AxoGraph X User Manual

Note: For Acrobat Reader Users...

• For the best figure quality when reading this document onscreen, the zoom setting should be 147 %.

• If the zoom setting has changed, type 147 % into the zoom field in the toolbar at bottom left.

• Search for a key-word by clicking on the binoculars icon in the toolbar at top right.

• Navigate this document using the bookmarks at left. The bookmarks can be toggled in and out by clicking the toolbar icon at bottom left.

• Alternatively navigate using the table of contents on following pages. Clicking on a chapter name will jump to that chapter in the main text.

Chapters

Contents

1 Introduction and Overview

1.1 Time Saving Information

A few minutes going over Chapters 2 and 3 is recommended. These sections introduce fundamental concepts and operating principles. Chapter 8 describes how to obtain the best graphical output quality.
• Navigate this document using bookmarks (see Adobe Acrobat Reader documentation).
• To search for a key-word, click on the binoculars icon in the toolbar (or select Find under the Edit menu).

1.2 About This Manual

A convention for specifying menu selections is employed throughout the manual.
File ➔ Open… indicates that the Open... item under the File menu is to be selected.

Worked examples in this manual rely on files provided in the Example Data folder.

1.3 Supported File Formats

A variety of import and export options are provided so that AxoGraph can be used in co–operation with other graphics and analysis programs. Supported file formats include :

Import and Export:
Igor Pro 3.0 binary wave, KaleidaGraph 3.0, CricketGraph III, and tab-delimited text
Import Only:
Clampex 6, Fetchex 6, pClamp 7 & 8, AxoScope, AxoTape 2, AxoData 1.2, Chart 3.6, Setup2, Neurocom, C Lab, space- or comma-delimited text
Export Only:
MacDraw PICT (can be read by ClarisDraw, Canvas, PowerPoint, etc.)
Copy to the clipboard in PICT format (can be pasted into Canvas, ClarisDraw, etc.)

Any single-run binary digitized data file recorded on any computer (IBM, Mac, etc.) can be imported into AxoGraph (acquisition parameters are entered via a dialog).

More complex data files in any format can be imported via a program written in AxoGraph's development environment. Modules are included for reading Igor binary data files (‘experiments’), CED, Neurocom and C Lab data acquisition files.

AxoBASIC users can create Clampex and Fetchex-style data files (ABF format) using routines supplied with AxoBASIC, and these files can be read directly into AxoGraph.

AxoGraph's own file format has been kept very simple. The format is described in the ‘Technical Information’ section. Programmers can use public domain routines for reading and writing AxoGraph files which are provided in the folder, AxoGraph Data File Format (in the Documentation folder).

1.4 Axon's Data Acquisition Software for MS-Windows and MS-DOS

AxoGraph can read data files generated by all of Axon's data acquisition software for PC and Windows.

AxoTape continuous acquisition for PC
AxoBASIC continuous and episodic acquisition for PC
pClamp 6 - CLAMPEX episodic acquisition for PC
pClamp 6 - FETCHEX continuous acquisition for PC
AxoScope continuous acquisition for PC and Windows
pClamp 7 continuous and episodic acquisition for Windows (ABF file format)
pClamp 8 continuous and episodic acquisition for Windows

With the exception of AxoBASIC, all these programs record data in the ABF file format. The MS-Windows programs use a *.ABF filename extension, and the MS-DOS programs use a *.DAT filename extension.

1.5 System Requirements

AxoGraph will run on any Macintosh equipped with at least 8 MByte of RAM and a 68020, 030, 040, or PowerPC CPU. It will run under System 7, 8 and 9. At least 32 MBytes of RAM is recommended for optimal performance with System 8 or 9. At least 64 MBytes of RAM is recommended when using the data acquisition package (see Chapter 3.1 for more information). AxoGraph is compatible with virtual memory, but virtual memory is not recommended when using the data acquisition package.

1.6 Bug Reports

Axon Instruments provides technical support for AxoGraph via e-mail or Fax. Whenever possible, send a file that demonstrates the problem as an e-mail attachment. It is very helpful if you can supply a step-by-step description of the conditions that demonstrate the problem. It is often difficult for us to duplicate your problem without an example file. Send bug reports to:

AxoGraph.Support@webone.com.au	(e-mail)
+1 (510) 675-6200	(Fax)

Visit the AxoGraph web page for upgrade information by clicking on the following link…

http://www.axon.com/CN_AxoGraph4.html

2 Terminology

2.1 Digitized, Graph and Text Files

AxoGraph can open three types of files :

Digitized Files ...	binary digitized integer data.
Graph Files	binary floating-point data.
Text Files	words and numbers.

A graph or digitized file opens into a graph window.
A text file opens into a text editor window or a graph window.

AxoGraph attempts to blur the differences between digitized files and graph files, but some minor distinctions are retained. Raw digitized data is valuable, and should not be altered by an analysis program. For this reason digitized files are copied when they are opened, and all processing is applied to the copied data. To save the processed data, it must be exported to a graph file.

2.2 Continuous and Episodic Data

Digitized data files are of two types :

Continuous ...	one long data trace recorded without interruption.
Episodic	several short data traces synchronized to a stimulus.

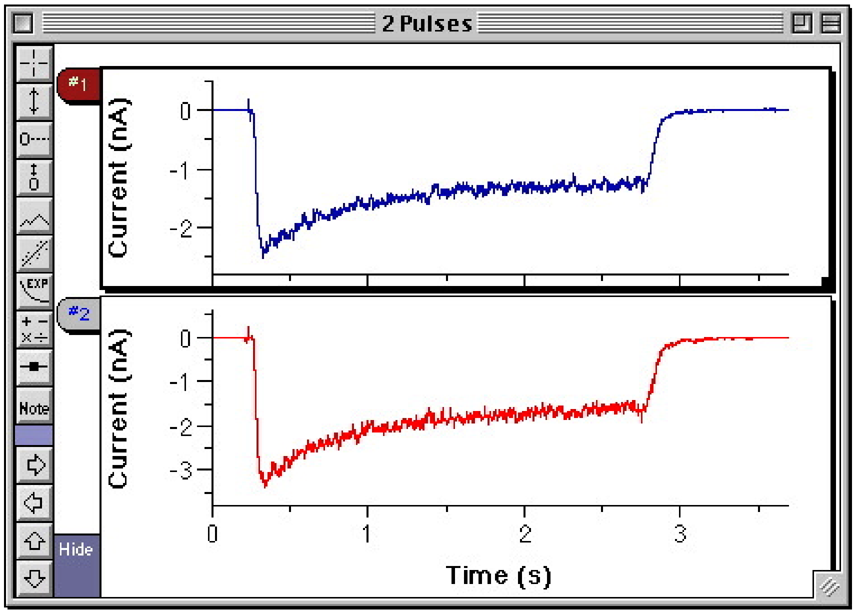
Continuous acquisition is used to record spontaneous or unsynchronized events. Episodic acquisition is used to record evoked or synchronized events. Some display and analysis features (Review, Average) only work with episodic data.

AxoScope, AxoTape and Fetchex record continuous data.
Clampex 6 and AxoData record episodic data.
The AxoGraph data acquisition package and Clampex 7 and 8 record both continuous and episodic data.

A graph file can be continuous or episodic. If two or more traces are combined in a group (see following section) then the file is treated as episodic.

2.3 Traces and Groups

For a graph file, a **Trace** is a single array or column of data.
For a digitized file, a **Trace** is a single episode recorded on one A-D channel.



Traces can be organized into numbered **Groups**. When more than one group is present in a graph window, each is displayed in an outlined sub-window with a numbered 'tag' in the top left corner.

A group can be moved by dragging its tag. It can be resized by dragging the handle (small black square) in the bottom right hand corner.

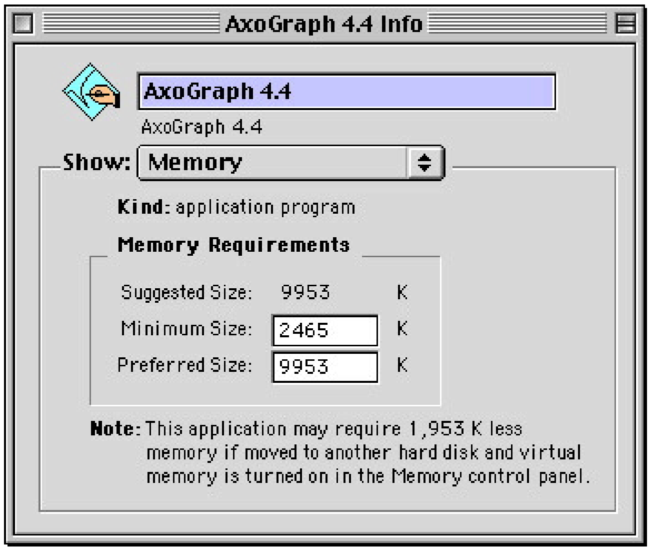
A group can be hidden by dragging its tag into the gray 'Hide' region at the bottom left of the graph window. **Analysis functions are not applied to hidden groups.**

Groups have a front to back order which affects both display and analysis. The front group is outlined in black, and the tag of the front group is dark red (group #1 above). All other tags are gray. **Some analysis functions only apply to the front group**, although most functions permit optional processing of all groups. To bring a group to the front, click once on any part of its sub-window. To send the front group to the back, click on its dark red tag.

For a graph file, traces can be grouped in any combination (see chapter 7).
For an episodic digitized file, all traces (episodes) corresponding to a given A-D channel are automatically combined into a group, and this grouping cannot be altered.

3 Operating Principles

3.1 AxoGraph Memory Management



Kind: application program

Memory Requirements

Suggested Size:	9953	K
Minimum Size:	2465	K
Preferred Size:	9953	K

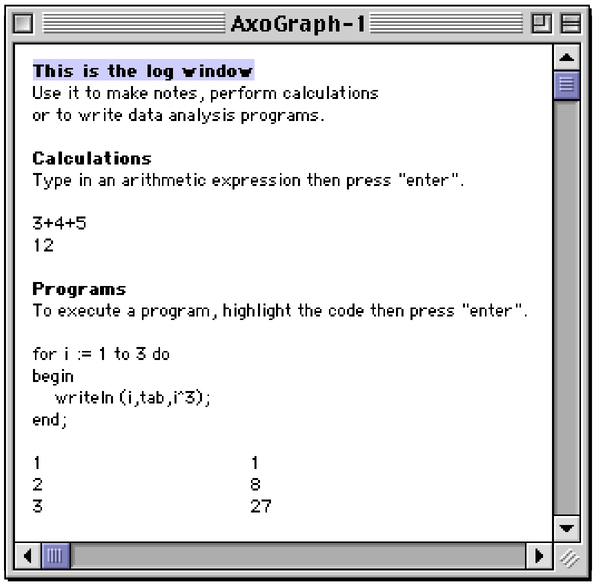
Note: This application may require 1,953 K less memory if moved to another hard disk and virtual memory is turned on in the Memory control panel.

To review or edit AxoGraph's memory partition, highlight the AxoGraph application icon in the Finder, then select File → Get Info → Memory

AxoGraph uses its ‘internal’ 9.9 MByte memory partition for graph style and layout information, and ‘external’ system memory for graph data.

When a graph file is opened, all data is loaded into system memory and style information is loaded into AxoGraph's memory partition. The default memory partition is 9.9 MBytes (reduces to 8 MBytes if virtual memory is turned on). This is sufficient to open and manipulated files with a total of 3,500 traces. To handle additional traces, increase the size of AxoGraph's memory partition by 2 MByte per 1,000 traces (increase Preferred size in the above dialog). AxoGraph must be re-launched for the new memory partition size to take effect.

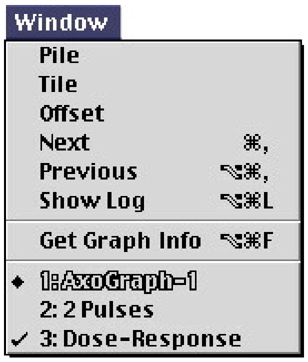
3.2 The Log Window



```
This is the log window
Use it to make notes, perform calculations
or to write data analysis programs.

Calculations
Type in an arithmetic expression then press "enter".

3+4+5
12

Programs
To execute a program, highlight the code then press "enter".

for i := 1 to 3 do
begin
   writeln (i,tab,i^3);
end;

1                    1
2                    8
3                    27
```

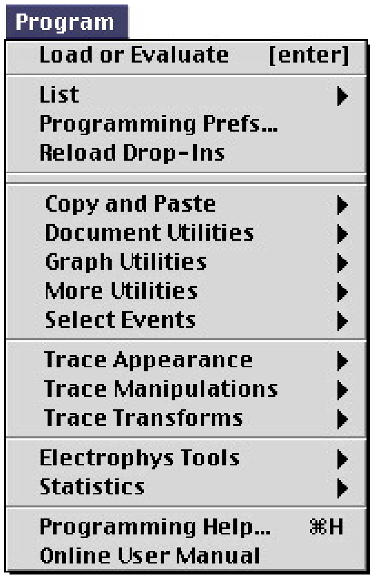
When AxoGraph is started up, a text editor window is opened automatically. This ‘log’ window will receive analysis results from graph windows. It can also be used to enter notes, edit data, perform calculations and write or execute programs. The log window is indicated in the Window menu with a black diamond beside its name (see figure on next page).

3.3 The Window Menu



The names of all open windows are listed at the end of the Window menu. Select a window's name to bring that window to the front. The front window has a check mark next to it. The windows are numbered in the order they were opened. These numbers are important when running AxoGraph custom analysis programs. Text editor window names are displayed in outline style. The log window is indicated with a black diamond.

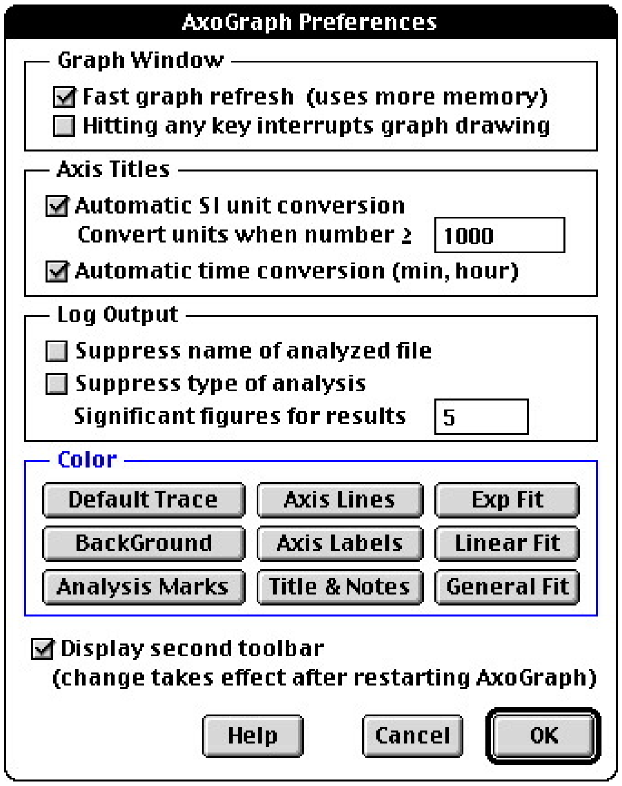
3.4 The Program Menu



The names of all available analysis programs are listed under the Program menu or in sub-menus of the Program menu. Selecting a program's name executes that program. To reconfigure this menu, quit AxoGraph, move text files or sub-folders that contain program source code into or out of the Plug-In Programs folder, then re-run AxoGraph. See Chapter 14 for a complete description of the Program menu.

Select Program ➔ Programming Help to access online documentation describing AxoGraph's multi-language programming environment.

3.5 AxoGraph Preferences



AxoGraph Preferences

Graph Window

☑ Fast graph refresh (uses more memory)
☐ Hitting any key interrupts graph drawing

Axis Titles

☑ Automatic SI unit conversion
Convert units when number ≥ 1000
☑ Automatic time conversion (min, hour)

Log Output

☐ Suppress name of analyzed file
☐ Suppress type of analysis
Significant figures for results 5

☑ Display second toolbar
(change takes effect after restarting AxoGraph)

Edit ➔ Preferences… A dialog appears which controls global aspects of AxoGraph's operation including...

- Redrawing graph windows
- Automatic unit conversion in axis titles
- Numerical precision of results sent to log window
- Default colors in graph windows
- Whether to display a second toolbar

When the Fast graph refresh check box is switched on, AxoGraph takes a snapshot after it has finished drawing a graph, and uses the snapshot to refresh the graph when parts of it are covered then uncovered again. Using the snapshot is much faster than redrawing the graph, but takes up additional memory. To gain extra free memory at the cost of slower performance, turn this check box off before opening any graph windows.

The Hitting any key interrupts graph drawing check box is self–explanatory. This option is only useful when dealing with very large data files that take a long time to draw. A draw operation that is interrupted can be restarted by typing the '*clear*' key or selecting Edit ➔ Clear.

AxoGraph automatically handles Standard International (SI) units if they are present in the axis title. For example, if the initial units are pA, and the axis range is -1500 pA to 2000 pA, then the displayed axis range will automatically switch to -1.5 nA to 2.0 nA. The axis range threshold above which the automatic conversion takes place can be set in the Convert units when number ≥ text edit box. Turn off the Automatic SI unit conversion check box to disable the unit conversion feature. In addition, AxoGraph automatically converts a time axis from seconds to minutes or hours if a very long axis range is specified. Turn off the Automatic time conversion check box to disable this feature.

The two check boxes in the Log Output box determine what information will be written to the log window during data analysis. When the Suppress name of analyzed file and Suppress type of analysis check boxes are on, analysis results will be less cluttered with information. This may help to construct a compact table of results. (See also section 10.4, 'Tidy Selected Text'). The Significant figures for results field determines how many significant figures will be used when sending numerical results to the log window. A setting of 5 is recommended.

The buttons in the Color box set the default colors for displaying graphs and fitted curves. Default Trace and Background set the graph trace color and background color. Default Trace and Background set the graph trace color and background color. Axis Lines, Axis Labels and Title & Notes set the colors of the corresponding elements of a graph. Analysis Marks sets the color of measurement feedback marks (baseline and measurement cursors, etc.).

By default, two toolbars are displayed at the bottom of the screen. The second toolbar can be removed to reclaim some screen space by turning off the Display second toolbar check box.

4 Open, Import, Save and Export Files

4.1 Open a File

There are four items under the File menu that can be used to open files.

Open…	can be used to open graph, digitized, text or foreign files.
Open Digitized…	will only open digitized data files.
Open Text…	will open text files into a text editor window.
Import…	will open and import foreign data files (Igor Pro, etc.).

The File Open dialog remembers the format of the last file opened via a given item. This helps when files of several different format (graph, digitized, text and foreign) need to be opened consecutively in a single session.

For information about the plug-in programs that import foreign data files, read About Foreign File Import in the Foreign File Import / Export folder which is in the Extras folder. It is possible to import any foreign data file format via an import program.

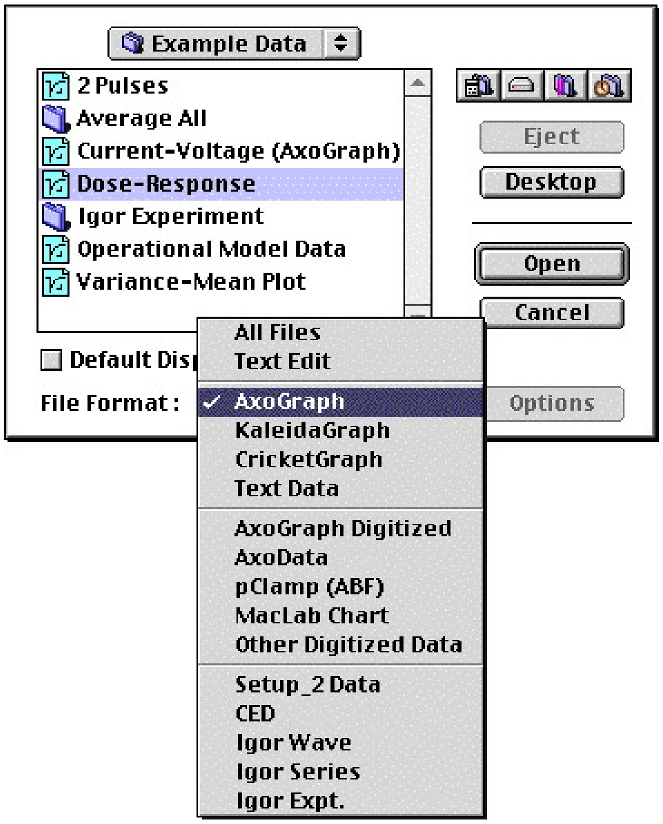
4.2 Create a New Text Editor Window

File ➔ New Text creates a new text editor window that can be used to enter numeric data, perform calculations and write or execute programs.

4.3 Open a Text File for Editing

File ➔ Open Text… then find the Example Data folder and open the text file Current-Voltage (Tab Text). The file opens into a text editor window and contains two tab-delimited columns of numeric data.

4.4 Open an AxoGraph Data File



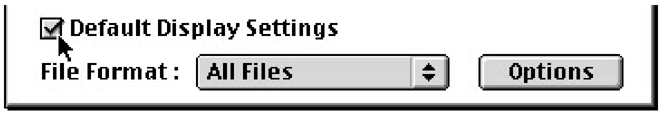
File → Open… then change the File Format popup setting to AxoGraph.

Use the standard file dialog to find the Example Data folder, then select and open an AxoGraph data file (2 Pulses for example).

If another file is opened, then the File Format will default to its most recent setting (AxoGraph in this case).

4.5 Default Display Settings in the Open File Dialog

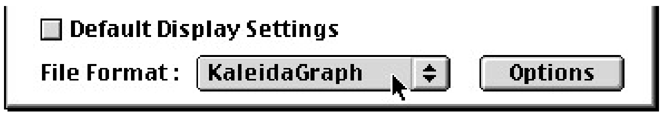
The current display format settings (symbols and colors, trace grouping, axis ranges, etc.) are saved when a graph is closed, and restored when it is re-opened.



To override the stored settings, turn on the Default Display Settings check box in the File Open dialog. When a graph is opened, the default display settings will be used instead of the display settings saved with the file. The defaults are the last settings that were manually selected in an open graph window.

When a file is opened, the default display settings will be used instead of the display settings saved with the file. The defaults are the last settings that were manually selected in an open graph window.

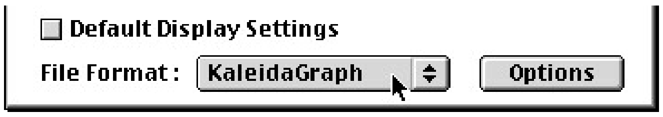
4.6 Import a KaleidaGraph or CricketGraph Data File



File → Open… then change the File Format setting to KaleidaGraph or CricketGraph. Find the Example Data folder, then select and open a data file (Current Voltage (KaleidaGraph) for example).

4.7 X-Axis Data in Imported Files

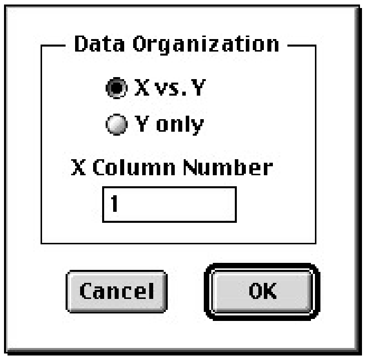
AxoGraph data files are always organized with the x-axis data in the first column. Imported data files may not share this organization. Some have no x-axis data, and others may place it in a different column.



☐ Default Display Settings

File Format: KaleidaGraph

File → Open… then change the File Format setting to KaleidaGraph or CricketGraph. Click on the Options button in the File Open dialog.



Data Organization

(●) X vs. Y

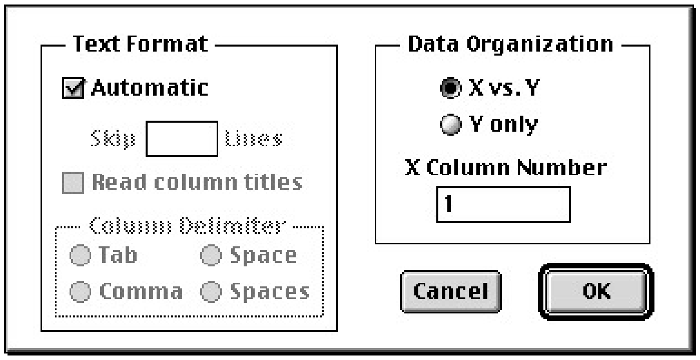
( ) Y only

X Column Number: 1

The text import dialog appears. Enter the number of the x-axis data column in the imported file. If there is no x-data, click on the Y only radio button, then specify the sample interval (data points are assumed to be uniformly spaced).

4.8 Open a Text Data File

AxoGraph can import text data files containing lists of numbers. The numerical data columns can be separated by tabs or commas. There can be several lines of description at the start of the file, and the data columns may or may not have text titles. AxoGraph will try to guess the layout of the text data. If this does not work, the layout can be specified in detail as follows.



Text Format

☑ Automatic

Skip [ ] Lines

☐ Read column titles

Column Delimiter: ( ) Tab ( ) Space ( ) Comma ( ) Spaces

Data Organization

(●) X vs. Y

( ) Y only

X Column Number: 1

File → Open… then change the File Format setting to Text Data (see section 4.4). Click on the Options button in the File Open dialog, and the text import dialog appears.

If the Automatic check box is on, AxoGraph scans the text file and attempts to guess the text data format (number of columns, column delimiter, etc.). This will only work for Tab or Comma delimited text data. To manually specify the text data format, turn off the Automatic check box. Turn on the Read column titles check box if column titles are present. Specify how many lines to skip at the start of the file (there is often several lines of header information to skip). Select the column delimiter (Tab, Comma, single Space or multiple Spaces).

To import a text data file in a non-standard format (e.g. some other delimiter), the text will need to be pre-processed using a text editor.

4.9 Append a Graph File to an Open Graph Window

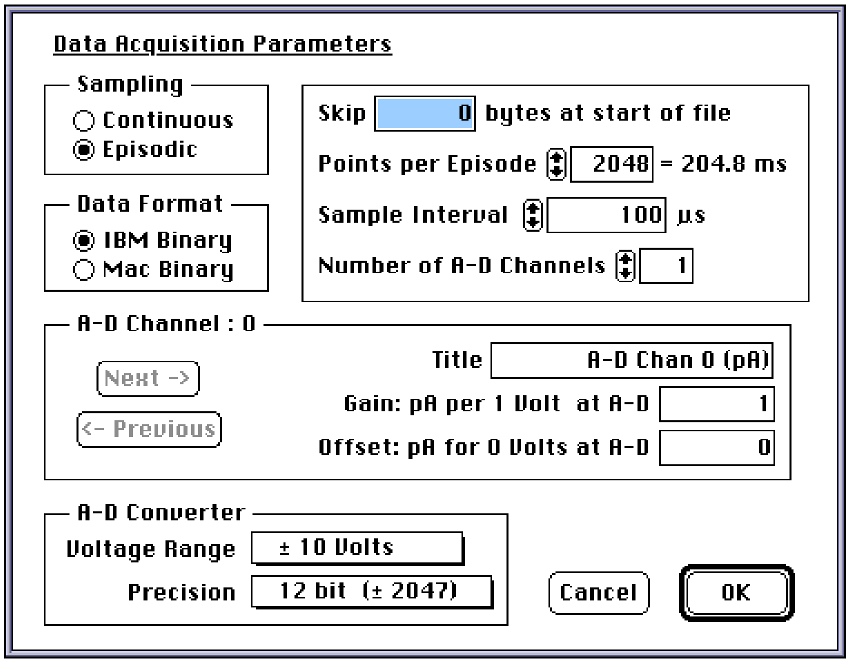
Open a graph file (2 Pulses for example). File ➔ Append… then select another graph file (or select the same file again for this demonstration). New traces are added in the bottom half of the graph window, but are partially overlaid. The x-axis data from the appended file is also appended to the combined graph.
The x-axis data columns can be viewed or edited by selecting Display ➔ X vs Y and Error Bars...
After appending a graph, select Trace ➔ Tile to tidy up the new traces or Trace ➔ Combine to combine old and new traces into a single group.

4.10 Open an AxoData or pClamp Digitized Data File

File ➔ Open Digitized… then change the File Format setting to AxoData or pClamp. Open a digitized data file (Current.DAT for example). The graph window is titled Current.DAT Copy because AxoGraph is working with a copy of the original data. Any changes made to the data cannot be saved back to the original file. This protects the raw digitized data. Use Save a Copy… to save changes to digitized data files.

4.11 Open a Foreign Digitized File

File ➔ Open Digitized… then change the File Format setting to Digitized Data and turn on the Default Display Settings check box in the File Open dialog. Select and open the file ForeignDigitized.DAT. This single-run foreign data file was digitized on an IBM PC and written to a disk file with no header information. When a digitized data file which lacks a pClamp or AxoData header is opened, AxoGraph presents the user with a dialog where the acquisition parameters (sampling rate, gains, etc.) can be specified.



Data Acquisition Parameters

Sampling: Continuous [✓ Episodic]

Data Format: [✓ IBM Binary] Mac Binary

Skip 0 bytes at start of file

Points per Episode 2048 = 204.8 ms

Sample Interval 100 µs

Number of A-D Channels 1

A-D Channel : 0

Title A-D Chan 0 (pA)

Gain: pA per 1 Volt at A-D 1

Offset: pA for 0 Volts at A-D 0

A-D Converter

Voltage Range ± 10 Volts

Precision 12 bit (± 2047)

Set Skip bytes at start of file to 0, since this file has no header. If a foreign file had a 512 byte header, then this parameter would be set to 512.

Set Data Format to IBM Binary. This determines the byte order of the digitized data. PCs based on Motorola chips (DEC, VAX, Atari) generally use the same byte order as the Macintosh, while machines based on Intel CPU chips (IBM and clones) use the reverse.

Set the Voltage Range to ±10 Volts, and the Precision to 16 bits. The voltage range is the minimum to maximum voltage at the input of the A-D converter, and the precision is the maximum numerical range of the digitized values. It is not strictly the precision. Some 12-bit A-D converters left shift the converted values for compatibility reasons (the 4 low order bits are set to zero). In this situation Precision should be set to 16 bit.

