

EPhys Matlab Toolbox

Electrophysiology Experiment Software for TDT

Designed & Written by Daniel Stolzberg, Ph.D.

daniel.stolzberg@gmail.com

last updated on 3/16/2014

Copyright © by Daniel Stolzberg, Ph.D. 2013

Table of Contents

List of Figures	3
Introduction	4
Hardware Recommendations, Software Requirements & Installation	5
Hardware recommendations	5
Required software	5
Obtaining and Using SVN.....	5
Adding EPhys software to the MATLAB path	6
EPhys Toolbox Overview	7
Creating a New Experiment.....	7
RPvds Requirements	7
Protocol Design Utility (ProtocolDesign).....	9
Creating a New Protocol.....	9
Adding/Removing Modules	9
Working With Parameter Tags.....	9
The \$ flag.....	10
Function column	10
Values column.....	10
Buddy column	11
Rand column	11
WAV column	11
Calib. (calibration) column.....	12
Protocol Design Options.....	13
EPhys Control Panel (EPhysController)	14
Calibration Utility (CalibrationUtil).....	16
Tank Registration (TankReg).....	17
Database Browser (DB_Browser).....	18
The "Quick Plot" Function.....	23
The "Export Unit Data" Function.....	29
Analysis Tool: RF_FreqVsTime	35

List of Figures

Figure 1. SVN Checkout.....	6
Figure 2. Unpopulated (top) and populated (bottom) device navigator	7
Figure 3 Required RPvds Components	8
Figure 4 Add/Remove module	9
Figure 5 Example RPvds circuit using parameter tags (ParTag component).....	9
Figure 6 Protocol Design Utility parameter table with example of parameter values controlling “Frequency” and “Duration” parameter tags (ParTag) specified in the RPvds circuit.....	10
Figure 7 Example of a compiled protocol	11
Figure 8 Protocol Design options	13
Figure 9 EPhys Control Panel	15

Introduction

The EPhys toolbox for Matlab integrates with Tucker-Davis Technologies (TDT) OpenEx software package and real-time processing hardware to facilitate the design and control of electrophysiology experiments. The toolbox includes a control panel which runs the experiment, a protocol design utility for generation parameters for an experiment, and a utility for the automated calibration of simple acoustic stimuli. Some familiarity with the Matlab environment and basic principles of R郑vds circuit design are important for implementing the EPhys toolbox. Creating new experiment protocols, however, requires no programming by the user; however, advanced dynamic Matlab scripts can be written to create more complex experiments.

There were several motivations for the designing the EPhys toolbox to extend OpenEx's capabilities. One motivation was to facilitate the design of complex multiparametric experiment protocols in a simple but powerful graphical user interface (GUI). This is accomplished with the Protocol Design Utility (**ProtocolDesign**) (p. 9).

A second motivation was to enable the use of high sampling rates (~200 kHz on some hardware) required for producing ultrasonic stimuli on the TDT real-time modules. The **Error! o bookmark name given.** (p. 10) accomplishes this by offloading stimulus triggers to the host computer. This approach, which has certain considerations discussed later, reduces the overhead (number of circuit components) required on to run an R郑vds circuit which frees up additional processing power for circuit components required for various stimulation paradigms. An additional advantage of using the host computer to control stimulus triggering is the added capability for using custom Matlab scripts to dynamically select the next stimulus parameters based on previously acquired data (such as spike counts, field potential power, button presses, etc).

Thirdly, a Calibration Utility (**CalibrationUtil**) (p. 16) automates calibration of acoustic sources and creates standardized calibration files which can be used to normalize sound levels of parameters (such as stimulus frequencies).

Finally, the use of high-channel electrode arrays has dramatically increased the size and complexity of electrophysiological datasets. Efficiently maintaining, managing, and accessing these very large datasets is greatly facilitated by the use of a common database structure. To address this, the EPhys toolbox includes GUIs which can be used to upload acquired data and stimulus parameters to an SQL server as well as GUIs to navigate, access, and visualize stimulus-evoked responses.

Hardware Recommendations, Software Requirements & Installation

Hardware recommendations

The minimum requirements of EPhys toolbox are those of the installed version of Matlab and TDT software. The EPhys toolbox was developed on a standard quad-core processor personal computer with 4 GB of memory running Windows 7 64-bit operating system.

Required software

The EPhys toolbox was designed in Matlab version R2011a and higher. The EPhys toolbox does not require any additional toolboxes to function; however, EPhys uses the OpenDeveloper ActiveX controls which can be obtained from TDT. OpenDeveloper is designed to interface with electrophysiology data structures and the OpenEx software suite to run an experiment.

If installing Matlab for the first time, it is recommended to Matlab install to a directory such as “C:\MATLAB\R2012b” (or whatever the current version) and not to the default “Program Files” directory. Installing into the default “Program Files” directory may cause problems with permissions when updating with TortoiseSVN on Windows Vista and later.

Currently, the EPhys toolbox is distributed using a software versioning (svn) system hosted by GoogleCode and additional software will be required to obtain the latest releases of the EPhys toolbox software (see next section).

Obtaining and Using SVN

The software versioning system (svn) requires third-party software to be installed locally on the computer which will be used with the EPhys toolbox. TortoiseSVN is a free software package and can be downloaded here: <http://tortoisesvn.net/downloads.html>. Download and install the latest TortoiseSVN software version for your operating system (32-bit or 64-bit) using default options. Once installed, follow these steps to obtain the EPhys software:

1. Create a new folder called “work” in the Matlab root directory such as “C:\MATLAB\work”
2. Open the work folder in a Windows Explorer, right-click in the white space to open a context menu, and click the “SVN Checkout...” menu item.

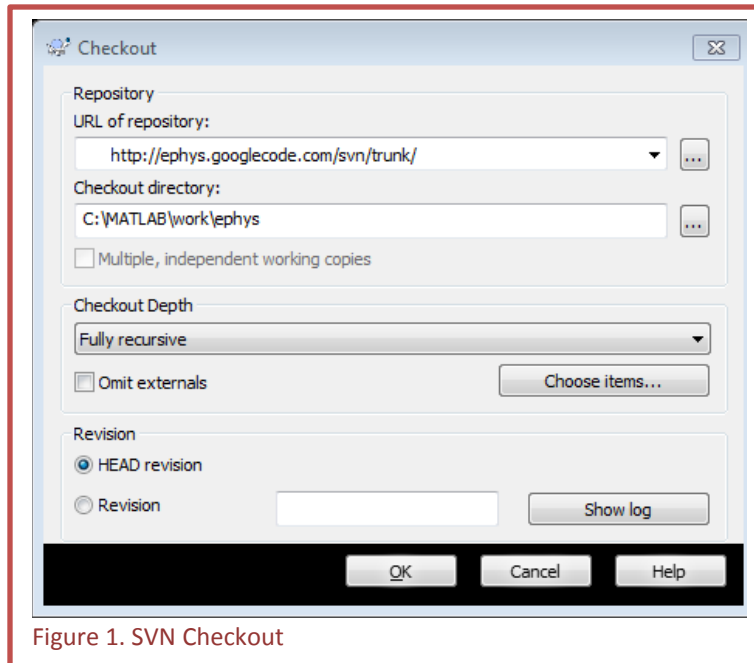


Figure 1. SVN Checkout

3. Enter the following information into the Checkout prompt and click OK.
 - a. URL of repository: <http://ephys.googlecode.com/svn/trunk/>
 - b. Checkout directory: C:\MATLAB\work\ephys
4. A dialog should appear as the EPhys software is downloaded.

The versioning system allows those using the software to easily update files for updates and bug fixes by right-clicking the parent folder (such as: “C:\MATLAB\work\ephys”) and selecting “SVN Update” from the context menu. This system also provides the opportunity for distributed collaboration on future software developments.

A wiki page can be found at <https://code.google.com/p/ephys/wiki/Intro>. This wiki provides a place where issues can be raised, discussed, and addressed with software fixes. This requires a Google account.

Adding EPhys software to the MATLAB path

To make sure MATLAB recognizes the software, open MATLAB, type **pathtool** in the command window, and add the directory of the ephys svn checkout (suggested, C:\MATLAB\work\ephys). Click the Save button and close the path tool dialog. Finally, type the command **ephys_startup** in the command window and appropriate paths will automatically be added to the path.

EPhys Toolbox Overview

The EPhys toolbox for Matlab is designed to integrate with the OpenEx software package provided by TDT. The OpenWorkbench program, included in the OpenEx package, should be used to define which TDT real-time hardware will be used during an experiment. Please refer to the TDT OpenWorkbench documentation for details on how to do this.

Creating a New Experiment

The procedure for designing new experiments with EPhys begins with assigning labels and RPvds files to the TDT hardware modules using the Device Navigator of OpenWorkbench (TDT). While labels are arbitrary in reality, descriptive labels should be used in practice. One example would be to use the label **Stim** for a module (such as an RP2.1, RX6, RZ6, or other module) which will be used to control stimulation during the experiment, such as a speaker or light. If multiple stimulus modules are to be used in an experiment then names such as **Stim1** and **Stim2**, or **SpeakerStim** and **LightStim** can be used (**Error! Reference source not found.**). One or multiple data acquisition modules (such as an RX5, RZ5, or other module) can be assigned a label in a similar manner (**Acq**, for acquisition device, is used in Figure 2). Note that not all modules need to be populated. The labels assigned in the Device Navigator of OpenWorkbench will be used to direct parameters during protocol design using the EPhys toolbox.

