CAMBRIDGE ELECTRONIC DESIGN LIMITED

eNEWSLETTER

#11
April 2021

Contents

Welcome

Training days

Latest software

Here to help

Script Spotlight

- MatlabXXX()

Spike2

- Getting started with spike sorting
- Script – ECG artifact remove

Signal

- Memory views
- Script – Pulse train setup

Scripters corner

- Toolbars

Recent questions

- Decontaminating equipment after COVID-19

Contact Us

Welcome

Thank you for downloading our April newsletter. Spike2 version 10.09c and Signal versions 7.06 and 6.06 were all released this month; please ensure to update your copy.

Our software engineers have recently been working on importing more third-party file formats into Spike2 and Signal. A researcher recently requested to import extensible data format files (XDF), which we have achieved and will soon be ready for release. We have had further requests for import and export of NWB (Neurodata Without Borders, an open-source repository of data), as well as HDF5 (hierarchical data) formats; we would like to hear your thoughts on these file formats along with any others you would like available for import into Spike2.

We would also like to hear from you if you are using Lab Streaming Layer (LSL), a system for centralised collection and viewing of data, and are interested in seeing its integration with Spike2 via a Talker module.

In a similar vein, our software engineers have been improving the current library of importers. We would be interested to hear from you about which of the available file formats you import most often and how they could be improved. Whilst we endeavour to maintain accurate records of the file formats we support; developers sometimes alter the structures of files without our knowledge.

Please send your comments to Marjorie@ced.co.uk along with small snippet of a data file for testing.

Training

Our first free 2021 online are now complete; however, the full recordings may be viewed at any time on our website. We have more sessions planned for later in the year, for which we will share details later.

We also offer remote training sessions by Skype/Zoom/Teams either one-to-one or with groups. Join us and learn how to make the best use of Spike2 and Signal to save hours of repetitive analysis. Our sessions are free to arrange and are suitable for both existing and prospective users of our data acquisition and analysis systems. If you would like to schedule a session, please get in touch: Marjorie@ced.co.uk.

If you see these buttons in our newsletters, it means a file or script relating to the section is available to download:



Latest versions of Spike2 and Signal

Spike2	Released	Signal	Released
Version 10.09c	04/2021	Version 7.06	04/2021
Version 9.13	02/2021	Version 6.06	04/2021
Version 8.21	02/2021	Version 5.12a	02/2018
Demo	04/2021	Demo	04/2021

Back to contents

Here to help

We know access to laboratories has been erratic for the past year, but with the current lockdown measures easing for some we hope that conditions will continue to improve. We are now able to offer site visits when necessary, however we will continue to support remotely in the first instance. CED will also continue to do all in our power to support you for increased home working. Should you require any help or wish to discuss your system, email Marjorie@ced.co.uk and we can arrange for a video call via Skype/Zoom/Teams.

We also have tutorial videos for both Spike2 and Signal available on our website for you to peruse at your leisure. There are new videos and updates in the pipeline. If you have a particular topic you think could benefit from a tutorial video, please let us know. All these videos are also available through our YouTube channels: Spike2 / Signal.

Try the CED Forums bulletin board for further software and hardware support. Ask your questions and receive valuable input from our super users.

Back to contents

Script Spotlight – MatlabXXX()