Set the remaining parameters as shown above. The acquisition parameters only need to be set once. The next time the file is opened, they will be applied automatically. Also, they become the default parameters, aiding import of multiple files with similar acquisition parameters. The Acquisition Parameters dialog can be accessed after a file is opened by selecting Edit → Acquisition Params…

4.12 Save a File

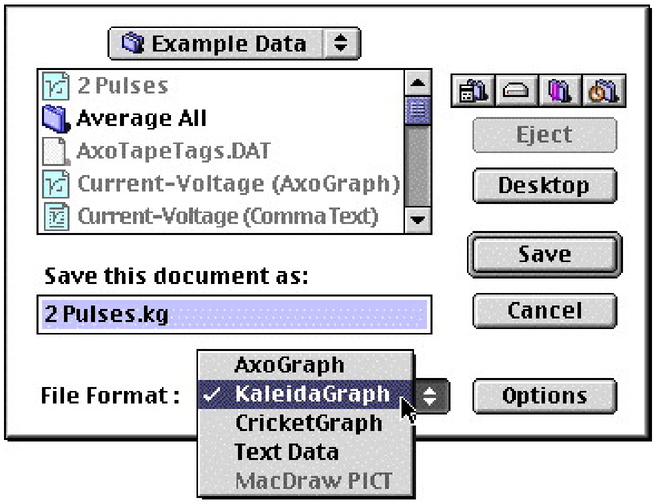
There are three items under the File menu that can be used to save the front graph or text window to a disk file:

Save…	saves to an existing file
Save as…	saves all graph data a new file, then switches to that file
Save a Copy…	saves only the displayed graph data to a new file

For graph windows, the Save as… and Save a Copy… items let the user specify the format of the new data file (AxoGraph, KaleidaGraph, etc.).

The Save… and Save as… items do not work for digitized data files. This protects these valuable files from being overwritten. Use Save a Copy… to save changes to digitized data files.

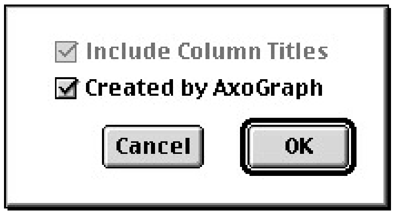
4.13 Export a Graph Window to a KaleidaGraph File



Save this document as: 2 Pulses.kg

File Format: ✓ KaleidaGraph

Open the AxoGraph file 2 Pulses. File → Save as… then change the File Format setting to KaleidaGraph as shown. Also, change the file name so that the original file is not overwritten.

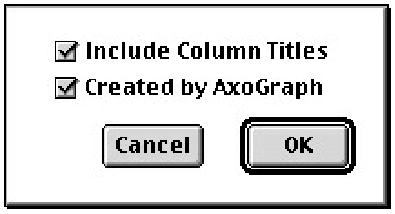


☐ Include Column Titles
☑ Created by AxoGraph

Now, click on the Options button. If the Created by AxoGraph check box is turned on, the new file's creator flag will be set to AxoGraph, otherwise it will be set to KaleidaGraph. This determines which program is run when the file's icon is double clicked.

4.14 Export a Graph Window to a Text Data File

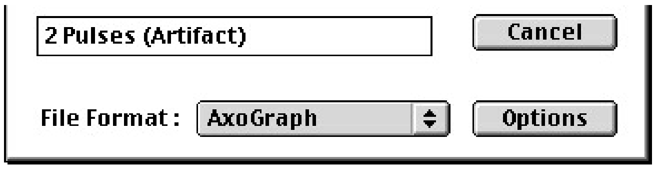
Graphs can be exported to tab-delimited text files. This is a standard data file format that can be read by most graphics and spreadsheet programs. Open the file 2 Pulses in the Example Data folder.
File → Save as… then change the File Format setting to Text Data and change the file name to 2 Pulses (Text). Now, click on the Options button.


☑ Include Column Titles
☑ Created by AxoGraph

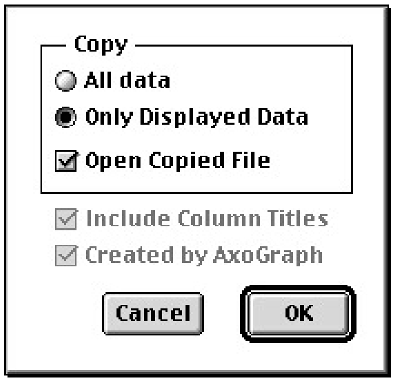
By default, the first row of the new file contains the column titles followed by the numerical data in subsequent rows. If the Include Column Titles check box is turned off, the column titles row is not written.

4.15 Export Only Part of a Graph Window

A selected section of a graph can be saved to a new file. Open the file '2 Pulses' in the Example Data folder. Drag trace #2 to the Hide area, then zoom in on the stimulus artifact of trace #1 (see section 5.2).


2 Pulses (Artifact)
File Format : AxoGraph

File → Save a Copy… then change the File Format setting to AxoGraph and change the file name to '2 Pulses (Artifact)'. Now, click on the Options button.


Copy
○ All data
● Only Displayed Data
☑ Open Copied File
☑ Include Column Titles
☑ Created by AxoGraph

If the All Data radio button is on, the copied file will contain all the data from the original file.

If the Only Displayed Data radio button is on, the copied file will only contain the currently visible data (the artifact from trace #1 in this example).

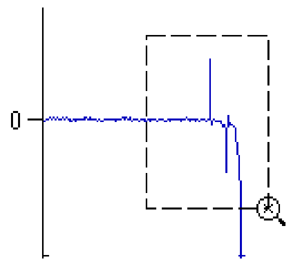
If the Open Copied File check box is turned on, the copied file will automatically be opened after it is created.

5 Browse Graph Data

5.1 Open a Graph Window

All sections in this chapter assume that a graph window is open.
File ➔ Open… then select a graph file (2 Pulses in the Example Data folder for example).

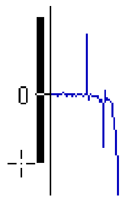
5.2 Zoom and Unzoom



Move the mouse cursor into the middle of a graph. It changes to a magnifying glass. Hold the mouse button down and drag a zoom box around a feature of interest.

The selected feature is enlarged and the x- and y-axes change accordingly.

With the mouse cursor still in the middle of a graph, double click the mouse button to zoom back out. Each double click zooms further out until the default axis range is reached.

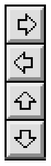


Move the mouse cursor over one of the axes. It changes to a cross hair. Hold the mouse button down and drag a zoom bar besides or below a feature of interest.

Only one axis is zoomed.

Display ➔ Default Axis Ranges sets the x- and y-axis ranges back to their default values.

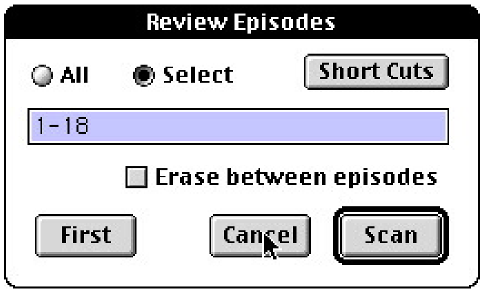
5.3 Scroll



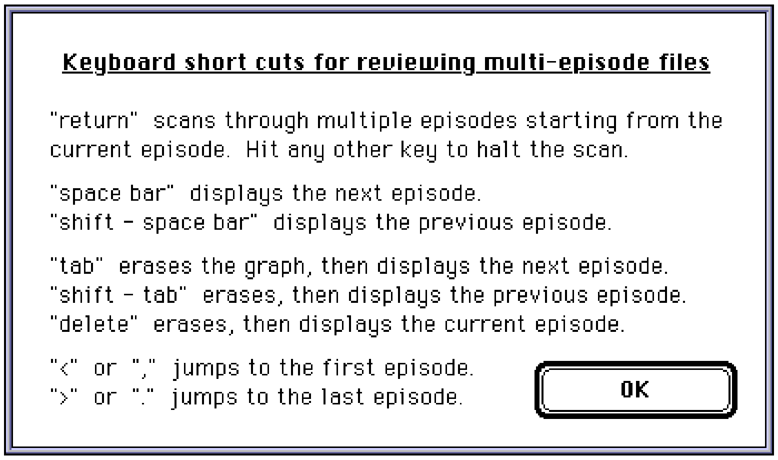
Zoom in on a feature of interest in a graph window, then use the keyboard arrow keys to scroll the graph. Alternatively, use the mouse to activate the scroll buttons in the bottom-left hand corner of the graph window. These provide smoother, faster scrolling than the keyboard.

5.4 Review Multi-Episode Data

File → Open Digitized… then select and open the file VoltageClamp.DAT in the Example Data folder. This is a Clampex file containing 10 episodes sampled on two A-D channels.



Display → Review Episodes… By default, all episodes are reviewed. To review only selected episodes, enter the episode numbers separated by commas or spaces. A range of episodes may be reviewed using the first and last episode number separated by a dash.



Keyboard short cuts for reviewing multi-episode files

"return" scans through multiple episodes starting from the current episode. Hit any other key to halt the scan.

"space bar" displays the next episode.
"shift - space bar" displays the previous episode.

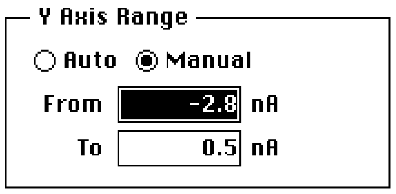
"tab" erases the graph, then displays the next episode.
"shift - tab" erases, then displays the previous episode.
"delete" erases, then displays the current episode.

"<" or "," jumps to the first episode.
">" or "." jumps to the last episode.

Click the Short Cuts button, and a dialog appears listing keyboard techniques for browsing multi-episode files.

6 Display Options

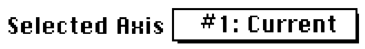
6.1 Axis Range and Title



Y Axis Range
Auto [✓ Manual]
From -2.8 nA
To 0.5 nA

All sections in this chapter assume that a graph window is open. File → Open… then select a graph file (2 Pulses in the Example Data folder for example).

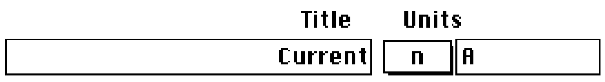
Double click on the x- or y-axis in a graph window. A dialog appears that controls the axis range and style. Display → Change Axis Range… opens the same dialog.



Selected Axis #1: Current

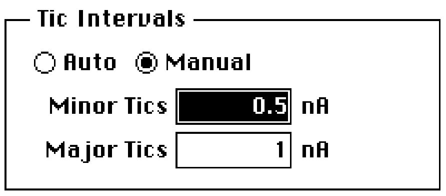
All the x- and y-axes can be modified without leaving this dialog by using the Selected Axis popup menu to move between axes.

To select the default axis range, turn on the Auto radio button in the Axis Range box. To manually set the range, enter values in the From and To text edit boxes.



Title: Current
Units: n A

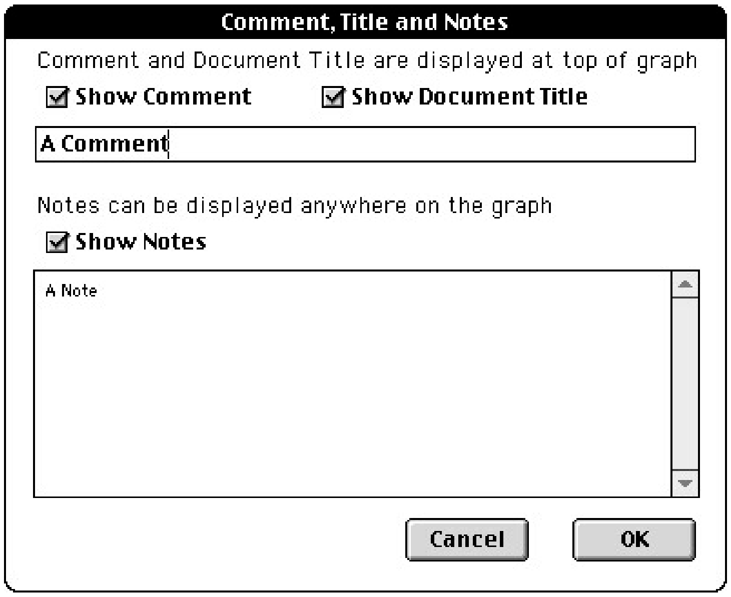
Enter the axis title and units in the Title and Units text edit boxes.



Tic Intervals
Auto [✓ Manual]
Minor Tics 0.5 nA
Major Tics 1 nA

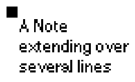
By default, major and minor tick intervals are selected automatically. To set these manually, click on the Manual radio button in the Tick Intervals box and enter values in the Major Ticks and Minor Ticks text edit boxes.

6.2 Comment, Title and Notes Options


Comment, Title and Notes
Comment and Document Title are displayed at top of graph
☑ Show Comment
☑ Show Document Title
A Comment
Notes can be displayed anywhere on the graph
☑ Show Notes
A Note
Cancel
OK

Display ➔ Comment and Notes… To add a short, one line comment to the top of a graph, turn on the Show Comment check box and enter the comment in the text edit box immediately below. To display the name of the graph file, turn on the Show Document Title check box. The comment and title are added at the top of the graph window, and cannot be moved.

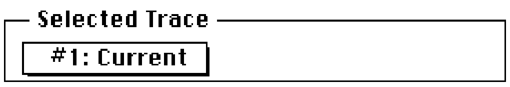
To add multi-line text to a graph, turn on the Show Notes check box. Notes can be typed or pasted into the text edit area below this check box.



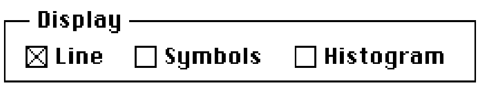
Notes are displayed in the graph window using a small font size. A small black square at the top left of the notes can be used to drag them anywhere in the graph window.

6.3 Symbol and Line Display Options

Display ➔ Symbols and Lines… The resulting dialog can change the symbols and/or lines used to display each trace.

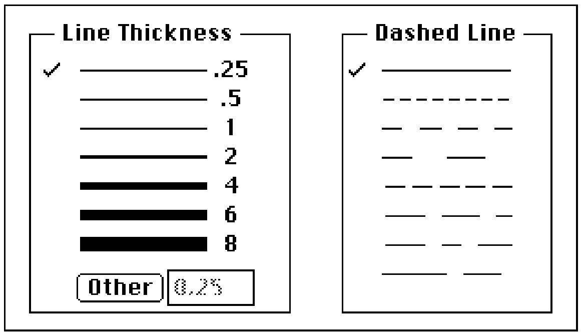


The symbol and line options for all traces can be edited without leaving the dialog by using the Selected Trace popup menu to move between traces.



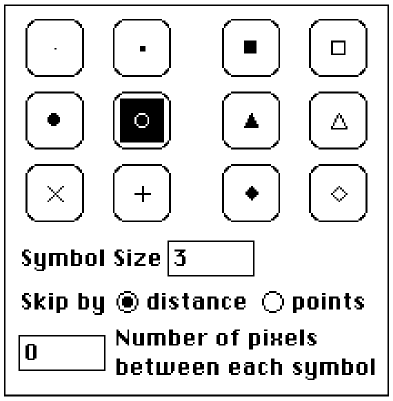
Each trace can be displayed as a line, a set of symbols (circles, squares, triangles, etc.), as a histogram, or as any combination.

The color of each trace can be specified by clicking on the Color button.



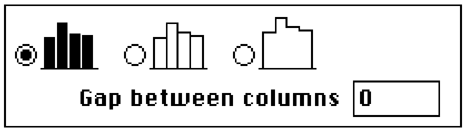
Line thickness and dash style can be controlled by clicking on the desired setting, or typing into the Other text edit box. Fractional thickness settings cannot be accurately represented on the screen, but will affect the appearance of the line when printed.

The symbol used for each trace is selected from a palette. The symbol size is entered as the radius of the symbol in pixels. For example, the default symbol size is 3 which generates a symbol 6 or 7 pixels in diameter.


Symbol Size 3
Skip by ◉ distance ○ points
0 Number of pixels between each symbol

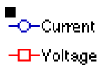
By default, a symbol is displayed for every data point. To improve the graph's appearance, some symbols can be skipped. When the Skip by distance radio button is selected, a symbol is displayed only when a data point is more than N pixels away from the previously displayed data point. The value of N is entered in the box labeled Number of pixels between each symbol. This style will display symbols where they are needed most, on steeply sloping regions of a signal. However, it can produce undesirable results when the signal is noisy, preferentially displaying the points with the greatest deviation from the underlying signal.

When the Skip by points radio button is selected, a gap of N data points is left between each displayed symbol. The value of N is entered in the box labeled Number of points between each symbol. This style displays data points in a more regular manner, but may leave steeply sloping regions of a signal looking bare.



Histograms are displayed as filled or outlined columns, or as a ‘cityscape’. A gap can be left between neighboring columns. The width of the gap in pixels is entered in the Gap between columns text edit box.

6.4 Symbol Legend

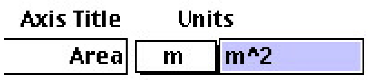


Display → Show Symbol Legend displays a movable symbol legend. It lists the y-axis titles for all displayed traces, and indicates the symbol used to display each trace. Drag the small black square to move the legend.

6.5 Automatic Axis Units

AxoGraph automatically handles Standard International (SI) units if they are present in the axis title. For example, if the initial units are pA, and the axis range is adjusted to -1500 pA to 2000 pA, then the displayed axis range will automatically switch to -1.5 nA to 2.0 nA.

This feature can be switched off. Edit → Preferences… then turn off the Automatic SI unit conversion check box (see section 3.5).



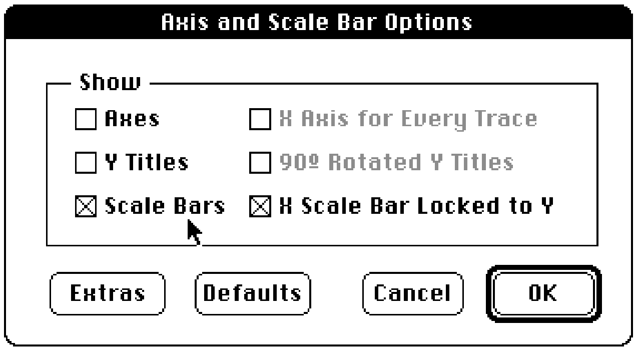
To raise an SI unit to a power, use the caret symbol (^). For example, millimeters squared would be indicated as mm^2 in the Units field of the axis title, as shown here.

6.6 Font, Size and Style

The font, size and style of all text in a graph window can be changed using the Font, Size and Style sub-menus under the Text menu. To modify the font, size and style of individual text elements, copy and paste the graph to a drawing or page layout program. See section 8.4 for more information about copy and paste.

6.7 Axis and Scale Bar Display Options

All sections in this chapter assume that a graph window is open.
File ➔ Open… then select a graph file (2 Pulses in the Example Data folder for example).


Axis and Scale Bar Options

Show
- ☐ Axes
- ☐ X Axis for Every Trace
- ☐ Y Titles
- ☐ 90º Rotated Y Titles
- ☒ Scale Bars
- ☒ X Scale Bar Locked to Y

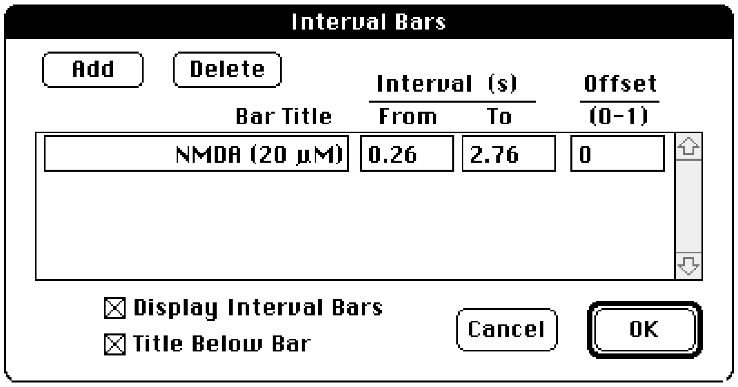
Display ➔ Scale Bars and Axes… To display labeled axes, turn on the Axes check box. To display y-axis titles, turn on the Y Titles check box. By default, the y-axis title is displayed horizontally at the top of the y-axis. To rotate the title to the vertical and display it beside the y-axis, turn on the 90º Rotated Y Titles check box. To display scale bars, turn on the Scale Bars check box. By default, the x- and y-scale bars are locked together so that when one is dragged, both move. To move the x- and y-scale bars independently, turn off the X Scale Bar Locked to Y check box.

6.8 Axis Tick Length and Line Thickness Options

Display ➔ Scale Bars and Axes… then click the Extras button. The dialog expands and options for controlling the axis tick length and axis and scale bar line thickness are presented. Ticks may be located on the inside or outside of the axis line. Selecting both inside and outside causes the tick to stroke through the axis line. Line thickness is specified in pixels.

6.9 Interval Bars

Interval bars are similar to scale bars, except they are tied to a specific region of the x-axis. They are useful for indicating a period of drug application or other experimental condition.


Interval Bars

Bar Title	Interval (s) From	Interval (s) To	Offset (0-1)
NMDA (20 µM)	0.26	2.76	0

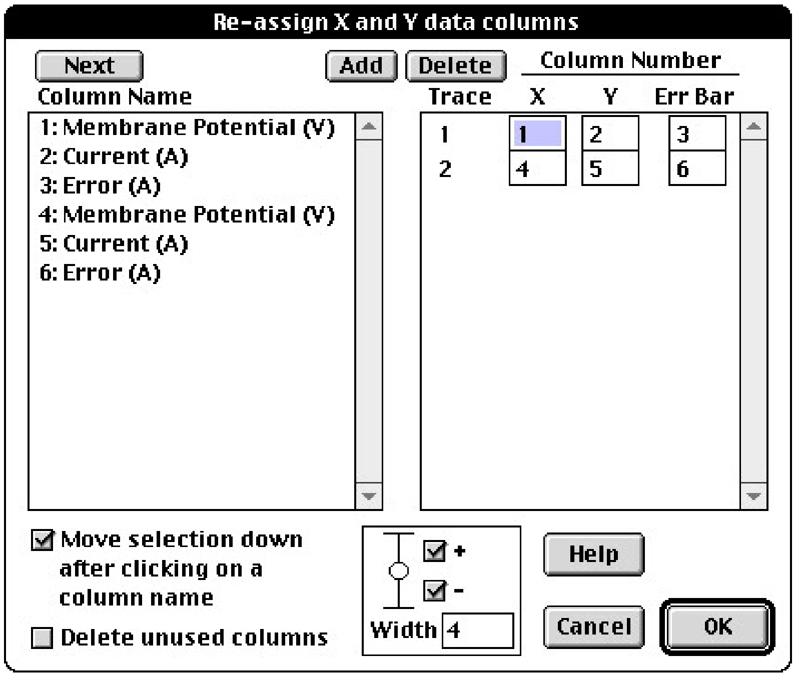
- ☒ Display Interval Bars
- ☒ Title Below Bar

Display ➔ Interval Bars… To add an interval bar, click on the Add button, then enter the Bar Title and specify the interval (From -> To) in x-axis units.

Any number of bars can be added to a graph. By default bars are drawn at the bottom of the lowest trace or group. They can be positioned higher by setting Offset to a value in the range 0 to 1. If Offset is set to 1, the bar is drawn at the top of the lowest trace or group. By default the interval bar title is drawn above the bar. Turn on the Title Below Bar check box to display the title underneath the bar. Interval bar thickness is the same as scale bar thickness (see section 6.8).

To hide all interval bars, turn off the Display Interval Bars check box. To delete an interval bar, highlight its Bar Title and click the Delete button.

6.10 Set the X- and Y-Axis and Error Bar Data Columns



Column Name
1: Membrane Potential (V)
2: Current (A)
3: Error (A)
4: Membrane Potential (V)
5: Current (A)
6: Error (A)

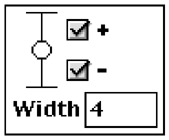
Trace	X	Y	Err Bar
1	1	2	3
2	4	5	6

☑ Move selection down after clicking on a column name
☐ Delete unused columns
☑ +
☑ -
Width 4

Display → Error Bars… A dialog appears for specifying which data columns contain the X, Y and Err Bar data for each trace. In the example shown here, 6 data columns (left hand panel) have been assigned to 2 traces (right hand panel).

The first trace will plot column # 2, Current on the Y axis against column #1, Membrane Potential on the X axis. Each data point will have an error bar attached with length specified in column #3, Error. The Err Bar column numbers are optional, and can be left blank or set to zero.

Display → X vs Y… Brings up the same dialog, but with the error bar settings hidden. This dialog is simpler to work with when error bars are not required.



When error bar columns are specified, the error bar display format can be controlled as follows. Turning on the + check box to displays a bar above each data point, turning on the - check box displays a bar above each data point, and the length in pixels of the cross-bar at the end of each error bar can be entered in the Width field. The specified error bar format will be applied to all traces. To set a different error bar format for each trace, see the following section (6.11).

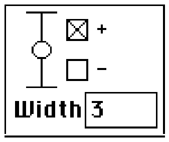
Column Numbers in the right hand panel can be entered automatically by clicking on a Column Name in the left hand panel.

The Move Down check box is on, the text entry selection will move down after clicking on a Column Name otherwise it will move to the right.

Click on the Next button to move down, or to the right without altering the currently selected item.

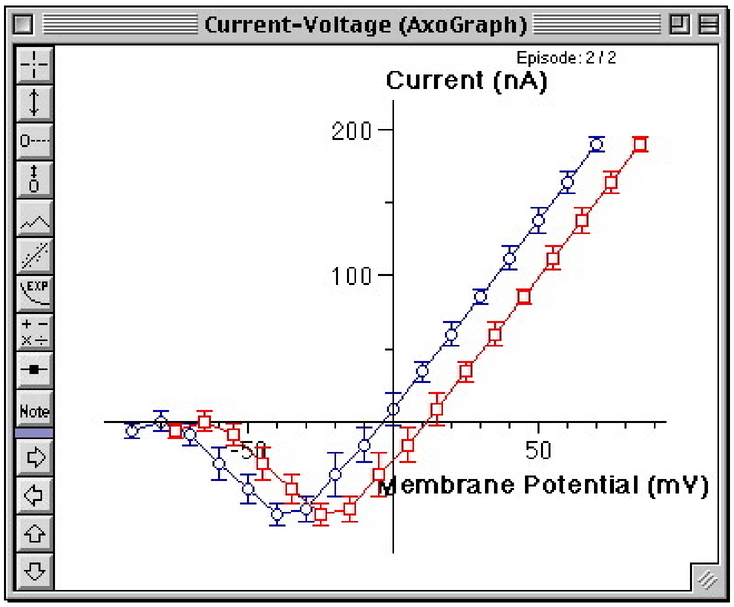
Click on the Delete button to mark the currently selected trace for deletion. A marked trace has X and Y column numbers set to dashes.

6.11 Error Bar Options



Display → Symbols and Lines… The symbols dialog also includes the error bar display options. These will be dimmed unless an error bar column has been assigned to the selected trace (see section 6.10 above). Error bars will be drawn above (+ check box) and/or below (- check box) each data point. The Width of the horizontal line at the end of each bar is given in pixels.

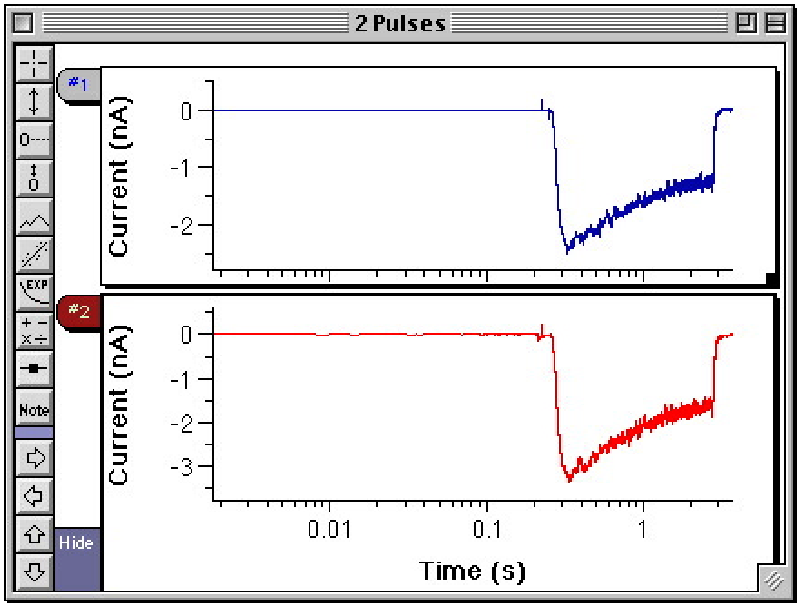
6.12 Zero Crossing Axes



Open the file Current-Voltage (AxoGraph) in the Example Data folder, and turn on the Default Display Settings check box. The graph shows a simulated current-voltage graph.

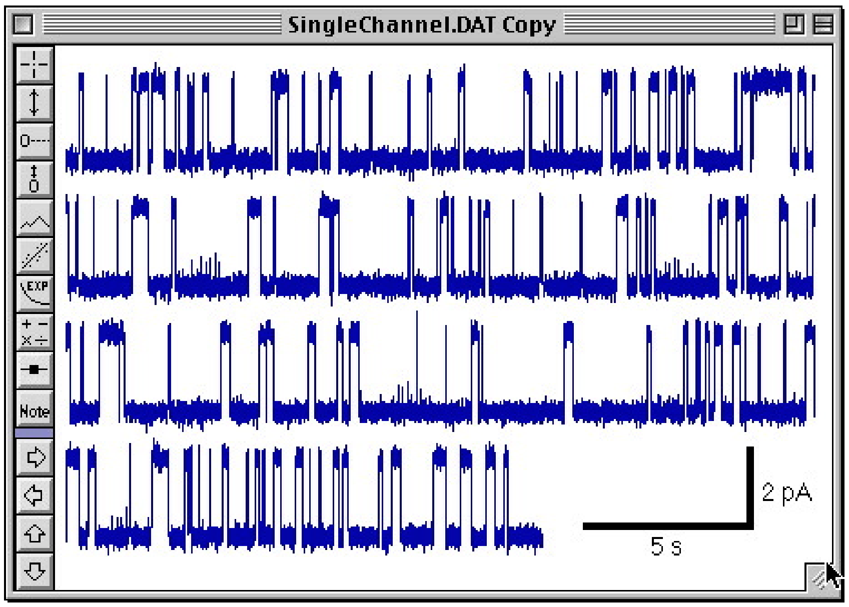
Display → Type of Axes → Zero Crossing This moves the axes so they cross at (0,0). This is the standard style for a current-voltage graph. The axis range should span zero for both x- and y-axes.