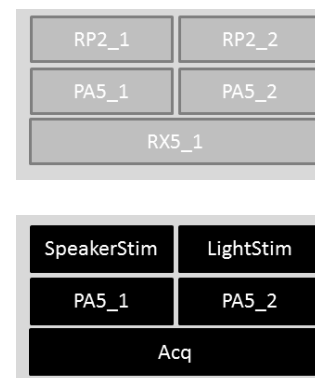


Figure 2. Unpopulated (top) and populated (bottom) device navigator

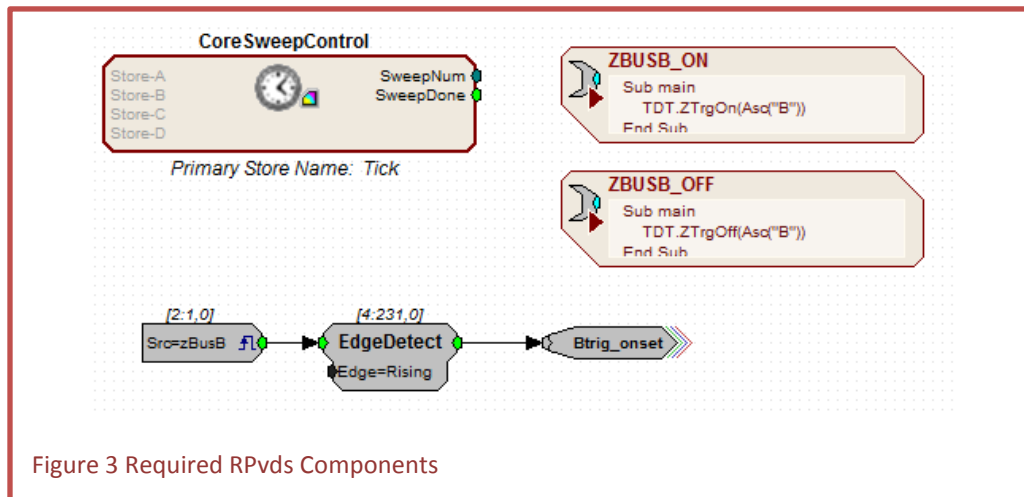
NOTE: Programmable Attenuator PA5 modules should be labeled PA5 followed by a suffix if desired as is in the example in Figure 2.

An RPvds file (with the extension .rcx) should be assigned to each real-time module which is defined in the Device Manager (see TDT documentation for details). The conventions for designing RPvds files for use with OpenWorkbench are outlined in the TDT documentation and should be followed. There are a few requirements for RPvds circuit designs when using EPhys to control experiments that will be discussed next.

RPvds Requirements

For the majority of experiment designs, the EPhysController (discussed on page 14) controls parameter values and timing of stimulation of the TDT hardware modules (defined in

the OpenWorkbench Device Navigator; see page 7). EPhysController first updates all parameter tags on real-time modules as well as any PA5 programmable attenuators, and then uses the **zBusB** trigger issue a synchronised trigger pulse across all real-time modules. In order to



accomplish this from Matlab the custom scripts named **ZBUSB_ON** and **ZBUSB_OFF** (see Figure 3) must be included on one of the RPvds modules in use.

The **CoreSweepControl** macro (left component in Figure 3) handles module synchronization and is required to be included in every RPvds file. This macro is available in RPvds by clicking the Components menu and selecting Circuit Macros.

Note: If using multiple RPvds circuits, each CoreSweepControl macro must have a unique primary store name. This can be modified by double clicking the macro object in RPvds, clicking the Setup tab, and then Change button. Any four letter name can be used (eg, Tock).

Note: When using high sampling rates on RX generation devices, the CoreSweepControl macro utilizes too much of the processing capabilities. Modified versions of this macro are included in the “ephys\examples\useful macros” directory and may be used to resolve this issue.

Protocol Design Utility (ProtocolDesign)

The Protocol Design Utility is a graphical user interface which is used to specify the values of parameters for an experiment. A simple example of this would be to specify all frequency and sound level combinations for a tone-evoked receptive field experiment. Instead of manually specifying every frequency/level combination, of which there may be hundreds or thousands of combinations, one can use the Protocol Design Utility to specify which frequencies and which sound levels should be presented and the utility permutes these parameters to create a list of stimuli for the experiment.

Creating a New Protocol

Before creating a new protocol, an OpenWorkbench project should be created as described above (see Creating a New Experiment on p. 7). The Protocol Design Utility can be launched in two ways. The command **ProtocolDesign** can be entered into the Matlab command window directly. Alternatively, clicking the “P” icon on the **Error! No bookmark** (p. 10) toolbar will launch the utility.

Adding/Removing Modules

Creating modules in OpenWorkbench was covered in an earlier section (see Creating a New Experiment, p 7). The labels for modules defined in OpenWorkbench can be added to or removed from the protocol design by clicking the “+” or “-” buttons, respectively. Clicking the “+” button will launch a dialog box in which the user should enter the label (capitalization matters) of one module at a time.

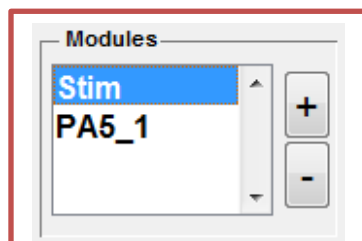


Figure 4 Add/Remove module

Working With Parameter Tags

Once a module has been added to the protocol, parameters can be specified using the table on the right of the Protocol Design Utility.

The RPDs circuit on the right is a simple example of an RPDs circuit which plays tones of various frequencies and durations. The RPDs circuit to the right is one realization of such a design. The

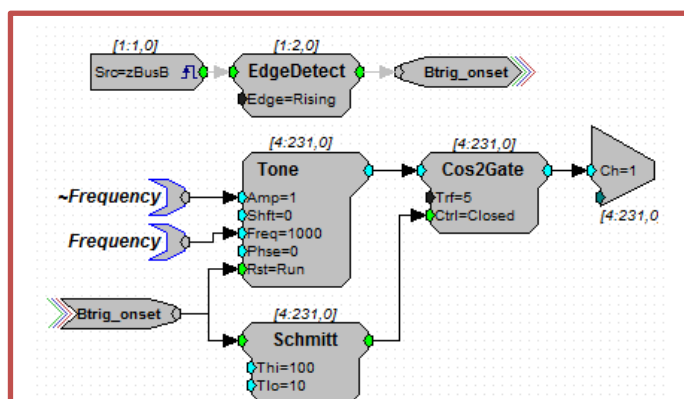


Figure 5 Example RPDs circuit using parameter tags (ParTag component)

parameter tags (ParTag components) called **Frequency** and **Duration** are linked to the **Freq** parameter of the Tone generating component and the **Thi** parameter of the Schmitt logic component, respectively (the **~Frequency** tag is discussed on page 12).

The **Frequency** and “Duration” parameter tags are entered (case sensitive) into the “Tag” column in the Protocol Design Utility parameter table. The other columns of the table are used to specify the values and how the parameter functions.

Tag	Function	Buddy	Values	Rand	WAV	Calib.
Frequency	Write/Read ▼	< NONE > ▼	1000 2000 4000 8000 16000	<input type="checkbox"/>	<input type="checkbox"/>	< NONE > ▼
Duration	Write/Read ▼	< NONE > ▼	25 50 100	<input type="checkbox"/>	<input type="checkbox"/>	< NONE > ▼

Figure 6 Protocol Design Utility parameter table with example of parameter values controlling “Frequency” and “Duration” parameter tags (ParTag) specified in the RPvds circuit.

The \$ flag

Some experiment designs require the values of parameters to be specified when beginning the recording. This can be done by prepending a **\$** (dollar sign) to a parameter tag name (a **\$** should also prepend the parameter tag name in the corresponding RPvds file); for example, **\$Frequency** or **\$Duration**. The **compile at runtime** option must be checked to enable this functionality. A prompt will automatically appear for the user to enter a value for the parameters with the **\$** flag when clicking the **Record** button on the EPhysController GUI (see page 14Figure 9) to begin an experiment.

Function column

The **Function** column can take one of three options selected from a drop-down list. This field indicates whether the parameter tag is updated before each trial (“Write”) or the value is read by EPhys Control Panel after each trial (“Read”). If both functions are required, then the “Write/Read” option should be selected. While most parameters will simply be used to update the parameter tags on the RPvds circuit, the “Read” functionality is useful for dynamic scripts.

Values column

The **Values** column is the primary focus of the parameter table. This field is used to specify the values that the parameter (specified in the **Tag** column of the same row) will have for a protocol. In the example parameter table above, the **Frequency** parameter tag is associated with the values: **1000 2000 4000 8000 16000** (separated by spaces). These values will be used to update the **Frequency** parameter tag in the RPvds circuit example above. The same goes for the **Duration** parameter tag for the values: **25 50 100** (specified in milliseconds according to the Schmitt component requirements).

Instead of manually specifying a list of variables, the **Values** column will accept any Matlab code which evaluates to a numeric scalar or vector. Any code which runs as a line item, including function calls, in the Matlab command window will work in this field. For example, instead of specifying the values manually as in Figure 6, we could have also entered: `2000*2.^(0:3)` with the same result. Another example of valid input: `logspace(log10(1000),log10(42000),40)`.

When using the `$` flag with a parameter tag, the value entered will be the default values when starting an experiment (see page 10 for details).

Clicking the **View Compiled** button (located below the table) brings up a new window with all permutations of specified parameter (Figure 7). In this Compiled Protocol table, each row starts with a trial number with each subsequent column a parameter value. Note that the total number of trials displayed in the table will be truncated if over 2000 trials are generated. The figure caption will indicate the total number of trials.

Buddy column

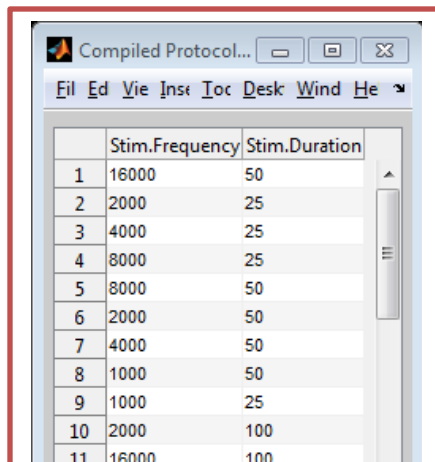
The **Buddy** column of the parameter table is used to have the values of more than one parameter co-vary. In other words, making a buddy variable will make the values of two or more parameters to always be the same. One stipulation is that the parameters with a selected buddy variable must have the same number of values. Multiple buddy variables can be specified.

Rand column

When activated, the **Rand** field works in conjunction with the **Values** field to provide a random value, selected at run time, between two values specified. For example, if a parameter tag is to be updated with a random value between 50 and 200, then enter `50 200` in the **Values** field and check the **Rand** field. Before the parameter is updated during the experiment, a random value will be selected from a uniform distribution (using the Matlab `rand` function) with the lowest possible value of 50 and a highest possible value of 200. This behavior is indicated on the compiled table (click View Compiled) as the two scalar values `[50 200]`.