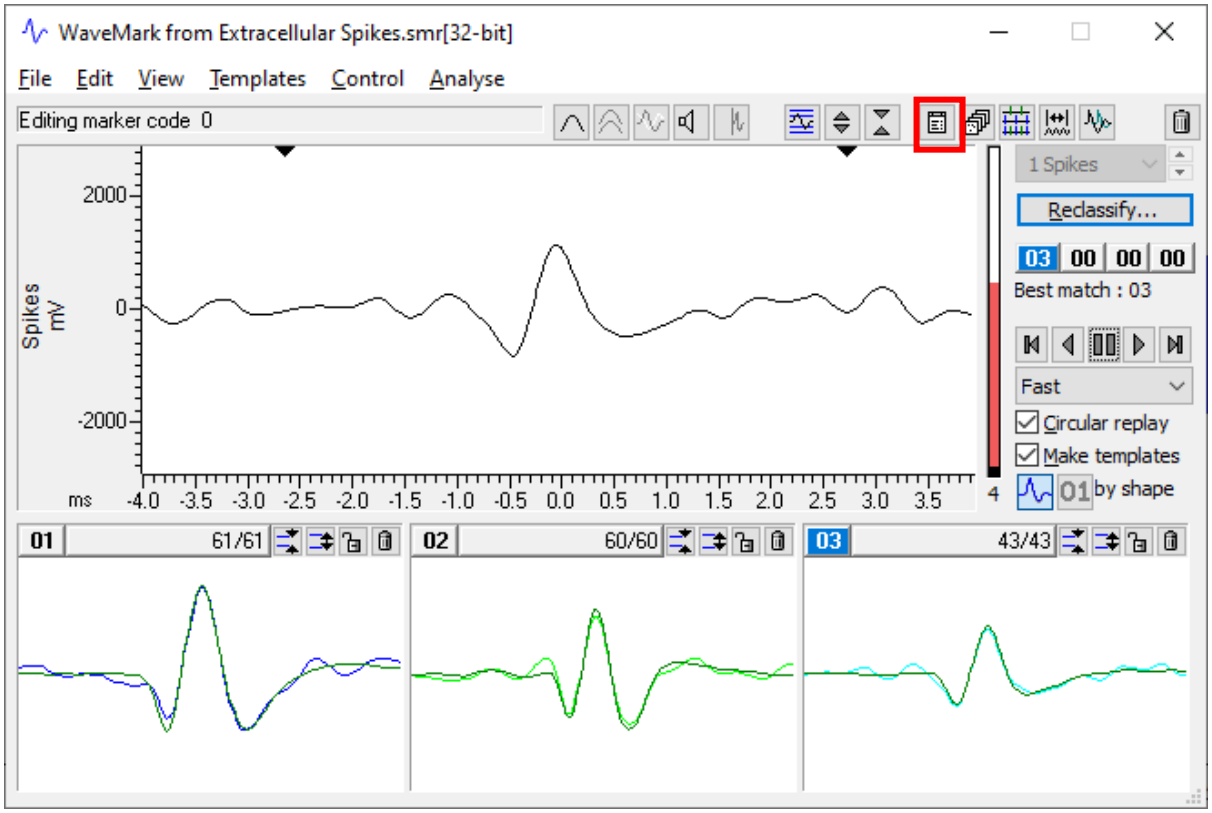
Many of our users wish to transfer data from Spike2 or Signal to MATLAB, and vice versa, during experiments. If you have MATLAB™ installed on your computer, both Spike2 and Signal can interact directly with a copy of MATLAB using the built-in MatlabXXX() family of script commands which let you start a MATLAB process to use as a computational engine. These include functions to transfer script variables back and forth between a MATLAB workspace, allowing you to use MATLAB for additional data processing, before transferring the results back to Spike2 or Signal.

To access the MatlabXXX() script commands, you should select the MATLAB script support option when installing Spike2 and Signal. An example script to check if the script support is installed and working is included in the Spike2 on-line help topic *Script language* > *Alphabetical script function index* > *M* > *MATLAB script support* (Signal users can view this on our website here). The example illustrates use of all the functions and most of the data transfer options. Full details of the functions may also be found in the same help location.

Back to contents

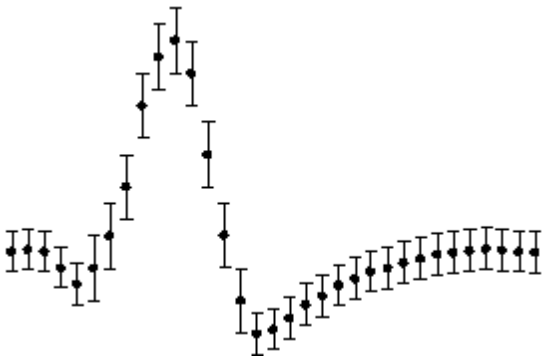
How do I get started with Spike sorting?

