

VERSION 4.0



TS-WAVE™

User's and Developer's Guide

Visual Numerics, Inc. – United States

Corporate Headquarters
12657 Alcosta Blvd, Suite 450
San Ramon, CA 94583

PHONE: 925.415.8300

FAX: 925.415.9500

e-mail: info@vni.com

Westminster, Colorado

10955 Westmoor Drive, Suite 400
Westminster, CO, 80021

PHONE: 303.379.3040

FAX: 303.379.2140

e-mail: info@vni.com

Houston, Texas

2500 Wilcrest, Suite 200
Houston, TX 77042

PHONE: 713.784.3131

FAX: 713.781.9260

e-mail: info@vni.com

Visual Numerics S. A. de C. V.

Florencia 57 Piso 10-01
Col. Juarez
Mexico D. F. C. P. 06600
MEXICO

PHONE: +52-55-5514-9730 or 9628

FAX: +52-55-5514-5880

e-mail: avadiillo@mail.internet.com.mx

Visual Numerics International Ltd.

SoanePoint
6-8 Market Place
Reading, Berkshire RG1 2EG
UNITED KINGDOM

PHONE: +44 118.925.5910

FAX: +44 118.925.5912

e-mail: info@vniuk.co.uk

Visual Numerics, Inc.

7/F, #510, Sect. 5
Chung Hsiao E. Road
Taipei, Taiwan 110
ROC

PHONE: +88 622-727-2255

FAX: +88 622-727-6798

e-mail: info@vni.com.tw

Visual Numerics International GmbH

Zettachring 10
D-70567 Stuttgart
GERMANY

PHONE: +49-711-13287-0

FAX: +49-711-13287-99

e-mail: info@visual-numerics.de

Visual Numerics Korea, Inc.

HANSHIN BLDG. Room 801
136-1, MAPO-DONG, MAPO-GU
SEOUL, 121-050
KOREA SOUTH

PHONE: +82-2-3273-2632 or 2633

FAX: +82-2-3273--2634

e-mail: info@vni.co.kr

Visual Numerics SARL

Immeuble le Wilson 1
70, avenue du General de Gaulle
92058 Paris La Defense, Cedex
FRANCE

PHONE: +33-1-46-93-94-20

FAX: +33-1-46-93-94-39

e-mail: info@vni.paris.fr

Visual Numerics Japan, Inc.

GOBANCHO HIKARI BLDG. 4TH Floor
14 GOBAN-CHO CHIYODA-KU
TOKYO, JAPAN 102-0076

PHONE: +81-3-5211-7760

FAX: +81-3-5211-7769

e-mail: vnijapan@vnij.co.jp

© 1990-2004 by Visual Numerics, Inc. An unpublished work. All rights reserved. Printed in the USA. 2004

Information contained in this documentation is subject to change without notice.

IMSL, PV- WAVE, Visual Numerics and PV-WAVE Advantage are either trademarks or registered trademarks of Visual Numerics, Inc. in the United States and other countries.

The following are trademarks or registered trademarks of their respective owners: Microsoft, Windows, Windows 95, Windows NT, Fortran PowerStation, Excel, Microsoft Access, FoxPro, Visual C, Visual C++ - Microsoft Corporation; Motif - The Open Systems Foundation, Inc.; PostScript - Adobe Systems, Inc.; UNIX - X/Open Company, Limited; X Window System, X11 - Massachusetts Institute of Technology; RISC System/6000 and IBM - International Business Machines Corporation; Java, Sun - Sun Microsystems, Inc.; HPGL and PCL - Hewlett Packard Corporation; DEC, VAX, VMS, OpenVMS - Compaq Computer Corporation; Tektronix 4510 Rasterizer - Tektronix, Inc.; IRIX, TIFF - Silicon Graphics, Inc.; ORACLE - Oracle Corporation; SPARCstation - SPARC International, licensed exclusively to Sun Microsystems, Inc.; SYBASE — Sybase, Inc.; HyperHelp - Bristol Technology, Inc.; dBase - Borland International, Inc.; MIFF - E.I. du Pont de Nemours and Company; JPEG - Independent JPEG Group; PNG - Aladdin Enterprises; XWD - X Consortium. Other product names and companies mentioned herein may be the trademarks of their respective owners.

IMPORTANT NOTICE: Use of this document is subject to the terms and conditions of a Visual Numerics Software License Agreement, including, without limitation, the Limited Warranty and Limitation of Liability. If you do not accept the terms of the license agreement, you may not use this documentation and should promptly return the product for a full refund. Do not make illegal copies of this documentation. No part of this documentation may be stored in a retrieval system, reproduced or transmitted in any form or by any means without the express written consent of Visual Numerics, unless expressly permitted by applicable law.

Table of Contents

Preface [v](#)

Typographical Conventions [vi](#)

Technical Support [vii](#)

Chapter 1: Introduction [1](#)

TS-WAVE's Features [3](#)

Starting and Stopping TS-WAVE [7](#)

Chapter 2: User's Guide [11](#)

Navigating through TS-WAVE's Main Interface [12](#)

Getting Started and Configuring Your Session [14](#)

Opening a Data File [16](#)

Designing your Page [19](#)

Displaying Your Data [23](#)

Analyzing Your Data [41](#)

Exporting Your Data [51](#)

Saving and Using Templates [59](#)

Using Resource Files [62](#)

Printing [65](#)

Batch Processing [66](#)

Chapter 3: Getting Started Tutorial [69](#)

Defining a Data Source [70](#)

Data Analysis Using the Standard Function: Smooth [74](#)

Plotting Parameters [76](#)

Creating and Viewing a Tab File [85](#)

Creating a Batch File 88

Chapter 4: User's Reference 89

TS-WAVE Menus and Dialogs 89

Resource Files 164

Using Shortcut Keys 171

FORTTRAN Format Strings 173

Chapter 5: Developer's Reference 175

General Requirements for Adding TS-WAVE
User Functions and Custom Data Handlers 176

TS-WAVE User Functions 180

Tips for Creating User Functions 183

Creating a Derived Parameter 183

Creating a Help File 187

Creating a New Plot Window 187

TS-WAVE Data Handlers 196

Setting Up Your Data Handler 199

The Data Handler Functions 204

<dhtype>_ReadFile Function 204

<dhtype>_ReadData Function 205

<dhtype>_WriteFile Function 205

Tips For Creating Data Handlers 207

TS-WAVE's Utility Functions 209

TS-WAVE's DataManager Functions 209

DM_addFCN Function 209

DM_addPSrc Function 210

DM_getSrcList Function 211

DM_getSrc Function 211

DM_getParmList Function	212
DM_getParm Function	213
TS-WAVE's User Interface Functions	214
GU_newParmlist_Component Function	214
GU_updateParmList_Component Function	215
GU_readParmList_Component Function	216
TS-WAVE's PrintManager Functions	217
PM_getDefaultDriver Function	217
PM_getCurrentDriver Function	218
PM_getDefaultPrinter Function	219
PM_getCurrentPrinter Function	219
PM_getDriverList Function	220
PM_getPrintFile Function	221
PM_isPrintToFileEnabled Function	221
PM_setPrintFile Function	222
TS-WAVE's Miscellaneous Utility Functions	223
TS-WAVE System Variables	223
Customization Options	225
 TS-WAVE Index	 229

Preface

TS-WAVE User's Guide and Developer's Reference explains how to use TS-WAVE, a powerful tool from Visual Numerics, Inc. for quickly analyzing and plotting data. This manual contains the following parts:

- **Chapter 1, *Introduction*** — Provides an overview of the scope of the most commonly used features of TS-WAVE.
- **Chapter 2, *User's Guide*** — This section describes the process of reading in data, creating and customizing graphs, and creating output.
- **Chapter 3, *Getting Started Tutorial*** — This section provides a user new to TS-WAVE with quick mini tours of the basic functionality.
- **Chapter 4, *User's Reference*** — A complete guide to the menus and dialog boxes in TS-WAVE.
- **Chapter 5, *Developer's Reference*** — This section is intended for the experienced PV-WAVE programmer who will create customized Data Handlers and user functions.
- ***TS-WAVE Index*** — Contains an alphabetical list of subjects described in this manual with page references.

Typographical Conventions

- The following notation means “select the **Graph Object** function from the **Create** menu”:

 Select **Create=>Graph Object**.

- In this user’s guide, “click” means to press and release the left mouse button quickly; “drag” means to hold down the left mouse button while moving the mouse.
- Keyboard keys are written like this <Keyname>. For example, <Return>, <Control>, and <X>.
- <Control>-<X> means to press the <X> key while holding down the <Control> key.

Technical Support

If you have problems installing, unlocking, or running your software, contact Visual Numerics Technical Support by calling:

Office Location	Phone Number
North American PV-WAVE Family Technical Support Westminster, Colorado	303-379-3033
Visual Numerics Corporate Headquarters San Ramon, California	925-415-8300
North American IMSL Family Technical Support Houston, Texas	713-784-3131
France	+33-1-46-93-94-20
Germany	+49-711-13287-0
Japan	+81-3-5211-7760
Korea	+82-2-3273-2633
Mexico	+52-55-5514-9730
Taiwan	+88-622-727-2255
United Kingdom	+44-118-925-5910

Users outside the U.S., France, Germany, Japan, Korea, Mexico, Taiwan, and the U.K. can contact their local agents.

Please be prepared to provide the following information when you call for consultation during Visual Numerics business hours:

- The name and version number of the product. For example, TS-WAVE 4.0.
- The type of system on which the software is being run. For example, SPARCstation, IBM RS/6000, HP 9000 Series 700.
- The operating system and version number. For example, HP-UX 11.0 or IRIX 6.5.3.
- A detailed description of the problem.

FAX and E-mail Inquiries

Contact Visual Numerics Technical Support staff by sending a FAX to:

Office Location	FAX Number
North American PV-WAVE Family Technical Support Westminster, Colorado	303-379-2140
Visual Numerics Corporate Headquarters San Ramon, California	925-807-0145
North American IMSL Family Technical Support Houston, Texas	713-781-9260
France	+33-1-46-93-94-39
Germany	+49-711-13287-99
Japan	+81-3-5211-7769
Korea	+82-2-3273-2634
Mexico	+52-55-514-5880
Taiwan	+88-622-727-6798
United Kingdom	+44-118-925-5912

or by sending E-mail to:

Office Location	E-mail Address
North American PV-WAVE Family Technical Support Westminster, Colorado	support@vni.com
Visual Numerics Corporate Headquarters San Ramon, California	info@vni.com
North American IMSL Family Technical Support Houston, Texas	support@vni.com

Office Location	E-mail Address
France	support@vni-paris.fr
Germany	support@visual-numerics.de
Japan	support@vni-j.co.jp
Korea	support@vni.co.kr
Mexico	avadillo@mail.internet.com.mx
Taiwan	support@vni.com.tw
United Kingdom	support@vniuk.co.uk

Electronic Services

General e-mail	info@boulder.vni.com
Support e-mail	support@vni.com
World Wide Web	http://www.vni.com
Anonymous FTP	ftp.boulder.vni.com
FTP Using URL	ftp://ftp.boulder.vni.com/VNI/
PV-WAVE Mailing List:	Majordomo@boulder.vni.com
To subscribe include: in the body of your message:	subscribe ts-wave YourEmailAddress
To post messages	ts-wave@boulder.vni.com

x

TS-WAVE

Introduction

The Most Extensive Application Foundation for Custom Time-series Data Analysis

TS-WAVE, the time series analysis component of the PV-WAVE[®] Family of products, is the perfect balance between an off-the-shelf application and custom development. The core of TS-WAVE offers the most complete set of data management and display techniques for the visualization of time-based data, and it provides the most reliable base for organizations building time-series data analysis applications. In addition, TS-WAVE provides a comprehensive interface for developing custom Data Handlers for reading and managing data. TS-WAVE's unparalleled ability to handle large, proprietary legacy datasets sets it apart. The powerful PV-WAVE data import and user interface development features underlying TS-WAVE makes the development of custom Data Handlers easy. For custom data analysis, TS-WAVE provides an interface for extending the standard analysis capabilities of the TS-WAVE. For more thorough discussions on using this functionality, see the *User's Guide* and *User's Reference* chapters of this manual.

Optimum Tool for Analysis and Visualization of Time-based Data

TS-WAVE provides a comprehensive and intuitive graphical user interface for the analysis and visualization of time-series data sets. The TS-WAVE user has complete control over the layout of pages and graph appearance. Through the use of templates, standard reports can be developed and run on new data sets as they

become available. Batch processing allows standard reports to be run automatically without user intervention.

Easily Customized for Proprietary Data Formats

Often, TS-WAVE is used for telemetry data gathered from flight tests, range tests or satellites. As is true for many types of time-series data, there is no industry standard format for telemetry data sets. Custom data readers are needed to efficiently read and manage the data. Often the data sets contain additional information about the source of the data and the parameters available in the data set. This additional information is important for the TS-WAVE user to easily display and analyze the data.

The Data Handler in TS-WAVE is a combination of standard user interfaces that perform such tasks as file selection and custom user interfaces for choosing subsets of the data for processing. Once the data source is selected, the standard TS-WAVE user interfaces are used for parameter selection, data display and processing. For more information on the Data Handling features of TS-WAVE, see [TS-WAVE Data Handlers](#) beginning on page 196 of this manual.

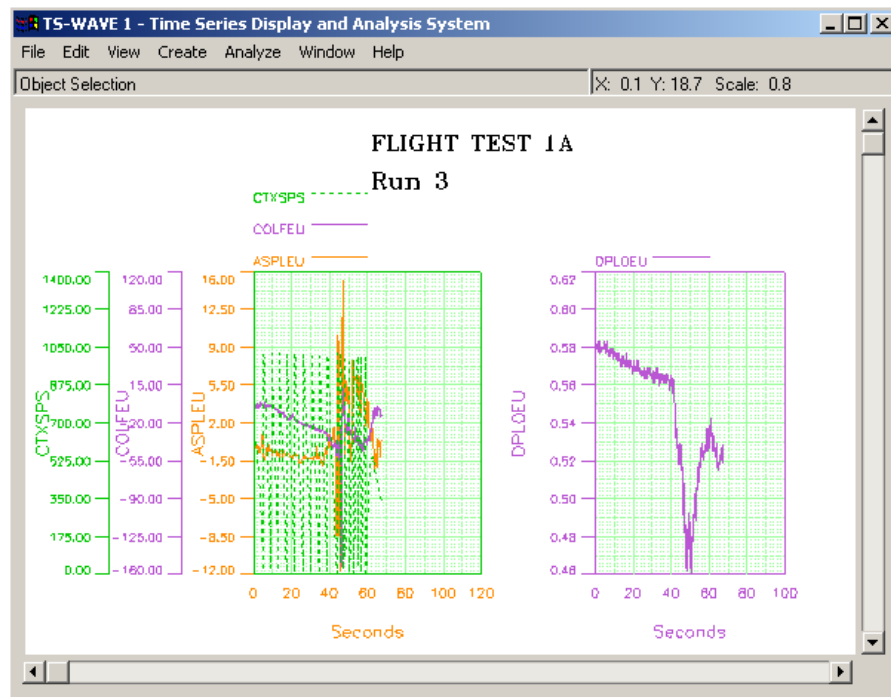


Figure 1-1 Example of Flight Test Analysis

Custom Analysis Using the Full Functionality of PV-WAVE and the IMSL™ Numerical Libraries

Even though TS-WAVE is the most comprehensive time-series analysis package of its kind, users may require other more specialized analyses. To enable this, TS-WAVE provides easy to use interfaces for expanding the analysis capabilities of the core application. These allow users to take full advantage of the extensive features in PV-WAVE for flexible and sophisticated algorithm development.

TS-WAVE developers are provided with standardized user interface components and functions to select parameters from data sources, retrieve data from the Data Handlers and return new parameters to TS-WAVE. The user functions are written using PV-WAVE Advantage procedures, providing developers with the full suite of the IMSL™ Numerical Libraries mathematical and statistical algorithms for use in their user functions. For more details, see the user function discussion under the [Analyze Menu](#) on page 159.

Not only do users get the full advantages of the PV-WAVE applications development environment, and the complete features set of TS-WAVE for advanced time series analysis, they also get the pure power of the IMSL Numerical Libraries for deep analysis of their data. These factors are an unmatched advantage of using TS-WAVE.

TS-WAVE's Features

This section provides an overview of the main features in TS-WAVE. The main features are:

- [*TS-WAVE's Main Interface*](#)
- [*TS-WAVE Data Handlers*](#)
- [*Customization*](#)
- [*Analysis Functions*](#)
- [*Annotation Objects*](#)

TS-WAVE's Main Interface

The main TS-WAVE interface consists of a Title Bar, and multiple menus that are accessible through the Menu Bar, a Message Area where feedback information is returned to the user, and a drawing area where your work is displayed. Throughout this manual, the complete interface is referred to as the TS-WAVE Page.

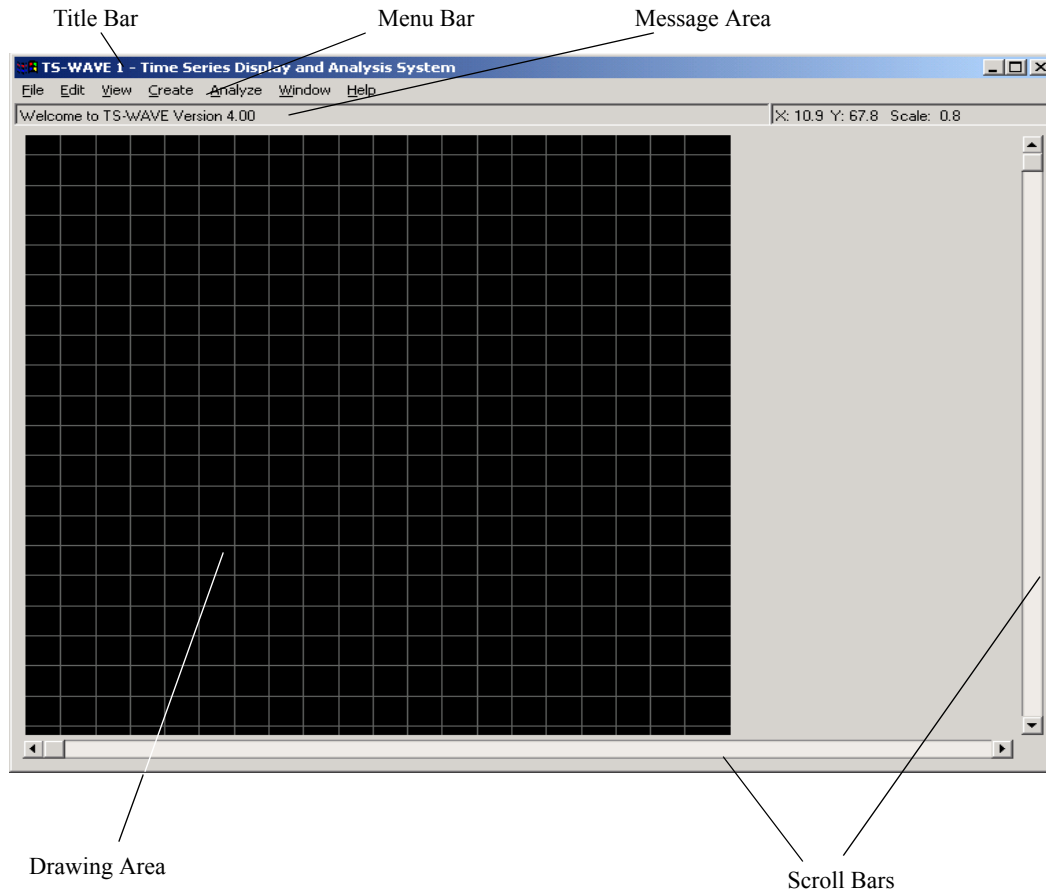


Figure 1-2 The TS-WAVE Main Window

The message area just below the Main menu initially says “Welcome to TS-WAVE 4.0”. This message area typically displays information on the current task that you are performing.

The drawing area is the area in which you create and/or modify the graphical display of your data. By default, TS-WAVE opens one TS-WAVE Page for you to work with, but you can easily open another Page by selecting **New Page** from the **File** dropdown menu ([New Page](#) on page 90). The layout and appearance of this Page are determined by the various resources that are defined in your resources files and can be modified to fit your needs. For more details on resources files, see [Using Resource Files](#) on page 62 in this manual.

TS-WAVE Data Handlers

A Data Handler is the mechanism used by TS-WAVE to read and write specific data file formats. It is a set of PV-WAVE procedures that have been designed specifically for your site-specific data file format. TS-WAVE Data Handlers are typically written to make the most efficient use of memory utilization. For smaller datasets, the data might be read into memory all at once. For larger more complex data files, TS-WAVE uses a technique where only the data being accessed by TS-WAVE (plotting, extraction, analytical calculations, etc.) is loaded into memory at the appropriate time. In this case you may be dealing with many megabytes of data in the actual file, but TS-WAVE only reads the necessary data to accomplish the task.

Data Handlers are created by programmers trained in the use of our PV-WAVE product with specialized training in TS-WAVE development techniques or by Visual Numerics' Consulting group. For more information on Data Handlers, see [TS-WAVE Data Handlers](#) on page 196.

Customization

TS-WAVE provides the ability for an individual user or group of users to customize the product to meet their individual needs and preferences through the use of resource files. Resources that can be modified include output and display options, default printers, and much more. For a complete discussion, see [Resource Files](#) on page 164.

Analysis Functions

TS-WAVE contains built-in standard functions that include many of the analysis routines needed when analyzing data. The standard functions are Bias, Difference, Differentiate, FFT, Gain, Gauss Fit, Smooth, Trim, and Wildpoint. These functions are typically used to create new parameters from the data that can then be saved to a file or displayed on your page. For more information, see [Using the Standard Functions](#) on page 48 and the [Analyze Menu](#) on page 159.

Tab File Output

TS-WAVE provides the ability to export data to tabular ASCII data files, called Tab Files. Tab Files can be imported into third-party applications, incorporated into reports, read back into TS-WAVE and viewed while you are running your TS-WAVE session. Two methods are available for selecting data for export to a Tab File, you can export all the data points for selected parameters or you can interactively select a subset for export. For more information on Tab Files, see [Working with Tab Files](#) on page 52 of the *User's Guide*.

Batch File Processing

Batch Processing is a robust and easy to use automated feature that enables TS-WAVE to be run as a self-executing background process without the use of an interactive graphical user interface. Batch Jobs are capable of performing most tasks possible in an interactive TS-WAVE session. A Batch Job and its related files carry instructions for the automatic loading of templates, data, and printer settings as well as output information. Graphic output can be sent directly to a printer or it can be saved to a file in any of the graphics formats supported by TS-WAVE. Data and data analysis results can be written to a Tab File. Batch Jobs are executed from a central shell script (UNIX) or batch file (Windows) that can be scheduled to run using any automated job scheduler. This gives TS-WAVE users who work with data that changes the ability to automate and standardize the output, analysis and graphical presentation of each new data set. A common use for a TS-WAVE Batch Job is in the processing of data that is collected daily by a separate automated process. You can schedule a job to be run at a specified time that automatically reads the data from the collection process and sends the plotted output to a printer. For more information on Tab Files, see [Batch Processing](#) on page 66 of the *User's Guide*.

Annotation Objects

These are the tools used for annotating your graphs, identifying a spot on a contour plot or adding informative text to your Page. These tools are lines, circles, boxes, text boxes, and headers. For a detailed discussion on each of these tools, see [Create Menu](#) on page 111.

Starting and Stopping TS-WAVE

Starting TS-WAVE

The section explains how to get started using TS-WAVE running under Windows or UNIX.

Starting TS-WAVE on a Windows System

After TS-WAVE is installed, the PV-WAVE Product Family program group icon is created for you. To start TS-WAVE,

Click **Start=>Programs=>PV-WAVE 8.0 Product Family =>TS-WAVE 4.0=>TS-WAVE 4.0**.

A PV-WAVE command console opens and the main window of TS-WAVE opens.

Starting TS-WAVE on a UNIX System

After TS-WAVE is installed, a shell script is created in the PV-WAVE installation directory. Check with your local system administrator if you are unsure of the PV-WAVE installation directory at your site.

Source the setup file for PV-WAVE, where *maindir* is the path to the main installation directory (e.g., `/usr/local/vni`):

From a C-Shell:

```
source <maindir>/wave/bin/wvsetup
```

From a Bourne- or Korn-Shell:

```
.<maindir>/wave/bin/wvsetup.sh
```

Once you have sourced `wvsetup` (or run `wvsetup.sh`), enter:

```
tswave -c
```

at the shell command line.

The main window of TS-WAVE opens.

TS-WAVE also allows a number of execution-time options. To examine the list of available options, enter:

```
tswave -h
```

Execution Time Options

The table below lists many of the flags in TS-WAVE that you can set at run time that will allow you more flexibility.

Executable Flags	Execution-time Options
<code>tswave -h</code>	Print the help message
<code>tswave <No options></code>	Start in runtime (blocking) mode and print output to default logfile ./ts-wave.log for UNIX or C:\ts-wave.log for Windows.
<code>tswave -n</code>	Start in non-blocking mode (Requires a TS-WAVE Developer License)
<code>tswave -c</code>	Print runtime output to the console window
<code>tswave -j jobfile</code>	Run the specified Batch jobfile
<code>tswave -d</code>	Start in debug mode
<code>tswave -m</code>	Print messages to standard output
<code>tswave -p "[x,y]"</code>	Main Window upper left corner position in pixels (enclose in quotes)
<code>tswave -xwidth</code>	Main Window width in pixels (UNIX only)
<code>tswave -ywidth</code>	Main Window height in pixels (UNIX only)
<code>tswave -l logfile</code>	Print runtime output to specified logfile. If logfile not specified, use default logfile ./ts-wave.log for UNIX or C:\ts-wave.log for Windows. Overrides the -c option

Stopping TS-WAVE

The section explains how to stop TS-WAVE.

Step 1 From the **File** menu, select **Exit**.

NOTE From now on, we will use the notation **Menu=>Function** to describe a menu selection. For example: Select **File=>Exit**.

Step 2 (*Non-blocking mode only*) Type `EXIT` at the `<WAVE>` prompt in the PV-WAVE command window to exit the PV-WAVE session.

User's Guide

TS-WAVE is an easy to use, customizable time-series analysis tool. This means that you can quickly begin analyzing your data. This chapter guides you through the process of reading in data, creating and customizing Graph objects and creating output.

The User's Guide contains the following sections:

- *Navigating through TS-WAVE's Main Interface*
- *Getting Started and Configuring Your Session*
- *Opening a Data File*
- *Designing your Page*
- *Displaying Your Data*
- *Analyzing Your Data*
- *Exporting Your Data*
- *Saving and Using Templates*
- *Using Resource Files*
- *Printing*
- *Batch Processing*

Navigating through TS-WAVE's Main Interface

The TS-WAVE menubar contains seven menus that are used to control specific functionality. The menus are **File**, **Edit**, **View**, **Create**, **Analyze**, **Window** and **Help**. Selecting a menu in the menubar displays the options for each menu. Each menu's options is summarized below. However, a thorough discussion on using each menu is found in the [User's Reference](#) beginning on page 89 of this manual.

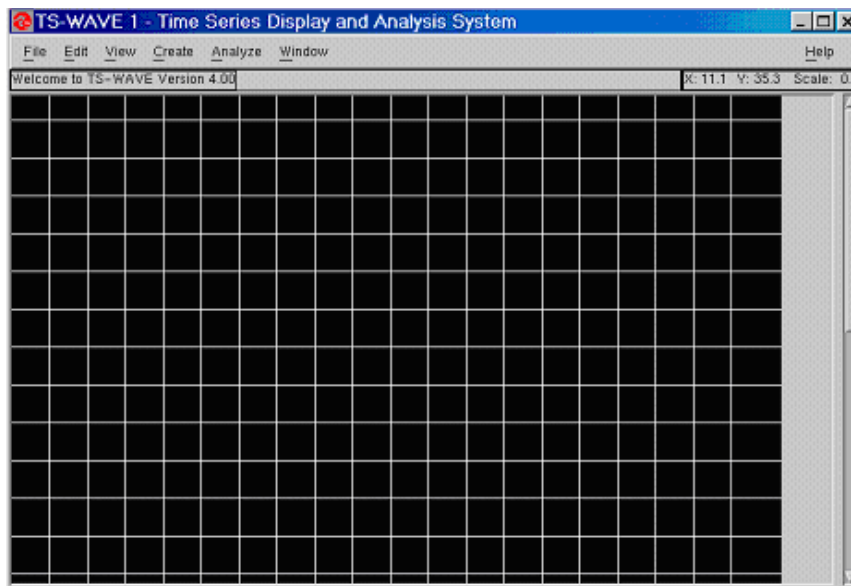


Figure 2-1 The TS-WAVE main interface

The **File** menu is where the user can create, print and close TS-WAVE Pages, manage Data Source, export data, save Template and Session Files, open Template and Session Files, and exit TS-WAVE.

The **Edit** menu is used to edit objects on your Page, but first you must select the object(s). Selecting **Edit=>Object Select** allows you to select the object(s) to be modified. At that point you can cut, copy, paste, and delete any object(s) that have been selected. You can also align Graph objects, access the graph attributes, select and deselect all objects, redraw the Page, group and ungroup objects, and move selected objects in front of or behind other unselected objects.

The **View** menu allows you to set your page magnification, zoom in on your plots, set axis scaling, control the grid settings and colors, and set your preferences for

displaying data and template information, and loading of your data and redrawing the Page.

The **Create** menu is used to add objects to your Page. Available objects are: Graph object, Header object, Contour object, Text object, Line object, Ellipse object and Box object. Additionally, this menu is used to create Batch Jobs and Tab and Pick Files, and view Tab files.

The **Analyze** menu is where analysis tools are located. The submenus are Standard (where the TS-WAVE supplied analysis tools are located) and User (where the user created analysis routines are located).

The **Window** menu allows you to quickly switch between TS-WAVE Pages.

The **Help** menu contains Manuals Online, User Fcn Help, and About TS-WAVE.

Getting Started and Configuring Your Session

TS-WAVE opens with a blank Page that is used as your work area. You may want to configure your session to accommodate the way in which you prefer to work. For example, you can set preferences such as background and foreground colors, grid sizes and more. All of these options are available via the **View** menu shown in [Figure 2-2](#).

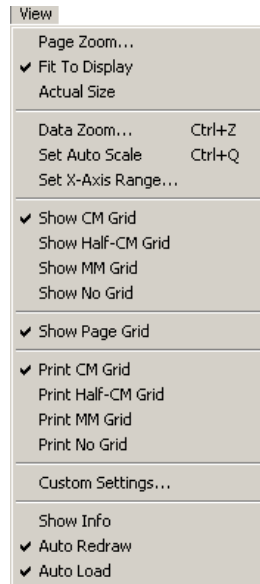


Figure 2-2 TS-WAVE's View Menu dialog

Use the **View** menu to determine how your TS-WAVE Page and your desired background grids will be displayed.

For example, you may want to have your Page fit to your display by selecting **Fit To Display**. Note that depending on your View settings, a centimeter grid may or may not be exactly one centimeter on your display. However, it will always be exactly one centimeter on your printed output device. Also, this will change your **Page Zoom** setting to have the entire Page fit on your screen. When you choose the **Actual Size** view option, the page zoom setting will change to 1.0 and the centimeter grid will be exactly one centimeter. With this setting activated, your displayed Page and printed Page are identical and appear as the actual output size.

Selecting the **View=>Custom Settings** options brings up the **Custom Settings** dialog. From this dialog, you can modify all of the display colors as well as the format and position of the information displayed by the **View=>Show Info** option.

The final three items on the **View** menu allow you to select whether you want the location of your data and Template Files displayed on your Page using **Show Info**, select whether or not you want your Page to redraw automatically using **AutoRedraw**; and select whether or not you want new data that is selected to be displayed automatically on a Graph object using **AutoReload**.

The options that you select affect only the current Page. TS-WAVE always starts using the default values. If you decide that you want to always use different settings, the default values can be modified in the appropriate resource file. (See [Customizing TS-WAVE](#) on page 164.)

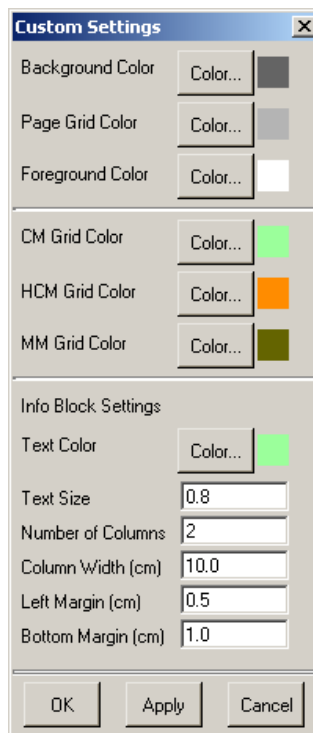


Figure 2-3 The Custom Settings dialog

Opening a Data File

TS-WAVE has two methods of opening a data file:

- by selecting a new data file via the **File=>Open Data Source** dialog
- or
- by opening a Session File that already contains data.

When you select **File=>Open Data Source** the file selection dialog appears allowing you to select a data file as shown in *Figure 2-4*.

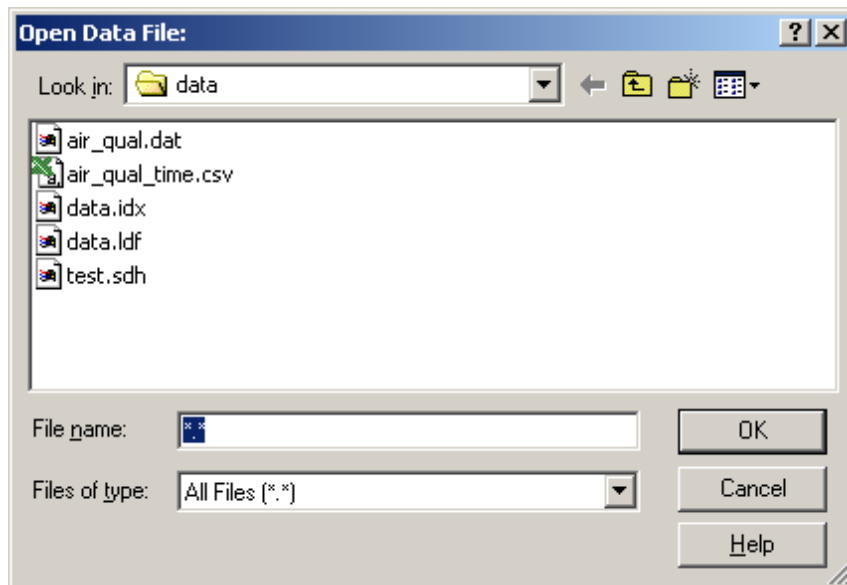


Figure 2-4 The Open Data File dialog

Once the data file is selected, the **Name and Type of Data Source** dialog appears that allows you to provide a name for this data source (called the Source ID) and identify the format of the file. (See the **Name and Type of Data Source** dialog in *Figure 2-5*.)

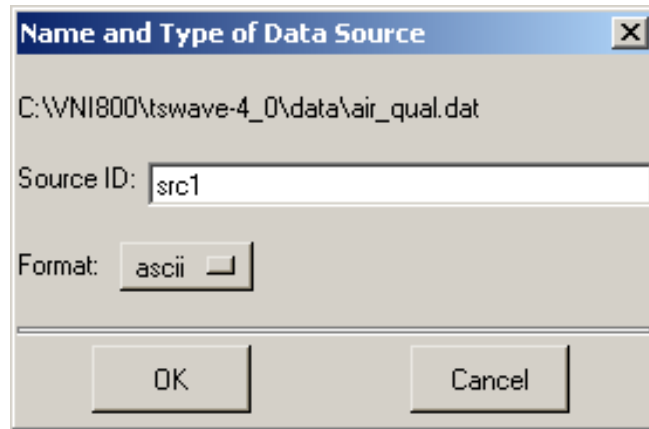


Figure 2-5 TS-WAVE's Name and Type of Data Source dialog

Source IDs are used throughout the TS-WAVE session to identify specific data sets. These data sets may originate from the same data file or they may be from different files. They may have similar parameters or may be completely different. You can modify the data set associated with a Source ID, close an open Source ID (i.e. discontinue access to a data set), or copy the data set associated with one Source ID to another Source ID. When a Source ID is modified, any TS-WAVE object using the data within that Source ID are redrawn with the new data and any parameters that were derived from its parameters are re-calculated. A series of plots can be created and then re-created with different data sets simply by modifying the data set associated with the Source ID.

When opening a new data source, the **Name and Type of Data Source** dialog ([Figure 2-5](#)) requests the file type to be identified. Data files are accessed through the use of TS-WAVE Data Handlers. TS-WAVE is shipped with five Data Handlers including Data Handlers for reading in ASCII (.dat), Loral Data Format (.ldf), and TS-WAVE Tab (.tab) formatted files. Most sites also have custom Data Handlers used to read your proprietary data types.

The user interface for a Data Handler may range from very simple with no options to more complex with many options for subsetting the data depending on the type of the file and the design of the Data Handler.

Below is an example using the ASCII Data Handler to read in the `air_qual.dat` file.

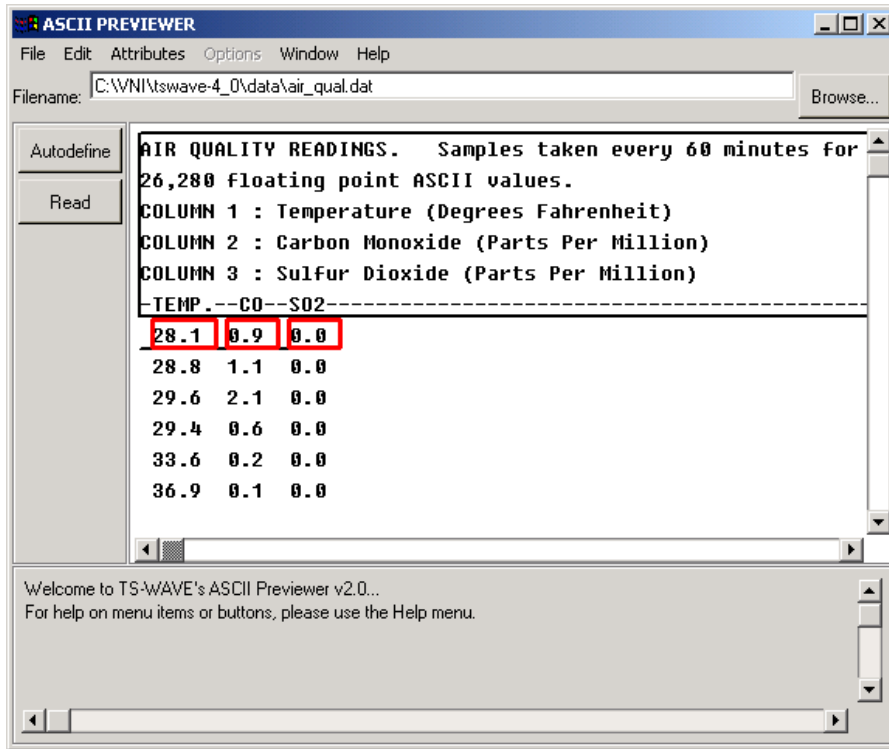


Figure 2-6 ASCII Data Handler Example

The ASCII Data Handler allows you to preview and read in your ASCII file based on what it identifies as the header of the file, the record length and the size and location of an individual field. You can override these definitions by designating the header, the record size and the field location and length. For more details on using the ASCII Data Handler, see [TS-WAVE Data Handlers](#) on page 196.

Other Data Handlers may have very different user interfaces based on the file type and attributes.

Designing your Page

With your session configured and the data loaded, you can start to layout your Page by adding objects using the **Create** menu.

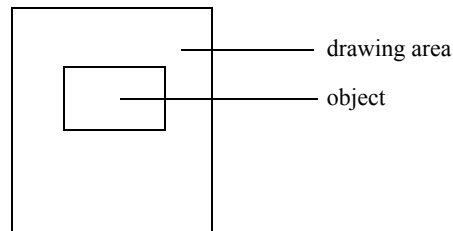
What are Objects?

Objects are the building blocks that you will use to create your data display. Available objects are Graph, Header, Contour, Text, Line, Box, and Ellipse. Each object has an attribute interface that allows you to define how you would like the object to be displayed.

Working with Objects

Creating an Object

To create an object, select an object from the **Create** menu then use your mouse to define the location of your object on the Page. Position the mouse pointer on the Page where you want the upper left-hand corner of the object to be located and hold down the left mouse button. Drag your mouse to define the object's region then release the mouse button to set the desired size as shown in the diagram below.



Selecting an Object

Individual objects are selected by choosing **Edit =>Object Select** from the main TS-WAVE menu then clicking on the object to be selected. When you are in **Object Select** mode, the **Message Area** will display 'Object Selection'. 'Handles' appear at points on the perimeter of the selected object. These handles indicate that the object is selected. When selected, the object can be resized, removed, or copied.

Resizing an Object

To resize an object, move the pointer over one of the handles, hold down the left mouse button, and drag the handle. Release the mouse button to complete the resizing. Note that when you press down the mouse button, the pointer changes to a double arrow. If the pointer does not change, be sure that the pointer is just inside the handle of an object when you press down the mouse button.

Removing an Object

To remove the selected object, select either **Edit=>Cut** or **Edit=>Delete**. The object is removed from the Page. The **Cut** function places the object in temporary storage. To paste the object back on any open Page, select **Edit=>Paste**. Had you selected **Edit=>Delete**, the object would be permanently removed.

TIP Notice on the **Edit** menu that most of the functions have keyboard shortcuts. For a complete listing, see [Using Shortcut Keys](#) on page 171.

Copying an Object

To copy a selected object, use **Edit=>Copy**. A copy of the object is placed in temporary storage. The copied object may be pasted onto any open TS-WAVE Page.

Selecting Multiple Objects

To select multiple objects, hold down the <Shift> key and click on the objects you wish to select; or, choose **Edit=>Select All** to select all objects on the Page at once; or, press and hold the left mouse button and drag the selection box around the objects you wish to select.

Deselecting an Object

To deselect an object, hold down the <Shift> key and click on a selected object. To deselect all objects, choose **Edit=>Deselect All**.

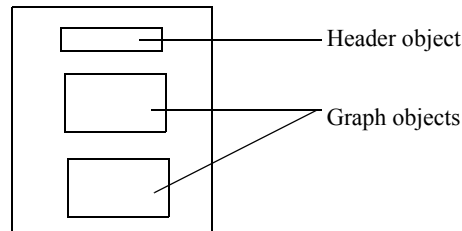
Moving a Graph Object

To move an object, make sure it is your currently selected object, then position the pointer inside the object, and hold down the left mouse button (this causes the pointer to change its shape). You can then drag the object to its new position.

Creating your Page Layout

To get started with the display of your data, you may want to layout the design of your Page before working with the data. Use the **Create** menu to place objects on your Page in the locations you would like as you get started.

For example, it is common to put a Header object on your Page followed by one or more Graph objects.



Working with Multiple Pages

TS-WAVE allows you to open and interact with multiple Pages. The first Page opens automatically upon starting TS-WAVE. Additional Pages are opened by selecting **File=>New Page** from any open Page.

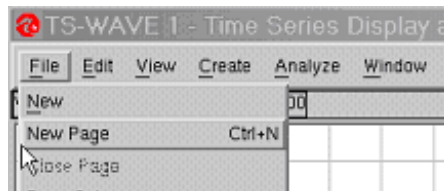


Figure 2-7 Example of File=>New Page menu selection

Each new Page appears with either the same color and page layout settings as the Page from which it was created (the default) or from your default TS-WAVE Page and layout settings depending on a resource setting. All TS-WAVE Pages share all open Source IDs. If you have two open Pages and you open or modify a Source ID in one, the new/modified Source ID becomes available immediately in the second Page and any Pages that are subsequently opened.

You may cut, copy and paste objects between TS-WAVE Pages. Each pasted object will appear in the same location it occupied on the Page from which it was copied.

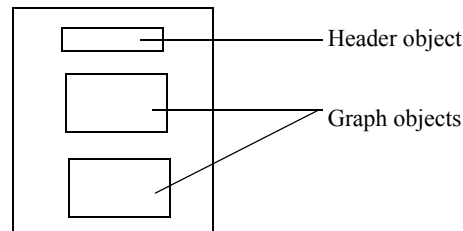
TIP Multiple pages allows you to lay out all of your plots on a main Page and use a second Page as a scratch area by cutting and pasting plots between them. Changes made to a plot in this second Page will not cause the plots on the main Page to redraw. This can substantially speed up development time when you are creating many plots with large data sets.

Changes made to an open Page's color, page layout, grid settings, etc. affect only the Page in which the changes are made, as do certain other menu options like **View=>Set Auto Scale**.

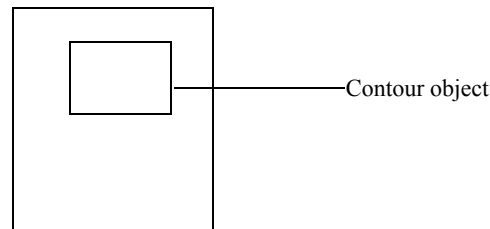
Displaying Your Data

To demonstrate the techniques used to display your data, we will build two Pages, one with a Header object and two Graph objects, and the other with a Contour object. The data file `air_qual.dat` that was read in earlier will be used for the examples.

We will use the following layout for the first Page:



with a single Contour object and some annotations on the second Page.



Adding a Header to Your Page

Text can be added to your Page by simply adding a Text object containing your text string or by using the more complex Header Object, which allows the creation of headers or subheaders and the ability to embed simple statistics from your data.

To use a Text object, select **Create =>Text Object** then click on the desired location for the text. This brings up the **Text Attributes Interface** where you can enter your text string and modify your text color, background color, font, style, size and thickness and the angle at which the text is displayed.

To use a Header object, select **Create =>Header Object** then place the desired object on your Page by selecting the location of the upper left corner of your Header Object with your left mouse button, dragging the mouse to the desired size

then releasing the mouse, Select and double click anywhere inside your Header object to bring up the **Header Attributes Interface** as shown in [Figure 2-8](#).

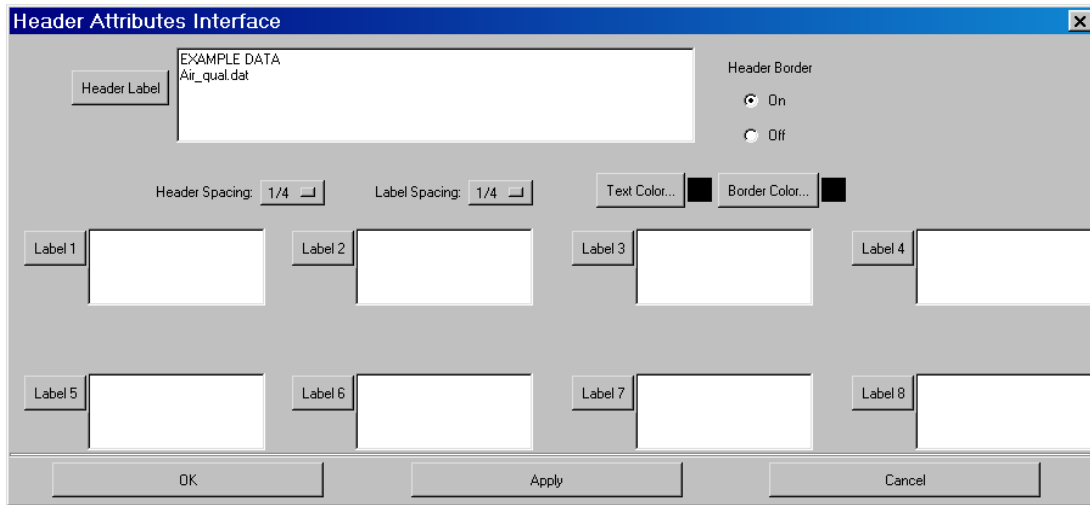


Figure 2-8 The Header Attributes Interface

You can enter your text either by typing directly into the text boxes on the **Header Attributes Interface** or by clicking on the **Label** button next to each text box and use the **Advanced Label Editing Interface** to select your formatting, print size and text.