WAV column

Activating the **WAV** field will launch a new window. Use this window to add WAV format files to a list. The file



	Stim.Frequency	Stim.Duration
1	16000	50
2	2000	25
3	4000	25
4	8000	25
5	8000	50
6	2000	50
7	4000	50
8	1000	50
9	1000	25
10	2000	100
11	16000	100

Figure 7 Example of a compiled

names, sampling rate, and duration of each WAV file will be displayed in the table in the new window. The order in which the WAV files are listed in the table are the order they will be specified for the parameter for which the WAV field was clicked. Please note that the sampling rate of the WAV file should be the same as the TDT real-time module will be running. If this is not followed, then aliasing will occur during playback resulting in unintended stimulus artifacts.

Calib. (calibration) column

The **Calib.** field is used to select a calibration file to be associated with a parameter tag. Calibration files are generated using the Calibration Utility (**CalibrationUtil**) (p. 16). Clicking the drop-down box in this column will launch a Windows dialog requesting the user to locate a calibration file (*.cal) which was created using the CalibrationUtility. Select the calibration file to be associated with the parameter in the same row. When compiled (after saving or clicking the 'View Compiled' button), an additional parameter will be automatically generated which will contain the calibrated value. This associated parameter will have the same name as the parameter but will be prefixed with a tilde (~).

For example: The circuit in Figure 5 (page 9) demonstrates a basic gated tone function. The **Frequency** parameter tag is linked to the **Freq** input of the Tone component and a second parameter labeled **~Frequency** is linked to the **Amp** input of the Tone component. This associated parameter should not be explicitly added to the parameter table in the Protocol Design Utility, but will be automatically generated when adding a calibration to the **Frequency** parameter. The values generated for the **~Frequency** parameter will be selected or calculated from the normalization curve in the calibration file. The result of this will be that each tone frequency presented during an experiment will be generated at a voltage which could produce a sound level (for example, 80 dB SPL) set during the calibration procedure. A PA5 programmable attenuator can then be used to attenuate the voltage to the intended sound level during an experiment.

Note: Values which are not explicitly calibrated are calculated from the the voltage normalization curve using a piecewise cubic hermite interpolating polynomial (the PCHIP function in Matlab).

A calibration file can be used in conjunction with parameters using the **\$** flag in the same manner as with the normal parameters as described above. For example, **\$Frequency** would be associated with the calibrated **~\$Frequency** parameter in RPvds.

Protocol Design Options

A few options (Figure 8) go along with each protocol file which control presentation of trials. The **Randomize** option will reorder the sequence of trials in a pseudorandom fashion.

By default the order of trials is compiled (meaning the actual trial sequence is generated) when the protocol file is saved. This will ensure that the same sequence of trials (even if randomized order) will be presented each time the experiment is run. This behavior can be altered by enabling the **Compile at Run-Time** option. When enabled, the protocol is compiled (the trial sequence generated) each time the experiment is run. This has some advantages if using a custom script.

The **# Reps** field accepts a scalar value which indicates how many times to present each combination of parameters. A value of **5** would repeat all combinations of parameters 5 times.

The inter-trigger interval (ITI) field accepts either a single scalar value or two scalar values separated by a space (eg. **200 500**). If a single value is specified, then a constant time interval will be used between stimulus triggers. If two values are entered, then a random time interval will be used between trials within the specified range. The field is in milliseconds.

Finally, the name of a custom written function can be specified in the **Trial Select Function**. This function supplants and should extend the default trial selection functionality. This is an advanced topic which should be tested thoroughly before running an experiment. If a custom function is specified then it must be somewhere on the Matlab path.

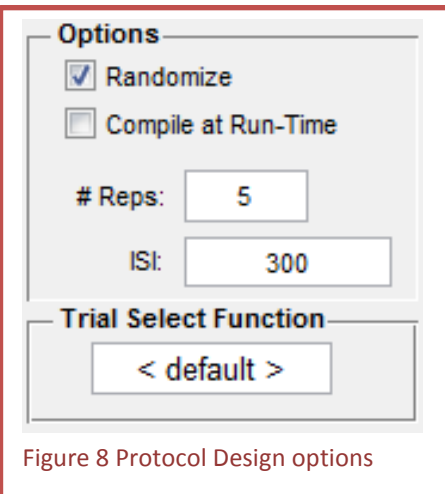
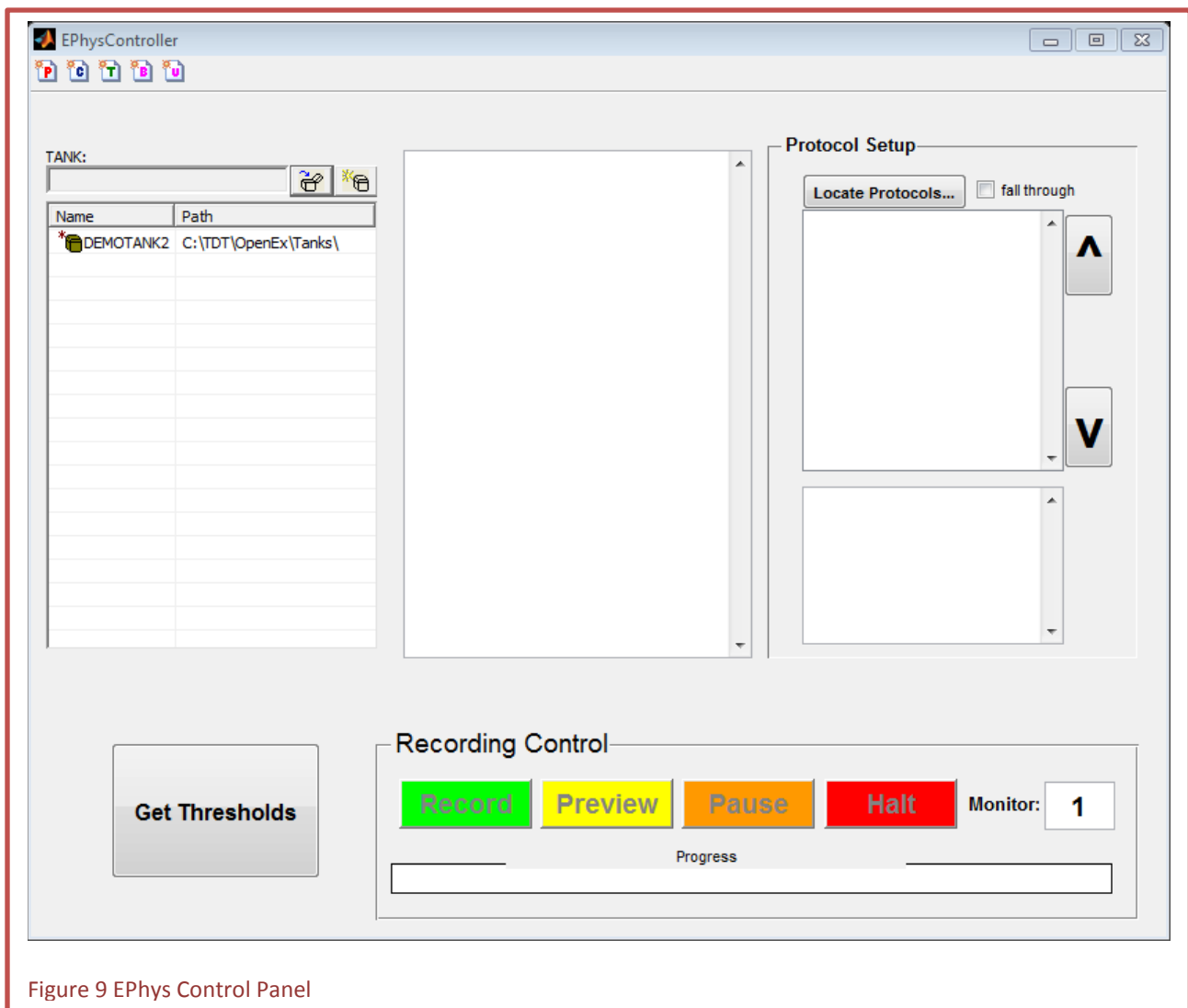


Figure 8 Protocol Design options

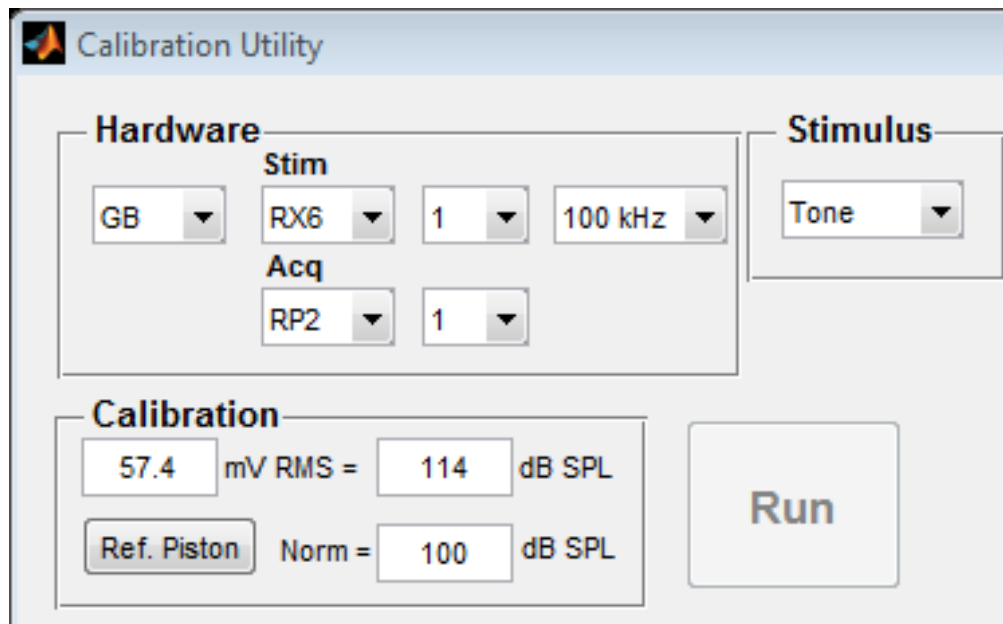
EPhys Control Panel (EPhysController)

The EPhys Control Panel controls physiology experiments in conjunction with OpenEx software. To launch the program, enter **EPhysController** into the Matlab command window and hit enter. To begin an experiment:

1. OpenProject must be open on the same computer and the OpenWorkbench Device Manager (see on page 7) should be set for the intended experiment.
2. Create a new tank or select an existing tank using the left panel of the EPhys GUI.
3. Click the **Locate Protocols...** button and select a directory containing one or multiple protocol files.
4. Click **Record** to begin an experiment.
5. Progress of the experiment will be displayed at the bottom of the EPhysController and the software will automatically stop after finishing and reset for the next protocol.