In our last newsletter we covered off-line template creation but did not delve in the rules applied from the template settings dialog. There are several parameters that control spike template formation and how incoming spikes are matched to templates. You can modify these parameters in the *Template Settings* dialog accessed through *Ctrl + Home* or from the toolbar button (outlined in red) in the spike shape window:



As described in the previous newsletter, the vertical bar to the right of the data display area indicates the number of provisional (grey) and confirmed (black) templates. If your choice of settings or template area generates many provisional templates, the provisional bar turns red, indicating a potential problem. In the case above, the template area includes more baseline than spike shape; reducing the area (by dragging the black triangles) to the range -1.0 to 1.3 ms would fix this.

A template is stored as a series of data points. Each point has a width range associated with it that represents the expected error in a signal that matches the template at that point. Spikes match a template if more than a certain percentage of the points in a spike fall within the template. The template settings dialog has four areas for: creation of new templates, matching incoming spike data to templates, template modification, and processing the raw waveform data into spikes.

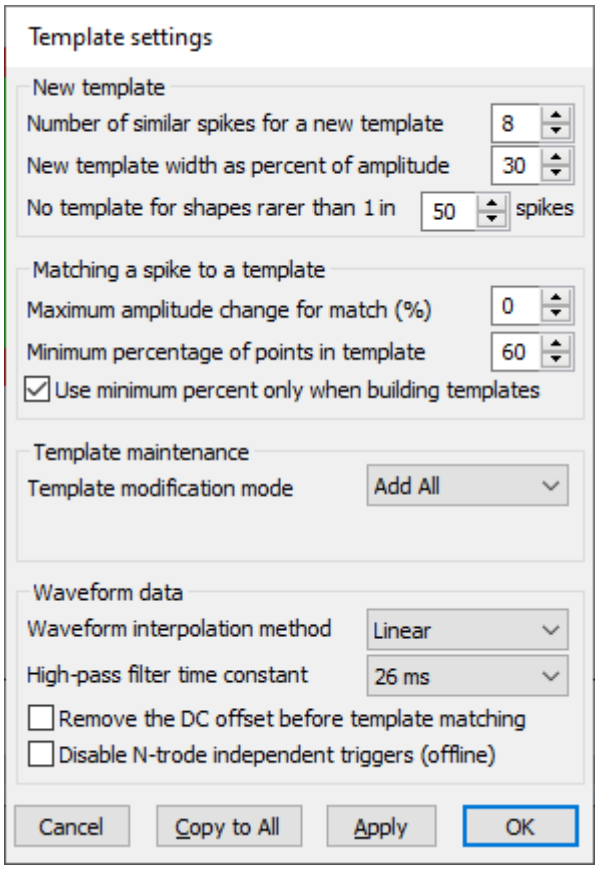


New template – controls the creation of new templates.

Number of similar spikes for a new template

Sets the number of spikes required to promote a provisional template to a confirmed template. The default of 8 is usually a good starting point. If set too low, you will generate multiple confirmed templates for similar spikes.

New template width as percent of amplitude
Determines the initial provisional template width (when we have only one spike to estimate it). As spikes are added to a template, the template width changes to represent the amplitude variation of the spikes and general noise. 30% is a reasonable starting value but should be increased if you suspect too few templates are being generated.



No template for shapes rarer than 1 in n spikes
This is roughly the same as saying that you are not interested in spike classes which occur less often than once every n spikes. It works by decrementing all the provisional spike counts every n spikes. If a provisional count reaches 0, that provisional template is deleted. If you want to keep all spikes as potential templates, set this to a large number.

Matching a spike to a template – This section determines how incoming spikes are matched to existing templates.

Unless *Maximum amplitude change for match* is non-zero, if the error between a spike and a template exceeds a value calculated from the template then the spike is rejected before applying the following rules. This avoids wasting time interpolating spike shapes for data that could never match.

