologic event are arranged in the order of their descending magnitude, the percent of time for each magnitude to be equaled or exceeded can be computed. A plotting of the magnitudes as ordinates against the corresponding percents of time as abscissas results in a so-called duration curve. If the magnitude to be plotted is the discharge of a stream, the duration curve is known as a flow-duration curve.*” Features of duration graphs are as follows:

- The zoom feature is disabled for this graph type.
- Although duration curves have traditionally been applied to streamflow or reservoir data, duration graphs can be created for any time series data.
- Noticeable breaks in the curve are caused by a limited number of data points and/or values that are measured as rounded values.

Period of Record Graph

The period of record graph is used to display the availability of data over a period, as shown in the following figure:



Example Period of Record Graph showing Monthly Streamflow

Characteristics of the period of record graph type are:

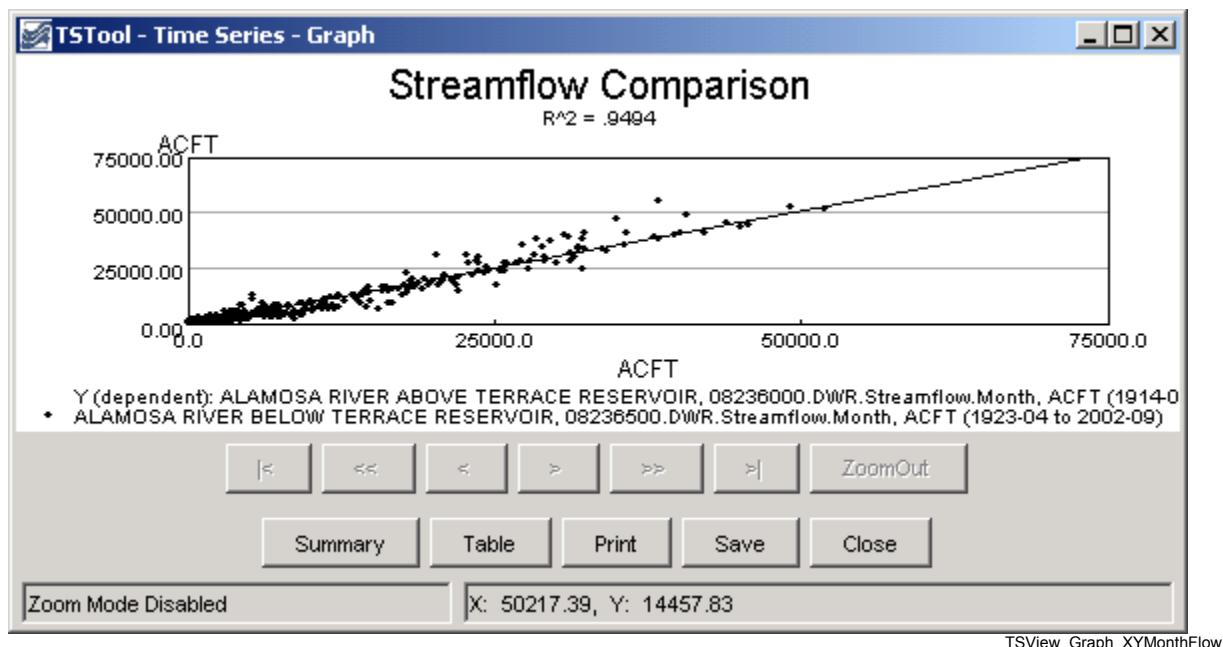
- Horizontal lines are drawn for each time series, with breaks in the line indicating missing data.
- Zooming is fully enabled, however, it may be difficult to see small breaks in the lines – it may be necessary to display symbols at the data points. The data limits properties of each time series can also be used to check for missing data. The TSTool application provides reporting features to summarize data coverage.
- Because data values are not plotted, the y-axis is labeled with a legend index number. This also allows the graph window to be compressed vertically, if desired.

XY-Scatter Graph

The XY-Scatter graph type can be used to compare data having the same data interval (units can be different). This graph type is often used for the following comparisons:

1. The dependent time series (Y) requires filling and multiple independent time series (X) are analyzed to find the best time series to use as the independent time series. One or more independent time series can be plotted on the same graph.
2. The dependent time series (Y) contains observed data and one or more independent simulated time series (X) are analyzed to determine which simulation is closest to actual observations.
3. The independent (X) and dependent (Y) time series are compared to determine whether any type of relationship exists between data points. In this case, a single dependent time series may be compared with multiple independent time series on the same graph.

Currently the XY-Scatter graph can have only a single dependent time series but can have one or more independent time series. The following figure shows a typical graph.

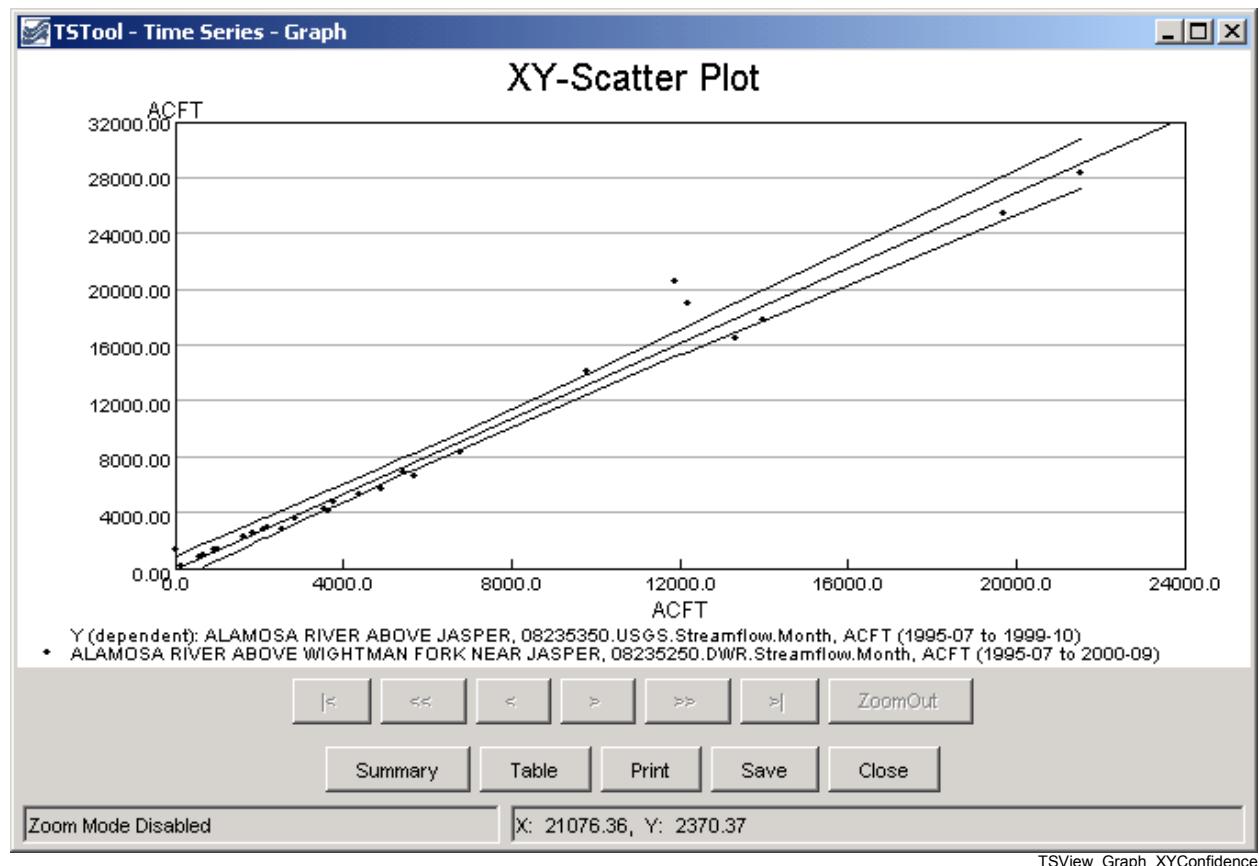


Example XY Scatter Graph showing Monthly Streamflow

Characteristics of the XY Scatter graph are:

- Labels and legend are automatically generated. See the **Time Series Product Properties** section below for information about changing the appearance of the graph.
- Simple linear regression is initially performed to determine a line of best fit. See the **Analysis** tab in the **Time Series Product Properties** section below for information about curve fit methods.
- A 45 degree line is currently **not displayed** because time series of different types and units may be compared. Graph properties do allow the line of best fit to be forced to zero. The limits on the axes are not automatically set to equal values for the same reason; however, a property to force the values to be the same will be added.
- Zooming is disabled.
- Two or more time series must be specified and must have the same interval.

- Confidence intervals can be turned on, as shown in the following figure:



TSView_Graph_XYConfidence

The confidence intervals provide a useful way for assessing the quality of a point estimate. When a regression line is of interest, the confidence interval on the line as a whole permits one to make confidence statements about a number of values of the predictor variables simultaneously. Confidence limits for the line are a function of the level of confidence (e.g., gamma = 95% or 99%), and the F-test statistic (2, n-2 degrees of freedom, and level of significance =1-gamma). The equations used to plot the confidence intervals are shown below (note that because the curves depend on the data points, the shape and smoothness of the curves will depend on the number of points; the points are sorted to generate a continuous line).

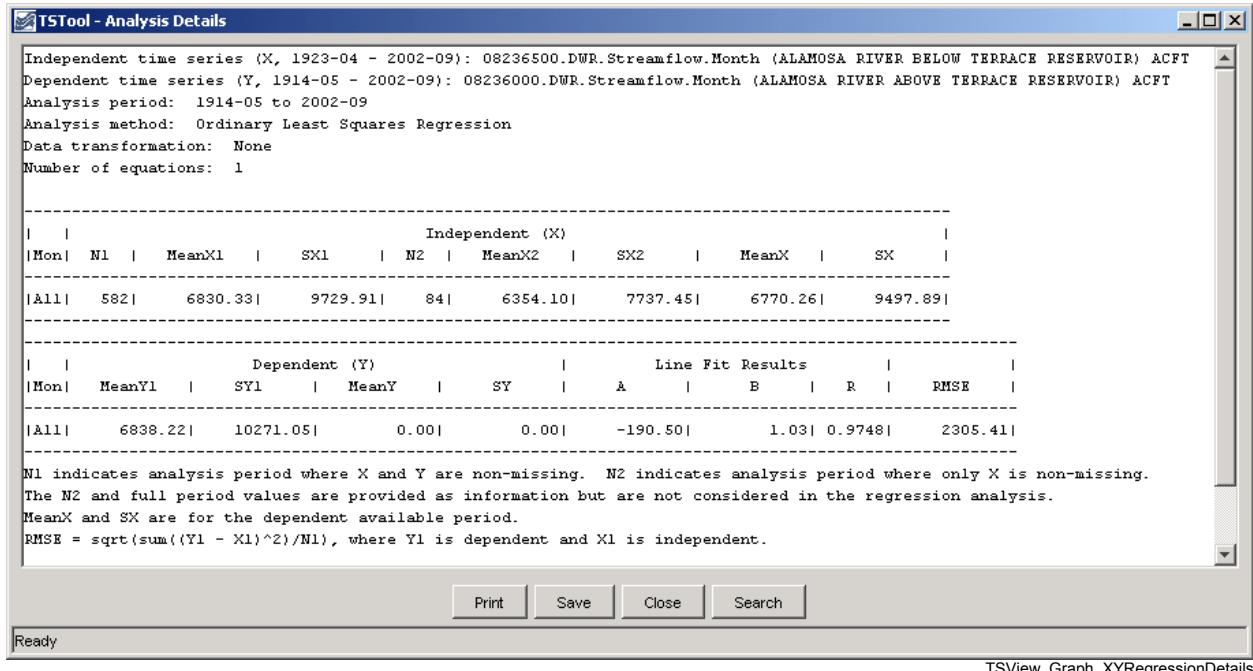
$$\hat{y}_{Cl_i} = [\bar{y} + B(x_i - \bar{x})] \pm \sqrt{2F} \left[\frac{1}{n-2} \sum_{j=1}^n (\hat{y}_j - y_j)^2 \right]^{1/2} \left[\frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^n (x_j - \bar{x})^2} \right]^{1/2}$$

where:

- \hat{y}_{Cl_i} = confidence interval y value at x_i
- \bar{y} = mean of y
- B = slope of regression line equation $y = A + Bx_i$
- x_i = x value where Y_{Cl_i} is being computed
- \bar{x} = mean of x

- F = F distribution at $(2, n-2)$ degrees of freedom and gamma significance
 n = number of points with x and y values
 \hat{y}_i = y predicted by the equation $y = A + Bx_i$
 y_i = y value of data point corresponding to x_i

- The best fit line can be turned off.
- Right-clicking on the graph displays the **Analysis Details** menu, that, if selected, displays curve fit information about the time series, as illustrated in the following figure:



Example Analysis Details

The RMS error (or *RMSE*) is calculated in the following way:

$$\begin{aligned} SSE &= \sum(X_i - Y_i)^2 = \text{Sum of Square Errors} \\ MSE &= SSE/N = \text{Mean of Sum of Square Errors} \\ RMSE &= \sqrt{MSE} = \text{Square Root of the MSE} \end{aligned}$$

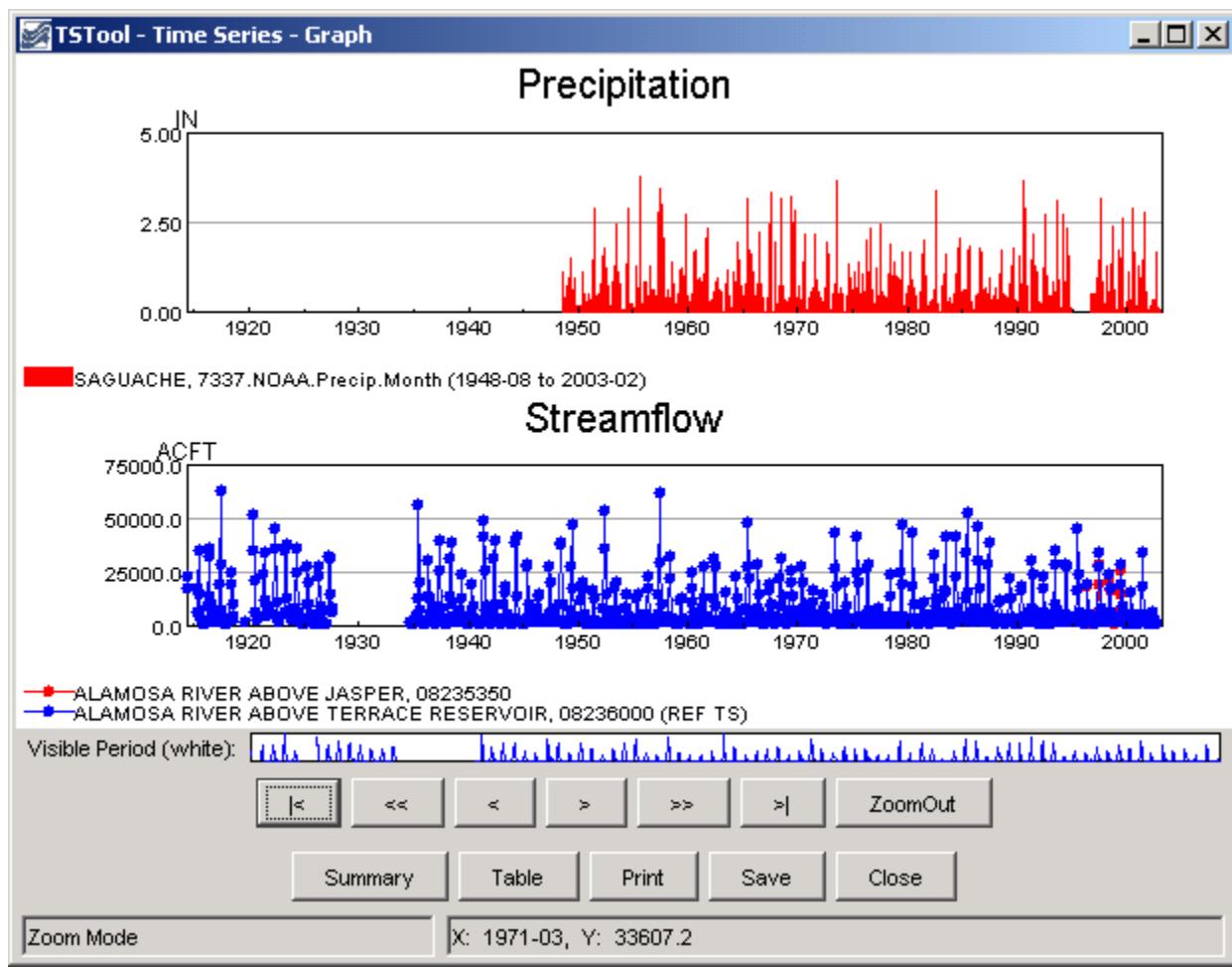
The *RMSE* can have different meanings, depending on how the data are being analyzed:

1. If a measured (X) and a simulated (Y) time series are being compared to determine, for example, to determine how well a model is simulating actual observations, then the *RMSE* indicates the error of a simulation when compared to the actual (comparing the values).
2. If two time series are evaluated to determine if the relationship between the time series can be used to estimate missing values in one of the time series, then the difference between estimated values (Y_{est}) and the line of best fit (e.g., $A + BX$) is used to compute the *RMSE*. For a perfect fit, the *RMSE* would be zero. Values of *RMSE* can be used to evaluate the estimator for data filling.

To provide as much information as possible for multiple uses, the **XY-Scatter Graph Analysis Details** provides both *RMSE* values. The default is to display a line of best fit, which is usually desirable information. The graph properties allow the analysis to be done for data filling, if desired.

Time Series Product Properties

A time series product is one or more time series graphs, tables, or reports on a “page”, although currently TSView focuses on graph products. Time series product properties can be displayed by right-clicking on a graph of interest and selecting the **Properties** menu item from the popup menu. Interactively changing properties allows graphs to be configured as desired. The following figure illustrates a time series product that has two graphs (see the **Time Series Product Reference** section for information about how to define time series product files, which can be used to save a product).



