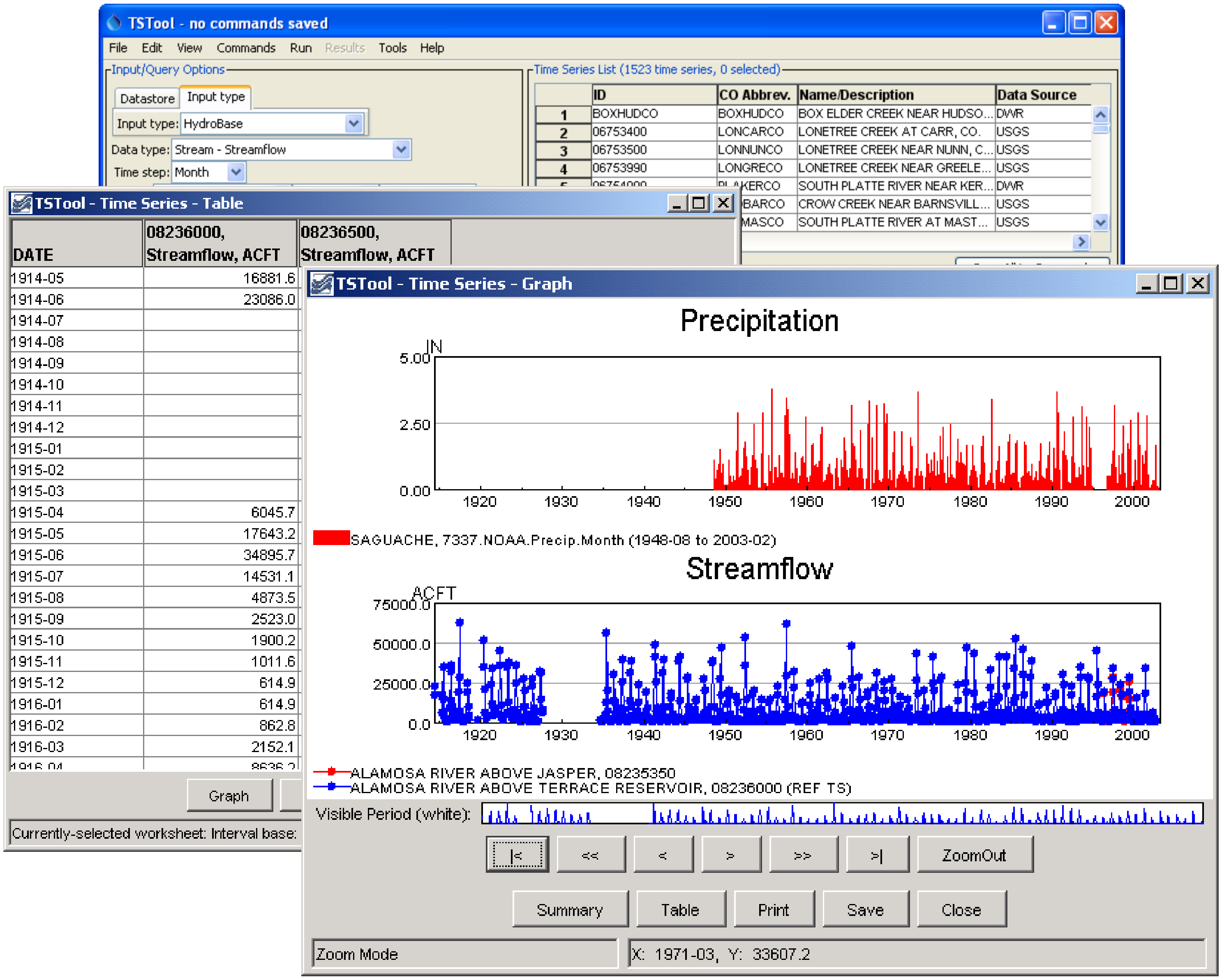
TSTool – Time Series Tool User Manual


TSTool - no commands saved
File Edit View Commands Run Results Tools Help
Input/Query Options
Datastore
Input type
Time Series List (1523 time series, 0 selected)
TSTool - Time Series - Table
TSTool - Time Series - Graph
Precipitation
IN
Streamflow
ACFT
Visible Period (white):
|< << < > >> >| ZoomOut
Summary Table Print Save Close
Graph

Input type: HydroBase
Data type: Stream - Streamflow
Time step: Month

	ID	CO Abbrev.	Name/Description	Data Source
1	BOXHUDCO	BOXHUDCO	BOX ELDER CREEK NEAR HUDSO...	DWR
2	06753400	LONCARCO	LONETREE CREEK AT CARR, CO.	USGS
3	06753500	LONNUNCO	LONETREE CREEK NEAR NUNN, C...	USGS
4	06753990	LONGRECO	LONETREE CREEK NEAR GREELE...	USGS
		PLAKERCO	SOUTH PLATTE RIVER NEAR KER...	DWR
		BARCO	CROW CREEK NEAR BARNSVILL...	USGS
		MASCO	SOUTH PLATTE RIVER AT MAST...	USGS

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	

Currently-selected worksheet: Interval base:

🟥 SAGUACHE, 7337.NOAA.Precip.Month (1948-08 to 2003-02)

ALAMOSA RIVER ABOVE JASPER, 08235350
ALAMOSA RIVER ABOVE TERRACE RESERVOIR, 08236000 (REF TS)

Zoom Mode
X: 1971-03, Y: 33607.2

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

Developed by:
RIVERSIDE
global science solutions

Version 10.21.00, 2013-07-14

This page is intentionally blank.

This document is formatted for double-sided printing.

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

DISCLAIMER for CDSS Products

2002-02-16

CDSS products include data and software from State of Colorado sources and from external sources like the U. S. Geological Survey (USGS). The following disclaimer applies to CDSS products:

CDSS products and associated access are under development at this time. Access is provided solely to test and demonstrate CDSS capabilities. In the future, this access may be restricted or offered for a fee. The State assumes no legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed herein. It is the user's responsibility to determine the fitness of the data for a particular purpose.

This page is intentionally blank

1 Acknowledgements

Version 10.00.01, 2011-05-16

TSTool has been developed by Riverside Technology, inc. (Riverside). Significant funding has been provided by the State of Colorado, Colorado Water Conservation Board (CWCB), as part of Colorado's Decision Support Systems (CDSS).

Enhancements to the CDSS version have also been made by Riverside in order to support a wider variety of uses and data formats.

Users are encouraged to provide general feedback to Riverside 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. Riverside'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.

2 Introduction

Version 10.21.00, 2013-06-16

TSTool can be thought of as a time series calculator. TSTool reads, displays, manipulates, analyzes, and writes time series data, either interactively or in batch (automated) mode. The TSTool graphical user interface (GUI) provides access to viewing and analysis features, command editors, and provides error feedback. Time series can be read from and written to a variety of file and database formats. Although a graphical user interface is provided, the heart of TSTool's analytical features is a command workflow processor. Depending on the task being performed, the command language may be used extensively or not at all. This flexibility makes TSTool useful for basic data viewing and advanced analysis. The documentation is divided into the following volumes:

Volume 1 – User Manual (for overall information about understanding and using TSTool)
Volume 2 – Command Reference (detailed documentation for every command)
Volume 3 – Datastore Reference (detailed documentation for every datastore, including file formats, databases, and web services)

This **User Manual** documentation is divided into the following chapters:

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.

Chapter 9 – Quality Control provides guidelines and examples for using TSTool to quality control data processing.

Chapter 10 – Spreadsheet Integration provides information about using TSTool with spreadsheet software.

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

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

The **TSView Time Series Viewing Tools** appendix provides a general reference for time series viewing features. These features are used throughout TSTool.

The **GeoView Mapping Tools** appendix provides a general reference for the GeoView map interface. The mapping interface is being phased in.

This documentation can be printed double-sided and is best viewed as PDF to use the navigable table of contents and bookmarks.

2.1 TSTool Data Flow Concepts

Although TSTool can be used simply to browse and view data, the main function of TSTool is to automate data processing using a workflow of commands. Figure 1 illustrates overall data flow concepts.

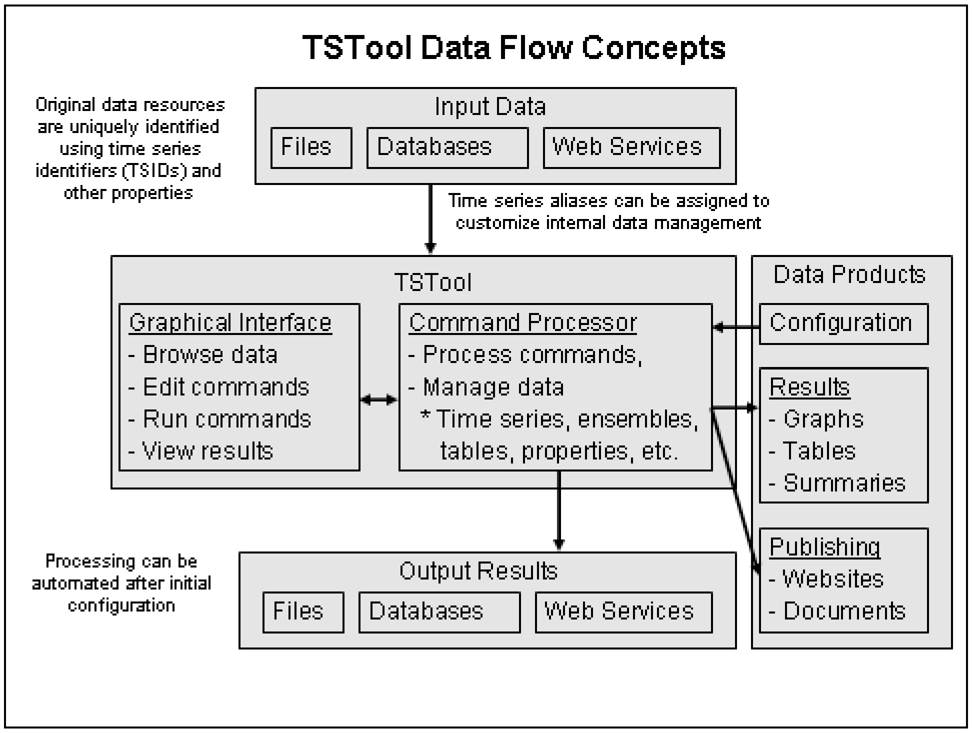

TSTool Data Flow Concepts
Original data resources are uniquely identified using time series identifiers (TSIDs) and other properties
Input Data
Files
Databases
Web Services
Time series aliases can be assigned to customize internal data management
TSTool
Graphical Interface
- Browse data
- Edit commands
- Run commands
- View results
Command Processor
- Process commands,
- Manage data
* Time series, ensembles, tables, properties, etc.
Data Products
Configuration
Results
- Graphs
- Tables
- Summaries
Publishing
- Websites
- Documents
Processing can be automated after initial configuration
Output Results
Files
Databases
Web Services

Figure 1 – TSTool Data Flow Concepts

Input data are retrieved using unique identifiers, such as time series identifiers discussed in the next section. The TSTool graphical user interface (GUI) is used to browse data, edit commands, run commands (by calling the command processor), and view results. The command processor internally manages data including lists of time series, time series ensembles, tables, and processor properties, and provides access to the data in interactions with the GUI. The command processor also can output results to files, databases, web services, and can create data products. Once configured, processing can be automated.

The interpretation of input data and definition of workflow and data products depends heavily on standards and conventions, some of which are defined by data providers, and some of which are defined by the TSTool design. Although standards define specific features, TSTool allows users a great deal of flexibility in defining workflows and producing data products.

2.2 Time Series Objects and Identifiers

TSTool handles time series as objects that are read (or internally created), manipulated, viewed, and output. Time series data include properties (also referred to as attributes or metadata) and a series of date/time and data value pairs (and optionally value data flags). Data generally consist of floating point values; however, time series may contain other data such as strings or images. TSTool primarily focuses on numerical time series. Time series are defined either as 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 and processing is less complicated.

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 properties 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 `NaN` [Not a Number])
- Comments (often station comments, if available)
- Processing history (a list of comments about how the time series was created and manipulated)

To identify time series in the original data and manage time series internally, TSTool assigns each time series a time series identifier (TSID) that uses the notation:

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

LocationType:Location.Source.Type.Interval.Scenario[Seq]~DataStoreName
```

The TSID can be thought of as a unique resource identifier, similar to a URL for web resources. The first five parts (`Location.Source.Type.Interval.Scenario`) are used to identify time series within the original data (e.g., to find the time series in a file or database):

- `LocationType` – optionally used where necessary to uniquely identify locations (e.g., use a location type of `Basin` or `Subbasin` where other identifier information would result in ambiguous interpretation of the identifier parts
- `Location` – typically a physical location identifier, such as a station, basin, or sensor identifier.
- `Source` – a data provider identifier, usually a government or system identifier (e.g., `USGS`, `NWS`), necessary because sometimes the provider for a location changes over time or a database may contain time series from multiple data providers
- `Type` – the data type, typically specific to the data (e.g., `Streamflow`, `Precip`) – TSTool does not try to institute global data type definitions).
- `Interval` – the data interval, indicating the time between data values (e.g., `6Hour`, `Day`, `Irregular`).
- `Scenario` – an optional item that indicates a scenario (e.g., `Hist`, `Filled`, `Max`, `Critical`).

- `Seq` – an optional item 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, collectively known as ensembles). Typically the sequence number is a four-digit year corresponding to the data input year.

The last part of the TSID (`~InputType~InputName` or `~DataStoreName`) indicates input information, which allows TSTool to locate and read the time series from a file, database, or the internet. The input information was introduced starting with TSTool version 5.04.00. The datastore convention was introduced in TSTool version 9.08.00 and will be phased in as the software is enhanced. The datastore design allows any name to be used to define a datastore, which allows more flexibility in defining data connections. The details about datastore configuration are defined in a simple properties file.

The following table lists datastore types that are supported or are under development. See the datastore appendices for information about how time series identifiers are formatted for specific datastores. TSTool features for a specific datastore may only be available if at least one datastore for a specific type is configured and enabled. Datastores are available for databases and web services because the "connection" information can be configured once and used throughout the TSTool session. User-assigned datastore names are used in TSTool commands to ensure that command files are transportable and datastore configurations can be updated without requiring commands to be updated.

Datastores Supported by TSTool

Datastore Type	Category	Description
ColoradoWaterHBGuest	Web service	State of Colorado's historical data.
ColoradoWaterSMS	Web service	State of Colorado's real-time data.
Generic Database	Database	A generic datastore to work with the `ReadTableFromDataStore()`, `WriteTableToDataStore()`, `ReadTimeSeriesFromDataStore()`, and `WriteTimeSeriesToDataStore()` commands, which can be configured to use an Open Database Connectivity (ODBC) Data Source Name (DSN).
HydroBase	Database	State of Colorado database (see also legacy and default treatment as an input type in following table). This is not available by default due to the legacy practice of allowing HydroBase to be selected at TSTool startup.
RCC-ACIS	Web service	Regional Climate Center Applied Climate Information System.
ReclamationHDB	Database	United States Bureau of Reclamation HDB database.
RiversideDB	Database	Riverside Technology, inc. database used for real-time and historical time series data as part of RiverTrak® System software.
USGS NWIS Daily	Web service	United States Geological Survey (USGS) National Water Information System (NWIS) daily values.
USGS NWIS Groundwater	Web service	USGS NWIS groundwater values.
USGS NWIS Instantaneous	Web service	USGS NWIS instantaneous (real-time) values.

Datastore Type	Category	Description
WaterOneFlow	Web service	WaterOneFlow web services (uses WaterML format for time series) – under development.

The following table lists input types that are supported or are under development. The legacy "input type" terminology continues to be used but in the future may be replaced with "file datastore" or similar. See the input type appendices for information about how time series identifiers are formatted for specific input types. TSTool features for a specific input type may only be available if the input type is enabled in the TSTool configuration file. Input types may utilize an "input name", for example when a time series identifier needs to include the input type and the name of a file being read. Reading time series from a single file using this approach makes sense because there is no need to configure a datastore with a configuration file for every data file. Most legacy input types that could be migrated to datastores have been migrated.

Input Types Supported by TSTool

Input Type	Category	Description
DateValue	Single file	General delimited date/value file with extended header information, able to store one or more time series.
Delimited	Single file	Generic column-delimited format (see the `ReadDelimitedFile()` command).
DIADvisor	Database	DIADvisor real-time environmental monitoring software, from OneRain, Inc. Earlier versions of TSTool provided DIADvisor integration; however, this capability has not been actively maintained.
HEC-DSS	Binary file database	Army Corp of Engineers binary time series database file used with Hydrologic Engineering Center (HEC) software.
HydroBase	Database	State of Colorado database. HydroBase also can be accessed as a datastore (see previous table and HydroBase datastore appendix). The input type convention is being retained in the short term until a decision can be made about how to configure HydroBase datastores consistently between multiple users and projects.
MexicoCSMN	File group database	Hydrometeorological database for Mexico Coordinación Servicio Meteorologico Nacional (CSMN, similar to US National Weather Service). This input type was implemented years ago and may no longer be of value.
MODSIM	Single file	Colorado State University MODSIM model, version 7. MODSIM version 8 uses a database but full support for this database has not been added (see Generic Database datastore as one option).
NWSCard	Single file	National Weather Service River Forecast System (NWSRFS) card file format for hourly data.
NWSRFS_ESPTraceEnsemble	Binary file	NWSRFS Ensemble Streamflow Prediction binary file.

Input Type	Category	Description
NWSRFS_FS5Files	Binary file database	NWSRFS binary FS5Files preprocessor and processed database.
RiverWare	Single file	University of Colorado Center for Advanced Decision Support for Water and Environmental Systems (CADSWES) RiverWare model data format.
SHEF	Single file	Standard Hydrologic Exchange Format, a common data format used by United States government agencies. Only limited support is provided.
StateCU	Single file	State of Colorado consumptive use model time series input and output file formats.
StateCUB	Binary file database	State of Colorado consumptive use model binary output file.
StateMod	Single file	State of Colorado StateMod model time series input and output file formats.
StateModB	Binary file database	State of Colorado StateMod model output binary file.
USGS NWIS Rdb	Single file	USGS NWIS RDB format files.
WaterML	Single file	WaterML format file.

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 daily streamflow file might be identified using:

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

In these examples, the optional scenario (fifth part) and sequence number are not used. TSID strings can be saved in a command file or time series product description file to indicate the time series to process. The TSID with input information allows TSTool to determine how to access the time series and is useful for managing time series. The TSTool GUI typically handles creation of all time series identifiers; however, command files can be edited with a text editor. The path to files can be absolute or relative to the command file. The latter is recommended to improve portability of files between computers.

TSTool only shows the input type and input name parts of the identifier when editing 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 (see the next section).

2.3 Using Time Series Aliases

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:

```
ReadTimeSeries(Alias=”09010500”,TSID=”09010500.USGS.Streamflow.Month~HydroBase”)
```

The following older syntax was phased out in TSTool version 10.00.00, and is automatically converted to the above syntax when a command file is read:

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

If an alias is defined for a time series, the alias will be used instead of the alias and will be displayed in command editors and results (although the TSID is often also shown in output). TSTool ignores upper/lower case when comparing identifiers, aliases, and other commands, although it is good practice to be consistent. The general term "TSID", when used as a command parameter to identify time series to process, allows a TSID or alias to be specified.

TSTool commands allow the alias to be defined using time series properties. For example, the parameter `Alias="%L"` will assign the alias as the location part of the time series identifier. A combination of the special formatting strings and literal characters can be used to create unique aliases for time series. It is recommended that aliases not contain spaces or other whitespace and that periods be avoided because they are used in TSIDs.

When defining a series of commands as a workflow to perform a task, a user generally will develop a rough draft of the process by using TSTool's interactive data browsing and command editing features. However, it is useful to develop a data flow description more identifiers that are more appropriate than the default TSIDs provided by TSTool. Consequently, at an appropriate time in the workflow definition, it is useful to take a step back and review the identifiers being used, and insert aliases where possible. For example, an inventory of time series may be defined to describe time series used in a process, similar to the following (where the bold indicates information that will be replaced with a specific value):

Alias	Description
Location-Streamflow-Month-Original	Original monthly streamflow.
Location-Streamflow-Month-QC	Monthly streamflow data after quality control.
Location-Streamflow-Month	Monthly streamflow data after filling data gaps, ready for analysis.
Location-Streamflow-Year	Annual streamflow data, accumulated from **Location**-Streamflow-Month.

By using aliases, it is possible to switch the data inputs with minor changes to the processing logic. If it is important to indicate the source of information, such as when comparing the same data stored in two different databases, then the alias can be defined to include the input name. Regardless of whether an alias is used, the original TSID also is maintained with the time series and is available for reports and displays.

In summary, if an alias is assigned to a time series, it will take precedence over the TSID when identifying the time series.

2.4 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.
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`, `Hour`). Consequently `24Hour` data has a base interval of `Hour` and a multiplier of `24`.

3. The period of a time series is defined by start and end date/time values, using 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) or water year (October to November). Additionally, a year type of NovToOct has been implemented to represent November to October and additional similar year types may be implemented in the future. Calendar year is the default but can be changed in some commands.

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 but generally are not used (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 formats commonly used in the United States and other locales. 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. The input type and datastore appendices discuss date/time issues with various data 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 screen pixels or page coordinates. 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. Some commands will allow an automatic conversion of 24Hour interval data to Day interval data to avoid hour offset issues.

TSTool understands leap years and days per month. Consequently data formats that do not properly implement leap years or simplify time by assuming a constant number of days per month may result in missing values in data when read into TSTool.

TSTool does have the capability to handle time zone in small interval (hour, minute) data. However, fully representing time zone and daylight savings offsets is somewhat complex and TSTool for the most part does not perform time zone conversions or normalization.

Input formats that have different conventions are handled by converting the data to TSTool conventions when reading the data and converting from TSTool conventions when writing the data.

2.5 Time Scale for Time Series Data

The time scale for time series data gives an indication of how data values 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 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.

Input formats that have different conventions are handled by converting the data to TSTool conventions when reading the data and converting from TSTool conventions when writing the data.

2.6 Time Series Commands and Processing Sequence

Although TSTool can be run in batch mode (see **Chapter 3 – Getting Started**), it is possible 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.

TSTool commands fall into several categories:

1. Time series identifiers (see Section **2.2 – Time Series Objects and Identifiers**), which are equivalent to time series "read" commands (where the identifier input type or datastore is used to determine how to read from the original data format)
2. General commands, which are used to set properties like the period for output
3. Time series commands, which are used to read and manipulate time series and output results
4. Ensemble commands, which process ensembles of time series,
5. Table commands, which process tables of information.

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. Command syntax is 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:

- The parameter names are included in the command, which makes the command more readable as text.
- Because the parameter name is included, the parameters can be in any order. The command editor dialogs will enforce a default older.
- Parameters that have default values can be omitted from the parameter list, shortening commands.
- New parameters can be added over time, without requiring parameter order to change in existing commands.

Older commands used a fixed notation where parameters were required to be in a specific order. TSTool will attempt to update older command syntax to the current syntax when a command file is read. The **Command Reference** describes each command's function and parameters.

The following sequence occurs when processing commands:

1. **Check the command for run-time initialization errors** . Commands initially are parsed when a command file is opened or new commands are added in the GUI. When commands are run, additional checks are performed based on run-time data, such as the period that has been specified for the run. Example commands are shown below. If the command results in reading or creating a time series, step 3 is executed, as described below.

```
# Example commands
08235350.USGS.Streamflow.Month~HydroBase
08236000.DWR.Streamflow.Month~HydroBase
Add(TSID=”08235350.USGS.Streamflow.Month”,HandleMissingHow=IgnoreMissing,
  TSList=SpecifiedTSID,AddTSID=”08236000.DWR.Streamflow.Month”)
```

```
08235350.USGS.Streamflow.Month~HydroBase
```

2. **Read or create time series.** TSTool recognizes that certain commands should read or create 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. The input type in the identifier (information after the ~) is used to determine how to read the time series. By default, the entire time series will be read unless the `SetInputPeriod()` or `SetOutputPeriod()` commands have been specified.
   a. **Read data.** If the input type, and if needed, input name, are specified in the identifier, they are used to read the data. Time series properties such as units are assigned.
   b. **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.
   c. **Save time series in processor.** The time series that are read or created are managed by the processor, using the TSID and alias. The time series can then be manipulated by other commands, as per the following step.

3. **Manipulate time series or other data.** Commands that manipulate time series (fill, add, etc.) do not automatically read the time series or make another copy. Instead, time series that are in memory are located using the TSID or alias and are manipulated as per the command functionality. 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.DWR.Streamflow.Month~HydroBase
Add(TSID=”08235350.USGS.Streamflow.Month”,HandleMissingHow=IgnoreMissing,TSList
=SpecifiedTSID,AddTSID=”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 time series (those that have been read or created in previous commands) against the current time series of interest (`TSID=”08235350.USGS.Streamflow.Month”`), 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).

TSTool commands generally use the `TSList` parameter to locate time series in memory, where the default value depends on the command. For example, `TSList=AllTS` will process all time series. For commands that require a single input time series, TSTool often will default to `TSList=LastMatchingTSID`, which finds the first time series prior to the current command. It is possible to create more than one time series with the same identifier while allowing localized processing of each time series. For 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").

However, this may lead to confusion and generally should be avoided by using unique identifiers for each time series that is being processed.

During processing, extra time series can accumulate and will be available for output. Use the `Free()` command to free time series that are no longer needed. This removes the time series from memory. Output commands also may use the `TSList` parameter to indicate which time series are to be output.

4. 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 (also right click on time series listed in the results menu at the bottom of the main window). Time series also can 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 is not displayed. Processing the example shown above results in three time series in memory:

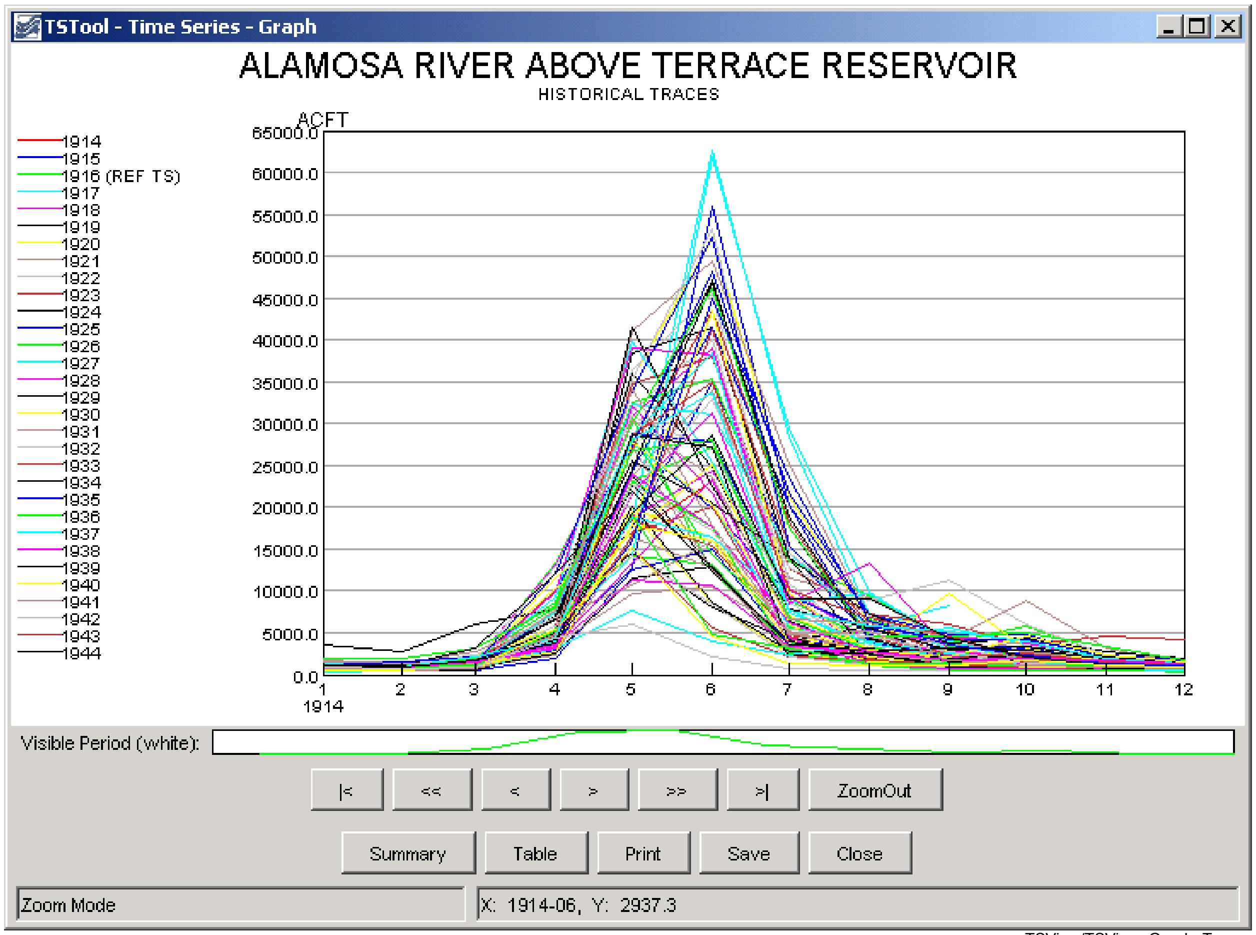
1. A time series identified by `08235350.USGS.Streamflow.Month`, containing the sum of the two time series.
2. A time series identified by `08236000.DWR.Streamflow.Month`, containing the input to the `Add()` command.
3. A time series **also** identified by `08235350.USGS.Streamflow.Month`, containing 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.

2.7 Time Series Ensembles

A time series ensemble is a group of related, typically overlapping, time series. Ensembles can be used to manage related scenarios (e.g., input and results of model scenarios) or as a way of shifting a historical time series so that years overlap. Many commands operate on a list of time series by using the parameters `TSList=EnsembleID` and `EnsembleID="SomeID"`. Statistics time series can be derived from ensembles, for example to calculate the average condition over time (although care must be taken in whether this can be interpreted as a time series of related values, for use as input to a process). Ensembles are assigned unique identifiers and are displayed at the bottom of the TSTool main window in the ***Ensembles*** results tab. Each time series in an ensemble is also available for individual processing and is listed in the ***Time Series*** results tab. Time series in ensembles often have a `[Seq]` sequence number in the TSID corresponding to the year of the data, as discussed in **Section 2.2**.

The following figure illustrates an ensemble of annual time series created from a long historical time series.



TSView/TSView_Graph_Traces

Example Trace Ensemble Plot Showing Historical Years

In general, ensembles provide a way to process a group of time series. Refer to the command reference for specific features and limitations related to ensemble processing.

2.8 Time Series Processor Properties

The time series processor performs data management tasks such as tracking the list of time series and providing requested time series to commands for processing (using the time series identifier and alias to match time series). The processor also has an understanding of standard global properties that serve as defaults for all commands, such as the input and output periods and working directory. In addition to global properties that are assigned by default, the `SetProperty()` command can be used to provide user-specified properties to the processor. For example, the user might want to define a property corresponding to a folder on the system for data files. The property then can be used in command parameters to insert the folder name. The syntax for using a property in a command parameter is:

```
Command(InputFile="${PropertyName}/SomeOtherInformation")
```

These processor properties should not be confused with time series properties, which are specific to each time series and are set with the `SetTimeSeriesProperty()` command. Command editors may be updated over time to provide drop-down choices of defined global properties; however, it may be necessary to enter the property name as text, in particular for file names. The following table lists standard global properties that can be used in command parameters that recognize the global properties (refer to specific command documentation to determine whether a command supports global properties). The internal representation of each property varies depending on the meaning of the property. For example, date/times are represented internally as DateTime objects. Regardless of internal representation, the value is converted to a string when used in string parameters such as filenames. Because some data types (such as date/times) can be formatted in a variety of ways, the default representations (such as the ISO 8601 YYYY-MM-DD representation for dates) may not be appropriate. The `FormatDateTimeProperty()` command can be used to format date/time string properties.

Command Processor Global Properties

Property Name	Type	Description
`CreateOutput`	Boolean	Indicates whether commands should create output files (true) or not (false) – this is useful for speeding processing during initial testing.
`InitialWorkingDir`	String	The initial working directory (folder), initially defaulted to the main command file folder. This property is used with the `RunCommands()` command to help the processor keep track of changing working directories.
`InputStart`	DateTime	The global input period start, for read commands.
`InputEnd`	DateTime	The global input period end, for read commands.
`OutputStart`	DateTime	The global output period start, for commands that create or write time series.
`OutputEnd`	DateTime	The global output period end, for commands that create or write time series.
`OutputYearType`	String	The output year type, for commands that create or write time series.
`WorkingDir`	String	The working directory (folder), initially defaulted to the command file folder. All other relative paths will be relative to this location. The `SetWorkingDir()` can be used to modify the value; however, this is not recommended because it can lead to hard-coding paths in command files, which limits portability.

3 Getting Started

Version 10.20.00, 2013-04-21

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

1. **Browse and view time series data**. In this capacity, a graph or summary can be created and then TSTool can be closed.
2. **Automate time series processing**. For example, format lists of time series for use with simulation models or other software. In this capacity, time series that are read and displayed can be incorporated into a command file, which can be run to generate time series files.
3. **Process time series products**. For example, create graphs for use on web sites or to facilitate review data or modeling results. In this capacity TSTool is used to generate data products in a streamlined fashion.

The remainder of this chapter provides an overview of the graphical user interface, in the general order of the main features and menus on the menu bar (left to right, top to bottom). The features necessary to accomplish the above tasks are described at an introductory level. See other chapters for more detailed information. See also the training materials that are available under the *doc\Training* folder of the software installation.

3.1 Starting TSTool

When using the State of Colorado's CDSS configuration for TSTool, the software can be started on Windows using ***Start…All Programs…CDSS…TSTool-Version*** (or ***Start…Programs…CDSS…TSTool-Version***). The menus vary slightly depending on the operating system.

TSTool also has been implemented for Linux and Mac OS X, in which the `tstool` script can be run to start the software.

To process a command file in batch mode without showing the user interface, use a command line similar to the following:

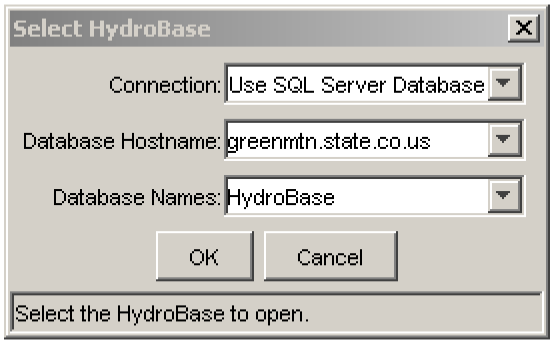
```
      tstool –commands commands.TSTool
```

It is customary to name command files with a *.TSTool* file extension. It may be necessary to specify a full (absolute) path to the command file when running in batch mode in order for TSTool to fully understand the working directory. See the **Batch Mode Execution** section at the end of this chapter for more information on running in batch mode.

3.2 Database Selection and User Authentication

Some data systems require authentication before a connection to the data can be established. A database selection also may be required. If database selection or authentication is required, TSTool may display a dialog at startup asking for information. **This convention is being phased out in favor of database datastores that are configured prior to runtime**.

For example, if the HydroBase input type is enabled (see the **HydroBase Input Type Appendix**), the HydroBase login dialog will be shown when TSTool starts in interactive mode. The dialog is used to select a server and database for the State of Colorado's HydroBase database. A HydroBase database can also be selected from the ***File...Open…HydroBase...*** menu.



Connection: Use SQL Server Database
Database Hostname: greenmtn.state.co.us
Database Names: HydroBase
Select the HydroBase to open.

Menu_Open_HydroBase

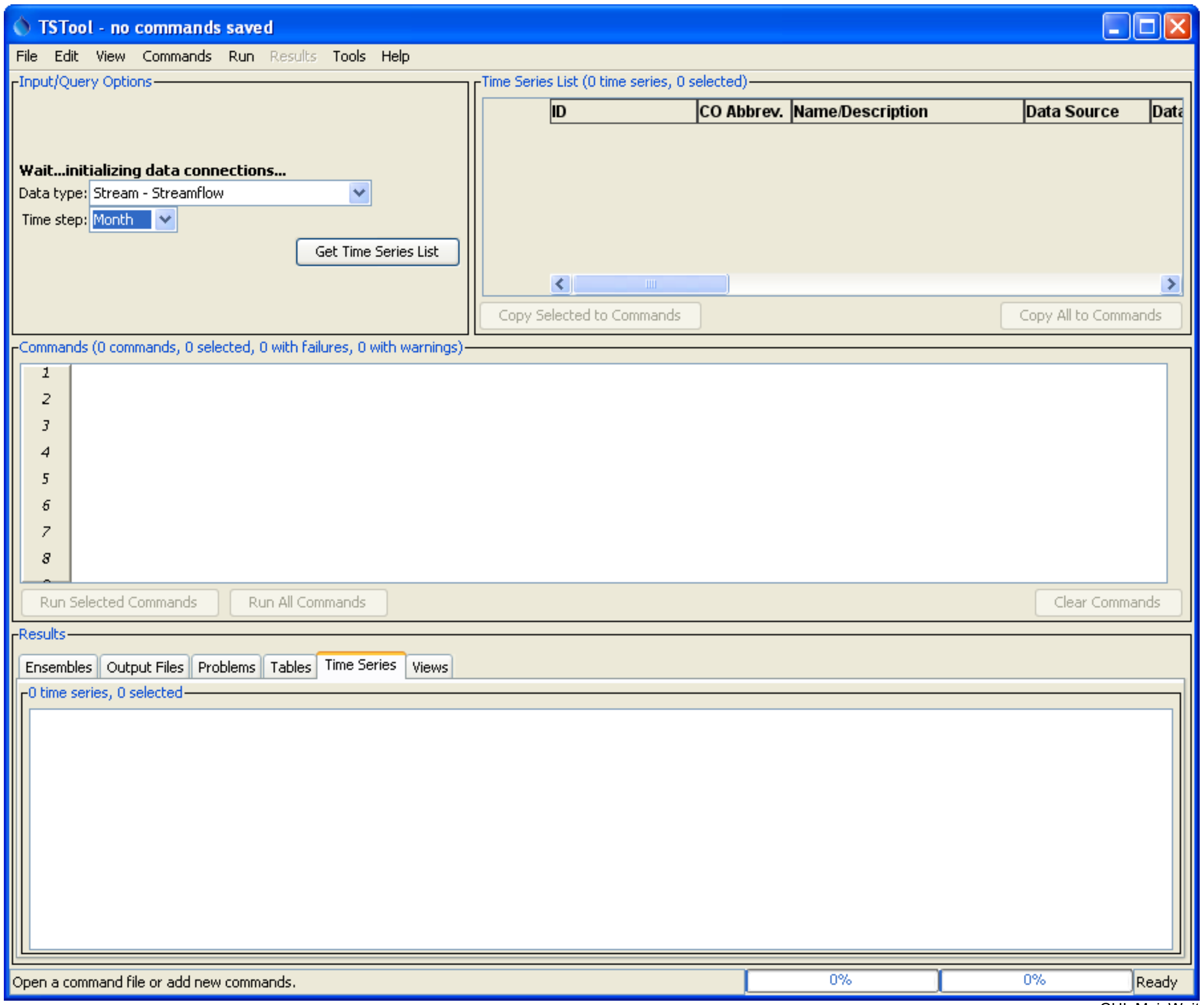
Select HydroBase Database Dialog

HydroBase features will be disabled if the HydroBase login is canceled.

Database connections typically are configured using configuration files so that the dialog confirmation can be avoided. See the **Installation and Configuration** appendix and information in the data store appendices. The ***Tools…Options*** menu allows input types to be enabled and disabled.

3.3 Main Interface

The following figure illustrates the main TSTool interface immediately after startup. A message is displayed in the upper left letting the user know that data connections are still being initialized for configured datastores. Trying to use commands that depend on data connections that have not been initialized will result in errors.


TSTool - no commands saved
File Edit View Commands Run Results Tools Help
Input/Query Options
Data type: Stream - Streamflow
Time step: Month
Get Time Series List
Time Series List (0 time series, 0 selected)
ID
CO Abbrev.
Name/Description
Data Source
Data
Copy Selected to Commands
Copy All to Commands
Commands (0 commands, 0 selected, 0 with failures, 0 with warnings)
Run Selected Commands
Run All Commands
Clear Commands
Results
Ensembles
Output Files
Problems
Tables
Time Series
Views
0 time series, 0 selected
0%
0%
Ready

Wait...initializing data connections...

Open a command file or add new commands.

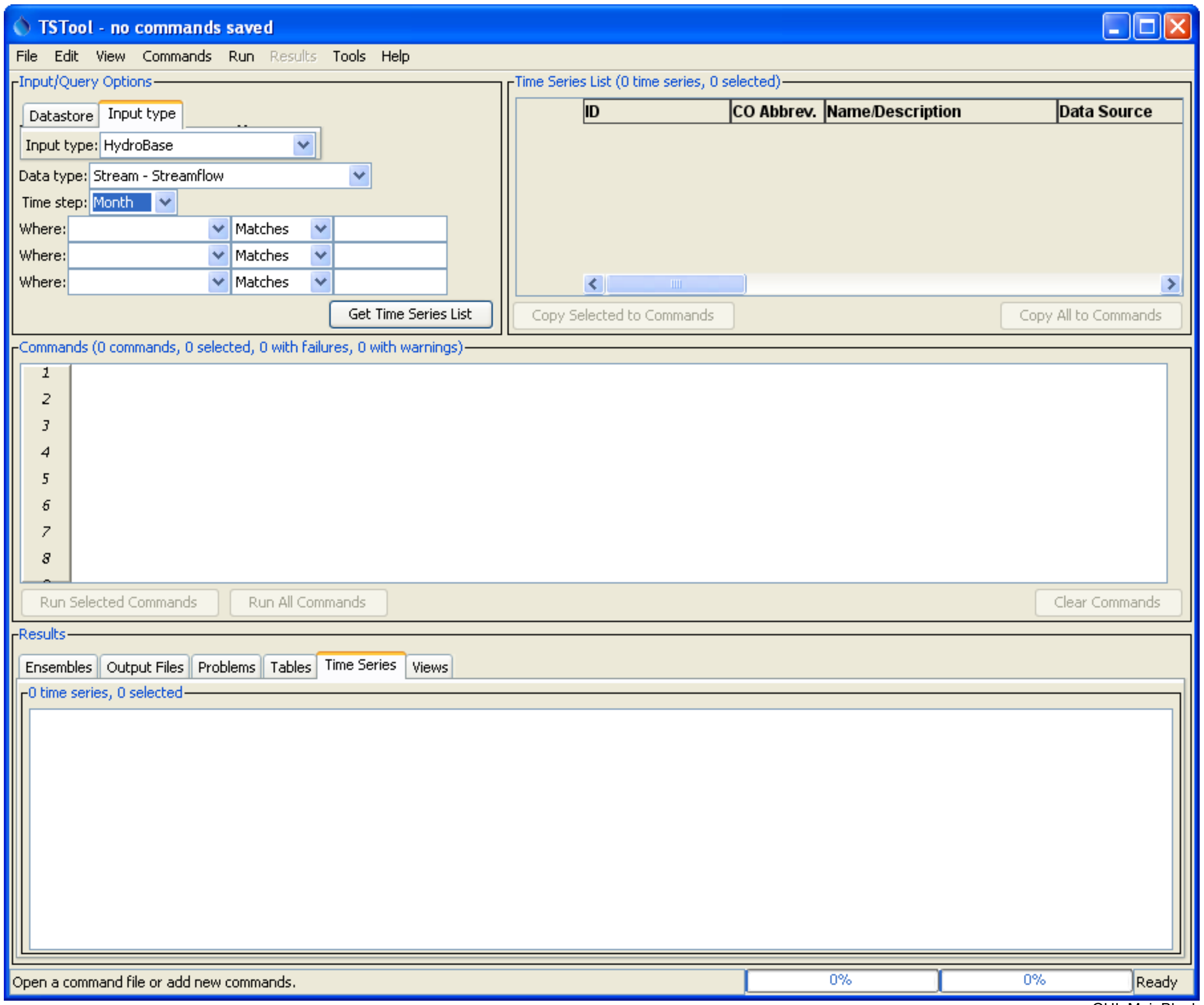
GUI_MainWait

TSTool At Startup, Waiting for Data Connections to be Initialized

The following figure illustrates the main TSTool interface after data connections are initialized. The interface is divided into three main areas:

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

Status and progress information is displayed at the bottom of the main window and also in the borders around main panels (e.g., to show how many items are in a list and how many are selected).



GUI_MainBlank

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 datastores (databases and web services) and input types (typically files and some databases). The interactive query interface is useful when selecting a time series from a file, database, or web service (internet). An alternative to the following interactive approach is to use read commands from the ***Commands*** menu (see the ***Commands*** chapter). To select time series, perform the following steps:

1. **Select the source of the data**. Select a ***Datastore*** or ***Input type***.
   - A data store is a repository that generally contains multiple time series (e.g., a database, web service, or file). The details about the datastore are included in a simple configuration file (see datastore appendices).
   - Input types define the storage format (e.g., database or file) for time series data, and may require an input name. Selecting some input types may prompt for a file, which is then listed in the ***Input name*** choices immediately below ***Input type***.

   The datastore design is being phased in because it provides more flexibility in defining data connections. In the future, all database connections will be transferred to datastores and remaining input types may be referred to as “file datastores”, which will use a filename.

   The DateValue input type is the default. More specific input types (e.g., HydroBase) may be the default if enabled. See the appropriate data store or input type appendix for more information.

   Depending on the input type, some of the remaining selection choices described below may be disabled or limited.

2. **Select the time series data type**. 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 data.

3. **Select the time series time step (interval)**. Select the ***Time step*** (if appropriate for the input type). The time step, also referred to as the data interval, generally will 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 input types where the time step is determined as data are read.

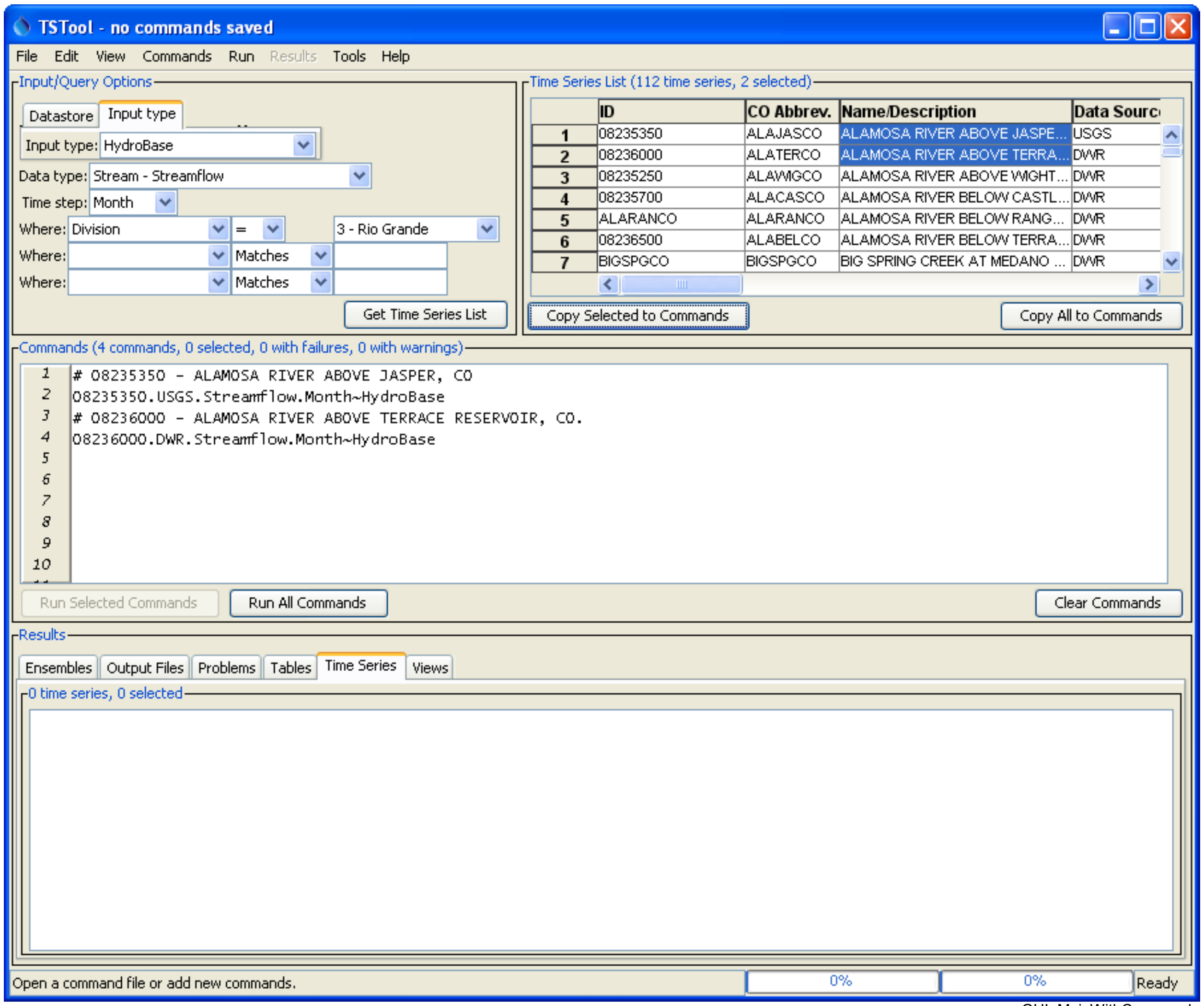
4. **Specify filter criteria for the time series list**. Specify the ***Where*** and ***Is*** clause(s) for the query (if appropriate for the input type). This information will limit the number of time series that are returned. The filters are highly dependent on the original data.

5. **Generate the time series list**. 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*** shows a list of matching time series, typically including location and time series properties. 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 sorts are alphabetical so some numeric fields may not sort as expected due to spaces, etc.

6. **Copy time series identifiers to the command list**. TSTool requires that time series identifiers (TSIDs) be created in the ***Commands*** list area in order to read the time series data values. To create TSIDs from the ***Time Series List***, selecting one or more rows in the ***Time Series List*** (note that the first column will not allow selections) and then press the ***Copy Selected to Commands*** button. Or, if appropriate, press the ***Copy All to Commands*** button.

7. **Read and display time series**. The ***Commands*** and ***Results*** areas are discussed below. To process time series having different data types or time steps, 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 selecting time series and copying to the ***Commands*** area, the main interface will appear similar to the following figure. As TSIDs are inserted, TSTool will attempt to read the time series properties to ensure that the TSID is correct, and an indicator will be shown for time series that could not be retrieved. This may result in a slight pause but helps ensure that commands are functional.


TSTool - no commands saved
File Edit View Commands Run Results Tools Help
Input/Query Options
Datastore
Input type
Get Time Series List
Copy Selected to Commands
Copy All to Commands
Run Selected Commands
Run All Commands
Clear Commands
Results
Ensembles Output Files Problems Tables Time Series Views
0%
0%
Ready

Input type: HydroBase
Data type: Stream - Streamflow
Time step: Month
Where: Division = 3 - Rio Grande
Where: Matches
Where: Matches

Time Series List (112 time series, 2 selected)

	ID	CO Abbrev.	Name/Description	Data Sourc
1	08235350	ALAJASCO	ALAMOSA RIVER ABOVE JASPE...	USGS
2	08236000	ALATERCO	ALAMOSA RIVER ABOVE TERRA...	DWR
3	08235250	ALAWIGCO	ALAMOSA RIVER ABOVE WIGHT...	DWR
4	08235700	ALACASCO	ALAMOSA RIVER BELOW CASTL...	DWR
5	ALARANCO	ALARANCO	ALAMOSA RIVER BELOW RANG...	DWR
6	08236500	ALABELCO	ALAMOSA RIVER BELOW TERRA...	DWR
7	BIGSPGCO	BIGSPGCO	BIG SPRING CREEK AT MEDANO ...	DWR

Commands (4 commands, 0 selected, 0 with failures, 0 with warnings)

```
# 08235350 - ALAMOSA RIVER ABOVE JASPER, CO
08235350.USGS.Streamflow.Month~HydroBase
# 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO.
08236000.DWR.Streamflow.Month~HydroBase
```

0 time series, 0 selected

Open a command file or add new commands.

GUI_MainWithCommands

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

3.3.2 Command List and Command Error Indicators

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***
- commands inserted using the ***Commands*** menu (see **Chapter 4 – Commands**).

Time series identifiers are added to the ***Commands*** list by selecting items in the ***Time Series List*** and copying the identifiers to the ***Commands*** list, as discussed above. 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 more control is needed during the data read or when processing more than one time series with one command.

The ***Commands*** and other TSTool GUI lists behave according to standard conventions for the operating system. For example, on Windows:

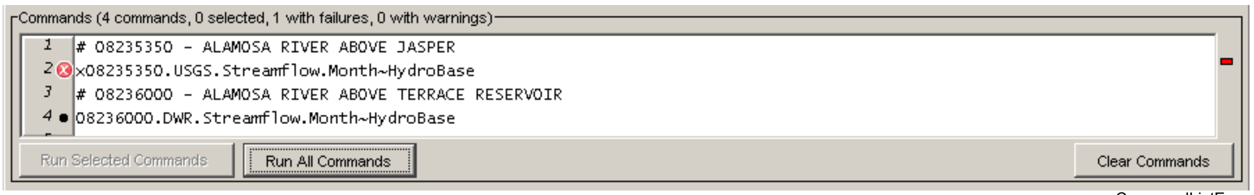
- ***Single-click*** to select one item.
- ***Ctrl-click*** to additionally select an item.
- ***Shift-click*** to select everything between the previous selection and the current selection.

Behavior on operating systems other than Windows may vary. 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. A summary of the pop-up menu choices is as follows:

Menu Choice	Description
Show Command Status Success/Warning/Failure	Displays the status of a command for each phase of command processing (more discussion below after table).
Edit	Edit the selected command using custom edit dialogs, which provide error checks and format commands. Double-clicking on a command also results in editing the command.
Cut	Cut the selected commands for pasting.
Copy	Copy the selected commands for pasting.
Paste (After Selected)	Paste commands that have been cut/copied, pasted after the selected row.
Delete Command(s)	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 Commands	Select all the commands.
Deselect All Commands	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.
Convert TSID command to general	Convert the selected TSID command to a `ReadTimeSeries()` command. This general command allows an alias to be assigned to the

Menu Choice	Description
ReadTimeSeries() command	time series.
Convert TSID command to specific Read…() command	Convert the selected TSID command to a specific `Read…()` command. The TSID is examined to determine a suitable read command. Specific read commands may provide parameters to control reading the time series, and may also allow multiple time series to be read.
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.

Commands are numbered to simplify editing. The command list also includes left and right gutters to display graphics that help with error handling. The following figure illustrates a command with an error (the first time series identifier has been edited to include an `x`, resulting in an invalid identifier).


Commands (4 commands, 0 selected, 1 with failures, 0 with warnings)

```
1   # 08235350 - ALAMOSA RIVER ABOVE JASPER
2 ⊗ x08235350.USGS.Streamflow.Month~HydroBase
3   # 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR
4 ● 08236000.DWR.Streamflow.Month~HydroBase
```

CommandListError

Command List Illustrating Error

The following error handling features are available:

- The graphic in the left gutter indicates the severity of a problem (see below for full explanation).
- The colored indicator on the right indicates the severity of a problem by its color and, when clicked on, positions the visible list of commands to display the command corresponding to the problem.
- Commands have three phases: 1) initialization, 2) discovery, 3) run. Initialization occurs when reading a command file or adding a new command. The discover phase is executed only for commands that generate time series for other commands and provides other commands with identifiers used in command editing. The run phase generates full output.
- Positioning the mouse over a graphic in the left or right gutter will show a popup message with the problem information. The popup is only visible for a few seconds so use the right-click popup menu ***Show Command Status (Success/Warning/Failure)*** for a dialog that does not automatically disappear.
- Clicking on the left gutter will hide and un-hide the gutter.

The meaning of the error handling symbols is described in the following table. The symbol for the most severe error will be displayed next to each command.

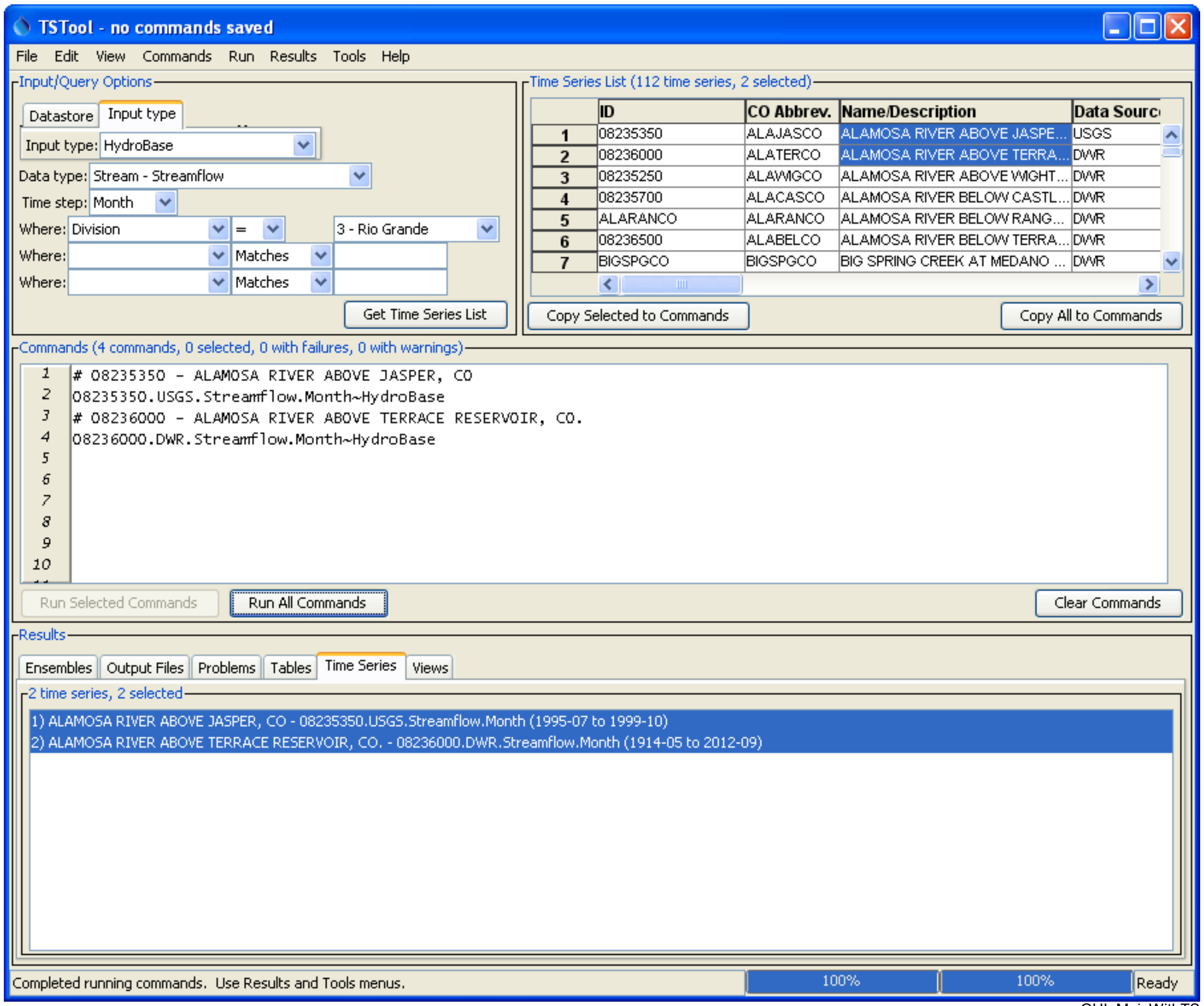
Command List Graphics for Problems

Problem Graphic	Description
No graphic	Command is successful (a warning or failure has not been detected).
●	The status is unknown, typically because the command has just been inserted.
⚠	The command has a problem that has been classified as non-fatal. For example, an input file has not been found. In general, commands with warnings need to be fixed unless work is preliminary.
⊗	The command has failed, meaning that output is likely incomplete. A problem summary and recommendation to fix the problem are available in the status information. Commands with failures generally need to be fixed. Software support should be contacted if the fix is not evident.

It is possible that a problem indicator will be shown during command editing and will be cleared when commands are run. For example, a command may depend on a file that is created by a previous command. It is important that errors displayed after running commands are resolved.

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 and other output that result from processing are listed in the bottom of the main interface, as shown in the following figure:



TSTool - no commands saved

Input/Query Options

Input type: HydroBase
Data type: Stream - Streamflow
Time step: Month
Where: Division = 3 - Rio Grande
Where: Matches
Where: Matches

Time Series List (112 time series, 2 selected)

	ID	CO Abbrev.	Name/Description	Data Sourc
1	08235350	ALAJASCO	ALAMOSA RIVER ABOVE JASPE...	USGS
2	08236000	ALATERCO	ALAMOSA RIVER ABOVE TERRA...	DWR
3	08235250	ALAWIGCO	ALAMOSA RIVER ABOVE WIGHT...	DWR
4	08235700	ALACASCO	ALAMOSA RIVER BELOW CASTL...	DWR
5	ALARANCO	ALARANCO	ALAMOSA RIVER BELOW RANG...	DWR
6	08236500	ALABELCO	ALAMOSA RIVER BELOW TERRA...	DWR
7	BIGSPGCO	BIGSPGCO	BIG SPRING CREEK AT MEDANO ...	DWR

Commands (4 commands, 0 selected, 0 with failures, 0 with warnings)