Maximum amplitude change for match %
Scales incoming spikes up or down by up to the value so that the area under the spike matches the area under the target template before attempting to match. Use this setting if you have spikes that maintain their shape but change amplitude during recording. As a default this setting is 0.

Minimum percentage of points in template
Sets the percentage of the spike data points that must fit in the template to confirm a match. This means that we do not consider a template if less than the user-specified percentage of data points lie within the template boundaries (defined as the mean template plus or minus the width at each point).

Use minimum percentage only when building templates
Specifies that the Minimum percentage... setting is only used when building templates. If this option is checked then the number of points in the template are ignored and spikes are matched to the template with the smallest error between the mean template and the spike.

Template maintenance – This section determines how the shape of a template changes over time as more spikes are added to it.

Template modification mode
Three options are available from the drop-down list:

Add All – add all spikes that match the template, modifying the template accordingly to represent the amplitude variation of each spike

Auto Fix – 'Lock' the template once a set number of spikes has been added. If you have several similar spikes, using this mode to lock templates after a small number of spikes can stop a template gradually changing shape and becoming the same as another template

Track – Normally, all spikes contribute equally to the template. With this option, the contribution of each spike to the template falls by a factor of 1/N as each new spike is added (N is the value entered). This weights the template shape towards the most recently seen spikes, so you can think of this as tracking the incoming spike shapes. The lower the N, the more aggressive the tracking; start with a value in the 60 to 100 range. This mode is useful for gradual changes in spike shapes; with fast changes, it is likely that templates will merge.

Waveform data – This final group of settings controls how the raw waveform data is processed for spike sorting.

Waveform interpolation method
Sets how the spike waveforms are aligned with templates using linear, parabolic, or cubic spline interpolation. For on-line sorting, linear interpolation should be used as it is fastest.

High-pass filter time constant
If your spike waveforms show baseline drift you can use this to remove slow changes in the baseline. Set the longest time constant that holds your data to the baseline to minimize spike distortion. This is applied before triggering levels are assessed.

Remove the DC offset
Subtracts the mean of each spike before matching; can be useful with sudden DC shifts. This is applied after triggering.

Disable N-trode independent triggers (off-line only) *Added in [10.01]*
When dealing with Stereotrode or Tetrode data, for triggering purposes all channels are treated independently with the provision that if two triggers occur within 4 samples of each other, only the first is used. This can lead to highly overlapped traces due to triggering events being detected on separate traces. By checking this box, new trigger events that occur within the last captured spike shape are ignored. Events can still overlap, but only by the pre-trigger time of the spike. This is implemented only for off-line sorting when creating new stereotrode or tetrode channels.

Back to contents

Scripts: Spike2

EMG electrodes often pick up ECG activity in addition to the muscle activity you are trying to record. This is especially problematic when trying to record the activity of small muscles or muscles located close to the heart, like the diaphragm. At best, the ECG artifacts can be a minor visual annoyance and at worst, can make responses difficult to analyse. Filtering EMG recordings often does not help as the frequency bands of ECG and EMG overlap.

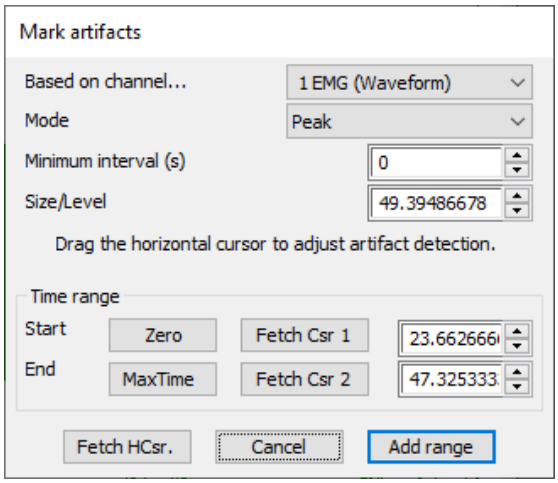
However, if the ECG artifact remains stable, it is possible to generate an average ECG artifact and subtract this average, point-for-point from each individual artifact. Our script writers have recently updated the ECGDelete.s2s script available from our website which does just that. This script replaces each artifact with the difference between the artifact and the mean artifact. This "reference noise subtraction" method will generally replace ECG artifacts with much smaller ones - provided that the artifacts remain consistent in size and shape over the time range analysed. If the artifacts vary widely in shape and size then the results may not be optimal, instead giving artifacts as bad or worse than the original.