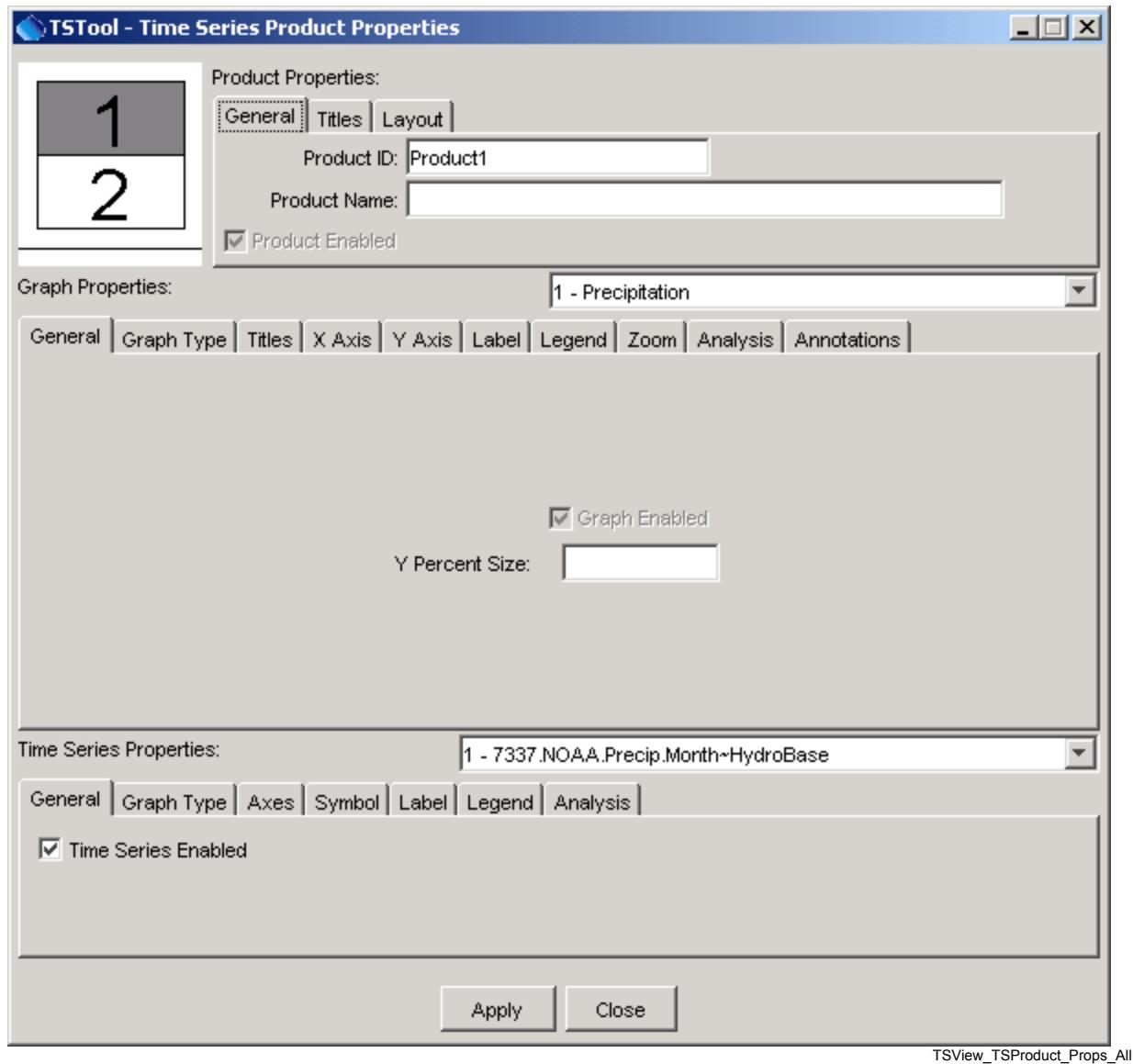
Example Graph Product showing Precipitation and Streamflow

In many cases, a graph product will consist of only a single graph (which may show one or more time series). However, it is also useful to display multi-graph products, especially when related data types are used. The TSView interface includes features to construct multi-graph products interactively, and the product files described in the **Time Series Product Reference** section can be created and processed. The TSTool application, for example, can interactively create or read a product file and display a graph similar to the one shown above. Important considerations for multi-graph products are:

- The product page has its own set of properties (e.g., titles and size).
- Each graph area has its own properties (e.g., titles, labels, graph type, legend). These properties comprise most of the properties for a product.

- Each time series has its own properties (e.g., symbol, color).
- If zooming is enabled, then zooming in one graph causes the same zoom to occur in related graphs. Each graph (and the reference graph) is assigned a *zoom group* number. This is used to indicate which graphs should zoom together. Currently, all graphs are in the same zoom group.

Right-clicking on a graph and pressing the **Properties** item in the popup menu will display the properties for the graph. The following figures illustrate the properties tabbed panel:



Tabbed Panel to Edit Time Series Product Properties

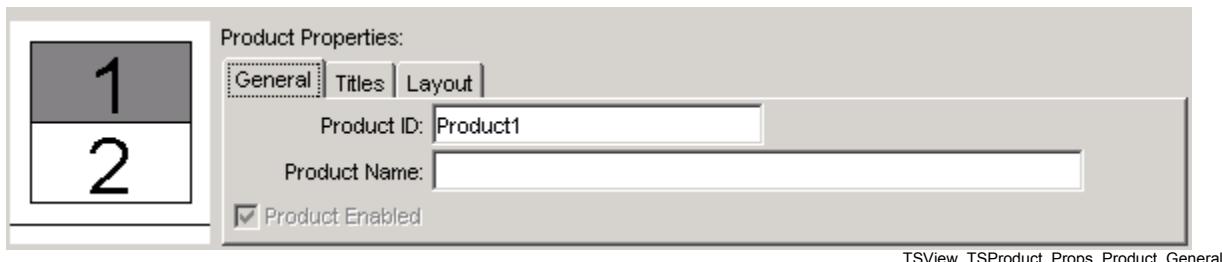
The time series product properties display as three layers of tabbed panels. Characteristics of the properties window are:

- The window is divided into a layout area (top-left) and tabs for different groups of properties. The layout window shows the overall layout of graphs on a page and allows manipulation of the time series product by dropping time series onto the layout.

- The top layer of tabs (**Product Properties**) are associated with product properties (the page).
- The middle layer of tabs (**Graph Properties**) are associated with subproduct properties (graphs on the page). The graph of interest is selected using the drop-down choice that shows the graph number and graph main title. When initially displayed, the selected graph is the one that was clicked on to display the **Properties** menu.
- The bottom layer of tabs (**Time Series Properties**) are associated with data (time series) properties. A time series within a graph is selected using the drop-down choice that shows the time series number within the graph, and the time series identifier. When initially displayed, the first time series for the selected graph is selected.
- The **Apply** button will apply the current properties and update the graph(s). **Warning - when changing between graphs and time series (where multiple graphs and/or time series exist for a product), properties that are changed are applied automatically. This behavior is being evaluated.**
- The **Close** button will apply the current properties, update the graph(s), and close the properties window.
- Only properties read from an original time series product file or that are set by the user will be saved if a time series product is saved. Internal defaults are not saved. This minimizes the size and complexity of product definition files.

The remaining discussion in this section illustrates each of the tabbed panels. The text-based properties that are displayed in the panels are described in the **Time Series Product Reference** section.

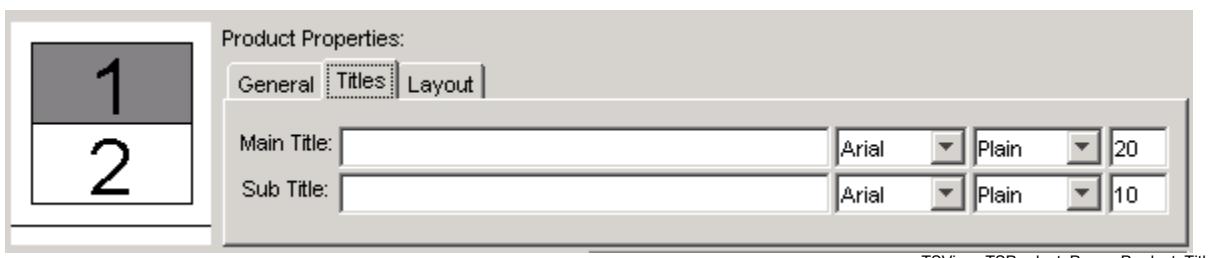
Product Properties - General



Example Product General Properties

The above figure illustrates the product **General** properties. The **Product Enabled** checkbox indicates whether the product is enabled (currently view-only). The **Product ID** is used when saving the product definition to a database. The **Product Name** is also used to when displaying lists of products.

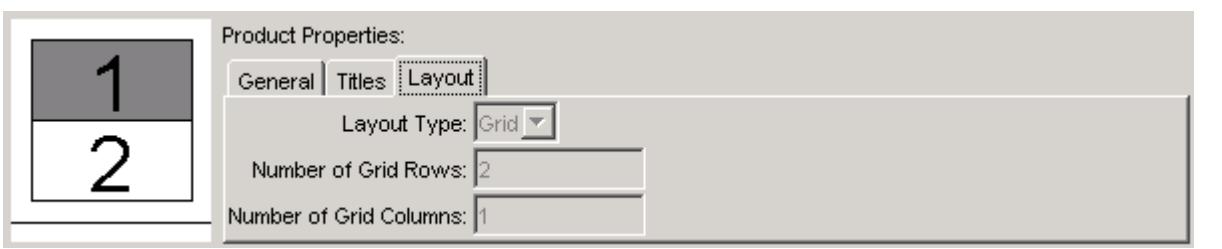
Product Properties - Titles



Example Product Title Properties

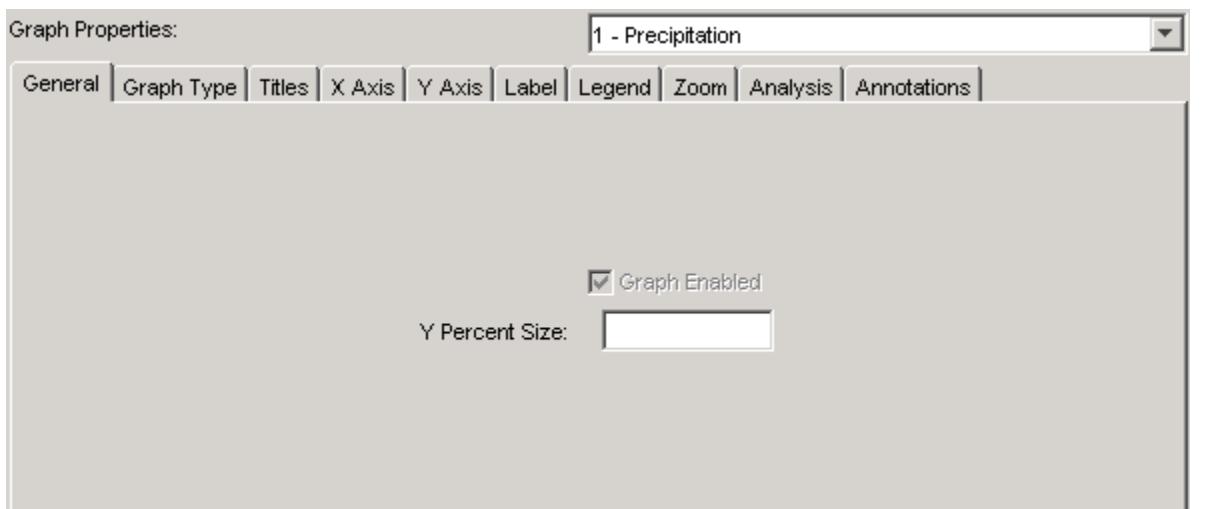
Product **Titles** properties include title and subtitle. If blank, no title will be shown. Because graphs (subproducts) also have a title and subtitle, the product titles are often only used when multiple graphs are included on a page.

Product Properties - Layout



Example Product Layout Properties

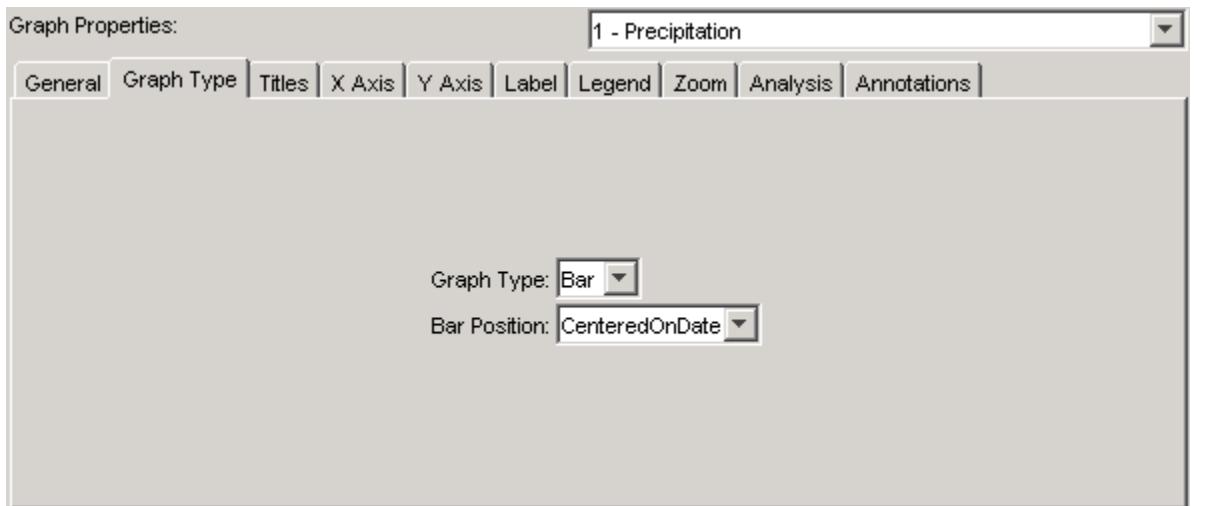
Product **Layout** properties describe how graphs are laid out on the page. Currently, graphs can only be organized in a vertical stack, although the design will support multiple columns. The layout properties are updated automatically as graphs are added to or deleted from the layout window at the left. The relative size of each graph on the page is controlled by using the **LayoutYPercent** general property for each graph on the page (see below).

Graph Properties - General

TSView_TSProduct_Props_Graph_General

Example Graph General Properties

The above figure illustrates graph **General** properties. The **Graph Enabled** checkbox indicates whether the graph is enabled (currently view-only). The vertical size of the graph on the page (percent) can also be specified (the default is to size all the graphs on the page equally).

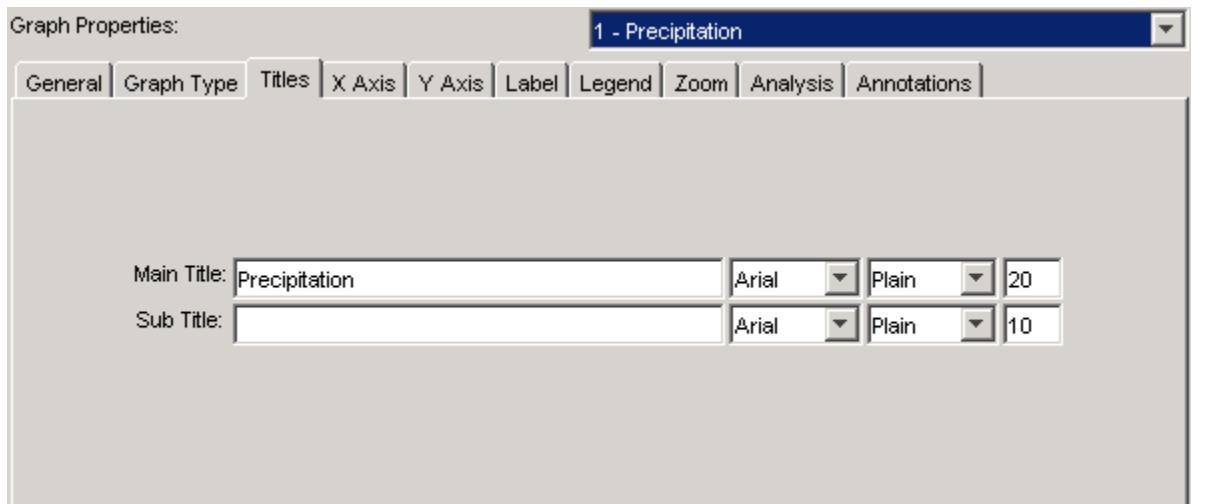
Graph Properties - Graph Type

TSView_TSProduct_Props_Graph_GraphType

Example Graph Graph Type Properties

Graph Type properties control the overall display of the data. The graph type can be changed after the initial display only when switching between simple graph types (e.g., line and bar graphs). Some graph types may have specific properties (e.g., bar width for bar graphs). If necessary, to change the graph type, you can usually select the type from a main application, and generate a new graph.

Graph Properties - Titles

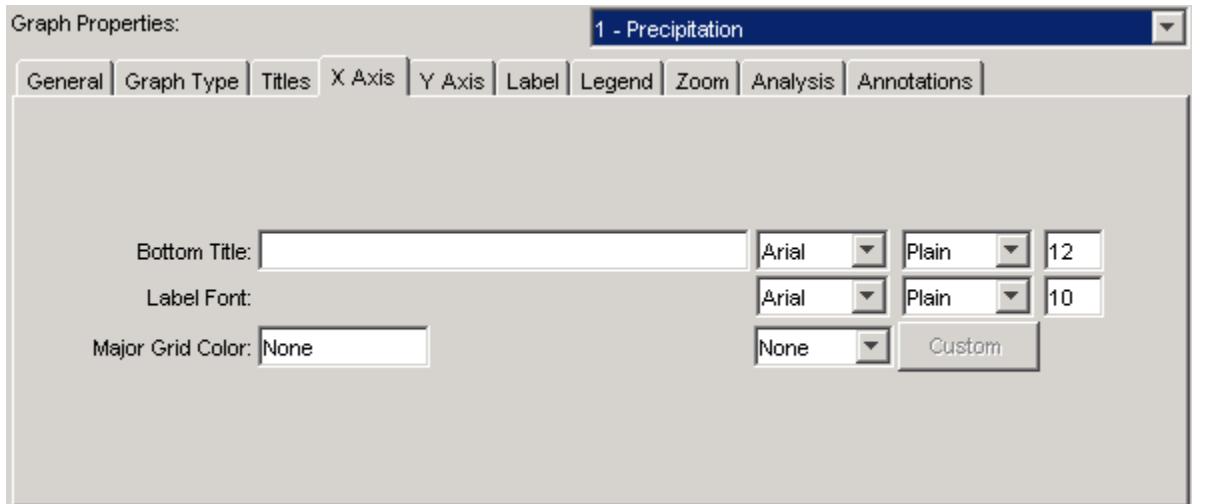


TSView_TSProduct_Props_Graph_Titles

Example Graph Title Properties

Graph **Titles** properties include title and subtitle. If blank, no title will be shown. Font properties can also be specified. After applying the a change to the main title, the title will be added in the list of graphs.

