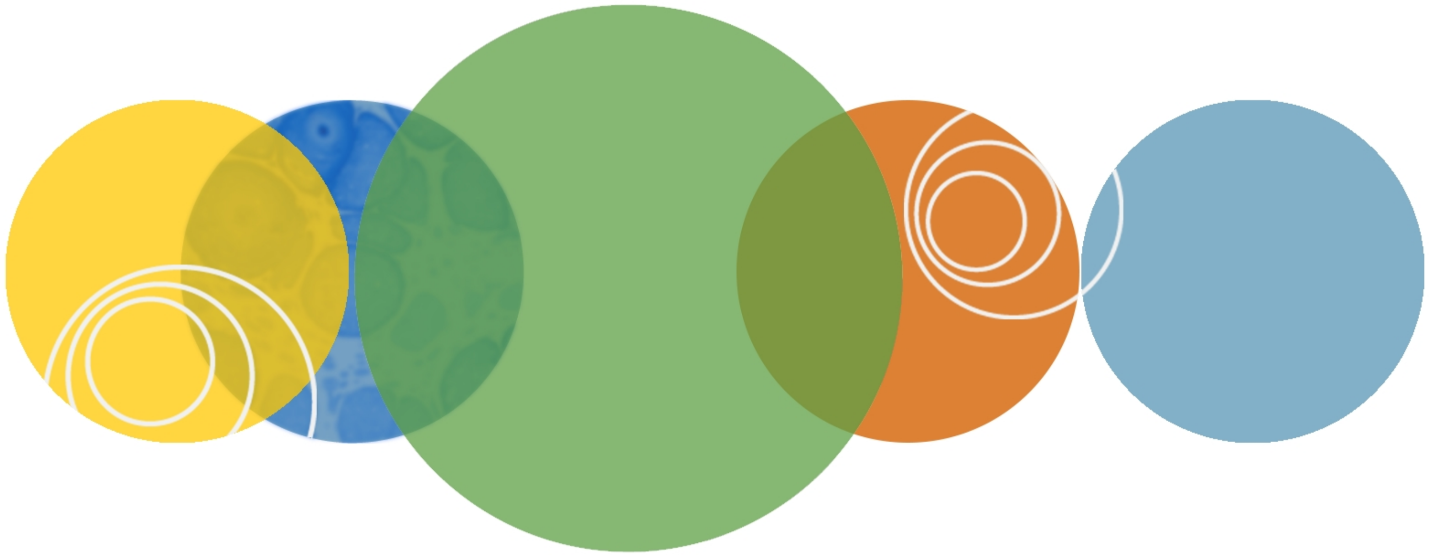


pCLAMP™ 10

Data Acquisition & Analysis Software

User Guide

1-2500-0180 G
January 2016

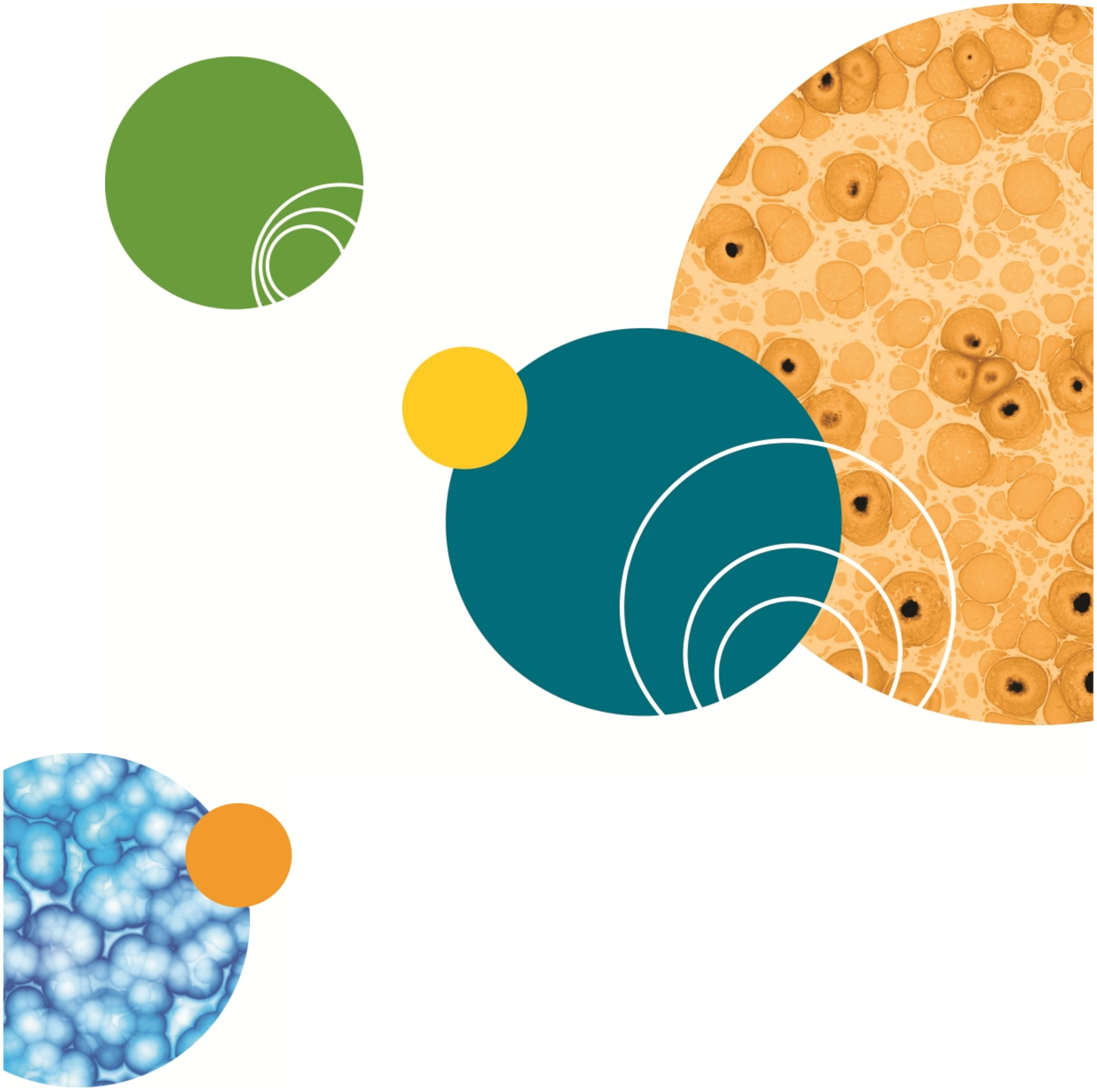
ISO 9001 REGISTERED COMPANY

For research use only. Not for use in diagnostic procedures.

Patents: http://www.moleculardevices.com/productpatents

Product manufactured by Molecular Devices, LLC.
1311 Orleans Drive, Sunnyvale, California, United States of America 94089.
Molecular Devices, LLC is ISO 9001 registered.

Contents



Chapter 1: Introduction

1

pCLAMP™ Data Acquisition & Analysis Software is the data acquisition and analysis suite from the Axon Instruments Conventional Electrophysiology product line of Molecular Devices. Designed for a variety of experiments, pCLAMP Software Version 10.6 is the latest version that has become the standard for electrophysiological experimentation and analysis. The flexibility that pCLAMP software offers allows researchers to adapt it to many uses outside its traditional applications in electrophysiology.

The pCLAMP Software Version 10.6 suite contains the following:

- Clampex Software, for data acquisition and production of stimulus waveforms.
- Clampfit Software, for data analysis.
- AxoScope Software , for background chart recording.

The optional accessory MiniDigi 2-Channel Digitizer can be ordered separately.

Clampex Software is a versatile and powerful software tool for acquiring digitized data of all types. While excellent for the acquisition of patch-clamp data, it is not limited to measuring voltage-clamp or current-clamp responses. Clampex Software can be used to measure any physical parameter that can be linearly converted into a voltage. For example, you can monitor and acquire end-plate currents, measure the fluorescence signal from a photomultiplier tube, measure pressure from a strain gauge, or acquire any other combination of analog signals.

Clampfit Software is powerful data analysis software with a wide variety of statistics, analyses, transforms, and layout tools for electrophysiological data.

Together, AxoScope Software and the optional MiniDigi digitizer provide the functionality traditionally performed by a separate chart recorder, for example, for concurrent background recording.

New Features

AxoScope Software and Clampex Software

See Enabling the HumSilencer System Controls on page 55, for further description of the following new feature.

- There are three models of the Digidata 1550B digitizer, one of which excludes the HumSilencer Adaptive Noise Cancellation System (ANC). The other two models of the Digidata 1550B digitizer with ANC provide configuration through Analog Input Channels #0, or Analog Input Channels #0, #2, #4 and #6 to eliminate 50 Hz or 60 Hz line-frequency noise and associated high-frequency harmonics.

Note: The HumSilencer system controls do not display in the Demo configuration of the AxoScope Software or Clampex Software. The controls also do not display without a connection to and configuration with a HumSilencer-enabled Digidata 1550A digitizer or Digidata 1550B digitizer.

CAUTION! The Digidata 1550B digitizer digital bits 12 through 14 are now reserved for use by the HumSilencer system and cannot be used for third-party custom programming.

- **Adapt between sweeps**: For long episodic stimulation experiments when using a HumSilencer capable digitizer, a 1.081 second inter-sweep learning period can be added to counteract the potential of noise pattern changing over time.

pCLAMP Software

Clampfit software has been updated to accommodate the new Clampex Software features. Otherwise, there are no other new features.

AxoScope Software

AxoScope Software is a subset of Clampex Software. The HumSilencer Adaptive Noise Cancellation System is available in the AxoScope Software version 10.6. AxoScope Software provides several continuous data acquisition modes, but has no episodic stimulation mode, which means that there is no capacity to generate stimulus waveforms. The Membrane Test is not included. Other advanced features found in Clampex Software, such as the LTP Assistant, the Junction Potential Calculator, and instrument telegraphs are also not included. With the optional accessory MiniDigi digitizer, AxoScope Software can be used as a background chart recorder running alongside Clampex Software during experiments.

MiniDigi Digitizer

The optional accessory MiniDigi 2-Channel Digitizer is a low-noise, two-channel digitizer, designed to function with AxoScope Software as a simple digital chart recorder. It has two independent, 16-bit analog inputs, each of which provides digitization at up to 1 kHz. The MiniDigi digitizer communicates with the host computer through a USB interface that also provides power to the digitizer.

Filtering

The MiniDigi digitizer uses your choice of minmax or analog-like filtering. If you select minmax, both the minimum and maximum values in each sample period are sent to the computer.

The analog-style filter is a low-pass, anti-aliasing filter with a cutoff frequency one fifth of the sampling rate.

Interface Description

The front panel of the MiniDigi digitizer has two BNC connectors for analog input channels 0 and 1. The back panel contains a USB connector and an LED to indicate power-on status. The LED slowly blinks to indicate communication with the software driver.

Specifications

Table 1-1: MiniDigi Specifications (analog input)

Item	Specification
Number of input channels	2 single-ended
Resolution	16-bit (1 in 65536)
Acquisition rate (per channel)	1 kHz
Input range	–10.000 V to +10.000 V
Maximum allowable input range	–50 V to +50 V
Input resistance	1 M
Gain value	1
Anti-alias filter (per channel)	Three-pole, 1.5 kHz Bessel

USB Interface

The MiniDigi digitizer has a low-power (< 100 mA), Universal Serial Bus (USB) 1 device to interface with the computer. The software driver is compatible with Windows XP Pro and Windows 7 operating systems.

pCLAMP Software Software Documentation

Extensive documentation is provided with pCLAMP Software, both to help you learn how to use the programs most effectively, and as a reference source for algorithms and other information.

- A thorough step-by-step PDF tutorial guides you through initial hardware/software setup for data acquisition.
- This user guide includes general introductory chapters for both Clampex Software and Clampfit Software (all AxoScope Software functionality is included in Clampex Software) and tutorials to help you get started with these programs. It also includes sections of general discussion on the use of pCLAMP Software, and algorithms and other reference material.
- Online Help provides specific help on each program command, as well as overview and “How to …” topics.

Setup Tutorial

The pCLAMP installation includes the PDF file tutorial, *Setting Up Clampex for Data Acquisition*, for the initial setup and configuration of Clampex Software with a digitizer and amplifier. Open the tutorial from the **Start > All Programs > Molecular Devices > pCLAMP 10.6 > Clampex 10.6 Tutorial**.

The tutorial has setup instructions covers everything necessary to start making real recordings. It takes you through:

- Cabling and telegraph setup.
- Channel and signal configuration, including setting scale factors.
- Basic protocol definition.

Help

Clampex Software, Clampfit Software, and AxoScope Software all have extensive Help. This provides detailed command-specific help as well as other more general help topics, and some “How to …” topics. The Help file can be accessed in a number of ways:

- Place the pointer over any command in the drop-down menus and then press **F1** to open Help at the topic for that command.
- Click **Help** in any dialog for help on the dialog.
- All tool buttons have associated tooltips, or pop-up descriptions of the button’s function that appear when the pointer is held over the button.
- Click **Help > Clampex Help** and find topics using the **Contents**, **Index**, or **Search** tabs, or use the internal topic links.

The heart of the application help is the “Contents” tab Menu Reference section, which matches the layout of the main window menus. All commands available in the pCLAMP Software appear in the main menus, although many of these commands are accessible through tool buttons and pop-up menus as well, and so the Menu Reference section contains an exhaustive list of the Help topics available for each command, dialog, and dialog tab.

The Help also contains an “Exploring Clampex Software/Clampfit Software/AxoScope Software” section, that introduces the main window and each of the specialty windows in the program, with extensive links to related topics within the Help.

In addition, a General Reference section has topics dealing with broader matters relevant to operating the program, but are not suited to the Menu Reference section.

All Help topics appear in the Table of Contents, but can also be accessed though the Index or Search.

User Guide

Since the application help contains extensive coverage on the use of features, the User Guide focuses on:

- Installation and setup of pCLAMP Software.
- Definitions and conventions used, and general discussion of data acquisition and analysis.
- Introduction to the main functions of each of the programs.
- Tutorials for data acquisition and analysis.
- Listing and discussion of the analytical tools used by pCLAMP Software, including algorithms.

Overview of User Guide

The chapters are organized as follows:

- Chapter 1: Introduction on page 9 lists features new to pCLAMP 10, and includes a section on the history of pCLAMP.
- Chapter 2: Description on page 17 includes definitions and conventions, and general discussions of data acquisition and file types.
- Chapter 3: Setup on page 35 lists the hardware and software requirements and recommendations for pCLAMP. It also includes detailed software installation and setup instructions.
- Chapter 4: Clampex Software Features on page 45 provides an introduction to the main Clampex Software features, with special attention to functions new to this version.
- Chapter 5: Clampex Software Tutorials on page 87 includes tutorials designed to guide new users through experimental setup, introducing them to a range of functions available in Clampex Software. It includes a discussion of the variety of experiments that can be conducted.
- Chapter 6: Clampfit Software Features on page 111 provides a general introduction to the main pCLAMP Software features, with special attention to functions new to this version.
- Chapter 7: Clampfit Software Tutorials on page 133 takes the reader step-by-step through analytical procedures common in the evaluation of physiological data.
- Chapter 8: Digital Filters on page 155 describes each digital filter and its characteristics.
- Chapter 9: Clampfit Software Analysis on page 185 is an introduction to Fourier analysis and its application to the power spectra of electrophysiological data.
- Chapter 10: pCLAMP Analysis on page 191 details the formulas used and discusses the various analyses in pCLAMP.
- Chapter 11: Curve Fitting on page 205 introduces the varieties of fitting methods available, and guides the reader in the choice of appropriate methods.
- Chapter 12: Fitting Functions on page 237 describes each of the predefined fitting functions and explains their use, restrictions, and any required data preprocessing.

Utility Programs

Along with the three main software programs of the suite (for which icons are placed on the computer desktop in installation), pCLAMP Software includes a number of utility software. These are all loaded in ..**\Program Files\Molecular Devices\pCLAMP 10.6**. You can open the **pCLAMP 10.6** folder in Windows Explorer and double-click on these programs to run them. The utility programs are:

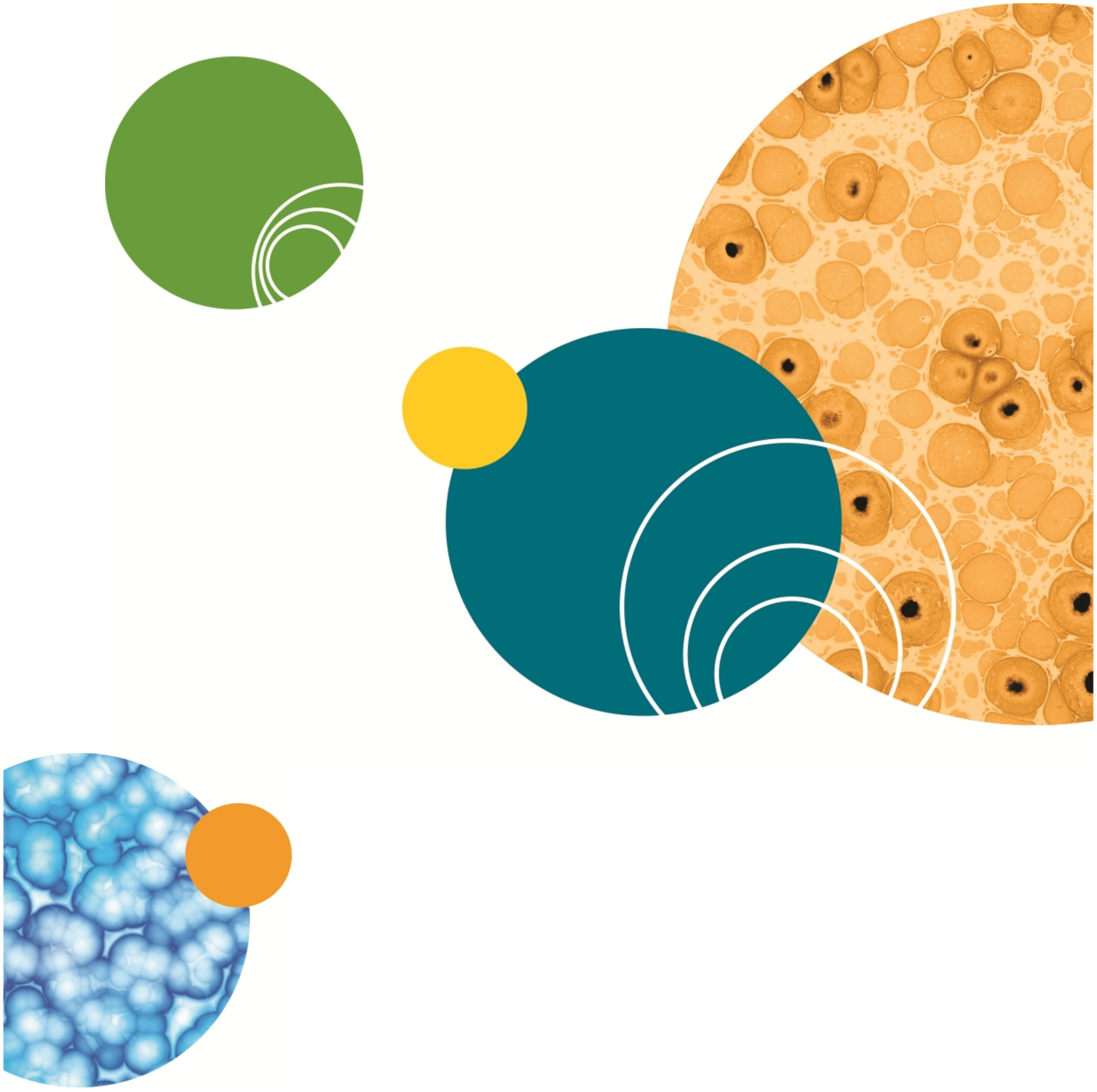
- **Reset to Program Defaults** (ClearRegistry.exe): This program is more conveniently run from the desktop: **Start > Programs > Molecular Devices > pCLAMP 10.6 > Reset to Program Defaults**. Use it when you encounter strange behavior in the program, or just want to set various Windows registry components back to their factory default settings.
- **ABF Info (ABFInfo.exe)**: This is a data file property viewer. Select ABF data files and view their header information in terse, normal or verbose modes. Several file headers can be viewed at once for comparison, but the header information cannot be edited. You can also view file headers one at a time in Clampex Software and Clampfit Software, by opening an ABF file and then selecting **File > Properties**.
- **DongleFind** (DongleFind.exe): A simple application that checks the computer for Axon Instruments software security keys (dongles), including network keys. The application can be set to report a range of information about the keys it finds.

History of pCLAMP

The first pCLAMP applications for controlling and analyzing electrophysiological experiments with computers emerged at the California Institute of Technology (Caltech) in 1973. They were originally used for kinetic studies on nicotinic acetylcholine receptors and on voltage-sensitive currents. These early versions were converted to PC-compatible software in mid-1982 and have been in use since 1983 (Kegel et al., 1985).

In 1984, with a view toward serving the entire community of cellular neurobiologists and membrane biophysicists, California Institute of Technology (Caltech) licensed the package to Axon Instruments, Inc. for continued development. pCLAMP continued to evolve into an extremely powerful suite of applications that were used by a broad spectrum of researchers in electrophysiology.

In 1998, the first Windows version of pCLAMP 7 was released. Full conversion from MS-DOS to Windows was completed in 2002 and included support for a variety of synaptic data (long-term potentiation/depression [LTP/LTD] and minis), as well as action potentials.



Chapter 2: Description

2

Starting with definitions of many of the terms used in pCLAMP, this chapter contains information of a general nature likely to be useful for pCLAMP users. Middle sections in the chapter discuss electrophysiology terminology conventions, and there is a discussion of the theory behind sampling. The final two sections list pCLAMP file types and a range of pCLAMP "vital statistics".

Definitions

A number of terms in standard use in Axon Instruments/Molecular Devices applications are defined below:

- A **waveform** consists of a series of analog voltage steps, ramps and/or trains of pulses, or arbitrary data in a file, generated on an output signal in order to stimulate a cell. Also termed the "command waveform" or "stimulus waveform", it can also have digital outputs associated with it.
- An **epoch** is a subsection of a waveform that can be defined as a step, ramp, or pulse train, and increased or decreased incrementally in amplitude and/or duration from sweep to sweep within a run.
- A **sample** is the datum produced by one A/D (analog-to-digital) conversion or one D/A (digital-to-analog) conversion. In analysis contexts, samples can be referred to as points.
- A **sweep** is the digitized data from all input signals for a defined number of samples. A sweep can contain up to one million samples, with all signals multiplexed at equal time intervals. A command waveform can be concurrently output during a sweep. Sweeps were known as episodes in older versions of pCLAMP. See Figure 2-1 for illustration of the relationship between runs, sweeps, channels and trials.
- A **run** is a set of sweeps. Sweeps within a run may all be the same, or they can be configured to have amplitude and/or duration changes from sweep to sweep. A run can contain up to 10,000 sweeps. If multiple runs are specified, all sets of sweeps are averaged together to produce a single averaged set of sweeps. See Figure 2-1 for illustration of the relationship between runs, sweeps, channels and trials.
- A **trial** is the data digitized from one or more runs, and saved as a single file. See Figure 2-1 for illustration of the relationship between runs, sweeps, channels and trials.

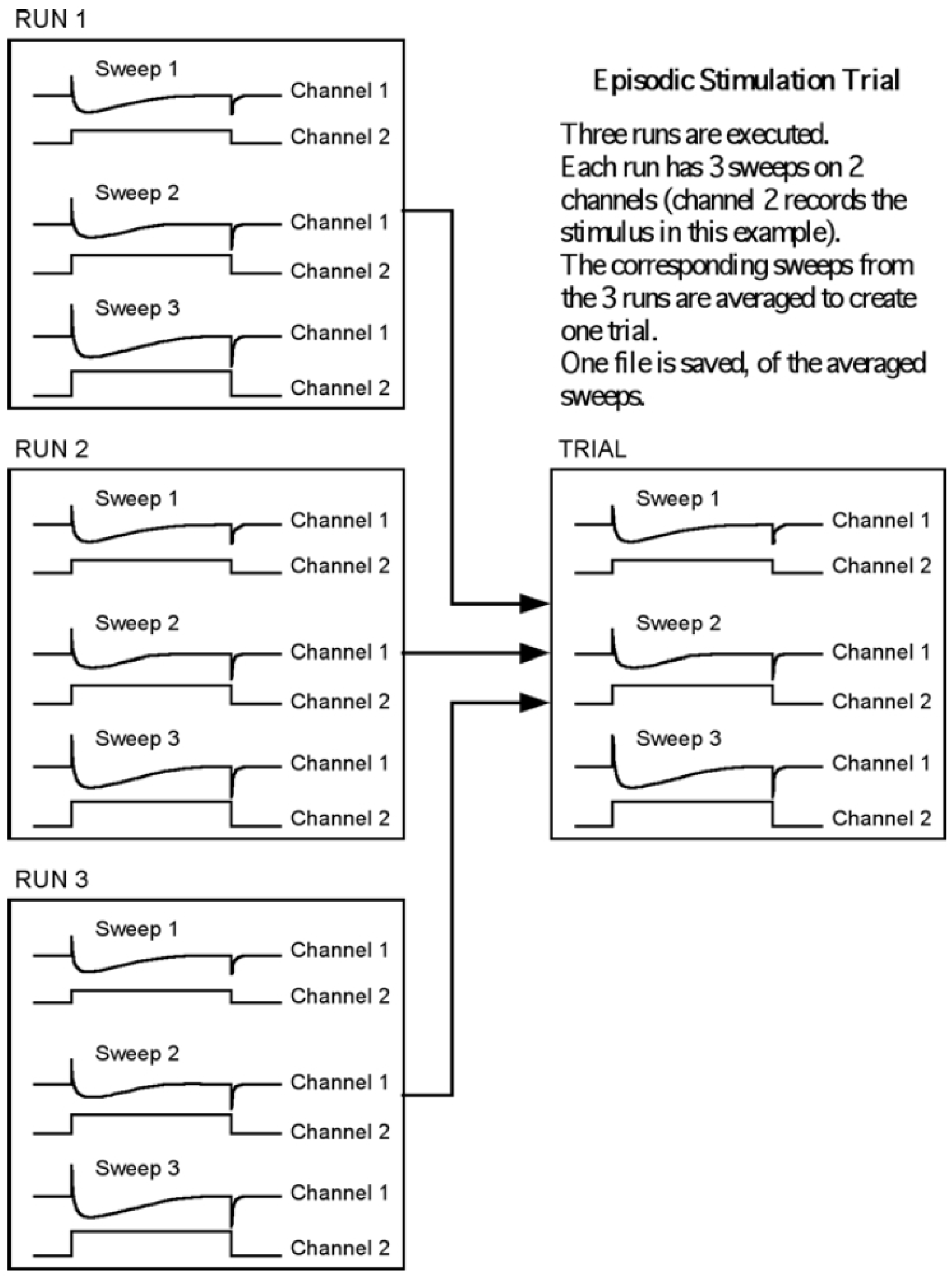


Figure 2-1: Clampex Software data structure showing relationship between runs and trial.

- A **trace** is a continuous set of data samples from a single input signal. When data are displayed as sweeps, each trace represents a sweep within that signal.

- A **point** is a single datum in a data file, similar to sample, above, although points can be created in a file without having been converted from an analog signal by an A/D converter.
- A **channel** is a physical connection through which analog and digital signals are received or transmitted. Channels are identified in pCLAMP by the name of the digitizer port where connection is made: e.g. Analog IN #0, Digital OUT #5.
- A **signal** is a set of name, unit, scale factor and offset, according to which:
    - Voltage inputs received at the analog input ports of the digitizer are represented in Clampex Software as the physical parameter (unit) actually being read by the amplifier or transducer, with correct scaling and offset, and a user-selected name.
    - Voltage outputs generated through the digitizer's analog output ports are represented in Clampex Software as the physical parameter (unit) actually being delivered to the preparation by the amplifier or transducer, with correct scaling and offset, and a user-selected name.

In Clampex Software, numerous signals can be configured in association with each analog input and output channel (in the Lab Bench). A specific signal is assigned to a channel in the protocol configuration.

- A **protocol** is a set of configuration settings for a trial. It defines the acquisition mode, the trial's hierarchy (for example the number of sweeps per run, and runs per trial), the sampling rate, the definition of the waveform, and many other options as well, which can all be saved into a *.pro protocol file.
- An **experiment** can be composed of several different protocols, and thus may result in several data files. In the context of sequencing keys, where protocols can be assigned to keys and also linked to follow one another, the *.sks files which define the keys and linkages can be said to define an experiment. Configurations created in the LTP Assistant, which also result in *.sks files, are similarly called experiments.
- An **event** is a discrete response of biological activity, usually relatively short, within an input signal. It can be characterized by event detection, and extracted for further data analysis.
- The **baseline** in an episodic sweep consists of the initial and ending points of the trace, during which the holding level is output. Or, it is the level in an input signal that a trace maintains during periods that no events occur.
- A **peak** is a point in a trace of local maximum deviation from the baseline. Peaks can be positive (above the baseline) or negative (below the baseline).
- The **rise** is that part of the trace that, in an event, goes from the direction of the baseline to the peak. In the case of a negative peak, the rise is a downwards movement of the trace. In previous versions of pCLAMP, rising phases were sometimes referred to with the term left, as in "greatest left slope".
- The **decay** is that part of the trace that, in an event, goes from the direction of the peak back to the baseline. In the case of a negative peak, the decay is an upwards movement of the trace. In previous versions of pCLAMP decay phases were sometimes referred to with the term right, as in "greatest right slope".

- The **mode** most commonly referred to in pCLAMP is the data Acquisition Mode, set in the Protocol Editor. This determines how data acquisition is triggered and stopped, and whether it is accompanied by a command waveform (see Data Aquisition Modes on page 20).

  Mode can also refer to amplifier mode, which is the amplifier state of current clamp, voltage clamp, I = 0, etc.
- **Acquisition** is the digitization of data by Clampex Software. Acquired data can be displayed in the Scope window at the same time they are recorded to disk, or viewed without recording.
- **Electrode resistance** (R_e), also called pipette resistance (R_p), is the resistance due to the electrode. It does not include resistance due to environmental current-impeding factors near the electrode tip, e.g. cellular debris, air bubbles, poorly conducting solution etc.
- **Access resistance** (R_a) is the sum of the electrode resistance and resistance due to current-impeding factors near the electrode tip, e.g. cellular debris, etc. Access resistance is sometimes called series resistance (R_s). This is the term used on Axon-made amplifiers.
- **Membrane resistance** (R_m) is the resistance across the cell membrane.
- **Total resistance** (R_t) is the sum of membrane resistance and access resistance. When an electrode seals against the membrane, if the seal is successful, for example a gigaohm seal, access resistance is a negligible component of the total resistance, so the total resistance is effectively equal to the seal resistance.
- **Seal resistance** is the resistance given by the seal between the electrode tip and the cell membrane.

Data Aquisition Modes

Clampex Software provides five distinct data acquisition modes.

Gap-Free Mode

This mode is similar to a chart or tape recorder, where large amounts of data are passively and continuously digitized, displayed, and saved without any interruptions to the data record.

Variable-Length Events Mode

Data are acquired for as long as an input signal has passed the threshold level, or for as long as an external trigger is held high. This mode is ideal for experiments such as recording of single-channel currents that are in a closed state for long periods of time and contain periods of random bursting.

Fixed-Length Events Mode

Data are acquired for same-length sweeps whenever an input signal has passed the threshold level, or when an external trigger occurs. This mode is ideal for recording synaptic events, action-potential spikes or other constant-width events. If during one fixed-length event a second trigger occurs, a second fixed-length event is started. The two events have overlapping data until the first event ends. In this way no events are lost, and each event has the same length.

High-speed Oscilloscope Mode

In this mode, data are acquired in sweeps, as with a standard oscilloscope. The input sweep can be triggered by either an external trigger, an autotrigger, or the input signal crossing a threshold level.

High-speed oscilloscope mode resembles fixed-length event mode, except that extra triggers occurring during a sweep do not initiate additional sweeps.

Episodic Stimulation Mode

An analog waveform, holding level, and/or digital pulses are output, while data are simultaneously acquired in fixed-length sweeps. Each sweep is non-overlapping and can be triggered by an internal timer or by a manual or external pulse.

Episodic stimulation mode is useful for studying voltage-activated currents using the whole-cell patch-clamp configuration. For example, Clampex Software can be used to drive a membrane potential to various potentials in controlled amplitude and duration increments or decrements. The cellular response to these test potentials is simultaneously acquired. Special features include pre-sweep trains, online leak current subtraction, online peak detection and statistics, and an online derived-math channel. Online statistics can be used to chart peak measurement values in real time.

Terms and Conventions in Electrophysiology

Current and Voltage Conventions

In many cases, Clampex Software is used to record membrane potential and current. There are various conflicting definitions of current and voltage polarities, so we take the opportunity to discuss here the conventions used for Axon Instruments products. This information is also presented in various instrument manuals and other publications provided by Axon Instruments/Molecular Devices.

Positive Current

In this discussion, and in all amplifiers manufactured by Axon Instruments/Molecular Devices, the term positive current means the flow of positive ions out of the headstage into the micropipette and out of the micropipette tip into the preparation.

Inward Current

Inward current is current that flows across the membrane, from the outside surface to the inside surface.

Outward Current

Outward current is current that flows across the membrane, from the inside surface to the outside surface.

Positive Potential

In this discussion, and in all amplifiers manufactured by Axon Instruments/Molecular Devices, the term positive potential means a positive voltage at the headstage input with respect to ground.

Transmembrane Potential

The transmembrane potential (V_m) is the potential at the inside of the cell minus the potential at the outside. This term is applied equally to the whole-cell membrane and to a membrane patch.

Depolarizing/Hyperpolarizing

The resting V_m value of most cells is negative. If a positive current flows into the cell, Vm initially becomes less negative. For example, V_m might shift from an initial resting value of -70 mV to a new value of -20 mV. Since the absolute magnitude of V_m is smaller, the current is said to depolarize the cell (for example, it reduces the “polarizing” voltage across the membrane). This convention is adhered to even if the current is so large that the absolute magnitude of V_m becomes larger. For example, a current that causes V_m to shift from -70 mV to +90 mV is still said to depolarize the cell. Stated simply, depolarization is a positive shift in V_m. Conversely, hyperpolarization is a negative shift in V_m.

Whole-Cell Voltage and Current Clamp

Depolarizing/Hyperpolarizing Commands

In whole-cell voltage clamp, whether it is performed by TEVC, dSEVC, cSEVC or whole-cell patch clamp, a positive shift in the command voltage causes a positive shift in V_m and is said to be depolarizing. A negative shift in the command voltage causes a negative shift in V_m and is said to be hyperpolarizing.

Transmembrane Potential vs. Command Potential

In whole-cell voltage clamp, the command potential controls the voltage at the tip of the intracellular voltage-recording micropipette. The transmembrane potential is thus equivalent to the command potential.

Inward/Outward Current

In a cell generating an action potential, depolarization is caused by a flow of positive sodium or calcium ions into the cell. That is, depolarization in this case is caused by an inward current.

During intracellular current clamp, a depolarizing current is a positive current out of the micropipette tip into the interior of the cell. This current then passes through the membrane out of the cell into the bathing solution. Thus, in intracellular current clamp, a depolarizing (positive) current is an outward current.

During whole-cell voltage clamp, sodium inward current flows in some cells after a depolarizing voltage step. This current is canceled by an equal and opposite current flowing into the headstage via the micropipette. Thus it is a negative current. When two-electrode voltage clamp was first used in the early 1950s, the investigators chose to call the negative current that they measured a depolarizing current because it corresponded to the depolarizing sodium current. This choice, while based on sound logic, was unfortunate because it means that from the recording instrument's point of view, a negative current is hyperpolarizing in intracellular current-clamp experiments but depolarizing in voltage-clamp experiments.

Because of this confusion, Axon Instruments/Molecular Devices has decided to always use current and voltage conventions based on the instrument's perspective. That is, the current is always unambiguously defined with respect to the direction of flow into or out of the headstage. Some instrument designers have put switches into the instruments to reverse the current and even the command voltage polarities so that the researcher can switch the polarities depending on the type of experiment. This approach has been rejected by Axon Instruments/Molecular Devices because of the real danger that if the researcher forgets to move the switch to the preferred position, the data recorded on the computer could be wrongly interpreted. We believe that the data should be recorded unambiguously.

Patch Clamp

The patch-clamp pipette current is positive if it flows from the headstage through the tip of the micropipette into the patch membrane. Whether it is hyperpolarizing or depolarizing, inward or outward, depends upon whether the cell is “cell attached”, “inside out” or “outside out”.

Cell-Attached Patch

The membrane patch is attached to the cell. The pipette is connected to the outside surface of the membrane. A positive command voltage causes the transmembrane potential to become more negative, therefore it is hyperpolarizing. For example, if the intracellular potential is –70 mV with respect to 0 mV outside, the potential across the patch is also –70 mV. If the potential inside the pipette is then increased from 0 mV to +20 mV, the transmembrane potential of the patch hyperpolarizes from –70 mV to –90 mV.

From the examples it can be seen that the transmembrane patch potential is inversely proportional to the command potential, and shifted by the resting membrane potential (RMP) of the cell.

A positive pipette current flows through the pipette, across the patch membrane into the cell. Therefore a positive current is inward.

Inside-Out Patch

The membrane patch is detached from the cell. The surface that was originally the inside surface is exposed to the bath solution. Now the potential on the inside surface is 0 mV (bath potential). The pipette is still connected to the outside surface of the membrane. A positive command voltage causes the transmembrane potential to become more negative, therefore it is hyperpolarizing. For example, to approximate resting membrane conditions, say V_m = –70 mV, the potential inside the pipette must be adjusted to +70 mV. If the potential inside the pipette is increased from +70 mV to +90 mV, the transmembrane potential of the patch hyperpolarizes from –70 mV to –90 mV.

From the example it can be seen that the transmembrane patch potential is inversely proportional to the command potential.

A positive pipette current flows through the pipette, across the patch membrane from the outside surface to the inside surface. Therefore a positive current is inward.

Outside-Out Patch

The membrane patch is detached from the cell in such a way that the surface that was originally the outside surface remains exposed to the bath solution. The potential on the outside surface is 0 mV (bath potential). The pipette interior is connected to what was originally the inside surface of the membrane. A positive command voltage causes the transmembrane potential to become less negative, therefore it is depolarizing. For example, to approximate resting membrane conditions, say V_m = -70 mV, the potential inside the pipette must be adjusted to –70 mV. If the potential inside the pipette is then increased from –70 mV to –50 mV, the transmembrane potential of the patch depolarizes from –70 mV to –50 mV.

The membrane potential is directly proportional to the command potential.

A positive pipette current flows through the pipette, across the patch membrane from the inside surface to the outside surface. Therefore a positive current is outward.

Summary

- Positive current corresponds to:

Cell-attached patch	patch inward current
Inside-out patch	patch inward current
Outside-out patch	patch outward current
Whole-cell voltage clamp	outward membrane current
Whole-cell current clamp	outward membrane current

- A positive shift in the command potential is:

Cell-attached patch	hyperpolarizing
Inside-out patch	hyperpolarizing
Outside-out patch	depolarizing
Whole-cell voltage clamp	depolarizing

- The correspondence between the command potential (V_{cmd}) and the transmembrane potential (V_m) is:

Cell-attached patch	V_m = RMP - V_{cmd}
Inside-out patch	$V_m = -V_{cmd}$
Outside-out patch	$V_m = V_{cmd}$
Whole-cell voltage clamp	$V_m = V_{cmd}$

The Sampling Theorem in pCLAMP Software

The sampling theorem states that an analog signal can be completely reproduced by regularly spaced samples if the sampling frequency is at least 2 times that of the highest frequency component in the signal. Thus the minimum sampling interval T is given by

$$T = \frac{1}{2f_h}$$

where f_h is the highest frequency component. For example, if the highest frequency component in an analog signal is 5000 Hz, the sampling rate should be at least 10,000 times per second if the signal is to be faithfully reproduced.

The maximum frequency f_h in an analog signal is generally referred to as the Nyquist frequency. The minimum sampling rate of $2f_h$ samples per second that is theoretically required to accurately reproduce the analog signal is referred to as the Nyquist rate. If the sampling rate is less than the Nyquist rate, then two types of errors are introduced. The first is that high frequency information will be irretrievably lost. The second is the introduction of artificial low frequency components, referred to as aliasing.

Aliasing is especially problematic if there are periodic components in the analog signal. This is illustrated in Figure 2-2, which represents a 2500 Hz sine wave sampled at a rate of 2000 Hz (two-fifths the Nyquist rate). The sampling points are shown as dark squares. The reconstructed 500 Hz waveform (heavy line) is only one-fifth the frequency of the original signal.

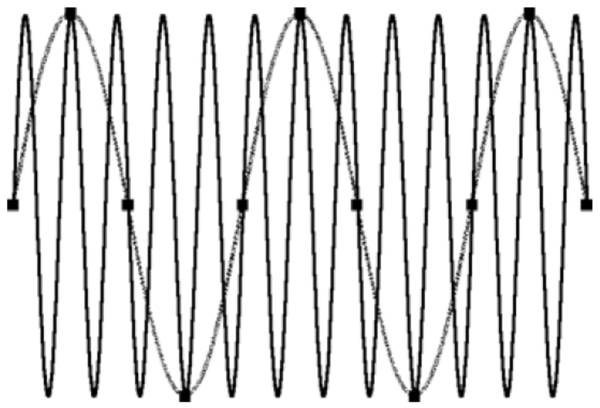


Figure 2-2: Illustration of aliasing.

In the real world, information regarding the exact spectral composition of analog signals is rarely available, especially in the presence of noise. To avoid distortion of the digital signal by frequencies that are above the Nyquist limit, a value of f_h should be selected in accordance with the particular experimental requirements, and the analog signal should be lowpass filtered before sampling to reject frequencies above f_h. Filters applied in this way are known as antialiasing filters or guard filters.

In practice, it is common to sample at a rate significantly faster than the minimum rate specified by the sampling theorem. This is known as oversampling. Exactly how much oversampling should be used depends on the experiment.

Optimal Data Acquisition

Computer-based data acquisition extends the range of the types, complexity and size of experiments that can be readily performed in the laboratory. To use these tools effectively, several key concepts that underpin all computerized data acquisition systems should be understood, and these are discussed briefly in the following sections.

Analog Data

The fundamental property of analog data is that it is continuous. Analog data can be obtained from transducers recording a wide variety of properties, including (but not limited to) voltage, current, pressure, pH, speed, velocity, light, sound levels, etc. The amplitudes of any of these signals may vary over wide ranges, and the increments are infinitesimally small. While analog signals can be recorded directly by transferring them from an analog output to an analog recording device (for example chart recorder, FM tape recorder, etc.), analysis and reproduction of these records always involves some signal degradation, due to the effects of noise or distortion.

Analog to Digital Conversion

The loss of analog signal fidelity can be minimized by the effective use of analog-to-digital conversion. This is the process of converting an analog signal into a digital representation. Such a representation can be stored on a computer disk, printed page, etc., without subsequent signal degradation. The popularity of the audio compact disc is based on the effective use of analog-to-digital conversion to store and reproduce the music recorded on it. The digital representation can then be replayed precisely, as often as desired, without the introduction of noise.

While seemingly obvious, the effective use of analog-to-digital conversion requires that one consider several conflicting goals carefully. In simplest terms, one must decide how best to preserve the fidelity of the analog signal, using an affordable quantity of recording media. Since an analog signal is continuous, A/D conversion inherently yields an approximation of the original data. The goal in A/D conversion is to make reasonable assumptions with respect to the necessary temporal and amplitude resolutions that are required to reproduce the original analog signal, and then to choose and set the acquisition parameters appropriately.

Temporal Resolution

Does the digital representation of the original signal faithfully reproduce the response of the analog signal in the time domain? It is obvious that a signal with a 10 Hz component will not be reproduced well by sampling the signal once per second (1 Hz). While an acquisition at 1,000 Hz is intuitively adequate, it uses unnecessary resources for the storage of the acquired signal.

So how fast do you need to sample to adequately reproduce the temporal characteristics of the analog signal? The Nyquist sampling theorem states that if a DC signal is sampled at a rate that is twice the analog bandwidth of the signal, the sampled values can reproduce the original signal. Thus, if the sampling rate is $1/\tau$ (where τ is the sampling interval), the signal must have no frequency components greater than $1/(2\tau)$. Sampling at a frequency of twice the analog bandwidth is the theoretical minimum required to reproduce the source signal.

If appropriate sampling frequencies are not used, two potential errors are introduced. The most obvious one is that the high frequency information is lost; the less obvious error is the introduction of aliasing. Aliasing is the introduction of a spurious low-frequency signal. For those old enough to remember 8 mm home movies (or fortunate enough to have video frame grabbers in their computers), frame rates of 8–12 frames per second often yield cars whose wheels appear to be turning backwards (while the car is moving forward!). This illusion is the effect of aliasing on our visual perception of the wheel motion.

Aliasing, Filtering, and Oversampling

In practice, it is common to sample at a rate significantly faster than the minimum rate specified by the sampling theorem. This is known as oversampling. Exactly how much oversampling should be used depends upon the type of experiment.

For experiments where the data will be analyzed in the frequency domain (for example noise analysis, impedance analysis), it is common to oversample only modestly. The main concern is to prevent aliasing. An antialiasing filter is introduced between the signal source and the analog-to-digital converter to control the bandwidth of the data.

The factor of twice the analog bandwidth required by the sampling theorem is only applicable if the antialiasing filter is ideal, for example the gain in the pass-band is unity and in the stop-band it abruptly changes to zero. Ideal filters cannot be realized, although they can be closely approximated. For frequency-domain analysis, it is common to use sharp cutoff filters such as Butterworth or Chebyshev realizations. Sampling is typically performed at 2.5 times the filter bandwidth. For example, if the data are filtered at 10 kHz, they should be sampled at about 25 kHz. Slower sampling rates are unacceptable. Faster sampling rates are acceptable, but offer little advantage, and increase the storage and analysis requirements.

For experiments where the data will be analyzed in the time domain (for example pulse analysis, IV curves), greater oversampling is required. This is because reconstruction of the analog signal requires not only an ideal antialias filter, but also an ideal reconstruction filter. The simplest and most common reconstruction filter is to join each sample by a straight line. Other techniques can be used, such as cubic-spline interpolation, but because of their much heavier computational requirements they are infrequently used.

There is no golden rule to determine how fast to sample data for time-domain analysis, but in general, 5 times the analog bandwidth is common, and 10 times is regarded as good.

Amplitude Resolution

The amplitude resolution of an A/D converter corresponds to the smallest increment in signal that it can resolve. This resolution is a result of two properties of the converter hardware: the number of “bits” in the conversion, and the full-range voltage input that the A/D converter can handle. Most high-speed A/D converters (such as those supported by pCLAMP) are binary devices with 12 to 16 bits of resolution. The number of possible A/D values is a power of two, often referred to as the number of bits.

Commonly, these values are:

8-bit converter = 28 = 256 value

12-bit converter = 212 = 4,096 values

16-bit converter = 216 = 65,536 values

The full voltage range of most A/D converters is typically ±10 V. The amplitude resolution is defined by the full-scale input range divided by the number of sample values (or quanta). Thus, for a 12-bit system, the amplitude resolution is 20 V/4,096 quanta or 4.88 mV/quanta. For a 16-bit system, the resolution is 0.305 mV/quanta.

If one wants to obtain the best resolution of the source signal, the need for amplifiers and/ or preamplifiers to scale the input signal appropriately becomes apparent. The goal is to have the input signal use as much as possible of the input voltage range of the converter, so that the resolution of the data signal can be as precise as possible. Thus, for a biological signal that varies over the range of ±100 mV, amplification with a gain of up to 100 is needed to fill the ±10 V data acquisition range.

File Formats

Binary Data

Clampex Software acquires and stores data in the Axon Binary Format file format (ABF). Binary encoding is compact, so data files do not occupy more space on disk than is necessary.

There are two types of ABF files:

- Binary Integer: When raw data are digitized, acquired and stored to a data file, they are saved as binary integer numbers.
- Binary Floating Point: When a data file is opened into an Analysis window for review, the data are internally converted to floating point numbers. This increases the amount of precision, which is necessary for applying certain mathematical operations to the data. When saving a data file from an Analysis window, you can save it in either integer format or floating-point format.

Clampex Software and pCLAMP Software can read integer or floating point ABF files. All pCLAMP 10 programs read all previous versions of pCLAMP ABF data files. In addition, there are several third-party software packages that directly support our integer binary data file format (see Programs and Sources on page 259).

Text Data

Clampex Software can also save data files in the Axon Text File file format (ATF), which is an ASCII text format. Thus, ATF files are easily imported into spreadsheet, scientific analysis, and graphics programs, and can also be edited by word processor and text editor programs.

Be aware that data stored in a text format occupies much more disk space than data stored in a binary format, and does not include the full header information in an ABF file, so the ATF format is only recommended for transferring data into other programs that do not support the ABF file format.

Plain Data Files

Clampex Software can read and write binary and text files that do not have a header. When such files are read, the header information must be manually re-created by the user. For this reason, plain binary and plain text files are not recommended, but they can sometimes be useful for data interchange between third-party programs and Clampex Software.

Header Information (ABFInfo)

The ABFInfo utility allows you to inspect the header information of ABF binary data files and ABF protocol files. The header information includes the entire protocol used to acquire the data, as well as information specific to the file, such as the time of acquisition, and the actual number of samples acquired.

Programming Information

The file format specifications are described in detail in ABFInfo and in the ABF File Support Pack (FSP) for programmers. This is available from the Molecular Devices web site. The Axon FSP includes library routines written in C++, and supports the reading and writing of ABF- and ATF-format files.

Axon Instruments' long-standing role in electrophysiology data acquisition systems has led to the support of several third-party programs for the analysis of pCLAMP data. Individual investigators can sometimes find such programs suited for their particular needs. In order for Clampex Software binary integer data files to be recognized by software packages compatible with pCLAMP 6 binary data files, it may be necessary to record the data with the Clampex Software Program Option configured to "Ensure data files and protocols are compatible with pCLAMP 6", and to change the data file's extension from "abf" to "dat".

With pCLAMP 10, the ABF file format has changed, so that third-party programs compatible with pCLAMP 9 data files might not be able to read pCLAMP 10 data files. In this case, pCLAMP 10 data files will need to be exported as pCLAMP 9 compatible files. Likewise, with the release of pCLAMP 10.6, the ABF file format has again been updated, and pCLAMP 10.6 data files cannot be read by earlier versions of pCLAMP 10.x. In this case, pCLAMP 10.x needs to be updated to version 10.6 – see the Molecular Devices Support Knowledge Base for downloads.

Other File Formats

pCLAMP uses a number of specialized file formats for files generated in specific contexts.

These are:

- **Axon Layout File (ALF)**: For data saved from the pCLAMP Software Layout window, where data can be arranged for presentation. The ALF file format has changed in pCLAMP Software and is not compatible with earlier versions.
- **Data File Index (DFI)**: For files saved from the Data File Index window in both Clampex Software and pCLAMP Software. The Data File Index window is a file management tool.
- **Junction Potential Calculation (JPC)**: For files saved from Clampex Software's Junction Potential Calculator.
- **Rich Text Format (RTF)**: This is a general-use text format not specific to pCLAMP. Lab Book files are saved in RTF.
- **Protocol File (PRO):** For protocol settings in the Clampex Software Protocol Editor. These files are stored in ABF format, like the binary data files.
- **Results File (RLT)**: For files saved from the Results window in Clampex Software or pCLAMP Software. Any graphs generated from Results window data are also saved in the RLT file.
- **Sequencing Key File (SKS)**: For sequencing key sets saved from **Configure > Sequencing Keys** in Clampex Software. Sequencing keys allow you to link protocols and the Membrane Test to follow in predefined sequences.

- **Search Protocol File (SPF)**: For search configurations saved from the three pCLAMP Software event detection searches (single-channel, threshold and template).
- **Statistics File (STA)**: For data from the Clampex Software **Online Statistics** window. If opened in pCLAMP Software, these files open both in a Statistics window and as text in the Results window.

pCLAMP Software Quantitative Limits

Maximum and minimum settings for a range of parameters in Clampex Software and pCLAMP Software are presented in the following tables:

Clampex Software

Table 2-1: Clampex Software Settings

Location	Parameter	Quantity
Protocol Editor Mode > Rate tab	Max. runs per trial	10,000
	Max sweeps per run	10,000
	Max. samples per sweep	1,032,258
	Max. samples per trial	2,147,483,647 including unacquired samples between sweeps
	Max. data file size	4 GB
	Max. sampling rate (kHz)	500 (DD1550A, DD1550) 250 (DD1440A)
	Min. sampling rate (Hz)	1
	Min. sampling interval (µs)	2 µs (DD1550A, DD1550) 4 µs (DD1440A)
	Min. start-to-start interval	Sweep length (for example zero delay between end of one sweep and start of next)
	First and last holding periods	1/64 of sweep length each
Protocol Editor Inputs tab	Max. analog input channels	8 (DD1550A, DD1550) 16 (DD1440A) 15 if a Math signal or P/N leak subtraction enabled
Protocol Editor Outputs tab	No. analog output channels	8 (DD1550A, DD1550,) 4 (DD1440A)
Protocol Editor Statistics tab	Max. samples in boxcar filter smoothing window	21 (for example 10 on either side of each sample)

Table 2-1: Clampex Software Settings (continued)

Location	Parameter	Quantity
Protocol Editor Inputs tab	Max. epochs	10
	Max. waveform channels	8 (DD1550A, DD1550,) 4 (DD1440A)
Protocol Editor Stim tab	Max. characters in User List	512 (including commas; unlimited with Repeat)
	Max. Leak Subtraction subsweeps	8
	Max. pre-sweep train pulses	10,000
Scope window	Max. open windows	4
Analysis window	Max. open windows	16
Sequencing keys	Max. defined keys	50, plus additional 32 MultiClamp amplifier-mode keys

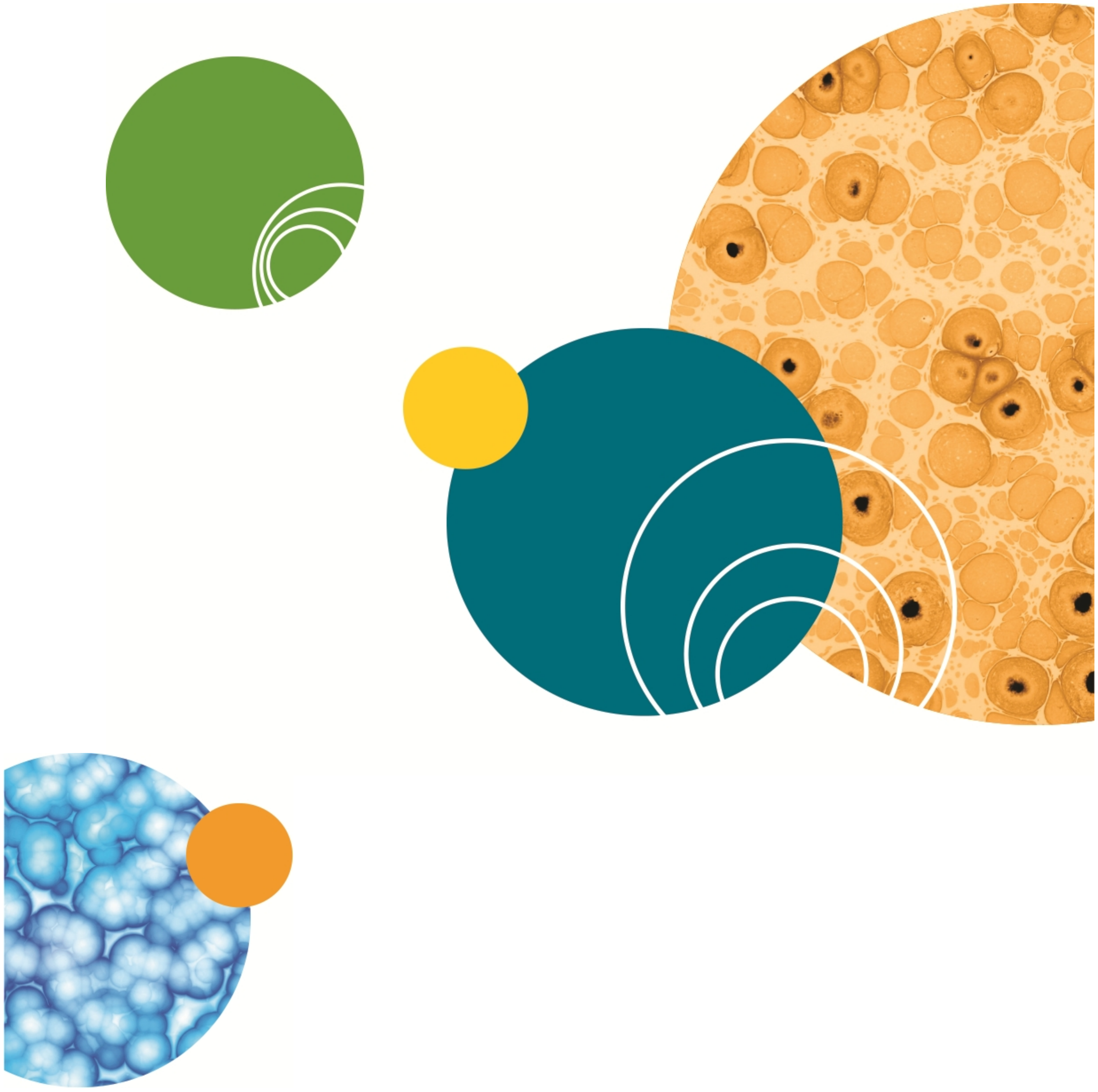
pCLAMP Software

Table 2-2: pCLAMP Software Settings

Location	Parameter	Quantity
Analysis window	Max. open windows	No limit
	Max. signals	12
	Max. characters in Select Sweeps user-entered list	512 (including commas)
	Max. samples transferred to Results or Graph windows	1,000,000
Fitting (Analysis, Graph, & Results windows)	Max. points that can be fitted	1,000,000
	Max. function terms	6
	Max. power	6
	Max. custom function parameters	24
	Max. independent variables in a custom function	6
	Max. independent variables in a custom function	1
	Max. points for functions containing a factorial term	170

Table 2-2: pCLAMP Software Settings (continued)

Location	Parameter	Quantity
Results window	Max. rows	1,000,000
	Max. imported columns	8,000
	Number of sheets	20
	No. of operations that can be undone	10
	Max. rows for Create Data	110,000
Event detection	Max. search categories (peak-time events)	9
	Max. levels (single-channel search)	8
Graph window	Max. plots	1000



Chapter 3: Setup

3

Computer System

Table 3-1: Computer System Requirements

Minimum System Requirements	Recommended System Requirements
PC with 2 GHz Pentium class CPU[a]	PC with 2 GHz (or faster) Pentium class CPU[a]
Windows XP Pro (32-bit)	Windows 7 Pro (32- and 64-bit)
1.2 GB RAM	4 GB RAM or higher
CD-ROM drive (for installation)	CD-ROM drive (for installation)
1024 X 768 display system for pCLAMP Software 800 X 600 display system (small fonts) for Clampex Software	1680 X 1050 (or higher) display
1 USB 1 port (for security key dongle) 1 USB 2.0 port (for Digidata 1440A or 1550)	High-speed built-in USB 2.0 ports

[a] Multiple processor systems are not supported.

pCLAMP Software Security Key

If you install the pCLAMP 10.6 software, insert the provided pCLAMP 10 security key (dongle) into any USB port on your computer. The dongle must be connected to a USB port on your computer for pCLAMP 10 Data Acquisition & Analysis Software (Clampex Software) use. Dongles for any previous versions of the pCLAMP Software are invalid.

Note: If the key is not installed, Clampex Software runs in Demo mode only, restricting you to simulated data.

Signal Connections

Clampex Software uses the following BNC connections on the Digidata 1550A digitizer series:

Table 3-2: Clampex Software Digidata 1550A digitizer BNC Connections

Clampex Software Signals	Digidata 1550 BNC Sections	Digidata 1550 BNC Names
8 Analog OUT Channels	ANALOG OUTPUTS (Front)	0–7
8 Analog IN Channels	ANALOG INPUTS (Front)	
4 Telegraph Inputs (Gain, Frequency, C_m Capacitance)	TELEGRAPH INPUTS (Rear)	0–3
8 Digital Outputs	DIGITAL OUTPUTS (Front)	0–7
1 Digitizer START Input	(Front)	START
1 External Tag Input	(Front)	TAG
Scope Trigger Output	(Front)	SCOPE

Clampex Software uses the following BNC connections on the Digidata 1550 digitizer series:

Table 3-3: Clampex Software Digidata 1550 digitizer BNC Connections

Clampex Software Signals	Digidata 1550 BNC Sections	Digidata 1550 BNC Names
8 Analog OUT Channels	ANALOG OUTPUTS (Front)	0–7
8 Analog IN Channels	ANALOG INPUTS (Front)	
4 Telegraph Inputs (Gain, Frequency, C_m Capacitance)	TELEGRAPH INPUTS (Rear)	0–3
8 Digital Outputs	DIGITAL OUTPUTS (Front)	0–7
1 Digitizer START Input	(Front)	START
1 External Tag Input	(Front)	TAG
Scope Trigger Output	(Front)	SCOPE

Clampex Software uses the following BNC connections on the Digidata 1440A digitizer series data acquisition systems:

Table 3-4: Clampex Software Digidata 1440A BNC Connections

Clampex Software Signals	Digidata 1440A BNC Sections	Digidata 1440A BNC Names
4 Analog OUT Channels	ANALOG OUTPUTS (Front)	0–3
16 Analog IN Channels	ANALOG INPUTS (Front)	0–15
Telegraphs (Gain, Frequency, C_m Capacitance)	TELEGRAPH INPUTS (Rear)	0–3
8 Digital Outputs	DIGITAL OUTPUTS (Front)	0–7
1 Digitizer START Input	(Front)	START
1 External Tag Input	(Front)	TAG
Scope Trigger Output	(Front)	SCOPE

Analog Output Signals

Clampex Software uses an analog output channel from the digitizer to control the holding level and/or command waveform of an experiment. For example, the ANALOG OUT #0 channel would be connected via a BNC cable to a microelectrode current/voltage clamp amplifier’s External Command Input. If the amplifier has an internal command generator, be sure to switch to external command control. The other analog output channels can be used to control a separate holding level or output other command waveforms.

Analog Input Signals

The output signals from an amplifier connect to the digitizer’s analog input channels. With Clampex Software, these analog signals are digitized, displayed on the computer screen, and optionally saved to a data file on the hard disk.

Digital Inputs

The Clampex Software trigger inputs allow other instruments to externally trigger the start of acquisition, as well as to trigger the insertion of time, comment or voice tag information directly into the data file. A TTL-compatible digital input is required.

Digital Outputs

Clampex Software supports eight TTL-compatible digital outputs. All of these can be configured with bit patterns that coincide with the command waveform, enabling you control other instruments, such as a solution changer or a picospritzer. All eight can be configured with a holding pattern, or changed via a sequencing key. Clampex Software can also output a dedicated scope trigger to synchronize signal digitization with an oscilloscope.

Telegraphs

Clampex Software can be configured to receive “telegraphs” from many amplifiers, reporting such amplifier settings as the variable gain, lowpass filter, and whole-cell capacitance compensation.

Older model amplifiers have a BNC port for each type of telegraph they generate. These must be cable-connected to the digitizer. Digidata 1440A digitizer series and Digidata 1550 digitizer series have dedicated telegraph BNC ports. MultiClamp 700 and Axoclamp 900A amplifiers are computer-controlled, so telegraphs are passed directly to Clampex Software digitally, with no telegraph cabling required. See Telegraphs on page 49 for more about telegraphs in Clampex Software.

HumSilencer Adaptive Noise Cancellation System

The HumSilencer™ Adaptive Noise Cancellation System is integrated into Analog Input Channel #0 of the Digidata 1550A digitizer, so no additional hardware installation is required. When it is in use, it eliminates electrical hum interference at 50 Hz or 60 Hz and the associated high-frequency harmonics. The ON/OFF switch for the HumSilencer system is software-controlled. The HumSilencer system works by learning the noise patterns that are synchronous with the AC power, known as hum, caching a signal-averaged replica of the hum, and subtracting the noise replica from the signal in real time. It adapts to noise changes in about 1 second, within 50 power-cycles at 50 Hz or 60 Hz. It is designed to work with peak-to-peak hum amplitudes that, when combined with the signal of interest and other non-line-synchronous noise sources, are within the -10 to +10V range of the Digidata 1550A digitizer analog inputs. It is not a filter and does not have a filtering effect on acquired signals; nor does it cause waveform distortion, such as, frequency change, amplitude attenuation, phase shift, or DC-voltage shift.

Software Setup and Installation

Before inserting the pCLAMP Software CD, exit all other Windows programs.

pCLAMP Software Version 10.6 supports Digidata 1440A digitizer and newer.

Note: pCLAMP Software version 10.3 is the last software version to support the Digidata 1320 series digitizer. You must upgrade your digitizer hardware to use newer versions of pCLAMP Software.

Windows XP/7

Clampex Software 10.6 runs under Windows XP, and Windows 7 (32- and 64-bit), for the Pro, Enterprise, and Ultimate editions. The Home edition is not supported. The pCLAMP Software Setup program automatically detects which operating system is running and loads the correct files.

Note: A Windows 7 (32-bit or 64-bit) operating system is recommended.

Automatic CD Loading

When the pCLAMP Software CD is inserted into the CD-ROM drive, it automatically loads and displays the Setup program. This process can take several seconds to complete.

If you prefer to manually start the pCLAMP Software Setup program, use Windows Explorer to go to the CD-ROM drive, or alternate location, and then double-click the Setup icon.

Installing AxoScope Software or pCLAMP Software: Standard Installation

Note: Before installing the software from a CD-ROM, verify on the Molecular Devices support web site it contains the latest version. If you do not have the latest version, you can download it from the support web site www.moleculardevices.com/support.html.

Install the AxoScope Software Version 10.6 or pCLAMP Software Version 10.6, which includes the Digidata 1550 digitizer and Digidata 1550B digitizer drivers. An AxoScope Software Version 10.6 CD is included. If you have purchased the pCLAMP Software Version 10.6, install it instead of AxoScope Software Version 10.6.

To install the software:

1. Insert the AxoScope Software Version 10.6 or pCLAMP Software Version 10.6 CD into your computer CD-ROM drive.
2. The setup dialog is displayed automatically; if not, use Windows Explorer to open the CD directory and double-click the **AxoScope_10_6_0.exe**, or **pCLAMP_10_6_0.exe** file. The installation menu appears.
3. Follow the on-screen instruction to install the software.
4. For Clampex Software to run properly, you may need to restart the computer. Before rebooting, if applicable, remove the pCLAMP Software CD from the drive.

Uninstalling pCLAMP Software Version 10.5

To uninstall pCLAMP Software Version 10.5:

1. Go to Windows **Start > All Programs >Molecular Devices> pCLAMP 10.5**.
2. Select **Uninstall pCLAMP 10.5 Software**.

File Locations

File locations depend on the software version.

- pCLAMP Software Version 10.x:

  User-related files, such as data and parameter files, are stored in their own folders in:

  \Documents and Settings\[user name]\My Documents\Molecular Devices\pCLAMP\...
- pCLAMP Software Versions 10.3 through 10.6:

  System-related files, such as for the Lab Bench, System Lab Book, and user-defined telegraphs, are stored in the hidden folder:

  C:\ProgramData\Molecular Devices\pCLAMP\

  Program application files are stored by default in the folder:

  C:\Program Files\Molecular Devices\pCLAMP 10.6
- pCLAMP Software Version 10.2 and earlier:

  Program application files and system-related files, such as for the Lab Bench, System Lab Book, and user-defined telegraphs, for are stored in:

  C:\Axon\pCLAMP X

Digitizer Configuration in Clampex Software

After you have installed pCLAMP Software and connected the digitizer to the computer, you must configure Clampex Software to communicate with the digitizer. This is done from the **Configure > Digitizer** dialog.

Detailed instructions on digitizer configuration are included in the *Digidata® 1550B Low-Noise Data Acquisition System plus HumSilencer™ Adaptive Noise Cancellation User Guide* and in the tutorial *Setting Up Clampex for Data Acquisition*, which was loaded onto the computer during the pCLAMP Software installation. Open the tutorial from the **Start > All Programs > Molecular Devices > pCLAMP 10.6 > Clampex 10.6 Tutorial**. The instructions here summarize those provided in the tutorial.

Demo Mode

When Clampex Software is first installed, it is in "Demo" mode, allowing you to experiment with the program without being connected to a digitizer. The demo digitizer is like having a built-in signal generator. It creates signals derived from episodic protocols and adds noise, making it perfect for creating sample data files. Or, from the **Configure > Digitizer** dialog with "Demo" selected, click **Configure** to alter the demo data output in non-episodic acquisition modes.

Configuring the Digidata 1550B Digitizer in the Clampex Software

Connect the digitizer to the computer. If you have a Digidata 1550B digitizer, the **Windows Found New Hardware Wizard** is displayed. Work through the prompts until the installation completes. No separate driver disk is needed, so it is recommended that you automatically search the hard disk for the driver.

Note: Configuring Digidata 1440A digitizer in Clampex Software requires the same procedures, except the **Digitizer Type** selection varies.

To configure the Digidata 1550B digitizer in Clampex Software:

1. Start the Clampex Software by clicking **Start > All Programs > Molecular Devices > pCLAMP 10.6 > Clampex Software 10.6**.
2. Select **Configure > Digitizer** dialog and click **Change**.
3. From the **Change Digitizer** dialog, select **Digitizer Type > Digidata 1550B Series**.
4. Click **Scan** to detect the digitizer. The first detected digitizer is assigned **0** and listed as **Available**. The **Configuration** changes from **Not present** to reporting the selected digitizer model number, serial number, firmware version, HumSilencer channel availability (0, 1, or 4), and the **OK** button is enabled.
5. Click **OK** to exit the dialog, then click **OK** to exit the **Digitizer** dialog.