```
# 08235350 - ALAMOSA RIVER ABOVE JASPER, CO
08235350.USGS.Streamflow.Month~HydroBase
# 08236000 - ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO.
08236000.DWR.Streamflow.Month~HydroBase
```

Results

2 time series, 2 selected

1) ALAMOSA RIVER ABOVE JASPER, CO - 08235350.USGS.Streamflow.Month (1995-07 to 1999-10)
2) ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO. - 08236000.DWR.Streamflow.Month (1914-05 to 2012-09)

Completed running commands. Use Results and Tools menus.

GUI_MainWithTS

TSTool after Running Commands

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

The following results may be available, depending on commands that were run:

- ***Ensembles*** – groups of time series with an ensemble identifier. Individual time series that are associated with an ensemble also are shown in the ***Time Series*** tab. Right-click on item to access viewing and analysis options.
- ***Output Files*** – files that are created during processing. Single click on a file to view.
- ***Problems*** – a full listing of warning and failure messages from all commands.
- ***Tables*** – column-oriented tables created during processing. Right-click on a table in the list to view the table.
- ***Time Series*** – time series created during processing. Right-click on one or more time series to view the time series.
- ***Views*** – alternate views of time series, other than the list of time series that is ordered based on command output.

Two progress bars at the bottom of the main window are updated during processing. The left progress bar indicates the overall progress in processing the commands (100% means that all commands have been processed). The right progress bar is used with commands that provide incremental progress during processing, a feature that will be phased in over time for commands that take longer to run. For example, if a single command processes many time series, this progress bar can be used to indicate progress in the command.

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

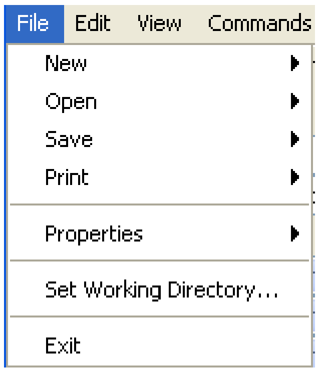
Menu Choice	Description
Graph – Area	Display a graph where the area below time series is filled in.
Graph – Area (stacked)	Display a graph where the area below time series is filled in, with the time series values being cumulative.
Graph – Bar (left of date)	Display bar graph for selected time series, 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 (HTML)	Display an HTML summary for selected time series using the default web browser.
Report – Summary (Text)	Display a text 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).

The **TSView Time Series Viewing Tools** appendix provides additional information about time series products.

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 or when time series have been processed.



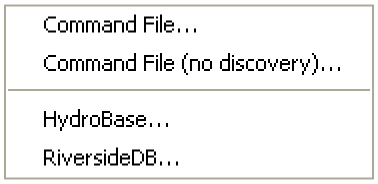
Menu_File

File Menu

3.4.1 File…New – Open Command File or Databases

The ***File…New…Command File*** menu item clears the current commands so that a new command file can be started. A new command file name will be requested when the commands are saved.

3.4.2 File…Open – Open Command File or Databases



Menu_File_Open

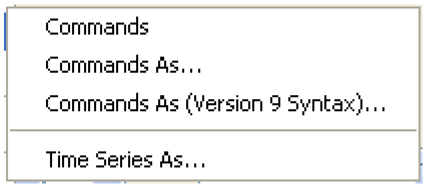
File…Open Menu

The ***File…Open…Command File*** menu item displays a dialog to select an existing command file. After a file is selected, the file contents replace the contents of the ***Commands*** list. A prompt is displayed if commands already exist in the ***Commands*** list and have been modified. Opening a command file causes the working directory (folder) to be set to the folder from which the command file was read. All other files specified with a relative path will be found relative to the command file. The ***File…Open…Command File (no discovery)*** menu item can be selected to load a command file without running discovery mode, which may be appropriate when loading large command files such as generated by expanding a template. Discovery mode is needed when editing commands because it provides lists of time series and other data to command editors.

TSTool automatically will attempt to update older command files to new syntax if a command has changed. If a change occurs, the command file will be marked as modified and will need to be saved to reflect the changes. If an error occurs updating a command, it will be marked with an error and a comment will be inserted with the original command indicating that an automated update could not occur. Unrecognized commands are marked with an error and will generate errors if run.

If appropriate for the TSTool configuration, other menu items will be displayed to allow opening databases. It is recommended that database connections be configured to automatically open; however, the menus are useful for development and troubleshooting.

3.4.3 File…Save – Save Command File, and Time Series



Menu_File_Save

File…Save Menu

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 command file is shown in the TSTool title bar and can be referred to when deciding a new command file name. All commands are saved, even if only a subset is selected. Saving a command file causes the working directory to be set to the folder where the command file was written. All other files specified with a relative path will be found relative to the command file.

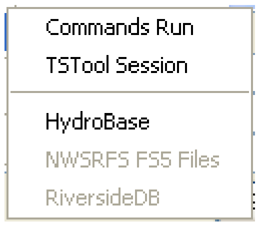
The ***File…Save…Commands As (Version 9 Syntax)…*** menu items saves the commands to a version 9 format command file. The primary difference between version 9 and later syntax is that the `TS Alias = Command(…)` syntax was replaced with `Command(Alias=…)` syntax in TSTool version 10.

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 and provider greater control.

3.4.4 Print Commands

The ***File…Print…Commands*** menu prints the contents of the ***Commands*** list. This is useful when editing and troubleshooting commands.

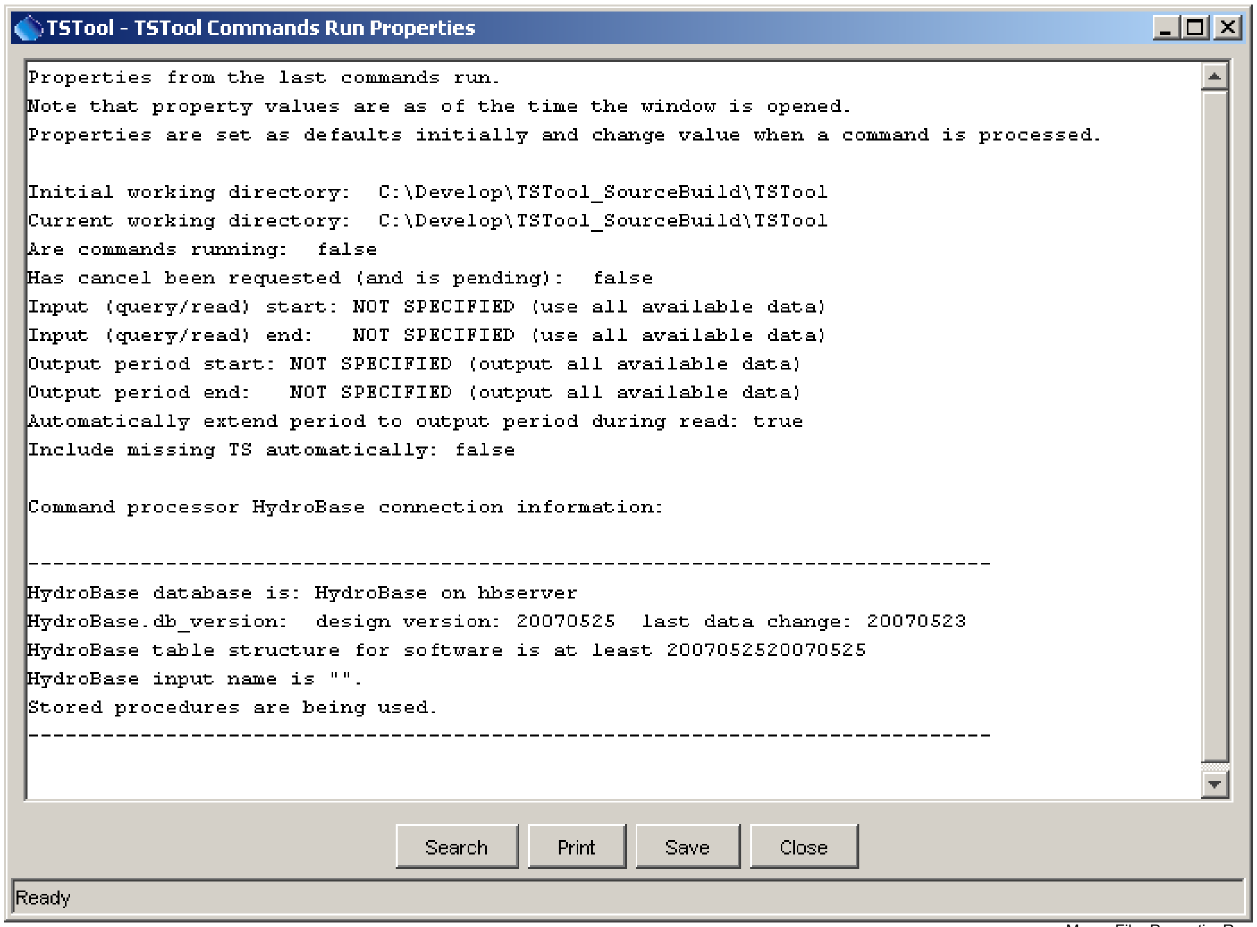
3.4.5 Properties for Commands Run, TSTool Session, and Input Types



Menu_File_Properties

File…Properties Menu

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



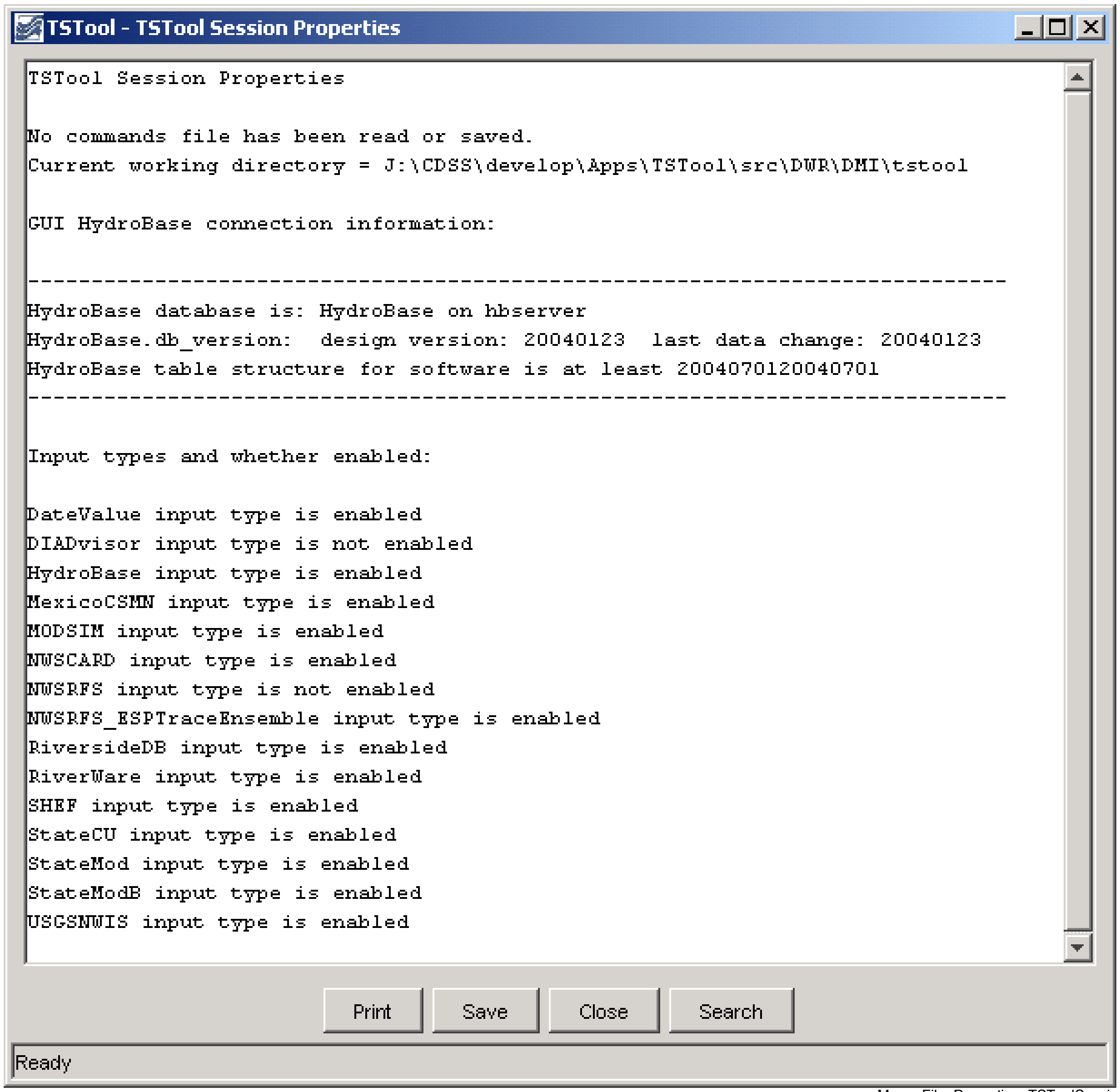
```
Properties from the last commands run.
Note that property values are as of the time the window is opened.
Properties are set as defaults initially and change value when a command is processed.

Initial working directory:  C:\Develop\TSTool_SourceBuild\TSTool
Current working directory:  C:\Develop\TSTool_SourceBuild\TSTool
Are commands running:  false
Has cancel been requested (and is pending):  false
Input (query/read) start: NOT SPECIFIED (use all available data)
Input (query/read) end:   NOT SPECIFIED (use all available data)
Output period start: NOT SPECIFIED (output all available data)
Output period end:   NOT SPECIFIED (output all available data)
Automatically extend period to output period during read: true
Include missing TS automatically: false

Command processor HydroBase connection information:

-----------------------------------------------------------------------------
HydroBase database is: HydroBase on hbserver
HydroBase.db_version:  design version: 20070525  last data change: 20070523
HydroBase table structure for software is at least 2007052520070525
HydroBase input name is "".
Stored procedures are being used.
-----------------------------------------------------------------------------
```

Menu_File_PropertiesRun

Properties of the Last Commands Run

This information is useful for troubleshooting processing.

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



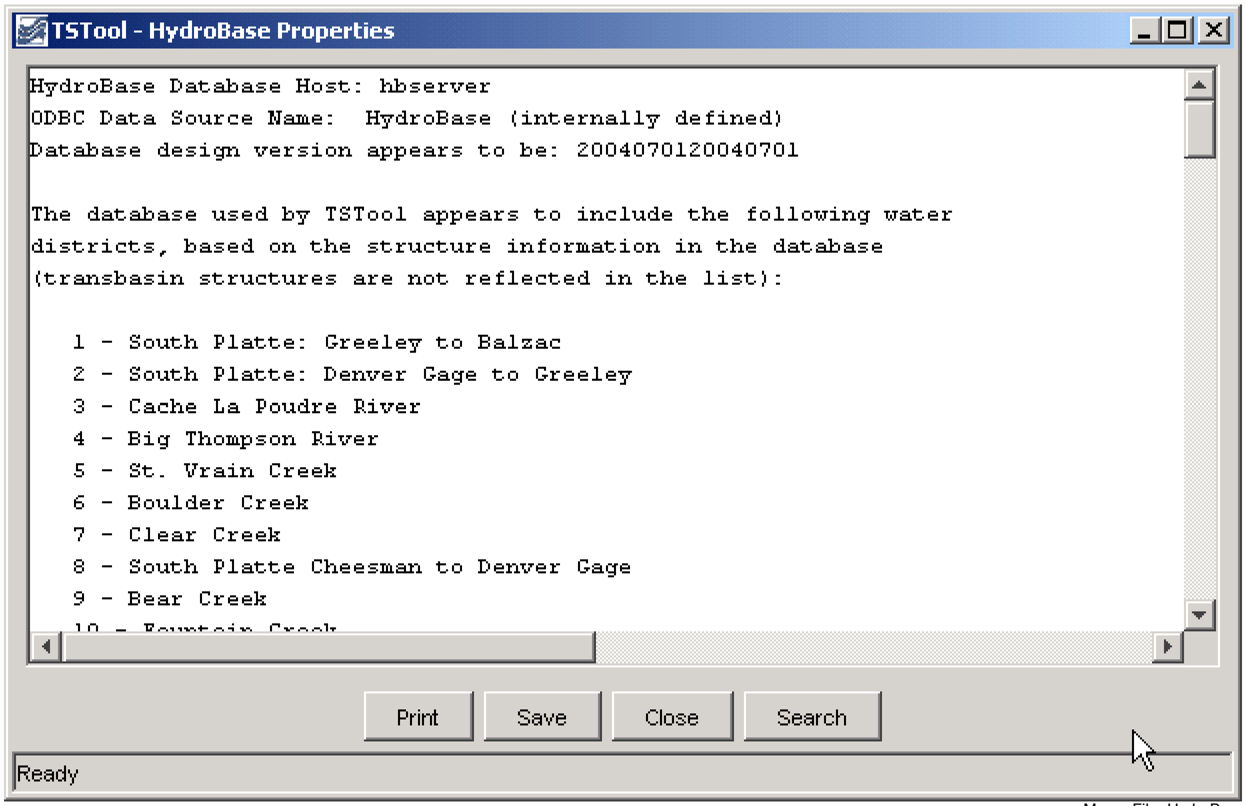
```
TSTool Session Properties

No commands file has been read or saved.
Current working directory = J:\CDSS\develop\Apps\TSTool\src\DWR\DMI\tstool

GUI HydroBase connection information:

-----------------------------------------------------------------------------
HydroBase database is: HydroBase on hbserver
HydroBase.db_version:  design version: 20040123  last data change: 20040123
HydroBase table structure for software is at least 2004070120040701
-----------------------------------------------------------------------------

Input types and whether enabled:

DateValue input type is enabled
DIADvisor input type is not enabled
HydroBase input type is enabled
MexicoCSMN input type is enabled
MODSIM input type is enabled
NWSCARD input type is enabled
NWSRFS input type is not enabled
NWSRFS_ESPTraceEnsemble input type is enabled
RiversideDB input type is enabled
RiverWare input type is enabled
SHEF input type is enabled
StateCU input type is enabled
StateMod input type is enabled
StateModB input type is enabled
USGSNWIS input type is enabled
```

Menu_File_Properties_TSToolSession

TSTool Session Properties

This information is useful for checking the TSTool configuration.

Additional properties displays may be available depending on enabled input types. For example, 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 Database Host: hbserver
ODBC Data Source Name:  HydroBase (internally defined)
Database design version appears to be: 2004070120040701

The database used by TSTool appears to include the following water
districts, based on the structure information in the database
(transbasin structures are not reflected in the list):

   1 - South Platte: Greeley to Balzac
   2 - South Platte: Denver Gage to Greeley
   3 - Cache La Poudre River
   4 - Big Thompson River
   5 - St. Vrain Creek
   6 - Boulder Creek
   7 - Clear Creek
   8 - South Platte Cheesman to Denver Gage
   9 - Bear Creek
```

Menu_File_HydroBase

HydroBase Properties Dialog

3.4.6 Set Working Directory

The ***File…Set Working Directory*** menu item displays a file chooser dialog that selects the working directory. The working directory is used by TSTool to local files specified with relative paths. The working directory normally is set in one of the following ways, with the current setting being defined by the most recent selection:

1. The startup directory for the TSTool program,
2. The directory where a command file was opened,
3. The directory where a command file was saved,
4. The directory specified by a `SetWorkingDir()` command (use of this command is discouraged because it hard-codes a system-specific folder in command files),
5. The directory specified by ***File…Set Working Directory***.

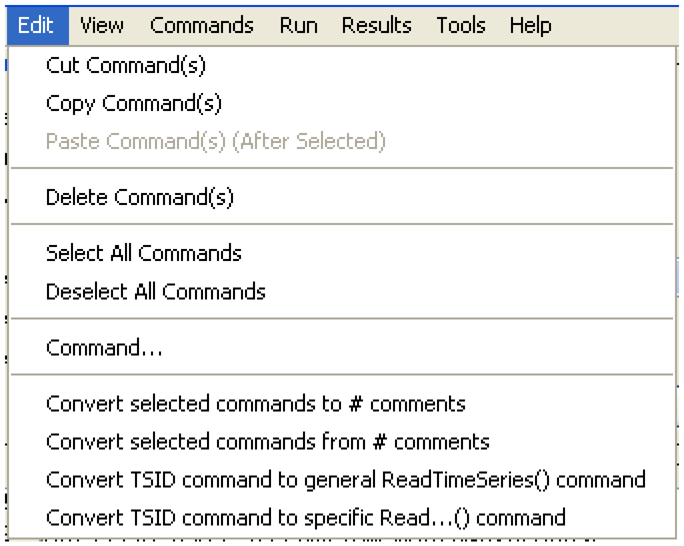
The menu item is provided to allow the working directory to be set before a command file has been saved (or opened).

3.4.7 Exit

The ***File…Exit*** menu exits TSTool. A confirmation prompt is displayed before exiting. If commands have been modified, they can be saved before exiting. Commands may have been automatically updated by TSTool if an old command file was read.

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 choices similar to those described below.


Edit
View
Commands
Run
Results
Tools
Help
Cut Command(s)
Copy Command(s)
Paste Command(s) (After Selected)
Delete Command(s)
Select All Commands
Deselect All Commands
Command...
Convert selected commands to # comments
Convert selected commands from # comments
Convert TSID command to general ReadTimeSeries() command
Convert TSID command to specific Read...() command

Menu_Edit

Edit Menu

3.5.1 Cut/Copy/Paste/Delete

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

Paste Command(s) (After Selected) 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 Command(s) (After Selected)***. 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 Command(s)*** choice. Additionally, after lines in the ***Commands*** list 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. A confirmation prompt is displayed if no commands are 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 Command

The ***Edit…Command*** 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 automatically will be detected and will be converted to new command syntax.** This feature is also accessible by right clicking on the ***Commands*** list and selecting the ***Edit*** menu item and by double-clicking on a command.

3.5.4 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.

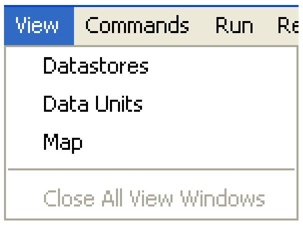
Note that the multi-line `/* */` comment notation can be inserted using the ***Commands…General – Comments*** menu.

3.5.5 Convert TSID to Read Commands

The ***Edit…Convert TSID command to general ReadTimeSeries() command*** inserts a new `ReadTimeSeries()` command using the TSID and replaces the original TSID command. The `ReadTimeSeries()` command allows an alias to be specified for the time series.

The ***Edit…Convert TSID command to specific Read…() command*** inserts a new read command using the TSID and replaces the original TSID command. Specific read commands may not be available for all input types and therefore the `ReadTimeSeries()` command may need to be used. Alternatively, insert a read command using the ***Commands*** menu choices.

3.6 View Menu – Display Useful Information and Map Interface



Menu_View

View Menu

The ***View…Datastores*** menu item displays a list of configured data stores, which is useful when troubleshooting whether a data store is properly configured. Note that some data stores are not listed in the main window ***Datastore*** choices but are available for use by commands.

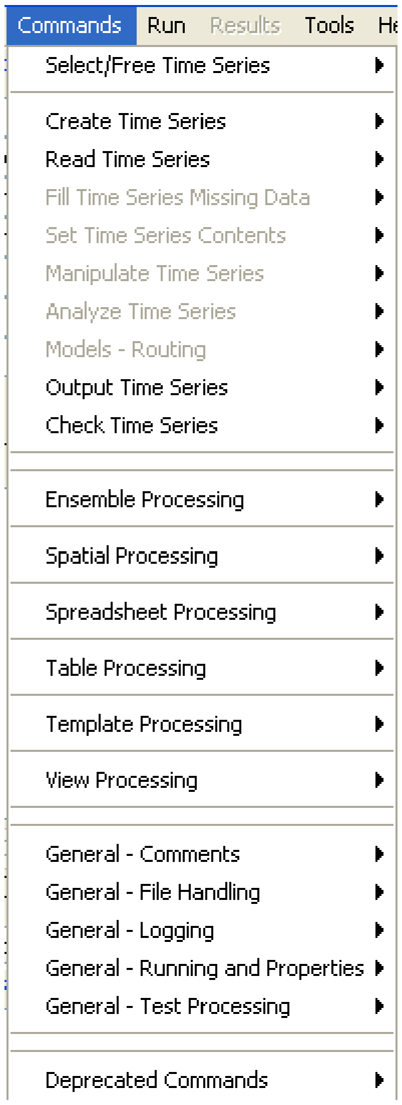
The ***View…Data Units*** menu item displays a list of configured data units, which are recognized by the `ConvertDataUnits()` command and other TSTool features that enforce data unit consistency. Data units from all data repositories are not automatically understood by TSTool, although additional capabilities may be added in the future.

The ***View…Map*** menu displays a map interface in a separate window. See the **Using the Map** chapter for more information.

The ***View…Close All View Windows*** menu closes all visible view windows, including graphs. This is useful if the command file has generated many graphs and the user wishes to close them all at once.

3.7 Commands Menu

The ***Commands*** menu provides several menus (as shown in the following figure), which insert commands into the ***Commands*** list.



Menu_Commands

Commands Menu

Time series commands are organized into the following categories:

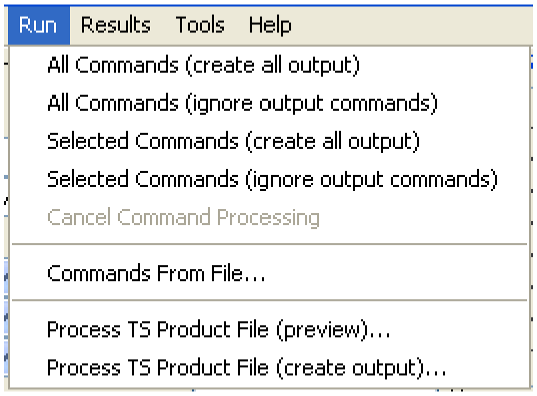
Select/Free Time Series – select or deselect time series for processing, free time series
Create Time Series – create one or more new time series
Read Time Series – read time series from a file or database
Fill Time Series Missing Data – fill missing data
Set Time Series Contents – set time series data or properties
Manipulate Time Series – manipulate data (e.g., scale a time series' data values)
Analyze Time Series – perform analysis on time series (e.g., determine wet/dry/average pattern)
Models – Routing – lag and attenuate time series
Output Time Series – write time series results to a file or produce graphical products
Check Time Series – check time series values and statistics against criteria
Ensemble Processing – commands that are specific to ensemble processing
Spatial Processing – commands that process spatial data
Spreadsheet Processing – commands that process spreadsheet files
Table Processing – commands that are specific to table processing

Template Processing – commands that are specific to template processing
View Processing – commands that are specific to view processing (alternate views of results)
General – Comments – insert comments
General – File Handling – commands to manipulate files and perform FTP and web retrieval
General – Logging – commands for logging (e.g., open a log file, set message levels)
General – Running and Properties – commands to control processing and run external programs
General – Test Processing – commands to process tests, to validate software and procedures
Deprecated Commands – commands that are planned for removal

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 results.


Run
Results
Tools
Help
All Commands (create all output)
All Commands (ignore output commands)
Selected Commands (create all output)
Selected Commands (ignore output commands)
Cancel Command Processing
Commands From File...
Process TS Product File (preview)...
Process TS Product File (create output)...

Menu_Run

Run Menu

The ***Run…All Commands (create all output)*** menu will process all the commands in the ***Commands*** list and create output.

The ***Run…All Commands (ignore output commands)*** menu will process the commands in the ***Commands*** list, ignoring commands that generate output products. 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…Cancel Command Processing*** menu items will be enabled if command processing is active, and allows the processing to be canceled. Processing may continue until the current command finishes.

The ***Run…Commands From File*** choice will run a command file but will 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.

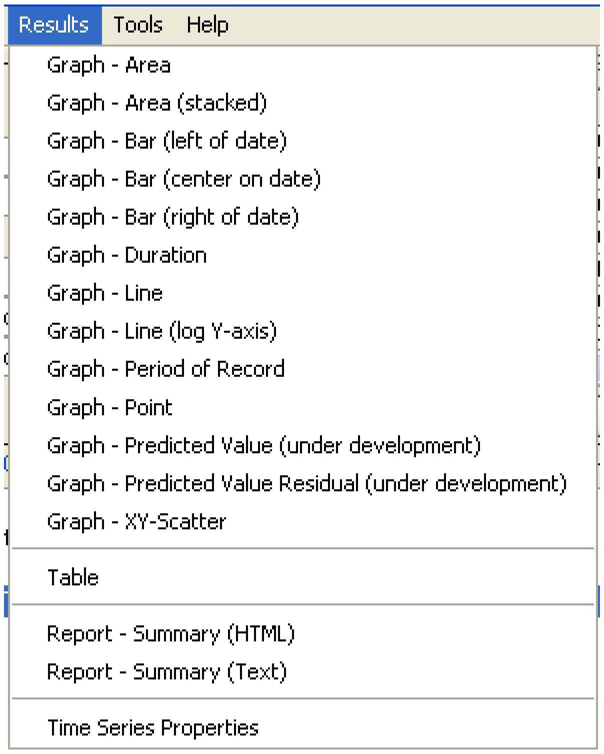
Menu items similar to the above also are available in a popup menu by right clicking on the ***Commands*** list.

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 ***Results*** list at the bottom of the TSTool main window. The time series can be viewed multiple times, using the same time series results.



Menu_Results

Results Menu

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.

Graphing time series results in slightly different viewing options being available, depending on the type of graph. Three views of time series are generally available: graph, summary, and table. Graph properties can be edited by right-clicking on the graph. The **TSView Time Series Viewing Tools Appendix** describes the graphing tools in detail. The following table summarizes ***Results*** menu items.

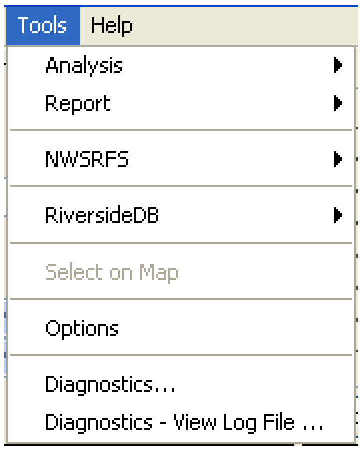
Results Menu Choices

Menu Choice	Description
Graph – Area	Display a graph where the area below time series is filled in.
Graph – Area (stacked)	Display a graph where the area below time series is filled in, with the time series values being cumulative.

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. Each time series' period is indicated by a horizontal line. An alternative to this graph type is to use the ***Tools…Data Coverage by Year*** report (see **Chapter 5 – Tools**).
Graph – Point	Display a graph using symbols but no connecting lines. This is useful for data that have infrequent measurements.
Graph – Predicted Value	Display a graph of data and the predicted values from regression. 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.
Graph – Predicted Value Residual	Display a graph of data minus the predicted values from regression. 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.
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 (HTML)	Display an HTML summary for selected time series using the default web browser.
Report – Summary (Text)	Display a text 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).

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.



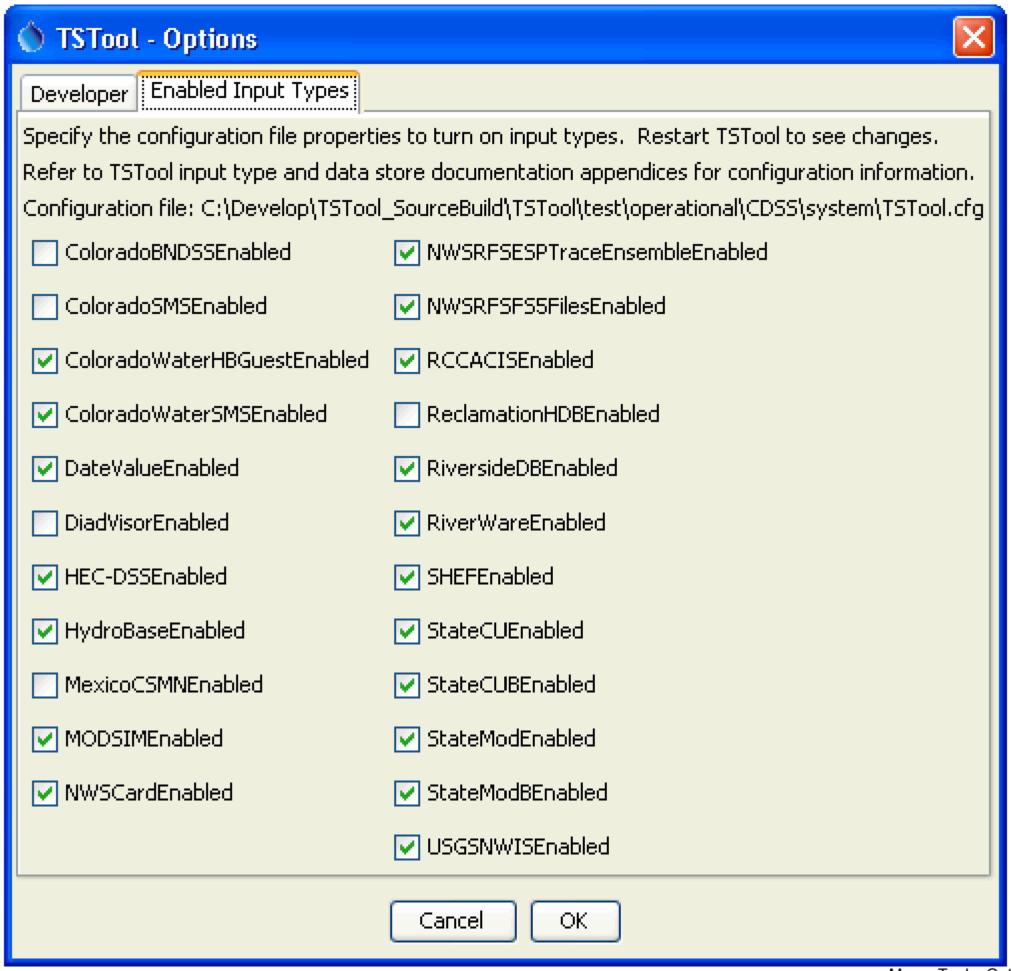
Menu_Tools

Tools Menu

Analysis tools are described in more detail in **Chapter 5 – Tools**. The following sections describe the ***Tools…Options*** and ***Tools…Diagnostics*** features. Some tools are provided based on enabled input types.

3.10.1 Options

The ***Tools…Options*** menu displays program options. The ***Developer*** tab configures information that should only be modified by software developers. The ***Enabled Input Types*** tab displays the input types that are enabled in the TSTool configuration file and allows the values to be changed. A warning will be shown if there are insufficient permissions to read or write the configuration file. TSTool must be restarted to reflect changes to the list of enabled input types. Enabling an input type may result in additional commands and tools being shown; however, additional configuration may be required to fully enable access to data. Refer to the input type and data shore appendices for more information..



Specify the configuration file properties to turn on input types. Restart TSTool to see changes.
Refer to TSTool input type and data store documentation appendices for configuration information.
Configuration file: C:\Develop\TSTool_SourceBuild\TSTool\test\operational\CDSS\system\TSTool.cfg

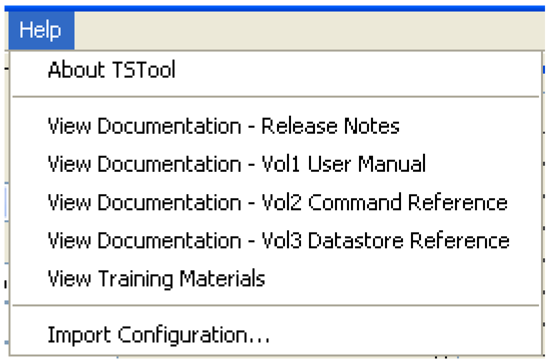
☐ ColoradoBNDSSEnabled	✓ NWSRFSESPTraceEnsembleEnabled
☐ ColoradoSMSEnabled	✓ NWSRFSFS5FilesEnabled
✓ ColoradoWaterHBGuestEnabled	✓ RCCACISEnabled
✓ ColoradoWaterSMSEnabled	☐ ReclamationHDBEnabled
✓ DateValueEnabled	✓ RiversideDBEnabled
☐ DiadVisorEnabled	✓ RiverWareEnabled
✓ HEC-DSSEnabled	✓ SHEFEnabled
✓ HydroBaseEnabled	✓ StateCUEnabled
☐ MexicoCSMNEnabled	✓ StateCUBEnabled
✓ MODSIMEnabled	✓ StateModEnabled
✓ NWSCardEnabled	✓ StateModBEnabled
	✓ USGSNWISEnabled

Menu_Tools_Options_InputTypes

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


Help
About TSTool
View Documentation - Release Notes
View Documentation - Vol1 User Manual
View Documentation - Vol2 Command Reference
View Documentation - Vol3 Datastore Reference
View Training Materials
Import Configuration…

Menu_Help

Help Menu

The ***Help…About TSTool*** menu item displays the program version number, for use in troubleshooting and support. Information about the software and system can be displayed from the version dialog, to help with roubleshooting.

The ***Help…View Documentation*** menu items display the PDF documentation using the default PDF viewer.

The ***Help…View Training Materials*** menu item opens a file system browser to the location of the training materials. Training materials are organized by topic and provide working examples.

The ***Help…Import Configuration*** menu item is used to import an old TSTool configuration file into a new software installation. It may be necessary to manually copy configuration old files, in particular for data store configuration. TSTool configuration files are saved in the *system* folder under the TSTool installation.

3.12 Batch Mode Execution

The TSTool interface provides immediate feedback on whether commands are properly defined and input data are available. However, once a command file has been defined, it may be appropriate to process the commands in "headless mode", without the graphical user interface. To do so, run TSTool on the command line as follows:

```
[PathToTSTool]tstool -commands commands.TSTool [parameters]
```

There are a number of potential issues that may impact command line execution:

- If the folder containing the TSTool executable (`tstool.exe` on Windows and `tstool` on *NIX) is in the `PATH` environment variable, then no leading path is needed. However, because different versions of TSTool may be installed on the system, specifying the leading path to the executable may be appropriate.
- On Windows, entering the TSTool command line in a Command Prompt window causes the prompt to be immediately returned, even though TSTool is still running. This can be disconcerting in particular because it may be difficult to know when TSTool has finished processing the command file. Placing the command in a batch file (**.bat*) can help.
- TSTool has the concept of "working directory", which is the root location to which relative paths are referenced. Normally this is the location of the command file once opened. However, when running TSTool in batch mode, there are a number of folders involved: the location of the TSTool executable, the location from which TSTool is run, and the location of the command file. These multiple locations can make it difficult to troubleshoot. One option is to use absolute paths in the TSTool command line for the executable and the command file so it is very clear where the TSTool executable and command file are located. These paths can be coded into a batch file (**.bat* on Windows, or shell script on *NIX).
- TSTool can be run in headless mode and still create graph windows (e.g., when TSTool supports the functionality of another application). Make sure to evaluate whether the `-nomaingui` parameter is needed in addition to `-commands`.
- The TSTool default log file under the logs folder of the installation is used when TSTool first starts up. If a `StartLog()` command is used in the command file, it will be used when the command file is run. Refer to the appropriate log file when troubleshooting.

The following table lists the TSTool command line parameters.

TSTool Command Line Parameters

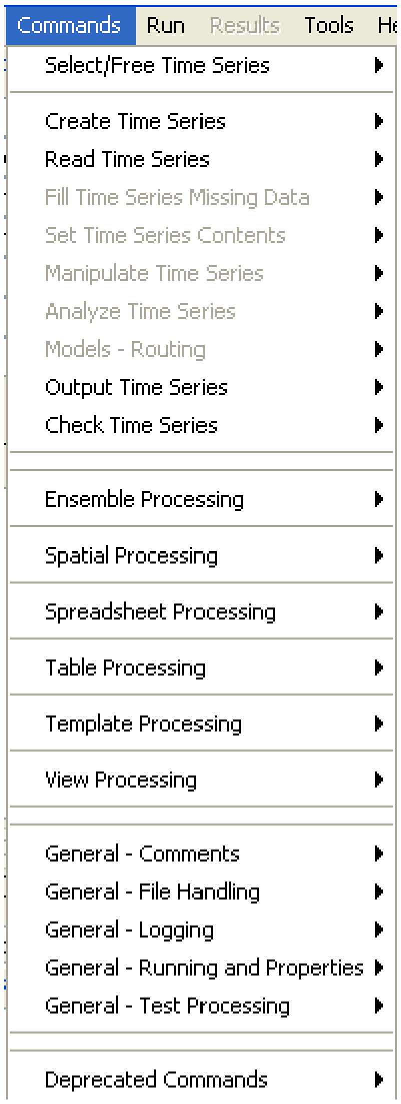
Parameter	Description	Default
`-commands CommandFile`	Specify the name of the command file to run in batch mode. TSTool will process the commands and exit.	Start TSTool in interactive mode.
`CommandFile`	The command file can be specified without `-commands` to start up the TSTool GUI and load the command file. This behavior occurs when selecting a `*.TSTool` file in Windows Explorer. See also `-runcommandsonload`.	
`-config ConfigFile`	Start TSTool using the specified configuration file. This is useful in software test environments.	Start TSTool using the *system/TSTool.cfg* file under the software installation.
`-dTerm,Log`	Specify the debug level (0+) for terminal and log file debug messages. This is useful for printing troubleshooting messages at startup. One or both values can be specified.	No debug messages are generated.
`-home InstallFolder`	Specify the install folder. This parameter is used internally by the TSTool launcher (Launch4J) and in the software development environment to specify the TSTool home, in order to locate other files.	Should always be set by the TSTool launcher.
`-nodiscovery`	Do not run discovery on commands as they are loaded. This can be used for large command files that will not be edited, in order to decrease load time.	Run discovery on commands as they are loaded.
`-nomaingui`	Do not show the TSTool main GUI when running commands, but do allow graph windows to be displayed. Closing the last visible graph window will close the TSTool application.	If `-commands` is specified, do not show any windows. If `-nomaingui` also is specified, allow graph windows to show.
`-runcommandsonload`	Load and run the commands, used to start the GUI with a specific command file.	
`-server`	Start TSTool as a REST restlet server (under development).	
`-test`	Run TSTool in test mode, for developers.	
`Property=Value`	Define TSTool global properties (similar to the configuration file). These properties typically are used during development to test specific features.	

This page is intentionally blank.

4 Commands

Version 10.17.00, 2013-02-18

The ***Commands*** menu provides choices to insert commands in the ***Commands*** list.



Menu_Commands

Commands Menu

Commands are organized into the following categories:

Select/Free Time Series – select or deselect time series for processing, free time series
Create Time Series – create one or more new time series
Read Time Series – read time series from a file or database
Fill Time Series Missing Data – fill missing data
Set Time Series Contents – set time series data or properties
Manipulate Time Series – manipulate data (e.g., scale a time series' data values)
Analyze Time Series – perform analysis on time series (e.g., determine wet/dry/average pattern)
Models – Routing – lag and attenuate time series

Output Time Series – write time series results to a file or produce graphical products
Check Time Series – check time series values and statistics against criteria
Ensemble Processing – commands that are specific to ensemble processing
Spatial Processing – commands that process spatial data
Spreadsheet Processing – commands that process spreadsheet files
Table Processing – commands that are specific to table processing
Template Processing – commands that are specific to template processing
View Processing – commands that are specific to view processing (alternate views of results)
General – Comments – insert comments
General – File Handling – commands to manipulate files and perform FTP and web retrieval
General – Logging – commands for logging (e.g., open a log file, set message levels)
General – Running and Properties – commands to control processing and run external programs
General – Test Processing – commands to process tests, to validate software and procedures
Deprecated Commands – commands that are planned for removal

The menus for each category are discussed in the following sections. Selecting a command menu item will display a command editor. The editor dialog and command syntax are described in detail in the **Command Reference** at the end of this documentation. Menus are enabled/disabled depending on the state of the application (e.g., whether time series are available). When ***OK*** is pressed in the command editor, the command will be inserted before the first selected command. If no commands are selected, the command will be added to the end of the command list. If necessary, right click on the command list and use ***Deselect All Commands*** – this will ensure that commands are added at the end of the list.

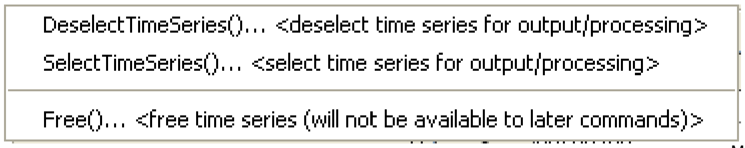
The `TSList` command parameter is available for many commands to specify the list of time series to process. Criteria can be specified to match the alias and TSID strings. The pattern typically can contain wildcards. For example, specify `TSID=A*` to match aliases starting with `A`, or `TSID=A*.*.*.Month` to match TSIDs with a location starting with `A`. An identifier string with periods indicates that the string may be a dot-delimited TSID string, and the TSID parts are compared individually.

For example, `Command(TSList=AllTS,…)` will process all available time series and `Command(TSList=AllMatchingTSID,TSID="A*",…)` will process all time series with an alias that starts with `A`. Refer to the command reference for a description of available parameters for a specific command. The standard values for the `TSList` parameter are as follows:

TSList Parameter Value	Description
`AllMatchingTSID`	Process all time series that match the `TSID` parameter value.
`AllTS`	Process all time series.
`EnsembleID`	Process all time series for the ensemble identifier specified by the `EnsembleID` parameter.
`FirstMatchingTSID`	Process the first time series (from the start of commands to the previous command) that matches the `TSID` parameter.
`LastMatchingTSID`	Process the last (previous to the current command) time series that matches the `TSID` parameter.
`SelectedTS`	Process all time series that have been selected with `SelectTimeSeries()` and `DeselectTimeSeries()` commands.
`SpecifiedTSID`	Process all time series in the specified list of time series identifiers (for example when used with the `Add()` command the `AddTSID` parameter is used to provide the specified identifiers).

4.1 Select/Free Time Series

The ***Commands…Select/Free Time Series*** menu inserts commands for selecting, deselecting, and freeing time series.


DeselectTimeSeries()... <deselect time series for output/processing>
SelectTimeSeries()... <select time series for output/processing>
Free()... <free time series (will not be available to later commands)>

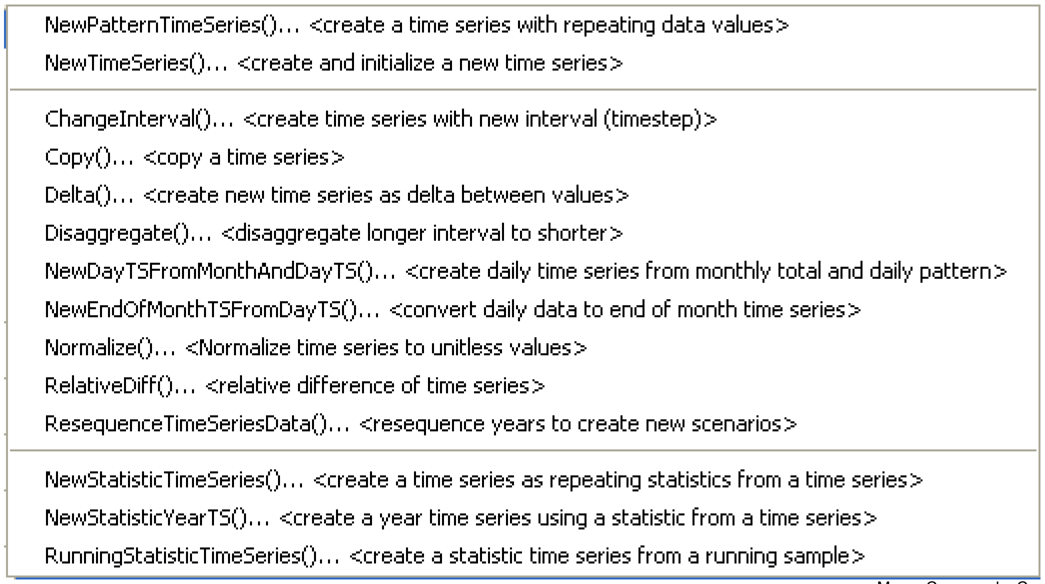
Menu_Commands_SelectTimeSeries

Commands...Select/Free Time Series Menu

Time series can be selected for specific actions and are processed by using the `TSList=SelectedTS` command parameters. The `Free()` command frees time series resources, for example when temporary time series are no longer needed or to free up memory for a large command file.

4.2 Create Time Series

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


NewPatternTimeSeries()... <create a time series with repeating data values>
NewTimeSeries()... <create and initialize a new time series>
ChangeInterval()... <create time series with new interval (timestep)>
Copy()... <copy a time series>
Delta()... <create new time series as delta between values>
Disaggregate()... <disaggregate longer interval to shorter>
NewDayTSFromMonthAndDayTS()... <create daily time series from monthly total and daily pattern>
NewEndOfMonthTSFromDayTS()... <convert daily data to end of month time series>
Normalize()... <Normalize time series to unitless values>
RelativeDiff()... <relative difference of time series>
ResequenceTimeSeriesData()... <resequence years to create new scenarios>
NewStatisticTimeSeries()... <create a time series as repeating statistics from a time series>
NewStatisticYearTS()... <create a year time series using a statistic from a time series>
RunningStatisticTimeSeries()... <create a statistic time series from a running sample>

Menu_Commands_CreateTimeSeries

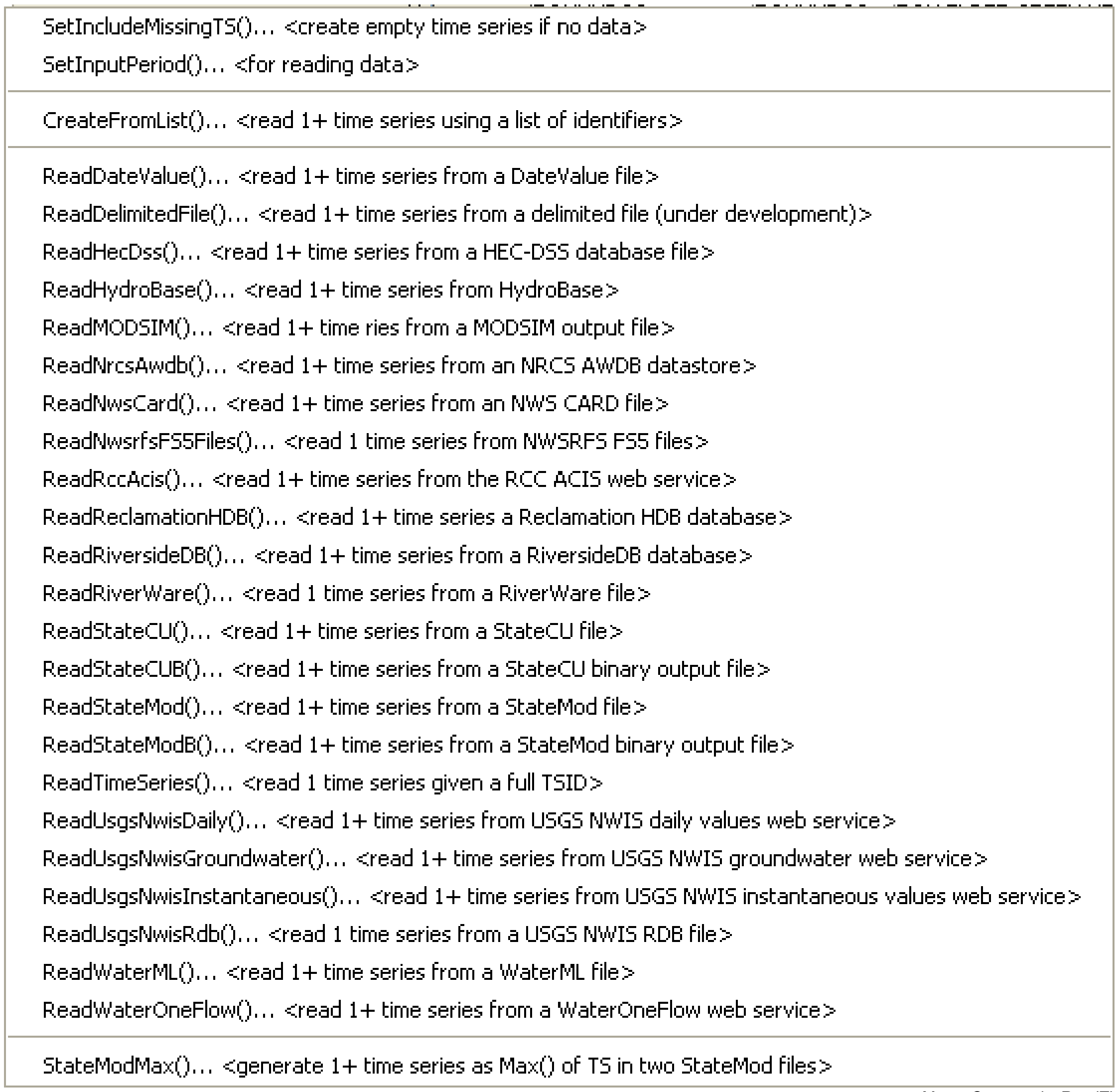
Commands...Create Time Series Menu

These commands create new time series from user-supplied data (see `NewTimeSeries()`) or data from input time series. A time series created from an existing time series is 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. Consequently, the commands force users to provide new TSID and/or alias information to identify the time series.

Commands may create a single or multiple output time series, although the trend is to continue enhancing commands to allow multiple time series to be processed. TSIDs and/or aliases for new time series are used during the discovery phase of command processing to provide identifiers for later commands. This allows command editors to provide time series choices even when the commands have not been run.

4.3 Read Time Series

The ***Commands…Read Time Series*** menu inserts commands to read time series from a database, file, or web service (internet).



SetIncludeMissingTS()… <create empty time series if no data>
SetInputPeriod()… <for reading data>

CreateFromList()… <read 1+ time series using a list of identifiers>

ReadDateValue()… <read 1+ time series from a DateValue file>
ReadDelimitedFile()… <read 1+ time series from a delimited file (under development)>
ReadHecDss()… <read 1+ time series from a HEC-DSS database file>
ReadHydroBase()… <read 1+ time series from HydroBase>
ReadMODSIM()… <read 1+ time ries from a MODSIM output file>
ReadNrcsAwdb()… <read 1+ time series from an NRCS AWDB datastore>
ReadNwsCard()… <read 1+ time series from an NWS CARD file>
ReadNwsrfsFS5Files()… <read 1 time series from NWSRFS FS5 files>
ReadRccAcis()… <read 1+ time series from the RCC ACIS web service>
ReadReclamationHDB()… <read 1+ time series a Reclamation HDB database>
ReadRiversideDB()… <read 1+ time series from a RiversideDB database>
ReadRiverWare()… <read 1 time series from a RiverWare file>
ReadStateCU()… <read 1+ time series from a StateCU file>
ReadStateCUB()… <read 1+ time series from a StateCU binary output file>
ReadStateMod()… <read 1+ time series from a StateMod file>
ReadStateModB()… <read 1+ time series from a StateMod binary output file>
ReadTimeSeries()… <read 1 time series given a full TSID>
ReadUsgsNwisDaily()… <read 1+ time series from USGS NWIS daily values web service>
ReadUsgsNwisGroundwater()… <read 1+ time series from USGS NWIS groundwater web service>
ReadUsgsNwisInstantaneous()… <read 1+ time series from USGS NWIS instantaneous values web service>
ReadUsgsNwisRdb()… <read 1 time series from a USGS NWIS RDB file>
ReadWaterML()… <read 1+ time series from a WaterML file>
ReadWaterOneFlow()… <read 1+ time series from a WaterOneFlow web service>

StateModMax()… <generate 1+ time series as Max() of TS in two StateMod files>

Menu_Commands_ReadTimeSeries

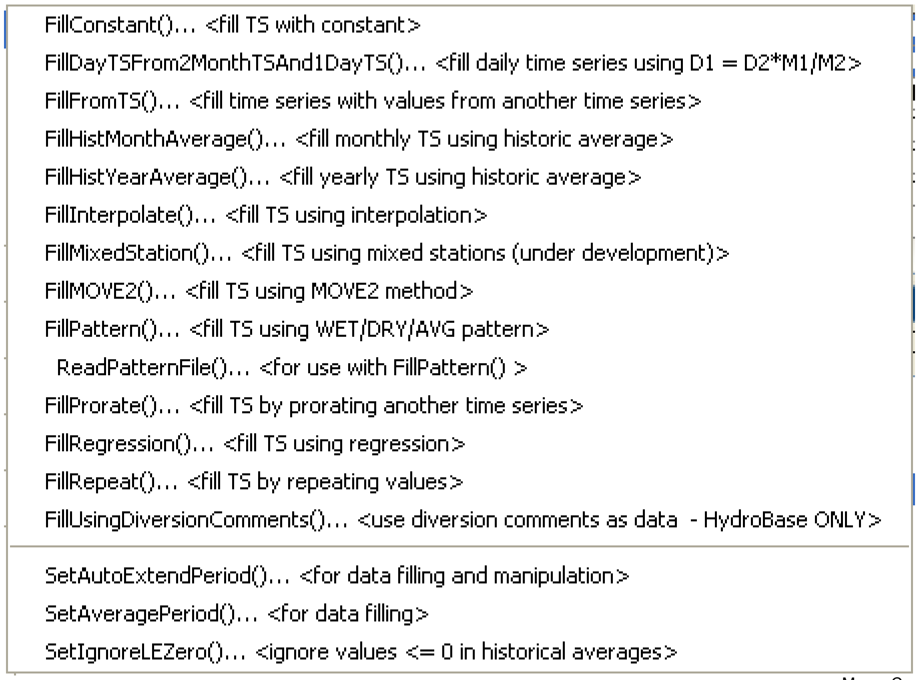
Commands...Read Time Series Menu

Read commands are alphabetized and are shown for enabled input types. The commands that are shown will depend on the input types and datastores that are enabled. Several commands perform supporting functions, such as setting the input period to read. Some data formats allow only a single time series to be read whereas other formats allow multiple time series to be read. Using read commands rather than TSID commands allows more control over the read, such as assigning an alias and converting data units.

The `CreateFromList()` command uses a delimited input file to provide location information and internally creates a list of TSIDs to read.

4.4 Fill Time Series Missing Data

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


FillConstant()... <fill TS with constant>
FillDayTSFrom2MonthTSAnd1DayTS()... <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>
ReadPatternFile()... <for use with FillPattern() >
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>

Menu_Commands_FillTimeSeries

Commands...Fill Time Series Missing Data Menu

Fill commands will change only values that are missing, whereas set commands (see next section) 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.

Time series may contain missing data due to the following or other 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 typically are 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 often is 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`, a similar extreme value, or `NaN` (not a number) 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 as blanks, by showing the missing data value, or a string (e.g., `NC`), depending on the constraints of the product. For example, an HTML time series highlights missing values and shows flags as a superscript.

Filled time series often are 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 estimation error and statistical characteristics of the final time series. The data analysis uses data that are 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. Fill commands often provide a `FillFlag` parameter, which allows filled values to be annotated. The flags can then be displayed in reports and graphs.

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 `SetInputPeriod()` 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

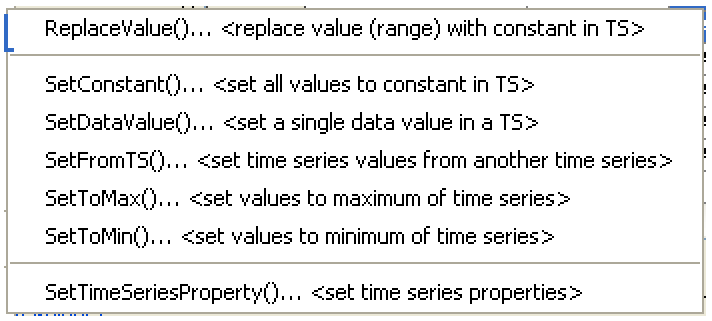
Technique	Command	Typical Use
Constant	`FillConstant()`	Use when missing data can be estimated as a constant. For example, if only the early period of a "regulated" (e.g., reservoir) time series is missing, it may be appropriate to set the values to zero.
Monthly total, daily pattern	`FillDayTSFrom 2MonthTSAnd1DayTS()`	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	`FillFromTS()`	Use non-missing values from a time series to fill missing values in another time series.
Historical Monthly Average	`FillHist MonthAverage()`	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).
Historical Year Average	`FillHist YearAverage()`	Use with yearly time series to estimate missing data as the average of annual values.
Interpolation	`FillInterpolate()`	Use to estimate missing data by interpolating between non-missing values. For example, use to estimate reservoir level changes.
Mixed Station	`FillMixedStation()`	This command tries various combinations of `FillRegression()` and `FillMove2()`

Technique	Command	Typical Use
		parameters with time series at different locations, to use the best combination.
Maintenance of Variance	`FillMOVE1()`	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	`FillMOVE2()`	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.
Historical Pattern Averages	`FillPattern()`	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 `SetPatternFile()` control command also be used.
Prorate	`FillProrate()`	Fill a time series by prorating known values with another time series.
Regression	`FillRegression()`	Use to estimate missing data by using ordinary least squares regression. For example, use to estimate streamflow from a related gage.
Repeat	`FillRepeat()`	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	`FillUsing DiversionComments()`	This command is only available with the HydroBase input type and uses diversion comments and the “not in use” flag 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 data and properties. Unlike fill commands, set commands reset values regardless of whether the values were missing in the time series.

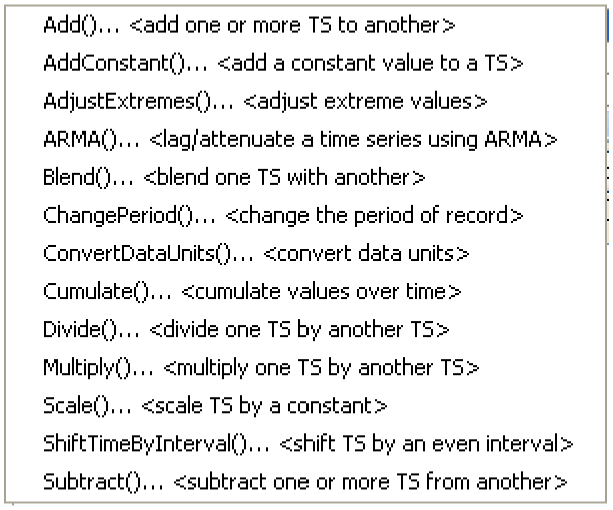

ReplaceValue()... <replace value (range) with constant in TS>
SetConstant()... <set all values to constant in TS>
SetDataValue()... <set a single data value in a TS>
SetFromTS()... <set time series values from another time series>
SetToMax()... <set values to maximum of time series>
SetToMin()... <set values to minimum of time series>
SetTimeSeriesProperty()... <set time series properties>

Menu_Commands_SetTimeSeries

Commands...Set Time Series Contents Menu

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>
ChangePeriod()... <change the period of record>
ConvertDataUnits()... <convert data units>
Cumulate()... <cumulate values over time>
Divide()... <divide one TS by another TS>
Multiply()... <multiply one TS by another TS>
Scale()... <scale TS by a constant>
ShiftTimeByInterval()... <shift TS by an even interval>
Subtract()... <subtract one or more TS from another>

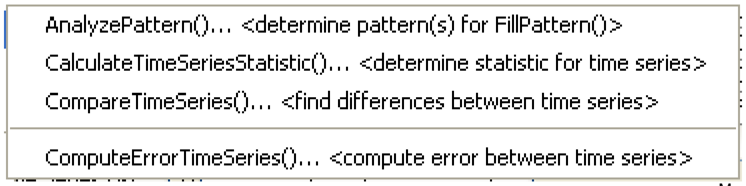
Menu_Commands_Manipulate

Commands...Manipulate Time Series Menu

Because the fundamental nature of the time series (e.g., data type, interval) is not changed by using these commands, the commands do not result in the creation of a new time series but change the data values in existing time series. Manipulation commands typically add comments to the time series history, which can be viewed with time series properties. If it is necessary to create a separate time series to contain the result of a manipulation, use a `Copy()`, `NewTimeSeries()`, or other command to create a "receiving" time series, and then manipulate this new time series.

4.7 Analyze Time Series

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

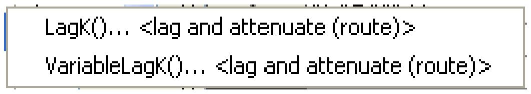

AnalyzePattern()… <determine pattern(s) for FillPattern()>
CalculateTimeSeriesStatistic()… <determine statistic for time series>
CompareTimeSeries()… <find differences between time series>
ComputeErrorTimeSeries()… <compute error between time series>

Menu_Commands_AnalyzeTimeSeries

Commands…Analyze Time Series Menu

4.8 Models

The ***Commands…Models – Routing*** menu inserts commands that perform tasks that are more complex than simple data processes. TSTool does not retrieve or save model states, leaving the handling of states to each model. Currently, the following routing models are available:


LagK()… <lag and attenuate (route)>
VariableLagK()… <lag and attenuate (route)>

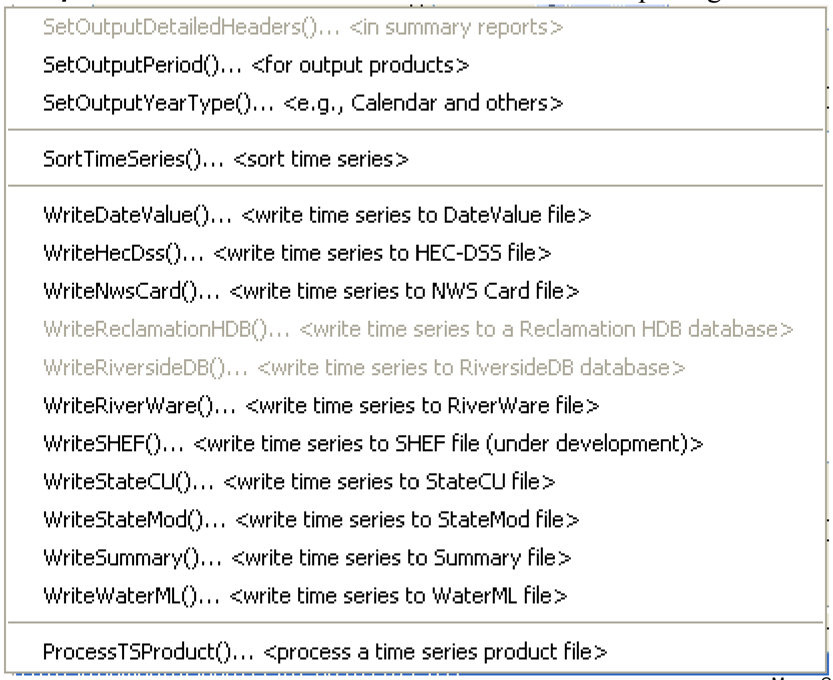
Menu_Commands_Models

Commands…Models Menu

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

4.9 Output Time Series

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


SetOutputDetailedHeaders()… <in summary reports>
SetOutputPeriod()… <for output products>
SetOutputYearType()… <e.g., Calendar and others>
SortTimeSeries()… <sort time series>
WriteDateValue()… <write time series to DateValue file>
WriteHecDss()… <write time series to HEC-DSS file>
WriteNwsCard()… <write time series to NWS Card file>
WriteReclamationHDB()… <write time series to a Reclamation HDB database>
WriteRiversideDB()… <write time series to RiversideDB database>
WriteRiverWare()… <write time series to RiverWare file>
WriteSHEF()… <write time series to SHEF file (under development)>
WriteStateCU()… <write time series to StateCU file>
WriteStateMod()… <write time series to StateMod file>
WriteSummary()… <write time series to Summary file>
WriteWaterML()… <write time series to WaterML file>
ProcessTSProduct()… <process a time series product file>

Menu_Commands_OutputTimeSeries

Commands...Output Time Series Menu

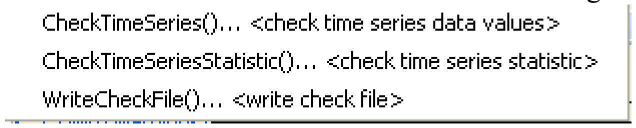
Commands that set global configuration values (e.g., output period) are listed at the start of the menu. Output commands are shown for input types and datastores that are enabled. The `ProcessTSProduct()` command is used to automate the production of graphs.