6.13 Log Axes

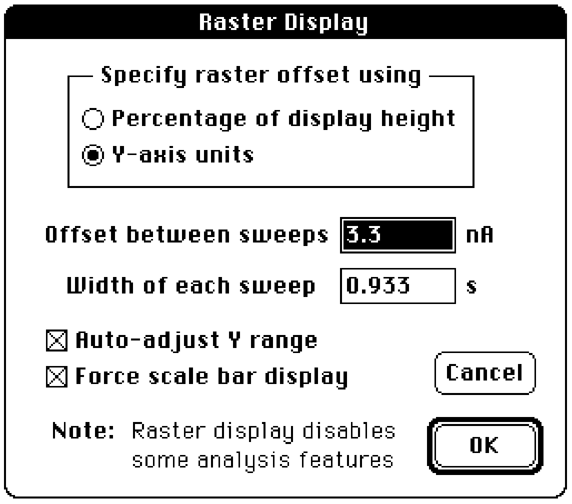


Make the 2 Pulses graph file, or a similar file the front window. Display → Type of Axes → Linear vs. Log changes the x-axis from linear to log. Both x- and y-axes can be switched between linear and log by selecting items in the Type of Axes sub-menu. The Tick Interval settings in the Change Axis Range dialog are not effective when a log axis is displayed.

6.14 Raster Display



Open the file SingleChannel.DAT in the Example Data folder. This graph is displayed in 'raster' mode. The continuous data trace is plotted in 4 separate segments, one below the other. The advantage of this display mode is that it sacrifices redundant y-axis resolution for improved x-axis (temporal) resolution.



Specify raster offset using
- ◯ Percentage of display height
- ◉ Y-axis units

Offset between sweeps 3.3 nA

Width of each sweep 0.933 s

☒ Auto-adjust Y range

☒ Force scale bar display

Note: Raster display disables some analysis features

Display → Type of Axes → Raster Display… Each trace is divided into several sections (sweeps) and the sweeps are displayed vertically offset from one another. The offset is specified in the Offset between sweeps text edit box, and the width of each sweep (and hence the number of sweeps) is specified in the Width of each sweep text edit box.

When the Auto-adjust Y range check box is turned on, the Y display range automatically increases to accommodate the raster sweeps.
When the Force scale bar display check box is turned on, the display parameters are automatically adjusted to hide the axes and show scale bars.

Episodic data is displayed with episodes offset from one another by Offset between sweeps.

Raster display mode was designed to aid figure production. It significantly complicates the issues of mouse interaction and screen feedback. For this reason, some types of analysis cannot work properly when a graph is displayed in raster mode. These analysis features are disabled when in raster display mode. Normal axes are recommended for data analysis.

6.15 Split Clock Support

File → Open Digitized… then select and open the file SplitClock.DAT in the Example Data folder. This is a Clampex file that was acquired at two sample clock rates. The first half of each trace was sampled at 50 kHz and the second half was sampled at 66.6 Hz. When the file is opened, only the first half of the acquisition period is shown, and the first sample rate is used. Display → Multiple Clocks → 2nd Clock switches the display to the second half of the acquisition period and changes to the second sample rate. To view and analyse split clock files with the entire width of the data trace displayed correctly, it is necessary to export the digitized data to a graph file. Display → Save a Copy… then click on the Options button and turn on the All Data radio button and the Open Copied File check box. Click OK then Save. This operation will take a few seconds. Trace → Combine overlays the traces. Zoom on the left end of the x-axis to view the data sampled at the faster rate.

6.16 Display Tags

File → Open Digitized… then select and open the file AxoTapeTags.DAT in the Example Data folder. This is an AxoTape file that includes tag markers. Tags in Chart files can also be displayed.

Display → Show Tags turns on the display of AxoTape tags. To hide the tags, select Display → Hide Tags. If no tags are present in a file, the Show Tags menu item will be dimmed.

7 Traces, Groups and Windows

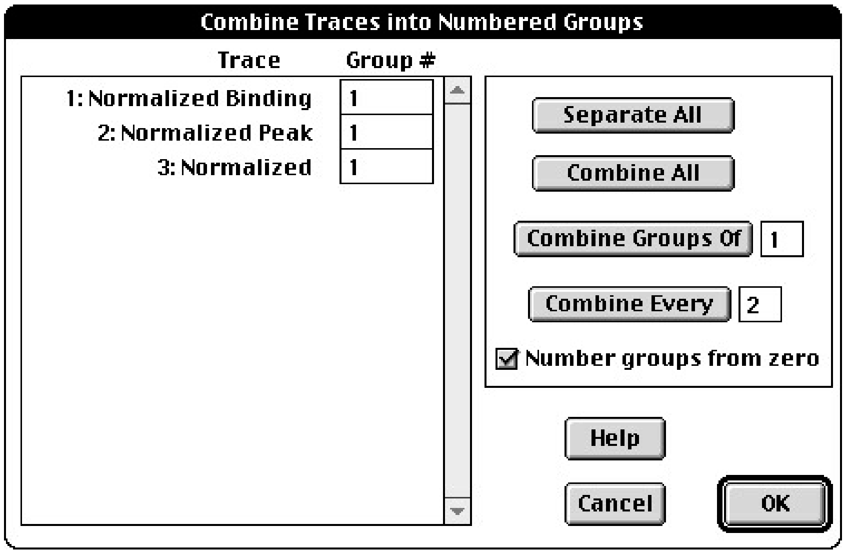
7.1 Separate or Combine All Traces

All sections in this chapter assume that a multiple trace graph window is open and that it is the front window. File → Open… then select a multiple trace graph file (Dose-Response in the Example Data folder for example).

Trace → Separate Split up any groups into individual traces. Each trace is displayed in a separate sub-window.

Trace → Combine All traces in a graph are combined into a single group. The combined traces are overlaid and plotted against the same axes. When several traces are grouped, they are treated as 'episodes' and can be browsed and analyzed using tools described in Chapter 12.

7.2 Combine Traces into Groups



Combine Traces into Numbered Groups

Trace	Group #
1: Normalized Binding	1
2: Normalized Peak	1
3: Normalized	1

Combine Groups Of: 1
Combine Every: 2
☑ Number groups from zero

Trace → Group… This dialog permits arbitrary grouping of traces. When two or more traces are placed in the same group (i.e. given the same group number), they will be overlaid and plotted against the same y-axis.

As an example of arbitrary grouping, assign trace 1: to Group # 1 and traces 2: and 3: to Group # 2, as shown below, then click OK.

Other buttons facilitate grouping when large numbers of traces are present.
Combine All merges all traces into a single group (#1).
Separate All separates groups by assigning each trace to a different group.
Combine Groups Of assigns sequences of N consecutive traces to the same group, where N is entered in the adjacent text edit box.
Combine Every assigns every Nth trace to the same group, where N is entered in the adjacent text edit box.
Number groups from zero begins the automatic group numbering from zero.

7.3 Pile, Tile or Offset Groups

Before proceeding, Trace → Separate to separate all groups of traces. This will aid the following descriptions.

Trace → Pile All active (not hidden) groups are expanded to fill the graph window. Click on the tag of the front group (top left). It is sent to the back and the next group is brought to the front.

Trace → Tile All active groups are rearranged into equal sized, non-overlapping sub-windows.

If the *shift* key is held down when this item is selected, the groups are not repositioned, but are resized into non-overlapping sub-windows. The best way to visualize the two types of Tile is to try them.

Trace → Offset All active groups are rearranged with a small offset between each group. This stacked arrangement is most useful when the y-axis titles are displayed horizontally (see section 6.7), because the titles will then form a list in the graph window.

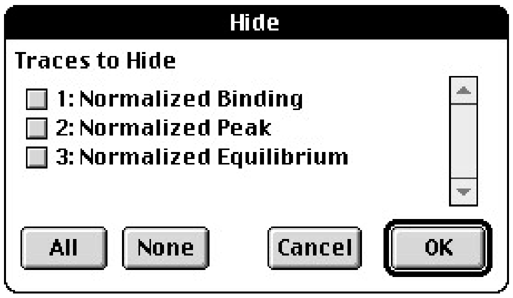
7.4 Next Group

Trace → Next The front group is sent to the back and the next group is brought to the front. This option is only active when there is more than one group in the graph window.

7.5 Show All Traces

Trace → Show All Hidden groups are moved back into the graph window, then all groups are rearranged with a small offset between each group.

7.6 Hide Traces



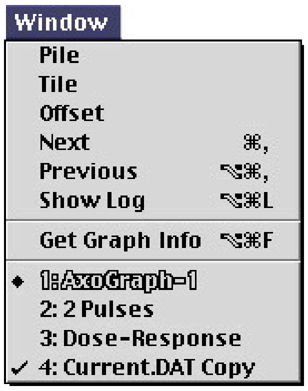
Trace → Hide… This dialog permits one or more trace to be hidden. While a trace is hidden, it will also be excluded from any analysis. For example, turn on the check box corresponding to trace #1 then click OK. Trace #1 is moved to the hide region. This dialog can also be used to move traces out of the hide region by turning their check boxes off.

When several traces are combined in a group, individual traces can also be hidden using the Display → Mask Selected… feature (see section 12.2).

7.7 Delete Traces

Trace → Delete… This dialog permits one or more traces to be permanently deleted from the graph. It is very similar to the 'Hide' dialog above. Click the Cancel button to avoid deleting any traces.

7.8 Manipulating Graph Windows



Open several graph windows. The names of the open windows are listed at the end of the Window menu. This menu can be used to move between the various windows.

The windows can be rearranged on the screen using the first three items at the top of the Window menu.

Window → Pile	Expand and overlay all graph windows.
Window → Tile	Resize and rearrange the graph windows so that all graphs can be seen.
Window → Offset	Resize and rearrange the graph windows so that the title bars of all windows can be seen.

To include text editor windows when using Pile, Tile or Offset, hold down the *shift* key while selecting the menu item.

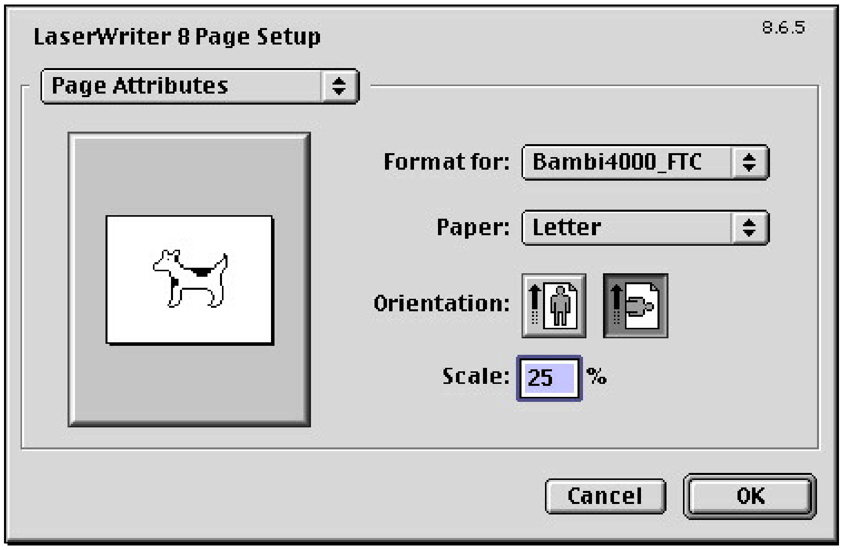
The remaining Window menu items are used to cycle through the open windows, get the log window or list file format information.

Window → Next	Send the front window to the back.
Window → Previous	Bring the back window to the front.
Window → Show Log	Bring the log window to the front.
Window → Get Graph Info	List information about the front graph window into a new text window.

8 Print or Export a Graph

8.1 Use a Page Setup Scaling of 25% for Optimal Output Quality

A graph can be output from AxoGraph in three different ways. It can be printed, exported to a MacDraw (PICT) file, or it can be copied and pasted to any program that supports graphics (all drawing programs and most word processors).



LaserWriter 8 Page Setup 8.6.5

Page Attributes

Format for: Bambi4000_FTC

Paper: Letter

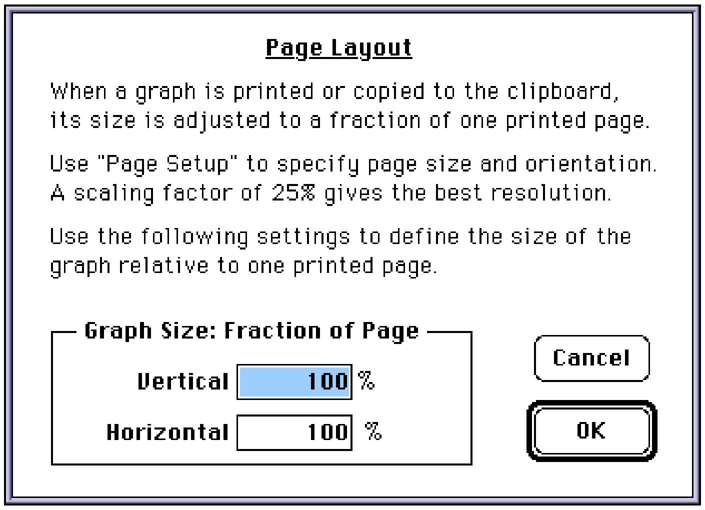
Orientation:

Scale: 25 %

For the best graph output quality, it is important to select File → Page Setup and set the Scale factor to 25% as shown at left.

For a full page graph, the Orientation should generally be set to horizontal, as shown.

8.2 Page Layout : Print a Full Page Graph



Page Layout

When a graph is printed or copied to the clipboard, its size is adjusted to a fraction of one printed page.

Use "Page Setup" to specify page size and orientation. A scaling factor of 25% gives the best resolution.

Use the following settings to define the size of the graph relative to one printed page.

Graph Size: Fraction of Page

Vertical 100 %

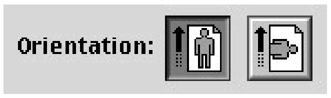
Horizontal 100 %

File → Page Layout… Determines how large the output graph will be in relation to a printed page. This not only applies to printed graphs, but also to graphs that are copied to the clipboard or exported to a PICT format file.

Set the Vertical and Horizontal fractions to 100% to specify a full page graph, then select File → Print…

8.3 Page Layout : Print a Half Page Graph

File → Page Layout… Set Vertical to 50%. The graph will now be output to the top half of a page.



The orientation of the output page should be changed to vertical, otherwise a very short wide graph will be produced. File → Page Setup… and set Orientation to vertical as shown.

File → Print… The graph is printed in the top half of a vertical page.

8.4 Copy and Paste a Graph to a Drawing Program

File → Page Setup… Set Reduce or Enlarge to 25%, and Orientation to horizontal (see section 8.1).

File → Page Layout… For the highest resolution and output to a full page, set both Vertical and Horizontal to 100%.

Edit → Copy Copies the graph to the clipboard.

Switch to a drawing program and paste the graph into a new document. In the drawing program, again change the Page Setup settings to Reduce or Enlarge 25%, and set Orientation to horizontal (see section 8.1). The pasted graph will occupy the top half of one page. It can be edited and then printed at full resolution.

Note that when the Page Setup scaling factor is set to 25%, AxoGraph increases the text point size by 400%. If the text was set to 12 point in AxoGraph, it will appear set to 48 point in the drawing program.

8.5 Copy and Paste a Graph to a Word Processor

File → Page Setup… Set Reduce or Enlarge to 100% and Orientation to vertical.

File → Page Layout… To place a graph in the top 1/3rd of a page, set Vertical to 30% and Horizontal to 100%.

Edit → Copy Copies the graph to the clipboard. Switch to a word processor and paste the graph into the document. The resolution of the pasted graph is the same as the screen (~72 dpi). When printed, it will not use the full resolution of the laser printer. This problem is, in part, due to the word processor's limited graphics handling ability.

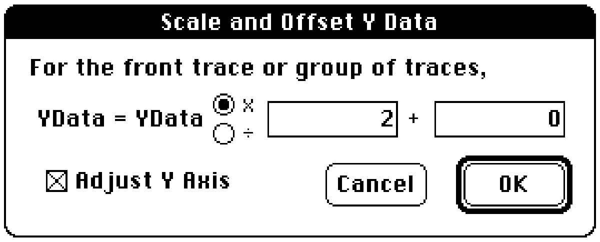
8.6 Export to a PICT File

File → Save a Copy… Change the File Format setting to MacDraw PICT, and change the file name to avoid over-writing the original graph file (for example, append ‘PICT’ to the file name). The exported PICT file can be imported by ClarisDraw, Canvas, PowerPoint or any other drawing or page layout program.

9 Data Analysis (Graph)

9.1 Scale and Offset X- and Y-Data

All sections in this chapter assume that you have a graph window open and that it is the front window. File → Open… then select a graph file (2 Pulses in the Example Data folder for example).


Scale and Offset Y Data

For the front trace or group of traces,

YData = YData [● ×] ○ ÷ 2 + 0

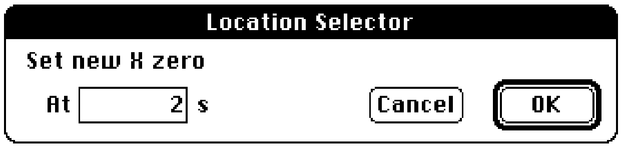
☒ Adjust Y Axis

Analyse → Scale and Offset Y… This opens the Scale and Offset dialog. Turn on the multiplication radio button and enter a scale factor of 2 and an offset of 0. Make sure the Adjust Axis check box is on so that the y-axis range will change to accommodate the scaled data. Watch the y-axis of the front group and click OK.

The scale and offset factors are applied to all traces in the front group.

Analyse → Scale and Offset X… A similar dialog is presented, but the scale and offset factors apply to the x-axis data only.

9.2 Change X-Axis Zero


Location Selector

Set new X zero

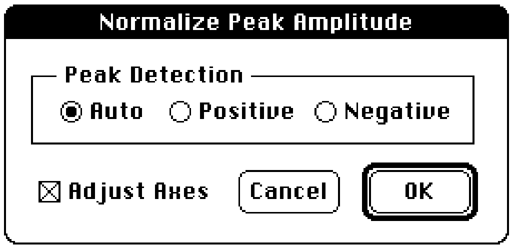
At 2 s

Analyse → Change X-Zero… A location marker (vertical line) is drawn at the x-axis zero point, and a location selection dialog appears in a movable window.

A new x-zero location can be entered by typing a value, or the location marker can be moved using the mouse. Point to a new x-zero location in the graph window and click the mouse button.

Click OK, and the x-axis data is offset so that zero is now at the selected location. The x-axis range is adjusted automatically.

9.3 Normalize Peak Amplitude



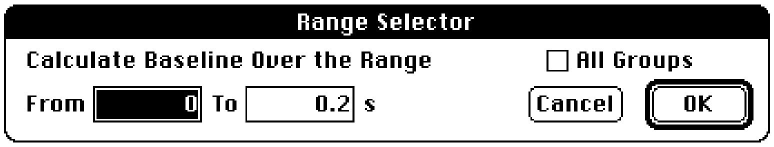
Analyse → Normalize… Each trace in the front group will be scaled so that its peak amplitude is set to 1. This dialog controls whether positive or negative peak amplitudes are normalized.

Select the Positive radio button to detect only positive peaks. Select the Negative radio button to detect only negative peaks. Select the Auto radio button to detect the largest absolute peak.

Peaks are only detected in the visible x-axis range. If an artifact is present that is larger than the signal peak, adjust the x-axis range to exclude the artifact.

By default, the y-axis range is automatically adjusted to display the normalized traces. To disable this adjustment, turn off the Adjust Axes check box.

9.4 Subtract Baseline

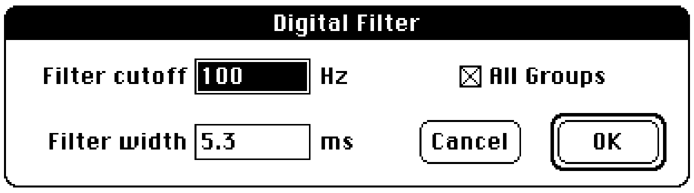


Analyse → Baseline… Two range delimiters (vertical lines) are drawn and a range selection dialog appears. Dashed horizontal lines indicate the baseline offsets (average amplitude of each trace in the selected range). Click OK, and the baseline offset is subtracted from each trace. The y-axis range is adjusted automatically.

The baseline range can be entered by typing minimum and maximum values, or the range delimiters can be positioned using the mouse. Point to one end of the range in the graph window and click the mouse button once. Alternatively, hold the mouse button down and drag the delimiter to the desired position. To switch to the other range delimiter, hit the ‘*tab*’ key, or double click the mouse button.

To restrict baseline subtraction to traces in the front group, turn off the All Groups check box.

9.5 Digital Filter

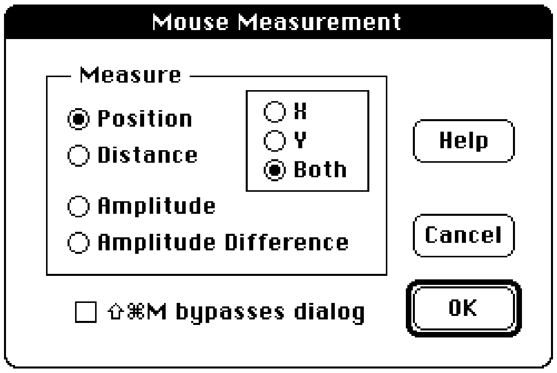


Analyse → Digital Filter… A digital filter will be applied to all visible traces. Filtering is performed by convolving a Gaussian with the data. The Filter width can be specified directly (the width equals 4 standard deviations of the Gaussian envelope), or it can be specified in more familiar terms as the Filter cutoff frequency. The default filter cutoff frequency is 1/5th the sampling frequency.

To limit application of the digital filter to traces in the front group, turn off the All Groups check box.

The filter algorithm is provided in Chapter 19, ‘Technical Information’. Filtering a large data file may take several seconds or tens of seconds. Type the ‘*esc*’ key or the ‘Cmd-period’ keys to interrupt a slow filter operation. Select File ➔ Revert to undo the interrupted filter operation.

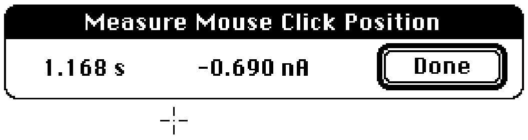
9.6 Mouse Measurements



Analyse ➔ Mouse Measure… The dialog specifies the type of mouse measurement that will follow. Select one of the radio buttons to measure :

By default, both x- and y-components of the mouse measurement are sent to the log window. If only one of these components is required, turn on either the X or Y radio buttons.

Position ...... the position where the mouse is clicked.
Distance ...... the distance between two mouse clicks.
Amplitude ... the trace amplitude at the mouse click position.
Amplitude Difference ... the difference in trace amplitudes between two mouse click positions.



Select the Position radio button, then click OK in the Mouse Measurement dialog.

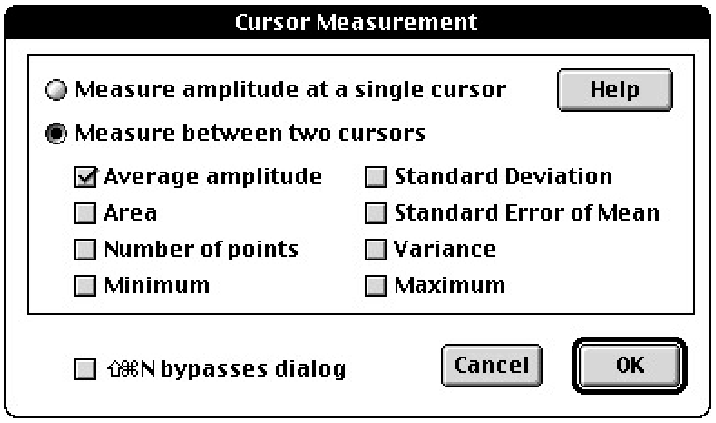
The mouse cursor changes to a cross-hair and the Measure Mouse Click Position dialog appears. This window shows the x- and y-coordinates of the mouse, and these values constantly update as the mouse is moved around in the graph window.

Click the mouse button with the cross-hair cursor positioned anywhere in the graph. The position of the click is marked on the graph, and its x- and y-coordinates are written to the log. More than one position measurement can be made, and the measurements accumulate in the log.

The appearance of the mouse cursor and the behavior of the measurement feedback window will vary slightly depending on the style of mouse measurement selected (Position, Distance, Amplitude or Amplitude Difference).

When a series of files is to be analysed, the repeated presentation of the Mouse Measurement dialog will slow the measurement process. To bypass this dialog, turn on the Shift-Cmd-M bypasses dialog check box and initiate mouse measurements using the Shift-Cmd-M key combination. When dialog bypass is active, the Mouse Measurement dialog can still be accessed by selecting Analyse ➔ Mouse Measure….

9.7 Cursors Measurements



Cursor Measurement

○ Measure amplitude at a single cursor
[✓ ● Measure between two cursors]

☑ Average amplitude	☐ Standard Deviation
☐ Area	☐ Standard Error of Mean
☐ Number of points	☐ Variance
☐ Minimum	☐ Maximum

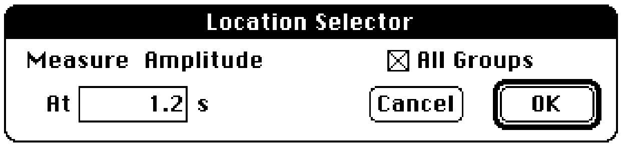
☐ ⇧⌘N bypasses dialog

Help · Cancel · OK

Analyse ➔ Cursor Measure… The dialog specifies the type of cursor measurement to make.

Select the Measure amplitude at a single cursor radio button to measure the amplitude of each trace at a single point only.

Select the Measure between two cursors radio button to perform measurements on each trace over a selected range of the x-axis. All data points located between the two cursors are analyzed. Turn on one or more of the check boxes (Average amplitude, Area, etc.) to indicate which measurements to perform.



Location Selector

Measure Amplitude ☒ All Groups

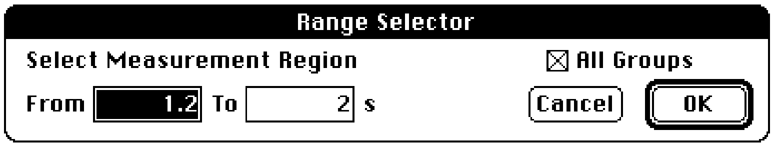
At 1.2 s

Cancel · OK

If the Measure amplitude at a single cursor radio button is selected, a location marker (vertical line) is drawn at the default amplitude measurement point, and the Location Selector dialog is presented.

The measurement location can be entered by typing it in, or it can be selected using the mouse. Point to the desired location in the graph window and click the mouse button once.

By default, all traces in the front group are measured. To measure all visible traces, turn on the All Groups check box. Click OK, and the amplitudes of each trace at the measurement location are displayed on the graph and written to the log (as a short cut, double click the mouse button at the desired location). Click OK again in the Cursor Measurements Complete dialog, and the amplitude measurements are cleared from the graph.



Range Selector

Select Measurement Region ☒ All Groups

From 1.2 To 2 s

Cancel · OK

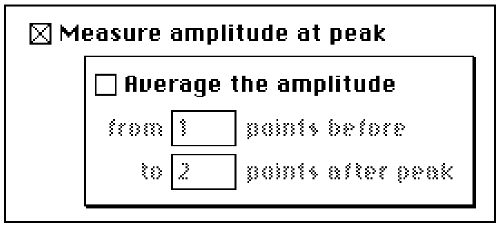
If the Measure between two cursors radio button is selected, two range delimiters (vertical lines) are drawn and the Range Selector dialog is presented.

The measurement range can be entered by typing minimum and maximum values, or the range delimiters can be positioned using the mouse. Point to one end of the range in the graph window and click the mouse button once. To switch to the other range delimiter, hit the '*tab*' key, or double click the mouse button.

By default, all traces in the front group are measured. To measure all visible traces, turn on the All Groups check box. Click OK, and the specified measurements are made for each trace over the selected range, and are displayed on the graph and written to the log. Click OK again in the Cursor Measurements Complete dialog, and the amplitude measurements are cleared from the graph.

When a series of files is to be analysed, the repeated presentation of the Cursor Measurement dialog will slow the measurement process. To bypass this dialog, turn on the Shift-Cmd-N bypasses dialog check box and initiate cursor measurements using the Shift-Cmd-N key combination. When dialog bypass is active, the Cursor Measurement dialog can still be accessed by selecting Analyse ➔ Cursor Measure….

9.8 Peak Amplitude Measurement


☒ Measure amplitude at peak

☐ Average the amplitude

from 1 points before

to 2 points after peak

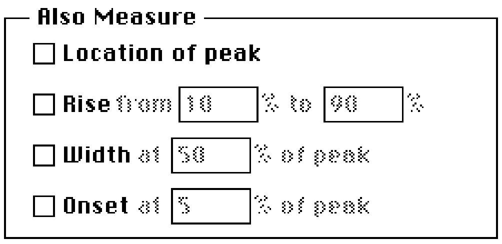
Analyse ➔ Peak Measure… The largest amplitude peak in each trace is detected and the amplitude and shape parameters are measured. The dialog controls which of these parameters are reported. Switch on the Measure amplitude at peak check box to display the amplitude of each peak.

If the signal contains high frequency noise, the peak measurement may overestimate the amplitude of the underlying signal. A more reliable estimate may be obtained by averaging the amplitude around the peak. To measure average peak amplitude, switch on the Average the amplitude check box, and enter the number of data points before and after the peak to include in the average.