The front panel **READY** light is continuously on only when the software connects to the digitizer. After the Digidata 1550B digitizer warms up (allow one hour), it is ready to do experiments.

If you receive an error message, see the Troubleshooting chapter of the *Digidata® 1550B Low-Noise Data Acquisition System plus HumSilencer™ Adaptive Noise Cancellation User Guide*.

HumSilencer Adaptive Noise Cancellation System

The HumSilencer system controls display only when connected to and configured with a HumSilencer Adaptive Noise Cancellation System-enabled Digidata 1550B digitizer. Not all Digidata 1550B digitizers have the HumSilencer Adaptive Noise Cancellation System (ANC). A *HumSilencer Options* sticker on the back of your digitizer specifies the number of ANC Inputs available.

Note: The HumSilencer Adaptive Noise Cancellation System controls do not display in the **Configure > Digitizer > Demo** configuration, nor is there a simulation mode for adaptive noise canceling.

When the Clampex Software starts, the HumSilencer system automatically starts learning the line-frequency noise, until you disable the learning. See Setting Inputs and Outputs on page 54.

MiniDigi Digitizer Installation

The optional accessory MiniDigi digitizer works with AxoScope Software only. Clampex Software does not support the MiniDigi digitizer.

To configure the MiniDigi digitizer in AxoScope Software:

1. Run the pCLAMP 10.6 installer before you connect the MiniDigi digitizer to the computer.
2. After pCLAMP has been installed, connect the USB cable to the USB port on the computer and to the MiniDigi digitizer.
3. In the **Windows Found New Hardware Wizard**, follow the instructions until Windows has installed the digitizer.
4. Start AxoScope Software by clicking **Start > All Programs > Molecular Devices > pCLAMP 10.6 > AxoScope 10.6**.
5. Open the **Configure > Digitizer** dialog and click **Change**.
6. In the **Change Digitizer** dialog, select **MiniDigi** from the **Digitizer Type** field.
7. Click **Scan** to detect the digitizer. **Available** is displayed, and **OK** is enabled.
8. Click **OK** to exit this dialog.
9. Click **Configure** to open the **Configure MiniDigi** dialog.
10. Select the style of filtering to use.
    - **Analog filtering** applies a low-pass, anti-aliasing filter with a cutoff frequency one fifth of the sampling rate.
    - **MinMax filtering** takes the minimum and maximum samples in every n samples, where n is determined by the sampling rate.
11. To calibrate the MiniDigi digitizer, attach a grounding plug to the Channel 0 BNC, and then click **Start**.
12. Repeat for Channel 1.
13. Click **OK** to exit this dialog.

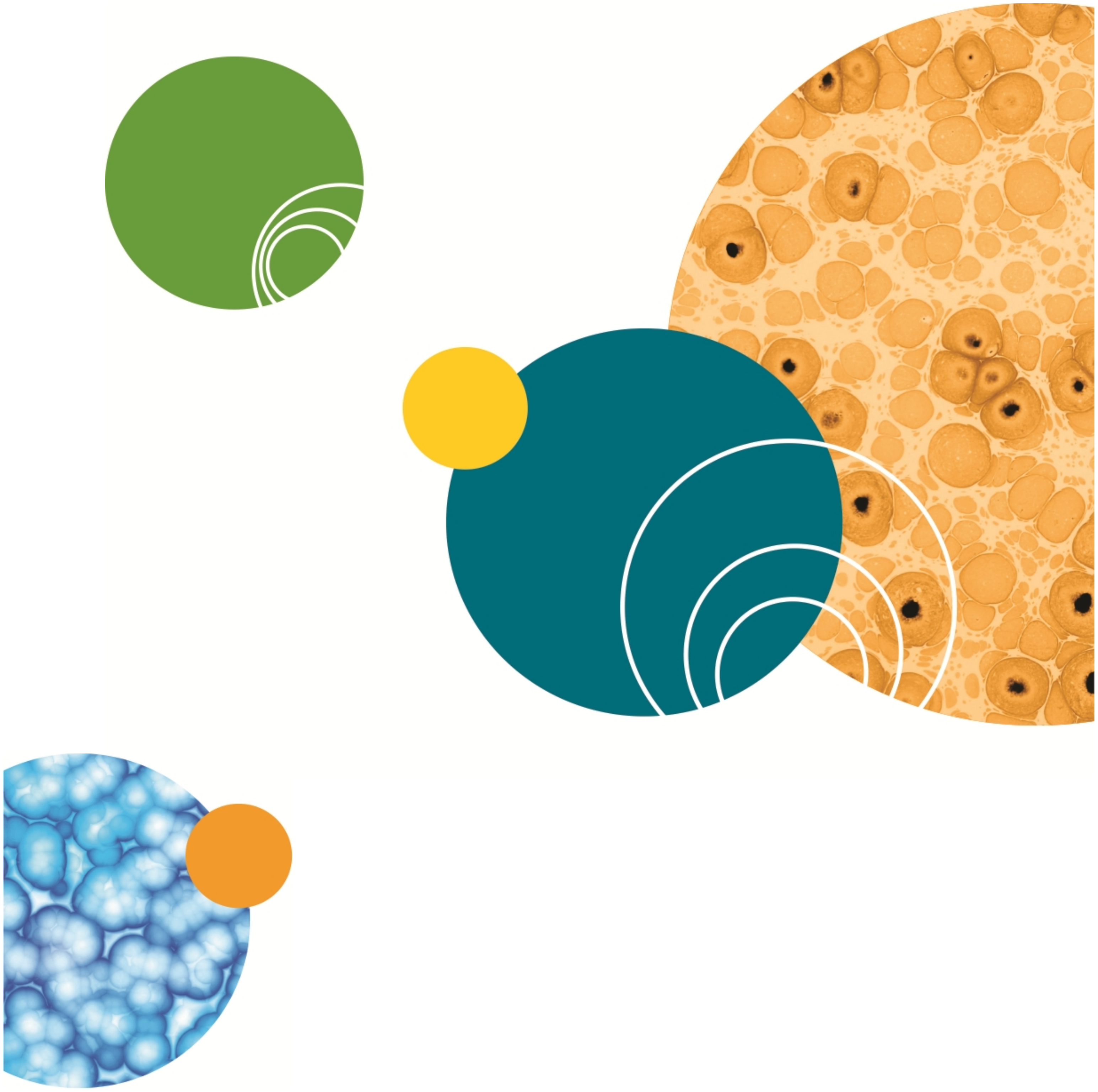
The MiniDigi digitizer is now ready to do experiments.

Resetting Program Defaults

The **Start > All Programs > Molecular Devices> pCLAMP 10.6** folder contains the utility **Reset to Program Defaults**, which resets pCLAMP Software settings back to their default values. Settings for other programs can be displayed in the list of registry items. Select the items relevant to your situation.

Printing

pCLAMP Software supports all printers and plotters that have been installed through the Microsoft Windows operating system.



Chapter 4: Clampex Software Features

4

This chapter introduces the major features of Clampex Software, including an extended discussion of the LTP Assistant. Having read the chapter, the new user can reinforce and extend their understanding by working through the tutorials in Clampex Software Tutorials on page 87. More detailed information on the features discussed in this chapter is contained in the Clampex Software Help file.

Clampex Software Windows

Clampex Software has a standard Windows format, with title bar, menus, and toolbars at the top, and a status bar along the bottom. As is typical of many Windows applications, you can choose which toolbars to display (from the **View** menu) and select the toolbuttons to appear in the toolbars (**Configure > Toolbars**). There is one dockable component—the Real Time Controls. By default, this opens as a panel attached to the left-hand side of the main window, but can be dragged away to be repositioned like a standard dialog box, or attached to the right-hand side of the main window.

All commands available in Clampex Software are included in the main menus, although many of them have toolbuttons or can be accessed from right-click popup menus as well. The main menu contents differ according to which of the windows (below) is highlighted.

Besides the main window, Clampex Software has seven window types:

- Analysis
- Data File Index
- Lab Book
- Membrane Test
- Results
- Scope
- Statistics

Within the main Clampex Software window, these windows can be maximized, minimized, resized and tiled. Right-clicking in each brings up a popup menu with options specific to the window, including in each case one that opens a **Properties** dialog box. This allows users to configure the general appearance of the window, and these settings can then be saved as defaults (**View > Window Defaults**).

The basic roles of each of the window types are as follows (see the application help for more details).

Analysis Window

The Analysis window displays data that have been saved in a file, for review and measurement. The data are displayed graphically, as traces, with sub-windows for each signal stored within the file. Open a file in an **Analysis** window from **File > Open Data**.

Data can be displayed as Sweeps, or in Continuous or Concatenated modes, controlled from **View > Data Display**. In sweep mode you can choose to view any subset of the sweeps (**View > Select Sweeps**), and toggle between viewing all the sweeps at once, a selected subset, or just one at a time (**View > Toggle Sweep List**). When more than one sweep is visible, step through them by highlighting one at a time with the "<" and ">" keys.

Up to sixteen "cursors"—vertical, repositionable lines—are available to assist in making simple measurements of the active sweep. Cursor text boxes display (optionally) time, amplitude and sample number, or delta values relative to a paired cursor. Configure these and other cursor options by double-clicking on a cursor to open the Cursor Properties dialog box. A number of measurements for the first two sets of cursors, and the sections of trace they bound, can be quickly sent to the Results window using the and buttons in the top-left of the Analysis window.

Data File Index

The Data File Index (DFI) window is a file management tool that allows you to construct an index of data files. Data files can be grouped together in separate DFI files, and then sorted according to a wide range of parameters reported for each file. These options give you great flexibility in being able to rapidly organize and find files from particular types of experiments—especially valuable in managing large amounts of data from archival sources such as CD-ROMs. Create a Data File Index from **File > New Data File Index**.

Lab Book

The **Lab Book** window is a text editor for logging events that occur while Clampex Software is running, for example that a protocol is opened, or when the holding level is changed. Events can be automatically written to it (see **Configure > Lab Book Options**) or add your own comments with the **Tools > Comment to Lab Book** command or by typing directly in the Lab Book. Several of the tools within Clampex Software offer the option of writing values to the Lab Book, for example the Membrane Test.

There is always a Lab Book window open, called the **System Lab Book**. Copies of this can be saved to disk for editing or archiving elsewhere.

Membrane Test

Membrane Test is a sophisticated utility for monitoring a number of parameters during the three stages of a patch-clamp experiment:

- **Bath**: Electrode resistance in the bath before patching (formerly called “Seal Test”).
- **Patch**: Patch resistance, to assist you in forming a gigaohm seal.
- **Cell**: Cell resistance and membrane capacitance.

You can switch between these three stages using the Stage buttons at the top of the dialog box. When you switch from one stage to another the current parameter values are recorded in the Lab Book. A selection of the following electrode and cell membrane properties are reported, depending on the current stage:

- Total resistance, Rt
- Access resistance, Ra
- Membrane resistance, Rm
- Membrane capacitance, Cm
- Time constant, Tau
- Holding current, Hold

Results Window

The Results window contains a spreadsheet for the display of measurements derived from cursors 1 and 2, and 3 and 4, in the Analysis window. These measurements include time and amplitude minimums, maximums, deltas, average slope, mean and standard deviation. You can only select contiguous rows and columns to perform standard copy and paste operations.

Like the System Lab Book, a Results window is kept open whenever Clampex Software is running. There can be only one Results file open at any one time (though you can view it in more than one window if you use the **Window > New Window** command). The Results window can be saved as a separate file at any time, and opened into pCLAMP Software.

Scope Window

The Scope window displays digitized data in real time during data acquisition (both View Only and Record). You can optionally use the **Acquire > View Only** command to preview data without writing it to disk, and then **Acquire > Write Last** to save the data to a file.

Multiple Scope windows can be opened with **Window > New Scope**. This can be useful for viewing incoming data at various magnifications, or with different display options.

In Episodic-mode acquisition, when Statistics have been enabled (from the **Edit Protocol > Statistics** tab), cursors define search region and baseline boundaries, and significant data points of the statistics measurements are marked with symbols in the Scope window. For gap-free or event-detected modes, trigger thresholds (**Edit Protocol > Trigger** tab) are shown with adjustable horizontal markers.

The **View > Store Sweep** command preserves the last acquired sweep in Episodic and Oscilloscope modes, so you can easily compare a particular sweep to other data.

Statistics Window

Statistics windows are graph-format windows, with a time X axis and subwindows for each of the statistical measurements recorded. Measurements from up to eight different search regions (configurable within each sweep for high-speed oscilloscope and episodic stimulation acquisition modes) are color-coded within the separate subwindows. A pane in the right-hand side of the window shows the search region color legend, and reports the most recent data in numerical form.

When statistics are recorded in Clampex Software, these are drawn into the Online Statistics window. This special Statistics window becomes active, accepting data, as soon as any statistics are generated. It continues to be active until Clampex Software is closed, recording all statistics measured. Once open, the **Online Statistics** window cannot be closed, though it can be minimized. It continues to accept data in either state. New subwindows are automatically added to the window for each new type of statistics measurement enabled.

Once activated, the Online Statistics window can be cleared of the data it contains and the X axis reset to begin at time zero (**Edit > Clear Statistics**). This also resets the number of subwindows, to those enabled in the currently loaded protocol. Before clearing, or at any other time, you can save the contents of the Online Statistics window into a standard statistics (STA) file. These files can then be opened into their own Statistics window (in both Clampex Software and pCLAMP Software) with the **File > Open Other > Statistics** command.

Measurements are written to the Online Statistics window when:

- In episodic and oscilloscope modes, Shape Statistics is enabled (**Edit Protocol > Statistics** tab).
- In gap-free and event-triggered modes, Threshold-based Statistics is enabled (**Edit Protocol > Trigger** tab).
- In episodic stimulation mode, the epoch Resistance Test is enabled (**Edit Protocol > Wave** tab).
- **Tools > Membrane Test** is run.

Telegraphs

For many amplifiers, Clampex Software can receive and incorporate a range of “telegraphed” amplifier settings. Depending on the type of amplifier you have, the variable gain, lowpass filter, and whole-cell capacitance compensation settings can be telegraphed, with Axoclamp 900A and MultiClamp amplifiers telegraphing amplifier mode and signal scale factors and units as well. Gain telegraphs automatically update the signal scaling of the input channel, based on changes to the amplifier’s gain knob. Lowpass filter telegraphs and capacitance compensation telegraphs store these settings in the data file header. With the Axoclamp 900A and MultiClamp amplifiers, amplifier mode changes can be linked to sequencing keys, so that, for example, Clampex Software automatically loads a suitable protocol when you switch between voltage or current clamp modes. The scale factor and units telegraphs from the Axoclamp 900A and MultiClamp amplifiers almost entirely automate signal setup in Clampex Software, leaving you only to name the input signals.

Note: Computer-controlled amplifiers have many of the fields grayed out because the amplifier software takes direct control of these parameters.

Clampex Software telegraphs are configured in the **Configure > Telegraphed Instrument** dialog:

- If using a computer-controlled amplifier such as the MultiClamp 700 or Axoclamp 900A, first turn on the amplifier and launch the corresponding Commander software.
- Select the digitizer input channel on which you receive the signal that the telegraphed information is relevant to.
- Select the amplifier type, which determines the configuration options you are offered. If the amplifier does not support telegraphs you can manually enter the information that would otherwise be telegraphed. To do this:
  - Select **(Manual)** as the telegraphed instrument in the configuration dialog.
- Enter the appropriate settings in the **Configure > Lab Bench > Input Signals** tab. The section of the tab where these entries are made reports telegraphed values in the case of normal telegraphing. Telegraphed filter, gain and capacitance compensation settings are also reported in the Real Time Controls.

Lab Bench

When setting up Clampex Software for data acquisition, having first configured the digitizer (Setup on page 35) you must configure input and output signals for it. This is done in the **Lab Bench** (**Configure > Lab Bench**). Clampex Software allows you to define several different signals for each digitizer channel. For each signal you need to set units and scaling so that Clampex Software displays data properly corrected for the parameter being read. Then, when you set up an acquisition protocol, you select the appropriate channels and signals from the **Input** and **Output** tabs in the **Acquire > Edit Protocol** dialog.

For the Analog IN #0 and Analog OUT #0 channels, Clampex Software has a number of predefined signals that are properly scaled for a variety of Axon Instruments amplifiers, but you are able to configure virtually any type of signal that you want, for any of the input or output channels.

If you are setting up your own signals, the **Scale Factor Assistant** helps calculate the correct scale factor. It asks a few basic questions, usually answered by reading the values from the front of the amplifier, and then computes the appropriate scale factor.

The Lab Bench has a tab each for input and output signals.

Setting Up Input Signals

Use the **Input Signals** tab for non-computer-controlled amplifiers to:

- Define units of measurement and any signal offsets.
- Define the scale factor by using the **Scale Factor Assistant**.
- Enable software filtering.
- Add additional gain or amplification to the signal before it is digitized.
- Enter the appropriate settings in the **Configure > Lab Bench > Input Signals** tab. The section of the tab where these entries are made reports telegraphed values in the case of normal telegraphing. Telegraphed filter, gain and capacitance compensation settings are also reported in the Real Time Controls.

If you configure a telegraphed instrument, amplifier settings such as output gain, filter frequency, and membrane capacitance values are also displayed.

Setting Up Output Signals

Use the **Output Signals** tab to:

- Define output signal units and scale factors (using the **Scale Factor Assistant**)
- And depending on the options you have selected in the **Overrides** dialog box (**Configure > Overrides**)
    - Set the holding level
    - Set digital OUT channels

Overrides

The **Overrides** dialog box (**Configure > Overrides**) gives you the option of switching the control of various parameters, normally set in the Protocol Editor, to other locations. Of most immediate interest here are the options for analog and digital holding levels. If the Overrides options for these are left unchecked, they are defined, protocol by protocol, in the Outputs tab of the Protocol Editor. If checked, then general levels are set from the Outputs tab in the Lab Bench. These levels then apply regardless of which protocol is being run. Whichever location has control, immediate changes can be made to holding levels from the Real Time Controls.

Handling Data

File Names

Before real data are recorded you should select a naming regime for new data files, and choose the default directory for these to be saved to. Both settings are made in the **File > Set Data File Names** dialog box. You can select date-based naming or choose a user defined prefix, and both options have long and short forms.

By default, data files are saved per user in **My Documents\Molecular Devices\pCLAMP\Data** and protocols are saved per user in **My Documents\Molecular Devices\pCLAMP\Params**.

After completing an experiment, you can inspect and edit the data by selecting **File > Last** Recording opens the most recently saved file in an Analysis window. The file opens using the display defaults which have been selected in the **File > Open Data** menu item.

Save As

The original data file from a recording contains all data from all signals configured in the protocol under which it was recorded. You can save selected sections from such a file, defining a region to save with the cursors, and selecting just those sweeps and signals that you want included.

To save selected sections:

1. Select **View > Window Properties > Show/Hide** to specify which signals remain visible.
2. Select **View > Select Sweeps**.to specify which sweeps remain visible.
3. Select **File > Save As > Options** to change the **Save As** dialog settings.

Edit Protocol

The Protocol Editor is of central importance to Clampex Software, because it is where the greater part of experimental configuration occurs. Opened with either of **Acquire > New Protocol** or **Edit Protocol**, it has options for setting up all the various aspects of data acquisition: the acquisition mode, trial length or hierarchy, sampling interval, the channels and signals that will be used, the shape of the command waveform, and whether or not to use adaptive noise cancellation, adapt between sweeps, triggering, statistics measurements, leak subtraction, pre-sweep trains, or math channels.

When a complete range of acquisition settings has been configured, this can be saved together as a protocol file (**Acquire > Save Protocol As**), and reopened later (**Acquire > Open Protocol**) as needed.

Setting the Mode/Rate

Once the hardware is set up and signals have been configured in the Lab Bench, the first step in setting up an experiment is to choose an **Acquisition Mode** (see Data Acquisition on page 60). This must be done first because many of the options change depending on the mode selected. In the **Edit Protocol** dialog, from the top section of the **Mode/Rate** tab, select an **Acquisition Mode**. Options include:

- Gap-free
- Variable-length events
- Fixed-length events
- High-speed oscilloscope
- Episodic stimulation

Trial Length

For **Acquisition Modes** other than **Episodic stimulation** you must select options for **Trial Length** on the **Mode/Rate** tab. Options include:

- **Use available disk space** - or until the data file reaches a 4 GB limit
- **Duration**

To help set these options, the amount of free space on the hard disk is reported on the tab, both in megabytes and in terms of the amount of recording time this gives you at the current settings. The time available is inversely related to the number of channels sampled and the sampling rate, so, if disk space is an issue, one way to free some up is to decrease the sampling rate. Another way to save disk space, if possible, is to use the **Acquisition Mode > Fixed-length events** or **Variable-length events**, instead of **Gap-free**.

Trial Hierarchy

When **Episodic stimulation** is selected the **Mode/Rate** tab has options for **Trial Hierarchy**. You need to enter the **Trial delay**, **Runs/trial**, **Sweeps/run**, **Sweep duration**. If you intend having more than one input channel, the Sweep duration that you enter is the same for each signal.

If you are keeping an eye on disk space, the file size—for files created under the protocol at its current configuration—is reported beside the **Sweeps/run** field. The number of runs per trial does not affect the file size—each run in a trial is averaged and only one final, averaged run is saved. The total amount of free disk space is reported at the bottom of the tab, both in megabytes and as the number of sweeps.

A breakdown of the sweep into **First holding** and **Last holding** periods, and **Epochs**, is also provided. This refers to the command waveform, where the First and Last holding periods are each automatically calculated as 1/64th of the specified sweep duration, during which time only the holding level is output. The command waveform is simultaneously output while recording data according to the hierarchy you configure here. The waveform is produced in each sweep, and the **Epochs** value reported here is the range of time available for you to configure its shape.

Sampling Rate

A trial’s sampling rate is also set in the **Mode/Rate** tab of the **Edit Protocol** dialog. The sampling rate is set per signal, so even if you acquire multiple signals, each signal has the displayed sampling rate.

The total throughput of the data acquisition system, for example of [sampling rate] x [number of signals] is displayed beneath the sampling interval field for non-episodic modes, and at the bottom-right of the tab for **Episodic stimulation**. This sampling rate is also set per signal in the generation of the command waveform.

pCLAMP supports split-clock acquisition, meaning that you can define two different sampling rates: a **Fast rate** and a **Slow rate**. You can specify when to use these rates in the **Sample rate** row on the **Waveform** tab.

Start-to-Start Intervals

In **Mode/Rate > Episodic stimulation > Start-to-Start Intervals** allows you to time the start of each sweep and/or run relative to the start of the one before it. This means that a **Start-to-Start Interval** must be equal to or longer than the sweep length or the total time of a run. **Pre-sweep Train**, **P/N Leak Subtraction**, **Membrane Test Between Sweeps**, and **Adapt between sweeps** are included in calculating the sweep length. If you just want the computer to execute sweeps or runs as quickly as possible, use the **Minimum** setting. The **Minimum** setting results in no lost data between sweeps, producing a continuous record divided into sweeps.

If a **Trigger > Trigger source** is selected to start a sweep, the **Start-to-Start Intervals** are disabled, because timing is now controlled externally.

If **Stimulus > Adapt between sweeps** is selected, 1.081 seconds is added to **Start-to-Start Intervals**.

Averaging

In **Mode/Rate > Episodic stimulation**, **Averaging** is enabled when more than one **Trial Hierarchy > Runs/trial** is specified. It is also an option in **High-speed oscilloscope** mode.

Two types of **Averaging Options** are available: **Cumulative** and **Most Recent**. In Cumulative averaging, each run contributes the same weighting into the average. As runs are accumulated, each successive run contributes a smaller percentage to the average, which becomes increasingly resistant to change. This option is recommended for data with fairly stable baselines.

In **Most Recent** averaging, only the last N runs are used to calculate the average, so the average is sensitive to changes in the data. This is recommended for data with significant baseline drift.

Up to 10,000 runs can be averaged. The average is updated with every run, but changes to the average can be removed allowing you to revert to an earlier saved average. If bad data are received midway through a trial, you can still save some results by reverting to the last average before the undesirable data were recorded. Options for this are set in the **Averaging > Options > Undo File**.

Setting Inputs and Outputs

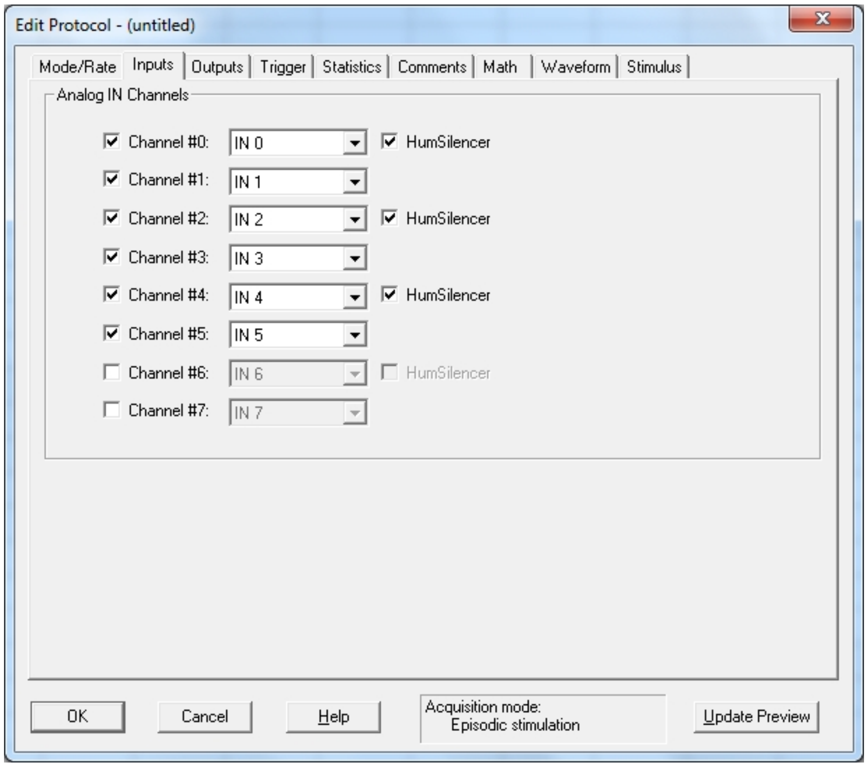
In the **Edit Protocol** dialog, in the **Inputs** tab, is where you select the number of input channels for receiving data, and also where you select the signals for each of these channels. These are the signals that were configured in the **Configure > Lab Bench Inputs** tab.

On the **Outputs** tab you similarly select signals for the output channels. You are also able to set both analog and digital holding levels on this tab. The levels you set here are specific to the protocol you are configuring. If the holding level fields simply report their values and cannot be adjusted, that means that you have set the **Configure > Overrides** options to give control of these parameters to the **Lab Bench**.

In setting analog holding levels, remember that levels set in Clampex Software are in addition to those set on the amplifier. Many users have their amplifier holding level control turned off and control this entirely from Clampex Software.

Enabling the HumSilencer System Controls

When you use a digitizer with the HumSilencer adaptive noise cancellation system, either a Digidata 1550A digitizer or Digidata 1550B digitizer, to use the software-based controls, you must first turn them on in the software by selecting the **HumSilencer** check box in the **Edit Protocol > Inputs** dialog tab for an **Analog IN Channel** (Figure 4-1). A blank check box indicates that the controls for the HumSilencer system in the Real Time Controls panel are disabled.



Analog IN Channels

✓ Channel #0:	IN 0	✓ HumSilencer
✓ Channel #1:	IN 1	
✓ Channel #2:	IN 2	✓ HumSilencer
✓ Channel #3:	IN 3	
✓ Channel #4:	IN 4	✓ HumSilencer
✓ Channel #5:	IN 5	
☐ Channel #6:	IN 6	☐ HumSilencer
☐ Channel #7:	IN 7	

Acquisition mode:
Episodic stimulation

Figure 4-1: Edit Protocol > Input dialog tab check box used to enable the Real Time Controls for the HumSilencer system

After enabling the **HumSilencer** controls in this **Edit Protocol > Inputs** dialog tab, you turn on and turn off the adaptive noise cancelling in the Real Time Controls panel by selecting or deselecting the **HumSilencer > Subtract** check box. See Using HumSilencer Adaptive Noise Cancellation Controls on page 62.

Note: The Digidata 1550A digitizer provides the **HumSilencer** controls only for **Analog IN Channel #0**.
The Digidata 1550B digitizer provides the **HumSilencer** controls only for **Analog IN Channels #0, #2, #4, and #6**.

Setting Triggers

Data acquisition triggering is set in the **Edit Protocol** dialog **Trigger** tab.

Your selection from the Start trial with field determines the way that you tell Clampex Software you are ready for data viewing or recording. This is always initiated in the first case by selecting **Acquire > Record or Acquire > View Only** (or clicking the equivalent icon buttons in the Acquisition Toolbar). Thereafter, the option you selected in **Start trial with** takes effect.

The trial can be set to start immediately, or on receipt of some external or keyboard signal. Rather than selecting a signal by name, you may want to go straight to the first signal enabled on the **Input** tab, with the selection of the **First Acquired Signa**l option.

Once a trial has been started, Clampex Software can be set to await a trigger before any data are recorded. In **Acquisition Modes** other than **Episodic stimulation**, you can select a specific input signal and then set a threshold level for it.

In all **Acquisition Modes**, except **Gap-free**, in the **Trigger** tab of the **Edit Protocol** dialog, when the **Trigger source** is set to **Digitizer START Input**, the **Timeout (ms)** field displays. Use it to set the maximum time the system polls for an external **START** trigger before it checks for user-intervention, such as clicking **Stop**, and responds to the intervention or resumes polling. Set this polling duration between 1000 ms to 60000 ms, making sure it is longer than your trigger interval.

The **Pretrigger length** field allows you to set how much of the trace, before a threshold crossing, you want recorded. This setting is also used for the period that recording continues at the end of an event in Variable-length events mode. To avoid noisy signals causing false triggers you can refine trigger settings in the **Hysteresis** dialog.

In **Episodic stimulation** mode, **Trigger source > Internal Timer** causes the runs and sweeps in the trial hierarchy to be controlled by the settings in **Mode/Rate > Start-to-Start Interval**.

Users can also enable a digital trigger signal from this tab. This uses the SCOPE front panel BNC on the Digidata 1550 digitizer and the Digidata 1440A digitizer to send a trigger to devices such as an oscilloscope. When enabled, the port outputs a 5 V TTL signal to coincide with data acquisition.

If you are in an acquisition mode other than Episodic stimulation, the trigger remains active for the duration of the time the signal remains over (or under, when you have selected negative polarity) the threshold level, selected in the **Trigger Settings** or **Threshold-Based Statistics** settings section immediately below the check box. Otherwise, in **Episodic stimulation** mode, this trigger is always output, and is held active from the start of the first holding period in each sweep until the start of the last holding period in each sweep.

Threshold-Based Statistics

When acquiring in **Gap-free**, **Variable-length events**, or **Fixed-length events** modes, you can monitor measurements such as the event frequency or percentage of time above the threshold level. These threshold-based statistics are set up in a protocol’s **Trigger** tab, and use the same setting as for the **Trigger source**.

When threshold-based statistics are enabled, the statistical data are recorded in, and can be saved from, the **Online Statistics** window. You can also choose to have the data automatically saved in a statistics file when a recording finishes.

Threshold-based statistics can serve as a handy indicator of the progress of an experiment. For example, it might be expected that in the presence of a certain drug, the channel activity will increase. If so, the percentage of time spent above threshold is a measure of this increased activity, and can be used as a criterion for continuing the experiment.

Setting Statistics

Available in **Episodic stimulation** and **High-speed oscilloscope** modes, **Shape Statistics** measure various parameters of evoked events such as peaks, areas, slopes and rise times, set in the **Statistics** tab of the **Edit Protocol** dialog.

Shape statistics can be measured from any of the input signals. Searches for different measurements can be configured for up to eight different search regions within the sweep.

After you start digitizing data, you see vertical cursor lines bounding the search regions in the **Scope** window. These are numbered according to search region, and the region boundaries can be reset during data acquisition by dragging the cursors to new positions.

Shape statistics are written to the **Online Statistics** window, with options to automatically save the data from each trial to their own statistics file, and to clear the Statistics window after each trial.

Setting Math Signals

Two analog input signals can be arithmetically manipulated and displayed as a separate online signal in the **Scope** window. This is set up in the **Math** tab of **Edit Protocol** dialog. Two signals can be scaled, offset, and arithmetically combined before being displayed, by using **Equation > General purpose**. Or, **Equation > Ratio dyes** can be used to measure cellular dye concentrations by ratioing the fluorescence signals from two photomultiplier tubes.

Setting Waveform

The waveform outputs are available only in **Episodic stimulation** mode. You can define analog and digital waveforms in the **Waveform** tab of the **Edit Protocol** dialog.

Up to eight analog stimulus waveforms can be generated simultaneously, one for each of the eight output channels tabs at the bottom of the dialog. Simultaneously, eight digital outputs can be enabled on one of the output channels. Epoch-driven waveforms are defined in an **Epoch Description** table. This consists of up to ten **Epochs** (sections of the sweep) preceded and followed by a holding-level period that is predefined as 1/64 in duration of the sweep length. For **Analog Waveform**, each of the epochs can be configured to step to and hold a particular voltage, to increase or decrease linearly in a ramp, or to produce a rectangular, triangular, sinusoidal or biphasic wave train. Click in the **Type** row for the epoch you want to configure, to assign one of these options. Amplitudes and durations of each epoch can be systematically increased or decreased by setting delta values, for example, the fixed amounts of change in a parameter with every sweep.

Train outputs can be configured for both analog and digital outputs. Trains can be generated as square pulses, sawtooth pulses, biphasic pulses and a cosine wave. A digital bit pattern can be specified with binary numbers (0, 1), and individual digital bits are enabled for trains by using the * symbol.

For more flexible control of the analog waveform, see User List. If the epoch-based analog waveform is still not flexible enough for your needs, the **Stimulus file** option can be used to read data from an ABF or ATF file and output it as an analog waveform.

While you are building the command waveform, you can see how it looks by clicking the **Update Preview** button in the bottom-right of the Protocol Editor. This opens an **Analysis** window displaying the waveform. You can keep this window open while you make changes in the **Epoch Description** table, clicking **Update Preview** again to see the latest changes made.

Setting Stimulus

Available only in **Episodic stimulation** mode.

Pre-sweep Trains

A train of pulses can be used to “condition” a cell prior to the main stimulus waveform. This is configured in the **Stimulus** tabs of the **Edit Protocol** dialog.

The Clampex Software pre-sweep train outputs analog pulses composed of repeated baseline and step levels. After the pre-sweep train is completed, the output can be held at a post-train level. If the number of pulses in the train is set to zero, only the post-train level is generated, which is a convenient way to change the holding level for a specified duration before each sweep.

The pre-sweep train can be output on either the same analog out channel as the stimulus waveform, or on the other analog out channel. Data are not acquired during the pre-sweep train period, so you cannot observe its effects in Clampex Software. To view these, run AxoScope Software concurrently in **Gap-free** mode with a MiniDigi digitizer.

P/N Leak Subtraction

P/N Leak Subtraction is a technique typically used in voltage-clamp experiments to correct for a cell’s passive membrane current, for example the leakage current. It is configured in the **Edit Protocol** dialog **Stimulus** tab.

In **P/N Leak Subtraction**, a series of scaled-down versions of the command waveform is generated, and the responses are measured, accumulated, and subtracted from the data. Scaled-down versions of the waveform are used to prevent active currents from being generated by the cell. The number of these scaled-down waveforms, entered as the number of subsweeps, is the “N” referred to in the name of this technique. The waveform (pulse P) is scaled down by a factor of 1/N, and this is applied N times to the cell. Since leakage current has a linear response, the accumulated responses of the subsweeps approximates the leakage current for the actual waveform. This is subtracted from the input when the actual waveform is run.

For added flexibility in preventing the occurrence of active currents, the **Polarity** of the P/N waveform can be reversed. In this case, the P/N accumulated response is added to the sweep of data. Also, to prevent a conditioning response from the cell, the P/N waveforms can execute **After** the main stimulus waveform.

In Clampex Software, both the raw data and the P/N corrected data are saved. The signal name for the corrected data channel is modified by a 'c' appending or replacing the last character of the signal name. During data acquisition, only one of these is displayed.

User List

The **User List**, on each of the **Stimulus** tabs, provides a way of customizing one of a range of analog and digital output features, overriding the generalized settings made elsewhere in the **Edit Protocol** dialog. The selected parameter can be set to arbitrary values for each sweep in a run by entering the desired values in the **List of parameter values** field.

So, for example, rather than having sweep start-to-start times remain constant for every sweep in a run, specific sweep start-to-start times can be set for each sweep. Alternatively, rather than being forced to increase or decrease the duration of a waveform epoch in regular steps with each successive sweep by setting a duration delta value, you can set independent epoch durations for each sweep. All aspects of the conditioning train, the number of subsweeps in **P/N Leak Subtraction**, and command waveform amplitudes and durations, besides other output features, can be overridden from the **User List**.

Membrane Test Between Sweeps

The **Membrane Test Between Sweeps** check box enables a Membrane Test to be run automatically between every sweep. In this mode the Membrane Test data are written out to the **Online Statistics** window and displayed in the Real Time Controls.

Adapt Between Sweeps

This feature is only available with a HumSilencer capable digitizer, either a Digidata 1550A digitizer or Digidata 1550B digitizer. Use this function to enable inter-sweep learning while using the HumSilencer Adaptive Noise Cancellation System during long experiments to counteract the potential of noise pattern changes over time.

The **Adapt between sweeps** check box is available from the **Stimulus** tab in the **Edit Protocol** dialog during the configuration of an Episodic stimulation acquisition protocol. The inter-sweep learning duration is 1.081 seconds. At least one **Inputs > Analog IN Channel** available for HumSilencer must be selected, and at least one **Waveform > Digital Outputs** channel must be selected.

Note: The **Start-to-Start Intervals** is calculated differently when using **Adapt between sweeps**. The minimum time required to accommodate the epoch durations for a protocol includes the settings for **Pre-sweep Train**, **P/N leak subtraction**, **Membrane Test Between Sweeps**, and the 1.081 seconds time to adapt between sweeps.

Data Acquisition

There are a number of options for data acquisition. Use the **Acquire** menu, or use the corresponding toolbar buttons.

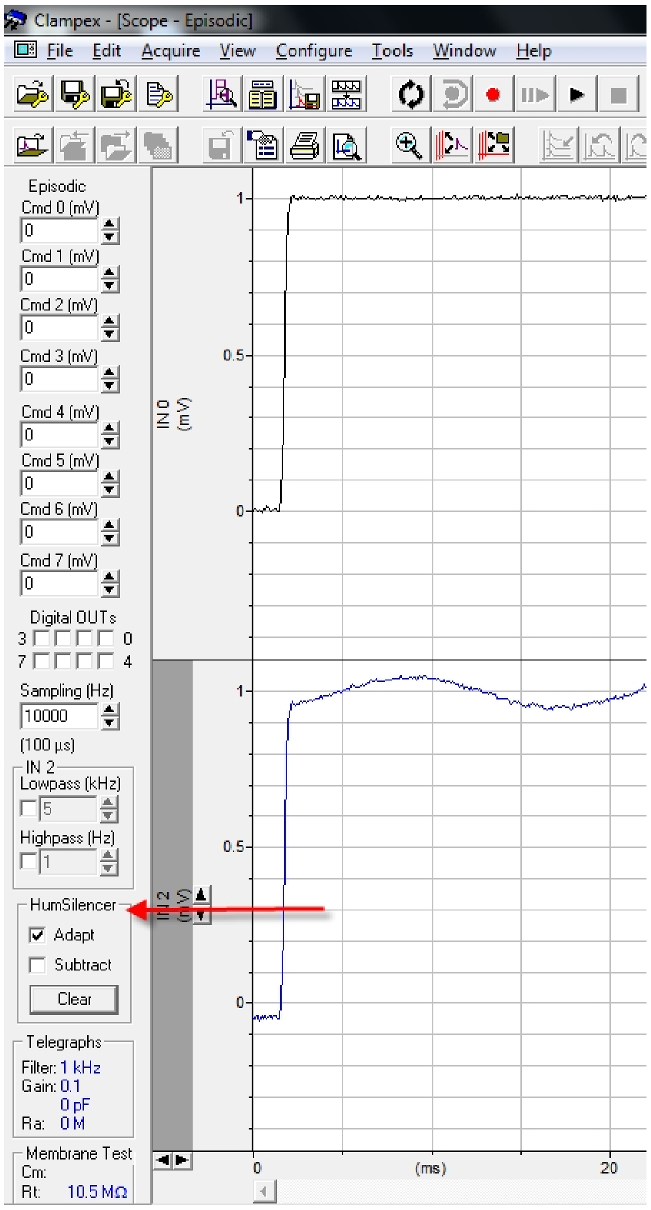
You can digitize data and view the result in the **Scope** window, without saving anything to disk, with **Acquire > View Only**. After the acquisition completes, you can save it to disk by selecting **Write Last**. Alternatively, you can press the **Record** button and the data are saved to disk during the acquisition. You can overwrite the last recorded data file by selecting the **Re-Record** button. You can also repeatedly perform the protocol, in either **View Only** or **Record** modes, by clicking the **Repeat** button.

Real Time Controls

The **Real Time Controls** panel, visible on the left side of the main window below the top toolbar, allows you to monitor the status of the experiment and easily control several selected input and output parameters while digitizing data (Figure 4-2).

You can test different experimental settings when viewing live data without having to open the **Edit Protocol** dialog or **Lab Bench** dialog, such as:

- Changing the holding levels of cells
- Controlling the digital outputs to a perfusion system
- Adjusting the sampling rate
- Applying filtering to the data signal currently selected
- Cancelling noise using HumSilencer system controls


Clampex - [Scope - Episodic]
File Edit Acquire View Configure Tools Window Help
IN 0 (mV)
IN 2 (mV)
(ms)
0
20
Clear

Episodic
Cmd 0 (mV) 0
Cmd 1 (mV) 0
Cmd 2 (mV) 0
Cmd 3 (mV) 0
Cmd 4 (mV) 0
Cmd 5 (mV) 0
Cmd 6 (mV) 0
Cmd 7 (mV) 0
Digital OUTs
3 ☐ ☐ ☐ ☐ 0
7 ☐ ☐ ☐ ☐ 4
Sampling (Hz) 10000
(100 µs)
IN 2
Lowpass (kHz) ☐ 5
Highpass (Hz) ☐ 1
HumSilencer
☑ Adapt
☐ Subtract
Telegraphs
Filter: 1 kHz
Gain: 0.1
0 pF
Ra: 0 M
Membrane Test
Cm:
Rt: 10.5 MΩ

Figure 4-2: Real Time Controls panel with HumSilencer system controls turned on

The **Real Time Controls** panel also reports where you are in an acquisition, in terms of elapsed time and sweep and run, and whether or not you have a conditioning train or **P/N Leak Subtraction** enabled.

To display the **Real Time Controls** panel:

- Select **View > Real Time Controls**.

  By default, the panel docks along the left frame of the main window.

To float the **Real Time Controls** panel:

- Click and drag the panel away from the from the left frame position.

The following rules apply to the parameters available in the Real Time Controls:

- Most parameters are available when acquiring data in **Acquire > View Only**. Use this mode to experiment with sampling rates and filter settings.
- A few parameters are available when recording data in non-episodic modes. Only analog and digital output levels can be changed, and a comment tag is inserted into the data at each change.
- Most parameters are disabled when recording data in **Episodic stimulation** mode. Only Digital Output levels can be changed, and a comment tag is inserted into the data at each change.

Using HumSilencer Adaptive Noise Cancellation Controls

When you use a digitizer with the HumSilencer Adaptive Noise Cancellation System, such as a Digidata 1550B digitizer, the controls for **HumSilencer** to learn real-time noise, activate noise cancellation, and reset noise-pattern learning, are displayed in the **Real Time Controls** panel after the HumSilencer-enabled digitizer is connected to a computer, and configured with pCLAMP Software Version 10.6 (Figure 4-2 and Figure 4-3).

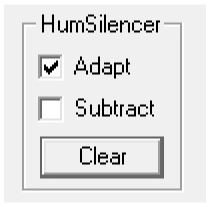


Figure 4-3: The HumSilencer System Real Time Controls in active learning mode, but not actively cancelling noise.

CAUTION! These **Humsilencer** controls in the Real Time Controls panel (Figure 4-3) do nothing until they are turned on by selecting the **Edit Protocol > Inputs > HumSilencer** dialog tab check box for an **Analog IN Channel**. See Enabling the HumSilencer System Controls on page 55.

The Real Time Controls for the **HumSilencer** system include:

- **Adapt** — Select this check box to turn on real-time noise pattern learning.

  To turn off the real-time noise pattern learning, deselect the **Adapt** check box.

  Note: Deselect **Adapt** when you have big steps or spikes in your signal, because fast changes will be visible as an excursion scaled to -1/50 of the fast signal amplitude in the output and will take 50 power cycles (about 1 second) to age out of the rolling averaged noise replica.

  The **Episodic stimulation** mode in Clampex Software automatically deselects **Adapt**.

- **Subtract** — Select this check box to turn on adaptive noise cancellation.

  To turn off adaptive noise cancelling, deselect the **Subtract** check box.

- **Clear** — Click this button to reset the collected cache of the learned real-time noise patterns.

  HumSilencer Start and **HumSilencer Stop** comment tags are displayed when a data file is opened in an Analysis window (Figure 4-4).

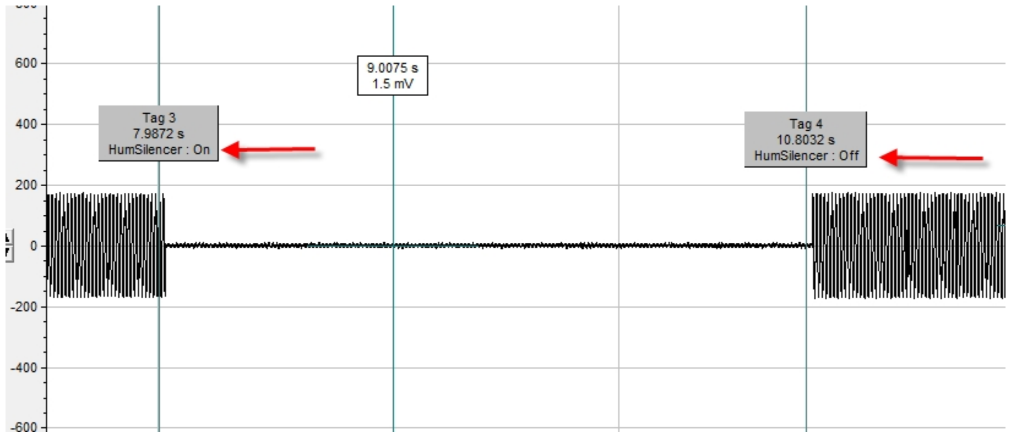


Figure 4-4: HumSilencer Start and HumSilencer Stop comment tags displayed in an open data file.

The way you use these controls depends on the selected protocol data acquisition mode. See Setting the Mode/Rate on page 52.

Note: When starting a recording with **HumSilencer** adaptive noise cancellation turned on, there can be a 21 ms recording-start delay while the HumSilencer system initializes.

For **Gap-free**, **Fixed-length events**, **High-speed oscilloscope**, and **Variable-length events** acquisition modes:

1. Make sure a **Edit Protocol > Inputs > HumSilencer** dialog tab check box for **Analog IN Channels** is selected.
2. If not already selected, in the Real Time Controls panel, select **Subtract** to turn on noise cancellation. The **Adapt** check box is automatically selected.
3. When you click **Record** or **View Only** to start data acquisition, during the data acquisition, the **HumSilencer** control settings can be changed.

For the **Episodic stimulation** acquisition mode:

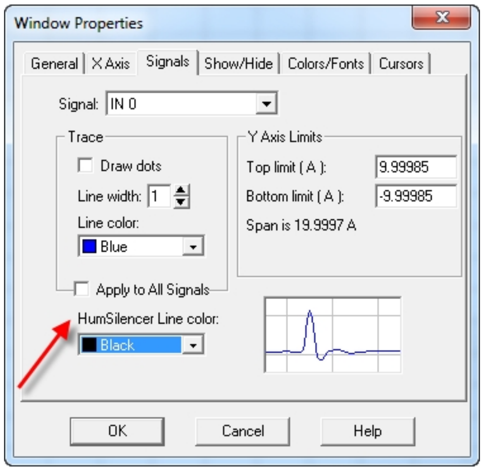
1. Make sure a **Edit Protocol > Inputs > HumSilencer** dialog tab check box for **Analog IN Channels** is selected.
2. If not already selected, in the Real Time Controls panel, select **Subtract** to turn on noise cancellation. Alternatively, you can leave **Subtract** deselected and noise cancellation turned off.
3. When **Record** or **View Only** is clicked to start data acquisition, the **Adapt** check box is automatically deselected, and the **Subtract** check box setting made in step 2 remains.

   During data acquisition, the **HumSilencer** control panel is disabled to prevent setting changes.

Note: The HumSilencer system cancels noise during the time between episodic sweeps, including during the **Edit Protocol > Stimulus** tab functions: **Pre-sweep Train**, **P/N Leak Subtraction**, and **Membrane Test Between Sweeps**.

In all data acquisition modes, **HumSilencer**-modified data can be displayed in a different color than the regular signal.

- Set the color from the **Scope** window in the **View > Window Properties > Signals** tab (Figure 4-5).



Signal: IN 0

Draw dots: ☐

Line width: 1

Line color: Blue

Apply to All Signals: ☐

HumSilencer Line color: Black

Top limit (A): 9.99985

Bottom limit (A): -9.99985

Span is 19.9997 A

Figure 4-5: HumSilencer line color change options in the View > Windows Properties > Signals tab

Seal and Cell Quality: Membrane Test

Tools > Membrane Test is a sophisticated set of controls for monitoring the electrode, the seal and the membrane, during and after the patching process. It calculates a range of accurate measurements by generating a square pulse of known dimensions and measuring the response signal.

The Membrane Test has three stages, each with its own set of parameters:

1. Bath stage monitors the electrode in the bath solution.
2. Patch stage monitors the creation of a gigaohm seal.
3. Cell stage monitors cell membrane resistance and capacitance.

The following values are calculated:

- Total resistance, Rt
- Access resistance, Ra
- Membrane resistance, Rm
- Membrane capacitance, Cm
- Time constant, Tau
- Holding current, Hold

Separate **Play**, **Pause** and **Stop** buttons allow you to control when the Membrane Test is running, as well as its settings and test history. Membrane Test can control the holding potential and stop generation of the test pulse. You can also configure and trigger a pulse train to interrupt the test pulse. Before using Membrane Test you must first define input and output signals through **Configure > Membrane Test Setup**.

Membrane Test measurements are displayed numerically in the **Membrane Test** window, and can also be charted in the **Online Statistics** window. You can automatically save the test measurements as an STA file with other statistics measurements from the **Online Statistics** window. A snapshot of the Measurements at the current time point can also be saved to the **Lab Book**.

Membrane Test can also be run automatically between every sweep. Configure this on the **Acquire > Edit Protocol > Stimulus** tab. In this mode the Membrane Test data are written out to the **Online Statistics** window and displayed in the Real Time Controls. The settings in **Configure > Membrane Test Setup** are also used in this mode. The **Membrane Test** dialog can be displayed, but it is not used when calculating **Membrane Test Between Sweeps**.

The Membrane Test pulse train is used to deliver a series of stimuli without closing the Membrane Test (the test pulse ceases while the train is run). The pulse train can be configured to use a different signal from the primary test pulse, possibly on a different output channel, so that you can send it to a separate stimulating electrode.

In order to calculate many of the measurements, the transient current response is fit by a single exponential. From this the decay time constant is calculated and used to ensure the signal reaches a steady state for reliable current measurements. Similarly, the fitted curve is used to calculate the transient peak, from which the access resistance is derived. The options in **Configure > Membrane Test Setup** let you define how much of the falling transient from each pulse edge to use for curve fitting. This same dialog also allows you to choose the timing and amplitude of the pulse train.

In addition to measuring important initial seal and cellular parameters, you can use the Membrane Test to perform an experiment. For example, you can use the pulse train to depolarize and stimulate a cell, and induce exocytosis in a secretory cell. Then, you can record the secretory cell's whole-cell response and monitor relative capacitive changes.

You can also include the Membrane Test in sequencing key series, for example to monitor seal or access resistance as part of a set sequence of steps.

For the algorithms used to calculate Membrane Test values, see Membrane Test on page 85. See also Membrane Test Tutorial on page 102.

Using HumSilencer in Membrane Test

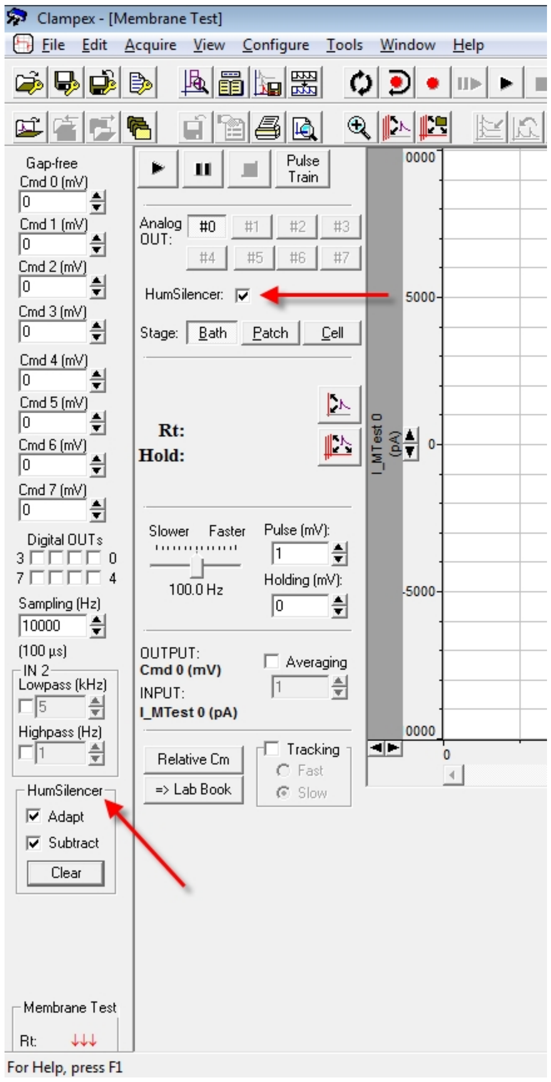
Note: There is no real-time noise learning during a Membrane Test.

To use the HumSilencer adaptive noise cancellation system in a Membrane Test:

1. In the **Membrane Test** dialog, select the **HumSilencer** check box to turn on the **HumSilencer** Real Time Controls (Figure 4-6).
2. If not already selected, in the Real Time Controls panel, select **Subtract** to turn on noise cancellation. Alternatively, you can leave **Subtract** deselected and noise cancellation turned off.
3. When **Record** or **View Only** is clicked to start data acquisition, the **Adapt** check box is automatically deselected, and the **Subtract** check box setting made in step 2 remains.

   During data acquisition, the **HumSilencer** control panel is disabled to prevent setting changes.

   Note: When starting a recording with **HumSilencer** adaptive noise cancellation turned on, there can be a 21 ms recording-start delay while the HumSilencer system initializes.



Gap-free
Cmd 0 (mV) 0
Cmd 1 (mV) 0
Cmd 2 (mV) 0
Cmd 3 (mV) 0
Cmd 4 (mV) 0
Cmd 5 (mV) 0
Cmd 6 (mV) 0
Cmd 7 (mV) 0
Digital OUTs
Sampling (Hz) 10000
(100 µs)
IN 2
Lowpass (kHz) 5
Highpass (Hz) 1
HumSilencer
[✓] Adapt
[✓] Subtract
Membrane Test
Rt: ↓↓↓

HumSilencer: [✓]
100.0 Hz
Pulse (mV): 1
Holding (mV): 0
OUTPUT: Cmd 0 (mV)
INPUT: I_MTest 0 (pA)
[ ] Averaging 1
[ ] Tracking
Fast
[✓ Slow]

Figure 4-6: HumSilencer system controls in the Real Time Controls panel and Membrane Test dialog

Membrane Test Measurements

In pCLAMP Software, the resistance due to an electrode alone is termed the electrode resistance (R_e). This is also sometimes termed the pipette resistance (R_p). Usually, in addition to electrode resistance, there is a degree of resistance due to largely unknown environmental factors near the tip of the electrode. Cellular debris, air bubbles and poorly conducting solution might all contribute to this additional resistance (which we will call R_{debris}).

The sum of the electrode resistance and resistance due to these additional factors is the access resistance (R_a):

$$R_a = R_e + R_{debris}$$

Access resistance is also commonly termed series resistance (R_s). While Rs is the term used with Axon Instruments amplifiers, we avoid its use in pCLAMP Software as it can be confused with seal resistance, dealt with below.

Resistance across the cell membrane is called membrane resistance (R_m).

The resistance between the headstage and ground is the total resistance (Rt). Strictly speaking, when the electrode is in contact with the cell, there are two pathways from the electrode tip to ground—one traversing the cell membrane and the other bypassing it, leaking directly from the tip into the bath. However, the comparative resistances in the two pathways are such that we can generally ignore one or the other pathway, depending on whether the electrode has access to the cell interior or is patched to the outer surface. The two pathways are illustrated in Figure 4-7.

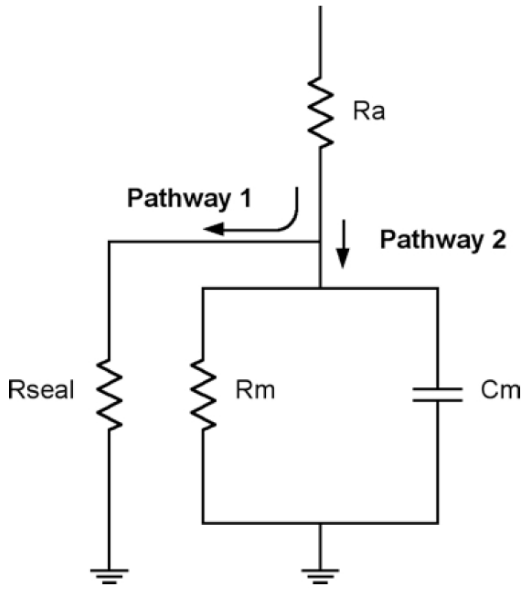


Figure 4-7: Idealized circuit showing two pathways from electrode tip to ground.

Patch Scenario

When a seal is created but the cell membrane is not ruptured, the resistance of the minute section of membrane in the patch is very high, so any current is likely to leak through the membrane/electrode seal. In Figure 1, this is equivalent to Pathway 2 being removed and all current taking Pathway 1. The resistance to this leakage, determined by the quality of the seal attained, is generally termed the seal resistance. In this case total resistance effectively consists of the access resistance and the seal resistance in series:

$$R_t = R_a + \textit{seal resistance}$$

In any successful seal, the seal resistance is orders of magnitude larger than the access resistance so that:

$$R_t \approx \textit{seal resistance}$$

Ruptured Patch Scenario

For a ruptured patch, when a satisfactory GΩ seal has been achieved, we hope that most current follows Pathway 2. In fact, with R_m and R_{seal} in parallel, we cannot distinguish these from each other. In this case, then, we assume all resistance is due to R_m. Under this assumption the total resistance consists of the access resistance and membrane resistance in series:

$$R_t = R_a + R_m$$

Terminology

As noted above, seal resistance is sometimes represented $\mathbf{R_s}$, which is also used to indicate series resistance (i.e. access resistance). For this reason we avoid the use of $\mathbf{R_s}$ in pCLAMP, favoring instead $\mathbf{R_a}$, $\mathbf{R_t}$ or the words “seal resistance” or $\mathbf{R_{seal}}$ when we need to refer to seal resistance specifically.

Bath

When the electrode is in the bath the total resistance will be just the electrode resistance (R_e). As the electrode approaches the cell, debris around the electrode tip may add further resistance (R_{debris}).

Seal

Only once a tight seal is formed with the cell membrane is the reported value effectively a measure of seal resistance (R_{seal}).

Membrane Test allows you to control the holding potential—a feature that is often used to hyperpolarize the cell to aid in seal formation. This control remains only for as long as the Membrane Test is run.

Time, Comment, and Voice Tags

Time Tags, Comment Tags, and Voice Tags let you annotate data while it is being collected. Each of these tags is available through the Acquire command during data acquisition, as well as from dedicated toolbuttons.

Time tags insert a simple numerical time-stamped tag into the data file. A **Comment** tag allows you to add an additional line of text to each tag. It is important to note that a comment tag is inserted when the tag is activated, not when you finish typing the comment. Clampex Software keeps a list of comment tags that have been used so that you can quickly recall a previous tag rather than retyping it.

If the computer includes a sound card, you can insert Voice (audio) tags into data files. This is analogous to recording your voice on an audio channel when saving data to a VCR tape. To configure Voice tags, choose **Configure > Voice Tags**. When you open the data file, double-click on a **Voice** tag to hear the audio comment.

Junction Potential Calculator

The Junction Potential Calculator provides a wealth of information on the measurement and determination of junction potentials created by solution changes. To start the calculation, choose **Tools > Junction Potential**. The **Calculate Junction Potentials** dialog box opens and lets you define the experiment type and temperature, and the type of ground electrode being used.

After providing this information, the Junction Potential Calculator graphically displays the various elements of the junction potentials. To determine the new junction potential, input the concentrations of the various ions in the new solution. The Junction Potential Calculator computes the junction potential that will be created by the solution. You can choose the ions from a supplied ion library list, or add your own ion definitions to the list. Additionally, you can copy the results of the Junction Potential Calculator to the Lab Book, save them to disk, or print them.

For more detailed information on how to use the Junction Potentials command, refer to the Help file.

Calibration Wizard

The Calibration Wizard lets you define the scaling and offset of a signal, after the data has been acquired. This is analogous to drawing a known scale bar on chart paper when recording continuous data to a pen recorder, and then measuring the scale bar afterwards to determine the scale factor for each signal of data. The Calibration Wizard is available when viewing data in the Analysis Window, and is activated by choosing the **Tools > Calibration Wizard** command.

When using the Calibration Wizard, you have the option of setting the scale factor based on measurements you make from the data file, or applying a known scale factor to the data. If you choose to set the scale factor based on the data file, you simply position Cursors 1 and 2 over two regions of known amplitude and indicate their values. The Calibration Wizard then computes the appropriate scale factor and offset, and optionally allows you to apply these values to the Lab Bench, so that any other data files that use the

same signal also have the correct scale factor and offset.

Alternatively, if you already have defined the scale factor and offset in the Lab Bench, you can directly update these settings in the data file using the Calibration Wizard.

Existing data files can also be rescaled in pCLAMP Software using **Edit > Modify Signal Parameters**.

Sequencing Keys

The Sequencing Keys command in the Configure menu lets you associate events, or a sequence of events, with a keystroke action.

For example, you can define a single key to flip a digital OUT high to activate a solution change, and another key to flip the digital OUT low to terminate the solution change. These keys are available as toolbar buttons in the Sequencing toolbar. You can also define a tooltip (a popup message) that reminds you what event is associated with each button.

In addition to setting various digital OUTs, the sequencing keys also let you change the holding levels, insert a comment tag, start a Membrane Test, load and run a protocol, or display a prompt.

In addition to associating a keystroke with an event, you can use the sequencing keys to link one event to another, and run an experiment in an automated fashion.

For example, you may want to start a Membrane Test, and when it finishes, run an I-V protocol, perform a solution change, and then run the I-V protocol again. Using the Sequencing Keys, you can link these events together and define the timing interval between them. You can save an entire sequencing series to disk, and maintain several sets of sequences for each type of experiment that you perform.

LTP Assistant

Long-term potentiation (LTP) and long-term depression (LTD) are terms for a broad range of experiments designed to investigate synaptic plasticity. In particular, it has been found that certain stimulus regimes alter synapse function, both to increase (LTP), and decrease (LTD) postsynaptic response. **Tools > LTP Assistant** provides a convenient interface for the setup of LTP and LTD experiments. This section provides an in-depth discussion of the organization and functionality of the **LTP Assistant** dialog.

The **LTP Assistant** dialog brings together a range of Clampex Software functions most commonly needed for LTP and LTD experiments that are formatted to follow a natural sequence for the setup of these types of experiments.

The **LTP Assistant** dialog should be of use to both new and experienced LTP experimenters, however it requires a basic knowledge of using Clampex Software (for example, the **Lab Bench**). A range of pre-configured protocols are included along with default stimulation waveforms. You can select the default protocol closest to your needs and adjust the settings. Alternatively, you can copy previously saved protocol files into the **LTP Assistant**.

If you want to design experiments with configuration options beyond those offered in the **LTP Assistant** dialog, you can use it for overall experiment management. For example, if the range of protocol definition options provided within the **LTP Assistant** dialog is insufficient, you can open the **Edit Protocol** dialog from within the **LTP Assistant** dialog (**LTP Assistant > Baseline > Edit**) and define your protocol with the broader range of options offered there.

The **LTP Assistant** dialog offers the following features:

- Creation and sequencing of the baseline and conditioning stages of an experiment.
- Configuration of the stimulus waveforms used during baseline and conditioning stages, for both digital and analog stimulation.
- Conditioning pulse trains of indeterminate length.
- Alternation of presynaptic stimulus between two pathways.
- Postsynaptic command stimulus to coincide with or follow presynaptic conditioning stimulus.
- Preconfigured statistics measurement.

The assistant has four tabs, used as follows:

- **Sequencing**: General configuration of the stages that make up the experiment.
- **Inputs/Outputs**: Configuration of Clampex Software for the input and output channel and signal connections in the experiment setup.
- **Baseline**: Configuration of the protocol used in baseline stages of the experiment. This defines the baseline stimulation waveform. Enabling of default statistics measurements is from this tab.
- **Conditioning**: Configuration of the protocols used in conditioning stages of the experiment. This includes conditioning stimulation waveforms and enabling and configuration of a paired postsynaptic command waveform.

Experiment and Data Organization

The set of configuration options defined within the **LTP Assistant** dialog is called an “experiment”. Experiments are saved from within the **Sequencing** tab. This creates a folder containing a sequencing key file (with the experiment’s name and *.sks extension) and as many protocol files (*.pro extensions) as are named in the **Sequencing** tab, for example are incorporated into the experiment. These constituent files are all saved at the same time as an experiment. Experiment folders are saved in **My Documents\Molecular Devices\pCLAMP\Params\LTP1**.

Protocols can be reused for different experiments, but in each case are copied to the relevant experiment folder. As each copy is placed into a different folder from the original file you can keep its original name, or rename it if you want, on the **Sequencing** tab.

To run an experiment:

1. Open an experiment in the LTP Assistant dialog (**Tools > LTP Assistant > Sequencing > Edit**).
2. Close the **LTP Assistant** dialog This loads the sequencing key file for the experiment into Clampex Software.
3. Start the experiment by pressing the sequencing key for the first stage.
4. Subsequent stages can also be started with sequencing keys, or might have been configured to run automatically when the previous stage finishes.

Each run of a protocol results in one data file,if you have chosen to record it. The data files generated during an experiment are named and located according to standard Clampex Software functionality, set in **File > Set Data File Names**. You can, if you want, later concatenate the files from an experiment in pCLAMP Software, with **Analyze > Concatenate Files**, provided they were recorded under similar conditions.

As normal for Clampex Software, all statistics are written continuously to the **Online Statistics** window. You should, then, clear the previously logged statistics (**Edit > Clear Statistics**) before you start an experiment. If the final (baseline) protocol in an experiment runs to completion, the statistics for the whole experiment are automatically saved in one file by default. If, on the other hand, you end the experiment manually, you need to manually save the statistics (**File > Save As** when the **Online Statistics** window is selected).

Signals and Channels

LTP and LTD experiments involve, at minimum, one source of presynaptic stimulation and one postsynaptic recording electrode. Stimulation might be delivered extracellularly, with a stimulus isolation unit (SIU), or intracellularly, with a standard amplifier electrode. The postsynaptic recording electrode can be intracellular or extracellular.

Beyond this, and depending on the number of amplifiers available to the experimenter (and the number of electrodes these support), many more configurations are possible. The LTP Assistant aims to streamline software configuration for the most common electrode placements.

Output Channels

The LTP Assistant has the capacity to enable and configure four output channels:

- Two analog outputs (Analog OUT #0 and #1)
- Two digital outputs (Digital OUT #0 and #1)

It is anticipated that the analog waveforms are output to standard amplifier electrodes, while the digital outputs drive SIUs.

The output channels are grouped on the **Inputs/Outputs** tab to allow configuration of one or two “pathways”— routes of neuronal activity across a synapse from a presynaptic to a postsynaptic cell. Thus, you can enable a maximum of two output channels for presynaptic stimulation. If you have enabled two digital presynaptic outputs, any further outputs (necessarily analog) can only be for postsynaptic electrodes, assumed to be in the same pathways.

The digital outputs can only be used for presynaptic stimulation. Given the potential for two pathways, it is possible to have one digital and one analog presynaptic command, which would then leave one analog channel for postsynaptic stimulation. When two presynaptic outputs are enabled, you can also choose to alternate the baseline protocol delivery between these, sweep by sweep. This is enabled from the **Baseline** tab.

Presynaptic and postsynaptic analog output channels have different waveform configuration options. Presynaptic channels can be configured to deliver pulses or pulse trains. Postsynaptic channels are restricted to delivering postsynaptic pairing steps or pulses (for conditioning stages).

Even if they have not been selected for presynaptic or postsynaptic commands, the analog output channels are always active, and always have a signal and holding level assigned to them. In this case, they can be used to clamp the voltage on a recording electrode, and you need to ensure on the **Inputs/Outputs** tab that you have the correct signal and holding level for the channel, and connect it to the recording electrode. In general, it is preferable to clamp recording electrodes from within the **LTP Assistant** dialog rather than with the amplifier, because the holding level is then written into the data file for later reference.

Input Channels

Four analog input channels (Analog IN #0, #1, #2, and #3) are available for recording or viewing data.

The LTP Assistant supports two analog output channels, so it is not possible to use the LTP Assistant to clamp four recording electrodes. In such a case, two of the recording electrodes can be clamped from within the LTP Assistant, and the two remaining electrodes would then need to be clamped by the amplifier holding level control.