Using the output commands allows the results of processing to be saved but does increase 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.

4.10 Commands to Check Time Series

The ***Commands…Check Time Series*** menu inserts commands for checking time series.


CheckTimeSeries()… <check time series data values>
CheckTimeSeriesStatistic()… <check time series statistic>
WriteCheckFile()… <write check file>

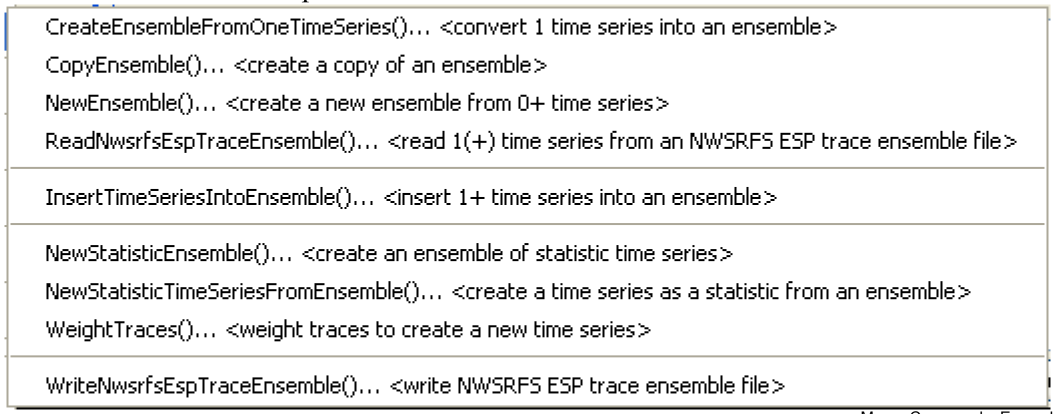
Menu_Commands_CheckTimeSeries

Commands...Check Time Series Menu

These commands can be used for quality control on raw data and processed time series. A summary of check results can be written to a file to preserve an artifact of data processing.

4.11 Commands for Ensemble Processing

The ***Commands…Ensemble Processing*** menu provides commands specific to time series ensembles. These commands can only be used with ensembles. However, many commands available in menus described above can be used to process ensembles by processing all of the time series in the ensemble. See the `TSList=EnsembleID` parameter in commands.


CreateEnsembleFromOneTimeSeries()… <convert 1 time series into an ensemble>
CopyEnsemble()… <create a copy of an ensemble>
NewEnsemble()… <create a new ensemble from 0+ time series>
ReadNwsrfsEspTraceEnsemble()… <read 1(+) time series from an NWSRFS ESP trace ensemble file>
InsertTimeSeriesIntoEnsemble()… <insert 1+ time series into an ensemble>
NewStatisticEnsemble()… <create an ensemble of statistic time series>
NewStatisticTimeSeriesFromEnsemble()… <create a time series as a statistic from an ensemble>
WeightTraces()… <weight traces to create a new time series>
WriteNwsrfsEspTraceEnsemble()… <write NWSRFS ESP trace ensemble file>

Menu_Commands_EnsembleProcessing

Commands…Ensemble Processing Menu

4.12 Commands for Spatial Data Processing

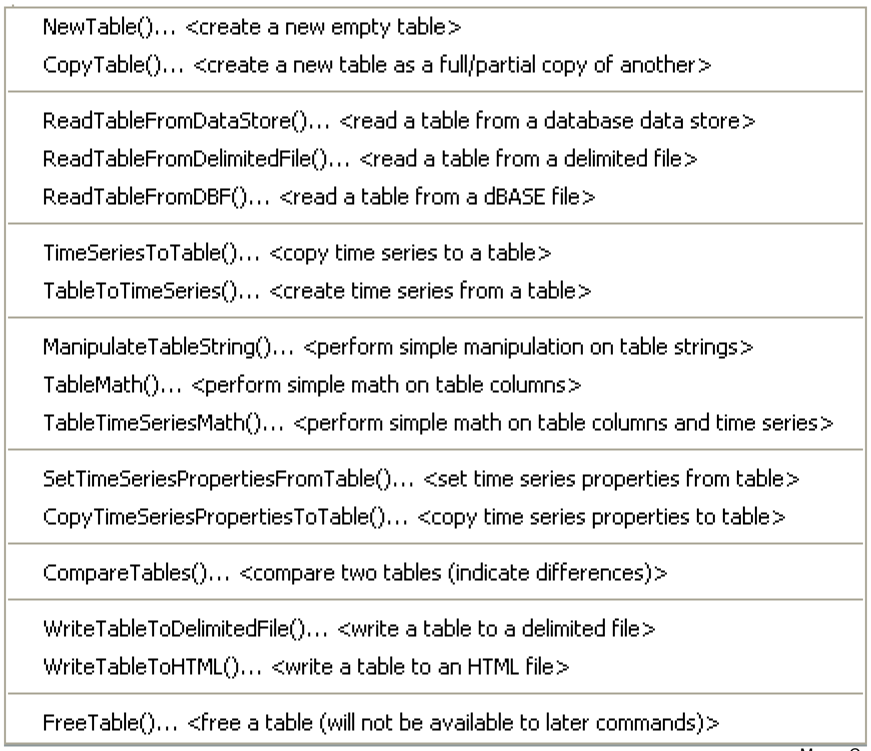
The ***Commands…Spatial Processing*** menu provides commands specific to spatial data processing. These commands are under development.

4.13 Commands for Spreadsheet Processing

The ***Commands…Spreadsheet Processing*** menu provides commands specific to spreadsheet processing. These commands are under development.

4.14 Commands for Table Processing

The ***Commands…Table Processing*** menu provides commands specific to table processing. Tables are defined as row/column data (e.g., from delimited files or databases) where comments can be present in the header and data records, the header defines labels for columns, and columns contain consistent data types (i.e., a column has all dates, all integers, all strings, all floating point numbers). Table rows can be related to a time series by using time series properties such as the location part of the time series identifier.

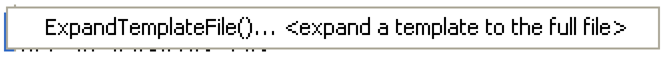

NewTable()… <create a new empty table>
CopyTable()… <create a new table as a full/partial copy of another>
ReadTableFromDataStore()… <read a table from a database data store>
ReadTableFromDelimitedFile()… <read a table from a delimited file>
ReadTableFromDBF()… <read a table from a dBASE file>
TimeSeriesToTable()… <copy time series to a table>
TableToTimeSeries()… <create time series from a table>
ManipulateTableString()… <perform simple manipulation on table strings>
TableMath()… <perform simple math on table columns>
TableTimeSeriesMath()… <perform simple math on table columns and time series>
SetTimeSeriesPropertiesFromTable()… <set time series properties from table>
CopyTimeSeriesPropertiesToTable()… <copy time series properties to table>
CompareTables()… <compare two tables (indicate differences)>
WriteTableToDelimitedFile()… <write a table to a delimited file>
WriteTableToHTML()… <write a table to an HTML file>
FreeTable()… <free a table (will not be available to later commands)>

Menu_Commands_TableProcessing

Commands…Table Processing Menu

4.15 Commands for Template Processing

The ***Commands…Template Processing*** menu provides commands specific to template processing.



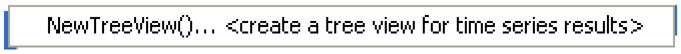
Menu_Commands_TemplateProcessing

Commands…Template Processing Menu

Templates are text files that can be expanded to reflect dynamic content. For example, a template command file can be used to repeat a block of commands for many time series. The documentation for the `ExpandTemplateFile()` command provides examples of how templates can be used.

4.16 Commands for View Processing

The ***Commands…View Processing*** menu provides commands specific to view processing.



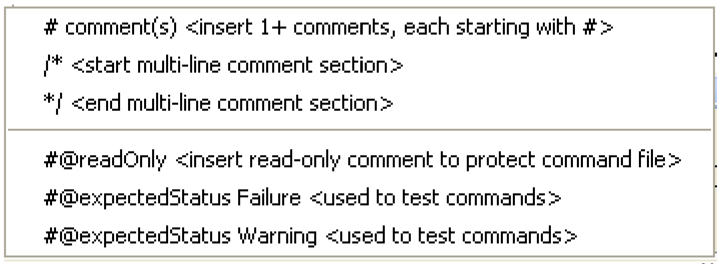
Menu_Commands_ViewProcessing

Commands…View Processing Menu

TSTool lists time series results in the order that time series are created by commands. However, this can lead to very long lists of time series that are difficult to review. The `NewTreeView()` command allows time series results to be organized in a way that is more appropriate. Other views may be implemented in the future to facilitate viewing results.

4.17 General Commands – Comments

The ***Commands…General – Comments*** menu provides choices to insert comments.



# comment(s) <insert 1+ comments, each starting with #>
/* <start multi-line comment section>
*/ <end multi-line comment section>
#@readOnly <insert read-only comment to protect command file>
#@expectedStatus Failure <used to test commands>
#@expectedStatus Warning <used to test commands>

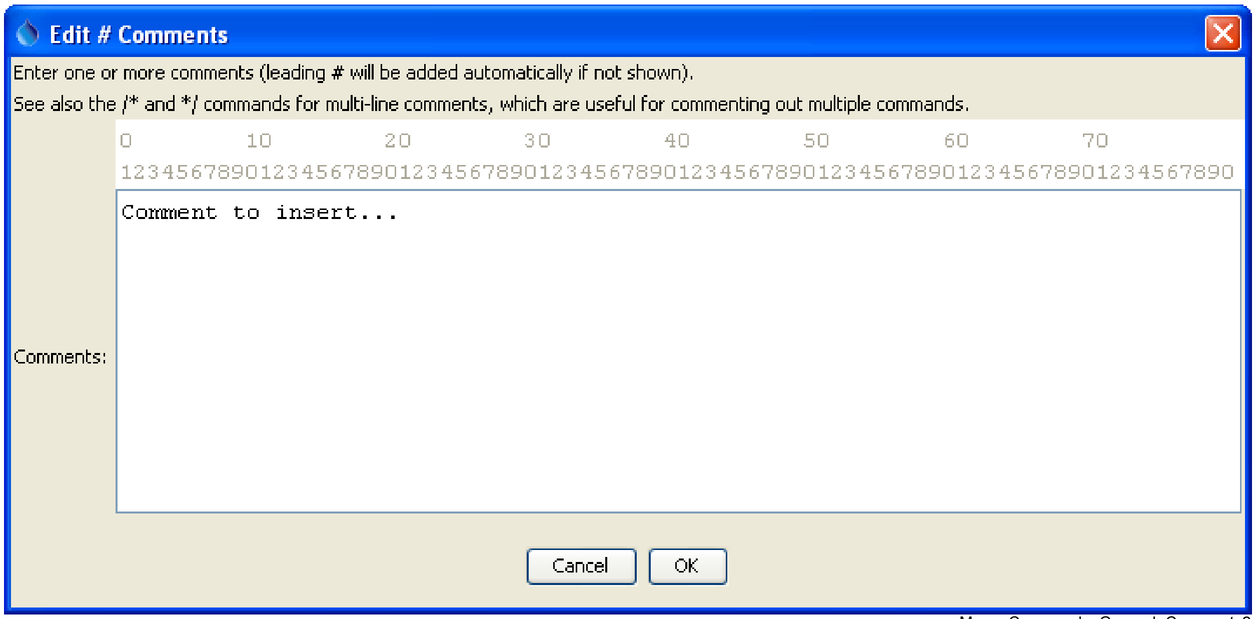
Menu_Commands_General_Comments

Commands…General – Comments Menu

Comments are treated as special commands. The `#` character indicates a single-line comment. The `/*` and `*/` commands indicate multi-line comments (for example to disable blocks of commands without removing them from the command file). The `#@readOnly` comment is used to protect a command file – TSTool will warn if an attempt is made to save the file. For example, this special comment is useful to protect old command files that are archived and should not be changed, even if TSTool command syntax changes.

4.17.1 Inserting # Comments

Comments are inserted by selecting a line in the ***Commands*** list and then selecting ***Commands…General – Comments…# 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.



Enter one or more comments (leading # will be added automatically if not shown).
See also the /* and */ commands for multi-line comments, which are useful for commenting out multiple commands.

```
0         10        20        30        40        50        60        70
1234567890123456789012345678901234567890123456789012345678901234567890123456789
```

Comments:

```
Comment to insert...
```

Menu_Commands_General_Comments2

Comment Editor

4.17.2 Start /* */ Comments

Multiple commands can be commented using the following notation:

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

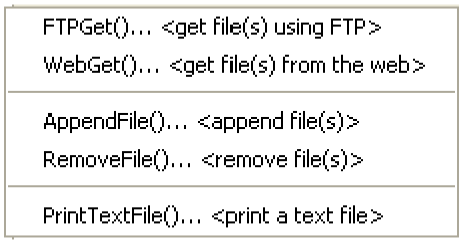
This syntax is consistent with a number of programming languages, including C, C++, and Java, and can be used to quickly disable multiple commands. Use the ***Commands…General – Comments…/* <start comment>*** menu to start a comment above the selected command. Matching start and end comments should be inserted. See also the `exit` control command. Currently there is no way to edit a block of commented code. The notation is meant to be used to comment large blocks of commands, for example during troubleshooting.

4.17.3 End /* */ Comments

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

4.18 General Commands – File Handling

The ***Commands…General – File Handling*** menu provides choices to insert commands that process files.


FTPGet()… <get file(s) using FTP>
WebGet()… <get file(s) from the web>
AppendFile()… <append file(s)>
RemoveFile()… <remove file(s)>
PrintTextFile()… <print a text file>

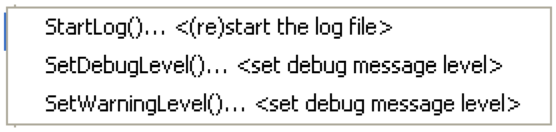
Menu_Commands_General_FileHandling

Commands…General – File Handling Menu

The `RemoveFile()` command is often used in testing. The `FTPGet()` and `WebGet()` commands are used to retrieve files from the internet.

4.19 General Commands – Logging

The ***Commands…General – Logging*** menu provides choices to insert commands used in logging. It is recommended that each command file use a `StartLog()` command as the first command, to create a log file that can facilitate troubleshooting and reviewing work at a later time. Setting the debug and warning level with commands can facilitate troubleshooting specific command logic.

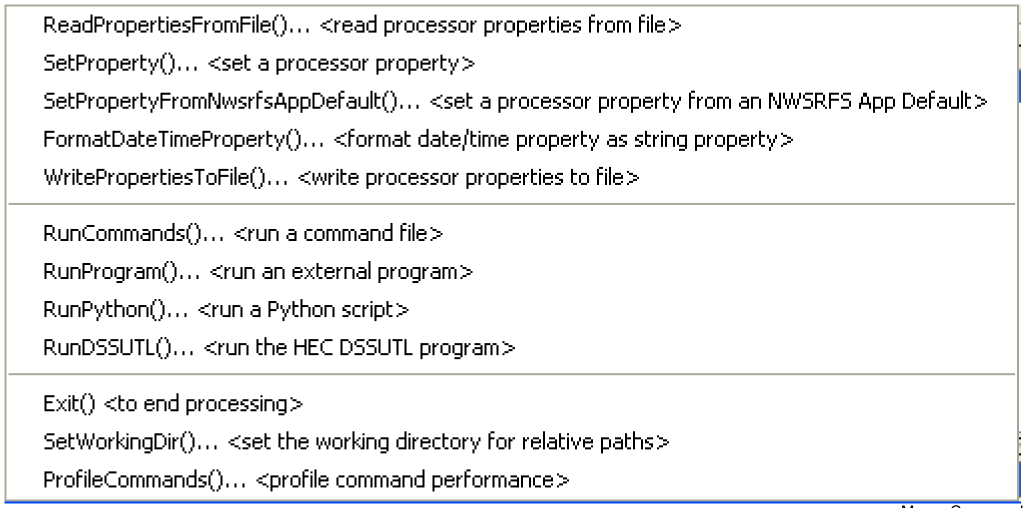

StartLog()… <(re)start the log file>
SetDebugLevel()… <set debug message level>
SetWarningLevel()… <set debug message level>

Menu_Commands_General_Logging

Commands…General – Logging Menu

4.20 General Commands – Running Commands and External Software

The ***Commands…General - Running*** menu provides choices to insert commands related to processing commands and external programs.


ReadPropertiesFromFile()… <read processor properties from file>
SetProperty()… <set a processor property>
SetPropertyFromNwsrfsAppDefault()… <set a processor property from an NWSRFS App Default>
FormatDateTimeProperty()… <format date/time property as string property>
WritePropertiesToFile()… <write processor properties to file>
RunCommands()… <run a command file>
RunProgram()… <run an external program>
RunPython()… <run a Python script>
RunDSSUTL()… <run the HEC DSSUTL program>
Exit() <to end processing>
SetWorkingDir()… <set the working directory for relative paths>
ProfileCommands()… <profile command performance>

Menu_Commands_General_Run

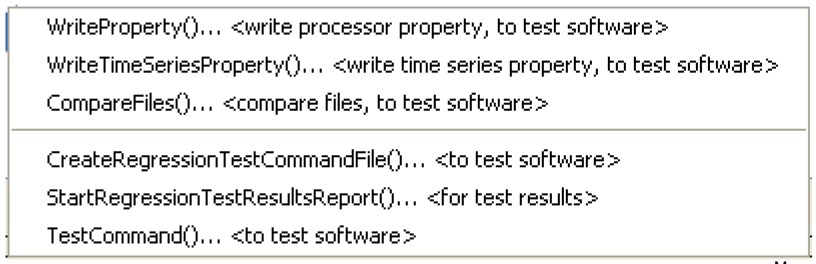
Commands…General – Running Menu

The command processor uses properties to manage controlling information. These properties can be set with commands to facilitate overall workflow logic, for example to allow configuration information to be defined at the start of a command file, and be used by other commands. TSTool commands will continue to be enhanced to utilize these properties.

The `RunCommands()` command can be used to create a "master" command file that runs other command files. This approach is used to create test suites to validate the TSTool software. Commands also are available to run external programs, Python scripts, and the Army Corps of Engineers DSSUTL software, which provides time series processing capabilities.

4.21 General Commands – Test Processing

The ***Commands…General – Test Processing*** menu provides choices to insert commands related to software and process testing. A test case can be a simple test (e.g., test of a single command with a specific combination of parameters) or a more complex test (e.g., a test of a command file used to process a data set file). The `CreateRegressionTestCommandFile()` can be used to search a folder and sub-folders for command files matching a pattern (e.g., *Test_*.TSTool*). This will create a master command file that includes `RunCommands()` commands. These commands are used by software developers to create test suites to verify TSTool software functionality and can also be used by software users to verify that a process is certified and gives expected results. Comparing the results from a specific software version with expected results is useful for diagnosing errors.


WriteProperty()… <write processor property, to test software>
WriteTimeSeriesProperty()… <write time series property, to test software>
CompareFiles()… <compare files, to test software>
CreateRegressionTestCommandFile()… <to test software>
StartRegressionTestResultsReport()… <for test results>
TestCommand()… <to test software>

Menu_Commands_General_Testing

Commands…General – Test Processing Menu

The following is an example command file to run the `CreateRegressionTestCommandFile()` command:

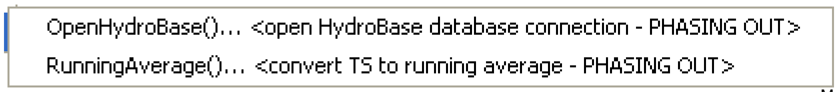
```
#
# Create the regression test runner for the
# TSTool/test/regression/TestSuites/commands_general files.
#
# Only command files that match Test_*.TSTool are included in the output.
# Don't append the generated commands, in order to force the old file to be
# overwritten.
#
CreateRegressionTestCommandFile(SearchFolder="..\..\..\commands\general",
OutputFile="..\run\RunRegressionTest_commands_general.TSTool",Append=False)
```

The following command file is generated from the above and can be run to execute the individual tests. Typically each test uses the `CompareTimeSeries()` or `CompareFiles()` command to generate a warning if results are not as expected.

```
StartRegressionTestResultsReport(
OutputFile="RunRegressionTest_commands_general.TSTool.out.txt")
RunCommands(InputFile="..\..\..\commands\general\add\Test_Add_1.TSTool")
RunCommands(InputFile="..\..\..\commands\general\add\Test_Add_Ensemble_1.TSTool")
/RunCommands(InputFile="..\..\..\commands\general\ChangeInterval\
Test_ChangeInterval_IrregINST_To_3HourINST.TSTool")
… etc. …
```

4.22 Deprecated Commands

The ***Commands…Deprecated Commands*** menu provides commands that are slated for removal at some point in the future. These commands should be avoided and newer alternatives should be used. Refer to the command documentation for information in migrating to other commands.


OpenHydroBase()... <open HydroBase database connection - PHASING OUT>
RunningAverage()... <convert TS to running average - PHASING OUT>

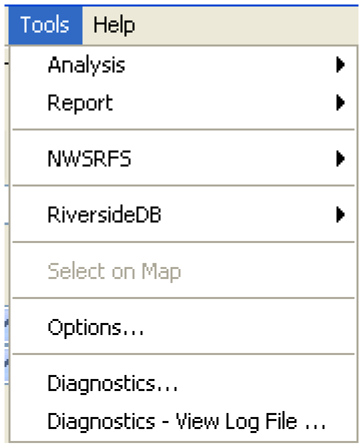
Menu_Commands_Deprecated

Commands…Deprecated Commands Menu

5 Tools

Version 10.20.00, 2013-04-21

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 are interactive, whereas commands can be used in automated processing.



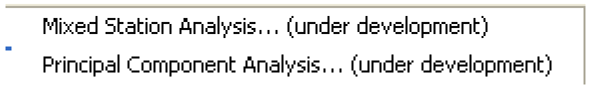
Menu_Tools

Tools Menu

Tools may be specific to a datastore type and consequently some tools may not be available in TSTool for all configurations. Where possible, TSTool functionality is included in commands to facilitate automation.

5.1 Analysis Tools

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


Mixed Station Analysis... (under development)
Principal Component Analysis... (under development)

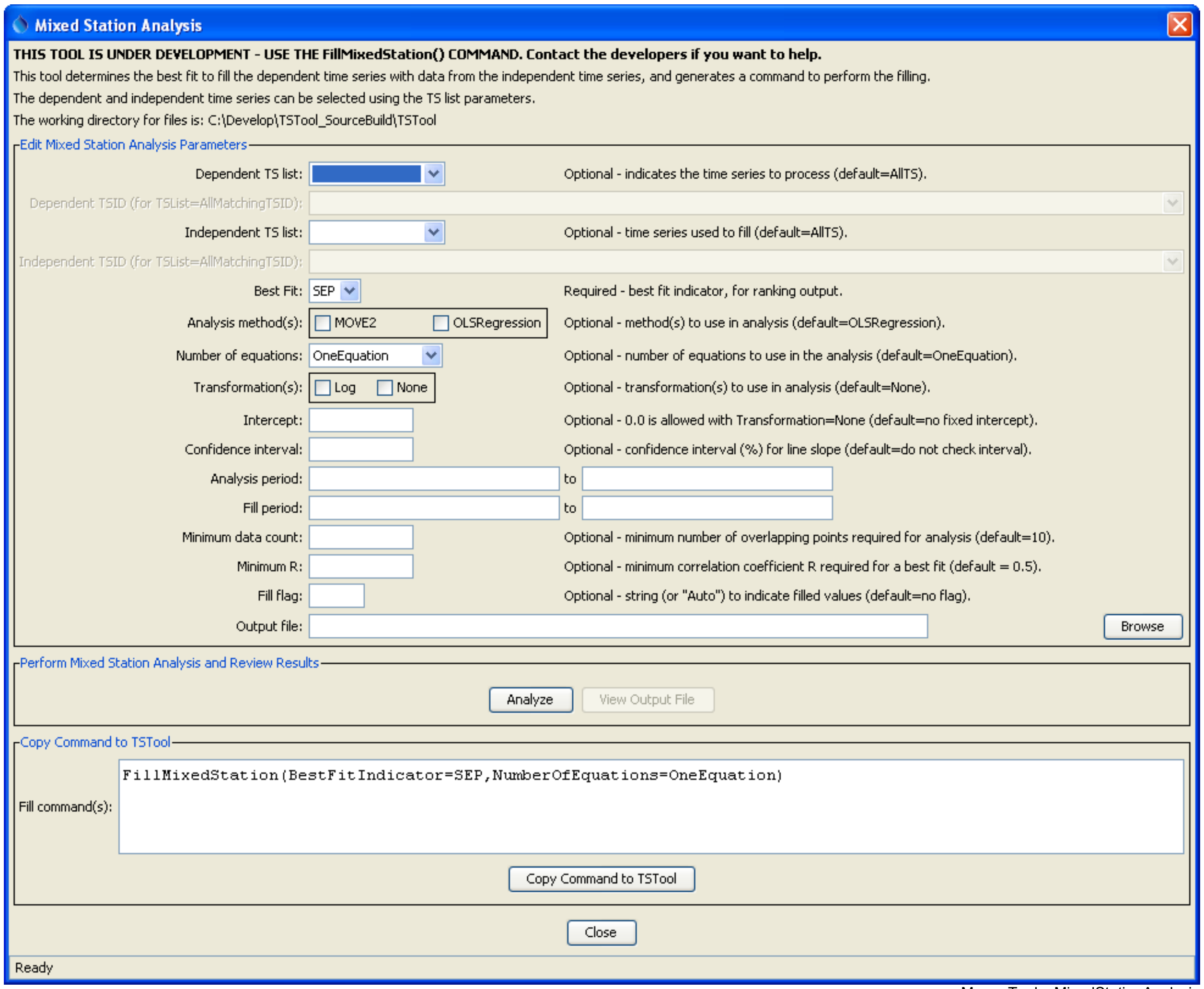
Menu_Tools_Analysis

Tools…Analysis Menu

5.1.1 Mixed Station Analysis

The Mixed Station Analysis tool is under development and not ready for production use. Instead, use the `FillMixedStation()` and `FillRegression()` commands, which provide most of the functionality envisioned by the interactive tool. The following documentation is retained for discussion purposes and to guide future enhancements.

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

THIS TOOL IS UNDER DEVELOPMENT - USE THE FillMixedStation() COMMAND. Contact the developers if you want to help.
This tool determines the best fit to fill the dependent time series with data from the independent time series, and generates a command to perform the filling.
The dependent and independent time series can be selected using the TS list parameters.
The working directory for files is: C:\Develop\TSTool_SourceBuild\TSTool

Edit Mixed Station Analysis Parameters

Dependent TS list:		Optional - indicates the time series to process (default=AllTS).
Dependent TSID (for TSList=AllMatchingTSID):		
Independent TS list:		Optional - time series used to fill (default=AllTS).
Independent TSID (for TSList=AllMatchingTSID):		
Best Fit:	SEP	Required - best fit indicator, for ranking output.
Analysis method(s):	☐ MOVE2 ☐ OLSRegression	Optional - method(s) to use in analysis (default=OLSRegression).
Number of equations:	OneEquation	Optional - number of equations to use in the analysis (default=OneEquation).
Transformation(s):	☐ Log ☐ None	Optional - transformation(s) to use in analysis (default=None).
Intercept:		Optional - 0.0 is allowed with Transformation=None (default=no fixed intercept).
Confidence interval:		Optional - confidence interval (%) for line slope (default=do not check interval).
Analysis period:		to
Fill period:		to
Minimum data count:		Optional - minimum number of overlapping points required for analysis (default=10).
Minimum R:		Optional - minimum correlation coefficient R required for a best fit (default = 0.5).
Fill flag:		Optional - string (or "Auto") to indicate filled values (default=no flag).
Output file:		Browse

Perform Mixed Station Analysis and Review Results

Analyze View Output File

Copy Command to TSTool

Fill command(s):

```
FillMixedStation(BestFitIndicator=SEP,NumberOfEquations=OneEquation)
```

Copy Command to TSTool

Close

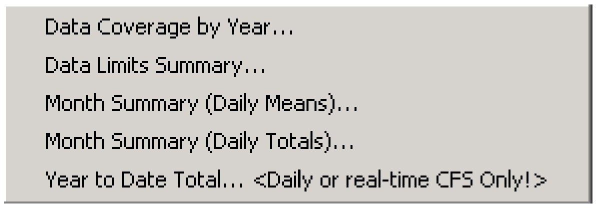
Ready

Menu_Tools_MixedStationAnalysis

Mixed Station Analysis Interface

5.2 Report Tools

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

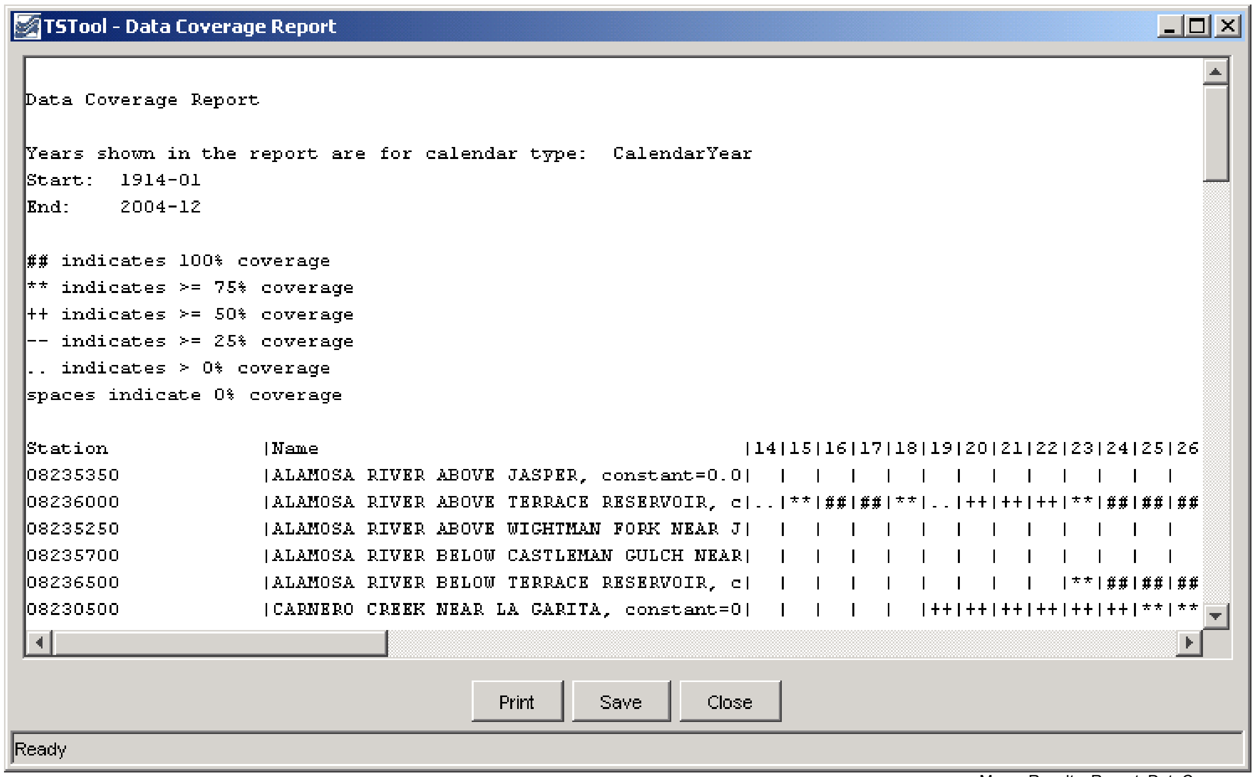

Data Coverage by Year...
Data Limits Summary...
Month Summary (Daily Means)...
Month Summary (Daily Totals)...
Year to Date Total... <Daily or real-time CFS Only!>

Menu_Results_Report

Tools…Report Menu

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.



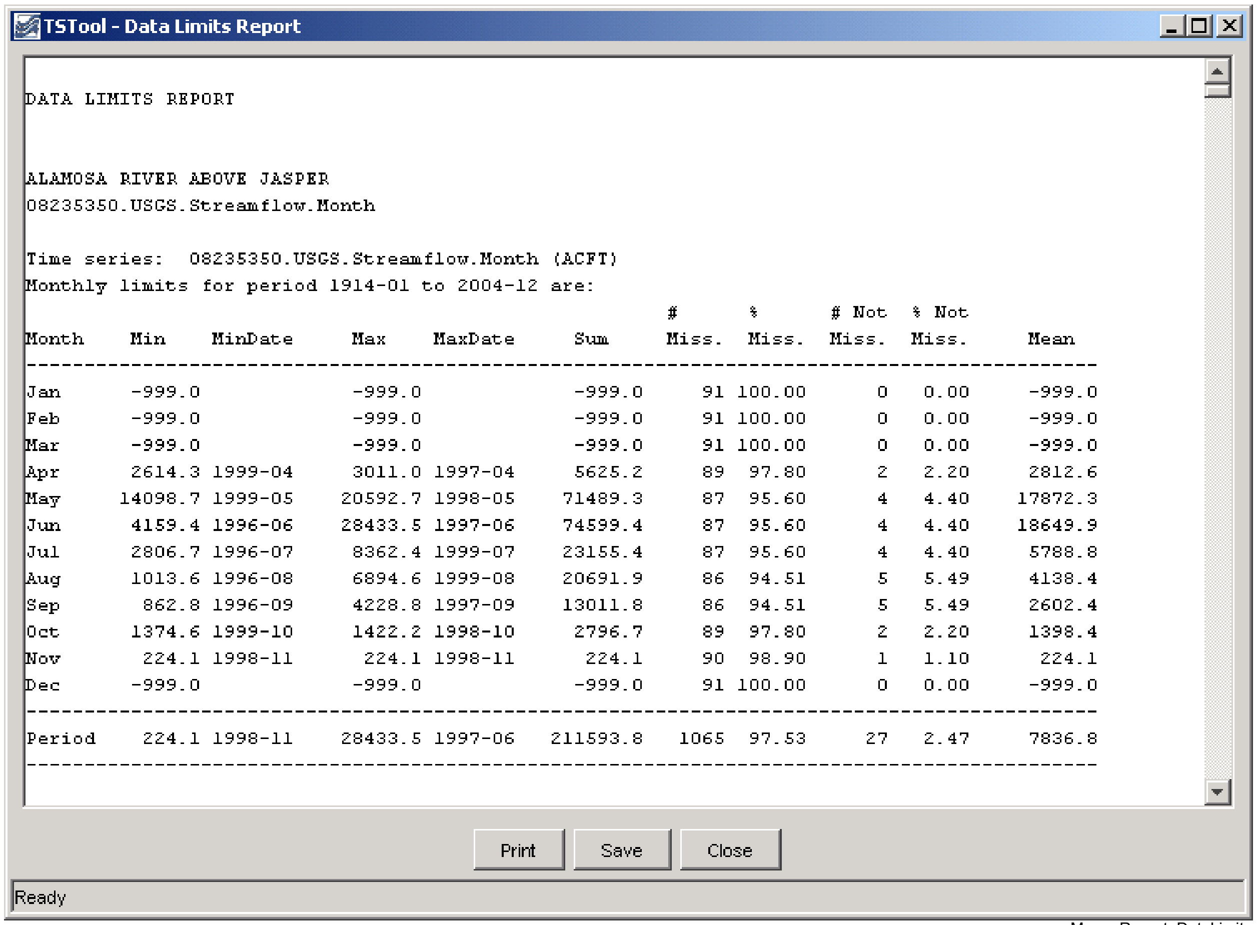
```
Data Coverage Report

Years shown in the report are for calendar type:  CalendarYear
Start:  1914-01
End:    2004-12

## indicates 100% coverage
** indicates >= 75% coverage
++ indicates >= 50% coverage
-- indicates >= 25% coverage
.. indicates > 0% coverage
spaces indicate 0% coverage

Station             |Name                                    |14|15|16|17|18|19|20|21|22|23|24|25|26
08235350            |ALAMOSA RIVER ABOVE JASPER, constant=0.0|  |  |  |  |  |  |  |  |  |  |  |  |
08236000            |ALAMOSA RIVER ABOVE TERRACE RESERVOIR, c|..|**|##|##|**|..|++|++|++|**|##|##|##
08235250            |ALAMOSA RIVER ABOVE WIGHTMAN FORK NEAR J|  |  |  |  |  |  |  |  |  |  |  |  |
08235700            |ALAMOSA RIVER BELOW CASTLEMAN GULCH NEAR|  |  |  |  |  |  |  |  |  |  |  |  |
08236500            |ALAMOSA RIVER BELOW TERRACE RESERVOIR, c|  |  |  |  |  |  |  |  |  |**|##|##|##
08230500            |CARNERO CREEK NEAR LA GARITA, constant=0|  |  |  |  |  |++|++|++|++|++|++|**|**
```

Menu_Results_Report_DataCoverage

Data Coverage by Year Report

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.



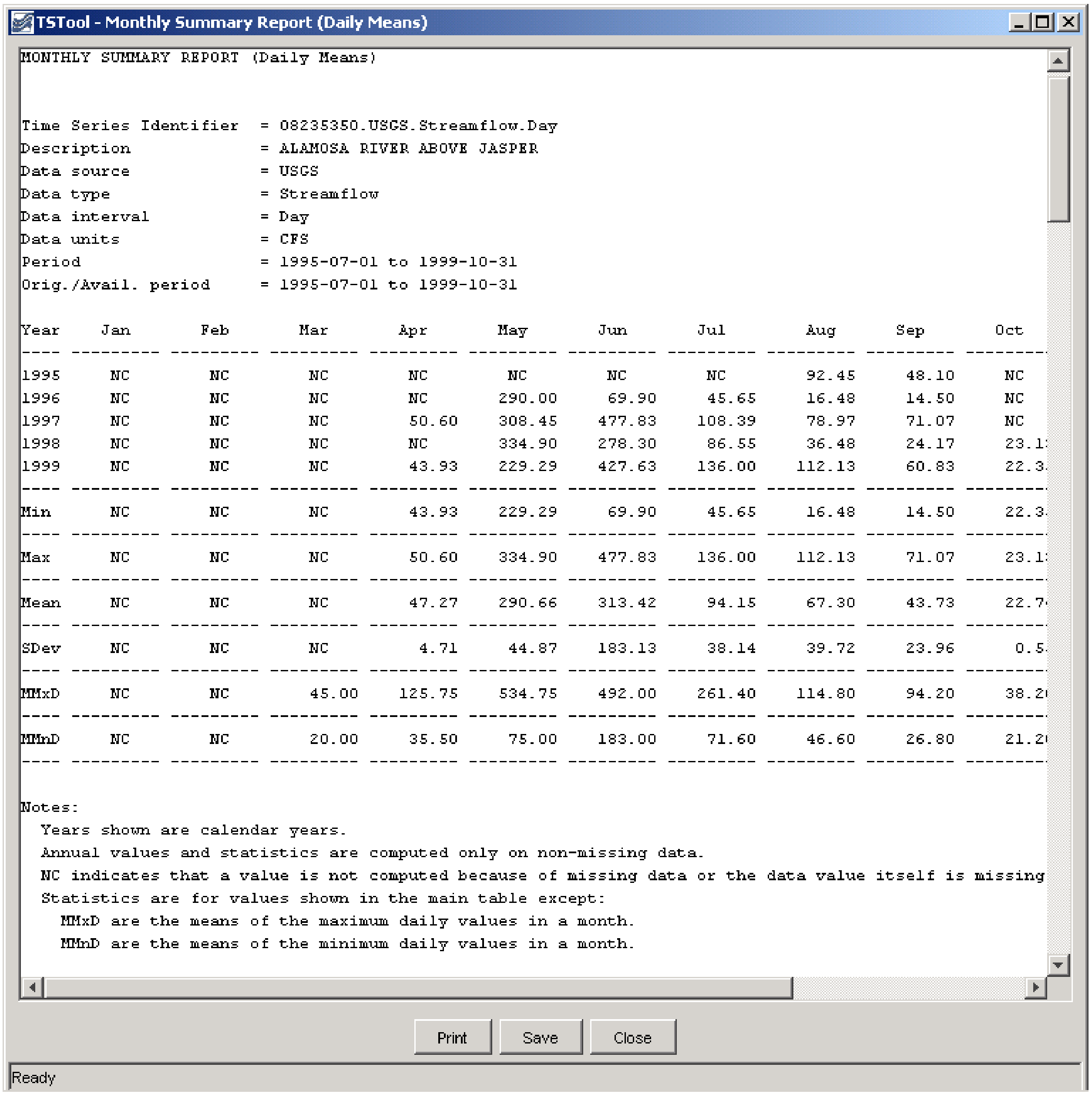
```
DATA LIMITS REPORT


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:
                                                        #      %      # Not  % Not
Month    Min     MinDate      Max    MaxDate      Sum     Miss.  Miss.  Miss.  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
-------------------------------------------------------------------------------------------
```

Menu_Report_DataLimits

Data Limits Summary Report

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.



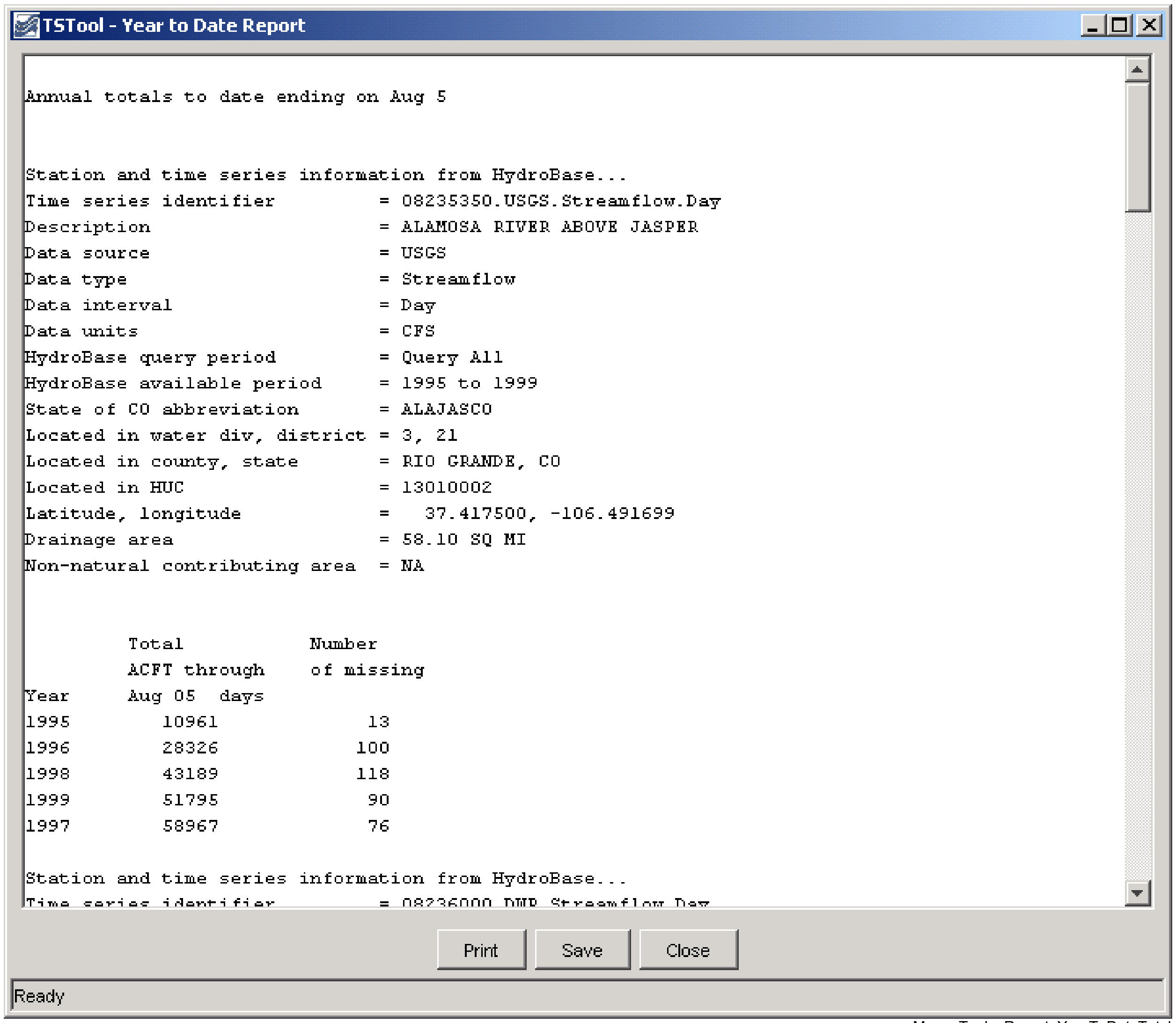
```
MONTHLY SUMMARY REPORT (Daily Means)


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     38.2
---- --------- --------- --------- --------- --------- --------- --------- --------- --------- ---------
MMnD     NC        NC      20.00     35.50     75.00    183.00     71.60     46.60     26.80     21.2
---- --------- --------- --------- --------- --------- --------- --------- --------- --------- ---------


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.
```

Menu_Tools_MonthSummaryDailyMean

Monthly Summary (Daily Mean) Report

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.**



```
Annual totals to date ending on Aug 5


Station and time series information from HydroBase...
Time series identifier          = 08235350.USGS.Streamflow.Day
Description                     = ALAMOSA RIVER ABOVE JASPER
Data source                     = USGS
Data type                       = Streamflow
Data interval                   = Day
Data units                      = CFS
HydroBase query period          = Query All
HydroBase available period      = 1995 to 1999
State of CO abbreviation        = ALAJASCO
Located in water div, district  = 3, 21
Located in county, state        = RIO GRANDE, CO
Located in HUC                  = 13010002
Latitude, longitude             =   37.417500, -106.491699
Drainage area                   = 58.10 SQ MI
Non-natural contributing area   = NA


         Total            Number
         ACFT through     of missing
Year     Aug 05   days
1995        10961             13
1996        28326            100
1998        43189            118
1999        51795             90
1997        58967             76

Station and time series information from HydroBase...
Time series identifier          = 08236000.DWR.Streamflow.Day
```

Menu_Tools_Report_YearToDateTotal

Year to Date Total Report

5.3 Map Tools

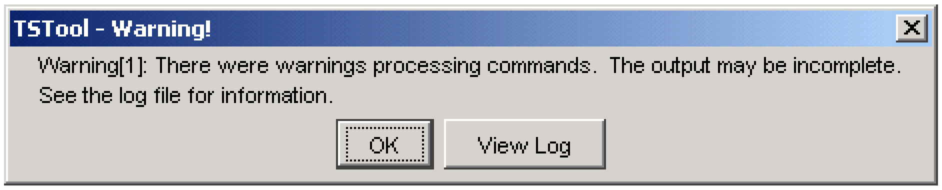
The ***Tools…Select on Map*** menu button is enabled when a map is displayed (using ***View…Map***) and time series are listed in the upper right part of the main window. The locations corresponding to selected time series or all time series in this list can be displayed on the map. See **Chapter 8 –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

5.4 TSTool Options

The ***Tools…Options*** menu provides a way to set TSTool options during the session. Minimal capabilities are provided. Instead, the TSTool environment is typically configured in its configuration file (see the **Installation and Configuration** appendix), data store configuration files, and with commands.

5.5 Diagnostics

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


TSTool - Warning!

Warning[1]: There were warnings processing commands. The output may be incomplete.
See the log file for information.

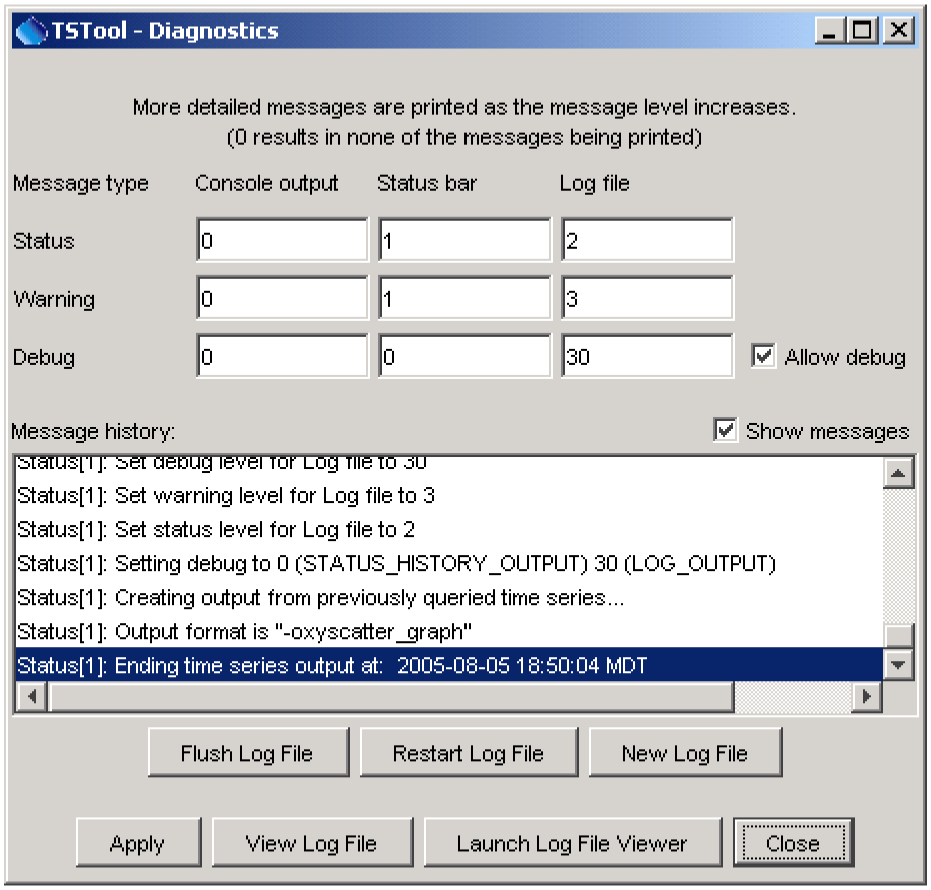
Diagnostics_Warning

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.

5.5.1 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.



More detailed messages are printed as the message level increases.
(0 results in none of the messages being printed)

Message type	Console output	Status bar	Log file	
Status	0	1	2	
Warning	0	1	3	
Debug	0	0	30	✓ Allow debug

Message history: ✓ Show messages

```
Status[1]: Set debug level for Log file to 30
Status[1]: Set warning level for Log file to 3
Status[1]: Set status level for Log file to 2
Status[1]: Setting debug to 0 (STATUS_HISTORY_OUTPUT) 30 (LOG_OUTPUT)
Status[1]: Creating output from previously queried time series...
Status[1]: Output format is "-oxyscatter_graph"
Status[1]: Ending time series output at:  2005-08-05 18:50:04 MDT
```

Diagnostics

Diagnostics Interface

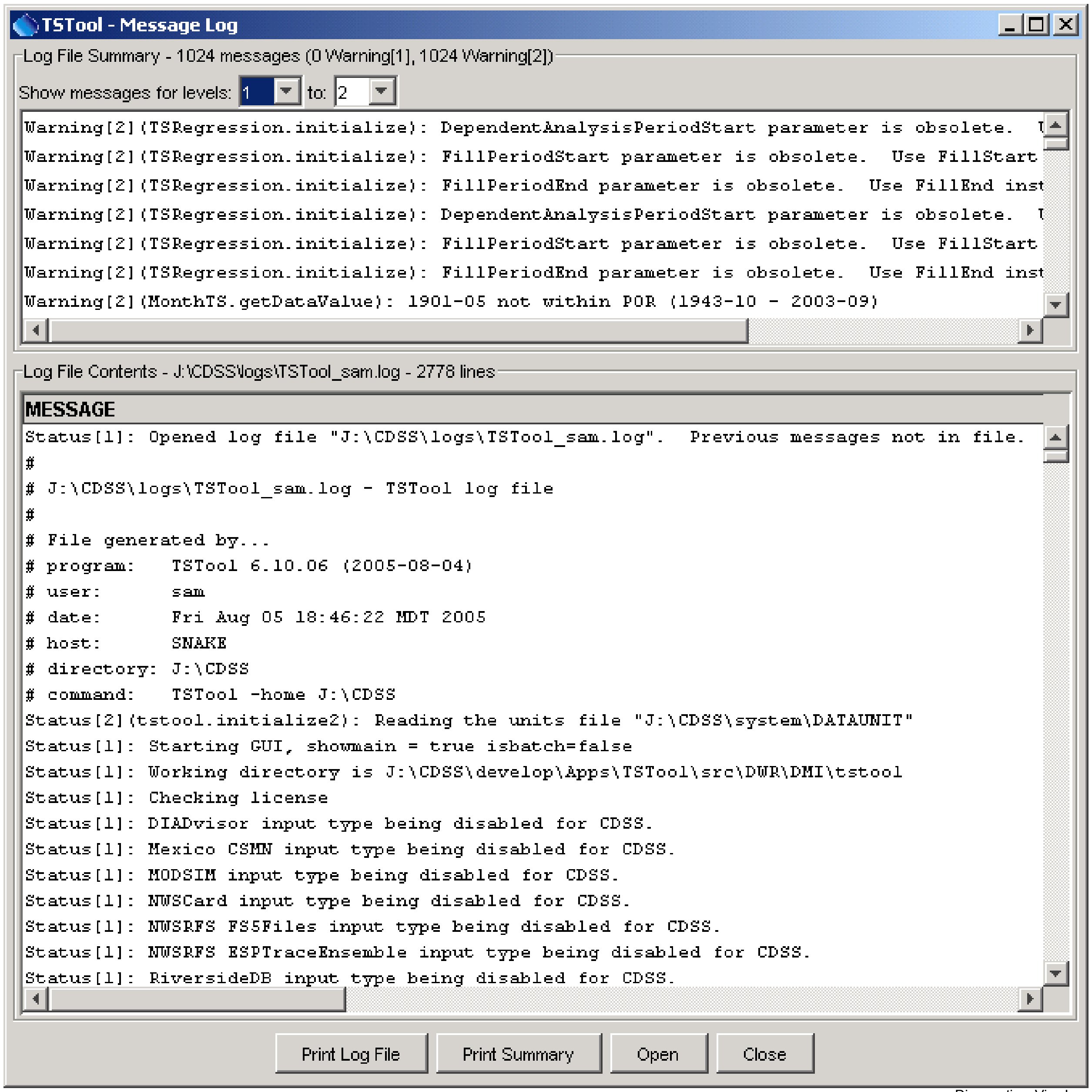
The settings at the top of the dialog are used to specify the level of detail for messages printed to the console window, the status area at the bottom of the main window (and the ***Diagnostics*** dialog), and the log file. The log file contains warning, status, and debug messages, many of which are 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.

5.5.2 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:


TSTool - Message Log
Log File Summary - 1024 messages (0 Warning[1], 1024 Warning[2])
Show messages for levels: 1 to: 2

```
Warning[2](TSRegression.initialize): DependentAnalysisPeriodStart parameter is obsolete.  U
Warning[2](TSRegression.initialize): FillPeriodStart parameter is obsolete.  Use FillStart
Warning[2](TSRegression.initialize): FillPeriodEnd parameter is obsolete.  Use FillEnd inst
Warning[2](TSRegression.initialize): DependentAnalysisPeriodStart parameter is obsolete.  U
Warning[2](TSRegression.initialize): FillPeriodStart parameter is obsolete.  Use FillStart
Warning[2](TSRegression.initialize): FillPeriodEnd parameter is obsolete.  Use FillEnd inst
Warning[2](MonthTS.getDataValue): 1901-05 not within POR (1943-10 - 2003-09)
```

Log File Contents - J:\CDSS\logs\TSTool_sam.log - 2778 lines
MESSAGE

```
Status[1]: Opened log file "J:\CDSS\logs\TSTool_sam.log".  Previous messages not in file.
#
# J:\CDSS\logs\TSTool_sam.log - TSTool log file
#
# File generated by...
# program:   TSTool 6.10.06 (2005-08-04)
# user:      sam
# date:      Fri Aug 05 18:46:22 MDT 2005
# host:      SNAKE
# directory: J:\CDSS
# command:   TSTool -home J:\CDSS
Status[2](tstool.initialize2): Reading the units file "J:\CDSS\system\DATAUNIT"
Status[1]: Starting GUI, showmain = true isbatch=false
Status[1]: Working directory is J:\CDSS\develop\Apps\TSTool\src\DWR\DMI\tstool
Status[1]: Checking license
Status[1]: DIADvisor input type being disabled for CDSS.
Status[1]: Mexico CSMN input type being disabled for CDSS.
Status[1]: MODSIM input type being disabled for CDSS.
Status[1]: NWSCard input type being disabled for CDSS.
Status[1]: NWSRFS FS5Files input type being disabled for CDSS.
Status[1]: NWSRFS ESPTraceEnsemble input type being disabled for CDSS.
Status[1]: RiversideDB input type being disabled for CDSS.
```

Diagnostics_ViewLog

Log File Viewer Window

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 10.00.01, 2011-05-16

This chapter provides examples for reading and manipulating time series data using TSTool. The heading for each section gives an indication of the example purpose. Where appropriate, input and output types are indicated to help users find an appropriate example. General examples are listed first, followed by more complex examples. Additional information can be found in the training materials (see ***Help…View Training Materials***)

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, you may need to select a database and provide a login.
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 ***Results*** area.

5. Use the ***Results*** and ***Tools*** menus to view the time series. 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()` transform the data

7. Repeat steps 4 – 5 to process and view time series.

6.1.2 General – Reproducing an Analysis with a Command File

To reproduce an analysis, save the commands shown in the ***Commands*** list to a command 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 command 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…Command 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 command 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 (the folder where the command file is saved) 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 command files to be portable. TSTool will internally set the working directory that the directory where a command 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 Command 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 commandfile
```

However, it is recommended that command 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. Review the release notes appendix when installing software updates.
2. Be familiar with this TSTool User Manual and, in particular, the command reference.
3. Run an existing command 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).
4. Open an existing command file using the ***File…Open…Command File*** menu . TSTool will attempt to convert commands to new syntax as the file is loaded. Most command files focus on a particular data type and manipulation. Therefore most updates will generally involve only a few changes.

A number of commands have been added/enhanced to promote reuse of command files in both batch and GUI run modes. For example, the `ProcessTSProduct()` command 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, for example, to ensure that periods overlap.

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

6.2.2 Modeling – Processing Reservoir Target (Input=HydroBase, Output=StateMod)

The following example illustrates how to create a monthly reservoir target file for the StateMod model using data from the State of Colorado's HydroBase. Note however that end of month data may not always be available in HydroBase due to data availability and quality control issues. If the data are not available in HydroBase, time series can be read from other sources.

```
# Reservoir target file commands
# Each reservoir needs a minimum (zero) and maximum time series (from HydroBase)
SetOutputPeriod(OutputStart="10/1974",OutputEnd="9/1991")
SetOutputYearType(OutputYearType=Water)
# CBT SHADOW MTN GRAND L
NewTimeSeries(Alias="ShadowMtn",NewTSID="513695.USBR.ResEOM.Month",
    Description="CBT SHADOW MTN GRAND L",Units="AF",InitialValue=0)
513695.USBR.ResEOM.MONTH~HydroBase
# CBT GRANBY RESERVOIR
NewTimeSeries(Alias="Granby",NewTSID="51460.USBR.ResEOM.Month",Units="AF",InitialValue=0)
514620.USBR.ResEOM.MONTH~HydroBase
# DILLON RESERVOIR
NewTimeSeries(Alias="Dillon",NewTSID="364512.USBR.ResEOM.Month",Units="AF",InitialValue=0)
364512.DWB.ResEOM.MONTH~HydroBase
# GREEN MOUNTAIN RESERVIOR
NewTimeSeries(Alias="GreenMtn",NewTSID="363543.USBR.ResEOM.Month",Units="AF",InitialValue=0)
363543.USBR.ResEOM.MONTH~HydroBase
# RIFLE GAP RESERVOIR
NewTimeSeries(Alias="RifleGap",NewTSID="393508.USBR.ResEOM.Month",Units="AF",InitialValue=0)
393508.USBR.ResEOM.MONTH~HydroBase
WriteStateMod(TSList=AllTS,OutputFile="coloup.tar")
```

ExamplesOfUse/Reservoir_EOM/Example_Reservoir_EOM.TSTool

6.2.3 Modeling – Filling Reservoir End of Month Contents with a Pattern File (Input=HydroBase, Output=StateMod)

The following example illustrates a command file for creating a StateMod reservoir historical end of month file, using pattern filling.

```
# eom.commands.TSTool
#
# commands in this file either pull historical EOM contents from the CRDSS database
# (i.e. Rifle Gap) or from user-defined *.stm files
#
SetOutputPeriod(OutputStart="10/1908",OutputEnd="09/2005")
SetOutputYearType(OutputYearType=Water)
ReadPatternFile(PatternFile="..\Diversions\fill2005.pat")
#
# GREEN MOUNTAIN RESERVOIR
363543...MONTH~StateMod~363543.stm
#
# UPPER BLUE RESERVOIR (ConHoosier)
# Data from HydroBase is used to better represent actual opperations of the reservoir in the
cm2005 update
# rather than setting the contents to its maximum as in previous model versions.
363570.DWR.ResMeasStorage.Day~HydroBase
NewEndOfMonthTSFromDayTS(DayTSID="363570.DWR.ResMeasStorage.Day",Alias="ConHoosier363570",
    Bracket=16)
Free(TSList=LastMatchingTSID,TSID="363570.DWR.ResMeasStorage.Day")
FillPattern(TSList=LastMatchingTSID,TSID="ConHoosier363570",PatternID="09037500")
SetConstant(TSList=LastMatchingTSID,TSID="ConHoosier363570",ConstantValue=0,SetEnd="03/1962")
FillInterpolate(TSList=LastMatchingTSID,TSID="ConHoosier363570",MaxIntervals=0,
    Transformation=None)
#
# CLINTON GULCH RESERVOIR
# Data from HydroBase is used to better represent actual opperations of the reservoir in the
cm2005 update
# rather than setting the contents to its maximum as in previous model versions.
363575.DWR.ResMeasStorage.Day~HydroBase
NewEndOfMonthTSFromDayTS(DayTSID="363575.DWR.ResMeasStorage.Day",Alias="ClintonGulch363575",
    Bracket=16)
Free(TSList=LastMatchingTSID,TSID="363575.DWR.ResMeasStorage.Day")
FillInterpolate(TSList=AllMatchingTSID,TSID="ClintonGulch363575",MaxIntervals=0)
FillPattern(TSList=LastMatchingTSID,TSID="ClintonGulch363575",PatternID="09037500")
SetConstant(TSList=LastMatchingTSID,TSID="ClintonGulch363575",ConstantValue=0,
    SetEnd="03/1977")
FillInterpolate(TSList=LastMatchingTSID,TSID="ClintonGulch363575",MaxIntervals=0,
    Transformation=None)
#
# DILLON RESERVOIR
364512...MONTH~StateMod~364512.stm
#
# WOLCOTT RESERVOIR
373639...MONTH~StateMod~zero.stm
… similar commands for other reservoirs omitted…
#
# Fill remaining missing data with historical averages
FillHistMonthAverage(TSList=AllTS)
#
WriteStateMod(TSList=AllTS,OutputFile="..\statemod\cm2005.eom",Precision=0)
```

Example_Reservoir_EOM.TSTool - From Colorado_1_2007 CDSS data set

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

It may be desirable to read a file containing a list of station/structure identifiers and process the corresponding time series. The following example illustrates a command file to use a list to read time series from HydroBase and output a StateMod data file.