Click OK. The peak is automatically detected and marked on the graph. The peak amplitude is displayed on the graph and written to the log.

Peaks are only detected in the visible x-axis range. If an artifact is present that is larger than the signal peak, adjust the x-axis range to exclude the artifact.

9.9 Peak Shape Measurement


Also Measure

☐ Location of peak

☐ Rise from 10 % to 90 %

☐ Width at 50 % of peak

☐ Onset at 5 % of peak

Analyse ➔ Peak Measure… The largest amplitude peak in each trace is detected and the amplitude and shape parameters are measured. The dialog controls which of these parameters are reported. Switch on the Location of peak check box to measure the x-axis location of the detected peak.

The remaining shape measurement options assume that the signal rises to a peak from a baseline with approximately zero amplitude, then falls away again towards zero. If this is not the case, then subtract a baseline from the signal by selection Analyse ➔ Baseline… before using the following options.

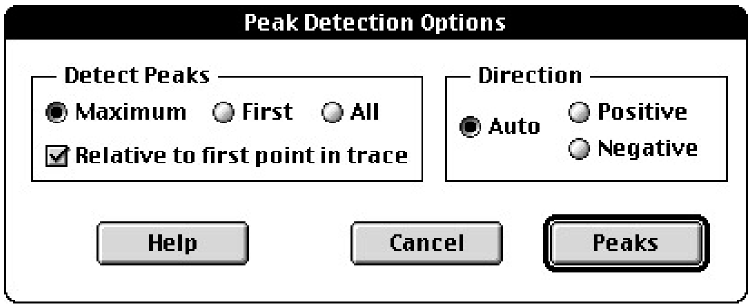
Switch on the Rise check box to measure the rise time of the peak.
The default settings measure the rise time from 10% to 90% of the peak.

Switch on the Width check box to measure the width of the peak.
The default setting measures the width at 50% of the peak.

Switch on the Onset check box to measure the onset time of the peak.
The default setting measures the onset at 5% of the peak.

If the signal is noisy, the performance of the shape measurement options may be improved by adjusting the settings. For example the rise could be measured from 20% to 80%, and the onset could be measured at 20%.

9.10 Peak Detection



Peak Detection Options

Detect Peaks: [✓ Maximum] First All

[✓ Relative to first point in trace]

Direction: [✓ Auto] Positive Negative

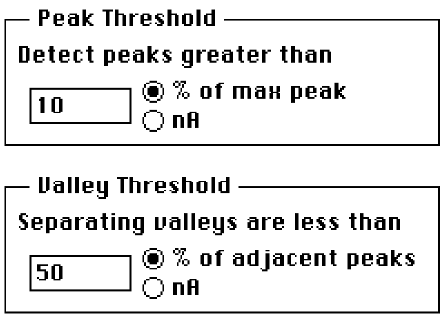
Analyse → Peak Detection… The dialog specifies the basic peak detection options. Peaks are only detected within the visible x-axis range.

Select the Positive radio button to detect only peaks > 0.
Select the Negative radio button to detect only peaks < 0.
Select the Auto radio button to detect peaks in the direction of the largest absolute amplitude.

Select the Maximum radio button to detect the largest absolute peak amplitude in the selected direction.

Select the First radio button to detect the first peak that satisfies the peak detection criteria (see below).
Select the All radio button to detect every peak that satisfies the peak detection criteria (see below).

Turn on the Relative to first point in trace check box to subtract a baseline from the data before peaks are detected. This option may be useful when the signal is offset from zero. It is only provided as a convenience feature, and it will generally make more sense to manually subtract a baseline from the signal (Analyse → Baseline…) before using the peak measurement features. When the 'Relative...' check box is on, the baseline level is averaged over a small region at the left end of the visible x-axis range, then subtracted from the signal. The baseline level is added back to the signal before measuring the peak amplitude or other peak parameters.



Peak Threshold

Detect peaks greater than 10 [✓ % of max peak] nA

Valley Threshold

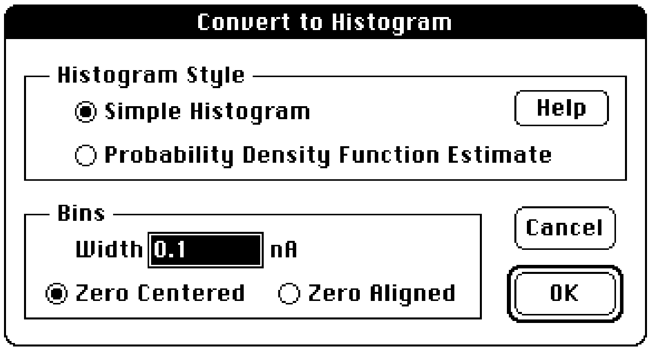
Separating valleys are less than 50 [✓ % of adjacent peaks] nA

When the First or All radio buttons are selected, the dialog expands to offer additional peak detection options. The additional detection criteria help select only the peaks of interest when they are embedded in a noisy signal. To be a significant event, a peak must rise out of the background noise, and it must be separated from an adjacent peak by an intervening valley that is deeper than the peak to peak noise.

Use Peak Threshold to set the minimum amplitude that a peak must reach before it can be detected. The peak thresholds can be specified in y-axis units, or as a percentage of the maximum peak amplitude.

Use Valley Threshold to set the amplitude that the signal must fall below in order to separate two peaks. When the % of adjacent peaks radio button is on, a valley will be detected if it dips below the selected percentage of the smallest of the two adjacent peaks.

9.11 Convert to Histogram


Convert to Histogram

Histogram Style

(●) Simple Histogram

( ) Probability Density Function Estimate

Bins

Width 0.1 nA

(●) Zero Centered ( ) Zero Aligned

Analyse → Convert to Histogram… The data points in each trace are grouped into 'bins' based on their amplitudes. The number of data points in each bin is then plotted against amplitude. The width of the bins can be specified using the Bin Width text edit box.

Select the Zero Centered radio button to construct bins with boundaries that fall on half bin widths. For example, if the bin width is 10, the bin boundaries will be at ... -15, -5, +5, +15, ...

Select the Zero Aligned radio button to construct bins with boundaries that fall on whole bin widths. If the bin width is 10, then bin boundaries will be at ... -10, +0, +10, +20, ...

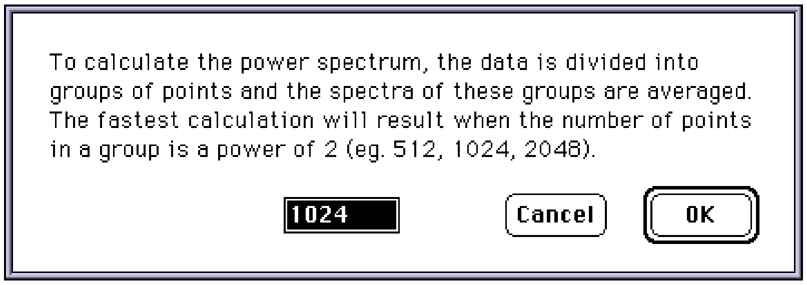
Select the Simple Histogram radio button to construct a conventional amplitude histogram (bin count plotted against amplitude).

Select the Probability Density Function Estimate radio button to construct an amplitude histogram that is normalized to have an area of one. This format has the advantage that its form is independent of the number of data points used to construct the histogram. It is also an estimate of the probability density function underlying the data.

After clicking OK, a range selector dialog appears which permits the analysis range to be restricted.

9.12 Power Spectrum

Analyse → Power Spectrum… The data trace is divided into segments containing the selected number of data points. The power spectrum of each segment is calculated using a fast Fourier transform (FFT)


To calculate the power spectrum, the data is divided into groups of points and the spectra of these groups are averaged. The fastest calculation will result when the number of points in a group is a power of 2 (eg. 512, 1024, 2048).

1024

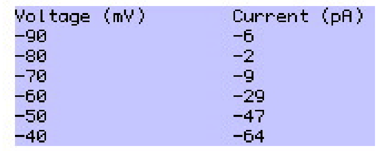
algorithm. The average power spectrum is then calculated and displayed. This processing is mathematically intensive and may take a few seconds to complete. Once started, it cannot be interrupted. In general, a larger group size (e.g. 4096) will result in faster processing and provide more low frequency information, but the spectrum will be noisier because fewer data segments will be averaged.

10 Data Analysis (Text)

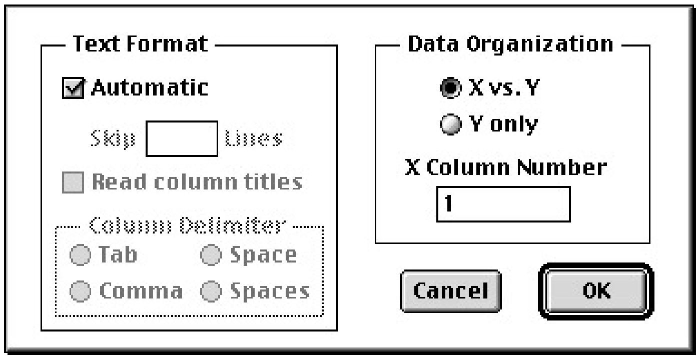
All sections in this chapter assume that you have a text editor window open and that it is the front window. File → Open Text… Select and open a file containing text data (Current-Voltage (Tab Text) in the Example Data folder for example). A text editor window is opened and the text data displayed. The data should be organized in a single column of numbers or in several tab delimited columns of numbers.

10.1 Create a Graph From the Text Editor

Select a list of data values in a text editor window by holding down the mouse button and dragging the cursor until the list is highlighted.



```
Voltage (mV)          Current (pA)
-90                   -6
-80                   -2
-70                   -9
-60                   -29
-50                   -47
-40                   -64
```



Text → Graph Selection… This is the Import Options dialog for text data (also described in section 4.8). If the Automatic check box is on, the selected text data is scanned and AxoGraph attempts to guess the data layout. To manually specify the layout, turn off the Automatic check box. Turn on the Read column titles check box if column titles are present. Specify how many lines to skip at the start of the file (usually 0), and the column delimiter (Tab, Comma or Space). Enter the number of the x-axis data column. If there is no x-data column, click on the Y only radio button, then specify the sample interval (data points are assumed to be uniformly spaced). Click OK.

A new graph window is created and the selected text data is plotted. This technique may be applied to any text data in any text editor window. For example, data analysis results in the log window can be conveniently plotted in this way.

Data to be graphed can also be organized as a series of lists, one below the other. For example....

```
X (s)
0.1584
0.2988
0.4437

Y (nA)
2152
570.4
124.3
```

To plot the list of three Y values in the second list against the list of three x-values in the first list, select both lists then Text → Graph Selection…

10.2 Simple Statistics in the Text Editor

Bring a text editor window to the front and select a list of data values. Do not select the column titles. Select only the numerical values to be analyzed.

Text → Selection Stats Summary statistics are calculated for the data values in each column of selected text. The average value, the standard error of the mean (S.E.M.) and the number of points in each column are output below the previously selected values. For example…

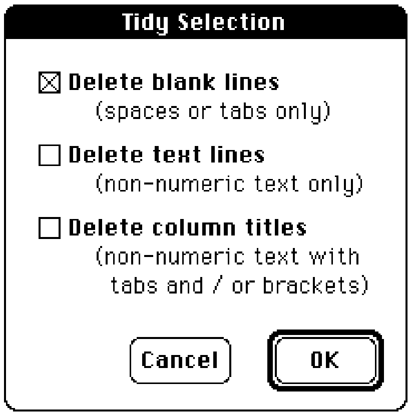
```
Amplitude (mV)          Area (mV.ms)
2.5294                  2.5294
2.615                   2.615
2.22                    2.22
2.5144                  2.5144
2.6094                  2.6094
2.68                    2.68
2.7919                  2. 7919
2.8494                  2.8494
2.2644                  2.2644


2.5638                  2.5638                  Average
0.07103                 0.07103                 S.E.M.
9                       9                       Number Of Points
```

10.3 Tidy Selected Text

The following text is in a format typical of the analysis results that accumulate in AxoGraph's log window.

```
Trial_A :  measure position
X (s)                   Y (nA)
0.1584                  -0.03036
0.2988                  -2.309
2.745                   -1.209

Trial_B :  measure position
X (s)                   Y (nA)
0.01                    2152
1.87                    570.4
```

A common task is to tidy up analysis output of this kind by deleting superfluous information and concatenating the relevant measurements into a single list. The list of measurements can then be plotted and subject to further analysis. The following text editor feature is designed to simplify this task. Bring a text editor window to the front and select (highlight) some analysis results.



Text → Tidy Up Selection… Turn on the Delete blank lines check box to delete all empty lines in the selected text. Lines that contain only space and tab characters are also deleted. This closes up the selection.

Turn on the Delete text lines check box to delete all lines in the selected text that contain only spaces and letters of the alphabet. Lines that contain numerals, tabs, brackets, etc. will not be deleted. This deletes file names and description of analysis generated by AxoGraph analysis routines.

Turn on the Delete column titles check box to delete all lines in the selected text that contain non-numeric characters. Lines with tabs and brackets will be deleted. Lines containing numeric data values will not be deleted. This deletes the column titles generated by AxoGraph analysis routines.

10.4 Convert Vertical Lists into a Tab-Delimited Table

Analysis results often accumulate as a series of lists, one below the other. For example, consider the following 9 lines of text.

```
X (s)
0.1584
0.2988
0.4437

Y (nA)
2152
570.4
124.3
```

To calculate ‘Simple Stats’ for the two lists, the data must first be reorganized as two tab-delimited columns (a two column table). Bring a text editor window to the front and select two or more lists that are organized one below the other (as shown above). The selected text should include the list titles.

Text → Selection to Table and the two vertical lists are reorganized into two tab-delimited columns, side by side.

```
X (s)                    Y (nA)
0.1584                   2152
0.2988                   570.4
0.4437                   124.3
```

This feature also works when there are more than two vertical lists, and when the vertical lists contain more than one column of data. The only requirement is that the vertical lists are separated by one or more blank lines.

10.5 Transpose a Tab-Delimited Table

It is sometimes necessary to transpose a table of results. In the following example, ‘Simple Stats’ has been applied to four data columns, and the results are...

```
40              112             150             190             Average
8.165           21.23           34.66           43.21           S.E.M.
```

To generate a bar graph of the four Average and S.E.M. values, the text table must first be rearranged so that the two rows become two columns (that is, the table must be ‘transposed’). Bring a text editor window to the front and select a tab-delimited table.

Text → Transpose Selection and the table is reorganized as follows.

```
40              8.165
112             21.23
150             34.66
190             43.21
Average         S.E.M.
```

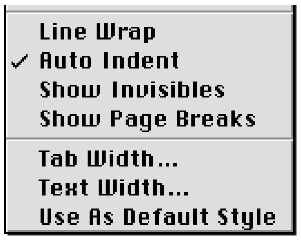
The first row has been converted to the first column and the second row to the second column. The transposed data values can now be selected and graphed (see section 10.2). For information on how to display the S.E.M column as error bars, see section 6.10.

10.6 Load Vertical Lists into Arrays

If one or more tab-delimited columns of numbers are highlighted in the front window, and Text → Selection to Array is selected, each column is loaded into a data array within AxoGraph. By default the arrays are named ‘arr1’, ‘arr2’, ... If the selected columns have titles at the top, the array names are taken from the titles instead. The arrays can then be manipulated and graphed using commands and functions described in Chapters 16, 17 and 18.

10.7 Text Editor Display Options

The appearance of text windows can be altered via the following items under the Text menu.

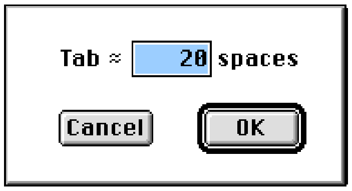


Text → Line Wrap toggles the Line Wrap setting. When Line Wrap is active, text lines longer than the current text editor width will automatically wrap to a new line (standard behavior for a word processor).

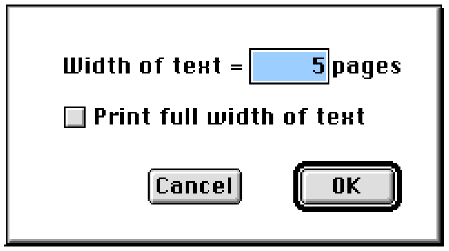
Text → Auto Indent toggles the Auto Indent setting. When Auto Indent is active, and a line starts with one or more spaces and tabs, then typing the ‘*return*’ key automatically inserts the same number of spaces and tabs at the start of the new line. This is particularly useful when writing AxoGraph programs.

Text → Show Invisibles toggles the Show Invisibles setting. When Show Invisibles is active, tab and return characters are displayed using special symbols.

Text → Show Page Breaks toggles the Show Page Breaks setting. When Show Page Breaks is active, page breaks between pages are shown as dashed lines. Calculating and displaying the page breaks slows the performance of the text editor.


Tab ≈ 20 spaces

Text → Tab Width… Specify the separation between the tab markers relative to a space character. A setting of 20 to 30 spaces is usually adequate to separate data columns. Tab markers are equally spaced across the full width of the page. Long column titles may require a larger separation. When writing source code for an AxoGraph program, a setting of 4 spaces is useful for indenting code blocks.


Width of text = 5 pages

☐ Print full width of text

Text → Text Width… The editor can work with text of any width. A text edit region wider than a single page is often convenient. For example, analysis results consisting of 7 or more tab delimited columns will be wider than a single page. Enter the width of the text edit region into the Width of text box. A setting of 5 pages is adequate for most situations. A width setting of 1 page combined with automatic line wrap is useful for simple word processing.

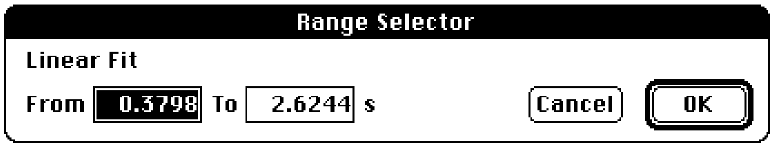
Turn on the Print full width of text check box to print the full width of the text editor region. If this check box is off, only a single page width is printed.

11 Data Analysis (Curve Fit)

All sections in this chapter assume that a graph window open and that it is the front window.
File → Open… then select a graph file (2 Pulses in the Example Data folder for example).

11.1 Fit a Line

Linear regression can be performed on the front trace or group of traces.

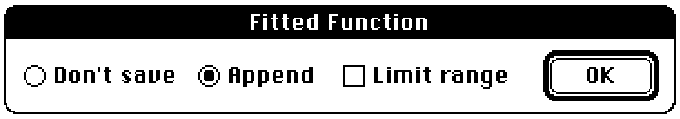


Linear Fit

From 0.3798 To 2.6244 s

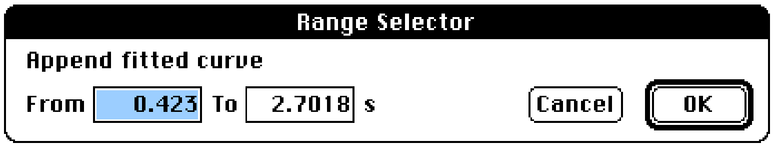
Analyse → Fit Line… Use the Range Selector dialog to select the range of data to fit. The range can be entered by typing minimum and maximum values, or the range delimiters can be positioned using the mouse. Point to one end of the range in the graph window and click the mouse button. This sets the first range delimiter. To set the second range delimiter, hit the '*tab*' key (or double click the mouse button with the mouse still at the first range delimiter) then point the other end of the range and click the mouse button again.

Click OK, and a line is fitted to each trace in the front group. The line is drawn on the screen, together with the equation describing the line.



Don't save [✓ Append] Limit range

Next, the Fitted Function dialog appears. This specifies how to handle the fitted line. By default the line is not saved. Turn on the Don't save radio button and click OK. The fitted line is erased. To incorporate the fitted line into the graph, turn on the Append radio button and click OK. The fitted line is appended to the graph window and added to the front group. The graph, including the fitted line, can now be printed or exported. The appended line can be limited to a selected range. In the Fitted Function dialog, turn on the Limit range check box and click OK. A Range Selector dialog appears.



Append fitted curve

From 0.423 To 2.7018 s

Use this dialog to specify the range limits for the appended line and click OK. The fitted line is appended to the graph window only over the selected range.

The slope, y-axis intercept and correlation coefficient and 'p' value for the fitted line are sent to the log window. The p value is the probability that the data points in the selected range are uncorrelated. If $p < 0.05$, the points are significantly correlated.

11.2 Fit an Exponential

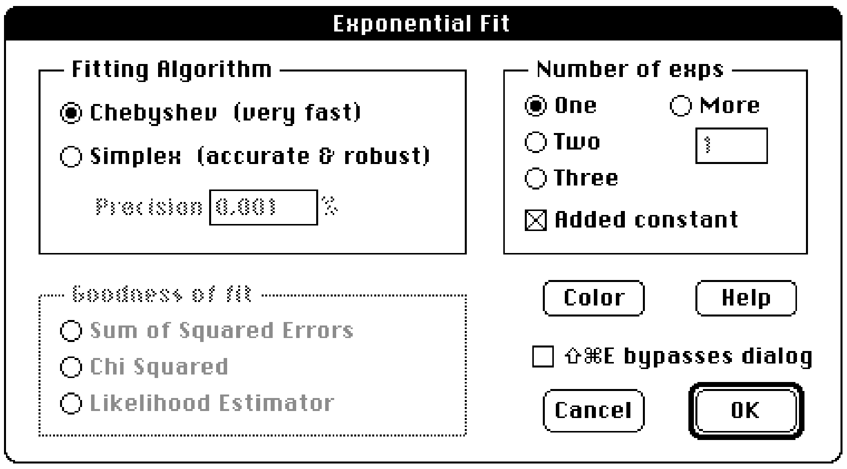
An exponential function can be fitted to the front trace or group of traces. The simplest equation that can be fitted is a single exponential :

$$a \exp(-x / t)$$

Additional exponential components and an added constant can optionally be included. The equation for a triple exponential with added constant is :

$$a1 \exp(-x / t1) + a2 \exp(-x / t2) + a3 \exp(-x / t3) + c$$

Analyse → Fit Exponential… The Exponential Fit dialog specifies the number of exponentials to fit, and the method used to fit them. Turn on the One, Two, Three or More radio button to select a fit with one, two, three or more exponential components. If the More button is used, type in the number of components. Use the smallest number of components that gives an adequate fit to the data. Turn on the Added constant check box to include a constant in the fitted equation when the data does not decay to zero.



Exponential Fit

Fitting Algorithm
- [✓ Chebyshev (very fast)]
- Simplex (accurate & robust)
- Precision 0.001 %

Goodness of fit
- Sum of Squared Errors
- Chi Squared
- Likelihood Estimator

Number of exps
- [✓ One]
- Two
- Three
- More: 1
- [✓ Added constant]

☐ ⇧⌘E bypasses dialog

The exponential equation can be fitted to the data using two approaches. The first is based on Chebyshev polynomials and is extremely fast. The second is based on a simplex optimization procedure and is slower, but more robust. Turn on the Chebyshev or Simplex radio button to select the fit method. See Chapter 19, 'Technical Information' for details about these two methods.

The Simplex fit can be fine tuned by setting the precision of the fit, and by selecting the best goodness of fit measure for the data. Precision should generally be set to less than 5%. The smaller the value, the more accurate the fit, but the longer it will take. A value of 0% will give the best possible fit. There are three goodness of fit measures to choose from. If there is no correlation between noise and signal amplitude, turn on the Sum of Squared Errors radio button. If the noise variance is approximately proportional to the signal amplitude, select the Chi Squared or Likelihood Estimator radio buttons. Likelihood Estimator is slightly superior in theory, but Chi Squared is usually more familiar and results in a statistically meaningful parameter.

Click OK and the Range Selector dialog appears. Use it to select the range of data values to be fitted. See section 11.2 for details on how to select the range.

Click OK and an exponential curve is fitted to each trace in the front group. The fitted exponential(s) are drawn to the screen, together with the decay constants for each exponential component of the fit. The decay constants and amplitudes of each exponential component, the added constant and the goodness of fit (usually SSE) are sent to the log window.

The Fitted Function dialog appears that specifies how to handle the fitted exponential curve. By default, the curve is not saved. To incorporate the fitted curve into the graph, turn on the Append to current file radio button and click OK. The fitted curve is appended to the graph window, and the new trace is added to the front group. The graph and fitted curve can now be printed or exported.

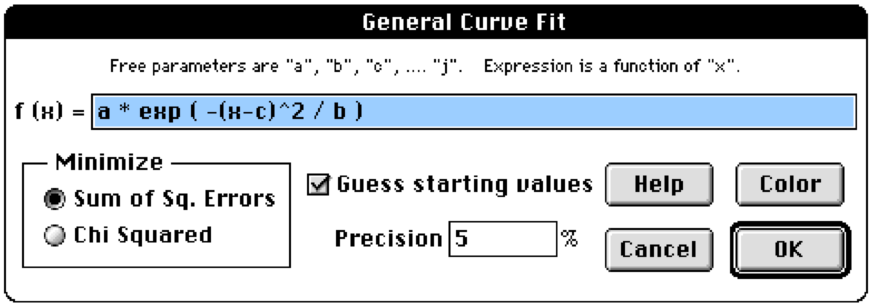
When analysing a series of files, the repeated presentation of the Exponential Fit dialog is unnecessary. To bypass this dialog, turn on the Shift-Cmd-E bypasses dialog check box and initiate exponential fits using the Shift-Cmd-E key combination. The Exponential Fit dialog can still be accessed by selecting Analyse → Fit Exponential….

11.3 Fit a General Equation

A general equation can be fitted to the front trace or group of traces. The equation should be a function of ‘x’, and the free parameters should be selected from ‘a’, ‘b’, ‘c’, ... ‘j’ (i.e. a maximum of 10 free parameters is supported). The equation can include any of the following :

abs (x)	Absolute value
sqrt (x)	Square root
sqr (x)	Square
exp (x)	Exponential (e raised to the power x)
ln (x)	Natural Log(to the base e)
exp10 (x)	10 raised to the power x
log10 (x)	Log to the base 10
trunc (x)	Next lowest integer
round (x)	Nearest integer
pi	Returns the value of pi(≈ 3.1415926)
Sin (x)	Sine
Cos (x)	Cosine
Tan (x)	Tangent
Sinh (x)	Hyperbolic Sine
Cosh (x)	Hyperbolic Cosine
Tanh (x)	Hyperbolic Tangent
ArcSin (x)	Inverse Sine
ArcCos (x)	Inverse Cosine
ArcTan (x)	Inverse Tangent

Analyse ➔ Fit General… Enter the equation to fit in the f (x) = text edit box. The equation shown here 'a * exp(-(x-c)^2 / b)' describes a Gaussian curve.


General Curve Fit

Free parameters are "a", "b", "c", ... "j". Expression is a function of "x".

f (x) = a * exp (-(x-c)^2 / b)

Minimize: [✓ Sum of Sq. Errors] Chi Squared

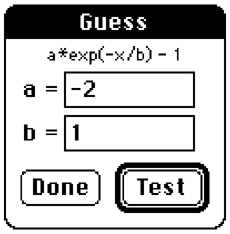
[✓ Guess starting values]

Precision 5 %

The Precision of the fit should generally be set to less than 5%. The smaller the value, the more accurate the fit but the longer it will take. A value of 0% will give the best possible fit.

The simplex algorithm will Minimize either the Sum of Squared Errors or Chi Squared. Sum of squared errors minimization should be used when the noise on the data is independent of the amplitude of the data. Chi squared minimization should be used when the noise on the data is approximately proportional to the amplitude of the data.

The fitting algorithm works best when reasonably good starting guesses are supplied for the parameter values. Turn on the Guess starting values check box and enter starting guesses in the subsequent dialog. Click OK.


Guess

a*exp(-x/b) - 1

a = -2

b = 1

If the Guess starting values check box is off, AxoGraph attempts to guess the starting values of the free parameters. This slows the fit and may not work. If the Guess starting values check box was on, the Guess dialog appears.

Enter starting guesses for the free parameters, 'a', 'b', etc., then click the Test button. The starting values are substituted into the general equation and the resulting curve is drawn to the screen. If no curve is seen, then the starting guesses are so bad that the curve is above or below the visible region of the graph. Try several starting guesses until the resulting curve loosely approximates the data, then click the Done button.

The Range Selector dialog appears. Use it to select the range of data values to be fitted. See section 11.2 for details on how to select the range.

Click OK and the general equation is fitted to each trace in the front group. The fitted curve(s) are drawn on the screen. The parameters that produced the best fit, and the goodness of fit measure (SSE) are sent to the log window.

The Fitted Function dialog appears that specifies how to handle the fitted curve. By default the curve is not saved. To print or export the graph together with the fitted curve, the curve must first be incorporated into the graph. Turn on the Append to current file radio button and click OK. The fitted curve is appended to the graph window, and the new trace is added to the front group.

Fit General… uses a Simplex optimization procedure to minimize the sum of squared errors between the general equation and the data trace. See Chapter 19, 'Technical Information' for details. The general equation is compiled into memory for maximum speed.

A general curve fit can be used perform a linear or exponential curve fit with one or more parameters held constant during the fit. For example a * exp(-x / 2) is a single exponential with a fixed time constant of 2). This is not possible with the dedicated linear and exponential fits described above (section 11.2 and 11.3).

11.4 Fit a User-Defined Function

Any user defined function can be fitted to a trace, or to several traces simultaneously using the Minimize procedure. Some simple programming is required, but this approach is completely general.

Three standard situations where Fit General… may not be adequate are :

1) A goodness-of-fit criterion other than sum of square errors or Chi sq. is required
2) Fit a single equation to several traces simultaneously.
3) Fit a function that cannot be expressed as a simple equation. For example,
   $f(x) = a * x$ when $x < 0$
   $f(x) = b * x$ when $x > 0$

An example program demonstrating the Minimize procedure is included with AxoGraph. It is in a file called Simultaneous Exp Fit in the More Analysis Programs folder.