Graph Properties - X Axis

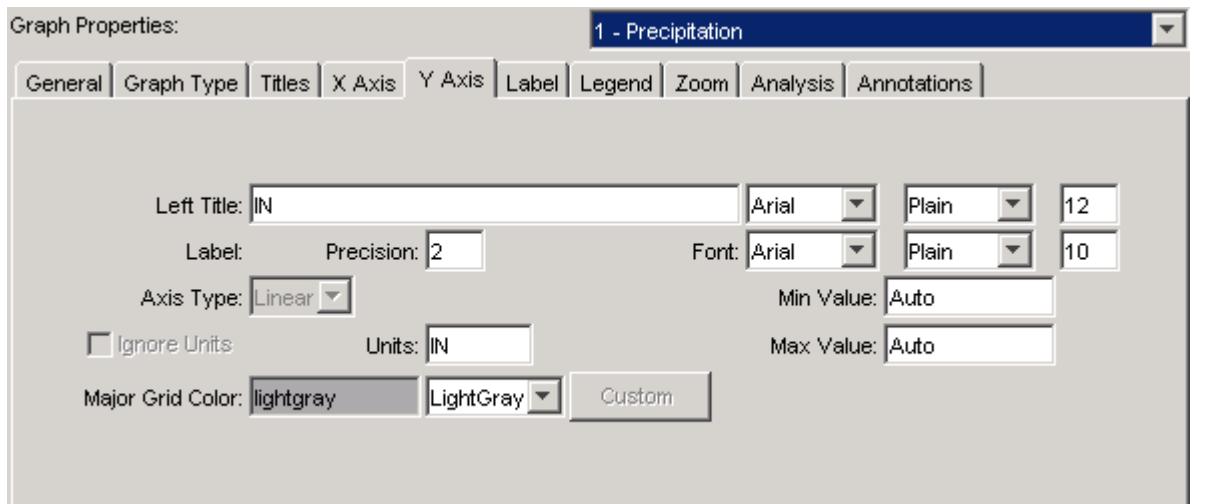


TSView_TSProduct_Props_Graph_XAxis

Example Graph X Axis Properties

Graph **X Axis** properties include title, label, and grid properties. The **Major Grid Color** can be specified by selecting from the available choices, which then fill in the text field with the given color selection.

Graph Properties - Y Axis



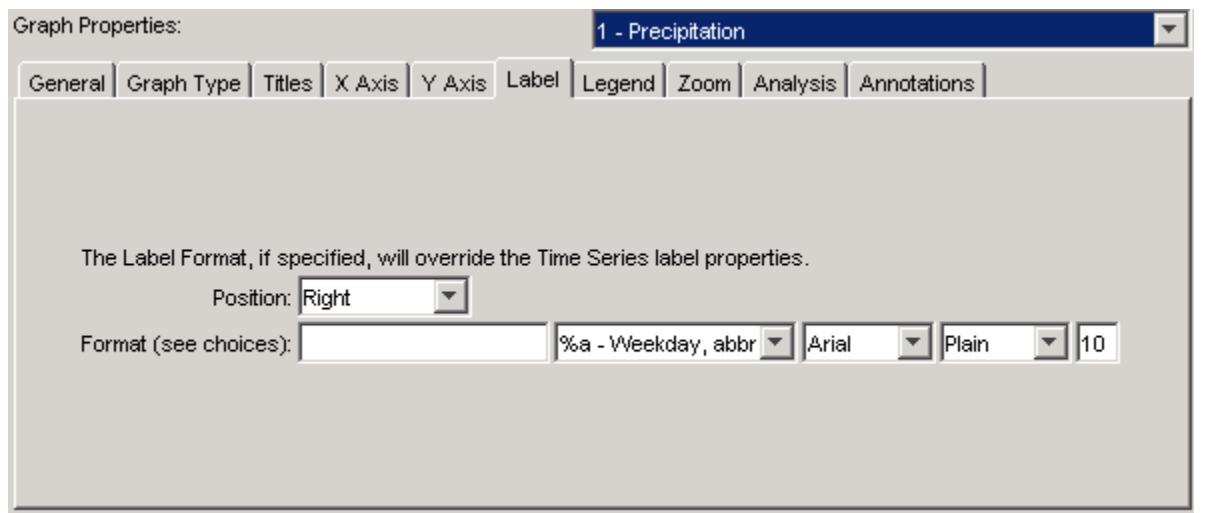
TSView_TSProduct_Props_Graph_YAxis

Example Graph Y Axis Properties

Graph **Y Axis** properties include the following:

- **Left Title** - this may be set to the data units but can be specified (the Y axis title is currently always placed at the top of the Y axis).
- **Label** - the font for labels and precision of numerical labels can be specified.
- **Axis Type** - currently this is view-only.
- **Min Value, Max Value** - currently this is view-only but can be set in time series product definition files (see the **Time Series Product Reference** section).
- **Units, Ignore Units** - currently these are view-only. If time series with incompatible units are graphed, **Ignore Units** will be checked and the units may be shown in the legend.

Graph Properties - Label

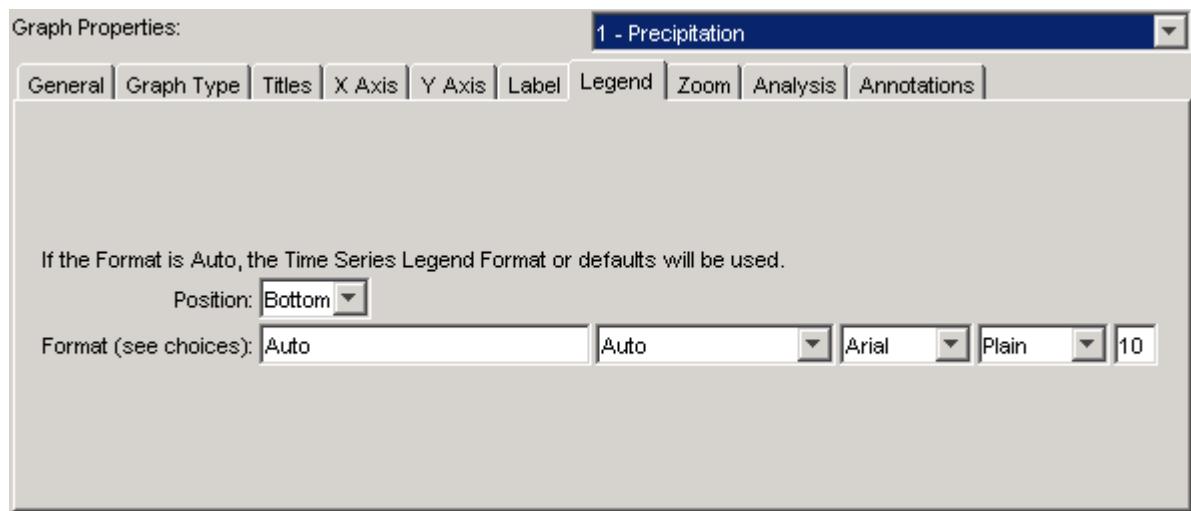


TSView_TSProduct_Props_Graph_Label

Example Label Properties

Data points are not labeled by default because there are usually too many data labels to be legible. However, for plots with limited data, or after zooming in, labels can be useful to identify points without referring to tabular data. The label format can be defined using the choices next to the text field or by entering literal text. For an XY Scatter plot, repeat the %v format (e.g., %v, %v) to show the independent (X) and dependent (Y) data values. See the **DataLabel** properties in the **Time Series Product Reference** section for label options.

Graph Properties - Legend

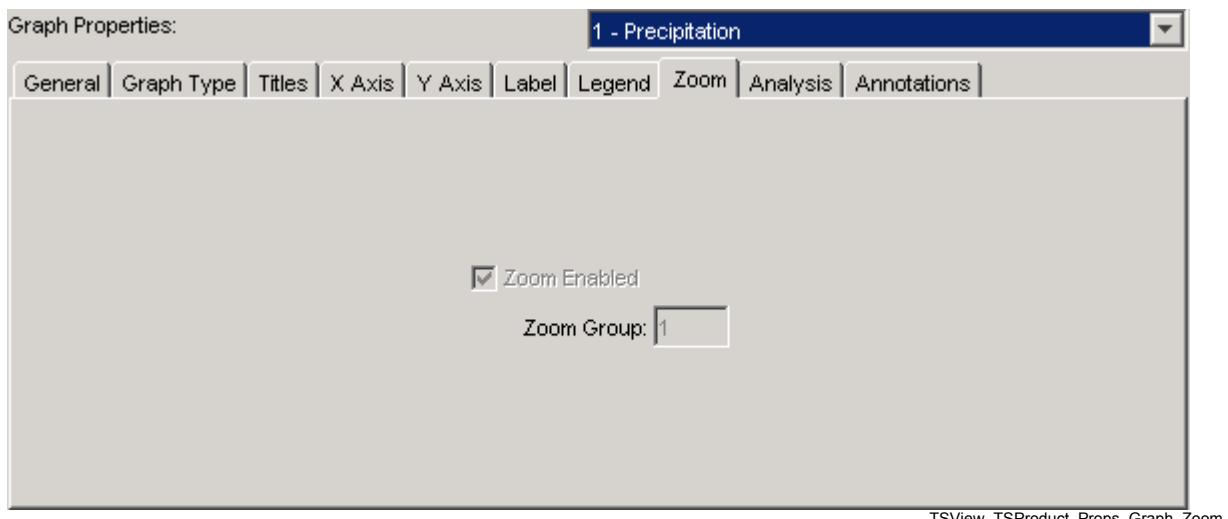


TSView_TSProduct_Props_Graph_Legend

Example Graph Legend Properties

Graph **Legend** properties include format and font properties. If the **Legend Format** is **Auto**, a default legend format will be constructed from the time series description, identifier, and period of record. See the **LegendFormat** property in the **Time Series Product Reference** section for legend formatting options.

Graph Properties - Zoom

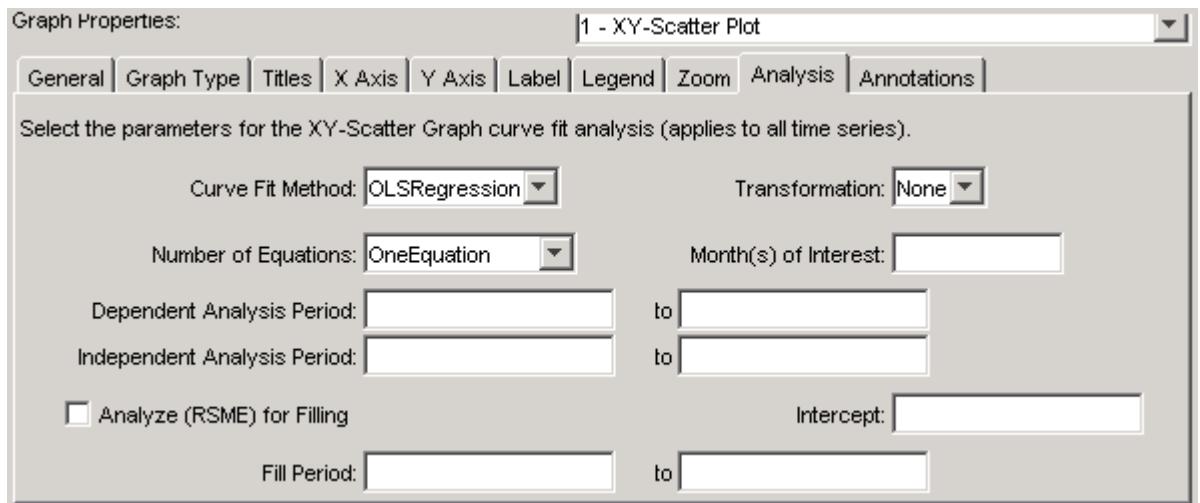


TSView_TSProduct_Props_Graph_Zoom

Example Graph Zoom Properties

Graph **Zoom** properties are currently view-only. Zoom will be enabled for graph types that support it (e.g., duration graphs do not). The zoom group indicates how graphs should respond when other related graphs on a page are zoomed and currently defaults to 1 for all graphs.

Graph Properties - Analysis

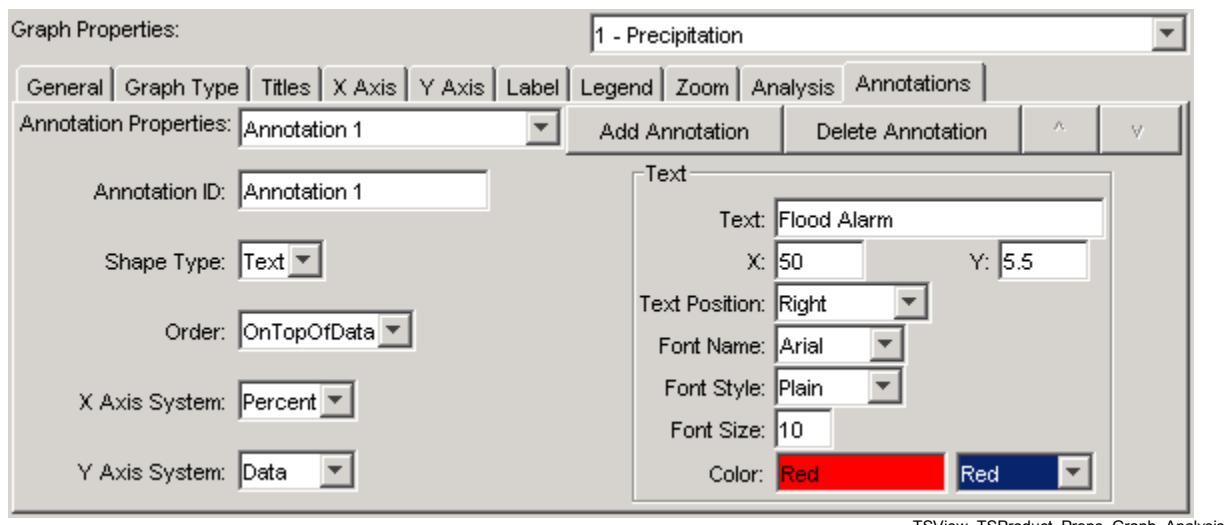


TSView_TSProduct_Props_Graph_Analysis

Example Graph Analysis Properties

Graph **Analysis** properties are available if the graph requires some type of analysis to produce the result (e.g., curve fitting). See also the analysis tab for individual time series. For help with input, place the mouse cursor over a field and a tool tip will be shown.

Graph Properties - Annotations

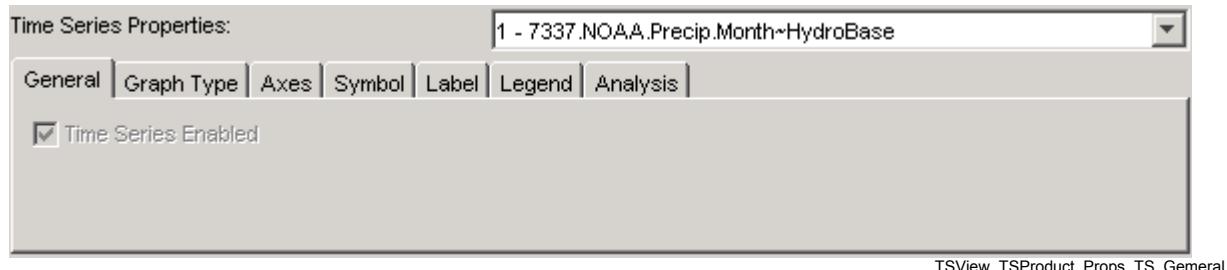


Example Graph Analysis Properties

Graph **Annotations** properties are used to add annotation objects to a graph. Annotations are text, line, or other simple shapes and are stored as simple text properties in time series products (see the **Time Series Product Reference** section below for more information). Annotations are placed on a graph using data units or a percent of the graph dimension. This allows annotations to move if a graph uses real-time data.

To add an annotation, press the **Add Annotation** button. Then select the **Shape Type** and specify annotation properties, as appropriate. The example shown in the above figure places the string “Flood Alarm” at the horizontal (X) center of the graph at a Y-coordinate of 5.5. A horizontal annotation line could also be drawn using 0 to 100 percent on the X axis at the same Y-coordinate.

Time Series Properties - General

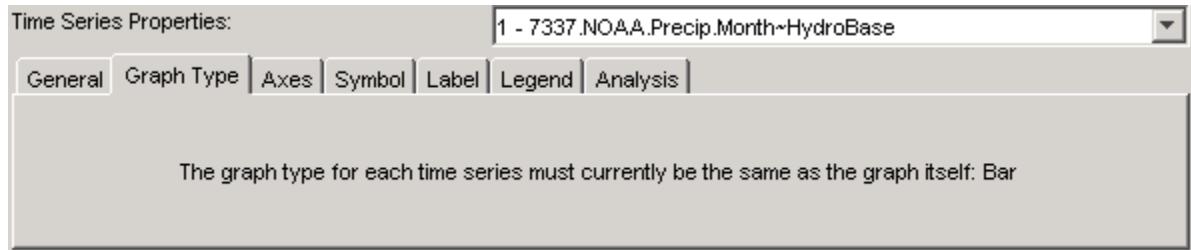


TSView_TSProduct_Props_TS_General

Example Time Series General Properties

Time series **General** properties are currently view-only and indicate whether the time series is enabled for the graph.

Time Series Properties - Graph Type

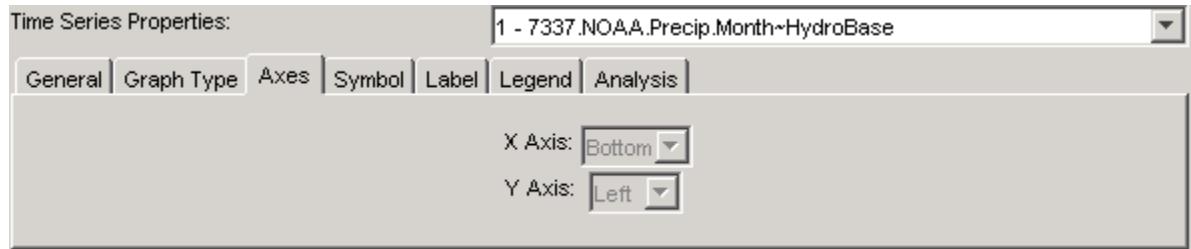


TSView_TSProduct_Props_TS_GraphType

Example Time Series Graph Type Properties

Time series **Graph Type** properties are currently disabled. Currently all time series in a graph must have the same graph type.

Time Series Properties - Axes

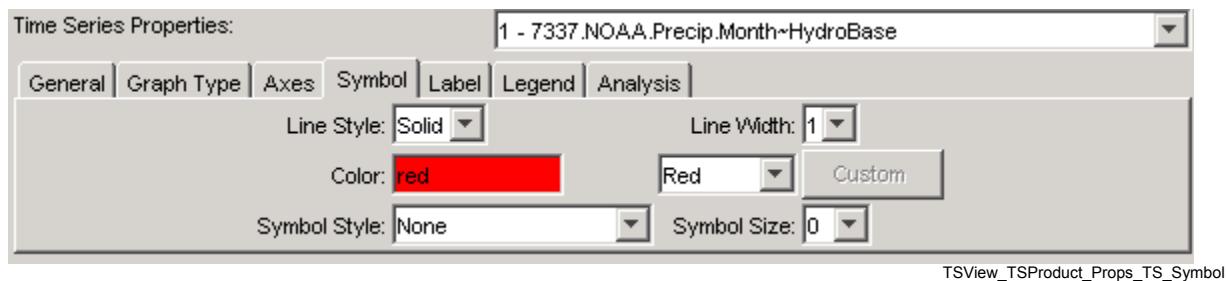


TSView_TSProduct_Props_TS_Axes

Example Time Series Axes Properties

Time series **Axes** properties are currently view-only and show the graph axes to which a time series is associated.