A powerful feature of the Header object is the ability to add embedded simple statistical functions into your header text using PV-WAVE language commands. The resulting calculation of the specified function will be printed with your text on the Page. The format used to embed a function is:

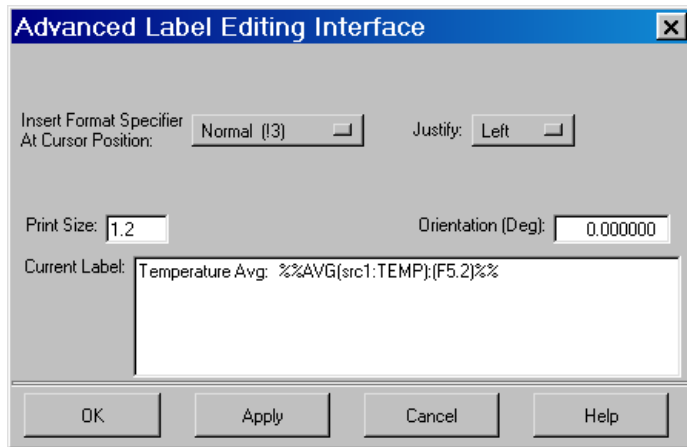
```
%%function_name(Source ID:parameter_name):(Format)%%
```

For example:

```
%%AVG(src1:TEMP):(F5.2)%%
```

The format specifier is optional.

After adding and entering the appropriate text in the interface it looks like:



The resulting Page is shown in *Figure 2-9*.

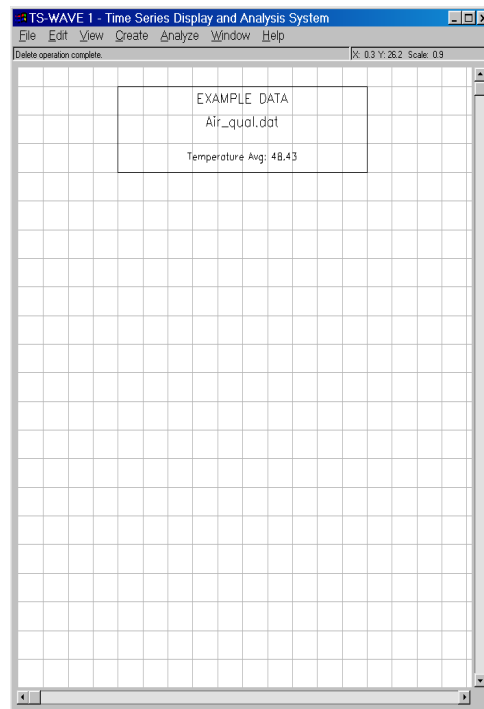


Figure 2-9 Example of a Page with an embedded function label in the Header object

For more details on creating and modifying Header objects, see the [Create Menu](#) on page 111.

Adding Graph Objects to Your Page

To add a Graph object to your Page, select **Create=>Graph Object** and position the Graph object on the Page. Once in Graph object mode, you can add multiple Graph objects to your Page. For the example, we will place two Graph objects on the first Page. (See [Figure 2-13](#) on page 29.)

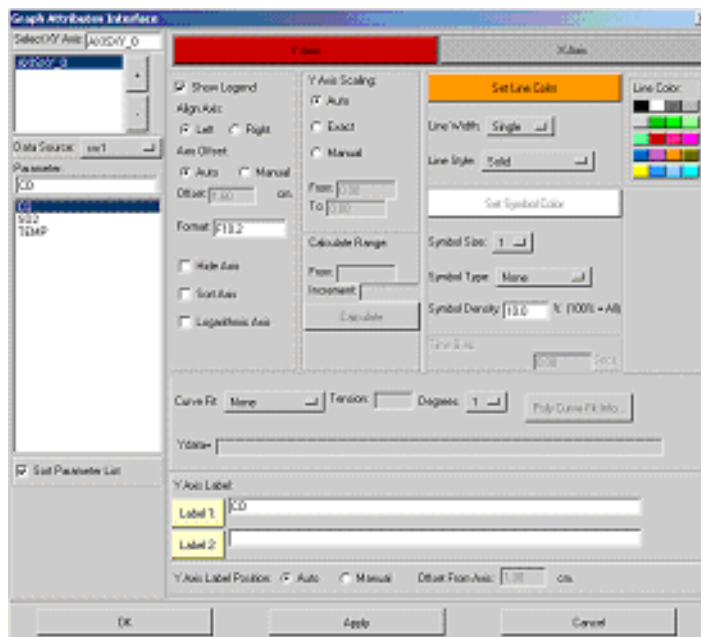


Figure 2-10 The Graph Attributes Interface

Select and double click on the Graph object to open the **Graph Attributes Interface**. Through this interface, you can add parameters and define the properties of your plot. When the interface first opens, it displays the attributes for the Y axis. The highlighted button near the top of the interface indicates the current axis.

Selecting the Data Parameters to Plot

The left side of the **Graph Attributes Interface** consists of parameter and axis information. When the interface is first opened, `AXISXY_0` is highlighted in the Axis List. To choose your parameter to plot, select the Source ID that has the desired parameter from the drop down **Data Source** menu. A list of the parameters available in that Source ID is displayed in the Parameter List. Click on the parameter name to select the parameter you want to plot. In [Figure 2-11](#) below, the parameter to be plotted CO(Carbon Monoxide) from Source ID `src1` is assigned to axis `AXISXY_0`.

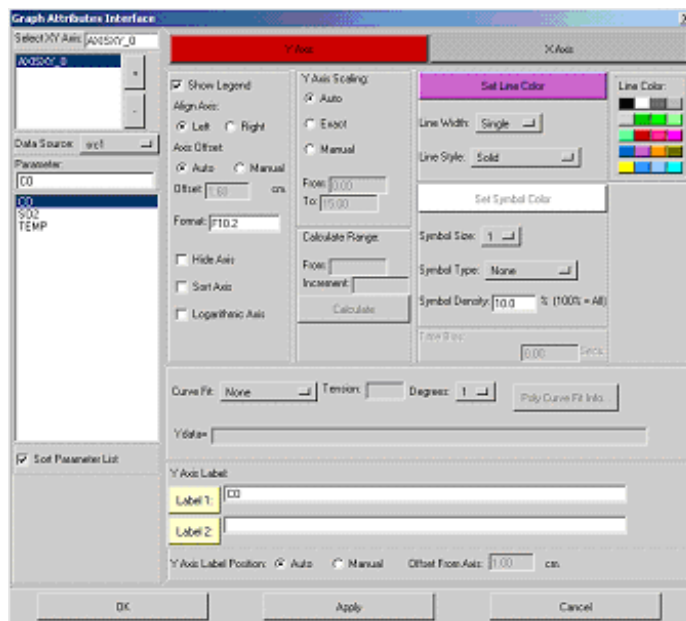


Figure 2-11 The Graph Attributes Interface with a parameter selected

Double clicking on a parameter name in the Parameter List brings up the statistical information about the data as shown below in the [Figure 2-12](#).

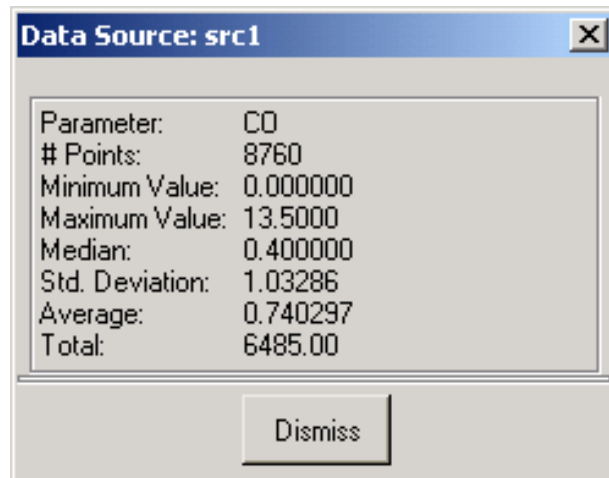


Figure 2-12 Example of Data Source dialog with parameter information displayed

After clicking on **OK** or **Apply** in the **Graph Attributes Interface**, the resulting Page displays the data for the chosen parameter, CO in this example, in the selected Graph object and will use the default settings for all the graph attributes.

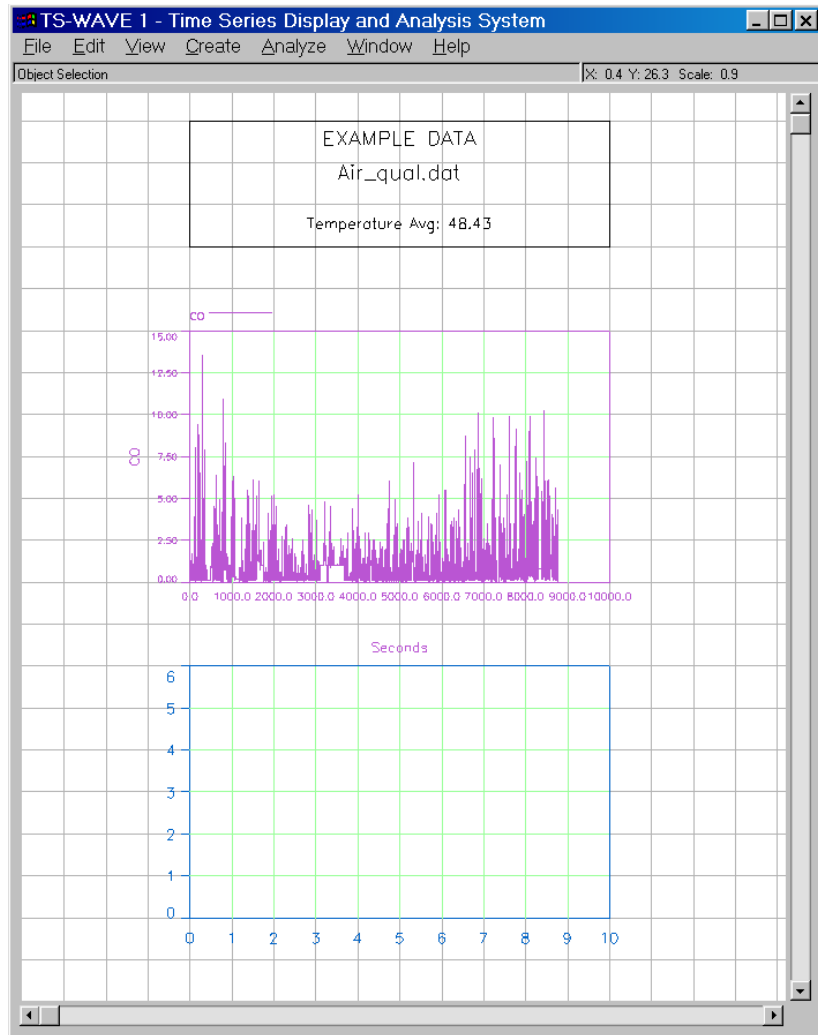


Figure 2-13 Page with a Header object and two Graph objects

Customizing Your Graph Object

It is likely that you will want to modify the attributes of your Graph object. This is done through the **Graph Attributes Interface** by selecting the **X axis** or **Y axis** button on the top of the interface. The appropriate attribute controls will then be displayed.

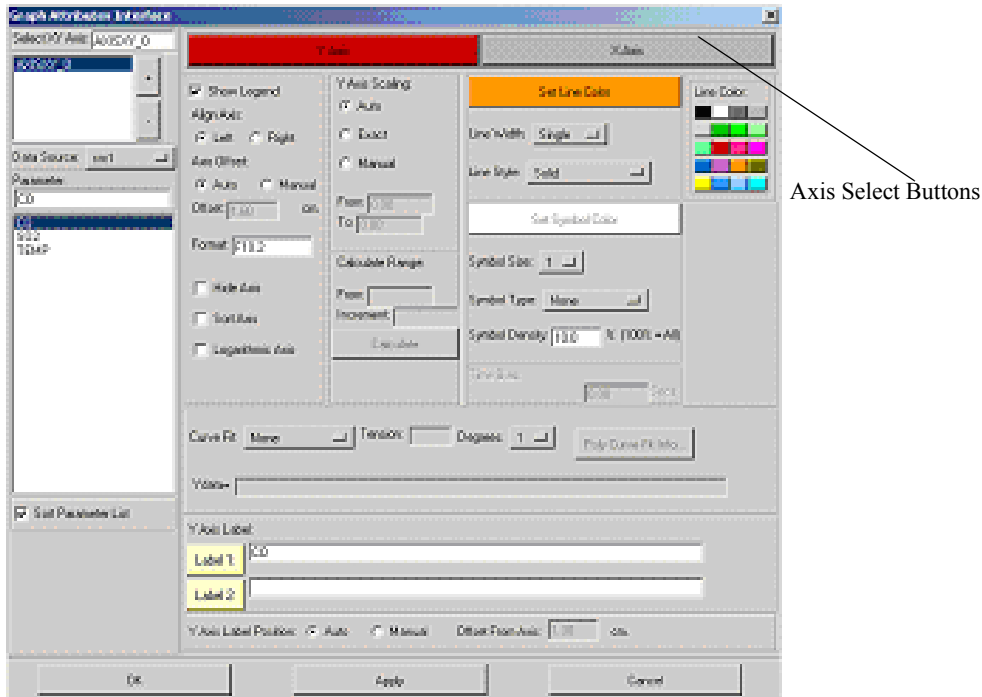


Figure 2-14 Diagram pointing to axis select buttons

Customizing Your Y Axis and Overall Graph Appearance

The **Y axis Attributes** Interface contains the controls for modifying the appearance of your Y axis as well as the general appearance of your plot.

Controls allow you to display a legend, place each axis on the right or left side of the Graph object, explicitly set the offset from the Graph object for each axis, set the formatting of the labels on the tick marks, hide the axis or set the axis to use a logarithmic scale.

The Y axis Scaling determines the axis range and can be set to **Auto**, **Exact** or **Manual**. Auto scaling means that the end points of the axis are based on the minimum and maximum data values, rounded in order to obtain even tick

increments on the axis. Setting **Exact** forces the axis to display the exact minimum and maximum values for the data as the axis endpoints. **Manual** allows the user to explicitly define the axis endpoints. You can calculate the **Manual** range based on a starting value and a given increment entered in the calculate range component and click on the calculate range button

By default, axis label 1 is the name of the chosen parameter. This can be modified by entering a new label string in the Y axis Label text box. You can also click on the label buttons to bring up the **Advanced Label Editing** Interface. The location for the axis label is set automatically, by default, but can be positioned differently by setting the **Y Axis Label Position** to **Manual** and specifying the offset in centimeters. You can also use this dialog to define a plot symbol to display at each data point.

The **Curve Fit** menu allows you to pass your data through a curve fitting algorithm before it is plotted. With the exception of the User Equation, these are predefined PV-WAVE algorithms. The options include various methods for fitting an n -degree polynomial curve through a set of data points using the least-squares method and a cubic spline interpolation algorithm. A User Equation is entered directly into the **Y data =** text box provided. For example, $y+5$ and $\sin(x)$.

Detailed descriptions on each of these choices are discussed in the [Create Menu](#) on page 111 under the *Graph Attributes Interface* heading.

Customizing Your X Axis

Attribute controls associated with the X axis are available by clicking on the **XAxis** button on the top of the **Graph Attributes Interface**.

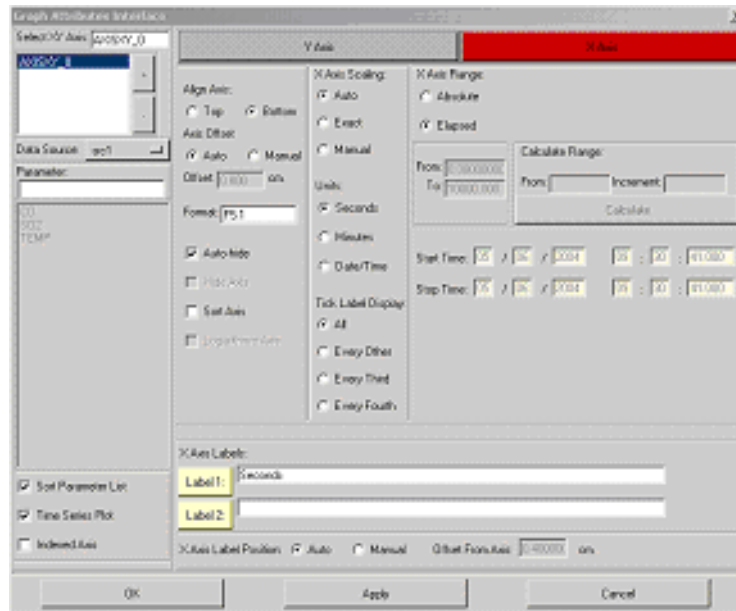


Figure 2-15 The Graph Attributes Interface with the X axis selected

There are three types of X axes: time series, cross plot and indexed. The default X axis is time series. A **Time Series Plot** uses the time component of your Y axis parameter.

A **Cross plot** graphs your Y axis parameter against a different parameter. Deselect the **Time Series Plot** toggle box then select the parameter you wish to use for your X axis. The Graph object now displays the Y axis parameter plotted against the selected X axis parameter.

Selecting **Indexed Axis** means that the value of your parameter will be plotted against its position in the data file. The n^{th} value of your Y axis parameter will be plotted at the $n - 1$ position on the X axis

To produce the image shown in [Figure 2-16](#), we also changed the **Tick Label Display** to every other label to improve readability.

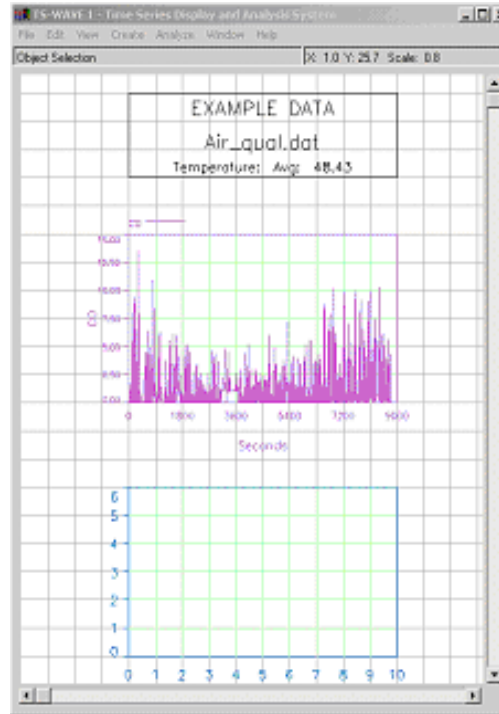


Figure 2-16 Page with modified X axis labeling

Adding a Second Parameter to Your Plot

Frequently you need to view more than one parameter on a single plot. Multiple parameters may be displayed on a single plot and each will be associated with its own X and Y axis.

To add a parameter to a plot, bring up the **Graph Attributes Interface** and select the the **Y axis** button. Click on the '+' which is located beside the Axis List on the upper left corner of the **Graph Attributes Interface**. A new axis named **AXISXY_1** (or higher number depending on the number of axes already created) will be added to the list and highlighted to indicate that it is the active axis. Select the parameter you wish to plot from the Parameter List. At this point you may also want to modify the X and Y attributes for the new axis. [Figure 2-17](#) displays the **Graph Attributes Interface** with two parameters and two axes.

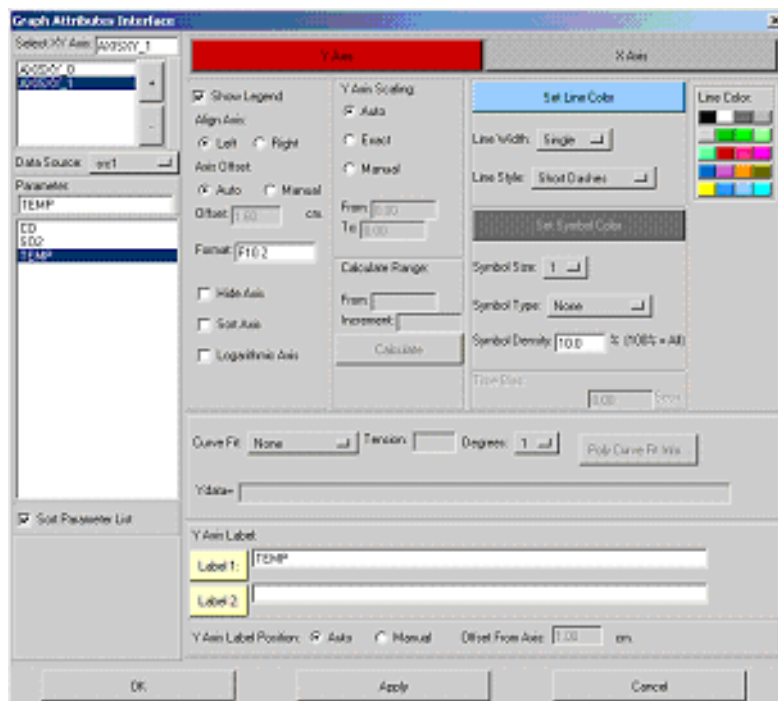


Figure 2-17 Graph Attributes Interface with two axes

The resulting plot is shown below in [Figure 2-18](#).

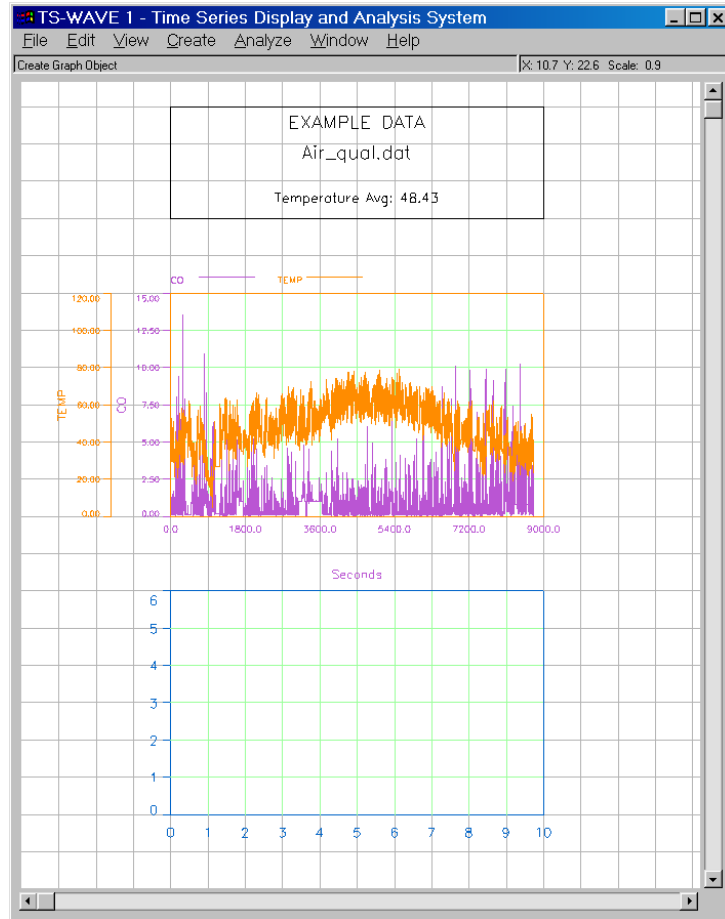


Figure 2-18 Graph with two parameters added

Going back to our previous example, modifying some of the individual attributes and plotting the SO₂ (sulfur dioxide) parameter in the second Graph object, results in the Page displayed below in [Figure 2-19](#).

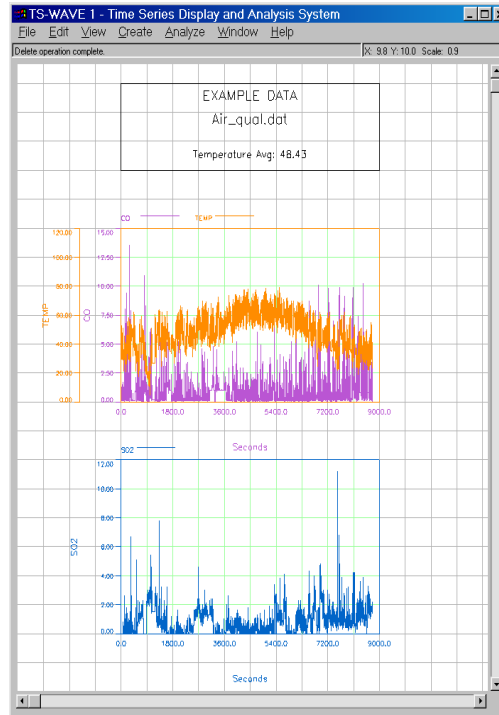


Figure 2-19 TS-WAVE Page with modifications to multiple plot attributes

Adding a Second Page

Select **File=> New Page** to create a new blank Page titled TS-WAVE 2. By default, the new Page(s) inherits all settings from the Page from which they are created.

Creating a Contour Plot

Contour plots are used to display the relationship of a dependent variable that is a function of two independent variables x and y ($z(x,y)$).

Contour objects are very similar to Graph objects and are created by selecting **Create => Contour Object**, positioning the mouse pointer on the Page where you want the upper left-hand corner of the Contour object to be located and holding the left mouse button; then dragging your mouse to define the plotting region and releasing the mouse button to set the desired contour plot area.

NOTE Remember, after an object is created, you must go to **Edit=>Object Select** to switch to Select mode to select the object and bring up the appropriate attributes dialog.

Double clicking on the selected Contour object will display the **Contour Attributes Interface** as shown in [Figure 2-20](#).

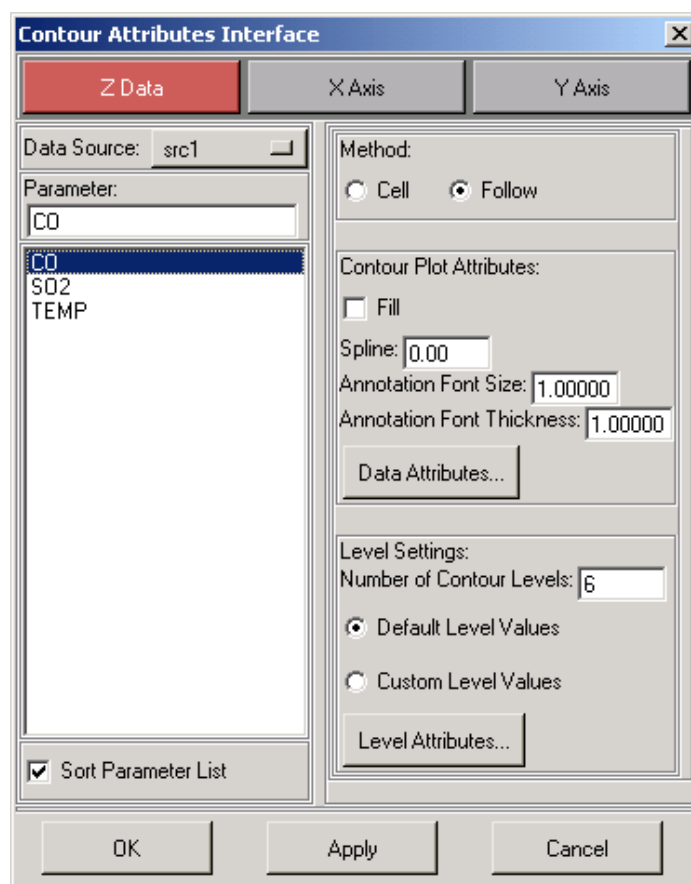


Figure 2-20 TS-WAVE's Contour Attributes Interface dialog

As with the **Graphic Attributes Interface**, the three buttons at the top of the **Contour Attributes Interface** allow you to display the attribute controls for the Z data values and the X and Y axis.

Starting with the **Z Data** interface, select the parameter name to be contoured from the Parameter List.

As with the **Graph Attributes Interface**, you can modify the overall appearance of the contour plot.

Two methods for calculating the contour lines are available. The first method, identified as the **Cell** method, examines each array cell and draws all contours emanating from that cell before proceeding to the next cell. This method is efficient in terms of computer resources, but does not allow contour labeling. The second method, listed as the **Follow** method, searches for each contour line and then follows the line until it reaches a boundary or closes. This method produces smoother lines when using dashed line styles and allows contour labeling, but requires more computing time. Although both methods draw correct contour maps, there may be slight differences between the final two plots.

Other contour plot options include the ability to fill in the area between the contours, choose the number of contour levels to display, define your own contour levels, select the level of a spline that is used in the calculation of the contour line and define the contour annotation attributes.

Selecting the **X** axis or **Y** axis buttons will display the appropriate attributes for each axis and allow you to associate one of the independent parameters with each axis.

These interfaces are shown below in [Figure 2-21](#).

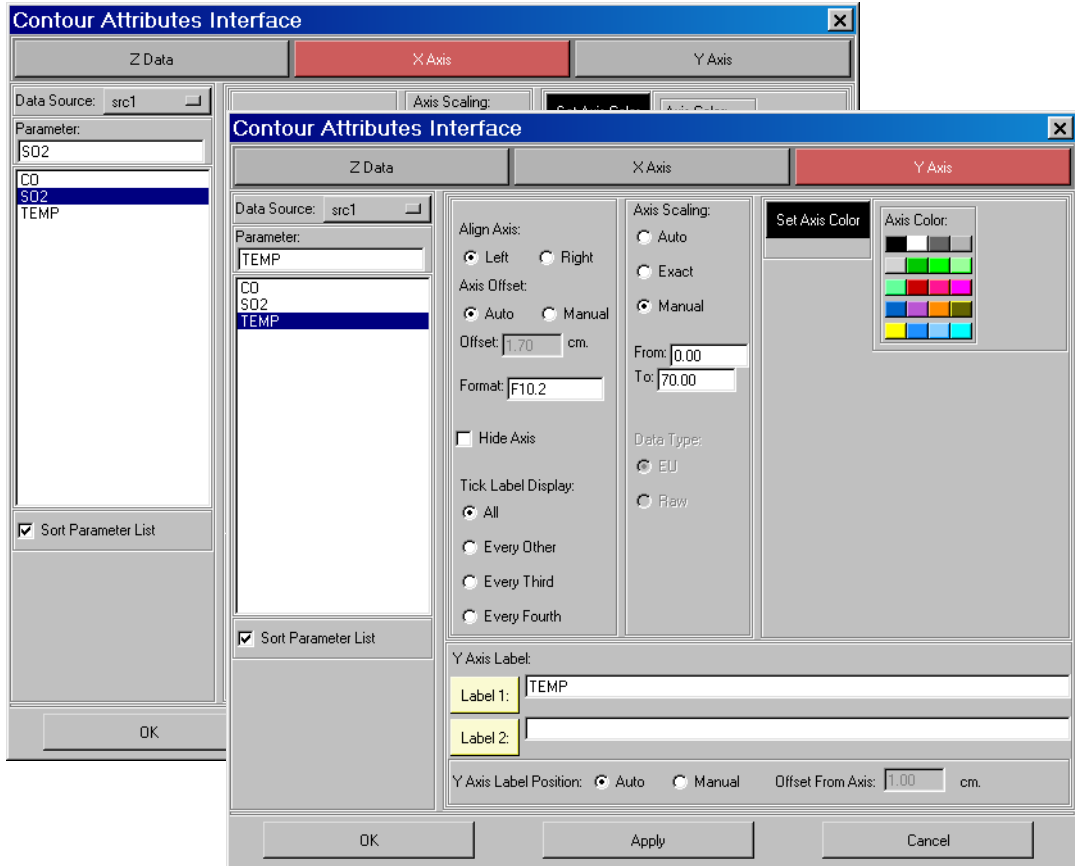


Figure 2-21 Contour Attributes Interface with X and Y axis selected

Below is the result of our example with sulfur dioxide (SO₂) and temperature (TEMP) being our independent variables and carbon monoxide (CO₂) the dependent variable.

NOTE All parameters used for a contour plot are required to have the same number of points.

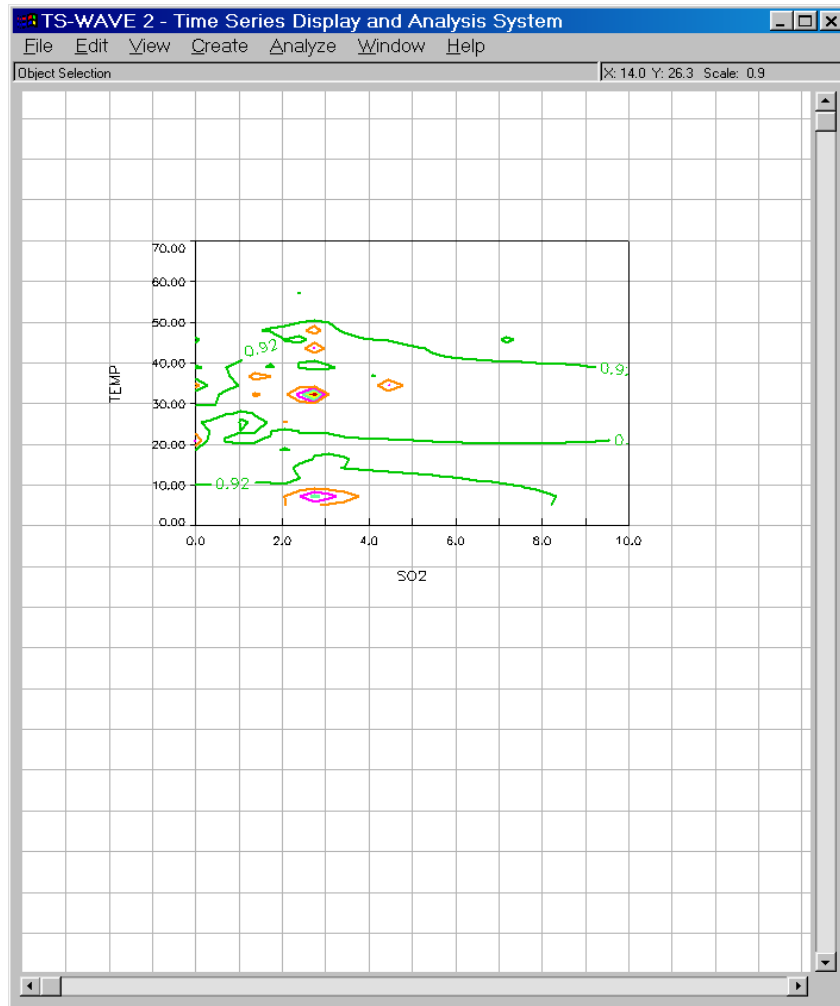


Figure 2-22 Example of independent and dependent variables plotted

Analyzing Your Data

Data analysis tools are built into almost every interface of the TS-WAVE product so that basic information regarding your data is quickly accessible. This includes everything from the curve fit routines built into the **Graph Attributes Interface** to the parameter summary pop-up windows available when you double click on a parameter in the Parameter List.

Visual Data Analysis Tools

Visual Data Analysis tools are found under the **View** menu (see page [104](#)) and include **Data Zoom** and **Examine Data Points**.

Data Zoom

The **Data Zoom** feature allows you to visually analyze your data by interactively zooming in on one or more parameters to examine your data in more detail.

To use the **Data Zoom** feature, select the Graph object you would like to zoom in on and then select **View=>DataZoom**. A new window will pop-up with the graph displayed as shown below in [Figure 2-23](#).

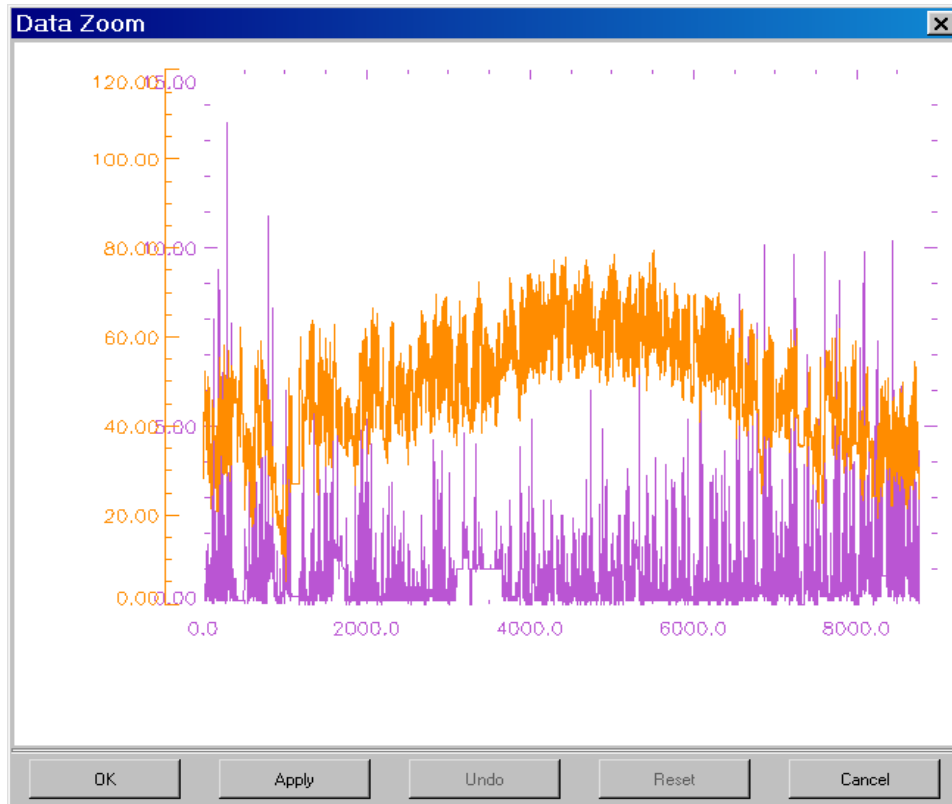


Figure 2-23 Data Zoom window

In the zoom window, click and drag your left mouse button to zoom in on a region of your data. A rubber band box will be created to indicate the area that you are selecting.

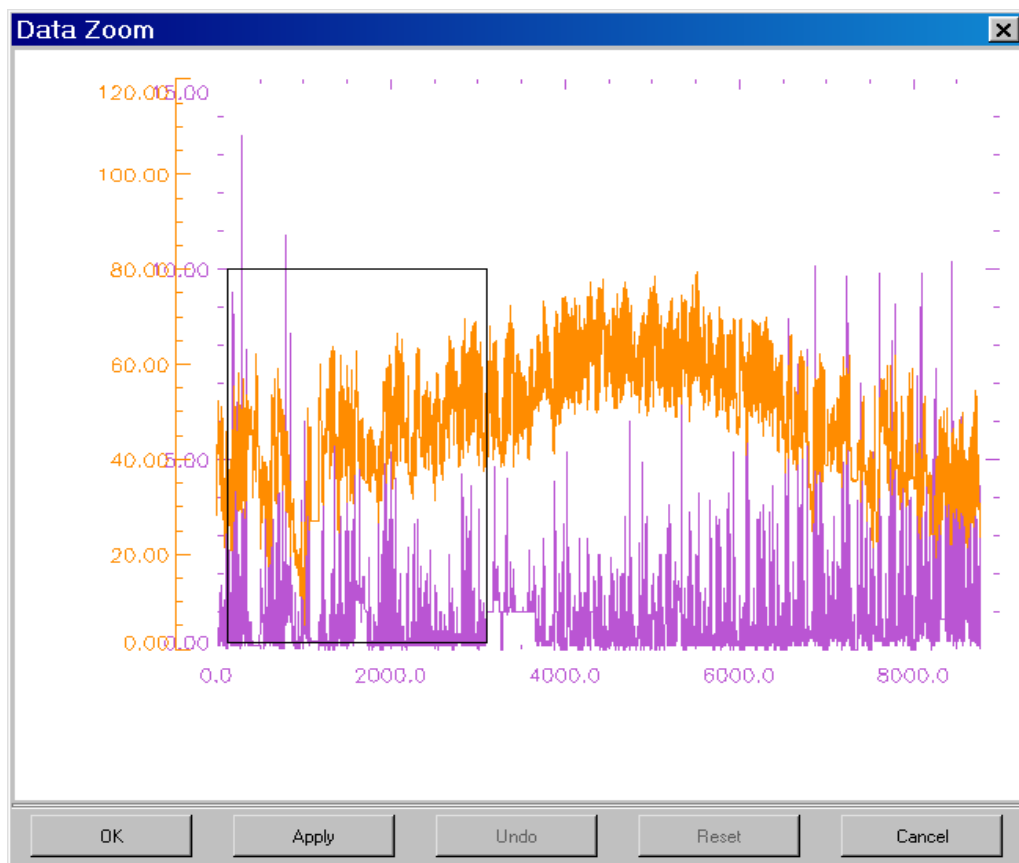


Figure 2-24 Data Zoom with area selected

Releasing your mouse button will zoom in on the data in the selected region. The X and Y axis ranges change to reflect the new graph region.

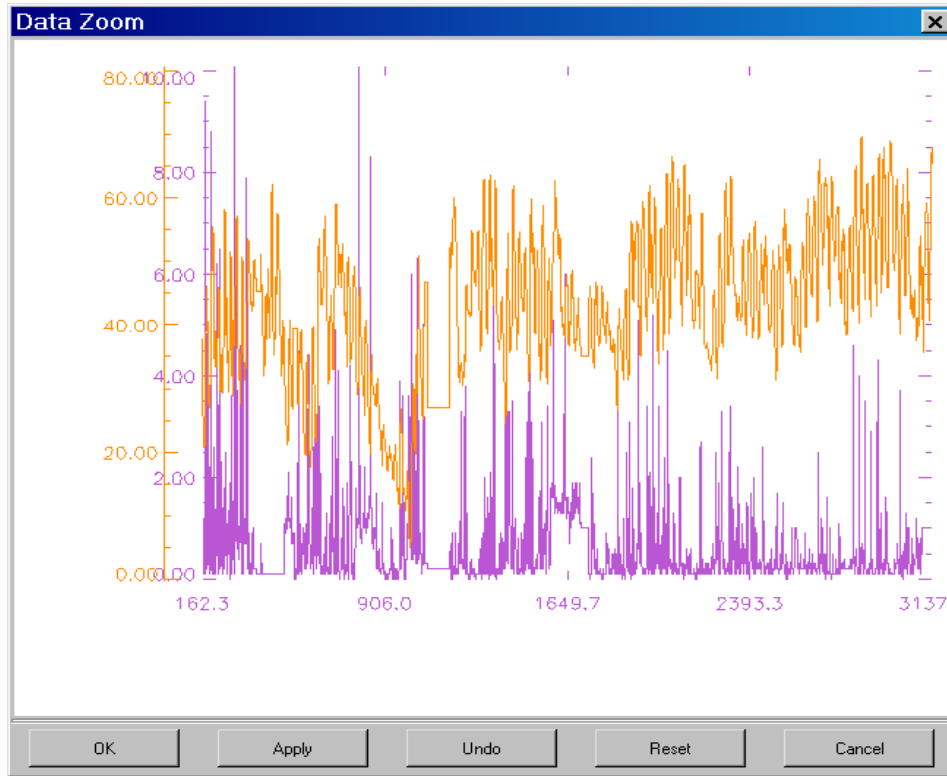


Figure 2-25 Display with Data Zoomed

The controls at the bottom of your Data Zoom window allow you to undo the last zoom, reset the plot back to the original data ranges or apply the new ranges to the graph or exit the interface without applying any changes to your original plot. The Data Zoom keyboard operations allow you to scroll in the Data Zoom window and are listed below.

Keyboard Operations on Data Zoom Window

Once the data has been zoomed, the following keys are active:

Left/Right Arrow — Pans the plot to the left or right.

Up/Down Arrow — Pans the plot up or down.

Home — Moves directly to the beginning of the original range.

End — Moves directly to the end of the original range.

The following modifier keys affect the amount by which the **Up/Down/Left/Right** keys pan within the range:

Shift plus Arrow— Allows the plot to be panned at an increased rate.

Ctrl plus Arrow — Allows the plot to be panned at a decreased rate.

Examine Data Points

Through **Examine Data Points** you can quickly retrieve the actual data values for interesting points in your data.

To use the **Examine Data Points** feature, select the Graph object you would like to examine and then select **View=>Examine Data Points**. An interactive window is displayed containing the selected Graph object. The graph has a vertical line cursor indicating the current X data coordinate. The values for each parameter will be displayed to the right of the plot while the X-value changes with the movement of the cursor.

If the selected Graph object has more than one parameter, a pop-up window appears requesting which X axis to use as shown in [Figure 2-26](#).

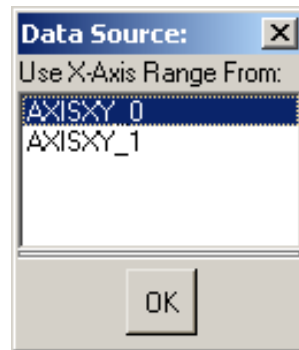


Figure 2-26 Point Data dialog showing the vertical cursor

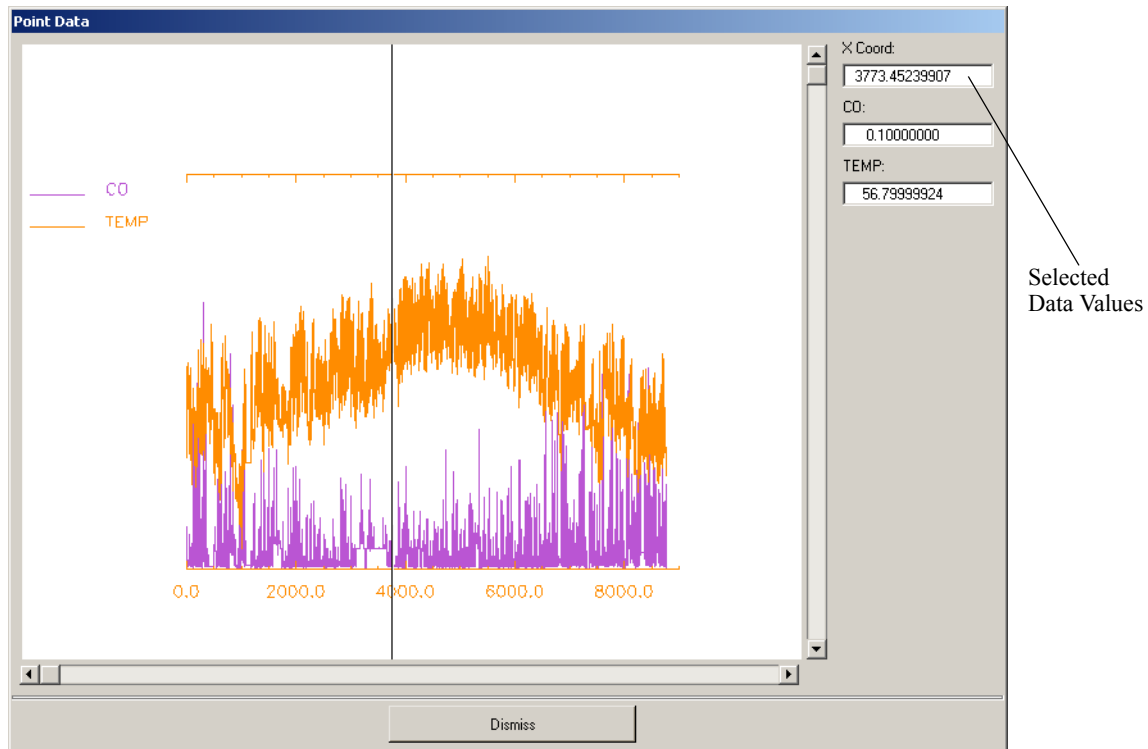


Figure 2-27 Diagram with Data Value selected

Analytical Tools

A number of analysis functions are provided in TS-WAVE. These utilities are listed under the **Analyze** menu and are separated into two categories: **Standard** and **User**. The **Standard** functions are supplied with the installation of the product and include some of the more commonly used functions. The **User** menu is provided to give you access to custom-developed functions. A few examples are included with the TS-WAVE installation. Details on creating user functions can be found in the Developer's Reference, *Writing a User Function* on page 181.

The **Standard** analysis routines create new parameters associated with the Source ID DERIVED. The default name of each new parameter is a combination of the function name, the Source ID with the parameter name and any additional arguments to the function. For example, SMOOTH(src1:CO, 7.0) as shown in *Figure 2-28*.