The program uses a weighted sum of squared errors (WSSE) to simultaneously fit two exponentials with added constants to two data traces in the front graph window. The WSSE is calculated as,

WSSE = Sum of ((Data - Fit) ^ 2 / abs(Fit))

The second fitted exponential is a scaled version of the first. The first exponential is fitted to the first trace and the scaled exponential is fitted to the second trace. The WSSE is calculated by adding together the WSSE's for each trace. The combined WSSE is a function of four free parameters: the exponential amplitude and time constant, the added constant and the scaling factor. One approach to this optimization problem is to fit the first three parameters to the first trace, then hold these parameters constant and find the optimal scaling factor to fit the second trace. A better approach is to fit both traces simultaneously and to minimize the combined WSSE. All four free parameters are optimized simultaneously in this approach.

To demonstrate the Simultaneous Exp Fit program, it needs to be loaded. Drag the program from the More Analysis Programs folder into the Plug-In Programs folder, then select Program → Reload Plug-Ins. Bring a graph window with at least two traces to the front (2 Pulses for example).
Program → Simultaneous Exp Fit A dialog appears requesting the numbers of the data traces to be fitted and the numbers of traces to receive the fitted exponentials. Click OK. The next dialog requests starting guesses for the exponential parameters. Click OK. A dialog appears requesting the region over which the two exponential curves are to be fitted. Select a region and click OK. The optimization procedure begins, and will take some time to converge (over a minute on slower machines). The slower performance is the price paid for the flexibility of the fitting procedure. When the search converges, the fitted exponentials are appended to the graph window, and the optimal parameters are sent to the log window.

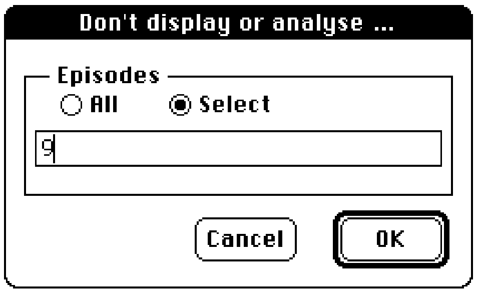
12 Data Analysis (Multi-Episode)

All sections in this chapter assume that a multi-episode graph window is open and that it is the front window. File → Open Digitized… then select a multi-episode graph file (NMDAStep.DAT in the Example Data folder for example - this is a Clampex file with 16 episodes).

12.1 Remove Corrupt Episodes from Analysis and Display

Display a single episode (for example episode 9 in the NMDAStep.DAT file). See section 5.4 for information on how to display selected episodes. The decay phase of this episode contains a small recording artifact and should probably be removed from the display and from subsequent analysis. This can be done as follows.

Display → Mask Displayed The currently displayed episode disappears and is replaced by the next episode. If another episode is masked, it is added to a cumulative list of masked episodes. If more than one episode is displayed when Display → Mask Displayed is selected, all the displayed episodes will be masked. When an episode masked, it will be skipped during subsequent analysis and display.


Don't display or analyse …

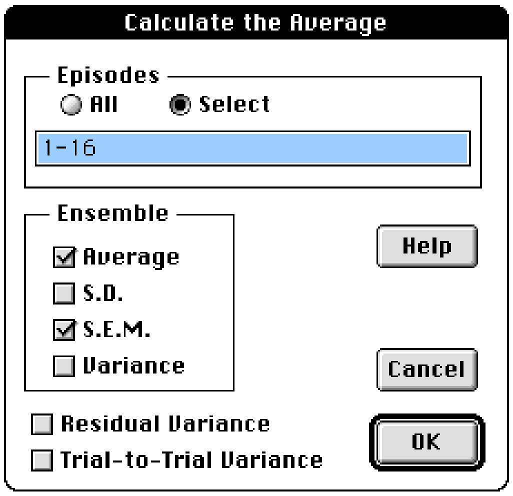
Episodes
All [✓ Select]
9

Display → Mask Selected… The dialog permits the list of currently masked episodes to be edited. Additional episodes can be masked by adding them to the list. Masked episodes can be unmasked by deleting them from the list.

The list consists of numbers separated be spaces or commas. A range of episodes can be included by entering a dash between the lowest and highest numbers in the range. For example, '1 3 5-8' would mask episodes 1, 3, 5, 6, 7 and 8.

12.2 Average Multi-Episode Data

The ensemble average and ensemble variance of a group of traces (episodes) can be calculated. The ensemble average is a trace with a waveform that is the average waveform of the selected episodes. The ensemble variance is a waveform that reflects the amount of episode-to-episode variability. See Chapter 19, for additional details.



Calculate the Average

Episodes
All [✓ Select]
1-16

Ensemble
[✓ Average]
S.D.
[✓ S.E.M.]
Variance

Residual Variance
Trial-to-Trial Variance

Analyse → Average Episodes… The dialog selects which episodes to average. Turn on the All radio button if all episodes are to be included in the average. Turn on the Select radio button if only selected episodes are to be included. To select episodes, enter a list of episode numbers in the Episodes text edit box. The list consists of numbers separated by spaces or commas. A range of episodes can be included by entering a dash between the lowest and highest numbers in the range.
For example, '1 3 5-8' would average episodes 1, 3, 5, 6, 7 and 8.

Turn on the check boxes to calculate :

Average	ensemble average
S.D.	ensemble standard deviation
S.E.M.	ensemble standard error of the mean
Variance	ensemble variance
Residual Variance	ensemble variance after subtracting the optimally scaled average from each episode
Trial-to-Trial Variance	ensemble variance of Ej - (Ej-1 + Ej+1) / 2 where Ej is the jth episode

The resulting traces are displayed in a new graph window.

The ensemble S.D., S.E.M. and variance all provide information about episode-to-episode variability. If this variability is due to background recording noise, the ensemble variance should be approximately flat. If some of the variability is due to episode-to-episode changes in the signal, this will show up as a positive deviation in the ensemble variance waveform. If the signal has a constant waveform but varies in amplitude, the ensemble variance time course will be approximately the signal squared. If the signal has a constant amplitude but varies slightly in waveform, the ensemble variance time course will be approximately the derivative of the signal.

Ensemble S.E.M. is useful for constructing an average with error bars. See Sections 6.10 and 6.11 for information on how to display the S.E.M. values as error bars on the average trace.

The Residual Variance calculation corrects for amplitude fluctuations in the signal by subtracting the optimally scaled average from each episode. The Trial-to-Trial Variance calculation corrects for gradual drift or run-down in the signal amplitude. For additional details on the algorithms used to calculate Ensemble Variance see Chapter 20, 'Technical Information'.

13 Analysis Case Studies

13.1 How to Use the Case Studies

A convention for specifying menu selections is employed throughout the case studies.
File → Open… indicates that the Open... item under the File menu is to be selected.

The case studies rely on files provided in the Example Data folder.

13.2 Construct a Current-Voltage Curve from Voltage-Clamp Step Data

Note: a program that automates current-voltage analysis is shipped with AxoGraph. It is described in the next chapter (section 14.3 under the sub-heading 'I-V Analysis'). The following manual method provides more control over intermediate steps in the analysis.

File → Open Digitized… Select the file VoltageClamp.DAT. This is a Clampex binary data file containing 10 episodes in two A-D channels or groups. Group #1 shows the whole-cell voltage-clamp current, and group #2 shows the corresponding voltage-clamp step commands.

Display → Review Episodes… Turn on the All radio button and click the Scan button. Wait until all 10 episodes are displayed. Optionally zoom the y-axis of group #1 to see the clamp current traces more clearly.

Analyse → Cursor Measure… Turn on the Measure between two cursors radio button, switch on only the Average amplitude check box and click OK. A Range Selector dialog is presented. Enter a range from 48 to 50 ms, switch on the All Groups check box, then click OK. In the Cursor Measurements Complete dialog, click OK again.

Switch to the log window (click on the AxoGraph-1 title bar, or select Window → Show Log). Select the two lists of amplitude measurements, including the titles.

Text → Selection to Table… The two measurement lists are converted to a tab-delimited table with current and voltage columns.

Text → Graph Selection… The Text Import dialog appears. Turn on the Automatic check box, turn on the X vs. Y radio button and set the X Column Number to 2. Click OK and a new graph window appears.

Display → Type of Axes → Zero Crossing The axes now cross at (0,0), the standard format for a current-voltage plot.

Display → Change Axis Range… Set the x-axis range to extend from -60 to +50 mV and set the minor x-tick interval to 10 mV and the major x-tick interval to 30 mV. Click OK.

Display → Symbols and Lines… Turn on both the Line and the Symbol check boxes. Click on the open triangle symbol, set Symbol Size to 4 and Skip By to zero, then click OK.

The current-voltage graph is now complete. It can be saved or closed and discarded.

13.3 Calculate an Amplitude Histogram and Open-Time Histogram from Single-Channel Data

AxoGraph's general purpose analysis features can be applied to the analysis and display of single-channel data. Several dedicated single-channel analysis programs are available for more specialized analysis (for example Fetchan and pStat on the IBM PC or TAC on the Macintosh).

File ➔ Open Digitized… Select the file SingleChannel.DAT. This is a Fetchex binary data file containing 80 seconds of simulated single-channel data. The data is free of offset drift and only a single, well resolved channel is present.

Analyse ➔ Convert to Histogram… Turn on the Simple Histogram radio button and set Bin Width to 0.03 pA, Zero Centered then click OK. A range selector dialog appears. Select all the data and click OK. A new graph window appears containing a point-by- point amplitude histogram of the single-channel data. The two peaks correspond to the closed and open states.

Window ➔ SingleChannel.DAT Copy Bring the SingleChannel.DAT Copy window to the front again.

Display ➔ Type of Axes ➔ Normal Change to normal axes, since peak detection and measurement does not work with raster display.

Analyse ➔ Peak Detection… Select the All radio button in the Detect cluster, turn off the Baseline at left check box, select the Positive radio button in the Direction cluster, set Detect peaks greater than to 50% and Separating valleys are less than to 50%. These settings implement a 50% crossing algorithm for detecting a channel opening. Click the Peaks button.

The Peak Measurement Options dialog appears. Turn off all the check boxes except for Width and set Width at to 50% of peak, then click OK.

Switch to the log window (click on the AxoGraph-1 title bar, or select Window ➔ Show Log). Select (highlight) the list of open-time measurements, including the column title.

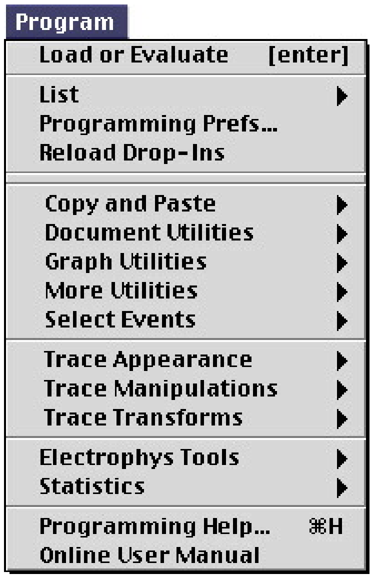
Text ➔ Graph Selection… The Text Import dialog appears. Turn on the Automatic check box and the Y Only radio button, set the X Sample Interval to 1, then click OK. The resulting graph shows the open-times as a function of opening number. This graph is useful for checking stationarity of channel properties with time. It can be converted to an open-time histogram as follows.

Analyse ➔ Convert to Histogram… Select the Simple Histogram radio button, set the Bin Width to 0.04 sec and select the Zero Aligned radio button, then click OK. A range selector dialog appears. Select all the data and click OK. Set the x-axis range from 0 to 700 ms for a clearer display.

Analyse ➔ Fit Exponential… Turn on the Simplex radio button, the Chi Squared radio button, and set Precision to 0%. In the Number of Exps group, turn on the One radio button, turn off the Added constant check box, then click OK. The Range Selector dialog appears. Select the range from 0 to 700 ms, then click OK. The exponential time constant suggests a mean open-time of 150 ms. The Save Exponential dialog appears. Turn on the Append to current file radio button and click OK.

14 The Program Menu

14.1 Overview of the Program Menu



Items in the Program menu are divided into six groups...The first group contains the single command, Load or Evaluate. This command evaluates numeric expressions, or executes program fragments.

The second group contains three commands that control the programming environment and display its current state. The first two groups of items are described in section 14.3.

The next three groups contain hierarchical menu items. These lead to sub-menus that contain all the ‘plug-in’ analysis programs that are currently loaded in memory. Selecting an item from these sub-menus runs the associated analysis program. The last group contains two items that link to online documentation. Sections 14.4 to 14.15 describe all the default plug-in analysis programs that ship with AxoGraph. These programs appear under the Program menu after AxoGraph is installed, but can be customized by the user. Section 14.2 describes how to customize the Program menu.

The last group contains two times that link to online documentation.

When the data acquisition package is loaded, an additional group of hierarchical menu items is added to the Program menu. See the online ‘Data Acquisition Manual’ for information on the acquisition package.

Section 14.16 describes two methods for loading new programs into the Program menu (manual load or via the Plug-In Programs folder).

14.2 Customize the Program Menu

The Program menu can be customized by dragging files or folders into or out of the Plug-In Programs folder. Changes will be reflected in the Program menu next time AxoGraph is launched. If AxoGraph is already running, then select Program → Reload Plug-Ins to dynamically update the Program menu.

Each of the hierarchical items under the Program menu (Copy and Paste through to Statistics) correspond to sub-folders of the Plug-In Programs folder. Most of these folders should be retained as an integral part of AxoGraph, but startup time and menu clutter can be reduced by removing any unused folders. The Electrophys Tools and Statistics folders contain more specialized programs that may not be needed. To remove the Statistics sub-menu, drag the Statistics folder out of the Plug-In Programs folder, then select Program → Reload Plug-Ins.

AxoGraph ships with many specialized analysis programs that are not pre-loaded. These include an event detection package, several electrophysiology analysis programs, a chemical-kinetic modeling package, and many other utility programs. To activate the event detection programs, open the Event Detection Package folder and drag the Event Detection folder into the Plug-In Programs folder, then select Program → Reload Plug-Ins.

14.3 Commands Under the Program Menu

Program → Load or Evaluate. This command takes the selected (highlighted) text from the front text window and loads any programs or functions from the text into memory for future execution. It then takes any remaining text from the selection and attempts to execute it immediately. If the remaining text contains one or more arithmetic expressions, they are evaluated and the numerical result appended to the text window. If no text is selected, the single line of text that contains the flashing cursor is processed as above.

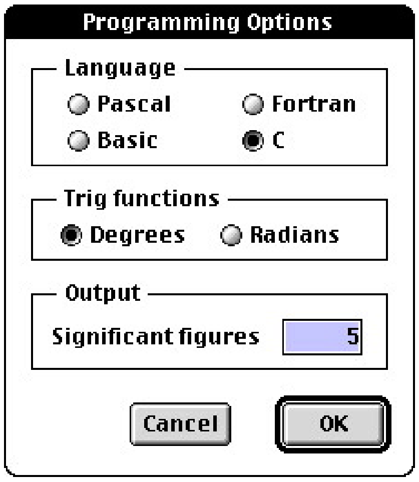
Hitting the '*enter*' key is the standard method of executing this command. The Load or Evaluate menu item is only included as an aid to new users. This command can be used as a calculator. For example, type '2+2' into the log window then hit the '*enter*' key. The result '4' will appear immediately below.

It is important that new programs, functions and global variables are given unique names. The three items under the Program → List menu can be used to check whether a function name or global variable name has been defined.

Program → List → Global Variables The names and values of all global variables currently in memory are listed to the front text edit window.

Program → List → Programs and Functions The names of all currently loaded programs, procedures and functions are listed to the front text edit window.

Program → List → Built In Functions The names of all built-in procedures and functions are listed to the front text edit window.


Programming Options

Language

Pascal Fortran

Basic [✓ C]

Trig functions

[✓ Degrees] Radians

Output

Significant figures 5

Program → Programming Prefs… This dialog sets the default behaviour of the programming environment. Most importantly it selects the default programming Language. It also specifies whether the arguments to trigonometric functions are in Degrees or Radians, and the number of Significant figures to be used when numerical results are output to the text window.

Program → Reload Plug-Ins... All global variables, programs, procedures and functions are deleted from memory, then all programs in the Plug-In Programs folder are reloaded. When programs or folders containing programs are dragged into or out of the Plug-In Programs folder, selecting Reload Plug-Ins dynamically reconfigures the Program menu. This feature is also useful when writing or debugging a plug-in program.

14.4 The Default Plug-In Programs

All the menu items that appear below Reload Plug-Ins under the Program menu are plug-in programs and folders that were automatically loaded into memory when AxoGraph was launched. Selecting a Program menu item runs the associated program. Section 14.2 describes how to customize the Program menu.

The ten default hierarchical items under the Program menu correspond to sub-folders of the Plug-In Programs folder. Functionally related programs are grouped under hierarchical menu items. The ten default hierarchical items are...

Copy and Paste	copy and paste display styles and graph data between windows
Document Utilities	open and close multiple graph, digitized or text files
Graph Utilities	combine, overlay, decimate or interpolate graphs
More Utilities	blank artifacts, normalize, align events, average with error bar
Select Events	select events based on shape parameters or via a mouse click
Trace Appearance	automatically set trace colors and symbols
Trace Manipulations	add, subtract or duplicate traces, subtract sloping baseline
Trace Transforms	integrate, differentiate and custom filter data traces
Electrophys Tools	accurately measure and analyse synaptic amplitudes
Statistics	test the quality of a theoretical fit, or whether two data sets are related

The programs under each of these hierarchical menus are described in the following sections.

14.5 Copy and Paste Display Styles

Program → Copy and Paste → Copy Style A dialog appears asking which internal style clipboard is to receive the display style from the front graph window. This style information includes for each trace the symbol type and size, line type and thickness, the color, the grouping of traces, group position, x- and y-axis ranges, the data columns assigned to each trace, etc. This style information can be pasted back into another graph window within AxoGraph (it cannot be pasted into another program). Up to 8 graph styles can be stored. Style clipboards are preserved when AxoGraph is quit and re-run.

Program → Copy and Paste → Paste Style A dialog appears asking from which internal style clipboard the display style is to be pasted. A second dialog then appears asking which aspects of the display style are to be pasted. The line and symbol styles for each trace are always pasted. Turning on the Paste Axis Titles check box pastes the title for each group. Turning on the Paste Axis Ranges check box pastes the axis range for each group. Turning on the Paste Trace Grouping check box groups the traces in the front graph window in the same way that they were grouped in the original window. Turning on the Paste Column Assignment check box assigns the data columns to the traces and error bars in the front graph window, in the same pattern that was used in the source window. This check box generally should be switched off unless error bars are present. It can produce confusing results. See section 6.10 for information on how to assign columns to traces and error bars. If there were N traces in the 'copy' graph, but there are >N traces in the 'paste' graph, then the style of trace 1 will be pasted to trace N+1, the style of trace 2 will be pasted to trace N+2, etc.

Program → Copy and Paste → Edit Style Names A dialog appears that permits the names of the 8 style clipboards to be changed. These names will appear in the dialog when the Copy Style or Paste Style commands are selected. Informative style names can be defined such as, 'Log-Linear Axes' or 'Thick Lines for All Traces'. The commands for copying and pasting graph styles are also available via the Copy and Paste toolbar at the bottom of the screen.

A Worked Example

File → Open Graph… then select a graph file that has a customized display style (2 Pulses in the Example Data folder for example).
Program → Copy and Paste → Copy Style Select clipboard #1.
File → Close to close the front window.
File → Open Graph… turn on the Default Display Settings check box, then select the same graph file. It will open and display the graph data using the default style settings.
Program → Copy and Paste → Paste Graph Style Select clipboard #1, then click OK. Turn on the Paste Axis Titles check box and the Paste Trace Grouping check box, and turn off the other two check boxes then click OK. The graph style information from clipboard #1 is applied to the front graph window.
Program → Copy and Paste → Edit Style Names Rename clipboard # 1 to 'Alternate Blue and Red Traces'.

14.6 Copy and Paste Graph Data

Program → Copy and Paste → Copy Data A dialog appears asking which subset of the data in the front graph window to copy to an internal clipboard. If Copy All X and Y Data is selected, then all the graph data will be copied. If Copy Y Data Traces from Front Group is selected, then only the traces in the front group will be copied. This data cannot be pasted into another program; it can only be pasted back into a graph window within AxoGraph.

Program → Copy and Paste → Paste Data A dialog appears asking which subset of the data on the internal clipboard to paste into the front graph window.

Program ➔ Copy and Paste ➔ Graph Copied Data A new graph window is created from the data on the internal clipboard. The display style of the original graph data is preserved. The commands for copying and pasting graph data are also available via the Copy and Paste toolbar at the bottom of the screen.

An alternative way to combine data from separate graph windows is by selecting
Program ➔ Graph Utilities ➔ Combine Selected Graphs This feature is described in section 4.8.

A Worked Example

File ➔ Open Graph… then select a graph file (2 Pulses in the Example Data folder for example).
Program ➔ Copy and Paste ➔ Copy Data   Select Copy Y Data Traces from Front Group, then click OK.
Program ➔ Analyse ➔ Scale and Offset Y   Enter a scaling factor of '2', then click OK.
Program ➔ Copy and Paste ➔ Paste Data   Select Paste Y Data, then click OK. The two copied traces are added to the front window. The display style of the original traces is preserved.

14.7 Document Utilities

The commands in the Document Utilities sub-menu can be used to open and close multiple graph, digitized or text files with a single menu selection.

Program ➔ Document Utilities ➔ List Documents This command lists the names of documents in a selected folder. A dialog appears asking which class of documents to list: Graph, Digitized, Text or All documents. This is followed by a standard dialog that indicates the folder containing the files to be listed. For example, choose the Graph radio button in the first dialog, then select the Example Data folder. The following file names should be listed...

2 Pulses
Current-Voltage (AxoGraph)
Current-Voltage (Comma Text)
Current-Voltage (CricketGraph)
Current-Voltage (KaleidaGraph)
Current-Voltage (Tab Text)
Dose-Response
Operational Model Data

Program ➔ Document Utilities ➔ Open Selected Documents The selected (highlighted) documents are opened. For example select the names 'Dose-Response' and 'Operational Model Data' in the above list then run the above program. The two selected graphs will be opened. This feature is useful when a large number of data files are collected in a single folder. Selected subsets of files can be conveniently opened for review and analysis.

Program ➔ Document Utilities ➔ Open All Documents All documents in the selected folder are opened. A dialog appears asking which class of documents to open: Graph, Digitized, Episodic, Text or All documents. This is followed by a standard dialog that indicates the folder containing the files to be opened.

Program ➔ Document Utilities ➔ Open Next Digitized Open the next file in the alphabetic list of digitized data files the current folder. It is assumed that the first digitized data file in the folder has been manually opened.

Program ➔ Document Utilities ➔ Open Prev Digitized Same as the above command, but open the previous digitized data file in the alphabetic list.

Program → Document Utilities → Close Graph Windows Close all currently open graph windows. A dialog appears asking whether to save changes before closing the windows.

Program → Document Utilities → Save & Close Graphs Save changes, then close all currently open graph windows.

Program → Document Utilities → Copy Graph Data to Log The data in the front graph window is copied to the log window in tab-delimited format.

Program → Document Utilities → Export Graphs to Text All currently open graph windows are exported to tab-delimited text data files. The export file names are constructed from the current file name plus the extension ‘.txt’.

Program → Document Utilities → Compare Text Windows Compare the contents of two text windows and list the non-matching lines to the log window.

Program → Document Utilities → Tidy AxoData Log The AxoData acquisition program generates a log file that is difficult to read because it contains a lot of extraneous detail. This command automatically deletes the extraneous information making the log file more usable. The command assumes that the front window is an AxoData log file.

14.8 Graph Utilities

Program → Graph Utilities → Combine Open Graphs All currently open graph and digitized files are combined into a single new graph. Traces with the same group number are overlaid in a single group in the combined graph. This command is useful for combining and overlying data traces from different data sets. The onset of each trace is aligned in the combined graph.

Program → Graph Utilities → Combine Selected Graphs A dialog with a list of currently open graph windows is presented. Traces from the selected windows are combined into a single new graph as above.

Program → Graph Utilities → Concatenate Open Graphs All currently open graph and digitized files are combined into a single new graph. Traces are merged end-to-end in the combined graph. This option is useful for combining data sets that were generated sequentially. The time sequence of the data points is preserved in the combined graph. The onset of each trace is aligned with the termination of the preceding trace.

Program → Graph Utilities → Overlay Open Graphs All currently open graph and digitized files are combined into a single new graph. Traces are merged and overlaid in the combined graph. This option is useful for combining several scatter plots into a single plot.

Program → Graph Utilities → Decimate Front Graph The number of data points in the front open graph is reduced by decimating the x- and y-axis data columns. This consists of replacing groups of N data points with either the last point in the group, or the average of the group.

Program → Graph Utilities → Decimate with Error Bars The number of data points in the front open graph is reduced as above. An additional data column is generated containing the Standard Error of each group of N data points. This column is used to generate error bars for each data point of the decimated graph.

Program → Graph Utilities → Interpolate Front Graph The number of data points in the front open graph is increased by interpolating between data points in the x- and y-axis columns. This can be used to facilitate the combination of data sets acquired at different sample rates, or binned with different bin widths.

Program → Graph Utilities → Tile Graph Windows Tile all currently open graph windows in a square pattern that fills the screen. A dialog appears asking how many horizontal and vertical tiles to use.

14.9 More Utilities

The commands in the More Utilities sub-menu perform some common types of manipulation on traces in a graph window.

Program → More Utilities → Blank Artifact A dialog appears which is used to select a region over which to perform blanking. Select a range that covers the artifact, but precedes the rise of the response, then click OK. An average is calculated over a small region immediately before the selected region, and then all points in the selected region is set to this value, effectively blanking out the artifact.

Program → More Utilities → Interpolate Artifact A dialog appears which is used to select a region over which to perform interpolation. Select a range that covers the last part of the baseline, and the first part of the rise of the response, then click OK. Data points in the selected region are adjusted so that they form a line between the first and last points of the region. This can be used to blank an artifact that occurs in a sloping region of a trace.

Program → More Utilities → Zero Artifact A dialog appears which is used to select a region over which to zero the trace. Select a range, then click OK. All data points in the selected region are set to zero. This simple behaviour may be preferred to Blank Artifact under some conditions.

Program → More Utilities → Normalize Over Range A dialog appears which is used to select a range over which to normalize. Select a range, then click OK. For each trace in the front group, the average is calculated over the selected range, and the amplitude of the trace is divided by the average. This is useful for normalizing noisy traces.

Program → More Utilities → Align at Onset A dialog appears asking for an onset threshold relative to baseline standard deviation. Select a value (typically 10 to 20) then click OK. A second dialog appears which is used to select a region of baseline that is common to all traces in the front group. Select a range, then click OK. For each trace in the front group, the onset of the response is identified (first data point with absolute value greater than the threshold), and the trace is adjusted so that the onset occurs immediately after the last point of the common baseline region. This aligns all the traces so that their onsets occur at the same time.

Program → More Utilities → Average Front Group An average is constructed of all traces in the front group, and the result is displayed in a new graph window.

Program → More Utilities → Average Aligned Traces A dialog appears asking for a range of windows and/or traces over which to average. The program assumes that all the chosen traces contain an event of variable latency. It presents each trace in turn together with a dialog for manually identifying the onset of the event. The program aligns the responses by discarding some of the baseline from traces with delayed responses, then constructs an average of the aligned events in a new graph window.

Program → More Utilities → Maximum Slope A dialog appears asking for the number of data points over which to calculate the local slope. Select a value (typically 2 to 20), then click OK. A second dialog appears which is used to select a region to search for the point of maximum local slope. Select a range, then click OK. For each trace in the front group, the point of maximum local slope is found and a red line is drawn indicating the point and the slope. Detailed numerical results are sent to the log window.

14.10 Select Events

The commands in the Select Events sub-menu are used to restrict automatically mask or unmask selected events in an episodic data file. Traces can be masked and excluded from subsequent analysis based on the amplitude or shape of the event they contain.

Program → Select Events → Mask Large Events All episodes in the front group are examined. Episodes that contain an event with an amplitude greater than a specified threshold are masked. This permits the selective display and analysis of small events. Large events can be displayed and analysed by subsequently selecting Swap in Masked Episodes. A dialog is used to select the amplitude threshold. A second dialog is used to limit the range over which to search for events.

Program → Select Events → Mask Misshaped Events All episodes in the front group are examined. Episodes that contain an event with shape parameters outside a selected range are masked. This permits the elimination of misshaped events (glitches, noisy traces, etc.). A dialog is used to select the shape parameters to be tested. The shape parameters that can be used are peak amplitude, 20-80% rise-time, half-width, event onset (20% of peak), baseline standard deviation, and baseline offset. A histogram is constructed displaying the distribution of each parameter, and cursors are used to limit the acceptable parameter range.

Program → Select Events → Mask Even Episodes All even numbered episodes are masked.

Program → Select Events → Pick One Episodes This program picks a single episode out of a the displayed episodes. A dialog appears requesting a mouse-click point. The episode that passes closest to that point is displayed and the episode number is shown. This is a convenient method for identifying an individual episode to be analysed or masked.

Program → Select Events → Swap in Masked Episodes All currently masked episodes are unmasked, and all currently unmasked episodes are masked. As an example, this program can selectively display and analyse all large events when applied following the Mask Large Events program.

Program → Select Events → Unmask All Episodes All episodes in the front graph window are unmasked.

14.11 Electrophys Tools

The commands in the Electrophys Tools sub-menu are designed to help measure and analyse synaptic currents recorded in whole-cell patch clamp recording mode.

Both high frequency noise contamination and synaptic latency jitter contribute to synaptic amplitude measurement errors. These errors can be reduced by calculating the average amplitude over a period that includes part of the rising and falling phases of the synaptic current. The average amplitudes will always be less than the peak amplitude and generally need to be scaled up to match the peak synaptic current. One of the following programs performs this two stage amplitude measurement.

When synaptic amplitudes are large (> 1 nA), they can be distorted by voltage clamp error due to the patch electrode series resistance (Rs). If the series resistance is known, then it is possible to compensate the recorded currents for this distortion. Two of the following programs measure Rs, and generate a series-resistance compensated synaptic current.

Program → Electrophys Tools → Measure Rs   This program assumes that the front window contains a whole-cell current trace which includes the response to a square voltage-clamp test pulse. The pulse parameters are entered in a dialog. The series resistance of the patch electrode is then calculated and written to the log.