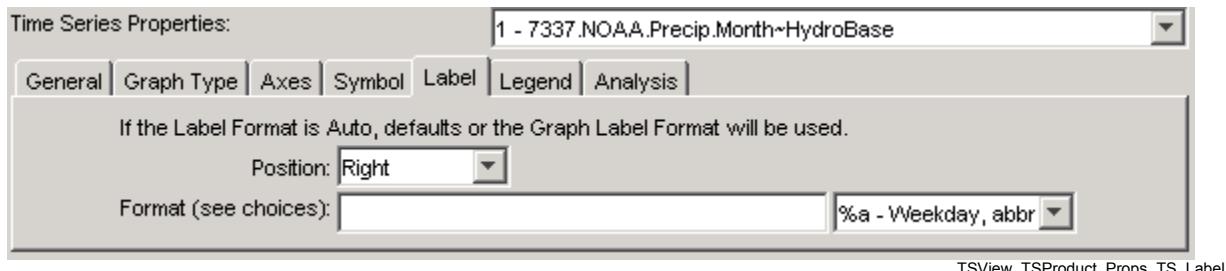
Time Series Properties - Symbol



Example Time Series Symbol Properties

Time series **Symbol** properties define the graphical appearance of time series data. Properties are enabled/disabled based on the graph type (e.g., the **Symbol Style** will be disabled if the graph type is Bar). The symbol properties are consistent with the GeoView tools used for maps.

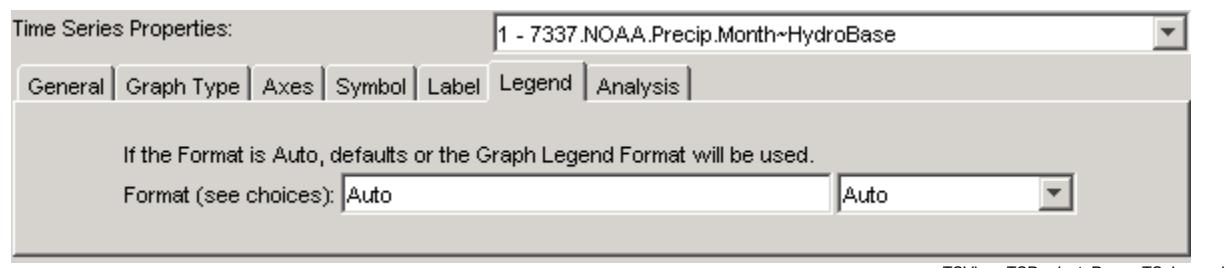
Time Series Properties - Label



Example Time Series Label Properties

Time series **Label** properties allow the data label to be changed. Data points are not labeled by default because there are usually too many data labels to be legible. However, for plots with limited data, or after zooming in, labels can be useful to identify points without referring to tabular data. The label format can be defined using the choices next to the text field or by entering literal text. For an XY Scatter plot, repeat the %v format to show the independent (X) and dependent (Y) data values. See the **DataLabel** properties in the **Time Series Product Reference** section for label options.

Time Series Properties - Legend

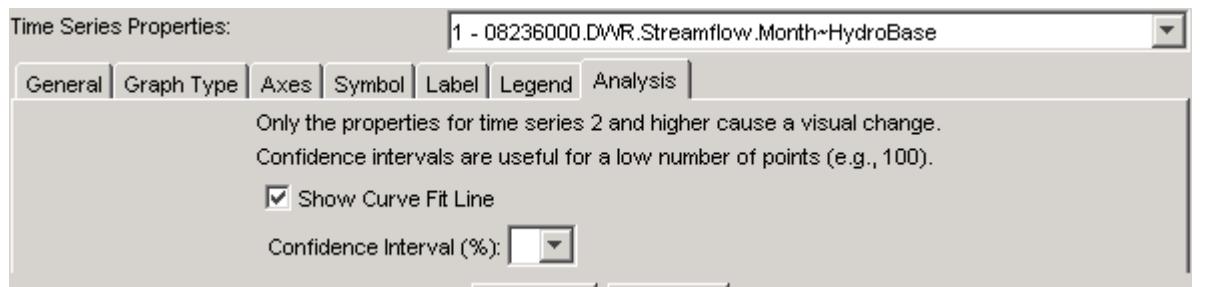


Example Time Series Legend Properties

Time series **Legend** properties allow the legend format to be changed. This is useful if the time series is to have different legend labeling than the other time series in the graph. If the **Legend Format** is Auto, a default legend format will be constructed from the time series description, identifier, and period of record.

See the **LegendFormat** property in the **Time Series Product Reference** section for legend formatting options.

Time Series Properties - Analysis



TSView_TSProduct_Props_TS_Analysis

Example Graph Analysis Properties

Time Series **Analysis** properties are available if the graph requires some type of analysis to produce the result (e.g., curve fitting).

Changing a Graph Page Layout

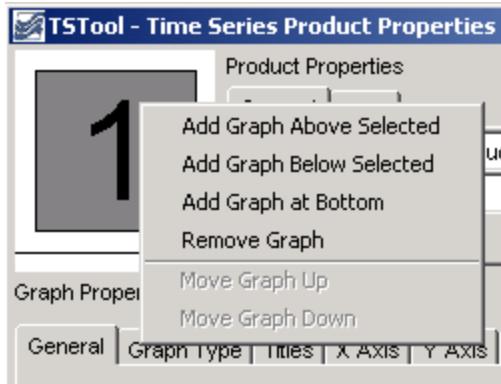
The default page layout for graphs is to display all time series in one graph. In this configuration, the layout area at the top-left corner of the time series product window will display as shown below:



TSView_Layout_1Graph

Layout Window Showing One Graph

The layout area can be used to split the single graph into multiple graphs on the page. For example, two graphs may be needed because of different units, time step, or graph type. Left-clicking on a graph in the layout area will select the graph – the selected graph is shown in gray. Right-clicking on the layout area displays a menu with available options:



TSView_Layout_Menu

Layout Window Menu

The actions taken by the menus are described below:

- | | |
|---------------------------------|---|
| Add Graph Above Selected | Add a new graph above the selected graph, renumbering the graphs as needed. |
| Add Graph | Add a new graph below the selected graph, renumbering the graphs as needed. |
| Add Graph at Bottom | Add a new graph below all existing graphs, giving the new graph the next number in the sequence. |
| Remove Graph | Remove the selected graph, renumbering the graphs as needed. |
| Move Graph Up | Move the graph up one in the sequence, renumbering the graphs as needed. The menu is enabled only when multiple graphs are available. |
| Move Graph Down | Move the graph down one in the sequence, renumbering the graphs as needed. The menu is enabled only when multiple graphs are available. |

When a new graph is added, it will not have any specific properties, time series data, or annotations, other than the default properties that are assigned (e.g., the default graph type is Line), and when drawn it will appear as a blank area. To see the graph, it will be necessary to set the graph's properties and provide it with data (and optionally, annotations). Properties and annotations are defined using the properties tabs as documented in previous sections – use the **Apply** button to apply and view the changes. To set graph properties, the graph to be modified should be selected from the choices at the top of the **Graph Properties** tab panel (or selecting the graph in the layout window).

To add time series data to the new graph (or an existing graph), two approaches can be taken:

1. Find the time series to be moved using the list in the time series properties panel. It may be necessary to select a graph to find the time series – selecting a graph will not impact the ability to move the time series to a different graph. In the list of time series, hold the left mouse button down over a time series choice and drag the time series to a graph on the layout area. During this process, the cursor will change to a new shape, as shown below:



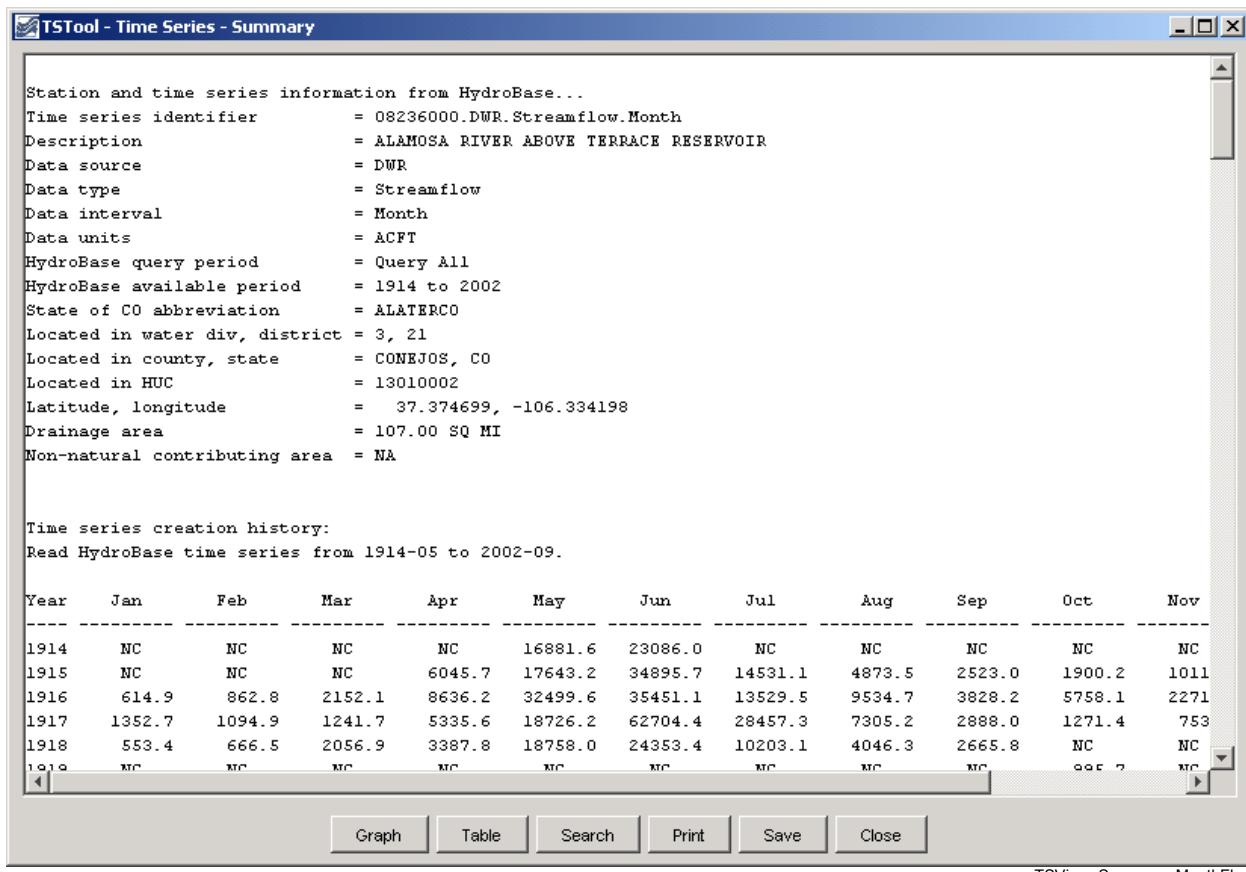
Release the mouse over the graph in the layout area that is to receive the time series. The time series will then be removed from the original graph and will be inserted into the new graph as the last time series in the list.

2. Some software programs will allow dragging a time series from a display to the time series product properties window. Similar to above, drag the time series onto the receiving graph in the layout area. Refer to documentation for the specific software program for additional information about whether this feature is available.

After adding a new graph and moving time series, it may be necessary to change the graph type for a graph. For example, the top graph may show precipitation and the bottom graph may show streamflow resulting from the precipitation. Precipitation is normally shown as bars and streamflow as a line. The graph will initially be shown using the graph type that was originally selected. Change the graph type in the new configuration, as appropriate, by selecting the graph to be changed and then use the **Graph Type** tab.

Time Series Summary View

The time series summary view can be selected from the graph or table view using the **Summary** button. Additionally, applications that use the TSView package may allow displaying a summary from a menu or button option. A time series summary view can usually be produced quickly, whereas the table view uses more resources. The following figure illustrates a typical summary view.



Example Summary View showing Monthly Streamflow

The summary view has the following characteristics:

- The graph view can be displayed using the **Graph** button and the table view can be displayed using the **Table** button.
- Each time series interval (e.g., Month, Day, Hour) has a default summary report format suitable for the interval. This format may be made more specific if time series are read from specific data types (e.g., if daily diversion time series are read from the HydroBase input type, the summary report will use the State of Colorado diversion coding report format).
- The contents of the view can be printed.
- The summary can be saved as a text file or DateValue time series file.
- Limited search capabilities are available to search for a string in the text area.

Time Series Table View

The time series table view can be selected from the graph or summary view using the **Table** button. Additionally, applications that use the TSView package may allow displaying a table from a menu or button option. The table view is useful for viewing date and data values in a spreadsheet-like display. A time series table view for a long period or many time series may require extra time to display, but usually only a few seconds are required. The following figure illustrates a typical table view.

The screenshot shows a Windows application window titled "TSTool - Time Series - Table". The main area is a grid table with two columns: "DATE" and "Streamflow, ACFT". The "DATE" column lists months from 1914-05 to 1916-04. The "Streamflow, ACFT" column lists corresponding values. The window has standard Windows controls (minimize, maximize, close) and buttons at the bottom labeled "Graph", "Summary", "Save", and "Close". A status bar at the bottom indicates "Currently-selected worksheet: Interval base: Month x 1".

DATE	08236000, Streamflow, ACFT	08236500, Streamflow, ACFT
1914-05	16881.6	
1914-06	23086.0	
1914-07		
1914-08		
1914-09		
1914-10		
1914-11		
1914-12		
1915-01		
1915-02		
1915-03		
1915-04	6045.7	
1915-05	17643.2	
1915-06	34895.7	
1915-07	14531.1	
1915-08	4873.5	
1915-09	2523.0	
1915-10	1900.2	
1915-11	1011.6	
1915-12	614.9	
1916-01	614.9	
1916-02	862.8	
1916-03	2152.1	
1916-04	8636.7	

Example Table View showing Monthly Streamflow

TSView_Table_MonthFlow

Characteristics of the table view are:

- The summary view can be displayed using the **Summary** button and the graph view can be displayed using the **Graph** button.
- The precision of dates matches the data interval for the time series.
- If time series with different intervals are selected, multiple tables will be displayed in the window.
- The table contents can be saved as a DateValue file, which is a useful delimited format file.

Time Series Product Reference

A *time series product* is a report, table, or graph, although currently TSView focuses on graph products. Examples of time series products and their use are:

- Reports and graphs generated from a database to perform quality checks.
- Reports and graphs generated from model input and output to check a calibration or model results.
- Reports and graphs generated from a database for real-time data products, to monitor current conditions or to create products for a web site.

The TSView package contains features to process time series product files in interactive and batch mode to produce time series products. Currently, only graph products are supported. The time series graph view allows a graph to be saved as a time series product file. This file describes the layout and contents of the product but does not include the time series data itself; therefore, the time series product is relatively small.

Time Series Product File Format

The time series product definition file format consists of comments (lines that start with #), sections (indicated by []), and simple `property=value` pairs. The following example illustrates the parts of a product file:

```
# Example Time Series Product file
# Comments start with #
# Sections are enclosed in [] and must be included

[Product]

# product properties - surround with double quotes if values contain spaces
xxxxx="xxxxxxxxxx"

[SubProduct 1]

# "sub-product", e.g., a graph on a page (page is product and may have
# multiple graphs)

[Data 1.1]

# First data item in the SubProduct (e.g., first time series).
TSID = ...

[Data 1.2]

# Second data item in the SubProduct (e.g., first time series).
TSID = ...

[SubProduct 2]

[Data 2.1]

# Annotations are associated with a SubProduct
[Annotation 2.1]

Annotation properties...
... etc. ...
```

Example Time Series Product File

Most properties, if not specified in the file, will default to reasonable values. The most important property is TSID, which indicates time series identifier to be read for data. The time series identifier follows the conventions described in the **Time Series Terminology** section. Some tools, like TSTool, will match the TSID against time series that have already been read into memory, or, if necessary, read the time series from a file or database if not in memory. The normal convention is to use a *.tsp* extension for time series product file names.

The list of properties that can be used in a time series product definition file is quite extensive and new properties are added as new features are enabled. As shown in the previous section, properties are defined as simple variable=value pairs. These properties are used internally by the graph view (and its properties window) regardless of whether the graph originated from a product file or interactively. The following tables list the properties that are currently supported or envisioned to be enabled in the future. The first set of properties are used to define the overall product (the full page).

Top-level Time Series Product Properties

Product Property	Description	Default
Current DateTime	The current date and time to be drawn as a vertical line on all graphs. If the property is not specified, no current date/time line will be drawn. If specified as Auto, the current system time will be used for the date/time. If specified as a valid date/time string (e.g., 2002-02-05 15), the string will be parsed to obtain the date/time. This property is often specified internally by the application at run time.	Not drawn.
Current DateTime Color	Color to use to draw the current date and time. Colors can be specified as named colors (e.g., red), hexadecimal RGB values (e.g., 0xFF0000), integer triplets (e.g., 255, 0, 0) or floating point triplets (e.g., 1.0, 0.0, 0.0).	Green
Enabled	Indicates whether the product should be processed. Specify as True or False.	True
LayoutNumber OfColumns	The number of columns in the product.	Currently always 1.
LayoutNumber OfRows	The number of rows in the product.	Currently equal to the number of graphs.
LayoutType	Indicates how the graphs in a product are laid out. Only Grid is supported.	Grid
MainTitle FontName	Name of font to use for main title (Arial, Courier, Helvetica, TimesRoman).	Arial
MainTitle FontSize	Size, in points, for main title.	20
MainTitle FontStyle	Font style (Bold, BoldItalic, Plain, PlainItalic).	Plain
MainTitle String	Main title for the product, centered at the top of the page.	No main title.
OutputFile	Output file when graph product is generated in batch mode. This property is often set at run time by the application.	C:\TEMP\tmp.jpg on windows, /tmp/tmp.jpg on UNIX
Owner	An identifier that indicates the owner of the TSProduct, used internally when saving TSProduct definitions to a database that implements permissions.	None – can be blank if permissions are not important.