Alternatively, the experiment protocols can be manually setup from within the protocol editor to control four analog output channels, and then configured for sequencing via sequencing keys.

Signals

Selection of signals for input and output analog channels follows standard Clampex Software procedure. Create and configure signals in the **Lab Bench**, in association with particular digitizer channels. Then, in the **LTP Assistant > Inputs/Outputs** tab, select specific signals for the channels you have enabled. This is the same as in the **Input** and **Output** tabs of the **Edit Protocol** dialog for standard protocols. If you are unfamiliar with the relation between signals and channels in Clampex Software, see Definitions on page 17.

Example Setup

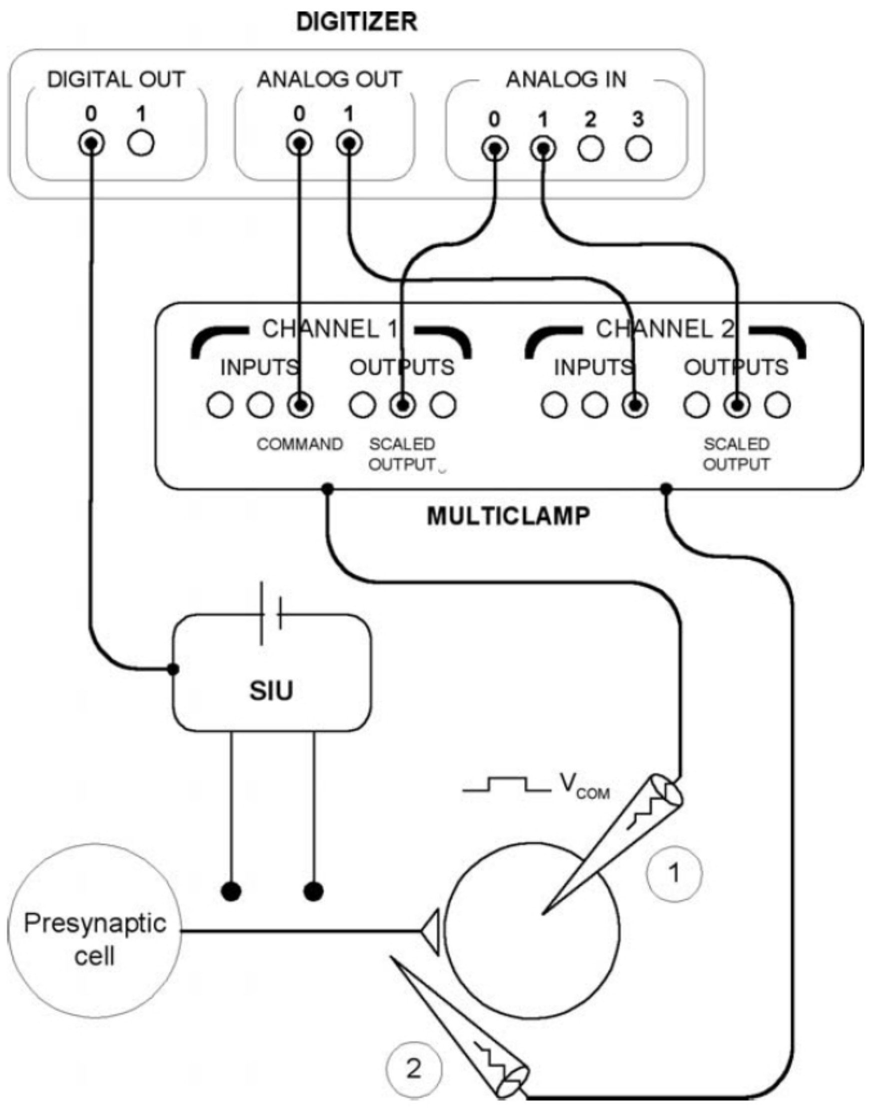


Figure 4-8: Example of LTP experiment setup, with digital presynaptic stimulation, postsynaptic intracellular voltage step command and current recording, and extracellular recording.

This setup has the following features:

- Presynaptic stimulation is delivered by an SIU via digital OUT #0.
- Postsynaptic intracellular electrode #1 is clamped via analog OUT #0, and can be configured to deliver a paired voltage step during conditioning stages. The electrode records to analog IN #0.
- Electrode 2 records extracellular responses to analog IN #1. It receives no command waveform, but is clamped via analog OUT #1.

Temporal Structure of LTP Experiments

LTP and LTD experiments focus on changes in synaptic behavior. The paradigm experiment thus typically has three stages:

- **Baseline**: The presynaptic cell is stimulated at a low frequency and the postsynaptic cell monitored to establish the base behavior of the synapse under investigation. Raw postsynaptic response data may or may not be recorded, but typically statistics are recorded, e.g. peak amplitudes and rise slopes. The baseline stage is also referred to as the “test”, or “test 1” period.
- **Conditioning**: A different stimulus is applied to the synapse to evoke LTP or LTD. This might involve one or more “trains” (or “bursts”) of pulses delivered via the presynaptic cell, or some pattern of presynaptic stimulation accompanied by a command signal to the postsynaptic cell (“pairing”). This stage is sometimes called “induction”, and sometimes “tetanus”, though this latter term more correctly names a particular type of conditioning stimulus.
- **Baseline**: the postsynaptic cell is monitored under the same stimulus regime used in the first baseline stage, in order to detect changes brought about during conditioning. This second application of the baseline is generally run for longer than the first. Again, this stage is sometimes referred to as “test” or “test 2”.

As the first step in creating a new experiment setup a default three-stage “baseline-conditioning- baseline” format like the one above is displayed in the **Sequencing** tab, to which you can add as many stages of either sort as you like.

Each stage of the experiment is identified as either a baseline or conditioning protocol, determining the range of waveform configuration options available for that stage. The conditioning protocol is further configured with default settings for tetanus, theta or LTD stimuli. You can change these settings later in the **Baseline** tabs or **Conditioning** tabs. You are able to use a given protocol in any number of stages; in fact, this is enforced in the case of baseline stages:

- Only one baseline protocol can be used in an experiment.

So you can have as many baseline stages as you want, but they must use the same protocol, and hence have the same stimulus waveform. On the other hand, there is no limit to the number of conditioning protocols you can use in an experiment, if you have more than one conditioning stage.

As well as determining whether a stage is baseline or conditioning on the **Sequencing** tab, you select whether each stage will be viewed only or recorded to disk. Statistics measurements can be taken in either case.

Stage Timing and Sequencing

The LTP Assistant creates and displays sequencing keys in the sequential order in which they were created. However, this ordering of experiment stages does not control the order in which they are actually executed. Each stage is automatically assigned a “Start key”, and a “Next key”—the Start key of the next stage to run. This allows you to have the stages run in any order you want, by changing the Next key assignment in a protocol’s “Properties”. So, you can configure the completion of one stage to automatically start the running of another stage, or manually interrupt a stage and start the next stage by pressing its sequencing key. You can also have a stage link to itself, so that it loops continuously until you manually start another stage.

The duration of each stage is set in the Sequencing tab. This can either be set to the acquisition time defined in the protocol Sweeps section, or you can override the protocol by setting a different time here. This setting can thus cut short a protocol’s running time or, if longer than the protocol, hold the experiment in a state of inactivity until the next stage is scheduled to start. Baseline stages are set to run for 1000 sweeps by default. This generally gives you a longer protocol run-time than needed, so that it is likely that a steady baseline response is achieved, after which you can then manually trigger the conditioning stage. In contrast, conditioning stages are usually short and should generally be set to run for the protocol acquisition time, automatically calling the following baseline stage when finished.

Protocols

The **LTP Assistant** dialog has one tab each for baseline and conditioning protocol configuration. These contain sub-sets of Clampex Software protocol-definition options. If you want further configuration options or simply want to check any default settings not revealed in the LTP Assistant, click **Edit** at the bottom of the tab to view the currently loaded protocol in the main protocol editor.

Both tabs allow you to configure a pattern of rectangular pulses for presynaptic stimulation. The same set of options is offered for digital and analog waveforms, except that you must enter pulse amplitudes for analog signals, the same pulse amplitude is used throughout the waveform. The default sampling rate for both baseline and conditioning protocols, not shown in the LTP Assistant, is 10 kHz.

In the **Sweeps** section, the **Start-to-start** time determines the frequency at which a waveform is delivered. This cannot be less than the duration of the sweep length, which is a combination of the **Waveform length** and derived pre- and post-waveform holding periods, each 1/64th of the waveform length. The **Waveform length** must be of sufficient duration to contain the presynaptic waveform, and if enabled, the resistance test or postsynaptic pairing. You can view, record, and take peak measurements from the input signals while the waveform is being output (i.e. for the duration of the sweep length), but you must then wait until the start of the next sweep, determined by the sweep start-to-start time, before more data are digitized.

For baseline protocols only, if you have enabled two presynaptic stimulation outputs, you can alternate delivery of these, sweep by sweep. First, sweep 1 of the protocol configured for OUT #0 is delivered, while OUT #1 is held inactive (for digital outputs) or at holding level (for analog outputs). Next, sweep 2 of the protocol configured for OUT #1 is delivered, while OUT #0 is held inactive or at holding level. For the next sweep, sweep 3 on OUT #0 is delivered, and so on. It is possible to alternate between digital and analog outputs, and combinations of these.

Conditioning protocols do not allow alternation. If two presynaptic output channels are enabled, whether or not these have alternation enabled in the baseline stage, they are both available for concurrent delivery during conditioning stages, unless both outputs are digital, in which case channel OUT #0 is available for configuration, and OUT #1 is held inactive.

LTP Assistant differences between options for the baseline and conditioning stage protocols are summarized in the following table:

Table 4-1: Options for the baseline and conditioning stage protocols

	Baseline Protocols	Conditioning Protocols
Alternating sweeps	Alternate stimulation waveforms, sweep by sweep, between two presynaptic channels.	Not available.
Number of sweeps	1000 sweep default.	User configurable.
Pulse trains	Not available. Up to four pulses per sweep.	Up to four trains per sweep.
Statistics	Peak amplitude, time of peak and slope measurements.	Not available.
Postsynaptic pairing	Not available.	Step postsynaptic voltage or current for entire conditioning period, or in a pulse following presynaptic stimulus.

With the “Copy From” button on each protocol tab, you can select a protocol to copy into the LTP Assistant. Protocol settings are automatically changed, if necessary, to be consistent with the settings current in the assistant at the time, for example the signals and sampling rate are changed to those of the protocol already loaded in the LTP Assistant. Once copied into the experiment, you can adjust the protocol’s settings at will.

The Preview button in the LTP Assistant opens an Analysis window displaying the waveform you have configured. This window updates automatically as you make changes to waveform configuration within the LTP Assistant.

Presynaptic Stimulus Waveform Definition

Waveforms for both baseline and conditioning protocols are built up from “pulses”, with rapid sequences of pulses (“trains”) configurable for conditioning protocols only. A pulse is an output (current or voltage) step, defined in terms of its amplitude and duration.

As noted above, digital and analog waveform definition is essentially the same, differing only in that you must set a pulse amplitude (“pulse level”) for analog signals. You must enter an absolute value for this, rather than a value relative to the baseline.

Existing Clampex Software functionality dictates that the basic unit for waveform configuration is the sweep. Each sweep has ten configurable subunits: the epochs. Epochs are not referred to in the assistant, but their role in waveform definition can be seen, if you want, by opening the protocol editor and viewing the Waveform tabs. You are able to set the total duration of the epochs, and hence of the time available for waveform configuration, with the “Waveform length” field in each of the Baseline and Conditioning tabs. With one epoch reserved for an optional initial delay period at the start of the sweep, and the final two for a postsynaptic pairing delayed pulse (in conditioning protocols), seven remain for presynaptic waveform configuration. This means that you can have a maximum of four pulses per sweep for baseline protocols, or four trains per sweep for conditioning protocols. One or two pulses per sweep is adequate for baseline stimulation in most experiments, and, with the capacity to repeat sweeps, four trains per sweep is sufficient for many conditioning stimuli as well. If not, you can build additional conditioning stages into the experiment and string these together on the Sequencing tab, so that the total conditioning stimulation might consist of several conditioning stages.

Pulses and trains can be described in terms of a number of different parameters. The parameters used for sweep waveform configuration in the LTP Assistant are illustrated in Figure 4-9.

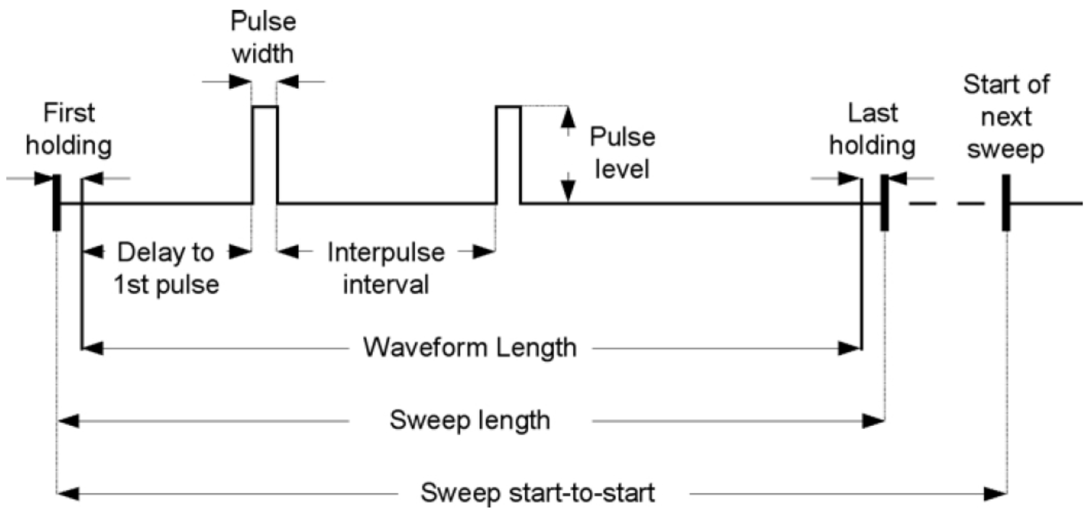


Figure 4-9: Baseline sweep with two pulses per sweep showing configuration parameters used on Baseline tab.

The first and last holding periods are not referred to in the waveform configuration interface, but are shown in the diagrams to illustrate the difference between the value entered in the "Waveform length" field and the reported "Sweep length".

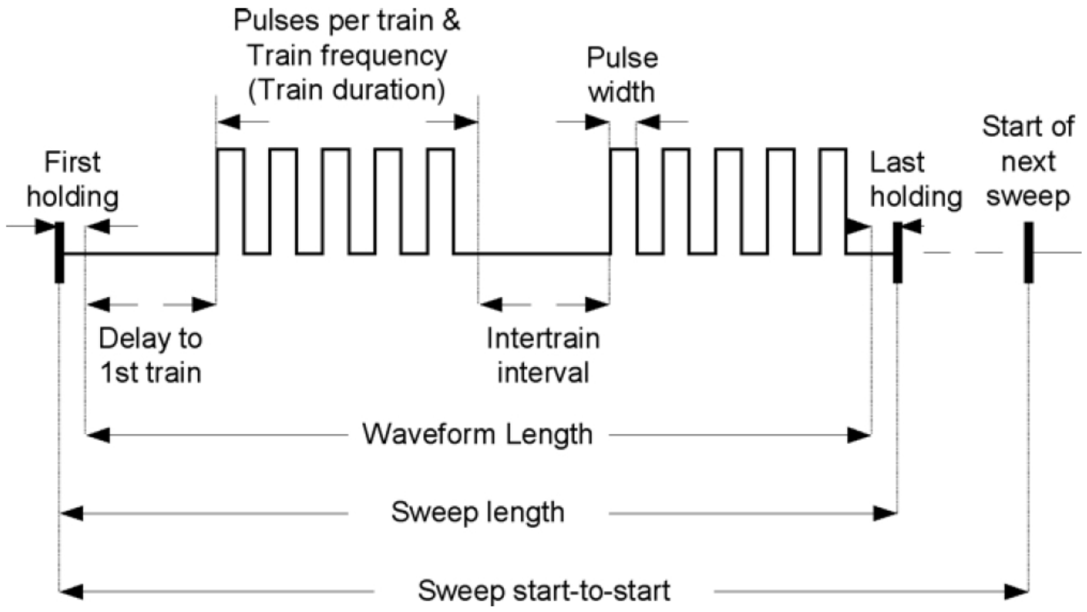


Figure 4-10: Conditioning sweep configuration with two trains per sweep, showing parameters used on Conditioning tab.

Because the pulses in a train each consist of a step and a following section at the holding level, trains similarly always begin with the start of a pulse, and end after a period at the holding level (equivalent to the interpulse interval within the train). The length of this period depends upon the pulse width and frequency you have set.

Predefined Protocols

Protocol definition does not start from a blank slate. When you first outline the overall structure of an experiment on the Sequencing tab, default protocol settings are loaded for each stage you create. One default protocol is used for all baseline stages, while you are given a choice of three default protocols for conditioning stages. This is simply to save you time, allowing you to select a starting point close to the configuration you want. The four default protocols have the following presynaptic waveform configuration.

Baseline: A single pulse of 1 ms at 0.05 Hz (20 s start-to-start interval). 51.6 ms of data is acquired for every pulse, starting 5.8 ms before the onset of the pulse.

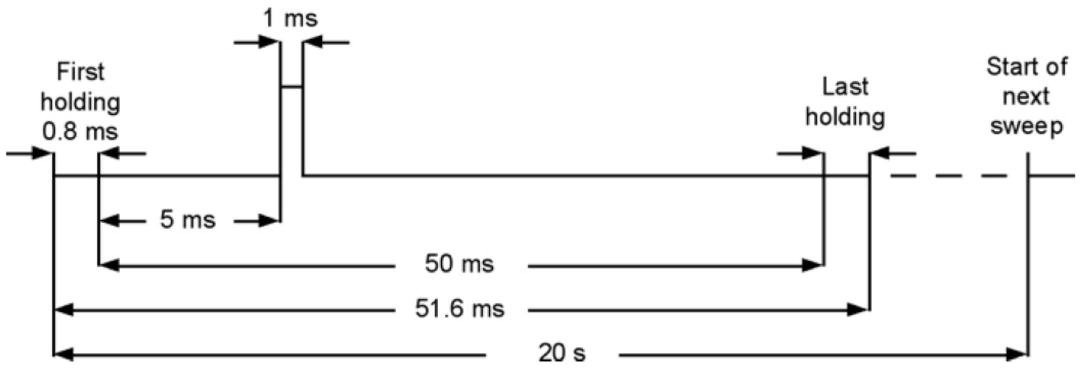


Figure 4-11: Default baseline presynaptic waveform.

- Sweep start-to-start: 20 s (0.05 Hz)
- Waveform length: 50 m
- Sweep length: 51.6 ms
- Pulses per sweep: 1
- Pulse level: 20 mV
- Delay to first pulse: 5 ms
- Pulse width: 1 ms
- Interpulse interval: not applicable
- Sampling interval: 100 µs (10 kHz)
- No. of sweeps: 1000

Tetanus: This is the default conditioning protocol that each new experiment starts with, and is the protocol used if you add a new “Tetanus” conditioning stage from the Sequencing tab Add dialog. Four 100 Hz, 1 s trains are delivered 20 s apart.

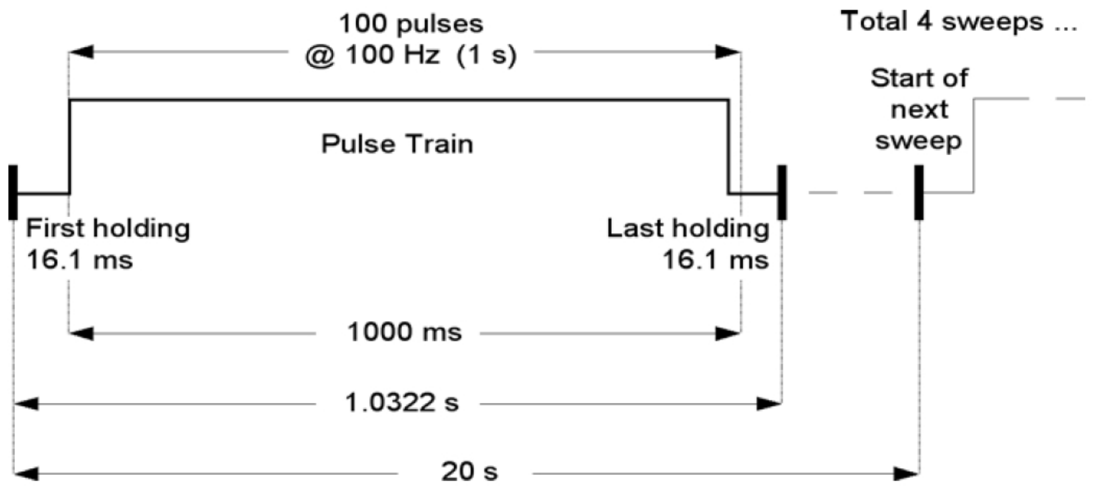


Figure 4-12: Default “Tetanus” conditioning stimulus.

- Sweep start-to-start: 20 s (0.05 Hz)
- Waveform length: 1000 ms
- Sweep length: 1.0322 s
- Pulses per train: 100
- Trains per sweep: 1
- Pulse level: 20 mV
- Delay to first train: 0 ms
- Train duration: 1000 ms
- Train frequency: 100 Hz
- Pulse width: 1 ms
- Intertrain interval: not applicable
- Sampling interval: 100 µs (10 kHz)
- No. of sweeps: 4

Theta: A series of ten short 100 Hz pulse trains, of five 1 ms pulses each, are delivered at a frequency of 5 Hz.

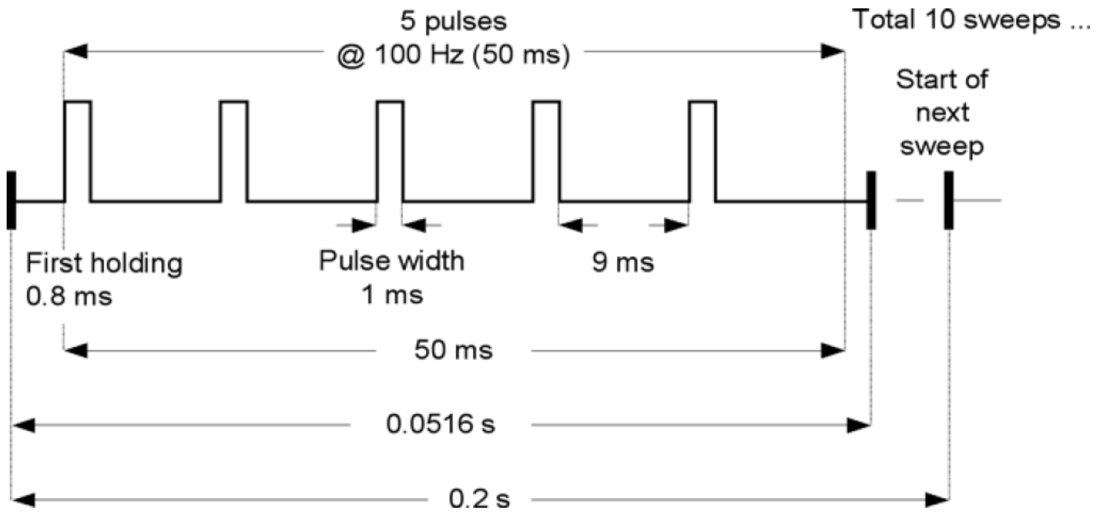


Figure 4-13: Default “Theta” conditioning stimulus.

- Number of sweeps: 10
- Sweep start-to-start: 0.2 s (5 Hz)
- Waveform length: 50 ms
- Sweep length: 0.0516 s
- Trains per sweep: 1
- Pulse level: 20 mV
- Delay to first train: 0 ms
- Train duration: 50 ms
- Intertrain interval: not applicable
- Pulses per train: 5
- Train frequency: 100 Hz
- Pulse width: 1 ms
- Sampling interval: 100 µs (10 kHz)

LTD: A single pulse at 1 Hz is delivered for 15 minutes.

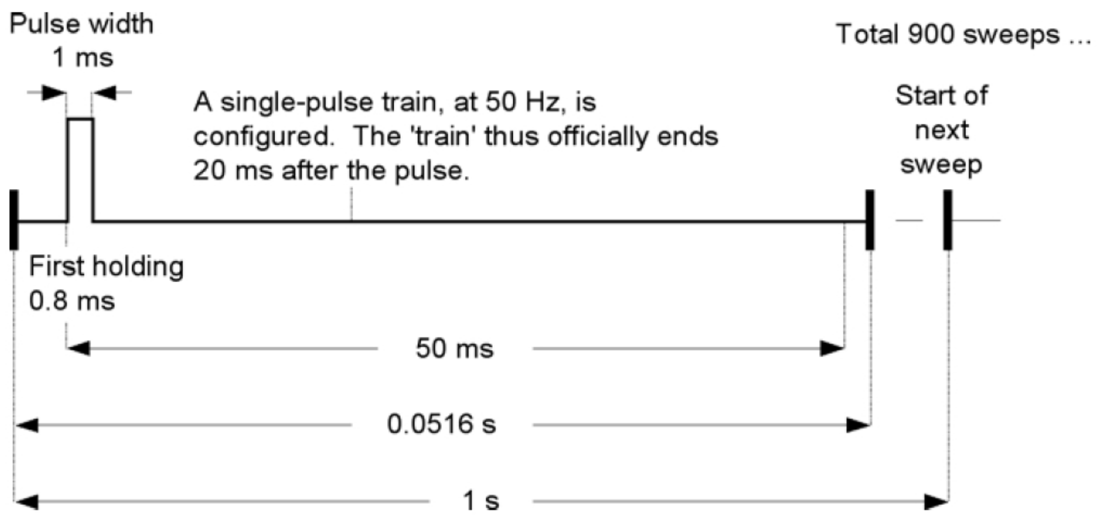

Pulse width
1 ms
Total 900 sweeps ...
A single-pulse train, at 50 Hz, is configured. The 'train' thus officially ends 20 ms after the pulse.
Start of next sweep
First holding
0.8 ms
50 ms
0.0516 s
1 s

Figure 4-14: Default “LTD” conditioning stimulus.

- Number of sweeps: 900
- Sweep start-to-start: 1.0 s (1 Hz)
- Waveform length: 50 ms
- Sweep length: 0.0516 s
- Trains per sweep: 1
- Pulse level: 20 mV
- Delay to first train: 0 ms
- Train duration: 20 ms, but not relevant
- Intertrain interval: not applicable
- Pulses per train: 1
- Train frequency: 500 Hz, but not relevant
- Pulse width: 1 ms
- Sampling interval: 100 µs (10 kHz)

Postsynaptic Pairing

In postsynaptic pairing a stimulus is delivered to a postsynaptic cell in association with the presynaptic conditioning stimulus. This option therefore requires that you have a postsynaptic command channel enabled, on the **Inputs/Outputs** tab.

Two types of postsynaptic pairing can be configured. Usually used in voltage-clamp experiments only, you can set a voltage step to be held for the duration of the conditioning stage. The postsynaptic command shifts to the set voltage after the first holding in the first sweep of the conditioning protocol and maintains this until the last holding in the final sweep, including intersweep periods where there is otherwise no output.

More often used in current-clamp experiments, you can also opt to deliver a single postsynaptic pulse, following a stipulated time after the end of the presynaptic stimulus. For the delayed pulse option, in order that the postsynaptic waveform is timed with respect to the end of the presynaptic stimulus, you must use a digital channel for presynaptic stimulation, and the analog postsynaptic channel of the same number for the paired stimulation, e.g. digital OUT #0 for presynaptic stimulation, and analog OUT #0 for the postsynaptic command. It is possible to pair postsynaptic stimulation with an analog presynaptic command, by enabling, say, analog OUT channels #0 and #1 for presynaptic and postsynaptic commands respectively, but in this case you need to time the postsynaptic waveform from the start of the sweep. The delay to the postsynaptic command, in this case, is not automatically calculated from the end of the presynaptic stimulus.

Membrane Test

Note: The **LTP Assistant** dialog no longer allows you to set up a resistance test to monitor seal and access resistance on a postsynaptic intracellular recording electrode during baseline stages.

To monitor access resistance over the length of an experiment, the experiment needs to be manually set up using **Edit Protocol > Stimulus > Membrane Test Between Sweeps** and sequencing keys.

Configure some settling time following the end of the presynaptic stimulus before the Membrane Test test pulse is generated. This provides the cell some time to settle after stimulation from the presynaptic cell.

Statistics

The **LTP Assistant** dialog allows three online statistics measurements to be recorded for all input signals during Baseline stages.The three measurements are:

- Peak amplitude
- Peak time
- Slope

These are enabled, initially, by checking the **Statistic**s check box for the channels you want, on the **Baseline** tab. Then later, when you have closed the **LTP Assistant** dialog and started the experiment, you must set appropriate **Search Regions** in the **Edit Protocol** dialog, to capture the measurements properly.

Two different **Statistics**, **Search Region** and **Baseline Region**, are used to measure these. Use **Search Region 1** to measure peak amplitude and peak time, and once you are acquiring stable baseline data you should drag the “1” cursor pair in the **Scope** window to capture the entire event evoked by the stimulus. Configure **Search Region 2** to measure **Slope**, by linear regression for all the data points within the region, and to measure the **Rise Slope** of the event. Cursor pair “2” should then be positioned to capture this feature of the event. The baseline should be set to a section of the sweep where there is no activity (see Figure 4-15):

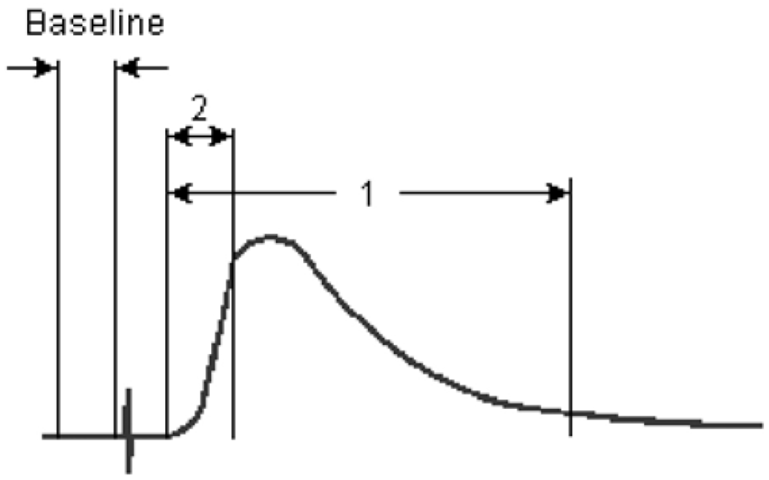


Figure 4-15: Intended statistics baseline and search region positions.

The regions you set are saved automatically when the protocol closes. This means that a baseline protocol opening automatically after a conditioning stage retains the settings you made in the earlier baseline stage.

Statistics measurements are displayed continuously, for the length of the experiment, in the **Online Statistics** window, from where they can be saved as an *.sta file in the normal way. Under default settings, the **Online Statistics** window is saved at the end of each protocol run. This means that so long as the final baseline protocol runs to completion, the statistics for the entire experiment are saved, except with gaps where, for example, a conditioning protocol with no statistics recorded was run. If, however, a very long baseline protocol is set and you stop the experiment manually, statistics are not automatically saved. In this case you need to save them manually with **File > Save As**.

As with all other protocol features in the **LTP Assistant** dialog, the full set of statistics configuration options can be accessed from the **Edit Protocol** dialog, if you want to take more or different measurements from the data.

Chapter 5: Clampex Software Tutorials

5

This chapter presents information to help you actually perform experiments. We walk you through a step-by-step tutorial on setting up experiments. We discuss how Clampex Software can be used to further manipulate data files that have already been recorded. We discuss how to use the Quick Graph feature in pCLAMP Software to automatically plot an I-V (current vs. voltage) curve while Clampex Software acquires data. We finish by presenting several scenarios for doing different types of experiments.

I-V Tutorial

This tutorial is designed to walk a user through the steps of setting up various types of experiments in Clampex Software. The goal of this section is to teach a user how to set up Clampex Software to perform a given experiment, rather than discuss each feature of Clampex Software in detail. For more information on a given feature, refer to the Help file. You may also want to first follow the tutorial *Setting Up Clampex Software for Data Acquisition*, which covers the basic setup procedures for a Digidata digitizer along with the Axopatch 200B, Axoclamp 900A and MultiClamp 700B amplifiers. See pCLAMP Software Software Documentation on page 12.

Goal of the experiment: To examine the I-V relationship of whole-cell calcium currents from cultured neurons in response to the application of a drug.

The experiment consists of:

1. Obtaining a whole-cell recording.
2. Monitoring access resistance.
3. Performing an I-V test.
4. Monitoring the peak current at a single voltage.
5. Applying a drug while measuring the peak current, and then performing an I-V test.
6. Washing off a drug while measuring the peak current, and then performing an I-V test.
7. Monitoring access resistance at end of experiment.

Hardware used:

- Axopatch 200B amplifier, headstage set to $\beta = 0.1$
- Electronically controlled solution changer
- Digidata 1550 digitizer

Selecting Digitizer

One of the first steps in performing an experiment is to select the appropriate hardware for acquisition. This is done with the **Configure > Digitizer** menu option. By default, Clampex Software is installed with the “demo” digitizer active. We will continue to use the demo digitizer for this example, but you must select and configure the digitizer before doing an actual experiment.

Creating Input and Output Signals in the Lab Bench

The **Lab Bench** is used to set the scaling and offset factors for all the input and output channels used in an experiment.

In this tutorial example, an Axopatch 200B amplifier is being used in voltage-clamp mode to record membrane currents in response to square voltage steps. We will set up Clampex Software to acquire the membrane currents on Analog IN #0, and to output the voltage command on Analog OUT #0.

Setting Up Input Signals

To set up the Lab Bench for this example:

1. Select **Configure > Lab Bench**.
2. Click the **Input Signals** tab, and then highlight the Digitizer channel Analog IN #0. You will notice that there are already a number of default signals associated with **Analog IN #0**.
3. Add another signal to **Digitizer Channel > Analog IN #0**, by clicking **Add**.
4. Give the signal the unique name “Im”.

5. Set the **Scaling** for this new signal by entering the **Signal units** and by entering the **Scale factor**. For most amplifiers, there is no need to adjust the **Offset** and therefore we will leave the **Offset** field value at the default "0".
   - In **Scaling > Signal units**, select the unit prefix “p”, and enter the unit type as “A”.
   - In **Scaling > Signal units**, we will use the **Scale Factor Assistant**.

     All we need to know is the gain settings of our amplifier, which can be read off of the front of the instrument. In our example, the signal is membrane current, and we will assume the Axopatch 200B amplifier is set to an α value of 1 and a β value of 0.1.

     To use the **Scale Factor Assistant**:
     - Click **Scale Factor Assistant** and indicate that you have an Axopatch 200 series amplifier connected to Analog IN #0.
     - Click **Next**.
     - Since the experiment will be performed in voltage-clamp mode, select **Mode Setting > V-Clamp**.
     - Select **Config Setting > Whole Cell (β = 0.1)** to match the β setting of our Axopatch 200B amplifier.
     - Select **Preferred Signal Units > pA**.
     - Set **Gain** to "1" to match the α setting of our Axopatch 200B amplifier.

       This is referred to as “unity gain”. The setting of unity gain is very important when you enable telegraphs, a functionality that we discuss in the next section. For now, be sure to set Gain to 1, and set up the Lab Bench to reflect this. We will discuss how Clampex Software deals with changing the gain in the telegraph section.
     - Click **Finish**.

       "1e^{-4} V/pA" automatically computes and is displayed in the **Scale factor** field.
6. Clampex Software has an online **Software RC Filter**with **Lowpass** and **Highpass** filter value options. For our example, leave these turned off.
7. You have now finished setting up the **Analog IN #0** channel to record “Im” signals from the Axopatch 200 amplifier. Since the Axopatch 200B amplifier has a number of different output gains, it is unlikely that you will need to use any additional hardware conditioning for the membrane current signal, otherwise we may want to have an additional gain applied to our signal before it is digitized, to ensure that we are maximizing the dynamic range of our digitizer.

We will now configure **Analog IN #1** to record the membrane potential. You will add the 10 V_m OUTPUT signal from the Axopatch 200 amplifier to **Analog IN #1**, and set the **Scaling unit** to record voltage in "mV". The **Scale Factor Assistant** should not be used for this signal, because it assumes that the **Analog IN** channel is connected to the scaled output of the Axopatch 200 amplifier, and not to other signals such as 10 V_m. Since the gain of the 10 V_m OUTPUT of the Axopatch 200 amplifier is x 10, manually set the **Scale factor** (V/mV) to “0.01” for the signal “V_m”.

We need to consider whether, with a gain of 10, the signal that we are recording adequately utilizes the dynamic range of the digitizer. Typically, the cellular membrane potential will be in the range of ±100 mV, and with a gain of 10, the Axopatch 200B amplifier will output a voltage in the range of ±1 V (100 mV x 10). Given that the dynamic range of the digitizer will be ±10 V, it is apparent that only ~10% of the dynamic range will be utilized (±1 V/±10 V). But, for a 16-bit digitizer such as the Digidata 1550A digitizer, while not optimal, this is resolved into about 6,500 steps, which is still quite adequate.

Setting Up Output Signals

Setting up the output signals is very similar to setting up the input signals. You need to determine an output Scaling factor and Offset for each signal that you create on a given Analog OUT channel.

To set up the Lab Bench for this example:

1. Select **Configure > Lab Bench**.
2. Click the **Output Signals** tab, and then highlight **Digitizer Channels > Analog OUT #0**. By default for **Analog OUT #0**, there are three associated **Signals**.
3. Add another signal to **Digitizer Channels > Analog IN #0**, by clicking **Add**.
4. Give the signal the unique name “VmCmd”.
5. In **Signal units**, select the unit prefix “m”, and enter the unit type as “V”.
6. Click **Scale Factor Assistant** to set up the **Scale factor**.
7. Select **Axopatch 200 series** amplifier as connected to Analog OUT #0.
8. Click **Next**.
9. Since the experiment will be performed in voltage-clamp mode, select **Mode Setting > V-Clamp**.
10. Select **Ext. Command Input > 20 mV/V**.
11. Click **Finish**.

    "20 mV/V" automatically computes and is displayed in the **Scale factor** field.

Setting the Holding Level

The holding level can be set in a number of locations in Clampex Software, including the **Lab Bench** if you check the appropriate **Configure > Overrides** check box. Default behavior has holding levels for each output channel set by the currently loaded protocol, **Edit Protocol > Outputs**. By selecting the Overrides from Lab Bench option, one holding level is maintained on each channel irrespective of the protocol that is loaded. However, wherever the override is set for, immediate changes to the holding level can be made from the Real Time Controls. In addition, if you are running the Membrane Test, you can change the holding level from within its window. The Membrane Test returns the holding level to where it was at the time the test was opened.

Telegraphs

A convenient feature of Clampex Software is the ability to detect various amplifier settings through “telegraphs”. These telegraphs allow you to change the settings of various amplifier controls, such as the gain, without having to reconfigure the Analog IN channels after each change in amplifier gain. For example, on an Axopatch 200 amplifier, the state of the gain (α), filter frequency and capacitance compensation settings can be directly used by Clampex Software. You may recall that we set the gain of our Analog IN #0 signal “Im” to a value of 1, or unity gain. When using telegraphs, Clampex Software performs all calculations relative to this unity gain, and therefore it is very important that unity gain is set up correctly for the signal that will be telegraphed. The telegraphs are configured through the **Configure > Telegraphed Instrument** menu item.

In our example, we will indicate that an Axopatch 200 CV 203BU headstage is being used, and that Analog IN #0 is connected to the scaled output of the amplifier. Since we want to telegraph all three settings, connect the appropriate outputs of the Axopatch 200B amplifier to the appropriate telegraph ports on the Digidata 1550 digitizer. Now, if we close the **Configure > Telegraphed Instrument** dialog box and return to the Lab Bench, we notice that the “Telegraphs” section at the bottom of the inputs tab reports the current status of these three telegraphed parameters. If you enable the telegraphs, you notice that the gain setting in the Scale Factor Assistant is automatically inserted via the telegraphs. Furthermore, any changes that are made to these parameters are automatically updated and used by Clampex Software.

Using Membrane Test

The first step in establishing a whole-cell recording is achieving a gigaohm seal.

Before starting Membrane Test, you need to ensure that it is configured correctly using **Configure > Membrane Test Setup**. By default, Membrane Test is set up to record input signals from the channel Analog IN #0 using a signal setting of I_MTest 0 (pA), and to output Membrane Test signals to the channel Analog OUT #0 on a signal setting of Cmd 0 (mV). We will change the input signal to “Im”, and the Membrane Test output signal to “VmCmd” to match the signals that were just created in the Lab Bench. You can also adjust the amplitude of the pulse when Membrane Test is started and set initial holding levels. We will set the initial holding level to a value of 0 mV.

After configuring Membrane Test, it can be opened from **Tools > Membrane Test**, or from the toolbutton on the Acquisition toolbar. After starting Membrane Test in Bath mode, you will observe the current response to square voltage steps.

You will see that Clampex Software continuously reports the Seal Resistance value. This value can be logged to the Lab Book at any time by pressing the => **Lab Book** button.

You can also adjust how often the pulses are delivered using the frequency slider.

In some cases, you may want to hyperpolarize the cell to aid in seal formation. This is done by setting the “Holding (mV)” value.

Editing a Protocol

The details of an acquisition are defined in the acquisition protocols. As this is a fairly complex issue, we discuss only the features that are relevant for setting up our example. In this case, we describe how to build two protocols: one that performs an I-V, and one that repetitively steps the cell to the same voltage. We have provided these demonstration protocol files for you, named **tutor_1a.pro** and **tutor_2a.pro**, as well as the sequencing key file, named **tutorA.sks**, in **..\Program Files\Molecular Devices\pCLAMP10.4\Sample Params**.

Protocol 1: Conventional I-V

To get a fresh protocol, select **Acquire > New Protocol**, which generates and opens a new protocol. We will examine each relevant tab of the protocol to ensure that the protocol is set up correctly.

Mode and Rate

To set the Mode/Rate part of the example protocol:

1. Select **Acquisition Mode > Episodic stimulation**.
2. Set the **Sweeps/run** to a value of 15, because our I-V will step from -80 to +60 in 10 mV increments, which needs 15 steps.
3. Keep the **Start-to-Start Intervals** time for **Sweep(s)** set to **Minimum** to have the protocol execute as fast as possible.
4. Since we will sample the data at 10 kHz, set the **Sweep duration(s)** to "0.2064 s", to ensure that each sweep is long enough to gather enough data.

Input and Output Channels

In the **Inputs** tab, you activate the **Analog IN Channels** that you are acquiring from, and indicate which signal that you created in the Lab Bench you would like to record on each channel. For our protocol example, activate **Channel #0** and **Channel #1**, and specify the “Im” and “Vm” signals respectively.

In the **Outputs** tab, you define which **Analog OUT Channels** and signals are active. For our protocol example, set the **Channel #0** to the signal “VmCmd” that you created in the Lab Bench. This also defines a **Holding Level** of "-80 mV".

Waveform

Defining the stimulus waveform on the **Waveform** tab is the next logical step.

1. For our protocol example, set the source of the **Analog Waveform** to be defined by **Epochs**, which are described by the table in the dialog.
2. **Digital Outputs** should be disabled.
3. Leave the **Intersweep holding level** default value of “Use Holding” setting.
4. Use the **Epoch Description** table to define the actual epochs.

5. Define the **A**, **B**, and **C** epochs as step epochs by clicking in the **Type** table cell and selecting **Step** for each of the three epochs. For our protocol example, set the following **Epoch Descriptions** in the table dialog:

   You can also use the <PgUp> and <PgDn> buttons to cycle through the epoch types.

   Table 5-1: Epoch descriptions for Protocol 1.

	Epoch A	Epoch B	Epoch C
Type	Step	Step	Step
Sample rate	Fast	Fast	Fast
First level (mV)	-80	-80	-80
Delta level (mV)	0	10	0
First duration (ms)	50	100	50
Delta duration (ms)	0	0	0
Digital bit pattern (#3-0)	1111	0000	0000
Digital bit pattern (#7-4)	0000	0000	0000
Train rate (Hz)	0	0	0
Pulse width (ms)	0	0	0

When finished, you can preview the waveform by clicking **Update Preview** at the bottom of the dialog.It is often useful to have the **Waveform Preview** dialog open while you are creating the protocol, and click **Update Preview** every time you want to see the effects of something you have changed. It is handy to shrink the size of the **Waveform Preview** dialog and keep it displayed in a corner of the screen.

Trigger

We will leave the **Trigger** tab in its default state for this protocol.

Stimulus

The **Stimulus** tab is where you define additional stimulation parameters, such as presweep trains, leak subtraction and arbitrary values from a user list. For this demonstration, using the demo digitizer, we have not added **P/N Leak Subtraction** to our protocol in the **Stimulus** tab, however, when recording from real cells you might add **P/N Leak Subtraction** to the protocol. We have defined a **P/N Leak Subtraction** stimulus in the protocol, but left it disabled. Check the check box to view our configuration: 5 subsweeps of polarity opposite to the waveform to occur before each individual sweep, with a minimum start-to-start time. The signal to which leak subtraction is applied is Im. You can optionally display the raw or corrected data.

Statistics

Clampex Software can display online various spike-oriented shape statistics on an acquired signal. Go to the **Statistics** tab of the protocol editor and check the box **Shape Statistics**. In our example, we will measure the peak amplitude and time of peak on the “Im” signal.

If we were recording actual inward calcium currents, we would want to set the polarity of the peak detection to negative. Since we are using the demo driver, we will set the polarity to positive to illustrate the functionality of the statistics display. We can define a search region and a baseline region either in terms of time or epochs. We’ll define the search region as Wave 0 Epoch B, and the baseline region as Wave 0 Epoch A. If you want to ensure that you are searching the correct regions of the waveform, use the Update Preview button. If **Shape Statistics** are enabled, the **Waveform Preview** window shows the **Baseline Region** and first **Search Region** used in the peak detection.

The statistics can be automatically saved whenever an acquisition is saved to disk, although if this option is not enabled, you can always manually save the statistics from the Online Statistics window with **File > Save As**. For now, we will not enable this option.

That completes setting up your first protocol. Save the protocol using the **Acquire > Save Protocol As** menu item, with a name such as “tutor_1.pro”.

Protocol 2: Repetitive Current Stimulation

The second protocol is one that is designed to repetitively activate the current by giving step pulses from -80 to +10 mV. This type of protocol if useful when one wants to monitor the effect of a drug over time.

To create this protocol, we will modify the existing protocol that we just created. Use the **Acquire > Edit Protocol** menu item to edit the current protocol. In the **Mode/Rate** tab, adjust the **Sweeps/run** to "6". For demonstration purposes we have made the protocol as fast as possible, but for real use you might want to adjust the **Sweep Start-to-Start Interval** to "10 s" and re-save the protocol. In the **Waveform > Channel #0** tab, change the **First level** of Epoch B to "+10 mV" and the **Delta level** to "0". This creates a protocol that steps to +10 mV for 100 ms in every sweep. Click the **Update Preview** button to ensure that you have changed the waveform correctly. Save the protocol under a new name, such as “tutor_ 2.pro” using **Acquire > Save Protocol As**.

At this point, you have configured Clampex Software correctly for the hardware that you are using for the experiment, and have defined the two protocols that you need to use in order to perform the experiment. Additionally, you have learned about the Membrane Test, which aids in the formation and monitoring of a gigaohm seal.

Before you save any data to disk, define the file names and locations where the data will be saved (and opened from). This is done in the **File > Set Data File Names** dialog box. For this example, we will use “tutor” as the filename prefix. Select long filenames, disable the date prefix, and type “tutor” in the file name field.

Using Sequencing Keys

Sequencing keys are an extremely powerful and flexible feature of Clampex Software that allow you to define an entire experiment composed of several different protocols, interspersed with Membrane Tests, and parameter changes such as digital outputs to control solution changes. Sequencing keys allow you to associate an event such as running a protocol, running a Membrane Test, or changing a parameter, with an individual keystroke and its associated toolbutton. Each keystroke can then be linked to another keystroke, allowing the user to set up a chain of events that define an entire experiment.

In this section we discuss how to use the information from the previous sections to design a complete experiment that can be run by the touch of a single button. Before setting up the sequencing keys, it is often helpful to determine the flow of the experiment. In our example, the experiment is to examine the I-V relationship of whole-cell calcium currents from cultured neurons in response to application of a drug.

We will use the sequencing keys to automate the experiment after whole-cell access has been achieved. We can think of the experiment in terms of the protocols and events, such as solution changes, that we use.

Table 5-2: Experimental steps, events, and protocols.

Step	Event	Protocol
1	Obtaining a whole-cell recording	
2	Monitoring access resistance	Membrane Test
3	Performing an I-V	Tutor_1.pro
4	Monitoring the peak current at a single voltage	Tutor_2.pro
5	Start applying a drug	Digital Out ON
6	Monitoring the peak current at a single voltage	Tutor_2.pro
7	Monitoring access resistance	Membrane Test
8	Performing an I-V	Tutor_1.pro
9	Washing off the drug	Digital Out OFF
10	Monitoring the peak current at a single voltage	Tutor_2.pro
11	Monitoring access resistance	Membrane Test
12	Performing an I-V	Tutor_1.pro

Now that we have established what the events are that we would like to define in the sequencing key setup, we can proceed to configure a sequencing series.

You set up the sequencing keys through the **Configure > Sequencing Keys** menu command or, if the Sequencing Toolbar is active, by pressing the first button on the **Sequencing Keys** toolbar:

- After you open the **Sequencing Keys** dialog, click **New Set** to define a new set of sequencing keys.
- To define a new event in the sequence, click **Add**. You are prompted as to what key you would like to define. Choose the default **Ctrl+1** for the **Add a key definition for** field.
- You can define the following operations that can occur when the key is activated: **Set Lab Bench Holding Levels** for Analog OUT channels, **Digital OUT Bit Pattern**, **Membrane test**, **Protocol** and **Prompt** messages.
- For each key you define, use the **Sequencing** tab to define what happens when the operation completes. In this case, we link each event to the **Next key**, so that we can set up a continuous sequence of events.
- We can also define when the next event will occur. For example, we can specify that we would like the next event to occur **When acquisition finishes**, or **After (s)** a fixed number of seconds from the start of the key.

In the following example, we will define <**Ctrl+1**> to run the Membrane Test for 1 s, and when it completes, we will immediately run the Tutor_1 protocol while saving the data to disk.

1. In the **Operations** tab, select **Membrane test**.
2. On the **Sequencing** tab, define the **Next key** as **Ctrl+2**.
3. Specify that you want to **Start next key After** "1" s has elapsed from **start of key**.
4. Click **OK** to finish defining <Ctrl+1>.
5. Click **Add**to add the <**Ctrl+2**> key.
6. Select **Operation > Protocol**.
7. Select **Action > Record**.
8. Click **Browse** and select the protocol file *Tutor_1*.
9. Set **Repetition count** to "1" so that the protocol only runs once.
10. On the **Sequencing** tab, set **Next key > Ctrl+3 > When acquisition finishes**.
11. Click **Add**to add the **Ctrl+3**> key.
12. Select **Operation > Protocol > Action > Record**.
13. Click **Browse** and select the protocol file *Tutor_2*.
14. On the **Sequencing** tab, set **Next key > Ctrl+4 > When acquisition finishes**.

To toggle an electronic solution changer on and off, you can use the **Digital OUT Bit Pattern** on the **Operations** tab. Select which digital bit you would like to turn on or off. For example, you could connect the solution changer to Digital OUT #0 and you would enable Digital OUT #0 when you wanted to activate a solution change, or disable the Digital OUT when you want to end the solution change.

In our experiment, the **<Ctrl+4>** key is defined to flip Digital OUT #0 high:

1. Add the <**Ctrl+4**> sequencing key.
2. In the **Operations** tab, select **Parameters** and enable **Digital OUT Bit Pattern**.

3. Deselect all bits except bit "0".

Note: You might have one bit unavailable here, overridden by the **Lab Bench > Output Signal > Set digital OUT bit high during acquisition** check box.

4. On the **Sequencing** tab, set **Next key > Ctrl+5**.

In our example, we want to keep our Digital OUT high until we set it back to low after we observe an effect of the drug on the cell. To keep the digital bits high when we run subsequent protocols, we must set in **Configure > Overrides > Use digital holding pattern from Lab Bench**. When this option is enabled, any change to the digital outputs are stored in the Lab Bench, and are only changed when the Lab Bench is changed, and not by loading and running a protocol.

Using the outline of the experiment you can set up the entire sequence. Use the **Copy** button to create new keys, followed by **Properties** to edit the new key. When you are done the sequence should look like the following table.

Table 5-3: Key sequence for experiment.

Key	Next Key	Type	Description
<Ctrl+1>	<Ctrl+2>	Membrane Test	Start-to-Start = 1
<Ctrl+2>	<Ctrl+3>	Protocol	Record using “Tutor_1.pro”. 1 reps. Protocol time takes priority.
<Ctrl+3>	<Ctrl+4>	Protocol	Record using “Tutor_2.pro”. 1 reps. Protocol time takes priority.
<Ctrl+4>	<Ctrl+5>	Parameters	Digital = 00000001 Start-to-Start = 1
<Ctrl+5>	<Ctrl+6>	Protocol	Record using “Tutor_2.pro”. 1 reps. Protocol time takes priority.
<Ctrl+6>	<Ctrl+7>	Membrane Test	Start-to-Start = 1
<Ctrl+7>	<Ctrl+8>	Protocol	Record using “Tutor_1.pro”. 1 reps. Protocol time takes priority.
<Ctrl+8>	<Ctrl+9>	Parameters	Digital = 00000000. Start-to-Start = 1
<Ctrl+9>	<Ctrl+Shift+1>	Protocol	Record using “Tutor_2.pro”. 1 reps. Protocol time takes priority.
<Ctrl+Shift+1>	<Ctrl+Shift+2>	Membrane Test	Start-to-Start = 1
<Ctrl+Shift+2>	None	Protocol	Record using “Tutor_1.pro”. 1 reps

Save the sequence with **Save Set**, naming it something like “Tutorial.sks”.

After closing the **Sequencing Keys** dialog, you can start the entire experiment by clicking <**Ctrl+1**>, or the second icon button in the **Sequencing** toolbar.

A text description of a sequencing set can be copied to the clipboard with the **Clipboard** button, and then pasted into a word processor, to allow you to print out the sequencing series for your lab notebook.

Displaying and Measuring Acquired Data

After you have acquired data, Clampex Software provides some basic analysis functionality. It should be noted that data is best analyzed in pCLAMP Software, but Clampex Software allows you to browse data and perform simple measurements.

You can open data files by selecting **File > Open Data**. This dialog also provides a number of options that control how the data is displayed once it is read into Clampex Software. For example, you can determine if each data file is opened into a new window, how to control the axis scaling, and you can set the number of sweeps or signals that you would like displayed. Open the last data file that was recorded by selecting **File > Last Recording**.

When the data is opened in the **Analysis** window, there are four cursors that can be moved by clicking and dragging. Many of the options that control the **Analysis** windows are available by clicking the right-mouse button while in various areas of the window.

For now, we will simply measure the peaks of our calcium currents that we acquired during the last I-V of the sequencing series, which was the last file that was recorded, and which should now be displayed. We will measure the minimum value between cursors 1 and 2:

1. Set cursors 1 and 2 so that they delimit the region of the peak inward calcium current.
2. Select **Tools > Cursors > Write Cursors**. This function can also be performed by pressing the "check" toolbutton at the top left of the Analysis window or from the **Analysis** toolbar.
3. Double-click on the **Results** window at the bottom of the screen, or select **Window > Results**. Observe that for each trace, a number of parameters were measured.
4. Select **View > Window Properties** when the **Results** window is displayed to specify which measurements to display and in which order they display.
5. Save these measurements to a text file by selecting **File > Save As**.

There is extensive control over various features of the Analysis window. The online Help covers this in detail. A useful reminder is that most properties can be set by right-clicking in various areas of the Analysis window. You can set various properties of the cursors and the Analysis window through this functionality.

Real-Time I-V Plotting in pCLAMP Software

Despite the power of Clampex Software itself, we will now coordinate its acquisition with I-V plotting in pCLAMP Software. For this demonstration you may first wish to return pCLAMP Software to its factory default values. Do this by closing all Axon Instruments software and then running the program **Reset to Program Defaults** found in the **Molecular Devices > pCLAMP 10.6** program folder. Select **Clampfit** as the application to return to default values.

Now load both pCLAMP Software and Clampex Software and in Clampex Software load your protocol file *tutor_1* or the protocol file provided as *tutor_1a.pro*. In pCLAMP Software, check **Apply automatic analysis** in **Configure > Automatic Analysis**. Using the Record function in Clampex Software (do not use the sequencing keys yet) record a file, which should then appear in an **Analysis** window in pCLAMP Software.

We will now use this file to create our I-V plotting routine and then we will set up pCLAMP Software to run an Automatic Quick I-V Graph concurrently with our experiment in Clampex Software:

1. With the focus on the **Analysis** window in pCLAMP Software, select **Analyze > Quick Graph > IV**. We want to plot the peak current Im against the command voltage VmCmd.
2. For the X Axis Waveform group choose Epoch, and “Epoch B level” for the X axis of the plot.
3. For the Y Axis, for Signal choose “Im”. We want to detect the positive peak in Epoch B so select Peak and choose “Positive” for the Polarity. For Region choose “Epoch B - Waveform 0".
4. We apply a smoothing factor of 5 points in peak detection.
5. Under Destination Option choose “Append” because we want to see all I-V curves in the experiment plotted together.
6. When you close the form a graph window will appear with an I-V plot.
7. Minimize all applications on the Windows desktop except Clampex Software and pCLAMP Software. Move the mouse to a blank part of the Windows taskbar, right-click and choose Tile Windows Horizontally to give equal place to the two applications. You may now want to arrange the Windows within each application to make maximal use of the desktop space. In pCLAMP Software minimize or close all windows except the Analysis window and the Graph Window and arrange the windows with a Window > Tile Vertical command in pCLAMP Software.
8. Before beginning the experiment, open Configure > Automatic Analysis, check Generate Quick Graph and select I-V. In this demonstration, make sure that Prompt for Arithmetic is not checked. When you run an actual experiment at slower rates of acquisition, you would enable Subtract Control File in order to subtract control sweeps from the sweeps to be analyzed automatically and plotted as I-V or Trace versus Trace graphs.
9. In Clampex Software, load the tutorial.sks sequencing file or the sequencing file provided as tutorA.sks with the command Configure > Sequencing Keys > Open Set. If the Online Statistics window is visible and has data in it, clear the data with the command Edit > Clear Statistics. Now start the experiment by pressing the <Ctrl+1> toolbutton. As the data is acquired you should see six I-V curves plotted in the Graph window.

Displaying and Measuring Acquired Data

After you have acquired data, Clampex Software provides some basic analysis functionality. It should be noted that data is best analyzed in pCLAMP Software, but Clampex Software allows you to browse data and perform simple measurements.

You can open data files by selecting **File > Open Data**. This dialog also provides a number of options that control how the data is displayed once it is read into Clampex Software. For example, you can determine if each data file is opened into a new window, how to control the axis scaling, and you can set the number of sweeps or signals that you would like displayed. Open the last data file that was recorded by selecting **File > Last Recording**.

When the data is opened in the **Analysis** window, there are four cursors that can be moved by clicking and dragging. Many of the options that control the **Analysis** windows are available by clicking the right-mouse button while in various areas of the window.

For now, we will simply measure the peaks of our calcium currents that we acquired during the last I-V of the sequencing series, which was the last file that was recorded, and which should now be displayed. We will measure the minimum value between cursors 1 and 2:

1. Set cursors 1 and 2 so that they delimit the region of the peak inward calcium current.
2. Select **Tools > Cursors > Write Cursors**. This function can also be performed by clicking the “check” toolbutton at the top left of the **Analysis** window or from the **Analysis** toolbar.
3. Double-click on the **Results** window at the bottom of the screen, or select **Window > Results**. Observe that for each trace, a number of parameters were measured.
4. Select **View > Window Properties** when the **Results** window is displayed to specify which measurements to display and in which order they display.
5. Save these measurements to a text file by selecting **File > Save As**.

There is extensive control over various features of the **Analysis** window. The Help file covers this in detail. A useful reminder is that most properties can be set by right-clicking in various areas of the **Analysis** window. You can set various properties of the cursors and the **Analysi**s window through this functionality.

Real-Time I-V Plotting in pCLAMP Software

Despite the power of Clampex Software itself, we will now coordinate its acquisition with I-V plotting in pCLAMP Software. For this demonstration you may first wish to return pCLAMP Software to its factory default values. Do this by closing all Axon Instruments software and then running the program **Reset to Program Defaults** found in the **Molecular Devices > pCLAMP 10.6** program folder. Select **pCLAMP Software** as the application to return to default values.

Now load both pCLAMP Software and Clampex Software and in Clampex Software load your protocol file *tutor_1* or the protocol file provided as *tutor_1a.pro*. In pCLAMP Software, check **Apply automatic analysis** in **Configure > Automatic Analysis**. Using the Record function in Clampex Software (do not use the sequencing keys yet) record a file, which should then appear in an **Analysis** window in pCLAMP Software.

We will now use this file to create our I-V plotting routine and then we will set up pCLAMP Software to run an Automatic Quick I-V Graph concurrently with our experiment in Clampex Software:

1. With the focus on the **Analysis** window in pCLAMP Software, select **Analyze > Quick Graph > IV**. We want to plot the peak current Im against the command voltage VmCmd.
2. For the **X Axis Waveform** group choose **Epoch**, and **Epoch B level** for the X axis of the plot.
3. For the Y Axis, for **Signal** choose “Im”. We want to detect the positive peak in Epoch B so select Peak and choose “Positive” for the Polarity. For Region choose “Epoch B - Waveform 0".
4. We apply a smoothing factor of 5 points in peak detection.
5. Under **Destination Option** choose “**Append**” because we want to see all I-V curves in the experiment plotted together.
6. When you close the form a **Graph** window will appear with an I-V plot.
7. Minimize all applications on the Windows desktop except Clampex Software and pCLAMP Software. Move the mouse to a blank part of the Windows taskbar, right-click and choose **Tile Windows Horizontally** to give equal place to the two applications. You may now want to arrange the Windows within each application to make maximal use of the desktop space. In pCLAMP Software minimize or close all windows except the **Analysis** window and the **Graph** window and arrange the windows with a **Window > Tile Vertical** command in pCLAMP Software.
8. Before beginning the experiment, open **Configure > Automatic Analysis**, activate **Generate Quick Graph** and select **I-V**. In this demonstration, make sure that **Prompt for Arithmetic** is not checked. When you run an actual experiment at slower rates of acquisition, you would enable **Subtract Control File** in order to subtract control sweeps from the sweeps to be analyzed automatically and plotted as I-V or Trace versus Trace graphs.
9. In Clampex Software, load the *tutorial.sks* sequencing file or the sequencing file provided as *tutorA.sks* with the command **Configure > Sequencing Keys > Open Set**. If the **Online Statistics** window is visible and has data in it, clear the data with the command **Edit > Clear Statistics**. Now start the experiment by clicking the <**Ctrl+1**> toolbutton. As the data is acquired you should see six I-V curves plotted in the **Graph** window.

Membrane Test Tutorial

This tutorial is designed to provide a quick overview of the performance and special features of the Membrane Test. It takes 10–15 minutes to complete the tutorial. It is important that you proceed through the tutorial in sequence for this tutorial to be useful.

The experiment consists of:

1. Set Clampex Software to demo mode.

   Go to the **Configure > Digitizer** dialog, select the **Change** button and select “Demo” from the **Digitizer Type** list.
2. Open the **Membrane Test**.

   Push the circuit diagram toolbar button or select **Tools > Membrane Test**. Confirm that the OUTPUT signal is Cmd 0 (mV) and that the INPUT signal is I_Mtest 0 (pA). If this is not the case, change the settings in **Configure > Membrane Test Setup** .You can do this without closing the **Membrane Test** window.
3. Select the **Membrane Test** mode.

   Click on the **Cell** button, and make sure that input channel #0 is selected. Then, click on the **Play** button to start the membrane test.
4. Change the voltage pulse amplitude.

   Change the pulse amplitude in the **Pulse** field to 20 mV, and then from 20 mV to 10 mV either by clicking on the down arrow button or by typing in the number followed by <Enter> on the keyboard. Notice that the capacitive transients from the current record in the **Scope** display decrease in amplitude. Now change the Pulse amplitude back to 20 mV and observe that the capacitive currents are larger.
5. Change the scale in the scope display.

   Click the up arrow button on the left side of the window by the Y axis a couple of times to change the amplitude scaling. Click the **Auto Scale** button next to the meters and observe the change.
6. Change the measurement update rate.

   Move the slider marked **Slower – Faster**. The frequency is shown to the underneath. Notice near the Slower end the calculations fail because there are too few points in the decay of the transient. As you move the slider, observe that the frequency value (measurement update rate) changes. The capacitive transient current records change, as well as the time scale on the X axis. The membrane parameter values also change, depending on the accuracy of the calculation, which depends on the accuracy of the curve fit (red line on the decay phase).
7. Check the **Averaging** check box to enable averaging.

   Click the **Averaging** check box to enable the averaging function. Notice that the noise of the capacitance signal decreases in the **Online Statistics** window. Click on the **Averaging** check box again and observe the noise increase (wait a moment for the new data to appear). Notice that the recording speed displayed below the **Slower – Faster** slider increases dramatically.
8. Change the number of pulses in the pulse train.

Open **Configure > Membrane Test Setup**. In the **Pulse Train Protocol** section, set **Number of pulses in train** to 4. Change the step level (mV) to 40. Select **OK** to return to the **Membrane Test** window. Click on the **Pulse Train** button and observe the 4 pulses appear sequentially in the **Scope** display. Notice also that the values in the **Online Statistics** window do not update during the pulses.

9. Save values to the Lab Book.

   Push the **Lab Book** button. The values of the five Membrane Test parameters are saved to the Lab Book. Confirm this by opening the **System Lab Book** window. Return to the **Membrane Test** window.

10. Save, then view results.

    Make the **Online Statistics** window active by clicking on its title bar. Select **File > Save As** to save the data in the **Statistics** window to a *.sta file. Note the file name and destination folder and click **OK**. Close the **Membrane Test** window. Open the file that was just created by selecting **File > Open Other > Results**. Change the **Files of type** field to “All Files (*.*)”. Select the *.sta file that you just saved. It is also a good exercise to import the file into a spreadsheet program such as Excel, SigmaPlot, or Origin.

Conclusion

This concludes the Membrane Test Tutorial. Be sure to reconfigure the Digitizer for data acquisition. If changes were made for the tutorial, you may need to change OUTPUT or INPUT signals to their pre-tutorial settings in the **Membrane Test Setup** dialog.

Scenarios

Cell-Attached Patch-Clamp Single-Channel Recording

The Clampex Software **Membrane Test** tool can be used to monitor seal resistance until a suitably high-resistance seal is obtained. Currents can then be recorded in **Gap-free** mode, or alternatively in **Variable-length events** mode. Important information like the time of the recording, the holding potential and telegraph information are stored in the file header and can be retrieved by selecting **File > Properties**. Additional information about the experimental conditions can be added as a voice tag or a typed comment tag. The toolbar buttons can be customized to present your preferred form of comment tag.

The simplest procedure would be to use gap-free recording. If event frequency is reasonably high, gap-free recording is probably the best mode to use. However, the event frequency is often not predictable until the data starts to appear. Based on the nature of the patch, it may be less than a second, or more than a minute, before enough events occur to permit a good analysis of the events. One way to be prepared is by having two protocols ready to run: a gap-free protocol if event frequency is high, and an event-triggered protocol if event frequency is low.

Sequencing keys can then be used to automatically load one protocol or the other, based on your initial impressions of the data. At this point, your subjective experience of your preparation and your specific experimental goals will determine how to proceed. You need to consider how long the patch is likely to be stable, and how many events are sufficient to make a particular experimental measurement.

For example, if you just want to generate amplitude histograms at multiple voltages to calculate a single channel I-V, then you probably just need a few hundred events at each voltage. As a consequence, the recordings may be relatively short. On the other hand, if you are primarily interested in knowing about channel kinetics, then relatively long recordings may be in order. Some experience with how long the patch is likely to be stable is another important factor, and also whether the single-channel activity itself is going to be stable or show rundown. The real-time display of threshold-based statistics may be useful for making decisions about how to best get the desired data from the patch. For example, you should be able to detect rundown by generating a running estimate of event frequency in the statistics display.

Since the **Analog OUT** channels and **Digital OUT** channels allow the user to change experimental parameters in real time, it is possible to get a clear representation of their cause and effect relationships in experiments. By using a BNC “T” connector to send the same signal to a peripheral device such as a valve controller, and to one of the acquisition channels, you can simultaneously record the output signals in addition to the experimental response (recordings may then be viewed in a continuous or segmented format). However, this dedicates an entire Analog IN channel to a parameter that changes only occasionally. For many applications, the better approach is to take advantage of the fact that changes in the output signal levels and the digital outputs can be tagged internally. Tags are added to the data file each time you change an output voltage or digital output. Or, within the **Trigger** tab of the **Edit Protocol** dialog, you can select **Options > External Tag**, and use the **TRIGGER IN/TAG BNC** on the digitizer to mark changes in the file while recording.

If you want to change the holding potential to determine the voltage dependence of conductance or kinetic parameters, the simplest thing is to stop the trial, adjust the V_{out} channel or the amplifier setting, and start another gap-free trial. However if you want to see if there are immediate (for example non-stationary) effects of the voltage jump, then you can change the V_{out} signal in real time and have a comment automatically inserted into the recording at the time of the voltage jump. Data ranges can be selected with the cursors and saved as separate binary integer files to be analyzed independently later.

When event frequency is low in single-channel experiments, a great deal of disk space can be conserved if the system only records during the time when the channel is active. Variable-length events mode is ideally suited for this. The pretrigger interval can be set to record data before the trigger, to ensure that baseline drift can be evaluated and corrected in the subsequent analysis of the data. The pretrigger interval also sets the amount of data that is saved after the signal recrosses the threshold, so that with this value set correctly, events are saved in their entirety.