Calibration Utility ([CalibrationUtil](#))



The screenshot shows the 'Calibration Utility' MATLAB GUI. It features three main sections: 'Hardware', 'Stimulus', and 'Calibration'. The 'Hardware' section includes dropdowns for 'GB', 'RX6', '1', and '100 kHz' under the 'Stim' label, and 'RP2' and '1' under the 'Acq' label. The 'Stimulus' section has a 'Tone' dropdown. The 'Calibration' section displays '57.4 mV RMS = 114 dB SPL' and 'Ref. Piston Norm = 100 dB SPL'. A large 'Run' button is positioned to the right of the 'Calibration' section.

Calibration Utility

Hardware

Stim: GB, RX6, 1, 100 kHz

Acq: RP2, 1

Stimulus

Tone

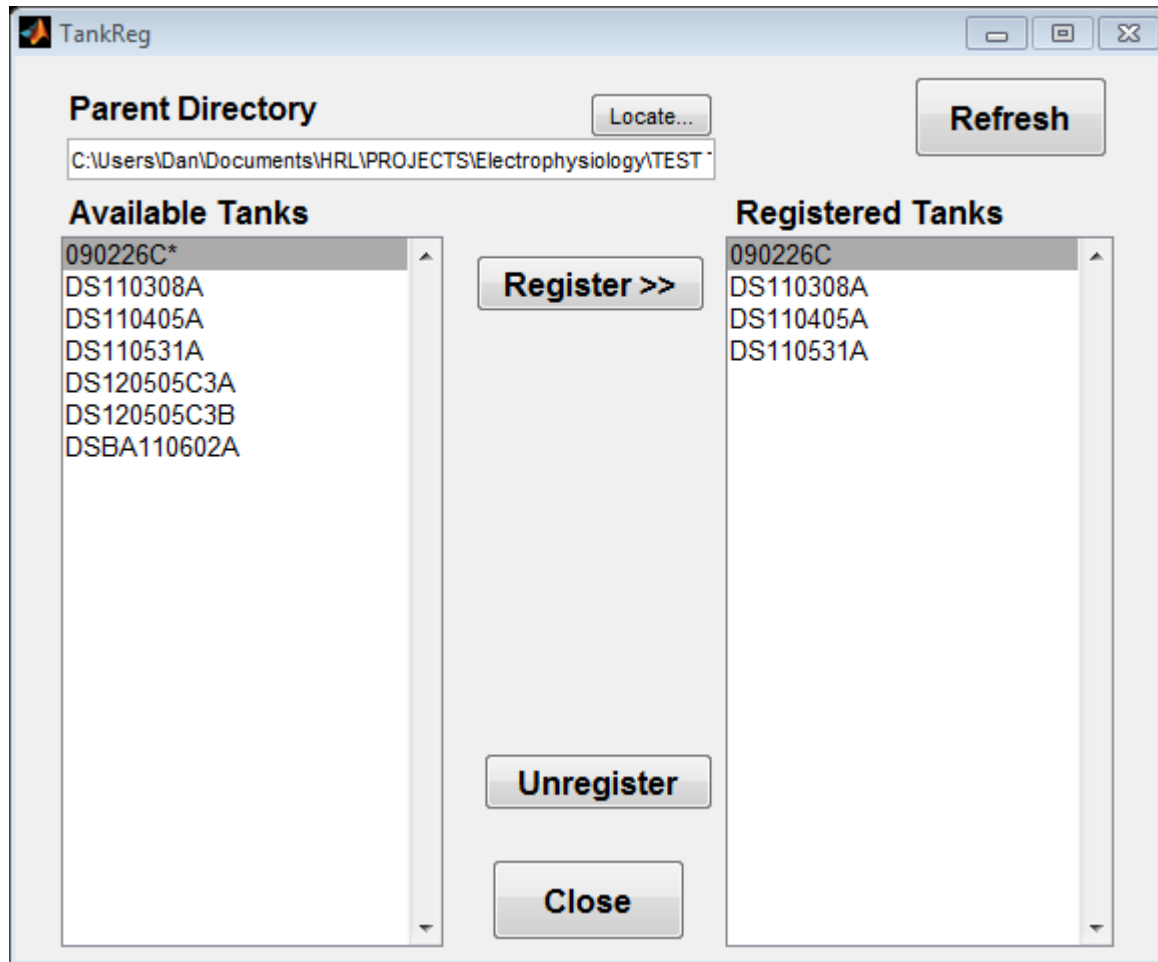
Calibration

57.4 mV RMS = 114 dB SPL

Ref. Piston Norm = 100 dB SPL

Run

Tank Registration (TankReg)



Database Browser (DB_Browser)

The DB_Browser window is the panel from which all of the analyses are controlled. It is the first figure that is seen when the application is run, and it is returned to when each analysis tool has served its purpose.

The first thing to take note of is the DB_Browser window's layout. The interface is divided between seven panels and fifteen buttons, which are indexed and indicated in figure 1-1. The panels are boxed in red and numbered 1-7, while the buttons are boxed in blue and lettered A-O. There are also seven checkboxes, all but two of which are similar in function and common across five of the panels. The common checkboxes are circled in purple, and the two unique checkboxes are double-circled in green.

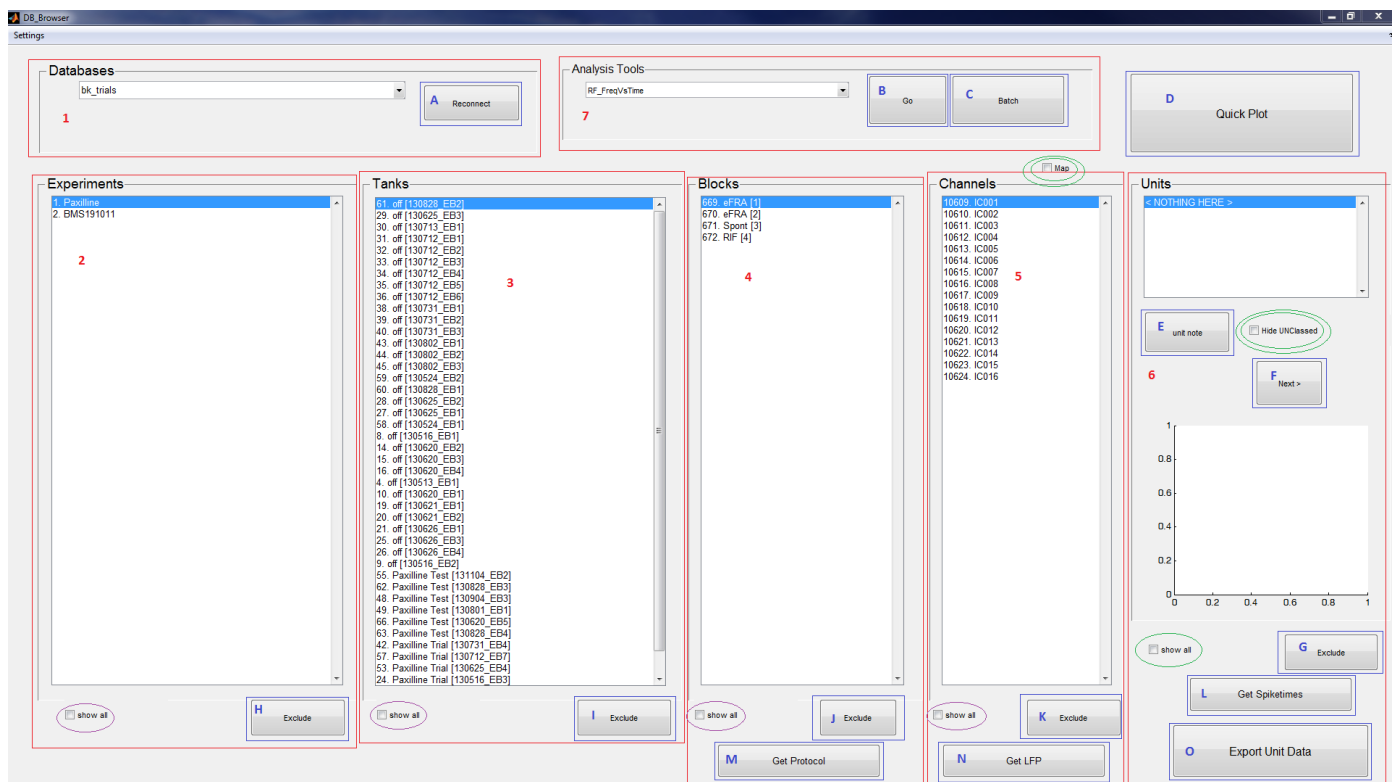


Figure 1-1 above. The panels are numbered 1-7 and boxed in red. The buttons are lettered A-O and boxed in blue. The two unique checkboxes are double-circled in green, and the five common checkboxes are circled in purple.

We'll go over each of the panels in order, also covering the buttons that are included in each panel (all but D). We'll then discuss the common checkboxes, then the buttons not included in any panel (only button D), which will get their own subsection starting on page five.

Panel 1 is labeled "Databases". Unsurprisingly, this panel's functionality pertains to database selection. The drop-down menu contains the databases on the local mysql server that are compatible with the DB_Browser's data structure. A database that is "compatible" is one that contains a table named "protocols" with the appropriate organization and content for the analysis functions of the DB_Browser. **Button A**, labeled "Reconnect" resets the connection to the server when clicked. This allows you to refresh your analysis if the database is updated while you are working, or if something goes wrong with the server.

Once a database is selected in window 1 and a connection is established, the list in **panel 2** will be filled in. This list contains the id numbers and titles of the types of experiments from which the data in the selected database originates. Only one such experiment title can be selected at one time. Hence the panel's label. **Button H**, labeled "Exclude", allows you to exclude the selected experiment from batch analysis (see page *fill this in later*). If button H is clicked while an experiment is selected, the button will turn red and remain red while that particular experiment is selected. This means that *all units in all channels in all blocks in all tanks in the selected experiment* will be ignored during batch analysis. Clicking it again while that experiment is selected will return button H to its usual gray coloration, and remove the exclusion effect.

