CORDUROY
Data acquisition and analysis for IGOR

Joshua Tate Dudman

Program of Neurobiology and Behavior
Columbia University

December 13, 2006

Contents

1 Overview

2 Installation

3 Analysis

3.1 The UTILITY panel

The utility panel provides basic functionality for getting data into IGOR in a way that Corduroy can understand and use for subsequent analysis. In order to maximize compatibility, Corduroy only requires that waves within a given data folder follow the naming convention $RecordA[trace_number]$ for channel 1 and *RecordB*[*trace_number*] for channel 2, etc. The name of the folder is arbitrary, but it generally works best to have the folder name be the name of the file number or experiment number relevant to your data acquisition software. (If you are using Corduroy to acquire your data the name of the folder will be the date and experiment number.) The first group of buttons are associated with getting waves into IGOR and ready for analysis. First, the *LoadComplete* button looks for ".dat" files in the user's DataDir path (you will be prompted to select the path the first time this button is used). A selected dat-file will the be loaded in its entirety and renamed according to the Corduroy formalism. If your waves are loaded using another macro or procedure they should be renamed using the *RenamePreload* button. This function sorts and organizes data to prepare it for subsequent analysis according to a rather simple formalism. Names should be divided by "_" that separate a description of the recording numbers. A prompt is created that allows the user to define the format of the wave names. It is preferable the user-defined loading functions name waves according to the expected formalism for compatibility. The $WriteOutWave$ button allows the user to export waves for analysis and use by complementary programs such as NEURON and Matlab using the $fprintf$ function built-in to IGOR.

The $RunWaveUtility$ button provides another class of functionality. The particular type of function executed is specified in the pulldown menu to the right. These utilities operate on either the top graph, the current active set, or from a user selected group of waves with the preferences searched in that order. A simple way to analyze a set when several graphs are used in the experiment (for example, during acquisition) is to use the *PlotSet* function to create a new top graph and then run the analysis. The functions used follow a rather simple naming formalism and can be found in the file *Corduroy_Analysis_util_lib.ipf* by advanced users. New functions can be defined following the naming convention and a few simple dependencies that must be recognized. In future versions, an object-oriented approach will be implemented to allow the user to add functionality in a stream-lined manner.

The middle group of utilities provide automated graphing of waves once loaded into an experiment. Full traces in a given data folder, individual waves, or waves defined within the active set can all be plotted easily. Two checkboxes allow automated updates of the graphs. The *Auto* checkbox plots individual traces as they are selected by the slider. The *AutoZoom* checkbox automatically zooms in to the region selected by the cursors on the top graph when checked.

Finally, data subsets can be defined by using the $DefineSet$ button at the bottom of the utility panel. This functionality is inspired by the NeuroMatic program and has much the same behavior. The idea is that arbitrary combinations of wave can be bound together as a set for graphing and analysis purposes. As above, in the absence of a top graph procedures will look to the *ActiveSet* pulldown menu for an active set that should be analyzed. The *PlotSet* function works on the selected, active set to place the waves in a new graph window. The final checkbox, $Default$, tells Corduroy that the active set should be used by default when running the utility functions.

3.1.1 Built-in utility functions

All of the utility functions described here are virtual in the sense that they do not alter the original data (with the exception of the RESCALE function) but create a subfolder within the data folder containing the plotted data or a folder containing a copy of the set of waves. Restoration of the original data is thus simple and the analyzed folder can be deleted at anytime to restore the experiment to only unaltered data.

3.1.2 AVERAGE

A very simple function that returns the average of an aribitrary number of waves using a relatively fast algorithm (MatrixMultiply with a column vector of 1's). The averaged data will be placed in the Altered sub data folder with the prefix "AVG_*index*".

3.1.3 CONCATENATE

Uses an IGOR extension to concatenate lists of waves. The waves are concatentated in the order that they are selected (listed) in the preferred source.

3.1.4 RESCALE

A prompt will be brought up that optionally allows the user to rescale the x and/or y axes by a fixed amount.

3.1.5 BASESUB

Generates a new set of waves which are translated by the mean of the wave between the two cursors. This is useful for removing a baseline offset from a series of waves. Cursors must be located on a wave in the top graph otherwise and error will be generated. As a result, this function will only work on data in the top graph, not an active set or data folder.