When recorded at a high bandwidth, single-channel data may be too noisy for event-triggered acquisition to be practical. Baseline drift can also be a problem with threshold-based event detection. Event-triggered acquisition can also require that the baseline is adjusted in real time to keep the position of the data constant relative to the trigger threshold. One option to get around these problems is to acquire the data on two Analog IN channels, one filtered and the other set at a higher bandwidth. Trigger information can then be taken from the filtered signal. The higher bandwidth setting on the second Analog IN channel preserves single-channel data for analysis. Since the filter settings in the Clampex Software Lab Bench are applied to each channel separately, this can be accomplished entirely within Clampex Software. For example, the Clampex Software lowpass filter of Analog IN channel 0 could be set to 5 kHz, while the lowpass filter of Analog IN channel 1 is set to 1 kHz along with a highpass filter set to 0.5 Hz. This basically makes Analog IN channel 1 AC-coupled and essentially removes baseline drift. With this setup, events are detected on the AC-coupled 1 kHz channel, while single-channel data is saved at 5 kHz.

An important caveat to this approach is that the minimum event duration is in part a function of the filter setting on the trigger channel, and in part a function of the filter setting on the data channel. Single, brief events may not trigger acquisition. However, brief events within bursts of activity will be resolved at the higher bandwidth. Therefore, this approach would best be applied to the analysis of kinetics within bursts of activity. If you are changing the holding potential while conducting a variable-length event-driven acquisition, you should also keep in mind that the change in channel amplitude also alters the event-detection sensitivity.

Isolated Patch-Clamp Single-Channel Recording

The big advantages of isolated patch-clamp recording (whether inside-out or outside-out) over cell-attached recording is that it provides the experimenter the ability to know precisely the transmembrane voltage, ion gradients and drug concentrations. While conditions on the pipette side of the membrane are usually held static, the conditions on the bath side of the pipette may be dynamically controlled as a way to study the regulation of P(open). As such, these patch-clamp configurations are ideal for the study of ligand-dependent channel activation. If ligands are applied slowly or have stationary effects once applied, data acquisition from these patch-clamp configurations can be treated the same as data acquisition from cell-attached patches.

However, the real strength of these recording modes comes from the ability to coordinate acquisition with solution changes made by an electronic valve controller or a piezo-electric switching device. This way, each patch can be used to establish both the control and the experimental response, as well as being used to follow non-stationary activity from the moment that a solution change is made.

To accomplish this, either the data acquisition should be controlled by the drug application system (DAS), or the drug application should be controlled from Clampex Software. In order to coordinate a single run of acquisition with the use of a DAS, a signal from the DAS can be sent either to the **Trigger source > Digitizer START Input** or to an acquisition channel. If a signal from the DAS is sent to the **Digitizer START Input**, then the **Digitizer START Input** must be appropriately configured in the **Edit Protocol** dialog **Trigger** tab. Once a start signal is received, acquisition continues until the **Stop** button is clicked, a preset duration is achieved, or the disk space limitation is reached. The disadvantage to this approach is that no pretrigger data are acquired.

The alternative approach is to feed the signal from the DAS into one of the Analog IN channels configured as a trigger for variable-length event-driven acquisition. In this configuration, a pretrigger interval can be set. The system can then be armed by clicking the **Record** button. Data are not written to disk until a signal is received from the DAS. Acquisition can be terminated by the **Stop** button, a preset duration, disk space limitation, or by the return of the DAS signal to a value below the trigger level.

A DAS may also be controlled in real time with either an Analog OUT or Digital OUT channel of the Digidata digitizer. In this case, the gap-free recording mode of Clampex Software may be used and configured to tag automatically the changes made to the output channel that provides the voltage-step commands to the DAS. While a voltage-step command may be ideal to gate the function of a valve driver, it is not a suitable input to a piezo-electric solution switcher. Too abrupt a change in the command voltage to a piezo-electric device causes ringing in the motion, and may even damage the piezo-electric crystal stack. However, it is possible to control a piezo-electric solution switcher, such as the EXFO Burleigh LSS-3000, if the digitizer's voltage step command is conditioned by a simple RC circuit.

Gap-free recording containing periods of channel activity corresponding to agonist (or other drug) applications can be prepared for analysis by opening the file into a Clampex Software **Analysis** window and using cursors 1 and 2 to delineate segments of interest. **File > Save As** can be used to save each segment as an *.abf file for subsequent analysis in pCLAMP Software.

If recordings were made in the variable-length event-driven mode, and the DAS step command was used as a trigger channel, when the saved data file is opened, it can be viewed as sweeps with **View > Data Display > Sweeps**. If the sweep-length exceeds the length of the longest triggered acquisition, then each triggered acquisition appears as a separate sweep. Data can then be sorted on the basis of sweeps and selectively saved.

Whole-Cell Recording of Responses to Drug Applications

The responses to drug applications in the recorded whole-cell mode normally take the form of macroscopic currents unsuitable for analysis as single-channel events. Rather, the responses are analyzed in terms of the peak amplitude and the kinetics of rise and fall. The Fixed-length events mode of event-triggered acquisition is ideal for this kind of data. The acquisition of the data as discreet sweeps provides the ability to detect, synchronize and average events. This mode is designed to capture any class of events that transpire within an anticipated (fixed) length of time. This mode creates a sequence of sweeps that can be averaged or analyzed for individual shape statistics in pCLAMP Software.

It is worth noting the three acquisition characteristics that distinguish fixed-length acquisition from high-speed oscilloscope acquisition, although for post-acquisition analysis there is no difference:

- In Fixed-length events mode there is one sweep per event.
- Fixed-length events mode guarantees not to miss any events, whereas in High-speed oscilloscope mode multiple events can occur during a sweep and are missed as individual events.
- Most important, the trigger for each sweep in Fixed-length events mode can be uniformly coordinated with an external process such as a drug application system (DAS). For these kinds of data, it is probably most practical to coordinate the DAS and the acquisition with an external pulse generator. The pulse generator would send a signal in parallel to the DAS and the digitizer’s Analog IN channel serving as the event trigger input. However, it is possible to use an Analog OUT signal from the digitizer for the same purpose. Using Real Time Controls in Clampex Software , if the Analog OUT signal is sent in parallel to the DAS and to the Analog IN channel serving as the event trigger, then data is written to disk only when a suitable Analog OUT signal is sent. The Analog OUT value can then be set back down and thus re-armed for another acquisition.

For example, with the data being read into Analog IN 0, and Analog OUT 0 used to control a valve driver, set the acquisition protocol for fixed-length event-driven acquisition with two Analog IN channels (0 and 1). Use a BNC “T” connector to send the Analog OUT 0 signal to both the valve controller and to Analog IN 1. In the **Trigger** tab of the **Edit Protocol** dialog, select IN 1 as the trigger and a trigger threshold value below what is required for the valve controller. For example, if the valve controller needs a +5 V signal to activate the valve, set the trigger threshold for +3 V. Now the acquisition session can be armed by clicking **Record**. When you want to activate the valve controller, you can type “5” into the text field for Analog OUT 0 and then trigger the valve when you click <Enter>. This should generate a response and a record of the relative timing of the valve control signal. To turn off the valve, type “0” into the text field and click <Enter>.

Whole-Cell Recording of Spontaneous Activity

While it is tempting to use Gap-free recording mode for events such as miniature synaptic currents, in fact, Fixed-length events mode of event-triggered acquisition is also ideal. Each event that passes a threshold triggers a sweep of uniform length, and while the sweeps are in a temporal sequence, they can also be overlapping in time. If multiple triggers are detected during the same interval, then the data are saved as multiple overlapping sweeps. This permits complete events to be saved independent of event frequency. Having the data in this format then permits quick analysis of shape statistics, such as amplitude and rise times, to be conducted in pCLAMP Software, instead of having to first rerun Event Detection on the raw data in pCLAMP Software to recapture the events for subsequent analysis.

Whole-Cell Recording of Evoked Activity

In most cases, **Fixed-length event**-triggered acquisition is best for recording whole-cell evoked activity. It is possible to record multiple kinds of responses to stimuli in the event-triggered modes by setting up the trigger channel to trigger off an external device, such as a stimulus isolation unit that is used to shock a presynaptic nerve bundle. The pulse controlling the stimulator may also be read into Analog IN channel 0, while postsynaptic responses are acquired on Analog IN channel 1. If Analog IN channel 0 is selected as the trigger channel for **Fixed-length event**-driven acquisition, data are acquired on channel 1 every time that the presynaptic nerve is stimulated. This way, sweeps of data on Analog IN channel 1 are recorded independent of any specific feature of the data, so that failures as well as evoked signals can be included in the analysis.

Alternatively, variable-length event-driven acquisition may be used if the data to be acquired is in the form of spike trains rather than single responses. However, in this case, the trigger should come from the data rather than the stimulus, so that complete runs of spikes are acquired.

Oocyte Recording

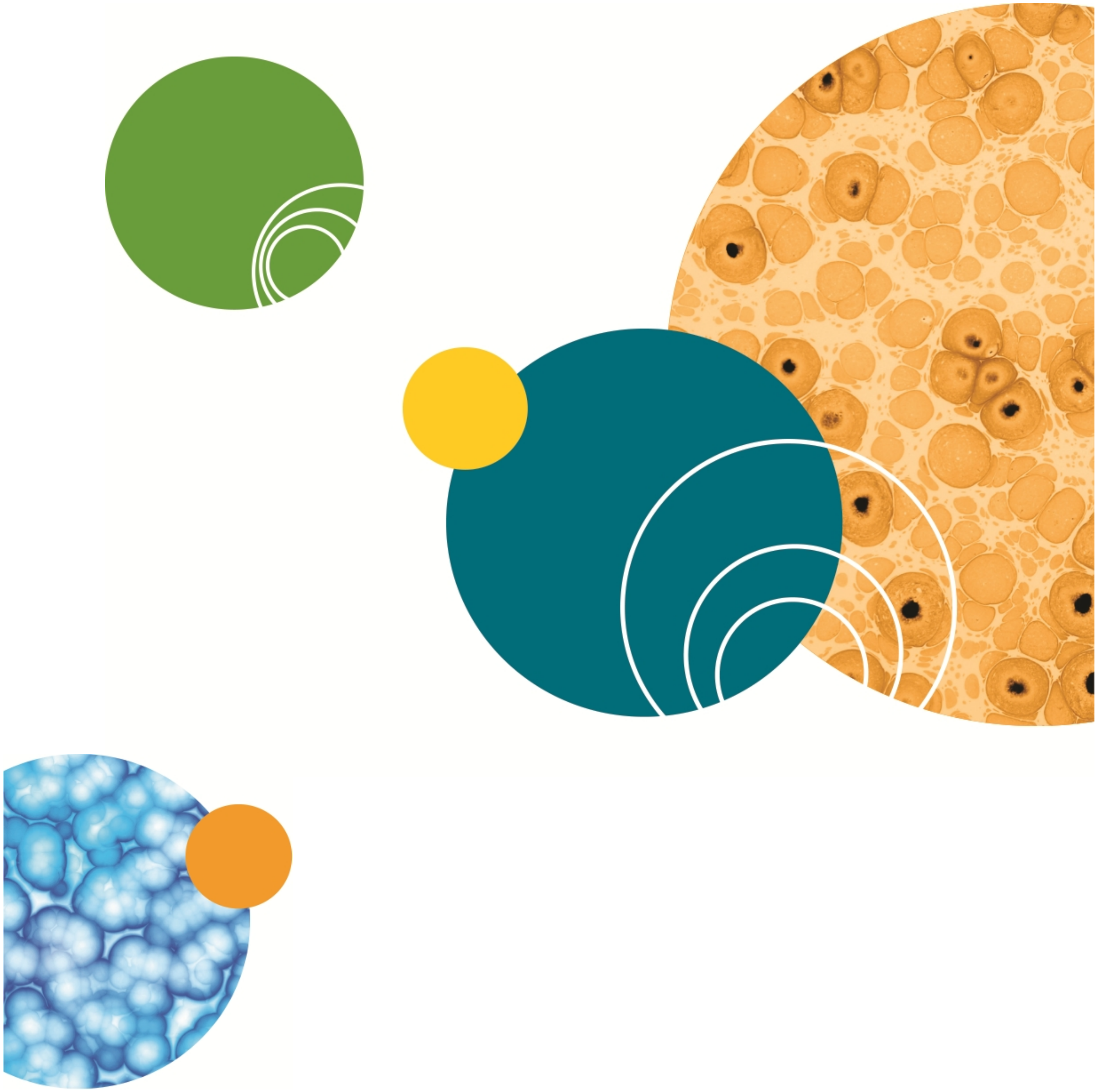
The use of **Real Time Controls** and **gap-free recording** is ideally suited for relatively low-frequency data acquisition, such as recording whole-cell responses from a *Xenopus oocyte*. If you recording the responses to bath-applied drugs, responses typically occur on the time scale of seconds, and so acquisition rates on the order of 10 Hz to 100 Hz are all that is needed. By double-clicking on the **Scope** window, you can call up the **Window Properties** dialog and select “Rolling display” to display the data as a chart record, suitable for acquisition on this time scale.

Bath application of drugs can be controlled in real time using the Digital OUT or Analog OUT signals. If an Analog OUT channel is used to control a valve driver, relatively accurate timing can be achieved by first typing the desired level in the Real Time Controls text field, and then waiting for a specific time mark (for example an elapsed time value) before hitting the <Enter> key. The Analog OUT only changes once the <Enter> key is struck. This way, a complete experiment containing multiple drug applications can be saved as a single file.

If you want to record a complete experiment in a single file but omit long times of inactivity such as when the drug is being washed out, Clampex Software can be toggled between the **Record** and **View Only** modes using the **Pause-View** toolbar button. Acquisition pauses, but the elapsed-time counter keeps counting.

Measuring Behavioral Data

Sometimes data need to be acquired during a behavioral experiment, where the animal may be unattended and inactive for long periods of time. An activity monitor or a photoelectric cell can be used on a trigger channel to control acquisition in the Variable-length event-driven mode, based on specific behavioral or environmental parameters.



Chapter 6: Clampfit Software Features

6

This chapter is an introductory overview of pCLAMP Software, outlining the main features and general organization of the program. Of special interest are the sections on event detection and single-channel analysis, introduced to pCLAMP Software in version 9. Included in the single-channel analysis section are guides for Fetchan and pSTAT for users upgrading to pCLAMP 10.

Clampfit Software Windows

pCLAMP Software has the same standard Windows format as Clampex Software. Wherever possible, pCLAMP Software uses the same commands and organizational principles as Clampex Software.

As in Clampex Software, all commands are included in the main menus, though many of these also have toolbuttons or can be accessed from right-click popup menus. Different menus are available according to the window type that has the focus, for example the **Analysis** window has no **Format** menu, while the **Results** window does, and the entries within menus of the same name similarly differ according to the selected window type.

pCLAMP Software has seven window types:

- Analysis
- Data File Index
- Graph
- Lab Book
- Layout
- Results
- Statistics

These windows can be maximized, minimized, resized and tiled. Bring open but hidden windows into view by selecting them from the list at the bottom of the Window menu. Right-clicking in different regions in each window type brings up popup menus with options specific to the window, including in each case one that opens a **Window Properties** dialog. This allows users to configure the general appearance of the window, and these settings can then be saved as defaults in **View > Window Defaults**.

The basic features of each of the window types are set out in the following, but also see the Help file for greater detail.

Analysis Window

The **Analysis** window displays saved data files. There is no limit to the number of open Analysis windows that can be open at once. The data are displayed graphically, as traces, with subwindows for each signal within the file. A signal can be copied to a new signal within the same file, where it can be modified and then easily compared with the original (**Edit > Create Duplicate Signal**). Similarly, the results of a curve fit can be appended to a file as a separate signal (**Edit > Create Fitted Curve Signal**), as well as the idealized command waveform (**Edit > Create Stimulus Waveform Signal**).

Open files into **Analysis** windows from **File > Open Data**, or configure Clampex Software and pCLAMP Software to automatically open newly recorded files in pCLAMP Software (**Configure > Automatic Analysis**). Configure the way files open into a new Analysis window or into an existent one, replacing the previous file, in **File > Open Data** Options.

Data can be displayed as Sweeps, or in Continuous or Concatenated modes, controlled from **View > Data Display**. In sweep mode you can choose to view any subset of the sweeps (**View > Select Sweeps**), and" toggle between viewing all the sweeps at once, a selected subset, or just one at a time (**View > Toggle Sweep List**). Step through sweeps one at a time with the "<" and ">" keys.

Many analyses can be performed on data in the **Analysis** window, including curve fitting, autocorrelation and cross-correlation, non-stationary fluctuation and V-M analysis. Data can be manipulated in a range of ways: subtracting a control file, averaging traces, adjusting the baseline, or averaging selected portions of the sweeps in a file to create an idealized sweep, to name a few.

Up to eighteen cursors are available to assist in making measurements and defining areas of the file for analysis. Cursors come in pairs, which you can add or remove from the **View > Window Properties > General** tab. Cursor pairs (cursors 1 and 2, 3 and 4, etc.) can be "locked" so they maintain position relative to each other as you move them about the file. Cursor text boxes display (optionally) time, amplitude and sample number, or, in the second of each cursor pair, delta values of these measurements relative to the first of the pair. Configure these and other cursor options by double-clicking on a cursor to open the **Cursor Properties** dialog box.

Since any number of Analysis windows can be open at time, fit data from multiple experiments can be directed to the same sheet in the **Results** window. In this way, fit parameters from control and treatment groups can be automatically tabulated and analyzed.

Selected parts of data files open in an Analysis window can be saved to new files, with the **File > Save As > Options** dialog. For example, you can save just the section of a file between cursors, or just the signals and parts of traces currently visible in the window.

Data File Index

The Data File Index (DFI) is a file management tool that allows you to construct an index of data files. Data files can be grouped together in separate DFI files, and within these sorted according to a wide range of parameters reported for each file. These options give you great flexibility in being able to rapidly find and organize files from particular types of experiments- especially valuable in managing large amounts of data from archival sources such as CDs or DVDs. Create a Data File Index from **File > New Data File Index**.

Graph Window

Graph windows display selected data in two-dimensional graphs. A range of graph types is available, such as scatter and line plots, and histograms. Any one graph can contain multiple plots, in which case a legend identifies these in the right-hand side of the window. Graphs can be named and the axes labeled in the **View > Window Properties** dialog, which contains a range of other configuration options as well. Axes for most graph types can be toggled between linear and logarithmic scaling with **View > Axis Type**.

A range of data manipulation options and analyses are available to apply to **Graph** window plots, including curve fitting, normalization and square root, amongst others. These are accessible from the **Analyze** menu when the **Graph** window is selected, and in the **Graph** window toolbar.

Graphs take their data from a **Results** window, and have no existence independent of their associated *.rlt (Results window) file. Graphs can thus be created from any Results window data, and when a graph is generated from some other source (for example a data file in an **Analysis** window with **Analyze > Quick Graph > Trace vs. Trace**), the data displayed in the graph is also written to an appropriate sheet in the **Results** window (for a Quick Graph, to the Quick Graph sheet). In consequence, to save a graph, you must save the Results window open at the time the graph is generated. If you manipulate data in the **Graph** window, for example, by normalizing it, the corresponding data in the Results window change to the new values resulting from the operation you have carried out in the **Graph** window.

Graphs are generated in pCLAMP Software in a number of ways:

- From an **Analysis** window, the **Analyze > Quick Graph** command has options for generating I-V and Trace vs. Trace graphs. These graphs can be set up to update automatically for data files received from Clampex Software, in **Configure > Automatic Analysis**. They can also be dynamically linked to the file they are generated from so that, under cursor-dependent configurations of the graphs, moving the relevant cursors in the data file automatically alters the graph to reflect the new cursor positions.
- Histograms can be created from **Analysis**, **Graph**, and **Results** windows, with the **Analyze > Histogram** dialog. For data files in **Analysis** windows, these bin and count the amplitudes of the sample points in the file. For **Graph** and **Results** window data, you select the parameters for binning and counting.

- With a **Results** window selected, graphs can be created directly from a sheet in the window by selecting data in a column and using the **Analyze > Create Graph** command. The selected data are graphed against their row number on the X axis. Toolbuttons in the **Results** window allow you to select columns for both Y and X axes, and to add further plots to the same graph. Alternatively, you are given many more configuration options if you use the **Analyze > Assign Plots** dialog.
- Several of the analyses in the **Analyze** menu (for the **Results** and **Analysis** windows) generate a graph as part of their output,for example the Kolmogorov-Smirnov Test, Autocorrelation and Nonstationary Fluctuation Analysis. In each case, the data in the graph is also written to a reserved **Results** sheet.
- With the **Results** window selected, graphs can be created with the **Analyze > Event Analysis > Fast Graph** dialog. This dialog is designed specifically for use with events data, which is usually found on the **Events** sheet. It can, however, be used to create graphs from any **Results** window sheet.
- The **Event Detection > Define Graphs** dialog allows you to create up to four graphs during an event detection session. The graphs are dynamically integrated with the other windows in the session, so that as new events are found they are plotted. You can select an event in all the windows linked within the session by clicking on the point corresponding to it in a **Define Graphs** scatter plot. For more information about this integration, see Event Detection on page 119.

Lab Book

Just as in Clampex Software, the Lab Book window is a text editor for logging events that occur while pCLAMP Software is running. You can set how much you want written to the Lab Book with **Configure > Lab Book Options**, and add comments with **Tools > Comment to Lab Book**, or in a free-form text editor fashion directly in the window. The results of several pCLAMP Software analyses,for example, the Chi-square and Mann-Whitney tests, are recorded in the Lab Book.

There is always a Lab Book window open, called the **System Lab Book**. Copies of this can be saved to disk for editing or archiving elsewhere.

Layout Window

The **Layout** window is a page layout editor for formatting spreadsheets and graphs ready for presentation. Traces and curve fits can be copied into the **Layout** window from the **Analysis** window. Graphs and charts created in the **Results** workbook can also be pasted into the **Layout** window with additional text and draw objects added to create a highquality figure. Files created within the **Layout** window are saved as *.alf files. Finished figures can be printed directly or copied and pasted as an enhanced metafile picture to other programs such as Corel Draw, Origin, or Microsoft Word or PowerPoint.

Results Window

The pCLAMP Software Results window is similar to the Clampex Software Results window but with twenty tabbed data spreadsheets, in contrast to one in Clampex Software. Clampex Software's one cursor measurement sheet corresponds to the first "Cursors" sheet in pCLAMP Software, and a saved Clampex Software results file can be opened directly into pCLAMP Software onto this sheet (when the sheet is then labeled "Clampex Software").

Many analyses, including fitting, are available for data presented in the Results window, and all graphs have their data stored in Results window sheets. The results of many analyses are recorded in the Results window, often on sheets dedicated to results from that particular analysis. New results may either be appended to previous values or else, at the user's option, written over previous data. If data you are trying to append has different columns to that already present on a sheet, then it cannot be appended. In this case you are prompted to see if you want to replace the data currently on the sheet.

There are thirteen sheets that receive results from specific functions:

Table 6-1: Results sheets, and the functions from which they receive results.

Results Sheet	Function	For Window Type
Cursor	Tools > Cursors > Write Cursors & Append Cursors	Analysis
Events	Event Detection	Analysis
Bursts	Analyze > Event Analysis > Burst Analysis	Results
Statistics	Analyze > Statistics & Power Spectrum	Analysis
Basic Stats	Analyze > Basic Statistics	Results
Fit Params	Analyze > Fit	Analysis, Graph and Results
Correlation	Analyze > Autocorrelation and Cross-Correlation	Analysis, Graph and Results
Fluctuation	Analyze > Nonstationary Fluctuation	Analysis
Histogram	Analyze > Histogram	Analysis, Graph and Results
Power	Analyze > Power Spectrum	Analysis
Resistance	Analyze > Resistance	Analysis
V-M Analysis	Analyze > V-M Analysis	Analysis
Quick Graph	Analyze > Quick Graph	Analysis

In addition, there are seven numbered sheets (sheets 14–20) for general use. Data can be moved into the open sheets from the predefined sheets or from other programs using the Windows clipboard (for example by cutting or copying and then pasting in the new location). Data can also be copied into the open sheets from the Analysis window by using the **Edit > Transfer Traces** command. Statistics (*.sta) files opened in pCLAMP Software transfer their data into a numbered Results window sheet as well as opening in a **Statistics** window.

As in Clampex Software, only one **Results** file can be open at any one time, though you can view this from multiple windows, for example, to compare two sheets, with the **Windows > New Window** command. A new **Results** file is opened with **File > New > Results**. You can save a **Results** window as a *.rlt file. This saves any graphs associated with the window at the same time.

Statistics Window

Statistics files created in Clampex Software can be opened in pCLAMP Software from the **File > Open Other > Results & Statistics** dialog. As well as opening into a **Clampex Software-style Statistics** window, the data are transferred to the first empty unreserved **Results** window sheet.

File Import

The Windows environment in which pCLAMP Software operates permits easy transfer of data and figures between pCLAMP Software and other applications in the same environment. Simple cut and paste operations permit you to bring analytical results from other programs into the Results sheet. In fact, any text file can be opened into a selected Results sheet. Open the **File > Open Other > Results** dialog and select the text file you want to open. A dialog opens for you to select the column delimiter in the file and set other options, before the file is displayed. Old pSTAT events lists and histogram files can also be opened with the same dialog. These automatically open into the Events and Histogram sheets, respectively, without any configuration necessary.

pCLAMP Software also imports numerical files for graphical display in the Analysis window. Any simple ASCII or binary output file can be opened into this window. When a non-Axon Instruments text or binary file is selected from **File > Open Data**, a conversions dialog opens. You need to complete this to convert the file into the scaled data that is displayed, specifying the number of signals present in the file and assigning signal names, units and scaling factors. A preview of the source file is provided, allowing you to specify a number of lines to be skipped so that header information is not misread as data. If the data file does not have a time column, you can specify the sampling rate, and pCLAMP Software generates the time values when you import the data.

Data Conditioning

Once raw data have been read into the Analysis window, pCLAMP Software supports several kinds of data conditioning, including filtering and baseline correction. In addition to conventional high and low pass options, the filter functions (**Analyze > Filter**) include notch and electrical interference filters. See Digital Filters on page 155 for a detailed description of pCLAMP Software filtering.

In addition to filtering, files can be baseline-corrected and spurious activity edited out. **Analyze > Adjust > Baseline** applies baseline adjustment as a fixed user-entered offset, or as an offset of the mean of the entire file, or of a particular region of the file. If there has been baseline drift in a sweeps file, for example, subtracting the mean of the first holding period can (depending on the file) bring each sweep to a common zero-value baseline suitable for further analysis.

pCLAMP Software can also correct for linear drift in the baseline by applying a slope correction. Often the most useful form of baseline correction is manual correction. As shown in the figure below, the method for manual baseline correction permits the user to define the baseline in data which requires complex correction functions. When this option is selected, a line is drawn over the data. The user defines the appropriate correction by clicking to create inflection points, and dragging to where the slope of the baseline changes. This form of correction may be ideal for single channel data prior to export (see Figure 6-1).

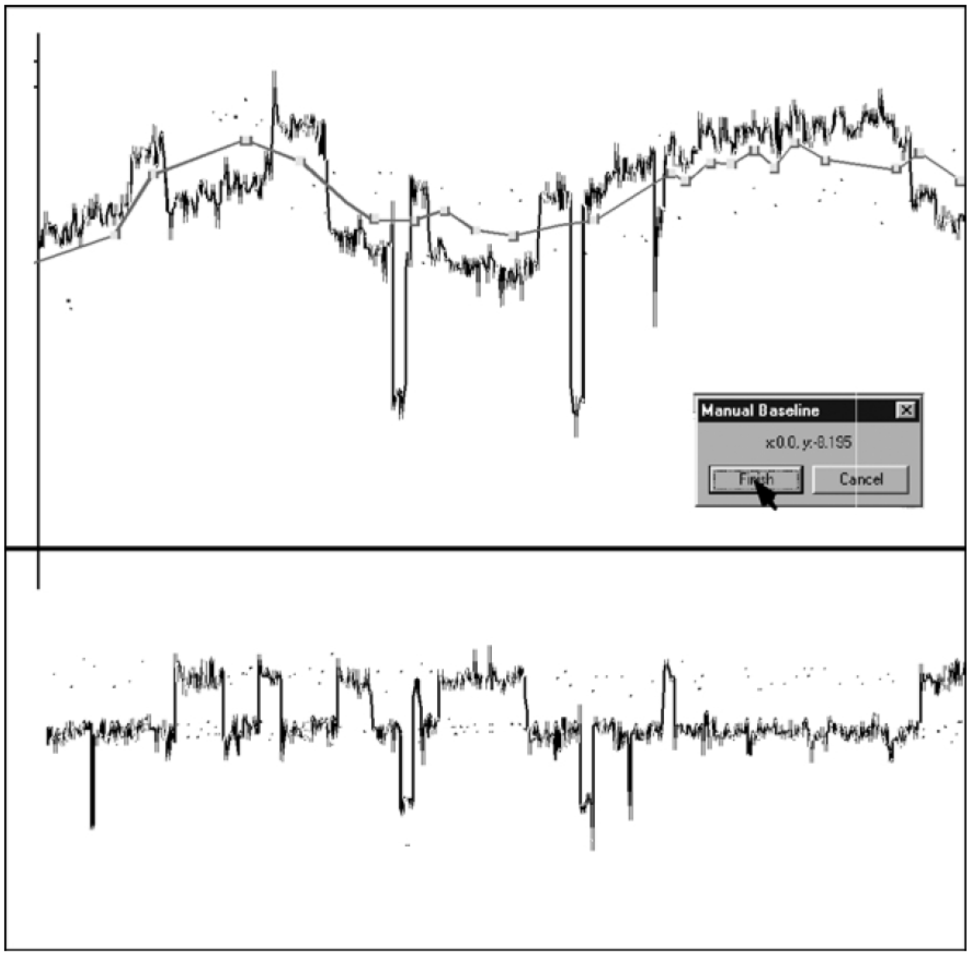


Figure 6-1: Manual baseline adjustment.

Another way of preparing a file for analysis is to remove the passive responses to the stimulation protocol. For this you can subtract a control file (**Analyze > Subtract Control**). If you run a stimulus protocol with lower voltages than those used in the actual test-in order to ensure you get a record of passive response without any cell activity you can then multiply the resultant file by the appropriate factor before it is subtracted.

If you have no independently recorded control file suitable for subtraction, you can create a file of the passive responses from the test file. For a sweeps file where some sweeps contain only passive responses, you can identify these sweeps and use **Analyze > Average Traces** to average them, then save the resultant sweep. If there are no sweeps without any activity, you can select inactive sections from different sweeps and build up an average passive response for the entire sweep: use **Analyze > Segmented Average**.

You can also remove unwanted artifacts from files with Analyze > Force Values, or save specific sections of data files by using appropriate settings in **File > Save As > Options**. Traces in a data file can be normalized prior to further analysis with **Analyze > Normalize Traces**. The dialog allows you to normalize the entire file or sweep, or to rescale the entire trace equally, but where only a selected duration maps to the zero-to-one range.

Any number of unmodified files, provided they are compatible (for example, recorded under the same conditions, with the same acquisition mode, sample rate, number of signals etc.) can be combined into one file with **Analyze > Concatenate Files**. The files are ordered in their order of acquisition within the new file.

Event Detection

Event detection is a dynamically integrated mode of operation that binds the functionality of graphs, the **Results** window and other features specific to event detection, with the **Analysis** window where the file being searched is displayed. One data file at a time can be searched for particular trace features likely to mark a biological "event", such as a synaptic potential or a single ion channel opening. Three main types of event detection search can be carried out (accessed from the **Event Detection** menu).

- **Single-channel search**

  Single-channel records are translated into idealized "events", which are categorized as belonging to user-defined levels corresponding to different ion channel amplitudes in the patch. pCLAMP Software's **Event Detection > Single-Channel Search** replaced "Events List" and "Latency" operation modes in Fetchan-pCLAMP's older single-channel analysis application.
- **Template searches**

  Templates are created by extracting and averaging segments of data that are manually identified as corresponding to an event. The template is then run through the data, identifying further events in the trace.
- **Threshold-based searches**

  Amplitude baseline and threshold levels are set, then the file searched for data that crosses the thresholds.

Event Detection Searches

Searches can be configured with a range of conditions to filter out false hits and to define events to ensure that event statistics-automatically taken for each event found-measure parameters in the desired way. Threshold and template searches can both have up to nine separate categories defined and concurrently searched for when a search is run.

All searches are able to accommodate high degrees of baseline drift provided this does not include abrupt shifts (in which case you are advised to first straighten the file with **Analyze > Adjust > Baseline**). As an additional aid to combat drifting baselines, levels can be manually changed while a search session is in progress. By dragging level markers in the Analysis window, events are immediately found in accord with the new levels, and statistics for these events recorded relative to the new levels. Levels are the only search parameters that can be changed on the fly. To change any other parameter in the search configuration you must stop the search, reconfigure the dialog and then click **OK** to restart the search.

When a set of event search parameters is confirmed (by clicking the **OK** button in the **Search** dialog) the event detection toolbar is inserted into the Analysis window containing the file being searched. Use this (or menu commands, or hot key keystrokes) to control the progress of the search. The first candidate event is highlighted in order that you can identify and assess it, with color-coded lines indicating its category (in a template or threshold search) or level (in a single-channel search). The program waits for you to accept the candidate event as a valid event, as a “suppressed” event (recognizably an event but with features such that you do not want to incorporate its statistics into the dataset), or to reject it entirely. Once you have concluded this, the next candidate event is automatically found and the process repeated. You are able to proceed in this manner for the entire length of the file (or for some defined portion of it) or you can choose to accept all automatically found events, and sit back and watch these accumulate. Once a search is finished you can configure a new search for the current file within the same event detection session, for example by repositioning cursors to search a different part of the file.

As the search proceeds you are able to view various types of events data, as they accumulate, in five other locations: the Event Monitor, Event Viewer, graphs, **Results** window, and **Event Statistics** dialog. Some of these are integrated so that selecting an event in one view brings it into view, highlighted, in other windows as well.

Event Monitor

Event Detection > Event Monitor is a small dialog that reports a few key statistics measurements for the current candidate event, providing a comparison with previously accepted events for that category (for peak-time events) or level (single-channel events). This is useful when assessing candidate events for inclusion into the data.

As each event is accepted, it is recorded, one line per event, in the **Events** sheet in the **Results** window. A range of measurements are always recorded, such as event start and end times, peak amplitude, rise and decay slopes, half-width, etc. for template and threshold events, and amplitudes and dwell times for single-channel events.

Graphs

Similarly integrated in the session, up to four graphs (**Event Detection > Define Graphs**) can be configured once search parameters have been defined and confirmed. Each graph can be configured as a conventional or logarithmic histogram or scatter plot, with all the measurements recorded in the **Results** window available for plotting. As with the **Results** window, these graphs dynamically update as new events are found.

Event Viewer

The Event Viewer (**Event Detection > Event Viewer**) contains a small Analysis window that accepts the trace segments identified as events in your search. These are overlaid one on another as sweeps within the window, and can be saved as an ABF or ATF file. Alone amongst the integrated event detection components, the **Event Viewer** can be kept open between event detection sessions and accumulate events from searches on different files.

Event Statistics

Lastly in the package of integrated data views, a summary of the statistics measurements recorded so far in any session can be displayed with **Event Detection > Event Statistics**. This opens the **Event Statistics** or **Single-Channel Statistics** dialog, depending on the search type, where means, standard deviations and data counts of all comparable events statistics taken in the session are reported.

For all these different “views” of events data, selecting an event in one view selects the same event in the other views. For example, clicking on a point in a scatter plot highlights the corresponding line in the Results window, highlights the event-marking lines on the trace in the **Analysis** window (and scrolls the window to bring it into view, if necessary), shows that event alone in the **Event Viewer**, and shows the values for the event in the **Event Monitor**.

The Help file has detailed discussion of all event detection functions, including an overview topic with links to all related Help topics. There is also a series of “How to” topics in the online Help for event detection. Additional information on single-channel event detection is also contained in the following section.

Single-Channel Analysis in pCLAMP Software

The following two subsections are written primarily as a guide to Fetchan and pSTAT users upgrading to the latest version of pCLAMP, but they should also benefit new users. They provide a general outline of where functionality from the old DOS single-channel analysis programs is found in pCLAMP Software, as opposed to a precise mapping of command paths from one program to the other. It is assumed, for example, that the user is familiar with general navigation and data display commands within pCLAMP Software, so these aspects of Fetchan and pSTAT are ignored.

pCLAMP Software for Fetchan Users

File Subtraction

Use **Analyze > Subtract Control** to subtract a control file from a test file. The dialog allows you to scale the control file before it is subtracted. Subtraction can be set up to be applied automatically as files are imported into pCLAMP Software, with **Configure > Automatic Analysis > Subtract control** file.

Any further baseline correction required must be carried out as a separate operation-use **Analyze > Adjust > Baseline**, where there are a number of options for this.

pCLAMP Software has no direct equivalent to Fetchan’s “Closest N Episodes” subtraction option. Subtracting an averaged trace and then applying suitable baseline adjustment can produce equivalent results. For example, subtracting the mean of a trace segment that has no activity in any of the sweeps (typically, the first holding will fit this criterion) brings a set of sweeps with rising or falling baselines to a common zero baseline.

Data Modification

The **Parameters > General > Modify Data** section of Fetchan has three settings.

- **Filter freq**

  Filtering is performed in pCLAMP Software in **Analyze > Filter**, where you have a full range of filtering options. Unlike Fetchan, you must filter a file as a separate step prior to running event detection.

  Because filtering rounds out level transitions in the trace, it is taken into account in the calculation of event amplitudes and in the definition of “short” events used for automatic level updating. It is important that you ensure a record of all filtering that has been applied to the file is recorded in the data file header.

  Filtering in the amplifier at the time of acquisition, if telegraphed, as well as CyberAmp filtering and any post-acquisition filtering applied in pCLAMP Software, are all automatically recorded in the file header. However, if you record without telegraphing the amplifier’s filter frequency, you should set the Clampex Software Configure > Telegraphed Instrument setting to “Manual”, and then enter the frequency information in the Lab Bench, which then writes it into the file header (see also Single-Channel Event Amplitudes on page 194).

- **Derivative**

  A data trace can be converted to a plot of its differential values with the “diff( )” function in Analyze > Arithmetic.

- **Change polarity**

  Change signal polarity in pCLAMP Software by changing the file scale factor. To do this, open **Edit > Modify Signal Parameters** and reverse the polarity of the scale factor. Note, however, that this can only be done with ABF files that have not already been modified in any way within pCLAMP Software.

Analysis Mode

In Fetchan, different sorts of analysis are enabled by changing the operation mode in the **Parameters > General > Analysis > Analysis** mode field, which then shows the Analysis menu commands relevant to the selected analysis mode. pCLAMP Software does not use this sort of organization, with all analyses being available once you have opened a file into an Analysis window. The following pCLAMP Software functions replace the Fetchan modes.

Episode Average

This Fetchan mode allows you to average all or selected sweeps from an episodic file. Use **Analyze > Average Traces** for this functionality in pCLAMP Software.

To select specific traces to include in the average, move through the sweeps in the file using the “>” and “<” keystrokes, viewing each sweep highlighted against the remainder of the sweeps in the file. Press <Delete> for sweeps you do not want to include in the average (this hides the sweeps-they can be brought back into view with **View > Select Sweeps**). Once you have gone through the file and only have the sweeps you want to average in view, use **Analyze > Average Traces**, making sure that you have “All visible traces” for the trace selection, and “Display resultant trace only” checked.

You can save the resulting trace as an ABF file (**File > Save As**). In the **Save As** dialog, be sure to use the **Data selection** option, “Visible sweeps and signals” so that the newly saved file contains only the averaged sweep. This is because all the original sweeps are still in fact in the file, simply hidden from view. If a file containing these were used as a control file for subtraction, all the hidden sweeps as well as the averaged sweep would be used in the subtraction.

Segmented Average

The functionality in this operation mode can be found in the **Analyze > Segmented Average** dialog in pCLAMP Software. The segmented average trace is built up in pCLAMP Software in a similar way to Fetchan, with the user scrolling through the sweeps in the file in the upper window of the dialog, using cursors to define segments for addition to the accumulating average trace in the lower window.

In pCLAMP Software, the entire sweep is always offered for creation of the segmented average, and the resulting trace saved from the dialog is for the entire sweep.

Pulse Average

pCLAMP Software has no dedicated command for generating pulse averages, however this Fetchan function is easily duplicated. During a threshold event detection search, each event is copied to the Event Viewer, and left aligned to the start of the event. Once the file has been searched, click **Copy As** in the **Viewer** to save (and open) the assembled events as an ABF file. Then average the events with **Analyze > Average Traces**, creating the pulse average trace.

To recreate Fetchan pulse average behavior, set only one level in the threshold search, at the amplitude you would set the threshold if running pulse average in Fetchan.

Events List

This Fetchan operation mode identifies level transitions in gap-free and variable-length files. **Event Detection > Single-Channel Search** replaces it in pCLAMP Software. Resulting data are written to the Results window Events sheet showing similar information to Fetchan EVL files.

In single-channel searches an idealized record is superimposed over the trace being searched, displaying the idealized events at each level. The duration of an idealized event is determined by the time the data crosses into and stays in the amplitude range of a defined level. The average amplitude of all the data points within that duration (less some points at the start and end of the event-see discussion of brief events below) is the amplitude of the idealized event. All the reported event statistics are taken from the idealized trace.

General configuration of single-channel searches-in Fetchan, set in **Parameters > Special > Events List Analysis**-is all carried out within the search configuration dialog in pCLAMP Software. Set the baseline and channel open levels either by entering a numeric value in the dialog, or drag the level markers in the Analysis window. Note that you must set levels as close as possible to the amplitude reached when a channel (or multiple channels) is fully open, rather than at some threshold value part-way to this. The amplitude halfway between each level is in fact the threshold (50% crossing) for categorization of an event in either level. As in Fetchan, the baseline and/or other levels can be set to update automatically to follow an unsteady baseline, and you can also manually adjust levels during a search.

Filtering, which in Fetchan was performed at the same time as the creation of the idealized event trace, must be carried out independently in pCLAMP Software, with **Analyze > Filter**.

For running the search itself, much Fetchan behavior is retained, although some of the commands (in the **Event Detection** menu, or use toolbuttons in the **Event Detection** toolbar, or keystrokes) have different names.

Table 6-2: Fetchan and pCLAMP Software Command Name Mapping

Fetchan	pCLAMP Software
Include	Accept
eXclude	Suppress
iGnore	Reject
Undo	Undo
Nonstop	Nonstop

As each event is found in a search you can view key statistics for it, along with comparisons to previous events, in **Event Detection > Event Monitor**. If you are unhappy with the amplitude or duration automatically given to an event you can adjust this (as in Fetchan) by dragging the relevant idealized event-defining lines in the **Analysis** window.

Brief Events

Fetchan's "short" events are called "brief" in pCLAMP Software, and labelled "B" in the State column in the Events sheet. They can be optionally excluded from analyses you go on to perform on the events data.

pCLAMP Software uses effectively the same algorithm for determining brief events as Fetchan. Since filtering rounds out level transitions in the trace, data points within two time constants of the start and end of an event are not used to calculate the amplitude (where the time constant is proportional to the amount of filtering applied). This means that if an event is shorter than four time constants (for example "brief") its amplitude cannot be calculated in the normal way. In these cases pCLAMP Software reports the amplitude of the midpoint of the event as the amplitude of the entire event (see also Single-Channel Event Amplitudes on page 194).

Latency

Fetchan's Latency mode is similar to the **Events List** mode, but applied to files generated under episodic stimulation where a stimulus was applied at the same time in each sweep. The time between the stimulus and the first response to this (for example the latency) is measured. In pCLAMP Software this is handled in a normal **Event Detection > Single-Channel Search**, configured to ensure latencies are correctly measured:

1. Set cursors in the Analysis window about the sweep region to be searched, ensuring the first cursor is positioned at the time of the start of the stimulus. To do this you can reveal the stimulus waveform with **Edit > Create Stimulus Waveform Signal** and drag the cursor to the onset of the pulse, or use the **Tools > Cursors > Cursor Properties** dialog to position the cursor at the start of the epoch containing the stimulus pulse.
2. In the search configuration dialog, select the cursor pair you positioned above to define the search region.
3. Check the **Latency Analysis** check box in the search configuration dialog.

With the **Latency Analysis** check box checked, event start and end times reported to the Events sheet are measured from the start of the search region rather than from the start of the sweep. Cursor placement, as above, and selection of the cursors for the search region, ensure these values are measured from the time the stimulus was delivered. This allows the program to use the event start time for the first event in each sweep as the latency measurement. These values are used in the analysis of latencies (from the Events sheet) in **Analyze > Event Analysis > Latency**.

In addition to the steps outlined above, you can set an "Ignore initial duration" setting in the search configuration dialog. This is equivalent to **Fetchan's Parameters > Special > First Latency Analysis** "Ignore" setting, whereby level transitions for the stipulated period following the start of the search region are ignored. This is typically used to skip over capacitance transients that immediately follow the stimulus pulse.

All Points Histogram

To create a histogram for the amplitudes of all points in a data file, open the **Analyze > Histogram** dialog when the Analysis window containing the file is selected. All configuration of the histogram is done in this dialog. You set the number of bins here and can restrict the data to be included in it: by trace, by range within the sweep, and by data value.

pCLAMP Software for pSTAT Users

pSTAT takes as its usual input the events list (EVL) files generated in Fetchan. Equivalent data in pCLAMP Software are written to the Results window Events sheet, so this sheet should be selected in order to access analyses and graph options of the sort found in pSTAT.

Histograms

Much pSTAT functionality involves production and analysis of histograms, for example, of dwell times or amplitudes. In pCLAMP Software, you can readily create simple histograms for any events measurements reported on the **Events** sheet with **Analyze > Histogram**:

1. Select the data column you want to display in a histogram.
2. Click the Histogram toolbutton in the **Results** window toolbar.
3. From the dialog choose the histogram type and bin width, and ensure that you have the correct Results window sheet and "All selected columns" in the Data section. You can restrict the data from the selected column so that you include only a specific row and/or data range.
4. Click **OK** to generate the histogram. The bin values are recorded on the Histogram sheet of the **Results** window.

By selecting more than one data column, the dialog allows you to include more than one parameter in a histogram graph (with their values combined or displayed separately, color-coded). You can also normalize the area under the histogram. The histogram function does not differentiate between data according to level or any other parameters, however. To do this, you need to use the **Event Analysis Fast Graph**.

Graphs

The **Analyze > Event Analysis > Fast Graph** dialog is a more powerful graph-generating dialog than Histogram, creating scatter plots as well as histograms. This highly configurable dialog is designed specifically for events data. It includes most of the options in the Histogram dialog, and also allows data selection by trace, search, and level, as well as by the value of a parameter different to the one being plotted.

Brief and/or suppressed events can be excluded from the plots. Where you have elected to plot events from more than one level, the plots are color-coded according to the level represented. **Fast Graph** is the general-purpose graph-generating dialog for all events data.

Fitting

Once a scatter plot or histogram is created, you have the full range of pCLAMP Software's fitting functionality in **Analyze > Fit** to fit a curve to this.

Besides displaying the fitted curve on the graph, fitting results are recorded on the **Results** window **Fit Params** sheet, and in the **Analyze > Fitting Results** dialog.

Frequency Analysis

Instantaneous frequency is automatically measured for all events as one of the basic measurements recorded on the Events sheet, as the events are found in an event detection search. You can use **Fast Graph** to display these data graphically if you wish.

pSTAT's Interval method frequency calculation is replaced by **Analyze > Convert to Frequency**:

1. Create a histogram of the "Event Start Time" values for the data, selecting a suitable bin width.
2. In the resulting histogram click **Convert to Frequency** to convert the count for each bin into a frequency, found by dividing the count by the bin width.

Burst Analysis

Burst Analysis in pCLAMP Software is carried out from **Analyze > Event Analysis > Burst Analysis**. The burst analysis provided in pCLAMP Software differs in a number of respects from that in pSTAT. There are two basic means of finding bursts in pCLAMP Software:

- Set an inter-event interval. Events found to be closer than or equal to this interval are classified as belonging to the same burst. The numerical value of pCLAMP Software's inter-event interval is the same as the inter-burst interval that would be used in pSTAT, the difference being that pCLAMP Software looks for closed-channel times less than the interval in order to cluster events together as belonging to a single burst, while pSTAT looked for closed-channel times greater than the interval, in order to separate bursts.
- Use "Poisson surprise", which measures the degree to which one is "surprised" that a frequency of occurrence deviates from a Poisson distribution.

You can elect to search for bursts of events belonging to a particular level, or for bursts of all levels. These all-level searches can be "merged", so that a sequence of nonzero-level events is treated as one event.

A range of data about the bursts found in an analysis are reported to the Bursts sheet, including the number of events in each burst, the burst duration, the mean intra-burst interval, the event frequency and the mean open time of the events. Histograms of these properties can be generated in the usual way.See also Burst Analysis on page 201.

P(open) Analysis

Analysis of the probability that a channel is open is performed with **Analyze > Event Analysis > P(open)**. This analysis reports the number of events included in the calculation, the total time range, and, on the basis of the way in which you configure the dialog, the probability of a single channel being open and the probability of any channel being open.

The dialog also gives the option of generating a P(open) histogram similar to pSTAT's. As in pSTAT, you can manually enter an interval setting for this, or have it done automatically by pCLAMP Software. See also P(Open) on page 203.

Latency Analysis

Latency analysis (**Analyze > Event Analysis > Latency**) is the measurement of the times from the application of a regular stimulus in each sweep until the start of the first event in the sweep. Optionally, you can create a histogram of the latencies from this dialog.

Events data need to have been collected under the right conditions in order that this analysis can be carried out. The latency values reported are simply the event start time (from the Events sheet) for the first nonzero-level event in each sweep. To ensure that this value is a measure of the time from the onset of the stimulus, you need to have configured the single-channel search (**Event Detection > Single-Channel Search**) appropriately. This involves checking **Latency Analysis** in that dialog, and setting the event detection search region to start at exactly the time of the stimulus.

See brief instructions above (pCLAMP Software for Fetchan Users on page 121) or more detailed instructions in the Help file.

Peri-event Analysis

This analysis is new in pCLAMP Software, with no equivalent in pSTAT. It measures the intervals between selected events and events that occurred within a stipulated time range of these. Events are selected as the central events for this analysis by tagging them. This can be done within a search session with **Event Detection > Accept and Tag or Event Detection > Process Selection**, or by typing "T" into the State column in the lines of the selected events on the Events sheet. See also Peri-Event Analysis on page 203.

Idealized Trace

pCLAMP Software displays an idealized trace showing dwell times and average amplitudes for each event, color-coded for the event level, when events are being found in a single-channel search. Once the event detection session is closed, the idealized trace cannot be re-created.

Curve Fitting for Single-Channel Results

pCLAMP Software provides the Gaussian function for fitting to amplitude data and exponential probability functions for fitting to either conventionally binned dwell-time data or dwell-time data that have been transformed by log interval binning.

Fitting to dwell-time data supports either least-squares minimization or log likelihood maximization. However, it is not recommended that conventionally binned data be fitted using maximum likelihood estimation, which is primarily recommended for fitting to log-binned data. pCLAMP Software provides an efficient and accurate algorithm for maximum likelihood fitting to log-binned dwell-time distributions, using the EM algorithm for initial parameter estimates and the variable metric fitting method to fine-tune the fit.

For example,Figure 6-2 shows variable metric fits of 2, 3 or 4 exponential terms to log-transformed open dwell-time. The fit improves with the higher order functions.

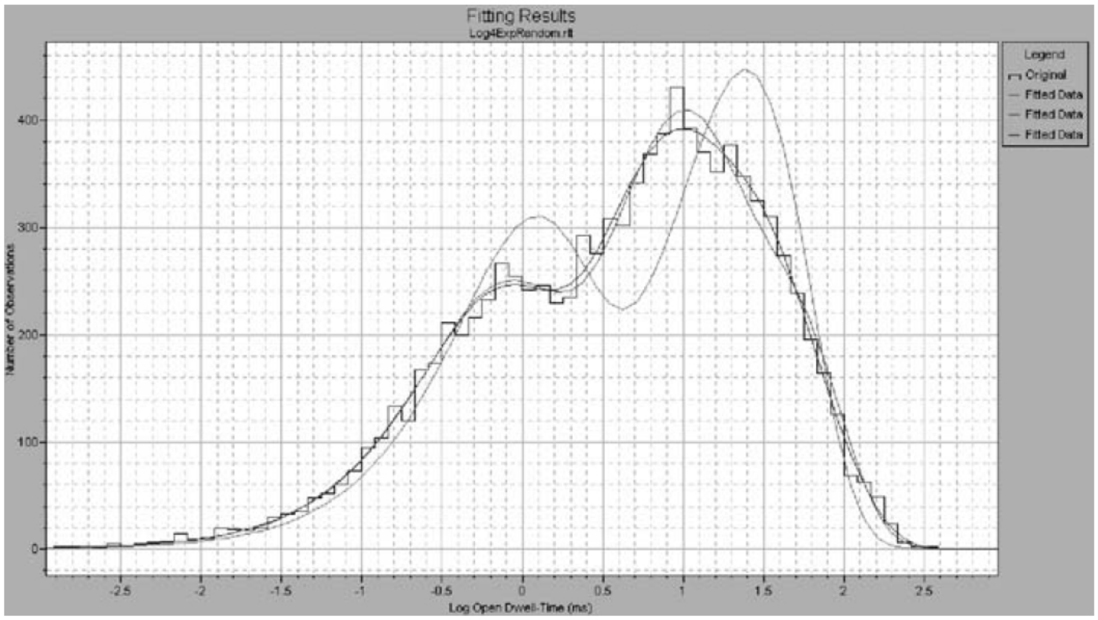
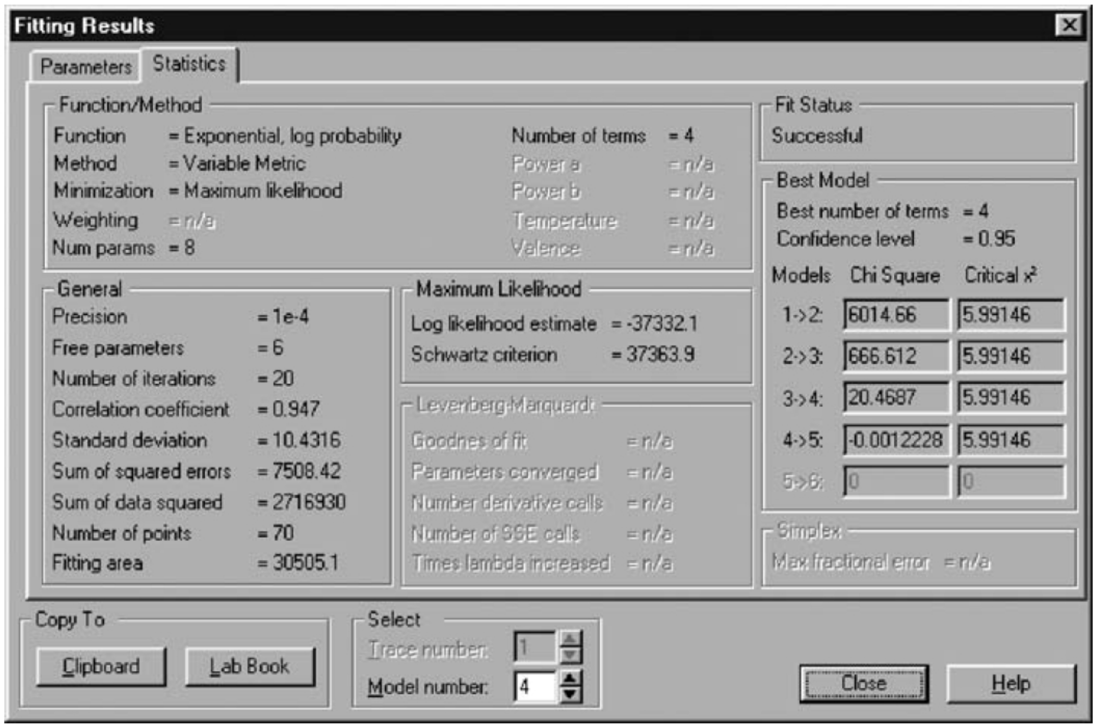


Figure 6-2: Variable metric fits with 2, 3 and 4 exponential terms.

Square root transforms of the binned data can be calculated using **Analyze > Square Root** column arithmetic in the results sheet. However, pCLAMP Software does not provide options for excluding zero-count histogram bins from the fit nor does it correct for binning and sampling promotion errors of conventionally-binned data, as did pSTAT. Even so, you might find that pCLAMP Software provides a more flexible environment for fitting, display and presentation of dwell-time and amplitude histograms, fitting residuals and component curves.

pCLAMP Software also provides an option for statistically determining the best fitting model (number of terms) for dwell-time or amplitude distributions; check **Compare Models** in the **Analyze > Fit > Data/Options** tab. Fitting models are iterated automatically. You can also specify confidence intervals for standard and maximum likelihood comparison statistics. The respective F-statistic or Chi-square table values are computed automatically and displayed along with the test statistics so you can evaluate the results without referring to statistical tables. This information is provided following the fit in the **Analyze > Fitting Results > Statistics** tab.

For example, Figure 6-3 shows the results of an automatic model comparison for the log-binned data shown in Figure 6-2.


Fitting Results

Parameters | Statistics

Function/Method

Function	= Exponential, log probability	Number of terms	= 4
Method	= Variable Metric	Power a	= n/a
Minimization	= Maximum likelihood	Power b	= n/a
Weighting	= n/a	Temperature	= n/a
Num params	= 8	Valence	= n/a

Fit Status

Successful

Best Model

Best number of terms	= 4
Confidence level	= 0.95

Models	Chi Square	Critical χ^2
1->2:	6014.66	5.99146
2->3:	666.612	5.99146
3->4:	20.4687	5.99146
4->5:	-0.0012228	5.99146
5->6:	0	0

General

Precision	= 1e-4
Free parameters	= 6
Number of iterations	= 20
Correlation coefficient	= 0.947
Standard deviation	= 10.4316
Sum of squared errors	= 7508.42
Sum of data squared	= 2716930
Number of points	= 70
Fitting area	= 30505.1

Maximum Likelihood

Log likelihood estimate	= -37332.1
Schwartz criterion	= 37363.9

Levenberg-Marquardt

Goodness of fit	= n/a
Parameters converged	= n/a
Number derivative calls	= n/a
Number of SSE calls	= n/a
Times lambda increased	= n/a

Simplex

Max fractional error = n/a

Copy To: Clipboard | Lab Book

Select: Trace number: 1 | Model number: 4

Close | Help

Figure 6-3: Fitting Results dialog Statistics tab.

The model comparison statistics are shown to the right in this information dialog. Since the difference in the Chi-square value for models 4 and 5 (that is, 4 and 5 terms) is not significant, the 4-term model is accepted as the best fit.

QuB

pCLAMP Software provides an option to export files in *.ldt format. Single-channel data exported in this format can be analyzed with QuB, a specialized suite of applications available from the University of Buffalo. These programs provide an alternative to pCLAMP Software for creating idealized records by applying maximum likelihood interval analysis to find the most likely rate constants for specific models. Information about these programs and downloads can be found on the QuB web site. These programs represent a highly specialized approach to single channel data, referred to as hidden-Markov modeling, and are neither directly supported nor endorsed by Axon Instruments. Therefore, pCLAMP Software users are advised to read the information and the references provided on the QuB web site before using the QuB programs to analyze their data. It should also be noted that the QuB methods are best suited for data that have not been heavily filtered.

Unlike the idealization of single channel records performed by Fetchan, the idealization procedure in the QuB suite does not permit the identification and editing of individual events. Therefore, the data must be carefully preprocessed in order to assure that they are suitable for idealization. This can be done either in pCLAMP Software or in the preprocessing module of QuB.

Preprocessing in pCLAMP Software is easy. First, data to be analyzed by QuB should be baseline-adjusted before export, best accomplished using the manual adjustment method described above. If a record has segments of noise from the breakdown of patch resistance or other spurious signals, these portions of the record can be removed before export by using the “blank” command. The recording is then exported as multiple segments, and the QuB programs can be applied to fit a model that would describe the activity within segments, information about time between segments will be lost in the QuB analysis. If a recording clearly contains multiple kinds of channel activity, distinguishable for example, by amplitude), the “blank” command can also be used to create a sequence of homogeneous segments, suitable for fitting to a single model.

Along with the hidden Markov model analysis, the SKM and MIL modules of the QuB suite do generate amplitude and event duration histogram files, respectively, which can be opened directly as Results sheets in pCLAMP Software. Once the Xaxis data in the QuB duration histograms is converted to Log values, using **Analyze > Column Arithmetic** in pCLAMP Software, the SKM-idealized data can be fit with conventional methods, as described above. This approach permits the direct comparison of QuB and pCLAMP Software idealizations.

Fitting and Statistical Analysis

pCLAMP Software boasts a wide range of predefined fitting functions for data displayed in Analysis, Graph and Results windows, or you can define customized functions. The **Analyze > Fit** dialog allows you to select the fitting method and configure a range of parameters relevant to the fitting function and method you select. While fitting in the Analysis and Graph windows allows the fit to be displayed along with the data, fitting in these windows is only suitable for simple X-Y pairs of data. Data in a Results window, on the other hand, can be fit with complex functions such as the Goldman-Hodgkin-Katz equation by assigning multiple variables to specific columns.

Fitting in pCLAMP Software is dealt with thoroughly in Curve Fitting on page 205 and Fitting Functions on page 237.

A powerful range of parametric and non-parametric statistical analyses are available to apply to data in the Analysis and Results windows. For analyses carried out on data files in the Analysis window you can identify specific epoch regions to which to apply the analysis from within the analysis configuration dialog. Alternatively, if you don’t want to analyze the whole file, set a cursor pair about the region you want to analyze before you open the configuration dialog, then select this region after you have opened the dialog. To select Results window data for inclusion in an analysis use the Select Columns dialog opened from the analysis dialog, or the **Analyze > Extract Data Subset** dialog for datasets based on conditional settings.

Other analyses, all located in the **Analyze** menu, are:

- **Burst Analysis**: finds closely packed groups of events in single-channel and peak-time events files and reports a range of statistics for these.
- **Latency**: measures the time between the onset of a stimulus and the response to this, in episodic files (for single-channel and peak-time events).
- **Peri-event Analysis**: measures the intervals between selected events and events that occurred within a stipulated time range of these (single-channel and peak-time).
- **P(open)**: analyzes single-channel data to calculate the probability that an ion channel is open.
- **Kolmogorov-Smirnov Test**: a non-parametric method used to assess the probability that data from two distributions belong to the same population.
- **Nonstationary Fluctuation**: computes the mean and variance for all the sweeps at each sample point within a selected area of trace activity and subtracts mean and variance values derived from a baseline section of the trace.
- **V-M Analysis**: generates a plot of variance against mean for data obtained under different experimental conditions to evaluate transmitter release.
- **Autocorrelation and Cross-correlation**: available for Results, **Analysis** and **Graph** windows data, these are time series analyses that look for cycles within the selected dataset, either by comparing time-shifted data with itself, or to data from a different file.

Creating Figures in the Layout Window

Data reduction and presentation often involve many steps, from simple fits of the raw data to extrapolation of complex functions based on those fits. pCLAMP Software is configured with a wide range of predefined fit functions to get both the raw data and final analyses into the same figure. For example, concentration-response data can be fit to the Hill equation, or reversal potential data from ion substitution experiments can be fit to the extended Goldman-Hodgkin-Katz equation. The Layout window can then be used to format a presentation of both raw data and plots of the final analysis.

The Layout window provides a place where the many elements of pCLAMP Software come together for final preparation and presentations. Graphs, raw data traces, and even tabulated data can all be copied directly into this window. To enter statistical results from the Lab book, first copy and paste them into the Results window, and from there, copy them into the Layout window.

While Layout window files can be saved, they can only be reopened within pCLAMP Software. Therefore, to transfer figures to another program, you need to use the standard Windows Copy and Paste functions. Using the option of Paste Special - Enhanced Metafile provides the best results. Otherwise, first copy and paste the figures into a Microsoft application such as Word, and from there, transfer them to the target application.

Chapter 7: Clampfit Software Tutorials

7

The scenarios presented in this chapter introduce Clampfit Software functionality by implementing procedures that are frequently used in the evaluation of physiological data. Occasionally the tutorials do not take a straightforward approach, in order to introduce you to a wider variety of features than would otherwise be the case. Once you master these features you may wish to perform the analyses in a more direct manner.

The scenarios highlight some of the features discussed in the last chapter. We will give menu commands for each step in the analysis, but you should also become familiar with toolbuttons and hotkeys. Hold the cursor over a toolbutton to read its tooltip, and hotkey combinations are reported beside menu commands. Also try the context-sensitive commands in the popup menu that a right mouse click opens. Context-sensitive online help is available everywhere in the program by hitting the <F1> key or through the Help buttons in the dialogs.

The sample files are found in the **...\Program Files\Molecular Devices\pCLAMP10.6\Sample Data** folder. When following the tutorials, you might want to save the changes from time to time. This is not explicitly mentioned, and is not really necessary, since you will find that most of the steps are reproduced quickly. If you want to save the analyzed files nevertheless, we recommend not replacing the original sample files. pCLAMP handles two different types of binary files. Integer ABF files are assumed to contain original data and cannot be overwritten from within the programs. Analyzed files are by default saved in floating point ABF format, and can be overwritten. Overwrite only ABF floating point files.

pCLAMP Software can be used to manipulate data in illegitimate as well as legitimate ways. Remember that it is your responsibility as a scientist to consider carefully the changes you introduce into the data and to maintain the integrity of the results.

Creating Quick Graphs

pCLAMP Software can generate current-voltage plots with just a few mouse clicks. Furthermore, in cooperation with Clampex Software, you can create I-V plots automatically, in real time, immediately after the application of a step protocol. This is explained in Real-Time I-V Plotting in pCLAMP Software on page 100.

A Quick I-V from Whole-Cell Currents

Use **File > Open Data** to open the sample file *cav1.dat*. It is a whole-cell patch clamp recording showing outward currents in response to a voltage step protocol. To see the protocol that was applied, select **View > Stimulus Waveform Display**, or click the toolbutton in the main toolbar. From a holding potential of -50 mV, depolarizing voltage steps were applied using the Analog Output Channel AO #0. We will plot the peaks of the elicited outward currents versus the step voltage:

1. Set cursors 1 and 2 to "32 ms" and "250 ms", respectively. You can do this by dragging the cursors to the new positions, or by double-clicking the cursor number, selecting

Time and entering the X axis value in the **Time** field, then clicking **OK**.

2. Select **Analyze > Quick Graph > I-V**, or click the toolbutton in the **Analysis** window toolbar.
3. There are two ways to assign the step voltage to the **X Axis (Voltage)**: specify it by the positions of cursors 1 or 3, or define it using epochs. In the example, place all four cursors in "Epoch B", so each of the three Waveform options in the X Axis group leads to the same result.
4. On the **Y Axis (Current)**, plot the **Peak** outward currents elicited during the voltage step. There is only one **Signal > IN 0 (A)**, which we can specify. To avoid the capacitive transient at the onset of the voltage step, restrict the search for a **Peak > Polarity > Positive** to the **Region > Cursors 1..2**.
5. pCLAMP Software offers the opportunity to smooth the signal while **Peak** is selected, minimizing the influence of high-frequency noise, or individual channel openings in relatively small cells. Up to 21 **Smoothing Points** are allowed, but you will have to determine the appropriate choice depending on the signal and sampling interval. To do so, either compare I-V curves with and without smoothing enabled, or smooth the signal using **Analyze > Filter**. In the third tutorial we will learn how to compare a filtered signal to the original.
6. For our comparatively slow currents, set **Smoothing Points** to "11" to provide a sufficiently smooth signal with little danger of over filtering.
7. Click **OK**.

pCLAMP Software creates a new **Graph** Window with the I-V plot, which can be customized in many respects after double-clicking to obtain the **Properties** dialog, or by using the items in the **View** menu.

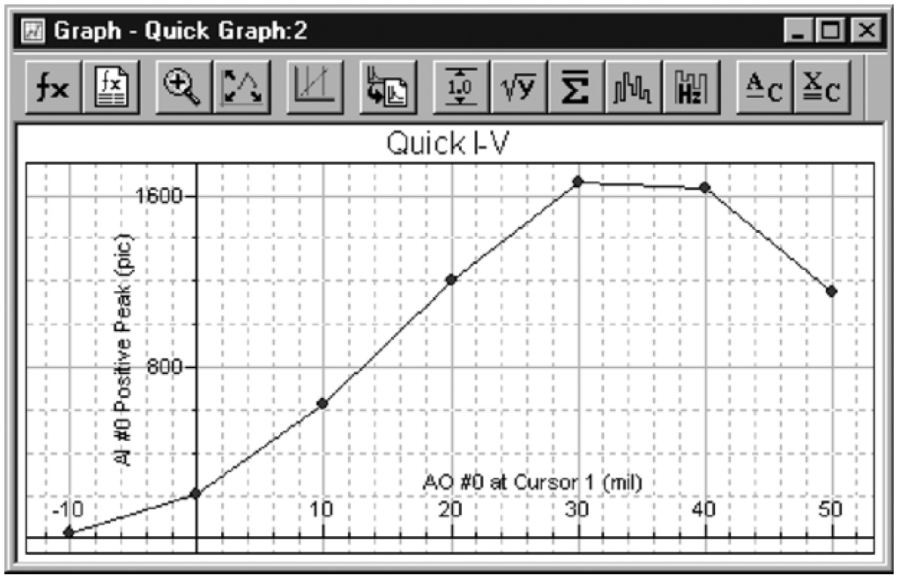


Figure 7-1: Quick I-V graph for cav.dat.

Quick Trace vs. Trace Plots from Whole-Cell Currents

The application of voltage ramps frequently contributes to the characterization of ligandgated currents (Okada et al. 1994, Tang & Papazian 1997). The sample file *inrctc02.dat* is an example for an inwardly rectifying current in response to a voltage ramp.