Like panel 2, the list in **panel 3** will populate once an experiment in the previous panel is selected. This list contains all of the tanks in the database that contain data from the experiment selected in panel 2. This list contains the id numbers of each tank, followed by the tank condition (how the tank relates to its parent experiment), followed by the tank name in brackets. At the time of writing, there are three possible tank conditions - Paxilline Test, Paxilline Trial, and off. Paxilline Test and Paxilline Trial are equivalent, while "off" indicates that the tank is not to be considered useful experimental data. Panel 3 also contains its own exclude button (**button I**), which applies to the selected tank. When clicked, button I will turn red for the selected tank. This indicates that *all units in all channels in all blocks in the selected tank* will be excluded during batch analysis (see page *fill this in later*). Upon a second click, button I will return to its gray hue, and the selected tank will no longer be excluded.

Panel 4 will populate once a tank in panel 3 is selected. This panel contains a list of all blocks in the tank selected in panel 3. Each entry in the list contains the block's id number, followed by its protocol type (the type of data that was collected during this block's experimental run). For explanations of the protocol types currently being used, please see that appropriate section on page 13. **Button M**, labeled with "Get Protocol", retrieves the parameter data of the selected block from the database into a matlab data structure. The MATLAB prompt will display this structure's dimensions (always 1x1, as right now only one

block at a time can be selected), and size in bytes. This will have no effect on the use of the gui, but the structure so created contains the data collected from the selected block's experimental run. Panel 4 also has **button J**, another "Exclude" button. Toggling this button to red will exclude *all units in all channels in the selected block* from batch analysis.

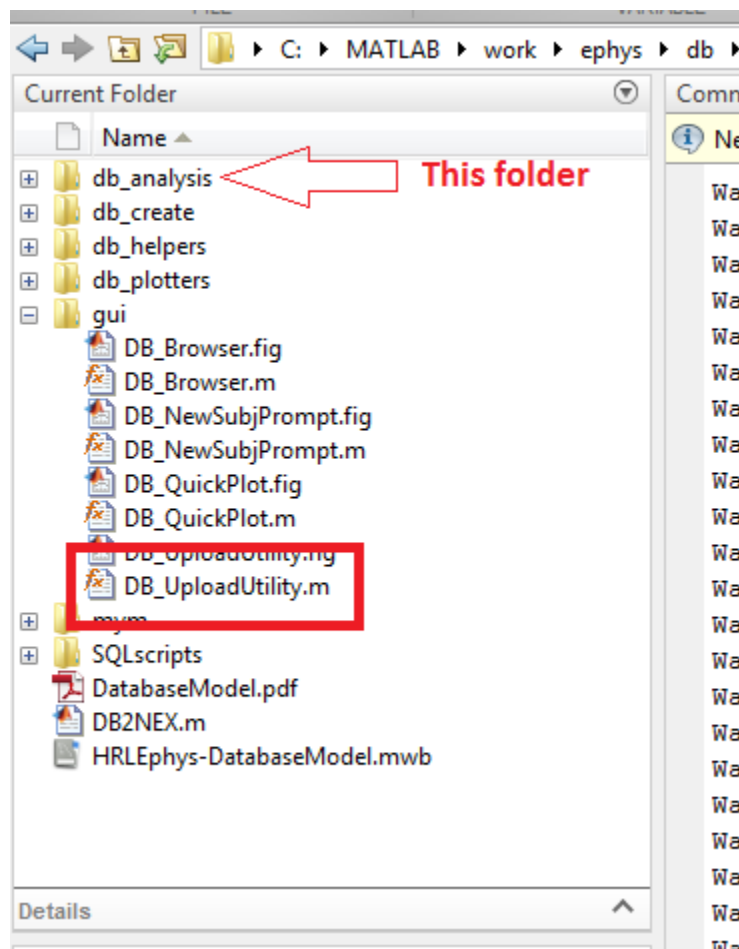
Panel 5, labeled "Channels" will populate once a block is selected. This panel's list contains entries with each channel's id number, followed by an identifier which contains the target area of the brain (IC for "Inferior Colliculus") as well as the channel of probe used. The channel id is unique across all blocks, but the channel of the probe will run between 001 and 016 in each block. The **button N**, labeled "Get LFP" is similar to button M in that it retrieves a data structure from the database into the MATLAB workspace. However, this structure has only two constituents - "wave" and "tvec". Both of these constituents are vectors of the same length. "tvec" contains the time interval over which the selected channel's data was collected. "wave" contains each of the corresponding values of the signal for each value in "tvec". Together, these can be used to plot the signal on a simple graph using the command "plot(lfp.tvec, lfp.wave)". **Button K** works like buttons J, I, and H. Toggling it to red while a channel is selected will exclude *all units in the selected channel* from batch analysis. The nearby unique checkbox, labeled "**Map**" and double-ringed in green in fig1-1, affects the list of channels in panel 5. Specifically, it toggles a reordering of the list according to the mapping order of the electrode's channel numbers. In our case, since we are using only one type of electrode, this will produce the same order each time. The order of the list will always become 009-008-010-007-013-004-012-005-015-002-016-001-014-003-011-006, as opposed to ascending order of channel number. This ordering corresponds to the "site_map" variable in the "electrode_types" table in the "db_utils" database.

Panel 6, labeled "Units", has a list and a space for a two-dimensional plot, as well as five different buttons (E, F, G, L, and O). It also contains the "**Hide UNClassed**" checkbox (double-ringed in green). The list will be populated once a selection is made in panel 5. This list's entries will contain each unit's id number, an indicator of whether the unit has multi-unit activity (MUA) or single-unit activity (SUA), and the number of its spikes in parentheses. If there are no units in the selected channel with usable data, then the list will contain one entry of "<NOTHING HERE>". When a unit is selected, a two-dimensional graph of that unit's response waveform will be plotted. The checkbox, "Hide UNClassed" will remove from the list units that have not been classified with autoclass. Since autoclass is a first step taken long before we analyze via gui, this will essentially do nothing. **Button E**, "unit note", opens a new window with a text box in which notes pertaining to the unit can be left. **Button F**, "Next", selects the next unit in the list. If there are no more units in the list, the first unit in the next

channel is selected. The **button G**, "Exclude" can be toggled to red to exclude *the selected unit* from batching. The **button L**, "Get Spiketimes", is similar to buttons M and N in that it retrieves data from the database into the MATLAB workspace. Clicking this button will retrieve the selected unit's spiketimes from the database's "spike_data" table in a simple column array. Like the other two buttons, the MATLAB prompt will then display this array's length and size in bytes. **Button O** will open a new window, from which you will be able to save the unit's spike data as a .tsv (tab-separated variables) file. For more information about this, see page 10.

Panel 7 contains two buttons (B and C) as well as a drop-down menu. The label of this section, "Analysis Tools", refers to the additional specialized guis that can be accessed from the drop-down. Each of these tools has a separate script in the "db_analysis" folder (see fig 1-2).

Figure 1-2 below. The DB_UploadUtility is boxed in red, and the db_analysis folder's location is indicated.



Each of the scripts in the db_analysis folder open and run a specialized gui that performs one type of analysis on the unit selected in panel 6. For more information about specific analysis tools, please see page *later this in fill*. **Button B**, labeled "Go", runs the selected analysis tool on the selected unit. **Button C**, labeled "Batch", runs the batch analysis process, which is discussed in its own section on page *fill this in later*.

The common checkboxes, ringed in purple and all labeled "show all", do the same thing for each of the five lists. Each checkbox toggles the addition of extra members to its associated list. When unchecked, only those members with

the "in_use" flag set to one (or "true") in their respective tables are shown. Checking this box shows all those members with the "in_use" flag not set to one. In other words, checking the

"show all" box shows all of the experiments/tanks/blocks/channels/units whether they have useful data or not.

The "Quick Plot" Function

When clicked, **button D** (aka "Quick Plot") opens a new window with information and options pertaining to the unit highlighted when the clicking took place. You may select new items in the DB_Browser window without having to close and reopen the Quick Plot window. The new window has five panels, two buttons, and two checkboxes, as shown in figure 1.1-1.

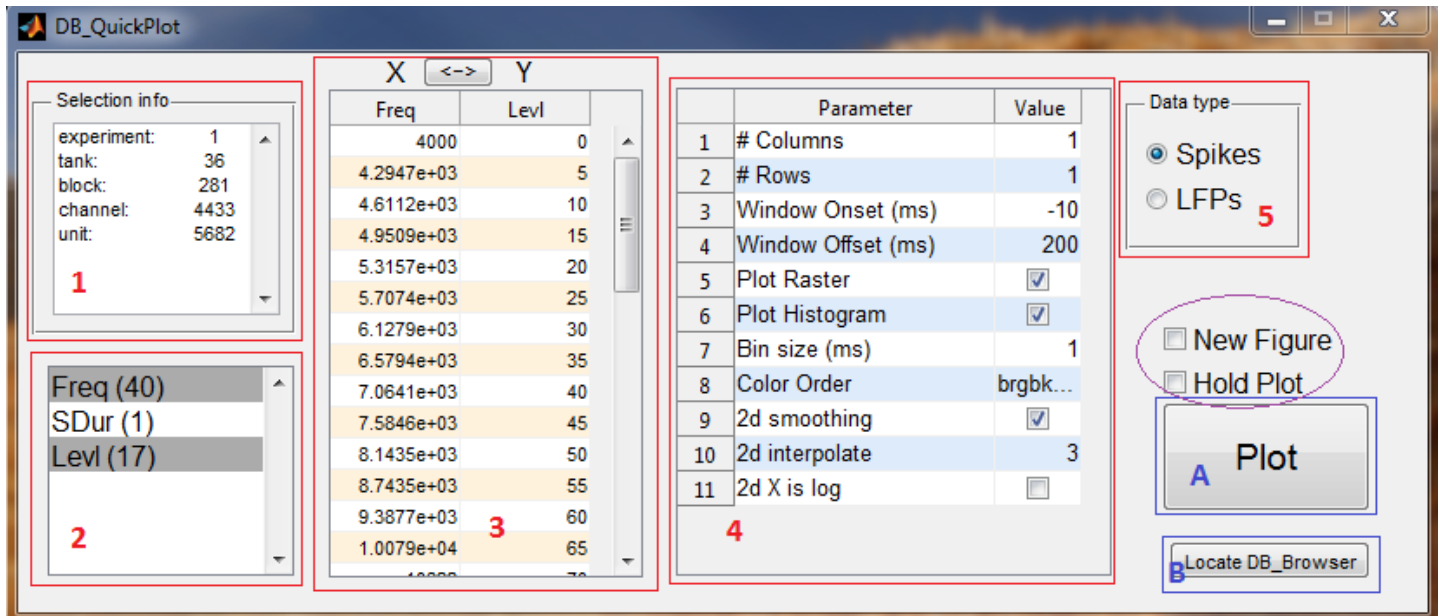


Figure 1.1-1 above. The panels are boxed in red and numbered 1 to 5. The two buttons are boxed in blue and lettered A and B. The two checkboxes are interrelated and are outlined together in purple.