Top-level Time Series Product Properties (continued)

Product Property	Description	Default
PeriodEnd	Ending date for time series data in the product. The date should be formatted according to common conventions (e.g., YYYY-MM-DD HH:mm), and should ideally be of appropriate precision for the data being queried. This property is often set at run time by the application.	Full period is read.
PeriodStart	Starting date for time series data in the product. The date should be formatted according to common conventions (e.g., YYYY-MM-DD HH:mm), and should ideally be of appropriate precision for the data being queried. This property is often set at run time by the application.	Full period is read.
PreviewOutput	Indicates whether the product should be visually previewed before output. This property is often set at run time by the application and is used to override generation of the outputFile.	false
ProductType	Currently only Graph is supported.	Graph
SubTitleFontName	Name of font to use for subtitle (see MainTitleFontName for font list).	Arial
SubTitleFontSize	Size, in points, for subtitle.	10
SubTitleFontStyle	Font style (see MainTitleFontStyle for style list).	Plain
SubTitleString	Subtitle for the product.	No subtitle.
TotalHeight	Height of the total drawing space, which may include multiple graphs, pixels.	400
TotalWidth	Width of the total drawing space, which may include multiple graphs, pixels.	400

The subproduct properties are associated with the graphs on a page. There can be one or more graphs on a page, each with different properties. It is envisioned that graphs can be grouped into several zoom groups, where zooming in on one graph will cause all graphs to scale similarly. However, at this time, all graphs in a product are placed in a single zoom group. It is also envisioned that graphs will could be placed anywhere on the page; however, at this time, multiple graphs on a page can only be stacked vertically, each using the full width of the page.

The following tables describe the subproduct (graph) properties.

Subproduct (Graph) Properties

Subproduct (Graph) Property	Description	Default
BarPosition	For use with bar graphs. This property controls how bars are positioned relative to the date and can have the values CenteredOnDate, LeftOfDate, or RightOfDate.	CenteredOnDate
BottomXAxisLabelFontName	Name of font for bottom x-axis labels (see Product MainLabelFontName).	Arial
BottomXAxisLabelFontSize	Bottom x-axis labels font size, points.	10
BottomXAxisLabelFontStyle	Bottom x-axis labels font style (see Product MainLabelFontStyle).	Plain
BottomXAxisTitleFontName	Name of font for bottom x-axis title (see Product MainTitleFontName).	Helvetica
BottomXAxisTitleFontSize	Bottom x-axis title font size, points.	12
BottomXAxisTitleFontStyle	Bottom x-axis title font style (see Product MainTitleFontStyle).	Plain
BottomXAxisLabelFormat	Format for X axis labels. Currently this is confined to date/time axes and only MM-DD is recognized.	Determined automatically.
BottomXAxisMajorGridColor	Color to use for the major grid.	Most graph types automatically set to None.
BottomXAxisMinorGridColor	Color to use for the minor grid. This property is not implemented.	None
BottomXAxisTitleString	Bottom X axis title string.	As appropriate for the graph type (often none if dates).
DataLabelFontName	Name of font for data labels (see Product MainLabelFontName).	Arial
DataLabelFontSize	Data label font size, points.	10
DataLabelFontStyle	Data label font style (see Product MainLabelFontStyle).	Plain

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description	Default
DataLabelFormat	<p>Format specifiers to use for labeling data points. If blank, no labels will be drawn. If specified, labels are drawn for line graphs and XY scatter plots.</p> <p>The following format specifiers are available (all other text in the format is treated literally). The last three specifiers are related to time series data and all others are related to the date for a point. The %v specifier can be specified twice for XY Scatter plots to display the X and Y values. If specified and the time series data property is not specified, the graph property will be used.</p>	Blank (no data point labels).
%%	Literal percent.	
%a	Weekday name abbreviation.	
%A	Weekday name.	
%B	Month name.	
%b	Month name abbreviation.	
%d	Day number.	
%H	Hour (0-23), 2-digit.	
%I	Hour (1-12), 2-digit.	
%J	Day of year.	
%m	Month 2-digit.	
%M	Minute, 2-digit.	
%p	AM, PM.	
%S	Second, 2-digit.	
%Y	Year, 2-digit.	
%Y	Year, 4-digit.	
%Z	Time zone.	
%v	Data value, formatted according to units.	
%U	Data units.	
%q	Data flag (e.g., quality).	

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description	Default
DataLabelPosition	Indicates the position of data labels, relative to the data point: UpperRight, Right, LowerRight, Below, LowerLeft, Left, UpperLeft, Above, Center. If specified and the time series data property is not specified, the graph property will be used.	Right
Enabled	Indicates whether the sub-product should be processed. Specify as true or false.	true
GraphHeight	Graph height in pixels. Currently this property is ignored (use Product TotalHeight instead).	Product TotalHeight (minus space for titles, etc.) if one graph, or an even fraction of Product TotalHeight (minus space for titles, etc.) if multiple graphs.
GraphType	Indicates the graph type for all data in a graph product. Available options are: Bar, Duration, Line, PeriodOfRecord, Point, XY-Scatter.	Line
GraphWidth	Graph width in pixels. Currently this property is ignored (use Product TotalWidth instead).	Product TotalWidth (minus space for titles, etc.).
LayoutXPercent	For the product grid layout, the width of the graph as a total width of the product, percent.	100 divided by the number of columns in the layout.
LayoutYPercent	For the product grid layout, the height of the graph as a total width of the product, percent.	100 divided by the number of rows in the layout.
LeftYAxisIgnoreUnits	Indicates whether to ignore units for the left Y axis. Normally, units are checked to make sure that data can be plotted consistently. If this property is set, then the user will not be prompted at run-time to make a decision. Specify as true or false.	If not specified, the units will be checked at run-time and, if not compatible, the user will be prompted to indicate whether to ignore units in the graphs. The property will not be reset automatically but will be handled internally using the interactively supplied value.

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description	Default
LeftYAxisLabelFontName	Name of font for left y-axis labels (see Product MainLabelFontName).	Arial
LeftYAxisLabelFontSize	Left y-axis labels font size, points.	10
LeftYAxisLabelFontStyle	Left y-axis labels font style (see Product MainLabelFontStyle).	Plain
LeftYAxisLabelPrecision	If numeric data, the number of digits after the decimal point in labels.	Automatically determined from graph type and/or data units.
LeftYAxisMajorGridColor	Color to use for the major grid.	Most graph types automatically set to lightgray.
LeftYAxisMax	Maximum value for the left Y Axis.	Auto, automatically determined. If the actual data exceed the value, the property will be ignored.
LeftYAxisMin	Minimum value for the left Y Axis.	Auto, automatically determined. If the actual data exceed the value, the property will be ignored.
LeftYAxisMinorGridColor	Color to use for the minor grid. This property is not implemented.	None
LeftYAxisTitleFontName	Name of font for left y-axis title (see Product MainTitleFontName).	Arial
LeftYAxisTitleFontSize	Left y-axis title font size, points.	12
LeftYAxisTitleFontStyle	Left y-axis title font style (see Product MainTitleFontStyle).	Plain
LeftYAxisTitleString	Left y axis title string. Note that due to limitations in Java graphics, the left y-axis title is placed at the top of the left y-axis so that it takes up roughly the same space as the y-axis labels. The top-most label is shifted down to make room for the title.	As appropriate for the graph type (often the data units).
LeftYAxisType	Left y-axis type (Log, or Linear).	Linear
LeftYAxisUnits	Left y-axis units. This property is currently used internally and full support is being phased in. See also LeftYAxisIgnoreUnits.	Units from first valid time series, or as appropriate for the graph type.
LegendFontName	Name of font for legend (see Product MainTitleFontName).	Arial
LegendFontSize	Legend font size, points.	10
LegendFontStyle	Legend font style (see Product MainTitleFontStyle).	Plain

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description		Default																																								
LegendFormat	<p>The legend format is composed of literal characters and/or time series data format specifiers, as follows.</p> <table border="1"> <tr><td>Blank</td><td>No legend will be displayed.</td></tr> <tr><td>%%</td><td>Literal percent</td></tr> <tr><td>%A</td><td>Time series alias</td></tr> <tr><td>%D</td><td>Description (e.g., RED RIVER BELOW MY TOWN)</td></tr> <tr><td>%F</td><td>Full time series identifier (e.g., XX_FREE.USGS.QME.24HOUR.Trace 1)</td></tr> <tr><td>%I</td><td>Full interval part of the identifier (e.g., 24Hour).</td></tr> <tr><td>%b</td><td>Base part of the interval (e.g., Hour).</td></tr> <tr><td>%m</td><td>Multiplier part of the interval (e.g., 24).</td></tr> <tr><td>%L</td><td>Full location part of the identifier (e.g., XX_FREE).</td></tr> <tr><td>%l</td><td>Main part of the location (e.g., XX).</td></tr> <tr><td>%w</td><td>Sub-location (e.g., FREE).</td></tr> <tr><td>%S</td><td>The full source part of the identifier (e.g., USGS).</td></tr> <tr><td>%s</td><td>Main data source (e.g., USGS).</td></tr> <tr><td>%x</td><td>Sub-source (reserved for future use).</td></tr> <tr><td>%T</td><td>Full data type (e.g., QME).</td></tr> <tr><td>%t</td><td>Main data type.</td></tr> <tr><td>%k</td><td>Sub-data type.</td></tr> <tr><td>%U</td><td>Data units (e.g., CFS).</td></tr> <tr><td>%z</td><td>Sequence number (used with traces).</td></tr> <tr><td>%Z</td><td>Scenario part of identifier (e.g., Trace1).</td></tr> </table>		Blank	No legend will be displayed.	%%	Literal percent	%A	Time series alias	%D	Description (e.g., RED RIVER BELOW MY TOWN)	%F	Full time series identifier (e.g., XX_FREE.USGS.QME.24HOUR.Trace 1)	%I	Full interval part of the identifier (e.g., 24Hour).	%b	Base part of the interval (e.g., Hour).	%m	Multiplier part of the interval (e.g., 24).	%L	Full location part of the identifier (e.g., XX_FREE).	%l	Main part of the location (e.g., XX).	%w	Sub-location (e.g., FREE).	%S	The full source part of the identifier (e.g., USGS).	%s	Main data source (e.g., USGS).	%x	Sub-source (reserved for future use).	%T	Full data type (e.g., QME).	%t	Main data type.	%k	Sub-data type.	%U	Data units (e.g., CFS).	%z	Sequence number (used with traces).	%Z	Scenario part of identifier (e.g., Trace1).	Auto, which uses Description, Identifier, Units, Period
Blank	No legend will be displayed.																																										
%%	Literal percent																																										
%A	Time series alias																																										
%D	Description (e.g., RED RIVER BELOW MY TOWN)																																										
%F	Full time series identifier (e.g., XX_FREE.USGS.QME.24HOUR.Trace 1)																																										
%I	Full interval part of the identifier (e.g., 24Hour).																																										
%b	Base part of the interval (e.g., Hour).																																										
%m	Multiplier part of the interval (e.g., 24).																																										
%L	Full location part of the identifier (e.g., XX_FREE).																																										
%l	Main part of the location (e.g., XX).																																										
%w	Sub-location (e.g., FREE).																																										
%S	The full source part of the identifier (e.g., USGS).																																										
%s	Main data source (e.g., USGS).																																										
%x	Sub-source (reserved for future use).																																										
%T	Full data type (e.g., QME).																																										
%t	Main data type.																																										
%k	Sub-data type.																																										
%U	Data units (e.g., CFS).																																										
%z	Sequence number (used with traces).																																										
%Z	Scenario part of identifier (e.g., Trace1).																																										
LegendPosition	<p>Position of the legend relative to the graph: Bottom, InsideLowerLeft, InsideLowerRight, InsideUpperLeft, InsideUpperRight, Left, None, Right.</p>		Bottom																																								

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description	Default
MainTitleFontName	Name of font to use for graph main title (see Product MainTitleFontName).	Arial
MainTitleFontSize	Size, in points, for graph main title.	10
MainTitleFontStyle	Graph main title font style (see Product MainTitleFontStyle).	Plain
MainTitleString	Main title for the graph.	None, or appropriate for graph type.
PeriodEnd	Ending date for time series data in the sub-product. The date should be formatted according to common conventions (e.g., YYYY-MM-DD HH:mm), and should ideally be of appropriate precision for the data being queried. This property is often set at run time.	Full period is read.
PeriodStart	Starting date for time series data in the sub-product. The date should be formatted according to common conventions (e.g., YYYY-MM-DD HH:mm), and should ideally be of appropriate precision for the data being queried. This property is often set at run time.	Full period is read.
RightYAxisLabelFontName	Name of font for right y-axis labels (see Product.MainLabelFontName). This property is not enabled.	Arial
RightYAxisLabelFontSize	Right y-axis labels font size, points. This property is not enabled.	10
RightYAxisLabelFontStyle	Right y-axis labels font style (see Product MainLabelFontStyle). This property is not enabled.	Plain
RightYAxisTitleFontName	Name of font for right y-axis title (see Product MainTitleFontName). This property is not enabled.	Arial
RightYAxisTitleFontSize	Right y-axis title font size, points. This property is not enabled.	12
RightYAxisTitleFontStyle	Right y-axis title font style (see Product MainTitleFontStyle). This property is not enabled.	Plain
RightYAxisTitleString	Right y axis title string. This property is not enabled.	

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description	Default
SubTitleFontName	Name of font to use for graph Sub title (see Product MainTitleFontName).	Arial
SubTitleFontSize	Size, in points, for graph sub title.	10
SubTitleFontStyle	Graph sub title font style (see Product MainTitleFontStyle).	Plain
SubTitleString	Sub title for the graph.	No subtitle.
TopXAxisLabelFontName	Name of font for Top x-axis labels (see Product.MainLabelFontName). This property is not enabled.	Arial
TopXAxisLabelFontSize	Top x-axis labels font size, points. This property is not enabled.	10
TopXAxisLabelFontStyle	Top x-axis labels font style (see Product MainLabelFontStyle). This property is not enabled.	Plain
TopXAxisTitleFontName	Name of font for Top x-axis title (see Product MainTitleFontName). This property is not enabled.	Arial
TopXAxisTitleFontSize	Top x-axis title font size, points. This property is not enabled.	12
TopXAxisTitleFontStyle	Top x-axis title font style (see Product MainTitleFontStyle). This property is not enabled.	Plain
TopXAxisTitleString	Top X axis title string. This property is not enabled.	As appropriate for the graph type.
XYScatterAnalyzeForFilling	Indicate whether the analysis should be used to analyze for filling. If true, then the XYScatterIntercept, XYScatterFillPeriodStart, and XYScatterFillPeriodEnd properties may be specified.	false
XYScatterDependentAnalysisPeriodEnd	Specify the ending date/time for the period to analyze the dependent time series data, to determine the best-fit line.	Blank (analyze full period).
XYScatterDependentAnalysisPeriodStart	Specify the starting date/time for the period to analyze the dependent time series data, to determine the best-fit line.	Blank (analyze full period).
XYScatterFillPeriodEnd	When XYScatterAnalyzeForFilling =true, indicates the ending date/time of the period to fill, using standard date/time string.	Blank (fill full period).

Subproduct (Graph) Properties (continued)

Subproduct (Graph) Property	Description	Default
XYScatterFillPeriodStart	When XYScatterAnalyzeForFilling =true, indicates the starting date/time of the period to fill, using standard date/time string.	Blank (fill full period).
XYScatterIndependentAnalysisPeriodEnd	Specify the ending date/time for the period to analyze the independent time series data, to determine the best-fit line.	Blank (analyze full period).
XYScatterIndependentAnalysisPeriodStart	Specify the starting date/time for the period to analyze the independent time series data, to determine the best-fit line.	Blank (analyze full period).
XYScatterIntercept	The value of A in the best-fit equation A + bx. If specified, the value of B is adjusted accordingly. This property cannot be used with transformed data and if specified must be 0.	Blank (do not force the intercept).
XYScatterMethod	Curve fit method used when analyzing data for the XY Scatter graph (OLSRegression or MOVE2).	OLSRegression
XYScatterMonth	One or more month numbers used when analyzing data for the XY Scatter graph, separated by commas or spaces (1=Jan).	Blank (analyze all)
XYScatterNumberOfEquations	Number of equations used when analyzing data for the XY Scatter graph (OneEquation or MonthlyEquations).	OneEquation
XYScatterTransformation	Data transformation used when analyzing data for the XY Scatter graph (None or Log). This property is not enabled.	None
ZoomEnabled	Indicates whether the graph can be zoomed (true) or not (false).	Graph types are evaluated and the property is automatically set. XY-Scatter and Duration graphs can't zoom.
ZoomGroup	Indicate a group identifier that is used to associate graphs for zooming purposes. For example, there may be more than one distinct group of graphs, each with its own overall period or data limits. The graph types may also be incompatible for zooming. This is an experimental feature and should currently not be specified in product files.	All graphs are assigned to zoom group 1.

Time Series Properties

Each subproduct (graph) includes time series data, and the presentation of each time series can be configured using data (time series) properties. In some cases, properties are layered, allowing a property to be defined for the subproduct (graph) for use by all time series (e.g., legend text).

The following tables list data (time series) properties.

Data (Time Series) Properties

Data (Time Series) Property	Description	Default
Color	Color to use when drawing the data. Examples are named colors (e.g., red), RGB triplets (e.g., 255, 0, 128), and hexadecimal RGB (e.g., 0xFF0088).	Repeating, using common colors.
DataLabelFormat	Data label format specifiers. See the graph DataLabelFormat property. If the graph property is specified and the time series property is not, the graph property will be used.	Blank (no labels).
DataLabelPosition	Data label position. See the graph DataLabelPosition property. If the graph property is specified and the time series property is not, the graph property will be used.	Right
Enabled	Indicates whether the data should be processed. Specify as true or false.	true
GraphType	Indicates the graph type for the data in a graph product. Available options are: Bar, Duration, Line, PeriodOfRecord, Point, XY-Scatter. Currently the sub-product property is used for all data. It is envisioned that this property will be enabled in the future to allow different data representations to be plotted together (e.g., monthly as bars, daily as line).	Property not enabled.
LegendFormat	The legend for the data can be specified and will override the SubProduct LegendFormat property (see that property for details).	Auto
LineStyle	Line style. Currently only None (e.g., for symbols only) and Solid are allowed.	Solid
LineWidth	Line width, pixels. Currently a line width of 1 pixel is always used.	1

Data (Time Series) Properties (continued)

Data (Time Series) Property	Description	Default
PeriodEnd	Ending date for time series data in the data item. The date should be formatted according to common conventions (e.g., YYYY-MM-DD HH:mm), and should ideally be of appropriate precision for the data being queried. This property is often set at run time.	Full period is read.
PeriodStart	Starting date for time series data in the data item. The date should be formatted according to common conventions (e.g., YYYY-MM-DD HH:mm), and should ideally be of appropriate precision for the data being queried. This property is often set at run time.	Full period is read.
RegressionLineEnabled	Indicates whether the regression line should be shown (currently only used with the XY-Scatter graph type). The line is drawn in black (there is currently not a property to set the line color).	true
SymbolSize	Symbol size in pixels.	0 (no symbol)
SymbolStyle	Symbol style. Recognized styles are: <ul style="list-style-type: none"> • None • Arrow-Down, Arrow-Left, Arrow-Right, Arrow-Up • Asterisk • Circle-Hollow, Circle-Filled • Diamond-Hollow, Diamond-Filled • Plus, Plus-Square • Square-Hollow, Square-Filled • Triangle-Down-Hollow, Triangle-Down-Filled, Triangle-Left-Hollow, Triangle-Left-Filled, Triangle-Right-Hollow, Triangle-Right-Filled, Triangle-Up-Hollow, Triangle-Up-Filled • X, X-Cap, X-Diamond, X-Edge, X-Square 	None

Data (Time Series) Properties (continued)

Data (Time Series) Property	Description	Default
TSID	Time series identifier.	Must specify.
XAxis	X-axis to use (Bottom or Top). This currently always defaults to bottom.	Bottom
XYScatterConfidenceInterval	This property is only used with XY scatter plots. If not blank, the value indicates that confidence level lines should be drawn on the XY Scatter plot for the given confidence interval, percent. Currently only 99 and 95 percent confidence intervals are supported. The lines will only be drawn if the curve fit line is drawn (see RegressionLineEnabled).	Blank (do not draw).
YAxis	Y-axis to use (Left or Right). This currently always defaults to left.	Left

Annotation Properties

Annotations are associated with subproducts (graphs) and are implemented as simple shapes that are drawn on normal graphs. It is envisioned that all shapes supported by the drawing package will eventually be supported but currently only text labels and lines can be specified as annotations.