Select **Analyze > Quick Graph > Trace vs. Trace**. This function allows plotting different elements of a data file versus each other sample by sample. The dialog provides two identical sections for the definition of the **X Axis** and **Y Axis** values, respectively. The available **Trace** , **Waveforms**, and **Signal** options are selectable from drop-down lists. Additionally, as in Quick I-Vs, you can **Invert** the values to account for data from cell-attached or inside-out recordings.

We want to plot the only available sweep of the input signal AI #15 versus the output signal AO #0, so the default settings apply-Sweep #1 of waveform AO #0 (mV) for the **X Axis**, against "Sweep #1" for the **Signal** AI #15 for the **Y Axis**. While in *cav1.dat* we had to avoid the capacitive transients at the beginning of the voltage steps, we now can perform the analysis on the entire ramp epoch. Therefore, specify **Region > Epoch C - Waveform 0**.

pCLAMP Software creates a plot that, in fact, looks similar to the original data file, but is calibrated in mV and pA, respectively, and does not include the holding sections and artifacts prior to or following the actual ramp waveform.

To compare with the current elicited under control conditions, open the file *inrctc01.dat* and repeat the previous steps, making sure that **Append** is selected in the **Destination Option** group.

Preparing Data for Presentation

We are next going to create a Layout window containing the original data files, the voltage step protocol, and the I-V curves:

1. Select the **Analysis** window that contains *inrctc02.abf*.
2. **Edit > Copy to Layout Options** allows you to choose between copying analyzed data files either in the fashion they are displayed in the **Analysis** window, or using the **Page Setup** settings. In addition, you can specify three parameters to be automatically included in the comment line: the file name, the acquisition date and time and the comment.
3. Select **Page Setup** settings and uncheck the three parameters for the comment.
4. Select **Edit > Copy to Layout Window**. pCLAMP Software opens a **Layout Window** and prompts where to place the graphic within a virtual grid on the canvas, and which comment to add.
5. To illustrate the voltage ramp that was applied, activate one of the Analysis windows, select **View > Stimulus Waveform Display**, scale and then click the **Clipboard** button. Select **View > Layout Window** and paste the waveform.

6. Return to the Graph window we created before. The **Trace vs. Trace** graph can be customized in the **Properties** dialog that is brought up by double-clicking, or through the right-click menu or if you select **View > Window Properties**. For the sample Layout above, starting from the factory default window properties, all fonts and the axis titles were altered, and the grid lines, the frame box and the legend title were removed. When the graph suits your needs, copy it to the **Layout Window**. You now have a figure illustrating the activation of an inwardly rectifying current by drug application.

Preconditioning Noisy Single-Channel Recordings

In electrophysiology, the best strategy to deal with noise is to avoid recording it. However, you will often have important data that cannot be analyzed without preconditioning. To learn more about pCLAMP Software's functionality during this tutorial, we will introduce artificial noise into a data file. Then we will remove the noise, while maintaining as much of the information in the file as possible.

The sample file *kchann.dat* is a patch clamp recording of a single active channel. It is the kind of recording often presented as "typical data" in publications. That means, hardly ever any recording is as clean as this one, so we are going to make it a little more realistic. We will work with a section of the file only:

1. Select **View > Data Display > Sweeps** and enter "20000" in the Sweep length field of the dialog box.
2. Now select the first sweep using the **View > Select Sweeps** command, or right-click in the data area for a popup menu option.
3. We use **Edit > Transfer Traces** to convert the binary data file into ASCII data in the **Results** window. **Edit > Transfer Traces** gives you the option to transfer the data from the **Analysis** window to the **Results window** or to a **Graph window**. Make sure **Results window** is selected.
4. The maximum number of samples pCLAMP Software can transfer is 1,000,000. The sweeps are 20,000 samples long, so we can select **Full trace** in the **Region to transfer** field.
5. We want to transfer the first sweep only, so the **Trace Selection** group should report **All visible traces**.
6. Click **OK** and look at the **Results** window to find the first portion of the data on Sheet 14.

Now we can start creating the artificial noise. We first generate line frequency hum:

1. Select column C by clicking on the column header and then go to **Analyze > Create Data**, or alternatively press the toolbutton in the **Results** window toolbar.
2. The **Create Data** dialog allows you to define the **Fill Range** in the **From** and **To** fields for columns and rows. The default setting is the current selection on the **Results** sheet, which is rows 1 to 20,000 of column C in our example. We can keep this default here.
3. The **Function** options group offers a set of predefined functions plus the **Expression** field for custom formulas.

4. In the **Data Limits** fields the values of variables used to generate the various predefined series can be specified. Click on different **Functions** in the Series Option list to see which variables can be defined.
5. If the **Fill Range** comprises more than one column and row, **Fill Direction** determines the way in which pCLAMP Software fills the cells. Your selection is represented graphically in the dialog.
6. Since we are going to create a sine wave, we use a custom expression function, which must be specified in terms of a function with the independent variable x. Let's use a line frequency of 50 Hz, corresponding to a 20 ms wavelength. Therefore, we will use a variation of the function:

   $y = \sin(2\pi \times \text{time} / \text{wavelength}) =$

   $\sin(2\pi \times \text{time} / 20\text{ ms}) =$

   $\sin(2\pi \times \text{time} / 10\text{ ms}) =$

   The constant π is accessed by typing in "pi". The amplitude of the sine wave will be about 1/4 the amplitude of the signals, which is almost 16 pA. So the expression we enter is:

   4 * sin(pi * x/10)
7. On the **Results** sheet, we find the time, in sampling interval steps, in the first column. Since this is the independent variable, we enter 0.1 into the X Increment field. The X Start Value can be any number in this example, since it only introduces a phase shift in a periodic function like a sine.
8. Click **OK**.
9. We additionally create harmonics in columns D-F, reducing the wavelength (10) in the term by integer division and the amplitude (4) in integer steps. To keep track which signal you created in which column, rename them by double-clicking on the header of column C, always clicking **Next** after you have typed in a convenient name.
10. Now that we have the original signal in column B, and line frequency hum in columns C through F, we are going to add the components together. Select **Analyze > Column Arithmetic**.
11. In the **Column Arithmetic** dialog there are several controls that assist in entering an **Expression**. The **Columns** button allows you to specify a list of columns using the "cA" notation explained in the dialog, or to select from the list. The **Operator** button opens a drop-down list of basic mathematical operators, while the **Function** button lists the functions supported by pCLAMP Software. See the Help file for a more detailed description. The items in the **Special** drop-down list yield the row number, the maximum number of rows in the columns the expression includes, and the constant π. **Undo** reverts the most recent change you did to the expression. The **Specify Region** group allows restricting the row range the expression is applied to. When only a subset of the rows is to be analyzed, if the **Force excluded region to zero** option is checked, else it remains unchanged.
12. We want to add all but the Time column and put the results out in an empty column. So the expression you should apply is:

    cG = cB + cC + cD + cE + cF

13. Both **OK** and **Apply** execute the expression, but after **Apply** the dialog remains open. Now the data in column G should be a patch clamp recording with a lot of line frequency hum. We are going to save this composite signal, open it as data and remove the noise.
14. Select **File > Save As**, set the file type to **Axon Text Files (*.atf)** and save with a name like "kc_noise". Then select **File > New > Results** to close the ATF file in the **Results** window.
15. Open the **Results** we just saved into an **Analysis** window with **File > Open Data** and again set the file type to *.atf.
16. pCLAMP Software by default assumes the multi-column ATF file to be a sweep type file and displays each column as one sweep. Select sweeps 1 and 6 only using **View > Select Sweeps** and go to **View > Distribute Traces**.
17. **Distribute Traces** allows you to offset the display of overlaid traces for a user-defined distance. The offset is only added to the display, not to the data. Enter "30" in the **Distance** field.
18. Select **Autoscale All Y Axes** from the right mouse popup menu.

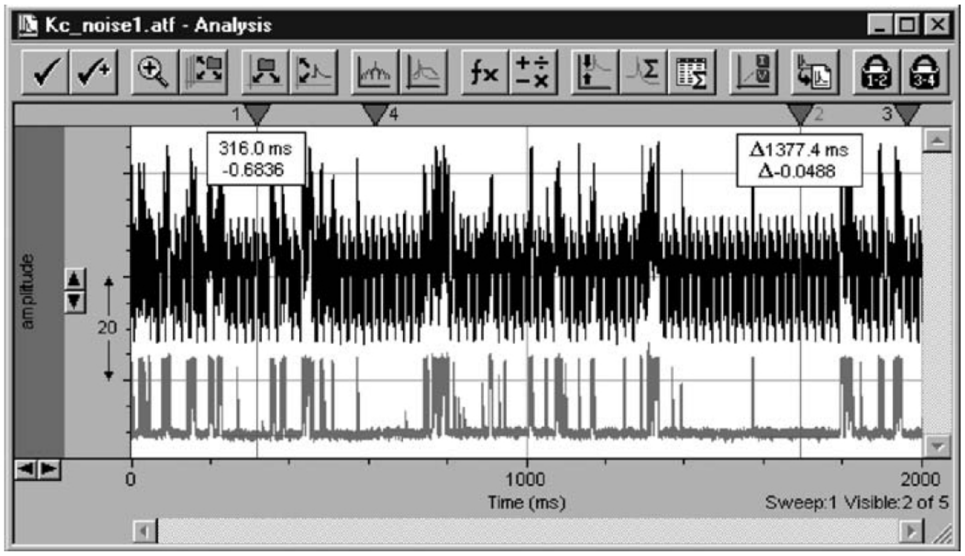


Figure 7-2: Analysis window after Distribute Traces.

19. The original recording, displayed in the lower trace, is now heavily contaminated with noise, as can be seen in the upper trace. Since in real life you would not have created the noise yourself, it is useful to perform a power spectrum before filtering:
    - Select **Sweep #6** only, **Autoscale** the **Y Axis** once more and go to **Analyze > Power Spectrum**.
    - A number of settings can be adjusted in the **Power Spectrum** dialog, which are explained in detail in Clampfit Software Analysis on page 185. For our purpose a **Window Type > Rectangular** together with the **Length > Maximum** will do. There is no need to Exclude spectral bins, and the analysis will apply to the full length of the **Active trace**. pCLAMP Software creates a Graph window with the power spectrum in a Log-Log plot.

- Right-click on one of the axes or go to **View > Axis Type** and select **Linear-Linear**. Zoom the first few hundred Hz by dragging the mouse cursor along the X axis.
- As expected, the spectrum exhibits four sharp peaks at 50 Hz and three harmonics.

20. Return to the **Analysis** window and select **Analyze > Filter**. The **Electrical Interference** section allows us to adjust several parameters. We know that there is no more than the fourth harmonic in our signal. pCLAMP Software determines which harmonics are present and only filters existing ones. So it is mainly a matter of execution speed to restrict the search for harmonics to a realistic number.
    - The **Cycles to average** setting determines the bandwidth of the filter. Since this parameter is limited by the number of samples in the filtered region, set it to 100.
    - pCLAMP Software can automatically determine the line frequency in the signal, but since it is usually known, you should set this parameter manually. After clicking **OK**, the filter has completely removed the hum, as you can convince yourself by performing another **Power Spectrum** and appending it to the existing plot.
    - The new power spectrum can be more easily compared if you increase the line thickness of the plots, via the **Graph Window Properties**. In addition, you can compare sweeps 1 and 6 once more.

Not only line frequency hum can contaminate recordings. Other influences such as mechanical vibration may produce broad bands instead of sharp peaks in the power spectrum. Now that you have learned how to produce and to remove a certain type of noise, you might try to introduce wide-band noise and remove it using the Notch Filter. By such practice you will learn a good deal about the benefits and limitations of different filter algorithms. The Help file and Digital Filters on page 155 provide valuable additional information.

If you have episodic data most of the steps we performed to introduce artificial noise can equally be done using **Analyze > Arithmetic** from the **Analysi**s window. This feature, which requires even less computational effort because no conversion of binary to ASCII data is necessary, is described in the next tutorial. We took a more complicated way to learn about pCLAMP Software's **Results** window, some of the features it offers, and the input and output of ASCII data. Furthermore, we learned how to use power spectra and digital filtering to improve the signal-to-noise ratio in our recordings.

Evaluation of Multicomponent Signals: Sensillar Potentials with Superimposed Action Potentials

In this tutorial we will learn how to use pCLAMP Software for separating the fast and slow components of a signal. This situation occurs especially in extracellular recordings, where contributions of different excitable cells, or different compartments of one cell coincide, forming a multi-component signal.

Open the sample file senspot.dat. Using software filters, we will separate the sensillar, or receptor, potential-the relatively slow downward deflection of the signal-from the much faster action potentials superimposed on it (Thurm 1972, Kaissling & Thorson 1980). Digital filtering requires a single, continuous sampling interval (see Digital Filters on page 155), which is not the case here with this older data file, as you can see if you select **File > Properties**. So we have to convert the two sampling intervals of 200 and 600 µs to a single interval using either **Analyze > Data Reduction**, or **Analyze > Interpolation**. For comparison we will do both.

Data reduction can be used to eliminate samples from a file acquired at an excess sampling rate, or which has to be reduced in size for other reasons (graphing, minimizing storage requirements, etc.). There are two parameters to set. The reduction factor *n* is the factor by which the number of samples in the file will be reduced.

The selection of the reduction method determines how pCLAMP Software eliminates the excess samples.

- **Decimate** retains only every nth data point, while the samples in between are deleted. This method minimizes the computational effort, but introduces the danger of aliasing, since the resulting data file is equivalent to one acquired at a lower sampling rate. That means no signal containing higher frequencies than half the sampling frequency in the reduced file may be processed using Decimate reduction.
- **Substitute average**, in contrast, does not introduce any artifacts. Here every n samples are averaged to yield the new data value. This method can eliminate transients and high frequency noise, and can even smooth the signal, since the averaged values are not restricted to the resolution of the A/D converter. The expense for these advantages is the slower performance, which should be relevant only on very slow computers, or with very large data files. Therefore, whenever possible, Substitute average should be the method of choice.
- **Min/Max** determines the largest and the smallest values out of every set of n data points and retains them. Therefore, the minimum for n is 2 with this method, which makes it inappropriate for our project. Because, as with Decimate, a subset of the samples in the original file is retained without a change, Min/Max also requires attention to prevent aliasing.

Similar to Data Reduction, Interpolation allows choosing the Interpolation factor and one of two algorithms, Straight line or Cubic spline.

- **Straight line** simply interconnects the existing samples, creating a step-like signal with newly introduced high-frequency components.
- **Cubic spline** generates a smooth signal, again at the price of a slightly slower performance.

In this tutorial, we do not want to increase or decrease the number of samples, but only create a file with a single sampling interval. Therefore, we use the interpolation factor 1:

1. Open the **Analyze > Interpolation** dialog.

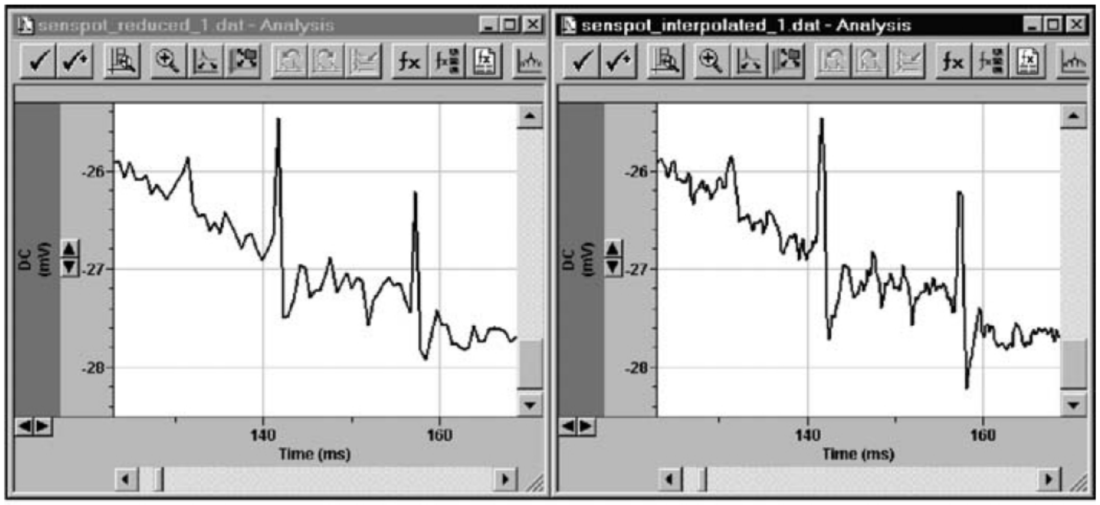

senspot_reduced_1.dat - Analysis
senspot_interpolated_1.dat - Analysis
DC (mV)
-26
-27
-28
140
160
Time (ms)

Figure 7-3: Comparison of original (left) and interpolated (right) files.

2. Select the **Cubic Spline** method.
3. Select an **Interpolation factor** of "1".
4. Click **OK**.

This creates a new file with the higher sampling rate as in the original file, as seen if you have the **Analysis** windows tiled, and compare the signals after zooming in around 1000 ms. Since we intend to use software filters in the next steps, and the sampling rate determines the range of the allowable cutoff frequency, we will use the interpolated file.

1. Select **Edit > Create Duplicate Signal**.
2. Make the new signal active by clicking on it, and select **Analyze > Filter**.
3. First we want to isolate the slow component of the response, that is, the sensillar potential. Several types of lowpass filters are available, which are described in detail in Digital Filters on page 155.
4. Select individual filters from the **Type** list to see the cutoff range reported in the Lowpass group change.
5. Select **Gaussian** as the filter type.
6. Specify a –3 dB cutoff frequency of 70 Hz.
7. In the **Trace Selection** section of the **Filter** dialog click **Select**.
8. In the **Signals** section of the **Select Traces** dialog and choose **The active signal**. Since there is only one sweep in the file, skip selecting **Traces**.
9. Click **OK** and your selection is reported in the Filter dialog.
10. Since we want to apply the filter to the entire length of the trace, set the **Region to filter** to **Full trace** and click **OK**. The action potentials are removed, except for small residuals.

The signal is called DC2 by default. To give it a more convenient name:

1. Select **Edit > Modify Signal Parameters**.
2. Type **Name > 70 Hz Lo**.
3. Click **Save**.

How can we be sure now that our choice of the cutoff frequency was appropriate? The filter should have removed the fast action potentials-which is easy to see-without affecting the shape of the slow sensillar potential-which is a little more difficult to determine. The strategy is to subtract the filtered signal from the original one

We can subtract the signals using the **Analyze > Arithmetic** command. The **Arithmetic** dialog is very similar to the **Column Arithmetic** dialog explained in the previous tutorial.

We want to compare the resulting trace to the existing signals in our file. Therefore, we must first create another signal to use as the destination:

1. Select the **70 Hz Lo** signal.
2. Select **Edit > Create Duplicate Signal** to create the new signal.
3. Open **Edit > Modify Signal Parameters**.
4. Rename the new signal **Subtracted**.
5. Click **Save**.

We are now ready to subtract the traces:

1. Open **Analyze > Arithmetic**.
2. Click **Traces** and select **Subtracted** from the **A** specified signal list.
3. Click **OK**.
4. Click the **Traces** button again and select **DC (mV)** from the **A** specified signal list.
5. Click **OK**.
6. From the **Operator** list select **- (subtract)**.
7. Click **Traces** again and select **70 Hz Lo (mV)** from the **A** specified signal list.
8. Click **OK**.

   The expression you then apply should resemble:

   T{VISIBLE} = DC:T{VISIBLE}-70 Hz Lo:T{VISIBLE}.
9. Click **OK**.

If you like, you can try other filter cutoff frequencies or filter types. The lower the cutoff frequency you specify, the more prominent is the transient before the sensillar potential in the subtracted signal. This indicates that at lower filter frequencies the faster elements of the sensillar potential are more seriously distorted. In contrast, the more you increase the cutoff frequency, the larger are the residuals of the action potentials. The 70 Hz we originally chose was a reasonable compromise.

As you try other filter types, you will notice that the infinite impulse response filters (for example all types, but Gaussian and Boxcar) require a comparatively long time to phase in, and additionally start at 0 mV. Digital Filters on page 155 explains these end effects in detail, and in the following paragraphs of this tutorial we will learn two ways to deal with them.

After we have isolated the sensillar potential, we will now take a look at the action potentials. The signal “Subtracted” already fulfills the most important criterion for the evaluation of action potentials: they sit on a non-fluctuating baseline. We will compare that to highpass filtering the signal:

1. Scale the **X Axis** so that the first 500 ms are on display.
2. Select the **DC (mV)** signal.
3. Select **Edit > Create Duplicate Signal** to create another duplicate of signal DC.
4. Select the new **DC2 (mV)** signal to make it the active signal.
5. Select **Analyze > Filter**.
6. Select **Highpass**.
7. Two types of **Highpass** filters are available: **RC (single pole)** and **Bessel (8-pole)**.
8. Select **RC (single pole)** and specify **100 Hz** as the **–3 db cutoff** frequency.
9. Click **OK**.
10. Select **View > Auto Scale > Active** Y axis to auto scale the end effect, which starts at a negative value near the original first data point.
11. To remove this transient:
    - Bring the cursors into the current region using **Tools > Cursors > Bring Cursors**.
    - Drag cursor 1 to the start of the file and cursor 2 to about 20 ms.
    - Select **Analyze > Force Values**. This function allows you to assign either values that result from the cursor positions, or a fixed value, to a region to force. The **Trace Selection** again allows specifying the signals and sweeps to include.
    - Select **Fixed value** and enter 0.
    - In **Region to force**, select **Cursors 1..2**.
    - Click **OK**.

The filtered trace exhibits a considerable residual from the steep portion of the sensillar potential, forming a downward deflection that the earlier action potentials are superimposed on. The flat region of the sensillar potential, where the late action potentials occur, has been effectively removed, however.

The other way of avoiding the end effects is removing the baseline offset before filtering:

1. Select the signal **DC (mV)**.
2. Set cursors 1 and 2 to 0 ms and 50 ms, respectively.
3. Select **Analyze > Adjust > Baseline**.

This feature corrects an offset or drifting baseline that you can specify using one of four methods. Subtract mean and Subtract slope use the region selected in the drop-down list:

- **Mean** calculates the average of all data points in the specified region
- **Slope** fits a regression line, then the result is subtracted from the entire trace, so that the specified region coincides with a straight baseline at y = 0.
- You can enter a fixed value, which is to be subtracted from the selected traces.
- You can correct an irregular baseline if you select **Adjust manually** and click on the pink correction line to create break points that can be dragged to follow an irregular curve.

Assuming the data values before the onset of the sensillar potential represent the baseline, the most suitable selection for our purpose is the mean between Cursors 1..2:

1. Select **Subtract mean of**.
2. Select **Cursors 1..2** from the list.
3. Ensure that **Active signal** is selected in the **Trace Selection** group, and click **OK**.

As you create duplicates of DC now, they start at zero and the end effects of the filters are minimized. Depending on the application, the amplitude offset in the original data file might be relevant. In this case, you should measure it before you adjust the baseline, or perform the adjustment in the duplicate signal, alternatively.

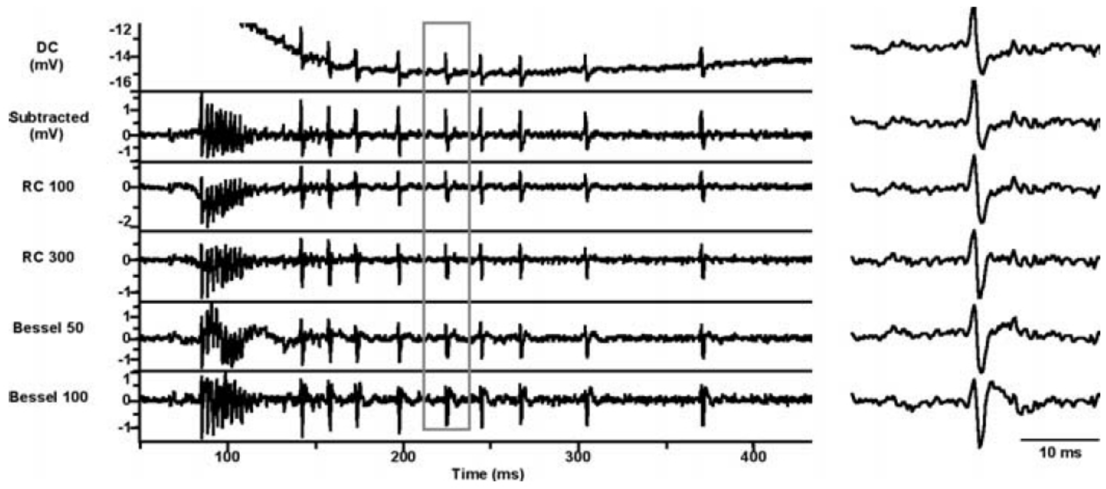


Figure 7-4: Effects of different filter settings.

As you apply a highpass filter to the next duplicate signal now, the end effect does not occur. Try the RC filter at a higher cutoff frequency to see the sensillar potential almost completely eliminated, but the action potential waveform increasingly distorted. At a comparatively low cutoff frequency of 50 Hz, the 8-Pole Bessel highpass filter introduces ringing in response to fast changes in the signal, such as the steep portion of the sensillar potential. If the cutoff frequency is increased to 100 Hz, the ringing disappears, but the action potential waveform is distorted more severely. The right side of the figure above illustrates the influence of the different filter algorithms and frequencies upon the indicated section on the left. Depending on the application, the choice of the suitable highpass filter is far more critical than for lowpass filters. Filtering individual regions of the signal using different algorithms or cutoff frequencies might be necessary for specific applications.

Now that we have separated the sensillar potential from the action potentials, we can go on evaluating parameters that describe the response. **Use Tools > Cursors > Write Cursors** to write the times of the action potentials to the Cursors sheet in the Results window or apply **Analyze > Statistics**. In addition, you can use the cursor measurements to characterize the sensillar potential. Or you can try to mathematically describe the time course of its initial phase:

1. Right-click in the data portion of the **70 Hz Lo** signal and select **Maximize Signal** from the right mouse menu.

2. Zoom the first 500 ms by clicking and dragging the cursor on the X axis.
3. Set cursors 1 and 2 to 85 ms and 240 ms, respectively.
4. Select **Analyze > Fit**.

pCLAMP Software provides highly sophisticated fitting features, giving you the opportunity to select from a set of predefined functions, or to specify a custom function. Four different **Search Methods** are available, in combination with various **Minimization** and **Weighting Methods**. A number of additional options are available on the **Data/Options** tab, and the **Seed Values** tab allows you to enter initial parameter estimates, either in a purely numerical way, or with the assistance of a graphical representation. Not all options available in the dialog apply to all datasets, methods, or functions. Since fitting is a fairly complex issue, it is beyond the scope of this tutorial to explain every detail. We strongly recommend that you read Curve Fitting on page 205 for more detailed information on the functions, the algorithms and trouble shooting. In addition, a number of ATF files can be found in the **..\Program Files\Molecular Devices\pCLAMP10.4\Sample Data** folder. Their file names indicate their functions and they are ideally suited to help you become more familiar with the **Fit** feature.

We are now going to investigate whether the steep initial phase of the sensillar potential can be described by an exponential function. Presumably several processes with different time constants superimpose to shape this waveform (Kaissling 1998, Vermeulen & Rospars 1998). Therefore, we will test exponential functions of different orders and compare these models. For this purpose pCLAMP Software provides an automatic comparison of models.

Fitting generally applies to the region between cursors 1 and 2. They already confine the region we are going to fit:

1. Select **Exponential, standard** from the **Predefined Function** group on the **Function/Method** tab.
2. Deselect **Select fitting methods automatically** and select **Levenberg-Marquardt**, and **Weighting Method > None**.
3. On the **Data/Options** tab, enable **Compare Models**.
4. The **Starting term** and **Ending terms** are automatically set to the maximum possible; the 95% **Confidence level** should be adequate.
5. On the **Seed Values** tab, you can review or alter the estimates pCLAMP Software has automatically generated. If you change the original values, they can optionally be restored with the **Auto-Estimate** button.
6. Click **OK**.

   pCLAMP Software starts fitting exponential functions of increasing order to the data. While the **Automatic** check box in the **Compare Models** group is enabled, the comparison stops if a higher-order model does not improve the fit. If the **Ending term** is set manually, however, pCLAMP Software continues up to the specified order.

7. Select **Analyze > Fitting Results**. The **Parameters** and the fit **Statistics** are displayed on two tabs. The **Model number** field at the bottom of the dialog allows you to review the values for the different models. In the **Best Model** group on the **Statistics** tab pCLAMP Software reports that, under the specified fitting conditions, a second order exponential function reasonably describes the fitted portion of the data.
8. Click **Copy to > Clipboard** or **Lab Book** to copy to the respective destinations the results for the model that is currently displayed in the dialog.

Separating Action Potentials by Their Shape

Electrophysiological recordings often comprise rather similar signals, which slightly differ in their amplitudes or kinetics because they originate from different cells. One well-known example of these signals are action potentials extracellularly recorded from insect sensilla (Frazier & Hanson 1986, Schnuch & Hansen 1990). This tutorial will demonstrate how pCLAMP Software can be used for characterizing the waveforms in a series of similar signals. Then we will investigate whether the differences in the measured parameters permit us to assign them to different subpopulations, and finally we will save them in two separate files for further evaluation.

1. Open the sample file *spikes02.abf*. The file contains spontaneous action potentials of two different types, which were extracellularly recorded from an insect olfactory sensillum. The file was acquired in high-speed oscilloscope mode, using the highpass-filtered signal AC as the trigger channel. For waveform analysis, only an unfiltered signal is relevant:
2. Right-click in the data area of the signal **DC** and select **Maximize Signal**.
3. One of the sweeps was obviously triggered by an artifact. You can exclude it from the analysis by deselecting it and performing all analyses on the **Visible Traces** only.
4. Click on the artifact to make it the active sweep. In the lower right corner of the **Analysis** window pCLAMP Software reports the number of the active sweep as **63**.
5. Open **View > Select Sweeps**.
6. Select **Sweep 63** and click **Invert**.
7. Click **OK**.
8. In **Sweep 39** there is another artifact. You can either remove it using **Analyze > Force Values**, or exclude it from the analysis by specifying the **Region to analyze** with cursors 1 and 2. We will use the second method here.

There is a slow drift in the baseline the action potentials sit on, as can be seen best in **View > Data Display > Concatenated** mode. Depending on the recording method, slow drifts can have different origins. Frequently seen is a slowly drifting electrode potential, when non-chlorided metal electrodes such as tungsten or platinum are used (Geddes 1972). Insect sensilla often exhibit slow, occasionally oscillatory, changes in their steady state potential, whose origin is not completely understood. A common way to deal with slowly drifting signals are AC-coupled recordings. In the previous tutorial, possible effects of highpass filtering on the signal waveform were demonstrated for digital filters. Comparing the signals AC and DC in the sample file, we can see that analog filters equally affect the waveform. Therefore, we remove the drift in the baseline using **Analyze > Adjust > Baseline**:

1. First, use **Edit > Create Duplicate Signal**. We are going to use this to compare the Subtract mean and Adjust manually baseline adjustment methods.
2. Right-click in the data area and select **Show Acquisition Signals** to show all signals.
3. Select the signal **DC**.
4. Select **View > Data Display > Sweeps**.
5. Set cursors 1 and 2 to 0 and 2 ms, respectively. If the X axis units are in seconds you can change to milliseconds using the Units list on the X Axis tab of the **View > Window Properties** dialog.
6. Select **Analyze > Adjust > Baseline**.
7. Select **Subtract mean of Cursors 1..2**. Ensure the **Trace Selection** is **Active signal** and **All visible traces**.
8. Click **OK**.
9. Select **View > Data Display > Concatenated**.
10. Select the signal **DC2** and select **Maximize Signal** from the right mouse menu.
11. Open the **Analyze > Adjust > Baseline** dialog and select **Adjust manually**. Click **OK**.
12. A pink correction line is displayed, which can be dragged to shape by clicking on it to create moveable break points. The X and Y coordinates of the most recently altered point are reported in a floating toolbar. When the line closely follows the drift of the baseline, click **OK**.
13. Select **View > Data Display > Sweeps** and **Show Acquisition Signals** to inspect the differences between the two methods.

While the manual adjustment exhibits a surprising accuracy and can lead to a satisfying result depending on the application, in this case Mean adjustment results in less variation between sweeps. So we will use the signal DC.

The next step in the waveform analysis is aligning the peaks of the action potentials. In the original file the signals are aligned at the threshold crossing point at 4 ms (100 samples). pCLAMP Software can shift sweep type data to either direction for a fixed time interval, which can be entered numerically, or alternatively defined by the position of one of the cursor pairs.

Analyze > Time Shift removes a number of samples corresponding to the shifted interval at one end of the **Analysis** window. There are two options to deal with these “wrapped” samples. **Rotate samples** adds them to the other end of the sweeps. This option is useful when you are not completely sure about the time interval, because a **Time Shift** can be undone by shifting in the opposite direction. However, the rotated data points are not separated from the rest of the data in any way. So you should be aware that every time shift affects the integrity of the data. The other option, **Replace wrapped samples with zeros**, makes it easier to recognize the samples that are not part of the actual data file. The time shift cannot be undone, however, unless the original file is reopened.

The waveform analysis we are going to do requires that all action potentials be aligned with a prominent element, namely their positive peaks:

1. Move cursors 1 and 2 so that they encompass the peaks but exclude the artifact in **Sweep 39**.
2. Open **Analyze > Time Shift**.
3. Select **Align Peaks**.
4. Select **DC (mV)** from the **Signal to search** list.
5. Select **Cursors 1..2** from the **Region to search** list.
6. Since the interval to shift is different for each individual sweep, which excludes undoing the shift anyway, select **Replace wrapped samples with zeroes**.
7. Ensure that **All visible traces** is set for the **Trace Selection**.
8. Click **OK**.

Next, using **Analyze > Statistics** we are going to determine waveform parameters that might be characteristic for the individual types of the action potentials. The measurements pCLAMP Software can perform on the data file are essentially the same that Clampex Software offers during acquisition, so if the recordings are well-reproducible regarding time course and amplitude, it will save time if you set up Clampex Software to perform the measurements in real-time.

1. Set the cursors 1 and 2 to 5 ms and 10.8 ms, respectively, so they include the action potentials at their bases, but exclude the transient in sweep 39.
2. Ensure that **DC (mV)** is selected in the **Trace Selection** group, and that **Peak Polarity** is set to **Positive-going**.
3. Ensure that the **Search Region** is set to **Cursors 1..2**.
4. For our action potentials, not all pCLAMP Software measurements are relevant. While the Peak amplitude is interesting, the Time of peak is identical for all sweeps, because we aligned the peaks. The Antipeak amplitude and the Time of antipeak might be different for the two types of spikes and can be further evaluated. The other measurements, such as the Area under the peaks and the kinetics of rise and decay, as well as the Half width might yield differences between the two spike types and should be included in the further evaluation. Select all relevant measurements.

   After **OK** pCLAMP Software writes the measurements to the **Statistics** sheet in the **Results** window.

We assume that the action potentials belong to two populations. First we will distribute them into two groups, evaluating their peak-to-peak amplitude. Then we will investigate whether the other parameters in these two groups are significantly different from each other.

In the **Results** window **Statistics** tab select the column named **R1S1 Time of Antipeak (ms)** by clicking on its header. Now select **Edit > Insert > Column**s to create a new column. Double-click on the new column's title and rename it **Peak-Peak**.

For computing the peak-to-peak amplitude, use **Analyze > Column Arithmetic** to subtract the **Antipeak** from the **Peak** column. See the second tutorial for details on Column Arithmetic. The peak-to-peak time can be calculated by subtracting 6 ms, for example the time of the positive peaks after aligning, from the **Antipeak (ms)** column.

There are different ways of determining the threshold between the large and the small action potentials. The simplest way is by creating a scatter plot. If there are clear differences, they should show up with this comparatively coarse method:

1. Select the **Peak-Peak** column on the **Statistics** sheet by clicking on the column header, and go to **Analyze > Create Graph**. By default, the selection is assumed to include the dependent variable Y, and is plotted versus an incrementing X.
2. Right-click on the plot and select **Properties**, or alternatively go to **View > Window Properties** in the main menu to open the **Window Properties** dialog.
3. On the **Plots** tab set the **Curve Type** to **Scatter** and click **OK** to remove the connecting lines.

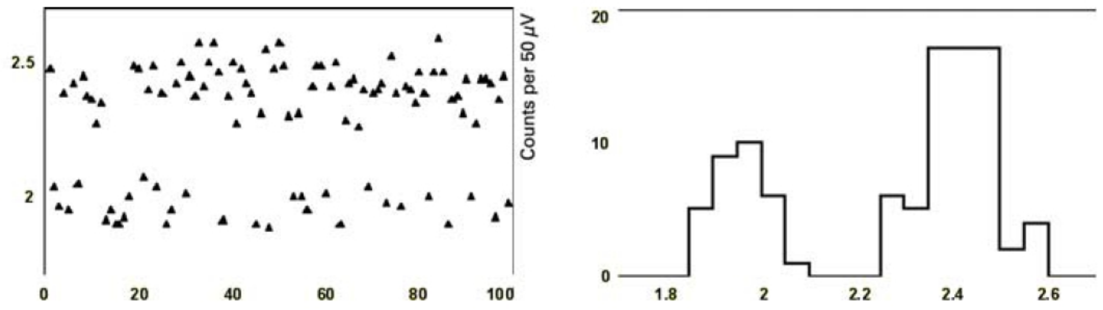


Figure 7-5: Scatter plot and histogram showing groupings in peak-peak measurements.

In our example, there are obviously no peak-to-peak amplitudes between 2.10 and 2.25 mV. Another way to determine the threshold, which can also reveal less evident differences, is a frequency distribution histogram. This requires the definition of bins.

1. With the **Peak-Peak** column selected, go to **Analyze > Basic Statistics**. Of the **Statistics Options** that are available in this dialog, only the **Number per category** is relevant for our project.
2. Select **Perform Breakdown Analysis**, select **Peak-Peak** in the **Category** column drop-down, check **Bin the categories**, and click **Specify Bins**. A dialog is displayed that allows you to specify bins of user-defined or fixed size. To cover the entire range of the occurring peak-to-peak amplitudes, specify 20 **Fixed Bin Size** with the **Initial Value 1.7 mV** and a **Width** of **0.05 mV**.

3. Click **OK** in the **Specify Bins** dialog and **OK** in the **Basic Statistics** dialog. pCLAMP Software writes the statistics on the **Basic Stats** sheet in the **Results** window.
4. Select the **Bin center** column first and click the **X** toolbutton to make it the X column, which is reported in the column header. Then select the #/Cat column and click **Y+**.
5. Now go to **Analyze > Create Graph**, or click the toolbutton. Unless a different **Graph** window is open, the default template is a **Line Graph**. Select **Properties** from the right-click menu and, on the **Plot** tab, set the **Plot Type** to **Histogram**. On the **Y Axis** tab, enter **0** in the **Bottom limit** field.

After clicking **OK**, the plot should look similar to the histogram in Figure 7-5. Again there is no amplitude around 2.15 mV. So we can state the null hypothesis that the action potentials with a peak-to-peak amplitude below this threshold are different from those having an amplitude above. In the following steps, we will test this hypothesis.

pCLAMP Software features a number of statistical tests, which are accessible in the bottom section of the Analyze menu. See the Help file for details on their use and their algorithms. For each of the parameters we determined, we want to compare two samples of unequal size, and calculate the probability that they originate from two different parent populations. Several tests can be used to investigate this question: an F-Test followed by an unpaired Student’s t-Test, a nonparametric Mann-Whitney UTest, and One-Way ANOVA (Sokal & Rohlf 1981). The general approach is identical for all statistics features available from the Results window, so only the F- and t-Tests are demonstrated here.

1. On the **Statistics** tab of the **Results** window select the columns **R1S1 Peak Amp** through **R1S1 Half-width (ms)** and go to **Analyze > F-Test and Student’s t-Test**.
2. Check **F-Test** and the two **Unpaired** options in the **Student’s t-Test** group. The lower half of the dialog is identical to the **Basic Statistics** dialog. The **Columns** selection should still be **All selected columns**, **Perform Breakdown Analysis** with **Bin the categories** from the column **Peak-Peak** still active.
3. In contrast to the previous procedure, this time we use two user-defined bins, comprising the large and the small peak-to-peak amplitudes, respectively. Select **Specify Bins** and select **Defined Bin Size**. Using the **Add** button, specify two bins, the first ranging from 0 to 2.15 mV, and the second from 2.15 to 5. The **Edit** and **Delete** buttons are only enabled when one of the bins in the **Bin Number** column is highlighted.

Upon OK, the Lab Book window is displayed, reporting a number of parameters for every column we included in the analysis. At the beginning of each column-related section, general information about the input values is reported, such as the sample size, the mean, and the variance within each bin. Then the test results are reported, which in our case includes the F-value and F-probability. This parameter indicates whether the variance of the two groups is different. Below that, the results of the t-test are listed: the t-value, the probability, and the degree of freedom.

Depending on the F-probability, use the t-probability from the pooled or the separate t-test in the further course, even if you find that there is virtually no difference for our example. A probability value of 0.0000 means that the probability that all peak amplitudes of the action potentials belong to the same population is less than 10^{-4}.

Table 7-1: Statistical results.

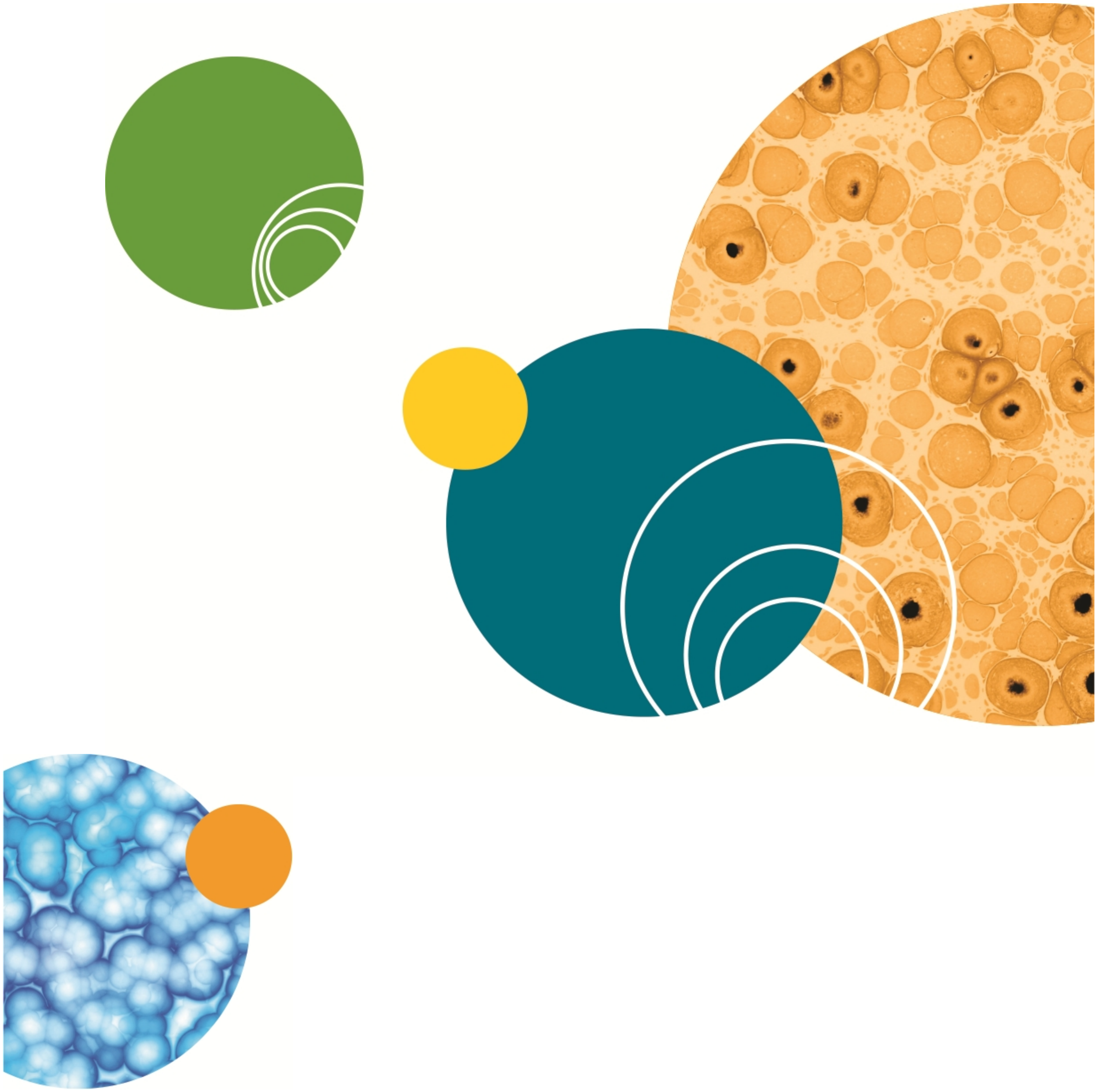
============ **H: Peak**	========= 0<=Bin 0<2.15	========= 2.15<=Bin 1<5	
Sample Size	31	68	
Mean	1.3132	1.7075	
Variance	0.0041	0.0052	
Degree of freedom ===========	30	67	
F-Value	1.2461		
Probability	0.5129		
	t-Value	**Probability (2-tail)**	**Degree of freedom**
============ **Unpaired (pooled var.)**	-26.1606	0.0000	97
Unpaired (separate var.) ============	-27.2656 =========	0.0000 =========	64 =========

To summarize, of the 12 parameters we evaluated, 5 are significantly different for the two types of action potentials we distinguished ($p< 10^{-4}$). They are all correlated with the amplitude. The peak-to-peak amplitude is not included, since it was the basis of our null hypothesis. For the reasons explained above, Time of peak, Mean and Standard deviation are also not considered here. Virtually all parameters that describe the time course of the action potentials are not significantly different for the two groups ($p > 0.05$). Only the time of the greatest right slope ($p = 0.04$) is different, but on an extremely weak basis. So we have collected good evidence that the data file contains two types of action potentials with different amplitudes. For further evaluation we are going to save them in separate files now.

First we sort them in the order of increasing peak-to-peak amplitude:

1. In the **Results** window, select the **Peak-Peak** column and go to **Analyze > Sort**. The current selection determines the default Key Column, but you can also highlight any other column in the drop-down list.
2. The default **Row Range** also depends on the selection, in our example it includes all non-empty rows. The **Columns To Sort** should normally comprise all columns, because the relation between individual columns gets lost if only a subset of them is selected here. The data can be sorted in either **Ascending** or **Descending Sort Direction**, and finally sorting can be performed in a Case sensitive way, if the Key Column contains strings.
3. Click **OK**.

4. After sorting, scroll down until you reach the gap in between the sub- and suprathreshold peak-to-peak amplitudes. The last subthreshold value should be found in Row 31. To better separate the two result blocks, select Row 32 by clicking on its header and go to **Edit > Insert > Rows**.
5. Select the subthreshold Rows in the Trace column and sort them in ascending order:
   - Tile the **Results** and **Analysis** windows vertically, select the **Analysis** window and go to **View > Select Sweeps**.
   - In the **By Selection** field, highlight those sweeps whose numbers are in the **Trace** column now, all the time holding down the **<Ctrl>** key.
   - Before you click **OK**, select **By user-entered** list and copy the contents of the **Range** field, which exactly reflects your selection, to the Windows clipboard. Depending on the application window size, you might have to select the sweeps in two portions, scrolling down in the **Results** window. This can easily be done, if you again click **<Ctrl>** before you start highlighting the remainder of the sweeps during the second turn. After clicking **OK**, only the small action potentials are on display
6. Save the file as *spikes_s.abf* now, making sure that the **Data** selection reported in the **Save Data As** dialog is Visible sweeps and signals.
7. Reopen the original data file, adjust the baseline, align the peaks and open the select sweeps dialog once more.
8. Paste the clipboard contents into the **Range** field and click the **Invert** button. the selection now includes the sweep with the artifact (#63), so deselect it by clicking while you hold down **<Ctrl>**. Then save the **Visible** sweeps as *spikes_l.abf* for further evaluation.



Chapter 8: Digital Filters

8

In digital signal processing a system is something that operates on one or more inputs to produce one or more outputs. A digital filter is defined as a system, in the case of pCLAMP Software, a software algorithm, that operates on digitized data to either pass or reject a defined frequency range. The objective of digital filtering is to remove undesirable frequency components from a digitized signal with minimal distortion of the components of interest.

There will be instances when it is necessary to filter experimental data after they have been digitized. For example, you might want to remove random noise or line frequency interference from the signal of interest. To this end, pCLAMP Software offers several types of digital filters.

The lowpass filters include Bessel (8-pole), boxcar, Butterworth (8-pole), Chebyshev (8- pole), Gaussian, a single-pole RC and an 8-coincident-pole RC. The highpass filters include Bessel (8-pole) and 8-coincident-pole RC. The Gaussian and boxcar filters are finite impulse response (FIR) filters while the Bessel, Butterworth, Chebyshev and RC filters are infinite impulse response (IIR) filters (see following section: Finite vs. Infinite Impulse Response Filters on page 155).

A notch filter is available to reject a narrow band of frequencies and an electrical interference filter is provided to reject 50 or 60 Hz line frequencies and their harmonics.

Finite vs. Infinite Impulse Response Filters

Digital filters can be broadly grouped into finite impulse response (FIR) filters and infinite impulse response (IIR) filters. FIR filters are also referred to as *nonrecursive* filters while IIR filters are referred to as recursive filters.

The output of FIR filters depends only on the present and previous inputs. The general “recurrence formula” for an FIR filter, which is used repeatedly to find successive values of *y*, is given by:

$$y_n = \sum_{k=0}^{M} b_k x_{n-k}$$

where y_n is the output value for the n^{th} point *x* and b_k is the k^{th} of *M* filter coefficients. In the case of the Gaussian and boxcar filters in pCLAMP Software, the *M* points ahead of the current point are also used, giving a general recurrence formula of:

$$y_n = \sum_{k=-M}^{M} b_k x_{n-k}$$

where the filter width is 2(*M* + 1) points.

The disadvantage of FIR filters is that they can be computationally inefficient as they might require several tens, hundreds or even thousands of coefficients depending on the filter characteristics.

The advantages are that FIR filters are inherently stable because there is no feedback and they possess ideal linear phase characteristics, exhibiting no phase distortion. That is, all frequency components passing through the filter are subject to the same pure time delay.

On the other hand, the output of IIR filters depends on one or more of the previous output values as well as on the input values. That is, unlike FIR filters, IIR filters involve feedback. The general recurrence formula for an IIR filter is given by:

$$y_n = \sum_{j=1}^{N} a_j y_{n-j} + \sum_{k=0}^{M} b_k x_{n-k}$$

where *a* and *b* are the *N* and *M* filter coefficients, where *a* represents the feedback coefficients. Note that the value of *y* for a given point *n* depends on the values of previous outputs y_{n-1} to y_{n-N} as well as the input values *x*.

The major advantage of IIR filters is that they are computationally more efficient, and therefore much faster, than FIR filters. The disadvantages are that IIR filters can become unstable if the feedback coefficients are unsuitable, and recursive filters cannot achieve the linear phase response that is characteristic of FIR filters. Therefore, all IIR filters introduce a phase delay to the filtered data.

The problem of potential instability of IIR filters is solved in pCLAMP Software by limiting the cutoff frequencies for all filter types to a range where the response is always be stable (see Cutoff Frequency Limitations on page 181.) However, the phase delay is not corrected.

The Nyquist rate (see The Sampling Theorem in pCLAMP Software on page 25) has important consequences for digital filtering in that the maximum analog frequency that a digital system can represent is given by:

$$f_h = \frac{1}{2T}$$

where *T* is the minimum sampling interval and f_h is the Nyquist frequency.

As a consequence of this, the maximum filter cutoff frequency of digital filters is limited to one-half the sampling rate. That is, the ratio of the cutoff frequency to the sampling rate (f_c/f_s) cannot exceed 0.5. In fact, only the Gaussian and single pole RC filters can realize an f_c/f_s ratio as high as 0.5; the Bessel, Butterworth, Chebyshev and notch IIR filters are limited to values that are somewhat lower than this because of performance degradation at higher f_c/f_s ratios (see Cutoff Frequency Limitations on page 181.)

The f_c/f_s ratio limitation should not present a problem if antialias filtering and oversampling are judiciously applied. For example, with a lowpass antialiasing filter cutoff of 4 kHz and a sampling rate of 40 kHz, an f_c/f_s ratio limited to as low as 0.1 allows a maximum cutoff frequency of 4 kHz, which is well above any useful cutoff frequency that might be applied to this particular digitized record.

Digital Filter Characteristics

An ideal filter would have a rectangular magnitude response with no attenuation in the passband and full attenuation in the stopband. However, ideal filters are *noncausal* in that the present output depends on future values of the input. They are, therefore, not realizable. However, realizable digital filters can approximate ideal filters in that the output can be delayed for a finite interval until all of the required inputs have entered the system and become available for determination of the output.

Different filter types optimize different characteristics of the ideal filter that they are meant to approximate. Therefore, the application of a particular filter type should be carefully considered in view of the specific requirements at hand.

Most filters introduce a time lag between the input and output signals. Depending on the filter type, some frequencies are subjected to a greater lag than others. As a consequence the output signal is distorted to some degree. This distortion takes the form of “ringing” and “overshoot” in the filter output given a step function input (for example. a square pulse). Filters that introduce equal time lags for all frequencies are said to have a constant “group delay”. Such filters exhibit minimal ringing and overshoot.

A filter can be characterized by its cutoff frequency and steepness of response. In the pCLAMP Software filters the cutoff frequency is defined as the frequency at which the signal amplitude decreases by a factor of 2. This corresponds to a drop in power of 1/√ 2, or -3 decibels (dB).

The steepness of a filter, its “roll off,” defines the rate at which a signal is attenuated beyond the cutoff frequency. It is desirable to have as steep a roll off as possible so that unwanted frequencies are maximally attenuated. However, filters that are designed for maximally steep rolloffs necessarily sacrifice constant group delay characteristics, and therefore exhibit ringing and overshoot.

The steepness of a filter response is also a function of its order (number of poles): the higher the filter order, the steeper the response. Apart from the single pole RC filter, the IIR filters in pCLAMP Software are 8-pole realizations.

End Effects

All software filters exhibit “end effects” at the beginning or end of a data record. In the case of the boxcar and Gaussian filters, end effects occur at both ends of the record because these filters use both previous and succeeding points to filter the current point. Clearly, at the beginning of the record only succeeding points are available. These filters are, therefore, phased in progressively as previous points become available. Towards the end of the record fewer and fewer following points become available so the filter is progressively phased out. The filter coefficients are adjusted during these phases in accordance with the available number of points.

Filters with fewer coefficients exhibit shorter end effects as the full operating width overlaps a fewer number of points. The number of settling points required for the Gaussian and boxcar filters is (filter length –1)/2. In the case of the Gaussian the filter length is equal to the number of coefficients, while in the case of the boxcar the filter length is equal to the number of averaging points.

IIR filters exhibit startup-transients only. Since these filters do not use points ahead of the current point for the filter output there is no phase-out transient. The output of these filters depends on previous outputs as well as the current input. Therefore, a certain number of points must be processed before these filters reach stability.

The rise time (T_r) of a lowpass filter can be estimated from $T_r = 0.35/fc$. As the filter settles to within about 10% of its final value in one rise time, a duration of 3 x T_r is sufficient to allow the filter to reach 0.1% of its final value, requiring about $(3 \times 0.35/f_c) \times f_s$ points.

Bessel Lowpass Filter (8 Pole) Specifications

- 10% to 90% step rise time: $0.3396/_{fc}$
- Maximum overshoot: 0.4%
- Attenuation: 114 dB at $f = 10\ _{fc}$

A Bessel lowpass filter has a maximally flat response over the entire frequency range (constant group delay characteristics), exhibiting minimal overshoot and ringing in response to a step function. As all frequencies are delayed equally the shape of the original signal is preserved. Because of these characteristics, Bessel filters are most commonly used for time-domain analysis of biological data, where the preservation of the shape of the original signal is critical.

Expected vs. Observed Overshoot

100 mV step pulse, fs = 10 kHz

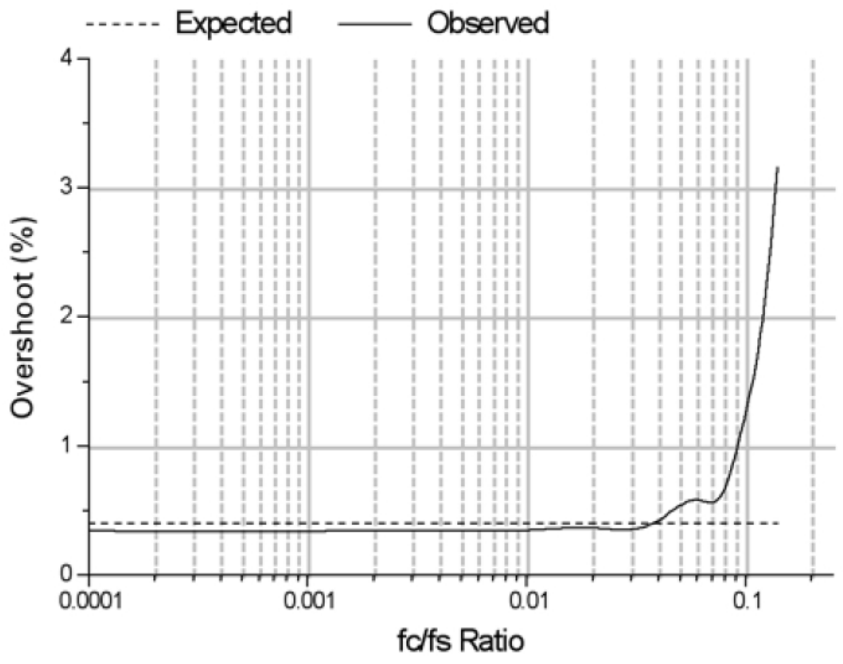


Figure 8-1: Bessel lowpass filter (8 pole) expected vs. observed overshoot.

Expected vs. Observed Rise Times

100 mV step pulse, fs = 10 kHz

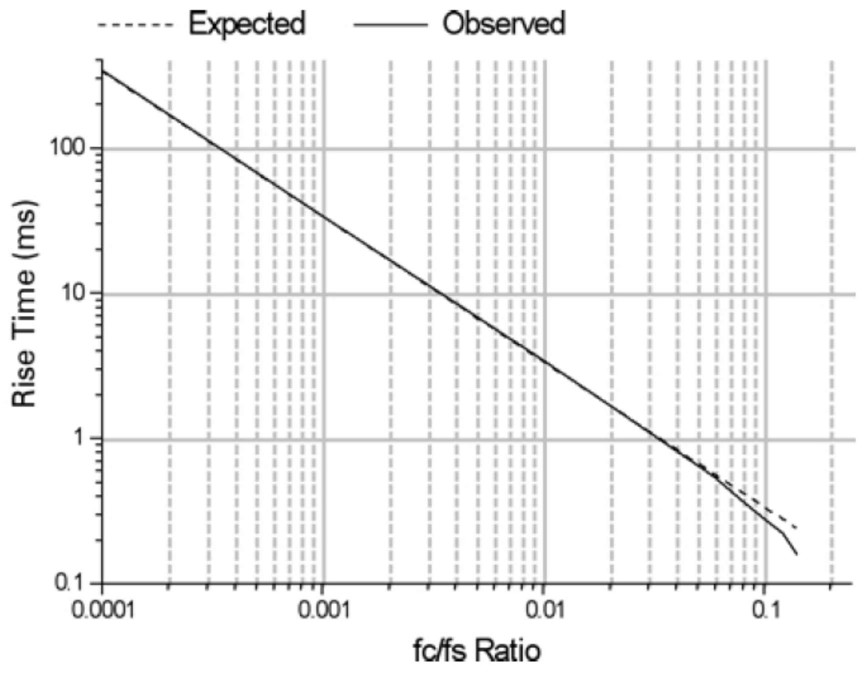


Figure 8-2: Bessel lowpass filter (8 pole) expected vs. observed rise time.

Normalized Frequency Response

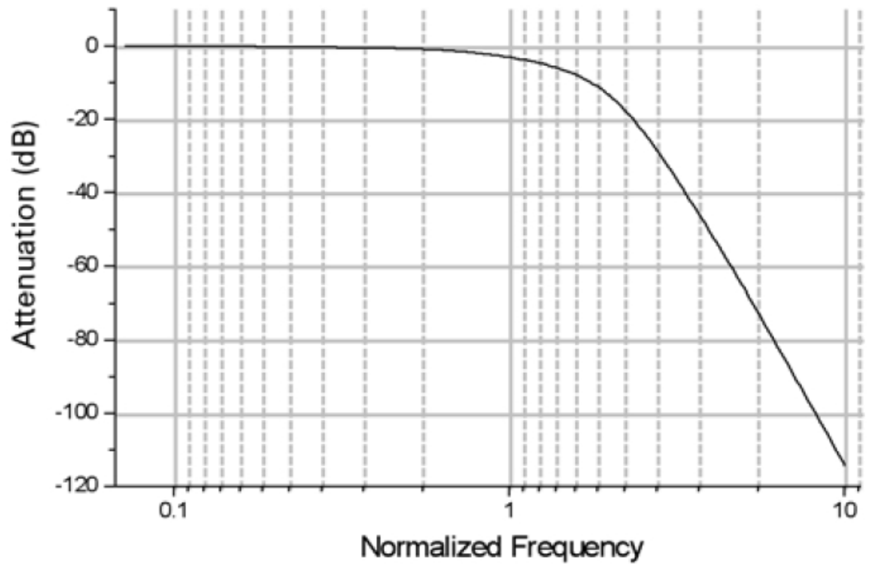


Figure 8-3: Bessel lowpass filter (8 pole) normalized frequency response.

Boxcar Smoothing Filter Specifications

Smoothing filters are generally used to remove high-frequency components from slowly varying signals and are therefore lowpass filters. The boxcar-smoothing filter uses the average of the current point and a given number of previous and succeeding points to replace the value of the current point. The recurrence formula for a boxcar filter is:

$$y_n = \sum_{k=-M}^{M} \frac{x_{n-k}}{P}$$

where x_n is the nth point to be filtered (at k = 0), P is the number of smoothing points (the filter width) and M = (P – 1)/2. The boxcar filter does not introduce a time lag.

Like the other filters, the boxcar filter also attenuates the signal; the degree of attenuation is directly proportional to the frequency of the signal and the number of smoothing points. The Figure 8.4 compares the attenuation of 10, 50 and 100 and 500 Hz sine waves (sampled at 10 kHz) at various filter lengths:

Boxcar Filter Attenuation vs. Number of Smoothing Points

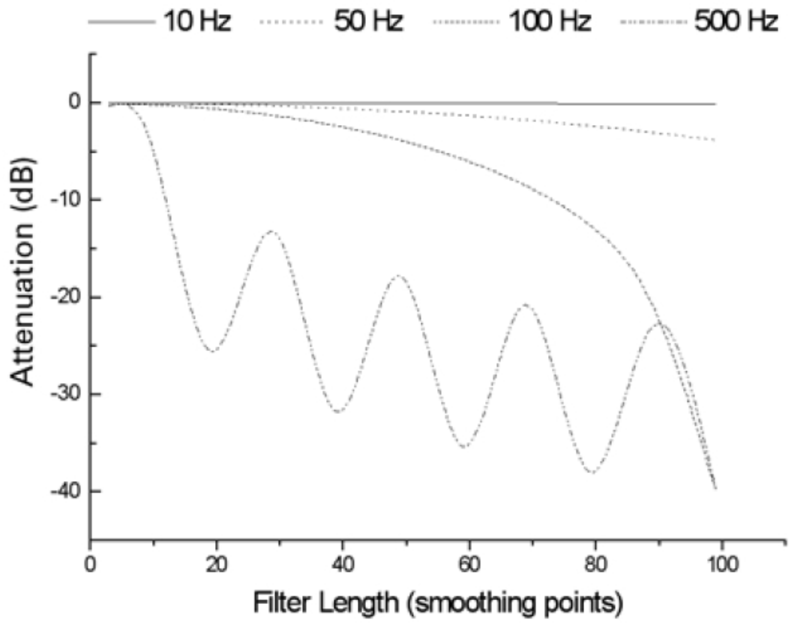


Figure 8-4: Boxcar filter attenuation vs. number of smoothing points.

Filtering periodic signals with the boxcar filter can introduce a periodic attenuation response, as seen with the 500 Hz signal. This occurs because the filter output for the current point is the mean of its value and the values of its immediate neighbors. The output therefore depends on the relative proportion of high and low data values within a given filter length.

Butterworth Lowpass Filter (8 Pole) Specifications

- 10% to 90% step rise time: $0.46/f_c$
- Maximum overshoot: 16.0%
- Attenuation: 160 dB at $f = 10\,f_c$

The Butterworth lowpass filter has a maximally flat response at low frequencies and a monotonically decreasing amplitude response with increasing frequency. The group delay is not constant so the Butterworth filter exhibits ringing and a substantial overshoot in response to a step function. This filter, however, has sharper roll-off characteristics than the Bessel filter. It is, therefore, better suited than the Bessel for frequency domain applications such as noise analysis. However, because of its nonconstant group delay characteristics this filter should generally not be used for time-domain analysis of biological data.

Expected vs. Observed Overshoot

100 mV step pulse, f_s = 10 kHz

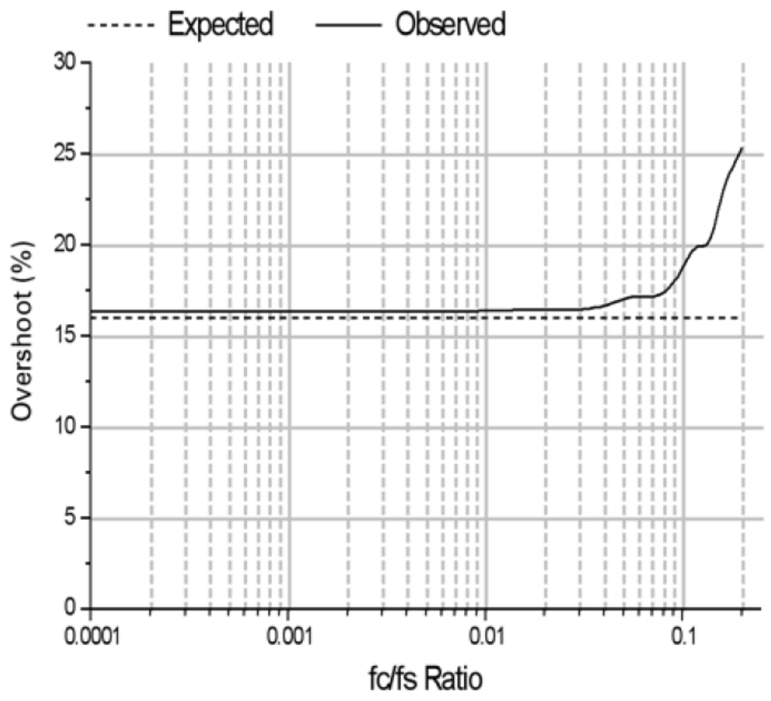


Figure 8-5: Butterworth lowpass filter (8 pole) expected vs. observed overshoot.

Expected vs. Observed Rise Times

100 mV step pulse, f_s = 10 kHz

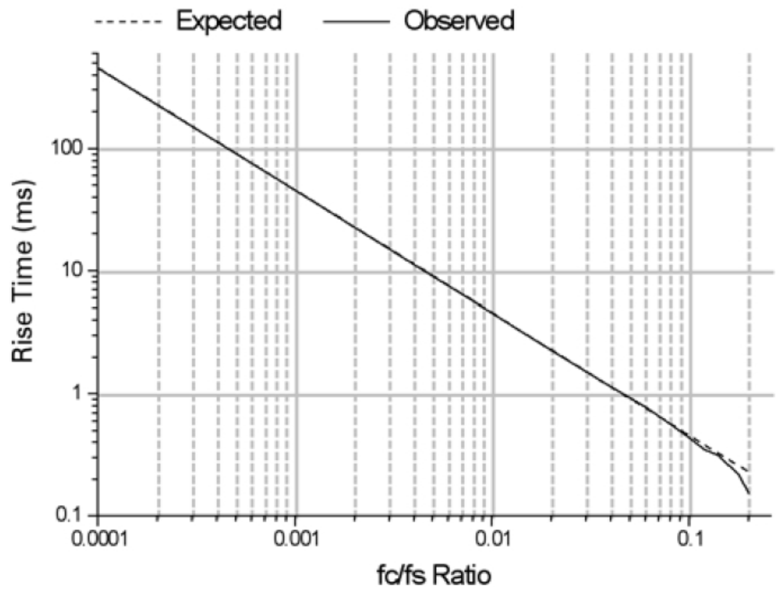


Figure 8-6: Butterworth lowpass filter (8 pole) expected vs. observed rise time.

Normalized Frequency Response

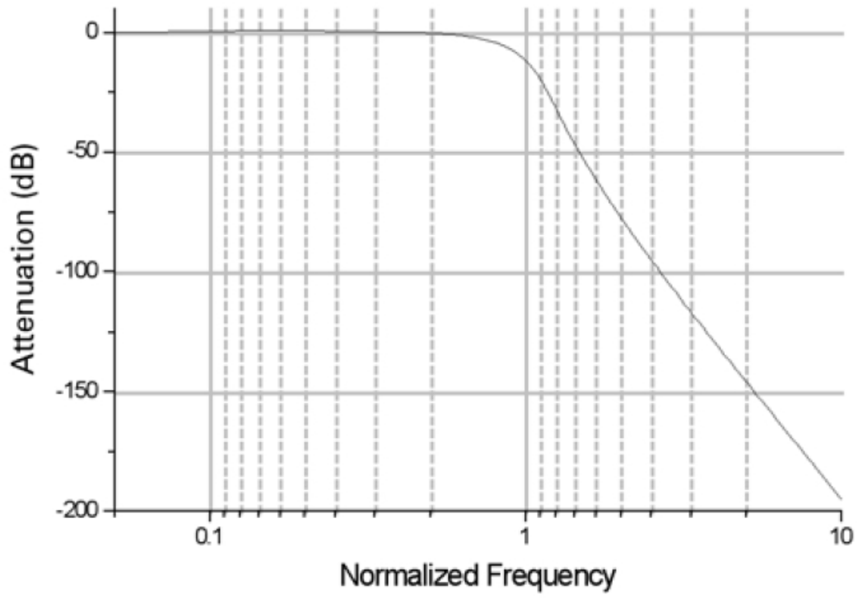


Figure 8-7: Butterworth lowpass filter (8 pole) normalized frequency response.