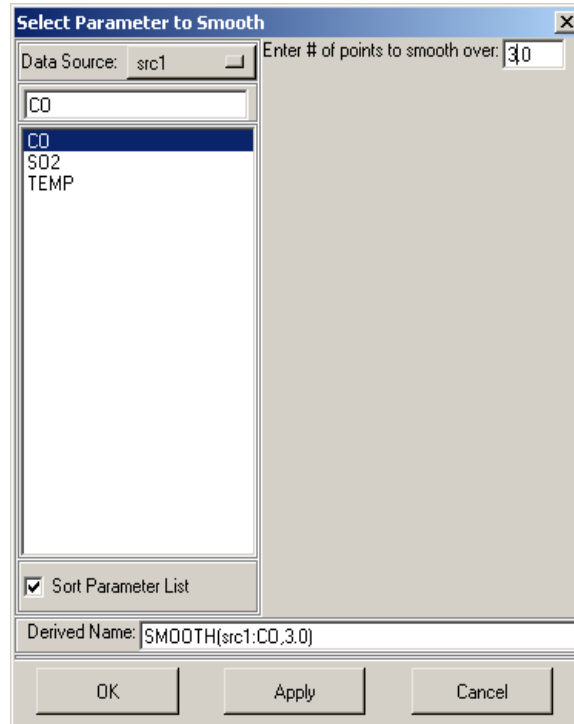


Figure 2-28 Select Parameter to Smooth dialog

The default name can be modified during the creation of the DERIVED parameter or by double clicking on the name of the DERIVED parameter in the Parameter List of any plot attributes interface and selecting **Rename**.

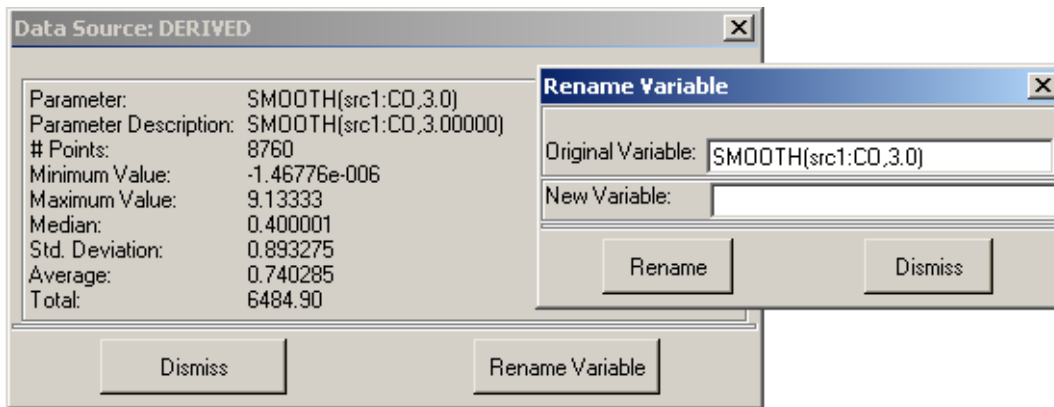


Figure 2-29 Dialogs used to rename a DERIVED parameter

Using the Standard Functions

The standard TS-WAVE functions are shown below.

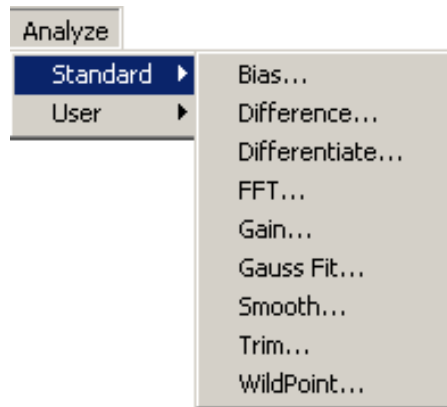


Figure 2-30 Analyze=>Standard menu

Standard Functions Descriptions

The **Analyze** menu contains built-in standard functions. These functions are used to create new derived parameters from the data from parameters in any open Source ID without changing the original parameters (see [Analyze Menu](#) on page 159).

The standard functions are:

- **Bias** — Adds a user-defined bias value to every point of a selected parameter. For more details, see **Standard=>Bias** under the *Standard Functions* section beginning on page 159.
- **Difference** — Computes the difference between any two parameters. The difference is computed by subtracting the value of the first parameter from the value of the second parameter. For more details, see **Standard=>Difference** under the *Standard Functions* section beginning on page 159.
- **Differentiate** — Computes the derivative of a specified parameter. For more details, see **Standard=>Differentiate** under the *Standard Functions* section beginning on page 159.
- **FFT** — Returns the fast Fourier transform (FFT) for the input variable. For more details on the FFT function, see *PV-WAVE User's Guide* and **Standard=>FFT** under the *Standard Functions* section beginning on page 159.
- **Gain** — Multiplies a user-defined gain value with every point of the selected parameter. For more details, see **Standard=>GAIN** under the *Standard Functions* section beginning on page 159.
- **GaussFit** — Fits a Gaussian curve through the data points of the original parameter. For more details, see **Standard=>GaussFit** under the *Standard Functions* section beginning on page 159.
- **Smooth** — Smooths the data points in the original parameter with a user-defined smooth factor. For more details, see **Standard=>Smooth** under the *Standard Functions* section beginning on page 159.
- **Trim** — Subtracts the value of the first data point of the selected parameter from each data point in that parameter. For more details, see **Standard=>Trim** under the *Standard Functions* section beginning on page 159.
- **WildPoint** — Places limits on the selected parameters data values according to user defined minimum and maximum limits . Values outside these limits are set to the appropriate minimum or maximum values. For more details, see **Standard=>Wildpoint** under the *Standard Functions* section beginning on page 159.

Working with User Functions

TS-WAVE's analytical ability can be extended with custom site-specific User Functions. These routines, written in the PV-WAVE Language, are added into TS-WAVE and invoked through the **Analyze=>User** menu.

The User Functions that are distributed with TS-WAVE are shown in [Figure 2-31](#) and are supplied both for example purposes as well as providing some handy tools. User Functions can be written to perform a specific action or to create a DERIVED parameter. User Functions can be as simple or as complex as you need and can make use of the full PV-WAVE functionality including creating graphics windows and calling external C functions.

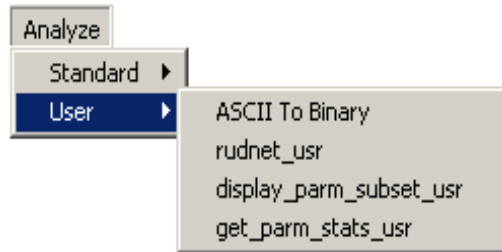


Figure 2-31 Analyze=>User menu

- **ASCII To Binary** — Converts a specifically formatted ASCII file to a general-purpose binary file. The resulting binary data can be read into TS-WAVE with increased speed over ASCII data. For more information see the **ASCII To Binary** topic under **Help->User Fcn Help** on the main TS-WAVE page.
- **rudnet_usr** — An example function that operates on two parameters and stores the result in a new derived parameter. This specifically calculates the difference between RPFLEU and RPFREU, two parameters found in the file data.ldf.
- **display_parm_subset_usr** — This is a handy interface for displaying simple statistics of a subset of a parameter using a user defined range.
- **get_parm_stats_usr** — Displays simple statistics for a selected parameter.

Exporting Your Data

TS-WAVE provides the ability to export your data to a Tab file or a site-specific file type. A Tab File is an ASCII text file that consists of a header block of information followed by tabular columns of data. [Figure 2-32](#) below displays part of a Tab File generated by TS-WAVE.

```
Created: 04/08/2004 11:43:40.000 v 4.00
C:\VNI\tswave-4_0\tab\tabfile.tab
Data Source: C:\VNI\tswave-4_0\data\data.ldf
Start: 05/17/1996 08:35:45.539 Stop: 05/17/1996 08:36:52.822
Identifier: Run=3 Sample Rate: 10.0
Title 1: A Title
Title 2: Another Title
Constant 1: A Constant Value
Constant 2: Another Constant Value
Long: Long_name_1 Long_name_2
Units: Feet Meters
SourceID: Tab_src1 Tab_src1
Param: COLFEU DPL0EU
05/17/1996 08:35:45.539 -6.1197 0.5789
05/17/1996 08:35:45.639 -7.0152 0.5802
05/17/1996 08:35:45.738 -7.3137 0.5802
05/17/1996 08:35:45.838 -7.6122 0.5802
05/17/1996 08:35:45.938 -8.8064 0.5789
05/17/1996 08:35:46.039 -9.1049 0.5815
05/17/1996 08:35:46.139 -8.2093 0.5789
05/17/1996 08:35:46.238 -6.4182 0.5802
```

Figure 2-32 Tabular Data file generated by TS-WAVE

Tab Files may contain all the data for selected parameters or only a subset of the data through Pick Files or Event Checking. Tab Files are useful for importing your data into third-party applications, incorporating the data into reports or reading back into TS-WAVE at a later date. If a custom Data Handler includes file writing, then you can access that ability through the **File=>Export** menu. For more details, see [Setting Up Your Data Handler](#) on page 199.

Working with Tab Files

To export your data to a Tab File you must open at least one Source ID then select **Create=>Create TabData**. A **Create Tab Data File** dialog appears for you to enter the name of your new data file. After entering the filename the **Select Tab Attributes** interface will be displayed as seen below in [Figure 2-33](#).

The left side of the **Select Tab Attributes** dialog displays the Source ID and parameter information for the current session. Use the **Data Source** drop-down menu to select the **Source IDs** you wish to use. The parameters available for that Source ID appear below it in the Parameter List.

To include a parameter in your output file, single click on the parameter name in the Parameter List and the Source ID, parameter name and the output format for that parameter will appear in the fields to the right as shown in [Figure 2-34](#).

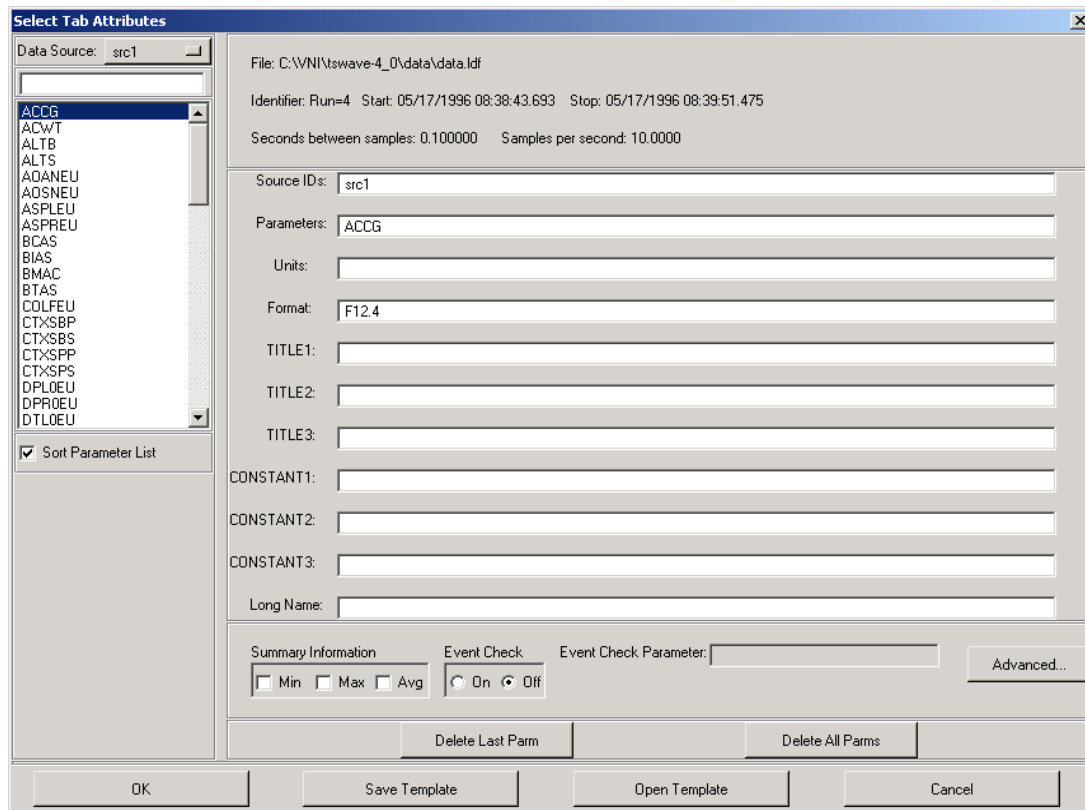


Figure 2-33 The Select Tab Attributes dialog

You may add parameters from any Source ID to the output file. The resulting output file will have your parameters grouped by Source ID, but within each Source ID data block, the parameters will appear in the order selected on the interface. Derived parameters are grouped with their originating Source ID. Duplicate selections are filtered out.

The fields on the right side of the interface define what goes into the Tab File. The only required fields are the Source IDs, Parameters and Format fields. These are automatically filled in for you as you select each parameter. The **Units**, **Title**, **Constant** and **Long Names** fields are optional. The Title and Constant entries appear in the Tab File header. The Units and Long Names entries appear above the column of data corresponding to the parameter in the same position in the Parameter field.

If you select any of the **Min**, **Max** or **Avg** check boxes at the bottom of the interface, each column of data will be followed by a summary containing the selected information.

The Event Check option allows you to subset each parameter according to the value of a different parameter, called the Event Check Parameter. The values for the output parameters will be printed only for the timestamps at which the Event Check Parameter has a value of 1.0.

The **Delete Last Parm** button deletes the last parameter in the Parameter field, along with the associated entries in the Format, Source IDs, Units and Long Names fields. The **Delete All Parms** button deletes all the selected parameters from the interface.

The **Advanced** button brings up the **Select Tab Attributes Advanced Settings** dialog. In this dialog, you can choose the number and location of the Tab File headers, the format and placement of the Date/Time output, further customize the output from the event checking option and reproduce your parameter selections for additional Source IDs.

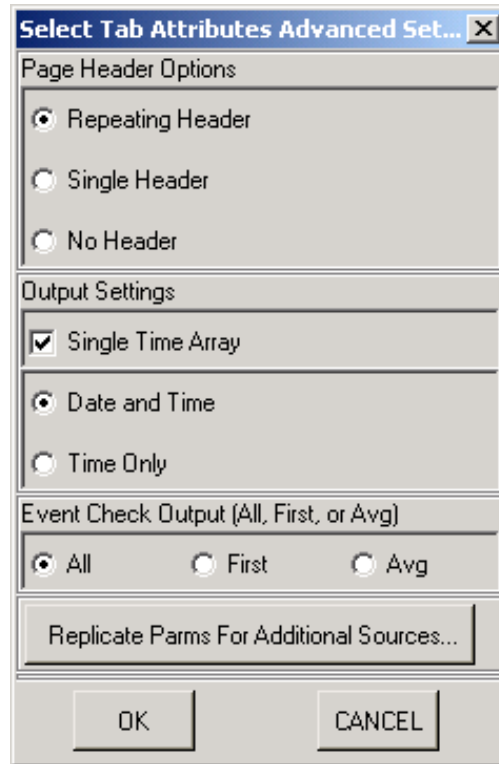


Figure 2-34 The Select Tab Attributes Advanced Setting dialog

Saving and Using Tab Templates

If you wish to reuse your Tab File attribute settings, create a Tab Template file. To do this select **Save Template** after your settings are complete. You will be prompted for a filename.

When you are ready to use a template, select **Open Template** and choose the Tab Template file you wish to use. The current settings are replaced by those in the Tab Template file.

It is possible to load a Tab Template file that contains parameters for a given Source ID that are not present in the current Source ID. When this occurs you will be asked about each missing parameter and given the opportunity to discard it and continue, or not continue with the write. You also have the option to **Discard All Missing Parameters**. If you choose this you will not be prompted about any other missing

parameters from that Source ID and the write will proceed with only those parameters that are present in the currently loaded Data Source.

Pick Files

A Pick File is a Tab File that is created by using the TS-WAVE **Pick File Interface** to graphically select a range of data values or specific data points. The selected points will then be written out in Tab File format. In order to create a Pick File, you must have a Source ID open and one or more plots created in a TS-WAVE Page.

To create a Pick File choose **Create=>Open Pick File**. A file selection dialog opens where you can specify a filename, then the **Select Tab Attributes** dialog opens. If your Graph object contains multiple parameters you will be prompted to pick a Master Axis before selecting the filename. This axis will determine the range, units and selection coordinates in the **Pick File** interface. The parameters present in the selected Graph object are already entered on the **Parameters:** line of the **Select Tab Attributes** interface. You can select additional parameters from the Source ID of the selected Graph object; these parameters will not appear in the **Pick File** interface, but will be included in the output to the Tab File.

After you have finished setting up your output file the **Pick Data** interface appears as shown below in [Figure 2-35](#).

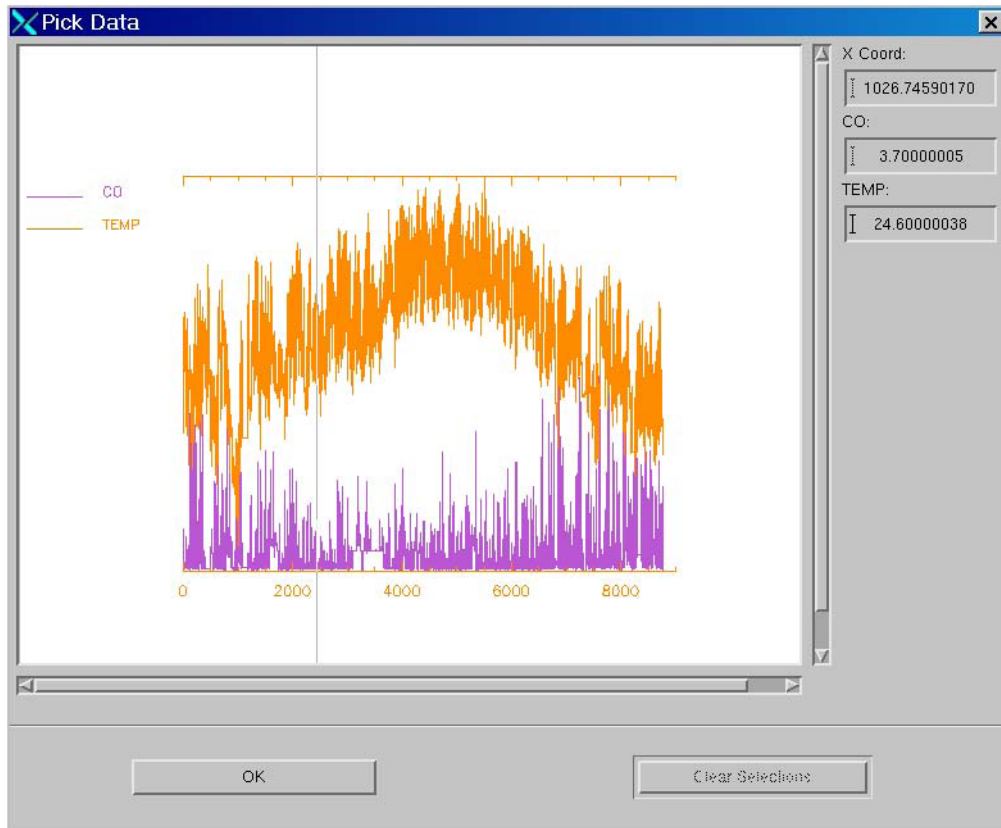


Figure 2-35 Pick Data interface with single point selected

A vertical line is displayed beneath the cursor. As you move the cursor over the plot, the coordinates of the X axis and each displayed parameter are shown on the right side of the interface. To select individual points, click and release mouse button 1 (MB1). A vertical line remains at the point you selected as shown below in [Figure 2-36](#).

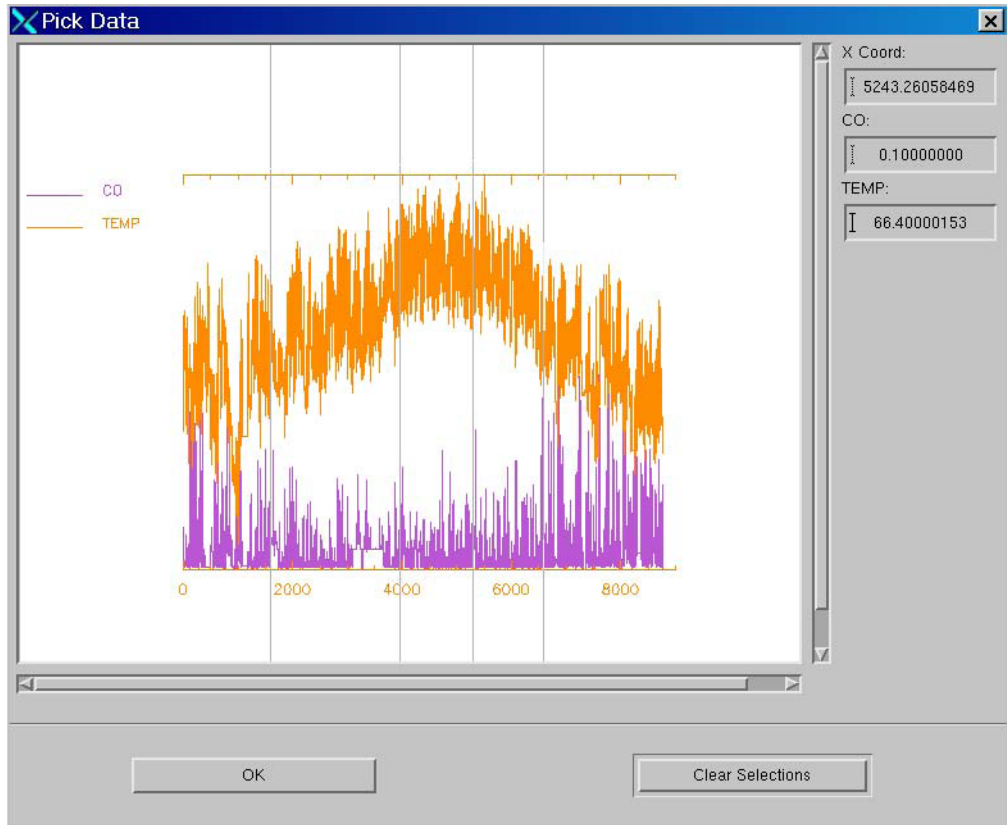


Figure 2-36 Pick Data with X values selected

To select a range of points, click and hold **MB1** and drag the cursor across the Graph object. A line appears where you first press **MB1** and another where you release it. A horizontal line will connect the endpoints of your selection region at the top of the Graph object to allow you to differentiate between multiple range selections. For an example, see the diagram below in [Figure 2-37](#).

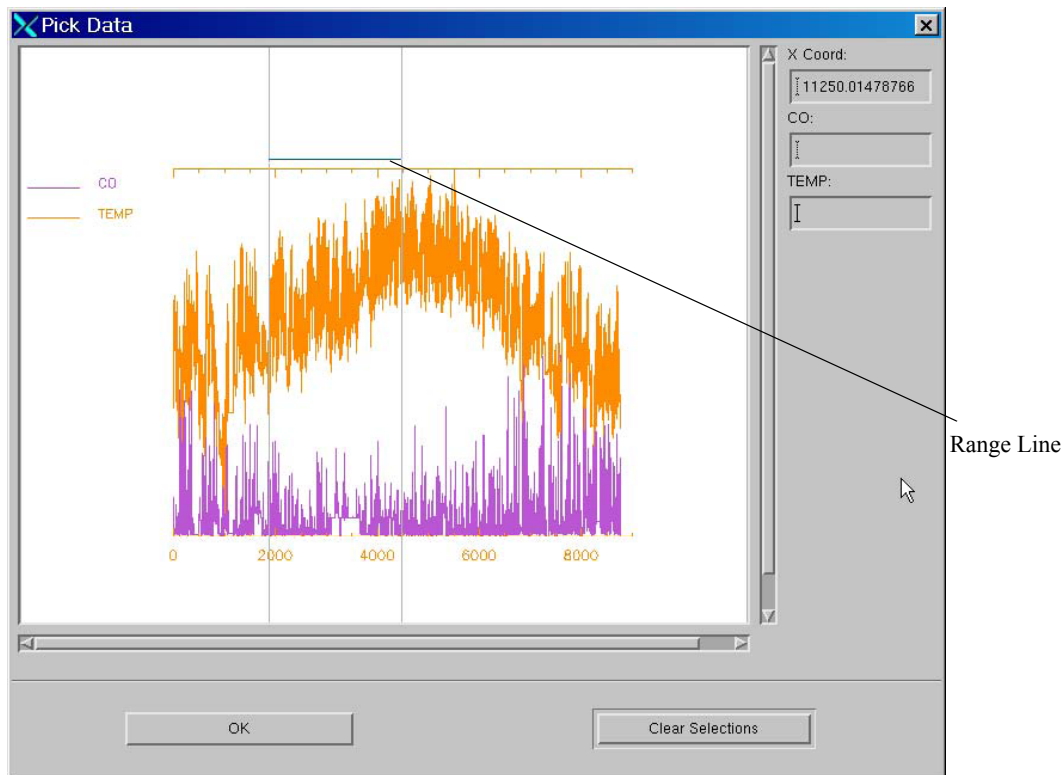


Figure 2-37 Diagram displaying a Pick File with range of data selected

NOTE You may select DERIVED parameters, but each must originate from the same Source ID as the other parameters in the selected Graph object. Parameters derived from two different Source IDs are not accepted.

Once a selection is made, the **Clear Selections** button will be enabled. Pressing this button will remove all the active selections.

To write data from a second Source ID to the same output file, select a Graph object containing only parameters from the second Source ID and select **Create=>Open Pick File**. Your new selections will be appended to the currently open output file. If you press **OK** in the **Pick Data** interface with no selections present, no data will be written for that Source ID, however, selections made on other Source IDs will still be written to the file when it is closed.

NOTE Repeat operations on the same Source ID will delete any previous selections for that Source ID.

To close your Pick File, select **Create=>Pick File** and the selected data will be written to the output Tab File.

Viewing your Pick File

Since a Pick File is a type of Tab File, you can view your Pick File from **Create=>View TabData File** in TS-WAVE or from a text editor. Pick Files use the *.tab* extension.

Saving and Using Templates

Often analysts of time-series data wish to look at the same data parameters collected over many different times. TS-WAVE lets you set up a Page layout and settings and save the entire configuration as a template that can be reused with different instances of the same parameters.

For instance, you might create a Page that contains three Graph objects that plot eight parameters. A template of this session contains the Graph objects, Text objects, Header objects, and other graphical elements, along with the selected parameter names. Multiple Pages are saved in a single Template File.

Saving Templates

To save your current TS-WAVE Pages as a Template File, select **File=>Save Template**. A standard file selection dialog appears where you can name your Template File.

By default, the Template File is saved in the `<tswave_dir>/template` directory. The extension given to Template Files is *.tpl*.

Using Your Template

To restore a previously saved Template File, select **File=>Open Template**. The **Restore** dialog appears and allows you to select a Template File. If your Template File references Source IDs the **Template Source Association** dialog displays as shown in [Figure 2-38](#).

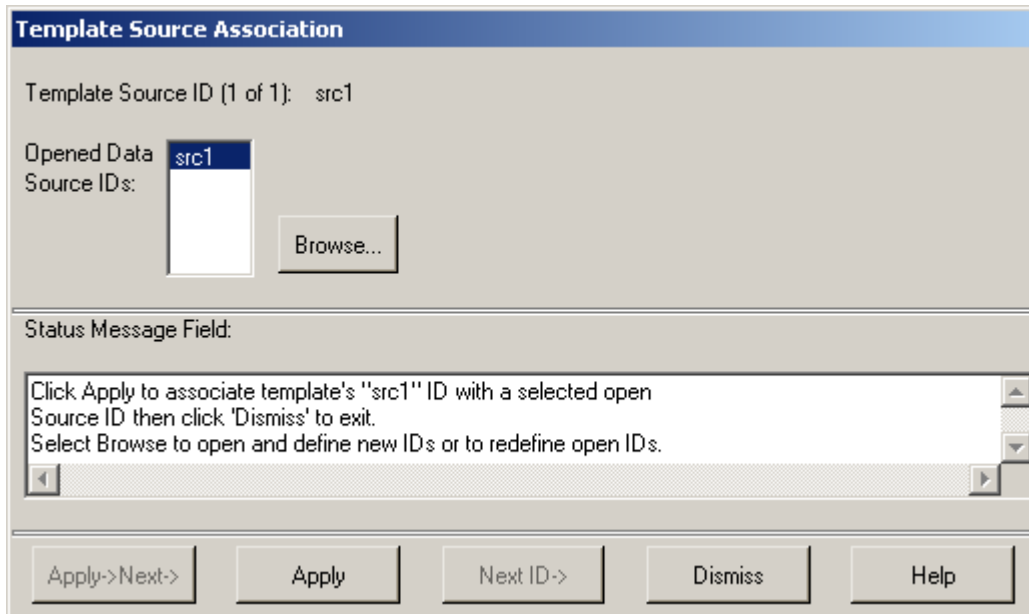


Figure 2-38 The Save Template Association dialog

The Source IDs contained in the Template File need to be associated with Source IDs in your current session. Selecting a current Source ID from the list of open Source IDs assigns the data set associated with the selected Source ID to the target template Source ID, creating it if it doesn't already exist in your session. Use the **Browse** button to add Source IDs to the list of open Source IDs.

If the template uses more than one Source ID, the **Template Source Association** dialog remains open until all Source IDs within the Template File are assigned.

Once the Template File is loaded, all of the Source IDs contained in your Template File will exist in your current session and the data sets associated with these Source IDs will be those you selected during the association process.

Saving Sessions

Session Files contain all of the information found in Template Files with the addition of the Source ID information. Use the **File=>Save Session** command to save an entire session, including the currently loaded Source IDs.

Using a Saved Session

Restore a saved session with **File=>Open Session**. The session is restored exactly as it was when it was saved. All Source IDs are reloaded from their original locations. Session Files are a good way to save work in progress.

Using Resource Files

Resource files are commonly used for customization of specific aspects of the TS-WAVE interface for personal or team preferences (such as background color or fonts used), customization of default behaviors, (such as snap-to-grid behavior and default data directories) or internationalization of the TS-WAVE interface.

Resource files are text files that contain information used by TS-WAVE to determine the default behavior for many aspects of TS-WAVE. The standard resource files are located in `<tswave_dir>/resource/` and are separated into two types of files that are identified by their extensions, `*.ad` and `*.ads`. The files with the `*.ad` extension are primarily used to modify the TS-WAVE Graphical User Interfaces (GUI) when internationalizing the TS-WAVE GUI. Files with the `*.ads` extension contain string and number resources that define information, such as default data directory, page colors, printer settings, and fonts. The `.ads` files are the files that are most commonly modified.

Information in the Resource files is stored as a Name:Value pair where the Name is the predefined attribute or characteristic in TS-WAVE and the Value is the requested behavior described by strings or numbers. For example, the default format for the X axis tick marks in the file

`<tswave_dir>/resource/ep_graphattribute.ads` is `F5.1`.

```
AXISXY_0_XTickFormat:          F5.1
```

NOTE Paths and environment variables discussed in this section are valid for both UNIX and Windows operating systems. For simplicity, all paths are shown using UNIX style separators and syntax (a ‘\$’ precedes UNIX environment variables). Windows users should translate this syntax to the Windows counterpart. For example, `$USER_RESPATH/user/` for UNIX is equivalent in Windows to `%USER_RESPATH%\user\`.

NOTE Some changes to resource files may make TS-WAVE inoperable. All changes should contain valid values for the specific resource. Resources that should never be changed are noted in the comments of the resource files that are located in `<tswave_dir>/resource/`.

Creating Your Own Private User Resources

You can define your own private resource settings whose values take precedence over the standard resources in the main TS-WAVE installation in `<tswave_dir>/resource/`. This means regardless of which TS-WAVE installation you run, your private user resource values will be substituted if they exist. User resources are especially useful in facilities where multiple users are running from one central TS-WAVE installation.

TS-WAVE can interact with Resource Files in two different ways, the behavior is determined by the location of the modified file, whether it is found in `$USER_RESPATH` or `$USER_RESPATH/user`. If this environment variable is not set, its value will default to:

Unix: `$HOME/ts_wave/`

Windows: `c:\ts_wave\`

although you may set it to any location you wish.

TIP Each Windows user sharing machines as well as common TS-WAVE installation should define `%USER-RESPATH%` as a Windows system user variable.

Modifying Only the Resources You Need

Most commonly, you will have a few resources that you find you are always modifying in TS-WAVE to match your personal preferences. Ideally you would like to change the default behavior on just these items and only for your use. To do this, create a subdirectory under `$USER_RESPATH` called “user”. Next, create a file of the same name as the Standard Resource File and save it in `$USER_RESPATH/user/`. In this file copy only the resources that you want to change.

For example, you may want to change the default background and foreground colors used when TS-WAVE is run from a shared installation without affecting other users. By default, these colors are black for background (0) and white for foreground (1). To change the default values complete the following steps:

Step 1 Create your default user resource directory:

(Windows)	<code>C:\ts_wave/resource/user\</code>
UNIX	<code>\$HOME/ts_wave/resource/user/</code>

Step 2 Locate the name of the resources that control the background and foreground colors in the `<tswave_dir>/resource/` directory and the

name of the file where they are defined. Background and foreground color resources are defined in

```
<tswave_dir>/resource/tswave_colors.ads as:
```

```
Background_Color: 0
```

```
Foreground_Color: 1
```

Step 3 Create the file:

```
$HOME/ts_wave/resource/user/tswave_colors.ads.
```

and copy only those two lines from

```
$VNI_DIR/<tswave_dir>/resource/tswave_colors.ads.
```

Then paste these two lines into this file and modify the values as you wish. This example switches background and foreground colors from their default settings:

```
Background_Color: 1
```

```
Foreground_Color: 0
```

Consequently, when you start TS-WAVE from any installation, your background color will be white and your foreground color will be black.

Because standard TS-WAVE resource files may change between releases (new resources may be added and unused resources may be removed), it is recommended that customized TS-WAVE resources be added to files in `$USER_RESPATH/user/` and should contain only the customized resource definitions. This practice makes it unnecessary to update customized resource files with each new TS-WAVE release.

Adding Other Resource Files to \$USER_RESPATH

Any resource file used by a custom Data Handler or custom TS-WAVE application can be placed in the `$USER_RESPATH/` directory. This can be useful during development for developers using a shared TS-WAVE installation. On startup, `$USER_RESPATH` is automatically appended to `$WAVE_RESPATH`, the internal TS-WAVE environment variable that defines the list of directories where TS-WAVE resource files reside.

Printing

Printing of your TS-WAVE Page is done through the **File=>Print** option. When this is selected the Print dialog (shown in [Figure 2-39](#)) is displayed allowing you to select the output format, printing queue name and select options specific to each driver.

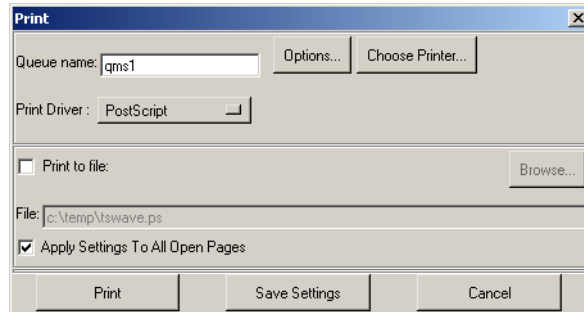


Figure 2-39 The Print dialog

The available printing formats in TS-WAVE are:

- **PostScript**
- **Computer Graphics Metafile (CGM)**
- **HP PCL**
- **HP GL**
- **Windows Metafile (Windows only) (EMF)**

The default printing information is specified in the resource file `<tswave_dir>/resource/tswave_print.ads` and includes the `Default_Printer` resource which by default sets the printer to `qms1`. You will need to change the printer name to your own printer name and you may also want to change the printing defaults for page format preferences and output format. If you wish to use Windows default printer, delete 'qms1' and leave the printer name field blank. Details on working with Resource Files are described in [Resource Files](#) on page 164.

CAUTION The `!Default_Printer` resource is case-sensitive.

Batch Processing

Batch Processing is a robust and easy-to-use, automated feature that enables TS-WAVE to run as a self-executing background process without the use of an interactive graphical user interface. A Batch Job and its related files carry instructions for automatic loading of templates, data and printer settings as well as output information. Graphical output can be sent directly to a printer or it can be saved to a file in any of the graphics formats supported by TS-WAVE. Data and data analysis results can be written to a Tab File and reviewed later. Batch Jobs are executed from a central shell script (UNIX) or Batch File (Windows) that can be scheduled to run by any automated job scheduler. This offers you the ability to run tests during the day, process the data with TS-WAVE that evening, and have the output waiting the next morning. This is especially useful for those who work with data that changes often and need the ability to automate and standardize the output, analysis and graphical presentation of their data.

For example, if you have data results that are collected daily by a separate automated process, you can schedule a TS-WAVE Batch script to be run at a specified time to read the data output by the collection process, prepare the requested displays and Tab Files and have the resulting Graph objects sent to a printer or file.

When a Batch File is run, the following steps occur:

- Step 1** A TS-WAVE session starts in background.
- Step 2** The first template is opened.
- Step 3** The template's associated data files are loaded.
- Step 4** The output is generated and sent to the printer and/or output file.
- Step 5** Steps 2, 3 and 4 are repeated until all templates in the Batch File are processed.
- Step 6** Batch mode is exited.

Creating a Batch Job

In general, the steps for creating and executing a Batch Job are:

- Step 1** Define a Source ID and design your TS-WAVE Page(s).

Step 2 Open a Batch Job by selecting **Create->Open Batch File** and name your Batch Job.

Step 3 Add your template(s) to the Batch File from **Create->Add to Batch File=>Add Template to Batch**.

Step 4 Close your Batch File by selecting **Create->Close Batch File**.

The shell script can be optionally executed via an automated job scheduler.

When the job is run, messages will be logged in the file

`<tswave_dir>/tswavebatch.log` if no other output location is specified.

When the Batch File is closed (after **Step 4**), multiple files are created with the chosen file name, but with different extensions and functions. These files include a Batch Job File (*.job), Template File or files (*.tpl), and a Batch Job Script (*.sh in UNIX, and *.bat in Windows). By default, all of these files are saved in the `<tswave_dir>/batch` directory. For information about specifying a default batch directory, see the .ads Resource discussion under [Standard TS-WAVE Resources](#) on page 165.

The Batch Job File contains the configuration parameters for the background TS-WAVE process, such as the name of your printer, the Source IDs used, the name of the Data Handler used, and the location of the Template Files. There may also be other Data Handler specific entries. An example of a *.job file can be seen on [page 152](#).

The Batch Template File is a standard TS-WAVE Template File that contains information about the drawing area configuration (Graph objects, Header objects, data, printer settings and so on). Each time you select **Create=>Add To Batch File=>Add Template To Batch** a new template is saved for the Batch Job File.

The Batch Job Script contains the TS-WAVE command to invoke TS-WAVE in batch mode with the specified Batch Job File and a log file to which messages generated from executing the Batch Job script are printed. For an example of a *.sh file, see [The Batch Shell Script \(.bat for Windows, .sh for Unix\)](#) on page 153.

Simple Batch Job Example

A simple example of a Batch File is to save the template created earlier with the `air_qual.dat` file and then rerun it as a Batch File (although in practice, this would be rerun with a second file containing different data values).

To prepare your Batch File layout, open your data source and design your page. Once this is done, check to see that the correct Queue name and Print Driver are

valid for your printer. If you make any modifications, you must click on **Save Settings** button in the **Print** interface to save the information for your Batch File.

Once you have set up your Page layout and printer you are ready to open the Batch File with select **Create=>Open Batch File**. By default, Batch Files are created in the <tswave_dir>/batch directory although a File Selection fnterface will pop-up and allow you to define the name and location of your file.

Next, select **Create=>Add To Batch File=>Add Template To Batch** to add the current Page configuration (Graph objects, Header objects, data, and so on) to the Batch File.

Lastly, select **Create=>Close Batch File** to close and save the Batch Job.

To test your Batch Job, execute the Batch Shell script, which by default is located in the <tswave_dir>/batch directory. The printed output is sent to the printer you selected.

NOTE Check <tswave_dir>/twavebatch.log file for printing status.

More details and specific examples can be found in the [User's Reference](#) beginning on page 89.

Getting Started Tutorial

This section is intended to provide a step-by-step introduction of the basic functionality to a user new to TS-WAVE. It covers, in short tutorials, all that you need to know to get yourself started using TS-WAVE and to begin producing useful results. The *User's Guide* in Chapter 2 contains more detailed information about the topics in this tutorial. If you are an experienced user of TS-WAVE, you may want to skip the tutorial.

After completing this tutorial, you will be able to:

- Define a data source
- Use a standard function to create a derived parameter
- Plot a parameter
- Print a TS-WAVE Page
- Create and view a Tab file
- Create and run a Batch file

This tutorial takes about an hour to complete.

Defining a Data Source

TS-WAVE is capable of reading a wide variety of data formats. This tutorial will use a sample data file included in your installation. The file is saved in the Loral data file format (*ldf*) and the *ldf* Data Handler is used to illustrate how data is typically imported into TS-WAVE. The sample *ldf* file consists of dozens of parameters that reflect measurements taken during five different trials organized in sections called runs.

- Step 1** If TS-WAVE is not already started, start it now. For instructions on starting TS-WAVE, see [Starting and Stopping TS-WAVE](#) on page 7.
- Step 2** Select **File=>Open Data Source**. The **Open Data File** dialog appears as shown in [Figure 3-1](#).

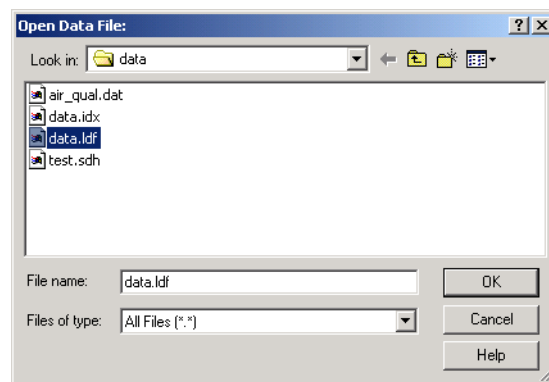


Figure 3-1 Open Data File dialog

- Step 3** Select the *data.ldf* file and click **OK**. The **Name and Type of Data Source** dialog appears as shown in [Figure 3-2](#).

The actual directory path might differ from the one shown here, depending on where you installed TS-WAVE.

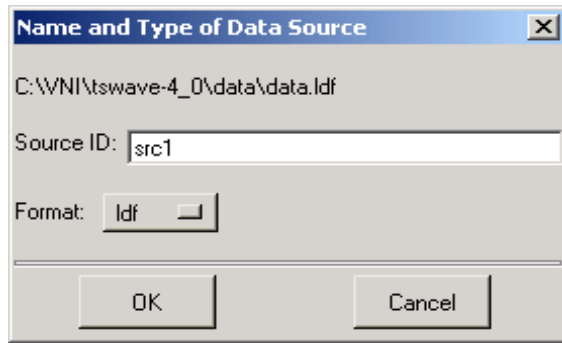


Figure 3-2 Name and Type of Data Source dialog

- Step 4** Notice the default Source ID is *src1*, and the file format type is *ldf*, click **OK**.
The **Select Data Run** dialog appears, as shown in [Figure 3-3](#). The **Select Data Run** dialog is specific to the ldf Data Handler and allows the user to select data based on features unique to the ldf format. Other TS-WAVE Data Handlers have different dialogs for controlling the selection of data.

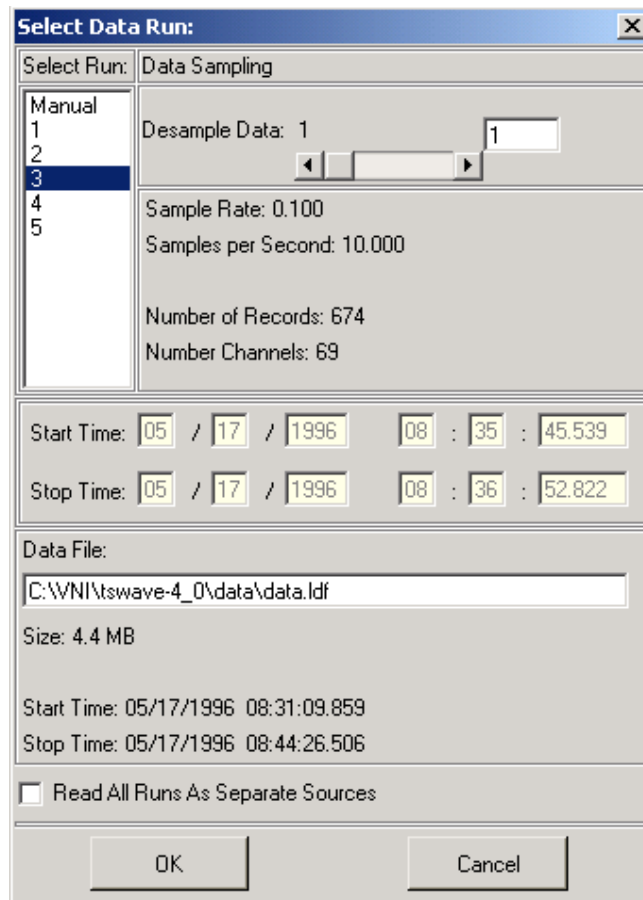


Figure 3-3 The Select Data Run dialog is used by the ldf Data Handler to select the run, or runs, to load from the data file.

Step 5 Click on run number **3** to select it.

TIP To read the entire file of test data at once, click on **Manual**. With **Manual** selected, you can edit the **Start Time** and **Stop Time** text fields to read specific time intervals.

NOTE The top section of the dialog shows information about the size of the selected data and the sampling rate. Once Run 3 is selected, **Start Time** and **Stop Time** are grayed out, however the time range for the selected run is displayed. The

bottom section of the dialog displays information about the data file: the name, the size, and the Start/Stop times for the entire file.

Step 6 Click **OK** to dismiss the dialog and return to the main TS-WAVE Page.

At this point, the Source ID `src1` is defined to be all the data for run number 3 in the selected data file, although, the `ldf` Data Handler does not actually read the data into TS-WAVE until it is accessed elsewhere.

Data Analysis Using the Standard Function: Smooth

TS-WAVE offers several ways to analyze your data both visually and analytically. The analytical capabilities are available in the **Analyze** menu.

TS-WAVE comes with built-in standard functions (see [Using the Standard Functions](#) on page 48) or you may use your own user functions ([Working with User Functions](#) on page 49).

Before creating a Graph object, we are

going to create a new parameter that contains a smoothed version of the data, which we will graph on top of the original data.

First, let's create the new parameter. This is done using the Standard User Function 'Smooth' to create a DERIVED parameter. For a detailed discussion about derived parameters, see [Creating a Derived Parameter](#) on page 183 of this manual.

Step 1 Select **Analyze=>Standard=>Smooth**. The **Select Parameter to Smooth** dialog appears as shown in [Figure 3-4](#).

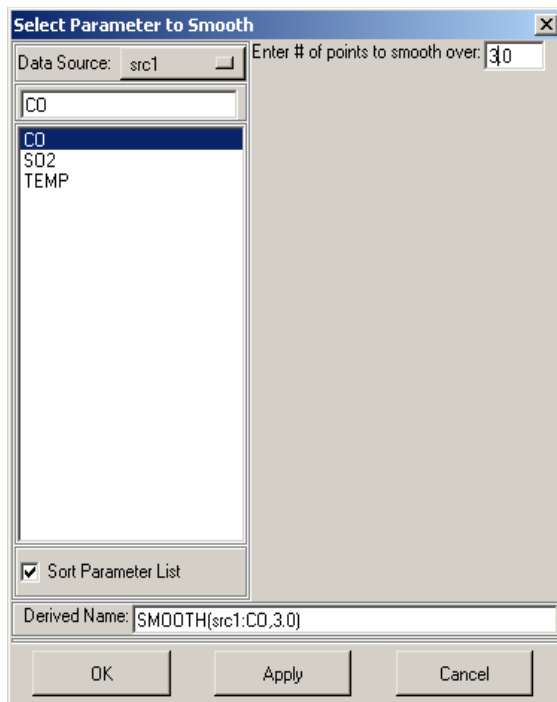


Figure 3-4 Select Parameter Smooth dialog

Step 2 Change the **# of points to smooth over** from **5.0** to **50**.