In contrast to the DB_Browser window, we'll first go through the two buttons in alphabetical order, followed by the five panels. We'll then discuss the checkboxes as a unit because they are closely related and mutually exclusive.

Button A, labeled "Plot" opens up yet another window. Unlike with Quick Plot, however, this window will only contain graphs. Plot takes the data in the selected unit, together with the filters and settings from the Quick Plot window, and creates a series of graphs. The type and number of graphs will vary depending on the Quick Plot settings. The new window and its associated graphs are static. Changing the Quick Plot settings or the unit selected will not cause the new window or its contents to change. In order to see the results of the altered settings, button A must be clicked again.

Button B, labeled "Locate DB_Browser", has a very simple function. When clicked, the DB_Browser window will be brought to the front of all other windows, and then the Quick Plot

window will be moved to a position just overlapping it. This button does essentially the same thing as clicking DB_Browser's box in the lowermost taskbar in a Windows distribution. However, it is very useful if one is not using Windows or is running many programs at once.

Panel 1, titled "Selection info", is a text box which contains the identification numbers unique to each of the items selected in the DB_Browser window. These identifiers are equal to the primary keys of the selected items within their respective tables in the database. These numbers specify the exact unit being looked at within the hierarchy of the database. Fittingly, there is no function besides listing this information about your selection in the Selection Info panel.

Panel 2, which has no title, is a list of the parameters relevant to the unit selected. The spike data for each unit is organized based on the values of these parameters. For more information about the parameters and how they relate to block protocols, see page [13](#). Multiple members of this list can be selected at once, but selecting more than two at a time is not recommended. Shift-clicking can highlight all members between the previous selection and the current one (as well as the current one). Ctrl-clicking can be used to build a selection of multiple members one at a time. The selected parameters gain a column in panel 3, and are used to create the axes in the graphs made with button A. If a single parameter is selected, several graphs will be plotted, one for each value of the parameter. If two parameters with multiple values are used, a single colormap will be generated. If a parameter with only a single value is selected in addition to the others, only a single graph will be made. This is likely the result of a bug, as the single graph is simply the first graph that plotted otherwise.

Panel 3 is populated with lists of the values of the parameters selected in panel 2. It contains one column for each parameter highlighted, up to two. If two columns are present, labels of "X" and "Y" will appear above the panel, each indicating which parameter will be used for each of the two axes. If the two columns contain the same number of members, the <-> button can be clicked to interchange the axes the columns will belong to. If the columns have different numbers of values, clicking this button will result in an error. However, it is possible to change the columns' assignments selecting them again in panel 2 in a different order. Specifically, the first parameter to be highlighted will be assigned to the X-axis by default. The next parameter will be assigned to Y. Individual values in the columns can be highlighted by clicking on them and dragging the mouse down the column. This will limit the values used in the plotting to those highlighted. In particular, if only one column is selected, graphs will be created only for the highlighted values, while the colormap's axes will be limited to the highlighted values if two columns are present. If there are two parameters selected, only one of them can be limited in this way. The other must use its full range of values.

Panel 4 contains settings for the style and layout of the graphs to be plotted. Each of the graph parameters in the "Parameter" column can be adjusted by clicking the checkbox or highlighting and editing the number in the "Value" column. The specific graph parameters displayed will differ based on the selection of the data type in panel 5. If "Spikes" is selected in panel 5, there will be 11 graph parameters. If "LFPs" is selected, the rows for "Plot Raster", "Plot Histogram", and "Bin size (ms)" will be replaced by "Error Band", for a total of 9 graph parameters. The individual definitions for the parameters common to both of panel 5's data types are as follows . . .

Columns - Number of vertical columns of graphs to be plotted. Decreasing this number such that the product of # Columns and # Rows is less than the total number of values in the single column in panel 3 will result in graphs being left out. Increasing this number such that the product of # Columns and # Rows is greater than the total number of values in the single column in panel 3 will result in the plot window being padded with extra space. This value and # Rows will both be set to 1 if more than one column is present in panel 3, and any changes to these values will be ignored.

Rows - Number of horizontal rows of graphs to be plotted. Decreasing this number such that the product of # Columns and # Rows is less than the total number of values in the single column in panel 3 will result in graphs being left out. Increasing this number such that the product of # Columns and # Rows is greater than the total number of values in the single column in panel 3 will result in the plot window being padded with extra space. This value and # Columns will both be set to 1 if more than one column is present in panel 3, and any changes to these values will be ignored.

Window Onset - Taken to be the time in milliseconds before the response to the presented stimulus appears. If this value t is negative, spikes that were recorded t milliseconds *before* the stimulus presentation will be included in the plotting. If this value t is positive, only spikes that were recorded t milliseconds *after* the stimulus presentation will be in the plotting. The total length of the graphs' x-axes will be equal to $(-t) + (u)$, where t is the value in Window Onset and u is the value in Window Offset. This value is set to -10 by default.

Window Offset - Taken to be the time in milliseconds it takes for the response to the presented stimulus to stop. This value sets the cutoff point of spikes counted for each presented stimulus. Only spikes that were recorded within u milliseconds after the stimulus presentation will be used in plotting, where u is this value. The total length of the graphs' x-axes will be equal to $(-t) + (u)$, where t is the value in Window Onset and u is the value in Window Offset. This value is set to 200 by default.

Color Order - When either of the checkboxes ringed in purple are checked, this determines the coloration of the additional graphs to be plotted. This value forms a list of colors, and each additional graph will be the next color down the list. Eventually, this will ring back around to the first color (blue) in this list.

2d smoothing - This graph parameter only applies if two columns are present in panel 3 so that a colormap will be plotted. When checked, the data to be plotted will be smoothed first.

2d interpolate - This graph parameter only applies if two columns are present in panel 3 so that a colormap will be plotted. This value indicates the degree of interpolation that will be used on the dataset before it is plotted.

2d X is log - This graph parameter only applies if two columns are present in panel 3 so that a colormap will be plotted. When checked, the x-axis of the resultant colormap will be logarithmic.

When the "Spikes" data type in panel 5 is selected, three additional graph parameters will be present. They are as follows . . .

Plot Raster - This graph parameter only applies if a single column is present in panel 3. When checked, each graph will include a scatterplot with points representing spikes. For these points, x represents the time at which the spike occurred, and y indicates which presentation of the graph's parameter (as in frequency or decibel level) value prompted it.

Plot Histogram - This graph parameter only applies if a single column is present in panel 3. When checked, each graph will include a histogram representing the number of spikes occurring in each bin. Each bin represents a duration of time within the total time of the window.

Bin size - This graph parameter only applies if a single column is present in panel 3. The value of this parameter is the bin width of the histogram, in milliseconds.

When the "LFPs" data type in panel 5 is selected, one additional graph parameter will be present . . .

Error Band - This graph parameter currently does nothing. It will likely be implemented some time in the future.

Panel 5, labeled as "Data Type", contains nothing more than a selection between either "Spikes" or "LFPs". Whichever is selected will be the data that will be plotted for the selected unit. The parameters in panel 2 are merely used to frame the presentation of this data. When

"Spikes" is selected, the spike times will be retrieved from the database (the "spike_data" table). These are discrete values indicating the location in time at which relevant neural activity was recorded. When "LFPs" is selected, the waveforms of the recorded local field potentials will be retrieved from the database (from the "wave_data" table). These are signal functions of the voltage changes indicating the localized cortical activity.

The checkboxes labeled **"New Figure"** and **"Hold Plot"** are mutually exclusive. If "New Figure" is checked, "Hold Plot" vanishes and its effect does likewise. If "Hold Plot" is checked, "New Figure" is automatically unchecked. When checked, "New Figure" causes new windows with new graphs to be created with each subsequent click of button A. Each new window will have its own independent set of graphs, which will be colored according to this window's position in the Color Order sequence from panel 4. "Hold Plot" causes new subsequently plotted graphs to be laid over ones already present in the foremost window. Each new overlay is drawn with the next color in the color order sequence in panel 4. These functions are useful for comparing data both within and across individual units.