Chebyshev Lowpass Filter (8 Pole) Specifications

- 10% to 90% step rise time: $0.53/f_c$
- Maximum overshoot: 16.0%
- Attenuation: 193 dB at $f = 10\ f_c$

The Chebyshev lowpass filter has a maximally sharp transition from the passband to the stopband. This sharp transition is accomplished at the expense of ripples that are introduced into the response. The Chebyshev filter in pCLAMP Software has a fixed ripple of 1 dB. Like the Butterworth, the sharp roll-off characteristics of the Chebyshev filter make it suitable for analysis of data in the frequency domain, such as noise analysis. Although the Chebyshev filter has a sharper roll-off than the Butterworth, it exhibits an even larger overshoot and more ringing. Therefore, it is also not generally suitable for time-domain analysis of biological data.

Expected vs. Observed Overshoot

100 mV step pulse, fs = 10 kHz

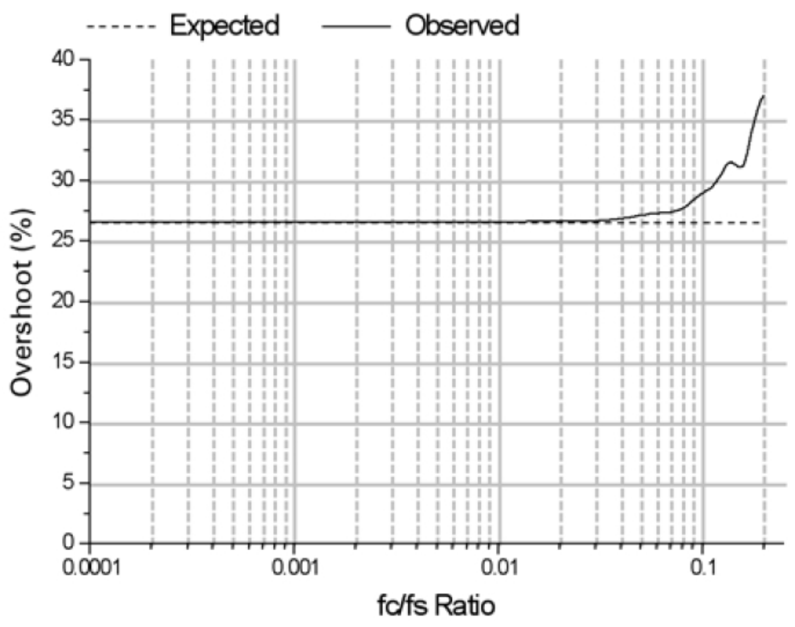


Figure 8-8: Chebyshev lowpass filter (8 pole) expected vs. observed overshoot.

Expected vs. Observed Rise Times

100 mV step pulse, fs = 10 kHz

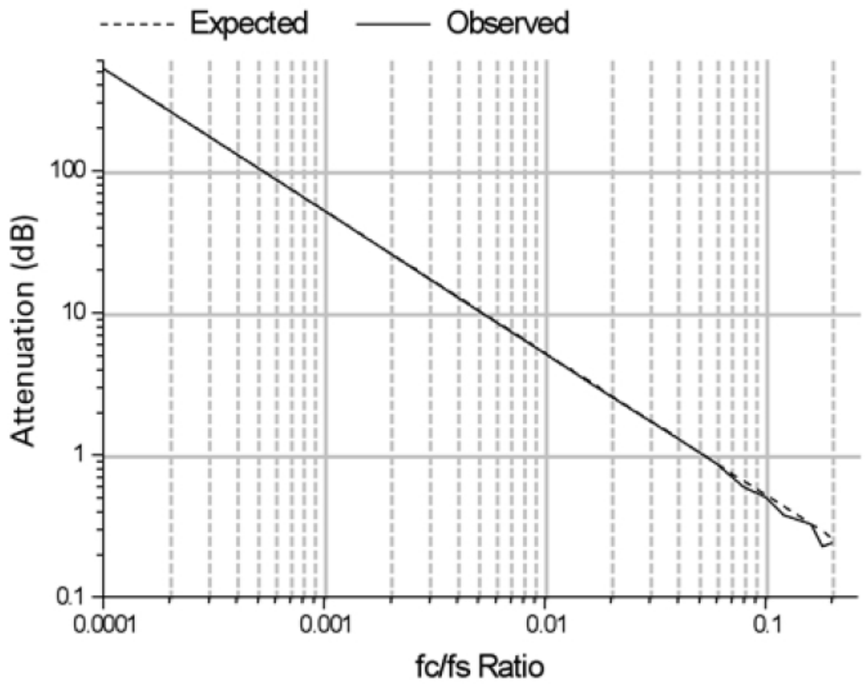


Figure 8-9: Chebyshev lowpass filter (8 pole) expected vs. observed rise time.

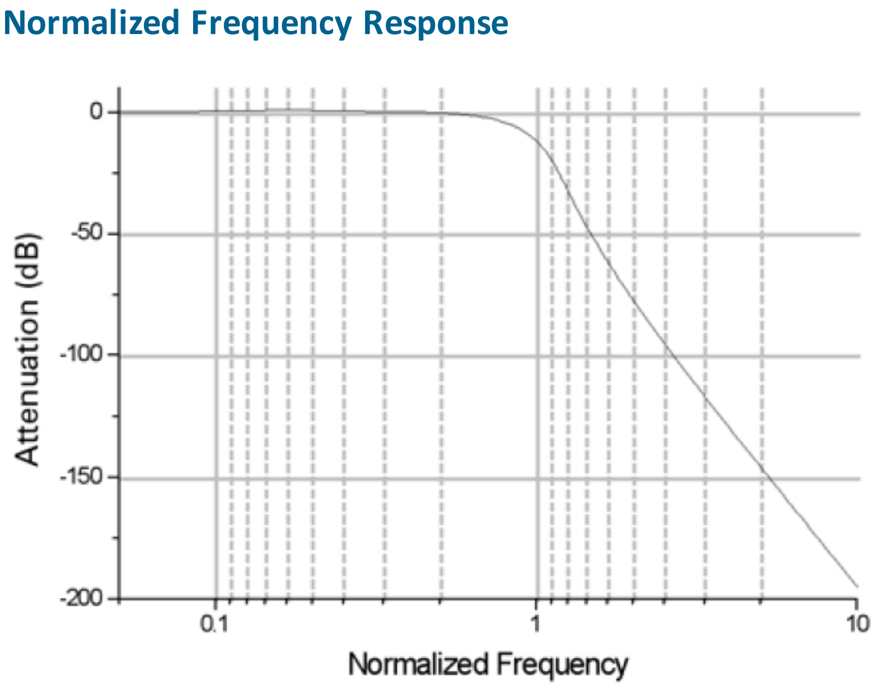


Figure 8-10: Chebyshev lowpass filter (8 pole) normalized frequency response.

Gaussian Lowpass Filter Specifications

- 10% to 90% step rise time: $0.3396/f_c$
- Maximum overshoot: 0%
- Attenuation: 90 dB at $f = 10\,f_c$

The Gaussian lowpass filter forms a weighted sum of the input values to form an output value according to the following recurrence formula:

$$y_i = \sum_{j=-n}^{n} a_j x_{i-j}$$

where a_j are the Gaussian coefficients that sum to unity. The algorithm for and properties of this filter are thoroughly described by D. Colquhoun and F.J. Sigworth (1995).

The Gaussian filter is particularly suited for filtering biological data for analysis in the time domain as it produces no overshoot or ringing and introduces no phase delay. The disadvantage is that it can be slow at high f_c/f_s ratios where the number of filter coefficients is large.

Expected vs. Observed Overshoot

100 mV step pulse, f_s = 10 kHz

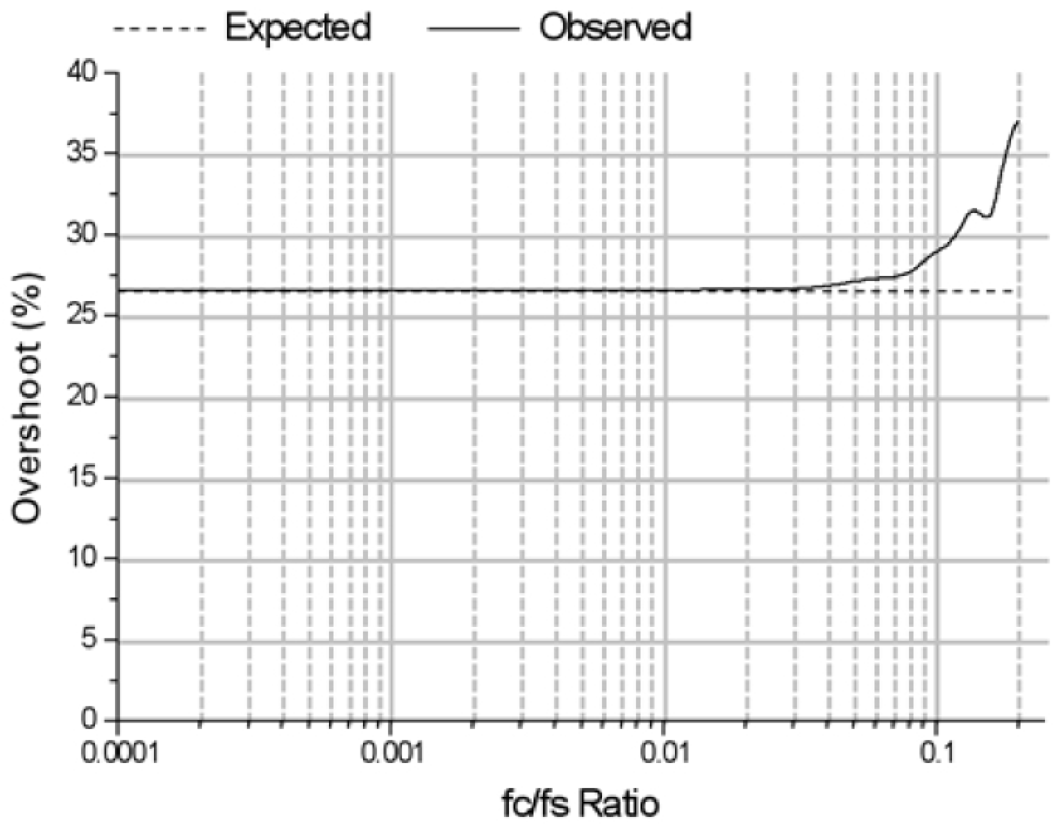


Figure 8-11: Chebyshev lowpass filter (8 pole) expected vs. observed overshoot.

Expected vs. Observed Rise Times

100 mV step pulse, f_s = 10 kHz

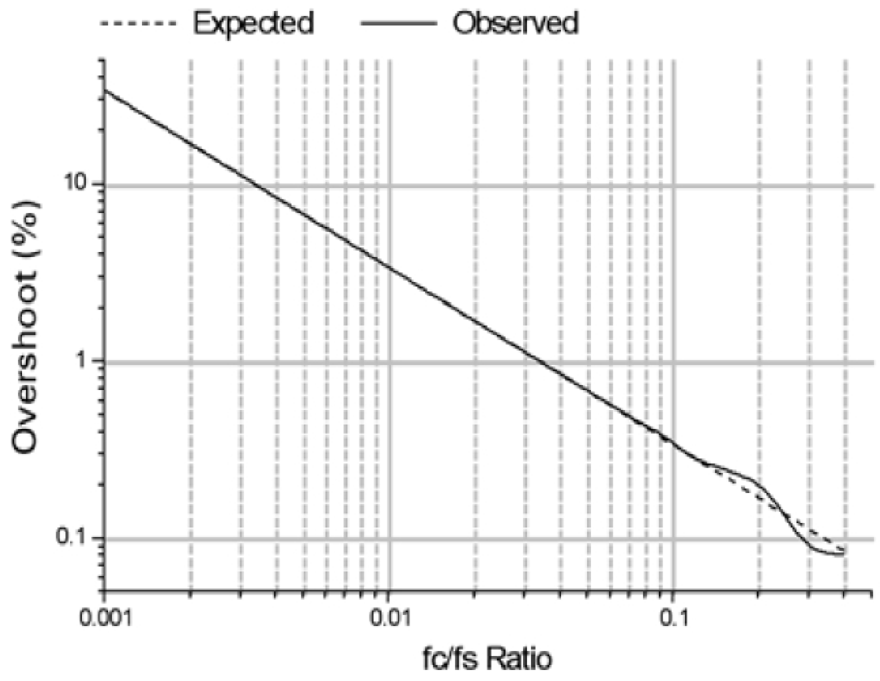


Figure 8-12: Gaussian lowpass filter expected vs. observed rise time

Normalized Frequency Response

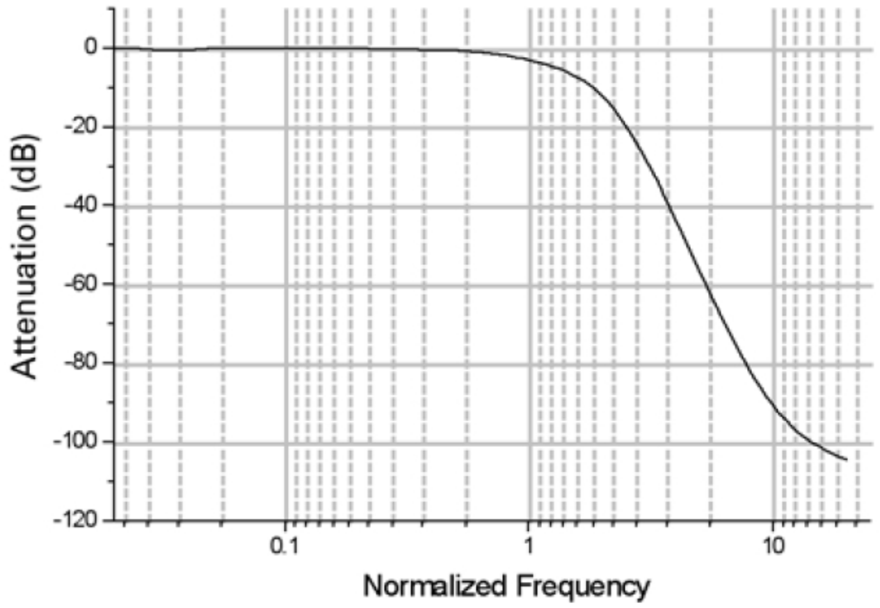


Figure 8-13: Gaussian lowpass filter normalized frequency response.

Notch Filter (2 Pole) Specifications

The number of poles of the notch filter is fixed at two. This filter has essentially zero gain (–inf dB) at its center frequency and about unity gain (0 dB) elsewhere. The notch filter has approximately zero phase shift except at its center frequency, at which the phase shift is undefined (because the gain is zero). In both respects (magnitude and phase) the resonator behaves like a “real” analog tuned circuit.

Figure 8-14 shows the frequency response of the notch filter with a 60 Hz center frequency with a 10 Hz –3 dB width.

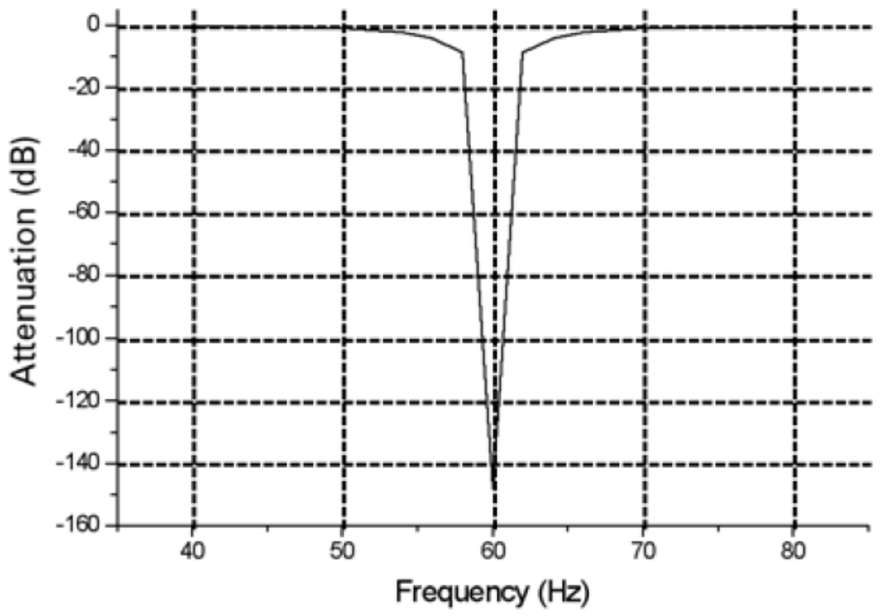


Figure 8-14: Notch filter (2 pole) frequency response (60 Hz center frequency, 10 Hz –3 dB width)

Settling Points

The –3 dB width has a significant influence on the number of points required for the notch filter to settle, the narrower the notch the greater the number of settling points. For example, for a 60 Hz sine wave sampled at a frequency of 1 kHz, applying a 60 Hz notch filter with a 1 Hz –3 dB width requires 2000 sampling points for the filter to reduce the amplitude to 0.1% of its original value (–60 dB). In contrast, a 60 Hz notch filter with a 10 Hz –3 dB width requires only 200 points.

The number of settling points also increases with increasing sampling frequency. For example, a notch filter with a –3 dB width of 10 Hz requires 2000 settling points for data sampled at 10 kHz compared to 200 for data sampled at 1 kHz.

The relationship between the number of settling points (P_s) and the sampling frequency (f_s) and –3 dB notch width (W_{-3dB}) is given by:

$$P_s = \frac{2f_s}{W_{-3dB}}$$

for attenuation of the center frequency by 60 dB.

RC Lowpass Filter (Single Pole) Specifications

- 10% to 90% step rise time: $0.3501/f_c$
- Maximum overshoot: 0%
- Attenuation: 20 dB at $f = 10\ f_c$

The RC lowpass filter function is equivalent to that of a simple first-order electrical filter made up of a single resistor and a single capacitor. The RC filter introduces a phase delay to the output, but the group delay is constant so that there is no ringing or overshoot.

The recurrence formula for this filter is:

$$Y(n) = X(n) + W + (Y(n-1) - X(n-1))$$

where *Y*(*n*) is the current output, *X*(*n*) is the current data point, *Y*(*n*–1) is the output for the previous point and:

$$W = e^{-dt/\tau} \qquad \text{where} \qquad \tau = 1/2\pi f$$

where *dt* is the sampling interval and f is the –3 dB cutoff frequency.

Expected vs. Observed Rise Times

100 mV step pulse, f_s = 10 kHz

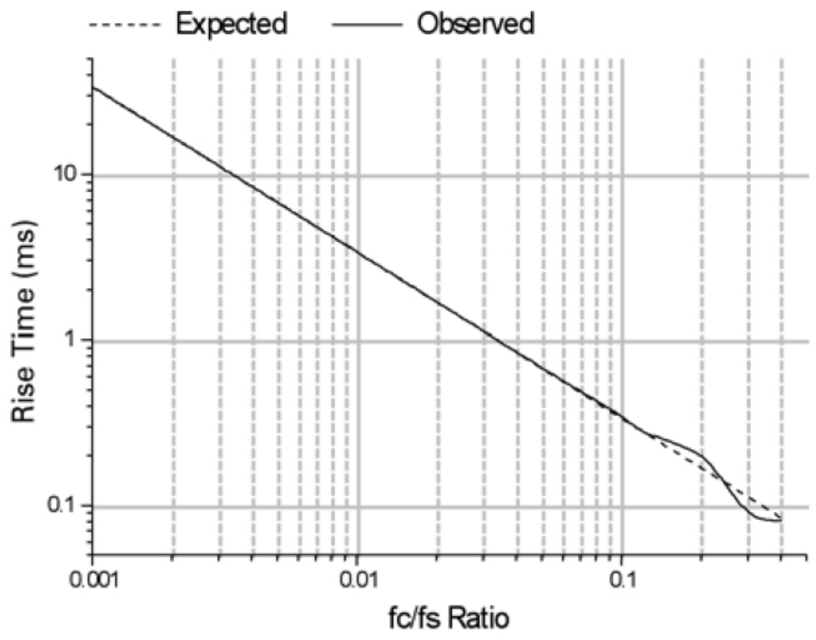


Figure 8-15: RC lowpass filter (single pole) expected vs. observed rise time.

Normalized Frequency Response

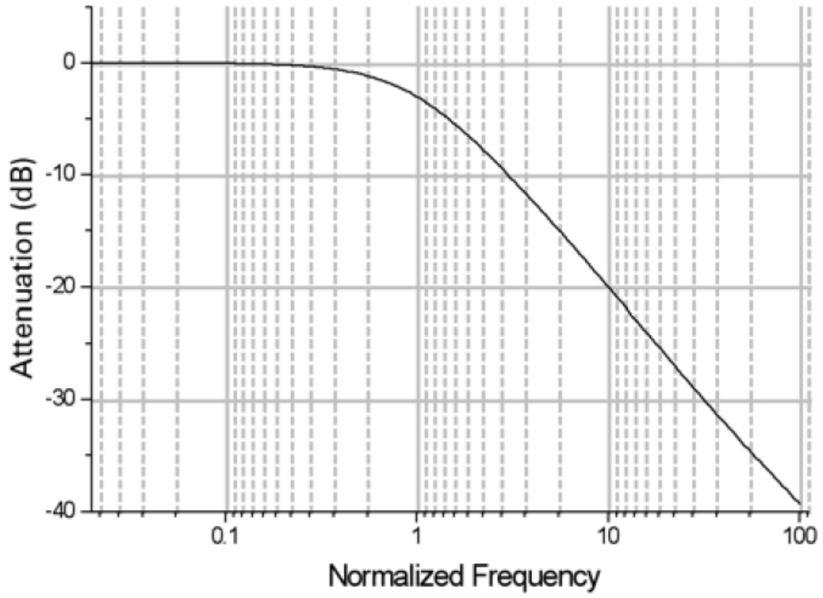


Figure 8-16: RC lowpass filter (single pole) normalized frequency response.

RC Lowpass Filter (8 Pole) Specifications

- 10% to 90% step rise time: $0.34/f_c$
- Maximum overshoot: 0%
- Attenuation: 80 dB at $f = 10\, f_c$

The 8-pole RC filter is a “multiple coincident pole” realization where the data points are filtered by applying 8 single pole RC sections in succession. The recurrence formula for this filter is, therefore, identical to that of the single pole RC filter except that the output from each previous pole is used as the input for the successive pole, where:

$$Y(n) = X(n_p) + W + (Y(n_p - 1) - X(n_p - 1))$$

or p = 1 to 8, where Y(n) is the output from the current pole, $X(n_p)$ is the filter output of the previous pole for the current point, $Y(n_p–1)$ is the output from the previous pole for the previous point and:

$$W = e^{-dt/\tau} \qquad \text{where} \qquad \tau = 1/2\pi(f/f_N)$$

where f_N is the normalized cutoff frequency. With the coincident pole design the cutoff frequency rises as the order of the filter is increased. The normalized cutoff frequency, f_N, is given by:

where *n* is the order (number of poles) of the filter. For an 8-pole filter this value is 3.32397. The specified cutoff frequency must be divided by the normalization factor in order to adjust the positions of the multiple poles. A consequence of this is that the maximum f_c/f_s ratio must be limited to the Nyquist frequency ($f_c/f_s = 0.5$) divided by the normalized cutoff frequency, or 0.5/3.32397 = 0.15.

Expected vs. Observed Rise Times

100 mV step pulse, f_s = 10 kHz

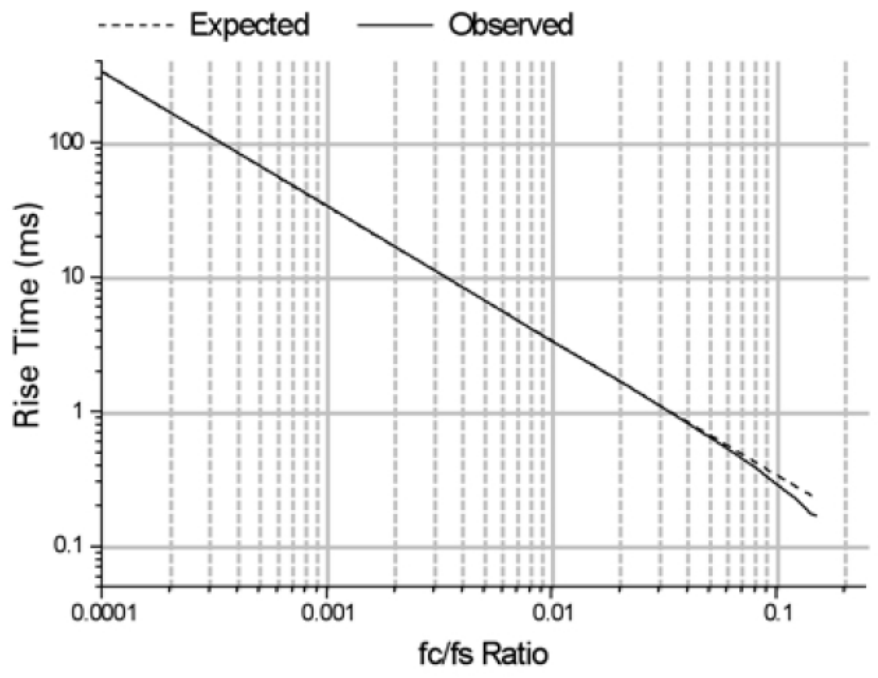


Figure 8-17: RC lowpass filter (8 pole) expected vs. observed rise time.

Normalized Frequency Response

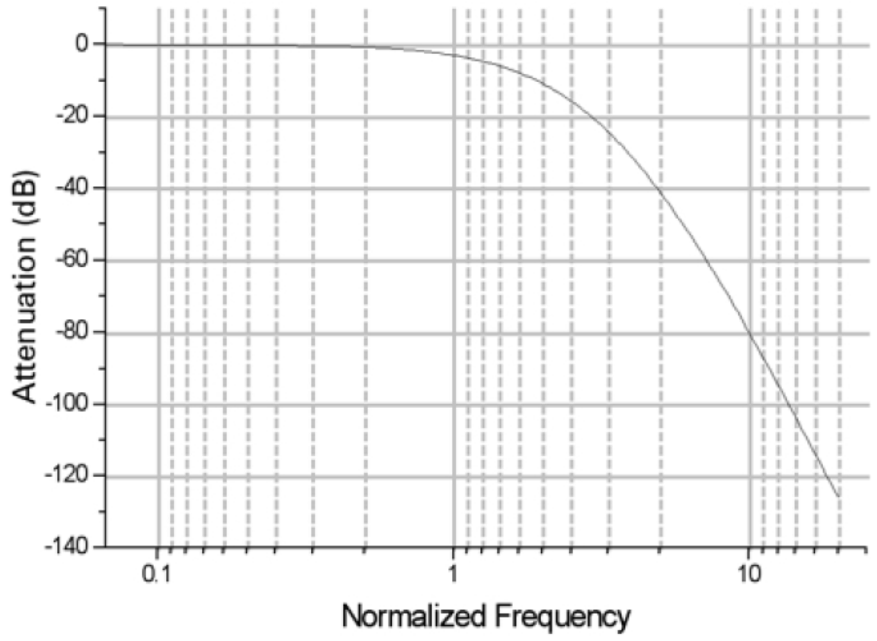


Figure 8-18: RC lowpass filter (8 pole) normalized frequency response.

RC Highpass Filter (Single Pole) Specifications

- Attenuation: 20 dB at $f = 0.1\ f_c$

The single-pole RC highpass filter operates by subtracting the lowpass response from the data. This is valid in this case because of the constant group delay characteristics of the single pole RC filter. The recurrence formula for this filter is, therefore:

$$Y(n) = X(n) - [X(n) + W + (Y(n-1) - X(n-1))]$$

where $Y(n)$ is the current output, $X(n)$ is the current data point, $Y(n–1)$ is the output for the previous point and:

$$W = e^{-dt/\tau} \qquad \text{where} \qquad \tau = 1/2\pi f$$

where *dt* is the sampling interval and ƒ is the –3 dB cutoff frequency.

Normalized Frequency Response

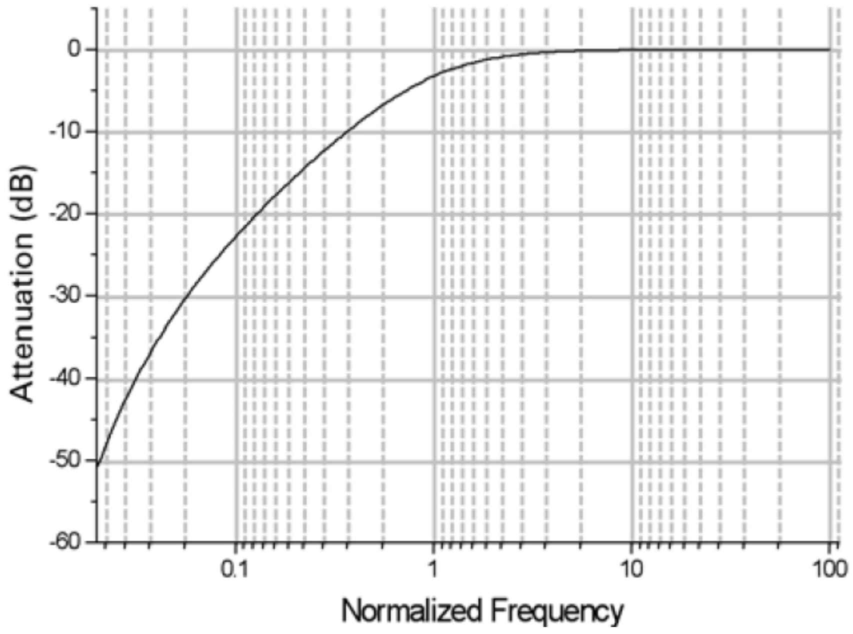


Figure 8-19: RC highpass filter (single pole) normalized frequency response.

Bessel Highpass Filter (8-Pole Analog) Specifications

- Attenuation: 114 dB at f = 0.1 f_c

The highpass Bessel filter has a sharper roll-off than the highpass RC filter. However, the Bessel filter deviates from ideal behavior in that it introduces ringing in the response to a step function, as shown below.

Highpass Bessel Filter Step Response

100 mV step pulse (dotted line), f_s = 10 kHz, f_c = 100 Hz

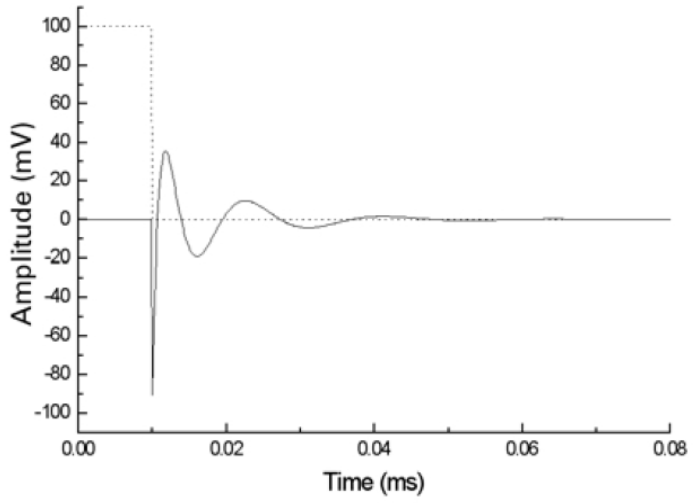


Figure 8-20: Highpass Bessel filter step response.

Normalized Frequency Response

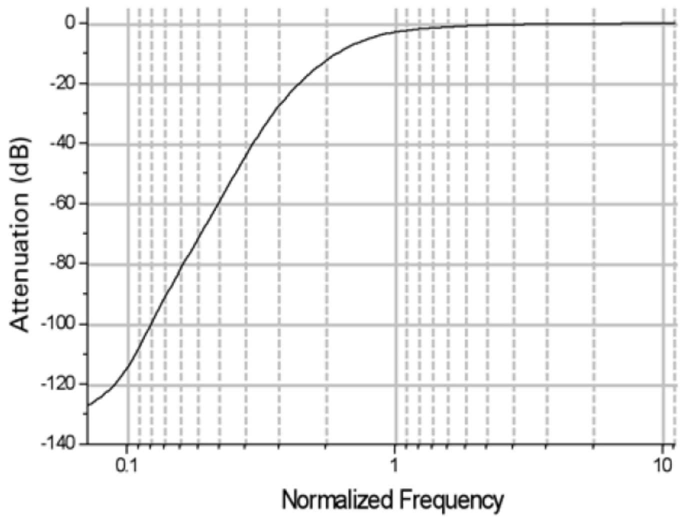


Figure 8-21: Highpass Bessel normalized frequency response.

The Electrical Interference Filter

The electrical interference (EI) Filter removes power line interference from an acquired data signal. This filter identifies and removes complex power line waveforms composed of multiple harmonics. The interference detection method is adaptive; that is, the filter always matches the actual line signal even if its properties (frequency, phase and shape) change during measurement.

The core component of the EI filter is the sine detector that discriminates sinusoids from other additive signal components. The sine detector generates a reference sine wave and adjusts its phase until it locks to a specific line-interference harmonic. The EI filter uses an array of sine detectors, one for each harmonic of the interference waveform.

The sine detector basically operates as a digital Phase Locked Loop (PLL) tuned to line frequency (50/60 Hz) or its multiple. The correlator (phase detector) detects phase difference between the reference and actual line harmonic. This phase difference is used as a feedback signal to adjust the reference phase until a perfect match is achieved.

Reference signals, each locked to a specific interference harmonic, are subtracted from the original signal, thus cancelling out the complete interference waveform.

Assumptions

1. The line interference and the data signal are statistically independent, for example, uncorrelated.
2. The line signal $x(n)$ is stationary across at least M of its periods. We can assume that the line signal does not significantly change when observed at any M of its periods.
3. The measured signal $s(n)$ is the sum of data signal $y(n)$ with scaled and phase shifted (delayed) version of the line signal: $s(n) = y(n) + Ax(n-d)$

Problem Statement

We want to identify line signal $x(n)$ incorporated inside $s(n)$. We assume that line signal is composed of a certain number of sinusoids with harmonic frequencies. According to assumption 3, we need to determine A and d for each harmonic.

Basic Theory

In order to detect the line signal we use the fact that data signal y(n) and line signal x(n) are uncorrelated. Practically, we say that cross correlation R_{xy} is equal to zero:

$$R_{xy}(d) = \frac{1}{M\Delta}\sum_{i=0}^{M\Delta} y(i)x(i-d) = 0$$

for each d, where Δ is the number of samples per period of the line signal. In order to keep the argument as simple as possible, we will use the term “correlation” R_{xy} meaning in most places actually “covariance”, implicitly assuming all signals to have zero DC component.

The correlation of the line signal x(n) and the measured signal s(n) should equal to:

$$R_{xs}(d) = R_{xy}(d) + R_{xx}(d)$$

where R_{xx} is the auto-correlation of the line signal. Since R_{xy} is equal to zero, we conclude that if we correlate the line signal and measured signal, we will obtain the auto-correlation of the line signal. The auto-correlation function $R_{xx}(d)$ is even and its maximum is at d = 0. Furthermore, since x(n) is periodic, R_{xx} is also periodic with the same period.

We use the above argument to determine both the delay d and the scale A of the line signal inside the measured signal (see assumption 3). The above discussion holds for our specific situation if we assume that x(n) is a sinusoidal reference signal.

Block Diagram

The adaptive line interference filter block diagram for the fundamental harmonic is shown in Figure 8-22. The same procedure is repeated for each harmonic by multiplying the frequency of the reference generator.

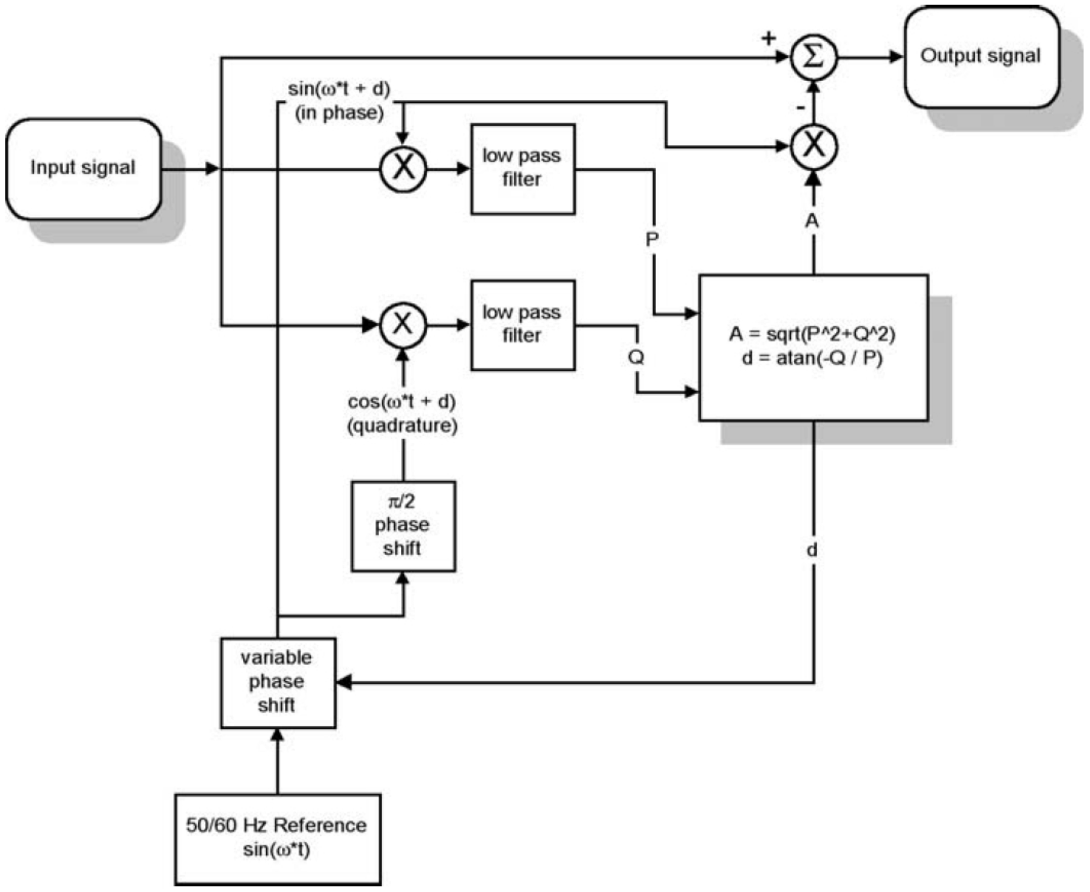


Figure 8-22: Adaptive line interference filter block diagram.

Implementation

The low pass filter is implemented as the simple average over M line periods. The average of the product of two signals, as in correlation definition, is equal to applying a low pass filter with box impulse response, or sin(x)/x transfer function.

The critical parameter for correlator performance is the averaging period M. The higher the value of M, the lower the bandwidth of the equivalent low pass filter. By keeping the bandwidth as low as possible, we reduce the correlation estimate error. Since we ultimately need to estimate the auto-correlation of the periodic signal x(n), the averaging period should be equal to a whole number of line signal periods M.

The EI filter generates a reference sine wave for each detected harmonic up to the maximum specified harmonic search number and subtracts reference waveforms from the signal, thereby cancelling line interference. Ideally, each reference sinusoid exactly matches the corresponding interference harmonic both in phase and amplitude, resulting in the perfect cancellation of line interference.

A practical EI filter never exactly detects phase and amplitude of interference harmonics. After subtraction of the generated reference sinusoids any discrepancy in amplitude and phase will result in artifact sinusoids, that is, components that were not present in the original signal. After filtering, artifactual components actually replace the original line harmonics. When harmonic detection is good, the total power of artifact components after filtering is much lower than line interference power in the original signal.

Harmonic detection errors come from noise and data signal components at line harmonic frequencies (multiples of 50/60 Hz). Generally, noise errors are less significant and can be successfully reduced by increasing the number of cycles to average. A more significant source of EI-filter errors are original data signal components that fall at or close to 50/ 60 Hz multiples (data signal leak).

Weak Harmonics

Weak line harmonics in the presence of noise cannot be accurately detected. In extreme cases EI-filtering artifact power for the single weak harmonic can be larger than the actual harmonic power. In such cases it might be better to reduce the harmonic search number in order to exclude weak harmonics. Also, increasing the number of cycles to average will always improve harmonic detection (if the noise is the main error source). However, excessively large number of cycles to average will negatively affect execution speed, tracking performance and start-up transient compensation.

Data Signal Components

Periodic

Any periodic components in the data signal with frequencies at or very close to 50/60 Hz multiples will leak through the EI filter harmonic detectors and will produce false line interference harmonics.

Figure 8-23 shows the result of filtering a pure (no noise and no line interference) square wave at 10 Hz with the harmonic search number set to 3 and reference frequency set to auto.

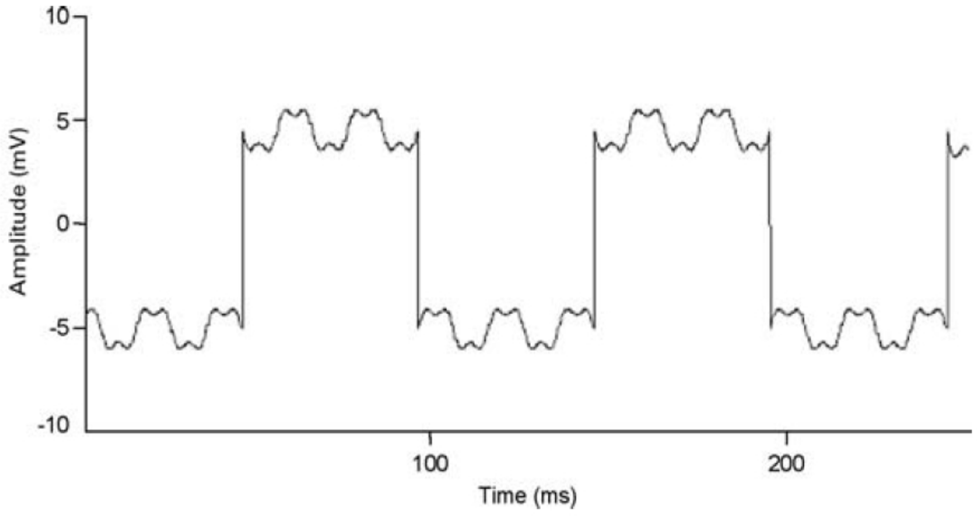


Figure 8-23: Results of filtering a square wave.

Since a 10 Hz square wave has strong components at multiples of 10 Hz, the EI filter locked on to the 5th (50 Hz), 10th (100 Hz) and 15th (150 Hz) harmonics, generating prominent artifacts.

Aperiodic

Strong and sharp pulses in the data signal may produce artifacts in the EI filtering process. Sharp pulses (spikes) have significant components at all frequencies including 50/60 Hz multiples. If the spike amplitude is two (or more) orders of magnitude larger than actual line interference amplitude, the EI filter will produce false line harmonics after the spike in the region whose size is equal to the number of cycles to average.

In the example in Figure 8-24, the EI filter was used at 10 harmonics with 50 cycles to average. Spike amplitude (not shown in full) was more than 200 mV. Notice the false line harmonics in the lower graph.

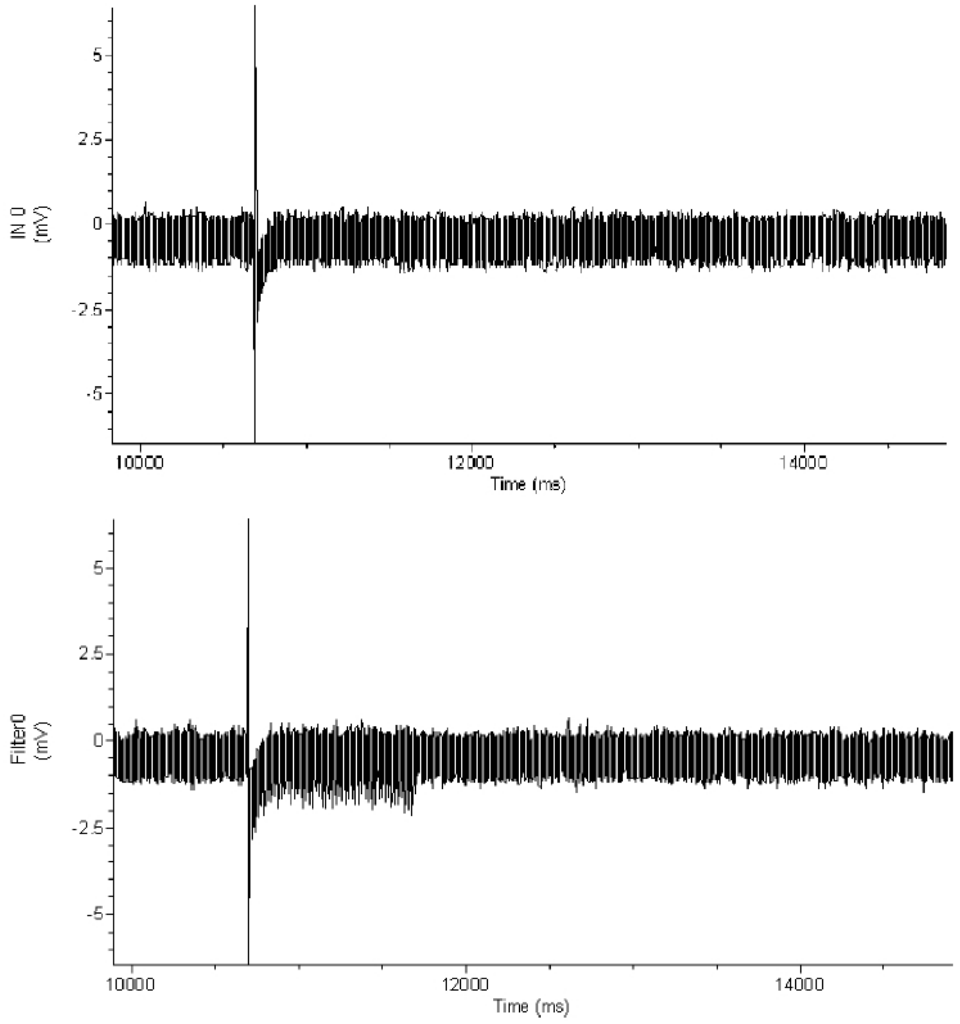


Figure 8-24: EI filtering showing introduction of false line harmonics.

Start-up Transients

Start-up transients are spurious, rapidly changing false harmonics at the very beginning of the filtered signal. When processing samples at the beginning of the signal file EI filter does not have enough information to accurately detect line harmonics. With every new processed sample the detection becomes better and better and the start-up transient becomes smaller and smaller. The filter reaches its steady state after a time equal to number of cycles to average.

The EI filter automatically compensates for start-up transients by turning off reference subtraction until it reaches its steady state. When it reaches steady state after the specified number of cycles to average the EI filter assumes that line interference parameters are accurately detected and filters the signal backwards using current reference parameters.

Potential Problems

The filter is too slow

When dealing with large datasets and high sampling rates the filter might be slow. If filtering is unacceptably slow try the following:

- Check if it is necessary to remove all harmonics specified in the harmonics field. Try removing only the first harmonic, and if the result is not satisfactory, increase the harmonic number and try again. The removal of the first three harmonics will often sufficiently reduce the interference.
- Decrease the value in the Cycles to average field. Often, smaller averaging lengths (time constants) do not significantly affect the output signal quality.

Interference is not fully removed

If the line interference is not fully removed try the following:

- If residual interference contains high frequencies then it is might be necessary to increase the value of the upper harmonic to be removed.
- If the fundamental line harmonic is still visible in the output signal then the number of cycles to average should be increased.

Cutoff Frequency Limitations

All digital filters have inherent limitations, and in some cases deviate significantly from their analog counterparts. The filters in pCLAMP Software have been restricted to an f_c/f_s ratio where the filter response is reasonably close to the theoretically expected response.

- The theoretical frequency range of digital filters is between 0 and the Nyquist frequency, which is one-half of the sampling frequency. This applies to all filters when filtering sampled data. However, the usable range of most software filters is considerably narrower than this theoretical range. The usable range depends on the nature of the filter (FIR or IIR) and the filter algorithm.
- The overshoot during a step response is a characteristic feature of Bessel, Butterworth and Chebyshev lowpass filters. For analog filters, the magnitude of the overshoot is constant over the full operating range. For digital IIR filters, however, the overshoot becomes increasingly larger as the ratio of f_c/f_s increases.
- The operating range of the Gaussian FIR filter is limited at the low end by a practical, rather than theoretical, limitation. Low ratios f_c/f_s result in the generation of a large number of filter coefficients. This creates two problems. The first is that smaller datasets cannot be accurately filtered because the filter length might be greater than the number of data points. The second is that the large number of coefficients is computationally inefficient. The number of Gaussian coefficients is inversely proportional to the f_c/f_s ratio where a lower cutoff frequency requires a greater number of coefficients for the filter realization.

The following table lists the numerical limitations for each filter type. The lower and upper cutoff frequencies are expressed as a factor times the sampling frequency (f_s). These limits are internally set by the filter algorithm and cannot be altered.

Table 8-1: Numerical limitations for each filter type.

Filter Type	Lower Cutoff Limit	Upper Cutoff Limit
Bessel (8-pole IIR)	10^{-4} x f_s	0.14 x fs
Boxcar (FIR)	See note 1.	n/a
Butterworth (8-pole IIR)	10^{-4} x f_s	0.2 x fs
Chebyshev (8-pole IIR)	10^{-4} x f_s	0.2 x fs
Electrical Interference	See note 2.	n/a
Gaussian (FIR)	10^{-4} x f_s. See note 3.	0.5 x fs
Notch (2-pole IIR)	10^{-3} x f_s	0.3 x fs
RC (single-pole IIR)	10^{-4} x f_s	0.5 x fs
RC (8-pole IIR)	10^{-4} x f_s	0.15 x fs. See note 4

Note 1: The boxcar filter requires that the number of smoothing points be specified. This must be an odd number in order for the filter to be symmetrical. The minimum number of smoothing points is 3. The maximum number of smoothing points is 99. However, the maximum number of smoothing points is also limited by the number of data points, n, such that the filter width is at least n/2. So if there are 50 data points the maximum number of smoothing points is 50/2 = 25. If this formula generates an even number then the maximum number of smoothing points will be one less. For example, if there are 52 data points, then the maximum number of smoothing points will be 52/2 – 1 = 25.

Note 2: The electrical interference filter does not have a lower or upper cutoff frequency limit as it is preset to remove either 50 Hz or 60 Hz interference and the associated harmonics (see The Electrical Interference Filter on page 175). However, there is a data point minimum, as this filter requires a specific number of points to reach steady state. This minimum is given by:

minimum points = samples per period x cycles to average

where the samples per period is the sampling frequency divided by the reference frequency (50 Hz or 60 Hz) and cycles to average is the number of cycles of the reference frequency which are averaged in the response. For example, for a sampling rate of 1 kHz, a reference frequency of 60 Hz and 20 cycles to average the minimum number of data points required is 1000/60 x 20 = 334 data points.

Note 3: The Gaussian filter width (see Finite vs. Infinite Impulse Response Filters on page 155) depends on the fc/fs ratio; the lower this ratio the greater the number of Gaussian coefficients (see Gaussian Lowpass Filter Specifications on page 165). In view of this, two criteria are used to limit the lower cutoff frequency.

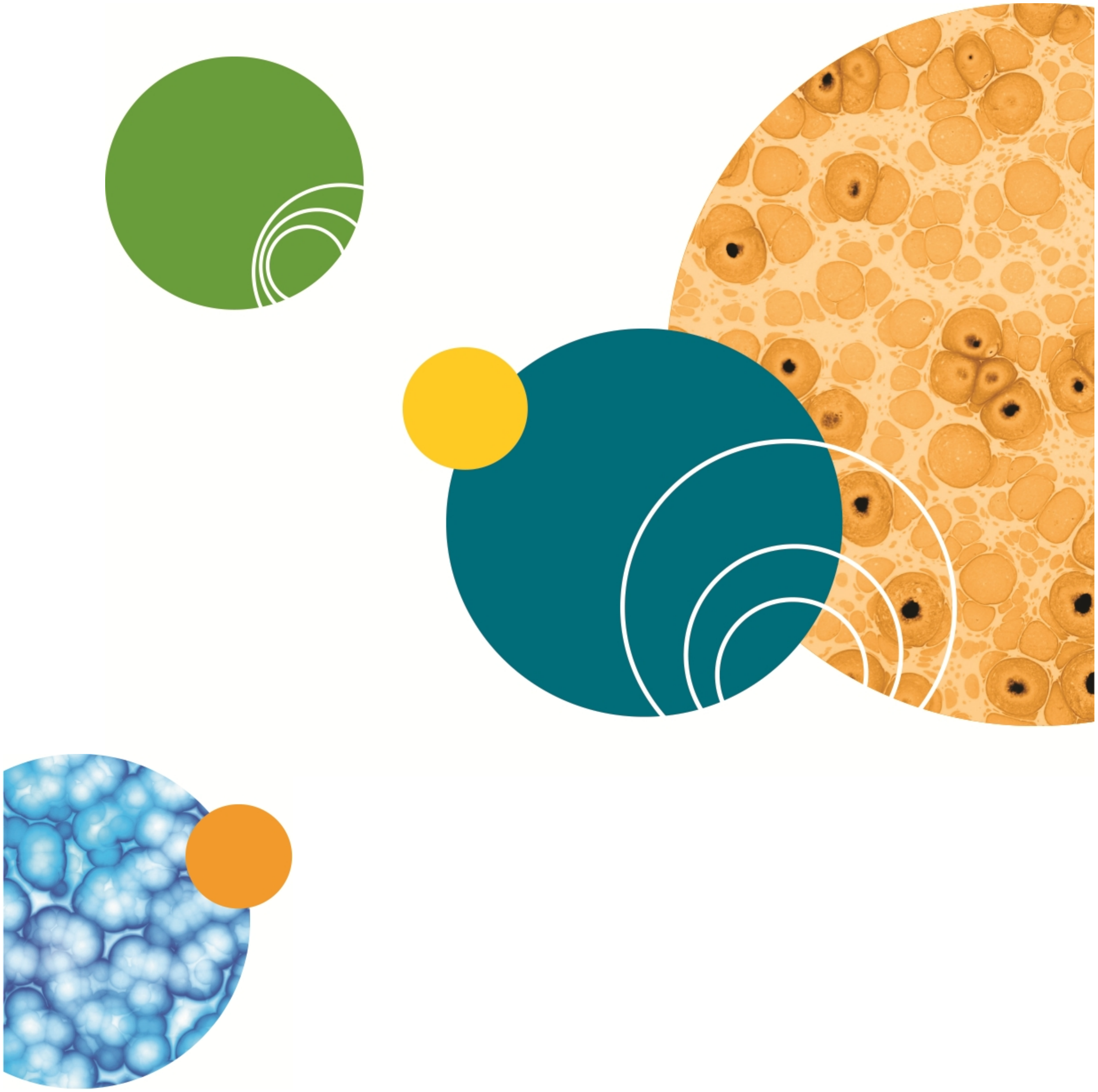
The first is that there must be enough data points to accommodate at least two Gaussian filter widths. That is, the minimum corner frequency will correspond to a filter width that is less than or equal to one-half the number of available data points.

The second is that the maximum number of Gaussian coefficients is limited to approximately 3500. This limit, which corresponds to an f_c/f_s ratio of about 3 x10^{-4}, is not exact because the automatically computed minimum corner ratio is generally rounded up. Therefore, the minimum corner ratio might correspond to a number of coefficients that is somewhat more or less than the 3500 limit.

Note 4: The 8-pole RC filter is a “multiple coincident pole” design where the –3 dB cutoff frequency rises with each pole by an amount given by:

$$f_N = 1/\sqrt{2^{1/n} - 1}$$

where f_N is the normalized cutoff frequency and n is the number of poles. Therefore, for an 8-coincident-pole filter the normalized cutoff frequency is actually 3.32397 times the specified cutoff frequency (see RC Lowpass Filter (8 Pole) Specifications on page 171). Consequently, the maximum f_c/f_s ratio must be limited to the Nyquist frequency ($f_c/f_s = 0.5$) divided by the normalized cutoff frequency, or 0.5/3.32397 = 0.15.



Chapter 9: Clampfit Software Analysis

9

Digital spectral analysis involves the decomposition of a signal into its frequency components. The purpose of such analysis is to reveal information that is not apparent in the time-domain representation of the signal. To this end the Fast Fourier Transform (FFT) is generally used because of its speed. The Fourier transform is based on the concept, first advanced by Jean Baptiste Joseph, Baron de Fourier, that nonperiodic signals can be considered as the integral of sinusoids that are not harmonically related.

The FFT samples in the frequency domain, just as a digital signal is sampled in the time domain. A signal processed by the FFT yields a set of spectral coefficients that can be regarded as samples of the underlying continuous spectral function. Although long transforms might look like a continuous spectrum, in fact they consist of discrete coefficients that define the “line spectrum” of the signal. The line spectrum indicates the “amount” of the various frequencies that are contained in the signal.

The Fourier Series

For a periodic digital signal x, the coefficients of the spectral distribution can be represented by the Fourier Series, which is defined by:

$$a_k = \frac{1}{N}\sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N}$$

where, for samples x = x_0 to x_{N-1}, ai is the kth spectral component, or harmonic, j is a complex number and N is the number of sample values in each period of the signal. The real and imaginary parts of the coefficients generated by this function can be expressed separately by writing the exponential as:

$$\cos(2\pi kn/N) - j\sin(2\pi kn/N)$$

If x[n] is a real function of n, the coefficient values except for a_0 are symmetrical. The real parts of coefficient a_0 are unique, but the real parts of coefficients a_1 and a_{N-1} are identical, as will the real parts of a_2 and a_{N-2}, and so forth. Likewise the imaginary parts also follow this pattern except that the sign changes so that $a_1 = -a_{N-1}$, and so forth. The imaginary part of coefficient a_0 (the zero-frequency coefficient) is always zero. This symmetry extends to coefficients outside the range 0 to N–1 in both the positive and negative direction. For example, coefficients a_N to a_{2N-1} are identical to coefficients a_0 to a_{N-1}. Thus periodic digital signals have spectra that repeat indefinitely along the frequency axis. This is also true for nonperiodic signals that are more likely to be encountered in the real world (see next section, The Fourier Transform on page 186).

A periodic digital signal with N samples per period can, therefore, be completely specified by in the frequency domain by a set of N harmonics. In fact, if x(n) is real then only half this number is required because of the inherent symmetry in the coefficient values.

Fourier analysis is concerned with determining the amount of each frequency that is present. This appears ambiguous for a digital signal since a whole set of spectral representations is possible. However, this is not a problem because spectra that are produced by digital Fourier analysis repeat indefinitely along the frequency axis, so only the first of these repetitions is sufficient to define the frequency content of the underlying signal, so long as the Sampling Theorem is obeyed (see The Sampling Theorem in pCLAMP Software on page 25).

If the “transform length” N contains an integral number of cycles then the natural periodicity of the signal is retained and each component of the spectrum occupies a definite harmonic frequency. This is because the natural periodicity of each component is preserved during spectral analysis. However, if x[n] contains sinusoids that do not display an exact number of periods between 0 and N then the resultant spectrum displays discontinuities when repeated end on end. Because of these discontinuities the spectrum displays a spreading of energy, or “spectral leakage”. In the real world, signals rarely if ever are so cooperative as to arrange themselves in integral numbers of cycles over a given transform length. Thus spectral leakage will, in general, always be a problem. Fortunately, there are procedures that can minimize such leakage (see Windowing on page 188).

The Fourier Transform

Most practical digital signals are not periodic. Even naturally occurring repetitive signals display some degree of variation in frequency or amplitude. Such signals are evaluated using the Fourier Transform, which is closely related to the Fourier Series defined by Equation 1.

Equation 1 applies strictly to a periodic signal with a period N. However, we can modify this equation to apply to nonperiodic signals. Starting with a periodic signal of length N, we select an arbitrary number of points in the center of the signal. We then “stretch” the signal by adding zeros to either side of the selected points. We can continue adding zeros until $N \rightarrow \infty$. At this point the neighboring repetitions have moved to $\pm\infty$ and we are left with a nonperiodic signal.

If the signal is stretched in this way then the coefficients, ak, must become smaller because of the 1/N term in Equation 1. They must also come closer together in frequency because N also appears in the denominator of the exponential. Therefore, in the limit as $N \rightarrow \infty$, the various harmonics become spaced extremely closely and attain vanishingly small amplitudes. We can think of this in terms of a continuous, rather than a discrete, spectral distribution.

As $N \rightarrow \infty$ the product Na_k remains finite although each spectral coefficient, a_k, becomes vanishingly small. We can write Na_k as X. Also, the term 2πk/N can be thought of as a continuous frequency variable that can be written as Ω. Thus, Equation 1 becomes:

$$x = Na_k = \sum_{n=0}^{N-1} x[n]e^{-j\Omega n}$$

Since x[n] is now nonperiodic the limits of summation should be changed. Also, since x[n] exists for both positive and negative values of n we sum between n = ±∞. We also write X as X(Ω) to make it clear that X is a function of the frequency Ω. Therefore, Equation 2 becomes:

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$$

which defines the Fourier Transform of the nonperiodic signal x[n].

Just as for periodic signals, the spectrum of a nonperiodic digital signal is always repetitive.

The Fast Fourier Transform

pCLAMP Software uses the Fast Fourier Transform (FFT) for spectral decomposition. Computing the Fourier Transform is a computationally intensive task that requires a great deal of redundant calculation. The Fast Fourier Transform is an algorithm that reduces these redundant calculations, in part by decomposing the Discrete Fourier Transform (DFT) into a number of successively shorter, interleaved DFTs.

For example, if we have signal x[n] with N points where N is a power of 2 then we can separate x[n] into two short subsequences, each of length N/2. The first subsequence contains the even-numbered points and the second contains odd-numbered points. Successive decomposition of an N/2-point subsequence can be broken down until each sequence is a simple 2-point DFT. This process is known as “decimation in time”. Computing the Fourier Transform at this point is still not trivial and requires factors to account for the displacement of the points in time.

For a more complete discussion of the FFT see Lynn and Fuerst 1994.

pCLAMP Software uses a decimation in time FFT algorithm that requires N to be an integral power of 2.

The Power Spectrum

The magnitude of the spectral components, S, derived by the Fourier transform is given by the sum of the squares of the real and imaginary parts of the coefficients, where:

$$S_i = a_{i(real)}{}^2 + a_{i(imag)}{}^2$$

Since this is the power averaged over N samples (where N is the transform (window) length) and since only N/2 of the components are unique, the power at each sampling point (Pi) is scaled such that, for a given sampling frequency, f:

$$P_i = S_i\frac{2N}{f}$$

The power spectrum is further scaled by the window factor, ϖ, where for a given window function, fw, the scale factor for a window of length N is given by:

$$\varpi = \frac{\sum_{n=1}^{N} (f_W)^2}{N}$$

and the power, expressed in units2, is given by:

$$P_i = S_i \frac{2N}{f\varpi}$$

Finally, the total root mean square (RMS) value is computed by taking the square root of the integral of the power spectrum, such that:

$$RMS = \sqrt{\sum_{n=1}^{N} P_i f/N}$$

where f/N is value of the frequency bin width in Hz.

Limitations

The limitations imposed on power spectral analysis are:

- The data must be sampled at evenly spaced intervals.
- The transform length must be an integral power of 2.
- The frequency spectrum will range from 0 to one-half the sampling frequency.

Windowing

If all frequency components in a signal have an integral number of periods in the transform length then each component occupies a definite harmonic frequency, corresponding to a single spectral coefficient. However, real-world signals are rarely so cooperative, containing a wide mixture of frequencies with few exact harmonics. Thus spectral leakage (see The Fourier Series on page 185) is almost always expected. In order to reduce spectral leakage it is common practice to “taper” the original signal before transformation to reduce edge discontinuities. This is done by multiplying the signal with a “windowing function”.

The ideal window has a narrow main lobe to prevent local spectral leakage and very low side lobe levels to prevent more distant spreading. As it turns out, these two requirements are in conflict. Consequently all windowing types allow some spectral leakage.

The simplest window is the rectangular (or “do-nothing”) window that does not alter the signal. This window has the narrowest possible main lobe but very large side lobes, which permit a substantial amount of spectral leakage with nonharmonics (there is no leakage of exact harmonic components). All other windowing functions taper the data to a greater or lesser degree.

pCLAMP Software offers several window functions. The choice of a windowing type depends on the nature of the signal and the type of information that is to be extracted. To assist in this decision the window function is displayed in the dialog box along with the time domain and frequency domain responses for each window type.

Segment Overlapping

Data can be divided into segments of equal length either for sequential processing to reveal a trend in the power spectra with time or to average the power spectra for all segments in order to reduce the variance. If the spectra are to be averaged then segmenting can either be overlapped or the segments can be generated without overlapping. Both options are available in pCLAMP Software.

The "segments" are equal to the window length, which in turn is equal to the FFT transform length.

If the segments are not overlapped then the reduction in the spectral variance is about M/2 where M is the number of segments. If the segments are overlapped by 50% then the reduction in spectral variance is about 9M/11, which is a significant improvement over the non-overlapped case (Press et al. 1992). For example, if 10 segments are averaged without overlapping the spectral variance is reduced by a factor of about 5. However, with 50% overlapping the variance is reduced by a factor of about 8.2.

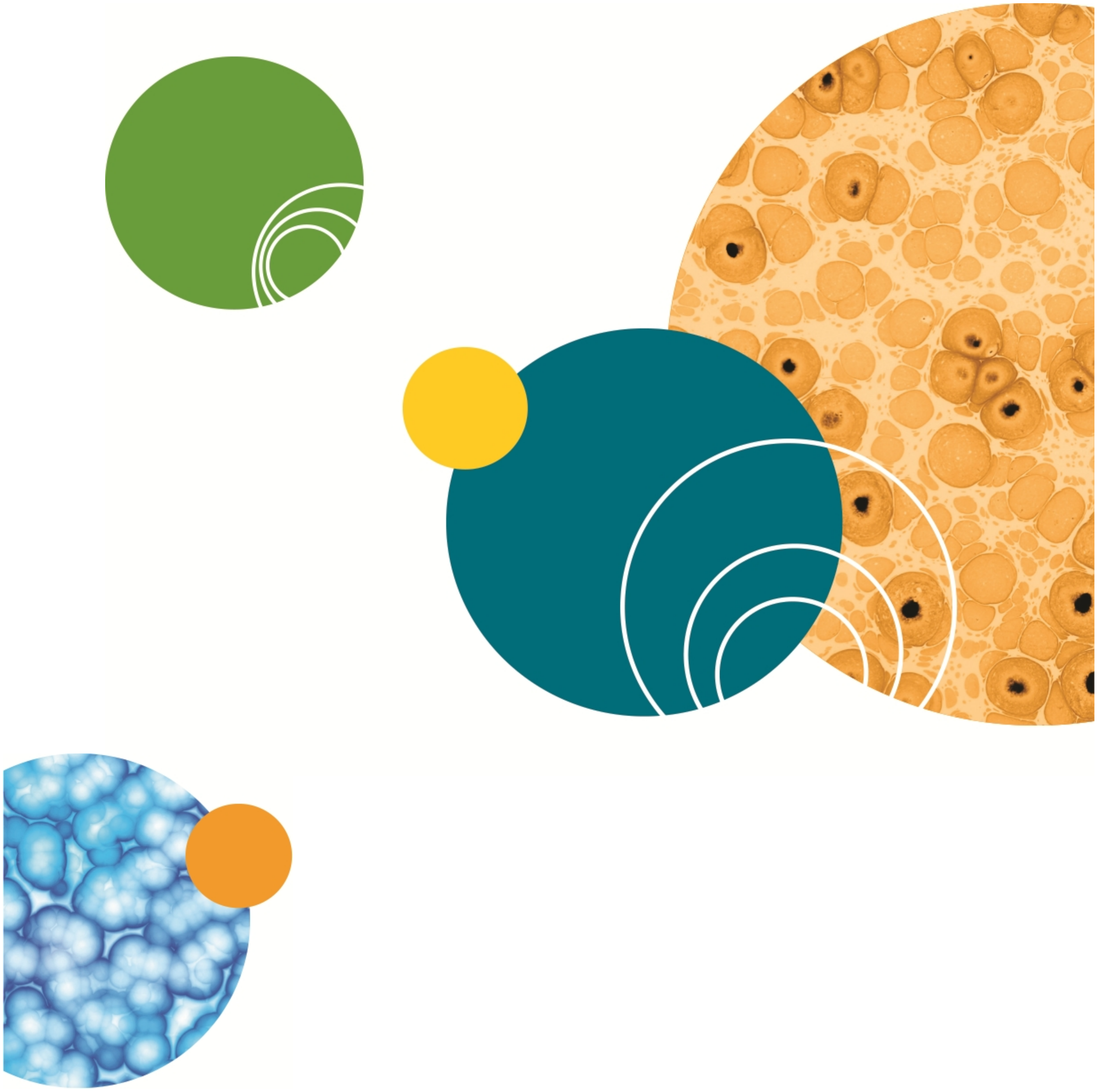
Transform Length vs. Display Resolution

The sampling theorem states that the highest frequency that can be contained in a digital record is equal to one-half the sampling frequency (see The Sampling Theorem in pCLAMP Software on page 25). Therefore, the highest frequency component that can be resolved by Fourier analysis is also only one-half the sampling frequency. Moreover, the symmetry inherent in the spectral coefficients means that only one-half of the coefficients are required to completely define the frequency spectrum of the digital signal. These issues are reflected in the display of the power spectrum in pCLAMP Software where the scale ranges from 0 to one-half the sampling frequency. A two-sided display would simply show a symmetrical (mirror image) distribution of spectral lines.