Step 3 Select parameter **COLFEU** from the parameter list. The result of this operation is a new parameter; the original parameter is unchanged. The new derived parameter is named **SMOOTH(src1:COLFEU, 50)**.

Step 4 Click **OK** to dismiss the **Select Parameter to Smooth** dialog.

We will now view the original data and the results of the Smooth analysis.

Plotting Parameters

TS-WAVE data is plotted on 2D graph objects, where the X axis by default is the Time axis, and the Y axis is the axis on which data parameters are plotted.

Customizing the TS-WAVE Page

First, let's set the background color and foreground colors for the TS-WAVE Page.

Step 1 Select **View => Custom Settings**. The **Custom Settings** dialog appears.

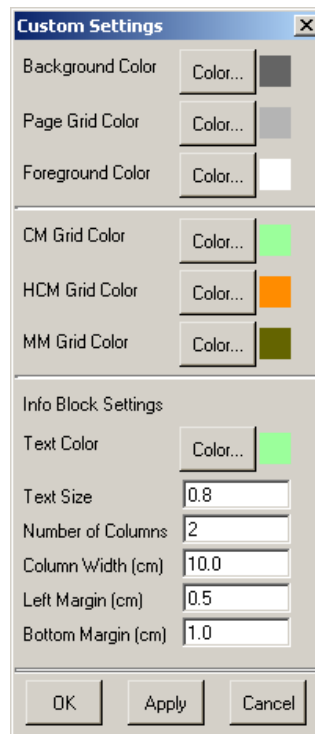


Figure 3-5 The Custom Settings dialog

Step 2 Click on the **Color** button next to the Background Color label. The **Background Color** dialog appears.

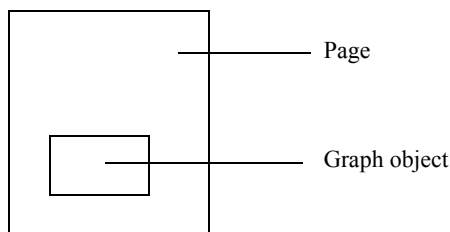
Step 3 Select the white block and click **Dismiss** to dismiss the dialog.

- Step 4** Click on the **Color** button next to the Page Grid Color label. The **Page Grid Color** dialog appears.
- Step 5** Select the light grey block and click **Dismiss** to dismiss the dialog.
- Step 6** Click on the **Color** button next to the Foreground Color label. The **Foreground Color** dialog appears.
- Step 7** Select the black block and click **Dismiss** to dismiss the dialog.
- Step 8** Click **OK** to dismiss the **Custom Settings** dialog. The Page is updated with the new color settings.

Creating a Graph Object

- Step 1** Select **Create=>Graph Object**.
- Step 2** Position the mouse pointer on the Page where you want the upper left-hand corner of the Graph object to be located. Then, hold down the left mouse button while dragging the mouse pointer to define the position of the lower-right corner of the Graph object. When you release the mouse button, the Graph object is displayed on the Page. The schematic below shows approximately where to position the Graph object on the Page.

After creating your Graph object, you will want to plot your data, which is described in the next section.



Graphing Data Parameters

You have already created a Source ID, a derived parameter, and a Graph object, now it is time to plot your data.

- Step 1** Select **Edit=>Object Select**.
- Step 2** Double click inside the Graph object. The **Graph Attributes Interface** appears, as shown in *Figure 3-6*.

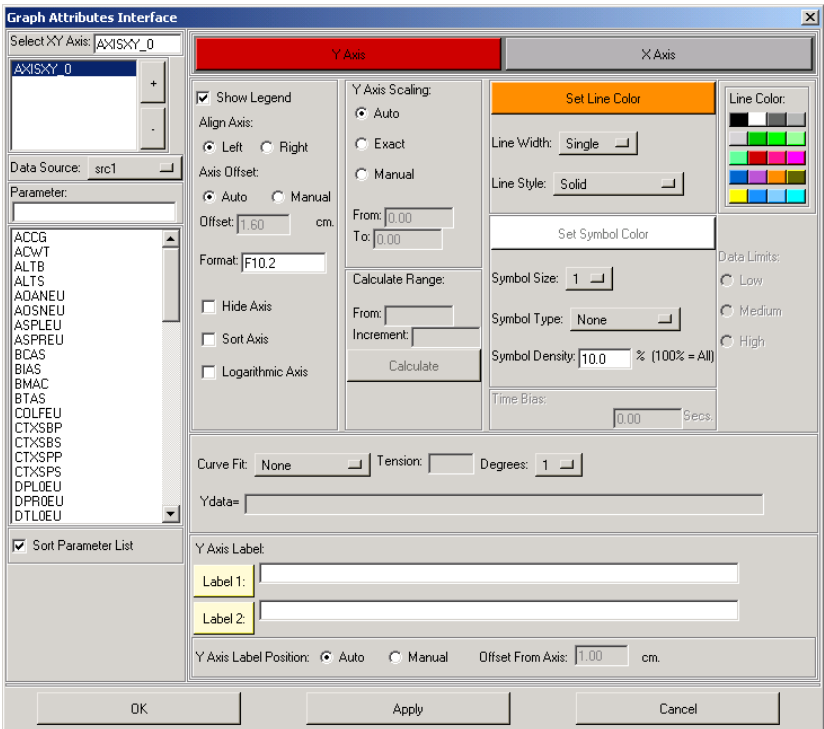


Figure 3-6 The Graph Attributes Interface

- Step 3** In the list of parameters, locate the **COLFEU** parameter and click on it. Notice the parameter name is now highlighted in the Parameter name field.
- Step 4** Double click on the **COLFEU** parameter. Double clicking on the variable name displays information about the variable including the: number of points, minimum and maximum value, median, standard deviation, average and total.

- Step 5** Click **Apply**. Notice the values in the grayed out **From:** and **To:** boxes under the Y axis Scaling radio buttons. Write down or remember these values, as you will use them in **Step 10**.
- Step 6** Near the top left of this interface, locate the small square button that has a '+' (plus symbol) in the center of the button. Click on this button to add a second **XY Axis** button. Notice **AXISXY_1** appears highlighted in the axis list.
- Step 7** Click the **Data Source** drop-down list.
- Step 8** Select **DERIVED** as your Source ID. Notice the derived parameter that you created earlier appears in the Parameter List.
- Step 9** Let's associate the new derived **Smooth** analysis parameter with the axis you created in the **Step 4**. Select **SMOOTH(src1:COLFEU, 50)** from the Parameter List. The derived parameter is highlighted.
- Step 10** Let's change the attributes of the **AXISXY_1** axis. Select **Manual** for the Y axis Scaling and enter the **From:** and **To:** values from **Step 5**. The Y axis Scaling for both of the Y axes is now the same.
- Step 11** Now we will change the appearance of the line. Select **Triple** from the **Line Width:** drop down list and **Solid** from the **Line Style:** drop down list.
- Step 12** Click **OK**. The dialog closes, and both parameters are plotted in the Graph object as shown in [Figure 3-7](#).

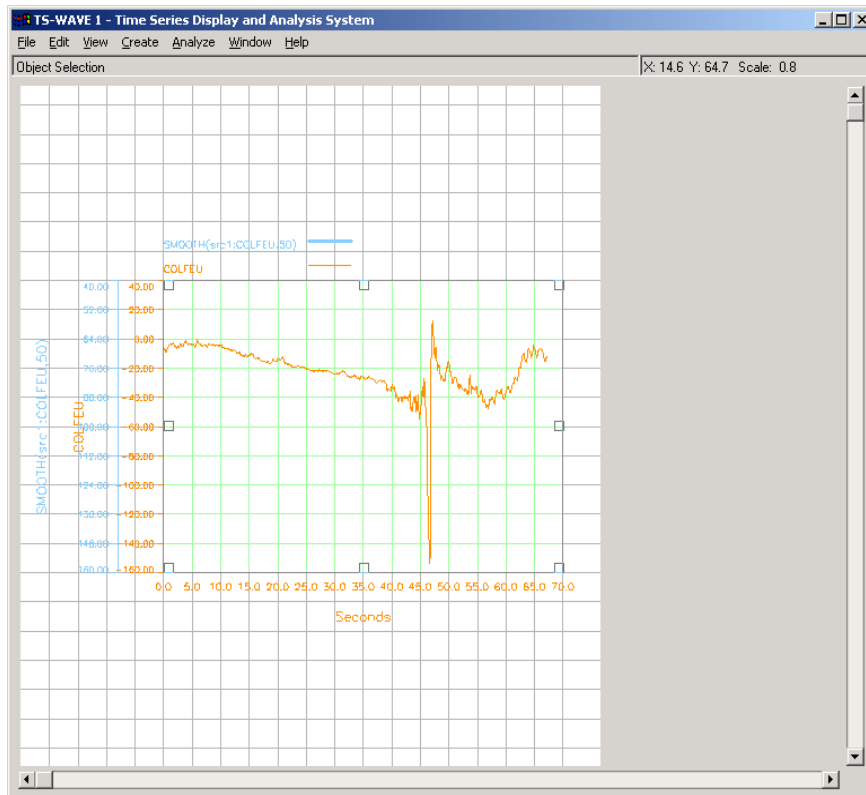


Figure 3-7 The Page contains data plotted from a new derived parameter that used the Smooth function.

Selecting, Resizing and Moving a Graph

You can resize or move a Graph object, but first you must select it.

Step 1 Click anywhere in the Graph object.

NOTE ‘Handles’ appear at points on the perimeter of the Graph object. These handles indicate that the Graph object is selected. When selected, the Graph object can be resized, cut, deleted or copied.

Step 2 Resize the Graph object. Move the pointer over one of the handles, hold down the left mouse button and drag the handle to resize the Graph object. Release the mouse button to complete the resizing.

TIP When you press down the mouse button, the pointer changes its shape. If the pointer does not change, be sure the pointer is just inside the handle of the Graph object as you press the mouse button.

Step 3 Move the Graph object. Position the pointer inside the Graph object, and hold down the left mouse button (this causes the pointer to change its shape). Then drag the Graph object to its new position.

TIP If necessary, move or resize the Graph object so that you can see both Y axes.

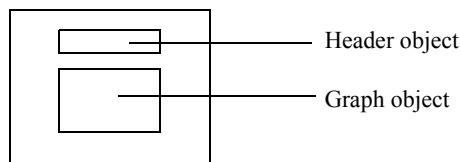
You have plotted two variables from the original data file. Let's add a Header object to your Page.

Adding a Header Object

A Header object is a text box that you can add to the Page. Header objects are created just like graph objects.

Step 1 Select **Create=>Header Object**.

Step 2 In the same manner as you created the Graph object, use the mouse to create a Header object just above the Graph object. The following schematic shows the approximate position of the Graph object and Header object on the Page:



Step 3 Select **Edit=>Object Select**.

Step 4 Double click anywhere inside the header box to bring up the **Header Attributes Interface**, shown in [Figure 3-8](#). This dialog is used to specify the contents of the Header objects and to modify the appearance of the text (font, size, color, and so on).

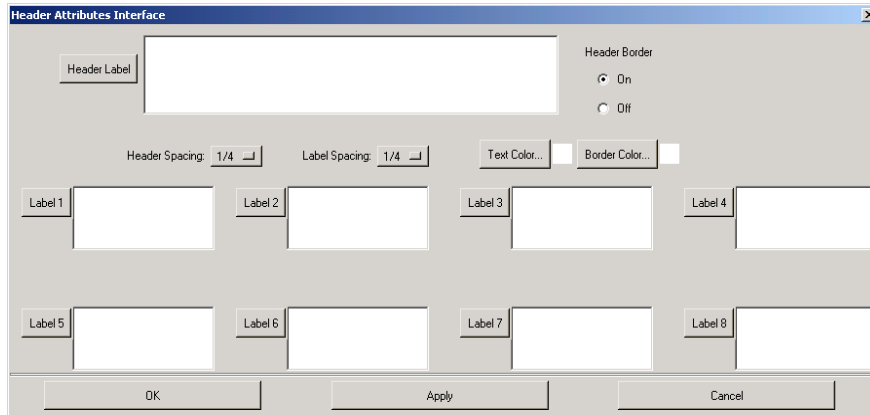


Figure 3-8 The Header Attributes Interface

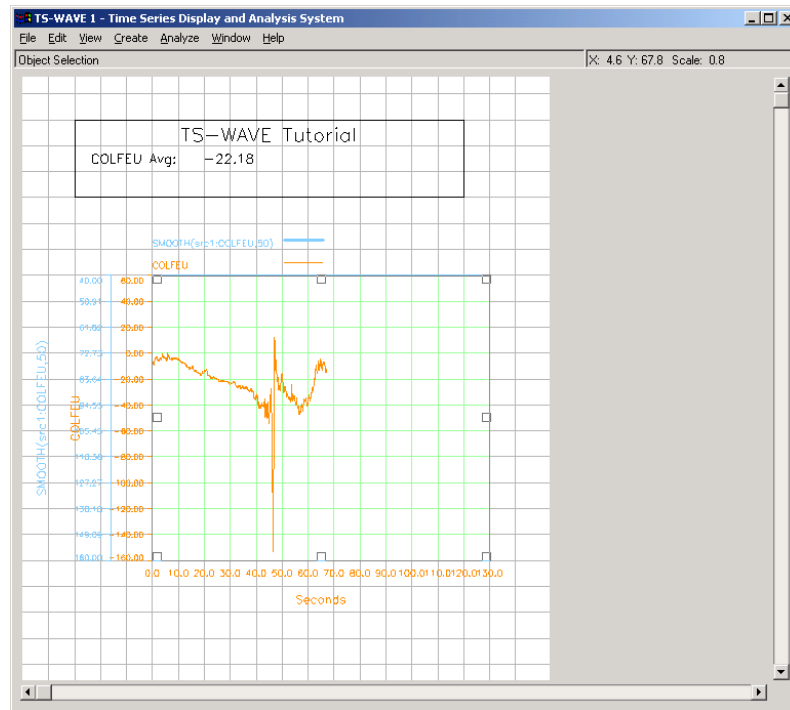
- Step 5** In the **Header Label** field, enter TS-WAVE Tutorial.
- Step 6** Click the **Header Label** button to bring up the **Advanced Label Editing Interface**. This dialog lets you change the font, text size, position, and other text characteristics. Change the **Print Size** to 1.5, and click **OK** to dismiss the **Advanced Label Editing Interface**. For more information on this dialog box, see the *Advanced Label Editing Interface* discussion under the Header Object discussion beginning on page 124.
- Step 7** In the **Label 1** text field, enter the following text:
- ```
COLFEU Avg: %%AVG(src1:COLFEU):(F8.2)%%
```
- This line demonstrates how you can embed PV-WAVE functions in a TS-WAVE header label. When a line with this syntax is included in a Header object, the specified calculation is made whenever a new data set is loaded into the TS-WAVE session. In this case, the average of the COLFEU parameter is calculated and displayed in the Header object. For more information on embedding PV-WAVE functions in header labels, see the *Header Attributes Interface* discussion under the Header Object discussion beginning on page 124.
- Step 8** Click **OK** to dismiss the **Advanced Label Editing Interface**.
- Step 9** Click **OK** to dismiss the **Header Attributes Interface**.

*Figure 3-9* shows the Page after the addition of the Header object.

---

**TIP** If you want to modify the header text, simply double click again inside the Header object . You may have to go into object select mode first by selecting **Edit => Object Select**. If you wish, experiment now with adding labels and with modifying the appearance of the text.

---



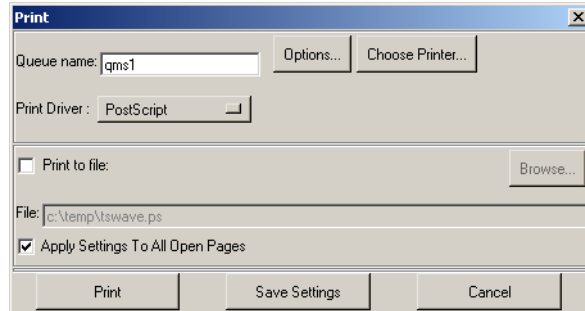
**Figure 3-9** The Page contains a Graph object and a Header object.

## Printing your Page

TS-WAVE provides options in the **Print** dialog for the most common printer features. The settings are dependent on the printer selection.

To print your Page:

**Step 1** Select **File=>Print**. The **Print** dialog appears as shown in [Figure 3-10](#).



**Figure 3-10** The Print dialog

---

**NOTE** The next three steps only apply to the Windows version of TS-WAVE. Unix users, enter the name of the printer in the **Queue Name** field, then skip to **Step 5**.

---

**Step 2** Click the **Choose Printer** button. The **Choose Printer** dialog appears.

**Step 3** Select a printer of your choosing from the list.

**Step 4** Click **OK** to dismiss the **Choose Printer** dialog.

**Step 5** If you wish to produce PostScript output, leave the printer driver set to PostScript. Otherwise, choose another driver from the **Print Driver** drop-down menu.

**Step 6** Click **Print**. Notice the **Print** dialog closes and the job is sent to the printer.

For a detailed discussion about customizing your printer settings, see [Using Resource Files](#) on page 62. For details on the options in the **Print** dialog, see [Print dialog](#) on page 98.



---

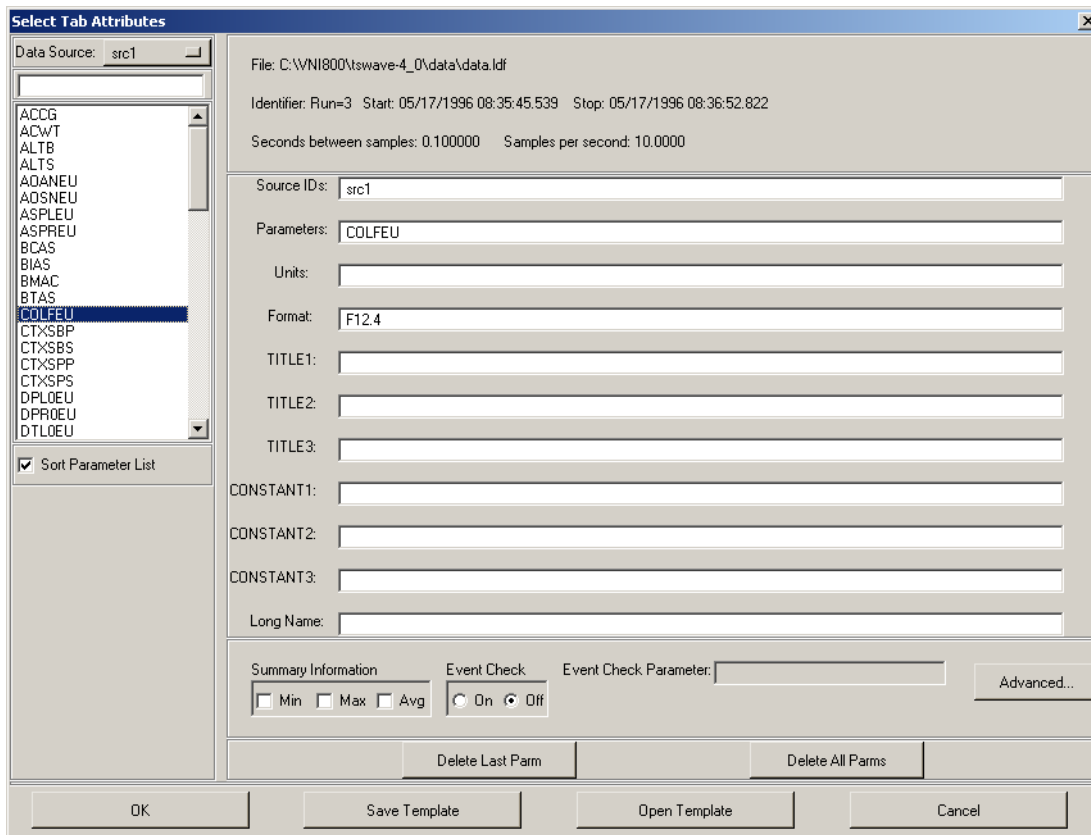
## ***Creating and Viewing a Tab File***

A tabular data file is an output file that contains a subset of the currently loaded data file(s). Tab Files are tabulary arranged ASCII files that consist of header information and columns of data. The Tab File can be used for reports, loaded into another application or read back into TS-WAVE for further analysis. For a detailed discussion on this functionality,<sup>74</sup>

see [\*Exporting Your Data\*](#) on page 51.

### **Creating a Tab File**

- Step 1**    Select **Create=>Create TabData**. The **Create Tab Data File** dialog appears.
- Step 2**    Enter a file name. For example, `test.tab`, then select **OK**. The **Select Tab Attributes** dialog appears.



**Figure 3-11** The Select Tab Attributes dialog

The left side of the dialog displays data source and parameter information for the current session.

The right side shows the information that is written to the Tab File.

**Step 3** Single-click on a parameter name in the Parameter List. For example, select **COLFEU**. Notice the Source ID, parameter name and the output format specifier for COLFEU appear in the fields to the right.

**Step 4** Select **OK** to write out the file. Notice a progress report in the upper left Status Message area on the main TS-WAVE interface.

## Viewing a Tab File

To view a Tab File in any text editor:

- Step 1** Select **Create=>View TabData File**. The **View Existing TabData File** dialog appears.
- Step 2** Select `test.tab` and select **OK**. The file is opened using the default application as specified by the `Windows_Tab_Viewer` and `Unix_Tab_Viewer` resources, depending on your current platform, in the file `<tswave_dir>resource/tswave_files.ads`.

---

**NOTE** The default application for Windows is `write.exe`. The default Unix resource is left blank and defaults to a PV-WAVE text widget. For more information, see [Working with Tab Files](#) on page 52 and [View TabData File](#) on page 148.

---

- Step 3** When you are finished viewing `test.tab`, close the viewer.

---

## Creating a Batch File

Batch Processing is a feature that allows TS-WAVE to be run as a background process without graphical user interface. Batch Processing is responsible for loading templates, data files, printing the generated graphs or creating a tabular output file. For a detailed discussion on this functionality, see [Batch Processing](#) on page 66 of this manual.

- Step 1** Select **Create=>Open Batch File**, then type or select a filename. For example, `test.job`. By default, Batch Files are created in the `<tswave_dir>/batch` directory.
- Step 2** Click **OK** to dismiss the **Open Batch File** dialog.
- Step 3** Select **Create=>Add To Batch File=>Add Template To Batch** to add the current Page configuration (Graph objects, Header objects, data, and so on) to the Batch File.
- Step 4** Select **Create=>Close Batch File** to close and save the Batch File. The process of closing the Batch File creates the `test.job` file, a script file (`test.bat` for Windows and `test.sh` for Unix) used to execute the Batch Job and other miscellaneous files.
- Step 5** Execute the Batch Job script. The Page is sent to the default printer specified in the resource file. Check `<tswave_dir>/twavebatch.log` file for any Batch messages.

## ***User's Reference***

This reference discusses:

*TS-WAVE Menus and Dialogs*

*Resource Files*

*Using Shortcut Keys*

*FORTRAN Format Strings*

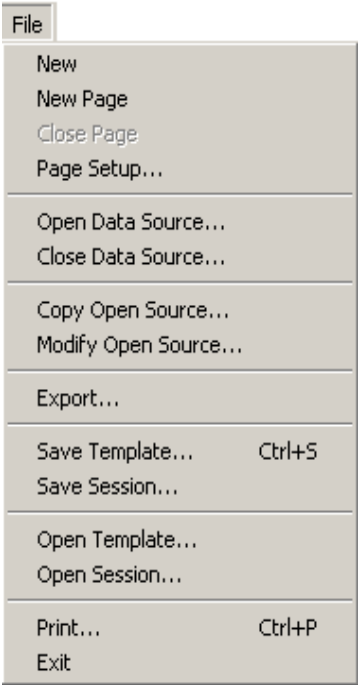
---

### ***TS-WAVE Menus and Dialogs***

This section describes the commands on the TS-WAVE menus:

- *File Menu*
- *Edit Menu*
- *View Menu*
- *Create Menu*
- *Analyze Menu*
- *Window Menu*
- *Help Menu*

# File Menu



## ***New***

Clears objects on the Page. Current Source IDs remain loaded.

---

**CAUTION** Be sure to save existing graphics before choosing the **New** function. Otherwise, your unsaved work will be lost.

---

## ***New Page***

Opens a new Page in your current TS-WAVE session.

---

**NOTE** By default, each new Page will be created with the page setup and color settings of the Page from which it was created. In order to have the new Page created with your default page settings, change the value of the Multi\_Use\_Default resource in <tswave\_dir>/resource/tswave.ads to 1.

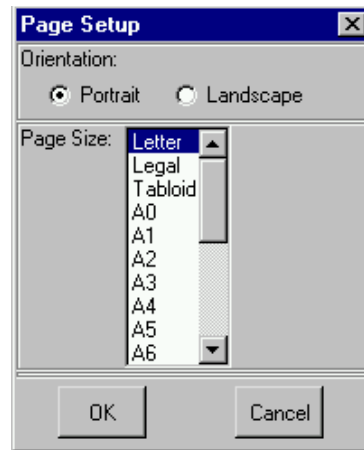
---

## **Close Page**

Closes the Page from which this menu item is selected. The main Page, 'TS-WAVE 1' cannot be closed without exiting the application.

## **Page Setup**

Brings up the **Page Setup** dialog. Use this dialog to change the orientation and size of the Page. These commands affect both the on-screen and printed versions of the Page.



**Figure 4-1** The Page Setup dialog

**Orientation** — Allows you to select the orientation of your output page.

- **Portrait** — If this option is selected, the Page is displayed in portrait orientation, with the X axis along the short dimension of the Page.
- **Landscape** — If this option is selected, the Page is displayed in landscape orientation, with the X axis along the long dimension of the Page.

**Page Size** — Displays the list of available paper sizes. Select the output paper size from the list. Letter is 8.5 by 11 inches. Legal is 8.5 by 14 inches. Tabloid is 11 by 17 inches. The A and B sizes adhere to their respective international paper sizes.

**OK** — Applies the changes and closes the dialog.

**Cancel** — Closes the dialog without applying any changes.

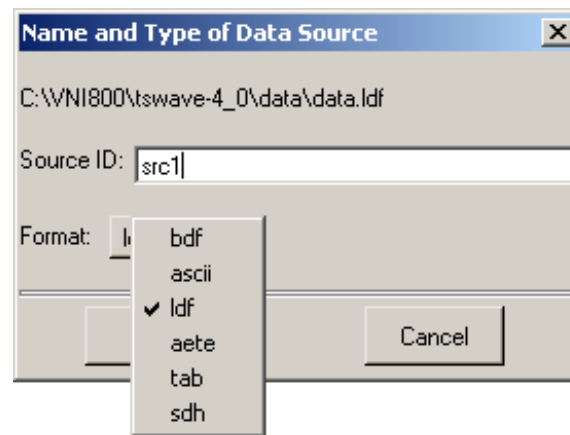
---

**NOTE** If you change the orientation of a Page that already contains graphical objects, you may have to relocate them so that they remain on the Page in its new orientation. The bottom left corner of the Page is the Page's 0,0 coordinate position.

---

### ***Open DataSource***

Brings up the **Open Data File** dialog, which is a standard file selection tool. Use this dialog to select a file to open. When you click **OK** in this file selection tool, the **Name and Type of Data Source** dialog appears.



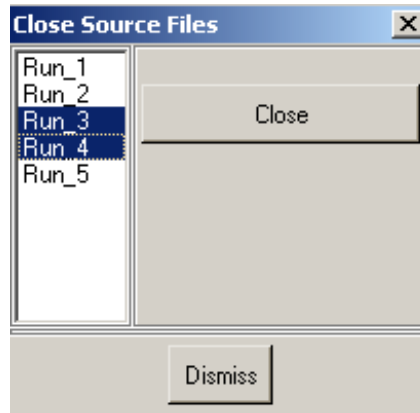
**Figure 4-2** The Name and Type of Data Source dialog

Use this dialog to define the Source ID and the file type of the data file you selected. Selecting **OK** in this dialog causes control to be passed to the Data Handler you selected.

### ***Close Data Source***

Brings up the **Close Source Files** dialog. Use this dialog to close any or all Source IDs in the current session.





**Figure 4-3** The Close Source Files dialog

The list on the left shows all of the open Source IDs. Select the Source ID(s) you wish to close and press the **Close** button. When you are finished select **Dismiss**.

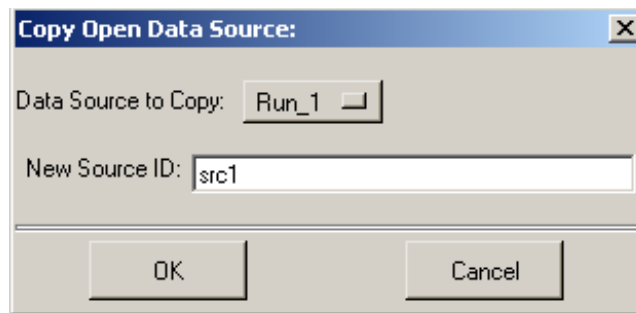
---

**NOTE** If a Source ID you are trying to close is in use by a graphical object you will not be able to close it.

---

### ***Copy Open Source***

Brings up the **Copy Open Data Source** dialog.



**Figure 4-4** The Copy Open Data Source dialog

Select the Source ID of the data set you wish to copy using the option menu. Enter the name of the new Source ID you wish create in the text box then click **OK**.

Control is then passed to the Data Handler appropriate for the Source ID you are copying.

### ***Modify Open Data Source***

Brings up the **Modify Open Source** dialog.

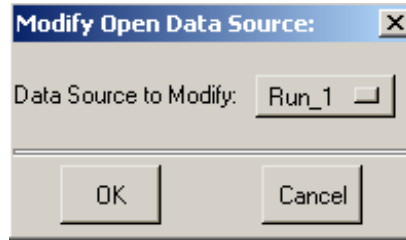


Figure 4-5 The Modify Open Data Source dialog

Use this dialog to modify data of an existing Source ID. Select the Source ID you wish to modify from the option menu then click **OK**. Control is then passed to the Data Handler appropriate for the data type you are modifying.

All open Source IDs except DERIVED can be copied or modified.

---

**NOTE** This option does not allow you to change the file format associated with the selected Source ID. To change this, select **File=>Open Data Source** and reuse the Source ID you wish to change.

---

### ***Export***

Brings up the **Export Data** dialog, which is a standard file selection tool. Use this dialog to select a file to which to export the data. When you click **OK** in the file selection tool, the **Select Export Format** dialog appears. Select a data file type to export your data from the option menu. Control is then passed to a **<dhType>\_writefile** procedure of the Data Handler for that data type. Data Handlers are not required to have a writefile procedure.

## ***Save Template***

Brings up the **Save Template As** dialog. This dialog is a standard file selection dialog that allows you to choose the filename and directory in which to save the Template File.

TS-WAVE Template Files are a snapshot of your TS-WAVE session, taken when the Template File is created. All configuration settings and graphical objects are stored in the file and will be restored when the file is read back into TS-WAVE. The only part of your session that is not restored is the open Source IDs because the purpose of Template Files is to allow you to re-use Graph objects and Page configurations with different data sets.

## ***Save Session***

Brings up the **Save Session As** dialog. This dialog is a standard file selection dialog that allows you to choose the filename and directory in which to save the file.

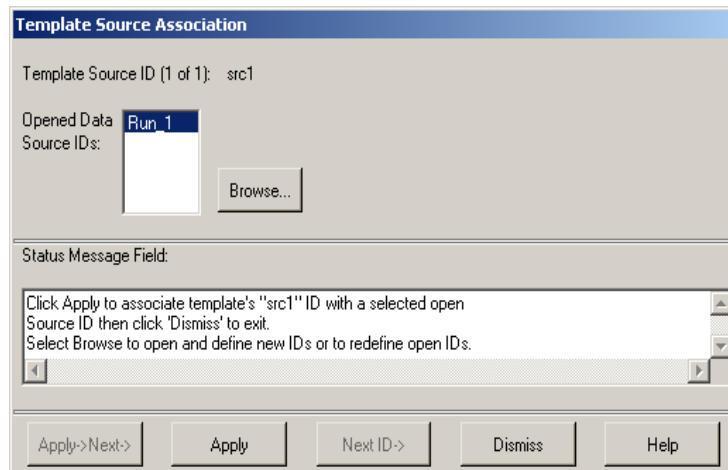
With one exception, a Session File is identical to a Template File. The exception is that a Session File also stores information about open Source IDs and restores them when the file is read back into TS-WAVE. This means that the data files associated with the Source IDs in your session must be available when the Session File is loaded. Session Files are handy for saving work in progress.

## ***Open Template***

Brings up the **Restore Template** dialog. Use this standard file selection dialog to open a previously saved template file. If the Template File contains Source IDs, the **Template Source Association** dialog appears as shown in [Figure 4-6](#).

### *Template Source Association Dialog*

The **Template Source Association** dialog allows you to match Source IDs in your Template File with Source IDs that exist in your current session. This dialog also allows you to open new Source IDs via the **Browse** button. The purpose of this functionality is to allow you to apply different data sets to an existing Page configuration.



**Figure 4-6** The Template Source Association dialog

For example, if you have a Template File that contains a single Graph object of a single parameter, 'src1:ALTB', and you open that Template File in a session that already has one open Source ID, 'Run\_1'. The **Template Source Association** dialog informs you that the Template File contains the Source ID 'src1' and that your current session contains the Source ID 'Run\_1'. If you associate 'Run\_1' with 'src1' in the dialog, the Graph object in the template file still uses the ALTB parameter from the 'src1' Source ID. Behind the scenes, TS-WAVE creates the 'src1' Source ID in your TS-WAVE session and the information from the 'Run\_1' Source ID is copied to it.

---

**TIP** Be careful when cross-associating multiple Source IDs since it is possible to inadvertently overwrite one that exists in your session with a copy of another. Review the **Status Message Field** on the **Template Source Association** dialog display to make sure you are doing what you intend.

---

The following is a step-by-step example of how to use the **Template Source Association** dialog:

- Step 1** Select an open Data Source ID from the list of Opened Data Source IDs. If the list is empty, select **Browse** to open and define one or more Data Source IDs then select one.

**Step 2** Click the **Apply** button to make the association. Review the **Status Message Field** message.

**Step 3** If the **Next ID** button is active, select it to continue processing of the next template Source ID.

Otherwise, when the **Next ID** button is inactive, indicating that the current template Source ID is the final one, make your final association, then select **Dismiss** to close the dialog.

### **Template Source ID**

This field names the current template Source ID being processed followed by a count of total Source IDs referenced in the template.

### **Opened Data Source IDs List**

This is a list of all open Source IDs in the current session. If none are open, it displays 'No Src files opened'. Additional Source IDs may be opened and defined using the **Browse** button. If an open Source ID matches the template Source ID being processed, it is automatically selected in this list.

### **Browse Button**

Select the **Browse** button to open and define a new Source ID or to redefine an existing open Source ID. If you re-use an existing open Source ID, the Source ID's new definition will override the previous definition. The **Browse** button has the same effect as selecting **File=>Open Data Source** from the main TS-WAVE interface.

### **Status Message Field**

Displays messages regarding the status of your last action or instructions about the next user action required.

### **Apply-> Next ID**

Using the **Apply ->Next ID** button is equivalent to clicking on **Apply** then **Next ID** when there is more than one template source to process and helps complete your associations more quickly. **Apply->Next ID** is recommended for advanced users who are familiar with the template association process. After **Apply->Next ID** is clicked, the **Template Source Association** dialog will automatically begin processing of the next template Source ID. The status of your last association does not display in the status message field. When all the template Source IDs are processed or if there are no open Source IDs available, this button becomes disabled.

## **Apply**

If a Source ID is selected from the Opened Source IDs list when the **Apply** button is clicked, the current template Source ID being processed is associated with the same data set as the open Source ID. When this button is clicked, a status message displays in the **Status Message Field**. When there are no open Source IDs available, this button becomes disabled.

## **Next ID**

If there is more than one template Source ID to process, the **Next ID** button becomes active. When selected, the **Associate Template** dialog continues to process the next template Source ID. While the final Source ID in the template is being processed, the **Next ID** button becomes disabled.

## **Dismiss**

Selecting the **Dismiss** button closes the **Associate Template** dialog.

---

**TIP** If your Template Source IDs are the same as your current open Source IDs, you can press **Dismiss** at any time and your Template Source ID will be associated with your current open Source IDs of the same name.

---

## **Help**

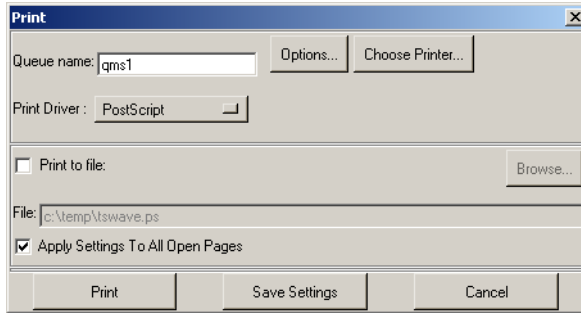
Brings up instructions on how to use this dialog.

## **Open Session**

Brings up the **Restore Session** dialog. Use this standard file selection dialog to open a previously saved Session File. There is no Source ID association step, as there is when loading a Template File, because all of the Source ID information in the Session File is loaded into your current TS-WAVE session. This means that duplicate Source IDs in your current TS-WAVE session are overwritten by those in your Session File.

## **Print**

Brings up the **Print** dialog.



**Figure 4-7** The Print dialog

**Queue Name** — Enter the name of the printer in this text field. The default printer name is specified in the resource file `<tswave_dir>resource/tswave_print.ads`. For more information on this resource file, see [Customizing TS-WAVE](#) on page 164.

**Options** — Brings up a dialog that lets you specify options for the currently selected **Printer Driver**. For detailed information about the options for each device, refer to *Appendix B, Output Devices and Window Systems*, in the *PV-WAVE Reference*.

**Choose Printer** — Opens a window that displays available printers.

**Print Driver** — Selects the type of printer or print file that you wish to use. The available drivers are:

- **PostScript** — Sends output to a PostScript device. If **Print to File** is selected, the output is saved in a PostScript file.
- **Windows Metafile** — (Windows only) Sends output to an Enhanced Metafile device. If **Print to File** is selected, the output is saved in an EMF file.
- **CG Metafile** — Sends output to a Computer Graphics Metafile device. If **Print to file** is selected, the output is saved in a CGM file.
- **HP PCL** — Sends output to a Hewlett Packard Printer Control Language device. If **Print to file** is selected, the output is saved in a PCL file.
- **HP GL** — Sends output to a Hewlett Packard Graphic Language device: If **Print to File** is selected, the output is saved in a GL file.

**Print to file** — Saves the output in a file instead of sending it to a printer.

**Apply Settings To All Open Pages** — When this checkbox is selected, all saved print settings are applied to all open TS-WAVE Pages. Settings are saved when either **Print** or **Save Settings** is selected.

**Print** – Save your settings and begin the print operation.

**Save Settings** – Saves your print settings for the current Page and closes the **Print** dialog.

**Cancel** – Closes the **Print** dialog without saving your settings.

### ***Exit***

Closes all open Pages and TS-WAVE. Exiting TS-WAVE is only available from the main TS-WAVE Page or (TS-WAVE 1). By default, exiting a TS-WAVE session will cause the close confirmation alert dialog to appear. To disable this alert for future sessions, change the resource `*TS_Wave*confirmClose` in `<tswave_dir>/resource/tswave.ad` to `False`.

---

**CAUTION** Be sure to save your work before you select the **Exit** command. Otherwise, your unsaved work will be lost.

---

---

**NOTE** You must also type `EXIT` in the PV-WAVE console to exit PV-WAVE if you are running in NoBlock mode.

---



## Edit Menu

| Edit               |             |
|--------------------|-------------|
| Object Select      | Ctrl+E      |
| Cut                | Ctrl+X      |
| Copy               | Ctrl+C      |
| Paste              | Ctrl+V      |
| Delete             | Ctrl+Delete |
| Align Graphs       |             |
| Plot Attributes... | Ctrl+F      |
| Select All         | Ctrl+A      |
| Deselect All       | Ctrl+D      |
| Redraw             | Ctrl+R      |
| Group              |             |
| Ungroup            |             |
| Front              |             |
| Back               |             |

### Object Select

Lets you select objects on the TS-WAVE Page. TS-WAVE objects are Graph objects, Contour objects, Header objects, Text objects, Lines objects, Box objects, and Ellipse objects, which are created using commands on the **Create** menu. Selected objects can be cut, copied, pasted, deleted, grouped, moved and resized. Double clicking on an object while in Object Selection mode brings up the attributes dialog for that object.

To select an object, activate the **Edit=>Object Select** menu item and single click on the object you wish to select. When an object is selected, 'Handles' appear at points on its perimeter indicating that it is selected.

---

**TIP** To select multiple objects: hold down the <Shift> key and click on the objects you wish to select; or, choose **Edit=>Select All** to select all objects on the Page at once; or, press and hold the left mouse button and drag the pointer around the objects you wish to select. You can also hold the <Shift> key to deselect individual objects.

---

## **Cut**

Cuts the selected objects to the clipboard, removing them from the Page.

---

**NOTE** The clipboard is an unseen location for temporary storage of the object(s) from the last **Cut** or **Copy** operation. The contents of the clipboard can be pasted to any TS-WAVE Page in your current session. Each new **Cut** or **Copy** overwrites the current contents of the clipboard.

---

## **Copy**

Copies the selected object(s) to the clipboard.

## **Paste**

Pastes the contents of the clipboard to the active TS-WAVE Page. The pasted objects are placed in front of other objects on the Page. Pasted objects appear in the same location on the new Page as they occupied when they were copied or cut. Pasting may be done between TS-WAVE Pages in a single session, but not between separate TS-WAVE sessions.

## **Delete**

Removes the selected object(s) from the Page and redraws the view. A deleted object does not overwrite the clipboard contents and cannot be retrieved.

## **Align Graphs**

Use this function to align two or more Graph objects. First, select one Graph object as the “master” Graph object, then select **Align Graphs**. All other Graph objects on the Page are moved and resized to match the master graph.

## **Plot Attributes**

Brings up the appropriate **Contour** or **Graph Attributes Interface** for the selected plot. This is equivalent to double-clicking on a Graph or Contour object. For more information on the **Graph** and **Contour Attributes Interfaces**, see [Adding Graph Objects to Your Page](#) on page 26 and [Creating a Contour Plot](#) on page 36, respectively.

## **Select All**

Selects all objects on the Page.

***Deselect All***

Deselects all selected objects on the Page.

***Redraw***

Redraws the Page.

***Group***

Groups the selected objects. Grouped items can be cut, copied, pasted, deleted, and moved as one unit.

***Ungroup***

Ungroups the selected grouped objects.

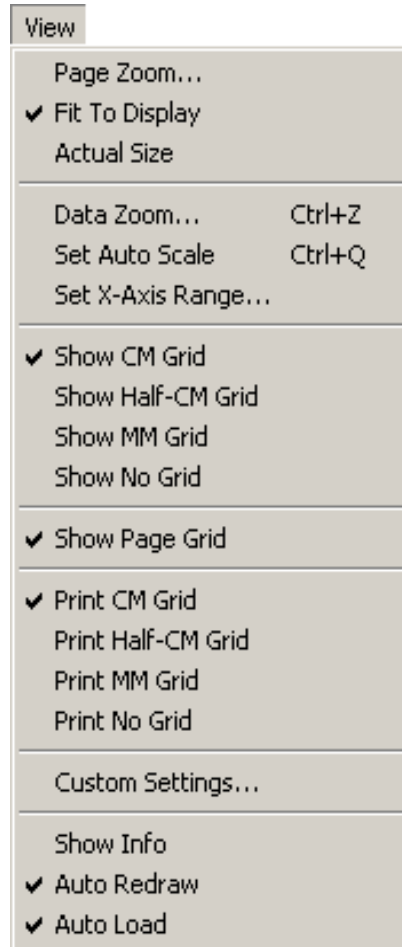
***Front***

Places the selected object(s) on top of the unselected objects.

***Back***

Places the selected object(s) behind the unselected objects.

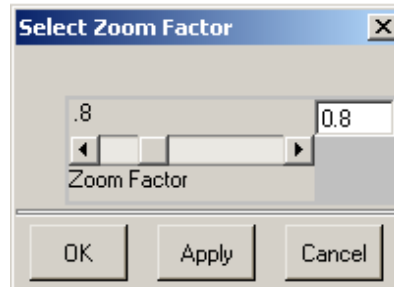
## View Menu



### **Page Zoom**

This command controls the magnification of the Page. Choose **View=>Page Zoom** to define a desired zoom factor from 0.1 to 3.0 in the dialog that appears.

To define the zoom factor, use the **Zoom Factor** slider or enter a value between 0.1 to 3.0 in the text field followed by a <Return> before clicking **OK** or **Apply**



**Figure 4-8** The Select Zoom Factor dialog

---

**TIP** Unlike Data Zoom, the Page Zoom feature does not change the axes ranges of your plot, it simply modifies your page view.

---

### ***Fit To Display***

This command automatically sets the Page Zoom factor so that a full Page of any size can be viewed on your display. The Fit To Display feature is automatically turned off if you select a different zoom factor from the Page Zoom menu.

### ***Actual Size***

Sets the Zoom Factor to 1.0 so the page appears at its actual physical size.

---

**TIP** If the Page does not refresh correctly, click in one or both of the scroll bars, or select **Edit=>Redraw**.

---

## Data Zoom

This command allows TS-WAVE users to interactively zoom in on one or more parameters, using the mouse. For more discussion on this topic, see [Data Zoom](#) on page 41.