3.1.6 FILTER, SMOOTH, DIFFERENTIATE and INTEGRATE

These functions use built-in IGOR operations. See the IGOR documentation for details on use.

3.1.7 Adding new utility functions

3.2 The ANALYSIS panel

The analysis functionality of Corduroy makes the following assumption about how data is collected during experimental work: a single trace recorded during an experiment often collects multiple pieces of data about the system under study. In order to facilitate the analysis of data collected in this manner, it is important to be able to define regions of the trace upon which certain analysis functions are executed. At the same time, it is frequently desirable to perform multiple types of analysis on individual traces. Before developing Corduroy I found myself writing entire analysis routines from scratch that involved all sorts of data handling. This is a very wasteful approach and I decided that the real goal is to reduce analysis to the least sophisticated data handling abilities and collect all the data handling into a single overarching structure. A pseudo-object oriented approach is used where analysis functions follow rather simple rules and naming conventions and as such are accessible to the overarching data handling structure of the Corduroy software. Advanced programmers can add their own analysis functions with relative ease and subsequent releases of the program will primarily focus on reducing the knowledge of IGOR required to add analysis functionality. The goal is not to provide all possible types of analysis with IGOR, but to facilitate the easy development of unique combinations of simple analysis that meet the complicated needs of real experiments. Below are described some of the basic analysis functions included with the software and a brief description of how a new analysis routine, recognized by Corduroy, is created.

3.2.1 Built-in Analysis functions

3.2.2 STATS

3.2.3 EVENTPEAK

3.2.4 SLOPE

3.2.5 KINETICS

3.2.6 SPIKES

3.2.7 Adding new analysis functions

3.3 The METAANALYSIS panel

3.3.1 Built-in Meta-analysis functions

3.3.2 NORMALIZE

3.3.3 BOXCAR

3.3.4 Adding new meta-analysis functions

4 Data acquisition

4.1 The structure of data in Corduroy

All data is structured as individual waves named according to the convention “Record”+The Channel+The sweep number. Thus, the first recorded sweep on channel A is called RecordA0, and so on. Individual protocols, potentially containing multiple sweeps, are gathered together within data folders which are sequentially named as the user executes various stimulus protocols. Many actions of the user are recorded into a Notebook file which can be used to reconstruct the relationship between a given data folder and the protocol that was used. The stimulus waves for each protocol are also loaded into memory and stored within the experiment file. A running list of the recorded protocols is saved within the list in the main panel and allows for easy display of past recordings by the user.

4.2 The MAIN panel

4.3 The TRACEPLOT panel

4.4 The UTILITYPLOT panel

The UTILITYPLOT panel is generated when the user presses the ”Test Pulse” button in the MAIN panel. This panel is useful for patch clamp recordings when it is necessary to monitor the resistance at the tip of the pipette either during the initial formation of a seal or to accurately compensate capacitance and series resistance in the whole cell configuration. For details about making a recording and accurately compensating for pipette parameters the manuals for many amplifiers from Axon Instruments (now Molecular Devices) provide excellent discussions.

The panel itself should be rather self-explanatory. The pulldown menu at the top left has options ”Test Pulse” or ”Bridge Balance”, which are useful in the voltage clamp and current clamp modes, respectively. The ”Channel” popup menu specifies the output channel through which the test pulse is sent. Finally, the start button is used to begin and end the application of the test pulse. In addition, online calculations of the resistance are calculated during application of the test pulse or bridge balance pulse. In the bridge balance mode the Err variable returns an estimate of the offset between the initial data during the charging of the membrane and the baseline holding potential. The absolute value of this Err variable reflects the quality of the bridge balance. Empirically, I find that something in the range of -150 to -200 is the optimal Err value based upon calibration with known electrical circuits.

4.5 The PROTOCOLS panel

The main role of the protocol panel within Corduroy is to provide a simple interface through which existing protocols can be loaded, quickly viewed, and sent to the Main panel for execution during recording. Additionally, the panel contains controls for adding updates to the notebook.