The transform (window) length relative to the sampling frequency determines the resolution of the power spectrum. In view of the restriction imposed by the frequency resolution, the frequency scale (X axis) for a one-sided display ranges from 0 to $f_s/2$, where f_s is the sampling frequency. The resolution is dependent on the transform length L. For example, if f_s is 10 kHz and L is 512 then the frequency scale, ranging from 0 to 5000 Hz, is divided into 256 bins (only one-half the transform length is used because of the transform symmetry), each having a width of 19.53 Hz (5000 Hz/256). If, on the other hand, L is 56 then the frequency scale is divided into 28 bins, each with a width of 178.57 Hz.

The bin width, W, in the spectral display is, therefore, given by:

$$W = f_s/L$$



Chapter 10: pCLAMP Analysis

10

Membrane Test

The Membrane Test in Clampex Software (**Tools > Membrane Test**) generates a definable voltage pulse and reads off a range of measurements from the current response:

- Membrane capacitance (Cm)
- Membrane resistance (Rm)
- Access resistance (Ra)
- Time constant (Tau)
- Holding current (Hold)

Membrane Test applies a continuous square wave voltage command and the current response is measured. Voltage-clamp mode is assumed.

Command Pulse

Each pulse period uses 500 samples, corresponding to 250 samples per step with a 50% duty cycle. The command pulses are measured relative to the holding level. The pulse height is expressed as peak-to-peak (p-p). Both edges of the pulse (for example both capacitive transients) are used for calculations.

A fast logarithmic exponential fit is performed on each transient or each averaged transient using a look-up table for the log transforms. The fit is performed between the 20% to 80% ordinates, or other ordinates, according to the settings in the **Proportion of Peak to Fit** section in the **Configure > Membrane Test Setup** dialog. The fit is displayed on the raw or averaged data as a superimposed red line.

Calculations

The transient portion of the current response is fit by a single exponential. From this exponential and the area under the curve the following parameters can be determined:

- The steady-state current (I_{ss} or HOLD)
- The time constant of the transient (τ or Tau) and the peak response
- The electrode access resistance (R_a)
- The membrane resistance (R_m)
- The membrane capacitance (C_m).

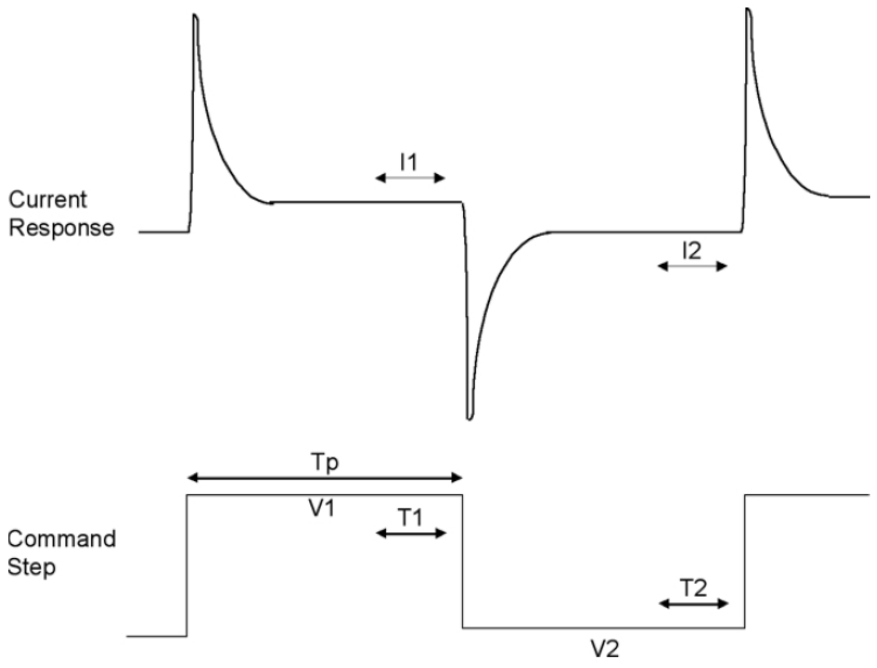


Figure 10-1: Regions of current response used in Membrane Test calculations.

The average current (I_1) is measured during period T1, which is 20% of the duration of T_p. The average current (I_2) of the second pulse in the pair is measured during period T2, and is the baseline for the first pulse. The average current (I_1) of the first pulse in the pair is the baseline for the second pulse.

When calculating the charge under the transient, the settling time of the membrane voltage step is corrected by adding ΔI x τ (Q_2; see below) to the integral, where $\Delta I = I_1 - I_2$. The steady-state current (I_{ss}) is calculated as the average of I_1 and I_2 [= $(I_1 + I_2)/2$] (see Figure 10-2).

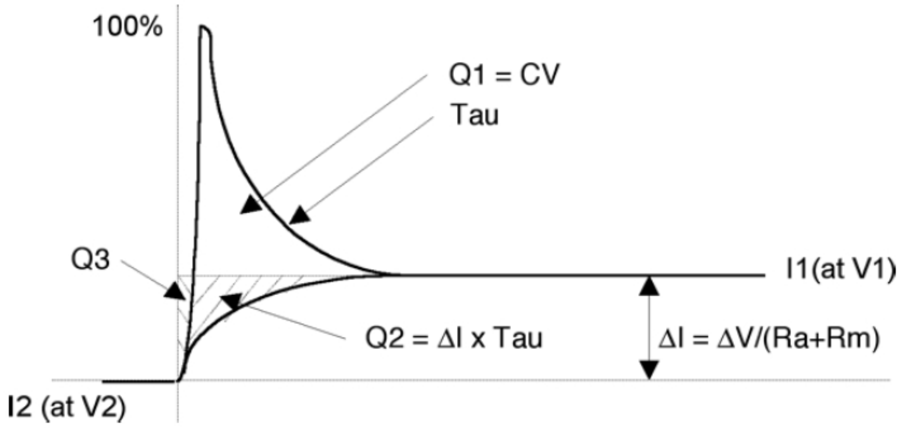


Figure 10-2: Derivation of Membrane Test results.

The total charge (Q_t) under the transient is the sum of the charge (Q_1) in the transient above the steady-state response plus the correction factor (Q_2). The small error charge (Q_3) introduced in the calculation of Q_2 is ignored. A logarithmic fit is used to find the time constant (τ) of the decay. Q_1 is found by integrating the charge above I_1. When integrating to find Q_1, I_1 is subtracted from It before integrating.

$$Q_2 = \Delta I \times \tau$$

$$Q_t = Q_1 + Q_2$$

C_m is derived from $Q_t = C_m \times \Delta V_m$

where ΔV_m is the amplitude of the voltage change across the capacitor, for example the change in the membrane potential. In the steady-state, the relation between ΔV_m and ΔV is:

$$\Delta V_m = \Delta V \times R_m / R_t = \Delta V \times R_m / (R_a + R_m)$$

where R_t, R_m and R_a are the total, membrane and access resistances, respectively. Substituting for ΔV_m from (2) into (1) derives:

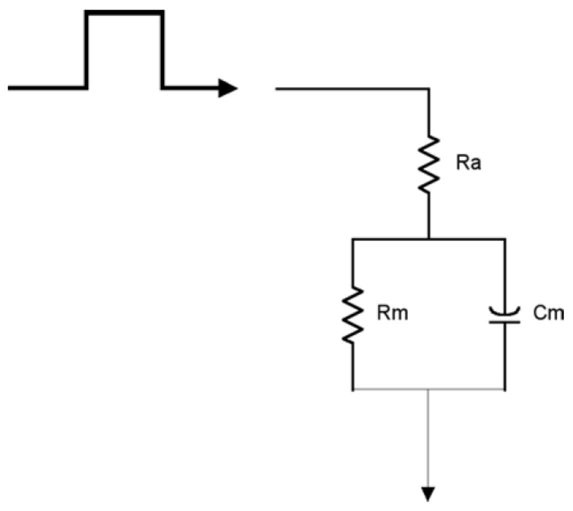


Figure 10-3: Idealized cell circuit.

The circuit elements R_a, R_m and C_m are readily derived as follows. Substituting (3) for Cm in the definition of the time constant = R_a x C_m:

$$\tau = Q_t \times R_a / \Delta V$$

hence:

$$R_a = \tau \times \Delta V / Q_t$$

which provides access resistance directly as a function of measured variables. The total resistance is calculated from the steady-state response:

$$R_t = \Delta V / \Delta I = R_a + R_m$$

from which one obtains:

$$R_m = R_t - R_a$$

C_m is then obtained from (3):

$$C_m = Q_t \times R_t / \Delta V \times R_m$$

When calculating the response for the second transient (the downward pulse), the same current measurements as the upward pulse (I_2 <–> I_1) are used and the second pulse is inverted in order to use the same calculation code.

[We thank Dr. Fernando Garcia-Diaz of the Boston University School of Medicine for his suggestions for these calculations.]

Template Matching

In event detection template matching in pCLAMP Software, (**Event Detection > Template Search**) the template is slid along the data trace one point at a time and scaled and offset to optimally fit the data at each point. Optimization of fit is found by minimizing the sum of squared errors (SSE) between the fitted template and the data. A detection criterion value is calculated at each position, by dividing the template scaling factor by a measure of the goodness of fit (derived from the SSE). You are able to set a threshold value for this criterion in the **Template Search** dialog, “Template match threshold”. All sections of the trace with a detection criterion greater than or equal to this value are automatically offered as candidate events.

When the template is aligned with an event the detection criterion is closely related to the signal-to-noise ratio for the event. Since background noise rarely exceeds four times the standard deviation of the noise, an (absolute) detection criterion value greater than four is unlikely unless there is an underlying event. For most data, then, four provides a close to optimum setting for the template match threshold value. Settings less than this run the risk of making false positives, while settings greater than this may miss genuine events.

For further information see Clements and Bekkers, 1997.

Single-Channel Event Amplitudes

Filtering data reduces the amplitude of rapidly occurring events (rapid with respect to the filter). In particular, short single-channel events have an attenuated amplitude. In addition, the points at the ends of long single-channel events have a smaller amplitude than in the absence of the filter.

When pCLAMP Software determines the amplitude of a single-channel event, it takes the average of all data points not affected by the filter. Data points within two time constants of the ends of each event are not included in the average. The time constant used takes into account the combination of digital filtering applied in pCLAMP Software prior to the single-channel search being run, and the acquisition filters, as follows.

$$n_f = \frac{1}{2\pi\Delta t}\sqrt{\frac{1}{f_{Inst}^2} + \frac{1}{f_{CyberAmp}^2} + \frac{1}{f_{Postacq}^2}}$$

where n_f is the settling time in sample points, t is the sampling period, f_{Inst}, and $f_{CyberAmp}$ are the –3 dB cutoff frequencies of the Instrument (or external) and CyberAmp filters used during acquisition, and $f_{Postacq}$ is the –3 dB cutoff frequency of pCLAMP Software's digital filter.

If a single-channel event is so short that all of its points are affected by the filter, no averaging is performed: the midpoint of the channel event is taken to be its amplitude, and "B" (for "brief") is written into the Results sheet State column for the event.

Note: Failing to specify correctly the instrument filter used (if any) when acquiring data invalidates the calculation of single-channel event amplitudes by pCLAMP Software.

Level Updating in Single-Channel Searches

To accommodate baseline drift in single-channel detection, the single-channel search dialog allows you to specify automatic level updating, applied as a search proceeds. You can opt to update the baseline only, without altering the relative amplitudes of the other levels, or update all levels independently.

When you select this option you must set the amount that new events contribute to their level's running amplitude. This weighting is entered as a percentage. With the default level contribution of 10%, for example, a new event contributes 10% of its amplitude, and the previous value, 90%, to the new level. For example, with a previous value of 10.0 pA, a new event with an amplitude 10.5 pA, and 10% contribution, the new amplitude for the given level is:

$$(0.9 \times 10.0\ pA) + (0.1 \times 10.5\ pA) = 10.05\ pA$$

The algorithm also includes a means to weight short events less, so that rapid events influence the level less than the level contribution you have stipulated. This is in part because short events, affected by signal filtering, return amplitudes less than actually occurred (amplitudes of longer events are measured from the central portion of the event, avoiding the filter-affected transition periods at the start and end of the event). Because the reduced weighting of short events is designed to mitigate filtering effects, the amount of filtering a signal has undergone is used to determine the length of "brief " events, for example the point at which reduced weighting begins to apply. An event is classified short when it is shorter than 50 sample points multiplied by two time constants. As for single-channel event amplitudes (the previous section), two time constants is the time equivalent of nf, for the number of samples, in the following equation:

$$n_f = \frac{1}{2\pi\Delta t}\sqrt{\frac{1}{f_{Inst}^2} + \frac{1}{f_{CyberAmp}^2} + \frac{1}{f_{Postacq}^2}}$$

Here, t is the sampling interval, and f_{Inst}, $f_{CyberAmp}$, and $f_{Postacq}$ the –3 dB cutoff frequencies of the instrument (for example amplifier), CyberAmp and pCLAMP Software digital filters respectively.

As an example of the application of this, in a file recorded at 400 µs sampling interval (2500 Hz), with 500 Hz lowpass filter setting, any event with less than 40 samples (16 ms) qualifies as short. The contribution such events make to the level update is directly proportional to the extent the event is shorter than this, for example with a level contribution setting of 10%, an event lasting 8 ms (for example half a “short” event for these sampling and filtering rates) contributes 5% to the new level. On the other hand, all events over 16 ms affect the level update equally, at 10%.

Kolmogorov-Smirnov Test

The two-sample Kolmogorov-Smirnov (K-S) test is used to compare two independent “samples” (or “datasets”) to test the null hypothesis that the distributions from which the samples are drawn are identical. For example, the K-S test can be used to compare two sets of miniature synaptic event amplitudes to see if they are statistically different. Student’s t-test (which assumes data are normally distributed) and the nonparametric Mann-Whitney test (which does not) are sensitive only to differences in means or medians. The K-S test, however, is sensitive to a wider range of differences in the datasets (such as changes in the shape of the distributions) as it makes no assumptions about the distribution of the data. This wider sensitivity comes at the cost of power-the K-S test is less likely to detect small differences in the mean that the Student’s t-test or Mann-Whitney test might otherwise have detected.

The K-S test should be used on data that are not normally distributed. If the data conform to a normal distribution, Student’s t-test is more sensitive.

The test statistic (“K-S statistic”, or “*D*”) is the largest vertical (Y axis) difference between cumulative histograms derived from the datasets. The p value represents the probability that the two samples could have come from the same distribution by chance. In the K-S test, as in other tests, a $p < 0.05$ is typically accepted, by convention, as “statistically significant”—the samples are significantly different.

As part of the analytical process, this test measures the maximum value of the absolute difference between two cumulative distributions. This value is referred to as the Kolmogorov-Smirnov D statistic (or the “K-S statistic”). Thus, for comparing two different cumulative distribution functions, $S_{N1}(x)$ and $S_{N2}(x)$, the K-S statistic is:

$$D = (-\infty < x < +\infty)\overset{max}{\left|S_{N_1}(x) - S_{N_2}(x)\right|}$$

The calculation of the significance of the null hypothesis that the samples drawn from two distributions belong to the same population uses a function that can be written as the following sum:

$$Q_{KS}(\lambda) = 2\sum_{j=1}^{\infty}(-)^{j-1}e^{-2j^2\lambda^2}$$

which is a monotonic function with the limiting values:

$$Q_{KS}(0) = 1 \qquad Q_{KS}(\infty) = 0$$

In terms of Equation 1, the significance level p of an observed value of *D*, as an evaluation of the null hypothesis that the distributions are the same, is approximated by the formula:

$$P(D > observed) = Q_{KS}\left(\left[\sqrt{N_e} + 0.12 + \frac{0.11}{\sqrt{N_e}}\right]D\right) \quad \text{(Stephens, 1970)}$$

where N_e, the effective number of data points, is given by:

$$N_e = \frac{N_1 N_2}{N_1 + N_2}$$

where N_1 is the number of points in the first distribution and N_2 is the number of points in the second distribution (Press et al. 1992). The closer the value of *p* is to 1, the more probable it is that the two distributions come from the same population.

pCLAMP Software supports both “one-dimensional” and “two-dimensional” K-S tests. In the case of the one-dimensional test, the values of the data samples that are to be compared are not altered prior to the application of the test. One simply selects two different sets of data and applies the K-S statistic. In the case of the two-dimensional test, the data from the two populations are binned into a cumulative histograms (either fixed or variable width), and the binned distributions are subsequently evaluated using both the *x* values (bin widths) and *y* values (bin counts).

Normalization Functions

Normalization functions are located in the following five locations in pCLAMP Software.

Normalize Traces

Analyze > Normalize Traces is used to normalize trace data in the Analysis window. It offers two normalization options:

Use All Points

The traces are adjusted such that all points span the range 0 to 1. The normalization function for each point *y* in the trace is:

$$f(y) = (y - y_{min})/(y_{max} - y_{min})$$

where y_{min} is the smallest value in the trace and y_{max} is the largest value in the trace.

Use Specified Regions

The traces are adjusted such that the mean of the “Baseline region” is at 0 and the traces peak at +1.0 or –1.0 in the “Peak region”. The normalization function for each point *y* in the trace is:

$$f(y) = (y - b_y)/p_{|y|}$$

where b_y is the arithmetic mean of the points in the specified baseline region and $p_{|y|}$ is the largest absolute excursion between the mean baseline value and the specified peak region, where, for points n in the peak region:

$$p_{|y|} = max|y_n - b_y|$$

If the largest excursion from by in the peak region is to a value more negative than b_y, the normalized trace will peak at –1.0 in the peak region. If the largest excursion from by in the peak region is to a value more positive than b_y, the normalized trace will peak at +1.0 in the peak region. If the response in the peak region contains values both above and below b_y, only the polarity that contains the largest absolute excursion from b_y will span a range of 1.0.

Normalize

Analyze > Normalize is used for normalizing data in Graph windows. Either the area under the plot is normalized, or the Y axis range.

Normalize Area

Plots are adjusted such that the area under the curves is equal to 1. The normalization function for each point y in the plot is:

$$f(y) = y/|y_{total}|$$

where | y_{total} | is the sum of the absolute *y* values in the plot. This feature is primarily intended for normalizing the area under histograms.

Normalize Range

The range option adjusts the points so that all points span the range 0 to 1. The normalization function for each point y in the trace is:

$$f(y) = (y - y_{min})/(y_{max} - y_{min})$$

where y_{min} is the smallest value in the trace and y_{max} is the largest value in the trace.

The Norm() Function in Trace Arithmetic

Data file traces in the Analysis window can be normalized using **Analyze > Arithmetic**. The norm() function adjusts the trace such that all points span the range 0 to 1. The normalization function for each point y in the trace is:

$$f(y) = (y - y_{min})/(y_{max} - y_{min})$$

where y_{min} is the smallest value in the trace and y_{max} is the largest value in the trace.

The Norm() Function in Column Arithmetic

Data in Results window columns can be normalized using **Analyze > Column Arithmetic**. The norm() function adjusts the column data such that all points span the range 0 to 1. The normalization function for each point y in the column is:

$$f(y) = (y - y_{min})/(y_{max} - y_{min})$$

where y_{min} is the smallest value in the trace and y_{max} is the largest value in the trace.

Normalize Histogram Area

When histograms are created from Analysis, Graph and Results window data using **Analyze > Histogram**, you have the option of creating a histogram with the area under the plot normalized.

The normalization function for each point *y* in the plot is:

$$f(y) = y/|y_{total}|$$

where $|\ y_{\text{total}}\ |$ is the sum of the absolute *y* values in the plot.

Variance-Mean (V-M) Analysis

Variance-mean analysis (V-M) is a new analytical method, developed by Clements and Silver (1999), to quantify parameters of synaptic function.

The starting point for V-M analysis is the premise that transmission at a synapse can be described by three parameters. The first is the average amplitude of the postsynaptic response to a packet of transmitter (*Q*), the second is the number of independent presynaptic release sites in the presynaptic terminal (*N*) and the third is the average probability of transmitter release from the presynaptic release sites (P_r). Synaptic strength is defined by these parameters with *Q* being an expression of the postsynaptic efficacy and P_r an expression of the presynaptic efficacy.

Presynaptic modulation will alter P_r, postsynaptic modulation will alter *Q* and a change in the number of functional release sites, for example, a change in the number of functional synapses, will alter *N*. V-M analysis proposes a method whereby these parameters can be extracted by measuring synaptic amplitude fluctuations.

The experimental approach involves the acquisition of a number of records of synaptic activity at different levels of P_r. These levels can be altered by adding calcium blockers such as Cd^{2+} to the experimental solution or by altering the Ca^{2+}/Mg^{2+} ratio. The variance and mean of the individual postsynaptic current amplitudes (PSC) are measured during a stable recording period in each of the different experimental solutions, and the variance is plotted against the mean (V-M plot). The general form of such plots will be parabolic, where the initial slope is related to *Q*, the degree of curvature is related to P_r and the amplitude is related to *N*. Theoretically, a visual comparison of the different curves under different experimental conditions can thus provide insight into which synaptic parameter was altered.

The complexity of the mathematical treatment of the results depends on the data. When the V-M plot is linear or when it curves and reaches a plateau but does not descend back towards the x-axis, P_r is likely to be in the low to moderate range. The plot can then be analyzed by fitting the parabolic function from equation (1) where y is the variance of the PSC and x is the mean PSC amplitude:

$$y = (ix - x^2)/N$$

The free parameters i and N can be used to calculate the average synaptic parameters:

$$Q_w = i/(1 + CV_I^2)$$

and:

$$P_{rw} = x(i/N)(1 + CV_I^2)$$

The w subscript indicates that P_{rw} and Q_w are weighted averages that emphasize terminals with higher release probabilities and larger postsynaptic amplitudes (Reid and Clements, 1999). The lower limit of the number of independent release sites is given by N. CV_I is the coefficient of variation of the PSC amplitude at an individual release site and is determined experimentally (see Reid and Clements, 1999). CV is defined as:

$$CV = SD/mean$$

Another way to think of CV is as the inverse of the signal-to-noise ratio. A small value for CV implies good signal-to-noise.

This parameter appears in two different ways in synaptic fluctuation analysis. The first is as an indication of the variability of the response to a single vesicle (quantal variability). This can be split into two main sources-the variability at a single release site (CV_i), and the variability from site-to-site (CV_{ii}). The other way CV is sometimes mentioned in synaptic fluctuation analysis is as the CV of the evoked synaptic amplitude. It usually appears as:

$$1/CV^2$$

The reason for calculating this can be seen if this is expanded using equation (4):

$$1/(SD/mean)^2 = mean^2/SD^2 = meanx(mean/Var)$$

The initial slope of the V-M plot is the quantal amplitude Q, assuming low P_r. Therefore:

$$1/CV^2 = mean/Q$$

If synaptic modulation is by a postsynaptic mechanism then both mean and Q will be scaled by the same amount and will be unchanged. pCLAMP Software can calculate the synaptic parameters described in equations (2) and (3) and write the results to the Lab Book.

If the V-M plot is approximately linear then P_{rw} is low and the plot can be analyzed by fitting a straight-line equation:

$$y = ix$$

This permits an estimate of Q_w from equation (2) but P_{rw} and N (equation 3) cannot be estimated.

Rundown Correction

Amplitude rundown is characterized by a progressive and steady decline in event amplitude. To correct for such rundown for V-M analysis, linear regression is applied to the amplitude data to obtain the relation:

$$A = mt + b$$

where *A* is the expected amplitude at time *t* (the time of the event relative to the start of the data record), *m* is the slope and *b* is the y-intercept. The data are then corrected for rundown by applying:

$$a' = a - mt$$

to each data point, where *a* is the observed amplitude and *a´* is the corrected amplitude.

As rundown correction necessarily alters the observed data, it should be applied only in those cases where the amplitude decline is noticeably linear.

Burst Analysis

Poisson Surprise

The “Poisson Surprise” method (Legéndy & Salcman, 1985) is a statistical technique for detecting bursts. The name derives from the definition, being the degree to which the burst surprises one who expects a train of events to be a Poisson process. The degree of variation from a Poisson process is used as a measure of the probability that the observed train is a burst.

The Poisson Surprise (*S*) requires an evaluation of how improbable it is that the burst is a chance occurrence. It is computed, for any given burst that contains n events in a time interval *T*, as:

$$S = -\log P$$

where *P* is the probability that, in a random train of events having the same mean rate *r* as the train of interest, a given time interval of length *T* contains n or more events. Assuming the random train of events is a Poisson process, *P* is given by Poisson’s formula, where:

$$P = e^{-rT} \sum_{i=n}^{\infty} (rT)^i / i!$$

Poisson Surprise Burst Analysis

In order to detect epochs of increased activity, a criterion is required for choosing the first and last events of what is to be a burst. For consistency with the definition of event rate, the number of events denoted by n in the equation above includes the first and excludes the last event.

Analysis begins with an initial pass through the data to compute the mean rate of the recorded events. In the second pass, the algorithm advances through the data, event by event, until it finds several closely spaced events, which are then taken to be a putative burst. The number of such events (N_B) can be specified, but the minimum number is 3. The interval (I), which is used to accept consecutive events, can either be specified or allowed to be set automatically. If automatically assigned, the minimum inter-event interval is one-half the mean inter-event interval of the entire dataset.

Once N_B events have been detected, the algorithm then tests whether including an additional event at the end of the train increases the Poisson Surprise of the burst. If so, the additional event is included. This process is repeated until a relatively long interval (greater than I) is encountered or the inclusion of up to 10 additional events fails to increase the Poisson Surprise. If automatically assigned, the rejection interval is set to twice the mean interval.

When the putative burst is terminated by either of the above criteria, the algorithm tests whether removing one or more events from the beginning of the burst further increases the Poisson Surprise.

Classification of bursts is accomplished by evaluating the Poisson Surprise value, which is reported with the parameters of each burst. Legéndy and Salcman classified bursts having S values of 10 and above as significant. For further details of statistical analysis of S values, the user is referred to the original reference.

Specified Interval Burst Analysis

Specified interval burst analysis uses a “burst delimiting” interval to detect and construct bursts. During analysis, this delimiting interval is compared to the “inter-event” interval. In the case of single channel data, the inter-event interval is defined as the beginning of one event to the beginning of the next. In the case of peak time data, the inter-event interval is the time between event peaks.

The algorithm examines each inter-event interval. If this interval is less than or equal to the specified delimiting interval then a putative burst is assumed. If the subsequent interevent intervals are less than or equal to the delimiting interval, those events are added to the burst. The burst is ended if an inter-event interval greater than the delimiting interval is encountered. If the number of events in the burst is equal to or greater than a specified minimum number of events then the burst is accepted.

Peri-Event Analysis

Peri-event analysis consists of the measurement of the time of occurrence of events over specified intervals relative to a “central” event (for example,. a stimulus-evoked action potential). This generates data of the form $T_b < T_c < T_a$ where T_b and T_a are the times of occurrence of events before and after the central event, respectively. Tc is the time of the central event, which is always set to zero and T_b and T_{aa} are adjusted accordingly. Thus, if T_{co} is the original, nonzero time of the central event, and T_{bo} and T_{co} are the original times of the events before and after T_{co}, for the time of occurrence of each event within the specified ranges before and after T_{co} we have:

$$T_c = T_{co} - T_{co} \quad (= 0)$$

$$T_b = T_{bo} - T_{co}$$

$$T_a = T_{ao} - T_{co}$$

These data can be displayed as a raster plot, where the data from each trace are displayed as individual points. A composite histogram having a specified bin width, using the times T_b and T_a from all of the data traces, can also be generated. T_c is at $x = 0$ in both types of graph.

Peri-event analysis is optimized to use event detection data that have been generated by pCLAMP Software. The central events must be “tagged” explicitly during event detection. This analysis cannot otherwise detect T_c. Data from other sources can also be used so long as peak time data and tagged events exist. This might require manual tagging of the results (a tag is designated by “T”), which would require setting up a separate tag column.

For general event detection, the times of occurrence of the events is the peak time relative to the start of the trace. For single-channel data the time of occurrence is the start time of each opening relative to the start of the trace.

P$_{(Open)}$

The probability of a channel being open (P$_{open}$) provides an indication of the level of activity of an ion channel. In the simplest case, it is given by:

$$P_{open} = \frac{t_o}{T}$$

where t_o is the total time that the channel was found in the open state and T is the total observation time. This equation holds true when there is a single channel in the patch and, consequently, no multiple openings in the data record. If a patch contains more than one of the same type of channel then:

$$P_{open} = \frac{t_o}{NT}$$

where *N* is the number of channels in the patch, and:

$$T_o = \sum Lt_o$$

where L is the level of the channel opening. This assumes that a level 2 opening, for example, is the result of two superimposed level 1 openings of the same type of channel, where a level 1 opening is assumed to be the opening of a single channel. If different levels are composed of different types of channels then this calculation no longer holds true.

In general, *N* should be assigned a value equal to or greater than the number of levels in the single-channel record. It cannot be assumed that *N* is equal to the number of levels because *N* channels in a patch can generate fewer than *N* levels if *N* channel openings never overlapped during the course of recording. This can easily be the case if the channel activity is low and/or the openings are brief. However, the minimum number of channels in the patch is at least the number of observed levels.

If the number of channels is not known and there is reason to suspect that the number of levels does not accurately reflect the number of channels, the probability of the channel being open, referred to as NP_o, can be computed by:

$$NP_o = \frac{T_o}{T_o + T_c}$$

where T_c is the total closed time.

Chapter 11: Curve Fitting

11

pCLAMP Software offers a powerful and flexible tool for fitting curves to data. An extensive set of options should satisfy the most demanding requirements. Four different search methods support a large number of predefined functions as well as user-defined (custom) functions. Automatic seeding of function “parameters” is provided for the predefined functions. Graphically assisted seeding is also available for more demanding fits that require accurate determination of initial seed values.

A “fitting method” is composed of a search method, a minimization method and an optional weighting method. The search method is the algorithm that reduces the difference between the data and the fitted function. The minimization method is used to specify the quantity that is to be minimized (or maximized in the case of maximum likelihood).

The search methods include Levenberg-Marquardt, variable metric, Simplex and Chebyshev.

The minimization methods include sum of squared errors, maximum likelihood, mean absolute and minimax.

The weighting methods include function weighting, data weighting, bin width weighting or no weighting.

Linear and polynomial regression routines are also provided. These non-iterative methods are automatically used when linear regression or polynomial regression is selected from the predefined function list. However, custom-defined linear or polynomial functions can only be fitted by means of one of the iterative search methods.

Fitting Model

A “fitting model”, or simply, “model” is defined as any function that is fitted to the data. Functions with different numbers of terms are considered to be different models. For example, a two-term exponential function and a three-term exponential function represent different models.

Fitting to Split-Clock Data

Some older data might have been acquired using two different sampling rates (split clock). Most predefined functions and all custom functions can be fitted to such data since the fitting deals with X-Y data pairs and is unaffected by the degree of uniformity of spacing along the X axis. Exceptions include those predefined functions that require the X axis to be either normalized or converted to integers, and functions being fitted with the Chebyshev search method, where uniform spacing along the X axis is required.

For custom functions, it is up to you to ensure that the function deals with split clock data properly. If the custom function does demand uniform data spacing along the X axis then the fit to split-clock data will most likely be compromised.

Dual sample-interval data can be converted to a single sample interval by using **Analyze > Interpolation**, as described in the Evaluation of Multicomponent Signals: Sensillar Potentials with Superimposed Action Potentials on page 140 of Clampfit Software Tutorials on page 133.

Function Parameters

The term “parameters” refers to those coefficients of the fitting function that are adjusted during fitting. For example, in the following function, A, τ and C are the function parameters. In all predefined functions the variable C is a constant offset in the y direction:

$$f(x) = Ae^{-t/\tau} + C$$

Parameters are or are not adjusted by the fitting routine depending on whether they are “free” or “fixed”. Fixed parameters, which are essentially constants, are always assigned a standard error of zero.

Parameter Errors

All search methods report a standard error for each parameter in the fitting function. Parameter errors are estimated by evaluation of a covariance matrix using Gauss-Jordan elimination. The method of error evaluation is identical for all fitting methods, so the results of fitting by different methods can be compared directly.

In some cases parameter errors cannot be estimated because the covariance matrix cannot be evaluated. In this unlikely event the message “Could not compute parameter errors.” is given in the Results window.

The parameter errors provide an estimate of the uncertainty in the determination of the parameters themselves. They do not necessarily provide information about the goodness of the fit. The correlation coefficient and the standard deviation of the fit are more reliable indicators of the quality of the fit. In fact, if the fit is poor the parameter errors are likely to be meaningless. In other words, the parameter errors are an indication of how reliably the parameters of a given model were determined for a particular dataset, where small errors suggest that the parameter estimates are reliable regardless of the quality of the fit. Therefore, the parameter errors can be quite small although the deviation between the fitted curve and the data might be quite large.

Alternatively, for small datasets the parameter error estimates can be quite large (perhaps as large or even larger than the parameter estimates themselves), but the fit, nevertheless, can still be quite good. Clearly, statistical parameters such as estimated errors cannot be as reliable with small datasets as with larger sets.

Fitting Failure

The fitting algorithms in pCLAMP Software are very robust and will perform satisfactorily in most cases. However, there is always the possibility that a fit will fail. The most likely reasons for a fit failure are:

- The data are very poorly described by the fitting function.
- The initial seed values were very inaccurate.
- Sign restrictions were applied and the search algorithm cannot assign acceptable positive values to the function parameters, for example the data cannot be reasonably well described by the sign-restricted fitting function.

In the event of a fit failure possible solutions are:

- Ensure that the data are indeed reasonably well-represented by the fitting function. If not, select or define a different fitting function. Also, try a different number of terms or run a model comparison.
- Assign more accurate seeds to the fitting function. Graphical seeding should be very helpful for this.
- Use the variable metric search method. This search method is the most reliable for forcing function parameters positive.
- Disable the Force parameters positive option.
- Reduce the tolerance. This could result in a poorer, although still acceptable, fit.
- Reduce the maximum number of iterations. Sometimes an acceptable fit can be achieved (as judged by the parameter errors and the quality of the fitted curve) even though the fit does not converge. This is especially true for Simplex, which can continue to search for many iterations even though it is very close to the function minimum.

In the event of a failed fit the error is reported in the Results window. When fitting multiple sweeps, errors do not cause execution to stop. If an error occurs while fitting a given sweep, the error is recorded in the Results window and fitting continues with the next sweep. Therefore, if you have fitted a series of sweeps you should check the fitting results to ensure that all sweeps have been successfully fitted.

Numerical Limitations

- The maximum number of data points that can be fitted is 110,000.
- The maximum number of function terms (where applicable) is 6.
- The maximum power (where applicable) is 6.
- The maximum number of parameters in a custom function is 24.
- The maximum number of independent variables in a custom function is 6.
- Only one dependent variable is allowed in a custom function.
- The maximum number of points for functions that contain a factorial term is 170.

Units

The fitting routines in pCLAMP Software do not make any assumptions about the units of the data. The variety of data sources and the potential for various data transformations makes the automatic tracking or assignment of units virtually impossible. Consequently, units are not displayed along with the reported parameter values. It is up to the user to determine the units of a particular function parameter.

The Levenberg-Marquardt Method

The Levenberg-Marquardt method supports the least squares, mean absolute and minimax minimization methods. The explanation given here is for least squares minimization but the general principle is the same for all minimization functions.

The sum of squared errors (SSE) is first evaluated from initial estimates (seed values) for the function parameters. A new set of parameters is then determined by computing a change vector P that is added to the old parameter values and the function is reevaluated. The value of P will depend on the local curvature in the "parameter space" that can be evaluated to determine the optimal rate and direction of descent toward the function minimum. This process continues until the SSE is "minimized" at which time the fit is said to have converged. The criteria by which is judged to be at its minimum are different for the different search methods.

The Levenberg-Marquardt search method combines the properties of the steepest descent and the Gauss-Newton methods. This is accomplished by adding a constant λ to the diagonal elements of the Hessian matrix that is associated with the gradient on the parameter space. If λ is large the search algorithm approaches the method of steepest descent. When λ is small the algorithm approaches the Gauss-Newton direction.

The method of steepest descent can optimally find major changes in the data and thus works best in the early stages of the fit when the residual of the sum of squares is changing substantially with each iteration. The Gauss-Newton method is best for smoothing out the fit in later stages when these residuals are no longer changing substantially (see Schreiner et al. 1985). The Levenberg-Marquardt method requires that the first derivative of the function *f (x,P)* be evaluated with respect to each parameter for each data point. These derivatives are used to evaluate the "curvature" in the local parameter space in order to move in the direction of the perceived minimum. For predefined functions the exact derivative is calculated. For custom functions a numerical derivative (central difference) is computed using a step size of 10^{-7}.

As the fit progresses some steps may result in a poorer (larger) value of the SSE. However, the general trend is a reduction in the SSE.

The Levenberg-Marquardt method does not report SSE during the fitting process but rather reports the standard deviation (σ). However, σ follows the same trend as the SSE, that is, if the SSE increases then σ also increases, and vice-versa. In fact, the standard deviation is reported by all search methods, providing a standard criterion for judging the fitting quality regardless of the search method. The standard deviation is given by:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (Obs - Exp)^2}{n - 1}}$$

where *n* is the number of data points, *Obs* is the observed value and *Exp* is the expected value as calculated using the fitting function.

Levenberg-Marquardt Convergence

Convergence is reached when the parameter change vectors go to zero, which occurs when a minimum is reached in the local parameter space. Note that because of this the fitting function might converge to a minimum that is not necessarily the lowest (global) minimum in the entire parameter response surface. Convergence to a local minimum often results in a poorly fitted curve and so is easily recognized. If you suspect that the fit has converged on a local minimum, you should specify new fitting seed values (graphically-assisted seeding is very useful here) and retry the fit. Alternatively, use a different fitting method. For example, the Simplex search method is not as prone to convergence at a local minimum.

The iterations are also stopped (convergence is assumed) when the change in the minimization function (for example, the SSE) is less than a preset value. This value can be set in the Precision field in the **Function > Method** tab of the fitting dialog. The default value is 10^{-6}.

Normally, it is preferable to allow the parameters to converge naturally. Convergence on a precision criterion can result in a poorer fit especially if the precision criterion is reached before the individual parameters have converged. On the other hand, some “difficult” fits might require hundreds or even thousands of iterations if only the change vector criterion is used for convergence. In order to favor convergence on the basis of change vectors but to also allow difficult fits to converge on the basis of an acceptable “precision” value, the fitting routine converges on the precision criterion only if this criterion has been met over at least 100 successive iterations. Given this criterion it is not likely that further improvements in the minimization function will lead to a better fit, so the iterations will stop.

Levenberg-Marquardt Precision

The default Levenberg-Marquardt precision is 10^{-6}.

The Levenberg-Marquardt precision sets the minimum absolute change in the minimization function (for example, the SSE) that signifies convergence. This minimum difference must be satisfied over at least 100 successive iterations. That is, if:

$$absolute(SSE(old) - SSE(new)) < Precision$$

over 100 consecutive iterations, convergence is assumed. A less stringent precision value could facilitate convergence for a particularly difficult dataset, but often at the expense of fitting accuracy. In any case, the statistics of the fit should always be carefully evaluated in order to determine whether or not the fit is acceptable.

The Simplex Method

The Simplex method supports the least squares, mean absolute, maximum likelihood and minimax minimization methods. The explanation given here is for least squares minimization but the general principal is the same for all minimization functions.

The Simplex search method is based on the algorithm of Nedler and Mead (1965), and is an example of a direct search approach that relies only on the values of the function parameters. It does not consider either the rate or the direction by which the function is approaching the minimum on the parameter response surface. However, the direction in which the function parameters proceed is not purely random but rather relies on a clever strategy that takes advantage of the geometry of the response surface.

A simplex is a geometric figure that has one more dimension than the parameter space in which it is defined. The vertexes of the simplex are first separated by adding an “offset” to the initial seed values. The function to be minimized is then evaluated at each vertex to identify the lowest and highest response values. For example, a simplex on a two dimensional space (corresponding to a two-parameter function) is a triangle that may have the following appearance:

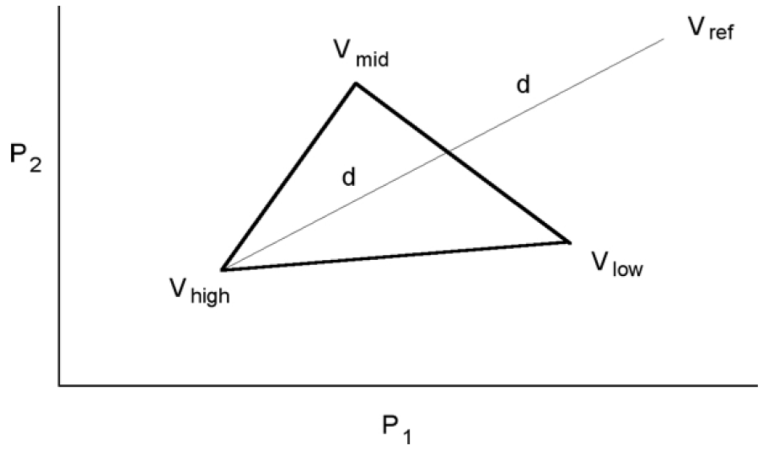


Figure 11-1: A simplex for a two-parameter function.

where P_1 and P_2 are the parameters, V_{high} is the vertex which has the highest (worst) function value, V_{low} is the vertex which has the lowest (best) function value and V_{mid} represents an intermediate function value. A “downhill” direction is established by drawing a line from V_{high} through a point midway between V_{mid} and V_{low}. The algorithm then tries to find a point along this line that results in an SSE which is lower than the existing vertexes.

The simplex changes shape by reflection, contraction, expansion or shrinkage. The first point tested is the reflected point V_{ref} which lies a distance of 2d along the line from V_{high}. This reflected simplex is accepted if its response is neither worse than V_{high} nor better than V_{low}. If the response is better than V_{low} then the algorithm performs an expansion by moving a distance of 4d along the line from Vhigh. The expansion is accepted if it has a lower (better) response than the previous best. Otherwise the reflection is accepted.

If the reflection results in a higher (worse) response than V_{high} then the algorithm tests a contraction by moving a distance of 0.5d toward the midpoint on the line. If this produces a better response then the simplex is accepted; otherwise shrinkage occurs where all vertexes except V_{low} move toward the midpoint by one-half of their original distance from it.

The advantage of the Simplex algorithm is that it is considerably less sensitive to the initial seed values than the gradient search algorithms. It will rapidly approach a minimum on the parameter response surface usually in the space of several tens of iterations for a multicomponent function.

The disadvantage of the Simplex fitting method is that its sensitivity does not increase when it is in the vicinity of a minimum on the parameter space.

Another problem can arise in that the Simplex algorithm may find what it perceives to be a local minimum but the fractional error is still greater than the convergence criterion (see below). In this case iterations may continue endlessly. To circumvent this problem, the fitting routine will stop when there is no change in the fractional error for 30 iterations. Even if the above error criterion is not met, the fit is assumed to have converged. This occurrence is reported as a “Stable Fractional Error” in the Results Window. The displayed value of σ may be the same for many more than 30 iterations before the fitting routine is terminated. This is because σ is reported as a single precision value whereas the fractional error is a double precision value.

The weighting options are not available with Simplex fitting. This is because weighting interferes with the “travel” of the simplex and greatly reduces the chances of convergence.

Simplex Convergence

The Simplex algorithm can converge in one of three ways. The iterations are stopped when the fractional error is less than or equal to a preestablished “precision” value.

The simplex is moved over the parameter space until the ratio of the response of the best and worst vertexes reaches a preset minimum fractional error, at which point the function is said to have converged:

$$\mathit{Fractional\ Error} = \frac{V_{high} - V_{low}}{V_{high}}$$

The quantities on the right-hand side of the equation are based on one of four minimization methods that Simplex can use, namely least squares, maximum likelihood, mean absolute or minimax.

The fractional error is computed for the simplex for each dimension (each having a V_{high} and V_{low} simplex). All simplexes must have a fractional error less than the value of precision for convergence.

Simplex Precision

The default Simplex precision is 10^{-5}.

The Simplex search is deemed to have converged when the fractional error is less than or equal to the Precision value. The fractional error can be based on one of four minimization methods, namely least squares, maximum likelihood, mean absolute or minimax.

The Variable Metric Method

The variable metric method supports the least squares and maximum likelihood minimization methods only.

Variable metric algorithms are designed to find either the minimum or the maximum of a multi-dimensional non-linear function ƒ. The minimum is used in chi-squared or least squares applications and the maximum in maximum likelihood estimation.

pCLAMP Software typically uses minimization of least squares, which is asymptotically equivalent to likelihood maximization for most applications (Rao, 1973). The parameter values that determine the global minimum of ƒ are optimal estimates of these parameters. The goal of the variable metric algorithm is to find these estimates rapidly and robustly by an iterative method.

pCLAMP Software uses the variable metric algorithm introduced by Powell (1978) and implemented by Dr. Kenneth Lange (UCLA). At each iteration the algorithm computes the exact partial derivative of f with respect to each parameter. The values of these derivatives are used in building an approximation of the Hessian matrix, which is the second partial derivative matrix of ƒ. The inverse of the Hessian matrix is then used in determining the parameter values for the subsequent iteration.

Variable metric algorithms have several desirable characteristics when compared with other methods of non-linear minimization. Like the simplex algorithm, variable metric algorithms are quite robust, meaning that they are adept at finding the global minimum of ƒ even when using poor initial guesses for the parameters. Unlike simplex, however, convergence is very rapid near the global minimum.

Variable Metric Convergence

The variable metric search method converges when the square of the residuals, or the maximum likelihood estimate if using likelihood maximization (see Maximum Likelihood Estimation on page 227), does not change by more than a preset value over at least four iterations. This value can be set in the Precision field in the **Function > Method** tab of the **Fitting** dialog. The default value is 10^{-4}.

Variable Metric Precision

The default variable metric precision is 10^{-4}.

The variable metric search is assumed to have converged when the square of the residuals, or the maximum likelihood estimate if using likelihood maximization, does not change by more than the Precision value over at least four iterations.

The Chebyshev Transform

The Chebyshev technique is an extremely rapid, stable, noniterative fitting technique with a goodness of fit comparable to that of iterative techniques. It was developed by George C. Malachowski and licensed to Axon Instruments. The explanation in this section only describes how the Chebyshev transform is used to fit sums of exponentials.

The Chebyshev Transform transforms a dataset to the Chebyshev domain (in this case, domain refers to a functional basis, in which datasets are considered in terms of scaled sums of functions), using a method equivalent to transforming data to the frequency domain using Fourier transforms. Instead of representing the data using a sum of sines and cosines, the Chebyshev transform uses a sum of the discrete set of Chebyshev polynomials (described below).

Transforming the data allows it to be fitted with various functions of biological interest: sum of exponentials, Boltzmann distribution and the power expression (an exponential plus a constant raised to a power). For each of these functions, it is possible to derive an exact, linear, mathematical relationship between the fit parameters and the coefficients of the transformed data. If the dataset has noise present, as is almost always the case, these relationships provide estimates of the fit parameters. However, as the relationships are linear, high-speed regression techniques can be applied to find the parameters.

This method has the following properties: extremely fast fitting at a rate that is independent of noise, comparable goodness of fit to that of iterative techniques, and it always finds a fit.

At present this technique has one limitation: it can only be used on datasets that have equally spaced data points. This makes it inappropriate at present for fitting histogram data with variable bin widths.

The Orthogonal Polynomial Set

An *N*th order polynomial is a function of the form:

$$P_N(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_Nx^N$$

where a_0, a_1, a_2, ... a_N are the coefficients of the polynomial, and a_N cannot be zero unless *N* is zero.

A set of polynomials consists of one polynomial from each order ($P_0(x)$, $P_1(x)$, $P_2(x)$, $P_3(x)$, ...), in which adjacent members of the set are related by a generating equation. Mathematicians usually represent a polynomial set with a single letter. In the case of the Chebyshev polynomials, the letter is “T” (from previous anglicization of Chebyshev as Tchebysheff).

A polynomial set is said to be orthogonal if every member in the set is orthogonal to every other member. In the case of the Chebyshev polynomials, T_0 is orthogonal to T_1, T_2, T_3, T_4, and so on, T_1 is orthogonal to T_2, T_3, T_4, T_5, and so on.

Orthogonal means at a right angle; synonyms are perpendicular and normal. Those familiar with vectors may recall that two vectors are tested for orthogonality using the dot product; two vectors are said to be orthogonal if their dot product equals zero. Similarly, two functions that are defined only at discrete points in time are said to be orthogonal if the sum of their product over all sampled points is zero. (This relation may only be true on a restricted range (for example $x \in [-1, 1]$), and with a weighting function present.)

Transforming Data to the Chebyshev Domain

All orthogonal function sets have the property of being able to represent a function in terms of a sum (or integral) of the members of the set. Depending on the function being represented, an appropriate scaling factor is chosen for each member of the set. A function *f (t)* can be represented as a sum over the Chebyshev polynomials by the relation:

$$f(t) = \sum_{j=0}^{\infty} d_j T_j(t)$$

where T_j *(t)* is the jth member of the Chebyshev polynomial set and dj is its scaling factor. In general, a sum of an infinite number of Chebyshev polynomials is required to represent a continuous function. In the case of a sampled dataset with N sampled points, a sum of only *N* Chebyshev polynomials, from order 0 to *N*–1, is required to exactly represent the dataset. In this case, *t* is not continuous, but is rather a set of data points t_j, where $_j$ runs from 0 to *N*–1. The above equation then becomes:

$$f(t_i) = \sum_{j=0}^{N-1} d_j T_j(t_i). \qquad \text{for } i = 0,\ldots, N-1$$

This sum of polynomials exactly equals the function at all points in time, even though the individual members may only cross the function at a few points.

A function that has been represented this way is said to have been transformed to the Chebyshev domain, and the scaling factors (the d_js) are usually referred to as the coefficients of the transform. Do not confuse these transform coefficients with the individual coefficients that make up each of the member Chebyshev polynomials! The member polynomials' coefficients never change; the coefficients of the transform change with *f(tj)*.

Calculating the Coefficients

The orthogonality property of the Chebyshev polynomials makes calculating the d_js straightforward. Recall that every member T_k is orthogonal to every other member T_j, for all $k \neq j$. To determine each coefficient (d_k), both sides of the above equation are multiplied by T_k, then summed over all values of t_i:

$$\sum_{i=0}^{N-1} T_k(t_i) f(t_i) = \sum_{i=0}^{N-1} T_k(t_i) \sum_{j=0}^{N-1} d_j T_j(t_i)$$

where T_k is the member whose coefficient is to be determined. Rearranging, and summing over all data points ti eliminates all T_js except T_k, leaving (after several steps):

$$\sum_{i=0}^{N-1} T_k(t_i)f(t_i) = d_k \sum_{i=0}^{N-1} T_k^2(t_i)$$

The summation on the right-hand side is usually written as a normalization factor (R_k).

Solving for d_k:

$$d_k = \sum_{i=0}^{N-1} T_k(t_i)f(t_i)/R_k$$

The Discrete Chebyshev Polynomials

The generating equation for the Chebyshev polynomials is given by (Abramowitz and Stegun, p. 792):

$$j(N-j)\cdot T_j(t) = (2j-1)(N-1-2t)\cdot T_{j-1}(t)-(j-1)(N-1+j)\cdot T_{j-2}(t)$$

where $T_j(t)$ is the Chebyshev polynomial being generated, and N is the number of data points. It is clear that each $T_j(t)$ depends on the previous two members in the set: $T_{j-1}(t)$ and $T_{j-2}(t)$. The zero^th member of this set is defined to be: $T_0 = 1$, from which all highe-order members may be derived. T_1 and T_2 are shown below:

$$T_1(t) = 1-\frac{2}{(N-1)}t$$

$$T_2(t) = 1-\frac{6}{(N-2)}t+\frac{6}{(N-1)(N-2)}t^2$$

Clearly, T_0 is a horizontal line, T_1 is a sloping line and T_2 is a parabola.

Isolating the Offset

The offset of a dataset can be isolated from the data in the Chebyshev domain. Consider the general case of a function containing an offset; we can rewrite this function in terms of its non-constant and its constant parts:

$$f(t_i) = g(t_i)+K$$

where $g(t_i)$ is the non-constant part of the function. Using this function in Equation 2 we derive:

$$d_j = \left(\sum_{i=0}^{N-1} T_j t_i g t_i/R_j\right)+\left(\sum_{i=0}^{N-1} T_j t_i K/R_j\right)$$

The above equation is simply the sum of the Chebyshev transforms of *g* and *K*. This can be seen if we write the transform coefficients of *g* as $d'_j(g)$, and the transform coefficients of *K* as $d''_j(K)$:

$$d_j(g + K) = d'_j(g) + d''_j(K)$$

However, $T_0 = 1$ implies that the Chebyshev transform of *K* is nonzero only for the zeroth coefficient ($d''_0(K)$). (*K* can be rewritten as KT_0, and T_0 is orthogonal to all other Chebyshev polynomials.) We therefore have:

$$d_0 = d'_0(g) + d''_0(K)$$
$$d_j = d'_j(g) \qquad \text{for } j>0$$

Once a dataset has been transformed to the Chebyshev domain, we can isolate the effect of the constant offset by not using d_0 in our calculations of the other parameters.

Transforming Data to the Chebyshev Domain

All orthogonal function sets have the property of being able to represent a function in terms of a sum (or integral) of the members of the set. Depending on the function being represented, an appropriate scaling factor is chosen for each member of the set. A function *f (t)* can be represented as a sum over the Chebyshev polynomials by the relation:

$$f(t) = \sum_{j=0}^{\infty} d_j T_j(t)$$

where T_j *(t)* is the jth member of the Chebyshev polynomial set and dj is its scaling factor. In general, a sum of an infinite number of Chebyshev polynomials is required to represent a continuous function. In the case of a sampled dataset with N sampled points, a sum of only *N* Chebyshev polynomials, from order 0 to *N*–1, is required to exactly represent the dataset. In this case, *t* is not continuous, but is rather a set of data points t_j, where $_j$ runs from 0 to *N*–1. The above equation then becomes:

$$f(t_i) = \sum_{j=0}^{N-1} d_j T_j(t_i). \qquad \text{for } i = 0,\ldots, N-1$$

This sum of polynomials exactly equals the function at all points in time, even though the individual members may only cross the function at a few points.

A function that has been represented this way is said to have been transformed to the Chebyshev domain, and the scaling factors (the d_js) are usually referred to as the coefficients of the transform. Do not confuse these transform coefficients with the individual coefficients that make up each of the member Chebyshev polynomials! The member polynomials' coefficients never change; the coefficients of the transform change with *f(tj)*.

Integrating an Exponential Function in the Chebyshev Domain

Let us say that we have a function *f(t)* that we wish to integrate, and that its integral is *F(t)*. In the discrete domain, where we only have a finite set of data points (usually evenly spaced), we write the discrete integral as:

$$\sum_{0}^{t-1} f(t')dt' = F(t)$$

The discrete integral *F(t)* is defined at each value *t* by the sum of the previous values from 0 to *t*–1. Using *t*–1 as the end point of the integration serves to ensure that the forward difference equation is equal to *f(t)*:

$$F(t+1) - F(t) = f(t)$$

A difference equation in the discrete domain is analogous to a differential equation in the continuous domain. The equation shown above in the continuous domain would be expressed as *dF/dt* = *f(t)*. Forward difference refers to using t and *t*+1 to form the difference.

How are the Chebyshev transforms of the integral *F* and *f* related? If we were to transform f to the Chebyshev domain, we would obtain a set of coefficients $d_j(f)$. Similarly, if we were to transform *F*, we would obtain a different set of coefficients $d_j(F)$. Comparing these two sets of coefficients, $d_j(f)$ and $d_j(F)$ we would find the relation:

$$D_j(F) = \frac{1}{2}\left(\frac{(N+j+1)}{2j+3}d_{j+1}(f) - d_j(f) - \frac{N-j}{2j-1}d_{j-1}(f)\right) \quad \text{for all } j > 1$$

where $d_j(F)$ is the j^{th} coefficient of *F(t)* and *N* is the number of points in the data. This equation cannot tell us the value of $D_0(F)$, as there is no $d_{-1}(f)$ coefficient.

(This formula is derived in *A Method that Linearizes the Fitting of Exponentials*, G. C. Malachowski). This equation is critical to the use of this technique. Proof of this relation is long; those interested may refer to the appendix of the above paper. Briefly though, it can be described as follows: integrating the Chebyshev transform of a function is the same as the sum of the integrals of each of the Chebyshev polynomials making up the transformation. The integral of a polynomial is itself a polynomial. It turns out that after much simplification and rearrangement, each coefficient in the transform of the integral is a sum of the two adjacent coefficients in the transform of the original function.)

If *f* is an exponential function, the following, very similar, relationship exists:

$$\frac{d_j(f)}{k} = \frac{1}{2}\left(\frac{(N+j+1)}{2j+3}d_{j+1}(f) - d_j(f) - \frac{N-j}{2j-1}d_{j-1}(f)\right) \quad \text{for all } j > 1$$

where *k* is defined as:

$$k = e^{-1/\tau} - 1$$

or, solving for τ:

$$\tau = \frac{-1}{log_e(k+1)}$$

Basically, these equations tell us that any adjacent triplet of Chebyshev coefficients forms an exact relationship that tells us the value of tau. Note how Equation 4 further restricts the value of *j* to be greater than one: it is only true for those coefficients that do not contain the constant offset term.

Thus, integrating an exponential function in the Chebyshev domain allowed us to determine the value of tau. This is similar to the case of integrating an exponential function in the continuous domain:

$$\int e^{-t/\tau} dt = -\tau e^{-t/\tau} + C$$

For reasons that will become clear in the section on the fitting of two exponentials, the right-hand side of Equation 4 can be written as *$d1_j(f)$*, which stands for the Chebyshev coefficients of the first integral of ƒ. Equation 4 then becomes:

$$\frac{d_j(f)}{k} = d1_j(f) \qquad \text{for } j > 1$$

where *k* is as defined in Equation 5.

Calculating Tau

Now we can calculate τ using Equations 4 and 5. Choose any triplet of Chebyshev coefficients, and use those values in Equation 4 to get the value of *k*. Then use *k* in Equation 5 to calculate τ. Every triplet has the same, redundant information built into it: triplet d_1, d_2, d_3, triplet d_2, d_3, d_4, and so on. The following is an example of the values of τ predicted using the 8 triplets of the first eleven Chebyshev coefficients.

Note: Remember that d_0 cannot be used as it contains information about the offset of the exponential, if present.

Chebyshev	Calculations of τ
d_1, d_2, d_3	25.00000003
d_2, d_3, d_4	25.00000075
d_3, d_4, d_5	24.99999915
d_4, d_5, d_6	24.9999998
d_5, d_6, d_7	25.00000096
d_6, d_7, d_8	25.00000257
d_7, d_8, d_9	25.00000464
d_8, d_9, d_{10}	24.99999885

The small differences between the calculated value and the actual value (25) are due to the limited precision of the coefficients used. In general, double precision numbers are used to calculate the fit parameters (τ, the amplitude and the offset).

Calculating the Amplitude

To determine the amplitude of the exponential, we must change directions completely, and generate an exponential dataset based on the value of tau just calculated. The generated dataset will have unity amplitude and zero offset:

$$g(t_i) = 1e^{-t_i/\tau} + 0$$

Transforming this generated set to the Chebyshev domain will give us a different set of coefficients (*d'_j (g)*). Now we can determine the value of the amplitude of our dataset by comparing the *d_j(f)*s from our dataset to those of the generated set *d'_j(g)*s. Recalling that the function that we are trying to fit is:

$$f(t_i) = a_0 + a_1 e^{-t_i/\tau}$$

we can transform this function to the Chebyshev domain as:

$$d_j(f) = \left(\sum_{i=0}^{N} a_0 T_j(t_i)/R_j\right) + a_1\left(\sum_{i=0}^{N} e^{-t_i/\tau} \cdot T_j(t_i)/R_j\right)$$

The Chebyshev coefficients for *g(ti)* are very similar to those for *f(ti)*:

$$d'_j(g) = \sum_{i=0}^{N} e^{-t_i/\tau} \cdot T_j(t_i)/R_j \qquad \text{for all } j$$

Comparing these two equations yields the following relationship between the Chebyshev coefficients of f and those of *g*:

$$a_1 = \frac{d_j(f)}{d'_j(g)} \qquad \text{for all } j > 0$$

The amplitude is contained redundantly in all of the coefficients of the two transforms, excluding the zeroth coefficient. This redundancy is similar to that seen in the calculation of tau.

$$a_1 = \frac{d_1(f)}{d'_1(g)} = \frac{d_2(f)}{d'_2(g)} = \frac{d_3(f)}{d'_3(g)} = \frac{d_4(f)}{d'_4(g)} = \ldots$$

Calculating the Offset

Calculating the offset is similar to calculating the amplitude: we compare the zeroth index coefficients from the two sets of transforms:

$$a_0 = d_0(f) - a_1 d'_0(g)$$

Unlike τ and a_1, the offset information is not redundantly stored in the transform coefficients.

Calculating the Fit Parameters in the Presence of Noise

If the dataset being fit contains noise, Equations 4, 5, 7 and 8 are no longer exactly true.

For example, when calculating τ, each triplet gives an estimate of τ.

Estimating τ

Recall that for **j** > 1, Equation 4 shows a relationship between *k* and each triplet of Chebyshev coefficients. In the case of noise, this relationship is not strictly true; $d_i(f)/k$ is an estimate of the right side of this equation. Equation 6 then becomes:

$$d_j(f) \cong k d1_j(f)$$

The value of *k* that minimizes the following expression will be *k´*:

$$\chi^2 = \sum_{j=1}^{n} (d_j - k d1_j^2(f))$$

where the sum generally does not include all *N* of the coefficients, but generally includes only those coefficients with a significant contribution to the transform; usually *n* is chosen to be 20. Expanding the squared term, differentiating with respect to *k*, setting the derivative to 0 and rearranging gives us our estimate of *k*:

$$k' = \sum_{j=1}^{n} d_j d1_j(f) / \sum_{j=1}^{n} d1_j^2(f)$$

Once *k* has been estimated, Equation 5 will give the corresponding best estimate of τ.

Estimating a₁

A similar technique is used to calculate the best estimate of the amplitude a_1. The ratios of the coefficients of our dataset [$d_j(f)$s] to the coefficients of the pure exponential [$d'_j(g)$s] now give estimates of the amplitude:

$$\frac{d_j(f)}{d'_j(g)} \cong a_1$$

This can be rewritten as:

$$d_j(f) \cong a_1 d'_j(g)$$

As in the case of estimating tau, we form a linear regression equation to find the best estimate of a_1 (a_1'):

$$\chi^2 = \sum_{j=1}^{n} (d_j(f) - a_1 d'_j(g))^2$$

Expanding the squared term, differentiating with respect to a_1, setting the derivative to 0 and rearranging gives us:

$$a'_1 = \sum_{j=1}^{n} d_j(f) d'_j(g) / \sum_{j=1}^{n} (d'_j(g))^2$$

Estimating a₀

The estimate of a_0 is calculated by substituting the above value of a'_1 into Equation 8.

Fitting the Sum of Two Exponentials

In the two exponential case, two taus must be found. To do so, we shall integrate the function to be fit twice, and solve the resulting set of simultaneous equations for τ_1 and τ_2. (This procedure is somewhat complicated.) Once we have the two taus, solving for the amplitudes and the offset is a simple extension of the procedure for fitting a single exponential.

Suppose we wish to transform a signal $f(t_i)$ that is the sum of two exponentially decaying functions to the Chebyshev domain, where $f(t_i)$ is defined as:

$$f(t_i) = a_0 + a_1 e^{-t_i/\tau_1} + a_2 e^{-t_i/\tau_2}$$

Let $g_{\tau 1}(t_i)$ represent a unity-amplitude exponential with time constant τ_1 and let $g_{\tau 2}(t_i)$ represent a unity-amplitude exponential with time constant τ_2. Then we can write:

$$f(t_i) = a_0 + a_1 g_{\tau_1}(t_i) + a_2 g_{\tau_2}(t_i)$$

Transforming both sides of this equation gives us:

$$d_j(f) = \left(\sum_{i=0}^{n} (a_0 + a_1 g_{\tau_1}(t_i) + a_2 g_{\tau_2}(t_i)) T_j(t_i) \right) / R_j$$

or

$$d_j(f) = d_j^{offset}(a_0) + a_1 d_j^{\tau_1}(g_{\tau_1}) + a_2 d_j^{\tau_2}(g_{\tau_2})$$

where $d_j^{offset}(a_0)$, $dj^{\tau 1}(g_{\tau 1})$, and $d_j^{\tau 2}(g_{\tau 2})$ are the Chebyshev coefficients of a_0, $g_{\tau 1}$ and $g_{\tau 2}$ respectively. To isolate the constant term from the calculations that follow, we shall only use coefficients where $j > 0$, yielding:

$$d_j(f) = a_1 d_j^{\tau_1}(g_{\tau_1}) + a_2 d_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 0$$

since the offset is only contained in the zeroth coefficients.

Using the Coefficients of f and its Integrals to Determine the Taus

What if we were to integrate both sides of Equation 12? Since we are dealing with discrete data points, we use a summation. From Equations 4, 5 and 6, we know the coefficients of the integral of an exponential function, and we know how those coefficients are related to the coefficients of the exponential function itself. Applying those relationships to this sum of two exponentials case yields:

$$d1_j(f) = a_1 d1_j^{\tau_1}(g_{\tau_1}) + a_2 d1_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 1$$

or using Equation 6 to rewrite in terms of the coefficients of and themselves:

$$d1_j(f) = \frac{a_1}{k_1} d_j^{\tau_1}(g_{\tau_1}) + \frac{a_2}{k_2} d_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 1$$

where

$$\tau_1 = \frac{-1}{log_e(k_1 + 1)} \quad \text{and} \quad \tau_2 = \frac{-1}{log_e(k_2 + 1)}$$

Integrating both sides of Equation 13 again:

$$d2_j(f) = \frac{a_1}{k_1} d1_j^{\tau_1}(g_{\tau_1}) + \frac{a_2}{k_2} d1_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 2$$

where we write the Chebyshev coefficients of the second integral of f as *d2$_i$(f)*. *j* now must be greater than two. (The exact reason for this is beyond the scope of this description (see Malachowski's paper) however, briefly, it is required to isolate the effect of the offset from the calculation of the taus.) Substituting in Equation 6 again gives us our final relation that we need to determine τ_1 and τ_2:

$$d2_j(f) = \frac{a_1}{k_1^2} d_j^{\tau_1}(g_{\tau_1}) + \frac{a_2}{k_2^2} d_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 2$$

Solving a Set of Simultaneous Equations to Determine k$_1$ and k$_2$

Equations 12, 14 and 16 now form the three relations that we need in order to determine the taus. Rewriting them below we have three equations in three unknowns *d$_j$*, *d1$_j$* and *d2$_j$*, and restricting *j* to be the same in all cases:

$$d_j(f) = a_1 d_j^{\tau_1}(g_{\tau_1}) + a_2 d_j^{\tau_2}(g_{\tau_2})$$

$$d1_j(f) = \frac{a_1}{k_1} d_j^{\tau_1}(g_{\tau_1}) + \frac{a_2}{k_2} d_j^{\tau_2}(g_{\tau_2})$$

$$d2_j(f) = \frac{a_1}{k_1^2} d_j^{\tau_1}(g_{\tau_1}) + \frac{a_2}{k_2^2} d_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 2$$

In order for these three equations to be simultaneously true, there must exist a pair of parameters x_1 and x_2 such that for all $j > 2$:

$$d_j + x_1 d1_j + x_2 d2_j = 0$$

The solution to this equation is a straight line in x_1–x_2 coordinate space. To solve it, we add up the three simultaneous equations, and gather the like terms to find:

$$d_j + x_1 d1_j + x_2 d2_j = a_1 d_j^{\tau_1}(g_{\tau_1})\left(1 + \frac{x_1}{k_1} + \frac{x_1}{k_1^2}\right) + a_2 d_j^{\tau_2}(g_{\tau_2})\left(1 + \frac{x_1}{k_2} + \frac{x_1}{k_2^2}\right)$$

The values of x_1 and x_2 that satisfy this equation are:

$$x_1 = -(k_1 + k_2)$$
$$x_2 = k_1 k_2$$

as can be seen by substituting these values into the above equation.

Our strategy will be to solve for x_1 and x_2, from them calculate the values of k_1 and k_2, and finally use Equations 15a and b to calculate the corresponding values of τ_1 and τ_2. To do so, we must first solve for k_1 and k_2 in terms of x_1 and x_2 (Equations 18a and b are the converse). There is no direct, algebraic method to do so, but we can recognize that Equations 18a and b are the roots of the quadratic polynomial:

$$k^2 + x_1 k + x_2 = 0$$

as can be seen by factoring the polynomial into the product of its roots:

$$k^2 + x_1 k + x_2 = (k - k_1)(k - k_2)$$

This means that we can determine k_1 and k_2 by using x_1 and x_2 to form the above quadratic polynomial, and then solving for its roots. For a quadratic polynomial, we use the quadratic formula. For higher-order polynomials, such as are used for fitting higher order exponentials, an iterative, root-finding method is used (Newton-Raphson iteration: see Kreyszig 1983, pp. 764–66).

What if there are not two, real roots? Recall that the quadratic formula either yields two real roots, one real root, or two complex roots. This corresponds geometrically to two crossings of the X axis, one tangential “touch” of the axis, or no crossings of the axis.

If there is one real root, then the data being fit only consisted of a single exponential. In this case, this technique would yield two taus with the same value as that of the single exponential, and with amplitudes each one-half of the amplitude of the single exponential.

If there are two complex roots, the data being fit is not a pure exponential. Rather, it is the product of an exponential and a harmonic function (for example, a cosine). This function is commonly called a ringing response, or an exponentially-damped cosine. This can be seen by substituting a complex number into Equations 15a and b (*a±bi*), rewriting the resulting number in terms of a complex exponential ($re^{i\theta}$), where *r* is $\sqrt{(a+1)^2 b^2}$ and θ is *tan⁻¹⁄ (b(a + 1))*, taking the logarithm, substituting back into Equation 11, and simplifying.