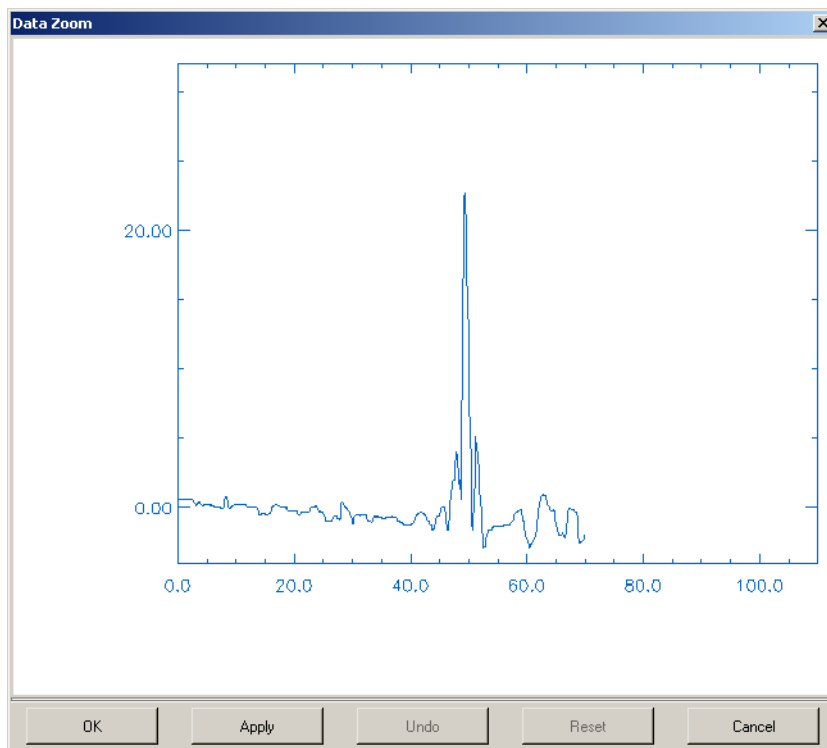


Figure 4-9 The Data Zoom interface

### Dialog Buttons

**OK** — Applies the zoomed range(s) to the selected Graph objects and dismisses the dialog.

**Apply** — Applies the zoomed range(s) to the selected Graph objects without dismissing the dialog.

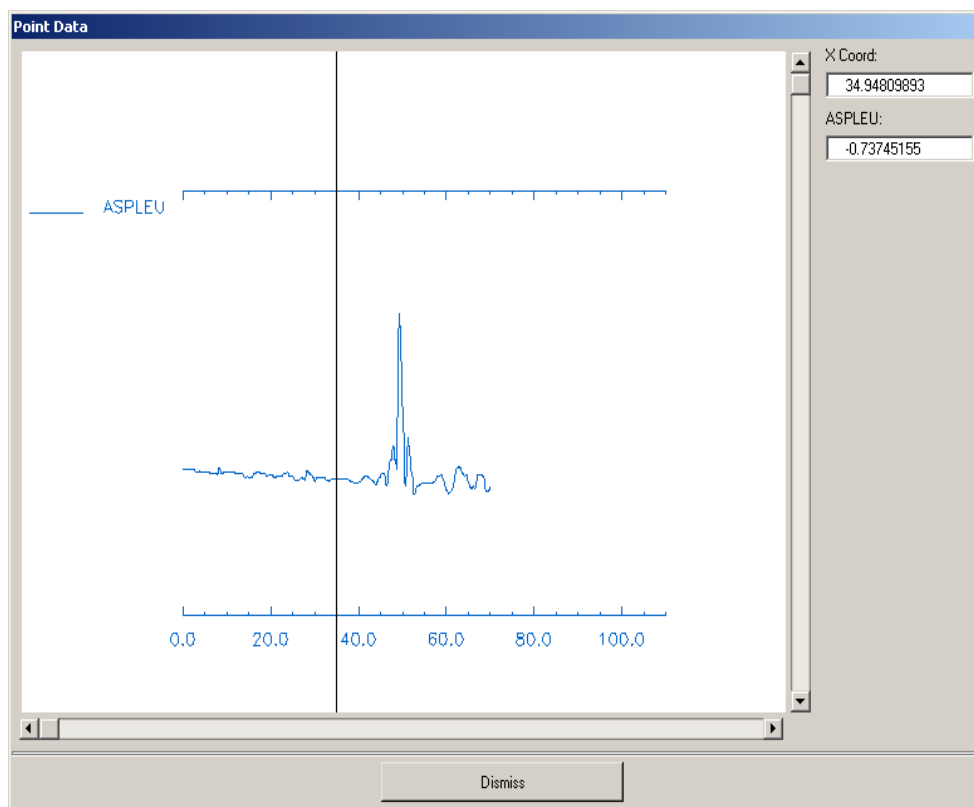
**Undo** — Reverts to the previous zoomed range(s) or the original range(s) if it was the first zoom.

**Reset** — Reverts to the original zoomed range(s) regardless of how many zooms/applies were performed.

**Cancel** — Exits the dialog without saving the current scales. If an **Apply** was performed, it will not undo these changes.

### ***Examine Data Points***

This command allows you to interactively view data values on a selected Graph object. When selected, it brings up a separate window containing the selected plot. The mouse cursor becomes a vertical line and the values for the plotted parameters at that position appear on the right side of the interface. This is essentially the same as creating Pick File data, except that no selections can be made and no data is written to an external file. All of the rules for creating a Pick File apply with the additional rule that your master X axis may not be a Date/Time axis.



**Figure 4-10** The Examine Data Points window

### ***Set Auto Scale***

Automatically sets X and Y axis scalings to Auto for all axes in each selected Graph object. This feature is not available for Contour plots.

### ***Set X Axis Range***

Sets the X axis scaling for every X axis in the selected Graph object to match the scaling of the axis you select in the **Data Source** dialog. If the X axis scaling of the selected axis is set to manual, the other X axis ranges are also set to match the manual range of the selected axis.

The **Show** grids only appear on your display and do not appear on your printed output. The **Print** grids only appear in your printed output and do not appear on your display.

### ***Show CM Grid***

Displays a one centimeter grid in Graph objects. (Default)

### ***Show Half-CM Grid***

Displays a one-half centimeter grid in Graph objects.

### ***Show MM Grid***

Displays a one millimeter grid in Graph objects.

### ***Show No Grid***

Turns off all grid displays in Graph objects.

### ***Show Page Grid***

Displays a one centimeter non-printable grid on the entire Page.

### ***Print CM Grid***

Displays a one centimeter grid in Graph objects on the printed output. (Default)

### ***Print Half-CM Grid***

Displays a one-half centimeter grid in Graph objects on the printed output.



### ***Print MM Grid***

Displays a one millimeter grid in Graph objects on the printed output.

### ***Print No Grid***

Turns off all grid displays in Graph objects on the printed output.

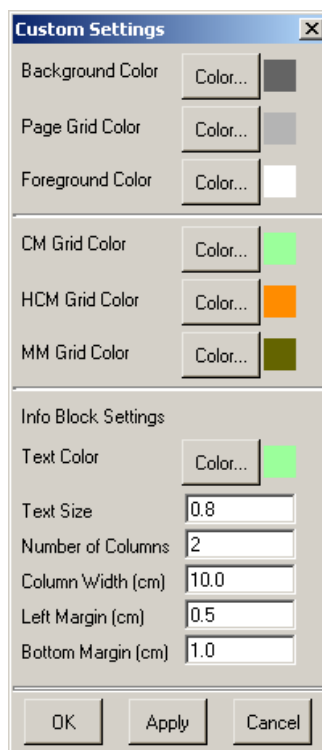
### ***Custom Settings***

Brings up the **Custom Settings** dialog that allows you to modify all of the display colors as well as the format and position of the information displayed by the **View=>Show Info** option.

---

**NOTE** These settings will be used for the current session only. For details on customizing your default settings, see [Resource Files](#) on page 164.

---



**Figure 4-11** The Custom Settings dialog

**Background Color** — Brings up the **Background Color** dialog. Use this dialog to change the background color of the Page.

**Page Grid Color** — Brings up the **Page Grid Color** dialog. Use this dialog to change the color of the page grid.

**Foreground Color** — Brings up the **Foreground Color** dialog. Use this dialog to change the foreground color of the Page.

---

**TIP** Make sure the background and foreground colors are not the same.

---

**CM Grid Color** — Brings up the **Select CM Grid Color** dialog. Use this dialog to change the color of the CM (one centimeter) grids.

**HCM Grid Color** — Brings up the **Select HCM Grid Color** dialog. Use this dialog to change the color of the HCM (half centimeter) grids.

**MM Grid Color** — Brings up the **Select MM Grid Color** dialog. Use this dialog to change the color of the MM (one millimeter) grids.

### ***Info Block Settings***

This portion of the Custom Settings dialog shows settings for the Info Block that is displayed when **View=>Show Info** setting is activated. By default, Info Block settings reflect settings defined in `<tswave_dir>/resource/tswave.ads`.

- **Text Color** — Defines the color for the text displayed in the Info Block.
- **Text Size** — Defines the size of the text to be displayed in the Info Block.
- **Number of Columns** — Defines the number of columns to be displayed in the Info Block.
- **Column Width (cm)** — Defines the widths of each Info Block column in centimeters.
- **Left Margin (cm)** — Defines the left margin of the Info Block in centimeters.
- **Bottom Margin** — Defines the bottom margin of the Info Block in centimeters.

### ***Show Info***

Use the TS-WAVE **View=>Show Info** menu to set and display session information on the TS-WAVE Page. This can be customized to your site specifications. For more information on customizing the Info Block, see [Show Info](#) on page 225

of this manual. By default, the Info Block lists open Source ID(s) and the data file(s) associated with each one, followed by the Template name, if one is loaded.

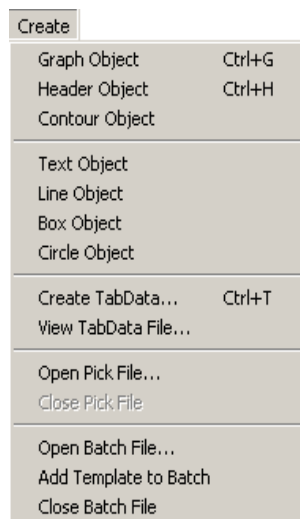
### ***Auto Redraw***

If this option is selected, the Page is automatically redrawn whenever a change is made that affects the displayed objects. If this option is not selected in order to save redisplay time, you may manually redraw your Page via **Edit=>Redraw**. This setting is not saved for Templates or Sessions.

### ***Auto Load***

If this option is selected, any new data that is selected is automatically loaded and displayed on the Page if it is being used. Otherwise, the Page is not updated when new data is selected. Use this feature to save time when initially laying out your Page and defining parameters for Graph objects, Contour objects or embedded Header functions. This setting is not saved for Templates or Sessions.

## **Create Menu**



### ***Graph Object***

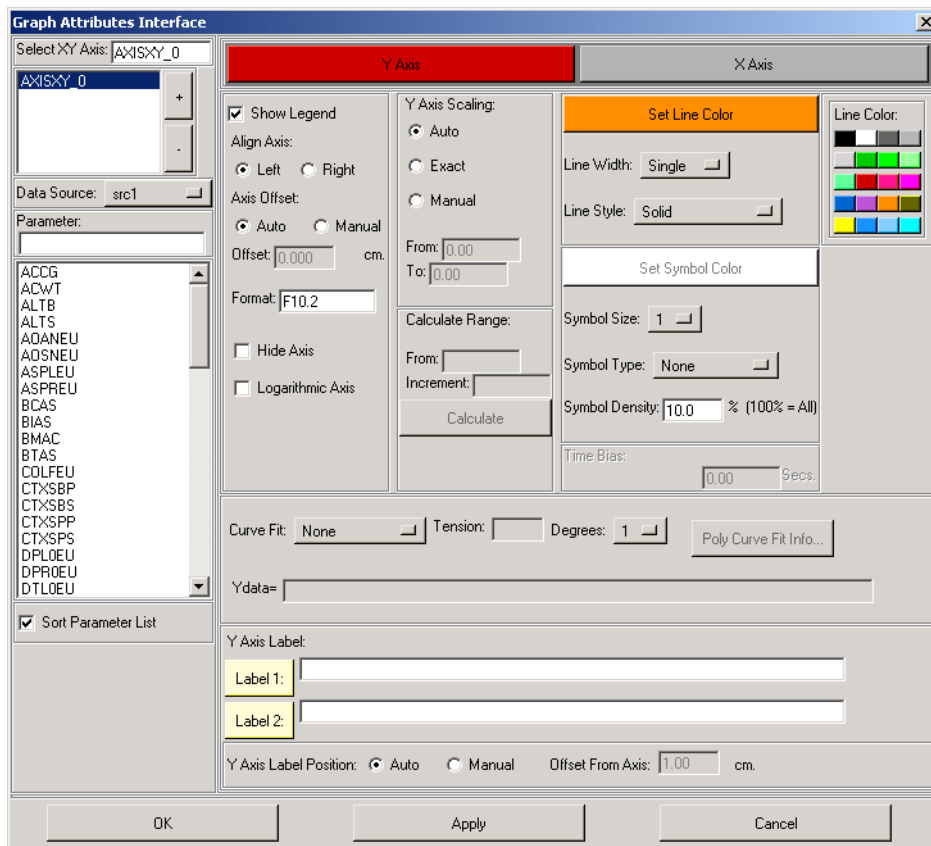
Lets you add a new Graph object to the Page.

To create a Graph object, select **Create=>Graph Object**, the message 'Create Graph Object' appears in the left side Status Message Area. Position your cursor

on the Page where you want the upper left-hand corner of the Graph object to be located then click and hold the left mouse button. Drag your mouse to define the plotting region then release the mouse button to finish. A blank Graph object appears in the defined location.

You can repeat this process as long as you are in graph creation mode as indicated by the message in the left side status message area.

Double clicking on a Graph object in Object Selection mode brings up the **Graph Attributes Interface** as shown in [Figure 4-12](#). Use this interface to add or delete parameters and modify various display settings for each parameter for both the Y and X axes.



**Figure 4-12** The Graph Attributes Interface with the Y axis selected

### *Graph Attributes Interface*

The **Graph Attributes Interface** first appears with the Y axis selected, as indicated by the highlighted button at the top of the interface. For details on adding a parameter to a Graph object, see the discussion about *Customizing Your Graph Object* in Chapter 2.

It should be noted that almost every default value in this interface can be modified and controlled via resource settings. For more information on using resources, see the *Customizing TS-WAVE* on page 164, or refer to the `<tswave_dir>/resource/ep_graphattribute.ads` file.

### *Y Axis Interface*

**Axis List** — Click on an axis in the **Axis List** to select it. The first axis is selected when the interface is created.

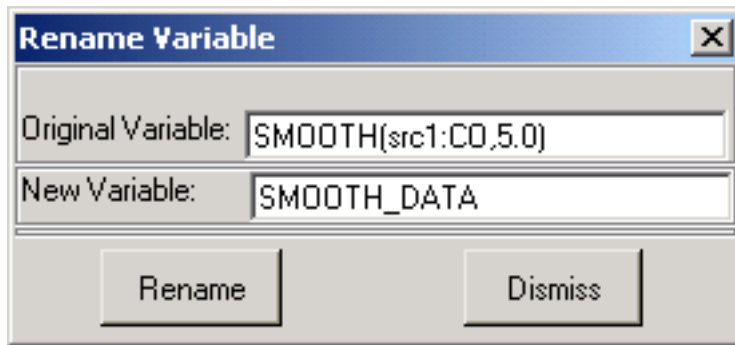
+ **and** - — Adds and deletes an axis from the Graph object.

To add additional axes, select the + button to the right of the Axis List. Each time you select the + button an additional axis will appear in the Axis List. The button deletes the selected axis.

**Data Source** — Use this drop-down menu to select from the list of open Source IDs. The Parameter List is updated to reflect the available parameters for the Source ID you select.

**Parameter List** — Single click on a parameter name to select it for plotting on the currently selected Y axis. When you select a parameter, its name appears in the **Label 1** field. Double click on a parameter in the Parameter List to bring up a window that displays general statistical information about the currently selected parameter.

If you double click on a parameter from the DERIVED Source ID, the information window will also contain a description of the derived parameter and a button labeled **Rename Variable**. Selecting this button will allow you to rename the selected derived parameter.



**Figure 4-13** The Rename Variable dialog

---

**NOTE** Existing Graph objects that contain the renamed parameter will automatically be updated to reflect the parameter's new name.

---

**Sort Parameter List** — When this checkbox is selected the Parameter List is sorted alphabetically. When unselected the Parameter List displays parameters in the order that they occur in the data file.

The following functions are grouped in the upper section of the dialog. These functions are mainly used to configure the appearance of the Y axis and plot lines.

**Show Legend** — Display line legends. Line legends appear above the Graph object. Line legends identify the parameter name along with its line style, line color, plot symbol and plot symbol color.

**Align Axis** — Positions the currently selected axis on the left or the right side of the graph.

- **Left** — Aligns the axis on the left on the plot.
- **Right** — Aligns the axis on the right on the plot.

**Axis Offset** — This text field specifies the offset in centimeters for the current axis from the edge of the plot.

- **Auto** — (default) Set the axis offset to a value that is based on the resource `Y Axis Space` in the file  
`<tswave_dir>/resource/ep_graphattributes.ads.`

- **Manual** — Allows you to define the distance the axis is offset from the edge of the plot. Manual axes are not taken into account when **Auto** axes are placed and overlap occurs.
- **Offset** — When **Manual** is selected this is the amount in centimeters to offset the current axis from the edge of the plot.

**Format** — Use this text field to modify the format code for the tick mark values on the currently selected Y axis. By default, the code F10.2 is used. If the **Y Axis Label Position** is set to **Auto**, increasing the number of format spaces will have the effect of moving the Y Axis label left or right.

---

**NOTE** You must use standard FORTRAN format codes in this field. For detailed information on FORTRAN format codes, see *[FORTRAN Format Strings](#)* on page 173.

---

**Hide Axis** — Allows a user to disable the drawing of the current axis. Other defined axis attributes (line color, symbol type, etc.) are applied to the parameter associated with the hidden axis. The axis itself is not displayed.

**Logarithmic Axis** — Allows a user to switch from linear distribution to logarithmic distribution.

**Y axis Scaling** — Allows the user to define the scaling or range for the current Y axis.

- **Auto** — Select **Auto** to let TSWAVE automatically calculate the Y range for you. The range will be divided in intervals which are exact multiples of 1.0, 2.0, 2.5, 4.0, or 5.0 times a power of 10, depending on the range of the data and on the size of the Graph object.
- **Exact** — Select **Exact** to set the axis end points to the minimum and maximum values of the parameter, thus maximizing the plot within the Y size of the Graph object you chose.
- **Manual** — Select **Manual** to specify the axis range yourself.

---

**NOTE** If you select **Manual**, the **From** and **To** text fields as well as **Calculate Range** become sensitized.

---

- **From** — Lets you specify the initial value on the Y axis.
- **To** — Lets you specify the end value on the Y axis.

**Calculate Range** — Allows you to maximize the plot according to your inputs within your chosen Graph object.

- **From** — Specifies an initial value for the axis.
- **Increment** — Specifies a value to increment between tick marks.
- **Calculate** — Calculates the axis range based on your entries in the **From** and **Increment** fields and the dimensions of your Graph object. If you resize your Graph object your axis increments will change.

**Curve Fit** — Lets you select a curve fitting algorithm to apply to the plotted points.

The choices are:

- **None** — Do not use any curve fitting. (Default)
- **User Equation** — If this method is selected, you can enter an equation in the **Ydata=** field immediately below this menu.
- **Polyfit** — Produces an  $n$ -degree polynomial curve through the set of data points using the least-squares method. Select the degree of the polynomial to be fitted to the data with the **Degrees** option menu. For more information on the Polyfit algorithm, refer to the POLY\_FIT function in the *PV-WAVE Reference*.
- **Polyfitn** — Fits an  $n$ -variate polynomial to some I-dimensional data points using the least-squares method. Select the degree of the polynomial to be fitted to the data with the Degrees option menu. For more information on the Polyfitn algorithm, refer to the POLYFITN routine in the *PV-WAVE Mathematics Reference*.

---

**NOTE** Polyfit and Polyfitn provide similar functionality, but use different algorithms in order to handle the widest range of situations.

---

- **Polyfitw** — This algorithm is a specialized version of the Polyfit and Polyfitn algorithms where the data points have been weighted inversely proportional to their magnitude. Select the degree of the polynomial to be fitted to the data with the Degrees option menu. For more information on the Polyfitw algorithm, refer to the POLYFITW function in the *PV-WAVE Reference*.

---

**NOTE** Selecting **Apply** with any of the Polyfit methods selected will cause the **Poly Curve Fit Information** button to sensitize.

---

- **Spline** — Performs a cubic spline interpolation of the parameter's data points. Splining results are best visible with smaller data sets. The X axis Parameter



must be monotonic and increasing to be used with the Spline curve fit. For more information on the Spline algorithm, refer to the SPLINE function in the *PV-WAVE Reference*.

**Tension** — Lets you enter the amount of tension to be put on the splined curve. This value controls the smoothness of the fitted curve. This text field is active when the **Spline** function is selected.

**Degrees** — Lets you pick the degree of the polynomial to be fitted to the data. This menu is active when any of the **Polyfit** functions are selected.

**Poly Curve Fit Information** — Selecting this button will open a dialog that displays the Equation used, the Coefficient(s) of the equation, the Correlation Coefficient, the Function Standard Deviation and the Y Standard Deviation of Each Point. This information is only available after selecting the **Apply** button on the interface. Selecting **Apply** with any of the Polyfit methods selected will cause the **Poly Curve Fit Information** button to sensitize. If you do not need to view the Y Standard Deviations for each point, set the resource `PolyFit.ShowYDev` in `<tswave_dir>/resource/ep_graphattribute.ads` to '0' to speed up calculation of Poly Fit information for large data sets.

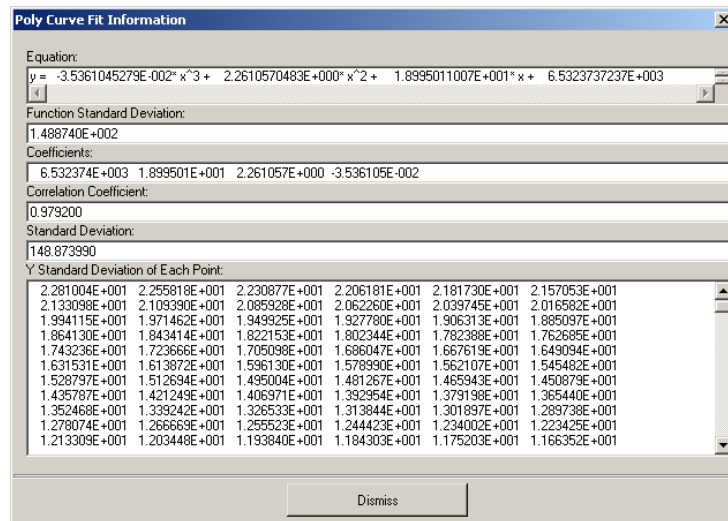


Figure 4-14 The Poly Curve Fit Information dialog

**Ydata** = — Enter any PV-WAVE expression in this field. For example,  $\sin(x)-y$  is a valid user equation. This text field is active when **User Equation** is selected from the **Curve Fit** menu.

---

**TIP** If your **Y Axis Scaling** is set to **Manual**, you may need to set it to **Auto** if your User Equation radically changes the range of your data.

---

### *Y axis Label Fields*

These text fields appear at the bottom of the dialog. They are used to add text labels to the Y axis and to modify the appearance of the text (fonts, text color, text size, and so on). Type your desired labels directly into the text boxes provided. Labels are placed on the Graph object in descending order, with Label 1 farthest from the axis.

---

**TIP** When a parameter is selected from the Parameter List its name is automatically entered into Label 1. If you wish to modify Label 1, do so after you have selected a parameter or your label will be overwritten.

---

**Label 1** — Click the **Label 1** button to bring up the **Advanced Label Editing Interface** dialog. This dialog lets you customize the appearance of the text entered in the adjacent field. You can set the font (normal, italic, bold, bold italic, Greek, and math), character position normal (normal, superscript, subscript), print size (1 is the default, 2 is twice as big, and so on), orientation (90 degrees is the default). For more information on the **Advanced Label Editing Interface** dialog, read the [Adding a Header to Your Page](#) of this manual.

**Label 2** — Click the **Label 2** button to bring up the **Advanced Label Editing Interface** dialog for this label.

---

**NOTE** The number of available labels is determined by the resource `Number_Y_Labels` in the file `<tswave_dir>/resource/ep_graphattributes.ads` and can range from 1 to 4.

---

**Y Axis Label Position**— Determines the distance the axis label is from the axis itself.

- **Auto** — TS-WAVE will automatically determine the label location based on the tick size and axis format code.
- **Manual** — Sensitizes the axis from the axis text field.
- **Offset From Axis** — Label offset in centimeters.

**Line Width** — Lets you specify the thickness of plot lines. Choices are: **Single**, **Double**, and **Triple**.

**Line Style** — Lets you specify the style of plot lines. Choices are: **None**, **Solid**, **Short Dashes**, **Long Dashes**, **Long-short Dashes**, and **Long-short-short Dashes**.

**Symbol Size** — Lets you pick a size for the plot symbols. Active only if **Symbol Type** is defined.

**Symbol Type** — Lets you pick a symbol type for the current Y axis variable plot. Choices are: **None**, **Plus (+)**, **Asterisk (\*)**, **Period (.)**, **Diamond**, **Triangle**, **Square**, **X**, and **User Defined**.

---

**TIP** The **Color** matrix on the right side of the interface is used to set both the line color and the symbol color. Click **Set Line Color** or **Set Symbol Color** before selecting a color in the matrix to determine to which the selected color will be applied. The label changes over the color matrix changes to indicate whether your selection will affect the line color or the symbol color.

---

---

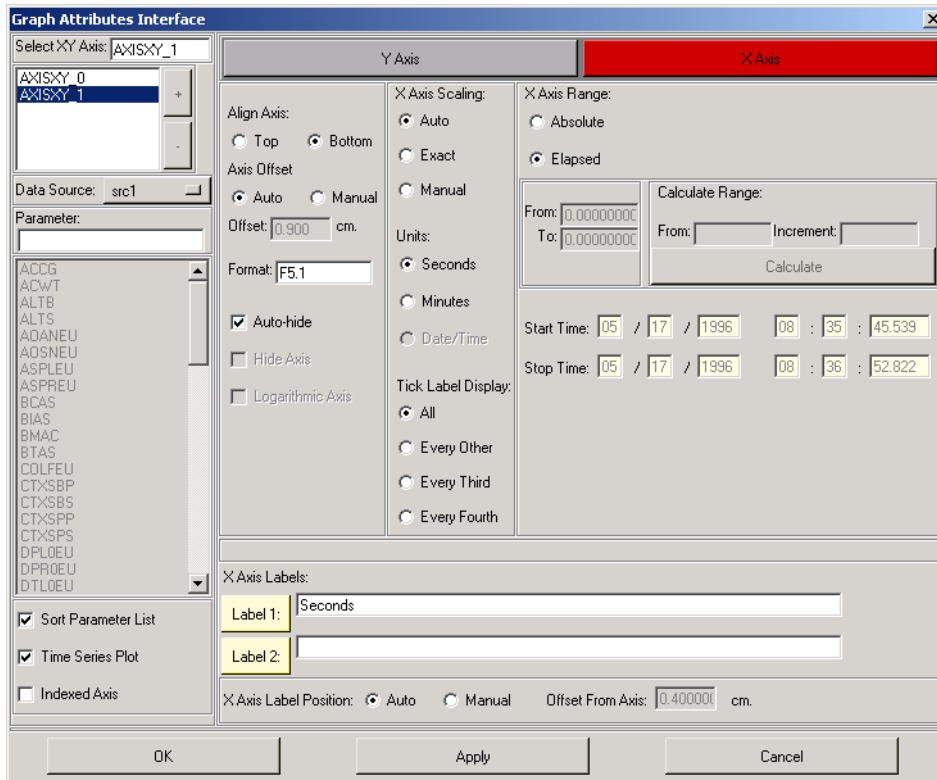
**NOTE** If you wish to add user-defined symbols to the **Symbol Type** menu, please contact Visual Numerics Customer Support for more information.

---

**Symbol Density** — This text field allows a user to specify a percentage that defines how often symbols are drawn on a data line. An entry of 50 implies a symbol is placed at every other data value location. An entry of 33 places a symbol at every third data value location. If the entry is 100, every data value location is indicated with a symbol.

## *X axis Button*

When you click on the **X Axis** button at the top of the **Graph Attributes Interface**, the interface switches to the X axis.



**Figure 4-15** The Graph Attributes Interface with the X axis selected

**Select XY Axis** — Displays the list of axes defined via the Y Axis interface and the currently selected Axis. Unlike the Y axis interface, you may only select among the available axes. You may not add or delete axes in the X axis interface.

**Data Source** — Use this drop-down menu to select from the list of open Source IDs. The Parameter List is updated to reflect the available parameters for the Source ID you select.

**NOTE** The Data Source menu and Parameter List are only enabled when both the **Time Series Plot** and **Indexed Axis** buttons are deselected.

**Parameter List** — When enabled, allows you to select a parameter to be plotted against the Y axis parameter. This type of plot is referred to as a cross plot.

**Sort Parameter List** — When this checkbox is selected the Parameter List is sorted alphabetically. When unselected the Parameter List displays parameters in the order that they occur in the data file.

**Time Series Plot** — When selected, this check box specifies that the Y axis parameter is displayed in a Time Series plot, where the X values are the time values associated with the Y axis parameter. This is the default.

**Indexed Axis** — When selected, the time values of the Y axis parameter are not used in the plot. Instead the index values of the Y axis data points are used on the X axis. This is useful if your data set contains repeating or out of sequence time values.

**Align Axis** — Positions the currently selected axis.

- **Top** — Places the current axis on the top of the plot.
- **Bottom** — Places the current axis on the bottom of the plot.

**Axis Offset** — This text field is only enabled when **Manual Axis Offset** is selected and specifies the offset in centimeters for the current axis from the top or bottom of the plot. The default is .8 cm.

**Format** — Use this text field to modify the format code for the tick mark values on the X axis of the currently selected graph. By default, the code F5.1 is used.

---

**NOTE** You must use standard FORTRAN format codes in this field. For detailed information on FORTRAN format codes, see *[FORTRAN Format Strings](#)* on page 173.

---

**Auto-hide** — Allows automatic hiding of identical axes for Time Series or Indexed plots. The Auto-Hide setting applies to all X axes of the graph. To selectively hide and show individual axes you must set this to off and set the **Hide** buttons individually for each axis. This is turned on by default.

- **Auto-hide toggle ON** — All identical X axes are displayed only once. If only one axis is present it will always be displayed.
- **Auto-hide toggle OFF** — Activates the ‘Hide Axis’ toggle. Axes will be hidden or displayed according to the ‘Hide Axis’ toggle setting for each axis.

**Hide Axis** — Allows a user to disable the drawing of the current axis. Other defined axis attributes (line color, symbol type, etc.) are applied to the parameter associated with the hidden axis. The axis is not displayed.

**Logarithmic Axis** — Allows a user to switch from linear distribution to logarithmic distribution.

**X Axis Scaling** — Allows the user to define the scaling or range for the current X axis.

- **Auto** — Select **Auto** to allow TS-WAVE to calculate the axis range based on the data values and the size of the Graph object. Use the axis range stored in the parameter. The range is divided in intervals which are exact multiples of 1.0, 2.0, 2.5, 4.0, or 5.0 times a power of 10, depending on the size of the X range and on the size of the Graph object.
- **Exact** — Select **Exact** to scale the X axis to the range of values stored in the parameter, thus maximizing the plot within the X size of the Graph Object you chose.
- **Manual** — Select **Manual** to specify the axis range yourself.

---

**NOTE** If you select **Manual**, the **From** and **To** text fields as well as **Calculate Range** become sensitized when **Units** is **Seconds** or **Minutes**. When **Units** is **Date/Time**, **Start Time** and **Stop Time** become sensitized.

---

**From** — Lets you specify the initial value on the X axis.

**To** — Lets you specify the end value on the X axis.

**Start Time** — Lets you specify a starting time value for a **Date/Time** axis.

**Stop Time** — Lets you specify an ending time value for a **Date/Time** axis.

**Calculate Range** — Allows you to maximize the plot according to your inputs within your chosen Graph object.

- **From** — Specifies an initial value for the axis.
- **Increment** — Specifies a value to increment between tick marks.
- **Calculate** — Calculates the axis range based on your entries in the **From** and **Increment** fields and the dimensions of your Graph object. If you resize your Graph object your axis increments will change.

**Units** — Lets you the units used on the X axis.

- **Seconds** — Time values are expressed in seconds.
- **Minutes** — Time values are expressed in minutes.
- **Date/Time** — You must select **X Axis Range** => **Absolute** to enable Date/Time Units. This option displays a Date/Time axis.

**Tick Label Display** — Tick Label Display allows you to control how many tick labels will be displayed. Choices are:

- **All** — Labels every tick.
- **Every Other** — Labels every other tick.
- **Every Third** — Labels every third tick.
- **Every Fourth** — Labels every fourth tick.

**X-Axis Range** — Defines the range of your X axis.

- **Absolute** — Displays the actual Date/Time value in Units of Seconds, Minutes or Date/Time, as recorded in the data provided. Choose your desired units in the **Units** field.
- **Elapsed** — Displays the Seconds or Minutes that have elapsed since the time stamp of the first data point. **Elapsed** deactivates the **Units Date/Time** button.

**X Axis Label** — By default, the **X axis** interface provides two text labels on the X axis. The labels are referred to as Label 1 and Label 2. The number of available labels is determined by the resource `Number_X_Labels` in the resource file `<tswave_dir>/resource/ep_graphattribute.ads`.

- **Label 1** — If the **Time Series Plot** check box is selected, the default for Label 1 is determined by the resource corresponding to for your selected **Units**. The default values are: ‘Seconds’, ‘Minutes’, and ‘Time’, corresponding to the **Units** selection. If **Indexed Axis** is selected, the default label is ‘Records’. If your plot is a cross plot mode, (both **Indexed Axis** and **Time Series Plot** off) the default is the name of the selected parameter. You may change the text in this field if you wish.

**X Axis Label Position** — Determines the position of the label on the X axis.

- **Auto** — Default position.
- **Manual** — When **Manual** is selected, **Offset From Axis** is activated, and allows you to enter a label offset in centimeters. Negative offset values are valid.

**OK** — Plots the specified parameter and closes the dialog.

**Apply** — Applies changes and redraws the Page, but does not close the dialog.

**Cancel** — Discards changes and closes the dialog.

## Header Object

Allows you to add a header object to the Page. A Header object is a set of grouped text fields that consist of a Header Label and up to eight equally-spaced sublabels below it. Each label can consist of one or more lines. Sublabels are the centered beneath the Header Label. The advantage of a Header object over a Text object is each label can contain multi-line text entries, they can be defined to have borders, and they can contain embedded functions. For more details, see [Adding a Header to Your Page](#) on page 23.

To create a Header object, select **Create=>Header Object** then position the upper left corner of your Header object with your left mouse button, drag the mouse to the desired size then release the mouse button to set the size of the header object size. Select **Edit=>Object Select**, and double click anywhere inside the header box to bring up the **Header Attributes Interface** dialog (shown below). Type any text in the Header Label field, then click **OK** to dismiss the **Header Attributes Interface** shown below.

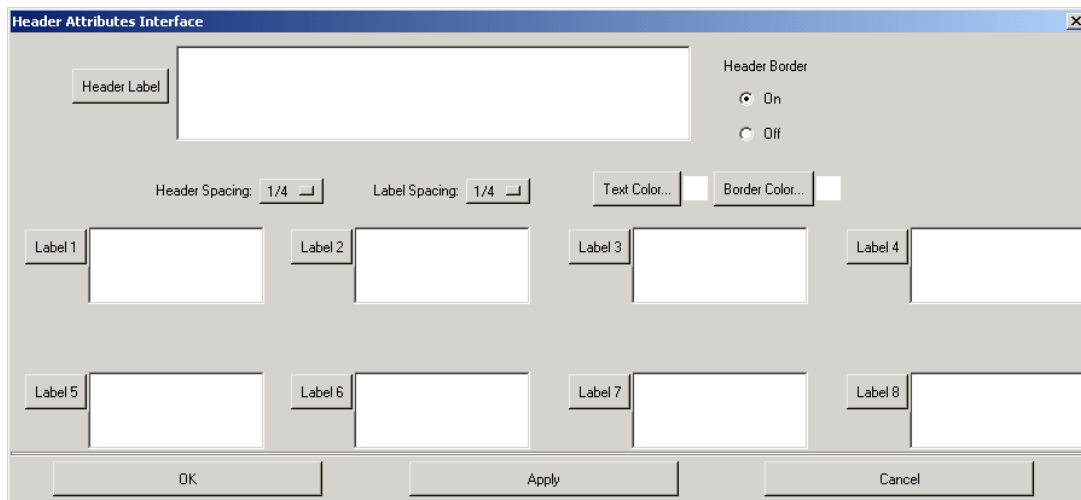


Figure 4-16 The Header Attributes Interface

TS-WAVE allows you to change the font, text color, and print size of any header text. It also has the ability to include calculations in a header by embedding PV-WAVE functions in any header text field.



### *Advanced Label Editing Interface*

This interface is used by all of the **Label** buttons on the **Header Attributes Interface** to define your heading. Your text is typed into the **Current Label** field and it is modified with the accompanying buttons and text fields.

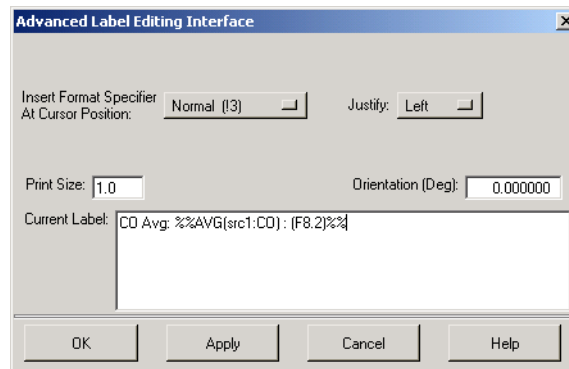
In addition to entering text, you can embed a PV-WAVE function in a label. The specified calculation is made whenever a new data set is loaded into the TS-WAVE session. The format used to embed a function into a label is:

`%%function_name(Source ID:parameter_name) : (Format)%%`

The format specifier is optional.

For example:

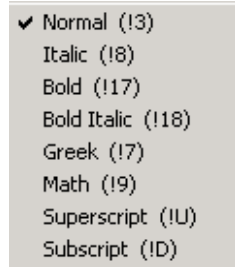
`CO Avg: %%AVG(src1:CO) : (F8.2)%%`



**Figure 4-17** The Advanced Label Editing Interface

**Insert Format Specifier At Cursor Position** — This allows you to integrate different font types into the header.

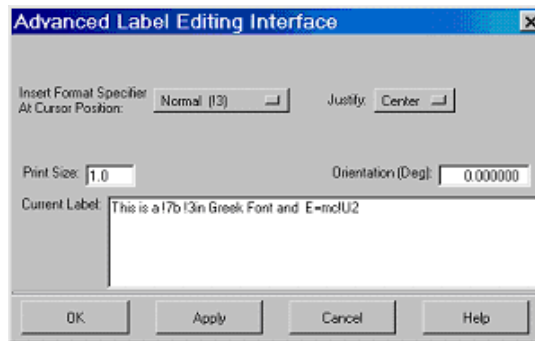
The available fonts are:



---

**NOTE** The software character set for each of these fonts is documented in the PV-WAVE Reference, Volume 2, Chapter 5, *Software Character Sets*. When changing font types, the font specifier will be added into the text at the location of the text cursor as shown below.

---



**Figure 4-18** The Advanced Label Editing Interface

- **Justify** — Allows you to left, center or right justify your text.
- **Print Size** — Sets the character size. The default size is 1.0.
- **Orientation** — Set the number of degrees to rotate the text.
- **Current Label** — Displays the current label.
- **OK** — Saves changes and closes the dialog.
- **Apply** — Saves changes and does not close the dialog.

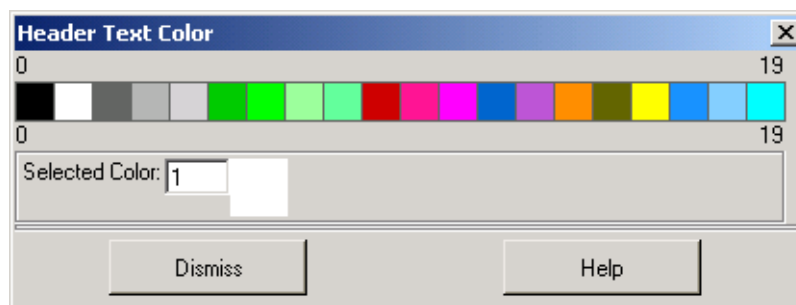
- **Cancel** — Discards unsaved changes and closes the dialog.

**Header Border** — Draws a border around the Header object when activated.

**Header Spacing** — The distance between multiple lines of the Header Label.

**Label Spacing** — The distance between multiple lines in a Label.

**Text Color** — Brings up a color selection dialog.



**Figure 4-19** The Header Text Color dialog

**Border Color** — Brings up a color selection dialog.

For more information on Header objects, see [Adding a Header to Your Page](#) on page 23.

### **Contour Object**

Lets you add a Contour object to the Page.

The Contour Plot feature is used to render a contour representation of a dependent variable  $z$ , as a function of two independent variables  $x$  and  $y$  ( $z(x,y)$ ). The Contour object feature creates a contour plot using data stored in three one-dimensional arrays. Each of X Axis, Y Axis, and Z Data must have parameters assigned to them before selecting **OK** or **Apply**. Each of the parameters must be of identical size and contain three or more data points.

To create a contour plot, select **Create => Contour Object** then position the mouse pointer on the Page where you want the upper left-hand corner of the Contour object displayed and select the left mouse button; then drag your mouse to define the plotting region and release the mouse button to set the desired contour plot area. Select your Contour object by choosing **Edit=>Object Select**. Double clicking on the Contour object displays the **Contour Attributes Interface** as shown in [Figure 4-21](#).

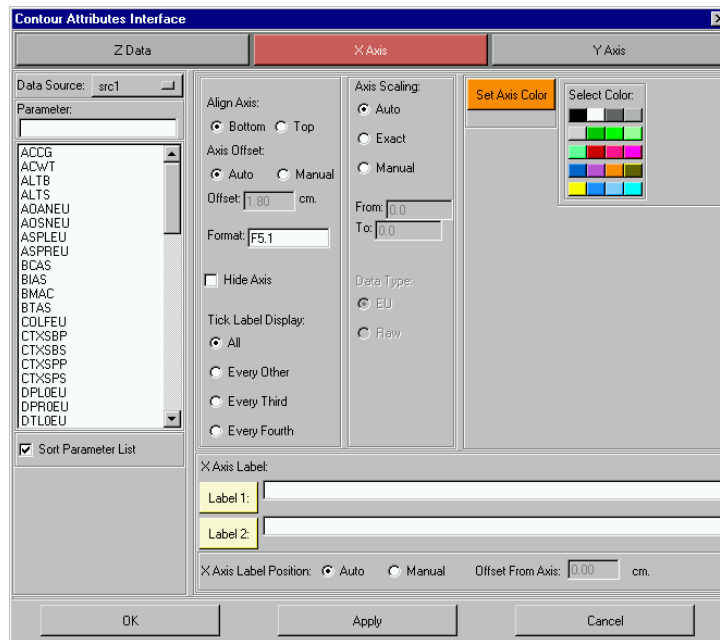
You must associate a parameter with each of the three axes. Associate data with **Z Data** by selecting the dependent parameter name from the Parameter List. Select the **X Axis** button to display the X axis attributes. Associate data with the X axis by selecting an independent parameter name from the Parameter List. Select the **Y Axis** button to display the Y axis attributes. Associate data with the **Y Axis** by selecting a second independent parameter name from the Parameter List. Click **OK** to dismiss the **Contour Attributes Interface**.

### *Contour Attributes Interface*

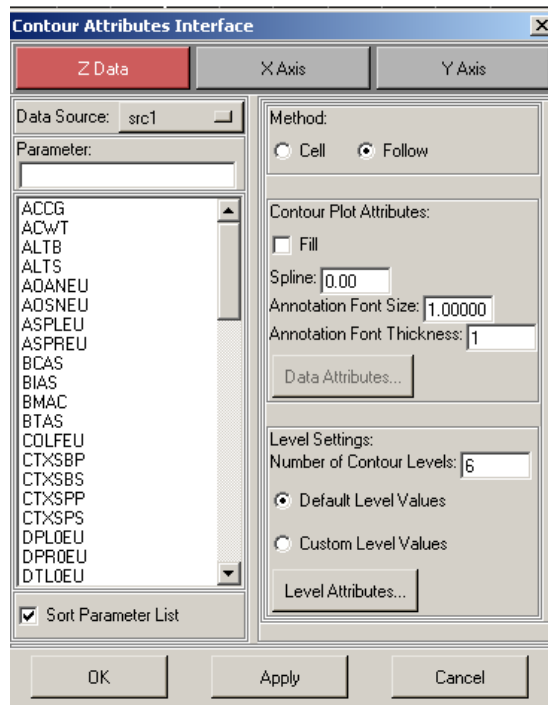
**Z Data** — The Z data parameter is the array of values that will make up a contour surface. Select a parameter from the Parameter List.

**X Axis** — The X axis parameter is the array of values that will be used to plot the X coordinates for the contour surface. Select a parameter from the Parameter List to assign the X data.

**Y Axis** — The Y axis parameter is the array of values that will be used to plot the Y coordinates for the contour surface. Select a parameter from the Parameter List to assign the Y data.



**Figure 4-20** The Contour Attributes interface with the X axis selected



**Figure 4-21** The Contour Attributes Interface with the Z Axis selected

**Data Source** — Use this drop-down menu to select from the list of open Source IDs. The Parameter List is updated to reflect the available parameters for the Source ID you select.

**Parameter List** — Click on a parameter name to select it for plotting on the currently selected axis

**Method** — Selects the method that will be used to create the contour lines. Although these two methods both draw correct contour maps, differences in their algorithms can cause small differences in the resulting plot.

- **Cell** — Examines each array cell and draws all contours emanating from that cell before proceeding to the next cell. This method is efficient in terms of computing resources but does not allow contour labeling.
- **Follow** — Forces the contour procedure to use the line-following method that searches for each contour line and then follows the line until it reaches a bound-

ary or closes. This method gives better looking results with dashed linestyles, and allows contour labeling, but requires more computing time.

### *Contour Plot Attributes*

- **Fill** — Fill the Contour objects with colors defined in the **Level Attributes** dialog. This attribute is available only for the Follow Method.
- **Spline** — Specifies that contour paths are to be interpolated using cubic splines. The appearance of contour plots of arrays with fewer data points may be improved by using spline interpolation. In rare cases, contour lines that are close together may cross because of interpolation. Splines are especially useful with small data sets (less than 15 array dimensions). With larger data sets the smoothing is not as noticeable and the expense of splines increases rapidly with the number of data points. This attribute is available only for the Follow Method. The Spline value specifies the length of each interpolated line segment in normalized coordinates. The default value is 0.005. Smaller values for this parameter yield smoother lines, up to the resolution of the output device, at the expense of more computations.

The Spline attribute is available only for the Follow Method.

- **Annotation Font Size** — Sets the overall character size for contour annotation. This attribute is available only for the Follow Method.
- **Annotation Font Thickness** — Sets the thickness of characters drawn. This attribute is available only for the Follow Method.
- **Data Attributes** — Brings up the **Contour Data Attributes** dialog as shown in [Figure 4-22](#). Your original one-dimensional data is gridded into three two-dimensional arrays before contouring occurs using PV-WAVE's GRIDN function. The attributes controlled through this dialog are parameters used for the GRIDN call. For more information on GRIDN, refer to the *P.V.-WAVE Reference*. Additionally, the **Resample Data** option allows you to resample your original data in order to reduce the total size of your data set for faster contour plotting.
  - **Resample Data** — The value entered here will be the factor by which the total number of data points may be resampled before plotting. For example, If you enter the value 2, your data set will be resampled to be half its original size. The value of this slider ranges from 1 to the size of your Z Parameter. If at least one Contour plot has not been generated, it will range from 1 to 500. The default resample rate is 1.
  - **Weight Order** — A scalar specifying the order of the weighting function. The dependent variable at a point is computed as a weighted average of the

variable over all neighborhood data points. The weighting function is  $1/e^w$  where  $e$  is the Euclidean distance between the grid point and the data point and  $w$  is the Weight Order of the function. This value should be  $0 \leq \text{weight} \leq 5$ . The default value is 2.

- **Neighborhood Size** — A scalar between 0 and 1 specifying neighborhood size. A value of one gives a maximal neighborhood which includes all data points, while lower values yield smaller neighborhoods. This value should be  $0 \leq \text{neighborhood} \leq 1$ . The default value is 1.
- **X Grid Size** — Final resampled dimensions along the X axis of the regridded data. The higher the grid dimensions, the more computing time is required, but the smoother your contour lines will be. The values entered here should be  $0 \leq \text{gridsize} \leq 1000$ .
- **Y Grid Size** — Final resampled dimensions along the Y axis of the regridded data. The higher the grid dimensions, the more computing time is required, but the smoother your contour lines will be. The values entered here should be  $0 \leq \text{gridsize} \leq 1000$ .
- **OK** — Accepts the specified values and closes the dialog.
- **Apply** — Applies the values to the Contour object and redraws the Page, but does not close the dialog.
- **Cancel** — Discards changes and closes the dialog.