```
#
# Example to illustrate how a delimited list of location identifiers can be used
# to create time series identifiers for processing.  This example creates
# time series identifiers to read from the State of Colorado's HydroBase, and
# outputs to a StateMod model file.
#
CreateFromList(ListFile="structure_list.txt",IDCol=1,DataSource="DWR",DataType="DivTotal",In
terval="Month",InputType="HydroBase",IfNotFound=Ignore)
SetOutputYearType(OutputYearType=Calendar)
WriteStateMod(TSList=AllTS,OutputFile="structure_list.stm")
```

ExamplesOfUse/CreateFromList/Example_CreateFromList

where the list file contains the following:

```
#
# Structures for which to process data
#
# WDID - State of Colorado Water District Identifier
# Name - Structure name (from HydroBase)
#
"WDID","Name"
0100501,EMPIRE DITCH
0100503,RIVERSIDE CANAL
0100504,ILLINOIS DITCH
```

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 State of Colorado’s 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 had 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(OutputStart="1950",OutputEnd="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(TSList=AllMatchingTSID,TSID="5484.NOAA.FrostDateL28S.Year",
    IndependentTSList=AllMatchingTSID,IndependentTSID="5487.NOAA.FrostDateL28S.Year")
FillFromTS(TSList=AllMatchingTSID,TSID="5484.NOAA.FrostDateL32S.Year",
    IndependentTSList=AllMatchingTSID,IndependentTSID="5487.NOAA.FrostDateL32S.Year")
FillFromTS(TSList=AllMatchingTSID,TSID="5484.NOAA.FrostDateF32F.Year",
    IndependentTSList=AllMatchingTSID,IndependentTSID="5487.NOAA.FrostDateF32F.Year")
FillFromTS(TSList=AllMatchingTSID,TSID="5484.NOAA.FrostDateF28F.Year",
    IndependentTSList=AllMatchingTSID,IndependentTSID="5487.NOAA.FrostDateF28F.Year")
Free(TSList=AllMatchingTSID,TSID="5487*")
#
# _________________________________________________________
FillHistYearAverage(TSList=AllMatchingTSID,TSID="*")
#
# _________________________________________________________
WriteStateCU(OutputFile="..\StateCU\Frost2002.stm")
```

ExamplesOfUse/FrostDates/Example_FrostDates.TSTool

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 4.3 – 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. Data filling involves a number of steps:

1. Initial review of the data (e.g., using the ***Tools…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 (see **Chapter 7** for limitations of the map interface).
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***).

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
#
#
# Generate some sample data for the example described above:
# ts1 has observed values from 1900-2000
# ts2 has observed values from 1920 to 1950 and regulated thereafter
#
# Although data are generated below, they could be read from files or a
# database.  In this case, the SetOutputPeriod() command might need to be used
# to ensure that the final result is for a required period.
#
NewPatternTimeSeries(Alias="ts1",NewTSID="ts1..Streamflow.Day",
    Description="Time series 1, all unregulated",SetStart="1900-01-01",
    SetEnd="2000-12-31",Units="CFS",PatternValues="1,2,4,7,12,6,2,1.5")
NewPatternTimeSeries(Alias="ts2",NewTSID="ts2..Streamflow.Day",
    Description="Time series 1, unregulated from 1920 to 1950, regulated after",
    SetStart="1900-01-01",SetEnd="2000-12-31",Units="CFS",PatternValues="2,3,6,10,15,3,2.5,2")
#
# Clear out the period 1919- in ts2 because it was not recorded in our example.
SetConstant(TSList=AllMatchingTSID,TSID="ts2",ConstantValue=-999,SetEnd="1919-12-31")
# Clear out the period 1951+ in ts2 because it is regulated and needs to
# be filled with the result of unregulated MOVE2 analysis.
SetConstant(TSList=AllMatchingTSID,TSID="ts2",ConstantValue=-999,SetStart="1951-01-01")
# Analyze and fill the second time series.  Transform the data to log10 and
# use monthly equations…
FillMOVE2(TSID="ts2",IndependentTSID="ts1",NumberOfEquations=MonthlyEquations,
    DependentAnalysisStart="1920-01-01",DependentAnalysisEnd="1950-12-31",
    IndependentAnalysisStart="1900-01-01",IndependentAnalysisEnd="2000-12-31",FillFlag="M")
```

ExamplesOfUse/Filling/Example_Filling.TSTool

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 Ensemble Examples

The general term *time series ensemble* refers to groups of a time series, often shown in overlapping fashion. Common ways to create ensembles 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 ensembles. Many other commands allow ensembles to be specified to provide a list of time series for processing. TSTool manages ensemble data by using an ensemble identifier and optionally using a trace (sequence) number for each time series, which when processing historical data is typically the starting historical year for the trace. For most functionality, the ensemble is simply a collection of the time series traces in the ensemble.

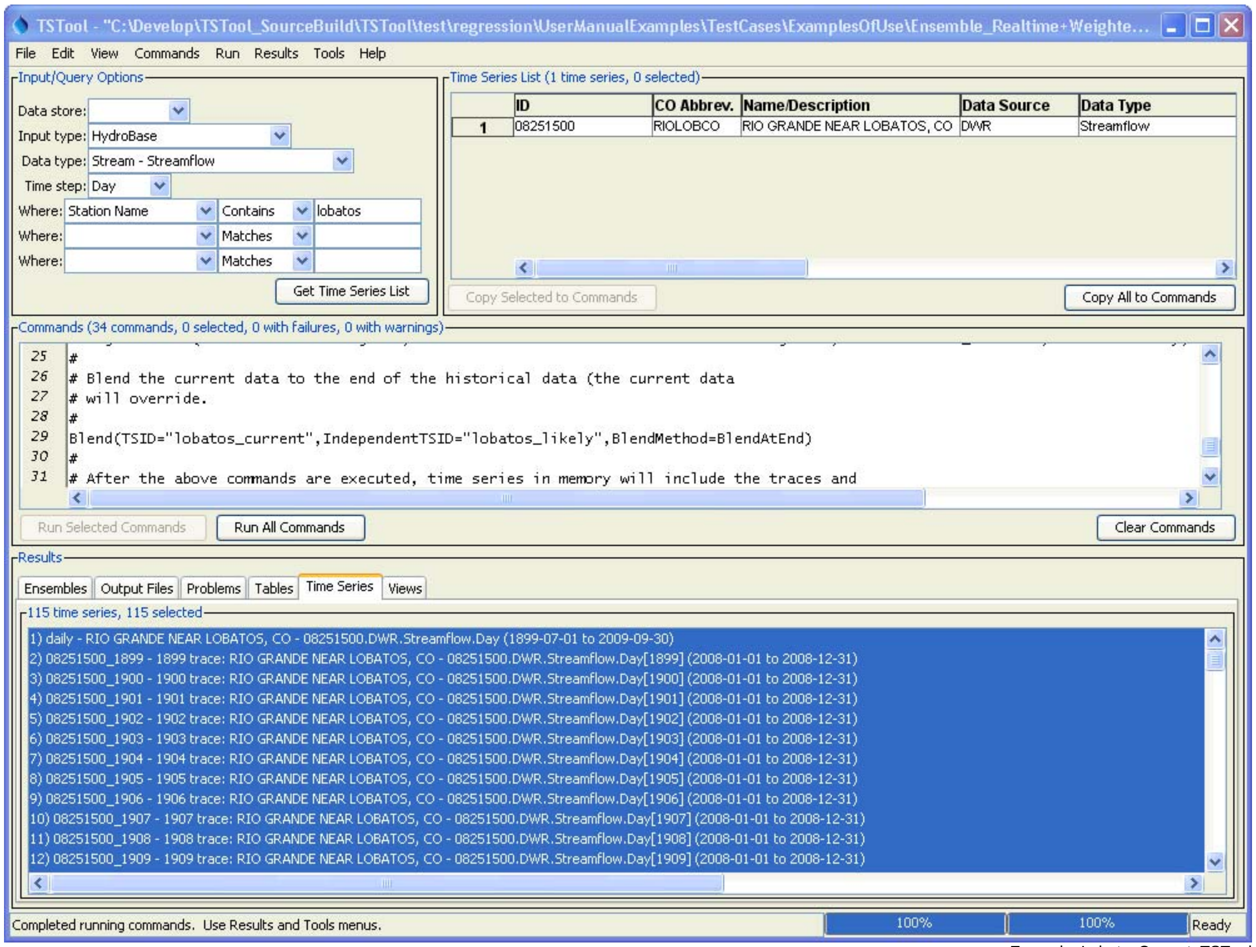
6.3.1 Time Series Traces – Comparing Historical and Current Conditions (Input=HydroBase)

The following command file illustrates how historical 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.

```
# StartLog(LogFile="Example_Ensemble.TSTool.log")
# These commands query historical 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.  In other words, the current year will be complete,
# with observed at the beginning and historical "likely" at the end.
#
# First get the historic daily Lobatos gage and convert to traces.  Shift each trace to
# 2008-01-01 (the current year) so that the data can overlay the current year's
# values.
ReadTimeSeries(TSID="08251500.DWR.Streamflow.Day~HydroBase",Alias="daily",IfNotFound=Warn)
CreateEnsembleFromOneTimeSeries(TSID="daily",TraceLength=1Year,EnsembleID="Lobatos",
    ReferenceDate="2008-01-01",ShiftDataHow=ShiftToReference)
#
# Now weight the traces using representative historic years.
#
WeightTraces(EnsembleID="Lobatos",SpecifyWeightsHow="AbsoluteWeights",
    Weights="1997,.5,1998,.4,1999,.1",NewTSID="Lobatos..Streamflow.Day.likely",
    Alias="lobatos_likely")
#
# 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
# 08251500 - RIO GRANDE RIVER NEAR LOBATOS
08251500.DWR.Streamflow-DISCHRG.Irregular~HydroBase
# Convert the irrigular instantaneous values to a daily instantaneous (midnight)
ChangeInterval(TSList=LastMatchingTSID,TSID="08251500.DWR.Streamflow-
DISCHRG.Irregular",Alias="lobatos_current",NewInterval=Day,OldTimeScale=INST,NewTimeScale=INST)
#
# Blend the current data to the end of the historical data (the current data
# will override.
#
Blend(TSID="lobatos_current",IndependentTSID="lobatos_likely",BlendMethod=BlendAtEnd)
#
# 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.
```

ExamplesOfUse/Ensemble_Realtime+WeightedHistorical/Example_Realtime+WeightedHistorical

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.


TSTool - "C:\Develop\TSTool_SourceBuild\TSTool\test\regression\UserManualExamples\TestCases\ExamplesOfUse\Ensemble_Realtime+Weighte...
File Edit View Commands Run Results Tools Help
Get Time Series List
Copy Selected to Commands
Copy All to Commands
Run Selected Commands
Run All Commands
Clear Commands
Ensembles
Output Files
Problems
Tables
Time Series
Views

Input/Query Options

Data store:
Input type: HydroBase
Data type: Stream - Streamflow
Time step: Day
Where: Station Name Contains lobatos
Where: Matches
Where: Matches

Time Series List (1 time series, 0 selected)

	ID	CO Abbrev.	Name/Description	Data Source	Data Type
1	08251500	RIOLOBCO	RIO GRANDE NEAR LOBATOS, CO	DWR	Streamflow

Commands (34 commands, 0 selected, 0 with failures, 0 with warnings)

```
#
# Blend the current data to the end of the historical data (the current data
# will override.
#
Blend(TSID="lobatos_current",IndependentTSID="lobatos_likely",BlendMethod=BlendAtEnd)
#
# After the above commands are executed, time series in memory will include the traces and
```

Results

115 time series, 115 selected

```
1) daily - RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day (1899-07-01 to 2009-09-30)
2) 08251500_1899 - 1899 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1899] (2008-01-01 to 2008-12-31)
3) 08251500_1900 - 1900 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1900] (2008-01-01 to 2008-12-31)
4) 08251500_1901 - 1901 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1901] (2008-01-01 to 2008-12-31)
5) 08251500_1902 - 1902 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1902] (2008-01-01 to 2008-12-31)
6) 08251500_1903 - 1903 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1903] (2008-01-01 to 2008-12-31)
7) 08251500_1904 - 1904 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1904] (2008-01-01 to 2008-12-31)
8) 08251500_1905 - 1905 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1905] (2008-01-01 to 2008-12-31)
9) 08251500_1906 - 1906 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1906] (2008-01-01 to 2008-12-31)
10) 08251500_1907 - 1907 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1907] (2008-01-01 to 2008-12-31)
11) 08251500_1908 - 1908 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1908] (2008-01-01 to 2008-12-31)
12) 08251500_1909 - 1909 trace: RIO GRANDE NEAR LOBATOS, CO - 08251500.DWR.Streamflow.Day[1909] (2008-01-01 to 2008-12-31)
```

Completed running commands. Use Results and Tools menus. 100% 100% Ready

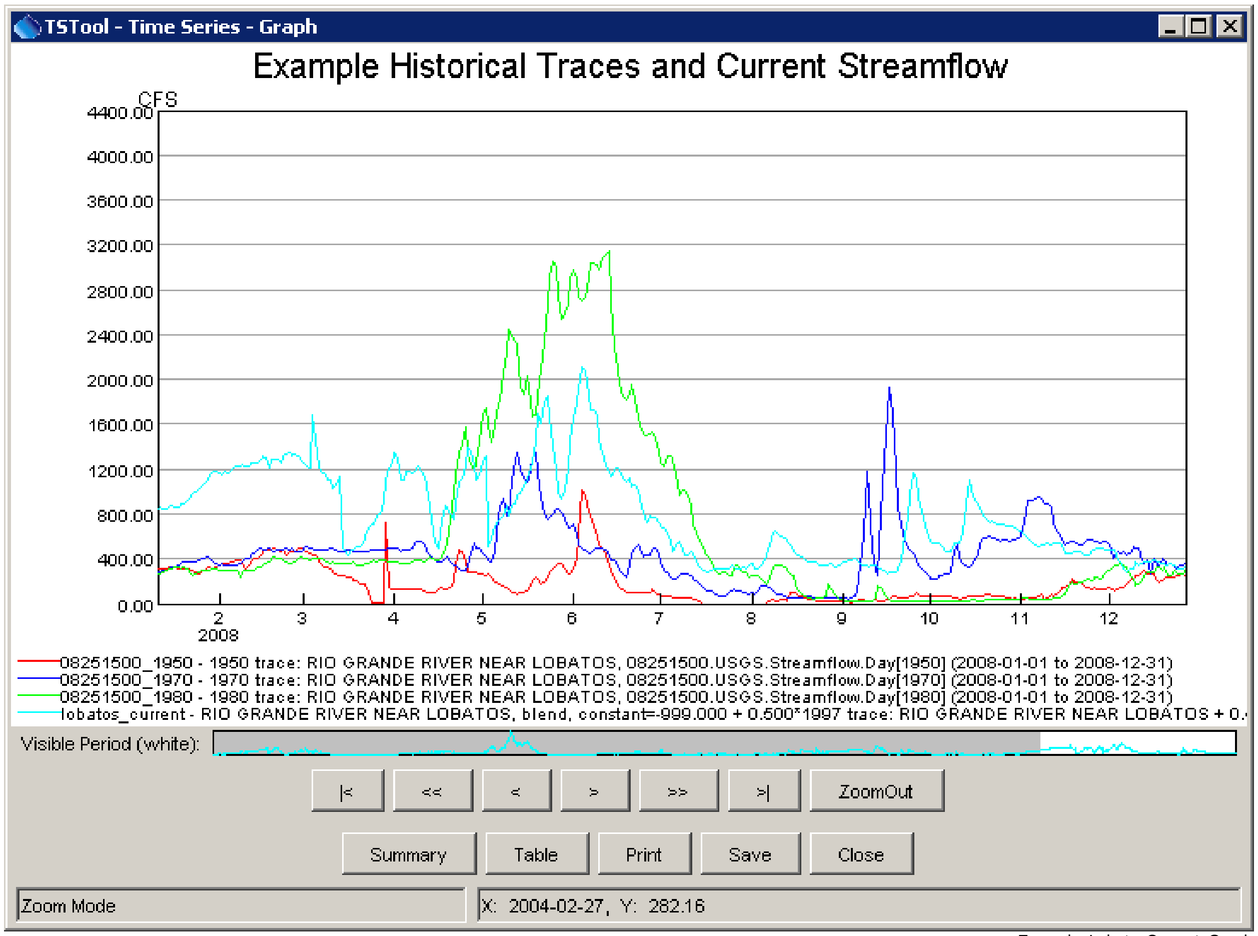
Example_LobatosCurrent_TSTool

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 an ensemble and then plot all the traces. Use the graph properties to turn traces on/off or use symbols to identify different traces.
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. The `NewStatisticYearTS()` and `NewStatisticTimeSeriesFromEnsemble()` commands create derived time series that are useful for evaluating ensemble time series.

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



08251500_1950 - 1950 trace: RIO GRANDE RIVER NEAR LOBATOS, 08251500.USGS.Streamflow.Day[1950] (2008-01-01 to 2008-12-31)
08251500_1970 - 1970 trace: RIO GRANDE RIVER NEAR LOBATOS, 08251500.USGS.Streamflow.Day[1970] (2008-01-01 to 2008-12-31)
08251500_1980 - 1980 trace: RIO GRANDE RIVER NEAR LOBATOS, 08251500.USGS.Streamflow.Day[1980] (2008-01-01 to 2008-12-31)
lobatos_current - RIO GRANDE RIVER NEAR LOBATOS, blend, constant=-999.000 + 0.500*1997 trace: RIO GRANDE RIVER NEAR LOBATOS + 0.

Zoom Mode

X: 2004-02-27, Y: 282.16

Example_LobatosCurrent_Graph

Example Graph of Traces and Combined Real-time/Historical Time Series

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 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 primarily is used as an interactive tool or to process command files in batch mode. However, it is also possible to run TSTool with a command 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**). In this run mode the main GUI is never made visible. 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.

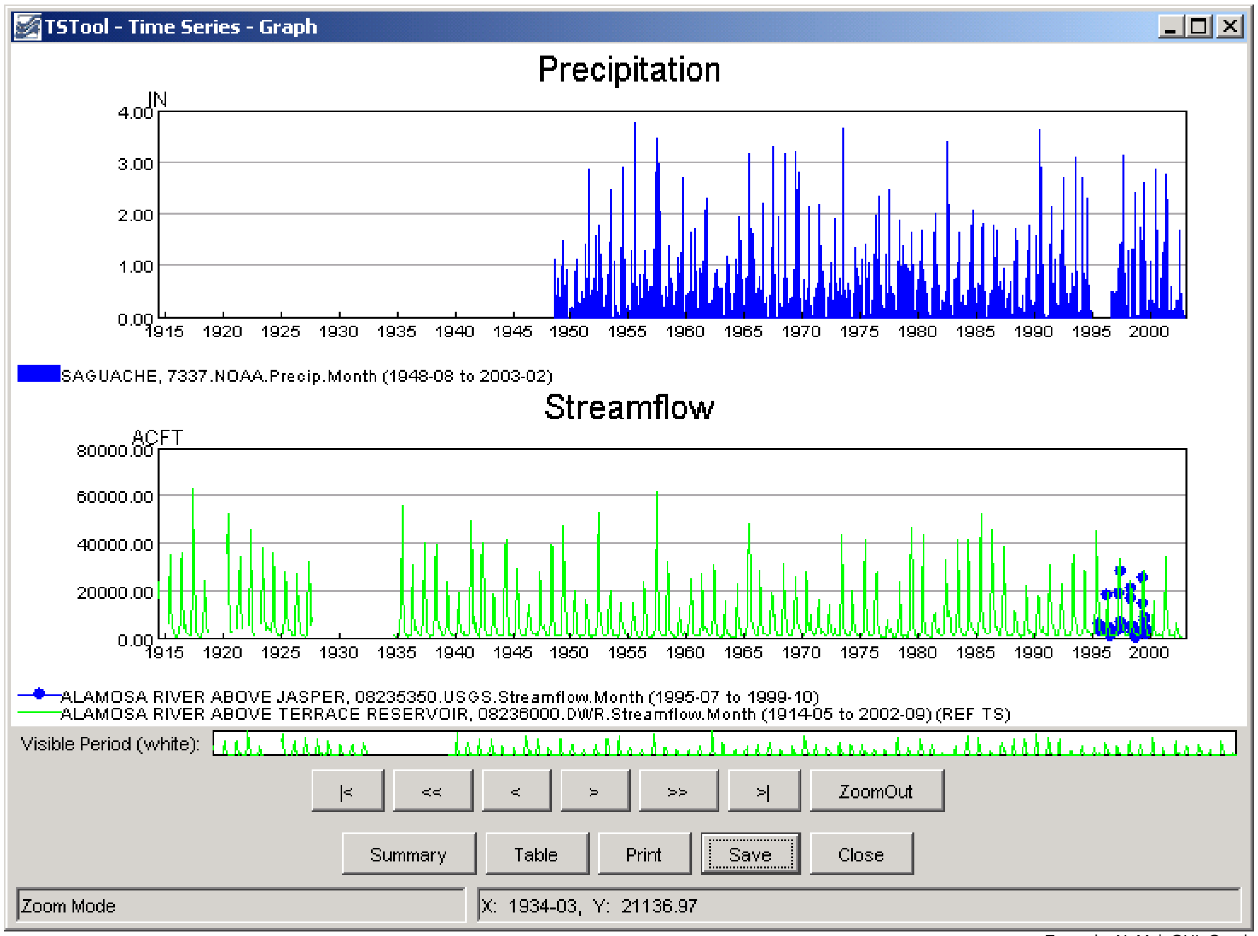
Although it is possible to display several graphs at the same time, it is currently assumed that only one graph will be shown. Closing the graph will close TSTool. The command file can be complex but in many cases will be simple because an application is calling TSTool to display a single graph. The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

```
# Example command file to run TSTool 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 only need to use
# TSTool in a supporting role.
#
# Process a time series product description file and display a plot window
# to the screen.
ProcessTSProduct(TSProductFile="test.tsp",RunMode=GUIAndBatch,View=True)
```

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.



Example_NoMainGUI_Graph

6.4.2 Automating Graphs to Compare Observed and Simulated Time Series

It often is useful to automate comparison of observed and simulated time series (e.g., during model calibration). For example, consider the simulated and observed time series stored in a DateValue file (*results.dv*), as follows (in this example the DateValue file was created by reading StateMod model input and output, and the `TSID` and `DataType` lines were hand-edited in the DateValue 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\Data\SWSI_Gunnison\StateMod\gunnv.rih
# 09152500.StateMod.River_Outflow.Month~StateModB~J:\CDSS\Data\SWSI_Gunnison\StateMod\gunnvb.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\gunnvb.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 . . .
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 . . .
```

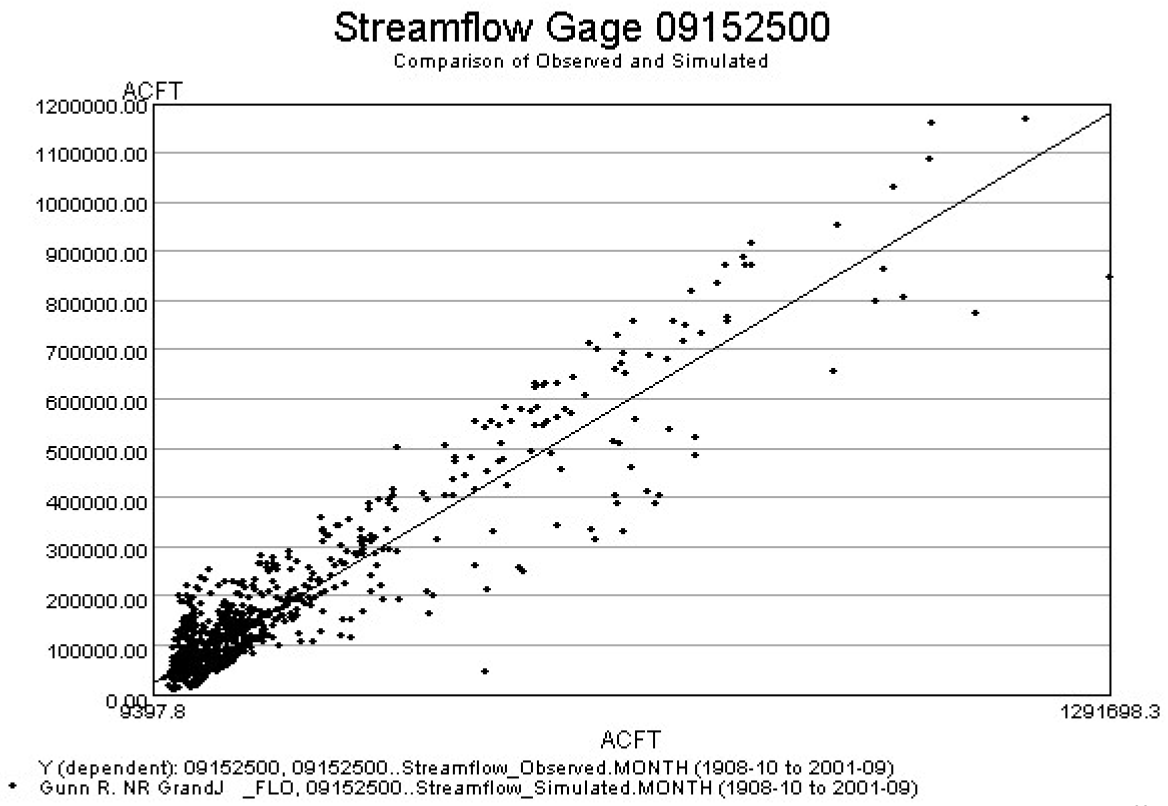
These time series can be read into TSTool, an XY graph produced, and the time series product saved (*results.tsp*), as shown in the following example. The original TSID properties have been inserted as comments, corresponding to the original data. The absolute paths to the time series files have also been replaced with relative paths, assuming that the command file to process the product is in the same folder as the product file.

```
[Product]

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

[SubProduct 1]

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

[Data 1.1]

#TSID = "09152500...MONTH~StateMod~gunnv.rih"
TSID = "09152500..Streamflow_Observed.MONTH~DateValue~results.dv"

[Data 1.2]

#TSID = "09152500.StateMod.River_Outflow.Month~gunnvb.b43"
TSID = "09152500..Streamflow_Simulated.MONTH~DateValue~results.dv"
```

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

```
ProcessTSProduct(TSProductFile="results.tsp",RunMode=GUIAndBatch,
View=True,OutputFile="09152500.png")
```

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 – resize the graph window to improve labeling):



09152500.jpg

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. The `TSAlias` property can also be used in product files and will take precedence over `TSID` properties. Using time series aliases can improve the readability of command files.

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 (for example see the `ExpandTemplateFile()` command).

This page is intentionally blank.

7 Spatial Data Integration

Version 10.15.00, 2013-01-09

This chapter will be updated when addition spatial data integration features are implemented. Much of the content at the end of this chapter is still relevant to the software, but will be updated significantly in the future.

Although the main focus in TSTool is time series, many time series are associated with a location such as a station, area, or sensor. This chapter discusses the relationship between time series and spatial data and provides an overview of using map-related features in TSTool. Time series concepts (such as time series identifiers) are discussed in detail in **Chapter 2 – Introduction**. Information about the built-in map display tool used in TSTool is provided in the **GeoView Mapping Tools Appendix**.

The map capability in the TSTool user interface is limited and has not been fully developed. However, commands related to tables can be used to link time series to tables, for example:

- the `ReadTableFromDBF()` command can be used to read the attribute data from an Esri shapefile and attributes can be attached to time series using the `SetTimeSeriesPropertiesFromTable()` command
- similarly, it is possible to use commands like `CalculateTimeSeriesStatistic()`, save to a table, and then join the table in a spatial data layer using GIS tools
- time series read from sources that provide location data typically have properties set during the read and these properties can be copied to a table with `CopyTimeSeriesPropertiesToTable()`, the table can be written using `WriteTableToDelimitedFile()`, and then the file can be used by GIS software as a point layer

It also is often possible to perform selections of time series based on spatial constraints, simply by using available attributes. For example USGS NWIS web services allow querying by county name and Hydrologic Unit Code. Of course, this requires that the locational properties for time series are available.

The remainder of this chapter describes map-related features and concepts. Future TSTool enhancements will build on features described in the above paragraph in order to allow automated processing of map data and products (similar to how the `ProcessTSProduct()` command processes time series products).

7.1 Time Series and Map Layer Relationships

An example is useful to provide an overview of the relationship between time series and map layers.

Map layers often indicate physical features (e.g., rivers, cities, roads, data collection stations) or features that are overlaid on physical features (e.g., political boundaries, weather fronts, regions or points of interest). A layer's data consist of:

1. Features – the coordinate information that defines the shape on the map.
2. Attributes – a tabular list of data values associated with the features.
3. Metadata – data about the layer, including the source, coordinate system, history, etc.
4. Projection – the coordinate system for the coordinates, which is usually noted in metadata but may also be indicated by a projection file or similar.

5. Symbology – the symbol information for each layer (e.g., point symbol, line width, polygon fill color), labels, and other visualization information.

The features and attributes are the primary data, and the other information facilitates using the features and attributes.

Consider a data collection station, represented by a point on the map. This station may be located near a river and collect streamflow stage (water depth). The station software may convert the stage to flow or may allow this to be done by other software. The station also may collect "climate" (meteorological) data such as precipitation, temperature, wind speed and direction, etc. Each measurement type requires a sensor and the cost of hardware typically controls the number and sensitivity of measuring devices. For data management, the station is typically assigned an alphanumeric station identifier and each data type that is collected is assigned an alphanumeric data type. The data are saved locally as date/value information and are then transmitted to or requested from a data collecting system. The date/value information is essentially time series. Data units and handling of missing data are considered during implementation of data collection systems. Measurements may be taken regularly (e.g., once every fifteen minutes) or may occur at irregular intervals, perhaps in response to some change in conditions. In nomenclature used with TSTool, the former are called regular time series and the latter irregular, reflected by the data interval (time step).

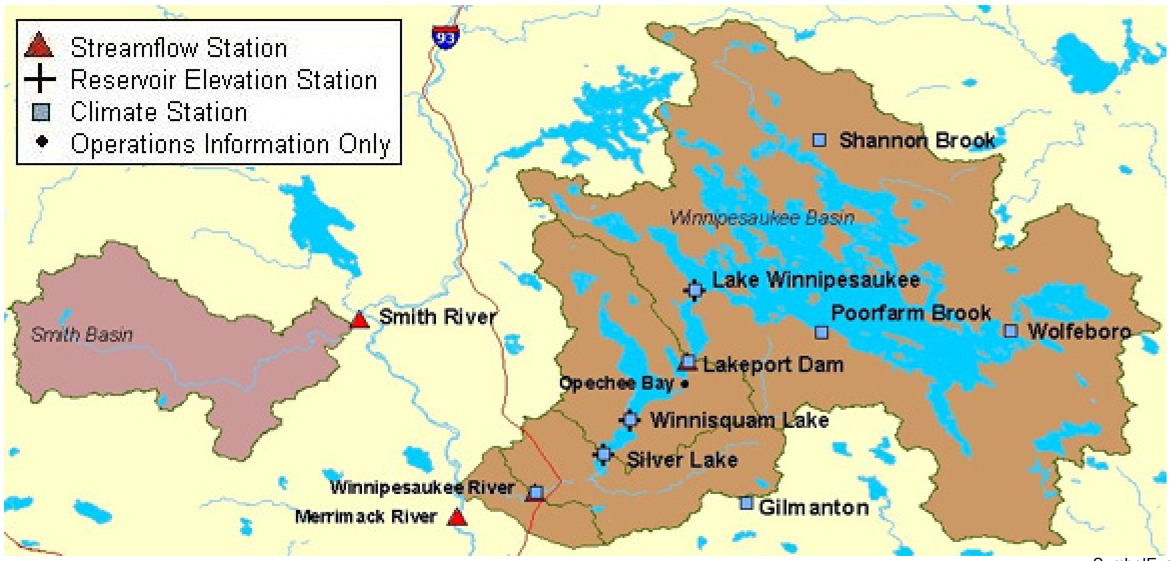
For the discussion purposes, consider only a meteorological station that measures precipitation and temperature. For mapping purposes, a choice may be made to focus on the physical nature of the map, in which case a single "Met Stations" (or "Climate Stations") layer may be shown, using a single symbol. This is suitable if the measurement types for such a station are consistent throughout a system or only one data type is of interest. It is frequently the case that the real-time data that are collected are managed in a database, with data being archived over time, for example resulting in the following time series for precipitation data:

1. Real-time data (often provisional data available for a short period).
2. Real-time extended data – real-time data collected for the past year, for example, having received limited or no quality control
3. Real-time archived data – real-time data for the historical period, quality controlled
4. Hourly accumulation – for example, convert real-time precipitation data to hourly totals
5. Daily accumulation – for example, convert hourly precipitation data to daily totals
6. Monthly accumulation – etc.
7. Yearly accumulation – etc.

The first two examples are often referred to as "real-time" data whereas the last five examples are often referred to as "historical" data. In a system that is homogeneous, a map layer that shows "Precipitation Stations" will imply that all of the above time series are available at the station. However, in a system where, for various reasons, not all stations have real-time and historical data, more attention to detail may be needed on maps.

To address this case, the map could show separate layers for real-time and historical stations (two layers). However, this does not address the issue that there may be multiple categories of real-time data and multiple categories of historical data. To address this issue, additional layers might be added for each time series type, using the same or similar symbols for each layer. The limitation in this approach is that the map now has many layers and many of the points will be the same and will therefore symbols will plot on top of each other.

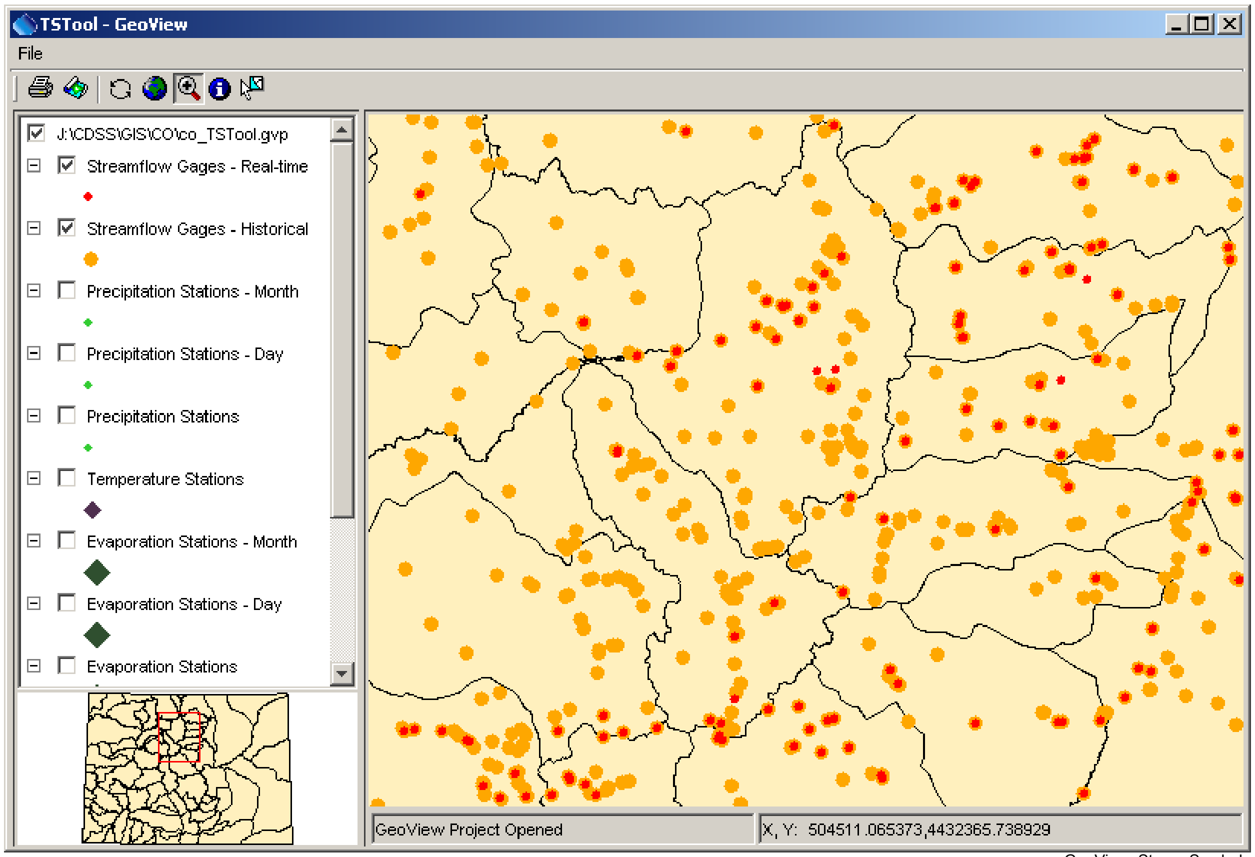
Layered symbols can be used to help with visualization. For example, use the symbol size and shape, and configure the order of layers to ensure that multiple symbols drawing on each other will still allow sufficient visibility of the symbols. This can be applied to indicate data types collected at a location, and also data type/interval information. The following example shows this approach to indicate station type:



SymbolExample

Example of Symbol Layering

Using many layers and symbols to indicate time series data interval may not be appropriate if the symbols are to be used for classification. For example, the symbol size or color may indicate a physical characteristic of the feature. For this reason and because maps usually focus on physical features, using symbols to indicate the various data intervals is not common. More common are maps that show a layer for real-time data and a layer for historical data, as shown in the following figure:


TSTool - GeoView
File
J:\CDSS\GIS\CO\co_TSTool.gvp
Streamflow Gages - Real-time
Streamflow Gages - Historical
Precipitation Stations - Month
Precipitation Stations - Day
Precipitation Stations
Temperature Stations
Evaporation Stations - Month
Evaporation Stations - Day
Evaporation Stations
GeoView Project Opened
X, Y: 504511.065373,4432365.738929

GeoView_StreamSymbols

Example Map Showing Real-time and Historical Data Layers

The map is useful because the user and software can determine where real-time time series SHOULD be available, and where historical data SHOULD be available. The mapping tools can be integrated with application software by hard-coding the data type and interval for real-time and historical data or use a configuration file (e.g., to indicate that “historical” data should always have an interval of Month).

TSTool is a generic tool and therefore configuration information is required to make the link between the map layers and time series. The approach that has been taken is to rely on a delimited lookup file, which allows users to determine at what level to categorize map layers. More specific information allows a more specific link (i.e., a time series in TSTool can be matched with a specific feature in a map layer) while less information results in a looser link (i.e., several time series in TSTool may match one station and selecting the station may result in more than one time series).

The following example illustrates the time series to map layer configuration 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.
#
# This file is discussed in the TSTool documentation.
#
# 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,””
```

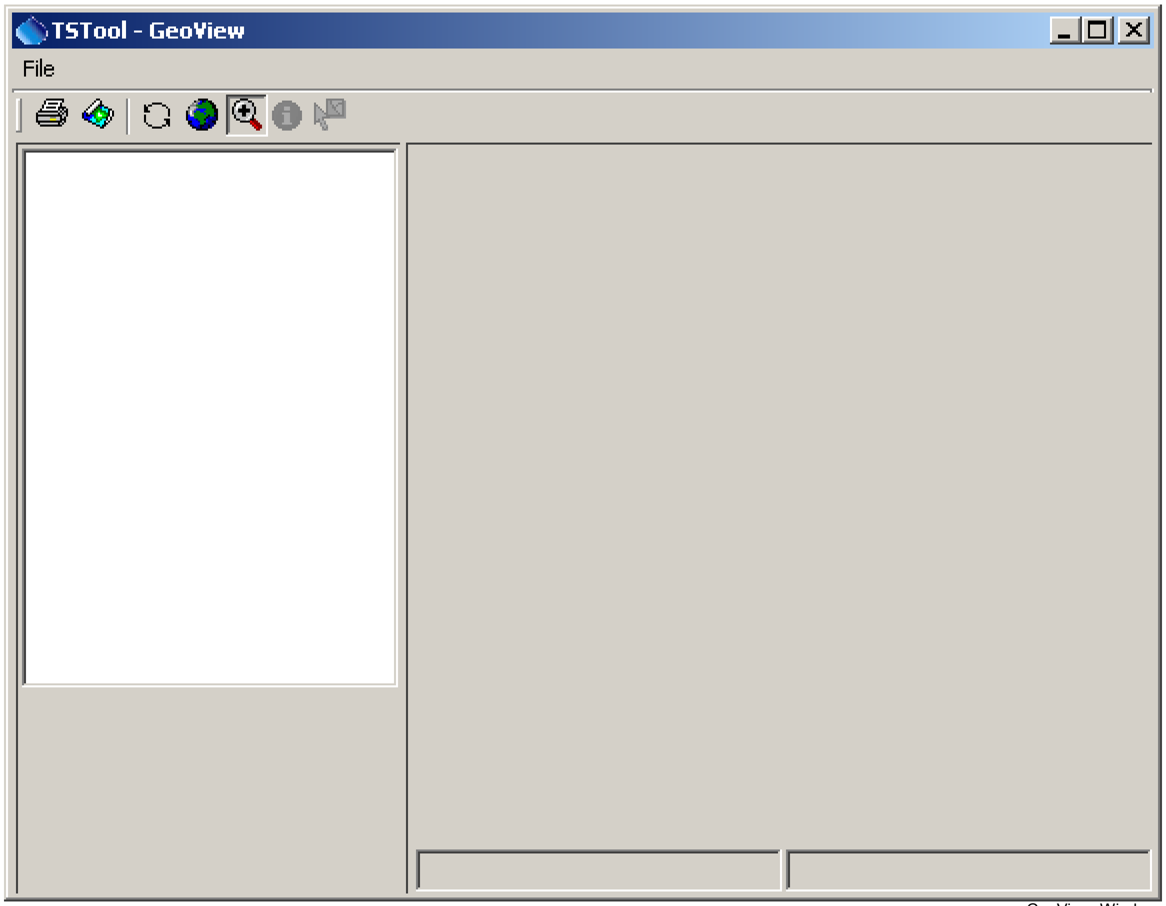
Example Time Series to Map Layer Lookup File

The intent of the file is to allow lookup of map layers from time series and to allow lookup of time series from map layers. The ability to perform these actions depends on the number of layers in the map and the attributes in map layers. For example, in the last two lines of the above example, historical streamflow time series are both linked to a single “Streamflow Gages – Historical” layer on the map. Consequently, it will not be possible from the map layer to indicate to TSTool that specifically day or month interval data are requested (both daily and monthly time series would be selected). This behavior may be appropriate, or more specific layers and attributes may be needed. Multiple maps are typically needed, in order to meet all the various needs of users and therefore maps with more detail may need to be configured for use with TSTool.

The following sections describe more specifically how to utilize the links between time series and map layers.

7.2 Opening a Map

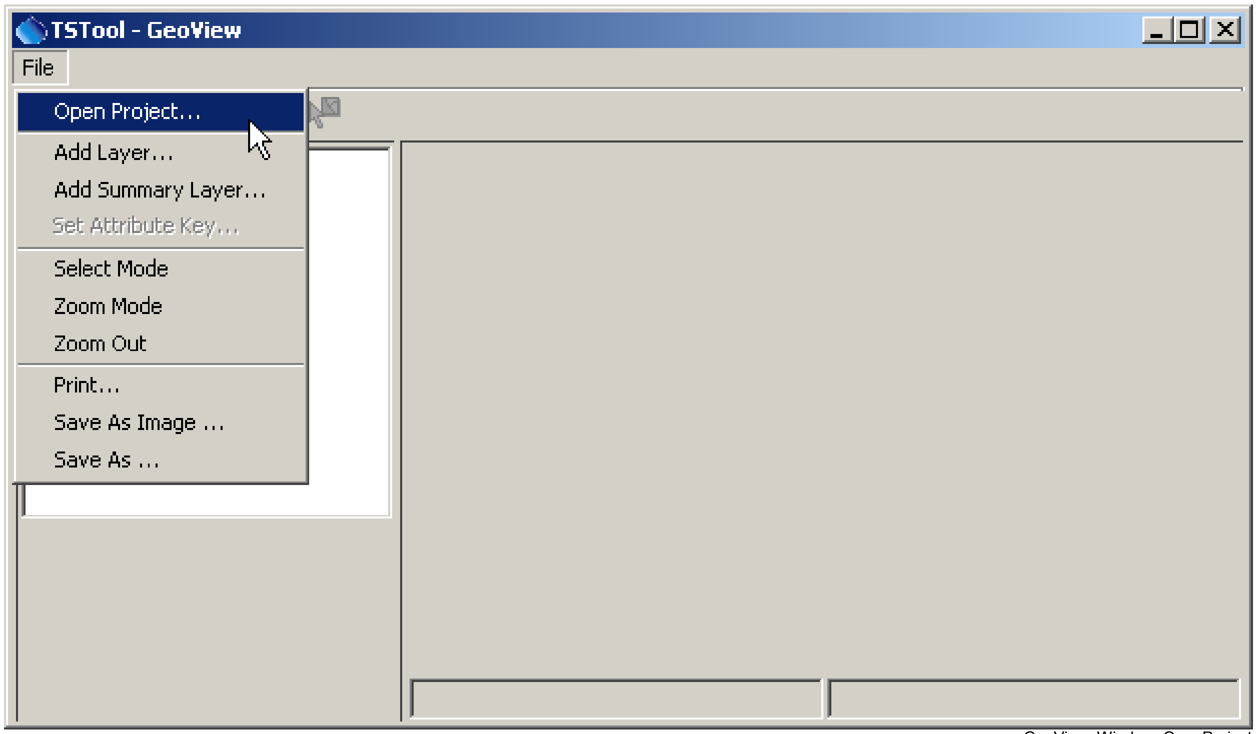
To open a map in TSTool, first select the ***View…Map*** menu item, which will display the following window:



GeoView_Window_Blank

Map (GeoView) Window when First Opened

In this window, select ***File…Open Project*** and select a GeoView Project File (**.gvp*), as shown below:



GeoView_Window_OpenProject

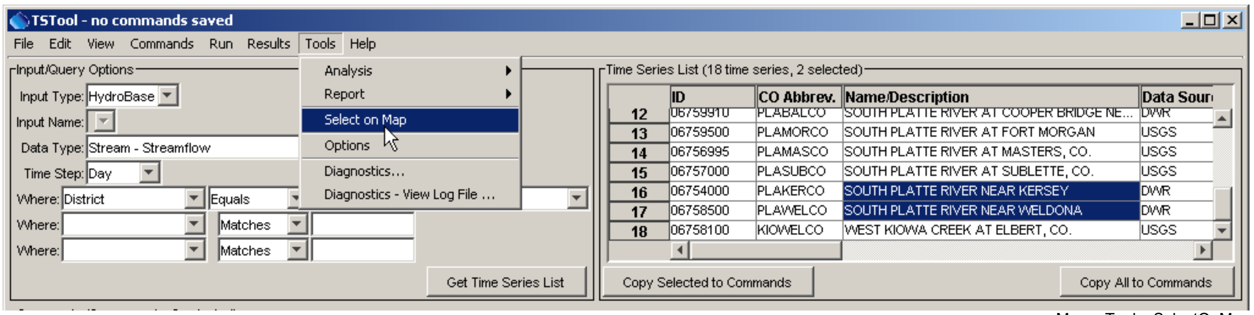
Opening a Map (GeoView Project) File

The format for a GVP file is described in the **GeoView Mapping Tools Appendix**. The file is a simple text file that can be manually edited. Although using an ESRI MXD or other file was considered, such file formats have been changing, are binary, and are proprietary in nature.

After opening the GVP file, a map will be displayed and the TSTool ***Tools…Show on Map*** button will be enabled when appropriate.

7.3 Using Time Series to Select Locations on the Map

To select time series on the map, first select time series in the upper part of the TSTool interface and then select the ***Tools…Select on Map*** menu item, as illustrated in the following figure:


TSTool - no commands saved
File Edit View Commands Run Results Tools Help
Input/Query Options
Analysis
Report
Select on Map
Options
Diagnostics...
Diagnostics - View Log File ...
Get Time Series List
Copy Selected to Commands
Copy All to Commands

Input Type: HydroBase
Input Name:
Data Type: Stream - Streamflow
Time Step: Day
Where: District Equals
Where: Matches
Where: Matches

Time Series List (18 time series, 2 selected)

	ID	CO Abbrev.	Name/Description	Data Sour
12	06759910	PLABALCO	SOUTH PLATTE RIVER AT COOPER BRIDGE NE...	DWR
13	06759500	PLAMORCO	SOUTH PLATTE RIVER AT FORT MORGAN	USGS
14	06756995	PLAMASCO	SOUTH PLATTE RIVER AT MASTERS, CO.	USGS
15	06757000	PLASUBCO	SOUTH PLATTE RIVER AT SUBLETTE, CO.	USGS
16	06754000	PLAKERCO	SOUTH PLATTE RIVER NEAR KERSEY	DWR
17	06758500	PLAWELCO	SOUTH PLATTE RIVER NEAR WELDONA	DWR
18	06758100	KIOWELCO	WEST KIOWA CREEK AT ELBERT, CO.	USGS

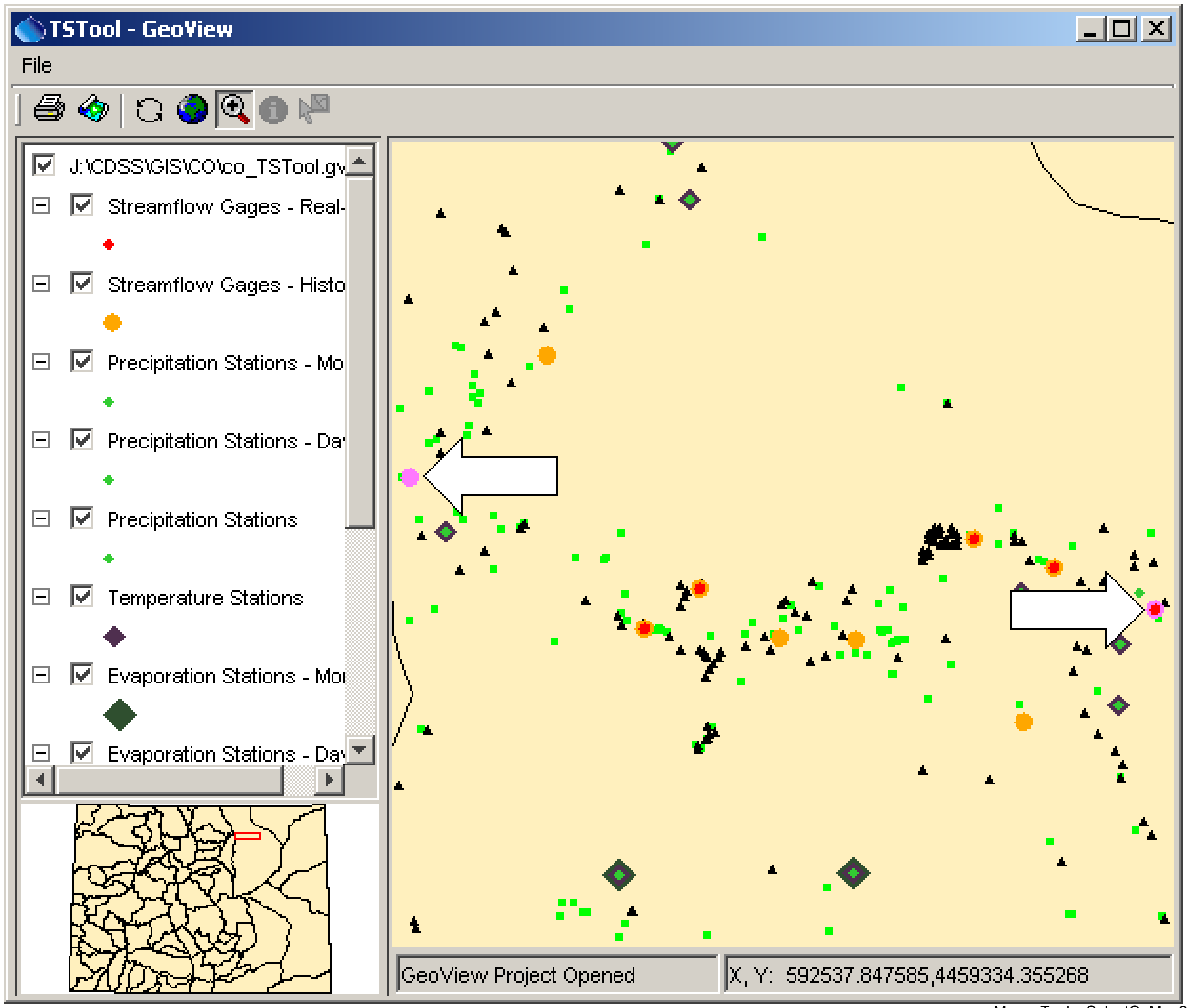
Menu_Tools_SelectOnMap

Example of Selecting Time Series on the Map

Note that in the above example, the selection is initiated from the time series input/query list (not the time series results at the bottom of the TSTool main window). The latter may be implemented in the future; however, it is faster to use the input/query list because only time series header information is processed.

When ***Tools…Select on Map*** is selected, features on the map are selected as follows:

1. The software tries to find appropriate map layers by matching the lookup file `TS_DataType` and `TS_Interval` column values with the ***Data Type*** and ***Time Step*** values in the input/query list.
2. The resulting layers (indicated by `Layer_Name`) are searched to match the ***ID*** (and optionally ***Data Source***) values in the input/query time series list with the attributes indicated by the `Layer_Location` and `Layer_DataSource` lookup file (`Layer_Interval` can also optionally be used).
3. Matching features are selected and the map zooms to highlight the features, as shown in the following figure (the arrows have been added for illustration).


TSTool - GeoView
File
J:\CDSS\GIS\CO\co_TSTool.gv
Streamflow Gages - Real-
Streamflow Gages - Histo
Precipitation Stations - Mo
Precipitation Stations - Da
Precipitation Stations
Temperature Stations
Evaporation Stations - Mo
Evaporation Stations - Da
GeoView Project Opened
X, Y: 592537.847585,4459334.355268
Menu_Tools_SelectOnMap2

Map after Using Time Series to Select Features

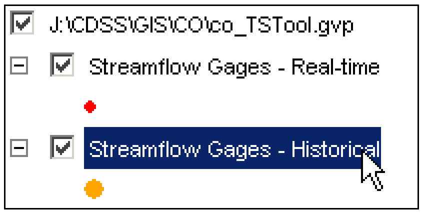
If the lookup file does not include a `Layer_DataSource`, then this value is ignored in the search. Once selected on the map, users can evaluate the significance of the distribution of stations, etc. and can use other map features.

To support this functionality, the attributes for the layer should include a column for the location identifier, and optionally the data source, where the values match the values shown in TSTool. Additionally providing an attribute for data interval allows another level of selection. In this case, the intervals in the attributes must match those shown in TSTool. Providing all information will result in record-level queries that allow a direct link between a time series and a layer.

7.4 Using Locations on the Map to Select Time Series

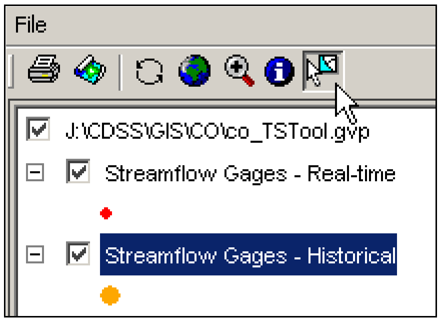
To select time series from the map:

1. Perform a time series query to create a list of time series (see the previous section for an example). **The map can be used to select time series in this list, but selecting from the map will not initiate a database query or file read. This is because the variety of input types that TSTool supports require specific information to query/read time series**.
2. Select a layer of interest on the map. This layer should correspond to time series in the list from the first step. For example, if the time series list contains daily streamflow time series, select the map layer that corresponds to such data. For example:


J:\CDSS\GIS\CO\co_TSTool.gvp
Streamflow Gages - Real-time
Streamflow Gages - Historical

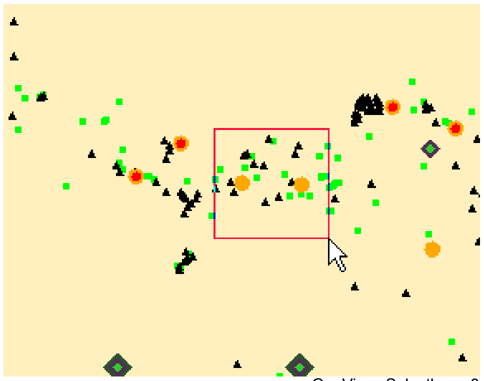
GeoView_SelectLayer

3. Activate the select tool on the map interface toolbar:

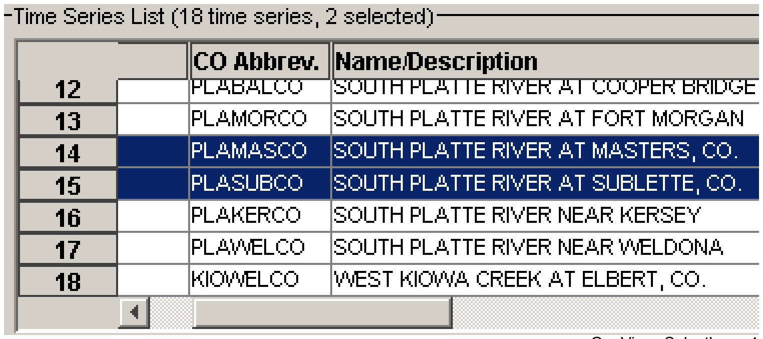

File
J:\CDSS\GIS\CO\co_TSTool.gvp
Streamflow Gages - Real-time
Streamflow Gages - Historical

GeoView_SelectLayer2

4. Draw a box around features of interest on the map. The software will attempt to match the features in the time series list in the TSTool main window. To do so, it first matches the layer name with the similar value in the lookup file. It then tries to match the ***ID*** (and optionally ***Data Source***) values in the input/query time series list with the attributes indicated by the `Layer_Location` and `Layer_DataSource` lookup file (`Layer_Interval` can also be used). Matched time series are selected in the list. Previous selections are not cleared – use the right click popup menu to clear selections first if appropriate.



Selecting Features on the Map



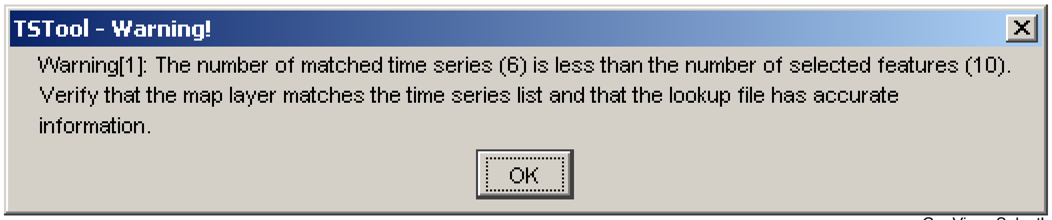
Time Series List (18 time series, 2 selected)

		CO Abbrev.	Name/Description
12		PLABALCO	SOUTH PLATTE RIVER AT COOPER BRIDGE
13		PLAMORCO	SOUTH PLATTE RIVER AT FORT MORGAN
14		PLAMASCO	SOUTH PLATTE RIVER AT MASTERS, CO.
15		PLASUBCO	SOUTH PLATTE RIVER AT SUBLETTE, CO.
16		PLAKERCO	SOUTH PLATTE RIVER NEAR KERSEY
17		PLAWELCO	SOUTH PLATTE RIVER NEAR WELDONA
18		KIOWELCO	WEST KIOWA CREEK AT ELBERT, CO.

Time Series List After Select from Map

5. Once time series have been matched, the information can be copied to the commands area for further processing. This capability is therefore useful for identifying available data for an area.

If not all selected features in a layer correspond to time series, a warning similar to the following may be shown:



TSTool - Warning!

Warning[1]: The number of matched time series (6) is less than the number of selected features (10). Verify that the map layer matches the time series list and that the lookup file has accurate information.

Selection Warning

This may indicate that the attributes in the spatial data are not detailed enough to do the lookup. For example, a data source attribute may exist but may only apply to one data interval (e.g., real-time instead of historical data types). A sufficient combination of layers and attributes in layers can avoid or minimize such problems.

The above procedure is not completely robust in that the user may select a layer that does not match the time series list. Additional features are being considered to minimize this possibility. However, the use of the map interface is considered an advanced feature and some reliance is made on a user's capability.

7.5 Spatial Analysis Commands

TSTool's commands provide a powerful analysis and data processing capability. The above sections provided an overview of how to link time series and spatial data. It is envisioned that in the future commands will be added to perform processing of time series, considering spatial data (e.g., weight time series based on their proximity to a point). The ability to perform spatial and temporal analysis in batch mode will be implemented as appropriate to meet user requirements.

8 Troubleshooting

Version 10.12.00, 2012-09-10

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.

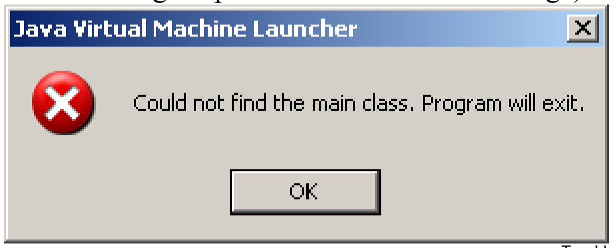
TSTool can run in interactive mode with a graphical user interface (GUI) and in batch mode. In both cases a log file contains messages from the program. By default, the log file is created in the logs directory under the main installation directory. It is recommended that the `StartLog()` command be inserted as the first command in each command file, using the name of the commands file in its name. The log file can then be viewed using the ***Tools…Diagnostics*** features (see **Chapter 3 – Getting Started**). However, in most cases, the log file should only be used for major troubleshooting because it contains technical details that may not be understandable by the user. The error-handling features of the GUI provide a status for each command. Often, an error in an early command leads to additional errors in other commands and therefore fixing the first error can resolve multiple problems.

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 (use the ***Help…About*** menu to list support contacts). You may need to email the log file to support to help determine the nature of a problem.

When running the TSTool GUI, major problems will be indicated with an icon next to the offending command (see **Chapter 3 – Getting Started** for a summary of command error handling features). When running in batch mode, warnings are only printed to the log file. In either case, the log file viewer can be used to pinpoint the source of problems. If the run has been successful the GUI will show no problem indicators and the log file will contain primarily status messages, which provide useful information about data processing.

The following table summarizes common errors and their fixes. If an error is occurring in batch mode, it is useful to run via the GUI to utilize error feedback features. Errors specific to a database are discussed in the documentation for the database (see input type and datastore appendices).

TSTool Errors and Possible Solutions

Error	Possible solutions
TSTool does not run on Windows (error at start-up).	1. If using the TSTool executable on Windows and the following is shown (or a command line message is printed with a similar message):  Java Virtual Machine Launcher Could not find the main class. Program will exit. OK Troubleshooting_LaunchError This error may be shown if software files have been manually moved. Reinstall using the installation program. 2. If TSTool is run on Windows using the batch file… The batch file (in the *bin* 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. Additionally, to help diagnose errors, try running the *TSTool.exe* or *TSTool.bat* 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.
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. Refer to the input type appendix for limitations on data handling.
Time series (of any data type) have `-999` or other missing 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`. Use fill commands to fill the missing data within the requested period.
TSTool fails on large queries or displays out of memory error.	TSTool may run out of memory on queries (hundreds or thousands of time series, depending on machine memory). More time series may be able to be handled if run in batch mode because GUI resources are minimized. To increase the amount of memory that TSTool will use: 1. If running on Windows using the TSTool.exe program (the default configuration), increase the value of the `-XmxNNNm` option in the *bin\TSTool.l4j.ini* file under the software installation folder. 2. If running on Windows using the *TSTool.bat* file, change the `-XmxNNNm` option after the JRE program name to tell Java to allow more memory (increase the number of MB `NNN` as appropriate for the amount of memory available on the machine – use a high number to force using hard disk swap space if desired). 3. If running on Linux or Unix using the *tstool* script, change the `-XmxNNNm` option after the JRE program name to tell Java to allow more memory

Error	Possible solutions
	(increase the number of MB `NNN` 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.	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: 1. TSTool has been developed using a version of Java that is indicated in the metadata for software files. Trying to use an older Java version may cause unexpected errors. This will only be a problem in custom installations where the default Java distributed with TSTool is not used. To determine the Java version that is being used to run TSTool and that was used to create the software, 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. 2. An unforeseen issue may be occurring. Contact support. You may need to provide the data and command file being used, which will allow troubleshooting and also allow developers to add additional tests to be run before software is released.
Unable to find files correctly.	The working directory is assumed to be the same as the location of the most recently opened or saved command file. The current working directory is generally displayed by editor dialogs that use a path and can also be displayed using ***File…Properties…TSTool Session***. If files are not being found, verify that the path to the file is correct, whether specified as an absolute path or relative to the command file.
When used with the HydroBase input type for a Microsoft Access database, an error occurs selecting the HydroBase database. Microsoft Access versions of HydroBase have not been used for years so this problem is unlikely.	1. TSTool tries to list the ODBC DSN that are available for HydroBase databases. It does so by running the *shellcon.exe* program, typically installed in *\cdss\bin* 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: • For Windows NT/2000/XP machines, add *\CDSS\bin* (or *\Program Files\RTi\RiverTrak\bin*) 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 *\CDSS\bin* (or *\Program Files\RTi\RiverTrak\bin*) to the `PATH` in the *autoexec.bat* file. Reboot to apply for all subsequent windows. You may have done this previously. A work-around is to manually type in the ODBC DSN in the entry field. 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.
TSTool is unable to	The HEC-DSS library files are distributed in the *TSTool-Version/bin* folder and

Error	Possible solutions
load HEC-DSS DLL files and therefore HEC-DSS features are unavailable.	by default this is where TSTool looks for the files. If the start folder for TSTool is changed from this folder, the files will not be found. Therefore, do not reconfigure TSTool to start in other than the *bin* folder.

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 with fixed parameter list was adopted:

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

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. Subsequent enhancements added new commands and converted 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. The above syntax is now standard throughout TSTool. Support for the older notation is provided where possible.

Prior to TSTool version 10, some commands used the syntax:

```
TS Alias = Command(…)
```

In version 10, this syntax has been made similar to all other commands:

```
Command(Alias=”….”,…)
```

In most cases, loading an old command file will automatically convert from old to new syntax. TSTool provides warnings for commands that are not recognized or are out of date and cannot automatically be updated –the command editor can be used to correct errors.

The following table lists obsolete commands. The `TSID` abbreviation, when inside parentheses for a command, is interchangeable with the time series alias.

TSTool Command Summary – Obsolete Commands

Command	Description	Replacement
`add(TSID,TSID1, TSID2,...)`	Add the 2nd+ 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.	Current `Add()`.
`-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 *tstool.bat* 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.	`OpenHydroBase()` and configuration information.
`-averageperiod MM/YYYY MM/YYYY`	Specify the period to be used to compute averages when the `-fillhistave` option is specified.	`SetAveragePeriod()`
`-batch`	Indicates to run in batch mode. This is automatically set if `-commands` is specified.	None – no longer used.
`-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.	None – no longer used.
`CreateTraces()`	Create an ensemble from a time series.	`CreateEnsemble()`
`-cy`	Output in calendar year format.	`SetOutputYearType()`
`-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.	`SetDebugLevel()`
`-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.	`CreateFromList()`
`-datasource ODBCDataSourceName`	Specify an ODBC Data Source Name to use for the HydroBase database.	`OpenHydroBase()` and configuration information.
`-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.	`CreateFromList()`

TSTool Command Summary – Obsolete Commands (continued)

Command	Description	Replacement
`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.	`NewEndOfMonthTS FromDayTS()` and `FillInterpolate()`
`-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.	`OpenHydroBase()` and configuration information.
`-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.	See output command parameters to control.
`fillCarryForward()`	Fill by repeating value.	`FillRepeat()`
`fillconst (TSID, Value)`	Fill the time series with a constant value.	`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.	`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.	`FillHistMonthAverage()` and `FillHistYearAverage()`
`Graph g = newGraph(GraphType, Visibility, TimeSeriesToGraph)`	Create a new graph window.	This command is no longer supported. See `ProcessTSProduct().`
`-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.	No longer used.
`-ignorelezero`	Treat data values <= 0 as missing when computing averages but do not replace when filling.	`SetIgnoreLEZero()`
`-include_missing_ts`	If a time series cannot be found, include an empty time series.	`SetIncludeMissingTS()`

TSTool Command Summary – Obsolete Commands (continued)

Command	Description	Replacement
`-informix`	Indicate that Informix is used for HydroBase.	Not used.
`-missing Value`	Use the specified value for missing data values (StateMod only). The default is `-999.0`.	See `WriteStateMod()`.
`-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.	`FillUseDiversion Comments()`
`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.	`SetOutputPeriod()`
`-o outputfile`	Specify output file name. This is used in conjunction with other `-o` options.	`Write*()` commands.
`-odatevalue`	Output a DateValue format file.	`WriteDateValue()`
`-ostatemod`	Output a StateMod format file.	`WriteStateMod()`
`-osummary`	Output a time series summary.	`WriteSummary()`
`-osummarynostats`	Output a time series summary without statistics (this is used with the data extension procedure developed by Ayres for CDSS).	No longer supported.
`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.	`FillRegression()`
`regress12(TSID1, TSID2)` `regressMonthly( TSID1,TSID2)`	Same as `regress()` except 12 separate monthly regressions values are calculated.	`FillRegression()`
`regresslog(TSID1, TSID2)`	Same as `regress()` except regressions values are calculated logarithmically.	`FillRegression()`
`regresslog12(TSID1, TSID2)` `regressMonthlyLog( TSID1,TSID2)`	Same as `regresslog()` except 12 monthly regressions values are calculated.	`FillRegression()`
`setconst(TSID,Value)`	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.	`SetConstant()`
`setconstbefore(TSID, Value,Date)`	The time series to the given value for all data on and before the specified date (`YYYY-MM or MM/YYYY`).	`SetConstant()`
`setConstantBefore()`	Set a value constant before a date/time.	`SetConstant()`
`SetMissingDataValue()`	Set the missing data value used in a StateMod time series.	See `WriteStateMod()`.

TSTool Command Summary – Obsolete Commands (continued)

Command	Description	Replacement
`setQueryPeriod(Start,End)`	Set the global period to query databases and read from files.	`SetInputPeriod()`
`-sqlserver`	Specify that SQL Server is used for HydroBase.	`OpenHydroBase()` and configuration information. SQL Server is also now the default because Microsoft Access is no longer supported.
`-slist File`	Create time series from a list file.	`CreateFromList()`
`-units value`	Output using the specified units (default is to use database units).	No longer used. If necessary, units can be converted by a number of commands including `ConvertUnits()`.
`-w#[,#]`	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.	`SetWarningLevel()`
`-wy`	Output in water year format.	`SetOutputYearType()`

9 Quality Control

Version 10.00.01, 2010-05-16

This chapter discusses how TSTool software is quality controlled and how to use TSTool for quality control.

9.1 Using TSTool to Quality Control Data

TSTool can be used to perform quality control on time series data. Two primary commands are:

1. `CheckTimeSeries()` – this command checks individual time series values for out of range, missing, greater than, etc. Values that are detected can be flagged with a string and optionally can be set to missing. The HTML summary (see `WriteSummary()` command) will indicate flagged values and graph products can label data points with flags. Flags can be written to data management systems if flags are supported.
2. `CheckTimeSeriesStatistic()` – this command calculates a statistic (e.g., count of data values greater than a criteria) and then checks the statistic against a criteria (e.g., is count greater than a criteria).

The above commands can be used to check for out of range and other unusual data conditions. If values are replaced with missing, the missing values can be filled using the fill commands. Additional quality control can be performed by using a combination of other commands. For example, the `NewStatisticYearTS()` command computes an annual statistic time series, which can then be checked using one of the above commands. The `Cumulate()` command also is useful for visualizing trends.

9.2 Using TSTool to Quality Control System Functionality

The previous section described how TSTool can be used to quality control data. Another aspect of quality control is ensuring that a system is working. For example, TSTool accesses data from many input sources and it may be difficult for the maintainers of data systems to know if systems are running (is the system up?) and that data are being properly returned (are values reasonable?). TSTool can be used as a test runner to check for system uptime, for example:

1. Blanket test. Generate a command file to read a representative number of time series (or all) from a system. It may be necessary to set the input period to be short. If any errors occur, they need to be resolved.
2. Data count test. Generate a command file to read a representative number of time series. Specify the input period as an appropriate value (e.g., short for real-time data). Use the `CheckTimeSeriesStatistic()` command to generate a count of data values in the period and check for reasonable values.

Other tests also can be performed. The following section describes how very specific tests can be run to test software functionality. A similar approach can be used to test systems, if it is known that expected results from the system will not change over time.

9.3 Quality Control for TSTool Software

TSTool software provides many data processing commands. Each command typically provides multiple parameters. The combination of commands and parameters coupled with potential data changes and user errors can make it difficult to confirm that TSTool software is itself performing as expected. To address this quality control concern, several commands have been built into TSTool to facilitate using TSTool itself to test functionality. Test cases can be defined for each command, with test cases for various combinations of parameters. The suite of all the test cases can then be run to confirm that the version of TSTool does properly generate expected results. This approach performs regression testing of the software and utilizes TSTool's error-handling features to provide visual feedback during testing.

Test cases are developed by software developers as new features are implemented, according to the following documentation. However, users can also develop test cases and this is encouraged to ensure that all combinations of parameters and input data are tested. Providing verified test data and results prior to new development will facilitate the new development.

9.3.1 Writing a Single Test Case

A single test case is illustrated by the following example (indented lines indicate commands that are too long to fit on one line in the documentation).