Program → Electrophys Tools → Correct for Rs   This program assumes that the front window contains a whole-cell current response containing large amplitude synaptic transients. The synaptic current is corrected for any systematic errors due to electrode series resistance. This is a simple, linear correction that assumes all membrane conductances have a reversal potential of zero.

Program → Electrophys Tools → Measure Synaptic Peaks   This program assumes that the front window contains evoked synaptic currents. It measure the peak amplitudes of synaptic responses by averaging over a region around the peak of each response, then scaling these measured amplitudes to match the peak amplitude of the ensemble average synaptic response.

Program → Electrophys Tools → Measure Two Peaks   This program measures the peak amplitudes of two separate synaptic responses using the same technique as above.

Program → Electrophys Tools → Calculate Variance-Mean   This program assumes that the front window contains evoked synaptic amplitudes plotted against stimulus number. It requests an analysis range, then measures the variance and mean of the synaptic amplitudes within that range. The program optionally corrects for rundown of the response and eliminates outlier amplitudes before calculating the mean and variance. The results are directed to the log.

When variance and mean have been measured under several different release probability conditions, the results accumulated in the log can be used to construct a variance-mean (V-M) plot. This can be done by tidying the results into two tab-delimited columns, the plotting them via **Graph Selection** under the **Text** menu. A V-M plot typically has a parabolic form.

Program → Electrophys Tools → Fit Var-Mean Parabola   This program assumes that the front window contains a V-M plot (see above). It fits a parabola to the V-M plot, thereby providing estimates of the average ‘quantal’ parameters for the synapse. These parameters are...
  the average amplitude of the response to a single vesicle of transmitter (Q_{av})
  the average probability of transmitter release from a terminal (P_{av})
  the number of synaptic terminals (N)

An article introducing variance-mean analysis was published in AxoBits recently. To access this article, click the following hyperlink...
http://www.axon.com/pub/axobits/AxoBits28.pdf

For a complete description of the variance-mean technique, see....

John D. Clements and R. Angus Silver
Unveiling synaptic plasticity: a new graphical and analytical approach.
Trends in Neurosciences (2000) **23**:105-113

To access this article, click the following hyperlink…
http://www.biomednet.com/library/abstract/TINS.etd00233_01662236_v0023i03_00001520

14.12 Statistics

The commands in the Statistics sub-menu perform some basic statistical tests on traces in a graph window. There are commands for assessing the quality of a theoretical fit to a data set using the Chi-Squared statistic, and for comparing two data sets using the Kolmogorov-Smirnov statistic.

Program ➔ Statistics ➔ Assess Quality of Fit Calculate the Chi-Squared statistic and probability level for an optimized fit to a data trace. The front graph should contain at least a data trace and a fitted trace. If the graph is not a histogram, then either a standard-error trace or error-bars on the data trace are also required. A dialog appears asking for the trace number of the data and of the fitted curve. (Cancel, then select Trace ➔ Group to help determine the correct trace numbers to enter). Enter the trace numbers, and the number of free parameters in the equation used to fit the data. If the trace is a binned histogram, then turn on the Fitted Data is a Histogram check box. Click OK. A second dialog appears requesting the range over which to calculate the Chi-Squared statistic. If the data trace is not a binned histogram, then the standard error of each data point is required. A third dialog appears asking for the trace number containing the standard-errors. If the data trace has error bars, then it is assumed that these represent ± 1 standard error. The results of the Chi-Squared analysis are displayed in a final dialog, and are also sent to the log window.

Program ➔ Statistics ➔ Compare Two Data Sets Test the hypothesis that two data sets are drawn from the same distribution. The test uses the Kolmogorov-Smirnov statistic. The data sets can be arranged as two sequences of unbinned data points, two binned histograms, or two normalized cumulative histograms. A dialog appears asking for the window and trace number of each data set. (Cancel, and select Trace ➔ Group to help determine the correct trace numbers). Enter the window and trace numbers, and click OK. A second dialog appears asking for the data format. Both data sets should be in the same format. If the traces contain a sequence of unbinned data values, then chose the Unbinned Data radio button. If the data values have been binned into histograms, then chose the Binned Data radio button. If the histograms have been scaled to give a probability density estimates (area under the curve = 1), then chose the Probability Density Estimate radio button. If the histograms have been integrated and normalized, then chose the Normalized Cumulative Histograms radio button. The results of the Kolmogorov-Smirnov analysis are displayed in a final dialog, and are sent to the log window.

14.13 Trace Appearance

The commands in the Trace Appearance sub-menu alter the display style (color, symbol, line thickness) of several traces simultaneously.

Program ➔ Trace Manipulations ➔ Set Color by Trace # The 1st trace is displayed in blue, the 2nd trace in red, and the 3rd trace in green. Up to 8 traces will automatically be set to different colors.

Program ➔ Trace Manipulations ➔ Set Color by Group # The 1st group is displayed in blue, the 2nd group in red, and the 3rd group in green. Up to 8 groups will automatically be set to different colors.

Program ➔ Trace Manipulations ➔ All Blue All traces and symbols are displayed in dark blue.

Program ➔ Trace Manipulations ➔ All Black All traces and symbols are displayed in black. This options is useful when preparing a graph for publication.

Program ➔ Trace Manipulations ➔ All White All traces and symbols are displayed in white. This options is useful when using a dark graph background.

Program → Trace Manipulations → Set Symbols by Trace # The 1st trace is displayed using open circles, the 2nd trace using open squares, and the 3rd trace using open diamonds. Up to 12 traces will automatically be set to use different symbols. If more than 12 traces are present, the symbol sequence will repeat.

Program → Trace Manipulations → Set Symbols by Group # The 1st group of traces is displayed using open circles, the 2nd group using open squares, and the 3rd group using open diamonds. Up to 12 groups will automatically be set to use different symbols. If more than 12 groups are present, the symbol sequence will repeat.

Program → Trace Manipulations → Set Line Width by Trace # The 1st trace is displayed using a line width of 1, the 2nd trace using a line width of 2, etc.

Program → Trace Manipulations → Set Line Width to 1 All traces are displayed using a line width of 1.

Program → Trace Manipulations → Remove All Symbols All symbols are removed. Traces will be displayed using only lines.

Program → Trace Manipulations → Remove All Lines All lines are removed. Traces will be displayed using only symbols.

14.14 Trace Manipulations

The commands in the Trace Manipulations sub-menu perform basic mathematical manipulations on traces in a graph window.

Program → Trace Manipulations → Group Math A dialog appears requesting the math operation to perform on two groups of traces. Select Subtract, Add, Multiply or Divide and click OK. A second dialog appears requesting the window and group numbers of groups to process. The selected math operation is applied to episode #1 of the first and second group, then episode #2 of the first and second group, etc. A new window is created that contains the resulting traces.

Program → Trace Manipulations → Trace Math A dialog appears requesting the math operation to perform on two traces. Select Subtract, Add, Multiply or Divide and click OK. A second dialog appears requesting the window and trace numbers of traces to process. A new window is created that contains the resulting trace.

Program → Trace Manipulations → Sort Data Points The data points in a scatter plot are re-ordered so that the x-axis data values are in ascending order. This may improve the appearance of a graph when using line display format.

Program → Trace Manipulations → Duplicate Each Group Each group of traces is duplicated into a new graph window. This program permits each group to be analysed or manipulated independently of the original data.

Program → Trace Manipulations → Duplicate Section of Trace A selected region of the front group of traces is duplicated into a new graph window. This program permits the selected sub-section of the displayed data to be analysed or manipulated independently of the original data.

Program → Trace Manipulations → Sloping Baseline A dialog appears which is used to select a region over which to calculate a sloping baseline. For each trace in the front group, a line is fitted to data points in the selected region, and the fitted line is subtracted from the trace.

Program → Trace Manipulations → Two Region Baseline   A dialog appears which is used to select two regions over which to calculate a sloping baseline. For each trace in the front group, a line is fitted through all the data points in the two selected regions, and the fitted line is subtracted from the trace.

14.15 Trace Transforms

The commands in the Trace Transforms sub-menu perform simple arithmetic transforms on traces in a graph window.

Program → Trace Transforms → Invert Front Group
Each trace in the front group is inverted (multiplied by -1).

Program → Trace Transforms → Rectify Front Group
Each trace in the front group is rectified (absolute value of all data points).

Program → Trace Transforms → Integrate Front Group
Each trace in the front group is numerically integrated.
The integrated traces are displayed in a new graph window.

Program → Trace Transforms → Differentiate Front Group
Each trace in the front group is numerically differentiated.
The differentiated traces are displayed in a new graph window.

Program → Trace Transforms → Delta-Y Front Group
Each data point in each trace in the front group is replaced by difference between that point and the preceding point. The result is similar to numerical differentiation. The resulting traces are displayed in a new graph window.

Program → Trace Transforms → Raise Front Group to Power
Each trace in the front group is raised to a power that is specified via a dialog.
The Y-axis units are not adjusted.

Program → Trace Transforms → Perfect Filter
Every trace in the front window is filtered.
FFT's are used to implement a ‘perfect’ filter (i.e. sharp cutoff above the selected frequency).

Program → Trace Transforms → Notch Filter
Every trace in the front window is filtered.
FFT's are used to implement a ‘perfect’ notch filter (i.e. remove a selected frequency).

Program → Trace Transforms → Box Car Filter
Every trace in the front window is filtered using a box-car filter. The advantage of this filter is that it is very fast. It replaces each point in a trace with the average of the surrounding points. The number of points to average is entered via a dialog.

Program → Trace Transforms → Three-Point Filter
Every trace in the front window is filtered using a three-point filter. This filter operation is very fast to calculate. It replaces each point in a trace with the center-weighted average of itself and its two neighboring points.

Program → Trace Transforms → High-Pass Filter

Every trace in the front window is filtered using a high-pass filter. FFT's are used to implement a ‘perfect’ high-pass filter (i.e. sharp cutoff below the selected frequency). Program → Trace Transforms → Sub-region Spectrum
The power spectrum of the selected sub-region of each trace in the front group is calculated. Either the average spectrum or the individual spectra are displayed in a new graph window.

14.16 Online Help

This section describes the last 2 items on the Program menu...

Program → Programming Help... A dialog appears with a list of help topics related to AxoGraph's programming languages, and its built-in commands and functions. Select one or more topic and click OK. An online documentation window is opened with information on the selected topic. The documentation is stored in text files in the Documentation : Help Menu Files folder.

Program → Online User Manual A documentation window appears with information about the two online user manuals. These include the manual that you are currently reading.

14.17 Add Custom Programs to the Menu

Manual Load
Programs can be written in Pascal, Fortran, Basic or C, then loaded into memory and run at a later time. The document containing the source code does not need to remain open once the program or function has been loaded. Programs, functions and procedures are loaded manually by selecting the source text (using Edit → Select All) then pressing the ‘*enter*’ key. When a program is loaded, its name is appended to the bottom of the Program menu. See the next chapter for more information about how to write a program in AxoGraph. A complete list of currently loaded programs, functions and procedures can be obtained by selecting Program → List Programs.

Automatic Load
When AxoGraph starts up, it looks for a folder called Plug-In Programs located in the same folder as the AxoGraph application, and opens any program or compiled module files that it finds in this folder. It also searches sub-folders in the Plug-In Programs folder for additional files. The contents of each file is loaded into memory, and the file is closed. The loaded programs have their names added to the Program menu. Files are loaded in alphabetical order and programs are added to the menu in the order they are loaded.

15 Plug-In Analysis Programs

15.1 The Analysis Programs

The programs described in this section are found in files and folders inside the Event Detection Package folder, the Electrophysiology Analysis folder and the Models and Utilities folder. These programs need to be loaded before they can be used. Instructions are provided in the online documentation on how to load each program. After an analysis program is loaded, it can be run via a menu item or sub-menu item under the Program menu.

The Event Detection Package folder contains documentation and programs for detecting, capturing, sorting and analysing asynchronous events in continuous or episodic data files. Detection algorithms include amplitude threshold, first-derivative threshold, and template matching techniques.

The Electrophysiology Analysis folder contains the following analysis packages:

Current-Voltage (I-V) Analysis
Population Spike Analysis
Quantal Analysis

A brief overview of these packages is supplied below. Additional information is supplied in online documentation that can be accessed via the Program menu. For example, after loading the Current-Voltage Analysis package select the menu item Program ➔ Current-Voltage Analysis ➔ About I-V Analysis.

The Models and Utilities folder contains the following analysis packages:

Chemical-Kinetic Model
Electrophysiology Models
Foreign File Import / Export
More Analysis Programs
Utility Programs

The Chemical-Kinetic Model folder contains a package of programs for modeling the evolution in time of a chemical-kinetic model (Markov model). This package can be used to model the properties of voltage and ligand gated channels with complex reaction schemes. It can also be used to estimate reaction rate constants by fitting the predicted current to a recorded current transient.

The Electrophysiology Models folder contains the following three modeling packages:

MK-801 Model
Vesicle Release Model
Operational Model Fit

A brief overview of these packages is supplied below.

The Foreign File Import / Export folder contains programs for importing and exporting data files in various formats. It also contains documentation on how to write programs of this type.

The More Analysis Programs folder and the Utility Programs folder contain programs for analyzing and manipulating data that are not likely to be widely useful. They are provided here as examples, and potential starting points for people wishing to write custom analysis programs in AxoGraph.

Current-Voltage Analysis

The analysis programs in this package are intended for data recorded with a voltage clamp step protocol. A typical protocol starts each episode at a constant command potential (holding potential), then steps to a test potential that changes from episode to episode. An I-V curve is constructed by plotting the clamp current vs. the voltage during the step.

Population Spike Analysis

The population spike is the a transient in an extracellular field recording due to the synchronous firing of a population of neurones. The ‘spike’ is typically negative going when the recording electrode is in a region containing predominantly cell bodies. The population spike is superimposed on a slower field epsp, and this is typically positive going. Thus, the population spike appears as a brief negative peak in the middle of a broad positive hump. This creates two small positive peaks and one large negative peak.

The program ‘Population Spike Analysis’ automatically identifies the two small positive peaks and the negative population spike peak. A line is extrapolated between the two positive peaks, and the population spike amplitude is estimated relative to this line.

This program also estimates the amplitude of the underlying EPSC by calculating the maximum slope on the rise of the EPSP field.

Quantal Analysis

The program ‘Simulation’ generates a synaptic amplitude histogram using a binomial model of transmitter release. The model incorporates quantal variability and stimulus failures. The program also generates a simulated list of amplitude measurements and the theoretical distribution from which the amplitudes were sampled.

The program ‘Analysis’ fits a theoretical curve to an amplitude histogram. It then tests the adequacy of the fit using the Chi Squared statistic. The theoretical curve uses a binomial model to describe transmitter release statistics, and incorporates quantal variability and stimulus failures.

MK-801 Model

This program models the progressive block of an NMDA receptor mediated synaptic current in the presence of the irreversible open channel blocker, MK-801. It is assumed the synaptic transmission is mediated by two populations of synaptic terminals with high and low probability of transmitter release. Each time a terminal releases transmitter in the presence of MK-801, a fraction of the postsynaptic NMDA receptors are irreversibly blocked. The peak amplitude of the synaptic current is plotted as a function of stimulus number. It typically has a double exponential form in the presence of MK-801.

Vesicle Release Model

This program models the kinetics of vesicle release. It assumes that each terminal contains a reservoir of vesicles, and a cluster of proteins and cytoskeletal elements attached to the terminal membrane that together constitute the ‘release complex’. Each release complex has several vesicle docking sites. These serve to bind and hold vesicles in a position close to a single ‘release machine’. One vesicle at a time can be loaded from a docking site into the release machine. The loading rate is proportional to the number of vesicles docked to the complex. Loaded vesicles are released at a very slow rate under control conditions. However, the release rate is a time dependent variable, and can be increased during a stimulus period. The other rates (docking and loading) are invariant.

This model describes release from a single terminal. If several terminals are present, and if all terminals are identical (contain the same number of reservoir vesicles and release complexes), then the parameters for all terminals can be lumped into a single meta-terminal. In this case, the parameter values describe the meta-terminal.

Operational Model Fit

Two dose-response curves are simultaneously fitted using the operational model of agonism. The parameter, tau, may be different for the two dose-response curves.

The operational model is defined by the equation...

Em * tau ^ n * A ^ n / ((Ka + A) ^ n + tau ^ n * A ^ n)

where
A = Agonist concentration
Em = Maximum amplitude of response
tau = Transducer ratio
Ka = Affinity
n = Hill coefficient (index of cooperativity)

15.2 Case Studies Using Analysis Programs

Case studies are presented for two of the analysis programs, Current-Voltage Analysis and Population Spike Analysis. The case studies refer to files supplied in the Example Data folder.

Current-Voltage Analysis

The commands in the Current-Voltage Analysis sub-menu generate current-voltage (I-V) curves from multi-episode data. These programs can be applied to data recorded with a variety of different protocols.

File → Open Digitized… then select a multi-episode digitized data file containing current and voltage traces (VoltageClamp.DAT in the Example Data folder for example).
Program → Current-Voltage Analysis → I-V Setup and Run Turn on the Voltage Data was Recorded check box, and turn off the Leak Data was Recorded check box. Specify the current and voltage group numbers (1 and 2 respectively for the VoltageClamp.DAT example file). Select the range over which to measure the current and the voltage. The measurements are made by averaging the data points over the selected range for each episode. The resulting current measurements are plotted against the voltage measurements in a new graph window. The axis is automatically set to 'zero-crossing' which is the standard for I-V curves.

The Setup process only needs to be done once.
To perform I-V analysis on another file select Program → Current-Voltage Analysis → I-V Run

Voltage data is not required. It can be filled in by the I-V analysis programs.
File → Open Digitized… then select a multi-episode digitized data file containing no voltage traces (CurrentTTX.DAT in the Example Data folder for example).

Program → Current-Voltage Analysis → I-V Setup and Run Turn off the Voltage Data was Recorded check box. Specify the current group number (1), then specify the voltage for the first voltage-clamp step (-60 mV), and the step increment (5 mV). Select the range over which to measure the current. The current measurements are made by averaging the data points in the selected range for each episode. The resulting current measurements are plotted against the calculated voltages in a new graph window.

Leak subtraction can be performed by the I-V analysis programs. File → Open Digitized… then select a multi-episode digitized data file containing leak current data (CurrentTTX.DAT in this example). Open another digitized file containing the currents of interest (Current.DAT in the Example Data folder for example). Make sure the data of interest is in the front window, and the leak data in another window. Program → Current-Voltage Analysis → I-V Setup and Run   Turn on the Leak Data was Recorded check box. Specify the current group number (1), then specify the voltage for the first voltage-clamp step (-60 mV), and the step increment (5 mV). Specify the numbers of the windows containing the test (Current.DAT) and leak (CurrentTTX.DAT) current traces. Select the range over which to measure the current. For each episode, the leak current trace is subtracted from the test current trace and The resulting current measurements are plotted against the calculated voltages in a new graph window.

Population Spike Analysis

File → Open… then select a multi-episode digitized data file containing population spike data (Pop Spike (AxoData) in the Example Data folder for example). Program → Population Spike   Set Regression points for max slope to 7, and click OK. Select the region for the population spike analysis. This region must exclude the stimulus artifact, and include two positive going peaks on either side of a negative going population spike (e.g. from 11 to 25 ms). Next, select a point close to the peak of the population spike (~15 ms). The point of maximum slope on the rising phase is located using the selected number of regression points, and is indicated by a red line. The two positive going peaks and the negative going population spike are detected and marked with black crosses. A line is interpolated between these two peaks and the population spike amplitude is measured from the population spike minimum (negative peak) to the interpolated line. The interpolated line and the population spike amplitude are drawn in green. All results are output to the log window.

16 Writing Programs

16.1 Introduction to Programming AxoGraph

All the information in this chapter, additional information about the programming languages and simple example programs, are available via AxoGraph's online help... Program → Programming Help.

AxoGraph contains a built in programming environment. Programs can be written in Pascal, Fortran, Basic or C and executed from any text window or via the Program menu. The default language is specified using the menu command Program → Programming Prefs, but can be overridden if the first line of the source file is of the form...

LocalLanguage *LanguageName*

where *LanguageName* is Pascal, Basic, Fortran or C.

Many useful programs are supplied in the Plug-In Programs folder the Electrophysiology Analysis folder, the More Analysis Programs folder and the Extras folder. These programs can be customized or adapted to perform new tasks. AxoGraph's startup time and memory overhead can be reduced by removing unused programs and folders from the Plug-In Programs folder.

Programs can manipulate data and change display parameters in any open graph window. Programs can interact with the user via standard dialogs. Frequently used programs can be assigned names which are then appended to the Program menu. Programs can be automatically executed or loaded onto the menu each time AxoGraph is run.

A minimal knowledge of one of the above programming languages and of AxoGraph's built in functions is required to program AxoGraph effectively. The time invested leaning to program AxoGraph greatly extends analysis flexibility and can lead to automation of routine data analysis.

The programming environment can be used as a sophisticated scientific calculator. Language syntax has been relaxed to make numerical calculations easier. Calculations are performed by typing them into a text window then pressing ‘enter’. A record of the calculation is maintained that can be cross-checked or printed. Standard scientific functions (sin, cos, exp, ...) and many statistical functions are supported. Several array functions (FFT, FitLine, etc.) are also supported.

The programming languages are ‘interpreted’, which means that code can be run instantly (it does not need to be compiled first). This provides many advantages. For example, code can be executed one line at a time as it is written, intermediate results can be checked at any point, and subroutines can be executed and tested before being integrated into a larger program.

The Language Implementations are Not Standard
AxoGraph's implementations of Pascal, Basic, Fortran and C are designed for writing simple data analysis programs. Some advanced language features are not implemented. In particular, AxoGraph C is a greatly simplified version of this large, complex language. None of the languages is case sensitive. Normally C is

case sensitive. Even if you are very familiar with one of the above languages, please take a few moments to check the documentation for that language to see what features are available.

16.2 A Programming Example in Basic

Here is an example of how to create a simple program using the Basic language. First select Program _ Preferences… and turn on the Basic radio button, then click OK. A program always begins with the word 'program'. This word is followed by a string enclosed in quotes. The string is appended to the Program menu when the program is loaded. If the second last character of the string is a slash (/), then the last character becomes a command key equivalent for running the program. Load the program by selecting (highlighting) it in the text editor window, then hitting the '*enter*' key.

```
program "Example Program/0"
    Print ""
        Print "Calculate factorial numbers from 1 to 10"
    f = 1
    For i = 1 to 10
        f = f * i
        Print "Factorial ",i," = ",f
    Next i
end
```

If the program loads successfully, Example Program should now be the last item under the Program menu.

To run the program, select Program _ Example Program.

16.3 A Programming Example in Fortran

Here is an example of how to create a simple program using the Fortran language. First select Program _ Preferences… and turn on the Fortran radio button, then click OK. A program always begins with the word 'program'. This word is followed by a string enclosed in quotes. The string is appended to the Program menu when the program is loaded. If the second last character of the string is a slash (/), then the last character becomes a command key equivalent for running the program. Load the program by selecting (highlighting) it in the text editor window, then hitting the '*enter*' key.

```
program 'Example Program/0'
    write ('Calculate factorial numbers from 1 to 10',newLine)
    f = 1
    do i = 1, 10
        f = f * i
        write ('Factorial ',i,' = ',f,newLine)
    endDo
end
```

If the program loads successfully, Example Program should now be the last item under the Program menu.

To run the program, select Program → Example Program.

16.4 A Programming Example in Pascal

Here is an example of how to create a simple program using the Pascal language. First select Program → Preferences… and turn on the Pascal radio button, then click OK. A program always begins with the word 'program'. This word is followed by a string enclosed in quotes. The string is appended to the Program menu when the program is loaded. If the second last character of the string is a slash (/), then the last character becomes a command key equivalent for running the program. Load the program by selecting (highlighting) it in the text editor window, then hitting the '*enter*' key.

```
program 'Example Program/0'
begin
    writeln ('Calculate factorial numbers from 1 to 10');
    f := 1;
    for i := 1 to 10 do
        begin
        f := f * i;
        writeln ('Factorial ',i,' = ',f);
    end;
end;
```

If the program loads successfully, Example Program should now be the last item under the Program menu.

To run the program, select Program → Example Program.

16.5 A Programming Example in C

Here is an example of how to create a simple program using the C language. First select Program → Preferences… and turn on the C radio button, then click OK. A program always begins with the word 'program'. This word is followed by a string enclosed in quotes. The string is appended to the Program menu when the program is loaded. If the second last character of the string is a slash (/), then the last character becomes a command key equivalent for running the program. Load the program by selecting (highlighting) it in the text editor window, then hitting the '*enter*' key.

```
program "Example Program/0"
{
    printf ("Calculate factorial numbers from 1 to 10");
    f = 1;
    for (i = 1; i<=10; i++) {
        f *= i;
        printf ("Factorial ",i," = ",f);
    }
}
```

If the program loads successfully, Example Program should now be the last item under the Program menu.

To run the program, select Program → Example Program.

17 Manipulating Graph Data and Display Parameters

17.1 Access a Data Trace

AxoGraph incorporates several languages (Basic, Pascal, Fortran and C). In addition to the standard features of these languages, AxoGraph implements many language independent commands and functions which can be used to access and manipulate data.

Data is organized in columns, as in a spread sheet. A column is a list of numbers and is equivalent to a one dimensional array. A data column can be passed to data array in one of the programming languages. One or more columns (typically just one) contains x-axis data values. The remaining columns contain y-axis data values. Each trace on the screen represents a y-axis data column plotted against an x-axis data column.

Xdata (w, t) is an x-axis data column
Ydata (w, t) is a y-axis data column or trace

The parameters are
w = window number (shown in the Window menu)
t = trace number (shown in the Group dialog under the Trace menu)

The parameters (w and t) can be numbers, variables or expressions. Floating point values are rounded to the nearest integer.
If ‘w’ is set to zero, the front window is assumed.
If ‘t’ is set to zero, the front trace is assumed.
If ‘t’ is omitted from xData (w), the default xData column is assumed.

Traces can be combined into groups. For episodic digitized data (e.g. Clampex, or AxoData), traces are automatically grouped by A-D channel number. Individual traces can be accessed by group number and trace number within the group.

Edata (w, g, e) is a trace or episode of data

g = group number (or A-D channel number for episodic data)
e = trace number within group (or episode number)

Some episodic digitized data files (e.g. AxoData) have a hierarchical structure and episodes can be organized into ‘runs’. Individual traces can be accessed by A-D channel number, run number and episode number.

REdata (w, g, r, e) is an episode of data (Multi-run digitized file)

r = run number for a multi-run data file

Xdata, Ydata, Edata and REdata act like ordinary array variables, and can be used to manipulate graph data. For example :

```
Ydata (2, 1) = Ydata (2, 1) * 2
```

This command multiplies the 1st trace of the 2nd window by 2.

Passing an array to a non-existent trace will cause a new trace to be appended to a graph. For example :

```
Ydata (1, 999) = Ydata (1, 2) * 2
```

This command multiplies the 2nd trace of the 1st window by 2 and graphs the result in a new trace ('999' is an arbitrary large number). However, this will not work with digitized data files.

The trace commands can subtract two columns or episodes of data. Data from more than one window can be manipulated. For example :

```
Edata(2, 1, 1) = Edata (2, 1, 1) - Edata (3, 1, 1)
```

This command subtracts the 1st episode in the 3rd window from the 2nd episode in the 2nd window.

Data can be passed to a temporary array variable created on the fly. For example :

```
bufferArr = Ydata (2, 1)
```

This command copies the data column from the 1st trace of the 2nd window, and passes it to the array variable 'bufferArr'. Modifying the array variable does not affect the original data. For example :

```
bufferArr = bufferArr * 2
```

17.2 Graphs with Multiple X-Axis Data Columns

Typically a graph has only a single x-axis data column. However, some graphs will have different x-axis data columns for different traces (for example two traces with different sample intervals).

Two numbered lists of data columns are maintained for each graph: the list of X data columns, and the list of all data columns (including X data columns).

The following commands apply to the list of X data columns. The parameter 'xc' refers to the position of a data column in that list.

NXColumns (w)	is the number of X data columns
XColumn (w, xc)	is the XCth data column in the X data column list
GetTraceXColumn (w, t, xc)	returns the list position (xc) of the X column assigned to the specified trace
SetTraceXColumn (w, t, xc)	assign the specified X data column (xc) to the specified trace

The following commands apply to the list of all data columns. The parameter ‘c’ refers to the position of a data column in that list.

NColumns (w)	is the total number of data columns
Column (w, c)	is the Cth data column
GetColumnTitle (w, c, cStr)	returns the column's title string
GetXColumn (w, t, c)	returns the number of the data column assigned to the X axis of the specified trace
SetXColumn (w, t, c)	assigns the Cth data column to the X axis of the specified trace
GetYColumn (w, t, c)	returns the number of the data column assigned to the Y axis of the specified trace
SetYColumn (w, t, c)	assigns the Cth data column to the Y axis of the specified trace
GetErrColumn (w, t, c)	returns the number of the data column assigned to the error bars of the specified trace
SetErrColumn (w, t, c)	assigns the Cth data column to the error bars of the specified trace

17.3 Access a Section of a Data Trace

```
Xrange (w, t, minRange, maxRange)
Yrange (w, t, minRange, maxRange)
Erange (w, g, e, minRange, maxRange)
RErange (w, g, r, e, minRange, maxRange)
```

where minRange and maxRange define a range of x-axis values in the currently visible units.

Individual data points should generally be manipulated by loading a trace into a temporary array, then accessing elements of the array. For example the following short Pascal program calculates the square root of each non-negative data point in the 1st trace of the 2nd window :

```
..............................................................................
localLanguage Pascal

{• Square root example •}
{• Pass the trace to a temporary work array •}
tempArr = Ydata (2,1)
{• Process each element of the array •}
FOR i = 1 to ArraySize(tempArr) do
begin
    if tempArr[i] > 0 then
    tempArr[i] = sqrt(tempArr[i])
    else
    tempArr[i] = 0
end
{• Pass the array back to the trace •}
Ydata (2,1) = tempArr
{• Throw away the work array •}
Unload (tempArr)
...........................................................................
```

Access a Single Point on a Data Trace

Xpoint (w, xLocation)
Ypoint (w, t, xLocation)
Epoint (w, g, e, xLocation)
REpoint (w, g, r, e, xLocation)

where xLocation defines the x-axis location in user units.
The data point nearest to xLocation is accessed.