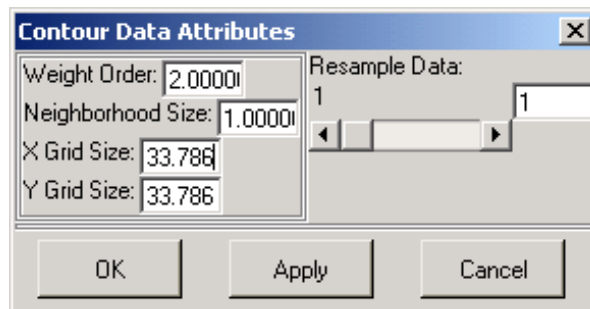


Figure 4-22 The Contour Data Attributes dialog

### Level Settings

- **Number of Contour Levels** — The number of equally-spaced contour levels that are produced. The maximum allowed is 150.
- **Default Level Values** — Calculate default contour level values. These values can be viewed in the **Level Attributes** dialog by first generating a plot using **Default Level Values**, then selecting **Custom Level Values** and clicking on the **Level Attributes** button.
- **Custom Level Values** — Assign custom level values for each contour. Values can be assigned by clicking on the **Level Attributes** button.
- **Level Attributes Button** — Brings up the **Contour Level Attributes** dialog.

### Contour Level Attributes Interface

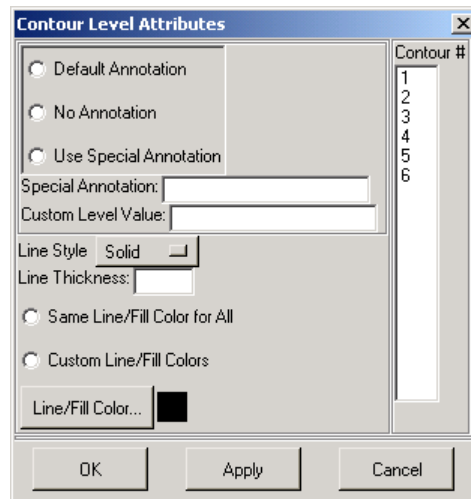


Figure 4-23 Contour Level Attributes interface

**Contour #** — The attributes at the left will be applied to the contour level selected in the ‘Contour #’ column. The number of levels that appear in this list are defined by the **Number of Contour Levels** field in the **Contour Attributes Interface**.

**Default Annotation** — Labels the selected contour level with its Z value.

**No Annotation** — Do not label the selected contour level.

**Use Special Annotation** — Labels the selected contour level using the string defined in the **Special Annotation** field.



**Special Annotation** — The string entered in this field will be used to label the selected contour level when **Use Special Annotation** is selected.

**Custom Level Value** — Specifies the contour level value assigned to the selected contour level. This field is available when **Custom Level Values** is selected in the **Contour Attributes Interface** dialog.

**Line Style** — Specifies the line style to use for the selected contour level.

**Line Thickness** — Specifies the line thickness of lines used to draw the selected contour level.

**Same Line/Fill Color for All** — When selected, all lines or fill colors will be set to the same color. You must select a contour level before selecting **Same Line/Fill Color for All**.

**Custom Line/Fill Colors** — Allows you to individually select a color for each contour level.

**Fill/Line Color** — Defines the color used to draw the selected contour level. To select a color, either click on a color or type in the color index, from 0-19, then <Return>.

**OK** — Accepts the specified values and closes the dialog.

**Apply** — Applies the values to the Contour object and redraws the Page, but does not close the dialog.

**Cancel** — Discards changes and closes the dialog.

## Text Object

Adds words or phrases to your TS-WAVE Page.

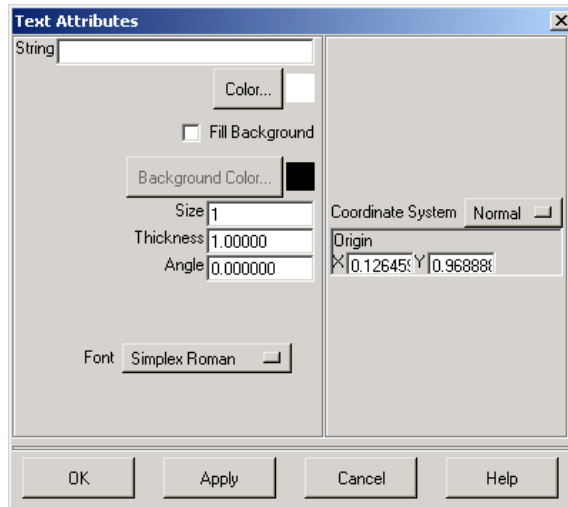


Figure 4-24 The Text Attributes dialog

The general steps for adding a Text object to the Page includes these steps:

1. Select **Create=>Text Object**.
2. Click where you want the text to start. The **Text Attributes** dialog appears.
3. Enter the text and, optionally, specify text characteristics (font, color, and so on) in the **Text Attributes** dialog.
4. Click **OK** when you are finished entering the text. You can continue to create Text objects until you make another menu selection.
5. To resize or move your Text object, use the object select mode (**Edit=>Object Select**).
6. To edit the Text object, select **Edit=>Object Select**, then double click on the Text object to bring up the **Text Attributes** dialog.

### *Text Attributes Dialog*

**String** — The text that is displayed.

**Color** — The color of the text.

**Fill Background** — Fills in the background of the text box with the Background Color.

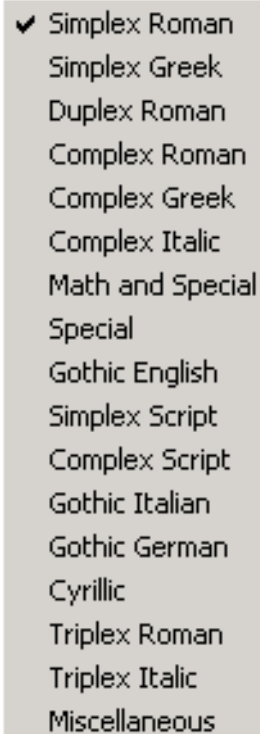
**Background Color** — Available if Fill Background is selected. Use this field to select the background fill color.

**Size** — Sets the character size. The default size is 1.0. For other sizes, the text is scaled relative to the normal size. For example, a size of 2 produces text twice as large as 1.

**Thickness** — Defines the thickness of the characters. The default thickness is 1.0, 2.0 is double-wide and so on.

**Angle** — Sets the number of degrees to rotate the text.

**Font** — The available fonts are:



- ✓ Simplex Roman
- Simplex Greek
- Duplex Roman
- Complex Roman
- Complex Greek
- Complex Italic
- Math and Special
- Special
- Gothic English
- Simplex Script
- Complex Script
- Gothic Italian
- Gothic German
- Cyrillic
- Triplex Roman
- Triplex Italic
- Miscellaneous

The software character set for each of these fonts is documented in Chapter 5 of the *PV-WAVE Reference*, Volume I-Z. When changing font types, the font specifier will be added into the text at the location of the text cursor.

**Coordinate System** — The object's coordinates can be defined in data or normal coordinate systems. The default coordinate system is normal. The data coordinate system is not recommended.

- **Normal** — The coordinate system of the TS-WAVE page ranges from (0.0,0.0) in the lower-left corner to (1.0,1.0) in the upper right corner.

**Origin X & Y** — Enter your values in normalized coordinates.

**OK** — Accepts the specified values and closes the dialog.

**Apply** — Applies the selected attributes but does not exit the dialog.

**Cancel** — Discards changes and closes the dialog.

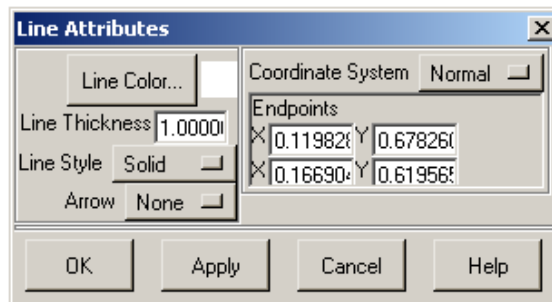
**Help** — Provides help on the **Text Attributes** dialog.

### **Line Object**

Adds lines to emphasize a particular value or to visually connect text to some feature of interest.

The general procedure for adding lines to the Page includes these steps:

1. Select **Create=>Line Object**.
2. Use the mouse to draw the line. Do this by pressing the left mouse button and dragging the pointer. When you release the mouse button, the line is fixed and you can draw another line if you wish. You can continue to draw lines until you make another menu selection.
3. To resize or move a Line object, select **Edit=>Object Select** then resize or move the Line object.
4. To edit the Line object (color, width, etc.), select **Edit=>Object Select**, then double click on the Line object to bring up the **Line Attributes** dialog.



**Figure 4-25** The Line Attributes dialog

### *Line Attributes Dialog*

**Line Color** — Brings up the **Line Color** dialog. Use this dialog to choose a color for the selected lines.

**Line Thickness** — Enter a value for the line thickness. A value of 1 is the default, 2 is double-wide, and so on.

**Line Style** — Select one of the standard PV-WAVE linestyles from the option menu.

**Arrow** — Select an arrow from the option menu. The arrow you select will be used to terminate line segments. The size of the arrowhead is proportional to the width of the line.

**Coordinate System** — The object's coordinates can be defined in data or normal coordinate systems. The default coordinate system is normal. The data coordinate system is not recommended.

- **Normal** — The coordinate system of the TS-WAVE Page ranges from (0.0,0.0) in the lower-left corner to (1.0,1.0) in the upper right corner.

**Endpoints** — Lets you enter specific endpoints for the selected line segment.

**OK** — Saves the specified values and closes the dialog.

**Apply** — Saves the selected attributes but does not exit the dialog.

**Cancel** — Discards unsaved changes and closes the dialog.

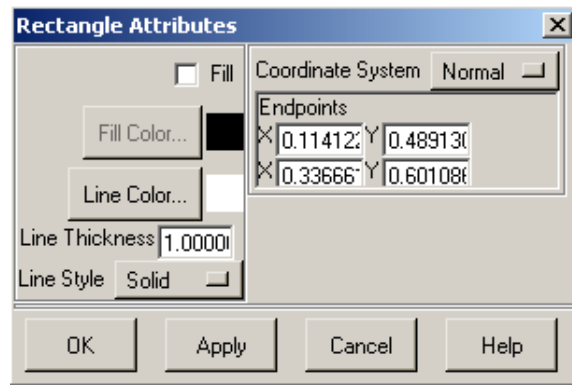
**Help** — Provides help on the **Line Attributes** dialog

### ***Box Object***

Adds a rectangle to your TS-WAVE Page.

The general steps for adding rectangles to the Page are:

1. Select **Create=>Box Object**.
2. Use the mouse to draw the box. Do this by pressing the left mouse button and dragging the pointer. When you release the mouse button, the box is fixed. You can continue to draw boxes until you make another menu selection.
3. To resize or move a Box object, use the object select mode (**Edit=>Object Select**)
4. To edit the Box object (color, line width, etc.), select **Edit=>Object Select**, then double click on the Box object to bring up the **Rectangle Attributes** dialog.



**Figure 4-26** The Rectangle Attributes dialog

### *Rectangle Attributes Dialog*

**Fill** — When this button is selected, the selected rectangle is filled with the color selected with the Fill Color option.

**Fill Color** — Brings up the **Fill Color** dialog. Use this dialog to select a fill color for the selected rectangle. This option is greyed out unless the Fill button is selected.

**Line Color** - Brings up the **Line Color** dialog. Use this dialog to select a color for selected rectangle.

**Line Thickness** — Enter a value for the line thickness. A value of 1 is the default, 2 is double-wide, and so on.

**Line Style** — Allows you to select one of the standard PV-WAVE linestyles from the option menu.

**Coordinate System** — The object's coordinates can be defined in data or normal coordinate systems. The default coordinate system is normal. The data coordinate is not recommended.

- **Normal** — The coordinate system of the TS-WAVE page ranges from (0.0,0.0) in the lower-left corner to (1.0,1.0) in the upper right corner.

**Endpoints** — Lets you enter specific endpoints for the selected line segment.

**OK** — Saves the specified values and closes the dialog.

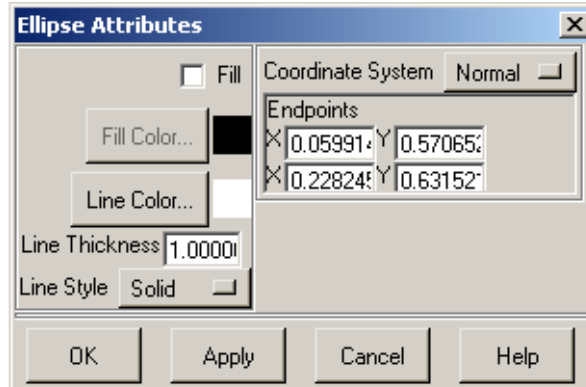
**Apply** — Saves the selected attributes but does not exit the dialog.

**Cancel** — Discards unsaved changes and closes the dialog.

**Help** — Provides help on the **Rectangle Attributes** dialog

### ***Ellipse Object***

Adds a circle or ellipse to your TS-WAVE Page.



**Figure 4-27** The Ellipse Attributes dialog

The general steps for adding an ellipse to the Page are:

1. Select **Create=>Ellipse Object**.
2. Use the mouse to draw the ellipse. Do this by pressing the left mouse button and dragging the pointer. As you drag the pointer, a box-shape circumscribes the ellipse. When you release the mouse button, the ellipse is fixed. You can continue to draw circles until you make another menu selection.

#### *Ellipse Attributes Dialog*

**Fill** — When this button is selected, circles are filled with the color selected with the Fill Color option.

**Fill Color** — Brings up the **Fill Color** dialog. Use this dialog box to select a fill color for the selected ellipse. This option is greyed out unless the **Fill** button is selected.

**Line Color** — Brings up the **Line Color** dialog. Use this dialog box to select a color for the selected ellipse.

**Line Thickness** — Enter a value for the line thickness. A value of 1 is the default, 2 is double-wide, and so on.

**Line Style** — Select one of the standard PV-WAVE linestyles from the option menu.

**Coordinate System** — The object's coordinates can be defined in data or normal coordinate systems. The default coordinate system is normal. The data coordinate system is not recommended.

- **Normal** — The coordinate system of the TS-WAVE page ranges from (0.0,0.0) in the lower-left corner to (1.0,1.0) in the upper right corner.

**Endpoints** — Lets you enter specific endpoints for the selected ellipse.

**OK** — Saves the specified values and closes the dialog.

**Apply** — Saves the selected attributes but does not exit the dialog.

**Cancel** — Discards unsaved changes and closes the dialog.

**Help** — Provides help on the **Ellipse Attributes** dialog.



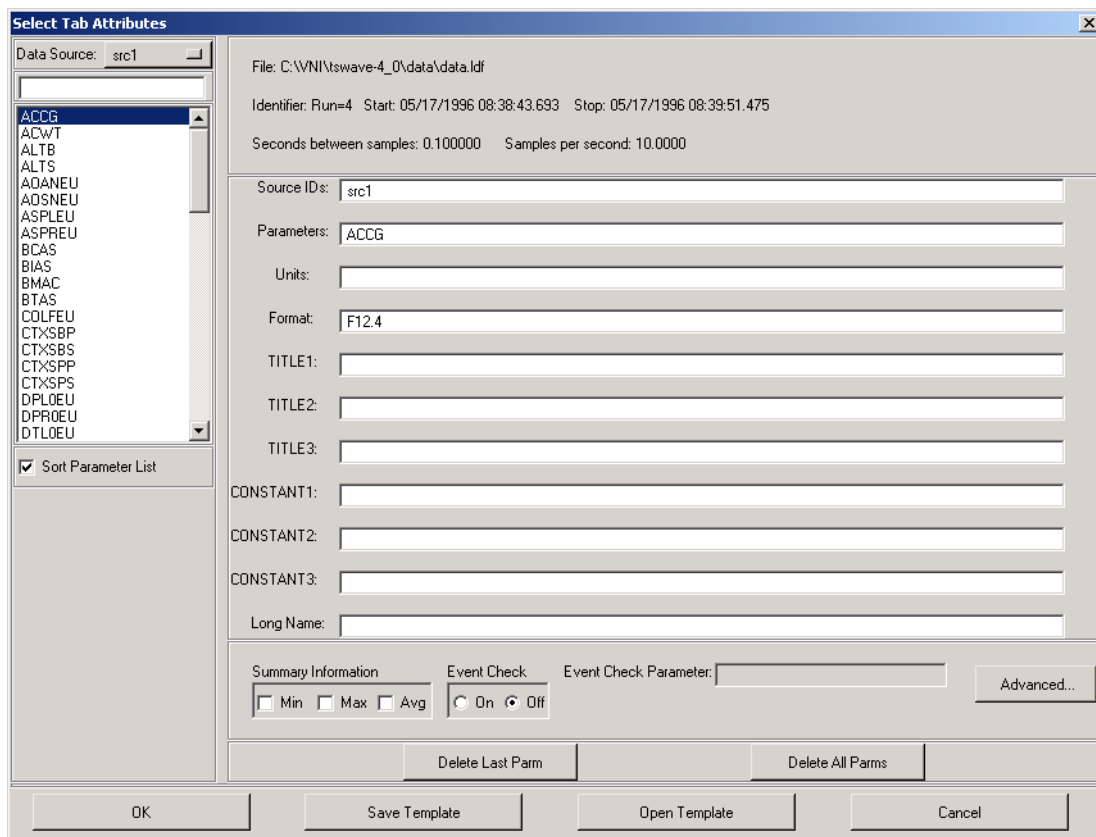
## Create TabData

A Tab File is an output file that contains may contain an entire data file or a subset of the currently loaded data file(s). Tab Files are tabulary arranged ASCII files that consist of header information and columns of data as shown in [Figure 4-28](#).

```
Created: 04/08/2004 11:43:40.000 v 4.00 C:\VNI\tswave-4_0\tab\tabfile.tab
Data Source: C:\VNI\tswave-4_0\data\data.ldf
Start: 05/17/1996 08:35:45.539 Stop: 05/17/1996 08:36:52.822
Identifier: Run=3 Sample Rate: 10.0
Title 1: A Title
Title 2: Another Title
Constant 1: A Constant Value
Constant 2: Another Constant Value
Long: Long_name_1 Long_name_2
Units: Feet Meters
SourceID: Tab_src1 Tab_src1
Param: COLFEU DPL0EU
05/17/1996 08:35:45.539 -6.1197 0.5789
05/17/1996 08:35:45.639 -7.0152 0.5802
05/17/1996 08:35:45.738 -7.3137 0.5802
05/17/1996 08:35:45.838 -7.6122 0.5802
05/17/1996 08:35:45.938 -8.8064 0.5789
05/17/1996 08:35:46.039 -9.1049 0.5815
05/17/1996 08:35:46.139 -8.2093 0.5789
```

**Figure 4-28** A Tab Data file generated by TS-WAVE.

To create a Tab File you must have at least one open Source ID. Select **Create=>Create TabData**. A **Create Tab Data File** dialog appears for file selection before the **Select Tab Attributes** dialog appears as shown in [Figure 4-29](#). Tab Files use the .tab extension and are placed in the <tswave\_dir>/tab directory by default.



**Figure 4-29** The Select Tab Attributes dialog

The left side of the dialog displays Source ID and parameter information for the current session. The right side shows the information that will be written to the Tab File. Use the **Data Source** option menu to select the Source ID you wish to use. The parameters available for that Source ID appear in the Parameter List. When you single click on a parameter name in the Parameter List, the Source ID, parameter name and the output format specifier for that parameter appear in the fields to the right.

### *Select Tab Attributes Dialog*

**Data Source** — An option menu showing the Source IDs of the open Data Sources.

**Parameter List** — This list shows all of the available parameters for the selected Source ID. When you click on a parameter all of the required fields of the interface are automatically filled in: its name appears in the **Parameters** field to the right, its format code appears in the **Format** field and its Source ID appears in the **Source IDs** field.

**Sort Parameter List** — When this checkbox is selected the Parameter List is sorted alphabetically. When unselected the Parameter List displays parameters in the order that they occur in the data file.

**Source IDs** — **[Required]** The Source IDs of the selected parameters. All selected parameters are grouped together by Source ID in the output file. Except in special cases, DERIVED parameters are grouped with the Source ID of the parameter from which each was derived. Source IDs appear at the top of each data column.

**Parameters** — **[Required]** The parameters that are to be written to the tabular data file. To add a parameter to this field, click on its name in the **Parameter List**. Each parameter is written to the file in the order it is selected, within the Source ID group. Parameter names appear at the top of each data column

**Units** — The units for each selected parameter, separated by spaces. Each separate unit notation appears above its corresponding column of data in the Tab File. The **Units** field is optional. If you choose to use the **Units** field, the number of elements must match that of the required field.

**Format** — **[Required]** The format code for each selected parameter. When a parameter is selected the default format code is automatically entered in this field. The default code is specified by the `Tab_DataFormat` resource in the resource file `<tswave_dir>/resource/tswave_files.ads`. The default code is F12.4 and may be modified by either changing this resource or changing it directly in the **Format** field for individual parameters. For more information on FORTRAN format codes see, on page 172.

**Title** — Use these text fields to add titles to the Tab File header. The number of available titles is determined by the `Tab_TitleLines` resource in the resource file `tswave_files.ads`. The default is 3.

**Constant** — Use these text fields to add constant values to the Tab File header. The number of available constants mined by the `Tab_ConstantLines` resource in the resource file `tswave_files.ads`. The default is 3.

**Long Name** — Lets you add descriptive, parameter specific text above each parameter's column of data in the Tab File header. The **Long Name** field is

optional. If you choose to use the **Long Name** field, the number of elements must match that of the required fields.

---

**NOTE** The length of the printed values that appear at the top of each data column is dictated by the format code for that column. Values longer than the length permitted by the format code will be truncated.

---

**Summary Information** — Lets you select the type of summary information you wish to appear at the end of each parameter's output column. The choices are **Min**, **Max**, and **Avg**.

**Event Check** — Turns the event checking mechanism on or off. Filters out all values from the selected parameters for timestamps where the Event Check Parameter does not have a value of '1'. This is useful when you are interested in the values of certain parameters when a certain event is taking place.

**Event Check Parameter** — Use this field to enter the name of the parameter that is used to signal the state of an event. The name you enter is case sensitive. If **Event Check** is turned on, data is written to the tabular file whenever this event parameter has the value '1'.

For Source IDs containing parameters sampled at different rates, the event is considered active until the event check parameter transitions to a non '1' value. For example, a data parameter is sampled ten times per second and the event check parameter is only sampled once per second. If the event parameter is 1 at 10 seconds and 0 at 11 seconds, the values of the output parameter between 10 and 11 seconds will be printed.

**Delete Last Parm** — Deletes the Source IDs, Parameter name, Unit, Format and Long Name for the last parameter listed in the **Parameters** fields.

**Delete All Parms** — Clears the **Source ID**, **Parameters**, **Unit**, **Format**, and **Long Names** fields.

**OK** — Saves the selected attributes, closes the dialog and writes the file.

**Save Template** — Creates a Tab Template File. All of the current tab attributes are saved.

**Open Template** — Opens an existing Tab Template File and populates the open **Select Tab Attributes** dialog with the stored information. Upon selecting **Open Template** the file selection dialog appears.

**Cancel** — Discards changes and closes the dialog. Output is not written to the file.

**Advanced** — Brings up the **Select Tab Attributes Advanced Settings** dialog.

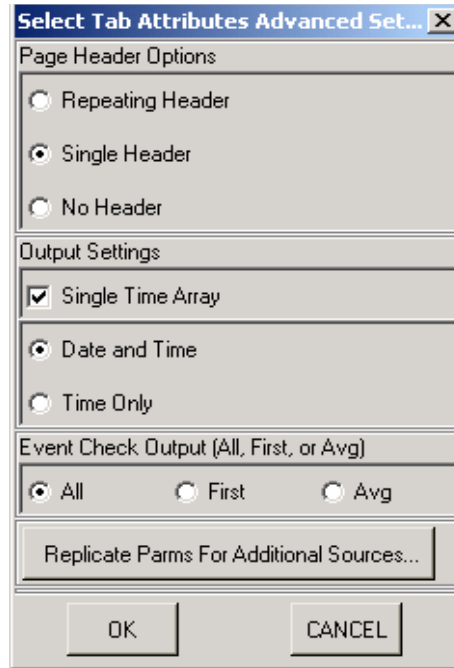


Figure 4-30 Select Tab Attributes Advanced Settings dialog

**Page Header Options** — These options determine how the header block is written in the resulting Tab File.

- **Repeating Header** — A Tab File header is printed at the top of each output Page. Page counts and page breaks are determined by the Tab\_PageLines resource in the file <tswave\_dir>/resource/tswave\_files.ads.
- **Single Header** — A Tab header is printed only at the beginning of each Source ID's section. Page count is determined by the Tab\_PageLines\_No\_Header resource in <tswave\_dir>/resource/tswave\_files.ads.
- **No Header** — No header information is printed in the file. Page count is determined by the Tab\_PageLines\_No\_Header resource in <tswave\_dir>/resource/tswave\_files.ads.

### **Output Settings**

- **Single Time Array** — When this checkbox is enabled, only the first column of the Tab File will show the timestamps for the data. When it is unchecked, each column of data will be preceded by its own column of timestamps.

- **Date and Time vs. Time Only** — These buttons determine if timestamps should show both the date and the time, or only the time.

Default values for the **Page Header Options** and **Output Settings** are resources in the file `<tswave_dir>/resource/tswave_files.ads`.

**Event Check Output** — There are three buttons that are used to determine how output that is filtered by an Event Check parameter should be displayed. The buttons are:

- **All** — Prints all points during the event.
- **First** — Prints only the first datapoint as the event first becomes active.
- **Avg** — Prints only the average of the datapoints while the event is active.

**Replicate Parms For Additional Sources** — This button allows you to duplicate current parameter selections and settings for other Source IDs. For example, you have four different Source IDs open and want the same parameters from each to be output to the Tab File. Select all of the parameters you want from a single Source ID, fill in or modify any additional information for the parameters (Units, Formats, Long Names, etc.) and select **Advanced...=> Replicate Parms For Additional Sources...** A list of the other open Source IDs appears. Select the Source ID(s) for which you want the current settings replicated. The additional parameters and settings appear in the **Tab Attributes** dialog.

## **Tab File Tips**

### *Output Performance*

Files created with a ‘Single’ or ‘No Header’ are written out much more quickly than files with repeating headers and may be more suitable for importation into another application.

### *Duplicate Timestamps*

All timestamp/data pairs for the selected parameters from a given Data Source are written to the file. Duplicate timestamps are not filtered out.

### *Output From Sparse Data Sources*

Output for parameters from a single Data Source that are sampled at different rates will appear slightly different from the normal Tab output. A value will be printed for each parameter at each timestamp in the selection. If a particular parameter does not have data at a particular timestamp, the value of the system variable `!INVALID` will be printed instead. The value of `!INVALID` is controlled by the resource `Invalid_Value` in the file `resource/ep_graphattribute.ads`.

### *Non-Numeric Characters*

After you create a Tab File, you should quickly scan it for Non-Numeric characters. Most likely, these will be asterisks placed into your data columns to indicate that the value it attempted to print out was too long for the given format code. Reading in files with these characters will produce an error at your command line or in your tswave.log file. The data for that point will be set to zero. If your source datafile contains sparse data, make sure your format specifier can accommodate your !INVALID value.

### *Tab Files and Source IDs*

When a Tab File is read in, the data in the file is associated with the source ID specified in each section's header. This means that if a Source ID in your session is the same as that in a Tab File, reading in that Tab File will replace that Source ID in your session. There is a resource in `tswave_files.ads` called `Tab_SourceName_Prefix`. The value of this resource will be prepended to the Source IDs in the Tab header as it is written to the Tab File to reduce conflicts. Its default is 'Tab\_', so the Source ID in the Tab File for data written from 'src1' will be 'Tab\_src1'. That will be the Data Source created when the Tab File is read in. This resource can be left blank, if you desire.

When a Tab File is read, it looks to the column headers for information about the source file ID and the parameter names. These column headers are subject to the same format specifiers as the data in that column. If they are longer than the format specifier allows they are truncated. You should avoid using format specifiers that are short enough to cause this header information to be truncated. When you select **File=>Open** to read a Tab File, you are prompted for a Source ID to assign to that file. This value is ignored and the Source ID(s) written in the Tab File itself are created in your session. There is currently no way to re-map the Source ID(s) written in the Tab File. 'DERIVED' data types are grouped with the Source ID of the first parameter in the DERIVED parameter's name string. Once a formerly DERIVED parameter is read from a Tab File, its value is static.

### *Tab File Headers*

If you create a Tab File with the 'No Header' option selected from the **Advanced Attributes** dialog, you will not be able to read that Tab File back into TS-WAVE via the TS-WAVE Tab reader. The reader relies on the Source ID and parameter lines in the header to correctly set up the data in your session. The 'No Header' option should be used exclusively to create output to be read in by another application that would have trouble dealing with the Tab headers. If you wish to read a Tab File that has no headers into TS-WAVE, read it in as an ASCII file. You will lose the parameter names, but the data is there.

### *Tab Files and Derived Data*

If a Tab File contains a DERIVED parameter, once it is read back in the parameter will be static with its values fixed to those in the Tab File. Output from user or standard functions need to be handled specially if the function replaced the X values with non-time or non-index values, an example of this is the output from the FFT standard function. FFT parameters are each written separately to the Tab File under the unique Source ID 'FFT\_N', where N is a unique number. DIFFERENCE parameters that are derived from two different Data Sources are all written together under the Source ID 'INDEXED', since their X values are replaced by index values.

### **View TabData File**

This option allows you to view a Tab File created by TS-WAVE. The file is opened using the default application as specified by the `Windows_Tab_Viewer` and `Unix_Tab_Viewer` resources, depending on your current platform, in the file `<tswave_dir>/resource/tswave_files.ads`. The default application for Windows is `write.exe`. The default Unix resource is left blank and defaults to a PV-WAVE text widget.

---

**NOTE** The default **View TabData** text window may use proportional fonts, which can cause some irregularity in the alignment of data in the columns. When you print your data file, the columns will be aligned properly. To see properly aligned columns on your screen, you must use a text editor that is set to use a nonproportionally spaced font.

---

### **Open Pick File**

A Pick File is a Tab File that contains data points at selected times or within selected time ranges. The points and ranges are selected graphically using your mouse in the **Pick File** interface.

In order to create a Pick File, you must have a Source ID open and one or more Graph objects created in one of your TS-WAVE Pages. If a Pick File is not already open, you will be prompted to select a file name. Pick Files use the `.tab` extension and are placed in the `<tswave_dir>/tab` directory by default.

When you choose **Create=>Open Pick File** the **Select Tab Attributes** dialog opens. The parameters present in the selected Graph object are already entered on the **Parameters** line of the **Select Tab Attributes** dialog. In this dialog you can select additional parameters from the Source ID of the selected Graph object.



These parameters will not appear in the **Pick Data** interface, but any selections made are applied to them as well and output to the Pick File.

For additional information about the **Select Tab Attributes** dialog, see the discussion earlier in this section.

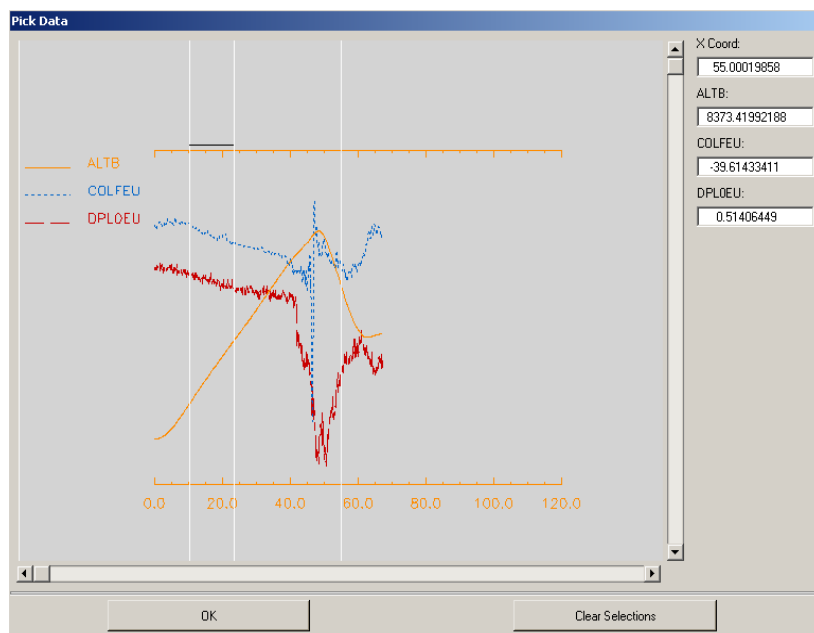
---

**NOTE** If your Graph object contains multiple parameters you will be prompted to pick a Master Axis. This axis will determine the range, units and selection coordinates in the **Pick Data** interface. Only this X axis will be displayed in the interface. Graph objects containing data from multiple Source IDs will not be accepted. Each data selection operation occurs entirely within the context of a single Source ID. You may add data from any open Source ID to the Pick File by repeating the data selection on a Graph object containing only data from that Source ID.

---

Until you select **Create=> Close Pick File**, all selections will be added to the open file.

**Create=>Close Pick File** will only be enabled when a Pick File is already open. Once you select **OK** in the **Select Tab Attributes** dialog, The **Pick Data** interface appears.



**Figure 4-31** The Pick Data dialog displaying a selected range

A vertical line is displayed beneath the cursor. As you move the cursor over the plot, the data coordinates of the X axis and each displayed parameter are shown on the right side of the interface. To select individual points, click and release mouse button 1 (MB1). A vertical line remains at the point you selected. To select a range of points, click and hold MB1 and drag the cursor across the graph. A line will appear where you first press MB1 and another where you release it. A horizontal line at the top of the graph connects the endpoints of each selected region.

Once a selection is made, the **Clear Selections** button will be enabled. Pressing this button removes all active selections. Pressing **OK** stores your selections for that Source ID. Repeated operations on the same Source ID deletes any previous selections for that Source ID. To write data from multiple Source IDs to the same output file, select a Graph object containing parameters from a different Source ID and select **Create=>Open Pick File**. Your new selections are appended to the currently open output file. If you press **OK** in the **Pick Data** interface with no selections present, no data will be written for that Source ID. Selections made on other Source IDs will still be written to the file when it is closed.

### ***Close Pick File***

Closes the Pick File and writes the selected data to the output Tab File.

## **Batch Processing**

The **Open Batch File**, **Add to Batch File** and **Close Batch File** menu items refer to the steps required for Batch processing.

Batch processing is a feature that allows TS-WAVE to be run as a background process without a graphical user interface. Batch Processing is responsible for automatically loading templates and data files, printing the generated TS-WAVE Page(s) and creating Tab File(s).

The steps for creating a Batch Job are:

- Step 1** Define a Source ID and design your TS-WAVE Page(s).
- Step 2** Insure that your printer is set to the correct printer and settings.
- Step 3** Open a Batch Job by selecting **Create->Open Batch File** and naming your Batch Job.
- Step 4** Add your template(s) to the Batch Job using **Create->Add to Batch File=>Add Template to Batch**.

**Step 5** Close your Batch Job by selecting **Create->Close Batch File**.

**Step 6** Execute the Batch Shell script.

Details on each step are described under each of the appropriate menu descriptions below.

### ***Open Batch File***

Selecting **Open Batch File** brings up a standard file naming tool where you specify a name and directory for the Batch Job file. By default this file is created in the <tswave\_dir>/batch directory and has a .job extension. Once the Batch Job is open the **Add to Batch File** and **Close Batch File** menu items are active.

### ***Add to Batch File***

Allows you to add one or more TS-WAVE Templates or Tab Templates to the Batch Job.

**Add Template to Batch** — Adds a TS-WAVE Template File for the open Batch Job. By default this Template File is placed in the <tswave\_dir>/batch directory.

**Add Multi Template to Batch** — Adds a TS-WAVE Template File for the open Batch Job and uses this template as a master Template to be applied to each Source ID selected in the Add Multi-Template to Batch dialog that appears when this menu item is selected. Batch entries will be automatically generated for each selected Source ID for use with the master template. By default this Template File is placed in the <tswave\_dir>/batch directory.

This option is active when both of the following conditions are met:

- Your Page contains one or more Graph objects, Contour objects or Header objects.
- Only one Source ID is being referenced by all objects in the current session

The **Add Multi Template to Batch** dialog displays only the Source IDs that contain all the Parameters referenced by the objects displayed in your current session.

Batch entries are automatically generated for each selected Source ID for use with the master template.

**Add Tab Template to Batch** — Adds a Tab Template for the open Batch Job. Brings up the Tab Attributes dialog where you can set attributes for the Tab File to be generated when the Batch Job is executed. By default this Template File is

placed in the <tswave\_dir>/batch directory. The output Tab File will be written to the <tswave\_dir>/tab directory.

**Add Tab Multi-Template to Batch** — Adds a Tab Template for the open Batch Job and uses this template as a master Tab Template to be applied to each source selected in the **Add Tab Multi-Template to Batch** dialog that appears when this menu item is selected. Batch Tab Files will be generated for each selected Source ID using the master Tab Template.

This feature is valid only when your Tab Template references just one Source ID. All Parameters referenced in the tab template must be valid for each of the Source IDs selected. By default this is placed in the <tswave\_dir>/batch directory. The output Tab Files will be written to the <tswave\_dir>/tab directory.

### ***Close Batch File***

Saves the Batch .job File.

After closing the Batch File, the TS-WAVE Batch process will have created a Batch Job file, Template File, and a Batch shell script. By default, all of these files are saved in the <tswave\_dir>/batch directory.

### ***The Batch Job (\*.job) File***

This is a text file with a .job extension that uses the TS-WAVE Batch command language to define printers, data files, templates, and other information specific to your Batch Job.

### ***Example .job file***

In this example .job file, <tswave\_dir> is c:\VNI\tswave-4\_0. This Batch Job was created from a TS-WAVE session that used an ldf data file and a PostScript printer named 'PS3':

```
BEGIN
 PRINTER =PS3
 PRINT_DRIVER =PS
 PRINT_TO_FILE=<Not Set>
 Time_Start=05/17/1996 08:31:09.859
 Time_Stop =05/17/1996 08:44:26.506
 Source=src1
 Run=1
 Desample=90
 Data_Source=c:\VNI\tswave-4_0\data\data.ldf
 DataHandler=ldf
 Template=c:\VNI\tswave-4_0\batch\my_batch.tpl
END
```

### ***The Batch Template File (\*.tpl)***

A TS-WAVE Template File is created automatically with the same name as the Batch Job file but with a .tpl extension. The Template File contains information about the Page configuration (including Graph objects, Header objects, and SourceIDs). Each time the **Create=>Add To Batch File=>Add Template To Batch** menu item is selected a new Template File is created.

### ***The Batch Shell Script (.bat for Windows, .sh for Unix)***

A BatchShell script is created automatically with the same name as the Batch Job but with a .bat extension for Windows and a .sh extension for Unix. The shell script invokes TS-WAVE in Batch mode with the specified Batch Job file and creates a log file to which messages generated from executing the Batch Shell script are printed.

#### **Example .bat file**

```
cd <tswave_dir>
..\wave\bin\bin.i386nt\tswave -j <tswave_dir>\batch\a.job ^
-l <tswave_dir>\tswavebatch.log
```

### **Batch Print-to-File Settings**

The **File=>Print** dialog contains a **Print to file** setting and a **File** text box. If **Print to file** is set when a TS-WAVE Template is added to your Batch Job, the graphical output goes to the file specified in the **File** text box. Each Batch Job file writes all of its output to a single PostScript file unless you change the output filename in the **Print Setup dialog** between adding templates with the exception of EMF and CGM, which are described below.

**EMF** (Windows only) and **CGM** — The output for each template you add to the batch file must be written to a separate file. You can use the **Print Setup** dialog to change the output file name before adding each template. If you use the same filename for more than one template, TS-WAVE will append the incremented template number to each as shown below:

```
<FileName>_<Template_Number>.<Driver_Name>
```

Templates containing multiple TS-WAVE Pages will also have the Page number appended:

```
<FileName>_<Template_Number>_<Page_Number>.<Driver_Name>
```

## Executing the Batch Job

After you add templates and select **Close Batch File**, you can run the Batch Job by executing the shell script that was created by the Batch command using the following:

### On Windows Systems

The newly created Batch Job can be run in one of three ways on a Windows system as listed below:

- Double click on the \*.bat file in the WindowsExplorer
- Enter the job name at the the command line prompt
- Use an automated job scheduler

### On UNIX Systems

The newly created Batch Job can be run in one of the methods listed below:

- Enter the following command at the shell prompt:  
`<tswave_dir>/batch/batch_name.sh`
- Execute the shell script as part of a cron job or other process.

---

**NOTE** The Batch files (.job and shell script) can be modified to change any settings automatically written to them on creation.

---

## A Simple Batch Job Example

The following steps describe how to save a create a Batch Job that prints a Graph object:

- Step 1** Select **File=>Print** and check that your Queue name and Print Driver are valid for your printer. Click **Save Settings** to save your printer settings for Batch.
- Step 2** Open a Data Source. See [Defining a Data Source](#) on page 70.
- Step 3** Plot a data parameter. See [Adding Graph Objects to Your Page](#) on page 26.
- Step 4** Select **Create=>Open Batch File**, then type or select a filename. By default, Batch Files are created in the `<tswave_dir>/batch` directory. Click **OK** to dismiss the **Open Batch File** dialog.

- Step 5** Select **Create=>Add To Batch File=>Add Template To Batch** to add the current Page configuration to the Batch Job.
- Step 6** Select **Create=>Close Batch File** to close and save the Batch Job.
- Step 7** Execute the Batch Shell script. By default, the Batch Shell script is located in the `<tswave_dir>/batch` directory.

The printed output is sent to the printer you selected in **Step 1**.

---

**NOTE** Check `<tswave_dir>/twavebatch.log` file for status messages.

---

## Batch Multi-Template Example

The following steps describe how to repeat a template file for multiple Source IDs using the example data file `<tswave_dir>/data/data.ldf`:

- Step 1** Select **File=>Print** and check that your Queue name and Print Driver are valid for your printer. Click **Save Settings** to save your Printer settings for Batch.
- Step 2** Open the `<tswave_dir>/data/data.ldf` data file. See [Defining a Data Source](#) on page 70. Select the **Read All Runs As Separate Sources** checkbox. This will create one Source ID for each of the five Runs in this data file. Each Source ID contains identical parameters, each with different data.
- Step 3** Select **Create=>Graph Object**. Add to it Graph object parameters from only one Source ID. See [Adding Graph Objects to Your Page](#) on page 26.
- Step 4** Select **Create=>Open Batch File**. TS-WAVE displays the **Open Batch File** dialog. Enter a filename then click **OK**.
- Step 5** Select **Create=>Add To Batch File=>Add Multi-Template To Batch**. TS-WAVE displays the Batch Data Sources dialog. Select one or more additional Source IDs for which to repeat your TS-WAVE Batch Job.
- Step 6** Select **Create=>Close Batch File** to close and save the Batch Job.
- Step 7** Execute the Batch Shell script. By default, the Batch Shell script is located in the `<tswave_dir>/batch` directory.

The printed output is sent to the default printer set up in **Step 1**. A page will be printed that uses the Source ID plotted in the graph object in your session in addition to one page for each of the Source ID(s) you selected in the **Batch Data Sources** dialog in **Step 5**.

## Batch Print-to File Example

The following steps describe how to set your printer name and create a Batch Job print to a file.

- Step 1** Select **File=>Print** and check that your Queue name and Print Driver are valid for your printer. Click the Print to file toggle and enter a filename in the **File:** text field. Click **Save Settings** to save your Printer settings for Batch.
- Step 2** Open a Data Source. See [Defining a Data Source](#) on page 70.
- Step 3** Select **Create=>Graph Object**. See [Adding Graph Objects to Your Page](#) on page 26. Plot a data parameter.
- Step 4** Select **Create=>Open Batch File**. TS-WAVE displays the Open Batch File dialog.
- Step 5** Open a Batch Job then click **OK**.
- Step 6** Select **Create=>Add To Batch File=>Add Template To Batch** to add the current Page configuration to the Batch Job.
- Step 7** Select **Create=>CloseBatch File** to close and save the Batch Job.
- Step 8** Execute the Batch Shell script. The Batch Shell script is located in the same directory as the job file.
- Step 9** Output is sent to the file defined in **Step 1**.

## Adding a Tab Data to a Batch Job Example

The following contents describe how to generate Tab Files from a to Batch Job. See [Working with Tab Files](#) on page 52.

- Step 1** Select **File=>Print** and check that your Queue name and Print Driver are valid for your printer. Click **Save Settings** to save your printer settings.
- Step 2** Open a Data Source. See [Defining a Data Source](#) on page 70.



- Step 3** Select **Create=>Open Batch File**. TS-WAVE displays the **Open Batch File** dialog. Enter a name for your Batch Job then click **OK**.
- Step 4** Select **Create=>Add To Batch File=>Add Tab Template To Batch**. The **Select Tab Attributes** interface will appear. Select one or more parameters in the **Select Tab Attributes** interface.
- Step 5** Select **Create=>Close Batch File** to close and save the Batch Job.
- Step 6** Execute the Batch Shell script. By default, the Batch Shell script is located in the `<tswave_dir>/batch` directory.
- Step 7** The Tab File created in **Step 5** is automatically generated. Since we did not add Graph, Contour, or Header objects to our template, no output is sent to the printer.

## Adding a Tab Multi-Template to Batch Example

The steps below describe how to repeat tab data for multiple Source IDs, each of which contain the same parameters.

- Step 1** Open the `<tswave_dir>/data/data.1df` data file. See [Defining a Data Source](#) on page 70. Select the **Read All Runs As Separate Sources** checkbox. This creates one Source ID for each of the five runs in this data file. Each Source ID contains identical parameters, each with different data.
- Step 2** Select **Create=>Open Batch File**. TS-WAVE displays the **Open Batch File** dialog. Enter a name for your Batch Job then click **OK**.
- Step 3** Select **Create=>Add To Batch File=>Add Tab Multi-Template To Batch** and select an output Tab File.
- Step 4** In the **Select Tab Attributes** interface, select parameters from only one Source ID then click **OK**. TS-WAVE displays the **Batch Data Sources** dialog. Select additional Source IDs to repeat your tab data.
- Step 5** Select **Create=>Close Batch File** to close and save the Batch Job.
- Step 6** Execute the Batch Shell script. The Batch Shell script is located in the same directory as the job file.

The number of generated Tab Files should equal the number of Source IDs added to the Batch Job.

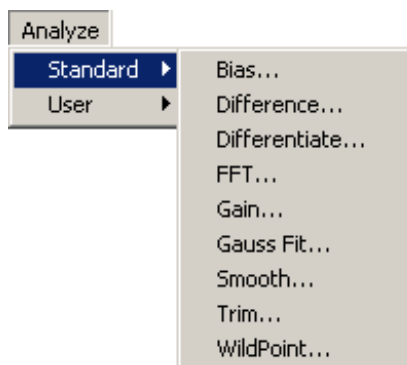
## Adding More than One Template to Batch Example

The following steps describe how to add multiple templates to Batch Processing:

- Step 1** Open a Data Source. See *Defining a Data Source* on page 70.
- Step 2** Select **Create=>Graph Object**. See *Creating a Graph Object* on page 77. Plot a data parameter. See *Selecting the Data Parameters to Plot* on page 27.
- Step 3** Select **Create=>Open Batch File** to open your data file. *Adding Graph Objects to Your Page* on page 26 displays the **Open Batch File** dialog. Enter a name for your Batch Job then click **OK**.
- Step 4** Select **Create=>Add To Batch File=>Add Template To Batch** to add the first template to the Batch Job.
- Step 5** Select **Edit=>Object Select** and double click on the graph object. TS-WAVE brings up the **Graph Attributes Interface**.
- Step 6** Select a different parameter from the Parameter List drop-down menu.
- Step 7** Click **OK**. TS-WAVE displays the modified Graph object.
- Step 8** Select **Create=>Add To Batch File=>Add Template To Batch** to add a second template to the Batch Job.
- Step 9** Select **Create=>Close Batch File** to close and save the Batch Job.
- Step 10** Execute the Batch Shell script. The Batch Shell script is located in the same directory as the job file.

Two pages are sent to the printer. The first page contains just the original Graph object for the first template that was added in **Step 5** and the second page contains the modified Graph object from the second template that was added in **Step 8**.

## Analyze Menu



### Standard Functions

These are the standard analysis functions that are included with TS-WAVE.