```
# Test filling with interpolation where maximum gap interval to fill is 2.
StartLog(LogFile="Results/Test_FillInterpolate_MaxIntervals=2.TSTool.log")
RemoveFile(InputFile="Results/Test_FillInterpolate_MaxIntervals=2_out.dv",
    IfNotFound=Ignore)
NewPatternTimeSeries(Alias="ts1_day",NewTSID="ts1...Day",Description="test data 1",
    SetStart="2000-01-01",SetEnd="2003-05-13",
    PatternValues="1,2,3,2,1,-999,5,1,-999,-999,-999,1,3,5")
FillInterpolate(TSList=AllMatchingTSID,TSID="ts1_day",MaxIntervals=2)
# Uncomment the following command to regenerate expected results.
# WriteDateValue(OutputFile="ExpectedResults/Test_FillInterpolate_MaxIntervals=2_out.dv")
WriteDateValue(OutputFile="Results/Test_FillInterpolate_MaxIntervals=2_out.dv")
CompareFiles(InputFile1="ExpectedResults/Test_FillInterpolate_MaxIntervals=2_out.dv",
    InputFile2="Results/Test_FillInterpolate_MaxIntervals=2_out.dv",IfDifferent=Warn)
```

Example Test Case Command File

The purpose of the test case command file is to regenerate results and then compare the results to previously generated and verified expected results. The example illustrates the basic steps that should be included in any test case:

1. **Start a log file to store the results of the specific test case.** The previous log file will be closed and the new log file will be used until it is closed. The log file is not crucial to the test but helps with troubleshooting if necessary (for example if evaluating the test case output when run in a test suite, as explained later in this chapter).
2. **Remove the results that are to be generated by the test**. This is necessary because if the software fails and old results match expected results, it may appear that the command was successful. Using the `IfNotFound=Ignore` parameter is useful because someone who is running the tests for the first time may not have previous results. Test developers should use `IfNotFound=Warn` when setting up the test to confirm that the results being removed match the name that is actually generated in a later command, and then switch to `IfNotFound=Ignore`.

3. **Generate or read time series data**. The `NewPatternTimeSeries()` command is used in the example to create a time series of repeating values. This is a useful technique because it allows full control over the initial data and minimizes the number of files associated with the test. Synthetic data are often appropriate for simple tests. If the test requires more complicated data, then time series can be read from a DateValue or other time series file. For example, if functionality of another software program is being implemented in TSTool, the data file from the original software may be used.
4. **Process the time series using the command being tested**. In the example, the `FillInterpolate()` command is being tested. In many cases, a single command can be used in this step. However, in some cases, it is necessary to use multiple commands. This is OK as long as each command or the sequence is sufficiently tested with appropriate test cases.
5. **Write the results**. The resulting time series are written to a standard format. The DateValue format is useful for general testing because it closely represents all time series properties. Note that two write commands are used in the example – one writes the expected results and the other writes the results from the current test. The expected results should only be written when the creator of the test has confirmed that it contains verified values. In the example, the command to write expected results is commented out because the results were previously generated. Some commands do not process time series; therefore, the `WriteProperty()` and `WriteTimeSeriesProperty()` commands can be used to write processor properties (e.g., global output period) and time series properties (e.g., data limits). Additional properties will be enabled as the software is enhanced.
6. **Compare the expected results and the current results**. The example uses the `CompareFiles()` command to compare the DateValues files generated for the expected and current results. This command omits comment lines in the comparison because file headers often change due to dynamic comments with date/time. If the software is functioning as expected, the data lines in the file will exactly match. The example illustrates that if the files are different, a warning will be generated because of the `WarnIfDifferent=True` parameter. Other options for comparing results include:
    a. **Use the CompareTimeSeries() command.** This command expects to find matching time series and will compare data values to a precision. For example, read one time series from a DateValue file and then compare with the current time series in memory. Using this command avoids potential issues with the DateValue or other file formats changing over time (and requiring the expected results to be reverified); however, doing a file comparison is often easier to troubleshoot because a graphical difference program can visually illustrate differences that need to be evaluated.
    b. **If testing a read/write command, compare the results with the original data file**. For example, if the test case is to verify that a certain file format is properly read, then there will generally also be a corresponding write command. The test case can then consist of a command to read the file, a command to write the results, and a comparison command to compare the two files. This may not work if the header of the file uses comment lines that are not recognized by the `CompareFiles()` command.

If the example command file is opened and run in TSTool, it will produce time series results, the log file, and the output file. If the expected and current results are the same, no errors will be indicated. However, if the files are different, a warning indicator will be shown in the command list area of the main window next to the **`CompareFiles()`** command.

General guidelines for defining test cases are as follows. Following these conventions will allow the test cases to be incorporated into the full test suite.

- Define the test case in a folder matching the command name.

- Name the command file with prefix *Test_*, extension *.TSTool*, and use the following guidelines:
  - for the default case using the filename pattern *Test_CommandName.TSTool*
  - If there is a reason to define a test for a specific data set or input, add additional information to the filename, for example: *Test_CommandName_RiverX.TSTool* or *Test_CommandName_6Hour.TSTool*
  - If defining a test for legacy syntax, name the command file as follows (and see the `@readOnly` comment tag described in **Section 9.3.3**): *Test_CommandName_Legacy.TSTool*
  - If defining a test for parameter values other than the default values, use a command file name similar to the following, where the parameters are listed at the end of the file name body: *Test_CommandName_Param1=Value1,Param2=Value2.TSTool*
    Although this can result in very long names, the explicit naming clarifies the purpose of the test. The name of the example command file shown above is *Test_FillInterpolate_MaxIntervals=2.TSTool.*
- Add a short comment to the top of the test case explaining the test.
- Use as little data as possible to perform the test – long time series cause tests to run longer and take up more space in the repository that is used for revision control. Even though hundreds or thousands of tests may ultimately be defined, it is important to be able to run them in a short time to facilitate testing.
- If possible, test only one command in the test – more complicated testing is described in **Section 9.3.4**.
- If an input file is needed, place it in a folder named *Data*, if necessary copying the same input from another command – this may require additional disk space but ensures that each command can stand alone.
- Write the expected results to a folder named *ExpectedResults.*
- Write the generated results and other dynamic content, including log file, to a folder named *Results*.
- (Recommended) When creating output files, use *_out* in the filename before the extension and use an extension that is appropriate for the file content – this helps identify final output products in cases where intermediate files might be produced.

9.3.2 Creating and Running a Test Suite

The previous section described how to define a single test case. However, opening and running each test case command file would be very tedious and inefficient. Therefore, TSTool provides a way to generate and run test suites, which is the approach taken to perform a full regression test prior to a software release.

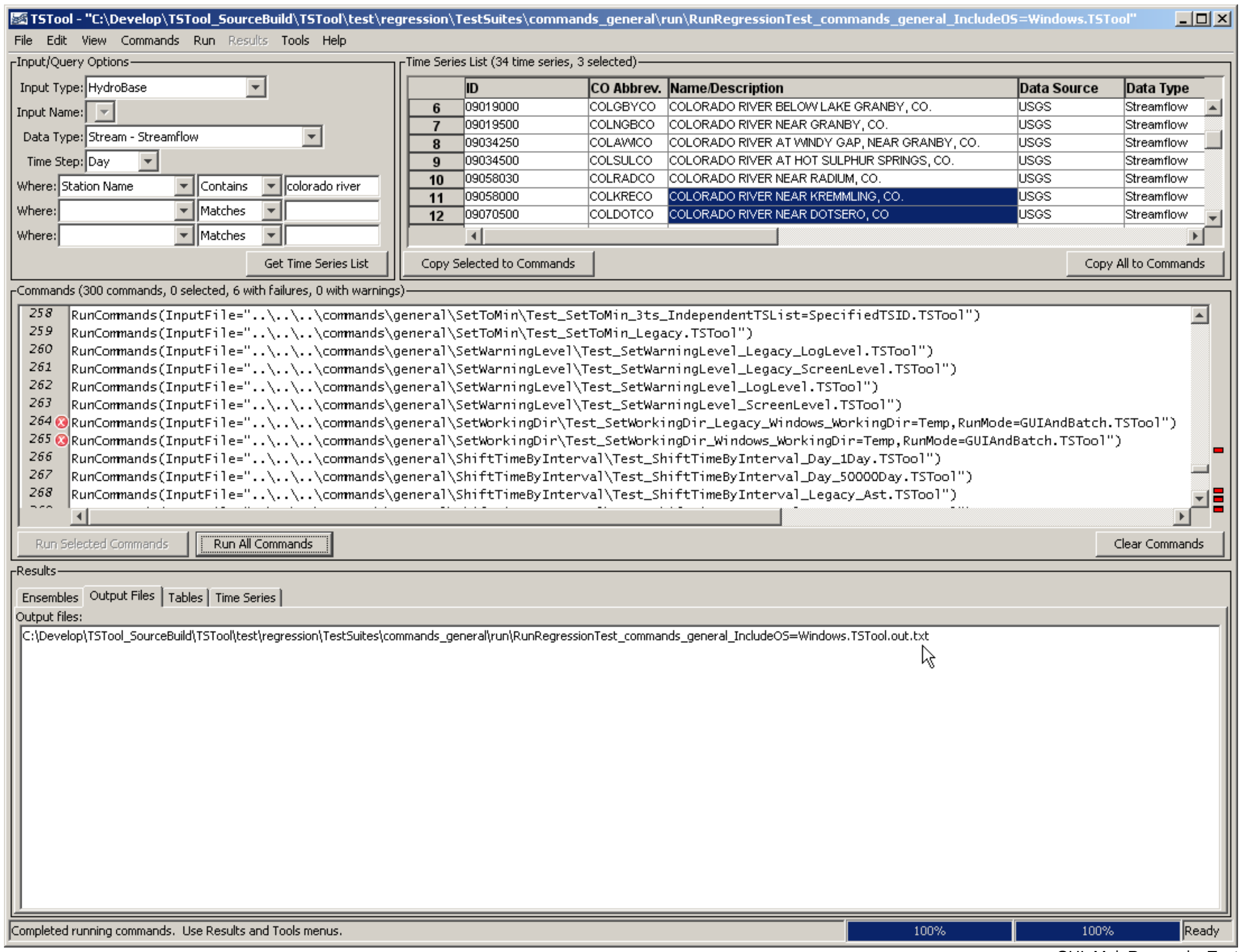
The following example command file illustrates how to create a test suite:

```
#
# Create the regression test runner for the
# TSTool/test/regression/TestSuites/commands_general files.
#
# Only command files that start with Test_ are included in the output.
# Don't append the generated commands, in order to force the old file to be
# overwritten.
#
CreateRegressionTestCommandFile(SearchFolder="..\..\..\commands\general",
   OutputFile="..\run\RunRegressionTest_commands_general_IncludeOS=Windows.TSTool",
   Append=False,IncludeTestSuite="*",IncludeOS="Windows")
```

When the command file is run, it searches the indicated search folder for files matching the pattern *Test_*.TSTool*. It then uses this list to create a command file with contents similar to the following example (its contents are truncated in the following figure due to length). This file will be listed as an output file after running the above command file. The `IncludeTestSuite` and `IncludeOS` parameters are described in **Section 9.3.3**.

```
# File generated by...
# program:      TSTool 9.00.04 (2009-01-20)
# user:         sam
# date:         Tue Jan 20 22:56:17 MST 2009
# host:         SOPRIS
# directory:    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\TestSuites\commands_general\create
# command line: TSTool -home test/operational/RTi
#
# The following 287 test cases will be run to compare results with expected results.
# Individual log files are generally created for each test.
# The following test suites from @testSuite comments are included: *
# Test cases for @os comments are included: Windows
StartRegressionTestResultsReport(
    OutputFile="RunRegressionTest_commands_general_IncludeOS=Windows.TSTool.out.txt")
RunCommands(InputFile="..\..\..\commands\general\Add\Test_Add_1.TSTool")
RunCommands(InputFile="..\..\..\commands\general\Add\Test_Add_Ensemble_1.TSTool")
RunCommands(InputFile="..\..\..\commands\general\AddConstant\Test_AddConstant_1.TSTool")
RunCommands(InputFile="..\..\..\commands\general\AddConstant\Test_AddConstant_Legacy_Ast.TSTool")
RunCommands(InputFile="..\..\..\commands\general\AddConstant\Test_AddConstant_Legacy_NoAst.TSTool")
…omitted…
RunCommands(InputFile="..\..\..\commands\general\WriteSummary\Test_WriteSummary_1.TSTool")
```

The above command file can then be opened and run. Each `RunCommands()` command will run a single test case command file. Warning and failure statuses from each test case command file are propagated to the test suite `RunCommands()` command. The output from running the test suite will be all of the output from individual test cases (in the appropriate *Results* folders) plus the regression test report provided in the TSTool ***Results*** list in the main window. An example of the TSTool main window after running the test suite is shown in the following figure. Note the warnings and errors, which should be addressed before releasing the software (in some cases commands are difficult to test and more development on the test framework is needed).


TSTool - "C:\Develop\TSTool_SourceBuild\TSTool\test\regression\TestSuites\commands_general\run\RunRegressionTest_commands_general_IncludeOS=Windows.TSTool"
File Edit View Commands Run Results Tools Help
Input/Query Options
Input Type: HydroBase
Input Name:
Data Type: Stream - Streamflow
Time Step: Day
Where: Station Name Contains colorado river
Where: Matches
Where: Matches
Get Time Series List
Time Series List (34 time series, 3 selected)
Copy Selected to Commands
Copy All to Commands
Commands (300 commands, 0 selected, 6 with failures, 0 with warnings)
Run Selected Commands
Run All Commands
Clear Commands
Results
Ensembles
Output Files
Tables
Time Series
Output files:
Completed running commands. Use Results and Tools menus.
100%
100%
Ready

	ID	CO Abbrev.	Name/Description	Data Source	Data Type
6	09019000	COLGBYCO	COLORADO RIVER BELOW LAKE GRANBY, CO.	USGS	Streamflow
7	09019500	COLNGBCO	COLORADO RIVER NEAR GRANBY, CO.	USGS	Streamflow
8	09034250	COLAWICO	COLORADO RIVER AT WINDY GAP, NEAR GRANBY, CO.	USGS	Streamflow
9	09034500	COLSULCO	COLORADO RIVER AT HOT SULPHUR SPRINGS, CO.	USGS	Streamflow
10	09058030	COLRADCO	COLORADO RIVER NEAR RADIUM, CO.	USGS	Streamflow
11	09058000	COLKRECO	COLORADO RIVER NEAR KREMMLING, CO.	USGS	Streamflow
12	09070500	COLDOTCO	COLORADO RIVER NEAR DOTSERO, CO	USGS	Streamflow

```
RunCommands(InputFile="..\..\..\commands\general\SetToMin\Test_SetToMin_3ts_IndependentTSList=SpecifiedTSID.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetToMin\Test_SetToMin_Legacy.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetWarningLevel\Test_SetWarningLevel_Legacy_LogLevel.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetWarningLevel\Test_SetWarningLevel_Legacy_ScreenLevel.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetWarningLevel\Test_SetWarningLevel_LogLevel.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetWarningLevel\Test_SetWarningLevel_ScreenLevel.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetWorkingDir\Test_SetWorkingDir_Legacy_Windows_WorkingDir=Temp,RunMode=GUIAndBatch.TSTool")
RunCommands(InputFile="..\..\..\commands\general\SetWorkingDir\Test_SetWorkingDir_Windows_WorkingDir=Temp,RunMode=GUIAndBatch.TSTool")
RunCommands(InputFile="..\..\..\commands\general\ShiftTimeByInterval\Test_ShiftTimeByInterval_Day_1Day.TSTool")
RunCommands(InputFile="..\..\..\commands\general\ShiftTimeByInterval\Test_ShiftTimeByInterval_Day_50000Day.TSTool")
RunCommands(InputFile="..\..\..\commands\general\ShiftTimeByInterval\Test_ShiftTimeByInterval_Legacy_Ast.TSTool")
```

C:\Develop\TSTool_SourceBuild\TSTool\test\regression\TestSuites\commands_general\run\RunRegressionTest_commands_general_IncludeOS=Windows.TSTool.out.txt

GUI_MainRegressionTest

TSTool Main Interface Showing Regression Test Results

An excerpt from the output file is shown below (normally the test number would be sequential from 1 to the number of tests but only a few examples are included below).

```
# File generated by...
# program:      TSTool 9.00.04 (2009-01-20)
# user:         sam
# date:         Wed Feb 25 16:59:52 MST 2009
# host:         SOPRIS
# directory:    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\TestSuites\commands_general\run
# command line: TSTool
#
# The test status below may be PASS or FAIL.
# A test can pass even if the command file actual status is FAILURE, if failure is expected.
#     Test   Commands   Commands
#     Pass/  Expected   Actual
# Num Fail   Status     Status     Command File
#----------------------------------------------------------------------
   1  PASS   SUCCESS    SUCCESS    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       Add\Test_Add_1.TSTool
   2  PASS   SUCCESS    SUCCESS    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       Add\Test_Add_Ensemble_1.TSTool
  34  PASS   Warning    WARNING    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       CreateFromList\Test_CreateFromList_InputType=HydroBase,
                                       IDCol=1,DataSource=DWR,DataType=DivTotal,
                                       Interval=Month,IfNotFound=Ignore.TSTool
  35  PASS   Failure    FAILURE    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       CreateFromList\Test_CreateFromList_InputType=HydroBase,
                                       IDCol=1,DataSource=DWR,DataType=DivTotal,
                                       Interval=Month,IfNotFound=Warn.TSTool
  36  PASS   Success    SUCCESS    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       CreateFromList\Test_CreateFromList_InputType=HydroBase,
                                       OutputPeriod,IDCol=1,DataSource=DWR,DataType=DivTotal,
                                       Interval=Month,IfNotFound=Default.TSTool
  37  PASS   Warning    WARNING    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       CreateFromList\Test_CreateFromList_Legacy.TSTool
  38  PASS   Failure    FAILURE    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       CreateTraces_Alias\Test_CreateTraces_Legacy_1.TSTool
 251 *FAIL*  SUCCESS    FAILURE    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       SetWorkingDir\Test_SetWorkingDir_Legacy_Windows_WorkingDir=Temp,
                                       RunMode=GUIAndBatch.TSTool
 252 *FAIL*  SUCCESS    FAILURE    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       SetWorkingDir\Test_SetWorkingDir_Windows_WorkingDir=Temp,
                                       RunMode=GUIAndBatch.TSTool
 287  PASS   SUCCESS    SUCCESS    C:\Develop\TSTool_SourceBuild\TSTool\test\regression\commands\general\
                                       WriteTimeSeriesProperty\
                                       Test_WriteTimeSeriesProperty_PropertyName=DataLimitsOriginal.TSTool
#----------------------------------------------------------------------
# FAIL count = 6
# PASS count = 281
```

A test passes if its expected status (by default SUCCESS) matches the actual status, and the test fails otherwise. Note that there are cases where a test case is actually intended to fail, in order to test that TSTool is properly detecting and handling the failure (rather than ignoring it or crashing).

The features built into TSTool can therefore be used to efficiently test the software, contributing to increased software quality and efficient software releases. See the next section for more information on controlling the test process.

9.3.3 Controlling Tests with Special Comments

The previous two sections described how to define individual test cases and how to automatically create and run a test suite comprised of test cases. However, there are special conditions that will cause the normal testing procedures to fail, in particular:

- tests depend on a database that is not available
- tests depend on a database version that is not available (data in the "default" database have changed)
- tests can only be run on a certain operating system
- tests depend on a specific environment configuration that is not easily reproduced for all users

Any of these conditions can cause a test case to fail, leading to inappropriate errors and wasted time tracking down problems that do not exist. To address this issue, TSTool recognizes special comments that can be included in test case command files. The following table lists tags that can be placed in # comments in command files to provide information for to the `CreateRegressionTestCommandFile()` command and command processor. The syntax of the special comments is illustrated by the following example:

```
#@expectedStatus Failure
```

Special # comment Tags

Parameter	Description

`@expectedStatus Failure`
 	
`@expectedStatus Warning`
	The `RunCommands()` command `ExpectedStatus` parameter is by default `Success`. However, a different status can be specified if it is expected that a command file will result in `Warning` or `Failure` and still be a successful test. For example, if a command is obsolete and should generate a failure, the expected status can be specified as `Failure` and the test will pass. Another example is to test that the software properly treats a missing file as a failure.

`@os Windows`
`@os UNIX`
	Using this tag indicates that the test is designed to work only on the specified platform and will be included in the test suite by the `CreateRegressionTestCommandFile()` command only if the `IncludeOS` parameter includes the corresponding operating system (OS) type. This is primarily used to test specific features of the OS and similar but separate test cases should be implemented for both OS types. If the OS type is not specified as a tag in a command file, the test is always included.

`@readOnly`	Use this tag to indicate that a command file is read-only. This is useful when legacy command files are being tested because TSTool will automatically update old syntax to new. Consequently, saving the command file will overwrite the legacy syntax and void the test. If this tag is included, the TSTool interface will warn the user that the file is read-only and will only save if the user indicates to do so.
`@testSuite ABC`	Indicate that the command file should be considered part of the specified test suite, as specified with the `IncludeTestSuite` parameter of the `CreateRegressionTestCommandFile()` command. Do not specify a test suite tag for general tests. This tag is useful if a group of tests require special setup, for example connecting to a database. The suite names should be decided upon by the test developer.

Using the above special comment tags, it is possible to create test suites that are appropriate for specific environments. For example, using `@testSuite HydroBase` indicates that a test case should be included in the HydroBase test suite, presumably run in an environment where a connection to HydroBase has been opened. Consequently, multiple test suites can be created and run as appropriate depending on the system environment.

9.3.4 Verifying TSTool Software Using a Full Dataset

The previous sections described how to test TSTool software using a suite of test cases. This approach can be utilized when performing general tests, for example prior to a normal software release. However, there may be cases where TSTool has been used to produce a large data set and it is desirable to confirm that a software release will still create the full dataset without differences. For example, for the State of Colorado’s Decision Support Systems, large basin model data sets are created and are subject to significant scrutiny. Approaches previously described in this chapter can be utilized to verify that TSTool is functioning properly and creates the dataset files. The following procedure is recommended and uses CDSS as an example:

1. If not already installed, install the data set in its default location (e.g., *C:\CDSS\data\colorado_1_2007*) – these files **will not be modified** during testing.
2. Create a parallel folder with a name indicating that it is being used for verification (e.g., *C:\CDSS\data\colorado_1_2007_verify20090216*).
3. Copy the data set files from step 1 to the folder created in step 2 (e.g., copy to *C:\CDSS\data\colorado_1_2007_verify20090216\colorado_1_2007*) – these files **will be modified** during testing.
4. Create a TSTool command file in the folder created in step 2 that will run the tests (e.g., *VerifyTSTool.TSTool*). It is often easier to edit this command file with a text editor rather than with TSTool itself. The contents of the file are illustrated in the example below. Some guidelines for this step are as follows:
    a. Organize the command file by data set folder, in the order that data need to be created.
    b. Process every **.TSTool* command to verify that it runs and generates the same results.
    c. If command files do not produce the same results, copy the command file to a name with "-updated" or similar in the filename and then change the file until it creates the expected results. This may be required due to changes in the command, for example implementing stricture error handling. These command files can then be shared with maintainers of the data set so that future releases can be updated.
    d. As tests are formalized, it may be beneficial to save a copy of this file with the original data set so future tests can simply copy the verification command file rather than recreating it (e.g., save in a *QualityControl* folder in the master data set). This effort will allow the creator of the data set to quality control their work as well as helping to quality control the software.
5. Run the command file – any warnings or failures should be evaluated to determine if they are due to software or data changes. Software differences should be evaluated by software developers. It may be necessary to use command parameters such as `Version`, available for some commands, to recreate legacy data formats.

The following example command file illustrates how TSTool software is verified using the full data set (indented lines indicate commands that are too long to fit on one line in the documentation). Note that intermediate input files that would normally be modified by other software (e.g., StateDMI for CDSS data sets) could impact TSTool verification. However, a similar quality control procedure can be implemented for StateDMI.

Guidelines for setting up the each test in the command file are as follows:

1. Remove output files that are generated from each individual command file that is run using `RemoveFile()` commands. This will ensure that test does not use old results for its output comparison.
2. Run each individual command file using the `RunCommands()` command.
3. Compare the results of the run with the original data set file using the `CompareFiles()` command.

```
StartLog(LogFile="VerifyTSTool.TSTool.log")
# This command file verifies the TSTool functionality by recreating a released
# StateMod/StateCU data set.  The general process is as follows:
# 1) Copy the entire original data set to this folder (e.g., do manually).
# 2) Commands below will remove output files from product and StateMod/StateCU
# folders.  This is done in case regeneration stops - don't want any confusion
# with original output and what should be created here.
# 3) Commands below will run the command files used to generate the model files.
# 4) Commands below will use CompareFile() commands to compare results.  Comment
# lines are ignored so only data differences (processing output) will be
# flagged.
# If run interactively from TSTool, indicators will show where results are
# different.  Differences must then be evaluated to determine if input data,
# process, or software have changed.  Differences may be valid.
#
##############################################################################
# Diversions
##############################################################################
RemoveFile(InputFile="colorado_1_2007\Diversions\514634.stm")
RunCommands(InputFile="colorado_1_2007\Diversions\514634.stm.commands.TSTool")
CompareFiles(InputFile1="colorado_1_2007\Diversions\514634.stm",
    InputFile2="..\colorado_1_2007\Diversions\514634.stm",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\Diversions\954699.stm")
RunCommands(InputFile="colorado_1_2007\Diversions\954699.commands.TSTool")
CompareFiles(InputFile1="colorado_1_2007\Diversions\954699.stm",
    InputFile2="..\colorado_1_2007\Diversions\954699.stm",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\Diversions\Fraser.stm")
RunCommands(InputFile="colorado_1_2007\Diversions\Fraser.commands.TSTool")
CompareFiles(InputFile1="colorado_1_2007\Diversions\Fraser.stm",
    InputFile2="..\colorado_1_2007\Diversions\Fraser.stm",WarnIfDifferent=True)
#
##############################################################################
##############################################################################
# instream
##############################################################################
##############################################################################
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005.ifm")
RunCommands(InputFile="colorado_1_2007\instream\ifm.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005.ifm",
    InputFile2="..\colorado_1_2007\statemod\cm2005.ifm",WarnIfDifferent=True)
#
##############################################################################
# reservoirs
##############################################################################
#
RemoveFile(InputFile="colorado_1_2007\reservoirs\363543-add.stm")
RemoveFile(InputFile="colorado_1_2007\reservoirs\364512-add.stm")
RemoveFile(InputFile="colorado_1_2007\reservoirs\503668-add.stm")
RemoveFile(InputFile="colorado_1_2007\reservoirs\513709-add.stm")
RemoveFile(InputFile="colorado_1_2007\reservoirs\514620-add.stm")
RemoveFile(InputFile="colorado_1_2007\reservoirs\723844-add.stm")
RemoveFile(InputFile="colorado_1_2007\reservoirs\838713-add.stm")
RunCommands(InputFile="colorado_1_2007\reservoirs\res.stm.commands.tstool")
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\363543-
add.stm",InputFile2="..\colorado_1_2007\reservoirs\363543.stm",WarnIfDifferent=True)
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\364512-add.stm",
    InputFile2="..\colorado_1_2007\reservoirs\364512-add.stm",WarnIfDifferent=True)
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\503668-add.stm",
    InputFile2="..\colorado_1_2007\reservoirs\503668-add.stm",WarnIfDifferent=True)
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\513709-add.stm",
    InputFile2="..\colorado_1_2007\reservoirs\513709-add.stm",WarnIfDifferent=True)
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\514620-add.stm",
    InputFile2="..\colorado_1_2007\reservoirs\514620-add.stm",WarnIfDifferent=True)
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\723844-add.stm",
    InputFile2="..\colorado_1_2007\reservoirs\723844-add.stm",WarnIfDifferent=True)
# CompareFiles(InputFile1="colorado_1_2007\reservoirs\838713-add.stm",
    InputFile2="..\colorado_1_2007\reservoirs\838713-add.stm",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005B.tar")
RunCommands(InputFile="colorado_1_2007\reservoirs\Btar.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005B.tar",
    InputFile2="..\colorado_1_2007\statemod\cm2005B.tar",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005C.tar")
RunCommands(InputFile="colorado_1_2007\reservoirs\Ctar.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005C.tar",
    InputFile2="..\colorado_1_2007\statemod\cm2005C.tar",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005.eom")
RunCommands(InputFile="colorado_1_2007\reservoirs\eom.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005.eom",
    InputFile2="..\colorado_1_2007\statemod\cm2005.eom",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005H.tar")
RunCommands(InputFile="colorado_1_2007\reservoirs\Htar.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005H.tar",
    InputFile2="..\colorado_1_2007\statemod\cm2005H.tar",WarnIfDifferent=True)
#
##############################################################################
# streamSW
##############################################################################
#
RemoveFile(InputFile="colorado_1_2007\streamSW\404657.stm")
RunCommands(InputFile="colorado_1_2007\streamSW\404657.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\streamSW\404657.stm",
    InputFile2="..\colorado_1_2007\streamSW\404657.stm",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005.rbd")
RunCommands(InputFile="colorado_1_2007\streamSW\rbd.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005.rbd",
    InputFile2="..\colorado_1_2007\statemod\cm2005.rbd",WarnIfDifferent=True)
#
RemoveFile(InputFile="..\colorado_1_2007\statemod\cm2005.rid")
RunCommands(InputFile="colorado_1_2007\streamSW\rid.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005.rid",
    InputFile2="..\colorado_1_2007\statemod\cm2005.rid",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\statemod\cm2005.rih")
RunCommands(InputFile="colorado_1_2007\streamSW\rih.commands.tstool")
CompareFiles(InputFile1="colorado_1_2007\statemod\cm2005.rih",
    InputFile2="..\colorado_1_2007\statemod\cm2005.rih",WarnIfDifferent=True)
#
##############################################################################
# TSTool - do after others, in case differences might cascade
##############################################################################
#
RemoveFile(InputFile="colorado_1_2007\TSTool\fill2005.pat")
RunCommands(InputFile="colorado_1_2007\TSTool\fill2005.pat.commands.TSTool")
CompareFiles(InputFile1="colorado_1_2007\TSTool\fill2005.pat",
    InputFile2="..\colorado_1_2007\TSTool\fill2005.pat",WarnIfDifferent=True)
#
RemoveFile(InputFile="colorado_1_2007\TSTool\Patgage.xbg")
RunCommands(InputFile="colorado_1_2007\TSTool\pattern_gage_raw_data.commands.TSTool")
CompareFiles(InputFile1="colorado_1_2007\TSTool\patgage.xbg",
    InputFile2="..\colorado_1_2007\TSTool\patgage.xbg",WarnIfDifferent=True)
```

This page is intentionally blank.

10 Spreadsheet Integration

Version 10.20.00, 2013-03-30

TSTool and spreadsheet software offer many similar capabilities. However, there are significant differences in the software features and approaches to structuring and processing data. It may be appropriate to use one tool and not the other for certain tasks. And, it also may be appropriate to use the tools in conjunction (whether in sequence or in parallel). This chapter provides a comparison of TSTool and Microsoft Excel and describes how they can be integrated. Microsoft Excel is the spreadsheet of choice in this chapter; however, other spreadsheet software, such as Open Office Calc, are similar, in particular when **.xls*, **.xlsx*, and **.csv* file formats are used. Common TSTool and Excel integration needs include:

- How can Microsoft Excel be executed from TSTool?
- How can TSTool be executed from Excel?
- How can data in an Excel file be extracted (pulled) by TSTool?
- How can data in TSTool be inserted (pushed) into an Excel file?
- How can data produced by TSTool be imported (pulled) into an Excel file?
- How can data produced by Excel be exported (pushed) into a TSTool file?
- How can TSTool results (e.g., charts and formatted reports) be made to look “more like Excel” (this may be related to functionality or simply a cosmetic preference for Excel)

Integrating TSTool with Excel can be challenging because:

- TSTool software is written in Java and Excel uses Microsoft Technologies – the toolkits for integration are different and involve different knowledge and skills
- Excel file formats may be binary, often are complex, and formats may not have clear documentation (it often is necessary to reverse engineer Microsoft formats) and
- The complexity of Excel files make it difficult to simply manipulate the files and therefore libraries need to be used to retain the integrity of the files

TSTool and Excel can be run in sequence to perform data processing. This approach can involve manual or automated execution. Both approaches require appropriate integration, whether using intermediate data files or direct connections, and direct connections must ensure that sufficient error handling is in place.

Many software developers and users face the challenges of integrating their software with Excel and various solutions are available. The appropriateness of any approach depends on the specific need and technologies that are available. This remainder of this chapter explains how to integrate TSTool with Excel, and will be expanded as successful techniques are confirmed. The focus is on recent software versions and technologies.

10.1 TSTool and Excel Comparison

The following table compares features of TSTool and Excel. In summary:

- TSTool processes data using a sequential workflow of commands, whereas Excel processes data using an internal solver that understands data dependencies.

- TSTool provides immediate feedback on errors when editing commands, but does not fully process data until commands are run; Excel processes formulas immediately and provides feedback on errors.
- TSTool is designed to separate data and processing logic whereas Excel allows users to mix.
- TSTool data representations are generally human-readable, whereas Excel files are binary and can be difficult to interpret.

TSTool and Excel Comparison

Feature	TSTool	Excel
Software language	Java	Microsoft technologies, depend on version.
File extension	**.TSTool* for commands, others for data and configuration files.	**.xls* (pre-2007 version), binary **.xlsx* (2007 version) zipped XML.
Data file formats supported	See datastore appendices for time series formats, also DBF and CSV files for tables.	**.xls*, **.xlsx*, **.dbf*, and other formats.
In-memory data representation	Lists of objects: • Time series • Tables • Processor properties	Workbook of worksheets, each containing a grid of cells, with each cell having a type (e.g., number, text, formula) and formatting properties.
Data processing logic representation	Commands text, saved in text command file. Commands used named parameters (`Parameter="Value"`) that allow defaults and any parameter order. Standards are enforced or recommended by TSTool. For example, TSTool uses a standard time series identifier (TSID) convention.	Formulas defined for cells (select cell to see formula). Formula arguments must be in a specific order, although arguments are optional in some cases. Excel is a free-form tool and does not impose many standards. However, there are limitations such as character restrictions for named ranges.
Data processing mechanism	Commands access data objects and manipulates their contents.	A formula in a cell modifies one or more cells.
Command editing	Editor dialogs are provided for editing commands in TSTool. Text command files can be edited directly.	Formula editors are provided and can also be edited as the cell contents.
Data and processing separation	Processing logic and graph configuration is clearly separated from data.	Data can be saved on separate sheets, with computations on other sheets. However, it is easy to mix data and processing logic.
User-defined code	• Call external Python script or other software externally (see General commands) • Templates • Plug-in commands (planned for future)	• Macros programmed in VBA • Call external programs • Third party extensions • Microsoft libraries (e.g., available in IronPython)
Third-party plug-ins	Not yet supported in integrated way but can read/write file formats and call Python, etc.	Available from third parties.

Feature	TSTool	Excel
Processing workflow	Sequential, first command to last (subset of commands can be executed).	Based on cell formula dependencies. (Excel contents are kept up to date as per cell contents and user interactions).
Transparency	Commands and data are visible as text, can review sequential processing.	Formulas can be viewed, sequence of processing is determined by Excel based on cell dependencies and may not be obvious to user.
Data identifiers	Time series and tables have identifiers. Time series also have aliases. Time series identifiers are based on TSTool conventions and data from the original source.	Named ranges can be assigned and Excel allows tables to be defined using column headings.
Data size limitations	None, other than memory of computer and Java virtual machine.	Excel row and column limits vary by Excel version.
Scalability	Built into design of data management and processing, for example with TSList command parameters that accept wildcards, database query filters, and templates.	Add more worksheets, columns, and rows.
Templates	Used to scale processing of large amounts of data.	Not available (have to replicate worksheets, rows, columns).
Automation	Fundamental part of TSTool design.	Workbooks tend to be an accumulation of data and processing, although Excel can be called in a process to perform specific work.
Command line (batch mode)	Run with: `tstool -commands …`	Can run `excel.exe` with switches.
Time series graphs	• Built in with default properties • Extensive graph properties • Can automate graphs using text time series product files and `ProcessTSProduct()` command. • Focus on time series, not general data series • Can graph different data intervals on same chart	• Built in charts • Use third party tools • Time series can be graphed but are essentially arrays of numbers • Cannot easily graph different intervals on same chart due to grid formatting of data
Database connectivity	• Integrated for specific datastores • Use ODBC DSN defined for Excel workbook and GenericDatabaseDataStore	• Use ODBC DSN defined for Excel workbook
Accessing external data	Use read commands to read from standard file formats, databases, and web services, including delimited files.	Use features under the ***Data*** menu, including accessing from databases, web services, and files.
Missing data	Handled transparently in most cases, including special data values (e.g.,	No specific handling, typically must ensure blanks in data cells so that

Feature	TSTool	Excel
	`NaN, -999`).	missing values will not generate errors.
Data units	Handled transparently in most cases, based on units in time series metadata, with checks in place to prevent incompatible manipulation.	No specific handling. User must guard against incompatible manipulation.
Data validation	Commands are available to check and compare data.	Formulas can be used, can enable data validation for input controls.
Software/logic testing	Built in framework and commands available to automate testing.	Not sure.
Logging	Built in, with text log file.	Not sure.

10.2 Running TSTool from Excel

Reasons for running TSTool from Excel include:

- TSTool is used to acquire and/or process data as input to Excel
- TSTool performs an analysis/visualization function

To run TSTool from Excel, run it like any other external system call, using a VBA macro in Excel (see: "How to Launch a Win32 Application from Visual Basic http://support.microsoft.com/kb/129797). TSTool can be run in batch mode using a command line:

```
TSTool -commands CommandFile.TSTool
```

See the **Getting Started** chapter for more information about running TSTool in batch mode. In the future, a TSTool server mode may be implemented to allow Excel to "drive" TSTool, for example using REST web services.

10.3 Running Excel from TSTool

TSTool can run Excel by using its `RunProgram()` command, and appropriate Excel command-line switches (see: http://office.microsoft.com/en-us/excel-help/command-line-switches-for-excel-HA010158030.aspx) (is there a better reference, for example to explain how to run in headless mode and exit?).

It also is possible to run and control Excel. However, this is not integrated into TSTool. One option is to use the TSTool `RunPython()` or `RunProgram()` command to run an IronPython script, which then interacts with Excel. For example, see:

http://www.ironpython.info/index.php/Interacting_with_Excel

10.4 Manipulating Excel Files from TSTool

Excel files (**.xls*, **.xlsx*) contain the definition of a workbook, worksheets, data, formulas, formatting, charts, and other information. Excel files are complicated and can be difficult to manipulate, especially when trying to ensure that different versions of Excel files are properly handled. Although it may be possible to modify the files directly (e.g., search and replace strings in the **.xlsx* XML), this can lead to file corruption.

A more robust way to manipulate Excel files is to utilize a software package or library that has been written specifically to manipulate Excel files through an application programmer interface (API). The following options are available for integrating TSTool and Excel and an appropriate option should be chosen for the specific environment, problem, and user:

- Use built-in TSTool features to interact directly with Excel **.xls*, **.xlsx* files:
    - The TSTool `ReadTableFromExcel()` command will read a table from Excel and the table can be processed by other table commands. For example, use the `TableToTimeSeries()` command to convert data read from Excel into time series that can be further processed with TSTool commands.
    - There currently is no TSTool command implemented to write to an Excel workbook, although this may be implemented in the future.
    - There currently is no TSTool command implemented to read time series directly from an Excel workbook (without using the `TableToTimeSeries()` command), although this may be implemented in the future.
    - There currently is no TSTool command implemented to write time series directly to an Excel workbook, although this may be implemented in the future.
    - The Apache POI package (http://poi.apache.org) is used for integration with Excel files. Functionality that is envisioned for future TSTool enhancements can be implemented using this package.
- Use built-in TSTool features to interact directly with Excel **.csv* files, which can be edited by Excel and do not include formatting, formulas, etc.:
    - Use the `ReadDelimitedFile()` command to read time series from a delimited file that was exported from Excel as a **.csv* file.
    - Use the `ReadTableFromDelimitedFile()` command to read a table from a delimited file that was exported from Excel as a **.csv* file.
    - Use the `WriteDateValue()` command to write a DateValue format file. The bottom part of the file can be read into Excel as a delimited file. A command may be implemented in the future to write time series to a delimited file (with no extra information).
    - Use the `WriteTableToDelimitedFile()` command to write a TSTool table to a delimited (**.csv*) file that can be opened in Excel.
- Use built-in TSTool features to interact with an Excel file as if it is a database:
    - Define an ODBC DSN for the Excel file and use a GenericDatabaseDataStore (see **Generic Database DataStore** appendix). Use `DatabaseEngine="Excel"` and specify the `OdbcName` property in the datastore configuration file.
    - Use the `ReadTableFromDataStore()` command:
        - The Excel worksheet must be relatively simple with column headings in the first row.
        - Need a good reference for EXCEL SQL statements, in particular showing complex queries and how to write to the sheet? What are the limitations on SQL when processing Excel files?
    - Optionally use the `TableToTimeSeries()` command to convert the table to time series.
    - There currently is no command to read a datastore table into a time series (without using the `TableToTimeSeries()` command); however, this capability may be added in the future.
- Use Python software to read/write Excel files and create files that can be processed with TSTool commands, such as a **.csv* file and the `ReadTableFromDelimitedFile()` command. The

use of Python or similar adds another level of communication between TSTool and Excel, but may be necessary to achieve the level of data manipulation and integration that is required. Python versions that have been tested with TSTool's `RunPython()` command include:

- Python xlrd (http://pypi.python.org/pypi/xlrd) and xlwt modules (http://pypi.python.org/pypi/xlwt):
  - The xlwt module does not handle **.xlsx* files as of version 0.7.4
  - Requires understanding the internals of Excel data representations
- IronPython (see: http://www.ironpython.info/index.php/Interacting_with_Excel):
  - Uses native Microsoft .NET libraries (tighter integration) but IronPython lags behind Python
  - Requires understanding the internals of Excel data representations
- Jython (http://www.jython.org/):
  - Is written in Java (see the TSTool `RunPython()` command for the version that is shipped with the TSTool software installer)
  - Has potential to allow running Python scripts with tight integration to TSTool software, including access to TSTool internal objects

10.5 Manipulating TSTool Files from Excel

There may be a need to create TSTool files from Excel. For example, TSTool supports the DateValue file format, which is essentially a delimited file with additional header information with time series properties. Other TSTool time series data files are text files that adhere to specifications of data providers (e.g., model formats). TSTool time series product files that describe graphs and commands are documented and can be written out with Excel or other software. Excel macros can be written to create these files and then TSTool can be called from Excel as described above.

In the future, standard Excel macros may be distributed with TSTool to facilitate Excel import/export capabilities.

10.6 Testing Excel and TSTool Software and Command Files

The **Quality Control** chapter explains how to use TSTool features for testing, including testing commands and user-defined command files. It also is possible to use TSTool test features in conjunction with Excel:

- to understand the similarity or differences between TSTool commands and Excel formulas (differences may be by design, due to precision of analysis, etc.)
- to validate a TSTool command by comparing to Excel, in particular for more complex functionality that cannot easily be validated with visual inspection
- to compare an analysis performed with multiple TSTool commands and the corresponding Excel formulas (for example to prototype the analysis in one of the tools and then fully implement in the other)

Specific issues related to testing with Excel include:

- TSTool's limited ability to read Excel files may limit testing. For example, see the `ReadTableFromExcel()` command documentation for limitations. One known issue is that Excel formula results may not be accessible to the `ReadTableFromExcel()` command because the formula results in the cell may be computed by Excel when the workbook file is opened and is not stored in the file. In this case it may be necessary to copy the Excel worksheet, paste special into a new worksheet, copying the values (or copy formatting first and then copy values). The values can then be read and used in the test. This is not ideal for reading Excel files for operational processing but may be a reasonable work-around for defining software tests.

- Some Excel integration approaches such as using a datastore or IronPython may result in Excel software (or underlying "headless" library processes) processes running that may impose restrictions such as locking files. If issues arise the work-around may be to read Excel files directly with `ReadTableFromExcel()` and similar commands.

10.7 Making TSTool Graph Output Look Like Excel Charts

TSTool has been developed primarily as a tool that automates and streamlines data processing. Features to visualize time series have been implemented in Java whereas Microsoft Excel is developed in Microsoft technologies. It has not been a focus to implement all TSTool visualization features to match Excel; however, it is recognized that many users are familiar with Excel charting and would like to produce similar charts in TSTool. The following table lists areas where TSTool does not match Excel, the rationale for the difference, and options for improvements that can be implemented in the future.

TSTool and Excel Time Series Graph Comparison

TSTool Graph Feature Compared to Excel	Rationale for Implementation in TSTool	Potential enhancement
TSTool default graphs are simple, with no titles	The default graphs that are displayed have relatively basic formatting because TSTool processes data from many sources. For example, it is not implemented in the current software to use blue to draw all time series of data type "streamflow". Time series properties for titles are also difficult to implement in generic way.	Allow default graph properties to be configured so that TSTool defaults are more appropriate for a particular system. Users can edit many graph properties by right-clicking on graph area and use the `ProcessTSProduct()` command.
TSTool has limited fonts	The core fonts supported in Java are offered. It can be an issue if non-standard fonts are used because they are not offered on all computers.	Expand fonts based on latest Java capabilities and use standard default if font is not available.
TSTool X-axis graph labels cannot be rotated	The label features were implemented with an older version of Java that did not support rotated text. Also, the TSTool auto-labeling features for date/times provide intelligent horizontal labeling.	Enable label rotation as an option using latest Java capabilities.
TSTool Y-axis graph labels cannot be rotated	See above. Vertical Y-axis labels would be beneficial, especially for long Y-axis labels. The current work-around is to show the Y-axis label horizontally at the top of the axis.	Enable label rotation as an option using latest Java capabilities.

TSTool Graph Feature Compared to Excel	Rationale for Implementation in TSTool	Potential enhancement
TSTool lines look a bit blocky	TSTool has no limitations on the number of data points that can be graphed. However, handling missing data as gaps in lines, sometimes results in blocky transitions through sharp angles.	Improve the TSTool drawing algorithm to take advantage of improved Java drawing internals (use less moveto/lineto and instead draw more polylines). Also implement a line property to draw lines by connecting points (the current default) and as step-function to show continuous value for interval.
TSTool bar charts do not look as nice as Excel	TSTool bar charts are designed to handle large number of data points and the drawing logic sometimes results in non-optimal space between bars.	Spend more time looking at bar representation, in particular case when bars are left off the ends of the graph.
TSTool legend default is too complicated	The TSTool legend provides more information than simple graphs in Excel or other tools. The user can configure simpler legends with graph properties.	Evaluate whether default legend should be simpler, and allow user to specify a global default.
TSTool does not draw data in middle of interval (e.g., should place point in middle of month, not at start)	TSTool graphing tools can display large numbers of data points and are intended more for scientific visualization than business data visualization.	Implement a graph option for whether to plot data points at the start, middle, or end of the data interval. This has been done to control bar positions but has not been implemented for point or line graphs.
TSTool does not by default format numbers as MM/DD/YYYY	The MM/DD/YYYY format is typical for dates in the USA. However, this format does not sort well and the ISO standard YYYY-MM-DD format is used in TSTool to promote consistency.	Provide optional graph properties to override default date/time format.
TSTool does not by default format numbers with commas to separate thousands.	This was not implemented due to resource limitations and because the default of no separator is appropriate in any locale.	Provide optional properties to display separators.
TSTool does not offer pie charts and some other Excel charts	TSTool was initially developed for time series visualization. More recent enhancements have enabled features to process tables and statistics and such information may be desirable to display in less data-intensive graphs such as pie charts.	Add pie charts and other graph types as part of table visualization enhancements.

Appendix: TSTool Installation and Configuration for CDSS

CDSS Version, 10.00.02, 2011-05-22

1. Overview

This appendix describes how to install TSTool in the CDSS (Colorado's Decision Support Systems) environment. 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

Standard locations of TSTool software files are as follows. Files are normally installed on Windows on the *C:* drive but can be installed in a shared location on a server.

C: \CDSS\TSTool-Version	Top-level install directory.
bin	Software program files directory.
batik.jar*	Scalable Vector Graphics (SVG) output packages.
Blowfish.jar*	Used for encryption/security.
cdss.domain.jar*	CDSS components.
h2.jar*	H2 embedded database.
HydroBaseDMI.jar*	State of Colorado HydroBase database interface package.
jcommon.jar, jfreechart.jar	Plotting package.
jsr173_1.0_api.jar, libXMLJava.jar	XML support.
jython.jar	Jython support.
msbase.jar *mssqlserver.jar* *msutil.jar*	Microsoft SQL Server packages.
NWSRFS_DMI.jar*	National Weather Service River Forecast System (NWSRFS) package.
RiversideDB_DMI.jar*	Riverside Technology, inc., RiversideDB database package.
RTi_Common.jar*	Riverside Technology, inc. supporting packages.
SatmonSysDMI.jar*	State of Colorado Satellite Monitoring System package.
StateMod.jar*	State of Colorado's StateMod and StateCU model packages.
TSCommandProcessor.jar*	Time series command processor package.
tstool	Shell script to run TSTool on Linux and Mac.

TSTool.bat	Batch file to run TSTool using the JRE software. This may need to be edited if the installation is not standard.
TSTool.exe	Executable program to run TSTool using the JRE software, recommended over batch file.
TSTool.l4j.ini	Configuration file for *TSTool.exe* launcher.
TSTool.jar*	TSTool program components.
doc\TSTool\UserManual\	Main documentation directory for TSTool.
TSTool.pdf	TSTool documentation as PDF.
examples	Example data and command files.
logs	Directory for TSTool log files (should be writable).
system	Directory for system files.
CDSS.cfg	CDSS configuration file for HydroBase database configuration.
DATAUNIT	Data units file.
TSTool.cfg	Configuration file to modify TSTool defaults.
*jre**	Java Runtime Environment used by TSTool

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, is recommended that the normal CDSS file structure be used.

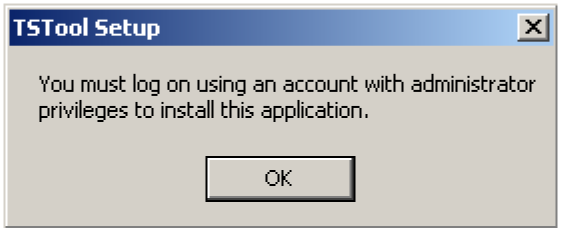
3.1 Installing TSTool from the “HydroBase data set Analysis Query Tools CD/DVD”

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. The version that is installed may be older than the version available on the CDSS web site; however, multiple versions can be installed and run independently.

3.2 Installing TSTool from the TSTool Setup File

Use the following instructions to install TSTool using the *TSTool_CDSS_Version_Setup.exe* installer program, for example if TSTool software was downloaded from the CDSS web site (*http://cdss.state.co.us*):

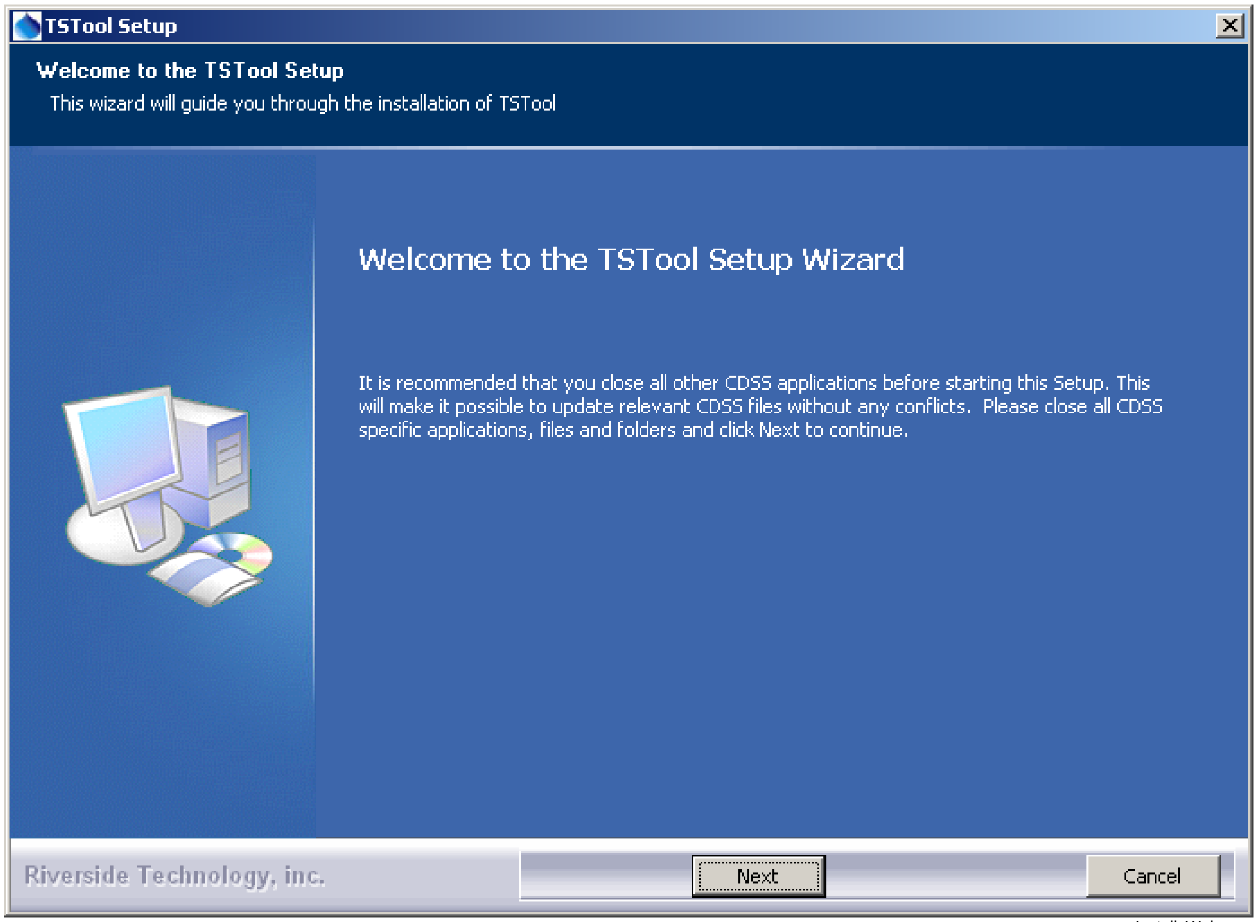
1. Run the *TSTool_CDSS_Version_Setup.exe* file by selecting from Windows Explorer, the ***Start… Run…*** menu, or from a command shell. You must be logged into the computer using an account with administrator privileges. Otherwise, the following warning will be displayed:


TSTool Setup

You must log on using an account with administrator privileges to install this application.

Install_AdministratorWarning

If you have administrative privileges, the following welcome will be displayed, and the installation can continue:


TSTool Setup

Welcome to the TSTool Setup

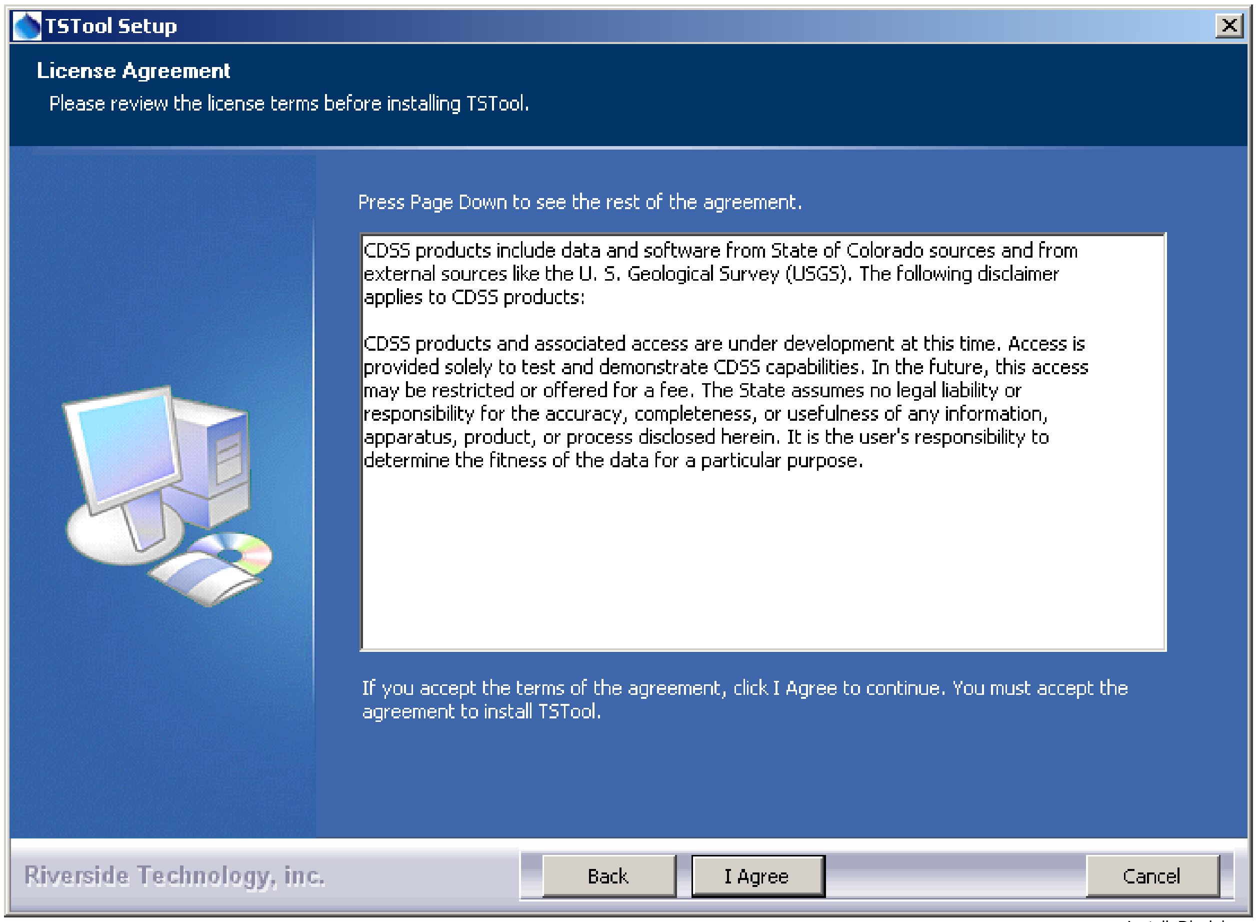
This wizard will guide you through the installation of TSTool

Welcome to the TSTool Setup Wizard

It is recommended that you close all other CDSS applications before starting this Setup. This will make it possible to update relevant CDSS files without any conflicts. Please close all CDSS specific applications, files and folders and click Next to continue.

Install_Welcome

Press ***Next*** to continue with the installation.


TSTool Setup
License Agreement
Please review the license terms before installing TSTool.
Riverside Technology, inc.
Back
I Agree
Cancel

Press Page Down to see the rest of the agreement.

CDSS products include data and software from State of Colorado sources and from external sources like the U. S. Geological Survey (USGS). The following disclaimer applies to CDSS products:

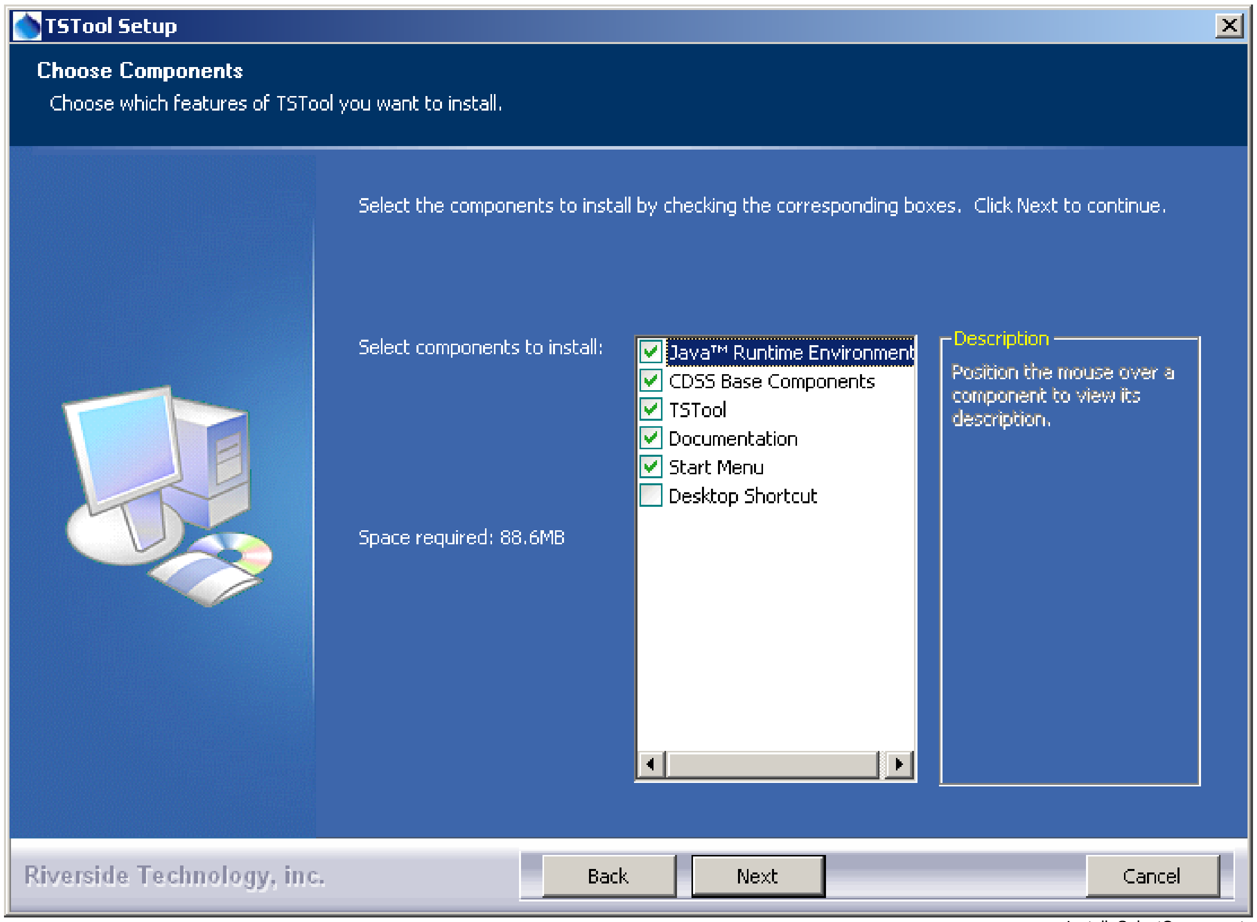
CDSS products and associated access are under development at this time. Access is provided solely to test and demonstrate CDSS capabilities. In the future, this access may be restricted or offered for a fee. The State assumes no legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed herein. It is the user's responsibility to determine the fitness of the data for a particular purpose.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install TSTool.

Install_Disclaimer

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.



TSTool Setup

Choose Components

Choose which features of TSTool you want to install.

Select the components to install by checking the corresponding boxes. Click Next to continue.

Select components to install:

- [✓] Java™ Runtime Environment
- [✓] CDSS Base Components
- [✓] TSTool
- [✓] Documentation
- [✓] Start Menu
- [ ] Desktop Shortcut

Space required: 88.6MB

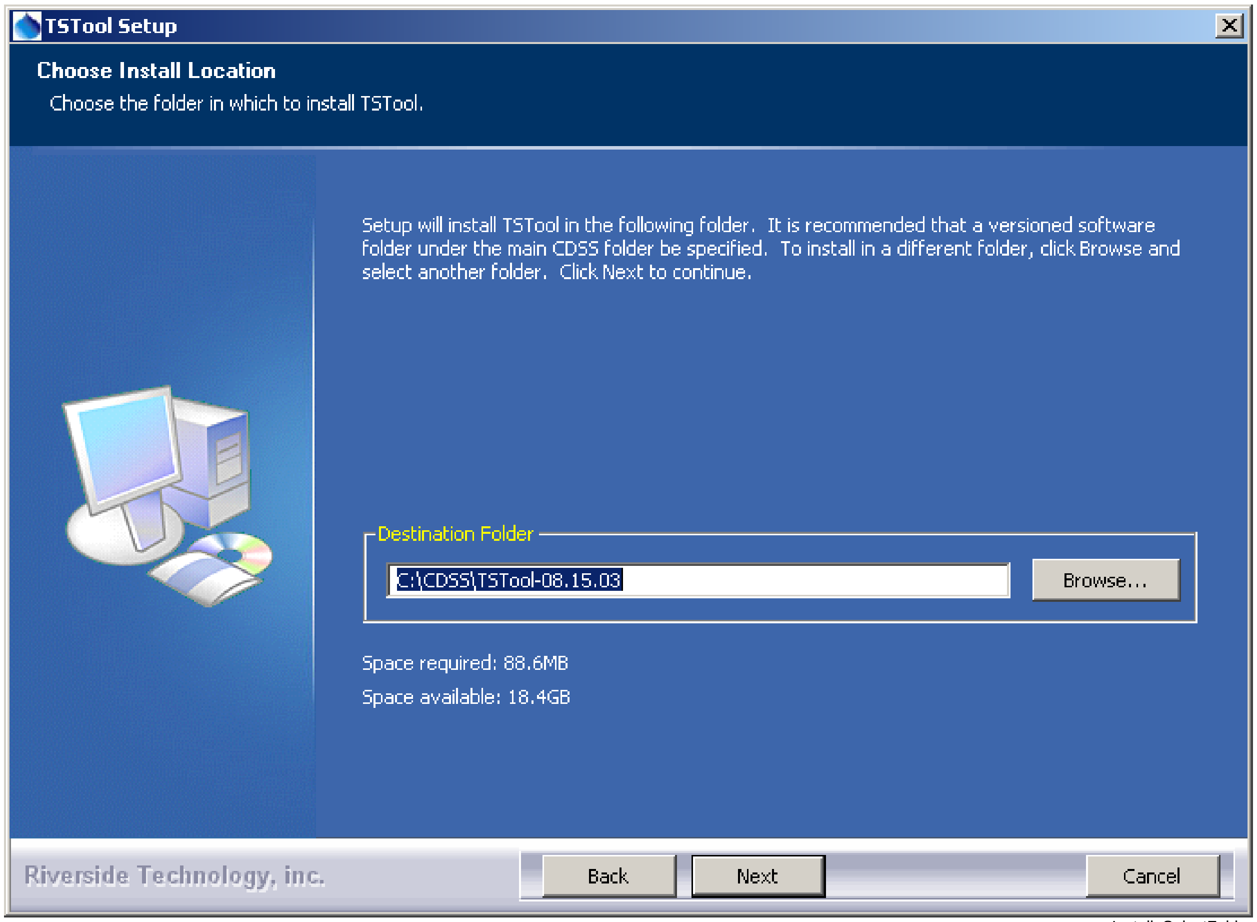
Description

Position the mouse over a component to view its description.

Install_SelectComponents

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. Multiple versions of TSTool can be installed and there are no dependencies between the versions. It is recommenced that the default install location shown is used.



TSTool Setup

Choose Install Location

Choose the folder in which to install TSTool.

Setup will install TSTool in the following folder. It is recommended that a versioned software folder under the main CDSS folder be specified. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder: C:\CDSS\TSTool-08.15.03

Space required: 88.6MB

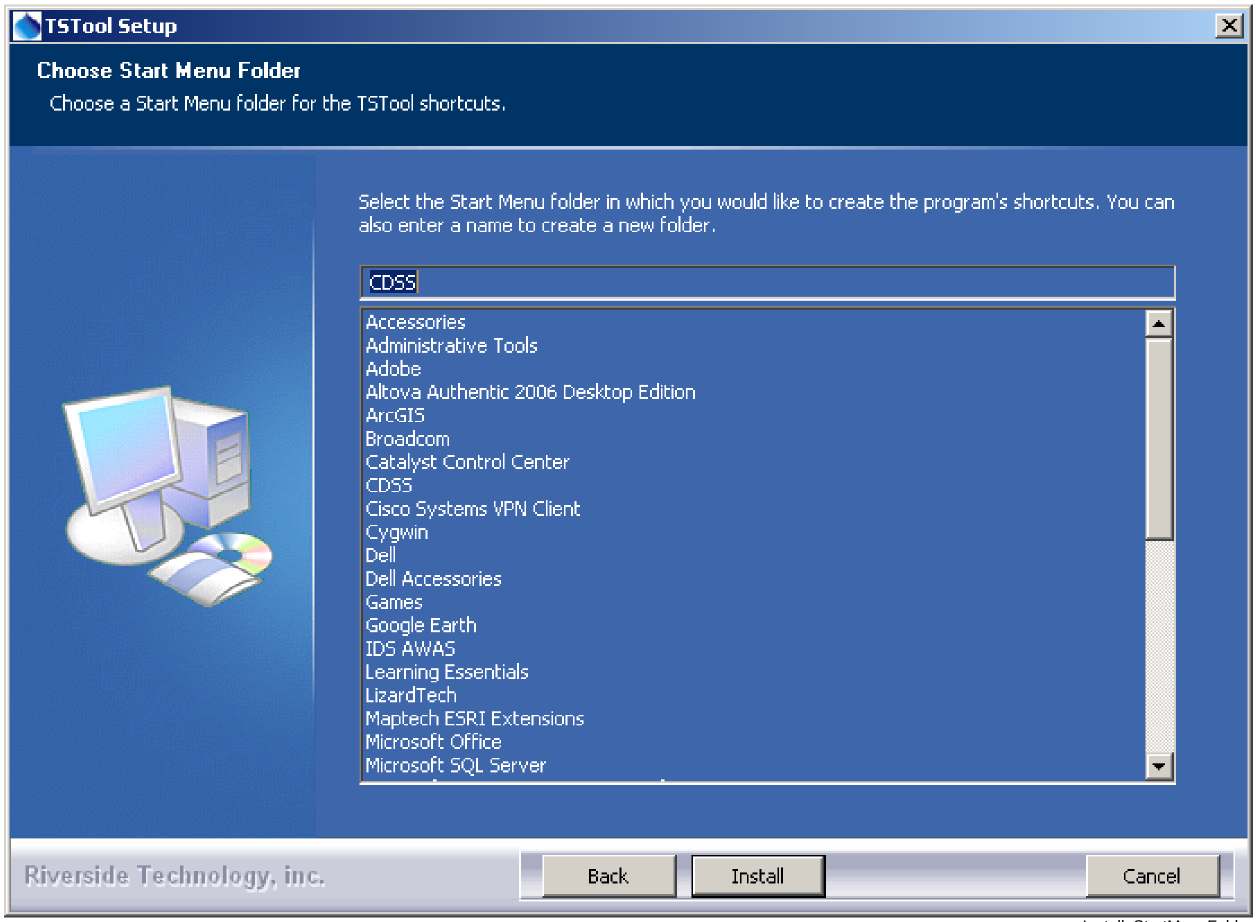
Space available: 18.4GB

Install_SelectFolder

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-Version\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:


TSTool Setup

Choose Start Menu Folder

Choose a Start Menu folder for the TSTool shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

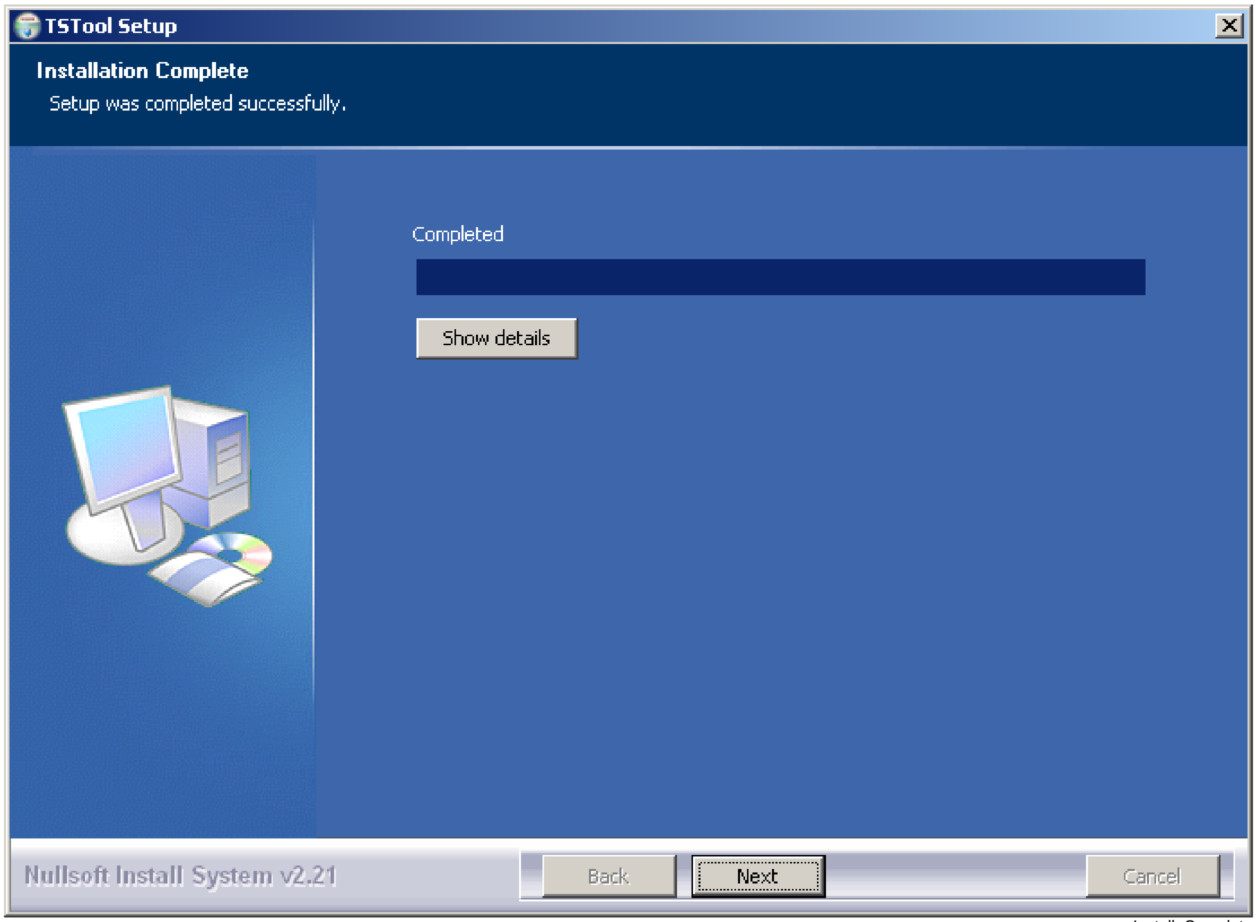
CDSS

Accessories
Administrative Tools
Adobe
Altova Authentic 2006 Desktop Edition
ArcGIS
Broadcom
Catalyst Control Center
CDSS
Cisco Systems VPN Client
Cygwin
Dell
Dell Accessories
Games
Google Earth
IDS AWAS
Learning Essentials
LizardTech
Maptech ESRI Extensions
Microsoft Office
Microsoft SQL Server

Install_StartMenuFolder

After selecting the folder, press ***Install***.

5. The following dialog will show the progress of the installation:



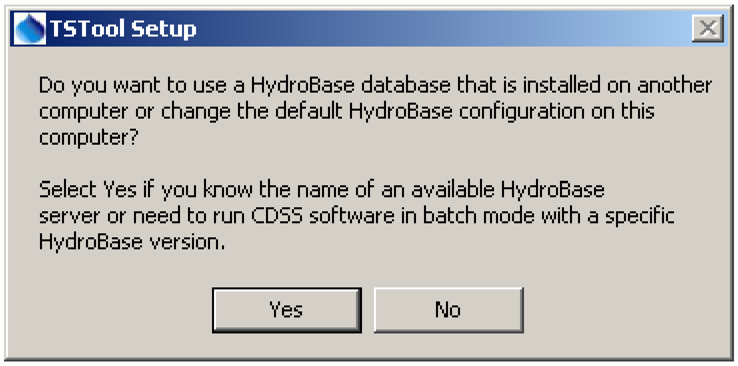
Installation Complete

Setup was completed successfully.

Install_Complete

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:



Do you want to use a HydroBase database that is installed on another computer or change the default HydroBase configuration on this computer?

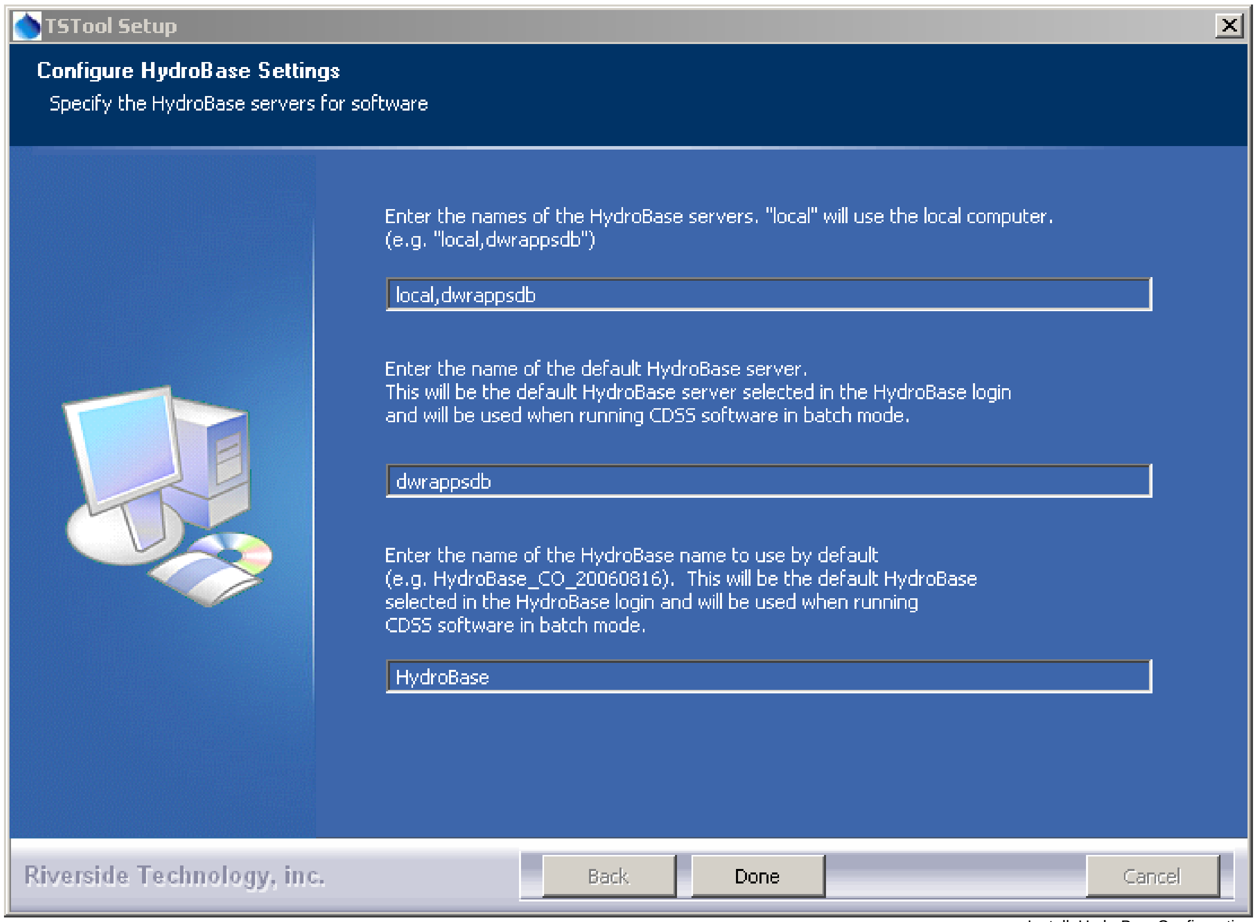
Select Yes if you know the name of an available HydroBase server or need to run CDSS software in batch mode with a specific HydroBase version.

Install_HydroBaseQuestion

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). Also press ***Yes*** if

TSTool will be run in batch mode because the specific HydroBase name must be specified in configuration files. 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). The dialog will initially show previous settings from the *\CDSS\TSTool-Version\system\CDSS.cfg* file and settings typically only need to be changed after installing a new HydroBase version.


TSTool Setup

Configure HydroBase Settings

Specify the HydroBase servers for software

Enter the names of the HydroBase servers. "local" will use the local computer. (e.g. "local,dwrappsdb")

local,dwrappsdb

Enter the name of the default HydroBase server. This will be the default HydroBase server selected in the HydroBase login and will be used when running CDSS software in batch mode.

dwrappsdb

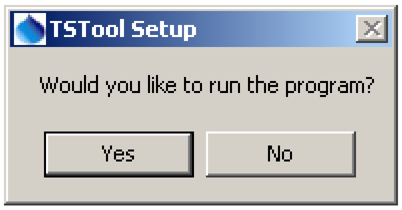
Enter the name of the HydroBase name to use by default (e.g. HydroBase_CO_20060816). This will be the default HydroBase selected in the HydroBase login and will be used when running CDSS software in batch mode.

HydroBase

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:


TSTool Setup

Would you like to run the program?

Install_RunTSToolQuestion

Press ***Yes*** to run the software or ***No*** to exit the installation procedure.

9. TSTool is distributed with a default configuration for CDSS. If you have edited the configuration properties, you can import the old configuration file using the ***Help…Import Configuration…*** menu'.

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 should typically 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):

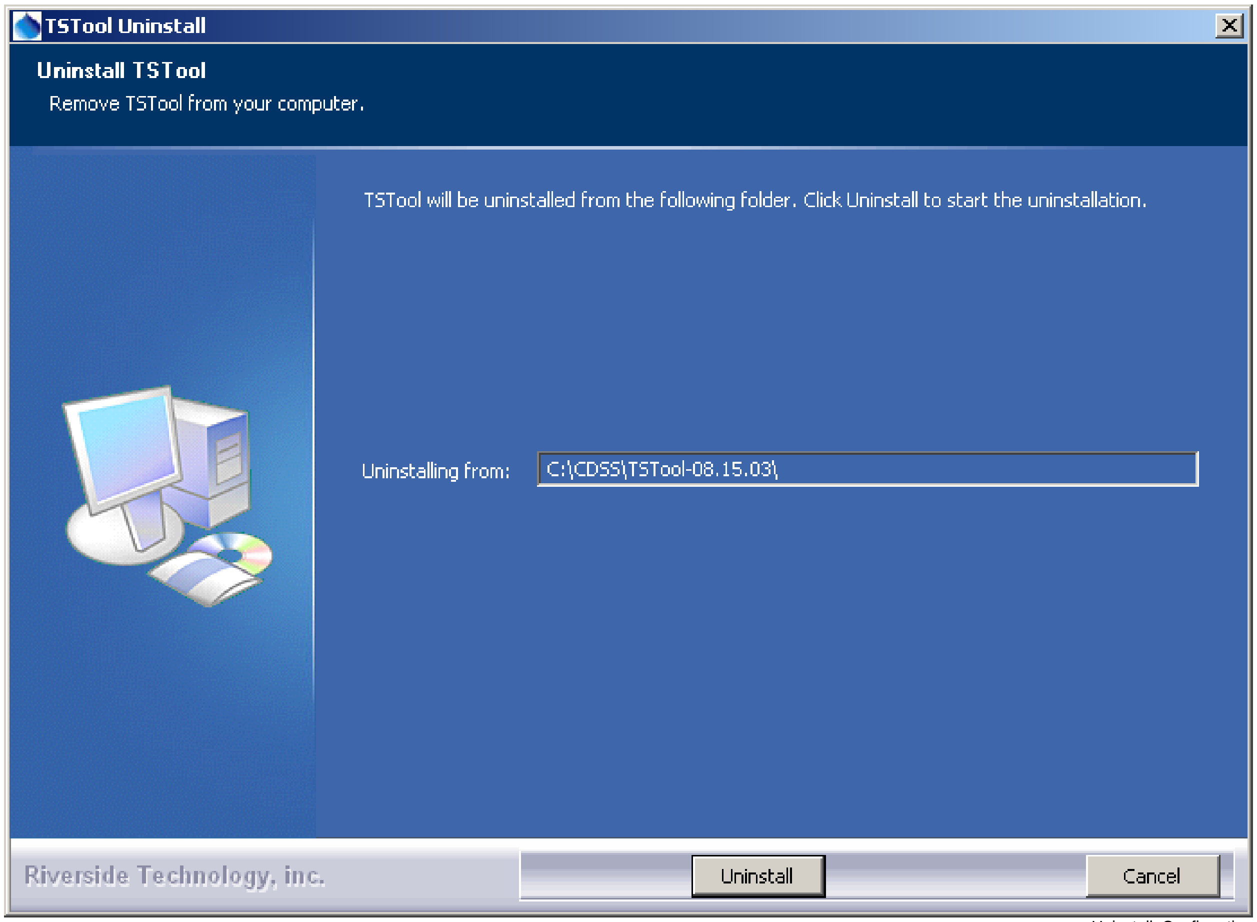
1. Run the *TSTool_CDSS_Version_Setup.exe* installer as described above. During installation specify the TSTool installation home using a drive letter and path for the server or specify a Universal Naming Convention (UNC) path (e.g., *\\ServerName\CDSS\TSTool-Version*).
2. Or….Copy the files from a local installation to a network location. The TSTool software will detect the file location when run using the *TSTool.exe* file. If the *TSTool.bat* file is used to run the software, it may need to be modified to specify the location of files on the server.

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.

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 *ServerName\CDSS\TSTool-Version\bin* folder. The software will expect that file locations use the same drives as when the software was installed.

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.



TSTool Uninstall

Uninstall TSTool

Remove TSTool from your computer.

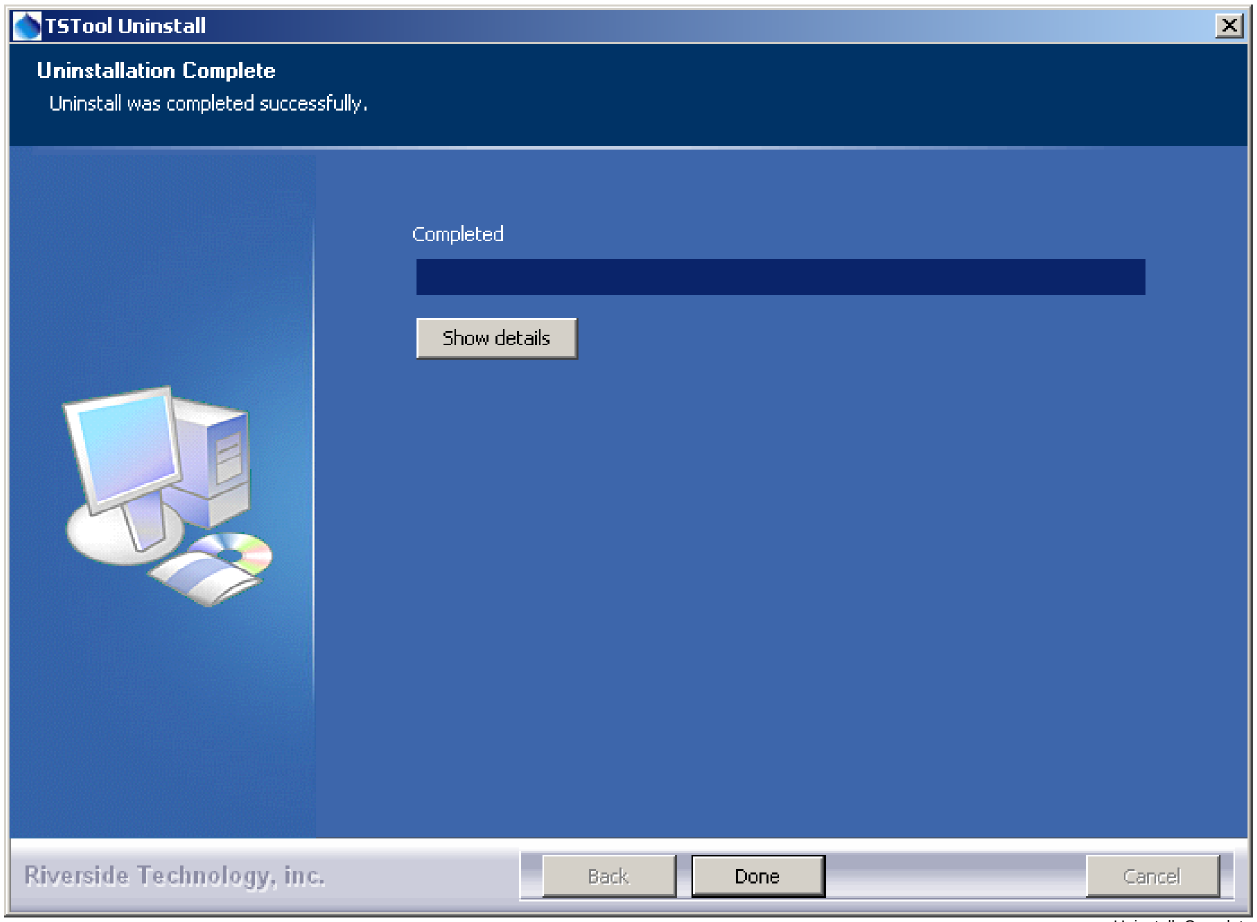
TSTool will be uninstalled from the following folder. Click Uninstall to start the uninstallation.

Uninstalling from: C:\CDSS\TSTool-08.15.03\

Uninstall_Confirmation

Press ***Uninstall*** to uninstall the software.

The following dialog shows the status of the uninstall.



Uninstall_Complete

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…CDSS…TSTool-Version*** (or ***Start… Programs… CDSS… TSTool-Version***) menu can be used to start the software. This runs the *TSToolInstallHome\bin\TSTool.exe* software.

5.2 Command Line Executable

The installation process does NOT add the *TSToolInstallHome\bin* folder to the path; however, this addition can be made by the user, allowing the TSTool software to be started anywhere by running *TSTool*. Running TSTool from any location will result in the software being run in the installation location. Specifying a command file on the command line or interactively will reset the working directory to that of the command file.

5.3 TSTool Batch File – Windows

A batch file can be used to run the *TSTool.exe* program, for example using the `–commands` command line parameter to specify a command file. In this case it may be necessary to specify the absolute path to the command file to ensure that the software can locate related files.

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 *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 developers 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. Use the ***Tools…Options*** menu for a dialog that helps with editing the `*Enabled` properties.

The optional `MapLayerLookupFile` property indicates the name of the time series to map layer lookup file. See the **Map Configuration** section below.

6.2 Data Units File

The *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 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
`AutoConnect`	If `False`, a HydroBase login dialog will be shown at startup. If `True`, the default database information in the CDSS configuration file (see next section) will be used to automatically connect to the database, and the login dialog will not be shown.	`False`
`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`

6.4 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 *\cdss\TSTool-Version\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_20080730"

[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, within the State of Colorado's offices.

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: TSView - Time Series Viewing Tools

2011-12-07, Original Maintained with TSTool

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.

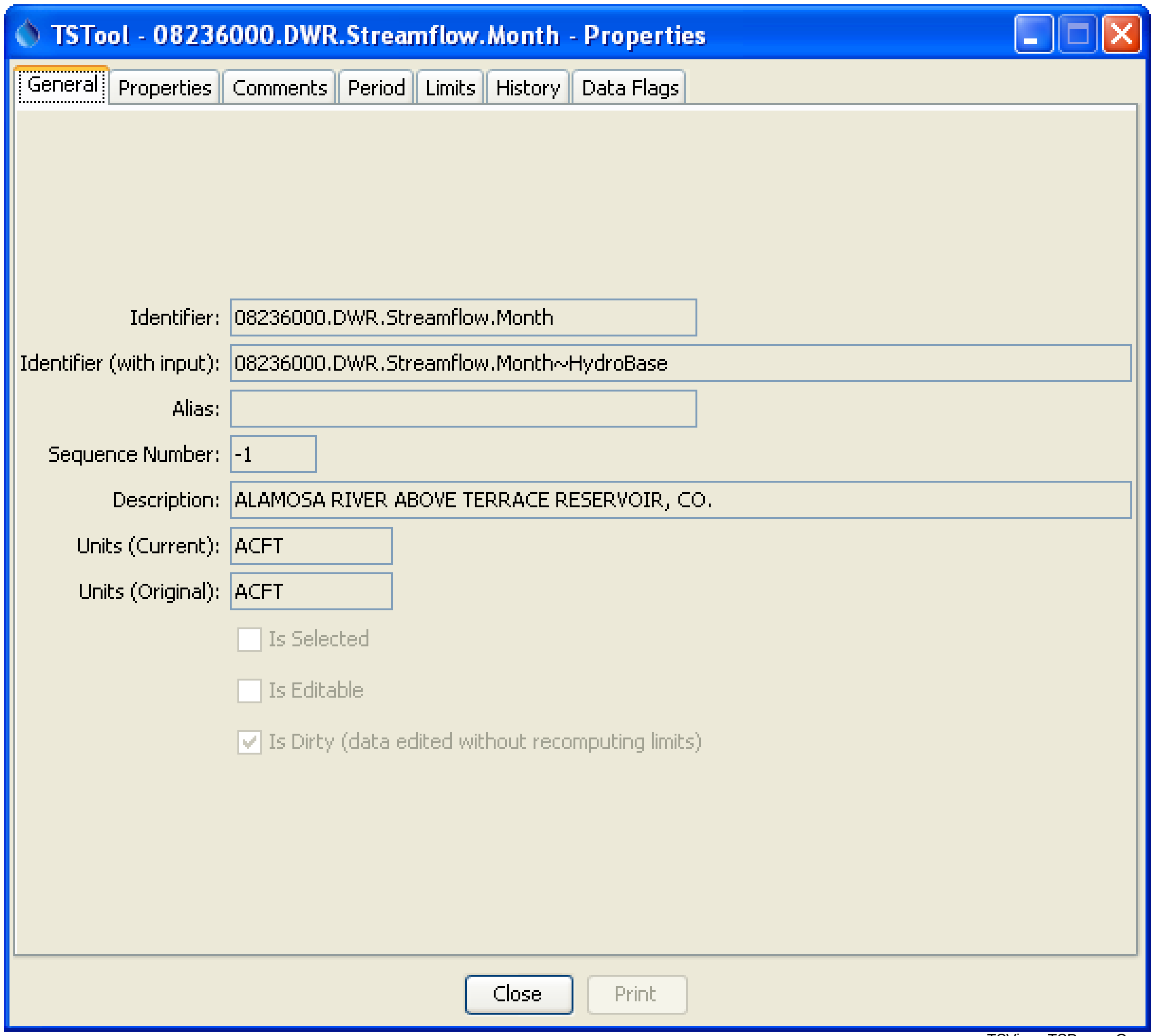
Time Series Terminology (listed alphabetically)

Term	Description
Data Interval	Time interval between time series data values. If a *regular* time series, the interval is constant. If an *irregular* 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.
Data Source	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).
Data Type	A string abbreviation for a data type, which is part of the time series identifier (e.g., `Streamflow`).
Date/Time Precision	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.
Input Name	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.
Input Type	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`).
Location	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).
Scenario	A string label that is part of a time series identifier, and serves as a modifier for the identifier (e.g., `HIST` for historical).
Sequence Number	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.
Time Series Product	A graph or report that can be defined and reproduced. See the **Time Series Product Reference** section.
Time Step	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



Identifier: 08236000.DWR.Streamflow.Month
Identifier (with input): 08236000.DWR.Streamflow.Month~HydroBase
Alias:
Sequence Number: -1
Description: ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO.
Units (Current): ACFT
Units (Original): ACFT
☐ Is Selected
☐ Is Editable
☑ Is Dirty (data edited without recomputing limits)

TSView_TSProps_General

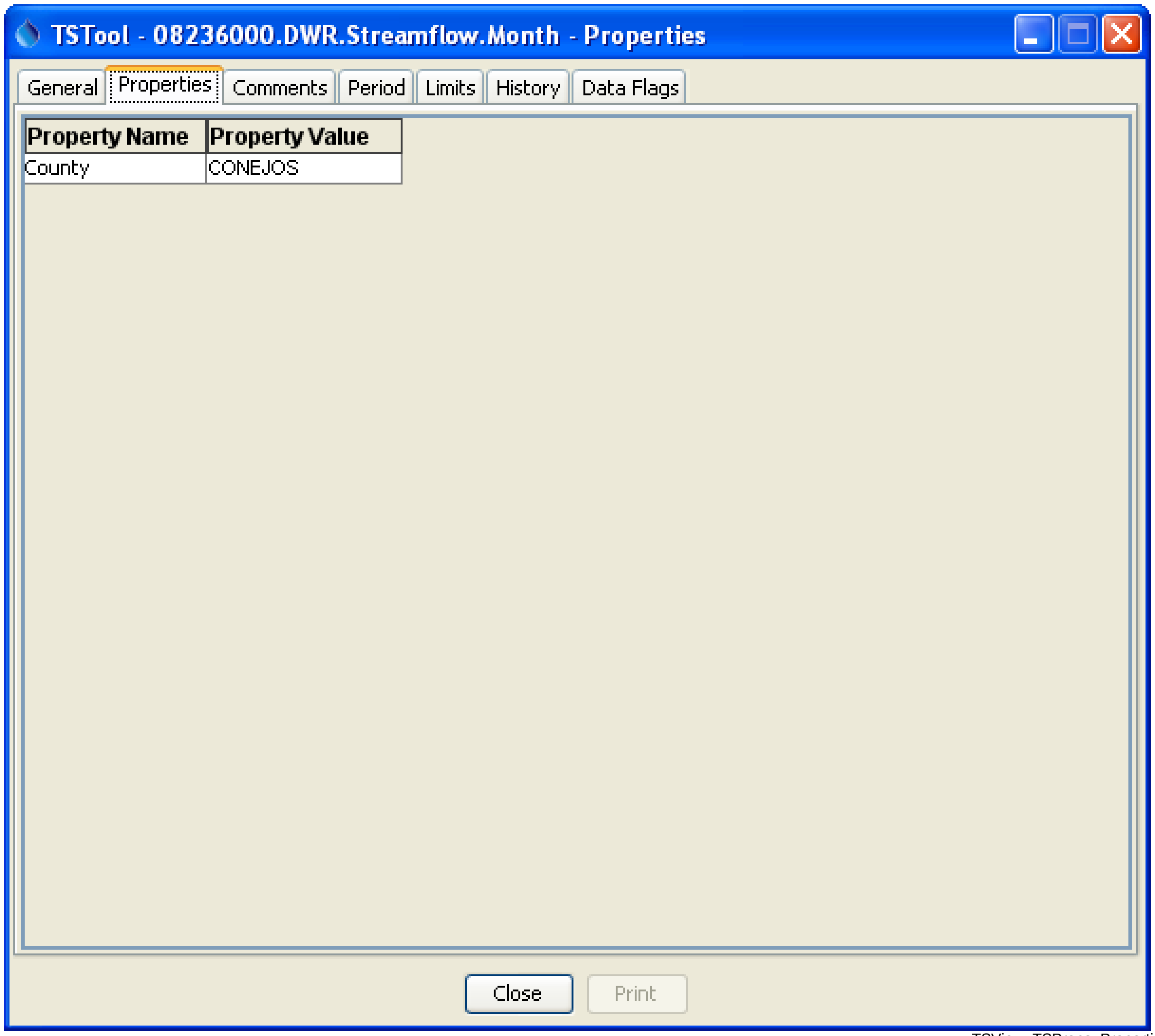
Time Series Properties - General

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 – (Dynamic) Properties



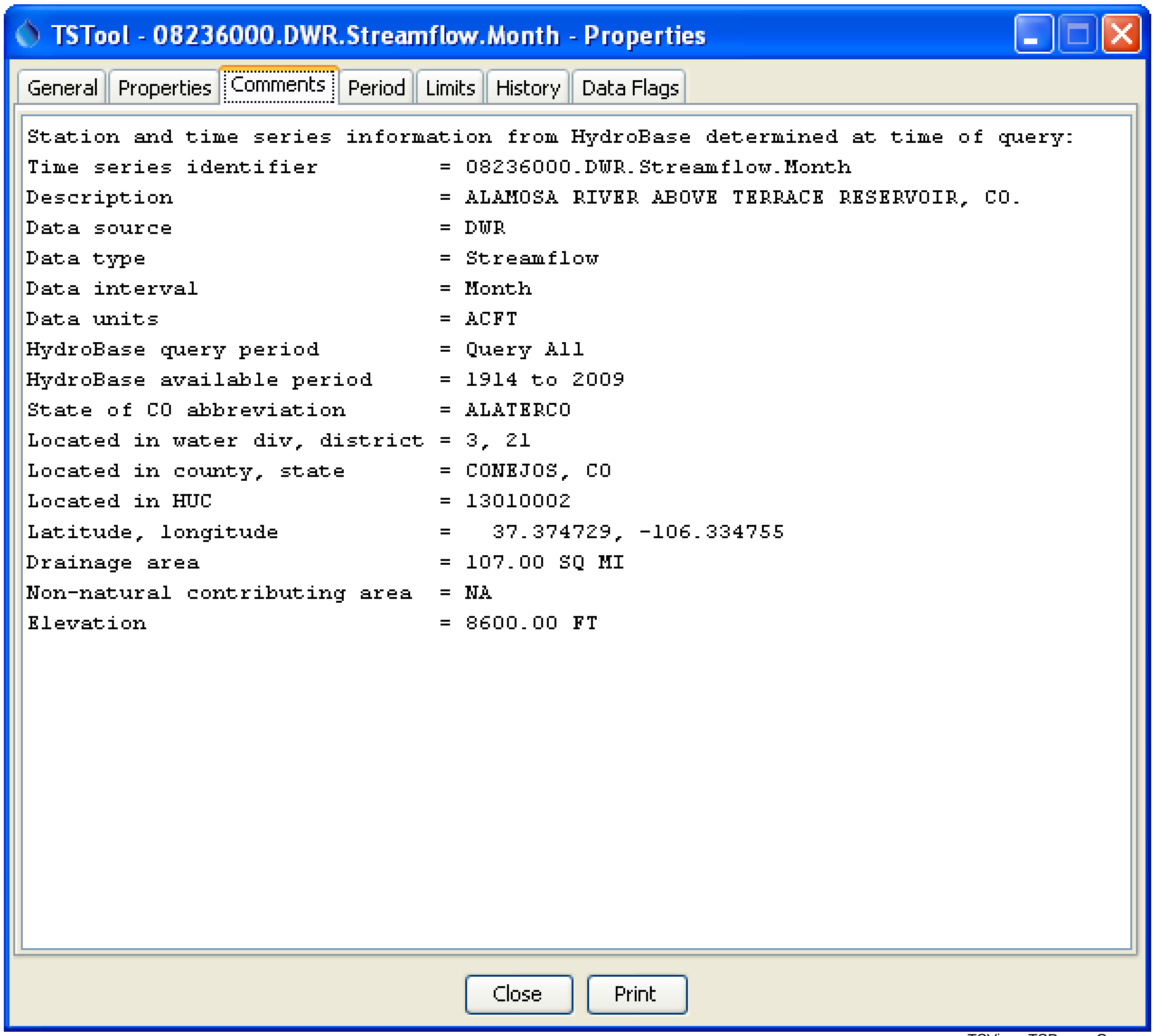
Property Name	Property Value
County	CONEJOS

TSView_TSProps_Properties

Time Series Properties – (Dynamic) Properties

Whereas the other properties tabs show standard time series properties, the ***Properties*** tab lists dynamic properties that are associated with a time series. For example, a TSTool command file may "tag" time series with properties based on quality control or other analysis. It also is possible to associate properties with time series when the data are read, for example to associate location and other properties queried from a database.

Time Series Properties – Comments



```
Station and time series information from HydroBase determined at time of query:
Time series identifier          = 08236000.DWR.Streamflow.Month
Description                     = ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO.
Data source                     = DWR
Data type                       = Streamflow
Data interval                   = Month
Data units                      = ACFT
HydroBase query period          = Query All
HydroBase available period      = 1914 to 2009
State of CO abbreviation        = ALATERCO
Located in water div, district = 3, 21
Located in county, state        = CONEJOS, CO
Located in HUC                  = 13010002
Latitude, longitude             =   37.374729, -106.334755
Drainage area                   = 107.00 SQ MI
Non-natural contributing area  = NA
Elevation                       = 8600.00 FT
```

TSView_TSProps_Comments

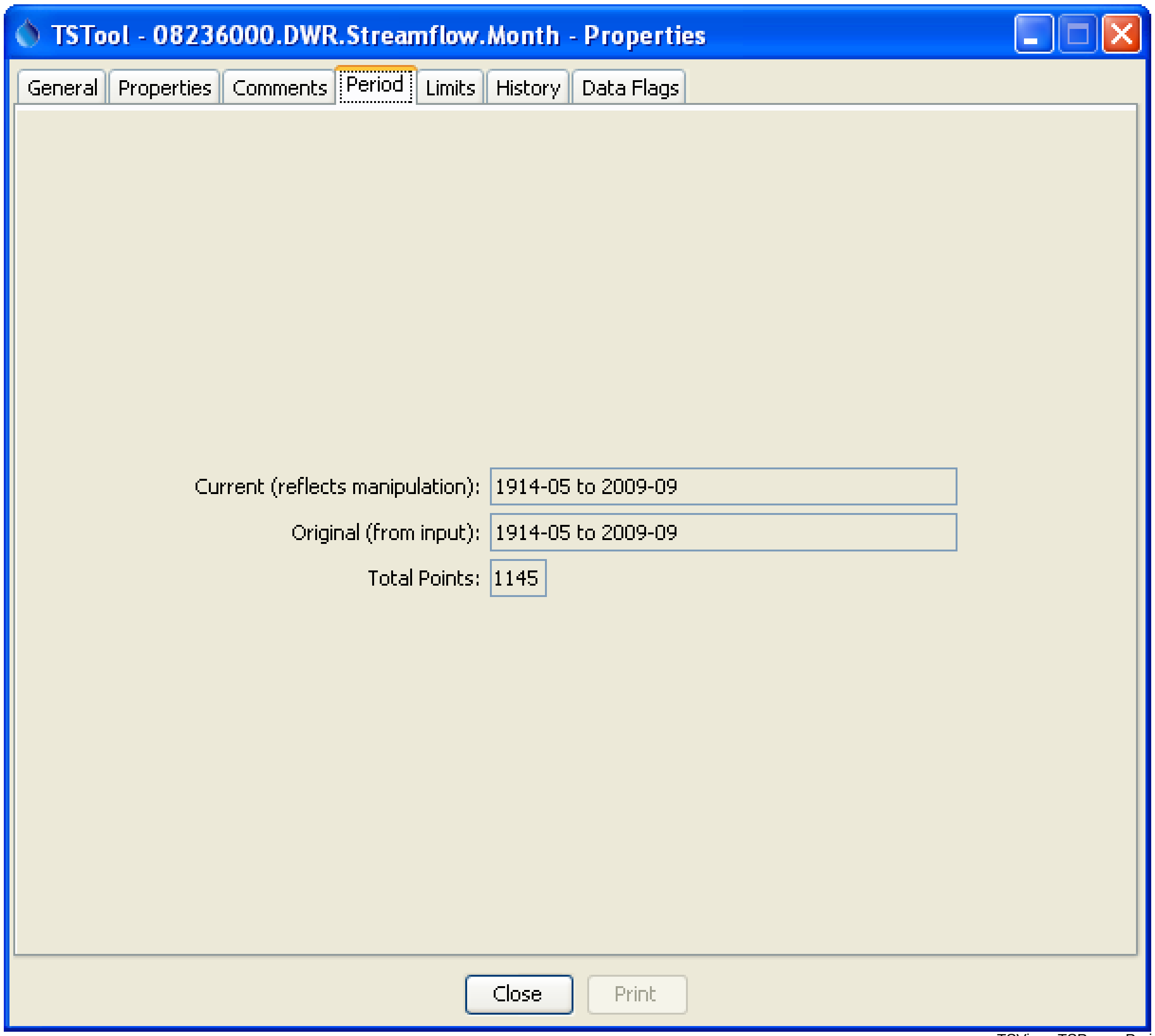
Time Series Properties - Comments

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).

In the future, the time series dynamic properties may be used more and text comments less.

Time Series Properties – Period



Current (reflects manipulation): 1914-05 to 2009-09
Original (from input): 1914-05 to 2009-09
Total Points: 1145

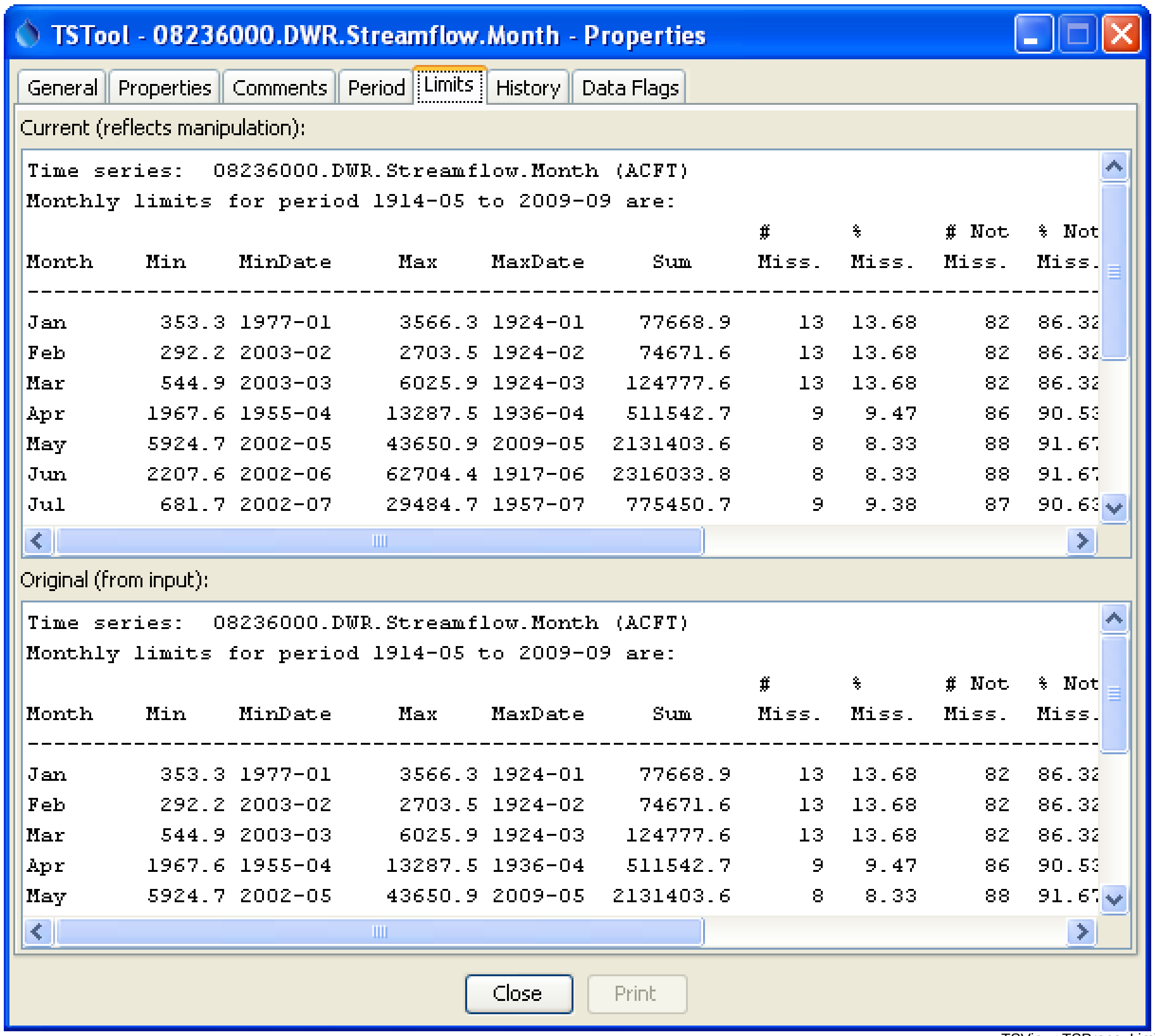
TSView_TSProps_Period

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 data points. If a regular time series, this is computed from the period. If an irregular time series, the number of points is the count of all data values. The data points may include missing data – see the data limits for additional information.

Time Series Properties – Limits



Current (reflects manipulation):

```
Time series:  08236000.DWR.Streamflow.Month (ACFT)
Monthly limits for period 1914-05 to 2009-09 are:
                                                       #      %      # Not  % Not
Month    Min    MinDate     Max    MaxDate      Sum     Miss.  Miss.  Miss.  Miss.
---------------------------------------------------------------------------------
Jan       353.3 1977-01     3566.3 1924-01    77668.9     13  13.68     82  86.32
Feb       292.2 2003-02     2703.5 1924-02    74671.6     13  13.68     82  86.32
Mar       544.9 2003-03     6025.9 1924-03   124777.6     13  13.68     82  86.32
Apr      1967.6 1955-04    13287.5 1936-04   511542.7      9   9.47     86  90.53
May      5924.7 2002-05    43650.9 2009-05  2131403.6      8   8.33     88  91.67
Jun      2207.6 2002-06    62704.4 1917-06  2316033.8      8   8.33     88  91.67
Jul       681.7 2002-07    29484.7 1957-07   775450.7      9   9.38     87  90.63
```

Original (from input):

```
Time series:  08236000.DWR.Streamflow.Month (ACFT)
Monthly limits for period 1914-05 to 2009-09 are:
                                                       #      %      # Not  % Not
Month    Min    MinDate     Max    MaxDate      Sum     Miss.  Miss.  Miss.  Miss.
---------------------------------------------------------------------------------
Jan       353.3 1977-01     3566.3 1924-01    77668.9     13  13.68     82  86.32
Feb       292.2 2003-02     2703.5 1924-02    74671.6     13  13.68     82  86.32
Mar       544.9 2003-03     6025.9 1924-03   124777.6     13  13.68     82  86.32
Apr      1967.6 1955-04    13287.5 1936-04   511542.7      9   9.47     86  90.53
May      5924.7 2002-05    43650.9 2009-05  2131403.6      8   8.33     88  91.67
```

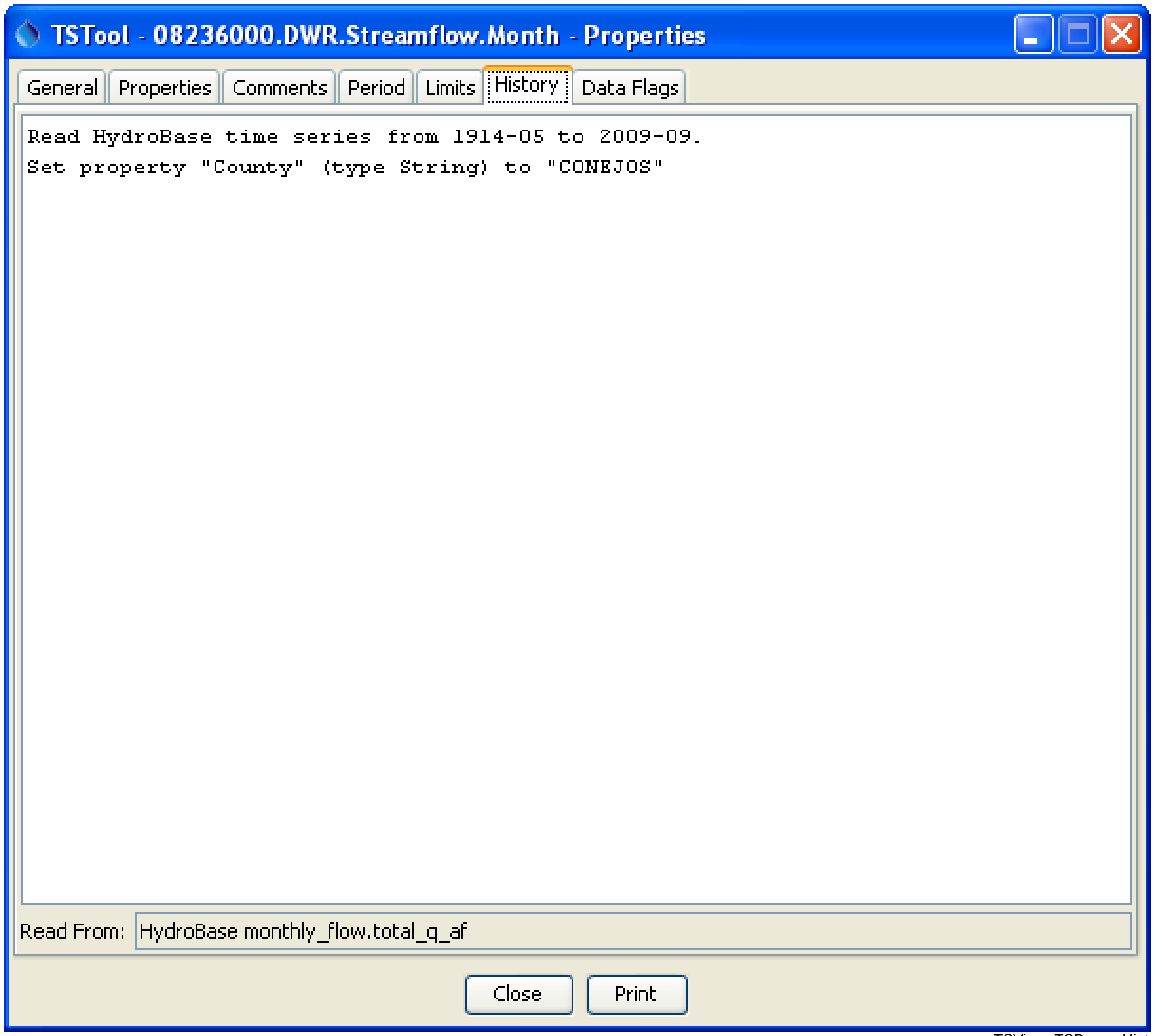
TSView_TSProps_Limits

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