NOTE: When more than one data point per trace is to be processed, do not use Xpoint, Ypoint, etc. Instead use Xdata, Ydata, etc. to create a temporary working array, then process elements of the array. For a worked example, see the square root example in the preceding section 'Access a Section of a Data Trace'.

17.4 Graph File Management

Additional commands are described in 'File Management Functions' (access this document via Program → Programming Help).

NewGraph (fileName)   Create a new graph file. Returns the window number.

OpenGraph (fileName)   Open the graph file with the specified name. It must be in the current folder.
If no fileName parameter is given, a standard file dialog is presented.
'OpenGraph' returns the window number of the newly opened file.
If the file is already open, its window number is returned.
If the file is not found, return -1.

Save (window)   Save the contents of the specified window to disk.

Close (window)   Close the specified window.
If the content have changed the user will be asked whether to save the changes.

17.5 Get a Location, Range or Point via a Dialog

The variable 'aStr' passes a message or prompt that is added to the dialog.

GetLocation (w, aStr, xLocation)   Pose a dialog and returns the user-selected x-axis location.

GetRange (w, aStr, xMin, xMax)   Pose a dialog and returns the user-selected x-axis range.

GetPoint (w, aStr, t, xPoint, yPoint)   Pose a dialog and returns the user-selected (x,y) point and the selected trace, t.

Baseline (w, t, xMin, xMax)   Subtract a baseline calculated over
the region xMin to xMax.

Note: The window (w) is brought to the front if necessary.

17.6 Set the Axis Title and Range

Xaxis (w)	is an x-axis title and displayed range
Yaxis (w,t)	is a y-axis title and displayed range

These commands are used in a similar fashion to Xdata and Ydata. For example, the following 2 lines copy the X and Y axes from an old window to a new window :

```
Xaxis(newWindow) = Xaxis(oldWindow)
Yaxis(newWindow,1) = Yaxis(oldWindow,1)
```

The following commands provide more detailed control over axis appearance. (‘xStr’ and ‘yStr’ are string variables.)

GetXTitle (w, xStr)	copy the x-axis title
GetYTitle (w, t, yStr)	copy the trace's y-axis title
GetGroupTitle (w, g, yStr)	copy the group's y-axis title
SetXTitle (w, xStr)	set the x-axis title
SetYTitle (w, t, yStr)	set the trace's y-axis title
SetGroupTitle (w, g, yStr)	set the group's y-axis title
GetXUnits (w, xStr)	copy the x-axis units
GetYUnits (w, t, yStr)	copy the trace's y-axis units
GetGroupUnits (w, g, yStr)	copy the group's y-axis units
SetXUnits (w, xStr)	set the x-axis units
SetYUnits (w, t, yStr)	set the trace's y-axis units
SetGroupUnits (w, g, yStr)	set the group's y-axis units
GetXRange (w, xMin, xMax)	copy the x-axis range
GetYRange (w, t, yMin, yMax)	copy the trace's y-axis range
GetGroupRange (w, g, yMin, yMax)	copy the group's y-axis range
SetXRange (w, xMin, xMax)	set the x-axis range
SetYRange (w, t, yMin, yMax)	set the trace's y-axis range
SetGroupRange (w, g, yMin, yMax)	set the group's y-axis range
Scroll (w, dx)	scroll the x-axis ‘dx’ units to the left
DefaultAxes (w)	set x-axis and y-axis ranges to default values

When ‘Automatic SI Unit Conversion’ is turned on, all the above ‘Get...’ procedures return range values or title units in absolute SI units instead of currently displayed units (for example, ‘V’ instead of ‘mV’). The following procedures are identical to the above, except they return their parameters in the currently displayed units.

DisplayedXTitle (w, xStr)	copy the x-axis title
DisplayedYTitle (w, t, yStr)	copy the trace's y-axis title
DisplayedGroupTitle (w, g, yStr)	copy the group's y-axis title
DisplayedXUnits (w, xStr)	copy the x-axis units
DisplayedYUnits (w, t, yStr)	copy the trace's y-axis units
DisplayedGroupUnits (w, g, yStr)	copy the group's y-axis units
DisplayedXRange (w, xMin, xMax)	copy the x-axis range
DisplayedYRange (w, t, yMin, yMax)	copy the trace's y-axis range
DisplayedGroupRange (w, g, yMin, yMax)	copy the group's range

DisplayedXScale (w, scale)
DisplayedYScale (w, t, scale)
DisplayedGroupScale (w, g, scale)

Returns the x- or y-axis display scale factor. This is the scaling between absolute units (e.g. ‘A’) and displayed units (e.g. ‘µA’ -> scale = 1,000,000).

17.7 Manipulate Graph Windows

NWindows	Returns the number of open windows
WindowTitle (w)	Returns a window's title
Save (w)	Save any changes to a window
Close (w)	Close a window
GetComment (w,aStr)	Get a window's comment
SetComment (w,aStr)	Set a window's comment
GetNotes (w,aStr)	Get a window's notes
SetNotes (w,aStr)	Set a window's notes
AddNotes (w,aStr)	Add aStr to the end of a window's notes
ShowNotes (w,display)	Show a window's notes when ‘display = True’
AddTag (w,tagPoint)	Add a tag to a window at the specified sample point To add a tag at a specified time, tagTime, calculate tagPoint = tagTime / sampleRate
ShowTags (w,display)	Show a window's tags when ‘display = True’
NIntervalBars (w)	Returns the number of interval bars

GetIntervalBar (w,BarNumber,aStr,xMin,xMax,offset)
SetIntervalBar (w,BarNumber,aStr,xMin,xMax,offset)

Get or set the display parameters of an interval bar
‘BarNumber’ is the number of the bar to modify
If the bar does not exist, a new bar is created
‘aStr’ is the bar label
‘xMin’ is the start of the interval
‘xMax’ is the end of the interval
‘offset’ is the y-location of the bar. Should be set in the range 0-1 (bottom to top)

ShowIntervalBars (w, showBars, titlesBelowBars)
"showBars" whether to display interval bars
"titlesBelowBars" whether the titles are displayed above or below the interval bars.

GetFront (w, t)	Returns the front trace and front window number
SetFront (w, t)	Move the specified window and trace to the front
Update (w,t)	Redraw the specified window and trace
NTraces (w)	Returns the number of traces in a window
NGroups (w)	Returns the number of groups in a window
NEpisodes (w)	Returns the number of episodes in a window
YNPoints (w, t)	Returns the number of data points in a trace
NPoints (w)	Returns the maximum number of data points across all traces in a window
deltaX (w)	Returns the x-axis sample interval for a window

GetScreenSize (xSize, ySize)
Returns the size of the main screen in pixels.
The menu bar at the top of the screen is not included.

GetWindowSize (w, xSize, ySize)
SetWindowSize (w, xSize, ySize)
Get or set the size of a window in pixels.

GetWindowLocation (w, xLoc, yLoc)
SetWindowLocation (w, xLoc, yLoc)
Get or set the location of a window in pixels.
The point, (xLoc, yLoc) defines the top left corner of the window.
The y-coordinates increase down the page (a counter-intuitive Macintosh standard).

IsATextWindow (window)
Returns ‘True’ when the specified window is a text file.

IsAGraphWindow (w)
Returns ‘True’ when the specified window is a graph file.

IsADigitizedWindow (w)
Returns ‘True’ when the specified window is a digitized file.

17.8 Manipulate Traces and Groups

Pile (w)	Pile all the visible groups in a window
Tile (w)	Tile all the visible groups in a window
Stack (w)	Stack or Offset all the visible groups in a window

Show (w,t)	Reveal a hidden trace
Hide (w,t)	Hide a visible trace
DeleteTrace (w,t)	Delete a trace
TraceIsShown(w,t)	Returns TRUE if a trace is not hidden
GetGroup (w,t,g)	Returns the group number a trace belongs to
SetGroup (w,t,g)	Assign a trace to a group

NthGroupNumber (w, n)	Returns the number of the nth group in a window
GetFrontOfGroup (w,g,t)	Returns the number of the front trace in the group
GetEpisodes (w,e1,e2)	Return the first and last displayed episodes
ReviewEpisodes (w,e1,e2)	Review the specified episodes
EraseEpisodes (w)	Erase all episodes
EpisodeIsDisplayed (w,e)	‘True’ if the episode is currently displayed
EpisodeIsMasked (w,e)	‘True’ if the episode is masked

SetEpisodeDisplayed (w,e, displayed)
 Displays the episode if ‘displayed=True’
SetEpisodeMasked (w,e,masked)
 Masks the episode if ‘masked=True’

GetYLocation (w,t,loc,size)
SetYLocation (w,t,loc,size)
 Get or set a group's vertical location and size.

GetSymbol (w,t,type,size)
SetSymbol (w,t,type,size)
 Get or set a trace's symbol type and size.
 ‘type’ is a number from 1 to 12 (0 = no symbols)
 ‘size’ sets the symbol diameter in pixels

GetSymSpacing (w,t,space)
SetSymSpacing (w,t,space)
 Get or set the spacing between adjacent symbols.
 If ‘space’ is positive, the separation is
 defined as number of data points. If ‘space’ is negative
 the separation is in number of pixels.

GetLine (w,t,width,dash)
SetLine (w,t,width,dash)
 Get or set the line width and style of dashes.
 ‘width’ is the line width in pixels (0 = no line).
 For a fine line on a laser printer, set ‘width = 0.25’.
 ‘dash’ is the style of dashed line (1 = no dash).

GetHistogram (w,t,type,separation)
SetHistogram (w,t,type,separation)
 Get or set the histogram type and bin separation.
 ‘type’ = 0 no histogram is drawn.
 ‘type’ = 1 indicates a histogram with filled bins.
 ‘type’ = 2 indicates a histogram with open bins.
 ‘type’ = 3 indicates a cityscape histogram.
 ‘separation’ is the separation between adjacent bins
 as a percentage of bin width (0-100).

GetColor (w,t,red,green,blue)
SetColor (w,t,red,green,blue)
 Get or set the trace color.
 ‘red’, ‘green’ and ‘blue’ are set to values from 0 to 1
 and specify the relative intensity of each color.
 If all are set to 0, the trace is black.

GetGraphStyle (w,axisType,axes,yTitle,yTitle90,scaleBars)
SetGraphStyle (w,axisType,axes,yTitle,yTitle90,scaleBars)
Get or set parameters defining the graph's style.
'axisType' = 0 indicates standard axes
'axisType' = 1 indicates zero-crossing axes
'axisType' = 2 indicates log-linear axes
'axisType' = 3 indicates linear-log axes
'axisType' = 4 indicates log-log axes
'axisType' = 5 indicates raster display
'axes' is true indicates axes are displayed
'yTitle' is true indicates y-axis titles are displayed
'yTitle90' is true indicates y-axis titles rotated 90º
'scaleBars' is true indicates scale bars are displayed

GetGroupRaster (w,g,offset,width)
SetGroupRaster (w,g,offset,width)
'offset' is the offset between raster sweeps
'width' is the width of each raster sweep
If 'width' = 0, the full width of a trace is used.
This is the typical setting for episodic data.

17.9 Temporary Analysis Feedback

When creating a custom analysis program it is sometimes useful to display some feedback in a graph window indicating the progress of the analysis. The following functions permit drawing of lines, symbols and text into a window, but these features are temporary and should be erased when analysis is complete.

DrawInitialize	Sets up for drawing to the front window. Must be called first, before other Draw commands.
DrawMove(x,y)	Move current position to (x,y)
DrawLine(x,y)	Draw a line from current position to (x,y) x and y are in absolute axis units (not displayed units).
DrawPixelMove(x,y)	Increment the current position by (x,y) screen pixels
DrawString(aStr)	Display a character string at the current position aStr is a Pascal style string (up to 255 characters)
DrawSymbol(sym,size)	Draw a symbol at the current position
DrawErase	Erase all temporary feedback
DrawClear	Same as DrawErase
DrawSetLine(thick)	Line thickness for subsequent calls to DrawLine
DrawSetDash(dash)	Dash setting for subsequent calls to DrawLine
DrawSetFont(font)	Set text font for subsequent calls to DrawString
DrawSetSize(size)	Set text size for subsequent calls to DrawString
DrawSetStyle(style)	Set text style for subsequent calls to DrawString
DrawSetColor (red,green,blue)	Change the color used by subsequent Draw commands

aStr is a Pascal style string (up to 255 characters)

'red', 'green' and 'blue' are set to values from 0 to 1 and specify the relative intensity of each color. If all are set to 0, the trace is black.

18 Math, String and Array Functions

18.1 Math Functions

abs (x)	Absolute value
sqrt (x)	Square root
sqr (x)	Square
exp (x)	Exponential (e raised to the power x)
ln (x)	Natural Log (to the base e)
exp10 (x)	10 raised to the power x
log10 (x)	Log to the base 10
trunc (x)	Next lowest integer
round (x)	Nearest integer
odd (i)	Returns TRUE if ‘i’ is an odd number

18.2 Trigonometric

pi	Returns the value of pi (≈ 3.1415926)
Sin (x)	Sin
Cos (x)	Cos
Tan (x)	Tan
Sinh (x)	Hyperbolic Sin
Cosh (x)	Hyperbolic Cos
Tanh (x)	Hyperbolic Tan
ArcSin (x)	Inverse Sin
ArcCos (x)	Inverse Cos
ArcTan (x)	Inverse Tan

18.3 String Manipulation

NewString (variableName)

This procedure creates a new 255 character string. It provides an alternative to formally declaring a string variable at the start of a program. String variables can also be created 'on the fly'. For example...

aString = 'abc123'

StringToNum (aStr)	Function that converts ‘aStr’ to a number
NumToString (aNum)	Function that converts ‘aNum’ to a string
PtoCString (aStr)	Function that converts aStr into an C string array
CtoPString (anArr)	Function that converts anArr into a Pascal string
Length (aStr)	Length of string
Pos (subStr, aStr)	Find ‘subStr’ in ‘aStr’ and returns position
Concat (aStr,x,bStr)	Concatenate strings and numeric variables
Copy (aStr, start, len)	Return a sub-string of ‘aStr’ starting from character ‘start’ with length ‘len’.
Delete (aStr, start, len)	Delete ‘len’ characters starting from ‘start’
Insert (iStr, aStr, start)	Insert ‘iStr’ into ‘aStr’ before ‘start’

18.4 Byte and Bit Manipulation

GetBit (aNum, Nth, Bit)	Procedure gets the Nth bit of aNum, and passes it to the variable, Bit
SetBit (aNum, bit, Bit)	Procedure that sets the Nth bit of aNum to the first bit of the variable, Bit
GetByte (aNum, Nth, Byte)	Procedure gets the Nth byte of aNum, and passes it to the variable, Byte
SetByte (aNum, bit, Byte)	Procedure that sets the Nth byte of aNum to the first byte of the varible, Byte
GetWord (aNum, Nth, Word)	Procedure gets the Nth word of aNum, and passes it to the variable, Word
SetWord (aNum, bit, Word)	Procedure that sets the Nth word of aNum to the first word of the varible, Word

18.5 Array Functions

arraySize (arr)	Total number of elements in array
sum (arr, from, to)	Sum of array elements (S)
av (arr, from, to)	Average of array elements
mean (arr, from, to)	Alternative average
stdDev (arr, from, to)	Standard deviation (s)
sd (arr, from, to)	Alternative standard deviation
stdErr (arr, from, to)	Standard error
sem (arr, from, to)	Alternative standard error
variance (arr, from, to)	Variance
min (arr, from, to)	Minimum array element
max (arr, from, to)	Maximum array element

count (arr, from, to)	Number of elements processed
copyArray (arr, from, to)	Return a copy of all or part of the array

Note: The ‘from’ and ‘to’ parameters are optional.
They specify a range of array elements to be processed.
If they are omitted, the entire array is processed.
Array indexing in C starts from zero, but in all other languages it starts from one.

18.6 Advanced Array Functions

• WriteArray (arr, from, to)
Write the contents of the array to the front window

• Stats (arr, from, to)
Calculate the following summary statistics for the specified array and output the results to the front window :
Mean, Variance, S.D., S.E.M., Minimum, Maximum and Count.

Note: the ‘from’ and ‘to’ parameters are optional.
They specify a range of array elements to be processed.
If they are omitted, the entire array is processed.

• FitLine (xArr, yArr, slope, yIntercept, correlation)
Performs a linear regression of ‘xArr’ and ‘yArr’, and returns the result in the parameters ‘slope’, ‘yIntercept’ and ‘correlation’.

• FitExp (dx, yArr, amplitude, timeConstant)
Fits a single exponential to the data in ‘yArr’, and returns the result in the parameters ‘amplitude’ and ‘timeConstant’. The parameter, ‘dx’ is the sample interval.

• FitExpPlus (dx, yArr, amplitude, timeConstant, addedConstant)
Fits a single exponential with added constant to the data in ‘yArr’, and returns the result in the parameters ‘amplitude’, ‘timeConstant’ and ‘addedConstant’.
The parameter, ‘dx’ is the sample interval.

• FitDoubleExp (dx, yArr, a1, tc1, a2, tc2, addedConstant)
Fits a double exponential with added constant to the data in ‘yArr’, and returns the result in the parameters ‘a1’, ‘tc1’, ‘a2’, ‘tc2’, and ‘addedConstant’.
The parameter, ‘dx’ is the sample interval.

• FitMultiExp (dx, yArr, a1, tc1, a2, tc2, a3, tc3, addedConstant)
Fits a triple exponential with added constant to the data in ‘yArr’, and returns the result in the parameters ‘a1’, ‘tc1’, ‘a2’, ‘tc2’, ‘a3’, ‘tc3’ and ‘addedConstant’. The parameters ‘a1’, ‘a2’ and ‘a3’ are the amplitudes of the three exponentials, and the parameters ‘tc1’, ‘tc2’ and ‘tc3’ are the corresponding time constants. The parameter, ‘dx’ is the sample interval.
If any of the four amplitude parameters, ‘a1’, ‘a2’, ‘a3’ and ‘addedConstant’, are set to zero before calling FitMultiExp, they will be constrained to zero during the fit procedure.

• FFT (arr)
Performs a fast Fourier transform on the contents of ‘arr’, and returns the

result in the same array.

• ComplexFFT (realArr, imaginaryArr)
  Performs a complex fast Fourier transform on the contents of ‘realArr’ and ‘imaginaryArr’, and returns the result in the same arrays. Both arrays should be the same size.

Array Manipulation

• NewArray (variableName, arrSize)
  This procedure creates a new array with ‘arrSize’ real elements, and sets all elements to zero. It provides an alternative to formally declaring an array variable at the start of a program.

• SetArraySize (variableName, arrSize)
  This procedure changes the size of an existing array to ‘arrSize’ elements, and does not alter the value of existing elements.

• SetArray (arr, value, from, to)
  A procedure to fill an array with a single numerical value.
  The ‘from’ and ‘to’ parameters are optional.
  For all i in the selected range,
  arr[i] = value

• FillArray (arr, first, increment, from, to)
  A procedure to fill an array with regularly spaced numerical values.
  The ‘from’ and ‘to’ parameters are optional.
  For all i in the selected range
  arr[i] = first + increment * (i-1)

• ConcatArrays (arr1, arr2)
  A procedure for combining two arrays.
  The arrays, arr1 and arr2, are combined into arr1.

• DecimateArray (arr, first, increment, average)
  A procedure to decimate (reduce the number of points) in the array.
  The size of the array is reduced by approximately (1/increment).
  Groups of ‘increment’ points are replaced with either the last point in the group, or with the average of the group depending on the setting of ‘average’. Groups start at the ‘first’ element.
  If average = False then arr[i] is replaced by arr[first+increment*(i-1)]
  If average = True then arr[i] is replaced by the average of the array elements from [first+increment*(i-1)] to [first+(increment*i)-1]

• CopyArray (theArray, from, to)
  Returns a copy of a subrange of theArray. The parameters ‘from’ and ‘to’ specify the subrange to copy. If these parameters are omitted, the entire array is copied.

• CombineArrays (arr1, arr2)
  A function for combining two arrays.
  The arrays, arr1 and arr2, are combined into a new array and returned.

18.7 General Purpose Procedures

Beep	Generates the alert or beep sound
Random	Returns a random number between 0 and 1
Wait (sec)	Halt execution for the specified period in seconds Also ‘Pause’ or ‘Sleep’
CurrentTime	Return the time in seconds since Jan 1st 1904
PauseUntil(sec)	Halt execution until the specified time is reached
GetDate	Return a string containing today's date
GetTime	Return a string containing the time (hours : min : sec)
FreeMemory	Return the currently available memory in bytes. For example, check memory before creating an array (4 bytes per element).
nWindows	Return the number of open windows
ListBuiltIns	List all built in procedures and functions
ListPrograms	List all loaded programs, procedures and functions
ListGlobals	List all global variables, and their current values
GetSigFigs (sf)	Get the default number of significant figures for output
SetSigFigs (sf)	Set the default number of significant figures for output
GetKeyDown	Returns the ASCII code of any key that is currently down, or zero if no keys are down. Useful for interrupting a loop.

• FlowControl (Background, SpinCursor, CheckKeys, TempMem)
Set parameters that control the execution of large programs within AxoGraph. The four boolean parameters are...
Background: AxoGraph can switch to background while a program is running.
SpinCursor: Display a spinning cursor while program is running.
Suppress this for time-critical programs.
CheckKeys: Check for Cmd-period escape and other key strokes
TempMem: Use temporary (system) memory when creating arrays, etc.

• Unload (varName, programName, procedureName, functionName, ...)
Unload all the specified global variables, programs, procedures and functions. This frees up memory space allocated to the arrays and procedures. It also removes the specified program names from the ‘Program’ menu.
This procedure may be useful for freeing up memory associated with large global data structures, but in practice it is rarely used.

18.8 Create Array and String Variables

• NewArray (variableName, arrSize)
This procedure creates a new array with ‘arrSize’ real elements, and sets all elements to zero. It provides an alternative to formally declaring an array variable at the start of a program.

• CopyArray (theArray, from, to)
Returns a copy of a subrange of theArray. The parameters ‘from’ and ‘to’ specify the subrange to copy. If these parameters are omitted, the entire array is copied.

• SetArraySize (variableName, arrSize)
This procedure changes the size of an existing array to ‘arrSize’ elements, and does not alter the values of existing elements.

• NewString (variableName)
This procedure creates a new 255 character string. It provides an alternative to formally declaring a string variable at the start of a program. String variables can also be created on the fly. (aString = ‘abc’;)

18.9 User Dialogs

• Alert ('A string', a, b, c,...)
This procedure converts its parameter list to a line of text (as for Writeln, Printf, etc.), and sends the text to an alert dialog. String, numeric and boolean parameters can be included in the list. Any number of parameters may be present, and they may appear in any order.

• PoseDialog ('Introduction', 'Describe A', a, 'Describe B', b, ...)
This procedure requests parameter values from the user via a standard dialog. The parameter list must have the following format.

- The 1st parameter is a string containing general information.
- The 2nd parameter is a string describing the first requested variable.
- The 3rd parameter is a numeric, boolean or string variable which
  will receive the requested value.

Subsequent parameters are optional, but if present, they must alternate text string, then numeric, boolean or string variable. After executing this procedure, the values entered into the dialog by the user are returned in the variables, ‘a’, ‘b’, ...

• ChoiceDialog ('Introduction', 'Alternative', 'Default', userChoseDefault)
This procedure offers the user an ‘either - or’ choice via a standard dialog. The parameter list must have the following format.

- The 1st parameter is a string containing general information.
- The 2nd parameter is a short string describing the alternative option.
- The 3rd parameter is a short string describing the default option.
- The 4th parameter is a variable that returns ‘True’ if the default
  option is selected, or ‘False’ if the alternative option is selected.

• RadioDialog ('Introduction', 'Describe A', a, 'Describe B', b, ...)
This procedure requests the user to select one item from a list via a standard radio button dialog. The parameter list must have the following format.

- The 1st parameter is a string containing general information.
- The 2nd parameter is a string describing the first requested variable.
- The 3rd parameter is a boolean variable.

Subsequent parameters are optional, but if present, they must alternate text string, then a boolean variable. After executing this procedure, the boolean variable, ‘a’, ‘b’, ... corresponding to the item selected by the user will be set to ‘true’, and all others will be set to ‘false’.

• VariableDialog ('Introduction', 'Describe A', a, display_a, 'Describe B', B, display_b, ...)
This procedure requests parameter values from the user via a standard dialog. Similar to 'PoseDialog', but only a subset of the parameters will be displayed. The parameter list must have the following format.

- The 1st parameter is a string containing general information.
- The 2nd parameter is a string describing the first requested variable.
- The 3rd parameter is a numeric, boolean or string variable which

will receive the requested value.
- The 4th parameter is a boolean expression which determines whether this item will be displayed in the dialog.

Subsequent parameters are optional, but if present, they must repeat the sequence text string, then numeric, boolean or string variable, then boolean expression. After executing this procedure, the values entered into the dialog by the user are returned in the variables, ‘a’, ‘b’, ...

• VariableRadioDialog ('Introduction', 'Describe A', a, display_a, 'Describe B', B, display_b, ...)
This procedure the user to select one item from a list via a standard radio button dialog. Similar to 'RadioDialog', but only a subset of the parameters will be displayed. The parameter list must have the following format.

- The 1st parameter is a string containing general information.
- The 2nd parameter is a string describing the first requested variable.
- The 3rd parameter is a boolean variable.
- The 4th parameter is a boolean expression which determines whether this item will be displayed in the dialog.

Subsequent parameters are optional, but if present, they must repeat the sequence text string, then a boolean variable, then boolean expression. After executing this procedure, the boolean variable, ‘a’, ‘b’, ... corresponding to the item selected by the user will be set to ‘true’, and all others will be set to ‘false’.

• Report ('Report Message')
This procedure opens a temporary report window (modeless dialog) that displays a message in bold-blue text. It remains open until it is closed by calling the procedure 'CloseReport'. It provides a mechanism for informing the user that a procedure will take some time to complete.

• CloseReport
Closes the report window.

18.10 Array Procedures

• WriteArr (arr, from, to)
Write the contents of the array to the front window.

• Stats (arr, from, to)
Calculate the following summary statistics for the specified array and output the results to the front window: Mean, Variance, S.D., S.E.M., Minimum, Maximum and Count.

• FitLine (xArr, yArr, slope, yIntercept, correlation)
Performs a linear regression of ‘xArr’ and ‘yArr’, and returns the result in the parameters ‘slope’, ‘yIntercept’ and ‘correlation’.

• FFT (arr)
Performs a fast Fourier transform on the contents of ‘arr’, and returns the result in the same array.

• ComplexFFT (realArr, imaginaryArr)
Performs a complex fast Fourier transform on the contents of ‘realArr’ and ‘imaginaryArr’, and returns the result in the same arrays.

18.11 Advanced Procedures

• Evaluate (aStr, bStr, ...)
Execute the contents of the string parameter(s), as if the user had selected the string(s) and typed ‘enter’. A string can be built and executed within a program or procedure. It can access variables local to that procedure. The following example uses the ‘Evaluate’ command to perform a user specified analysis on a newly opened file. The name of the analysis procedure is contained in the string ‘AnalysisProcedure’, and the procedure takes a window number as its parameter.

```
{ Open a graph file and analyse it. }
    newWindow = openGraph
    Evaluate (AnalysisProcedure,'(newWindow)')
```

• Minimize [precision] UserFunction (a, b, c, ...)
Find the parameters, ‘a, b, c, ...’ that minimize a user defined function. ‘precision’ is optional and controls the convergence criterion for the simplex optimization algorithm. The function can have up to 25 free parameters. The value of the parameters at the time ‘Minimize’ is called are used as the starting guess for the optimization. This curve fitting approach is much more flexible than ‘Fit General...’ under the ‘Analyse’ menu, but it is slower. The following example uses the ‘Minimize’ command to find the minimum point of a parabola.

```
{ Find the minimum of a parabola }
    Function Parabola (x)
    begin
        Parabola = 5*x*x - 5*x + 5
    end

    xMin = 1      { The starting guess }
    Minimize Parabola (xMin)
    writeln ('\nThe minimum of the parabola was found at x = ',xMin)
```

19 File Management Functions

19.1 Change Folders

ChangeFolder (folderName)
Set the current working folder. If no parameter is given, a standard file dialog is presented.
If the folder name is a string literal, the brackets are optional.
The command ‘cd’ is equivalent to ‘ChangeFolder’.

GetFolder (refNumber)
Get the reference number of the current working folder.

SetFolder (refNumber)
Make the folder with the specified reference number, the current working folder.

19.2 Create and Open Files

Files
List all files in the current folder.
‘dir’ and ‘ls’ are equivalent to ‘Files’.
If followed by ‘/t’ or ‘-t’, only text files are listed.
If followed by '/g' or '-g', only graph files are listed.
If followed by '/d' or '-d', only digitized files are listed.
If followed by '/f' or '-f', only folders are listed.

FileExists (fileName)
Returns true if the named file exists in the current directory.

IsATextFile (fileName)
Returns ‘True’ when the named file exists in the current directory, and is a text file.

IsAGraphFile (fileName)
Returns ‘True’ when the named file exists in the current directory, and is a graph file.

IsADigitizedFile (fileName)
Returns 'True' when the named file exists in the current directory and is a digitized data file.

IsATextWindow (window)
Returns 'True' when the specified window is a text file.

IsAGraphWindow (window)
Returns 'True' when the specified window is a graph file.

IsADigitizedWindow (window)
Returns 'True' when the specified window is a digitized file.

NewText (fileName)
NewGraph (fileName)
Create a new file. Returns the window number.
If no fileName parameter is supplied, a default name is used.

OpenText (fileName)
OpenGraph (fileName)
OpenDigitized (fileName)
Open the file with the specified name and type. It must be in the current folder.
If no fileName parameter is given, a standard file dialog is presented.
‘Open.....’ returns the window number of the newly opened file.
If the file is already open, its window number is returned.
If the file is not found, return -1.
If the file is not of the specified kind, return -2.