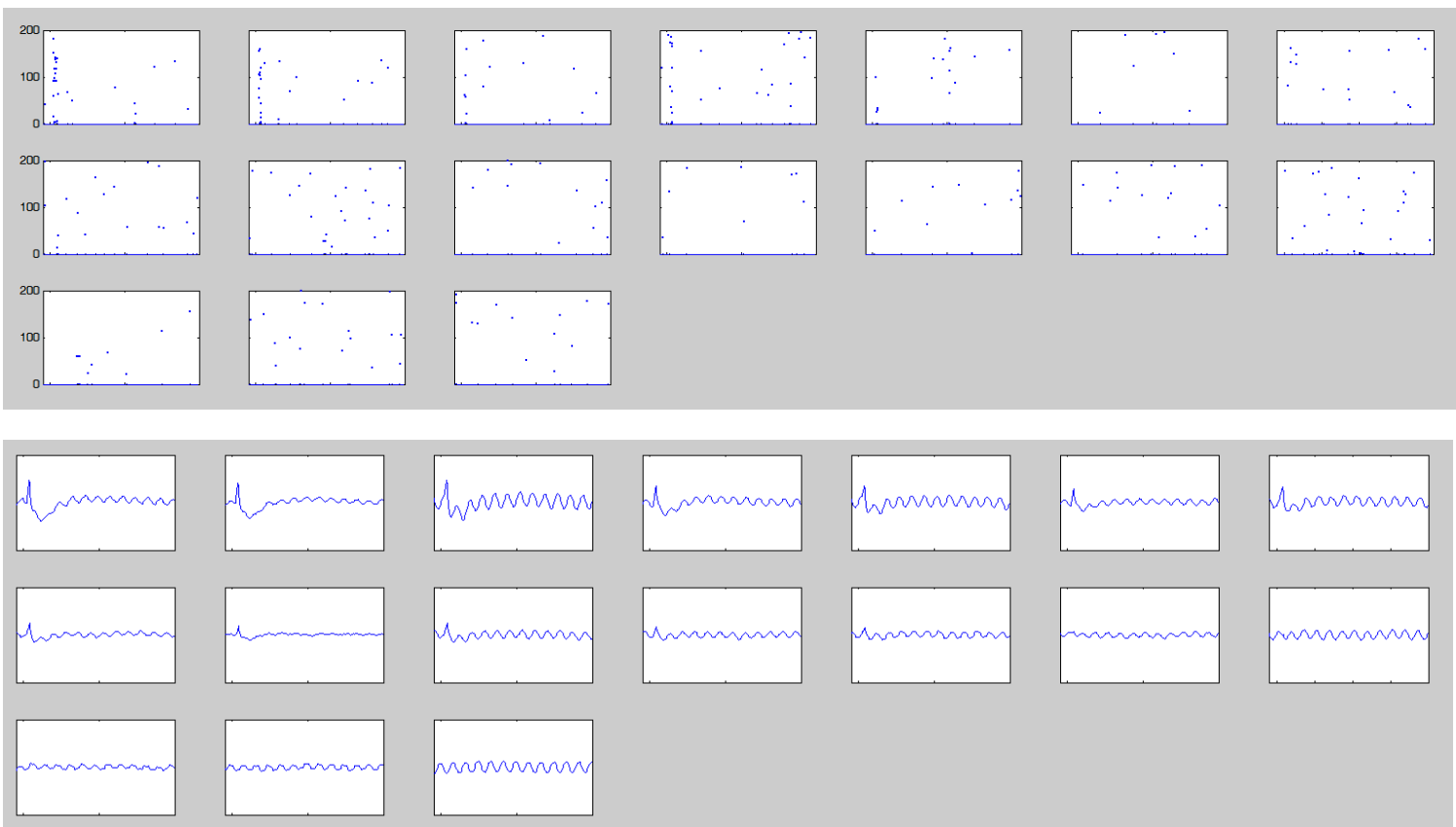


Fig 1.1-2 above. The upper image shows an output of the plotting with the "Spikes" data type selected in panel 5. Note the histograms along the bottom of each graph, and the scatterplot overlays. The lower image shows an output of the plotting with the "LFPs" data type selected in panel 5. Each of these graphs is simply a waveform of the LFP data.

The "Export Unit Data" Function

In the initial DB_Browser window, **button O** is labeled with "Export Unit Data". When clicked, this button opens a new window as shown in figure 1.2-1. It always has three panels (boxed in red), two buttons (boxed in blue), and a single drop-down menu (circled in purple). The operations performed from this window involve only the block containing the unit selected in the DB_Browser window (the "exporting block") when the "Export Unit Data" button was clicked.

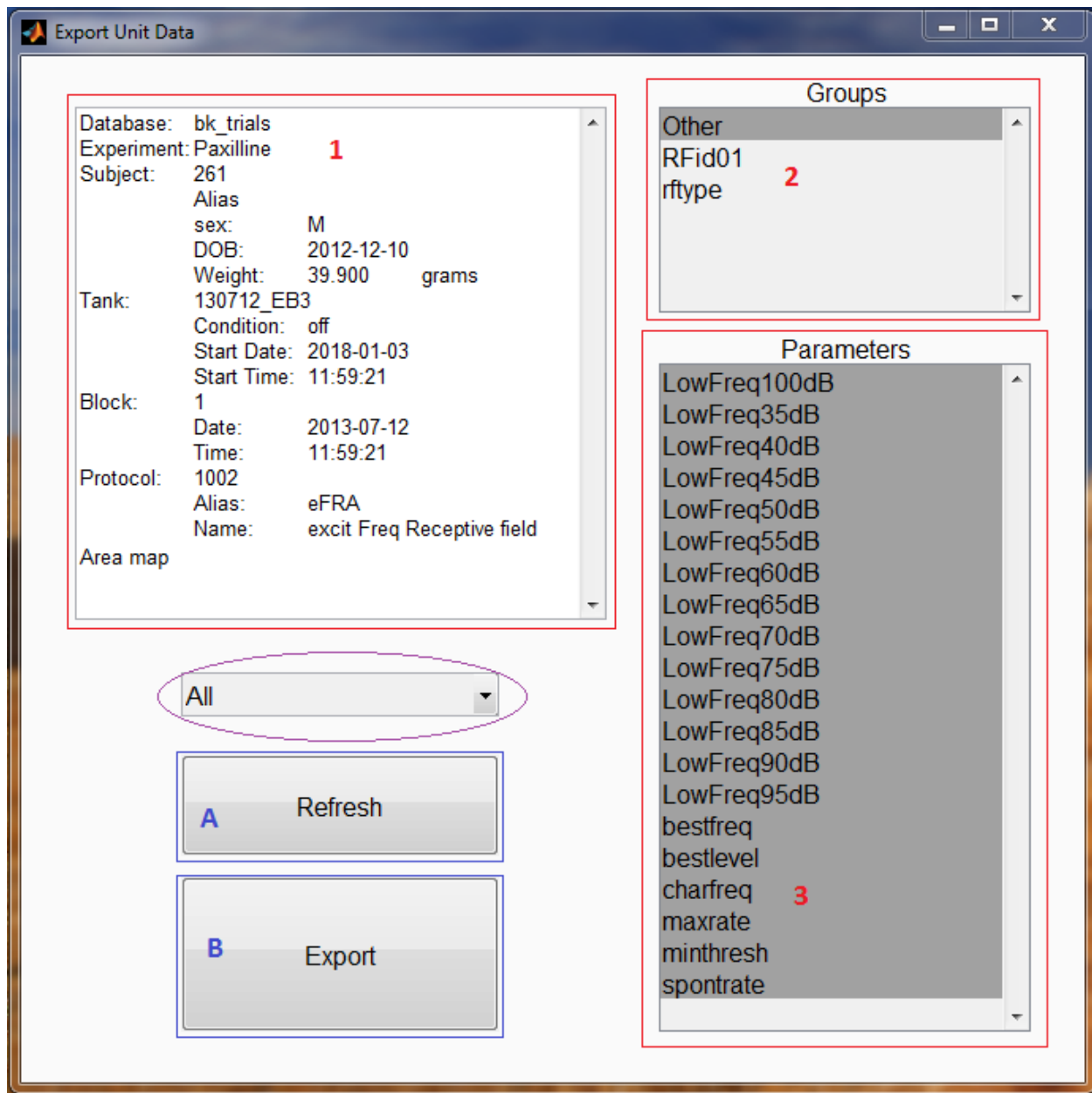


Fig 1.2-1 above. The panels are boxed in red and numbered 1 to 3. The buttons are boxed in blue and lettered A and B. The lone drop-down menu is circled in purple.

This window is populated with lists of unit parameters and the associated data drawn from the database. We'll go through each of the three panels in numerical order, then the drop-down menu, and finally the two buttons in alphabetical order.

Panel 1 is simply an information panel. It describes the details of the unit that are higher up in the hierarchical data structure. The lines beginning with "Database" and "Experiment" list the name of the database in which the unit is found and the type of experiment the unit was part of, respectively. The line "Subject" lists the identification number

of the actual mouse the exporting block was derived from. The four indented lines following this one list information specific to the mouse subject. The "Alias" line is left blank, as the mice are identified with their subject numbers. Following this, the subject mouse's sex, date of birth, and weight at the time of the experiment are listed. "Tank" refers to the name of the tank to which the selected unit and its block belong. The "Condition" line refers to whether the tank is off or a Paxilline test, and contains the same condition information referred to in panel 3 of the initial DB_Browser window. The "Start Date" line *should* refer to the date on which the recording began, but due to an error with the timestamps this number will always be incorrect. This is no big loss, as the name of the tank includes the date in its first six characters (yymmdd), by our lab's convention. The "Start Time" refers to the time at which the recording for this tank began, using military time notation. The "Block" line indicates exporting block's identifier within its tank. This number is taken from the set of blocks within the tank, and is not unique across all blocks. This section also has (correct) "Date" and "Time" lines indicating when each block was recorded. The "Protocol" line lists the identifier of the stimulus presentation and recording protocol used for this block. The "Alias" line here refers to the protocol abbreviation, while the "Name" line refers to the protocol's full name. For more information about experimental protocols, see page [13](#). The "Area map" line will be blank, as there is no corresponding data structure for the units, blocks, or tanks in the database.

Panel 2 is titled with the header, "Groups", and refers to the groups of parameters that are used to describe the data belonging to each unit. These unit parameters are organized into several overlapping groups, each of which contain units from certain types of blocks. The groups displayed in this list will depend on the category option chosen from the single drop-down menu as well as the properties of the categories to which the exporting block belong. Only one member of this list can be highlighted at a time, and this selection will affect what parameters are displayed in panel 3. Essentially, the group selection is the framework in which the exported unit data will displayed.

Panel 3 is headed with "Parameters", and contains a list of the properties describing the data associated with the units contained within the exporting block. The parameters listed will depend on the group selected in panel 2, with some parameters being exclusive to specific groups. All parameters will start out highlighted, but a narrower selection can be chosen. Multiple members of this list can be highlighted at once by shift-clicking, and a piecemeal set of parameters can be constructed one at a time by ctrl-clicking. The highlighted set of parameters will form the columns generated when button B is clicked and the unit data is exported.

The drop-down menu, **ringed in purple**, contains a list of the options "All", "Experiment", "Tank", and "Block". Each of these options is a choice as to the scope of the data

to be exported. Likewise, the possible selections of the lists in panels 2 and 3 will be narrowed to match what is contained in the selected scope. If "Block" is selected, only the exporting block will be prepared for exportation. If "Tank" is selected, then all blocks within the exporting block's tank that share its protocol will be prepared. If "Experiment" is selected, all blocks within the exporting block's experiment that share its protocol will be prepared. If "All" is selected, then all blocks within the database that share the exporting block's protocol will be prepared. Each time a new scope is selected, the data kept on hand for this window is reassembled, essentially refreshing the dataset.

Button A is labeled "Refresh" and reassembles the lists of groups and parameters for the exporting block. Each member of the lists in panel 2 and panel 3 is retrieved from the database along with its associated values. This is done when the "Export Unit Data" window is first opened, and clicking button A repeats this process. This is useful for assuring the fidelity of the exporting block's data in the case of the database's content being updated or something going wrong with the connection.