```
Read HydroBase time series from 1914-05 to 2009-09.
Set property "County" (type String) to "CONEJOS"
```

Read From: HydroBase monthly_flow.total_q_af

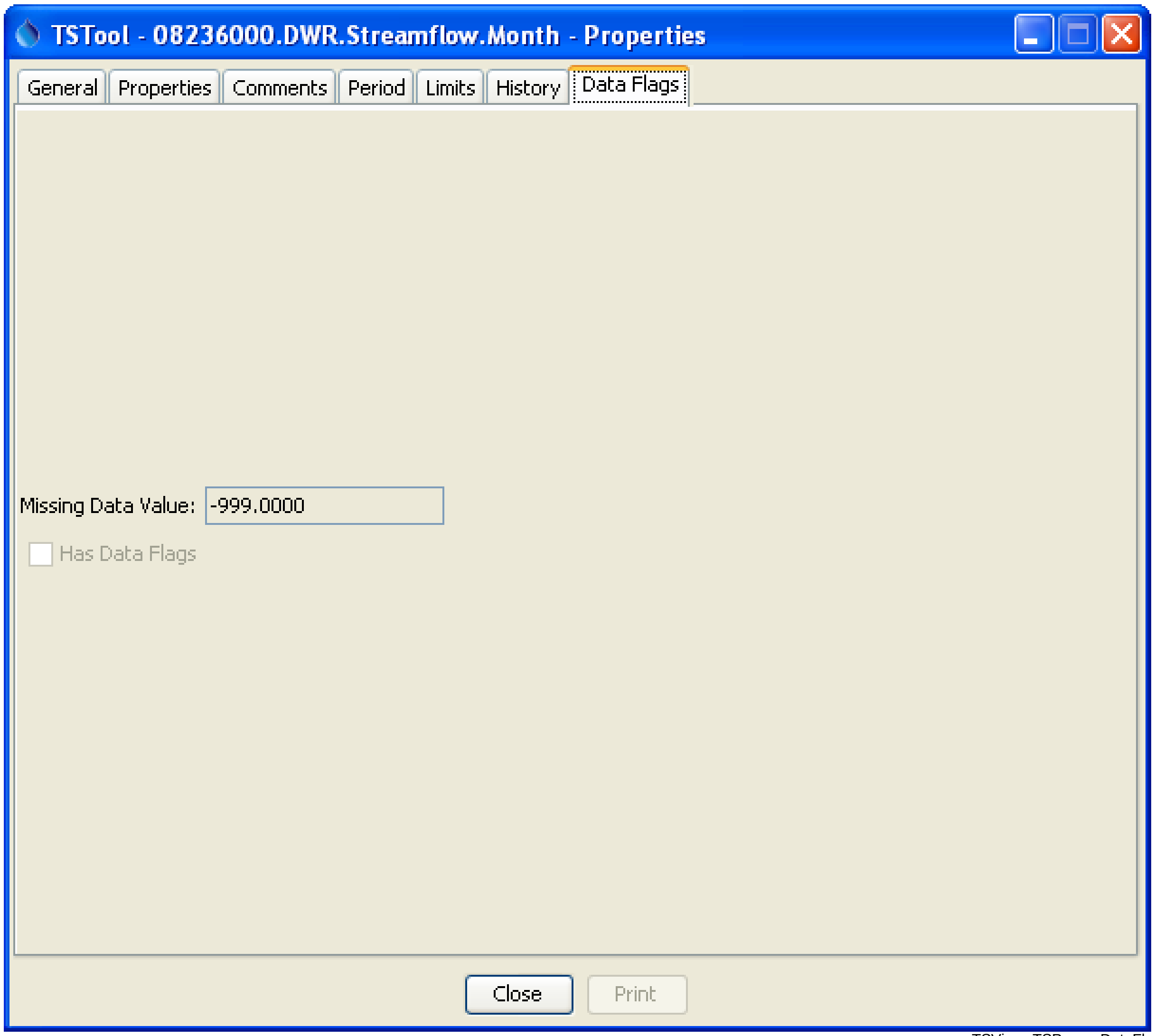
TSView_TSProps_History

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



Missing Data Value: -999.0000

Has Data Flags

TSView_TSProps_DataFlags

Time Series Properties - Data Flags

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).

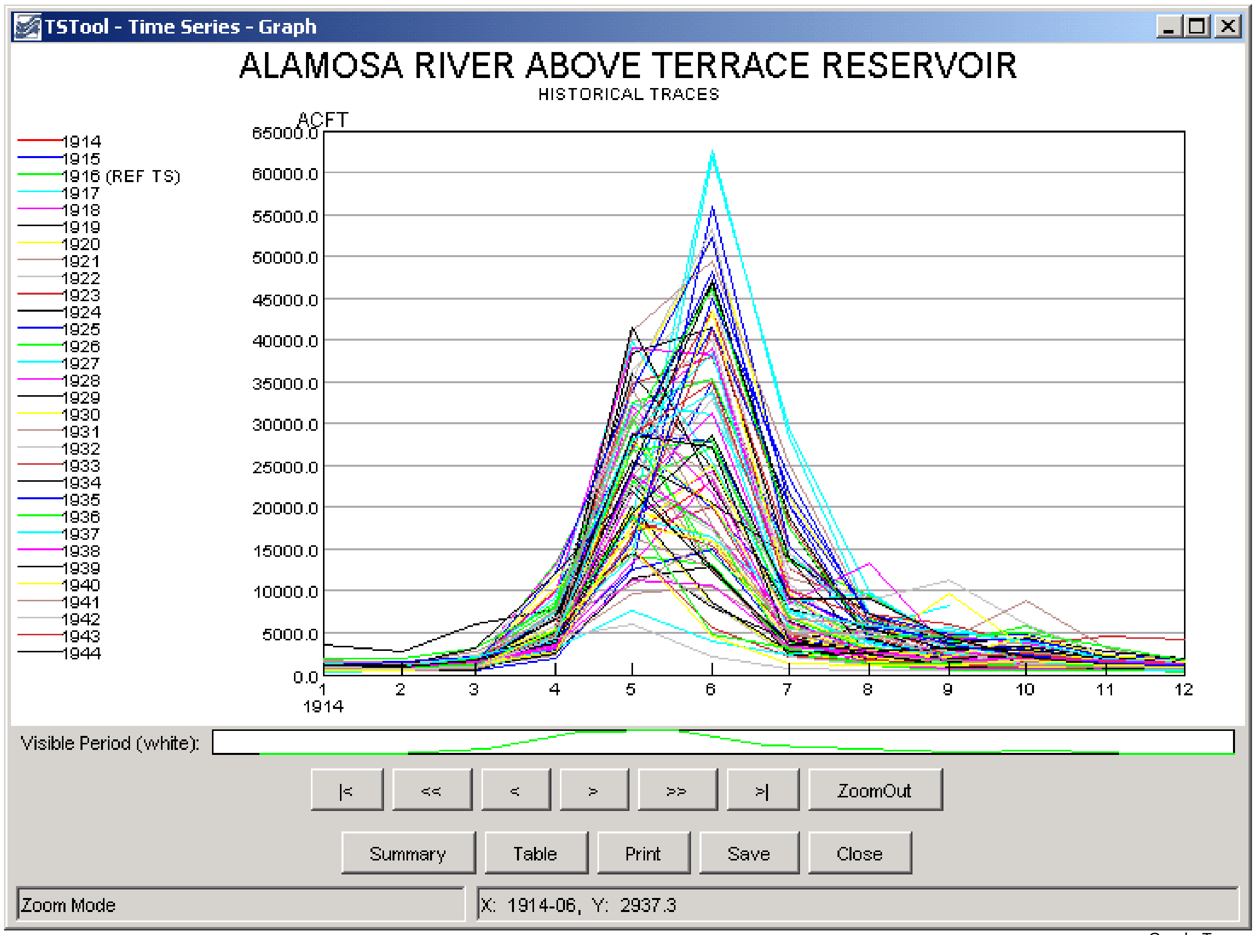
One of the issues with fully supporting data flags is that different input types (and even different data within an input type) treat data flags inconsistently. Therefore, it is easier to add data flags to time series visualization tools (e.g., label points on a graph with the flag) than to integrate data flags in data filling and analysis features. Features related to data flags will continue to be enhanced.

Time Series Traces

In general, the term *time series traces* 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.



Graph_Traces

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 use simple time series, most features also are 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

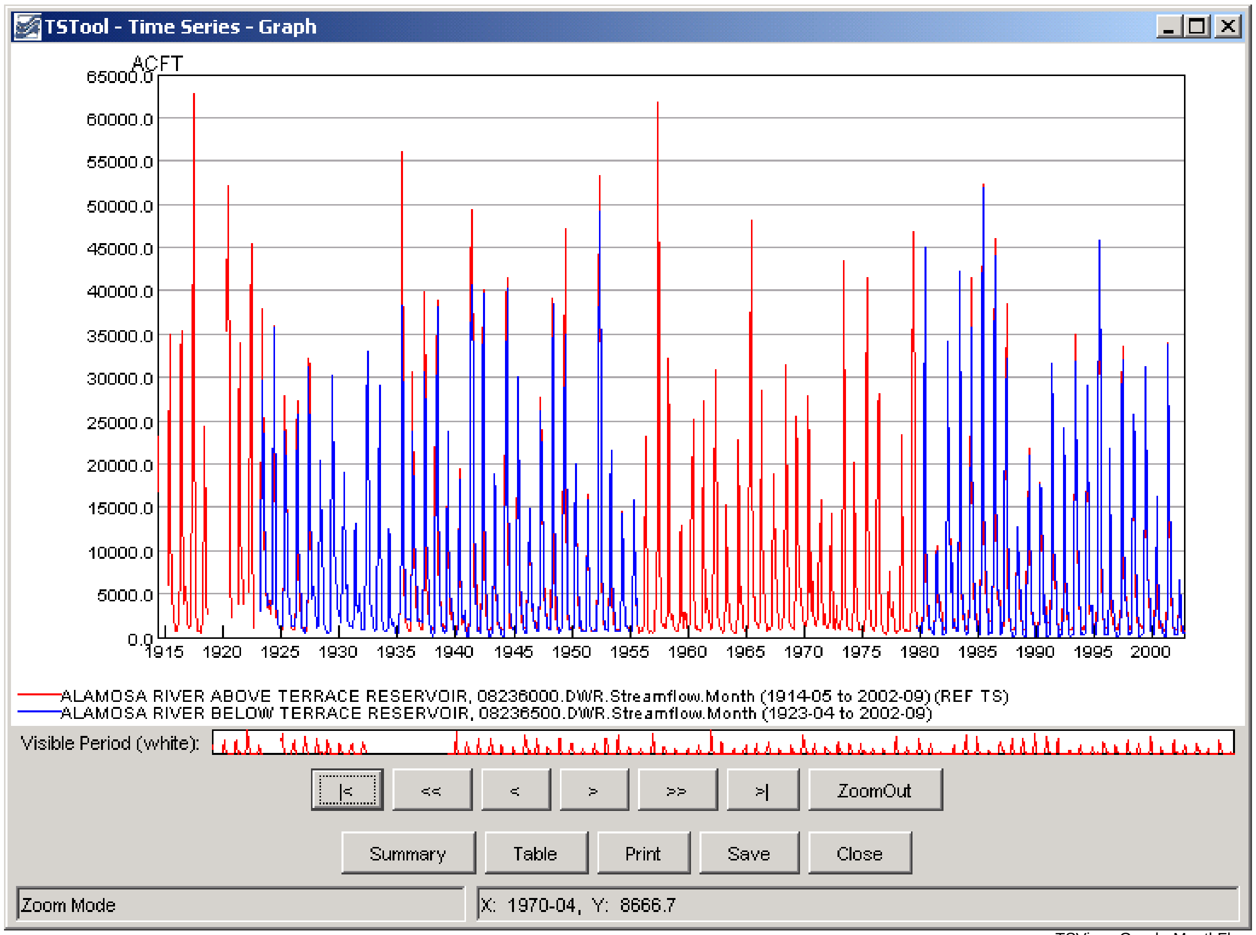
The initial view for a time series list is typically determined from the actions of the software user. For example, a ***Graph*** button may be displayed on a screen, which when pressed will result in a graph being displayed. The time series that are displayed in the view can typically contain one or 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.



ALAMOSA RIVER ABOVE TERRACE RESERVOIR, 08236000.DWR.Streamflow.Month (1914-05 to 2002-09) (REF TS)
ALAMOSA RIVER BELOW TERRACE RESERVOIR, 08236500.DWR.Streamflow.Month (1923-04 to 2002-09)

Zoom Mode

X: 1970-04, Y: 8666.7

TSView_Graph_MonthFlow

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 *Time Series Product* 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 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.

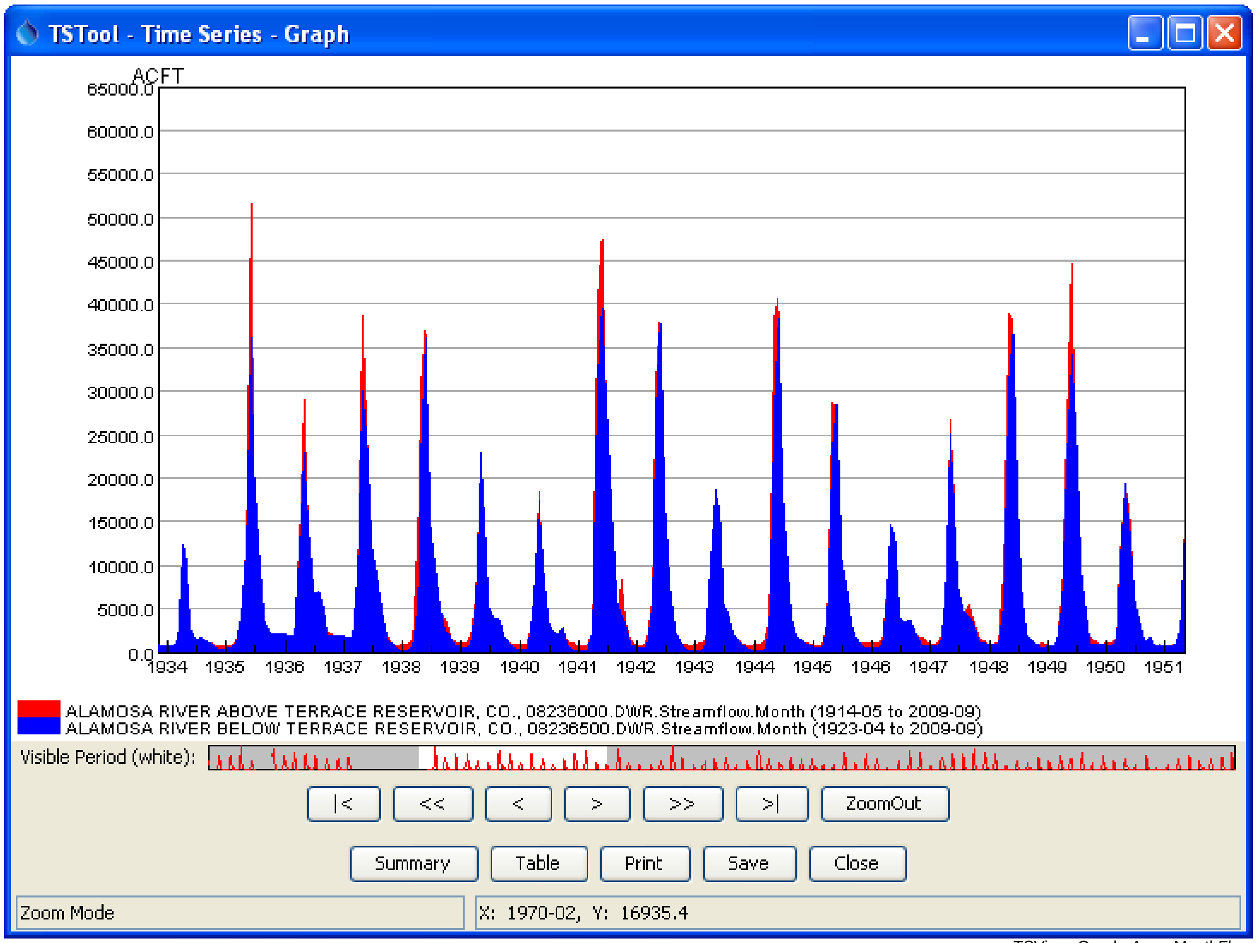
9. Graph types can be changed after the initial display, with limitations. Graphs can be switched between simple types like line and bar graphs; however, simple graphs cannot be changed to more complex types.

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.

Area Graph

An area graph draws each time series by filling in the area under the graph with a solid color. The time series are drawn in the order that they are specified and therefore this graph is most suitable for cases where the first time series has values that are larger than subsequent time series.

An alternative to the area graph is a bar graph where `BarOverlap=True` (by default multiple time series in a bar graph shows bars that are parallel at a point in time).



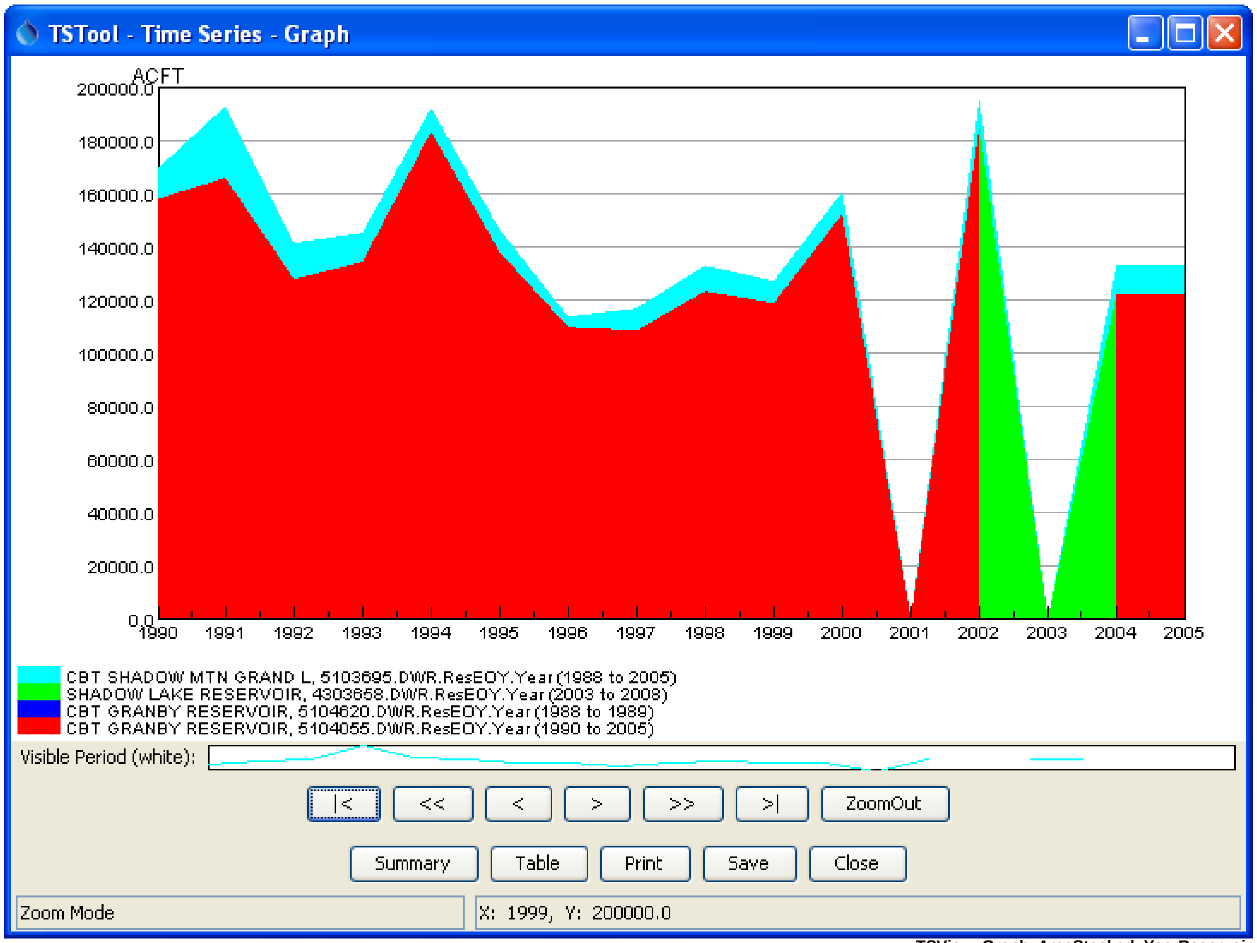
🟥 ALAMOSA RIVER ABOVE TERRACE RESERVOIR, CO., 08236000.DWR.Streamflow.Month (1914-05 to 2009-09)
🟦 ALAMOSA RIVER BELOW TERRACE RESERVOIR, CO., 08236500.DWR.Streamflow.Month (1923-04 to 2009-09)

Zoom Mode | X: 1970-02, Y: 16935.4

TSView_Graph_Area_MonthFlow

Stacked Area Graph

A stacked area graph is similar to the simple area graph in that it draws each time series by filling in the area under the graph with a solid color. However, instead of drawing the original data values, the cumulative values of the time series are used for the plotting coordinates. The time series are drawn in the order that they are specified (first on bottom). The legend order agrees with the representation of the graph.



🟦 CBT SHADOW MTN GRAND L, 5103695.DWR.ResEOY.Year (1988 to 2005)
🟩 SHADOW LAKE RESERVOIR, 4303658.DWR.ResEOY.Year (2003 to 2008)
🟦 CBT GRANBY RESERVOIR, 5104620.DWR.ResEOY.Year (1988 to 1989)
🟥 CBT GRANBY RESERVOIR, 5104055.DWR.ResEOY.Year (1990 to 2005)

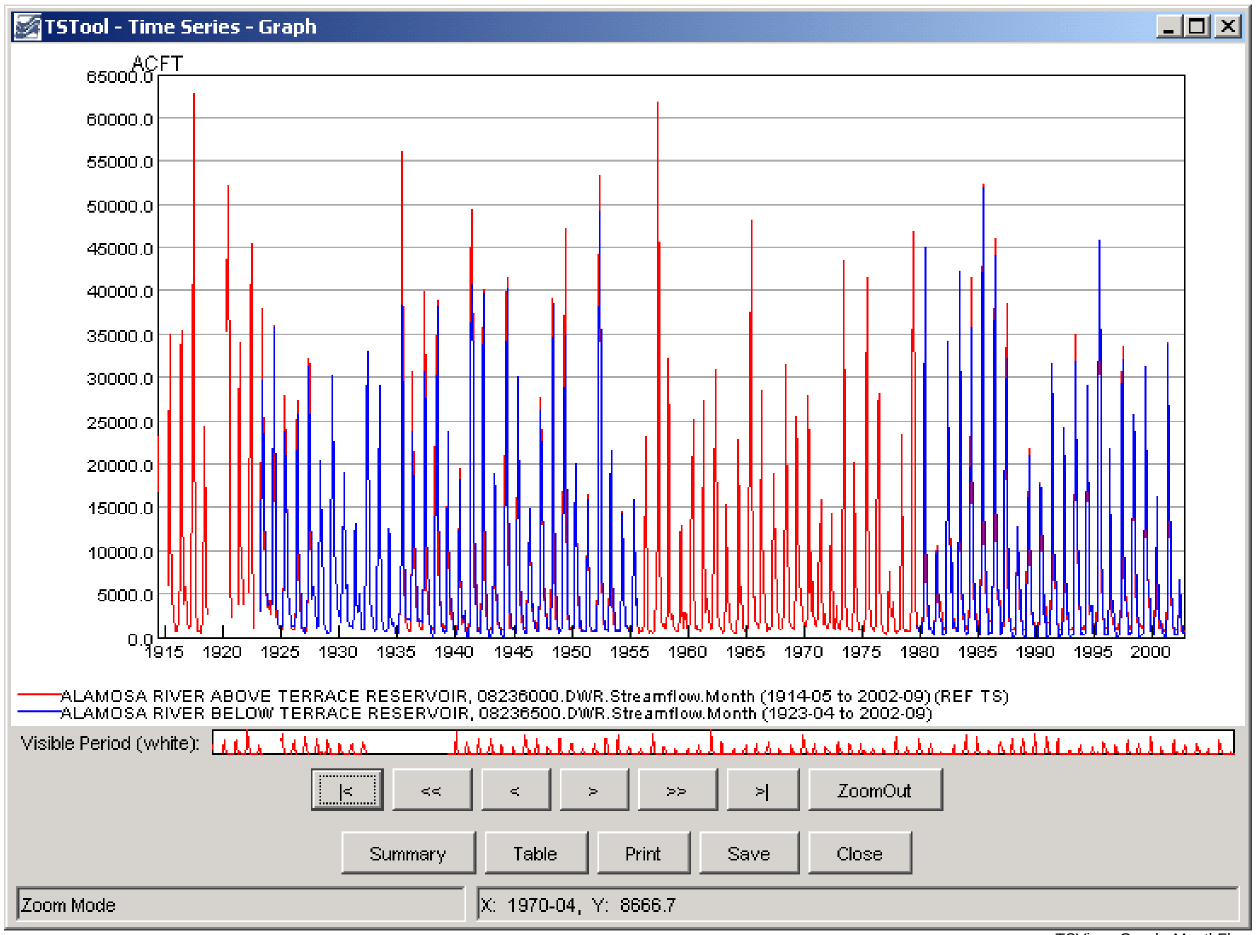
Zoom Mode

X: 1999, Y: 200000.0

TSView_Graph_AreaStacked_YearReservoir

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).



ALAMOSA RIVER ABOVE TERRACE RESERVOIR, 08236000.DWR.Streamflow.Month (1914-05 to 2002-09) (REF TS)
ALAMOSA RIVER BELOW TERRACE RESERVOIR, 08236500.DWR.Streamflow.Month (1923-04 to 2002-09)

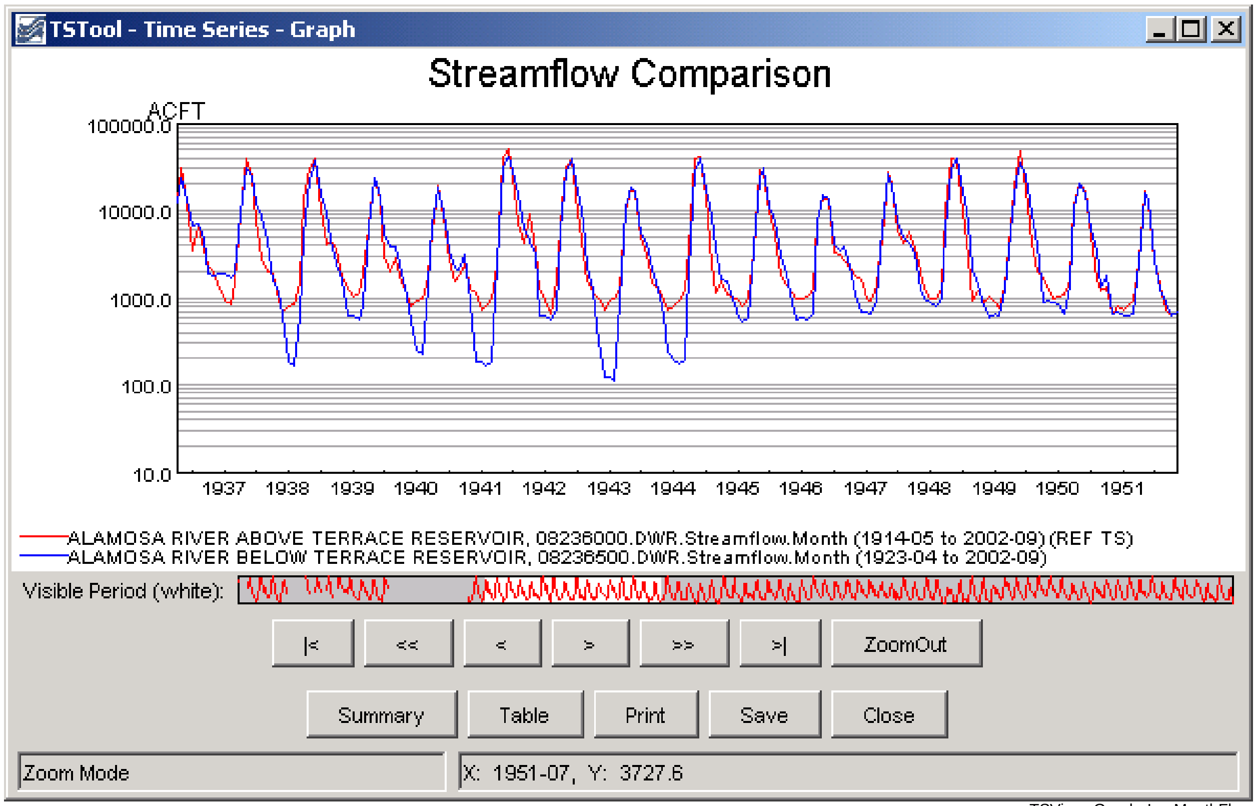
Zoom Mode

X: 1970-04, Y: 8666.7

TSView_Graph_MonthFlow

Line Graph - Log Y Axis

Log-axis line graphs are similar to simple line graphs. The following figure illustrates a typical graph.



ALAMOSA RIVER ABOVE TERRACE RESERVOIR, 08236000.DWR.Streamflow.Month (1914-05 to 2002-09) (REF TS)
ALAMOSA RIVER BELOW TERRACE RESERVOIR, 08236500.DWR.Streamflow.Month (1923-04 to 2002-09)

Zoom Mode

X: 1951-07, Y: 3727.6

TSView_Graph_LogMonthFlow

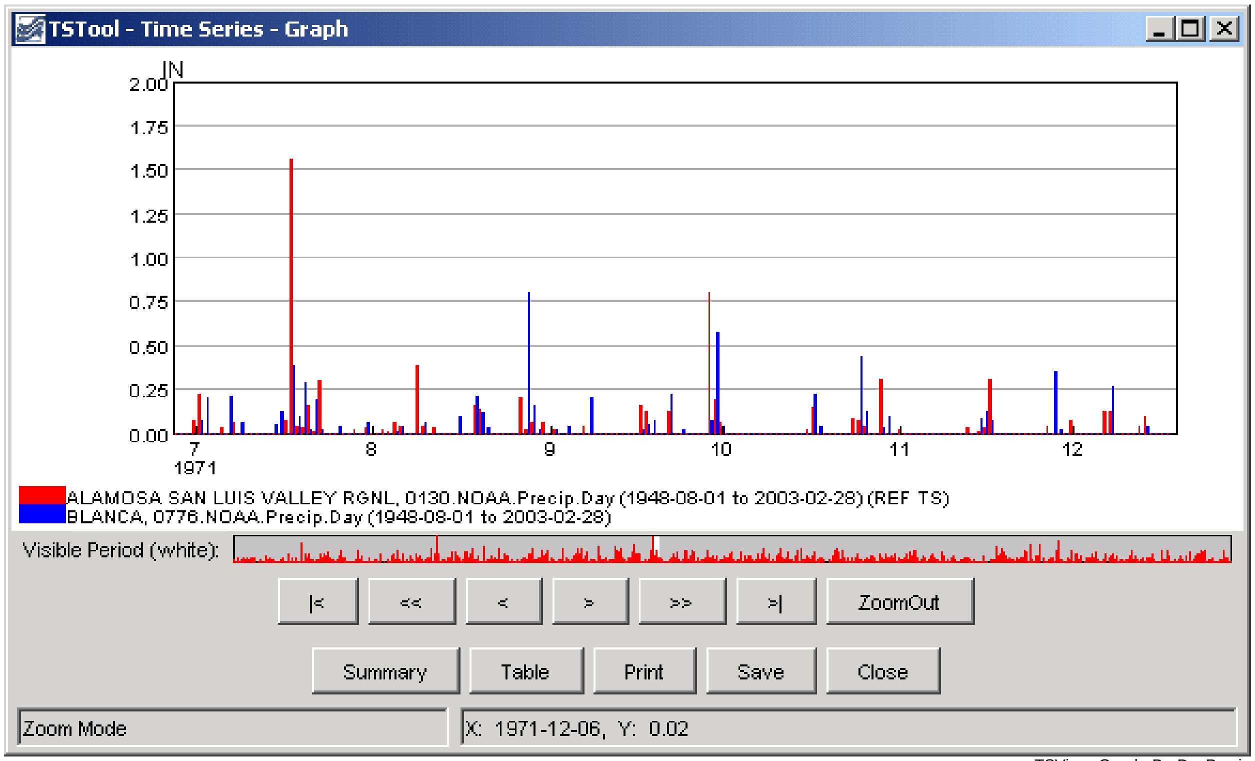
Example Log Y Axis Graph showing Monthly Streamflow

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 by default produces a graph with parallel vertical bars, as shown in the following example:



🟥 ALAMOSA SAN LUIS VALLEY RGNL, 0130.NOAA.Precip.Day (1948-08-01 to 2003-02-28) (REF TS)
🟦 BLANCA, 0776.NOAA.Precip.Day (1948-08-01 to 2003-02-28)

Zoom Mode

X: 1971-12-06, Y: 0.02

TSView_Graph_BarDayPrecip

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:

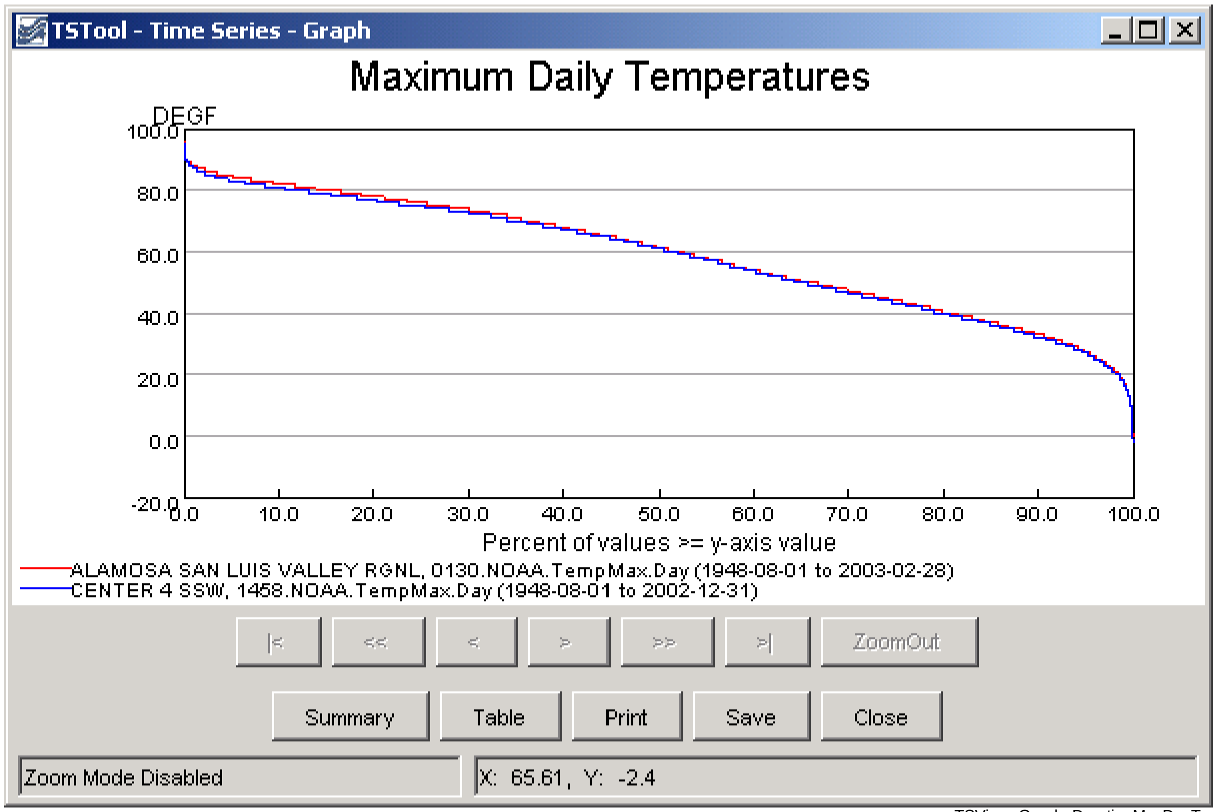
- Bars can be plotted centered on, to the left of, or to the right of the dates. If multiple time series are plotted, the overall total width of the bars will correspond to one data interval. If drawn to the left of the date, the bars for all graphed time series are drawn to the left of the date. If 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:


TSTool - Time Series - Graph
Maximum Daily Temperatures
DEGF
100.0
80.0
60.0
40.0
20.0
0.0
-20.0
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
Percent of values >= y-axis value
ALAMOSA SAN LUIS VALLEY RGNL, 0130.NOAA.TempMax.Day (1948-08-01 to 2003-02-28)
CENTER 4 SSW, 1458.NOAA.TempMax.Day (1948-08-01 to 2002-12-31)
|<
<<
<
>
>>
>|
ZoomOut
Summary
Table
Print
Save
Close
Zoom Mode Disabled
X: 65.61, Y: -2.4

TSView_Graph_DurationMaxDayTemp

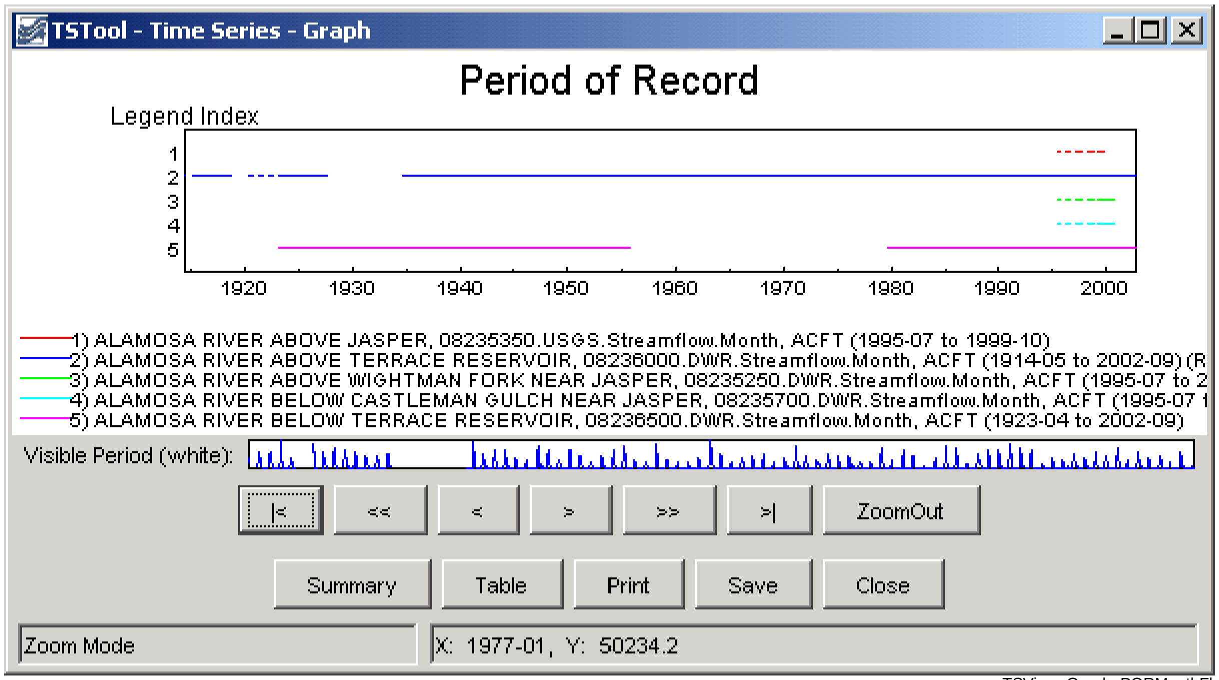
Example Duration Graph showing Maximum Monthly Temperatures

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:



1) ALAMOSA RIVER ABOVE JASPER, 08235350.USGS.Streamflow.Month, ACFT (1995-07 to 1999-10)
2) ALAMOSA RIVER ABOVE TERRACE RESERVOIR, 08236000.DWR.Streamflow.Month, ACFT (1914-05 to 2002-09) (R
3) ALAMOSA RIVER ABOVE WIGHTMAN FORK NEAR JASPER, 08235250.DWR.Streamflow.Month, ACFT (1995-07 to 2
4) ALAMOSA RIVER BELOW CASTLEMAN GULCH NEAR JASPER, 08235700.DWR.Streamflow.Month, ACFT (1995-07 t
5) ALAMOSA RIVER BELOW TERRACE RESERVOIR, 08236500.DWR.Streamflow.Month, ACFT (1923-04 to 2002-09)

Zoom Mode

X: 1977-01, Y: 50234.2

TSView_Graph_PORMonthFlow

Example Period of Record Graph showing Monthly Streamflow

Characteristics of the period of record graph type are:

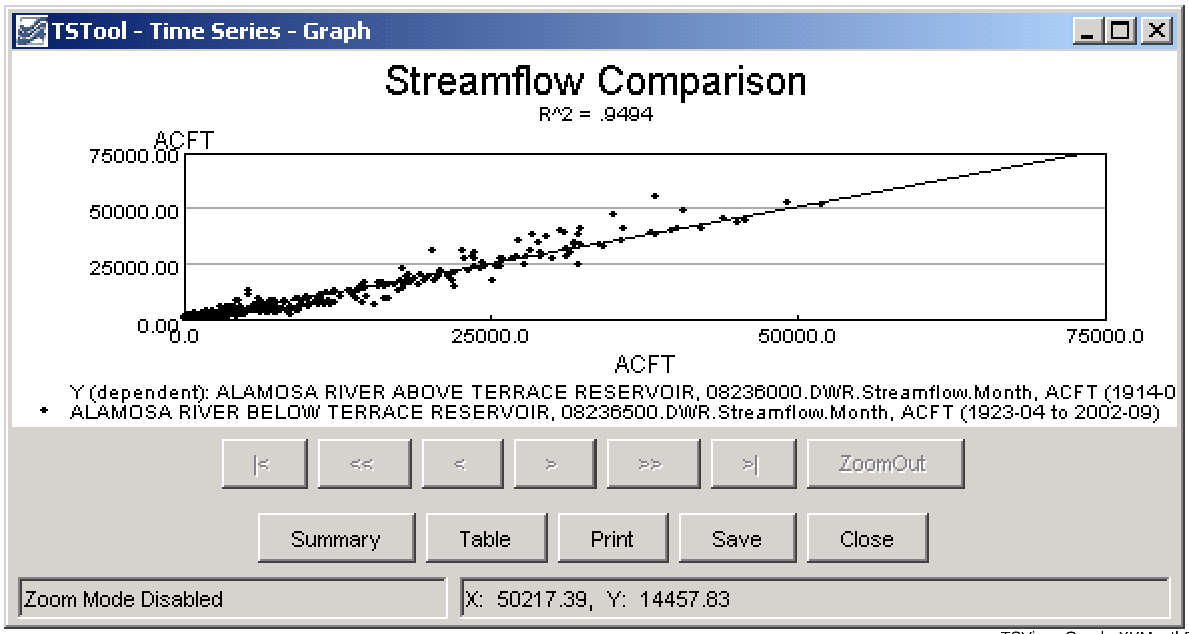
- Horizontal lines are drawn for each time series, with breaks in the line indicating missing data.
- Zooming is fully enabled, however, it may be difficult to see small breaks in the lines – it may be necessary to display symbols at the data points. The data limits properties of each time series can 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 closed to actual observations.
3. The independent (X) and dependent (Y) time series are compared to determine whether any time 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.



Y (dependent): ALAMOSA RIVER ABOVE TERRACE RESERVOIR, 08236000.DWR.Streamflow.Month, ACFT (1914-0
• ALAMOSA RIVER BELOW TERRACE RESERVOIR, 08236500.DWR.Streamflow.Month, ACFT (1923-04 to 2002-09)

Zoom Mode Disabled

X: 50217.39, Y: 14457.83

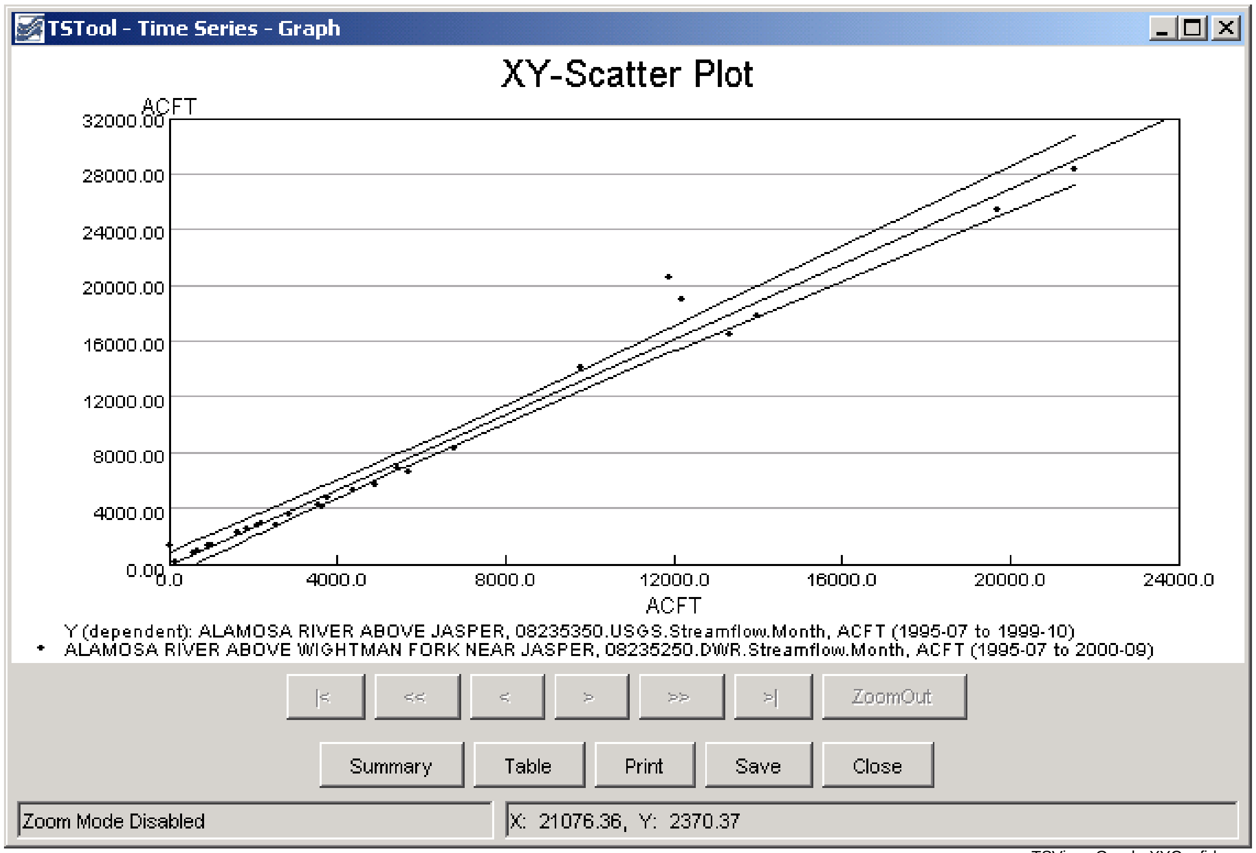
TSView_Graph_XYMonthFlow

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:



Y (dependent): ALAMOSA RIVER ABOVE JASPER, 08235350.USGS.Streamflow.Month, ACFT (1995-07 to 1999-10)
• ALAMOSA RIVER ABOVE WIGHTMAN FORK NEAR JASPER, 08235250.DWR.Streamflow.Month, ACFT (1995-07 to 2000-09)

Zoom Mode Disabled

X: 21076.36, Y: 2370.37

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).

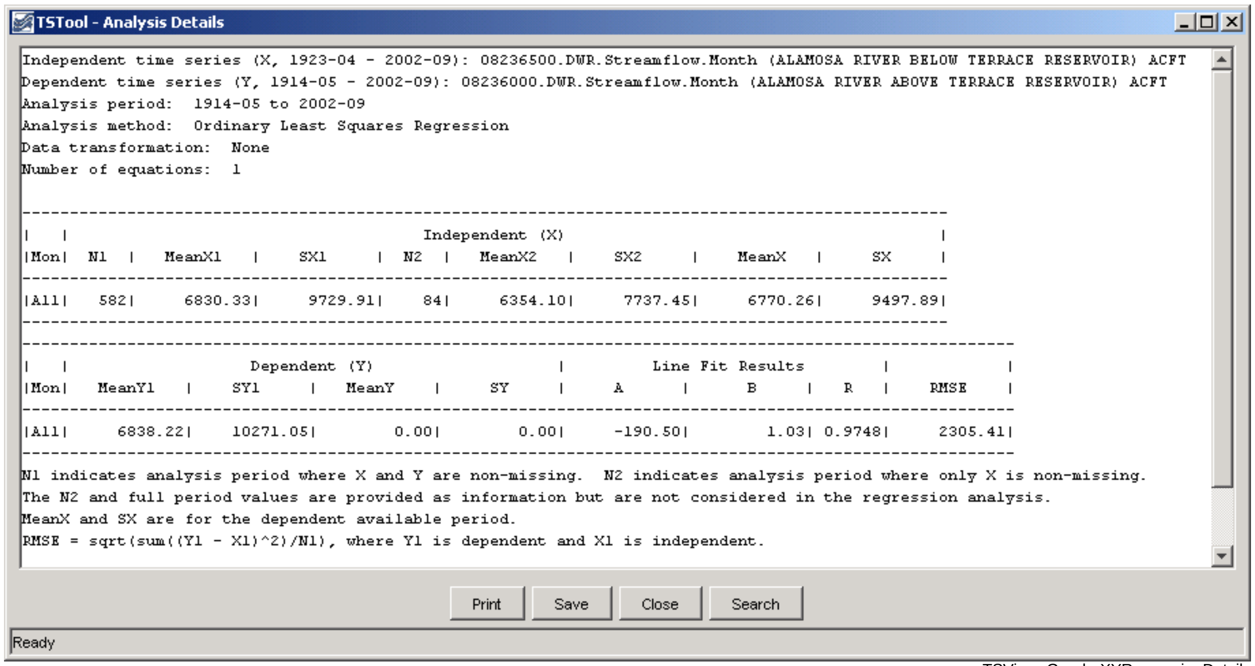
$$\hat{y}_{CI_i} = \left[\bar{y} + B(x_i - \bar{x})\right] \pm \sqrt{2F}\left[\frac{1}{n-2}\sum_{j=1}^{n}\left(\hat{y}_j - y_j\right)^2\right]^{1/2}\left[\frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^{n}(x_j - \bar{x})^2}\right]^{1/2}$$

where:

$\hat{y}_{CI_i}$	= 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_{CIi} 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:



```
Independent time series (X, 1923-04 - 2002-09): 08236500.DWR.Streamflow.Month (ALAMOSA RIVER BELOW TERRACE RESERVOIR) ACFT
Dependent time series (Y, 1914-05 - 2002-09): 08236000.DWR.Streamflow.Month (ALAMOSA RIVER ABOVE TERRACE RESERVOIR) ACFT
Analysis period:  1914-05 to 2002-09
Analysis method:  Ordinary Least Squares Regression
Data transformation:  None
Number of equations:  1