In the Protocols section of the panel there are three buttons and a pulldown menu. The LOAD button allows the user to load a pre-existing protocol from the system directory structure (located within the Corduroy system folder by default). Once the protocol has been loaded the experiment will contain full waves for the protocol and this protocol will be available in the ACTIVE popup menu. Selecting a protocol from the active menu will tell Corduroy that the next execution of a record or play (see Main panel) event will use the active protocol. Protocols can be rather large files and it is recommended that only the required protocols be loaded into a given experiment to keep the file size within reason. It is possible to set up a set of protocols to be loaded by default upon launching Corduroy, but this feature is only available to advanced users and is not supported in the manual.

In order to view or manipulate the existing protocol the user can use the VIEW and EDIT buttons. Pressing the VIEW button will generate a new graph window in which the active protocol will be plotted. Finally, pressing the EDIT button will launch the Protocol Generator with the values set to those used in the current protocol. Changes to the protocol can be made and the Save as... button used to overwrite the existing protocol with a modified version.

In the Data Log section there is a popup menu and a button. The popup menu is used to select a pre-written section of text. Once the MARK button is pressed the Notebook (or Data Log) will be appended with the text specified by the popup. In general, this feature is useful for making note of important events during the recording with a time stamp, allowing the user to follow the details of the experiment later, during analysis.

4.6 The PROTGEN panel

The design of the protocol generator panel is intended to be as modular as possible, thereby allowing the user to have maximal flexibility in the generation of protocols used during data acquisition. It is possible to build on top of the existing protocol types for advanced users who understand IGOR programming relatively well. Within the *Corduroy_DAQ_protgen* procedure file there are details for creating your own functions. It is expected, however, the currently available methods will satisfy the vast majority users and tasks.

A protocol within Corduroy DAQ is organized as a series of waveforms repeated an arbitrary number of times and in an arbitrary sequence. Each waveform is generated through the combination of segments which have distinct characteristics. The user describes segments individually and can combine these segments to create waveforms with unique combinations of features. The available segment types are Step, Function, SimSyn, TTL, LoadArb, nrnLink. Each of these segment types is described in more detail below. Once the user has created segments (either of individual types or through the combination of segment types) segments are stored to the disk as a description of properties, but not actual waves (a design that is intended to save space and improve IGOR performance). Pulldown menus allow the user to select individual segments applied to a specific DA or TTL output channel on the ITC18 board. This combination of selected AD channels, DA outputs, TTL outputs and description of timing parameters (Interlude, Length, Sweeps) is combined into a single protocol that can be used by the MAIN acquisition panel. Protocols are saved to disk as complete waves and can be loaded as necessary into individual experiment files. This alleviates problems that could occur if the user wants to repeat exact protocol repetitions, however, protocols can be easily modified and saved using the protocol generator and segments. All the generated segments are available to the user unless they are moved out of the segments folder on the disk.

NOTE: The Igor Pro program is not dedicated from the point of view of the operating system and as such is sensitive to processing overhead. This can cause irregular pauses in the execution of Igor calls to the operating system. As such there is a limit to the accuracy with which short intervals can accurately measured. If the interval of stimulation is critically important it is recommended that the protocol be combined into a single continuously executed sweep to ensure accurate timing. While this significantly increased the overhead in the sense of having large experiment files it is probably necessary at the moment.

4.6.1 Segment Types: Step

The Step segment type is the most simple. It allows for DC pulses to be applied after a specified delay and for a specified duration. The delay and duration of the DC pulse is specified in the TimeOn and TimeOff boxes in combination with the sweep duration. The height of the DC pulse is specified through the combination of the Amplitude and Factor boxes. Factor is a scaling factor that is multiplied by the sweep number (second sweep is 1) and added to the amplitude for each sweep. For all subsequent segment types, the Step specifications (boxes) remain in the dialogue. It is not necessary to separately specify a Step segment and more complicated segments.

4.6.2 Segment Types: Function

The Function segment type is very similar to the Step type except that it allows the user to specify a slightly more complicated waveform that a simple DC waveform. It supports the generation of sine, cosine and saw functions. More complicated functions are supported using the Design... dialogue. The support for the Design.. dialogue is currently weak, but it expected to be improved in future releases. For the time being the LoadArb segment type is a convenient work-around. Functions are permitted to have up to 2 parameters specified. For the pre-packaged functions the parameters P1 and P2 specify the frequency factor (equivalent to the amplitude factor in the Step type) and the rate of change in frequency during the pulse (Hz/ms; useful in the creation of ZAP functions), respectively.