To allow flexibility, annotations can be placed using two coordinate systems. For example, if a product is generated using real-time data, the date/time axis will have a different range over time. Therefore, placing an annotation using a fixed coordinate would cause the annotation to scroll off the graph as time passes. To resolve this issue and still allow absolute positioning of annotations, as appropriate, the following coordinate systems are supported, as specified by the XAxisSystem and YAxisSystem properties:

Data When using the data coordinate system, it is expected that the coordinates used to define the annotation will agree with the data units being drawn. For example, for a normal time series graph, the x-axis coordinate would be specified as a date/time to the necessary precision and the y-axis coordinate would be specified using data values.

It is envisioned that a notation +NNN and -NNN will be implemented in the future to allow offsets from the edges of the graph, in data units.

Percent When using the percent coordinate system, it is expected that the coordinates used to define the annotation are specified as a percent of the graph width or height, with 0 being the lower/left and 100 being the upper/right.

Each axis can have a different coordinate system (e.g., the y-axis value can be set using data units and the x-axis value can be set using percent).

The following tables list annotation properties.

Annotation Properties (All Shapes)

Annotation Property	Description	Default
AnnotationID	A string that identifies the annotation, to be used in software displays. If there are many annotations, this helps identify them when editing.	Annotation + annotation number (1+) (e.g., Annotation1).
Color	Color to use when drawing the annotation. Examples are named colors (e.g., red), RGB triplets (e.g., 255, 0, 128), and hexadecimal RGB (e.g., 0xFF0088).	Black
Order	The drawing order for the annotation, either BehindData to draw behind time series data or OnTopOfData to draw on top of time series data.	OnTopOfData
ShapeType	The type of shape to be drawn for the annotation. Currently accepted values are Text and Line.	None – must be specified.
XAxisSystem	Indicates the system for X coordinates: <ul style="list-style-type: none">• If Data, the X coordinates that are specified will be in data units.• If Percent, the X coordinates are percent of the graph (0% is left and 100% is right).	Data
YAxisSystem	Indicates the system for Y coordinates: <ul style="list-style-type: none">• If Data, the Y coordinates that are specified will be in data units.• If Percent, the Y coordinates are percent of the graph (0% is bottom and 100% is top).	Data

Annotation Properties (Line Shape)

Annotation Property	Description	Default
LineStyle	Line style. Currently only None and Solid are allowed.	Solid
LineWidth	Line width, pixels. Currently a line width of 1 point (pixel) is always used.	1
Points	X and Y coordinates for the line endpoints, as follows: X1 , Y1 , X2 , Y2 or X1 , Y2 X2 , Y2.	None – must be specified.

Annotation Properties (Text Shape)

Annotation Property	Description	Default
FontSize	Annotation text font size, points.	10
FontStyle	Annotation text font style (see Product MainLabelFontStyle).	Plain
FontName	Annotation font name (see Product MainTitleFontName).	Arial
Point	X and Y coordinates for the text position, as follows: X1 , Y1	None – must be specified.
Text	The string to display.	Blank
TextPosition	Indicates the position of text, relative to the point: UpperRight, Right, LowerRight, Below, LowerLeft, Left, UpperLeft, Above, Center.	Right

This page is intentionally blank.

Appendix

GeoView Mapping Tools

Color, 2004-05-27, Original Maintained with TSTool, Acrobat Distiller

Overview

[GeoView Terminology](#)

[The GeoView Panel](#)

[Interacting with the GeoView Map](#)

[Setting GeoView Properties](#)

[Viewing a Layer's Attributes](#)

[Using GeoView with an Application](#)

[Limitations](#)

[GeoView Configuration – the GeoView Project File](#)

[GeoView Project File Examples](#)

Overview

The GeoView package contains integrated software components that can be used with software to enable map-based interfaces. The main purpose of the GeoView package is to provide simple and flexible map displays that can be used in a variety of software applications with little or no reconfiguration.

The GeoView package has been developed by Riverside Technology, inc., using Java technology. GeoView interfaces can be embedded in Java applications (e.g., use GeoView as the main window interface), can be enabled as a separate floating window (e.g., to support an application's features without being embedded in the main window), and can be used in web pages either as embedded map applets or stand-alone map windows. GeoView tools operate similarly on Microsoft Windows and UNIX operating systems.

This appendix describes general GeoView features and can be used as a reference for how to configure and use GeoView components. Specific uses of GeoView in a software program are discussed in the documentation for the specific software. Some of the figures shown in this documentation were generated using GeoView with the TSTool software and consequently title bars include TSTool in the wording.

GeoView Terminology

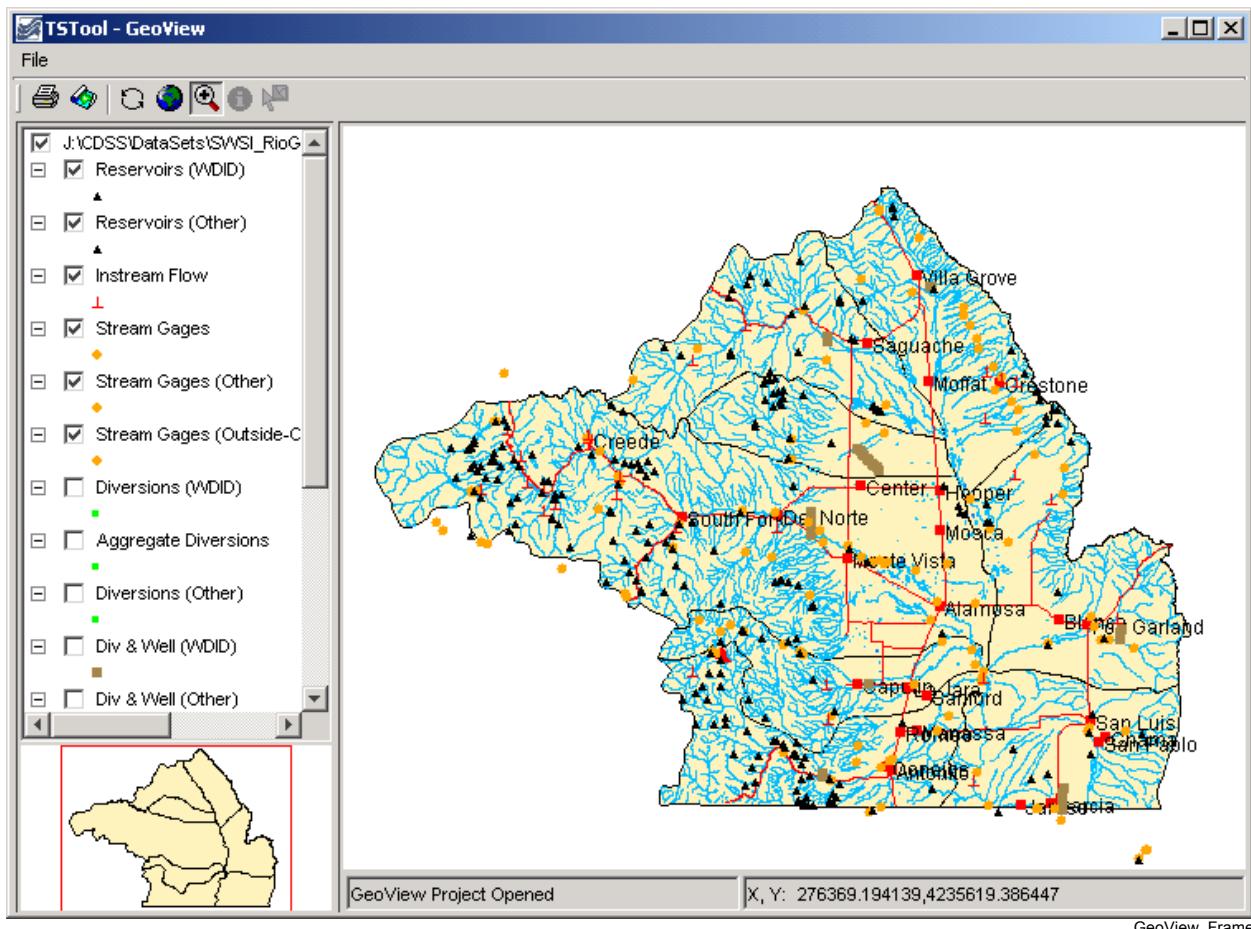
GeoView terminology is similar to other GIS product terminology. Important terms are shown in the following table. These terms are used infrequently in most user interfaces and applications but are visible at times in property dialogs and configuration files.

GeoView Terminology

Term	Description
<i>GeoView</i>	The visible map window where maps are displayed. Currently there can be only one main GeoView. A reference GeoView may be used to show the zoom extents in the main GeoView.
<i>GeoLayer</i>	A data layer, in its "raw" form (e.g., an ESRI Shapefile). The more generic term "layer" is often used.
<i>GeoLayerView</i>	A view of a GeoLayer, with symbol properties for visualization. The more generic term "layer view" is often used. This is equivalent to a "theme" in some software packages.
<i>Feature</i>	A general term describing an item on the map, consisting of shape and attribute data.
<i>Shape</i>	A general term defining the type of feature (e.g., point, polygon). The shape type defines the ways that a feature can be symbolized and used in analysis. Shape types can typically be determined automatically from input data.
<i>Attributes</i>	A general term defining non-shape data that are associated with a feature. Often, attributes are stored in a tabular form, such as a relational database table. Attributes are usually associated with a shape using some type of index number (shape index).
<i>Symbol</i>	The combination of properties used to visualize a layer (e.g., symbol style, color, labeling, classification). The feature shape type controls how the feature can be symbolized.
<i>GeoView Project</i>	A GeoView Project file (.gvp) can be used to define the layers and global viewing properties for a GeoView. The contents of this file are described in more detail in the GeoView Configuration – the GeoView Project File section at the end of this appendix.
<i>Application Layer Type</i>	Because GeoView is a generic tool, it has no implicit understanding of the types of data that are important to an application. The AppLayerType is a property that can be assigned to layers in a GeoView Project file to help an application know that a layer is important to the application. An application layer type of "BaseLayer" indicates that a layer should be used for background information and is not specific to the application. See the Using GeoView with a Software Application section below for more information.

The GeoView Panel

An example GeoView interface is shown in the following figure. This example uses a floating GeoView window. Some programs use a GeoView that is embedded in the main application window, and some rely on secondary map windows (as shown below). If the map is in the main window, the menus at the top of the window will be those specific to the software (whereas below the single GeoView **File** menu is shown).



Example GeoView Interface (from TSTool)

The GeoView Panel is a self-contained component that offers a standard map-based interface that can be used in many applications. In the above figure, the GeoView Panel includes everything shown, except for the top menu bar (with the **File** menu). The general purpose GeoView Frame includes the menu bar and a GeoView Panel. The GeoView Panel contains the following components:

Table of Contents (left edge)	The table of contents displays a list of layer views, showing the top-most layer at the top of the legend. Layers can be enabled/disabled by toggling the check box. A layer can be selected/deselected by clicking on the layer in the table of contents. Layers that are selected can be acted on (e.g., properties can be viewed). The table of contents also indicates the symbol for the layer.
Main GeoView (large map)	The main GeoView displays the enabled layers and allows you to interact with the map using the mouse and keyboard (e.g., zoom, select).

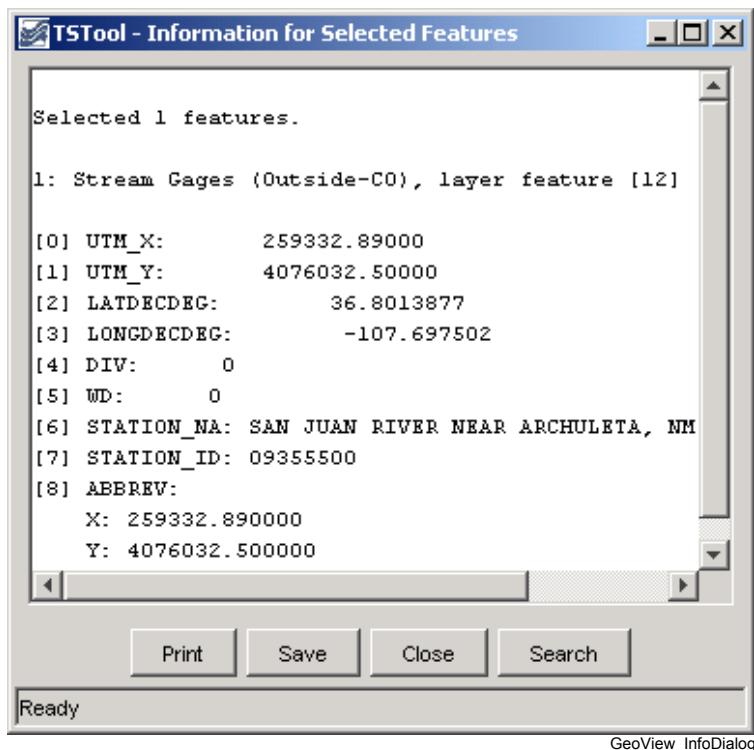
Reference GeoView (lower left)	The reference GeoView displays layers that have the property <code>ReferenceLayer</code> set to <code>true</code> . This view shows the current zoom extent relative to the maximum extent of the data and can also be used to initiate a zoom to a region on the main map (the reference map is always in zoom mode).
Standard Controls (top, below menu bar)	Standard controls perform actions on the visible map as follows:
	Print Print the visible map. You will be able to pick the printer and orientation.
	Save Image Saves the map as a Portable Network Graphics (PNG), JPEG, or other supported graphic file format.
	Refresh Refresh the map display by redrawing features in enabled layers that are in the visible window. This does not re-read the original data. GeoView normally refreshes automatically as needed.
	Zoom Out Zoom to the maximum data extents.
	Select the Mode as Zoom, Info, or Select Select the interaction mode. The Zoom mode allows a rectangle to be drawn on the map to zoom to the specified region and is the default mode if no layer is selected. The Info mode allows features to be selected (by clicking on or drawing a box around), after which geographic information about the features is displayed. The Select mode is similar to Info ; however, its purpose is to select features for an additional action (e.g., exporting data or performing a query). The Info and Select modes are only enabled if one or more layers are selected in the table of contents.
Message Areas (bottom)	See the next section for more information about using these features.

The GeoView Panel components work with each other to provide interaction with the maps, as described below.

Interacting with the GeoView Map

The layers shown on the map are initially displayed according to the GeoView Project settings. Once displayed, you can interact with the map in the following ways:

Disable/Enable a layer view	<p>Layers can be enabled/disabled to make the map more readable or useful:</p> <ol style="list-style-type: none"> 1. Use the check boxes in the table of contents to disable and enable layer views, as appropriate. The map will automatically refresh, resulting in a slight delay as the map is redrawn. 2. If necessary, use the Refresh tool () to cause the map to be updated (automatic refresh may be disabled for some applications, due to performance reasons).
Change layer view order	Currently the layer view order can only be changed by editing the GeoView Project file.
Zoom in/out	<p>Zooming is useful make symbols and labels more readable. To zoom in:</p> <ol style="list-style-type: none"> 1. Set the GeoView interaction mode to "Zoom" by selecting the zoom tool () at the top of the window. 2. Use the mouse to draw a box around an area of interest (left mouse button down to start, move the mouse, and then release). The main GeoView map will zoom to the selected region and the reference map will show the zoom extent. 3. Use the Zoom Out tool () to zoom to the full extent or use the reference GeoView to zoom to a different region.
Change symbols for a layer view	<p>To change the symbols and labels for a layer view:</p> <ol style="list-style-type: none"> 1. Select the layer view in the table of contents 2. Right-click and select the Properties menu. See the Setting GeoView Properties section below for information about the properties.
Display geographic information for features	<p>The GeoView interface can display information about geographic features (shape and attribute data) from the original geographic data. To do so:</p> <ol style="list-style-type: none"> 1. Select layer views in the table of contents that are to be searched for information. 2. Set the GeoView interaction mode to Info () 3. Click near the feature or draw a box around multiple features. The layers will be searched and the following dialog will be shown.



GeoView_InfoDialog

The resulting dialog will show information about the selected features, including basic layer information, and information about the specific shapes and attributes. **The display is for geographic data only.**
Attribute names and values are as they appear in the original data.
Additional application-specific data are typically provided by a separate software interface.

Select features

Features can be selected for a number of reasons. Currently, GeoView has limited select tools, which are mainly used internally when integrated with an application (e.g., an application can select features internally, which are then highlighted on the map). In the future, interfaces to select features from the GeoView interface using query criteria may be added.

Features can be selected () similar to the **Info** mode described above. The selected features are highlighted on the map. In the past, yellow, or cyan have been used to highlight selected features. However, yellow is not clearly visible when earth-tone colors are used for background layers and cyan is not clearly visible when water-tone colors are used for background layers. Therefore, GeoView is phasing in a magenta/pink selection color, which is rarely used for background layers.

Setting GeoView Properties

GeoView properties are initially set in a GeoView Project file or are assigned internally by the software. Most properties control how layers are displayed (colors, labels, etc). To view general GeoView properties, right click on the GeoView map and select the **Properties** menu. Some properties are currently view-only. Refer to the **GeoView Configuration – the GeoView Project File** section below for a complete list of properties that can be defined in a GeoView Project file.