-----------------------------------------------------------------------------------------------
|   |                                    Independent (X)                                      |
|Mon| N1  |  MeanX1    |    SX1     | N2  |  MeanX2    |    SX2     |   MeanX    |     SX     |
-----------------------------------------------------------------------------------------------
|All|  582|     6830.33|     9729.91|   84|     6354.10|     7737.45|     6770.26|     9497.89|
-----------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------
|   |                  Dependent (Y)                    |        Line Fit Results         |            |
|Mon|  MeanY1    |    SY1     |   MeanY    |     SY     |     A      |     B      |   R   |    RMSE    |
--------------------------------------------------------------------------------------------------------
|All|     6838.22|    10271.05|        0.00|        0.00|     -190.50|        1.03| 0.9748|     2305.41|
--------------------------------------------------------------------------------------------------------
N1 indicates analysis period where X and Y are non-missing.  N2 indicates analysis period where only X is non-missing.
The N2 and full period values are provided as information but are not considered in the regression analysis.
MeanX and SX are for the dependent available period.
RMSE = sqrt(sum((Y1 - X1)^2)/N1), where Y1 is dependent and X1 is independent.
```

TSView_Graph_XYRegressionDetails

Example Analysis Details

The RMS error (or *RMSE*) is calculated in the following way:

$SSE = \Sigma(X_i - Y_i)^2$ = Sum of Square Errors
$MSE = SSE/N$ = Mean of Sum of Square Errors
$RMSE = \sqrt{MSE}$ = Square Root of the MSE

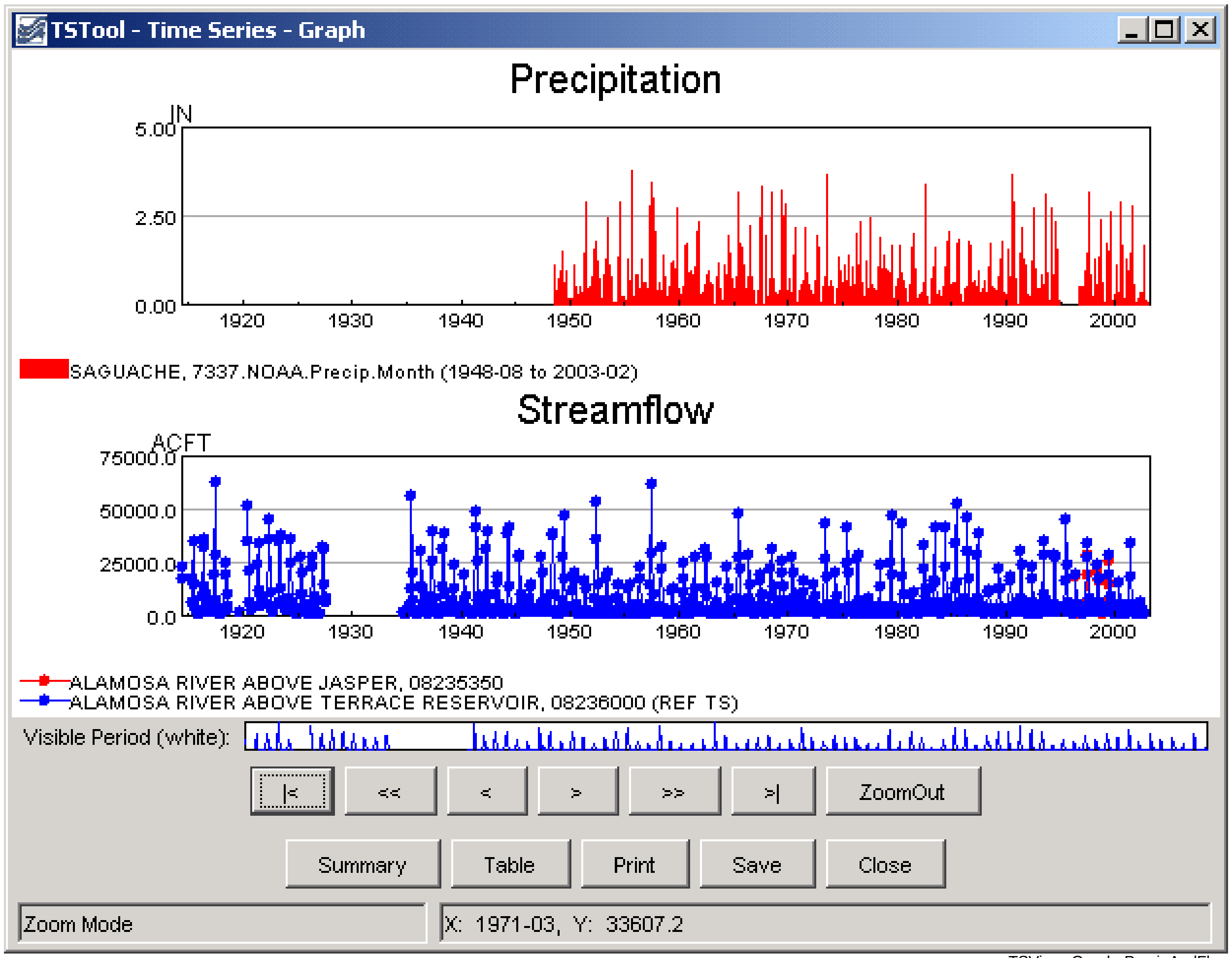
The *RMSE* can have different meanings, depending on how the data are being analyzed:

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 (*Yest*) 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).



TSView_Graph_PrecipAndFlow

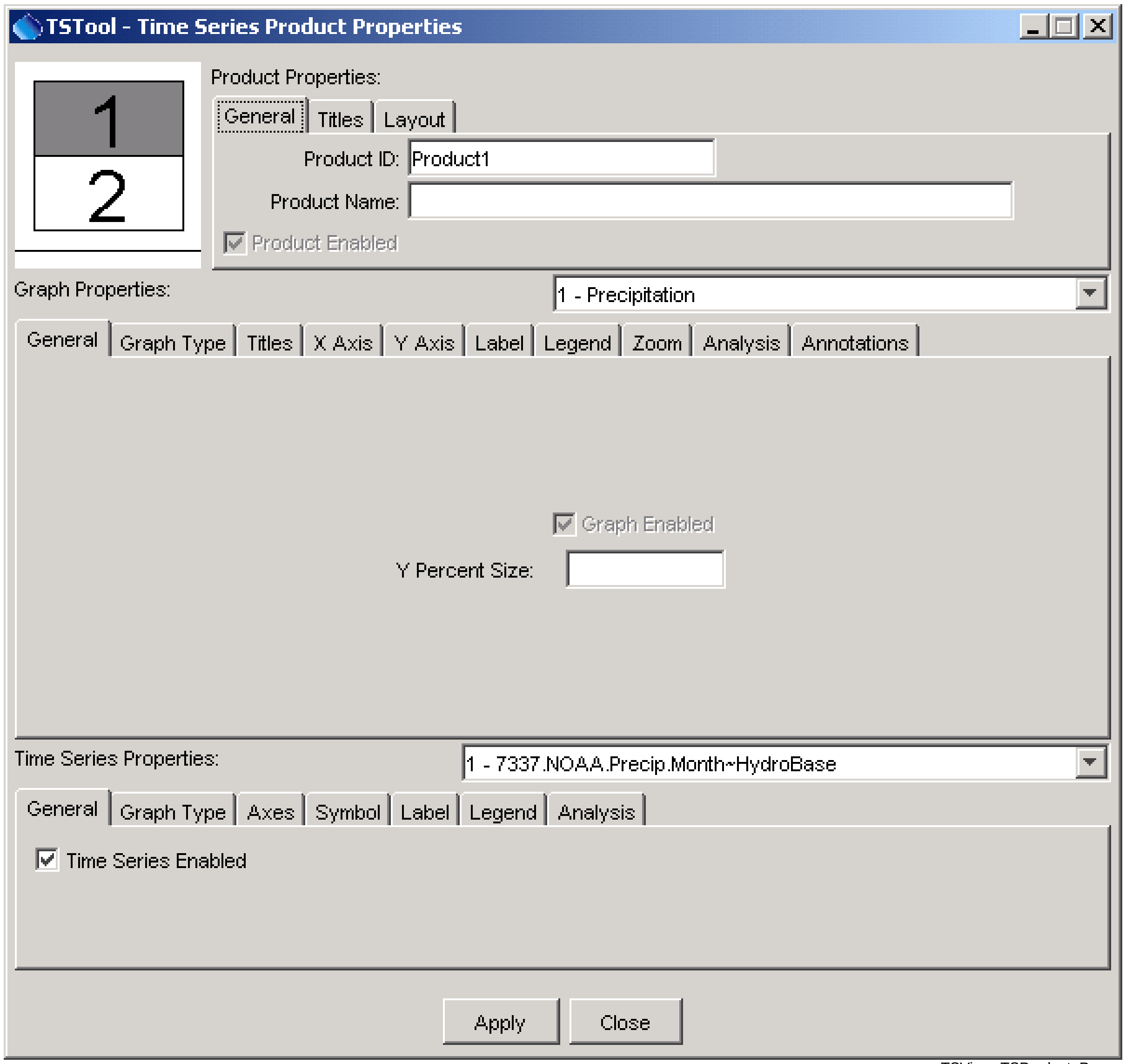
Example Graph Product showing Precipitation and Streamflow

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:



Product ID: Product1
Product Name:
☑ Product Enabled
1 - Precipitation
☑ Graph Enabled
Y Percent Size:
1 - 7337.NOAA.Precip.Month~HydroBase
☑ Time Series Enabled

TSView_TSProduct_Props_All

Tabbed Panel to Edit Time Series Product Properties

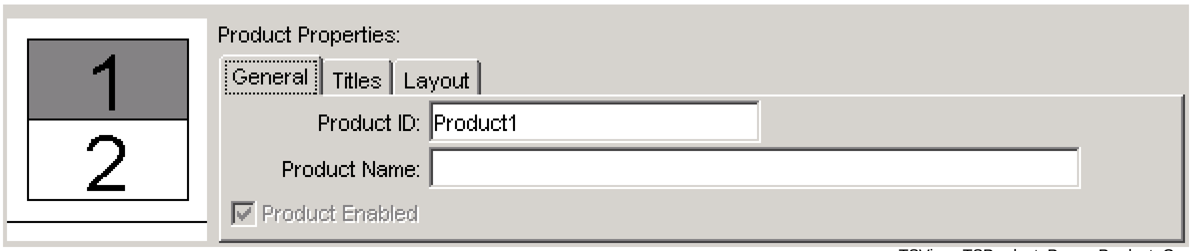
The time series product properties display as three layers of tabbed panels. Characteristics of the properties window are:

- 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***) is associated with product properties (the page).

- The middle layer of tabs (***Graph Properties***) is 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***) is 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



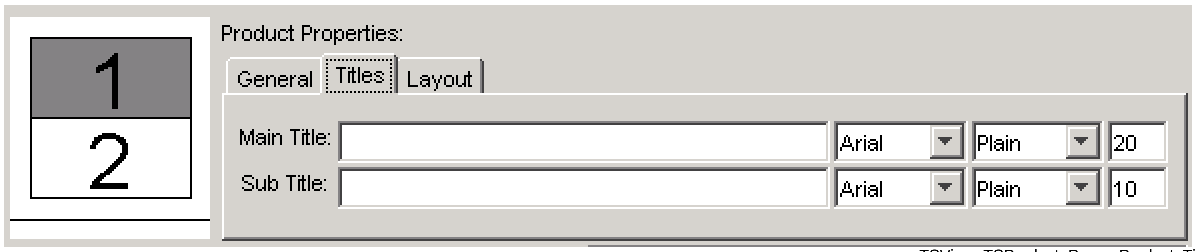
Product ID: Product1
Product Name:

TSView_TSProduct_Props_Product_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



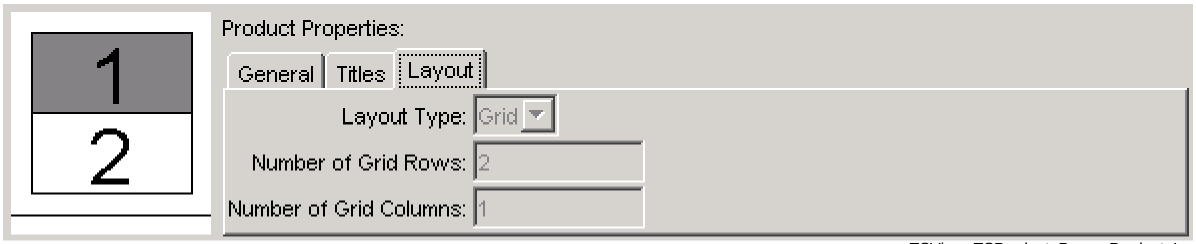
Main Title: | Arial | Plain | 20
Sub Title: | Arial | Plain | 10

TSView_TSProduct_Props_Product_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



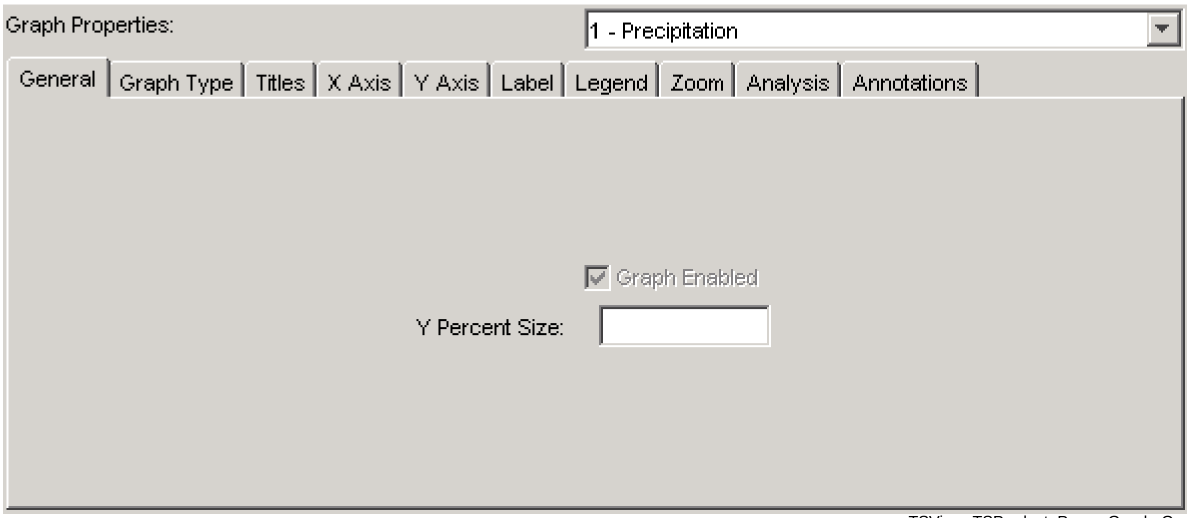
Layout Type: Grid
Number of Grid Rows: 2
Number of Grid Columns: 1

TSView_TSProduct_Props_Product_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

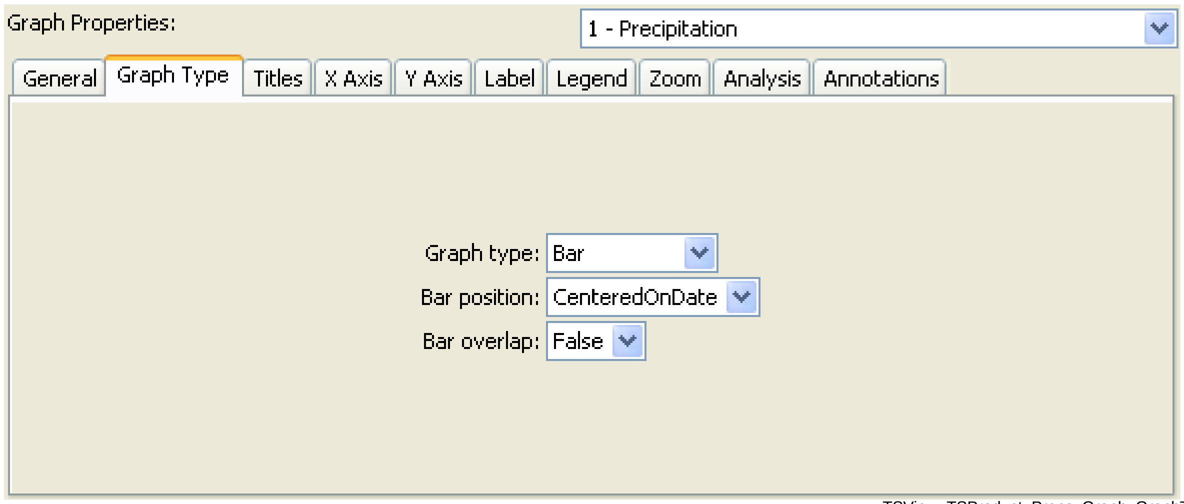


TSView_TSProduct_Props_Graph_General

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



Graph Properties: 1 - Precipitation

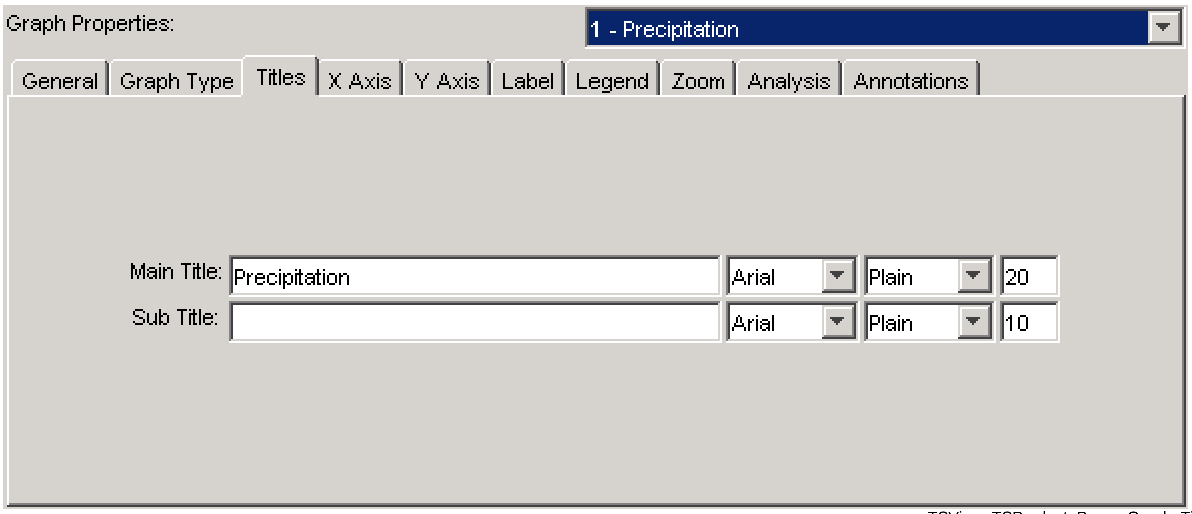
Graph type: Bar
Bar position: CenteredOnDate
Bar overlap: False

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



Main Title: Precipitation | Arial | Plain | 20

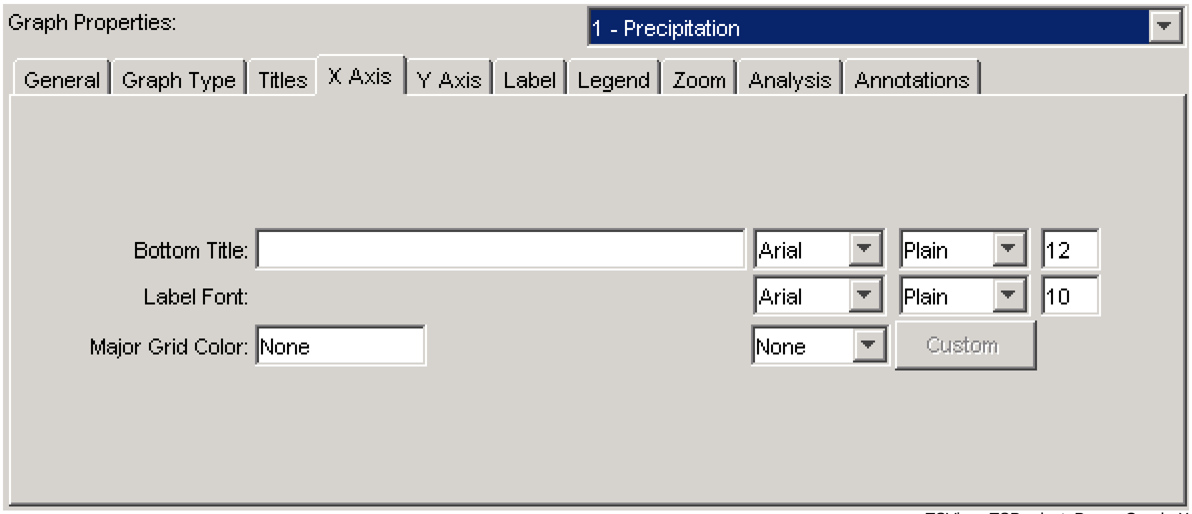
Sub Title: | Arial | Plain | 10

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



Bottom Title: | Arial | Plain | 12

Label Font: Arial | Plain | 10

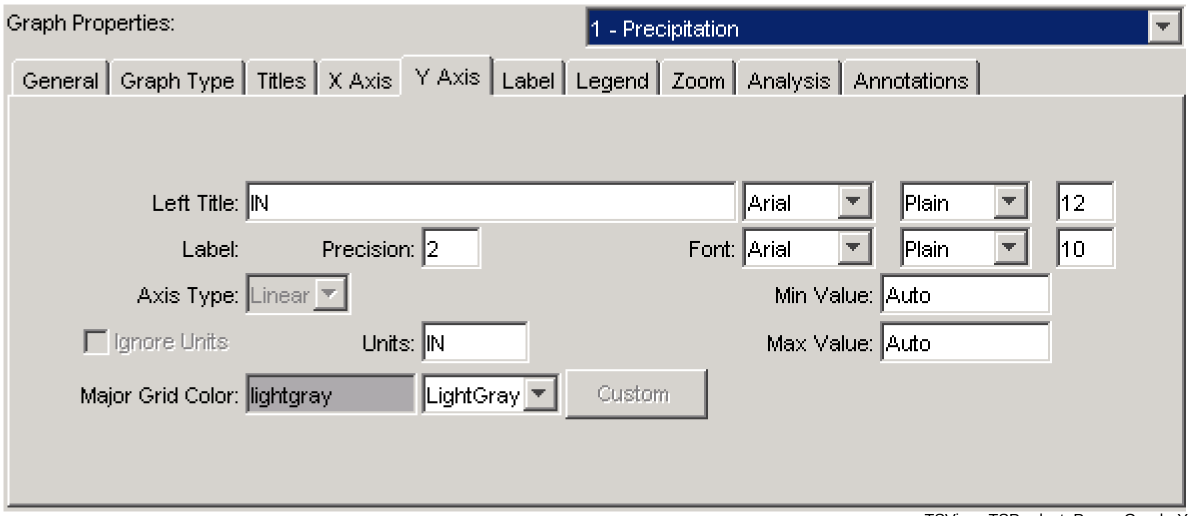
Major Grid Color: None | None

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



Graph Properties: 1 - Precipitation

Left Title: IN Arial Plain 12

Label: Precision: 2 Font: Arial Plain 10

Axis Type: Linear Min Value: Auto

Ignore Units Units: IN Max Value: Auto

Major Grid Color: lightgray LightGray

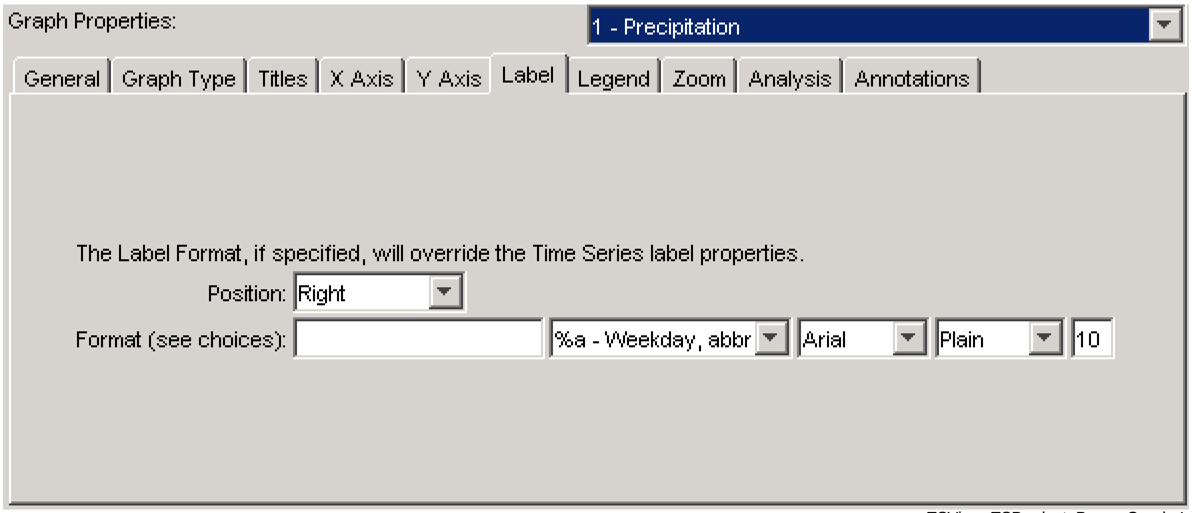
TSView_TSProduct_Props_Graph_YAxis

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



The Label Format, if specified, will override the Time Series label properties.

Position: Right

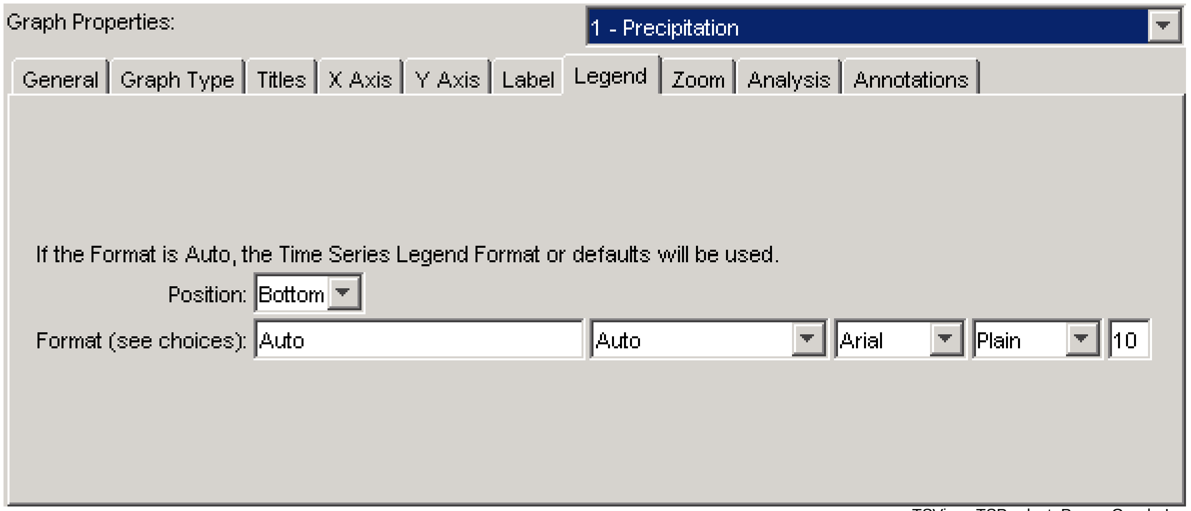
Format (see choices): %a - Weekday, abbr | Arial | Plain | 10

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



If the Format is Auto, the Time Series Legend Format or defaults will be used.

Position: Bottom

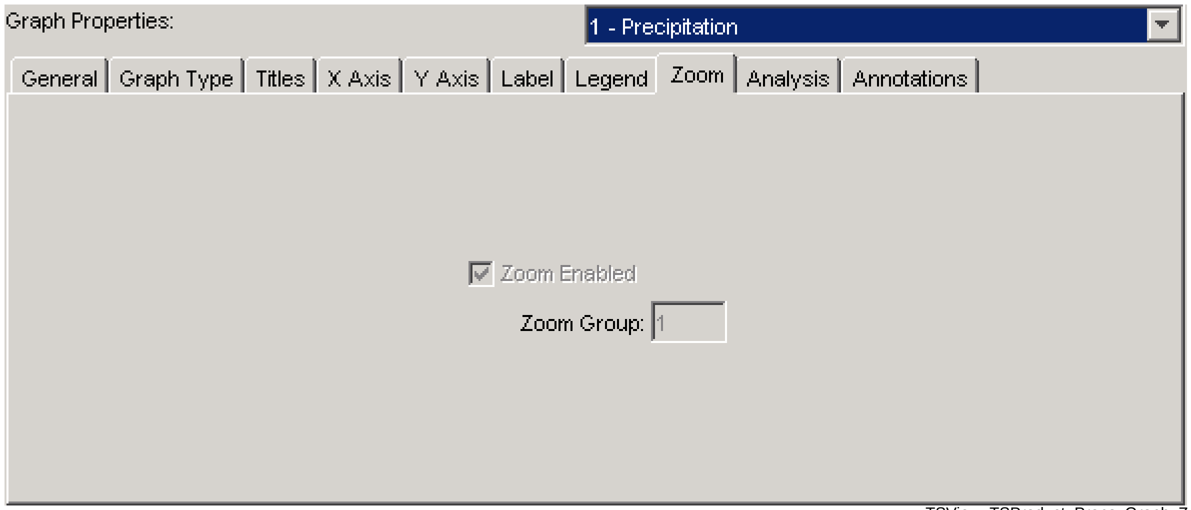
Format (see choices): Auto | Auto | Arial | Plain | 10

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



☑ Zoom Enabled

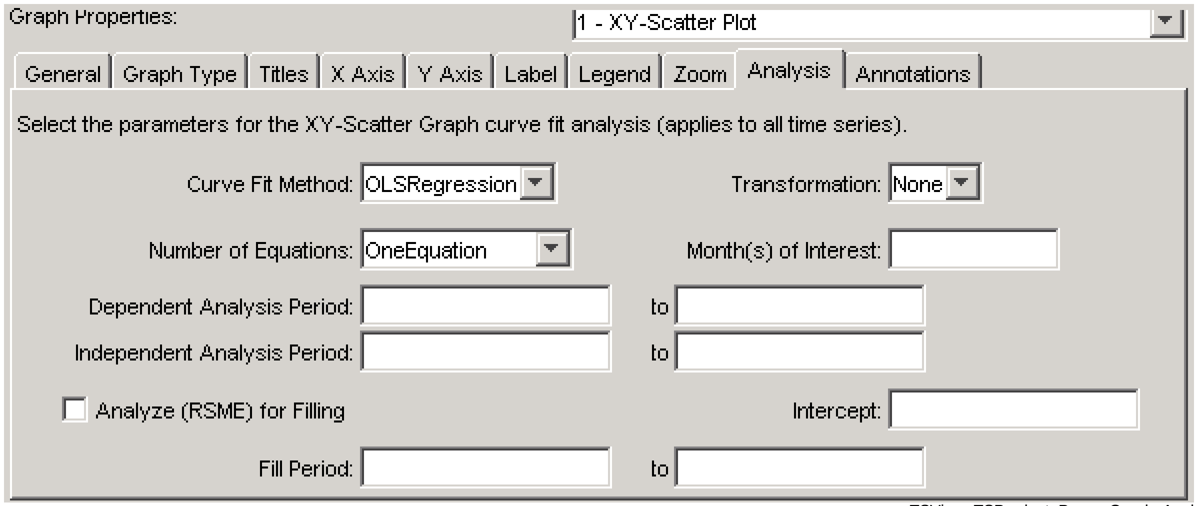
Zoom Group: 1

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



Select the parameters for the XY-Scatter Graph curve fit analysis (applies to all time series).

Curve Fit Method: OLSRegression   Transformation: None

Number of Equations: OneEquation   Month(s) of Interest:

Dependent Analysis Period: to

Independent Analysis Period: to

☐ Analyze (RSME) for Filling   Intercept:

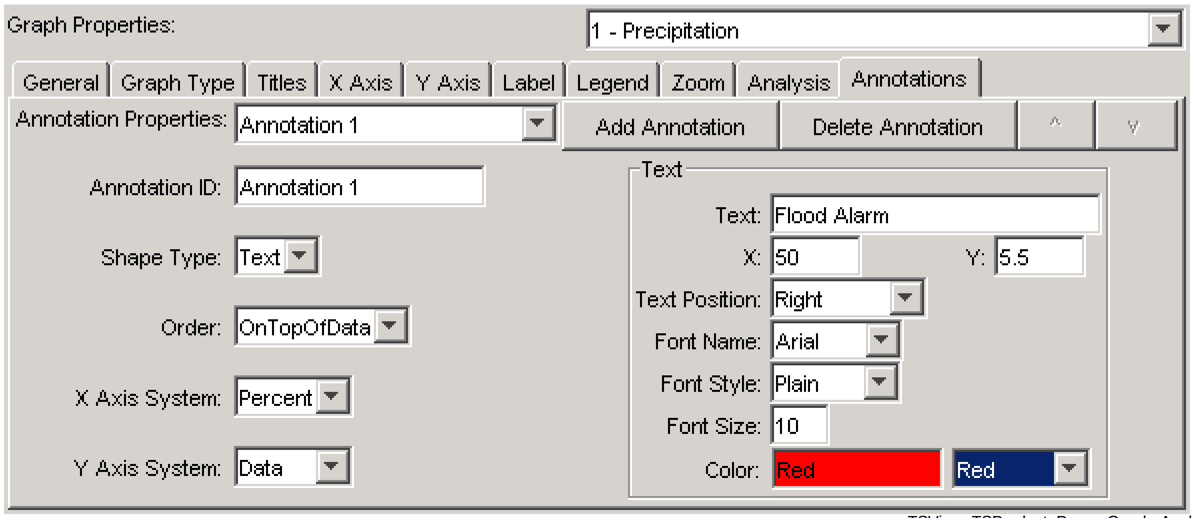
Fill Period: to

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



Graph Properties: 1 - Precipitation
Annotation Properties: Annotation 1
Annotation ID: Annotation 1
Shape Type: Text
Order: OnTopOfData
X Axis System: Percent
Y Axis System: Data
Text: Flood Alarm
X: 50
Y: 5.5
Text Position: Right
Font Name: Arial
Font Style: Plain
Font Size: 10
Color: Red Red

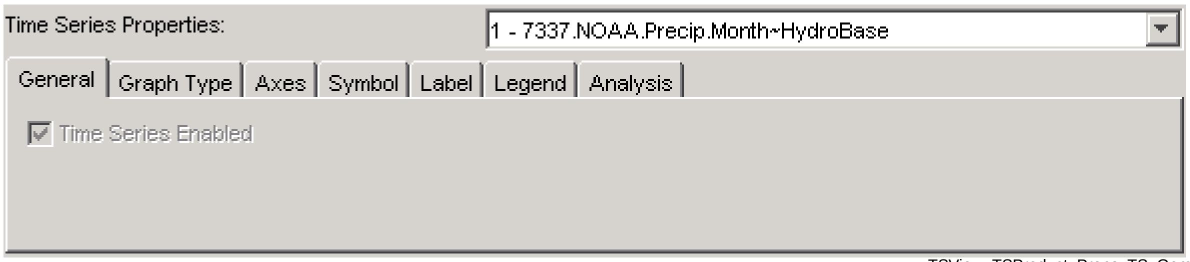
TSView_TSProduct_Props_Graph_Analysis

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

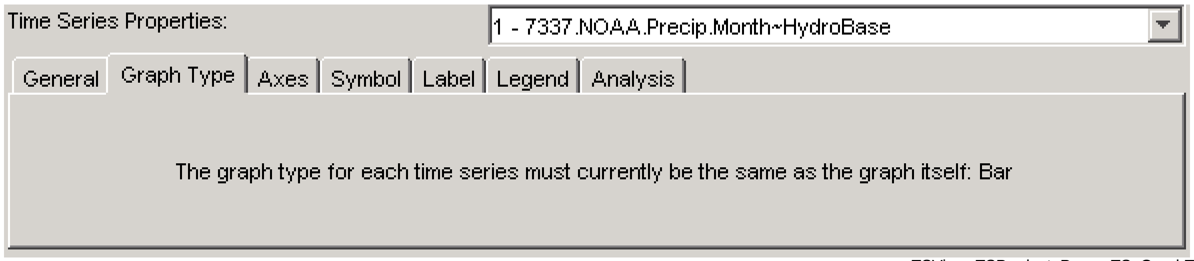


TSView_TSProduct_Props_TS_Gemeral

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



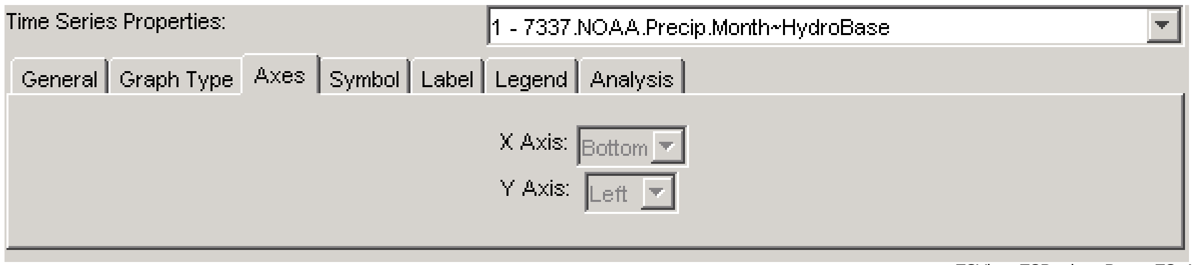
The graph type for each time series must currently be the same as the graph itself: Bar

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



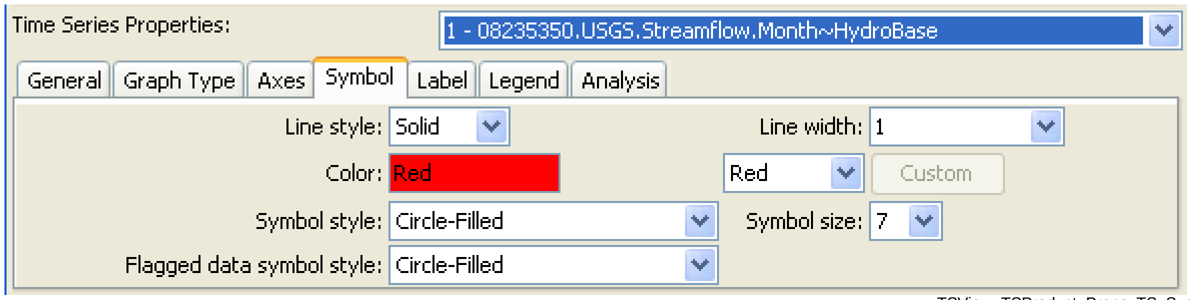
X Axis: Bottom
Y Axis: Left

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


Time Series Properties: 1 - 08235350.USGS.Streamflow.Month~HydroBase

Line style: Solid
Line width: 1
Color: Red
Red

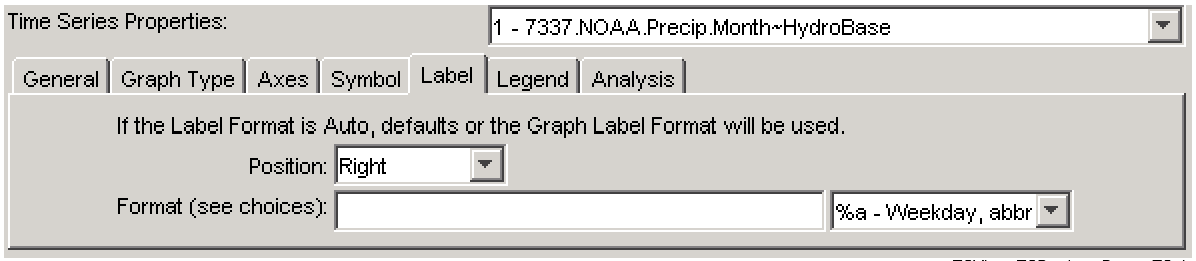
Symbol style: Circle-Filled
Symbol size: 7
Flagged data symbol style: Circle-Filled

TSView_TSProduct_Props_TS_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


Time Series Properties: 1 - 7337.NOAA.Precip.Month~HydroBase

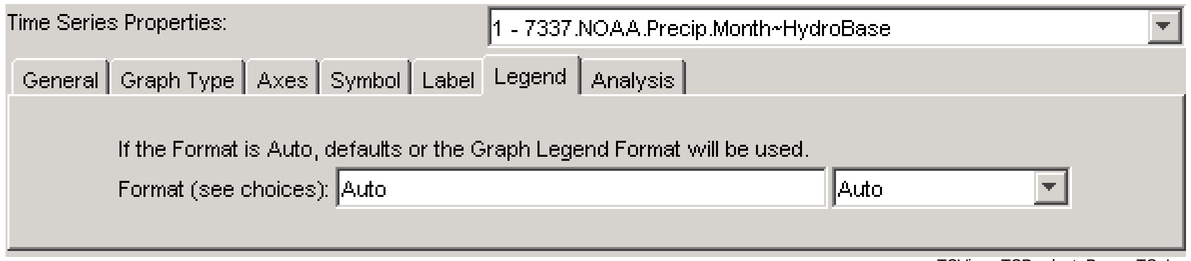
If the Label Format is Auto, defaults or the Graph Label Format will be used.
Position: Right
Format (see choices):
%a - Weekday, abbr

TSView_TSProduct_Props_TS_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


Time Series Properties: 1 - 7337.NOAA.Precip.Month~HydroBase

If the Format is Auto, defaults or the Graph Legend Format will be used.
Format (see choices): Auto
Auto

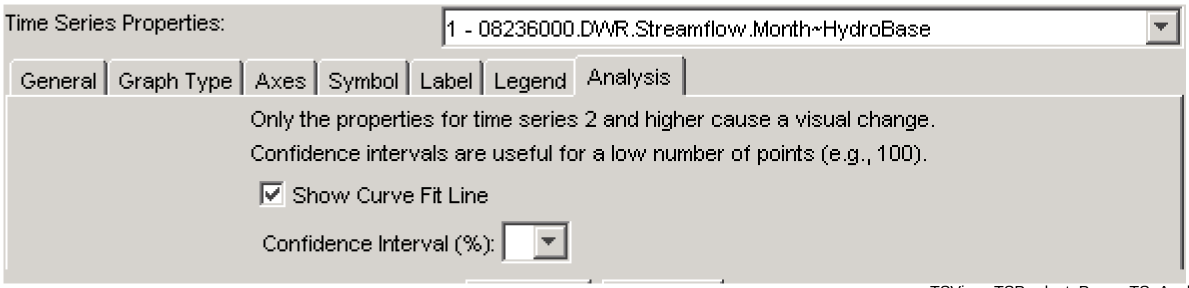
TSView_TSProduct_Props_TS_Legend

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 that 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


Time Series Properties: 1 - 08236000.DWR.Streamflow.Month~HydroBase

Only the properties for time series 2 and higher cause a visual change.
Confidence intervals are useful for a low number of points (e.g., 100).

☑ Show Curve Fit Line

Confidence Interval (%):

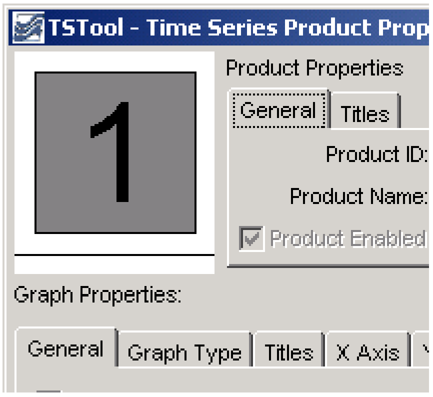
TSView_TSProduct_Props_TS_Analysis

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

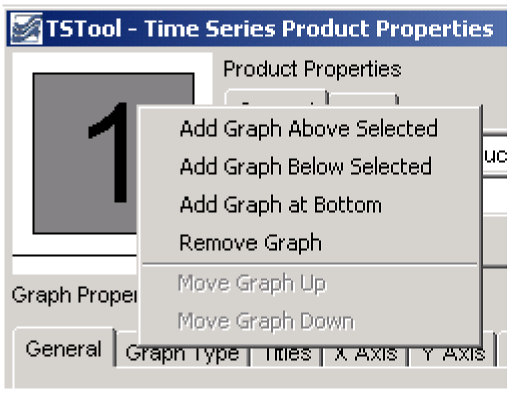
The default page layout for graphs is to display all time series in one graph. In this configuration, the layout area at the top-left corner of the time series product window will display as shown below:



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

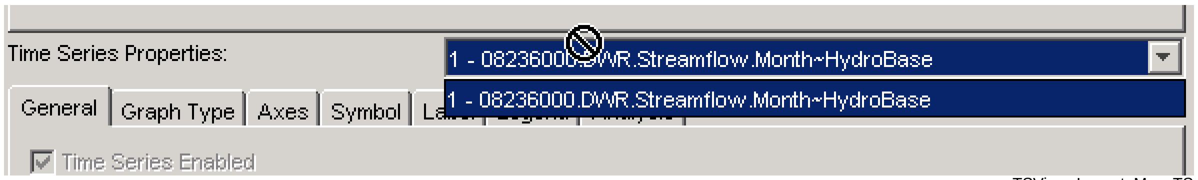
The actions taken by the menus are described below:

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 by 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:


Time Series Properties:
1 - 08236000.DWR.Streamflow.Month~HydroBase
1 - 08236000.DWR.Streamflow.Month~HydroBase
General
Graph Type
Axes
Symbol
Time Series Enabled

TSView_Layout_MoveTS

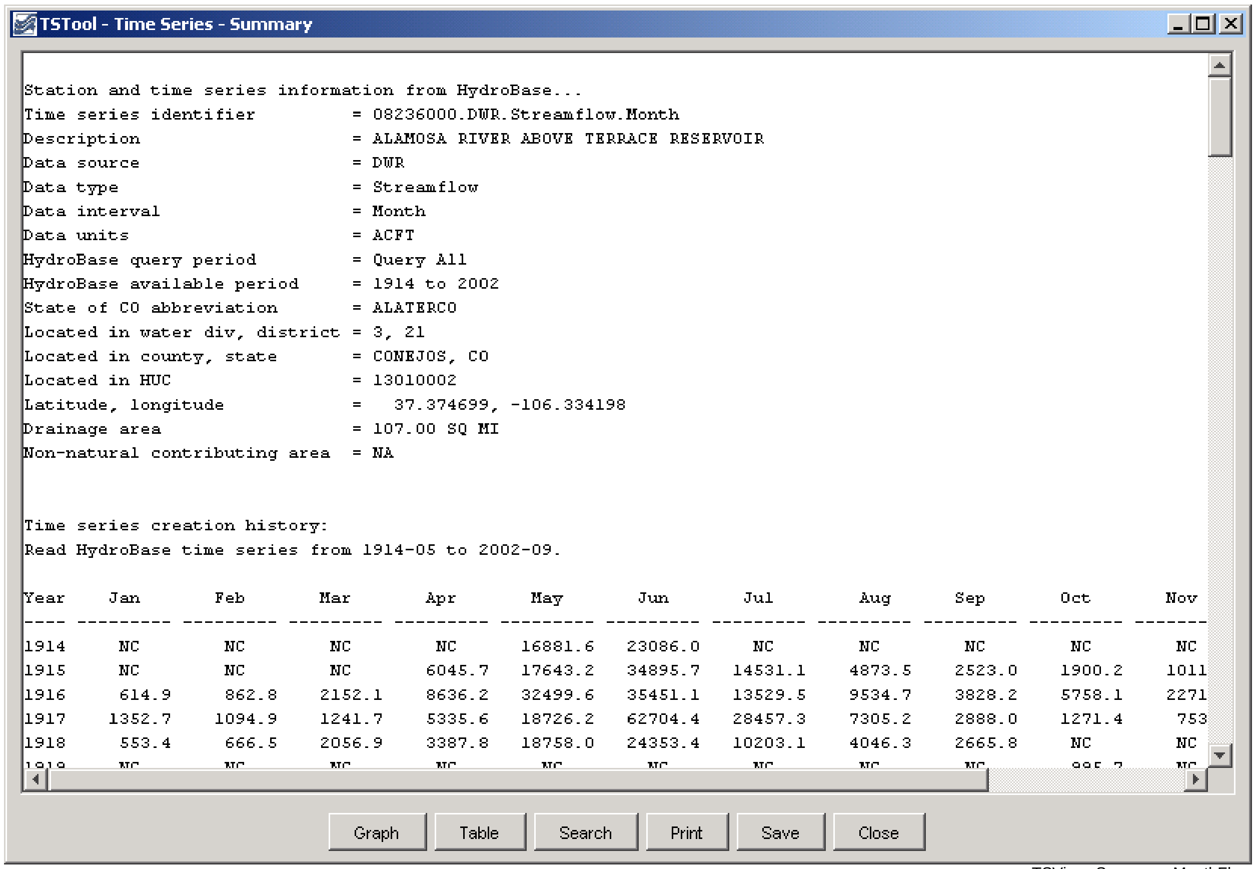
Release the mouse over the graph in the layout area that is to receive the time series. The time series will then be removed from the original graph and will be inserted into the new graph as the last time series in the list.

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.



```
Station and time series information from HydroBase...
Time series identifier         = 08236000.DWR.Streamflow.Month
Description                    = ALAMOSA RIVER ABOVE TERRACE RESERVOIR
Data source                    = DWR
Data type                      = Streamflow
Data interval                  = Month
Data units                     = ACFT
HydroBase query period         = Query All
HydroBase available period     = 1914 to 2002
State of CO abbreviation       = ALATERCO
Located in water div, district = 3, 21
Located in county, state       = CONEJOS, CO
Located in HUC                 = 13010002
Latitude, longitude            =   37.374699, -106.334198
Drainage area                  = 107.00 SQ MI
Non-natural contributing area  = NA


Time series creation history:
Read HydroBase time series from 1914-05 to 2002-09.

Year    Jan       Feb       Mar       Apr       May       Jun       Jul       Aug       Sep       Oct       Nov
---- --------- --------- --------- --------- --------- --------- --------- --------- --------- --------- -------
1914     NC        NC        NC        NC     16881.6   23086.0     NC        NC        NC        NC        NC
1915     NC        NC        NC     6045.7   17643.2   34895.7   14531.1    4873.5    2523.0    1900.2    1011
1916    614.9     862.8    2152.1    8636.2   32499.6   35451.1   13529.5    9534.7    3828.2    5758.1    2271
1917   1352.7    1094.9    1241.7    5335.6   18726.2   62704.4   28457.3    7305.2    2888.0    1271.4     753
1918    553.4     666.5    2056.9    3387.8   18758.0   24353.4   10203.1    4046.3    2665.8     NC        NC
1919     NC        NC        NC        NC        NC        NC        NC        NC        NC      995.7      NC
```

TSView_Summary_MonthFlow

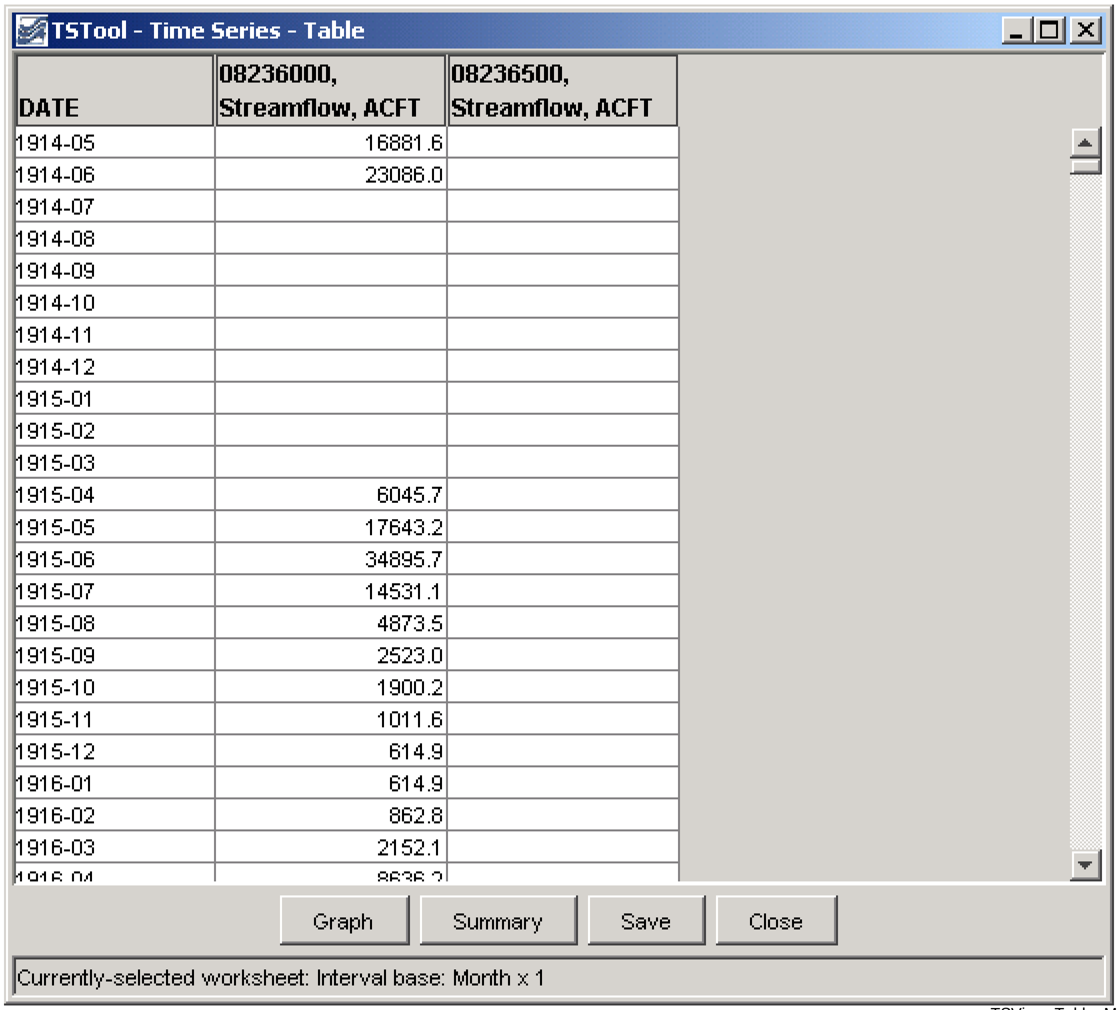
Example Summary View showing Monthly Streamflow

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.



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	

Currently-selected worksheet: Interval base: Month x 1

TSView_Table_MonthFlow

Example Table View showing Monthly Streamflow

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="xxxxxx   xx"

[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
`CurrentDateTime`	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.
`CurrentDateTimeColor`	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`
`LayoutNumberOfColumns`	The number of columns in the product.	Currently always `1`.
`LayoutNumberOfRows`	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`
`MainTitleFontName`	Name of font to use for main title (Arial, Courier, Helvetica, TimesRoman).	`Arial`
`MainTitleFontSize`	Size, in points, for main title.	`20`
`MainTitleFontStyle`	Font style (Bold, BoldItalic, Plain, PlainItalic).	`Plain`
`MainTitleString`	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.** This property is ignored for `ProductType=Report` and must be specified at a subproduct level.	*C:\TEMP\tmp.png* on windows, */tmp/tmp.png* 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`	Time series product type, one of: • `Graph` – graph (see graph subproduct properties). • `Report` – report (see report subproduct properties).	`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 or report files. 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 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
`BarOverlap`	For use with bar graphs. This property controls how bars are positioned relative to each other, one of: • `False` – bars will not overlap and will be side by side with space between. • `True` – bars will overlap. Time series should be specified so that time series with the largest values are drawn first.	`False`
`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`	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. 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.		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`	The legend format is composed of literal characters and/or time series data format specifiers, as follows.		`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.Trace1`)	
	`%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`	Position of the legend relative to the graph: `Bottom`, `InsideLowerLeft`, `InsideLowerRight`, `InsideUpperLeft`, `InsideUpperRight`, `Left`, `None`, `Right`.		`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`.

Report Subproduct Properties

The following table describes the subproduct (report) properties. Limited support for report products are currently enabled. Reports are defined as any format other than graphical output, including raw data formats like delimited and DateValue files. The number of properties for reports will continue to be expanded as additional features are enabled. An example of a report product file is as follows (in this case for NWSRFS FS5Files input type time series):

```
[Product]

ProductType = Report
Enabled = true

[SubProduct 1]

OutputFile = C:\Report_6_Hour
ReportType = DateValue

[Data 1.1]
TSID = FZRDR.NWSRFS.SPEL.6HOUR~NWSRFS_FS5Files

[SubProduct 2]

OutputFile = C:\Report_24_Hour
ReportType = DateValue

[Data 2.1]
TSID = FZRDR.NWSRFS.PELV.24HOUR~NWSRFS_FS5Files
```

Subproduct (Report) Properties

Subproduct (Report) Property	Description	Default
`OutputFile`	Output file when report product is generated in batch mode. If a relative path is given, the file will be written relative to the working directory for the software. **This property is often set at run time by the application.**	*C:\TEMP\tmp_report_N* on windows, */tmp/tmp_report_N* on UNIX
`Enabled`	Indicates whether the sub-product should be processed. Specify as `true` or `false`.	`true`

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`
`FlaggedData SymbolStyle`	Symbol style for data points that have non-empty data flag string. See the `SymbolStyle` property for possible values.	Same as `SymbolStyle`
`FlaggedData SymbolTable`	It is envisioned that this property will be enabled in the future to allow fine-grained control of symbols when data flags are used.	Property not enabled.
`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). It is possible to specify a Line graph type for area graphs.**	Property enabled only for area graphs.

Data (Time Series) Properties (continued)

Data (Time Series) Property	Description	Default
`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`
`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: • `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: • 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: • 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`

Appendix: GeoView Mapping Tools

Color, 2004-05-27, Original Maintained with TSTool, Acrobat Distiller

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

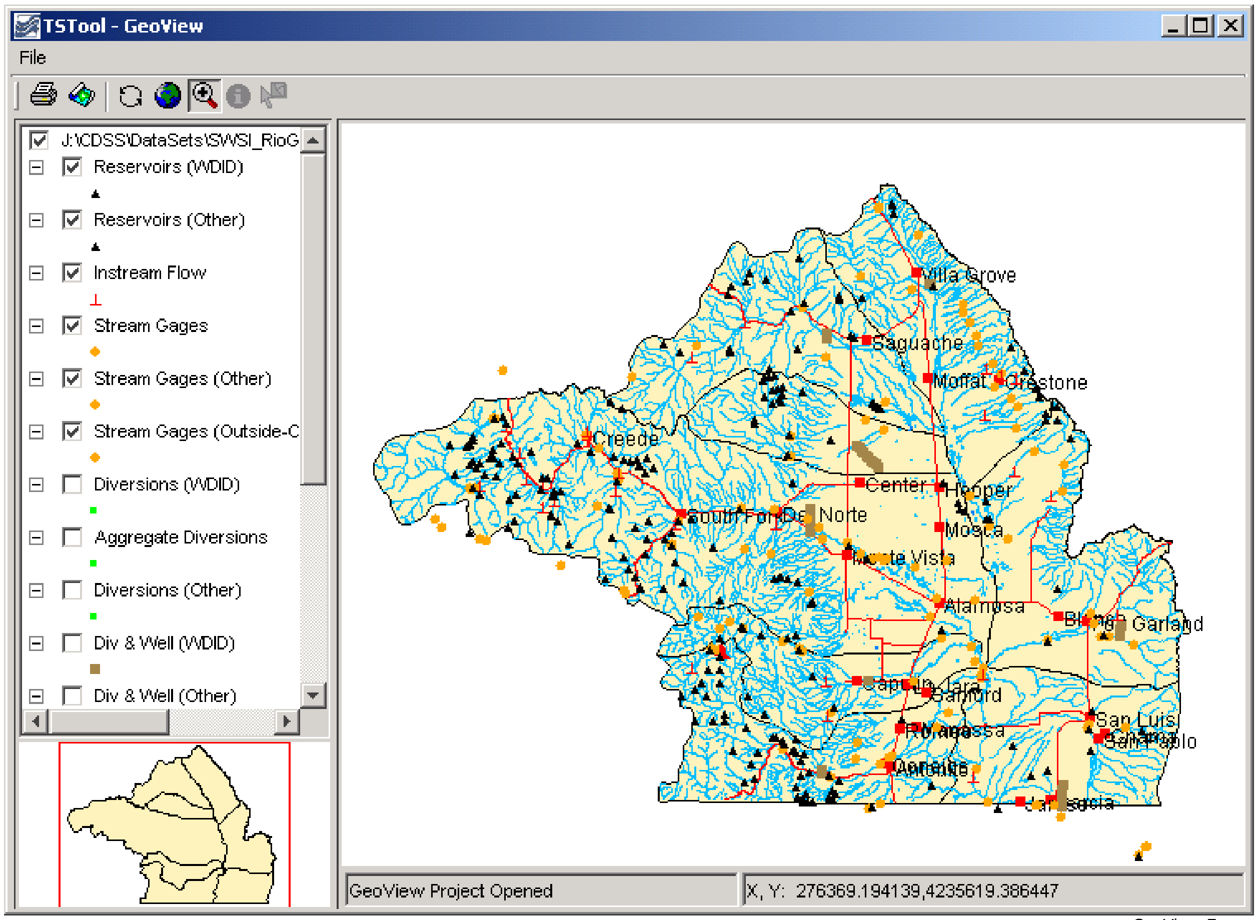
GeoView terminology is similar to other GIS product terminology. Important terms are shown in the following table. These terms are used infrequently in most user interfaces and applications but are visible at times in property dialogs and configuration files.

GeoView Terminology

Term	Description
GeoView	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.
GeoLayer	A data layer, in its "raw" form (e.g., an ESRI Shapefile). The more generic term "layer" is often used.
GeoLayerView	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.
Feature	A general term describing an item on the map, consisting of shape and attribute data.
Shape	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.
Attributes	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).
Symbol	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.
GeoView Project	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.
Application Layer Type	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).



GeoView_Frame

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 `ReferenceLayer` set to `true`. 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.
🔍 ℹ ↖	***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. See the next section for more information about using these features.

Message Areas (bottom)

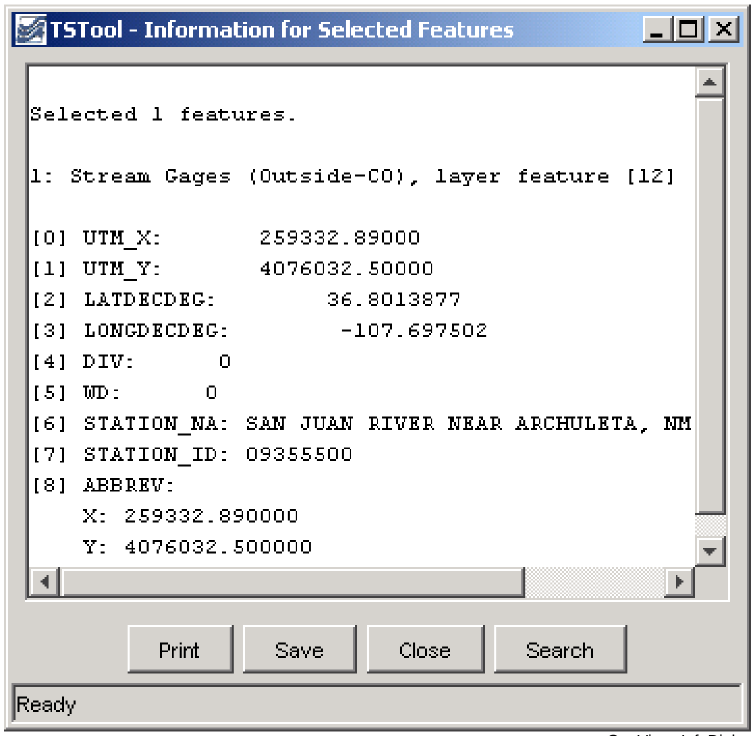
The message areas are used to display the mouse coordinates and provide other feedback.

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	Layers can be enabled/disabled to make the map more readable or useful: 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	Zooming is useful make symbols and labels more readable. To zoom in: 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	To change the symbols and labels for a layer view: 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	The GeoView interface can display information about geographic features (shape and attribute data) from the original geographic data. To do so: 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.



```
Selected 1 features.

1: Stream Gages (Outside-CO), layer feature [12]

[0] UTM_X:       259332.89000
[1] UTM_Y:       4076032.50000
[2] LATDECDEG:        36.8013877
[3] LONGDECDEG:        -107.697502
[4] DIV:      0
[5] WD:      0
[6] STATION_NA: SAN JUAN RIVER NEAR ARCHULETA, NM
[7] STATION_ID: 09355500
[8] ABBREV:
    X: 259332.890000
    Y: 4076032.500000
```

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.**

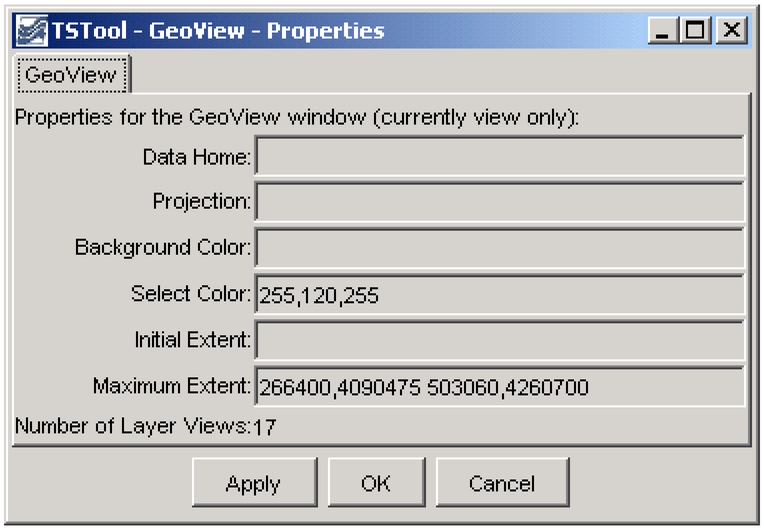
Select features

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.


TSTool - GeoView - Properties

GeoView

Properties for the GeoView window (currently view only):

Data Home:	
Projection:	
Background Color:	
Select Color:	255,120,255
Initial Extent:	
Maximum Extent:	266400,4090475 503060,4260700

Number of Layer Views:17

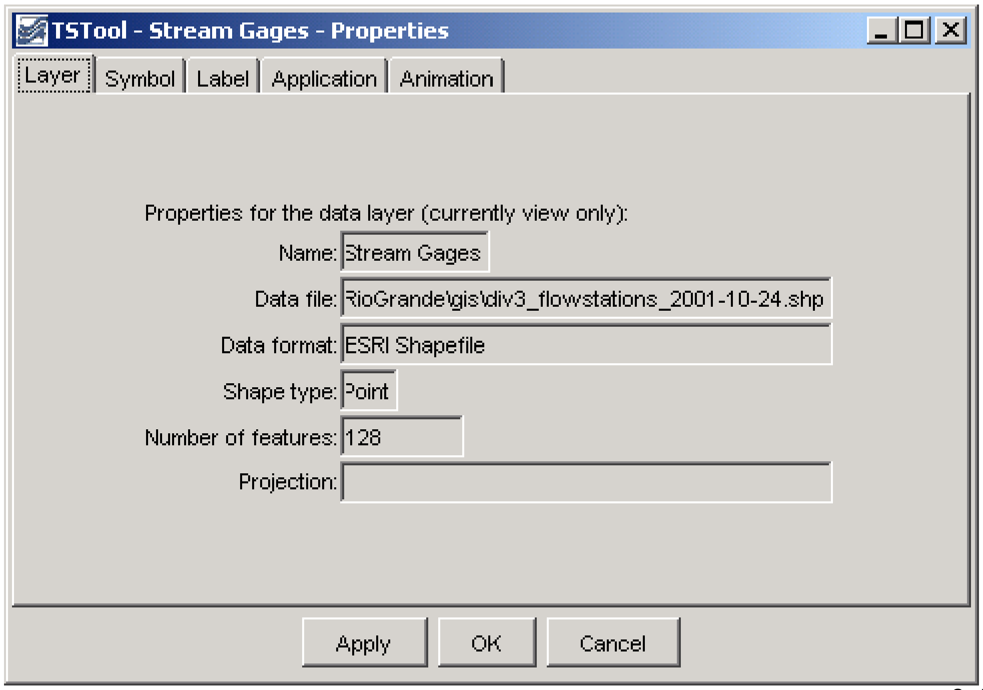
GeoView_Props

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.



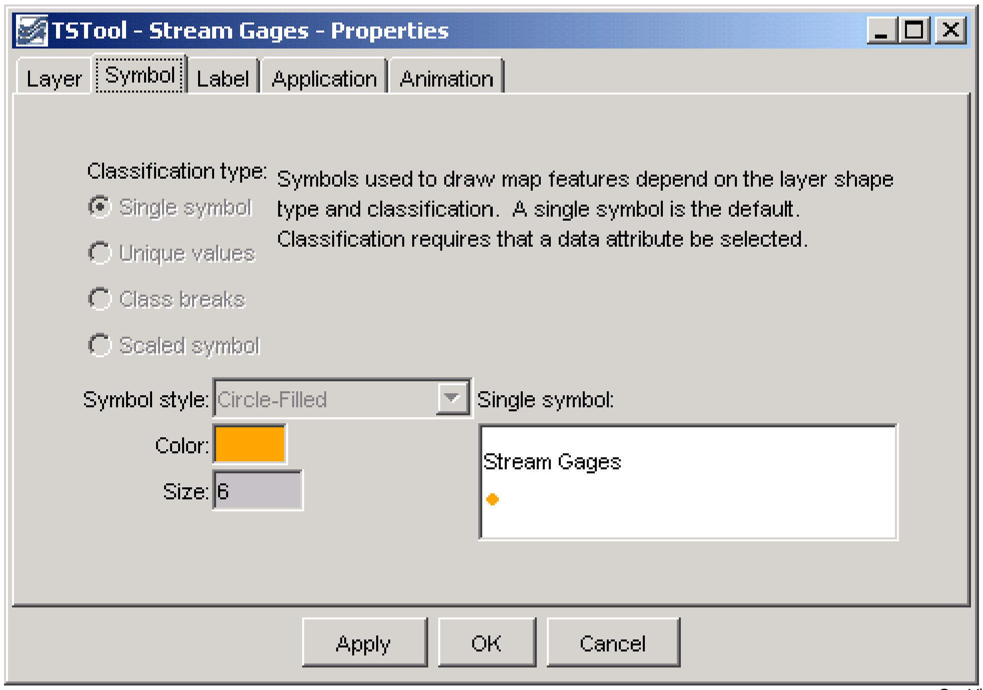
Properties for the data layer (currently view only):

Name:	Stream Gages
Data file:	RioGrande\gis\div3_flowstations_2001-10-24.shp
Data format:	ESRI Shapefile
Shape type:	Point
Number of features:	128
Projection:	

GeoView_Props_Layer

GeoView Layer Properties

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.



TSTool - Stream Gages - Properties

Classification type: Symbols used to draw map features depend on the layer shape type and classification. A single symbol is the default. Classification requires that a data attribute be selected.

[✓ Single symbol]

Unique values

Class breaks

Scaled symbol

Symbol style: Circle-Filled

Color: 🟧

Size: 6

Single symbol:

Stream Gages

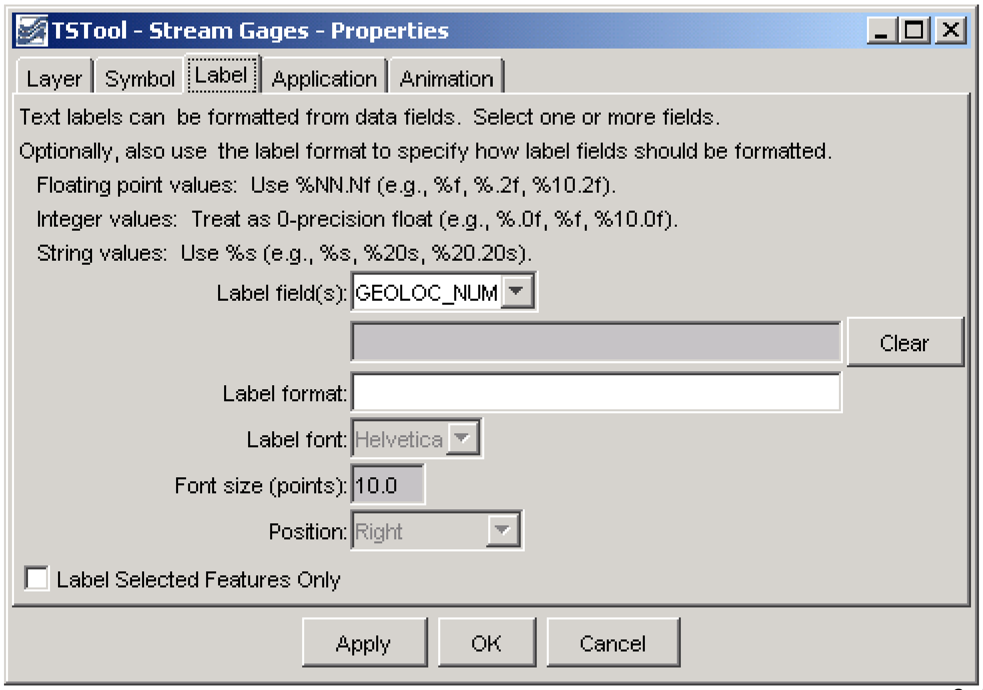
●

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.



Text labels can be formatted from data fields. Select one or more fields.
Optionally, also use the label format to specify how label fields should be formatted.
Floating point values: Use %NN.Nf (e.g., %f, %.2f, %10.2f).
Integer values: Treat as 0-precision float (e.g., %.0f, %f, %10.0f).
String values: Use %s (e.g., %s, %20s, %20.20s).

Label field(s): GEOLOC_NUM
Label format:
Label font: Helvetica
Font size (points): 10.0
Position: Right
Label Selected Features Only

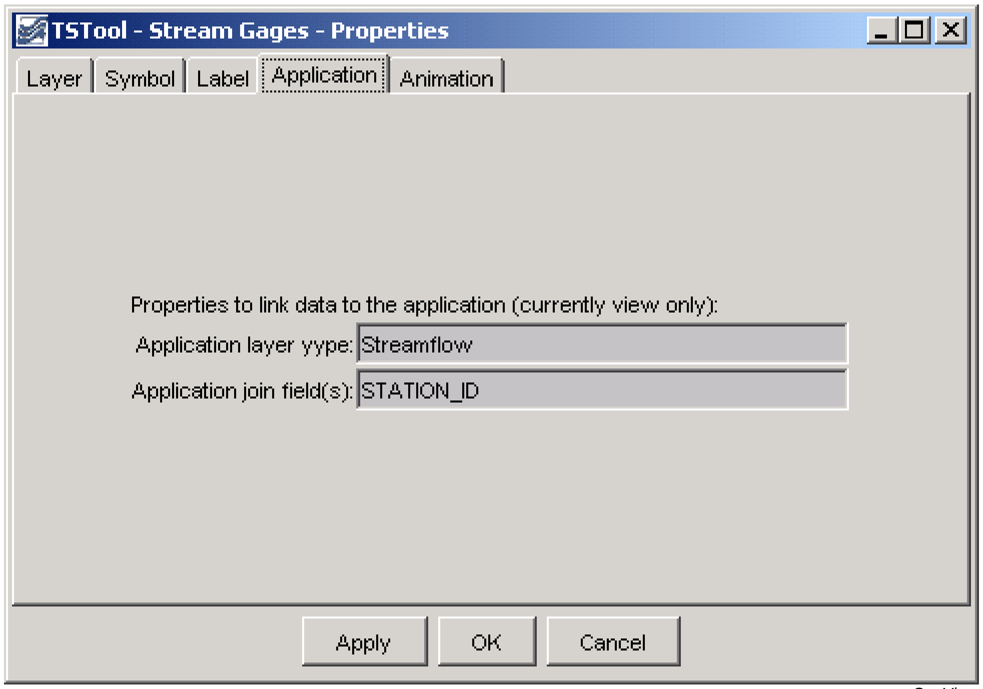
GeoView_Props_Label

Layer View Label Properties

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 field with 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
```



Properties to link data to the application (currently view only):

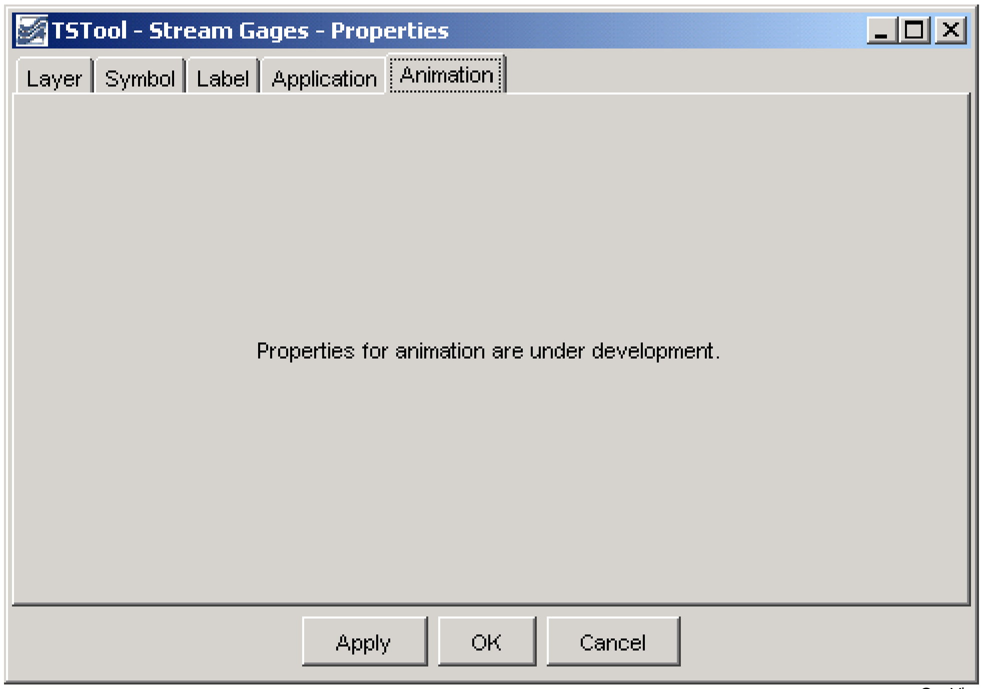
Application layer yype: Streamflow

Application join field(s): STATION_ID

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.



Properties for animation are under development.

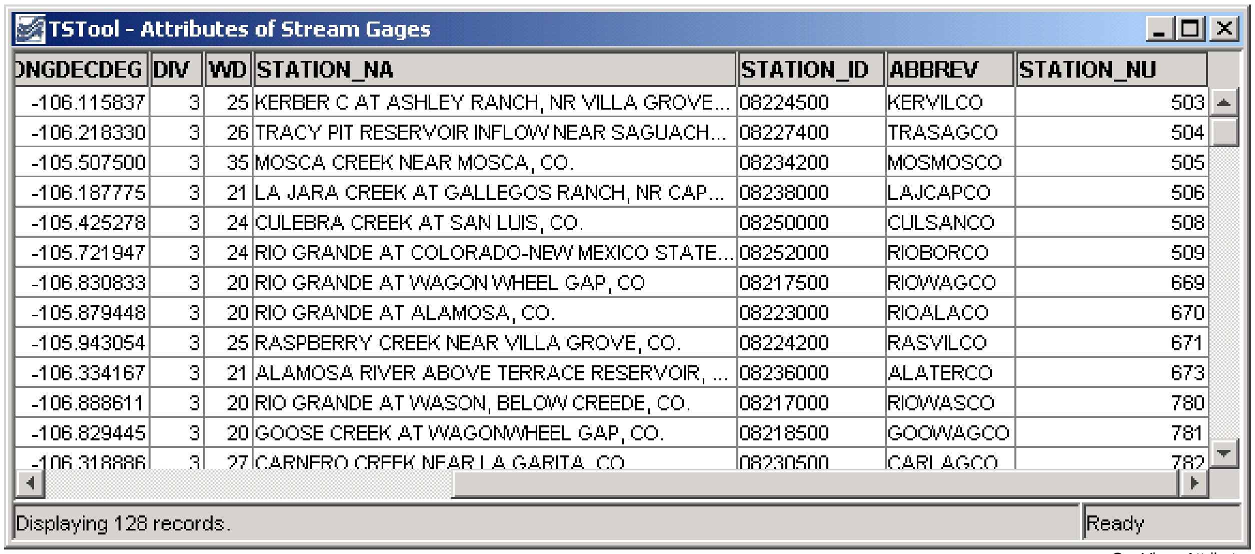
GeoView_Props_Animation

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:



TSTool - Attributes of Stream Gages

ONGDECDEG	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 CREEDE, CO.	08217000	RIOWASCO	780
-106.829445	3	20	GOOSE CREEK AT WAGONWHEEL GAP, CO.	08218500	GOOWAGCO	781
-106.318886	3	27	CARNERO CREEK NEAR LA GARITA, CO	08230500	CARLAGCO	782

Displaying 128 records. Ready

GeoView_Attributes

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 *.shp* file extension and checking the internal file format. No projection is assumed but the `Projection` property for the GeoView 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 `Projection` property is defined as `Geographic` if ASCII data and `HRAP` 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 `Projection` property is defined as `HRAP`.

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`	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: 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` **Only the first option is currently enabled**. See the discussion after this table for examples of how to specify colors.	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 considererd. 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	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 is 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	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 patter (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 and user-defined tables is 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 `SymbolClassField` 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 `SymbolClassField` 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"
```

This page is intentionally blank.

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.

Documentation Binder Spine Labels

This page, when printed, can be used for a spine in a binder.

Colorado's Decision Support Systems (CDSS)

TSTool - Time Series Tool