IndexOpenText (index, afterTime)
IndexOpenGraph (index, afterTime)
IndexOpenDigitized (index, afterTime)
Open a file by index Position in the current folder's file list.
Only open if it is was modified later than ‘afterTime’.
The optional ‘afterTime’ parameter is given in seconds since Jan 1st 1904 (Mac standard).
A reference time can be obtained using the ‘GetTime’ function.
‘IndexOpen...’ returns the window number of the newly opened file.
If the file is not found, return -1.
If the file is not of the specified kind, return -2.
If the file was created before ‘afterTime’, return -3.

OpenAllText (afterTime)
OpenAllGraphs (afterTime)
OpenAllDigitized (afterTime)
Opens all text, graph or digitized files in the current folder modified since ‘afterTime’.
The time parameter is in seconds since Jan 1st 1904 (Mac standard).
A reference time can be obtained using the ‘GetTime’ function.
If no time parameter is supplied, then all files in the current folder are opened.
Use the ‘nWindow’ function before and after ‘OpenAll....’ to check whether any new windows were opened.

SetGraphType (window, theType)
Set the graph file format for a subsequent save. The format is specified by the value of ‘theType’ as follows :

AxoGraph	when theType = 1
KaleidaGraph	when theType = 2
CricketGraph	when theType = 3
Tab-Text	when theType = 4

SetGraphCreator (window, theType)
Set the creator for a subsequent graph file save. The creator is specified by the value of ‘theType’ as follows :

AxoGraph	when theType = 1
KaleidaGraph	when theType = 2
CricketGraph	when theType = 3
Tab-Text	when theType = 4

Save (window)
Save the contents of the specified window to disk.

SaveAs (window, fileName)
Save the contents of the specified window to a new file.

SaveViaDialog (window, fileName)
Save the contents of the specified window to a new file.
Always present a standard save-file dialog.

SaveACopy (window, fileName)
Make a copy of the specified window and save to a new file.

Close (window)
Close the specified window.
If the content have changed the user will be asked whether to save the changes.

ForceClose (window)
Close the specified window. If the content have changed, the changes will be discarded.

DeleteFile (fileName)
Delete the specified file from the current folder.

Lock (fileName)
Lock the specified file.
It can not be modified or deleted when locked.

Unlock (fileName)
Unlock the specified file.

nWindows   Returns the number of open windows.

19.3 Edit Text

Select (window, start, end)
Select a range of text in the specified window extending
from the character ‘start’ to the character ‘end’.

GetSelection (window, start, end)
Returns the current text selection range in the specified
window. The range extends from the character ‘start’ to
the character ‘end’.

ShowSelection (window)
Scroll the selected text into view in the specified window.

ClearText (window)
Clear the selected text in the specified window.

CutText (window)
Cut the selected text from the specified window.

CopyText (window)
Copy the selected text from the specified window.

PasteText (window)
Paste any text on the clipboard into the specified window.

19.4 Read and Write Text Files

ReadFrom (window, offset)
The next Read or ReadLn will start from the specified window and character offset. If the offset parameter is omitted, the end of the current selection is used.

GetReadFrom (window, offset)
Returns the current window and character offset where the next Read or ReadLn will start from. The offset parameter is optional.

ReadLn (aStr) Read a line from a text file into the string 'aStr'
The next Read or ReadLn will begin at the start or the next line.

Read (aVar, bVar, ...)
Read numeric values from a text file into the variables 'aVar', 'bVar', ...
The next Read or ReadLn will begin at the start or the next line. Numeric values should be comma, tab or space delimited. Non-numeric text in the input line will be skipped automatically. If there are too few values to satisfy the parameter list, an error will result.

WriteTo (window, offset)
The next program output (e.g. from a WriteLn) will be directed to the specified window and character offset. The output buffer of the current window is flushed. If the offset parameter is omitted, the end of the current selection is used.

GetWriteTo (window, offset)
Returns the current window and character offset where the next output will be directed too. The offset parameter is optional.

EndOfFile (window)
Returns 'True' when the current selection (offset) is at the end of the text in the specified window.

TextLength (window)
Returns the total number of characters (bytes) in the text in the specified window.

Write ('A string', a, b, c,...)
WriteLn ('A string', a, b, c,...)
Print ('A string', a, b, c,...)
Printf ('A string', a, b, c,...)

These procedures convert their parameter list to a line of text and send it to the front window. ‘Write’ does not add a ‘return’ at the end of the output line. ‘WriteLn’, ‘Print’ and ‘Printf’ are interchangeable. Valid parameters are numerical, logical or string variables and expressions. Any number of parameters may be specified (including none), and they may appear in any order.

Note: Pascal and Fortran use single quotes (') to enclose character strings, but Basic and C use double quotes (‘). To include a quote, tab or new line in a character string, use \' or \’ for a quote, \t for a tab and \r or \n for a new line. As an alternative, the built in strings ‘tab’ and ‘newline’ can be added to the parameter list of an output statement.

FlushOutput

Text output from Write, WriteLn and Print is normally accumulated in a buffer and is output all at once when a program finishes running. Use FlushOutput to force the output to occur while the program is still running.

19.5 Read and Write Binary Data Files

Binary read and write routines can only be used with a file that already exists. These routines were designed for importing foreign data files into AxoGraph. See ‘Import Modules’ folder for examples of their use.

• CreateBinary (fileName, fileCreator, fileType)

This procedure creates a new, empty binary file with the specified name, creator and type. The three parameters are all string variables. Only the first 4 characters of the ‘fileCreator’ and ‘fileType’ parameters are used.

• OpenBinary (fileName)

This function opens the file with the specified name for binary input (read only). It returns a file ID number that is used by the binary read and close routines.
If the file is not found, return -1.

• OpenBinaryWrite (fileName)

This function opens the file with the specified name for binary input or output. It returns a file ID number that is used by the binary read, write and close routines.
If the file is not found, return -1.

• IndexOpenBinary (index)

Open a binary file by index position in the current folder's file list. ‘IndexOpen...’ returns a file ID number that is used by the binary read, write and close routines.

If the file is not found, return -1.

• CloseBinary (importFileID);
This procedure closes the specified binary data file.

• BinaryTitle (importFileID);
This function returns the name (or title)
of the specified binary file.

• BinaryLength (importFileID);
This function returns the length in bytes
of the specified binary file.

• ReadBinary (importFileID, offset, length, kDataType, dataArray)
• WriteBinary (importFileID, offset, length, kDataType, dataArray)
These procedures read or write binary data from the specified file. The operation starts at ‘offset’ bytes from the start of the file and continues for ‘length’ bytes. The ‘offset’ parameter is automatically incremented by ‘length’ bytes. Data is read into or written from ‘dataArray’ after optional processing. The processing is specified by ‘kDataType’ which can take the following values...

Integer, int:	1
Longint, long:	2
Real, float:	3
IBM Integer:	4
IBM Longint:	5
IBM Real:	6
Pascal String:	7
C String:	8
Double:	9
Extended:	10

The Integer types are 2 bytes per data value
The Longint and Real types are 4 bytes per data value
Double is 8 bytes per floating point data value
Extended is 10 bytes per floating point data value
IBM types have their byte order reversed as they are transferred
Pascal Strings begin with a length byte
C Strings terminate with a null byte

20 Technical Information

20.1 AxoGraph File Format

There are two different AxoGraph file formats, one for graph data and one for digitized data. Versions of AxoGraph prior to 4.1 only support the graph file format. The digitized file format was introduced in version 4.1 to store acquired data more efficiently. The formats of both files are very simple. Graph style information (line thickness, symbols, color, trace grouping, etc.) is stored in the 'resource fork' of the file. Graph and digitized files do not require a style resource, and files that lack a resource fork will be opened with the default style settings. The file format for graph style information is not documented.

Graph File Format

Header

Byte	Type	Contents
0	OSType	AxoGraph file header identifier = 'AxGr'
4	Integer	AxoGraph file format version number = 1
6	Integer	Number of columns to follow

Each column

Byte	Type	Contents
0	Longint	Number of points in the column (columnPoints)
4	String[79]	Column title - units should be in brackets 'Current (pA)'
84	Real*4	1st Data point
88	Real*4	2nd Data point
..	..	....
..	..	etc.

Note:
'OSType' is a 4 byte character identifier.
Pascal 'Integer' type is equivalent to C 'short' or 2 byte integer.
Pascal 'Longint' type is equivalent to C 'long' or 4 byte long integer.
Pascal 'Real*4' type is equivalent to C 'float' or 5 byte floating point.
Pascal 'String' type has no direct C equivalent. It is a character string with the string length stored in the first byte.

Digitized File Format (supported by version 4.1 and later)

Header

Byte	Type	Contents
0	OSType	AxoGraph file header identifier = 'AxGr'
4	Integer	AxoGraph file format version number = 2
6	Integer	Number of columns to follow

First column (X data)

Byte	Type	Contents
0	Longint	Number of points in the column (columnPoints)
4	String[79]	Column title - units should be in brackets 'Current (pA)'
84	Real*4	Sample Interval

Each subsequent column

Byte	Type	Contents
0	Longint	Number of points in the column (columnPoints)
4	String[79]	Column title - units should be in brackets 'Current (pA)'
84	Real*4	Scaling Factor
88	Integer*2	1st Data point
90	Integer*2	2nd Data point
..	...	....
..	...	etc.

20.2 Floating Point Data

AxoGraph uses three different formats to store floating point data depending on the context. The formats are...

1) IEEE 4 byte / 32 bit format (‘real’ in Pascal and ‘float’ in C)
2) IEEE 8 byte / 64 bit format (‘double’ in Pascal and C).
3) IEEE 10 byte / 80 bit format (‘extended’ in Pascal and C).

AxoGraph stores all graph data points using the 32 bit floating point format on both PPC and 68K machines. The 32 bit format contains 24 bits of precision. Thus, a 32 bit floating point number can be used to average approximately 4,000 integer data points acquired by a 12 bit A-D converter, without loss of information.

All intermediate values generated during arithmetic calculations and data manipulations are performed using a higher precision format. On 68K machines, the 80 bit ‘extended’ floating point format is used, and on PPC machines, the 64 bit ‘double’ format is used. All of AxoGraph's floating point parameters, and all floating point variables in its built-in languages are stored and processed internally using the higher precision floating point format. Elements of array variables are stored in 32 bit format. Higher precision results are rounded into the 32 bit format when they are passed into an array or a graph.

20.3 Formulas and Algorithms

Most of the formulas and algorithms used by AxoGraph for analysing data are given below. One exception is the fast Fourier transform (FFT) algorithm used to calculate the power spectrum. This is a standard algorithm that can be found in many numerical recipe books, and is too long to be included here.

Standard Deviation and Standard Error

Standard deviation (SD) and standard error of the mean (SEM) are calculated by several different analysis routines. The following formulas are used.

For a list of N data points (Yi , i = 1 to N), let,

```
SY = Sum over i ( Yi )
SYY = Sum over i ( Yi * Yi )

SD = Square Root ( (SYY - (SY * SY / N)) / (N-1) )

SEM = Square Root ( (SYY - (SY * SY / N)) / (N * (N-1)) )
```

Digital Filter

The digital filter uses a Gaussian filter algorithm to smooth data traces. It numerically convolves a Gaussian envelope with the data trace. The user selects a Filter cutoff or a Filter width in the Filter dialog box, and the selected parameter is used to calculate the standard deviation of the Gaussian envelope (Sigma) as follows,

Sigma = Filter width / 4

or,

Sigma = 0.132505 / Filter cutoff

Sigma is then converted from real units to number of data points (N) by dividing by the Sample Interval (assumed to be constant),

N = Sigma / Sample Interval

A set of N Gaussian filter coefficients (Fi) is calculated from Sigma as follows (Note: '*' is multiplication and '**' is exponentiation),

```
If Sigma < 0.62 then,
  N = 3
    b = Sigma * Sigma / 2
    F1 = b
    F2 = 1 - 2 * b
    F3 = b

If Sigma ≥ 0.62 then,
    N = 2 * Round (4 * Sigma) + 1
    b = -1 / ( 2 * Sigma ** 2)
    i = (N / 2) + 1
    Fj = exp (b * (j - i) ** 2)   for j = 1 to N
```

The data is then smoothed by sliding the Gaussian envelope, represented by the filter coefficients, along the data trace. Each data point (Yk) is replaced by a filtered data point (Y'k) as follows, Y'k = Sum over j (Fj * Yk+j-i)

Ensemble Average
The ensemble average of a selected list of episodes in a digitized data file is calculated as follows. Let Yij be the ith point of the jth episode. Assume the ensemble average is calculated over all N episodes in the file. Let,
SYi = Sum over j (Yij)
Now, the ith point of the ensemble average, Ai, is
Ai = SYi / N

Ensemble Variance
The ensemble variance of a selected list of episodes in a digitized data file is calculated as follows. In addition to the symbols defined above, let,
SYYi = Sum over j (Yij * Yij)
Now, the ith point of the ensemble variance, Vi, is
Vi = (SYYi - (SYi * SYi / N)) / N

Residual Variance
The ‘residual’ variance of a selected list of episodes in a digitized data file is calculated as follows. First, the ensemble average of the N episodes (Ai) is calculated (see above). Then, the ensemble average is optimally scaled to fit each episode. The optimal scale factor for the jth episode (Cj) is calculated as follows,
SYAj = Sum over i (Yij * Ai)
SAAj = Sum over i (Ai * Ai)
Cj = SYAj / SAAj

The optimally scaled average is subtracted from each trace,
Y'ij = Yij - Cj * Ai

The residual variance (V'i) is the ensemble variance of the N subtracted episodes,
SY'i = Sum over j (Y'ij)
SYY'i = Sum over j (Y'ij * Y'ij)
V'i = (SYY'i - (SY'i * SY'i / N)) / N

Trial to Trial Variance
The ‘trial to trial’ variance of selected episodes in a digitized data file is calculated as follows. The first and last episodes in the selected group are not processed. For each of the remaining episodes, the average of the previous and the following episodes is subtracted,
Y’ij = Yij - (Yi(j-1) + Yi(j+1)) / 2   for j = 2 to N-1

The trial to trial variance (V’i) is the ensemble variance of the N-2 subtracted episodes,
SY’i = Sum over j (Y’ij)   for j = 2 to N-1
SYY’i = Sum over j (Y’ij * Y’ij)   for j = 2 to N-1
V’i = (SYY’i - (SY’i * SY’i / (N-2))) / (N-2)

Linear Fit
Linear regression is a standard procedure described in many numerical recipe books. The slope (A) and y-axis intercept (B) of the fitted line are calculated as follows. Let (Xi, Yi) be the ith of N data points.

```
   SX = Sum over i ( Xi )
   SY = Sum over i ( Yi )
   SXY = Sum over i ( Xi * Yi )
   SXX = Sum over i ( Xi * Xi )

   A = ( SXY - (SX * SY) / N ) / ( SXX - (SX * SX) / N )
   B = ( SY - A * SX ) / N
```

The correlation coefficient (CC) of the fitted line is calculated as follows.
Let Fi be the ith point of the fitted line,

```
   Fi = A * Xi + B

   SY = Sum over i ( Yi )
   SF = Sum over i ( Fi )
   SYY = Sum over i ( Yi * Yi )
   SFY = Sum over i ( Fi * Yi )
   SFF = Sum over i ( Fi * Fi )
   Co = Square Root ( (SYY - SY * SY / N) * (SFF - SF * SF / N) )

     CC = (SFY - (SF * SY) / N) / Co
```

Auto- and Cross Correlation
The cross correlation between two traces is calculated as follows.
Let Yi be the ith point of one trace and Zi be the ith point of the second trace. Both traces contain N data points. Let,

```
   SY = Sum over i ( Yi )
   SYY = Sum over i ( Yi * Yi )
   SZ = Sum over i ( Zi )
   SZZ = Sum over i ( Zi * Zi )

   Co = Square Root ( (SYY - SY * SY / N) * (SZZ - SZ * SZ / N) )
```

Let CCi be the ith point of the cross correlation trace. It has 2 N - 1 points. To calculate the ith point first define,

```
   ko =  j - N
   M = N - Abs( ko )    where Abs() is the absolute value function.
   ka = ko
   kb = -ko

   if ka < 0 then ka = 0
   if kb < 0 then kb = 0

     SYi = Sum over k ( Yka+k )  for k = 1 to M
     SZi = Sum over k ( Zkb+k )  for k = 1 to M
     SYZi = Sum over k ( Yka+k * Zkb+k )  for k = 1 to M

     CCi = ( SYZi - (SYi * SZi) / M ) / Co
```

The autocorrelation of a single trace is calculated using the same algorithm as above, but with Yi substituted for Zi throughout.

20.4 Simplex Optimization Algorithm

AxoGraph uses a simplex algorithm to fit general equations, and to fit exponentials when the faster Chebyshev algorithm is not selected. It finds the combination of parameters that give the best fit between the fitted function and the data. This is usually done by minimizing the sum of squared errors (SSE) between the data and the fitted function. The simplex method is a standard algorithm that can be found in many numerical recipe books. It should not be confused with another type of simplex algorithm used in constrained optimization problems (e.g. airline bookings). The following description of the simplex algorithm used in AxoGraph is adapted from Chapter 2.6 of 'Methods for Unconstrained Optimization' by J. Kowalic and M.R. Osborne, Elsevier, New York (2068).

A simplex is a set of n+1 points in n dimensional space. In the case n = 2, the corresponding figure is a triangle, and in the case n = 3, it is a tetrahedron. The optimization search proceeds as a series of spatial operations on the simplex that tend to move it towards the function minimum. The three basic operations used in this method are reflection, expansion and contraction which are defined below.

First define three important vertices of the simplex :

Xmax is the vertex where
f(Xmax) = Maximum f(Xi) for i = 1 to n+1

X2nd is the vertex where
f(X2nd) = Maximum f(Xi) for i = 1 to n+1, i <> max

Xmin is the vertex where
f(Xmin) = Minimum f(Xi) for i = 1 to n+1

also define the centroid of all vertices except Xmax,
Xo = (1/n) Sum over i (Xi) for i = 1 to n+1, i <> max

The three operations used in the simplex method are defined as follows,

Reflection, the maximum vertex is reflected in the centroid of the remaining vertices, and Xmax is replaced by
Xreflect = 2 Xo - Xmax

Expansion, Xreflect is expanded in the direction along which a further improvement of the function value is expected. The distance from Xo to Xreflect is doubled and Xreflect is replaced by,
Xexpand = 2 Xreflect - Xo

Contraction, is used when reflection fails. The maximum vertex is replaced be the point half way between itself and the centroid, so Xmax is replaced by
Xcontract = 0.5 Xmax + 0.5 Xo

The search procedure can be viewed as the reflecting, expanding and contracting motion of the simplex toward the minimum. This motion is accomplished in the following way :

1) An initial simplex is formed (the vertices are guessed), and the function is evaluated at each of the vertices.

2) The points Xmax, X2nd, Xmin are determined and the centroid, Xo, is calculated. The reflection operation is performed, and the function is evaluated at Xreflect. Depending on the value of f(Xreflect), one of the steps 3, 4 or 5 is performed.

3) If f(Xreflect) < f(Xmin), the reflection was successful, so the simplex is expanded in the new direction and the function evaluated at Xexpand. If f(Xexpand) < f(Xmin), the expansion was successful and Xmax is replaced by Xexpand, otherwise Xmax is replaced by Xreflect. In either case the search procedure returns to step 2.

4) If f(X2nd) ≥ f(Xreflect) ≥ f(Xmin), the reflection was a partial success, so Xmax is replaced by Xreflect and the procedure returns to step 2.

5) If f(Xmax) > f(Xreflect) > f(X2nd), the reflection was a partial failure, so Xmax is replaced by Xreflect and contraction is tried and the function evaluated at Xcontract. But, if f(Xreflect) ≥ f(Xmax), then reflection was a complete failure, and contraction is tried without first replacing Xmax.
If f(Xcontract) < f(Xmax), the contraction was successful and Xmax is replaced by Xcontract and the procedure returns to step 2.
But if f(Xcontract) ≥ f(Xmax), the contraction was a failure and the last simplex is shrunk around the minimum vertex as follows :
Xi = 0.5 (Xi + Xmin) for i = 1 to n+1
The function is evaluated at each of the new vertices and the procedure returns to step 2.

Precision
The search continues until the convergence criterion is satisfied. This occurs when an estimate of the relative error for each search parameter drops below the specified ‘Precision’ value. This value is expressed as a percentage in the Curve Fit dialog. If there are N search parameters then the relative error is checked following every 10 x N calls to the function being optimized. The relative error (Err) in each parameter is estimated as,

Err = 10 x change in parameter since last check / initial step size

The ‘initial step size’ for a parameter is typically 10% of the starting guess for that parameter, but will be a small arbitrary value if the starting guess was zero.

If Err is set to zero, an alternative convergence criterion is used. Empirical observations suggest that contraction is unsuccessful only when the search has almost converged to the limits of computational accuracy. A tally is kept of the number of unsuccessful contractions, and when the tally reaches 8 the simplex search converges.

20.5 Chebyshev Algorithm

AxoGraph incorporates a very fast exponential fitting algorithm based on Chebyshev polynomials. This algorithm was written by George Malachowski of Melbourne University. The general fitting routine uses the discrete Chebyshev transform and the Chebyshev recursion algorithm to fit discrete data with a set of harmonics or exponentials.

The method attempts to fit the functions:

f(j) = a0 + a1*exp(-t/tau1) + a2*exp(-t/tau2)...

The fit transforms the input time series data into Chebyshev polynomial coefficients via the summation d[k] = Sum over j (f(j)*Tk(j))/Rk

where Rk is a normalization factor given by:

Rk = Sum over j (Tk(j)*Tk(j))

where the sum is performed from 0 to N. N+1 is the number of data points and Tk(j) is the kth order discrete Chebyshev polynomial.

The method exploits the fact that the Chebyshev transform linearizes the problem of fitting exponentials. If the input data is an exact exponential, then the resulting Chebyshev coefficients have the following interrelationship:

d[j] = k * ((N+j+2) * d[j+1] / (2*j+3) - d[j] - (N-j+1) * d[j-1]/(2*j-1))

k = (1 - exp(-1/tau))/(1 + exp(1/tau))

This expression holds for all the d[j]. Thus k can be directly calculated by regressing over all j.

The Two Exponential Case

The case of multiple exponentials is a more complex. Let

f(t) = a1*exp(-t/tau1) + b*(exp(-t/tau2)

Transform the data via the Chebyshev transform to obtain d[j]:

d[k] = Sum over j (f(j)*Tk(j))/Rk

Define:

db[j] = (N+j+2) * d[j+1]/(2*j+3) - d[j] - (N-j+1) * d[j-1]/(2*j-1))

If db[j] is calculated from the d[j], and f(t) is an exact sum of exponentials, then:

d[j] = a1*d_tau1[j] + a2*d_tau2[j]

where d_tau1[j] is the Chebyshev transform of a single exponential exp(-t/tau1) and d_tau2[j] is the transform of exp(-t/tau2). Calculating db[j] yields:

db[j] = a1*db_tau1[j] + a2*db_tau2[j]

However each db_tau1[j] is related to the d_tau1[j] via:

d_tau1[j] = k1*db_tau1[j]

where

k1/(1+ k1) = (1 - exp(-1/tau1))/(1 + exp(1/tau1))

Thus

db[j] = a1*d_tau1[j]/k1 + a2*d_tau2[j]/k2

Performing the operation:

dbb[j] = (N+j+2)*db[j+1]/(2*j+3) - db[j] - (N-j+1)*db[j-1]/(2*j-1))

and employing the same identity:

dbb[j] = a1*d_tau1[j]/k1^2 + a2*d_tau2[j]/k2^2

There are now three equations:

d[j]   = a1*d_tau1[j] + a2*d_tau2[j]

db[j] = a1*d_tau1[j]/k1 + a2*d_tau2[j]/k2

dbb[j] = a1*d_tau1[j]/k1^2 + a2*d_tau2[j]/k2^2

The left hand side coefficients are calculated from the d[j]. Now there must exist a pair of parameters x1 and x2 such that:

d[j] + x1*db[j] + x2*dbb[j]

= a1*d_tau1[j]*(1+x1/k1+x2/k1^2) + a2_d_tau2[j]*(1+x1/k2+x2/k2^2)

= zero for all j. The only way this is possible is if:

x1 = -(k1 + k2) and x2 = k1*k2.

Substituting x1 and x2 in the expression:

1 + x1/k1 + x2/k1^2 gives:

1 -(k1 + k2)/k1 + k1*k2/k1^2)

-> 1 - 1 - k2/k1 +k2/k1

-> 0

Thus if x1 = -(k1 + k2) and x2 = k1*k2:

d[j] + x1*db[j] + x2*dbb[j] = 0

for all j.

In general where f(t) is not an exact sum of exponentials, the equation

d[j] + x1*db[j] + x2*dbb[j]

is regressed against x1 and x2 to yield the linear equations in x1 and x2:

Sum(d[j]*db[j]) = -x1*Sum(db[j]*db[j]) - x2*Sum(dbb[j]*db[j])

Sum(d[j]*dbb[j]) = -x1*Sum(dbb[j]*db[j]) - x2*Sum(dbb[j]*dbb[j])

These equations can be solved and the solution of the polynomial:

```
  k^2 + x1*k + x2 = 0
```

Gives solutions k1 and k2 which are related to the time constants tau1 and tau2 via:

```
  k1/(1 + k1) = (1 - exp(-1/tau1))/(1 + exp(1/tau1))

      k2/(1 + k2) = (1 - exp(-1/tau2))/(1 + exp(1/tau2))
```

Calculate the Exponential Amplitudes

The amplitudes are obtained by regressing the original coefficients d[j] against coefficients that are the Chebyshev transform of the exponentials exp(-t/tau1) and exp(-t/tau2). These coefficients are obtained from a method called inward recursion which is detailed in Abramovitz and Stegun. Basically, the method relies on the fact that the transform of an exponential satisfies the expression:

```
  d[j] = k/(1 + k)*((N+j+2)*d[j+1] / (2*j+3) - d[j] - (N-j+1)*d[j-1]/(2*j-1))
```

where k/(1 + k) = (1 - exp(-1/tau))/(1 + exp(-1/tau))

The equation can be reorganised by taking the d[j] on the right over to the left side and dividing through by (1 + k):

```
  d[j] = k/(1 + k)*( (N+j+2)*d[j+1]/(2*j+3) - (N-j+1)*d[j-1]/(2*j-1) )
```

Then by further reorganising:

```
  d[j-1] = -(1+k)*(2*j-1)*d[j]/(N-j+1)/k
             + (N+j+2)*(2*j-1)*d[j+1]/(2*j+3)/(N-j+1)
```

Increasing the index by 1, j -> j + 1 gives:

```
  d[j] = -(1+k)*(2*j+1)*d[j+1]/(N-j)/k
          + (N+j+3)*(2*j+1)*d[j+2]/(2*j+5)/(N-j)
```

Thus d[j] is now expressed in terms of the two higher coefficients d[j+1] and d[j+2] as well as k. This expression has the property that if a reasonably large value of j is selected, say 64, then for any defined k, and arbitrary starting values for d[64] and d[65] (1.0 and 0.0), the expression can be recursed inward to j = 0, getting an estimate of d[j] at each step. The resulting coefficients are extremely accurate values of the coefficients that would be obtained by Chebyshev transforming the function exp(-t/tau). The method is much faster than performing the transform. At the end of the inward recursion, the values are exact, apart from a normalization factor. It turns out that the sum of the coefficients must be 1.0. Thus summing all the coefficients together during the inward recursion gives a factor at the end of the calculation that can be divided into each coefficient. This last step gives the Chebyshev coefficients of the exponential exp(-t/tau).

Having obtained the Chebyshev coefficients for tau1 and tau2 and calling the two arrays containing the newly calculated coefficients:

```
  d_tau1[j] and d_tau2[j]
```

Then another regression equation is formed:

```
  d[j] = a1*d_tau1[j] + a2*d_tau2[j]
```

Since d[j], d_tau1[j] and d_tau2[j] are known, then a1 and a2 can be calculated by simple linear regression. The linear equations are formed:

```
  Sum(d[j]*d_tau1[j]) =
               a1*Sum(d_tau1[j]*d_tau1[j])-a2*Sum(d_tau1[j]*d_tau2[j])

  Sum(d[j]*d_tau2[j]) =
              a1*Sum(d_tau1[j]*d_tau2[j])-a2*Sum(d_tau2[j]*d_tau2[j])
```

and the a1 and a2 amplitudes are calculated.

Calculate the Added Constant
The input data can be expected to have an offset:

```
  f(t) = offset + a1*exp(-t/tau1) + a2*exp(-t/tau2)
```

Here another property of the Chebyshev transform can be exploited. The offset term only appears in d[0]. Nowhere else. Since tau1 and tau2 now known, as well as the amplitude then:

```
  d[0] = offset + a1*c_tau1[0] + a2*c_tau2[0]
```

Thus the offset is obtained by rearranging this equation:

```
  offset = d[0] - a1*c_tau1[0] - a2*c_tau2[0]
```

Smoothing Window
In order to reduce noise and non-linear noise effects on the statistics of the fit, a smoothing window is carried across the raw coefficients d[j] and the derived coefficients db[j], dbb[j] etc. This is a window that averages the coefficients. The usefulness of this window is that because the expression for the fit parameters is linear, e.g. in the case of a double fit:

```
  d[j] + x1*db[j] + x2*dbb[j]
```

then the expression is also true of a sum of these over j:

```
  (d[j]+d[j+1]..) + x1*(db[j]+db[j+1]..) + x2*(dbb[j]+dbb[j+1]..)
```

The summation is carried over a window of usually ten coefficients. The actual window size is calculated using

```
  window = Min (Number of Data Points, 10)
```

The averaging does not alter the accuracy of the mathematics fit. However, the averaging tends to reduce the noise component, if one exists. The noise is usually distributed over the coefficients (not necessarily equally). Thus summing coefficients and using the averaged values has a significant ameliorating affect on noise. It is probably true that this is not the best way to sum over the coefficients because the transform does not produce a flat spectrum if white noise is presented to it. There are probably more sophisticated smoothing windows that could be run across the coefficients to reduce noise even more.