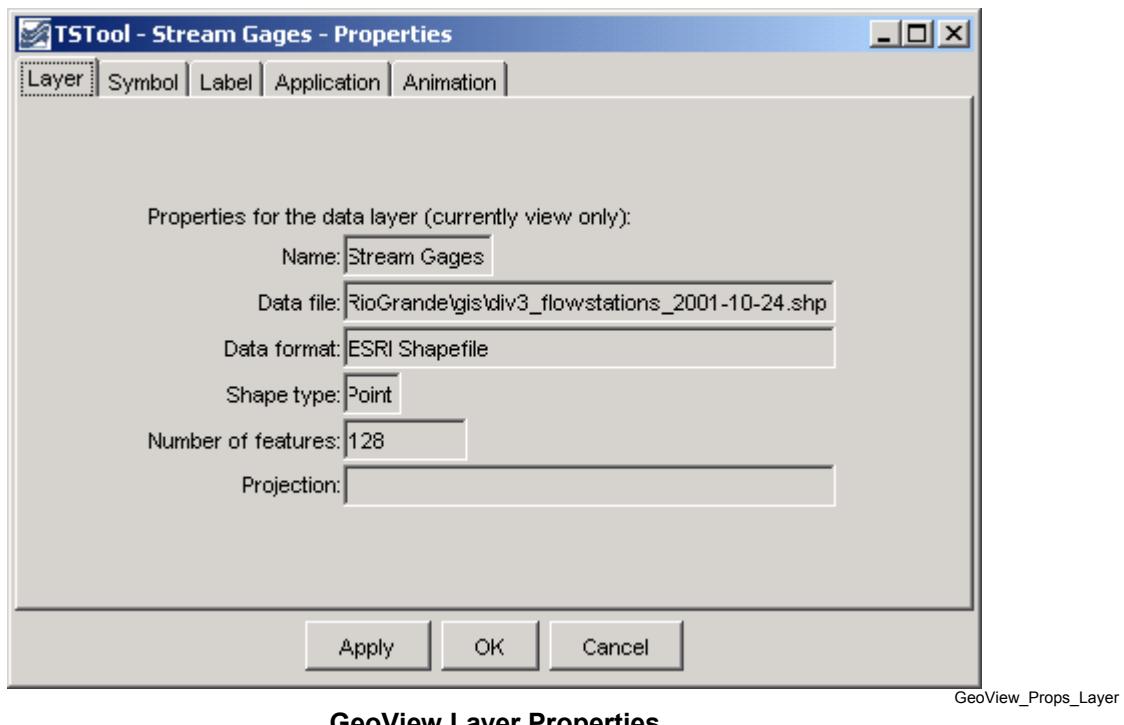
GeoView_Props

Main GeoView Properties

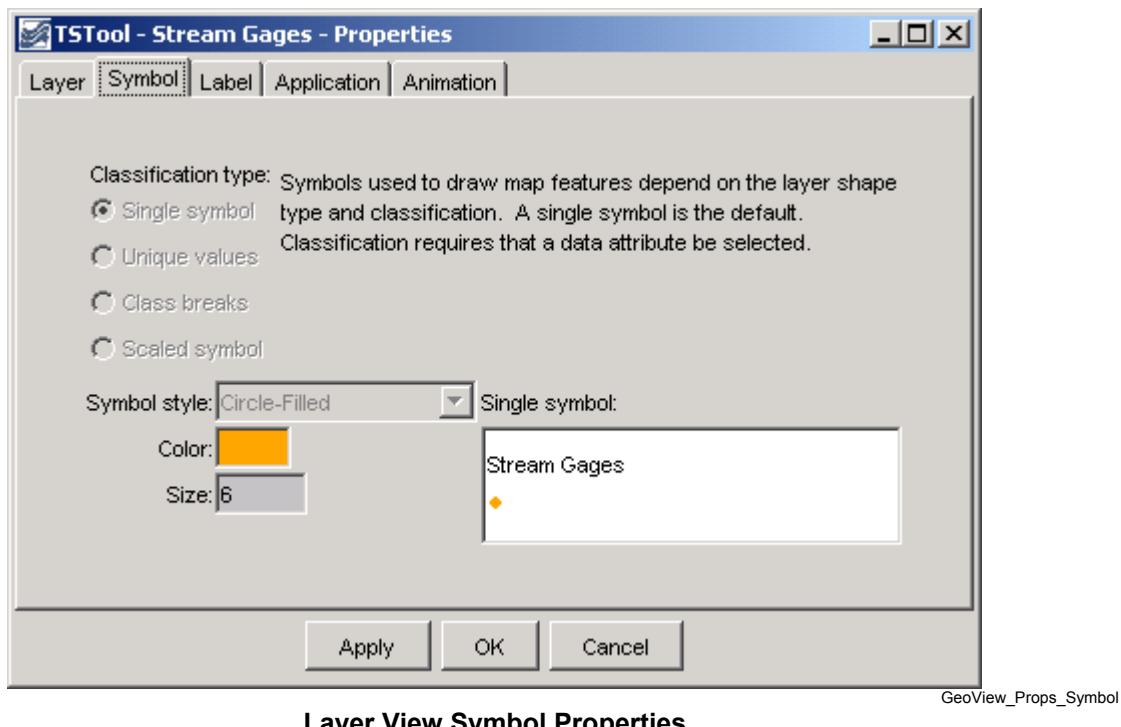
GeoView properties, as shown in the above figure, apply to the main GeoView and are shared between layers. These properties are typically not edited by end users. One important property is the projection property. If all data layers are projected consistently (e.g., for ESRI shapefiles) then a projection does not need to be defined. However, if the layers have different projects, a GeoView projection and projections for each layer can be defined to allow the GeoView to project data consistently for visualization.

If the **OK** or **Apply** buttons are pressed, the GeoView properties will be updated in memory (the GeoView Project file is not updated) and the map will redraw. Pressing **OK** will additionally close the properties dialog. The **Cancel** button causes the dialog to close, without updating the properties.

To view or change properties for a layer, select a layer view in the table of contents, right-click and select the **Properties** menu item. The following tabbed dialog will be displayed for the first selected layer view. The tabbed panels are discussed below the each figure.



GeoView layer properties, as shown in the above figure, apply to the input source. Currently these properties are used for information purposes and cannot be interactively edited.

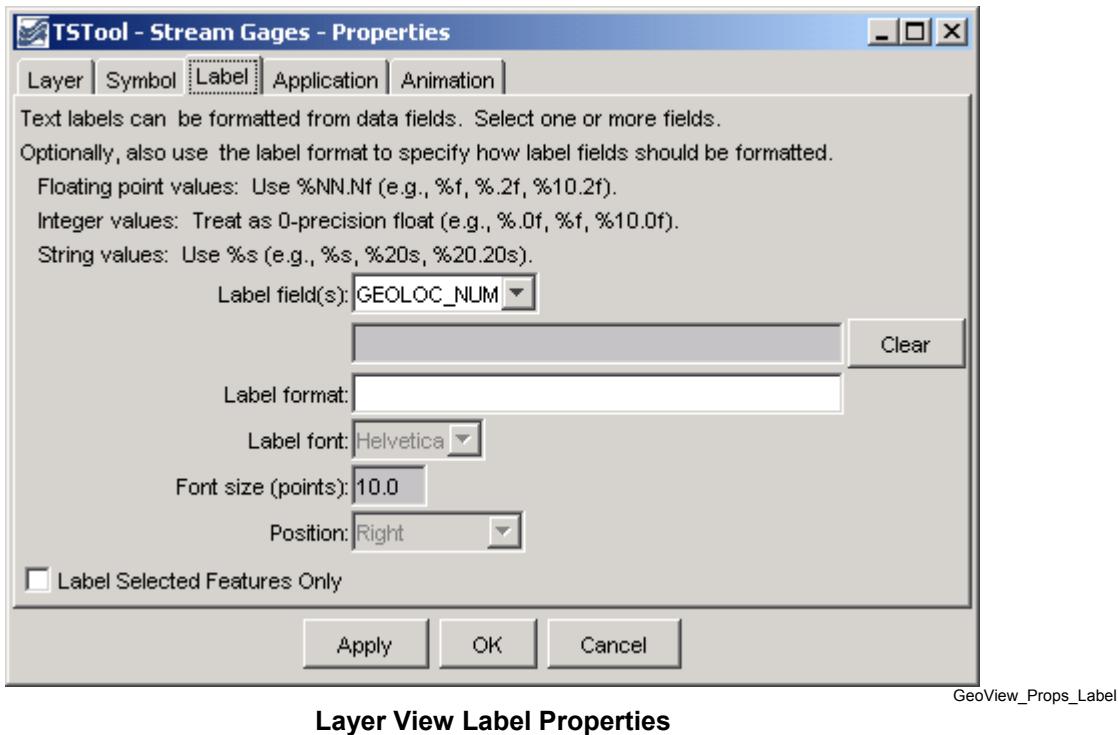


GeoView_Props_Symbol

Layer View Symbol Properties

Symbol properties, as shown in the above figure, indicate how the layer is to be drawn (symbolized) on the map and in the table of contents. A sample of the symbol is shown in the dialog, although it may appear slightly different on the map and table of contents.

Symbol terminology corresponds to standard GIS tools.



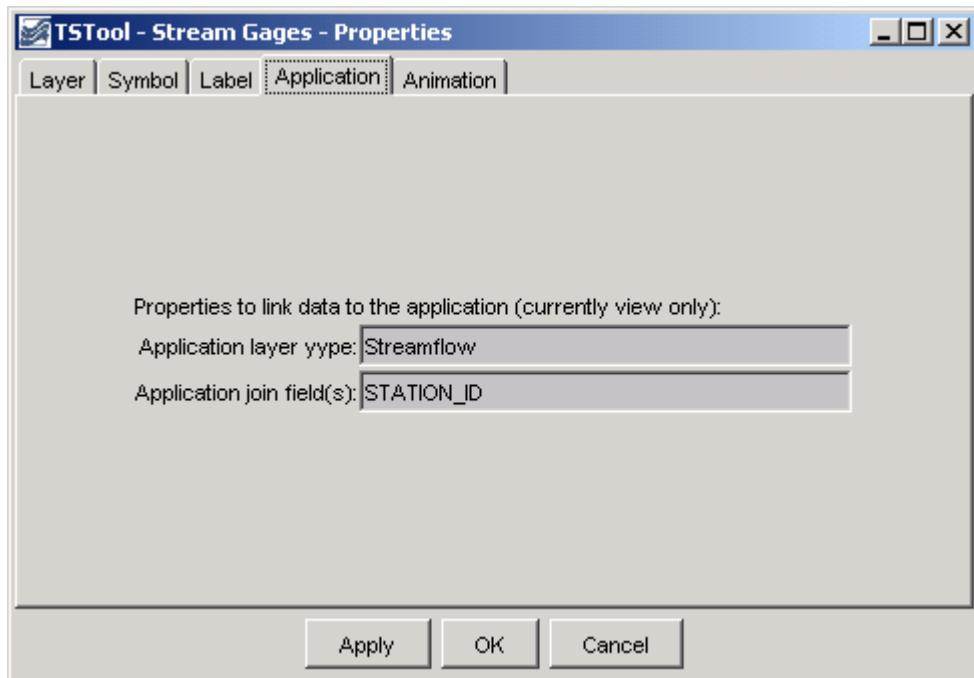
Layer View Label Properties

GeoView_Props_Label

Label properties, as shown in the above figure, can be modified to label features with attribute data or literal strings. Currently, only point features can be labeled. Labels can consist of a combination of attribute values. To label features, select the attribute fields from the available choices, in the order that they should appear in the label.

The label format, if not specified, defaults to the use the full width of the attribute. For example, if an attribute field is defined as being twenty characters wide, the label may be the full width, including leading and trailing spaces. More often, it is desirable to omit the spaces. To do so, or to format numbers using a more appropriate format than the full width default, use the **Label Format** information. The dialog box notes illustrate valid formats. For example, if a string field and an integer field are available, the following label format would show the labels with only a comma and one space between the values:

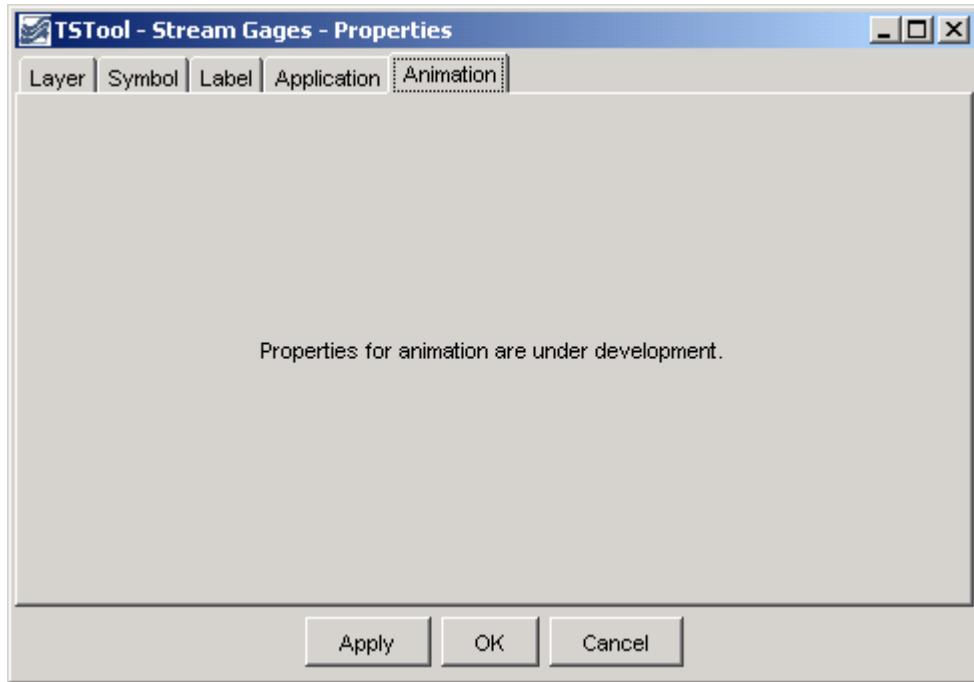
`%s, %d`



GeoView_Props_Application

Application Layer Type

Layer application properties (above) are used to link a layer's data to an application. This process allows general GeoView features to be used more specifically by specific software programs. The **Using GeoView with a Software Application** section (see below) describes this functionality in more detail.



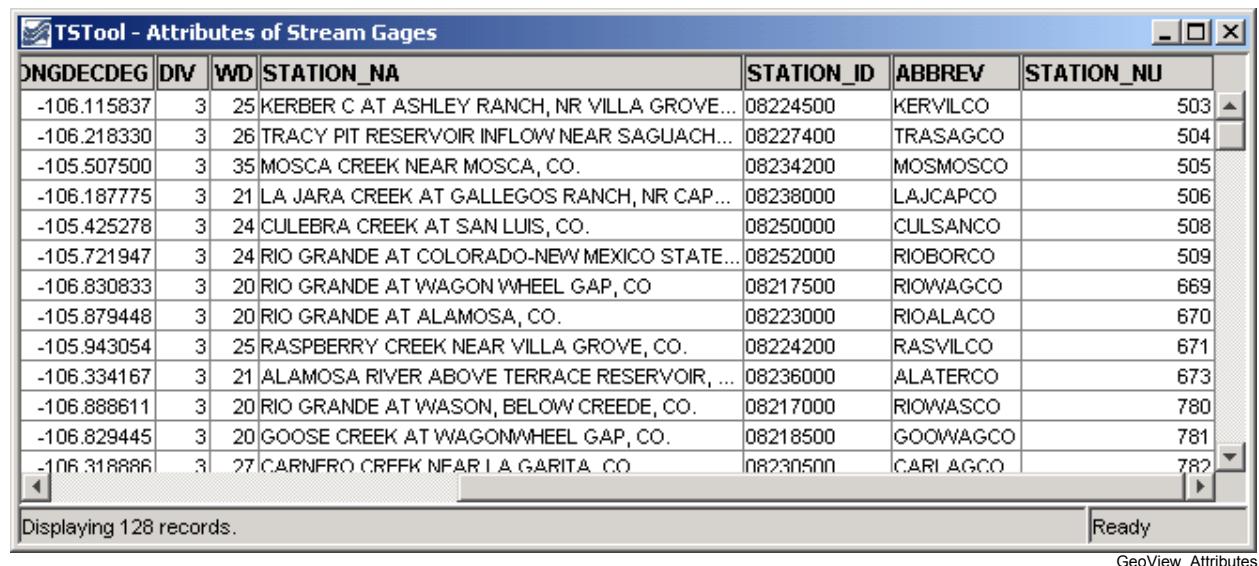
GeoView_Props_Animation

Layer View Animation Properties

Layer view animation properties are currently under development. Animation properties will define, for example, the time series data that are used for symbolization during animation.

Viewing a Layer's Attributes

Each feature in a layer includes geographic shape information (e.g., the coordinates that define a polygon). Each feature also can have attribute data, which are typically represented in a tabular fashion. To view the attributes for a layer, first select the layer in the table of contents, then right-click and press the **View Attribute Table** menu choice. A window similar to the following will be shown:



The screenshot shows a Windows application window titled "TSTool - Attributes of Stream Gages". The window contains a table with the following columns: DNGDECDEG, DIV, WD, STATION_NA, STATION_ID, ABBREV, and STATION_NU. The table lists 128 records of stream gage information. The last record listed is "106.318886 3 27 CARNFRO CRFFK NFAR LA GARITA CO". At the bottom of the window, it says "Displaying 128 records." and "Ready". The status bar at the bottom right says "GeoView_Attributes".

DNGDECDEG	DIV	WD	STATION_NA	STATION_ID	ABBREV	STATION_NU
-106.115837	3	25	KERBER C AT ASHLEY RANCH, NR VILLA GROVE...	08224500	KERVILCO	503
-106.218330	3	26	TRACY PIT RESERVOIR INFLOW NEAR SAGUACH...	08227400	TRASAGCO	504
-105.507500	3	35	MOSCA CREEK NEAR MOSCA, CO.	08234200	MOSMOSCO	505
-106.187775	3	21	LA JARA CREEK AT GALLEGOS RANCH, NR CAP...	08238000	LAJCAPCO	506
-105.425278	3	24	CULEBRA CREEK AT SAN LUIS, CO.	08250000	CULSANCO	508
-105.721947	3	24	RIO GRANDE AT COLORADO-NEW MEXICO STATE...	08252000	RIOBORCO	509
-106.830833	3	20	RIO GRANDE AT WAGON WHEEL GAP, CO	08217500	RIOWAGCO	669
-105.879448	3	20	RIO GRANDE AT ALAMOSA, CO.	08223000	RIOALACO	670
-105.943054	3	25	RASPBERRY CREEK NEAR VILLA GROVE, CO.	08224200	RASVILCO	671
-106.334167	3	21	ALAMOSA RIVER ABOVE TERRACE RESERVOIR, ...	08236000	ALATERCO	673
-106.888611	3	20	RIO GRANDE AT WASON, BELOW CREED, CO.	08217000	RIOWASCO	780
-106.829445	3	20	GOOSE CREEK AT WAGONWHEEL GAP, CO.	08218500	GOOWAGCO	781
-106.318886	3	27	CARNFRO CRFFK NFAR LA GARITA CO	08230500	CARIAGCO	782

Attributes Table for a Layer

The attributes are displayed in the order and format determined from the input data. Attribute names in ESRI shapefiles are limited to ten characters. Information in the table can be selected (use **Ctrl-A** to select all) and can be copied to other software.