- **Bias** — Brings up a dialog where you can apply a bias value to a parameter. The result of the operation is a new parameter with the value you entered in the amount of bias text field added to each value of the parameter's data. The original parameter is unchanged. By default, the new parameter is named:

BIAS( Source ID: parameter name, Bias value)

For example, if your original data values were [10.0, 20.0, 30.0, ...] and the bias value you entered was 5.0, the values in the new parameter are [15.0, 25.0, 35.0, ...]. The new parameter is named:

BIAS(src1:ALTB, 5.00000)

where *src1* is the name of the Source ID, and ALTB is the parameter selected from the **Select Parameter to Bias** dialog, and 5.00000 is the bias value. The new parameter is placed into the DERIVED Source ID.

- **Difference** — Brings up a dialog that lets you compute the difference between two selected parameters. The difference is computed by subtracting the value of the selected parameter in the right column from the value of the selected parameter in the left column. The result of the operation is a new parameter; the original parameter is unchanged. By default, the new parameter is named:

DIFFERENCE(Source ID 1: parameter name 1, Source ID 2:  
parameter name 2)

For example, where *src1* and *src2* are the Source IDs, and ASPLEU and CTXPB are the parameters selected from the **Select Different Parameters dialog**. CTXSBP values are subtracted from ASPLEU values. The resulting parameter is named:

DIFFERENCE(*src1*:ASPLEU, *src2*:CTXSBP)

The new parameter is placed into the DERIVED Source ID.

- **Differentiate** — Brings up a dialog that lets you compute the derivative of a specified parameter. The result of the operation is a new parameter; the original parameter is unchanged. By default, the new parameter is named:

DIFFERENTIATE(Source ID: parameter name)

For example, where *src1* is the Source ID, and CTXSPS is the parameter selected from the **Select Parameter to Differentiate** dialog. The derivative of CTXSPS is computed.

DIFFERENTIATE(*src1*:CTXSPS)

The new parameter is placed into the DERIVED Source ID.

- **FFT** — Brings up the dialog that lets you return the fast Fourier transform (FFT) for the input variable. For detailed information on the FFT function, see *PV-WAVE User's Guide*. By default, the new parameter is named:

FFT(Source ID: parameter name)

For example, where *src1* is the Source ID, and ASPREU is the parameter selected from the **Select Parameter to FFT** dialog. The FFT of ASPREU is computed and the resulting parameter is:

FFT(*src1*:ASPREU)

The new parameter is placed into the DERIVED Source ID.

- **Gain** — Brings up a dialog where you can apply a gain value to a parameter. The result of the operation is a new parameter with the value you entered in the amount of gain text field multiplied by each value of the selected parameter's data. The original parameter is unchanged.

GAIN(Source ID:parameter name, gain value)

For example, if your original data values were [10.0, 20.0, 30.0, ...] and the bias value you entered was 2.0, the values in the new parameter are [20.0, 40.0, 60.0, ...]. The new parameter is named:

GAIN(*src1*:ESPLEU, 2.00000)

where *src1* is the Source ID, and ESPLEU is the name of the parameter selected from the **Select Parameter to Apply Gain to** dialog. The ESPLEU values are multiplied by the gain value 2.00000.

The new parameter is placed into the DERIVED Source ID.

- **Gauss Fit** — Brings up a dialog that lets you fit a Gaussian curve through the data points of the original parameter. The result of the operation is a new parameter; the original parameter is unchanged. By default, the new parameter is named:

GAUSSFIT(Source name:parameter name)

For example, to compute the Gaussian curve of the parameter named COLFEU from the Source ID of *src1* the resulting parameter is:

GAUSSFIT(*src1*:COLFEU)

The new parameter is placed into the DERIVED Source ID.

- **Smooth** — Brings up a dialog that lets you smooth the data points in the original parameter. The result of the operation is a new parameter; the original parameter is unchanged. By default, the new parameter is named:

SMOOTH(Source ID:parameter name, smooth value)

For example, where *src1* is the Source ID, and DPLOEU is the name of the parameter selected from the **Select Parameter to Smooth** dialog. The values in DPLOEU are smoothed with a boxcar average of 5.00000.

SMOOTH(*src1*:DPLOEU,5.00000)

The parameter is placed into the DERIVED Source ID.

- **Trim** — Brings up a dialog that lets you create a new parameter whereby the initial value of the original parameter is subtracted from each subsequent point in the original parameter. The result of the operation is a new parameter; the original parameter is unchanged. By default, the new parameter is named:

TRIM(Source ID: parameter name)

For example, where *src1* is the Source ID, and ALTS is the name of the parameter selected from the **Select Parameter to Trim** dialog. The values in ALTS are subtracted from the initial value in ALTS.TRIM(*src1*:ALTS).

The new parameter is placed into the DERIVED Source ID.

- **WildPoint** — Places limits on the selected parameters data values according to user defined minimum and maximum limits. Values outside these limits are

set to the appropriate minimum or maximum values. By default, the new parameter is named:

WILDPPOINT(Source ID: parameter name, minimum value,  
maximum value)

For example, where *src1* is the Source ID, and BCAS is the name of the parameter selected from the **Select Parameter to WildPoint** dialog. The values between 10.000000 and 20.000000 are extracted from BCAS and the resulting parameter is:

WILDPPOINT(*src1*:BCAS,10.000000,20.000000)

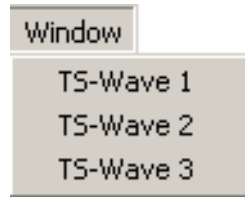
The new parameter is placed into the DERIVED Source ID.

### ***User Functions***

Use this menu to access user-defined TS-WAVE functions. For information on creating functions and adding them to this menu.

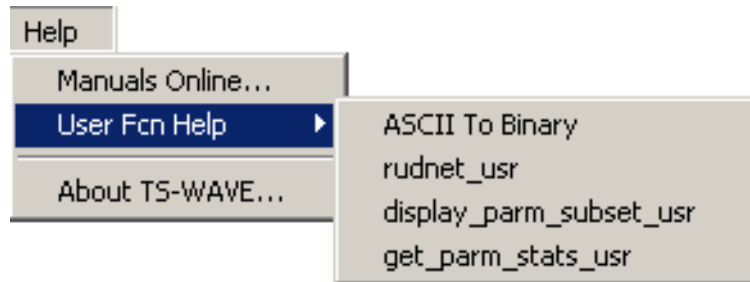
- **ASCII To Binary** — Converts a specifically formatted ASCII file to a general-purpose binary file. The resulting binary data can be read into TS-WAVE with increased speed over ASCII data. For more information see the **ASCII To Binary** topic under **Help->User Fcn Help** on the main TS-WAVE page. For detailed information, see [\*Working with User Functions\*](#) on page 49.
- **rudnet\_usr** — An example function that operates on two parameters and stores the result in a new derived parameter. This specifically calculates the difference between RPFLEU and RPFREU, two parameters found in the file data.ldf.
- **display\_parm\_subset\_usr** — This is a handy interface for displaying simple statistics of a subset of a parameter using a user defined range.
- **get\_parm\_stats\_usr** — Displays simple statistics for a selected parameter.

## Window Menu



Allows you to switch between active TS-WAVE Pages.

## Help Menu



### ***Manuals Online***

Brings up the on-line version of the TS-WAVE User's Guide and Developer's Reference.153

### ***User Fcn Help***

Displays help on user-created functions located under the **Analyze=>User** menu.

### ***About TS-WAVE***

Displays version information about TS-WAVE.

---

## Resource Files

### Customizing TS-WAVE

TS-WAVE's resource files give users the ability to modify and customize various default settings used by TS-WAVE. These resource files are located in `<tswave_dir>/resource/`.

Resource files are usually modified for customizing aspects of the TS-WAVE interface (such as background color or fonts used), customizing default behavior (such as snap-to-grid behavior and default data directories) and internationalizing TS-WAVE. Many of the resource files contain detailed documentation on their specific resources. For an example see `<tswave_dir>/resource/tswave_print.ads`.

---

**NOTE** Paths and environment variables discussed in this section are valid for both UNIX and Windows operating systems. For simplicity, all paths are shown using UNIX style separators and syntax (a '\$' precedes UNIX environment variables). Windows users should translate this syntax to the Windows counterpart. For example, `$USER_RESPATH/user/` for UNIX is equivalent in Windows to `%USER_RESPATH%\user\`.

---

### Resource File Syntax

Resource files are simply text files that contain Name:Value pairs. Values can consist of strings or numbers. Whenever TS-WAVE requires a value defined in a resource file, it loads the resource file that defines the 'Name' then reads and uses its associated 'Value'. An example of a Name:Value pair in the resource file `ep_graphattribute.ads` is:

```
Default_XAxisCharsize: .75
```

`Default_XAxisCharsize` is the resource 'Name' (or key) and `.75` is its associated 'Value'. This resource defines the default character size of the X axis labels to be `.75`. Increasing the value of 'Default\_XAxisCharsize' in `ep_graphattribute.ads` to `1.0` will have the effect of displaying a larger character font size for the X axis labels the next time TS-WAVE is started.

For more details on resource files, refer to the 'PV-WAVE Application Developer's Guide', Chapter 4, *Building VDA Tools*, and also refer to 'Using Resources and Strings in PV-WAVE Applications' contained in this chapter.



---

**NOTE** Some changes to resource files may make TS-WAVE inoperable. All changes should contain valid values for the specific resource. Resources that should never be changed are noted in the comments of the resource files that are located in `<tswave_dir>/resource/`.

---

## Standard TS-WAVE Resources

This section gives brief descriptions of the standard required resource files, which are located in `<tswave_dir>/resource/`. For more detailed information, refer to the resource files.

### ***\*.ad Resources***

Resource files with the **.ad** *extension* contain string and number definitions that are primarily used for internationalizing TS-WAVE Graphical User Interfaces (GUI), such as dialog labels and menu items.

*The .ad resource files are:*

**tswave.ad** — GUI resources for the TS-WAVE interface, such as confirm close behavior and menu labels.

**ec\_graphattribute.ad** — GUI resources for Contour.

**graph\_attribs.ad** — GUI resources shared by Contour and Eplot.

**tswave\_print.ad** — GUI resources used in the **Print dialog**.

### ***\*.ads Resources***

Resource files with the **.ads** *extension* contain string and number resources that define information such as the default data directory, Page colors, printer settings, and fonts.

*The .ads resource files are:*

**datahandler.ads** — TS-WAVE's general Datahandler resource settings.

**ec\_graphattribute.ads** — Allows customization of default values for Contour plot attributes such as X and Y axis label character sizes, axis line colors, axis offsets, tick number formats.

**ep\_graphattribute.ads** — Allows the settings for Plot attributes, such as axis attributes, !INVALID value, and Pick and Zoom window sizes.

**tswave.ads** — Defines various global resources used throughout TS-WAVE, such as default directory locations where TS-WAVE looks for User functions and data, snap to grid behavior, info block settings, and multi-page defaults.

**tswave\_colors.ads** — Contains default color definitions for TS-WAVE's interface, background and foreground colors, the graph attributes color palette, and various other color settings. Each button on the color palette is defined by 'Names' and 'Values' that define Red, Blue, and Green. Color names and their associated color definitions are listed in:

**Windows:** <VNI\_DIR>\wave\lib\std\windows\rgb.txt

**UNIX:** \$VNI\_DIR/hyperhelp/xprinter/rgb.txt

**tswave\_files.ads** — Contains default definitions for file-related TS-WAVE resources, such as tab data output defaults, and the default pattern for file open dialogs.

**tswave\_fonts.ads** — Defines TS-WAVE font resources used throughout the TS-WAVE interface and their various dialogs.

**tswave\_print.ads** — Contains settings for the default print driver, printer name, default Page size and orientation.

**<datatypeName>.ads** — Individual data handlers may use resource files that contain resources specific to that file type. These are stored in the directories <tswave\_dir>/datahandler/<datatypeName>/<datatypeName>.ads.

## Private User Resources

Individual users can define private resource settings whose values take precedence over the standard TS-WAVE resources. This means that regardless of which main TS-WAVE installation you run, your customized resource values will be used if they exist. User resources are especially useful in facilities where multiple users are running from one central TS-WAVE installation.

### Customizing Selected Resources

User resources that contain a selected subset of the standard TS-WAVE resources can reside in \$USER\_RESPATH/user/. If \$USER\_RESPATH is not set, this location defaults to:

**UNIX:** \$HOME/ts\_wave/resource/user

**Windows:** C:\ts\_wave\resource\user

Resource files in `$USER_RESPATH/user/` must be named the same as the standard TS-WAVE resource files, except that each can contain a partial list of customized resources from the standard file.

It is recommended that customized TS-WAVE resources added to files in `$USER_RESPATH/user/` should contain only the customized resource definitions. This practice makes it unnecessary to update customized resource files with each new TS-WAVE release and simplifies maintenance.

It is not required that the directory `$USER_RESPATH/user` exist if you do not intend to use private user resources.

---

**NOTE** Only resource files with the same names as standard TS-WAVE resource files that are placed in `$USER_RESPATH/user/` are recognized. Resource files with any other name are ignored.

---

---

**TIP** Windows users sharing machines as well as a common TS-WAVE installation should define `%USER_RESPATH%` as a Windows system user variable.

---

### ***Placing Other Resource Files in \$USER\_RESPATH***

Any User resource file used by a custom Data Handler or custom TS-WAVE application can be placed in the `$USER_RESPATH` directory. Resources defined in these resource files will be accessible in TS-WAVE on startup. If `$USER_RESPATH` is not set, this location defaults to:

**UNIX:** `$HOME/ts_wave/resource/`

**Windows:** `C:\ts_wave\resource\`

This location can be useful during application development for developers using a shared TS-WAVE installation and would like to keep certain resources encapsulated so they do not interfere with another user's resource settings

---

**CAUTION** If resource files with the same names as standard resource files in the main TS-WAVE installation (`<tswave_dir>/resource/`) exist in `$USER_RESPATH`, they will be loaded in place of the standard resource files, making it necessary that these resource files contain the complete set of resource keys that TS-WAVE expects to be defined in that file or TS-WAVE may not function properly. Although it is allowed, it is not recommended that you place customized standard TS-WAVE resource files here because it will complicate

merging of your custom User Resources between new versions of TS-WAVE if the new resource files contain new resources.

---

### ***NOTE about \$USER\_RESPATH for TS-WAVE Developers***

On startup, `$USER_RESPATH` is automatically prepended to `$WAVE_RESPATH`, the internal TS-WAVE environment variable that defines the list of directories where TS-WAVE resource files reside. Strings from this file can be accessed by using the PV-WAVE routines `WoLoadStrings()` and `TmGetMessage()`. If a colon-separated Name:Value pair exists in a user resource file named, for example, `$USER_RESPATH/myresfile.ads`, in which the resource `MY_RESOURCE_KEY` is defined, the value can be retrieved in a TS-WAVE function like this:

```
WoLoadStrings, 'myresfile.ads'
returnval = TmGetMessage('MY_RESOURCE_KEY')
```

Alternatively, user resource file strings can be accessed by using:

```
returnval = TmGetMessage('myresfile.ads', 'MY_RESOURCE_KEY')
```

This last `TmGetMessage()` call that includes a filename argument will both load all strings in `myresfile.ads` into TS-WAVE's internal resource database and return the value of the string assigned to `MY_RESOURCE_KEY` in the `returnval` variable. For more detailed information on these two routines, refer to the PV-WAVE documentation.

Because `$USER_RESPATH` is automatically added to `$WAVE_RESPATH`, it is not necessary to include the full path to resource filenames used in the `TmGetMessage()` or `WoLoadStrings()` calls. On the other hand, `$USER_RESPATH/user/` is not added to `$WAVE_RESPATH` since it is intended to hold only standard, user-customized TS-WAVE resource files.

### ***Example***

This example shows how to create and use a private user resource. For example, this user wants to change the default background and foreground colors used when TS-WAVE is run from a shared installation without affecting other users. By default, these colors are black for background (0) and white for foreground (1).

**Step 1** Create your default user resource directory:

**(Windows)** `C:\ts_wave/resource/user\`

**UNIX** `$HOME/ts_wave/resource/user/`

- Step 2** Locate the name of the resources that control the background and foreground colors in the `<tswave_dir>/resource/` directory and the name of the file where they are defined. Background and foreground color resources are defined in

```
<tswave_dir>/resource/tswave_colors.ads as:
```

```
Background_Color: 0
```

```
Foreground_Color: 1
```

- Step 3** Create the file:

```
$HOME/ts_wave/resource/user/tswave_colors.ads.
```

Copy only those two lines from

```
<tswave_dir>/resource/tswave_colors.ads
```

and paste into this file, then modify the values in your new user resource file. This example switches background and foreground colors from their default settings:

```
Background_Color: 1
```

```
Foreground_Color: 0
```

Now when you start TS-WAVE from any installation, your background color will be white and your foreground color will be black.

### ***Private User Function Example***

In this example, `$HOME/ts_wave/usrfcn` is set up to be your private user function path. Note that user function directories must end in the directory name 'usrfcn' (i.e., `<any_path>/usrfcn/`).

In the steps below, the HOME directory for Windows is C:\.

- Step 1** Create the directory `$HOME/ts_wave/usrfcn/` and place your user function file(s) in this directory.
- Step 2** Copy `<tswave_dir>/usrfcn/userfunction.list` to your private user function directory, `$HOME/ts_wave/usrfcn/`, and add your user function's name to this list.
- Step 3** Create the file `$USER_RESPATH/user/tswave.ads`. Add the key `UserFunction_Root_Dir` to this file and define its value to be `$HOME/ts_wave`.  
For example, if your home directory is named `/home` and you use the

default value of \$USER\_RESPATH, this line would be:  
UserFunction\_Root\_Dir: /home/ts\_wave/

---

**NOTE** Do not use environment variables to define paths. Either expand the paths or show full path definitions.

---

Note that the final 'usrfcn' directory is omitted in this definition because it is assumed this directory will exist under UserFunction\_Root\_Dir.

The next time you start TS-WAVE from any installation, your new userfunction will appear in the list of user functions under the **Analyze=>User** menu.

You will need to open a data file to sensitize the **Analyze=>UserFunction** menu.

---

## Using Shortcut Keys

TS-WAVE provides many shortcut keys so that you can use the keyboard while you work. For example, you can press [CTRL+C] to copy the selected text or object.

| Shortcut      | Description            |
|---------------|------------------------|
| Ctrl + A      | Edit=>Select All       |
| Ctrl + C      | Edit=>Copy             |
| Ctrl + D      | Edit=>Deselect All     |
| Ctrl + Delete | Edit=>Delete           |
| Ctrl + E      | Edit=>Object Select    |
| Ctrl + F      | View=>Plot Attributes  |
| Ctrl + G      | Create=>Graph Object   |
| Alt + G       | Edit=>Group            |
| Ctrl + H      | Create=>Header Object  |
| Ctrl + N      | File=>New Page         |
| Ctrl + O      | File=>Open Data Source |
| Ctrl + P      | File=>Print            |
| Ctrl + R      | Edit=>Redraw           |
| Ctrl + S      | File=>Save Template    |
| Ctrl + T      | Create=>TabData File   |
| Ctrl + U      | Edit=>Ungroup          |
| Ctrl + V      | Edit=>Paste            |
| Ctrl + X      | Edit=>Cut              |
| Ctrl + Z      | View=>Data Zoom        |

In addition, you may use the <Alt> key with the underlined letter in each menu name to open that menu, and then press the underlined letter of the menu item you wish to activate on your keyboard.

|         |                        |
|---------|------------------------|
| Alt + F | Brings up File Menu    |
| Alt + E | Brings up Edit Menu    |
| Alt + V | Brings up View Menu    |
| Alt + C | Brings up Create Menu  |
| Alt + A | Brings up Analyze Menu |
| Alt + W | Brings up Window Menu  |
| Alt + H | Brings up Help Menu    |

---



---

## ***FORTRAN Format Strings***

FORTRAN format strings are used by TS-WAVE to specify the size and appearance of numeric values when converted to strings for output, e.g. Tab Files or tick labels.

The following tables show the FORTRAN format specifiers that you can use in TS-WAVE.

The following table shows data conversion characters, which specify the type of data that is being transferred.

### ***FORTRAN Format Specifiers***

FORTRAN Format Codes that Transfer Data

| <b>Conversion Character</b> | <b>How the Data is Transferred</b>                                                                                                                                                                                                                                                                              |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [n]A[w]                     | Transfers character data. <i>n</i> is a number specifying the number of times to repeat the conversion. If <i>n</i> is not specified, the conversion is performed once. <i>w</i> is a number specifying the number of characters to transfer. If <i>w</i> is not specified, all the characters are transferred. |
| [n]D[w.d]                   | Transfers double-precision floating-point data. <i>n</i> is a number specifying the number of times to repeat the conversion. <i>w</i> specifies the number of characters in the external field, and <i>d</i> specifies the number of decimal positions.                                                        |
| [n]E[w.d]                   | Transfers single-precision floating-point data using scientific (exponential) notation. The options are the same as for the D conversion character.                                                                                                                                                             |
| [n]F[w.d]                   | Transfers single-precision floating-point data. The options are the same as for the D conversion character.                                                                                                                                                                                                     |
| [n]G[w.d]                   | Uses E or F conversion, depending on the magnitude of the value being processed. The options are the same as for the D conversion character.                                                                                                                                                                    |
| [n]I[w] or<br>[n]I[w.m]     | Transfers integer data. <i>n</i> is a number specifying the number of times to repeat the conversion. <i>w</i> specifies the width of the field in characters. <i>m</i> specifies the minimum number of non-blank digits required.                                                                              |
| [n]O[w] or<br>[n]O[w.m]     | Transfers octal data. The options are the same as for the I conversion character.                                                                                                                                                                                                                               |

## FORTRAN Format Codes that Transfer Data (Continued)

| Conversion Character | How the Data is Transferred                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [n]Z[w] or [n]Z[w.m] | Transfers hexadecimal data. The options are the same as for the I conversion character.                                                                                                                                                                              |
| Q                    | Obtains the number of characters in the input record to be transferred during a read operation. This conversion character is used for input only. In an output statement, the Q format code has no effect except that the corresponding I/O list element is skipped. |

---

**NOTE** Characters in square brackets [ ] are optional.

---

## ***Developer's Reference***

The Developer's Reference contains the following sections:

- *General Requirements for Adding TS-WAVE User Functions and Custom Data Handlers*
- *TS-WAVE User Functions*
- *Tips for Creating User Functions*
- *Creating a Derived Parameter*
- *Creating a Help File*
- *Creating a New Plot Window*
- *TS-WAVE Data Handlers*
- *Setting Up Your Data Handler*
- *The Data Handler Functions*
- *Tips For Creating Data Handlers*
- *TS-WAVE's Utility Functions*
- *TS-WAVE's DataManager Functions*
- *TS-WAVE's PrintManager Functions*
- *TS-WAVE's Miscellaneous Utility Functions*
- *TS-WAVE System Variables*
- *Customization Options*

---

## **General Requirements for Adding TS-WAVE User Functions and Custom Data Handlers**

This section explains the requirements necessary for adding TS-WAVE User Functions and Custom Data Handlers. The list below shows the four requirements.

- TS-WAVE Developer's License
- Becoming Familiar with the PV-WAVE Programming Language
- Understanding the WAVE\_PATH Environment Variable
- Compiling PV-WAVE Routines

### **TS-WAVE Developer's License**

You must be able to run TS-WAVE in Developer mode (NoBlock) to compile your PV-WAVE .pro files, which requires a TS-WAVE developer license.

### **Becoming Familiar with the PV-WAVE Programming Language**

In order to write and add your own TS-WAVE user functions and custom Data Handlers, you will need to be familiar with the PV-WAVE programming language. If you are not, you should refer to the PV-WAVE Tutorial, PV-WAVE Programmer's Guide and the PV-WAVE User's Guide that are installed in PDF format by the default TS-WAVE installation. Contact Visual Numerics, Inc. for available training at 303-379-3040.

For information on creating your own graphical user interfaces, refer to the PV-WAVE Application Developer's Guide, chapters 4, 9, 10, and 11. PV-WAVE Online Help is also available for individual command reference.

These documents together with the full set of PV-WAVE documentation provide a comprehensive overview of the PV-WAVE language. To start the PV-WAVE online documentation or wavedoc:

**Windows**      via the Start Menu under the PV-WAVE program group.

**UNIX**          source <vni\_dir>/wave/bin/wvsetup then type wavedoc.

### **Understanding the WAVE\_PATH Environment Variable**

In order for your routines' .pro and .cpr files to be found by TS-WAVE, they must be located either in the current directory or in a directory that is included in the path

list defined by the TS-WAVE environment variable `WAVE_PATH`. By default, all directories distributed with TS-WAVE under `<vni_dir>` that contain `.pro` files are included in `WAVE_PATH` as well as the current directory and your private user function directory, if it exists. For more details on private user functions, see [Private User Resources](#) on page 166.

Under UNIX, `WAVE_PATH` is defined after you have sourced `wvsetup`. For details about adding more directories to `WAVE_PATH` for UNIX, see [Customized Startup Command Files](#) on page 225.

Under Windows, `WAVE_PATH` is internally generated on startup. If `%WAVE_PATH%` exists in the user's environment, it is appended to TS-WAVE's internal `%WAVE_PATH%`.

## Compiling PV-WAVE Routines

PV-WAVE `.pro` routines must be compiled before they can be executed. They may be compiled either interactively, automatically, or pre-compiled into `.cpr` files.

TS-WAVE runtime mode requires that all routines' `.pro` files be pre-compiled into `.cpr` files. TS-WAVE Developer mode (NoBlock) will use a routine's pre-compiled `.cpr` file if it exists or will automatically compile the routine's `.pro` file if a `.cpr` file does not exist. For details on running TS-WAVE in NoBlock mode, see [TS-WAVE Developer Mode \(NoBlock\) Startup File](#) on page 226. You may also interactively compile a `.pro` file in NoBlock mode by using the PV-WAVE `.RUN` executive command.

Here's a simple example that interactively compiles a procedure using the `.RUN` executive command then creates a pre-compiled `.cpr` file of the compiled procedure. For this example, we will compile a user-created routine named **`yourfcn_usr`**.

---

**NOTE** In this example, text following a semicolon indicates a comment.

---

**Step 1** Create a PV-WAVE procedure named **`yourfcn_usr`** in a file named `yourfcn_usr.pro` and place it in `<tswave_dir>/usrfcn`. For an actual code example, refer to the [TS-WAVE User Functions](#) on page 180 of this document. (See the [Example](#) on page 181.)

**Step 2** Start TS-WAVE in NoBlock mode.

**Step 3** At the TS-WAVE command prompt, change directories:

```
CD, GETENV("TSWAVE_ROOT") + '/usrfcn'
```

#### Step 4 Compile and generate a compiled .cpr:

```
.RUN yourfcn_usr
;At this point, if yourfcn_usr had no syntax errors, it has
;been compiled and is now active in the TS-WAVE session.
;Note: The .run command must not have spaces before it.
 COMPILE, /All, FileName='yourfcn_usr'
;yourfcn_usr.cpr now exists in the current directory
;now the compiled PV-WAVE routines are saved in binary ;format in
 the file yourfcn_usr.cpr in the current directory.
```

#### Step 5 Repeat Steps 3 and 4 for all other .pro files you wish to compile.

---

**NOTE** User functions that should appear in the **Analyze=>User** menu may not show up until you exit and restart TS-WAVE.

---

For more information, refer to the PV-WAVE Programmer's Guide, "Chapter 1, Creating and Running Programs".

## Understanding the `tool_name` Argument

Many of the routines documented throughout this chapter accept an argument called *tool\_name*. *tool\_name* is a string that TS-WAVE uses internally to identify each open TS-WAVE Page. Page 1 is internally referred to as 'TS\_Wave\_0', Page 2 as 'TS\_Wave\_1' and so on, so that each Page is identified by a unique *tool\_name*.

When you read the next few sections in this chapter, you will find that custom Data Handlers and User functions called by TS-WAVE are passed the *tool\_name* as the first argument. In this way, the Page that invoked the routine is known.

## Example

*tool\_name* can be used to extract and set information relating to a specific Page within a Data Handler or User routine. For example, one of the routines documented in the Utility Routines section in this chapter is `PM_getDefaultPrinter`, which uses the *tool\_name* argument to return the default printer driver set for a specific Page.

To print the default printer driver for the first Page in a session where, in this example, two Pages are open, you can add the following lines in a PV-WAVE routine:

```
tools=TmEnumerateToolnames()
INFO, tools
```

```

;Output is: TOOLS STRING = Array(2)
IF tools(0) NE '' THEN BEGIN
 PRINT, "The default printer for ", tools(0), " is ", $
 PM_getDefaultPrinter(tools(0))
;Output is: The default printer for TS_Wave_0 is qms1
ENDIF

```

To print the default printer for the first Page from the TS-WAVE command line, assuming Page 1 is open, use:

```

PRINT, PM_getDefaultPrinter('TS_Wave_0')

```

---

## TS-WAVE User Functions

### General

In addition to TS-WAVE's standard analysis functions, TS-WAVE provides a mechanism for adding a wide range of customized functionality (see [Working with User Functions](#) on page 49 of this manual). User functions can be added easily to integrate analysis routines that call a PV-WAVE procedure, an external C or FORTRAN library, generate a derived parameter, or display a custom plot.

### User Functions Specific Requirements

Custom user functions must meet some simple requirements in order to integrate properly into TS-WAVE. These are:

- They must be PV-WAVE procedures. Your procedure can do anything that standard PV-WAVE routines can do, including: call other PV-WAVE routines, call an external C or Fortran library, produce graphical output, read from or write to external data files or connect to a database.
- User functions must accept one parameter: a string containing the unique VDA tool name.
- User functions must be located in the directory specified by the combination of the resources `UserFunction_Root_Dir` and `UserFunction_File_Dir` in the resource file `tswave.ads`. The defaults for each are the `<tswave_dir>` and `usrfcn`, respectively.
- In order to appear in the **Analyze=>User** menu on the main TS-WAVE tool, user functions must be listed in the file specified by the resource `UserFunction_ListFile` in the file `tswave.ads`. The default file is `<tswave_dir>/usrfcn/userfunction.list`.
- User functions must have unique names in order not to conflict with other PV-WAVE or TS-WAVE routines.

### Modifying the User Function Directory

If you wish to store your user functions in a directory other than the default `<tswave_dir>/usrfcn` directory you must modify the `UserFunction_*` resources in the file `<tswave_dir>/resource/tswave.ads`. A section of this file is reproduced in the example below.



## Example

```
UserFunction_Root_Dir:
UserFunction_File_Dir: usrfcn
UserFunction_ListFile: userfunction.list
UserFunction_HelpFileExt: txt
```

---

**NOTE** By default, the resource `UserFunction_Root_Dir` is not set. This causes it to default to your TS-WAVE installation directory `<tswave_dir>`.

---

The example shows that TS-WAVE looks for user functions in the `<tswave_dir>/usrfcn` directory. It uses the file `<tswave_dir>/usrfcn/userfunction.list` to build the **Analyze=>User** menu of available user functions on the main TS-WAVE interface. The resource `UserFunction_HelpFileExt` specifies the file extension of the help file associated with each user function. For the setting in the above example, help files should have the same names as their associated user functions, except with the extension defined by `UserFunction_HelpFileExt`.

## Writing a User Function

---

**NOTE** In order to write a TS-WAVE user function, you need to be familiar with the PV-WAVE language. If you are not, you should make use of the PV-WAVE Programmer's Guide and the PV-WAVE User's Guide. Both are installed online in PDF format by the default TS-WAVE installation. Individual PV-WAVE routines and an overview of the PV-WAVE language can also be found in the online help. UNIX users should source `vni_dir/wave/bin/wvsetup` and type `wavedoc` to start the online manuals or `wavehelp` to start the PV-WAVE online help. Windows users can access either of these via the Start Menu.

---

Here are the basic steps for creating a user function.

**Step 1** Create a PV-WAVE procedure that accepts the `tool_name` argument.

### *Example*

```
PRO YourUsrFcn_usr, tool_name

;

END
```

- Step 2** Add to the procedure the code to perform your specific task.
- Step 3** Save your user function as, in this case, `yourusrfcn_usr.pro` in the `<tswave_dir>/usrfcn` directory.
- Step 4** Add **yourusrfcn\_usr** to the file `<tswave_dir>/usrfcn/userfunction.list`.

### ***Example***

Here is an example of the *userfunction.list* file with **yourusrfcn\_usr** added to the end.

```
 ; comment line
 ;
rudnet_usr
dudnet_usr
get_parm_stats_usr
yourusrfcn_usr
```

---

## ***Tips for Creating User Functions***

User functions can be as simple or as complex as you choose. The TS-WAVE utilities listed below are useful in developing user functions. For more information, see the documentation for these and other helpful routines under *TS-WAVE's Utility Functions* on page 209.

**Dm\_getSrcList** — Returns a list of all opened source names or data files.

**Dm\_getParmList** — Returns a list of available parameter names for a specified source name.

**Dm\_getParm** — Returns the data values and time in an associative array. Dm\_getParm is needed to create derived parameters.

In addition, TS-WAVE provides three routines to incorporate a parameter selection component into your interface.

**Gu\_newParmlist\_Component** — Creates the Parmlist\_Component graphical interface component. This can be added to your interface to allow Data Source and parameter selection.

**Gu\_updateParmlist\_Component** — Updates the Parmlist\_Component with the latest source files.

**Gu\_readParmlist\_Component** — Reads the Parmlist\_Component interface to get the selected source name and parameter name.

---

## ***Creating a Derived Parameter***

A TS-WAVE user function is commonly used to create a new parameter for graphical display in the main TS-WAVE interface. In order for TS-WAVE to be able to access this new parameter your user function must create what is called a derived parameter. Derived parameters are stored under the Source ID 'DERIVED' in the TS-WAVE parameter interfaces.

In order to create a derived parameter you need the following:

- Your top-level user function (**YourUsrFcn\_usr**, from above) must make a call to the DM\_addFCN and DM\_addPSRC routines (See *DM\_addFCN Function* on page 209, and *DM\_addPSrc Function* on page 210) to register your new parameter, its function call and its associated Source ID. The DM\_addFCN function takes three arguments: The unique VDA tool name, a string holding the name of your derived parameter as it will appear in the TS-WAVE interface

and a string containing the actual function call that processes the data for your derived parameter.

The `DM_addPSRC` function also takes three arguments: the `tool_name`, a string containing the derived parameter's name and a string containing the Source ID of the parameter from which it is derived.

- A separate function to actually process the derived data. No processing takes place until the derived parameter is displayed or written to an output file. At that time, TS-WAVE executes the function call you passed to `DM_addFCN` in your top-level user function.

For the example below, the second routine is named `YourUsrFcn`.

---

**NOTE** Each function should be placed in a separate file in the `<tswave_dir>/usrfcn` directory with each file named after the routine it contains. This is because the second function is called by TS-WAVE when the derived parameter is accessed. If the two functions are in the same file and the file is named after the first function, the second function will not be found by TS-WAVE if the first function has not already been called. This situation arises when the derived parameter is stored in a TS-WAVE Template or Session File, possibly for use in Batch Processing. For more information, see the PV-WAVE Programmer's Guide, "*Chapter 9, Writing Procedures and Functions*".

---

Here are the two functions. To keep things simple for now, our functions will not have any user interface. We will assume that you have read in the sample data file *data.ldf* with the Source ID '*src1*'.

### Function 1:

```
; The top-level user function
PRO YourUsrFcn_usr, tool_name

 ; forward declaration of the second function and the
 ; utility functions

 DECLARE FUNC, YourUsrFcn

 DECLARE FUNC, DM_addFCN, DM_addPSRC

; A string containing the name of the new derived parameter
; new_parm = "MyParm(src1:COLFEU)"
```

```

; Construct a string containing the function call to
; YourUsrFcn(). The values that the function uses are
; hardcoded here. Normally you would construct this
; string from input you gather from a user interface.

; For example:
;ret_data = GU_ReadParmList_Component(tool_name, $
; wids=parm_wids)

;parm_name = ret_data('PARAMETER')
;src_name = ret_data('SOURCE')
;function_call = "YourUsrFcn(tool_name, '"+src_name+" ' , ' ' " $
; +parm_name+" ')"

; The simpler, hardcoded example
function_call = "YourUsrFcn(tool_name, 'src1', 'COLFEU') "

; Register the new parameter's Source ID
status = DM_addPSRC(tool_name, new_parm, 'src1')

; Register the new derived parameter and it's function call
status = DM_addFCN(tool_name, new_parm, function_call)

RETURN

END

```

## Function 2:

```

; the function to actually create the derived data
FUNCTION YourUsrFcn, tool_name, src_name, parm_name

; forward declaration of utility routine to retrieve the
; original parameter
DECLARE FUNC, DM_GetParm

; get the data from the target parameter
data_array = DM_getParm(tool_name, src_name, parm_name)

; Do some error checking
IF size(data_array,/type) NE 11 THEN BEGIN
 Print, 'DM_getParm, did not return data'
 RETURN, ASARR()

```

```

ENDIF
; extract the data and time arrays from the associative
; array returned by DM_GetParm()
data = data_array('DATA')
time = data_array('TIME')

; do some simple math
data2 = data*data

RETURN, ASARR('DATA', data2, 'TIME', time)

END

```

***To use the example above, do the following:***

- Step 1** Copy function 1 and function 2 above into plain text files named `yourusrfcn_usr.pro` and `yourusrfcn.pro`, respectively.
- Step 2** Place the files in the `<tswave_dir>/usrfcn` directory and add **yourusrfcn\_usr** to the `userfunction.list` file in the `tswave_dir/usrfcn` directory.
- Step 3** Start TS-WAVE in NoBlock mode and, at the TS-WAVE prompt, change directory to your `<tswave_dir>/usrfcn` directory.
- Step 4** At the prompt, type:

```

DELVAR, /All
DELPROC, /All
DELFUNC, /All

.RUN yourusrfcn_usr
compile, /all, filename='yourusrfcn_usr'
DELPROC, 'yourusrfcn_usr'
.RUN yourusrfcn
COMPILE, /All, Filename='yourusrfcn'

```

- Step 5** Exit and restart TS-WAVE.
- Step 6** Load the sample `data.ldf` file as Data Source 'src1'.
- Step 7** Select **Analyze=>User=>yourusrfcn\_usr**.

- Step 8** Create a graph object, then open the **Graph Attributes Interface** and add the DERIVED parameter `MyParm(src1:COLFEU)` to it.
- Step 9** Click **OK** to dismiss the **Graph Attributes Interface**. Your new parameter is now plotted on the graph object on the Page.

---

## ***Creating a Help File***

You may include a help file for your user function. The help file must be a plain text file and have the same name as the function, but have the file extension specified by the resource `UserFunction_HelpFileExt` in `tswave.ads`, `.txt`, by default. Additionally, the file must be located in the same directory as the user function. Users are able to access this help file by selecting **Help=>User Fcn Help** on the main TS-WAVE menu bar.

---

## ***Creating a New Plot Window***

If a user function contains a `WwDrawing` window, it must manage the new window ID.

### **Example**

This example uses the WSET procedure.

```
; create a new drawing area
;

wid = WwDrawing (layout, windowid, 'AnExposeCallBack', $
 wsize, dsize, /NoMeta, /Noscroll, Area = area, $
 /Top, /Bottom, /Left, /Right)

; store the window ID for use by other routines
status = TmSetAttribute(tool_name, 'new', 'WINDOWID', windowid)

; make sure the tool_name is attached to the drawing area
status = WwSetValue(area, UserData=tool_name)

...

...

...
```

```

; an example of retrieving the window ID we set earlier
windowid = TmGetAttribute(tool_name, 'new', 'WINDOWID', $
 Default=-1)

;error checking
IF windowid EQ -1 THEN RETURN
status = WwSetValue(wid, /Show)
status = WwSetValue(wid,/Update)

; Windows Users may also have to do the following:
status = WtProcessEvent(/Drain)
; to force the window manager to finish creating the
; drawing area before you try to WSET to it...

; set the focus to the new window so you don't draw in the
; main TS-WAVE window.
WSET, windowid

```

## A Complete User Function (userfcn) Example

This complex example creates the following:

- A graphical user interface that contains a parameter list.
- A text box that contains statistics derived from the selected parameter.
- A drawing window with which one can interact with the mouse.

---

**NOTE** This user function displays an interactive window and does not create a derived parameter.

---

*To test this example, do the following:*

**Step 1** For your convenience the following example is provided as a .pro file. The file (udemo\_usr.pro) is located in <tswave\_dir>/usrfcn/udemo\_usr.pro. Add the user function **udemo\_usr** to <tswave\_dir>/usrfcn/userfunction.list.

For more information, see [Writing a User Function](#) on page 181.

**Step 2** Start TS-WAVE and load a data set, then select **Analyze => User => udemo\_usr** from the menu bar.



**Step 3** Select a parameter from the list and click **OK**. Notice a time series plot appeared in the drawing window. Click your left mouse button to select start and end points. The statistics in the box are updated and the selected time series segment is highlighted. Right click to return to the original plot. To access the derived parameter, click on the **Data Source** pull-down menu and choose **DERIVED** as the Data Source. To close the window, select **Cancel**.

## Example

```
;*****
; Drawing Callback
;*****

PRO udemo_tsdrawCB, wid, index
 COMMON stuff, tool_name, parm_wids, flag, x0, x1, y0, y1
 COMMON other_stuff, data, textwid, parm_name
 DECLARE FUNC, TmGetAttribute
 DECLARE FUNC, TmSetAttribute
 DECLARE FUNC, DM_getParm
 DECLARE FUNC, GU_read_Parmlist_component
 DECLARE FUNC, UDemo
 IF SIZE(flag, /Type) EQ 0 THEN flag = 0
 DPRINT, 'In udemo-tsDrawCB...'
 SetCursor, tool_name, /Wait
 IF SIZE(parm_wids, /Type) LT 8 THEN RETURN
 ret_data = GU_readParmlist_component(tool_name, wids=parm_wids)
 parm_name = ret_data('PARAMETER')
 src_name = ret_data('SOURCE')
 textwid = TmGetAttribute(tool_name, 'PARM_STATS_WIDS', $
 'TEXT_WID')
 data_array = DM_getParm(tool_name, src_name, parm_name)
 IF SIZE(data_array, /type) NE 11 THEN BEGIN
 MESSAGE, 'DM_getParm, did not return data', /Continue
 RETURN
```

```

ENDIF

data = data_array('DATA')
time = data_array('TIME')
; plot the new parameter
PLOT, data, title = STRSUBST(parm_name, '!', '!!!')
SetCursor, tool_name, /Reset
END

;*****
; Event Handler Callback
;*****
PRO UDemo_handlerCB, wid, shell, event
 COMMON stuff, tool_name, parm_wids, flag, x0, x1, y0, y1
 COMMON other_stuff, data, textwid, parm_name
 DECLARE FUNC, TmGetAttribute
 DECLARE FUNC, TmSetAttribute
 DECLARE FUNC, DM_getParm
 DECLARE FUNC, GU_read_Parmlist_component
 DECLARE FUNC, Display_Parm_StatsCB
 IF event.button EQ 1 THEN BEGIN ; Left Mouse Button
 xypos = WwGetPosition(event)
 xdev = xypos(0)
 ydev = !D.Y_Vsize - xypos(1)
 xydat = CONVERT_COORD(xdev, ydev, /Device, /To_Data)
 test = flag MOD 2
 IF (test EQ 0) THEN BEGIN
 x0 = xydat(0)
 y0 = xydat(1)
 XYOUTS, x0, xydat(1), '*', color='FFFF00'x1
 ENDIF ELSE BEGIN
 x1 = xydat(0)
 y1 = xydat(1)
 XYOUTS, x1, xydat(1), '*', color='FF00FF'x1
 PLOTS, [x0,x1], [y0,y1], color=5
 tmp = [x0,x1] & tmp = tmp(SORT(tmp))

```

```

 x0 = tmp(0) & x1 = tmp(1)
 datarange = data(FIX(x0):FIX(x1))
 OPLOT, FINDGEN(x1-x0)+x0-1, datarange, color=16
 ENDELSE
 flag = flag + 1
ENDIF ELSE BEGIN ; Not Left Mouse Button
 flag = 0
 test = 1
 PLOT, data, title = STRSUBST(parm_name,'!', '!!!')
 datarange = data
ENDELSE
IF (test NE 0) THEN BEGIN ; Update statistics
 parmsize = N_ELEMENTS(datarange)
 strparmsize = STRING(REPLICATE(32B,40))
 STRPUT,strparmsize,'Size of Parameter:'
 STRPUT,strparmsize,STRCOMPRESS(STRTRIM($
 STRING(parmsize),2)),29
 DH_MinMax, datarange, Max=maxval, Min=minval
 strminval = STRING(REPLICATE(32B,40))
 STRPUT,strminval , 'Minimum Value:'
 STRPUT,strminval,STRCOMPRESS(STRTRIM($
 STRING(minval),2)),29
 strmaxval = STRING(REPLICATE(32B,40))
 STRPUT,strmaxval, 'Maximum Value:'
 STRPUT,strmaxval,STRCOMPRESS(STRTRIM($
 STRING(maxval),2)),29
 meanval = MEDIAN(datarange)
 strmeanval = STRING(REPLICATE(32B,40))
 STRPUT,strmeanval, 'Median Value:'
 STRPUT,strmeanval,STRCOMPRESS(STRTRIM($
 STRING(meanval),2)),29
 stdval = STDEV(datarange)
 strstdval = STRING(REPLICATE(32B,40))
 STRPUT,strstdval , 'Standard Deviation:'
 STRPUT,strstdval,STRCOMPRESS(STRTRIM($

```

```

 STRING(stdval),2)),29
 totval = TOTAL(datarange)
 strtotval = STRING(REPLICATE(32B,40))
 STRPUT,strtotval,'Total of Parameter:'
 STRPUT,strtotval,STRCOMPRESS(STRTRIM($
 STRING(totval),2)),29
 rangeval = maxval - minval
 strrangeval = STRING(REPLICATE(32B,40))
 STRPUT,strrangeval,'Range of Parameter:'
 STRPUT,strrangeval,STRCOMPRESS(STRTRIM($
 STRING(rangeval),2)),29
 strret = STRING(10B)
 strtxt = strparmsize + strret + strminval + $
 strret + strmaxval + strret + strmeanval + $
 strret + strstdval + strret + strtotval + $
 strret + strrangeval + strret
 st = WwSetValue(textwid , strtxt)
ENDIF
END
;*****
; Display Stats Callback
;*****
FUNCTION Display_Parm_StatsCB, wid, index
 COMMON stuff, tool_name, parm_wids, flag, x0, $
 x1, y0, y1
 DECLARE FUNC, TmGetAttribute
 DECLARE FUNC, TmSetAttribute
 DECLARE FUNC, DM_getParm
 DECLARE FUNC, GU_read_Parmlist_component
 DPRINT, 'In Display_Parm_StatsCB...', index
 SetCursor, tool_name, /Wait
 IF(index EQ 3) THEN return,0
 udata = WwGetValue(wid, /Userdata)
 tool_name = udata(0)
 parm_wids = udata(1)
 grael_name = 'GET_PARM_STATS_USR'
 ret_data = GU_readParmlist_component(tool_name, $
 wids=parm_wids)
 parm_name = ret_data('PARAMETER')

```

```

src_name = ret_data('SOURCE')
textwid = TmGetAttribute(tool_name,'PARM_STATS_WIDS', $
 'TEXT_WID')
data_array = DM_getParm(tool_name, src_name, parm_name)
IF SIZE(data_array,/type) NE 11 THEN BEGIN
 PRINT, 'DM_getParm, did not return data'
 RETURN, ASARR()
ENDIF
data = data_array('DATA')
time = data_array('TIME')
; Gather the desired statistics
parmsize = N_ELEMENTS(data)
strparmsize = STRING(REPLICATE(32B,40))
STRPUT,strparmsize,'Size of Parameter:'
STRPUT,strparmsize,STRCOMPRESS(STRTRIM($
 STRING(parmsize), 2)),29
DH_MinMax, data, Max=maxval, Min=minval
strminval = STRING(REPLICATE(32B,40))
STRPUT,strminval , 'Minimum Value:'
STRPUT,strminval , STRCOMPRESS(STRTRIM($
 STRING(minval), 2)),29
strmaxval = STRING(REPLICATE(32B,40))
STRPUT,strmaxval, 'Maximum Value:'
STRPUT,strmaxval, STRCOMPRESS($
 STRTRIM(STRING(maxval), 2)),29
meanval = MEDIAN(data)
strmeanval = STRING(REPLICATE(32B,40))
STRPUT,strmeanval, 'Median Value:'
STRPUT,strmeanval, STRCOMPRESS(STRTRIM($
 STRING(meanval), 2)),29
stdval = STDEV(data)
strstdval = STRING(REPLICATE(32B,40))
STRPUT,strstdval , 'Standard Deviation:'
STRPUT,strstdval , STRCOMPRESS(STRTRIM($
 STRING(stdval), 2)),29
totval = TOTAL(data)
strtotval = STRING(REPLICATE(32B,40))
STRPUT,strtotval, 'Total of Parameter:'
STRPUT,strtotval , STRCOMPRESS(STRTRIM($
 STRING(totval), 2)),29

```

```

rangeval = maxval - minval
strrangeval = STRING(REPLICATE(32B,40))
STRPUT,strrangeval,'Range of Parameter:'
STRPUT,strrangeval,STRCOMPRESS(STRTRIM($
 STRING(rangeval), 2)),29
strret = STRING(10B)
strtxt = strparmsize + strret + strminval + strret + $
 strmaxval+ strret + strmeanval + strret + $
 strstdval + strret + strtotval + strret+ $
 strrangeval + strret
st = WwSetValue(textwid , strtxt)
UDemo_TSDrawCB, wid, index
SetCursor, tool_name, /Reset
RETURN, 1
END
;*****
; User Function UDemo_Usr
;*****
PRO UDemo_Usr, tool_name
; Forward declaration of TM routines
; -----
DECLARE FUNC, TmGetAttribute
DECLARE FUNC, TmSetAttribute
DECLARE FUNC, TmGetTop
DECLARE FUNC, WoGenericDialog
DECLARE FUNC, GetToolName
DECLARE FUNC, GU_newParmlist_component
DECLARE FUNC, GU_read_Parmlist_component
DECLARE FUNC, GU_updateParmlist_component
buttons = LONARR(3)
dialog = WoGenericDialog(TmGetTop(tool_name), $
 layout,'Display_Parm_StatsCB', $
 Title = 'Select Parameter', $
 /OK, /Apply, /Cancel, /NoDestroy, $
 Default = 1, Buttons = buttons)
Status = TmSetAttribute(tool_name, 'DIALOG_IDS', $
 'GET_PARM_STATS_USR_GUI', dialog)
; layout of widgets
lly = WwLayout(layout, /Top,/Left, /Bottom, /Form, /Frame)
rly = WwLayout(layout,/Top,/Bottom,/Form,/Frame, Left=lly)
dlay = WwLayout(layout,/Top,/Right,/Bottom,left=rly, $

```

```

 /Form,/Frame)
grael_name = TmSetAttribute(tool_name, 'TM', 'CURR_GRAEL', $
 'GET_PARM_STATS_USR')
ifc = TmSetAttribute(tool_name, 'GET_PARM_STATS_USR', $
 'CURR_IFC', 'B')
curr_axis = TmSetAttribute(tool_name, 'GET_PARM_STATS_USR', $
 'CURR_AXIS','GET_PARM_STATS_USR_1')
parm_wids = GU_newParmlist_component(tool_name, parent=l1lay)
rv = GU_updateParmlist_component(tool_name, $
 wids=parm_wids)

; text area
wid = WwText(r1lay, 'NoOpCB', Cols=30, Row=35, $
 Foreground='Black', Background='', /Bottom)
; drawing area
windowID = -1
wdraw = WwDrawing(d1lay, windowID, $
 'udemo_TSDrawCB', [512,512], [512,512], $
 area=area, /noscroll)
whandler = WwHandler(area,'udemo_handlerCB', $
 'ButtonPressMask')
rv = TmSetAttribute(tool_name,'PARM_STATS_WIDS', $
 'TEXT_WID',wid)

; Save widget IDs to set current values when OK or Apply
; selected
FOR i = 0, 1 DO $
 rv = WwSetValue(buttons(i),Usdata=LIST(tool_name, $
 parm_wids))
status = WwSetValue(dialog, /Show)
END

```

---

## TS-WAVE Data Handlers

### Overview

There are as many different data file formats as there are industries and companies collecting data. Rather than support only a few common formats and force you to modify your files to accommodate one of these, TS-WAVE provides a method for you to customize data access to your specifications. This method is called the Data Handler API.