Button B is labeled "Export" and creates a .tsv file out of the exporting block's data and the options set within this window. A .tsv file is a "tab-separated values" file, and can be opened in Microsoft Excel as a spreadsheet. This spreadsheet will appear similar to figure 1.2-2, and will contain the specifying information in panel 1 as well as a table. This table will have the values of the parameters selected in panel 3 for each unit in the exporting block. This table will also have columns for the unit pool (multi- or single-unit activity), the number of recorded spikes within it, and all of the hierarchical identification data contained within the scope selected with the drop-down menu. The new file is unnamed and can be saved to any directory.

E13

Database: bk_trials

Experiment: Paxilline

Subject: 261

Alias:

sex: M

DOB: #####

Weight: 39.9 grams

Tank: 130712_EB3

Condition off

Start Date 1/3/2018

Start Time 11:59:21

Block: 1

Date: #####

Time: 11:59:21

Protocol: 1002

Alias: eFRA

Name: excit Freq Receptive field Area map

Channel	Unit	Pool	Count	LowFreq1	LowFreq3	LowFreq4	LowFreq4	LowFreq5	LowFreq5	LowFreq6	LowFreq6	LowFreq7	LowFreq7	LowFreq8	LowFreq8	LowFreq9	LowFreq9	bestfreq	bestlevel	charfreq	maxrate	minthresh	spontrate
1	5497 MUA		272	5	7141	15275	6062	16401										12808	0.33333	57	0.022222		0
2	5499 MUA		1546	1	9431	16553	7715	19560										13879	2.6	52	0.066667		0
3	5501 MUA		1719	5	10095	15465	8921	16811	8207	17939								13088	3.0667	50	0.077778		0
4	5502 MUA		217	5	8232	15228												12572	0.93333	65	0.005556		0
5	5503 MUA		58		8743.46	12475.5	8743.46	10079.4										9411.43	0.13333	60	0		1
6	5504 MUA		67		10849.4	10849.4	6127.87	13394.7										10849.4	0.26667	60	0.005556		1
7	5506 MUA		2252	5	7573	14227	7010	16451	6310	22829								9788	2.2667	42	0.10556		0
7	5507 SU1		1900	1	8334	12152	7434	13794	7434	14628	6924	14948	6137	16149	4576	22759		9727	1.7333	16	0.12778		0
8	5508 MUA		7919	5	28169	37551	10221	41841	8031	43288	6610	46191						31582	7.7333	40	0.16667		0
9	5510 SU1		6640	5	11008	15755												13624	3.4667	65	0.37778		0
11	5512 MUA		304	5	6924	21930												8811	1	64	0.011111		0
12	5514 MUA		10532	1	8258	11424	7908	12152	7596	15275	7097	19560	5859	22971	4094	25359		9373	3.2667	15	0.55		0
13	5516 MUA		5732	5	7207	19083	5340	21066										11388	2.9333	60	0.26111		0
13	5517 SU1		6386	5	48159.3	59608.1	48159.3	59608.1	8143.46	59608.1	6579.37	59608.1	4950.91	59608.1			23	57883	2.3333	25	0.34444		1
15	5520 MUA		2329	5	29324	37089	9909	38970	6509	37784								32776	3.2	50	0.066667		0
16	5522 MUA		1384	1	27737	39455	26236	41712	18616	38730	16708	39090						32273	1.8	35	0.038889		0
16	5523 SU1		2243	5	40945	60070	39211	61006	40069	61384	40317	63704	41583	62921	41199	63704		49902	1.2	15	0.094444		1

Created on #####

Figure 1.2-2 above. This is an example of what the spreadsheet resulting from "Export Unit Data" will look like. The area boxed in red gives the identification details of the subject mouse, the tank, and the block. The area boxed in blue is the collected unit data contained within the selected scope. This example's scope was set to "Block" in the drop-down menu, which is ringed in purple in fig. 1.2-1.

1.3 - Protocols and Their Definitions

The stimulus presentation in each of the recording experiments varies along one of few different sets of parameters. In kind, each recording keeps track of one a few sets of variables along with the spike time capture. These sets are protocols, and their use allows the same experimental setup to be used to gather different types of information. The protocols currently in use are . . .

eFRA - "Excitatory Receptive Field Area". This protocol varies the frequency and intensity level of the stimulus, and records both quantities. This allows a three - dimensional representation to be made which plots the number of spikes at each frequency and intensity level pair.

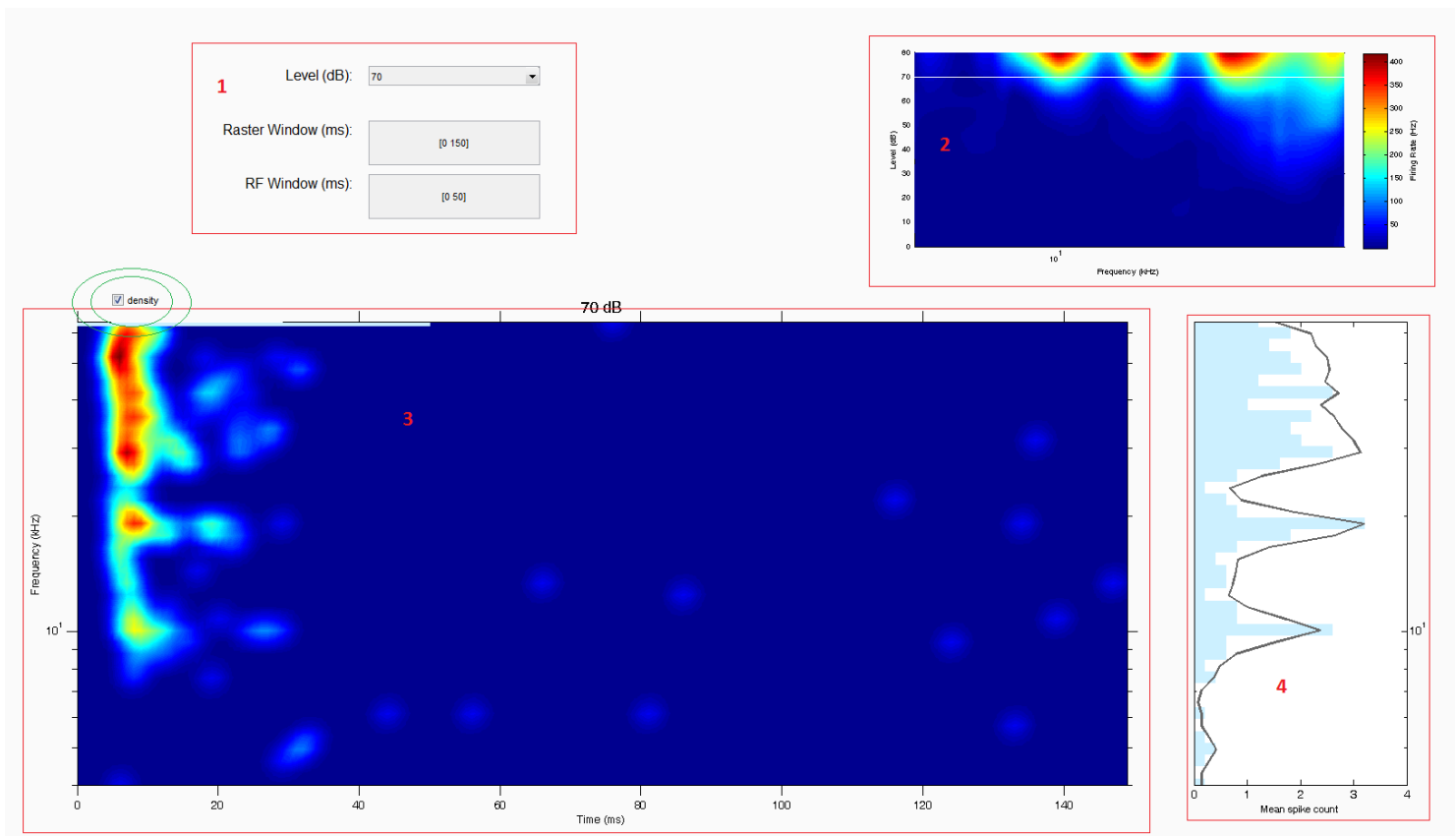
RIF - "Rate Intensity Function". This protocol varies the intensity level of the stimulus, and uses this plus the response activity to create a two - dimensional graph.

Gaps - This protocol creates pauses (hence "gaps") in the stimulus which produces additional neural activity as the subject detects and interprets the pauses in the sound. Variables keeping track of when the gap occurs, its length, and the frequencies of any background stimuli are recorded.

Add the rest of the protocols to this section later.

Analysis Tool: RF_FreqVsTime

In the section of the DB_Browser window labeled "Analysis Tools" is a drop-down menu. This menu contains a selection of add-on scripts that produce the results of specialized analyses. The first of these we'll go through is the "RF_FreqVsTime" script. To make use of this script, select the unit to be analyzed, then select this script from the drop-down menu and click "Go" (or "Batch", see page *fill this in later*). A new window will open up. As is shown in figure 2-1, this new window will have four panels (boxed in red), and a checkbox labeled "density"



(double-ringed in green).

Fig. 2-1 above. The four panels are boxed in red and numbered 1 to 4. The single checkbox "density" is circled twice in green.

We'll go through each of the panels in order. We'll go over the checkbox alongside panel 3, as they are interrelated.

Panel 1 contains a drop-down menu labeled "Level (dB)", and two text input boxes, labeled "Raster Window (ms)" and "RF Window (ms)". These form the majority of the controls

for the rest of the window's display. The vectors in the two text input boxes are the ranges which define the time interval over which the graph in panel 3 is plotted (Raster), and the duration after the stimulus is presented in which spikes are counted (RF Window). To change these and see the effects, simply click within the box and use the keyboard to edit the vectors. Hitting Enter will display the results. The drop-down menu will contain a list of the intensity levels, which usually span from 0 to 80 in steps of 5. Also take note of the white line across the graph in panel 2. This line corresponds to the level selected in the drop-down menu. Selecting a new level from this menu will change the position of the white line in panel 2, and clicking on the graph in panel 2 will change the value in the drop-down box accordingly.

Speaking of **Panel 2**, the colormap it contains is a representation of the Response Field Area, or the spike density at each frequency and each intensity.