Using GeoView with a Software Application

Software developers integrate the GeoView components with software applications and typically the software user does not need to know how GeoView works with the application. However, this section describes a few important concepts that will help facilitate setting up data for use by an application.

Basic GeoView implementations involve defining a GeoView Project (see the **GeoView Configuration – the GeoView Project File** section below for details on project file properties) and then interacting with the GeoView interface when the map is displayed. In a basic application, a GeoView can be added to show maps for reference purposes only. For example, the application may be an interface to a database containing location information. If a GeoView project file is defined with only base layers, then the zooming and features will allow a user to zoom into a region of interest, but there will be no interaction between the GeoView and the application.

In a more advanced application, layers in the GeoView Project file are assigned an AppLayerType property, which is recognized by the application. For example, a layer may be assigned an application type of "Streamflow" to indicate a streamflow gage. Additionally, the AppJoinField property can be defined to allow the application to join its data to the geographic data. This assignment in and of itself causes no effect in the GeoView. However, the application can now interact with the GeoView by asking for the "Streamflow" layer. This allows features in the GeoView to be selected from the application (e.g., in a database query screen) and allows the GeoView to provide information about the layer to the

application. For this type of implementation, it is important that the application layer types, feature (shape) type, and the required join fields are documented; consequently, new data layers can be used with the application with only a few configuration changes.

Some applications may automatically update the map interface by zooming to selected areas, selecting features, etc. Standard GeoView features are typically still available, as previously described.

Limitations

The GeoView components have been developed not to serve as full-featured GIS components, but to support many common GIS activities like selection, zooming, and symbolization. The components have been developed to integrate with existing applications and use other tool sets, including time series viewing tools. Basic features have been implemented to address important needs for applications; however, additional features are implemented as requirements change. The GeoView components are not envisioned as a replacement for pure GIS tools like ESRI's ArcGIS products. In many cases, ESRI or other tools can be used to develop the data for use with GeoView.

Currently, properties that are changed interactively cannot be saved to the GeoView Project file.

GeoView software currently does not examine projection files optionally distributed with ESRI shapefiles. Projections must be defined in the project file (or, if omitted, the projection is assumed to be consistent for data layers). Only a few projections are recognized, as needed by specific GeoView software implementations.

GeoView Configuration – the GeoView Project File

A GeoView display is configured mainly by using a GeoView Project (*gvp*) file, which is either read at software startup or when selected by the user. The purpose of the file is to persistently store the configuration of a map display so that it can be loaded again without redefining the configuration. The file format is simple text properties and can be read by applications implemented in various technologies running in various environments. An example of the file is shown below.

```
# Main properties global to the GeoView
# The format is:
# [Prefix]
# Prop=value
[GeoView]
GeoDataHome = .

# Properties for each GeoLayerView (data source and
# symbols)...
[GeoLayerView 1]
GeoLayer = xxx.shp

[GeoLayerView 2]
GeoLayer = xxx.shp
```

The GeoLayerViews listed first in the project file are drawn first and are therefore behind other layers on the map. For all properties, the comma is used as an internal delimiter and the semi-colon is used as a second layer of separation, as appropriate. Most properties will default to appropriate values if not specified (see tables below). The most important properties, as shown in the example above, are the `GeoDataHome`, which defines where data can be found, and `GeoLayer`, which defines where the data file is for each layer. Recognized layer file formats are listed in the following table and are described further in separate appendices. Support for additional layer types can be added as necessary.

Layer Type	Description
ESRI shapefile	ESRI shapefiles are commonly used with ESRI software such as ArcView, ArcMap, and ArcExplorer. GeoView determines the type by looking for the <code>.shp</code> file extension and checking the internal file format. No projection is assumed but the <code>Projection</code> property for the <code>GeoView</code> and individual layers can be used to indicate the projection.
NWSRFS GeoData files	The National Weather Service River Forecast System (NWSRFS) uses ASCII and binary files defining various geographic layers. This format is detected by checking the file names, which are predefined for NWSRFS. The <code>Projection</code> property is defined as <code>Geographic</code> if ASCII data and <code>HRAP</code> if binary data.
NWS XMRG Radar Files	XMRG files are gridded radar files produced by the National Weather Service. GeoView treats these files as grid files. This format is detected by looking for an "xmrg" string in the filename. The <code>Projection</code> property is defined as <code>HRAP</code> .

The following main GeoView properties can be defined in the project file. Graphical user interfaces to allow interactive editing of all properties are being implemented.

Main GeoView Property	Description	Default Value
Color	Background color for map. See the discussion after the properties tables for a discussion of how to define colors.	White.
FontName	Name of font to use for GeoView components (e.g., buttons). This property currently can only be set internally with software.	System-specific.
FontSize	Size of font, in points, to use for GeoView components (e.g., buttons). This property currently can only be set internally with software.	System-specific.
FontStyle	Font style to use for GeoView components (e.g., buttons). This property currently can only be set internally with software.	System-specific.
GeoDataHome	Directory where the GIS data exist. This directory will be prepended to layer files if they are not absolute paths already.	If not specified or if specified as ".", the directory will be set as the home of the GeoView Project file.
InitialExtent	Initial extent of the map display, in data coordinates. The coordinates should be specified as "XMIN, YMIN XMAX, YMAX", where the first pair is the lower-left corner of the extents and the second pair is the upper right. This property has not been implemented. See the MaximumExtent property.	No default. The initial extent will be the maximum data extent.
MaximumExtent	Maximum extent of the map display when zoomed out, in data coordinates. The coordinates should be specified as "XMIN, YMIN XMAX, YMAX", where the first pair is the lower-left corner of the extents and the second pair is the upper right.	No default. The maximum extent will be the maximum data extent.
Projection	Projection for the GeoView. The projection definition varies depending on the projection (some projections require more parameters). The following projections are currently supported: Geographic - no projection (decimal degrees) HRAP - used by National Weather Service UTM, Zone[, Datum, FalseEasting [, FalseNorthing][, CentralLongitude [, OriginLatitude][, Scale]] - Universal Transverse Mercator. The Zone is required (e.g., 13 for Colorado). Datum defaults to NAD83. The FalseEasting defaults to 500000. The FalseNorthing defaults to 0. The CentralLongitude is computed from the Zone. The OriginLatitude defaults to zero. The Scale defaults to .9996.	No default. All data are assumed to be the same projection.
ProjectAtRead	Indicates whether layer features are projected at read-time to the GeoView projection. This slows down the application initially but increases performance later during map refreshes.	false (it is usually best to project all data to a common projection rather than relying on GeoView to do projections)
SelectColor	Color to use for selected features. See the discussion after this table for examples of how to specify colors.	Yellow A more unique magenta/pink color with RGB 255,120,255 is being considered.

The following GeoLayerView properties can be defined, corresponding to each data layer/file:

GeoLayer View Property	Description	Default Value
AppJoinField	Specify the field(s) that should be used by an application to join the layer data to application data. If multiple fields are necessary, separate the field names by commas (e.g., "wd, id").	None.
AppLayerType	Indicate a layer type to be handled by an application. For example, a layer may be tagged as "Streamflow". The application can then use this information to treat the layer differently (e.g., to know how to join the data to application data). Valid AppLayerType values must be defined and understood by the application.	None.
Color	Color for features when the SymbolClassification is SingleSymbol. If point data, this is the main color for the symbol. If line data, this is the line color. If polygon data, this is the fill color. See the discussion after this table for examples of how to specify colors.	Random.
ColorTable	<p>Used when the SymbolClassification property is ClassBreaks or UniqueValues and requires more than a single color. The number of colors should be one more than the number of class break values if SymbolClassBreaks is used and equal the number of class values if UniqueValues is used. Color tables can be defined in three ways:</p> <ol style="list-style-type: none"> 1. ColorTableName;NumColors Predefined tables include Gray, BlueToCyan, BlueToMagenta, BlueToRed, CyanToYellow, MagentaToCyan, MagentaToRed, YellowToMagenta, and YellowToRed. These named tables choose primary colors where necessary to provide clean color breaks. 2. Ramp;NumColors;Color1; Color2 3. Custom;NumColors;Color1; ...;ColorN <p>Only the first option is currently enabled. See the discussion after this table for examples of how to specify colors.</p>	Named color table using Gray.
EventLayerView	Indicates if the layer view is an event layer (ESRI Map Object notation). This property is not currently used in the Java tools.	false
GeoLayer	Name of file for the data layer, typically with the file extension. If an ESRI shapefile, specify the .shp file. If a relative path, the GeoView.GeoDataHome property will be prepended to the file name. This property is used to detect a break in the GeoLayerView numbering, indicating the end of layer views.	No default. Should always be specified.
IgnoreDataOutside	Indicate a range of values that should be considered. Currently this applies only to grid layer types. Specifying a range can be used, for example, to draw only cells with positive data values. The range should be specified as two numbers separated by a comma (e.g., .00001,100.0).	Not specified. All cells are considered.
LabelField	Specify one or more fields to be used for the label, separated by commas. If a LabelFormat property is specified, use it to format the label; otherwise, format each field according to the field specifications from the attribute data source.	No default. Specify one or more fields for the label.

GeoLayer View Property	Description	Default Value
LabelFontName	Font to use for labels. This property has not been enabled.	Helvetica
LabelFontSize	Label font height, points. This property has not been enabled.	10
LabelFormat	Specify a C-style format string to format the fields. The format specifications must agree with the data types being formatted. For example, if two floating-point fields are specified with the LabelField property, the corresponding format may be "%10.1f, %5.2f".	If not specified, the label will be formatted using the field width and precision determined from the data table, with values separated by commas.
LabelPosition	Label position. If point data, the position is relative to the point coordinates. If line or polygon data, the position is relative to the centroid coordinates. The position of the text will be offset to not overwrite a symbol and can be UpperRight, Right, LowerRight, Below, LowerLeft, Left, UpperLeft, or Above .	Right
Name	The layer view name that should be displayed in the legend.	No default. If not specified, the file name will be used in the legend.
OutlineColor	Outline color for point or polygon symbols. See the discussion after this table for examples of how to specify colors.	Default to the same as main color).
Projection	Projection for the layer's data. See the main GeoView Projection property for available values.	No default. It is assumed that all data in a project have a consistent projection.
ProjectAtRead	Indicates whether features are projected at read-time to the GeoView projection. This slows down the application initially but increases performance later during map refreshes. This property can be set once in the GeoView main properties.	false (it is usually best to project all data to a common projection rather than relying on GeoView to do projections)
ReadAttributes	Indicates whether attributes should be read when the data are read. If possible, based on the layer data format, attributes will be read on the fly as needed. Reading the attributes (true) takes more memory but will result in faster performance.	false
ReferenceLayer	Indicates whether the layer should be drawn in the reference GeoView. Indicate as true or false. Typically only the most general boundary information should be used in the reference layer.	false
SelectColor	Specify the color to be used when drawing selected features. This property is useful if the default select color does not easily viewed.	Use the GeoView .SelectColor property.
SkipLayerView	Can be set to true to skip the layer altogether when reading the project file (useful for commenting out layers during development). If this property is used, the number sequence for the layer views can be kept the same.	false (the layer view will be displayed)
SymbolClassBreaks	Class breaks that correspond to the ClassBreaks SymbolClassification property. The number of values should be one less than the number of values in the ColorTable property for the classification.	No default, although an application may suggest values.
SymbolClassField	Attribute data field that is used when the SymbolClassification property is ClassBreaks or UniqueValues.	No default, although an application may suggest a value based on the available attributes.

GeoLayer View Property	Description	Default Value
SymbolClassification	Indicates how data are to be classified when displaying the shape symbols. Values can be SingleSymbol (e.g., single point symbol, line style, or polygon fill color), UniqueValues (display a unique symbol style for each value, currently not implemented), or ClassBreaks (display a unique symbol style for groupings of values - requires specification of the SymbolClassField and SymbolClassBreaks properties).	SingleSymbol
SymbolSize	For point data, specify the symbol size in pixels. For line data specify the line width in pixels. Not used for polygon data. This property may need to be expanded to properly handle printed output (might need to use points rather than pixels or allow the units of measure to be set in the property). This property is currently not enabled.	6 pixels for points, 1 pixel for lines.
SymbolStyle	Indicates the symbol style. If point symbols, the style is the symbol identifier (e.g., CircleFilled). If line data, the symbol style is the line style (currently only Solid is supported). If polygon data, the symbol style is the fill pattern (currently only FillSolid is supported). See below for a full discussion of symbol styles.	None for points, Solid for lines, FillSolid for polygons.

Color Specification

Colors are specified for a number of different properties, including the feature color and outline color. In order to allow flexibility in specifying colors, a number of formats are supported:

- Named color. Available colors are: None (transparent), Black, Blue, Cyan, DarkGray, Gray, Green, LightGray, Magenta, Orange, Pink, Red, White, Yellow
- Comma-separated Color Triplets as 0-255 (e.g., 255, 0, 0) or 0.0 -1.0 (e.g., 1.0, 0.0, 0.0).
- Hexadecimal: 0xRRGGBB (e.g., 0xFF0000 for red)

Color Tables

Color tables are simply a list of colors. Typically the symbol maintains a color table if the classification is other than SingleSymbol. The symbol will also keep track of unique values or class breaks and use this information to determine a color to display for a shape. A number of predefined color tables are supported but user-defined tables are supported in the property format.

Symbol Style - Point Data

Symbol styles for point data are the same as for time series viewing tools. The following styles are available:

- None
- Arrow-Down, Arrow-Left
- Asterisk
- Circle-Hollow, Circle-Filled
- Diamond-Hollow, Diamond-Filled
- Plus, Plus-Square
- Square-Hollow, Square-Filled

- Triangle-Down-Hollow, Triangle-Down-Filled, Triangle-Left-Hollow,
Triangle-Left-Filled, Triangle-Right-Hollow, Triangle-Right-Filled,
Triangle-Up-Hollow, Triangle-Up-Filled
- X-Cap, X-Diamond, X-Edge, X-Square

Classification

Classification is used to symbolize data. The following classifications are supported:

Classification Type	Description
SingleSymbol	This is the default for all layers if not specified. For point data, a single symbol is used, centered on the . For line data, a single line width and color is used. For polygon data, single fill and outline colors are used.
UniqueValues	The data values for the field specified with the <code>SymbolClassField</code> property is sorted and checked for unique values. Each value is then assigned a color in the color table.
ClassBreaks	The number of class breaks should be one less than number of colors in the color table for the symbol. Breaks are defined by using a groupings of features based on the values of the field specified with the <code>SymbolClassField</code> property: < first value >= first value < second value ... > last value

GeoView Project File Examples

This section provides several examples, extracted from GeoView Project files.

The following example illustrates how to configure base layers in a GeoView Project file:

```
# GeoView project file for Rio Grande basin.

# Main GeoView properties.

[GeoView]

# Main home for data
# If a directory is not specified, the directory will be determined when the
# GeoView project file is selected.
#GeoDataHome = "C:\cdss\statemod\data\rgtwday\gis"
# ArcView/ArcExplorer Default...
#SelectColor = Yellow
# Arc 8...
#SelectColor = Cyan
# All-purpose (magenta/pink)
SelectColor = "255,120,255"
MaximumExtent = "266400,4090475 503060,4260700"

# Now list the layer views. A layer view consists of specifying a data layer
# (e.g., shapefile) and view information (e.g., symbol). This is equivalent to
# the ESRI "theme" concept. The layers specified first are drawn on the bottom.
# Start with number 1 and increase the layer number sequentially as layers are
# added on top.

[GeoLayerView 1]
GeoLayer = div3_districts.shp
Name = "Water Districts"
# RGB 153 204 50 - green-yellow
#Color = "0x99CC32"
# tan
Color = "255,240,190"
OutlineColor = black
ReferenceLayer = true
AppLayerType = "BaseLayer"

[GeoLayerView 2]
GeoLayer = div3_lakes.shp
#GeoLayer = div3_lakes.shp
Name = "Lakes"
# - blue
Color = "165,250,254"
OutlineColor = "0,130,254"
AppLayerType = "BaseLayer"

[GeoLayerView 3]
Name = "Rivers"
GeoLayer = div3_rivers.shp
#GeoLayer = div3_rivers.shp
# RGB - blue
Color = "0,188,253"
AppLayerType = "BaseLayer"

[GeoLayerView 4]
GeoLayer = div3_highways.shp
Name = "Roads and Highways"
Color = "255,0,0"
AppLayerType = "BaseLayer"

[GeoLayerView 5]
GeoLayer = div3_cities.shp
Name = "Cities and Towns"
SymbolStyle = "Square-Filled"
SymbolSize = 6
Color = "red"
LabelField = "Name"
LabelPosition = RightCenter
AppLayerType = "BaseLayer"
```

The following example illustrates how to display point data layers. These properties should be inserted at the appropriate location in a GeoView Project file.

```
[GeoLayerView 15]
#SkipLayerView = true
GeoLayer = div3_flowstations_2001-10-24.shp
Name = "Stream Gages"
# orange
Color = "254,167,0"
SymbolStyle = "Circle-Filled"
SymbolSize = 6
AppLayerType = "Streamflow"
AppJoinField = "STATION_ID"
#LabelField = "STATION_NA, STATION_NA"
#LabelFormat = "%s, %s"

[GeoLayerView 18]
#SkipLayerView = true
GeoLayer = div3_reservoirs_2001-10-24.shp
Name = "Reservoirs (WDID)"
# black
Color = "black"
SymbolStyle = "Triangle-Up-Filled"
SymbolSize = 6
AppLayerType = "Reservoir"
AppJoinField = "ID_LABEL_6"
```

Appendix

Spatial Data Format – ESRI Shapefile

2004-05-24, Acrobat Distiller

Overview

ESRI Shapefiles are a relatively simple format for spatial data, consisting of a file containing shape information (*.shp*), a file containing attribute data (*.dbf*), and an index file (*.shx*). The GeoView package currently supports the following shapefile shape types:

- Point (shape type 1)
- Multi-point (shape type 8)
- Arc/Line (shape type 3)
- Polygon type (shape type 5)
- Null shape (shape type 0)

Support for additional shape types may be added in the future, consistent with the shapefile specifications.

This page is intentionally blank.

This page, when printed, can be used for a spine in a binder.

Colorado's Decision Support Systems (CDSS)

TSTool - Time Series Tool