4.6.3 Segment Types: SimSyn

The SimSyn segment type allows the user to simulate a pattern of synaptic inputs impinging upon the cell being recorded. The protocol generator is structured such that events are generated according to some probability distribution. This distribution can range from a linear distribution of fixed probabilities to various distributions with more complicated probability descriptions. A familiarity with Poisson and Gamma distributions is recommended before using these features. The shape of the simulated synaptic event is defined by the Tau1 and Tau2 boxes. These time constants (in ms) are combined to generate a biexponential current waveform that will be generated at each event. Users may want to specify very complicated combinations of excitatory and inhibitory synaptic potentials, for this type of protocol it is recommended to use the LoadArb segment type and generate the waveforms either in IGOR or another program (e.g. Matlab).

Simple probability distributions are supported and can be obtained by using the the pull-down menu. A "Linear" distribution assumes a fixed probability of 1 at each event time given by the frequency parameter (P1). To get a single TTL pulse specify a low frequency. In Linear mode the first event occurs at the time specified in the TimeOn box. Currently, four types of random distribution have been implemented, but any number more could easily be created by the user. Standard are three Poisson distributions (standard, fixed interval renewal, and an exponentially recovering renewal). For the standard Poisson distribution mode, P1 specifies the mean frequency (in Hz) of events and uses the IGOR random number generator to get random distributions (consult the help files in IGOR for a description of the *enoise* function). For PoissonR the second parameter (P2) specifies an absolute renewal period in which the probability of a spike drops to zero. And for PoissonRexp, p2 specifies an exponential time constant (in ms) during which the probability of generating a spike relaxes back towards the mean rate. For the Gamma distribution, P1 specifies the mean interval of points in the distribution (rate in Hz). Finally, the dynamic pulldown menu allows the user to decide whether each sweep of this segment type should have a unique, random set of event timings or whether the same random set of event timings should be used for each sweep.

4.6.4 Segment Types: TTL

The TTL segment type is distinct from other segment types because it allows control of the TTL outputs on the ITC18 board (2 total outputs are supported by the XOP). TTL pulses are useful for triggering external equipment such as stimulators or other recording devices. The protocol generator is structured such that (like SimSyn) events are generated according to some probability distribution. This distribution can range from a linear distribution of fixed probabilities to various distributions with more complicated probability descriptions. A familiarity with Poisson and Gamma distributions is recommended before using thee features. The width and height of the TTL pulse (default values are 0.1 ms and 5 V, respectively) is specified in the top two boxes.

4.6.5 Segment Types: LoadArb

If a desired segment type cannot be generated within the current protocol generator the user can load waves from other programs into protocols within Corduroy. This takes a good deal of user care to ensure that the wave being loaded matches the protocol's expected sampling rate and timing. However, it allows for maximum flexibility in the protocols that can be created. There is little support for this feature and requires the user to understand how the Corduroy functions structure a stimulus wave.

Briefly, the user must place waves named according to the convention ArbWave0, ArbWave1, etc. into the data folder containing the segment of interest. Note: these waves will be destroyed once the user uses them to make a particular protocol, so they should be backed up in another data folder. User must take care to enter appropriate values for segment length, protocol length, sample rate, etc. to match the user-generated waves.

4.6.6 Segment Types: nrnLink

This segment type is not supported by the manual. It is in development, and the code has been removed from the distribution form of Corduroy.

4.6.7 Segment Types: dClamp

The ITC18 board allows for linear dynamic clamp behavior using lookup tables. Currently, this is not implemented in the Corduroy acquisition program, but is reserved for a future release.

5 Future development

Please contact me at *j.dudman@gmail.com* with ideas, suggestions, requests, etc. No guarantees are made as to whether changes can be implemented but your ideas are welcome. Use the subject line "CorduroyDAQ comment." Updates and news can be tracked on the distribution website:

http://www.columbia.edu/~jtd2001/pages/main/research/software.html

6 Credits

This software was written while I was a graduate student at Columbia University in the laboratory of Dr. Steven Siegelbaum. Useful feedback was provided by a number of people, notably Matthew Nolan and Andrew Fink, for which I am thankful.