Finding the Taus in the Presence of Noise

In the presence of noise, $f(t_j)$ is not an exact sum of exponentials, and therefore the Chebyshev coefficients d_j, $d1_j$ and $d2_j$ do not lie along a straight line, but are scattered:

$$d_j + x_1 d1_j + x_2 d2_j \cong 0$$

To find the best line through the data, we form the following regression equation, and minimize the χ^2 value:

$$\chi^2 = \sum_{j=2}^{n} (d_j - x_1 d1_j - x_2 d2_j)^2$$

The best values for x_1 and x_2 are determined by expanding this relation, minimizing it first with respect to x_1, then with respect to x_1. After rearranging we have the following set of simultaneous equations:

$$\sum_{j=2}^{n} d_j d1_j = x_1 \sum_{j=2}^{n} d1_j^2 + x_2 \sum_{j=2}^{n} d1_j d2_j$$

$$\sum_{j=2}^{n} d_j d2_j = x_1 \sum_{j=2}^{n} d1_j d2_j + x_2 \sum_{j=2}^{n} d2_j^2$$

Direct solution of simultaneous equations is a well-known problem in mathematics; an iterative matrix technique is used here (Gauss-Seidel iteration, in Kreyszig, pp. 810–12. This technique has extremely fast convergence, particularly in the sum of *m* exponentials case). This allows us to easily solve for the more difficult case of finding the solution to a set of *m* simultaneous equations, which must be used when fitting the sum of *m* exponentials.

Finding the Amplitudes of the Two Exponentials

Once the taus of the sum of exponentials are known, a technique similar to the single-exponential, noise present, case is used to find the amplitudes and the offset. (The corresponding two-exponential case without noise is not shown.) We generate two exponential datasets based on the values of τ_1 and τ_2 just calculated; both datasets have unity amplitude and zero offset:

$$g_{\tau_1}(t_j) = 1e^{-t_j/\tau_1} + 0 \quad \text{and} \quad g_{\tau_2}(t_j) = 1e^{-t_j/\tau_2} + 0$$

Recalling from Equation 12 that the transform of f is the scaled transform of $g_{\tau1}$ and $g_{\tau2}$, the resulting coefficients of each of these datasets is scaled and added together. In the presence of noise, however, this relationship is not exactly true. Rather:

$$d_j(f) \cong a_1 d_j^{\tau_1}(g_{\tau_1}) + a_2 d_j^{\tau_2}(g_{\tau_2}) \qquad \text{for } j > 0$$

Linear regression of this equation yields the best possible values of a_1 and a_2 that satisfy:

$$\chi^2 = \sum_{j=1}^{n} \left(d_j(f) - a_1 d_j^{\tau_1}(g_{\tau_1}) - a_2 d_j^{\tau_2}(g_{\tau_2}) \right)^2$$

Solution of this equation is not shown, but involves expanding the square inside the summation, minimizing first with respect to a_1 and a_2, and solving the resulting simultaneous set of equations for a_1 and a_2.

Finding the Offset

Finally, to find the offset, a formula similar to Equation 8 is used; no regression is needed:

$$a_0 = d_0(f) - a_1 d_0^{\tau_1}(g_{\tau_1}) - a_2 d_0^{\tau_2}(g_{\tau_2})$$

Fitting the Sum of Three or More Exponentials

Fitting the sum of three or more exponentials is a simple extension of the fitting the sum of two exponentials case. A full description is not given here.

Speed of Fitting

In tests the Chebyshev technique fit speed was completely unaffected by noise. There was a slight dependence on the number of exponentials being fit.

In contrast to these results, with iterative techniques, fitting a sum of two exponentials usually requires twice as much time, fitting a sum of three exponentials requires three times as much time, and so on.

Goodness of Fit

In tests comparing the Chebyshev method to the Simplex iterative search method, both yielded the same values of the fit parameters in low-noise and no-noise conditions. The tests added varying amounts of noise to exponentials generated from known values. As the noise levels increased, these two methods produced slightly different values of the fit parameters for each test case, although the average of the parameters was the same. At extremely high levels of noise (the peak-to-peak noise reached 30% of the peak of the exponential), the Chebyshev search clearly did not fit as well as the Simplex method, in those times that the iterative Simplex converged at all.

Like all fitting methods, the Chebyshev method fits sums of exponentials data best if the dataset spans several times the largest time constant in the exponential. Although the Chebyshev method consistently outperforms other iterative techniques in this regard, even it can generate incorrect fits in this situation (for example trying to fit an exponential function with a time constant of 2000 ms to a dataset spanning just 10 ms!) In particular, as the amount of noise increases, its ability to fit an insufficiently sampled dataset decreases.

The Chebyshev method performs most poorly when fitting data with extremely low-frequency signals present. This may occur under the following circumstances: 60 Hz noise present (or other low-frequency noise) or insufficiently averaged data (as may occur by forming a macroscopic current from an insufficient number of single-channel records: for example the single channel events may still be seen). This occurs since low-frequency noise will appear most strongly in the low-index Chebyshev coefficients, the same coefficients that contain most of the information of the exponential. Although iterative techniques do not perform well in this case either, when they converge they do so to the correct result more often than the Chebyshev technique.

Success of Fitting

The Chebyshev technique very rarely fails to find a fit to the data when fitting exponentials or sums of exponentials, as it uses mathematical relationships and linear regression techniques. Experimentally though, it sometimes fails to find a good fit (see above conditions) and can even fail altogether with some kinds of data (for example, if the x and y data values are identical). In particular, since the Chebyshev technique always finds an answer so quickly, it is tempting to assume that this answer is always correct. Be sure to always compare the fitted data to the dataset.

The Chebyshev method can fail when fitting shifted Boltzmann or exponential power functions, although the failure in these cases is expected to be very rare. In the event of a failure, all function parameters will reported as zero.

Fitting Sums of Exponentials to Non-Evenly Spaced Datasets

At present, the Chebyshev method only works with datasets having equally spaced points, since the relations derived here depend upon that assumption. With datasets that were sampled at two different rates (split-clock acquisition), you must first use the **Analyze > Interpolation** command to fill in the missing data points using linear interpolation.

Note: At the time of this writing the Chebyshev search method had not been published in a peer-reviewed journal. While we have conducted many empirical tests that have confirmed the accuracy of the fits, you should occasionally compare Chebyshev-fitted results to those of one or more of the other fitting methods.

Maximum Likelihood Estimation

Data are fitted by maximizing the logarithm of the likelihood with respect to a set of fitting parameters. Exponentially distributed data (t_i) are described by one or more components (k) of the form:

$$F(t_i) = \sum_{j=1}^{k} a_j e^{-t_i/\tau_j}$$

where t_1, t_2,. . . t_n are the n measured data points and τ_j is the time constant of the jth component. Each a_j is a fraction of the total number of events represented by the jth component where:

$$\sum a_j = 1.0$$

The probability density function *f(t)* is obtained by taking the first derivative of Equation 1 with respect to t_i, where, for each t_i:

$$f(t_i) = \frac{d}{dt_i}F(t_i) = \sum_{j=1}^{k} a_j \tau_j e^{-t_i/\tau_j}$$

The likelihood (L) of obtaining a given set of observed data t_i, given the form of the distribution and the set of function parameters (denoted here by ϑ) is the product of the probabilities of making each of the N observations:

$$L = \prod_{j=1}^{N} f(t_i|\theta)$$

As the likelihood typically takes on very small values the numerical evaluation of its logarithm is preferable. In practice, limited frequency resolution of the recording system makes it impossible to record events shorter than t_{min}. To generalize this case it is also assumed that no events longer than t_{max} can be measured. Taking these corrections into consideration, the conditional PDF is given by:

$$L(\theta) = \sum_{i=1}^{N} ln[f(t_i|\theta)/p(t_{min},t_{max}|\theta)]$$

where:

$$p(t_{min},t_{max}|\theta) = Prob(t_{min} \le t < t_{max})$$

is the probability that the experimentally measurable dwell-times fall within the range delimited by t_{min} and t_{max} given the probability distribution with parameters θ. In pCLAMP Software t_{min} and t_{max} are defined by the lower and upper limits of the data's time base.

The fitting algorithm used here is not, strictly speaking, a maximum likelihood estimation. It is an iteratively re-weighted least squares fit to the number of elements in a bin (which should converge on the maximum likelihood estimates of the parameters in this case). The weighting here has to do with the expected variance of the number of elements in a bin, which is Poisson distributed. So the weighting factor is the inverse of the variance (remember that mean = variance in a Poisson distribution). The iterative aspect has to do with the fact that the number of elements per bin gets moved around to correct for censoring for example, short events are not measurable). Because of the simple form of the "weighted" sum of exponentials (that is a different "weight"), the derivatives of the minimized function with respect to each parameter can be written directly and corrected for this Poisson-distributed variance.

Maximum Likelihood Estimation

The log likelihood L_b of observing a particular set of bin occupancies n_i for a given set of dwell-time data of times t_i is calculated by:

$$L_b(\theta) = \sum_{i=1}^{N} n_i \; ln\left\{\frac{F(t_{i+1}|\theta) - F(t_i|\theta)}{p(t_s, t_k|\theta)}\right\}$$

where $F(t_i)$ and $F(t_{i+1})$ are the probability distributions at the lower and upper bounds of the i^{th} bin using the parameter values θ, and $p(t_s, t_k)$ is the probability that the experimental dwell-times fall within the range (k) of the histogram where

$$p(t_s, t_k|\theta) = F(t_s|\theta) - F(t_k|\theta)$$

In pCLAMP Software, the probability distribution function (Equation 1), rather than the probability density function (Equation 2), is used in the calculations. *F(t)* is evaluated at each bin edge. The difference gives the probability that an event falls in the bin. This is equivalent to integrating the PDF over the width of the bin.

For a sum of *m* exponential components the probability distribution function is given by:

$$F(t|\theta) = 1 - \sum_{i=1}^{m} a_i e^{-t/\tau_i}$$

where θ is the entire set of coefficients comprising the fraction of the total number of events in each i^{th} component a_i and the time constants τ_i. The coefficients ai sum to unity, where:

$$\sum_{i=1}^{m} a_i = 1$$

Therefore, the set of parameters θ has 2*m*–1 degrees of freedom. pCLAMP Software can use either the Simplex or variable metric search methods to find the set of parameters that maximize $L_b(\theta)$. Only the variable metric method constrains the coefficients a_i to sum to unity. In the Simplex method, these parameters are not constrained and the set of parameters has 2*m* degrees of freedom.

Maximum likelihood will operate on either logarithmically binned or conventionally binned histograms. It is, however, recommended that logarithmically binned data be used if MLE is the fitting method. With conventional binning, it is often not possible to select a bin width that can represent the data satisfactorily if the time constants are widely spaced. This problem can be avoided by binning the data into the variable-width bins of a logarithmic histogram. For detailed discussion see Sigworth & Sine (1987) and Dempster (1993).

The EM Algorithm

The EM (Expectation step – Maximization step) algorithm computes maximum likelihood parameter estimates for binned observations from a mixture of exponentials. This algorithm, described in Dempster et al. (1977), is used in pCLAMP Software to estimate initial seed values for maximum likelihood estimates for fitting exponential probability functions.

Model Comparison

In general, with non-linear fitting routines, when the order of the fitting function is increased the fitted curve will appear to improve (up to a point, of course, until it becomes difficult to distinguish between successive order fits). Although it is often possible to choose one fit over another by visual inspection, this is not always the case. Moreover, visual inspection is not an adequate substitute for a statistical comparison, especially if two different models (for example different orders) produce fits that are visually very similar. pCLAMP Software provides a means of statistically comparing different models that have been fitted with either a least-squares routine or with maximum likelihood (Horn, 1987, Rao, 1973).

When model comparison is selected, the text “compare models” appears next to the function name above the equation window. If the selected function does not support model comparison, the model comparison status text is blanked out.

Fixed fitting function parameters are not allowed when comparing models.

Maximum Likelihood Comparison

Suppose you wish to compare two models, *F* and *G*, which have been fitted using the maximum likelihood method. The probability densities for these models are given by $f(x, \vartheta)$ and $g(x, \beta)$ where *x* is the dataset, and β and θ are the function parameters with dimensions *kf* and *kg* (where *kg* > *kf*). The natural logarithm of the likelihood ratio (*LLR*) for *F* and *G* is defined as:

$$LLR = log\left\{\frac{sup_{\beta}(x, \beta)}{sup_{\theta}(x, \theta)}\right\} = log\left\{\frac{g(x, \beta)}{f(x, \theta)}\right\}$$

where *β* and θ are the parameter values (maximum likelihood estimates) that maximize the likelihood for each probability density. The suprema of *f(x, ϑ)* and *g(x, β)* are denoted by $sup_{\vartheta}(x,\vartheta)$ and $sup_{\beta}(x,\beta)$, respectively. When model *F* is true, 2LLR has a Chi-square distribution with $k_g - k_f$ degrees of freedom (Rao, 1973; Akaike, 1974, Horn, 1987).

For example at a confidence level of 0.95 ($p < 0.05$) for $k_g - k_f = 2$ degrees of freedom (as is always the case between successive models) Chi-square is 5.991. Therefore, if 2LLR < 5.991 then it is assumed that model G does not represent a significant improvement over model *F* and model *F* is selected as the best fit.

Least Squares Comparison

In the case of least squares fitting (Simplex or Levenberg-Marquardt fitting methods) the SSE for models *F* and *G* is defined as:

$$SSE_f = \sum_{i=1}^{n} [x_i - f(x_i|\theta)]^2 \qquad \text{for model } F$$

and:

$$SSE_g = \sum_{i=1}^{n} [x_i - g(x_i|\beta)]^2 \qquad \text{for model } G$$

where x_i are the n data points, $f(x_i\,|\,\theta)$ and $g(x_i\,|\,\beta)$ are the values predicted by models *F* and *G*, respectively, and θ and β are the set of function parameters that minimize SSE.

To compare models *F* and *G* a form of the likelihood ratio test is used. In the case of least squares fitting the statistic *T* (Rao, 1973) is defined by

$$T = \frac{(SSE_f - SSE_g)}{SSE_g} \cdot \frac{(n - k_g)}{k_f}$$

where SSE_f and SSE_g are the sums of squared errors for models *F* and *G*, respectively. *T* has an *F*-distribution which can be compared with a standard *F*-distribution table, with k_f and $n-k_g$ degrees of freedom. This statistic is denoted in pCLAMP Software by “F”.

The degrees of freedom (k_f and $n-k_g$) will be different for each successive model.

Defining a Custom Function

The rules for defining a custom function and associated issues are as follows:

- Only one dependent variable is allowed. The dependent variable is already expressed, as is the equal sign, so these should not be entered as part of the equation. Only the right-hand side of the equation should be specified.
- When fitting from Analysis window data, only one independent variable is allowed. This variable must be expressed as x1 (or X1).
- When fitting from a Results window, up to six independent variables are allowed. These variables must be expressed as x1...x6 (or X1...X6).

- The maximum number of function parameters is 24. Parameter names must be expressed as p1...p24 (or P1...P24).
- The maximum length of the function including parenthesis is 256 characters.
- Parameter labels (*p*), independent variable labels (*x*) and mathematical operations such as log, sin, etc. are case-insensitive, so you may use either lower or upper case when specifying these labels or operations.
- Automatic seeding is not available for custom functions. You must specify all seed values before the fit will commence. If you try to commence fitting (by clicking OK) before all parameter seeds have been specified you will receive an error message.
- Graphically assisted seeding is not currently available when fitting to data from a Results window. If you absolutely require graphical seeding for fitting Results window data, you can first save the data to an ATF file, then import the ATF data into an Analysis window for graphical seeding and fitting. However, the data can contain only positive, uniformly spaced independent variable values (X axis data) for proper display in the Analysis window. Also, keep in mind that only a single independent variable can be specified for fitting from the Analysis window.

The custom function is compiled when switching from the **Function > Methods** tab to either of the other tabs in the fitting dialog. Once compiled successfully, the equation will appear on the line above the equation window in the fitting dialog. If there is an error in the expression, compiler warnings will be issued.

Be careful with parentheses. For example, be aware that 2*x1+p2 is not the same as 2*(x1+p2). In the former case 2*x1 is evaluated before p2 is added to the product, whereas in the latter case, x1+p2 is evaluated before the multiplication is performed.

Multiple-Term Fitting Models

A fitting model can be expressed as a sum of identical functions (terms). Each term in a multiple-term model will be assigned a “weight” or “amplitude” that will reflect the contribution of that term to the fitted curve (Schwartz, 1978). For example, a two-term standard exponential will be of the form

$$f(x) = A_1 e^{-t/\tau_1} + A_2 e^{-t/\tau_2} + C$$

where A_1 and f_1 are the amplitude and time constant, respectively, for the first term and A_2 and f_2 are the amplitude and time constant, respectively, for the second term. The variable *C* is a constant offset term along the Y axis.

Multiple terms for custom functions must be explicitly defined within the custom function itself, for example, the above two-term exponential function would be specified as

$$f(x) = p1 \cdot exp(-x1/p2) + p3 \cdot exp(-x1/p4) + p5$$

where *p*1 and *p*3 are the amplitudes, *p*2 and *p*4 are the time constants and *p*5 is the constant offset term.

Minimization Functions

Sum of Squared Errors

Levenberg-Marquardt, variable metric and Simplex only.

The function to be minimized is

$$SSE = \sum_{i=1}^{N} (y_i - y)^2 = \sum_{i=1}^{N} [y_i - f(x, P)]^2$$

where SSE (sum of squared errors) is the sum of the squares of the difference between the data y_i and the fitting function $y = f(x, P)$ with a set of parameters P to be fitted over N data points. The optimal values for P are assumed to occur when SSE is at a minimum. Weighting may or may not be applied to modify this function.

Maximum Likelihood Minimization

Variable metric and Simplex only.

Maximum likelihood estimation (MLE) is available only for the standard and log-transformed probability exponential functions and only with the variable metric or Simplex search methods. Moreover, these functions are intended to be used with binned data. That is, the dependent variable values (X axis data) are assumed to be bin center values.

Strictly speaking, the likelihood is maximized so the use of “minimization method” might be deemed inappropriate here. However, the fitting method minimizes the negative of the log likelihood value, which is equivalent to maximizing the positive log likelihood.

See Maximum Likelihood Minimization on page 232 for a description of the algorithm.

Mean Absolute Minimization

Levenberg-Marquardt and Simplex only.

Mean absolute minimization is a linear criterion that weights a badly-fitted data point proportionately to its distance from the fitted curve, rather than the square of that distance. If some points have substantially more error than others then the best sum of squares fit might deviate from the more reliable points in an attempt to fit the more reliable ones. The mean absolute error fit is influenced more by the majority behavior than by remote individual points. If the data are contaminated by brief, large perturbations, mean absolute error minimization might perform better than sum of squares minimization.

The function E to be minimized is

$$E = abs\left(\sum_{i=1}^{N} (y_i - y)\right) = abs\left(\sum_{i=1}^{N} [y_i - f(x, P)]\right)$$

Minimax Minimization

Levenberg-Marquardt and Simplex only.

Minimax minimization yields a fit in which the absolute value of the worst-fitted data point residual is minimized. This might be useful for certain datasets when the fit must match the data within a certain tolerance.

The function *E* to be minimized is

$$E = abs(max(y_i - y)) = abs(max[y_i - f(x, P)])$$

where max ($y_i - y$) is the largest absolute difference between the data and the fit.

Weighting

For the search methods that support least squares minimization (Levenberg-Marquardt, variable metric and Simplex)) the sum of the squares (SSE) of the difference between the data (f_i^{obs}) and fitted curve ($f_i(q)$) is minimized, where

$$SSE = \sum (f_i^{obs} - f_i(\theta))^2$$

This works quite well when uncertainties are random and not related to the time or number of counts. However, the uncertainty observed is often a function of the measured value y, such that for larger values of y there are substantially greater deviations from the fit than for smaller values of y. To compensate for this, weighting becomes quite useful during the assessment of the fit, where:

$$SSE = \sum \left[\left(\left(f_i^{obs} - f_i(\theta) \right)^2 / f(\theta) \right) \right]$$

The Levenberg-Marquardt method is the only fitting method that supports weighting. The SSE minimization function can be weighted in one of the following four ways:

None

In this case the denominator $f(\theta)$ is 1.

Function

Weighting the sum of squared errors function generates the Chi-square (χ^2) function. The (χ^2) value for a function with a given set of parameters ϑ is given by:

$$\chi^2 = \sum_{i=1}^{m} \frac{f_i^{obs} - f_i(\theta)}{f_i(\theta)}$$

where *m* is the number of points in the histogram fitting range, $f_i(\vartheta)$ is the fit function calculated for the ith data point, and f_i^{obs} is the observed value of the ith point.

Data

This is a modified form of the χ^2 function, where:

$$\text{Modified } \chi^2 = \sum_{i=1}^{m} \frac{f_i^{obs} - f_i(\theta)}{f_i^{obs}}$$

where *m* is the number of points in the histogram fitting range, $f_i(\vartheta)$ is the fit function calculated for the i^{th} data point, and fiobs is the observed value of the i^{th} point.

Bin Width

Weighting by the bin width weights the sum of squared errors by the width of each i^{th} histogram bin, such that:

$$SSE_b = \sum_{i=1}^{m} \frac{f_i^{obs} - f_i(\theta)}{(x_i - x_s)}$$

where x_i is the right bin edge value and x_s is the left bin edge value. Bin width weighting is allowed only for the predefined log-transformed exponential function that assumes log-binned data. It is not available for custom functions.

The selected weighting type is also applied to mean absolute and minimax minimization.

If you wish to apply different weighting criteria you can export the data to a results sheet, modify the data using column arithmetic and subsequently fit the data directly from the results sheet.

Normalized Proportions

The normalized proportion is the absolute value of that parameter divided by the sum total of the absolute values of all of the proportion terms, that is:

$$P_{norm_k} = \frac{P_{abs_k}}{\sum_{j=1}^{n} P_{abs_j}}$$

Normalized proportions are most likely to be specified with the exponential probability functions or the Exponential, weighted/constrained function. The variable metric method (and only this method) constrains the proportion terms to sum to 1.0 during fitting (see The Variable Metric Method on page 212), but only for the standard or log-transformed exponential probability functions when using maximum likelihood or the weighted/constrained exponential with least squares.

Zero-shifting

In cases where the natural origin for a dataset is ambiguous or inappropriate, it would be desirable to shift the origin of the data that are to be fitted to zero. For example, if a time constant of an exponential curve is to be extracted from an arbitrary segment of a large dataset it would be reasonable to force the selected fitting range to start at zero. In the same vein, it might also be desirable to set a zero origin just after a stimulus artifact. To this end, the “zero-shift” option is provided. If zero-shift is enabled then for a set of *i* data points x_i, each point *x* is offset such that $x = x_i – x_0$ where x_0 is the value of the first data point.

However, it is important to note that zero-shifting can affect the values of some fitted parameters. Consequently, in some instances, zero-shifting the data might not be appropriate and could lead to unexpected results. For example, when fitting a “Z-delta Boltzmann” to current/voltage data the parameter V_{mid} (the voltage at which the current is half-maximal) will differ depending on whether or not zero-shift is enabled. In the following example, the data were zero-shifted prior to fitting.

Figure 11-2 shows a fit of the Z-delta Boltzmann function to current/voltage data. The fitted curve is the line without symbols.

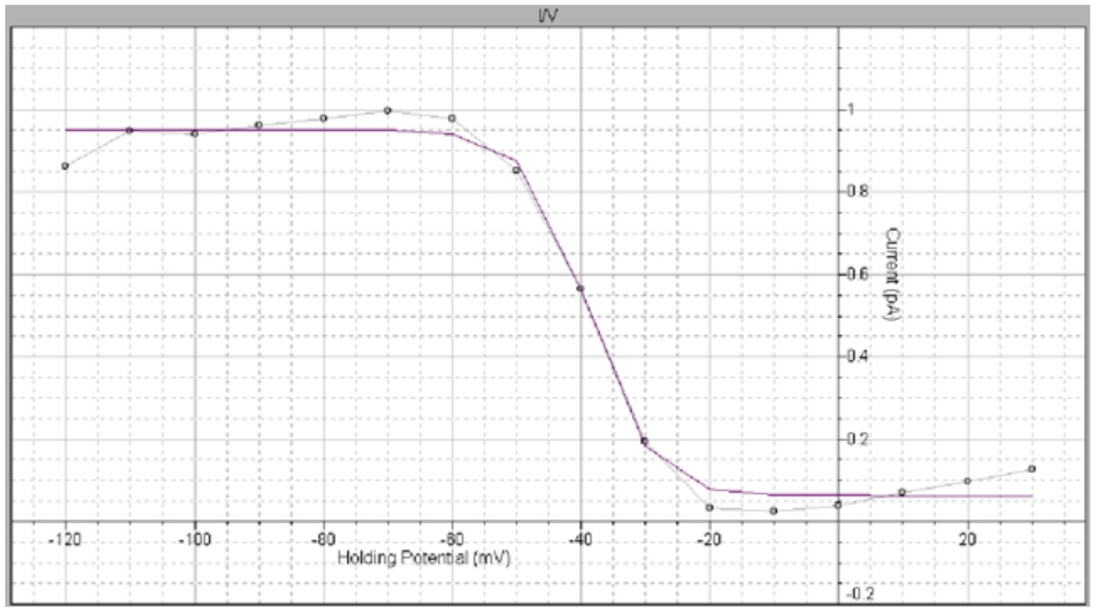


Figure 11-2: Fit of Z-delta Boltzmann function to current/voltage data.

In this fit V_{mid} is reported as +81.16 but from the data one would expect V_{mid} to be about –40 mV. Nevertheless, the value of +81.16 is in fact correct because the first point (originally at –120 mV) has been forced to zero such that the actual fitting range is from 0 to +140 mV. On this scale, the reported positive value is a reasonable estimate of V_{mid}.

If zero-shift is disabled, the fitted curve looks exactly the same but V_{mid} is now reported as –38.84 mV. The other parameters of this function are identical in both cases. As the expected value for this dataset is indeed in the range of –40 mV, it is clear that in this particular case it would not be appropriate to use zero-shifting. In fact, if zero-shift is inadvertently enabled it would appear that the fitting routine is malfunctioning, which is not the case. It is, therefore, very important that the status of the zero-shift option be known at all times when fitting.

Chapter 12: Fitting Functions

12

pCLAMP Software provides a large selection of predefined fitting functions to assist in data analysis. Some functions require that restrictions, such as forced positive parameters, are applied during the fit. Accordingly, some options in the Fit dialog might be automatically set to comply with mandatory fitting requirements. Furthermore, some functions require automatic preprocessing of the data, such as normalization, prior to fitting. In the function descriptions below, any automatic data preprocessing and forced option requirements for a function are listed on the line above the function formula.

The references cited provide sources for more detailed explanation of the functions and examples of specific experimental applications.

BETA Function

$$f(x) = x^{a-1}(1-x)^{b-1}/B(a,b) + C$$

$$B(a,b) = \int_0^1 x^{a-1}(1-x)^{b-1}dx$$

$$a = \alpha\tau, \qquad b = \beta\tau$$

- Requires a normalized X axis.
- All parameters except the offset (*C*) are forced positive.

The beta function is used to describe the steady state filtered amplitude distribution of a two-state process, where α and β are rate constants and τ is the time constant of a first-order filter. This method has been used to measure block and unblock rates in the microsecond time range even under conditions when individual blocking events were not time-resolved by the recording system (Yellen, 1984).

This function describes a probability distribution that requires the X axis data range to be 0 to 1. If the data do not meet this criterion the X axis values are automatically normalized prior to fitting. These values are rescaled after fitting so that the fitted curve will conform to the original data.

The beta function is intended to be used primarily with data that have been imported into a Results or Graph window, and therefore has limited utility with respect to time-based Analysis window data.

The fit solves for parameters *a*, *b*, *B(a,b)* and the constant *y*-offset *C*. The rate constants α and β can be obtained by dividing *a* and *b*, respectively, by the filter time constant, τ. pCLAMP Software does not perform this calculation.

The recommended fitting method is Levenberg-Marquardt.

Binomial

$$f(x) = \frac{n!}{(n-x)!x!}P^x(1-P)^{n-x}$$

- Requires an integral X axis.
- Requires normalized data.
- Maximum number of points = 170.
- The parameter P is forced positive.

The binomial distribution describes the probability, *P*, of an event occurring in *n* independent trials. For example, this function can be used to determine the probability for a particular number of n independent ion channels being open simultaneously in a patch (Colquhoun, *et al*. 1995). This function has also been applied to quantal analysis of transmitter release (Bekkers, *et al* 1995, Larkman, *et al* 1997, Quastel, 1997).

This function requires integer values of *x*. If the X axis values are not integers then they are converted to such prior to, and rescaled following, fitting. The ordinate data are also normalized so that they range from 0 to 1, and are rescaled after fitting so that the fitted curve conforms to the original data. Rescaling is such that the area under the fitted curve is equal to the area under the original data curve.

The number of sample points for this function is limited to 170 to conform to the computational limit for a factorial.

The binomial function has limited utility with respect to time-based Analysis window data. It is intended to be used primarily with data that have been imported into a Results or Graph window.

The fit solves for the probability variable, *P*. Since the X axis scale is integral the fitted curve will appear as a histogram.

The recommended fitting method is Levenberg-Marquardt.

Boltzmann, Charge-voltage

$$f(V) = \frac{I_{max}}{1 + e^{(V_{mid} - V)/V_c}} + C$$

This function can be used to examine activation and inactivation profiles of voltage-gated currents (Zhang, *et al*. 1995).

The charge-voltage Boltzmann distribution is given by

$$Q_{on}(E) = Q_{on-max} / [1 + exp((E_{mid} - E)/K)] + C$$

where Q_{on-max} is the maximum charge displaced, E_{mid} is the potential at which $Q_{on} = 0.5 \times Q_{on-max}$, K is the number of millivolts required to change Q_{on} e-fold, and E is the measured potential. C is a constant offset term. This function can be used to fit

current-voltage curves of the form:

$$I = I_{max} / [1 + exp((V_{mid} - V)/V_c)]$$

or

$$g = g_{max} / [1 + exp((V_{mid} - V)/V_c)]$$

where V is the membrane potential, V_{mid} is the membrane potential at which the current is half-maximal, and V_c is the voltage required to change I or g e-fold. If I or g are normalized then the data points should be input as I/I_{max} or g/g_{max} and the dependent variable I_{max} or g_{max} in the function above should be fixed to 1.0.

The fit solves for I_{max} (or G_{max}), V_{mid}, V_c and the constant y-offset C.

The recommended fitting method is Levenberg-Marquardt. The variable metric method also works well with this function but is slower to converge.

Boltzmann, Shifted

$$f(x) = \sum_{i=1}^{n} \frac{A_i}{1 + Be^{-x/\tau_i}} + C$$

Like the standard Boltzmann, this function also defines a sigmoidal curve. *A* is the amplitude, τ is the “slope” and *C* is a constant offset in the *y* direction (see Boltzmann, Standard on page 240). However, unlike the standard Boltzmann function, the shifted Boltzmann includes an offset parameter *B* that shifts the curve along the X axis such that the half-maximal amplitude is at $x = -ln(1/B) \cdot \tau$. Thus, this function is to be used when fitting a sigmoidal curve to data where the half-amplitude point is expected to be displaced from zero along the X axis.

The fit solves for *A*, *B*, τ and the constant y-offset *C* for each component *i*.

The recommended fitting method is Levenberg-Marquardt or Chebyshev if fitting only a single term.

Boltzmann, Standard

$$f(x) = \sum_{i=1}^{n} \frac{A_I}{1 + e^{-x/\tau_i}} + C$$

This function defines a sigmoidal curve. Unlike the shifted Boltzmann function, this function does not include an offset parameter along the X axis.

The physical correlates for *A* and *x* are not specified in this equation. It is up to the user to define these quantities. For example, *A* might be conductance and *x* might be cell membrane voltage (Bähring, *et al*. 1997). The parameter τ is the slope of the function that specifies the change in *x* required to produce an e-fold change in *A*. *A* is half-maximal at *x* = 0 (where $f(x) = A / (1 + e^{-0/\tau}) = A/2$). Consequently, the fitted curve is sigmoidal only if there are both positive and negative *x* data with the half-amplitude at or very close to zero. If the half-amplitude is offset significantly from zero the shifted Boltzmann function should be used.

The fit solves for the amplitude *A*, the width τ and the constant y-offset *C* for each component *i*.

The recommended fitting method is Levenberg-Marquardt.

Boltzmann, Z-Delta

$$f(V) = V_{min} + \frac{V_{max} - V_{min}}{1 + e^{\frac{Z_d F}{RT}\langle V - V_{mid}\rangle}}$$

This function can be used to analyze the voltage dependence of gating charges in ion channels (Hille, 1992).

V_{min} and V_{max} are the minimum and maximum voltages, Z_d is the magnitude of the charge valence associated with the electric field *V*, V_{mid} is the voltage at which *f(V)* is halfmaximal, *F* is the Faraday constant, *R* is the Gas constant, *T* is absolute temperature. The temperature is optionally specified (in °C).

The fit solves for V_{max}, V_{min}, V_{mid} and the constant *y*-offset *C*.

The recommended fitting method is Levenberg-Marquardt.

Current-Time Course (Hodgkin-Huxley)

$$f(t) = I'(1 - e^{-t/\tau_j})^a (k_\infty - (k_\infty - 1)e^{-t/\tau_k})^b + C$$

This is the Hodgkin-Huxley model for describing the time course of voltage-dependent ionic membrane currents. This equation was originally used to describe voltage-activated sodium and potassium currents (with *a* = 3 and *b* = 1). The term *k*• is the steady-state inactivation, *I´* is the maximum current that is achieved in the absence of inactivation, τ_j is the activation time constant, τ_k is the inactivation time constant, and the power terms a and *b* are empirically determined (Dempster, 1993, pages 140–142).

The fit solves for *I´*, τ_j, τ_k, *k*• and the constant *y*-offset *C*. The power terms a and b must be optionally specified.

The recommended fitting method is Levenberg-Marquardt.

Exponential, Alpha

$$f(t) = \sum_{i=1}^{n} A_i t e^{-t/\tau_i} + C$$

The alpha exponential function has been used to describe temporal responses at the neuronal soma to synaptic input (Gerstner, et al. 1992 and Gerstner, et al. 1993).

The fit solves for the amplitude *A*, the time constant τ and the constant *y*-offset *C* for each component *i*.

The recommended fitting method is Levenberg-Marquardt.

Exponential, Cumulative Probability

$$f(t) = \sum_{i=1}^{n} P_i(1 - e^{-t/\tau_i}) + C$$

This function fits data that have been binned cumulatively. That is, each successive bin contains its own data plus the data in all of the preceding bins.

This function should not be used for binned data because cumulative binning creates artificial correlations between successive bins. The correlation occurs because each successive bin contains all of the data in the preceding bins. The cumulative exponential function provides meaningful results only if the data values are not correlated.

The fit solves for the proportion (amplitude) *P*, the time constant τ and the constant *y*-offset *C* for each component *i*.

The recommended fitting method is Levenberg-Marquardt.

Exponential, Log Probability

$$f(t) = \sum_{i=1}^{n} P_i e^{[ln(t) - ln(\tau_i)]e^{ln(t) - ln(\tau_i)}}$$

- Can only be used with Results or Graph window data.
- The dwell-time data (*t*) must be input as $log_{10}(t)$.

This function describes dwell-time data, usually from single channel experiments, that have been binned on a logarithmic time scale. Logarithmic binning is often preferable to conventional linear binning because of its superior resolution of widely spaced time constants (Sigworth & Sine, 1987). Histograms can be imported from pSTAT or the QUB module MIL.

The fit solves for the proportion (amplitude) *P*, the time constant τ and the constant *y*-offset *C* for each component i.

The recommended fitting method is variable metric with maximum likelihood estimation.

Exponential, Power

$$f(t) = \sum_{i=1}^{n} A_i(1 - e^{-t/\tau_i})^a + C$$

The fit solves for the amplitude *A*, the time constant τ and the constant *y*-offset *C* for each component *i*. The power term *a* must be optionally specified.

The recommended fitting method is Levenberg-Marquardt or Chebyshev if fitting only a single term (Chebyshev can solve for a single term only).

Exponential, Probability

$$f(t) = \sum_{i=1}^{n} P_i \tau_i^{-1} e^{-t/\tau_i} + C$$

This function can be used to fit single channel dwell time distributions that have not been converted to Log duration. For each component of the distribution, the fit solves for the proportion *P*, the time constant τ and the constant *y*-offset *C* for each component *i*.

The recommended fitting method is Levenberg-Marquardt. Maximum likelihood estimation can also be used with either the variable metric or Simplex fitting methods, but convergence will be slower.

Exponential, Product

$$f(t) = \sum_{i=1}^{n} A_i \left(1 - e^{-t/\tau_{r_i}}\right)\left(e^{-t/\tau_{d_i}}\right) + C$$

This function can be used to fit postsynaptic events (excitatory or inhibitory postsynaptic potentials). The fit solves for the amplitude *A*, the rise time constant τ_r and the decay time constant τ_d for each component *i*.

The recommended fitting method is Levenberg-Marquardt.

Exponential, Sloping Baseline

$$f(t) = \sum_{i=1}^{n} A_i e^{-t/\tau_i} + mc + C$$

This function is used to fit an exponential of the standard form to data that are superimposed on a sloping baseline, for example resulting from a constant baseline drift.

The fit solves for the amplitude *A*, the time constant τ for each component *i*, and the common parameters, the slope *m* and constant *y*-offset *C* for each component *i*.

The recommended fitting method is Chebyshev.

Exponential, Standard

$$f(t) = \sum_{i=1}^{n} A_i e^{-t/\tau_i} + C$$

This is the most basic function used to fit changes in current or voltage that are controlled by one or more first-order processes. The fit solves for the amplitude *A*, the time constant τ, and the constant *y*-offset *C* for each component *i*.

The recommended fitting method is Chebyshev.

Exponential, Weighted

$$f(t) = K_0\left(\sum_{i=1}^{n} f_i e^{-K_i t}\right) + C$$

This function is identical to the constrained exponential function except that the sum of the f_i components is not constrained to 1.

The fit solves for the proportion (amplitude) *f*, the rate constant *K*, the "weight" K_0 and the constant *y*-offset *C* for each component *i*.

The recommended fitting method is Levenberg-Marquardt.

Exponential, Weighted/Constrained

$$f(t) = K_0\left(\sum_{i=1}^{n} f_i e^{-K_i t}\right) + C \qquad \text{where} \qquad \sum_{i=1}^{n} f_i = 1.0$$

- Requires the variable metric fitting method.

This function has been used to describe the recovery rate of ground-state absorption following photo-excitation of intercalated metal complexes bound to DNA (Arkin, *et al.* 1996).

The fit solves for the proportion (amplitude) *f*, the rate constant *K*, the weight K_0 and the constant *y*-offset *C* for each component *i*. The f_i terms sum to 1.0.

The fitting method must be variable metric.

Gaussian

$$f(t) = \sum_{i=1}^{n} A_i \frac{e^{-(x-\mu_i)^2 / (2\sigma_i^2)}}{\sigma_i \sqrt{2\pi}} + C$$

This is for data that can be described by one or more normal distributions. For *n* components, the fit solves for the amplitude *A*, the Gaussian mean μ, the Gaussian standard deviation σ and the constant *y*-offset *C* for each component i.

This function is generally used for describing amplitude distributions of single channel events (Heinemann, 1995).

The recommended fitting method is Levenberg-Marquardt.

Goldman-Hodgkin-Katz

$$f(x, y, z) = \frac{RT}{F} ln \frac{[X]_1 + \alpha[Y]_1 + \beta[Z]_1}{[X]_2 + \alpha[Y]_2 + \beta[Z]_2} \qquad \begin{aligned} \alpha &= (pY)/(pX) \\ \beta &= (pZ)/(pX) \end{aligned}$$

- This function can only be used with Results window data.

This function is used to describe the steady-state dependence of membrane voltage on ion concentrations and the relative permeability of those ions through the membrane.

The equation assumes that all the ions are monovalent. For positive ions $[X]_1$ refers to the concentration outside the membrane and $[X]_2$ refers to the intracellular concentration.

For negative ions $[X]_2$ refers to the concentration outside the membrane and $[X]_1$ refers to the intracellular concentration.

The fit solves for the permeability ratios α and β.

The recommended fitting method is Levenberg-Marquardt.

Goldman-Hodgkin-Katz, Extended

$$f(x, y, z^{2+}) = \frac{RT}{F} ln \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$a = [X]_2 + 4\beta[Z]_2 + \alpha[Y]_2$$
$$b = [X]_2 - [X]_1 + \alpha([Y]_2 - [Y]_1)$$
$$c = -[X]_1 - \alpha[Y]_1 - 4\beta[Z]_1$$

$$\alpha = pY/pX \qquad \beta = pZ/pX$$

- This function can only be used with Results window data.

This function is used to describe the steady-state dependence of membrane voltage on ion concentrations and the relative permeability of those ions through the membrane.

However, this formulation extends the Goldman-Hodgkin-Katz relationship to include the effect of a divalent ion such as calcium or magnesium. By measuring the dependence of resting potential or reversal potential on varying concentrations of the relevant monovalent and divalent ions, this equation can be used to calculate the relative permeability of the activated conductance(s) to calcium compared to sodium, potassium, or other experimental ions (Piek, 1975; and Sands, *et al*. 1991).

The fit solves for the permeability ratios α and β.

The recommended fitting method is Levenberg-Marquardt.

Hill (4-parameter Logistic)

$$f(x) = I_{min} + \frac{I_{max} - I_{min}}{1 + (C_{50}/[x])^h}$$

This is a modified form of the Hill equation that is useful for fitting dose-response curves. The half-maximal concentration is determined directly. I_{min} refers to the baseline response, I_{max} refers to the maximum response obtainable with drug *x*, C_{50} is the concentration at half-maximal response (inhibitory or excitatory) and *h* is the Hill slope.

The fit solves for I_{min}, I_{max}, C_{50} and *h*.

The recommended fitting method is Simplex.

Hill, Langmuir

$$f(x) = \sum_{i=1}^{n} \frac{I_{max_i}[x]^{h_i}}{C_{50_i}^{h_i} + [x]^{h_i}} + C$$

The Langmuir-Hill equation allows for fitting a sum of Langmuir components. It is useful for fitting data where non-specific binding of the agonist is linear. I_{max} refers to the maximum response obtainable with drug x, C_{50} is the concentration at half-maximal response (inhibitory or excitatory) and h is the Hill slope. C is a constant y-offset.

The fit solves for I_{max}, C_{50}, h and C.

The recommended fitting method is Simplex.

Hill, Steady State

$$f(S) = \frac{V_{max}[S]^n}{K^n + [S]^n} + C$$

This is a general equation that can be applied to many kinds of concentration-dependent pharmacological or ion channel responses. V_{max} refers to the maximum response obtainable with drug S. By definition a partial agonist will have a V_{max} value that is less than the V_{max} of a full agonist. K is indicative of potency but it is not equal to the concentration at half-maximal velocity except when $n = 1$. The value of n places some limitations on the degree of cooperativity of the ligand-dependent processes. In order to define a concentration-dependent inhibitory process, n can be seeded with a negative value.

The fit solves for V_{max}, K and n.

The recommended fitting method is Simplex.

Lineweaver-Burk

$$f(S) = \frac{K_m}{V_{max}[S]} + \frac{1}{V_{max}} \qquad \text{where } S = 1/x$$

This equation, derived by taking the reciprocal of both sides of the Michaelis-Menten equation, describes a straight line with a slope of K_m/V_{max} and a y-intercept of $1/V_{max}$. It is useful for gaining information on enzyme inhibition (Lehninger 1970, page157).

The fit solves for K_m and V_{max}.

The recommended fitting method is Levenberg-Marquardt.

Logistic Growth

$$f(X) = \frac{R_{max}}{1 + Ae^{-Bx}} + C$$

This function describes an exponential growth that is subject to saturation that occurs at the limiting value R_{max}. The parameter A is the number of times the initial value must grow to reach R_{max}, and B determines the rate and direction of growth. The function will increase when B is positive and decrease when B is negative.

The fit solves for Rmax and A, B and the constant y-offset C.

The recommended fitting method is Levenberg-Marquardt.

Lorentzian Distribution

$$f(x) = \sum_{i=1}^{n} \frac{2A_i\omega}{4\pi(x-\mu)^2 + \omega^2} + C$$

The Lorentzian distribution function is generally used to characterize energy transition spectra that exhibit homogenous broadening of the peak. For example, the natural line shape in spectroscopy can be characterized by this function. The peak of a spectral line is narrow but broadens as a result of uncertainties in the energy level of the excited state. It, nevertheless, retains the Lorentzian line shape.

The fit solves for the area A under the curve, the half-width ω of the peak, the X axis center μ of the peak (generally the center frequency) and the constant y-offset C for each component i.

The recommended fitting method is Levenberg-Marquardt.

Lorentzian Power 1

$$S(f) = \sum_{i=1}^{n} \frac{S(O)_i}{1 + (f/f_{c_i})^2}$$

The power spectra of current fluctuations can be described by the sum of one or more Lorentzian functions of this form.

The time constant is related to the cutoff frequency, f_c, at which the power has declined to $S(0)/2$ by

$$\tau = \frac{1}{2\pi f_c}$$

(Dempster 1993 pages 196–197; Stevens 1981).

The fit solves for $S(0)$ and f_c for each component i.

The recommended fitting method is Levenberg-Marquardt.

Lorentzian Power 2

$$S(f) = \sum_{i=1}^{n} \frac{S(O)_i}{1 + (2\pi f \tau_i)^2}$$

The power spectra of current fluctuations produced by ion channels can be described by the sum of one or more Lorentzian functions of this form where:

$$\tau = \frac{1}{1/\tau_o + 1/\tau_c}$$

If the probability of the channel being open is low relative to the probability of the channel being closed then the channel closed time constant τ_c can be ignored and τ can be equated to the channel open time constant τ_o. At low frequencies the function tends toward *S*(*0*). At high frequencies the spectrum decays in proportion to f^2 (Dempster, 1993 and Stevens, 1981).

The fit solves for *S*(*0*), τ and the constant *y*-offset *C* for each component *i*.

The parameter τ has units of *s* if the frequency, ƒ, is in Hz.

The recommended fitting method is Levenberg-Marquardt.

Michaelis-Menten

$$f(S) = \frac{V_{max}[S]}{[S] + K_m} + C$$

This is a general equation to describe enzyme kinetics where [*S*] refers to the concentration of substrate. In this equation, V_{max} refers to rate of catalysis when the concentration of substrate is saturating. K_m refers to the Michaelis constant ($(k_{-1} + k_2)/k_1$) where k_1 and k_{-1} are forward and backward binding rates of the enzyme-substrate complex, respectively, and k_2 is the first-order rate constant for the formation of product from the bound enzyme-substrate complex.

The fit solves for V_{max}, K_m, and the constant *y*-offset *C*.

Nernst

$$V(x) = \frac{RT}{zF} ln \frac{[x]_1}{[x]_2}$$

This function describes the condition where an equilibrium exists between the energy associated with membrane voltage (*V*) and the energy associated with a concentration gradient of an ion species *x*. Hence, the Nernst potential for a given ion is often also referred to as the equilibrium potential for that ion.

The fit solves for the concentration $[x]_2$ given a series of concentrations $[x]_1$.

Parabola, Standard

$$f(x) = Ax^2 + Bx + C$$

This is the parabolic function.

The fit solves for the parameters *A*, *B* and the constant *y*-offset *C*.

Parabola, Variance-Mean

$$f(x) = ix - x^2/N$$

This is a form of the parabolic function used to describe synaptic event data where *i* is the unitary synaptic current amplitude and *N* is the number of release sites (Clements & Silver, 2000).

The fit solves for the parameters *i*, and *N*.

Poisson

$$f(x) = \frac{e^{-\lambda}\lambda^x}{x!}$$

- Requires an integral X axis.
- Requires normalized data.
- Maximum number of points = 170.
- The data will be automatically zero-shifted.

The Poisson distribution describes the probability of getting x successes with an expected, or average number of successes denoted by λ. This is similar to the binomial function but is limited to cases where the number of observations is relatively small.

This function requires integer values of *x*. If the *X* axis values are not integers then they are converted to such prior to, and rescaled following, fitting. The ordinate data are also normalized so that they range from 0 to 1, and are rescaled after fitting so that the fitted curve conforms to the original data. Rescaling is such that the area under the fitted curve is equal to the area under the original data curve.

The number of sample points for this function is limited to 170 to conform to the computational limit for a factorial.

The fit solves for the λ given a series of observed probability values *x*. Since the *X* axis scale is integral the fitted curve will appear as a histogram.

The recommended fitting method is Levenberg-Marquardt.

Polynomial

$$f(x) = \sum_{i=0}^{n} a_i x^i$$

The fit solves for the polynomial coefficients a_i. The term a_0 always exists in the solution. A first-order (or "one term") polynomial is, therefore, given by $f(x) = a_0x^0 + a_1x^1 = a_0 + a_1x$, which is a straight-line fit.

The maximum order is 6.

Straight Line, Origin at Zero

$$f(x) = ix$$

This function is used to fit variance-mean data (V-M analysis) to estimate the unitary current, *i*. The fit solves for the slope *i*, forcing the origin to zero.

Straight Line, Standard

$$f(x) = mx + b$$

The straight-line fit solves for the slope *m* and the *y*-intercept *b*.

Voltage-Dependent Relaxation

$$f(V) = \frac{1}{a_0 e^{V/\alpha} + b_0 e^{-V/\beta}} + C$$

This function describes the relaxation kinetics for a two-state voltage-dependent process. The forward and reverse rate constants are α and β, respectively. The term a_0 is value of α at $V = 0$ and b_0 is value of β at $V = 0$.

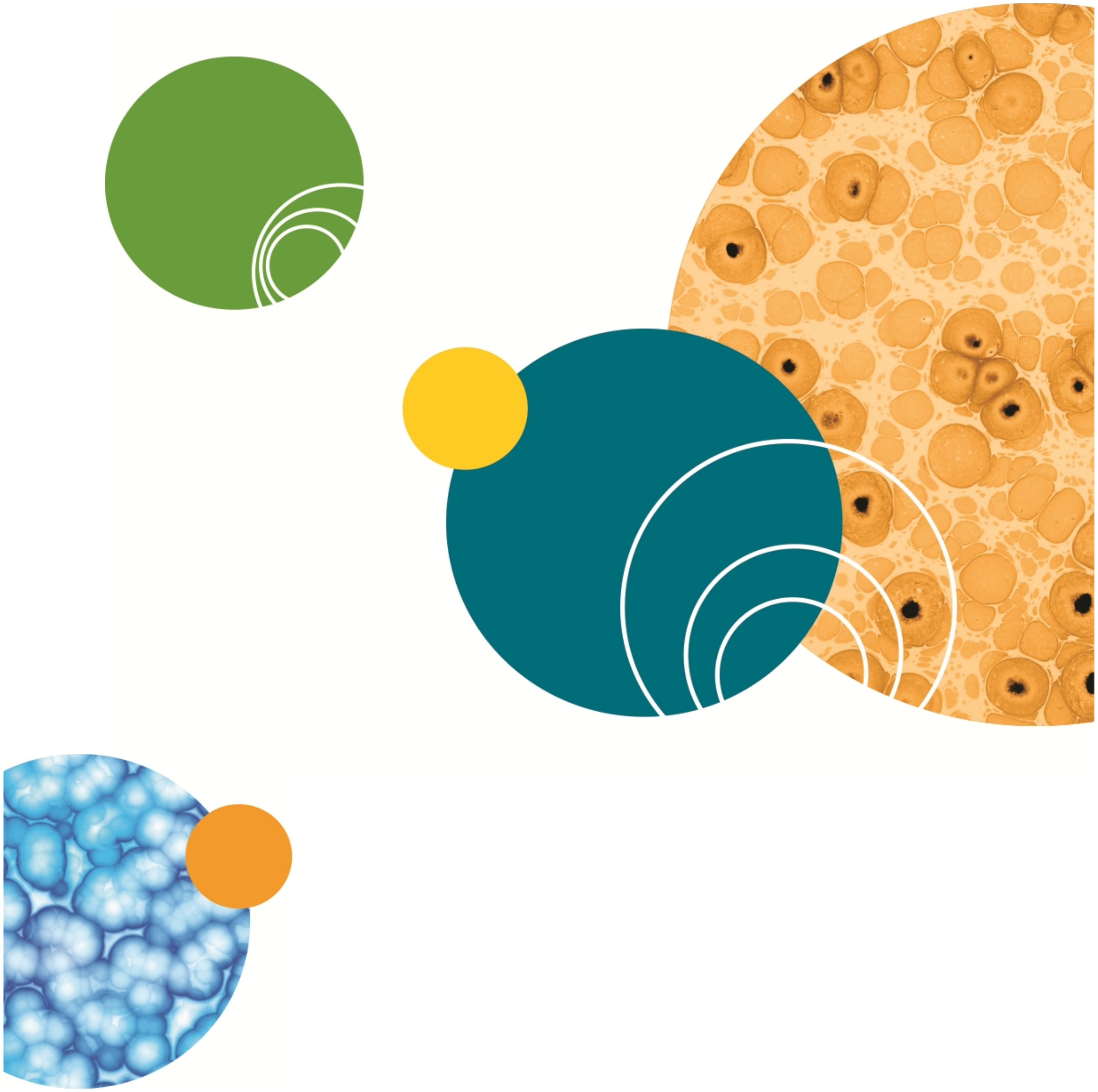
The fit solves for α, β, a_0, b_0 and the constant *y*-offset *C*.

The recommended fitting method is Levenberg-Marquardt.

Constants

F = Faraday constant = 9.648456×10^4 C/mol.

R = gas (Rydberg) constant = 8.31441 J/mol-deg K



Appendix A: References

A

Primary Sources

Abramowitz, M. and Stegun, I.A., eds. *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*. Dover Publications, New York, 1972.

Akaike, H. “A new look at statistical model identification.” *IEEE Transactions on Automatic Control* AC-19, 1974.

Arkin, M.R., Stemp, E.D.A., Holmlin, R.E., Barton, J.K, Hörmann, A., Olson, E.J.C. and Barbara, P.F. “Rates of DNA-mediated electron transfer between metallointercalators.” *Science* 273, 475–480, 1996.

Bähring, R., Bowie, D., Benveniste, M. and Mayer, M.L. “Permeation and block of rat GluR6 glutamate receptor channels by internal and external polyamines.” *J. Physiol*. 502, 575–589, 1997.

Bekkers, J.M. and Stevens, C.F. “Quantal analysis of EPSCs recorded from small numbers of synapses in hippocampal cultures.” *J. Neurophysiol*. 73, 1145–56, 1995.

Clements, J.D. and Bekkers, J.M. “Detection of Spontaneous Synaptic Events with an Optimally Scaled Template.” *Biophysical Journal* 73, 220–229, 1997.

Clements, J.D. and Silver, R.A. “Unveiling Synaptic Plasticity: A New Graphical and Analytical Approach.” *Trends in Neurosciences* 23, 105–113, 2000.

Colquhoun, D. and Hawkes, A.G. “The Principles of the Stochastic Interpretation of Ion-Channel Mechanisms” in: Single-Channel Recording, Second Edition, eds. Sakmann, B. and Neher, E. Plenum Press, New York. 432, 1995.

Colquhoun, D and Sigworth, F.J. “Fitting and Statistical Analysis of Single-Channel Records.” Chap. 19 in *Single-Channel Recording*, Second Edition, eds. Sakmann, B. and Neher, E. Plenum Press, New York, 1995.

Dempster, A.P., Laird, N.M. and Rubin, D.B. “Maximum likelihood from incomplete data via the EM algorithm.” *J. R. Statist. Soc.* B, 39,1–38, 1977.

Dempster, J. *Computer Analysis of Electrophysiological Signals*. Biological Techniques Series. Academic Press, London, 1993.

Donaldson, J.R. and Tryon, P.V. The Standards Time Series and Regression Package National Institute of Standards and Technology (formerly the National Bureau of Standards), Internal Report *NBSIR* 86–3448, 1990.

Frazier, J.L. and Hanson, F.E. “Electrophysiological recording and analysis of insect chemosensory responses” in: *Insect-Plant Interactions*, eds. Miller, J.R. and Miller, T.A. Springer Verlag, New York, pp. 285–330, 1986.

Geddes, L.A. *Electrodes and the Measurement of Bioelectric Events*. John Wiley and Sons, New York, 1972.

Gerstner, W., and van Hemmen, J.L. “Associative Memory in a Network of ‘Spiking’ Neurons.” *Network* 3, 139–164, 1992.

Gerstner, W., Ritz, R., and van Hemmen, J.L. “A Biologically Motivated and Analytically Soluble Model of Collective Oscillations in the Cortex: I. Theory of Weak Locking.” *Biol. Cybern*. 68, 363–374, 1993.

Heinemann, S.H. “Guide to Data Acquisition and Analysis” in: *Single-Channel Recording*, Second Edition, eds. Sakmann, B. and Neher, E. Plenum Press, New York. pp. 68–69, 1995.

Hille, B. *Ion Channels of Excitable Membranes*, Second Edition. Sinauer Associates, Massachusetts 1992.

Horn, R. “Statistical Methods for Model Discrimination: Applications to Gating Kinetics and Permeation of the Acetylcholine Receptor Channel.” *Biophys.J*. 51, 255–263, 1987.

Kaissling, K.E. “Pheromone deactivation catalyzed by receptor molecules: a quantitative kinetic model.” *Chem. Senses* 23:385–395, 1998.

Kaissling, K.E. and Thorson, J.T. “Insect olfactory sensilla: structural, chemical and electrical aspects of the functional organization” in: *Receptors for Neurotransmitters, Hormones and Pheromones in Insects*, eds. Satelle, D.B., Hall L.M., Hildebrand, J.G. Elsevier/North-Holland Biomedical Press, Amsterdam, pp. 261–282, 1980.

Kreyszig, E. *Advanced Engineering Mathematics*, 5th edition. John Wiley and Sons, New York, 1983.

Larkman, A.U., Jack, J.J. and Stratford, K.J. “Quantal analysis of excitatory synapses in rat hippocampal CA1 in vitro during low-frequency depression.” *J. Physiol*. (Lond.) 505, 457–71, 1997.

Legéndy, R.C. and Salcman, M. “Bursts and recurrences of bursts in the spike trains of spontaneously active striate cortex neurons.” *J. Neurophysiol*. 53(4), 927–939, 1985.

Lehninger, A.L. *Biochemistry*. Worth, New York p. 157. (1970).

Lynn, P.A. and Fuerst, W. *Introductory Digital Signal Processing with Computer Applications*, Revised Edition. John Wiley and Sons, New York. 1994.

Nelder, J.A. & Meade, R. “A Simplex Method for Function Minimization.” *Comput J*. 7, 308–313, 1965.

Okada, Y., Teeter, J.H., and Restrepo, D. “Inositol 1,4,5-trisphosphate-gated conductance in isolated rat olfactory neurons.” *J. Neurophys*. 71(2):595–602, 1994.

Penner, R. “A practical guide to patch clamping” in: *Single-Channel Recording*, Second Edition, eds. Sakmann, B. and Neher, E. Plenum Press, New York, pp. 3–52, 1995.

Piek, T. “Ionic and Electrical Properties” in: *Insect Muscle*, ed. Usherwood, P.N.R. Plenum Press, New York. pp. 281–336. 1975.

Powell, M.J.D. “A fast algorithm for non-linearly constrained optimization calculations” in: *Proceedings of the 1977 Dundee Conference on Numerical Analysis*, ed. Watson, G.A. Springer Verlag. 1978.

Press, W.H., Teukolsky, S.A., Vetterling, W.T and Flannery, B.P. *Numerical Recipes in C. The Art of Scientific Computing*, Second Edition. Cambridge University Press, Cambridge. 1992.

Quastel, D.M. “The binomial model in fluctuation analysis of quantal neurotransmitter release.” *Biophys*. J. 72, 728–53, 1997.

Rao, C.R. *Linear Statistical Inference and Its Applications*, Second Edition. John Wiley Publications. Ch. 4 and 5, 1973.

Reid, C.A. and Clements, J.D. “Postsynaptic Expression of Long-Term Potentiation in the Rat Dentate Gyrus Demonstrated by Variance-Mean Analysis.” *Jnl. of Physiology* 518.1, 121–130, 1999.

Sands, S.B. and Barish, M.E. “Calcium permeability of neuronal nicotinic acetylcholine receptor channels in PC12 cells.” *Brain Res*. 560, 38–42, 1991.

Schnuch, M. and Hansen, K. “Sugar sensitivity of a labellar salt receptor of the blowfly Protophormia terraenovae.” *J. Insect Physiol*. 36(6):409–417, 1990.

Schreiner, W. Kramer, S.K. & Langsam, Y. “Non-linear Least-squares Fitting.” *PC Tech Journal* May, 1985.

Schwartz, G. “Estimating the dimensions of a model.” *Ann. Statistics* 6, 461–464, 1978.

Sigworth, F. and Sine, S.M. “Data transformation for improved display and fitting of single-channel dwell time histograms.” *Biophys*. J. 52, 1047–1054, 1987.

Sokal, R.R. and Rohlf F.J. *Biometry*, 2nd Edition. Freeman, San Francisco, 1981.

Stephens, M.A. *Journal of the Royal Statistical Society*, ser. B. 32, 115–122, 1970.

Stevens, C.F. “Inferences about molecular mechanisms through fluctuation analysis” in: *Membranes, Channels, and Noise*, eds. Eisenberg, R.S., Frank, M., and Stevens, C.F. Plenum Press, New York. pp. 1–20., 1981.

Tang, C.Y. and Papazian, D.M. “Transfer of voltage independence from a rat olfactory channel to the Drosophila ether-à-go-go K+ channel.” *J. Gen. Physiol*. 109:301–311, 1997.

Thurm, U. “The generation of receptor potentials in epithelial receptors” in: *Olfaction and Taste IV*, ed. Schneider, D. Wissenschaftliche Verlagsgesellschaft, Stuttgart, pp. 95–101, 1972.

Vermeulen, A. and Rospars, J.P. “Dendritic integration in olfactory sensory neurons: a steady-state analysis of how the neuron structure and neuron environment influence the coding of odor intensity”. *J. Comput. Neurosci*. 5:243–266, 1998.

Yellen, G. “Ionic permeation and blockade in Ca2+-activated channels of bovine chromaffin cells.” *J. Gen. Physiol*. 84, 157–186, 1984.

Zhang, L. and McBain, C.J. “Voltage-gated potassium currents in stratum oriens-alveus inhibitory neurones of the rat CA1 hippocampus.” *J. Physiol*. (Lond.) 488, 647–660, 1995.

Further Reading

Bishop, O.N. *Statistics for Biology*. Longmans, London, 1966.

Finkel, A.S. “Progress in instrumentation technology for recording from single channels and small cells” in: *Cellular and Molecular Neurobiology*, eds. Chad, V. and Wheal, H. Oxford University Press, New York, pp. 3–25, 1991.

Hamill, O.P., Marty, A., Neher, E., Sakmann, B., and Sigworth, F.J. “Improved patchclamp techniques for high-resolution current recording from cells and cell-free membrane patches.” *Pflügers Arch*. 391:85–100, 1981.

Sakmann, B. and Neher, E., eds. *Single-Channel Recording*, Second Edition. Plenum Press, New York, 1995.

Appendix B: Troubleshooting

B

Software Problems

If you have any installation questions or problems, first check the website Knowledge Base linked through the Help menu within each application. We are continually improving our products, and often a problem that you are having has already been fixed in a newer version of the program. You can download the latest version of pCLAMP and software release notes from the website.

It is also a good idea to test your hard disk and file structure when encountering problems. For problems not related to data acquisition, try more than one data file to see if your problem is due to a corrupted data file. If the problem is associated with one protocol file, that file may be corrupted; use the **Acquire > New Protocol** command to recreate the protocol. If the program is generating severe errors, use the utility in the Molecular Devices folder, **Reset to Program Defaults**, to clear the registry entries. Please report any known bugs to Molecular Devices.

Context-sensitive menus are used. If you are unable to find or access a particular feature, it might be associated with a different type of window from the one that is active, for example that has its title bar highlighted. Or, a certain acquisition mode might need to be active for associated operations to be available. Sometimes, a certain parameter will need to be selected to enable additional parameters. Refer to the Help file.

Hardware Problems

If you have other equipment hooked up to your data acquisition system, start by disconnecting everything, except for a single BNC cable connecting Analog Out #0 to Analog In #0, to isolate the problem to the computer system. If you are still unable to determine if you have a hardware or a software problem, try moving your Digidata digitizer to another computer and see how it runs there. If it runs OK, then you have a computer-related problem.

Besides the data acquisition system itself, other common causes of intermittent hardware problems are defective cables. Try using different cables. Our technical support department will help you to further diagnose your problems.

Obtaining Support

Molecular Devices is a leading worldwide manufacturer and distributor of analytical instrumentation, software and reagents. We are committed to the quality of our products and to fully supporting our customers with the highest possible level of technical service.

Our support web site, www.moleculardevices.com/support , has a link to the Knowledge Base with technical notes, software upgrades, safety data sheets, and other resources. If you do not find the answers you are seeking, follow the links to the Technical Support Service Request Form to send an email message to a pool of technical support representatives.

You can contact your local representative or contact Molecular Devices Technical Support by telephone at 800-635-5577 (U.S. only) or +1 408-747-1700. In Europe call +44 (0) 118 944 8000.

Please have your instrument serial number or Work Order number, and your software version number available when you call.

Appendix C: Resources

C

Programs and Sources

Several software programs complement pCLAMP Software by providing additional analysis and plotting functions.

Note: Support of the pCLAMP 10 ABF file format has not been verified with these companies.

AxoGraph X (AxoGraph Scientific) directly reads ABF data files for whole-cell, minis analysis and graphics on Macintosh computers.

DataAccess (Bruxton Corp) imports ABF files into Excel, Igor Pro, Origin and SigmaPlot.

DataView (Dr. Heitler, University of St. Andrews) directly reads ABF files for data analysis.

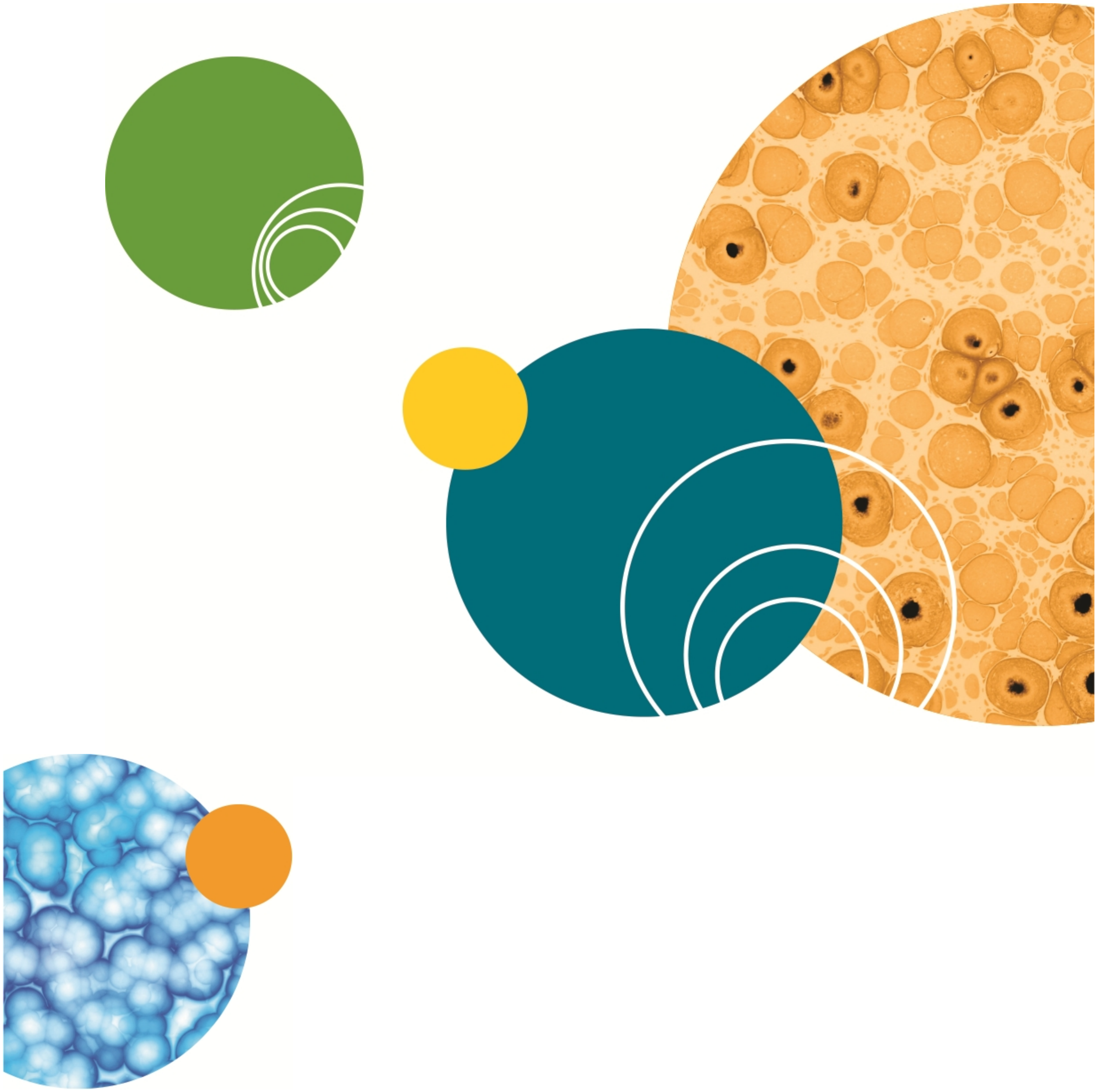
DATAPAC 2K2 (RUN Technologies) directly reads *.dat and ABF data files for spike train analysis.

Experimenter (DataWave Technologies) converts its data files to ABF files for use with pCLAMP's single-channel analysis.

Mini Analysis Program (Synaptosoft, Inc.) directly reads *.dat and ABF files for overlapping minis detection.

Origin (OriginLab Corp.) directly reads ABF data files for general analysis and graphing.

SigmaPlot (Systat Software Inc.) Electrophysiology Module directly reads ABF data files for general analysis and graphing.



Index