The Data Handler API consists of a set of support routines and a highly flexible data structure: the fileinfo associative array (if you are not familiar with the PV-WAVE associative array, please refer to the PV-WAVE Programmer's Guide, "*Chapter 6, Working with Structures*").

There are only two routines that you are required to create to be able to access your data from TS-WAVE. These are: `<dhtype>_ReadFile` and `<dhtype>_ReadData`, where `<dhtype>` is the name of your Data Handler. A third routine, `<dhtype>_WriteFile`, is only required if you wish to output data from TS-WAVE to an external file using your data file format.

### Required Routines Overview

`<dhtype>_ReadFile` — Initializes the settings needed to actually extract data from the file. This routine is where you would implement any user interfaces required to allow users to select settings specific to your data format. This routine does not actually read data into TS-WAVE. Its primary purpose is to set up the parameter list so users can select specific parameters for display.

---

**TIP** ASCII files are generally read in their entirety on the first data access.

---

`<dhtype>_ReadData` — This is the routine that actually reads and returns parameter values from your data file. The parameters are only read when they are displayed in a graph object or contour object or written to an output file, such as a TAB file.

Information about the data file is passed to your Data Handler in the fileinfo associative array. An associative array is a very flexible PV-WAVE data structure that stores information in key:value pairs and allows you to add keys as you need them.



## Data Handler Calling Sequence

When a user selects **File=>Open Data Source** from the main interface, TS-WAVE displays a file selection dialog. Once a file is selected the **Name and Type of Data Source** dialog appears. When the user selects ‘OK’ in this dialog, your **<dhtype>\_ReadFile** routine is called with an initialized fileinfo associative array.

## The fileinfo Associative Array

The fileinfo associative array that is passed to your **<dhtype>\_ReadFile** routine comes with some pre-defined keys, some have values already filled in and some are set to default values. With a few exceptions, you may use these keys to store whatever information your Data Handler needs. You are free to add additional keys if you need to. Below is a list of the keys that already exist when your **<dhtype>\_ReadFile** routine is called along with the values each contains. Keys set by TS-WAVE should not be modified.

---

**NOTE** Associative array keys are case sensitive.

---

## Required Keys

**PARAMETER\_LIST** — Your **<dhtype>ReadFile** routine MUST fill this key with a string array of the parameters available in the data file.

## Keys Set by TS-WAVE

**FILE\_FULLNAME** — A string set by TS-WAVE to the full path and filename of the data file.

**FILEPATH** — A string set by TS-WAVE to the path of the selected data file.

**FILENAME** — A string set by TS-WAVE to the name of the data file.

**FILETYPE** — A string set by TS-WAVE to the name of your Data Handler. This is what TS-WAVE uses in place of the word '<dhtype>' that you see in this document. For details on how to name your Data Handler see, [Setting Up Your Data Handler](#) on page 199.

**Listed below are the user keys you may modify or ignore and some suggested uses of each:**

**IDENTIFIER** — A string that can be used to further identify the data set. This string will be printed in the Tab File header when parameters for this data set are exported.

**UNITS** — A string array of units corresponding to the `PARAMETER_LIST`. Defaults to a blank string.

**SAMPRATE** — A string containing the sample rate associated with a file containing time-series data. This value will be output in the header of Tab Files created from this Data Source. Defaults to '1'.

**FILE\_SIZE** — A long integer value containing the size of the data file. Defaults to 0.

**START\_TIME** — A PV-WAVE Date/Time structure (!DT) containing the starting time of data files containing time-series data. Defaults to the current date/time. (For more information about PV-WAVE Date/Time variables, see the PV-WAVE User's Guide, "*Chapter 8, Working with Date/Time Data*".

**STOP\_TIME** — A PV-WAVE Date/Time structure containing the stop time of data files containing time-series data. Defaults to current date/time.

**DT\_BASE** — A PV-WAVE Date/Time structure used for date/time conversions. Defaults to the PV-WAVE !Dt\_base system variable.

The entire fileinfo associative array is stored by TS-WAVE and passed to your `<dhtype>_ReadData` routine when a parameter in the data file is accessed. If your `<dhtype>_ReadData` routine requires additional information you may add it to fileinfo using a command similar to:

```
fileinfo("NEW_KEY") = new_value
```

Since the associative array is passed by reference, any changes you make to it are automatically picked up by TS-WAVE. For more information on parameter passing mechanisms, see the PV-WAVE Programmer's Guide, "*Chapter 9, Writing Procedures and Functions*."

---

## Setting Up Your Data Handler

TS-WAVE relies on two files to properly initialize the Data Handlers. Both of these files are only read at startup, so you'll have to restart TS-WAVE to incorporate any changes you make. The two files are the datahandler.ads resource file, located in <tswave\_dir>/resource, and datahandler.list, located in <tswave\_dir>/datahandlers.

### The datahandler.list File

This file is a list of the available Data Handlers and the directories in which TS-WAVE should look for the routines for each. This list is what appears in the **Format** pull-down menu in the **Name and Type of Data Source** dialog.

### Example

Here is an example of the datahandler.list file.

```
ldf : $TSWAVE_ROOT/datahandlers/ldf
ascii : $TSWAVE_ROOT/datahandlers/ascii
tab : $TSWAVE_ROOT/datahandlers/tab
sdh : $TSWAVE_ROOT/datahandlers/sampledathandler
```

In this case, *ldf* is the name of a specific Data Handler and the *ldf\_readfile.pro* and *ldf\_readdata.pro* are located in the directory *\$TSWAVE\_ROOT/datahandlers/ldf*. The variable *\$TSWAVE\_ROOT* is automatically defined to be the directory in which TS-WAVE was installed. If you wish, you may use a different path instead of *\$TSWAVE\_ROOT*. Simply add your new Data Handler to the list using the same format.

### The datahandler.ads Resource File

The datahandler.ads resource file provides TS-WAVE with additional information about your Data Handler.

### Example

Here is an example of the datahandler.ads file.

```
!-----
! Name: datahandler.ads
! Description: Resource settings for TS-WAVE Data Handler.
```

```

!-----
!-----
! Specify Data Handler extensions
!-----

ldf: ldf
ascii: dat
tab: tab
sdh: sdh
DH_Root:
DH_Subdir: datahandlers
DH_Version: 4.0
DH_TypeListFile: datahandler.list
!-----
! Index File Creation Attributes
!-----
DH_Index_Version: 2.0
Update_Previous_IDX_Files: 0
DH_Index_Ext: idx

```

The first block of this file tells TS-WAVE which Data Handler to associate with specific file extensions. This causes the **Format** pull-down menu on the **Name and Type of Data Source** dialog to appear with the appropriate Data Handler selected for the data file selected by the user.

The second block contains information to help TS-WAVE find the list of available Data Handlers. The resource `DH_Root`, if left blank, defaults to the TS-WAVE installation directory. The settings above tell TS-WAVE to look for a file called `datahandler.list` in the `<tswave_dir>/datahandlers` directory. The resource `DH_Version` is included for backward compatibility.

The final block is used during the generation of index files. An index file is a special file that a Data Handler can use to speed data access. For example, the *ldf* file format contains blocks of identical parameters called 'runs'. The first time an *ldf* file is read, an index file is generated which holds the byte offsets of the beginning of each 'run' block. This allows the *ldf* Data Handler to jump directly to the beginning of a run to find a specific parameter instead of scanning the entire file, thus speeding up data access. Individual Data Handler developers are free to

implement tricks of this sort since the only requirement is that the two required routines follow the Data Handler API.

## Data Handler Example Overview

Since the methods used to read data from a data file are highly specialized for the format of that file, our example is largely pseudo code. However, the generalized tasks are the same for any data file. Here we focus only on the interaction with TS-WAVE. Your actual Data Handler would make use of PV-WAVE's Data Connectivity (DC) and file manipulation (POINT\_LUN, etc.) routines. For more information on data connectivity in PV-WAVE, see the PV-WAVE Programmer's Guide, "*Chapter 8, Working with Data Files*".

In this example, our Data Handler is called *ex1*. This name must be unique from all other Data Handlers and replaces **<dhtype>** in the previous sections.

---

**NOTE** Associative array keys are case sensitive.

---

## Example: **<dhtype>**\_ReadFile Function

```
FUNCTION ex1_readfile, fileinfo, tool_name

; Get the file name from the fileinfo associative array
;filename = fileinfo("FILE_FULLNAME")

; Open and read the file

...

; Gather the parameter names into the variable parmlist

...

; Fill the parameter list in the fileinfo Associative
; Array. At this point, the responsibilities of the
; ex1_readfile routine are complete, it has only
; to return a status value. For this example we
; will fill in a few more values and create some of
; our own.

fileinfo('PARAMETER_LIST') = parmlist
fileinfo('START_TIME') = starttime
fileinfo('STOP_TIME') = stoptime

; Some new key:value pairs for fileinfo to help us access the
;data in ex1_ReadData.
```

```

fileinfo('CURR_RUN') = current_run
fileinfo('RUN_OFFSET') = run_offset

; Return the status, 1=Success, 0=Failure
RETURN, 1
END

```

## Example: <dhtype>\_ReadData Function

```

FUNCTION ex1_readdata, fileinfo, parm, tool_name

; Use the information in the fileinfo Associative
; Array to gather the data for the parameter
; specified in the parm argument.

Filename = fileinfo("FILE_FULLNAME")
Offset = fileinfo("RUN_OFFSET")

; We can use the offset value to jump to the proper
; block in the file using the PV-WAVE POINT_LUN
; routine, not shown. For this example, we will
; assume that the file contains both data and time
; values with the time values represent elapsed
; seconds from the base time, stored as double
; precision floats.

...

; Read the data and time values into the variables
; values and time.

...

; Convert the time values into an array of PV-WAVE Date/Time
; structures using the PV-WAVE SEC_TO_DT function.

...

; Return the values and time variables in an associative array
; containing the keys 'DATA' and 'TIME'.
RETURN, ASARR('DATA', values, 'TIME', time)

; If the data file did not contain time values, we would have
; constructed the associative array with the following command:

RETURN, ASARR('DATA', values, 'TIME', $
 LINDGEN(N_ELEMENTS(values)))

END

```

The **ex1\_ReadFile** and **ex1\_ReadData** functions must be placed into separate files named `ex1_readfile.pro` and `ex1_readdata.pro`, respectively, and compiled into PV-WAVE .cpr files. For an example of compiling PV-WAVE procedures, see ‘**Steps 1-9**’, in [Creating a Derived Parameter](#) on page 183, or refer to the COMPILE command in the PV-WAVE Reference.

Now we will add information about the *exI* Data Handler to the `datahandler.list` and `datahandler.ads` files.

**datahandler.list:**

```
ldf : $TSWAVE_ROOT/datahandlers/ldf
ascii : $TSWAVE_ROOT/datahandlers/ascii
tab : $TSWAVE_ROOT/datahandlers/tab
sdh : $TSWAVE_ROOT/datahandlers/sampledathandler
ex1 : $TSWAVE_ROOT/datahandlers/example1
```

The change above tells TS-WAVE to look for the *exI* Data Handler files in the `<tswave_dir>/datahandlers/example1` directory.

**datahandler.ads:**

```
!-----
! Specify datahandler extensions
!-----
ldf: ldf
ascii: dat
tab: tab
sdh: sdh
ex1: ex1
```

The change above tells TS-WAVE to associate files with the `.ex1` extension with the *exI* Data Handler.

Once TS-WAVE is restarted, the *exI* Data Handler will be available for use.

---

## The Data Handler Functions

This section contains detailed descriptions of the Data Handler functions distributed with TS-WAVE.

---

### <dhtype>\_ReadFile Function

Opens the data file of <dhtype> format and initializes any settings needed to read data. It can also be used to display a user interface that allows users to select settings specific to the format, such as select a time slice from the data file, a subset of parameters, or sampling rates. The primary responsibility of this function is to set the Parameter list, which allows a user to choose a parameter from the file.

#### Usage

```
status = <dhtype>_READFILE(fileinfo, tool_name)
```

#### Input Parameters

**fileinfo** — A PV-WAVE associative array containing information about the source file opened. Since fileinfo is an associative array, your Data Handler can add additional keys.

**tool\_name** — A string containing the unique name of a VDA Tool. The information is set by the calling function.

#### Returned Value

**status** — Integer value indicating success or failure; 0 indicates failure, 1 indicates success.

#### Discussion

<dhtype>\_ReadFile reads and gathers header and setup information that is needed to read parameter data from the file at the time in which the parameter was requested.



---

## **<dhtype>\_ReadData Function**

Returns the data values for a specific parameter.

### **Usage**

*data* = <dhtype>\_READDATA(*fileinfo*, *parm*, *tool\_name*)

### **Input Parameters**

*fileinfo* — A PV-WAVE associative array containing information about the source file opened. Since *fileinfo* is an associative array, your Data Handler can add additional keys.

*parm* — String containing the parameter name.

*tool\_name* — String containing the unique VDA Tool name.

### **Returned Value**

A PV-WAVE associative array with two keys: **DATA** and **TIME**.

**DATA** — Contains the actual data values for the parameter.

**TIME** — An array of PV-WAVE Date/Time structures corresponding to the data values.

or

**TIME** — An indexed array of integers with the same number of elements as the parameter's data values.

### **Discussion**

<dhtype>\_ReadData retrieves parameter data values and optional time values associated with the data.

---

## **<dhtype>\_WriteFile Function**

An optional routine to write out data to a specific <dhtype> file format.

## Usage

*status* = <dhtype>\_WRITEFILE(*tool\_name*, *filename*)

## Input Parameters

*tool\_name* — String containing the unique VDA Tool name.

*filename* — String containing the filename selected from the file selection interface.

## Returned Value

*status* — Integer value indicating success or failure; 0 indicates failure, 1 indicates success.

## Discussion

TS-WAVE's unique tool name and the file name selected from the File Selection interface is passed as the input parameter for <dhtype>\_WriteFile to be used to generate a file of <dhtype>'s format.

---

## ***Tips For Creating Data Handlers***

Use this section as a quick reference for tips on creating Data Handlers.

### **Batch Processing**

When TS-WAVE is running a Batch Job, there is no Graphical User Interface (GUI) created. It loads the template you created when you created the Batch Jobfile and does not call your **<dhtype>\_ReadFile** routine. It does, however, call your **<dhtype>\_ReadData** routine. For this reason, if your Data Handler requires user input you should not attempt to create any GUIs in your **<dhtype>\_ReadData** routine, only in your **<dhtype>\_ReadFile** routine.

### **Copying and Modifying Data Sources**

The File menu on the main TS-WAVE interface contains a Copy Open Source option and a Modify Open Source option. When a user selects either of these your **<dhtype>\_ReadFile** routine is called directly, without an intervening file selection dialog. The fileinfo associative array passed to your routine is the one associated with the Data Source the user has chosen to copy or modify. In the rare case that your routine needs to change its behavior based on whether a Data Source is being copied or modified, as opposed to just being read. TS-WAVE can be instructed to set a /Copy keyword in the call to your **<dhtype>\_ReadFile** routine during Copy or Modify operations. To tell TS-WAVE that your Data Handler needs this keyword passed to it, modify the following section of the datahandler.ads resource file to include your Data Handler.

```
!-----
! Copy Keyword Required
!-----
```

```
ldf_Copy: 1
rrb_Copy: 1
bdf_Copy: 1
```

Add the following line to the ‘Copy Keyword Required’ section:

```
<dhtype>_Copy: 1
```

Add the Copy keyword to your routine’s declaration:

```
FUNCTION <dhtype>_ReadFile, fileinfo, tool_name, Copy=copy
```

Then use the PV-WAVE KEYWORD\_SET() function to test its value:

```
IF KEYWORD_SET(Copy) THEN BEGIN
```

```
...
```

```
ENDIF
```

If your Data Handler does not need to do any special processing in these cases then you may leave the ‘Copy Keyword Required’ section of the datahandler.ads file as it is.

The discussion of the special processing that you might need to do is intentionally vague. This is because it is very Data Handler specific. The Data Handlers that do use it generally use it to signal that a particular key in the fileinfo associative array needs to be reset to some value to inform other routines in the Data Handler what is going on. The odds are good that your particular Data Handler will not need to use this keyword, but if you find that you do, it is there.

For more information on PV-WAVE keywords, see the KEYWORD\_SET routine in the online help or see the PV-WAVE Programmer's Guide, *Chapter 10*, “*Checking For Parameters*”.

---

## ***TS-WAVE's Utility Functions***

TS-WAVE's utility functions are comprised of DataManager functions (DM\_\*), User Interface functions (GU\_\*), PrintManger functions (PM\_\*), and Data Handler functions (DH\_\*). These utility functions make it easy to manage data and integrate elements into a graphical user interface (GUI), and retrieve print and print manager statuses. Each utility function is described in the next section.

---

## ***TS-WAVE's DataManager Functions***

The DataManager functions (DM\_\*) make it easier to handle data and build interfaces for your user function. The DataManager routines include functions that allow a derived parameter to be accessed easily from other TS-WAVE routines, give you access to the source list and parameter list, and let you retrieve data values associated with a parameter (including properly created derived parameters).

TS-WAVE's DataManager functions are:

- *DM\_addFCN Function*
- *DM\_addPSrc Function*
- *DM\_getSrcList Function*
- *DM\_getSrc Function*
- *DM\_getParmList Function*
- *DM\_getParm Function*

---

### ***DM\_addFCN Function***

Adds the derived parameter to the Function List, making the derived parameter available to other TS-WAVE functions.

#### **Usage**

*rv* = DM\_addFCN(*tool\_name*, *newparm*, *exe\_cmd*)

#### **Input Parameters**

*tool\_name* — String containing the unique VDA tool name.

***newparm*** — Name of the new derived parameter.

***exe\_cmd*** — String containing the commands to create the derived parameter that will be executed by the PV-WAVE's EXECUTE command. For more information, see PV-WAVE's EXECUTE command in the PV-WAVE online help.

## Keywords

None.

## Returned Value

***rv*** — Integer value indicating success or failure; 0 indicates failure, 1 indicates success.

## Example

See the Function 1 in the [Creating a Derived Parameter](#) section of this chapter.

---

## ***DM\_addPSrc Function***

Adds the source names associated with a derived parameter to the DERIVED source list, making the Source IDs for derived parameter available to other TS-WAVE functions.

## Usage

***rv*** = DM\_addPSrc(*tool\_name*, *newparm*, *src\_names*)

## Input Parameters

***tool\_name*** — String containing the unique VDA tool name.

***newparm*** — Name of the new derived parameter.

***src\_names*** — String array containing the list of open Source IDs.

## Keywords

None.

## Returned Value

*rv* — Integer value indicating success or failure; 0 indicates failure, 1 indicates success.

## Example

See the Function 1 in the [Creating a Derived Parameter](#) section of this chapter.

---

## ***DM\_getSrcList Function***

Returns a string array containing the list of open Source IDs.

### Usage

```
src_list = DM_getSrcList(tool_name)
```

### Input Parameters

*tool\_name* — String containing the unique VDA tool name.

### Keywords

None.

## Returned Value

*src\_list* — String array containing the list of open Source IDs.

---

## ***DM\_getSrc Function***

Returns a string array containing all of the available information for the Source ID.

### Usage

```
fileinfo = DM_getSrc(tool_name, src_name)
```

## Input Parameters

*tool\_name* — String containing the unique VDA Tool name.

*src\_name* — String containing the open Source ID.

## Keywords

None.

## Returned Value

*fileinfo* — An associative array containing Source ID information for the *src\_name*, such as ‘FILE\_FULLNAME’ which contains the file location, ‘FILETYPE’ containing the format type, and ‘PARAMETER\_LIST’ containing the list of available parameters. See [The fileinfo Associative Array](#) on page 197 for a list of associative array keys.

*0* — Indicates failure.

---

## DM\_getParmList Function

Returns a string array of available parameters for a given Source ID.

## Usage

```
parm_list = DM_getParmList(tool_name, src_name)
```

## Input Parameters

*tool\_name* — String containing the unique VDA tool name.

*src\_name* — String containing the list of open Source IDs.

## Keywords

None.

## Returned Value

*parm\_list* — String array containing the names of the available parameters.

*-1* — Indicates failure.



---

## ***DM\_getParm Function***

Returns the data and time values associated with a specified parameter from a specified Source ID.

### **Usage**

*data\_values* = DM\_getParm(*tool\_name*, *src\_name*, *parameter*)

### **Input Parameters**

*tool\_name* — String containing the unique VDA tool name.

*src\_name* — String containing the name of the Source ID file.

*parameter* — String containing the open Source ID.

### **Keywords**

None.

### **Returned Value**

*data\_value* — A PV-WAVE associative array with two keys: ***DATA*** and ***TIME***.

***DATA*** — An array of scalar data values.

***TIME*** — A PV-WAVE DT (date-time) array of times corresponding to the data values.

or

***TIME*** — Or array of scalar values with same number of elements as data values.

---

## TS-WAVE's User Interface Functions

The User Interface functions (GU\_\*) allow easy integration of elements into a GUI, and can also create a list widget and later update or retrieve parameters contained in it.

TS-WAVE's User Interface functions are:

- *GU\_newParmlist\_Component Function*
- *GU\_updateParmList\_Component Function*
- *GU\_readParmList\_Component Function*

---

### GU\_newParmlist\_Component Function

Creates a new parameter list in an existing layout.

#### Usage

*wids* = GU\_newParmlist\_Component(*tool\_name*, parent=*parent*)

#### Input Parameters

*tool\_name* — String containing the unique VDA tool name.

#### Keywords

*parent* — The widget ID of the parent widget (in most cases, a layout ID).

#### Returned Value

*wids* — A structure containing the widget IDs created. These widget IDs can be used to directly interact with the elements of the Parameter List component. The table below lists the tags and their associated objects.

*-1* — Indicates failure.

| Tag Name       | Object                         |
|----------------|--------------------------------|
| parmtxt_id     | Text ID for selected parameter |
| srclist        | Source Selection List ID       |
| parmlist_id    | Parameter List ID              |
| sort_button_id | Sort parmlist button ID        |

---

**NOTE** Once the main component has been created, GU\_updateParmList\_Component must be called to populate the widgets with Source IDs and parameters.

---



---

## ***GU\_updateParmList\_Component Function***

Updates the ParmList\_Component with Source IDs and parameters. If a Source ID is specified, it is selected, and its parameters are used to populate the Parameter List box, otherwise the first Source ID in the list is selected.

### **Usage**

```
rv = GU_updateParmList_Component(tool_name, wids=wids, src=src, $
parm=parm)
```

### **Input Parameters**

***tool\_name*** — String containing the unique VDA tool name.

### **Keywords**

***wids*** — Structure of widgets returned by the GU\_newParmList\_Component function.

***src*** — Optional keyword to pass a specified Source ID with which to update the Parameter list widget.

***parm*** — Optional keyword to pass a specified parameter to be highlighted in the Parameter list widget.

## Returned Value

*rv* — Integer value indicating success or failure; 0 indicates failure, 1 indicates success.

---

## ***GU\_readParmList\_Component Function***

Reads the *ParmList\_Component* to obtain the selected Source ID and Parameter name.

## Usage

*rv* = GU\_readParmList\_Component(*tool\_name*, *wids* = *wids*)

## Input Parameters

*tool\_name* — String containing the unique VDA tool name.

## Keywords

*wids* — Structure of widgets returned by the GU\_newParmList\_Component routine.

## Returned Value

*rv* — Associative array containing the Source IDs and parameter name selected in the ParmList\_Component in the keys SOURCE and PARAMETER.

## Discussion

Extracting the Source ID and parameter from the Returned Value:

```
Source_name = rv('SOURCE')
```

```
Parm_name = rv('PARAMETER')
```

---

## ***TS-WAVE's PrintManager Functions***

The PrintManager functions (PM\_\*) make it easy to retrieve printer and printer driver statuses.

TS-WAVE's PrintManager functions are:

- *PM\_getDefaultDriver Function*
- *PM\_getCurrentDriver Function*
- *PM\_getDefaultPrinter Function*
- *PM\_getCurrentPrinter Function*
- *PM\_getDriverList Function*
- *PM\_getPrintFile Function*
- *PM\_isPrintToFileEnabled Function*
- *PM\_setPrintFile Function*

---

### ***PM\_getDefaultDriver Function***

Retrieves the name of the default printer driver.

#### **Usage**

*defd* = PM\_getDefaultDriver(*tool\_name*, *DefaultDriver*)

#### **Input Parameters**

***tool\_name*** — String containing the unique VDA tool name.

***DefaultDriver*** — String containing the name of the default printer driver to use if a default has not been assigned.

Acceptable values are: PS, CGM, PCL, HPGL.

#### **Keywords**

None.

## Returned Value

*defd* — Default printer driver or the value of `DefaultDriver` (if a driver was not previously set and the parameter `DefaultDriver` is passed in).

---

## ***PM\_getCurrentDriver Function***

Retrieves the name of the current printer driver.

### Usage

*currd* = `PM_getCurrentDriver(tool_name, CurrentDriver)`

### Input Parameters

*tool\_name* — String containing the unique VDA tool name.

*CurrentDriver* — String containing the name of the current printer driver to use if a default has not been assigned.

Acceptable values are: PS, CGM, PCL, HPGL

### Keywords

None.

## Returned Value

*currd* — Current printer driver or the value of `CurrentDriver` (if a driver was not previously set and the parameter `CurrentDriver` is passed in).

### Example

```
;Gets the default printer driver
defd = PM_getDefaultDriver(tool_name)

;Now get the current driver. If one is not set, use the default
;driver.
currd = PM_getCurrentDriver(tool_name, defd)
```

---

## ***PM\_getDefaultPrinter Function***

Retrieves the current printer name.

### **Usage**

```
defp = PM_getDefaultPrinter(tool_name)
```

### **Input Parameters**

*tool\_name* — String containing the unique VDA tool name.

### **Keywords**

None.

### **Returned Value**

*defp* — String containing the name of the default printer.

---

## ***PM\_getCurrentPrinter Function***

Retrieves the current printer name.

### **Usage**

```
currp = PM_getCurrentPrinter(tool_name, CurrentPrinter)
```

### **Input Parameters**

*tool\_name* — String containing the unique VDA tool name.

*CurrentPrinter* — String containing the name of the current printer to use if a one has not been assigned.

### **Keywords**

None.

## Returned Value

*currp* — Current printer or the value of CurrentPrinter (if a printer was not previously set and if the parameter CurrentPrinter is passed in).

## Example

```
Gets the default printer driver resource
defp = PM_getDefaultPrinter(tool_name)
currp = PM_getCurrentPrinter(tool_name, defp)
```

---

## ***PM\_getDriverList Function***

Retrieves the list of printer drivers.

## Usage

*dlist* = PM\_getDriverList(*tool\_name*)

## Input Parameters

*tool\_name* — String containing the unique VDA tool name.

## Keywords

None.

## Returned Value

*dlist* — Array of strings of printer drivers.

## Example

```
PRINT, PM_getDriverList(tool_name)
```

## Output

```
PS WMF CGM PCL HPGL
```



---

## ***PM\_getPrintFile Function***

Retrieves the name of the file to which output will be written if ‘Print to file’ is enabled .

### **Usage**

*pfname* = PM\_getPrintFile(*tool\_name*, Default=*default*)

### **Input Parameters**

*tool\_name* — String containing the unique VDA tool name.

### **Keywords**

*default* — A string containing the name of the file to use if no default printer file is set.

### **Returned Value**

*pfname* — The name of the file to which output will be written if print to file is enabled.

### **Example**

```
status = PM_isPrintToFileEnabled(tool_name)
IF status EQ 1 THEN $
 pfname = PM_getPrintFile(tool_name, Default='temp.ps')
```

---

## ***PM\_isPrintToFileEnabled Function***

Checks whether print to file is enabled. If print to file is enabled, the next time print commands are executed, output is written to the filename returned by PM\_getPrintFile.

### **Usage**

*status* = PM\_isPrintToFileEnabled(*tool\_name*)

## Input Parameters

*tool\_name* — String containing the unique VDA tool name.

## Keywords

None.

## Returned Value

*status* — Integer value indicating success or failure; 0 indicates failure, 1 indicates success.

---

## ***PM\_setPrintFile Function***

Sets the name of the file to which output will be written if print to the name specified.

## Usage

*pfname* = PM\_setPrintFile(*tool\_name*, *PrintFileName*)

## Input Parameters

*tool\_name* — String containing the unique VDA tool name.

*PrintFileName* — String containing the name of the print file.

## Keywords

None.

## Returned Value

*pfname* — The name of the previously set print file.

---

# TS-WAVE's Miscellaneous Utility Functions

There are many useful utility functions that should be explored by developers located in <tswave\_dir>/datahandlers/lib/std, such as dh\_get\_time\_string, dh\_minmax, dh\_validata\_data, lm\_\*, and str\_is\_num. Some source code for these functions is included with the TS-WAVE distribution.

---

# TS-WAVE System Variables

This section describes system variables available to TS-WAVE developers writing user functions, Data Handlers and utility routines. System variables are variables created internally in a TS-WAVE session that are preceded by a '!' character. These variables can be read the same as any other variable. If a system variable is not read-only, it may also be assigned a new value. Read-only variables are indicated in the table below with (RO).

## Example

Here's an example of how to print and extract the value of a system variable.

```
PRINT, !TSWAVE_Root
;output C:\VNI\tswave-4_0
myfile = !TSWAVE_Root + !Dir_Sep + 'myfile.txt'
PRINT, myfile
;output C:\VNI\tswave-4_0\myfile.txt
```

| System Variable Name | Description                            | Default Value              |
|----------------------|----------------------------------------|----------------------------|
| !Cm2in               | Centimeters to inches conversion. (RO) | 0.393541877                |
| !Dir_Sep             | OS-specific Directory separator. (RO)  | Windows: '\',<br>UNIX: '/' |
| !In2cm               | Inches to centimeters conversion. (RO) | 2.54102564                 |
| !Invalid             | Invalid data value to be ignored.      | -999999.0                  |

| System Variable Name | Description                                                                                         | Default Value                                                                        |
|----------------------|-----------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| !Diag                | Flag to print diagnostic messages (dprint command output). 1=true, 0=False                          | 0                                                                                    |
| !LogDiag             | Used only if !Diag is 1. Prints diagnostic info output by dprint command to log file.               | 0                                                                                    |
| !Messages            | Determines whether TS-WAVE should print error messages or pop up WwAlert dialogs when errors occur. | 'ON'                                                                                 |
| !Mode                | Internal system variable that reports whether TS-WAVE is running in Batch or Interactive mode.      | 'I' if TS-WAVE is started in interactive mode. 'B' if TS-WAVE is started batch mode. |
| !Path_Sep            | OS specific path separator character. (RO)                                                          | Windows: ';',<br>UNIX: ':'                                                           |
| !Prompt              | Session prompt.                                                                                     | 'TS-WAVE -> '                                                                        |
| !TSWAVE_Date_Fmt     | Format to be used for date output.                                                                  | 'MM/DD/YYYY'                                                                         |
| !TSWAVE_Release      | Structure that contains the most current TS-WAVE release version and date.(RO)                      |                                                                                      |
| !TSWAVE_Root         | TS-WAVE root directory.                                                                             | <tswave_dir>                                                                         |
| !TSWAVE_Time_Fmt     | Format to be used for time output.                                                                  | 'HH:MM:SS.sss'                                                                       |

---

## Customization Options

This section describes how you can customize the behavior of TS-WAVE for your specific needs. Data Handlers are described in [TS-WAVE Data Handlers](#) on page 196, and UserFunctions are described in [Writing a User Function](#) on page 181.

### Show Info

Use the TS-WAVE **View->Show Info** menu to set and display a footer on a TS-WAVE Page. This output can be customized to your site specifications. The PV-WAVE source code is provided and resides in

```
<tswave_dir>\util\ vzshowinfo.pro
```

Routines DM\_getSrcList and DM\_getSrc are used for gathering information and are used in the vzshowinfo supplied. DM\_getSrcList returns the list of available Source IDs. Routine DM\_getSrc returns the file information associative array populated by the Data Handler. For more information on DataManager functions, see [TS-WAVE's DataManager Functions](#) on page 209.

---

**CAUTION** Before modifying **vzshowinfo.pro** and **vzshowinfo.cpr** be sure to back up the originals.

---

### Customized Startup Command Files

Users can customize their default TS-WAVE startup environment with user-defined startup files. Custom startup files can be used to do such things as adding additional TS-WAVE environment variables, expanding WAVE\_PATH or printing customized startup messages. For more information, see [Understanding the WAVE\\_PATH Environment Variable](#) on page 176.

### TS-WAVE Runtime Mode Startup File

The file defined by the WAVE\_RT\_STARTUP environment variable is used by both TS-WAVE and PV-WAVE's runtime modes. If WAVE\_RT\_STARTUP is not defined, the default startup file <vni\_dir>/wave/lib/std/rtwavestartup.cpr will be used. If WAVE\_RT\_STARTUP is defined, it must define a PV-WAVE procedure that has been compiled into a .cpr file.

Users can either add custom commands directly to `rtwavestartup.pro` and generate a new `rtwavestartup.cpr` or individual users can create their own private versions of runtime startup files; the latter method is recommended.

### To set up your private customized runtime mode startup file:

- Step 1** Create and compile a PV-WAVE procedure that contains custom commands to be executed when TS-WAVE starts up. For this example, we will create a startup file named `myrtstartup.pro` that for Windows is in `C:\usrfiles\` and for UNIX in `/usr/usrfiles/`. This startup file simply prints a statement. For information on compiling, refer to [Compiling PV-WAVE Routines](#) on page 177.

```
PRO myrtstartup
 PRINT, "This is my custom startup message."
END
```

- Step 2** Create the system variables that define the name and path of your `WAVE_RT_STARTUP` procedure.

**Windows:** `set WAVE_RT_STARTUP=myrtstartup`

`set WAVE_PATH C:\usrfiles\`

**UNIX (chs):** `setenv WAVE_RT_STARTUP myrtstartup`

`setenv WAVE_RT_PATH /usr/usrfiles/`

- Step 3** Start TS-WAVE in runtime mode. Your startup commands will execute before TS-WAVE starts up.

## TS-WAVE Developer Mode (NoBlock) Startup File

The startup file used by the NoBlock mode is a PV-WAVE command Batch File (not a PV-WAVE procedure file) which does not need to be compiled. This startup file must, at a minimum, contain the commands defined in

`<tswave_dir>/lib/tswave_start.cmd` for TS-WAVE to initialize properly.

The environment variable that defines the path and filename of your startup file is:

**Windows:** `WAVE_STARTUP`

**Note for Windows users:** `WAVE_STARTUP` is used by both TS-WAVE's and PV-WAVE's NonBlocking modes.

**UNIX:** `TS_STARTUP`

**Note for UNIX users:** WAVE\_STARTUP, if set under UNIX, is ignored by TS-WAVE.

If the startup environment variable is not defined, the default startup file <tswave\_dir>/lib/tswave\_start.cmd is used. Users can add custom commands directly to this file or individual users can create their own private versions of startup files; the latter method is recommended.

### **To create your customized Developer mode startup file:**

**Step 1** Copy the file <tswave\_dir>/lib/tswave\_start.cmd to the directory where your private user startup file will be located and edit this file to include your custom code. For this example, use the directory /usr/usrfiles/ for UNIX and C:\userfiles\ for Windows.

**Step 2** Create the system variable that defines the name and path of your TS-WAVE startup file.

**Windows:** set WAVE\_STARTUP=C:\usrfiles\tswave\_start.cmd

**UNIX csh:**

setenv TS\_STARTUP /usr/usrfiles/tswave\_start.cmd

**Step 3** Add a custom print statement to the beginning of this file.

**Step 4** Start TS-WAVE in developer's mode. Your print statement will execute before TS-WAVE starts up.

For more information on creating startup files, refer to the PV-WAVE Programmer's Guide, "Appendix B, Chapter 1, "Creating and Running a Command (Batch) File" and Chapter 9, "Compiling Procedures and Functions".

## **Resources**

The directory <tswave\_dir>/resources includes resource files that allow the user to modify many of TS-WAVE's default settings and strings for customization or internationalization purposes.

For a detailed discussion on resource files, see [Customizing TS-WAVE](#) on page 164, and PV-WAVE Application Developer's Guide, "Chapter 12, Building VDA Tools", "Using Resources and Strings in PV-WAVE Applications".





# TS-WAVE Index

## A

- adding a parameter 120
- Analyze Menu 159
  - Standard Functions 159
    - Bias 159
    - Difference 159
    - Differentiate 160
    - FFT 160
    - Gain 160
    - Gauss Fit 161
    - Smooth 161
    - Trim 161
    - WildPoint 161
  - User Functions 162
    - ASCII To Binary 162
    - display\_parm\_subset\_usr 162
    - get\_parm\_stats\_usr 162
    - rudnet\_usr 162

## B

- Batch Job File 152
- Batch Processing 66, 150–158
  - adding template 67, 151
  - closing 68, 152
  - Creating a Batch Job 66, 88
  - examples 67, 154–158
  - executing 154
  - opening 68, 150, 151
  - tswavebatch.log file 68
- Batch Shell Script 153
- Batch Template File 153
- Box objects
  - creating 19

## C

- configuring your session 14, 76
  - auto redraw 15
  - auto reload 15
  - custom settings 15, 109
- Contour objects
  - attributes 130
  - Contour Attributes Interface 37, 128
  - creating 19, 36, 127
  - levels 132
- Create Menu 111
  - Add Template to Batch 151
  - Box object 137
  - Close Batch File 152
  - Close Pick File 150
  - Contour objects 127
  - Create TabData 141
  - Ellipse object 139
  - Graph objects 111
  - Header object 124
  - Line object 136
  - Open Batch File 151
  - Open Pick File 148
  - Text object 134
  - ViewTabData 148
- Customization Options 225

## D

- Data Handlers 5, 176, 196–208
  - ascii Data Handler 17
  - datahandler.ads 199
  - datahandler.list 199
  - example 201
  - fileinfo associative array 197, 204
  - LDF Data Handler 71

- ReadData function 202, 205
- ReadFile function 201, 204
- Resource files 166
- Write File routine 51
- WriteFile function 205
- Data Zoom 41–45, 106
- DERIVED Parameters 46
  - creating 74, 159–162, 183
  - Pick Files 58
  - renaming 113
  - Tab Files 147
- drawing area 4

## E

- Edit Menu 101
  - Align Graphs 102
  - Back 103
  - Copy 102
  - Cut 102
  - Delete 102
  - Deselect All 103
  - Front 103
  - Group 103
  - Object Select 101
  - Paste 102
  - Plot Attributes 102
  - Redraw 103
  - Select All 102
  - Ungroup 103
- Ellipse objects
  - creating 19
- embedded functions 125
- Environment Variables
  - TS\_STARTUP 226
  - USER\_RESPATH 64, 166
  - WAVE\_PATH 62, 176, 226
  - WAVE\_RESPATH 168
  - WAVE\_RT\_STARTUP 225
  - WAVE\_STARTUP 226
- Examine Data Points 45, 107
- execution time options 8

## F

- File Extensions
  - .cpr 177
  - .pro 177

- File Menu
  - Close Data Source 92
  - Close Page 91
  - Exit 100
  - Export 94
  - Modify Open Data Source 94
  - New 90
  - New Page 90
  - Open Session 98
  - Open Template 95
  - Page Setup 91
  - Print 98
  - Save Session 95
  - Save Template 95
- FORTTRAN Format Strings 173–174

## G

- Graph objects
  - adding an axis 33
  - adding parameters 27, 78
  - creating 19, 26, 77, 111–112
  - Curve Fit 116–118
  - X axis attributes 32, 120, 123
  - Y axis attributes 30, 31, 118
    - scaling 30

## H

- Header objects 23
  - adding text 24
  - Advanced Label Editing 24, 82, 125
  - creating 19, 23, 81, 124
  - Embedded Functions 24, 82, 125
- Help
  - PV-WAVE Online documentation 176, 181
- Help Menu 163
  - About TS-WAVE 163
  - Manuals Online 163
  - User FCN Help 163

## L

- ldf data 70
- Line objects
  - creating 19

## M

- menus 4
- message area 4

## O

- objects
  - adding 19
  - copying 20
  - deselecting 20
  - moving 20
  - removing 20, 80
  - resizing 20, 80
  - selecting 19, 80
  - selecting multiple 20, 101

## P

- Page
  - adding a second page 36
  - layout 21
  - multiple pages 21
- Parameters
  - graphing 78
- Pick Files 55, 148
  - Selecting data 57
- printing 65, 84, 98
  - Batch settings 153
  - Page Setup 91
  - print to file 99
  - selecting a printer 99
- PV-WAVE
  - Programming Language 176, 226
  - example 178

## R

- Resources 62–64, 164–170
  - color 166
  - confirm close behavior 165
  - contour 165
  - graph 165
  - graph attributes 166
  - grid 166
  - info block settings 166
  - multi-page defaults 166
  - Private User Function

- example 169
- Private User Resources 63
  - example 168
- recommended 167
- Resource Files 165
  - ep\_graphattribute.ads 62, 113, 123, 146
  - tswave.ad 100
  - tswave.ads 90
  - tswave\_colors.ads 64
  - tswave\_files.ads 87, 143
  - tswave\_print.ads 99
- USER\_RESPATH 62, 64
- .ads and .ads resources 165

## S

- Saving Sessions
  - session files 60
- Session Files 60, 98
- Shortcut Keys 171
- Show Info 225
  - custom settings
  - info block settings 110
- Source IDs 16, 17, 27
  - Batch Jobs 151
  - closing 92
  - Contour objects 129
  - copying 93
  - creating 70, 92
  - modifying 94
  - Pick Files 148
  - Tab Files 53, 55, 141–144, 147
  - Template Files 95–97
- Standard Functions 46
  - Bias 49
  - Difference 49
  - Differentiate 49
  - FFT 49
  - Gain 49
  - GaussFit 49
  - Smooth 49, 74
  - Trim 49
  - Using 48
  - Wildpoint 49
- starting TS-WAVE 7
- Startup Command Files 225–227
- stopping TS-WAVE 9
- support, technical vii

System Variables [223](#)  
!Invalid [147](#)

## T

Tab Files [51](#)  
    attributes [85](#)  
    creating [52, 85, 141](#)  
    Select Tab Attributes [52, 142](#)  
    Tab Files Tips [146](#)  
    Tab Templates [54](#)  
    viewing [87, 148](#)  
technical support [vii](#)  
Template Files [59, 95](#)  
    Source ID association [95–97](#)  
Text objects  
    creating [19, 23](#)  
tool\_name [178](#)  
TS-WAVE Developer's License [176](#)

## U

User Functions [46, 49, 162, 176, 180](#)  
    ASCII To Binary [50](#)  
    display\_parm\_subset\_usr [50](#)  
    example [188](#)  
    get\_parm\_stats\_usr [50](#)  
    rudet\_usr [50](#)  
Utility Functions [209](#)  
    DataManager Functions [209](#)  
    Miscellaneous Functions [223](#)  
    PrintManager Functions [217](#)  
    User Interface Functions [214](#)

## V

View Menu [104](#)  
    Actual Size [105](#)  
    Auto Load [111](#)  
    Auto Redraw [111](#)  
    Custom Settings [109](#)  
    Data Zoom [106](#)  
    Fit To Display [105](#)  
    Page Zoom [104](#)  
    Print CM Grid [108](#)  
    Print Half-CM Grid [108](#)  
    Print MM Grid [109](#)  
    Print No Grid [109](#)

Set Auto Scale [108](#)  
Set X Axis Range [108](#)  
Show CM Grid [108](#)  
Show Half-CM Grid [108](#)  
Show Info [110](#)  
Show MM Grid [108](#)  
Show No Grid [108](#)  
Show Page Grid [108](#)

## W

Window Menu [163](#)  
wvsetup [176](#)