The script has several features to help cope with inconsistencies in artifact shape and amplitude:

- There is a checkbox option to scale the 'mean artifact' to the same amplitude as each 'target artifact' before subtraction. This may improve results depending on the situation.
- The script can compensate for longer term changes in the shape of the artifact during a recording by dividing the time range into a user-defined number of epochs, with a new average artifact generated for each time range.

- You can also analyse multiple time ranges interactively, with or without gaps between them, provided the data sections do not overlap and are analysed from left to right.

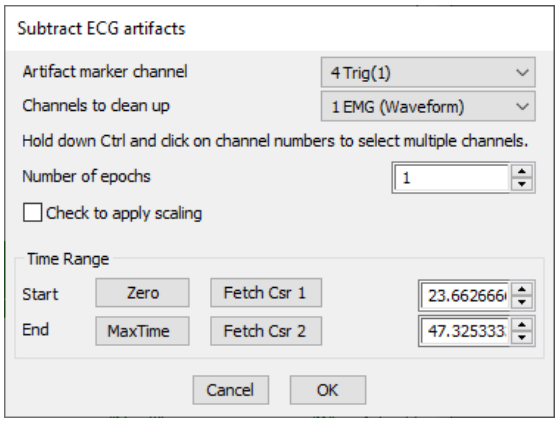
The script operates on the Front view which must be a time view containing the EMG data that you want to remove artifacts from. To use the script, first close all other time views then run the script.



Use the *Mark ECG* button to select a waveform channel as the source and create a trigger channel marking peaks, troughs, or level crossings of the ECG artifact. Note that artifact removal usually works best if you use peak or trough detection rather than level crossings. These triggers will be used to create an average artifact for subtraction. If the artifact is consistently larger than the EMG of interest, then you can use one of the EMG channels as the source. If not, you will need to record an additional channel of ECG for this purpose. The minimum Interval in the dialog sets the minimum time that must elapse between consecutive detected triggers.

The script uses a horizontal cursor to aid setting the detection threshold. Click on *Fetch HCsr* to add the cursor to the relevant channel and then drag it slowly down from the top edge of the channel to set the level. The number of triggers detected in the *Range* memory channel will adjust to the level that you set interactively. This should make it easy to set a level that marks the greatest possible proportion of the triggers you want to include. Note that the H cursor position represents the threshold level in "threshold" modes. In Peak or trough modes, the relevant parameter is the "level difference" between the horizontal cursor and low edge of the Y-axis. You can make fine adjustments to the position of the H Cursor by editing the *Size / Level* item in the dialog.

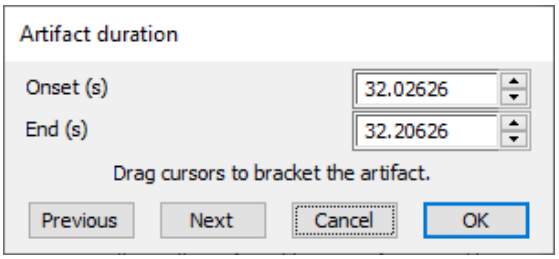
Next, select the time range to process either by dragging the cursors, pressing the associated buttons in the dialog, or by clicking the spinner arrows in the dialog time items. Click *Add Range* to save the current set of triggers to a disk channel. *Mark ECG* may be used again to add multiple time ranges each with their own detection criteria. The only proviso is that new ranges are added from left to right without overlaps. Note that you should use the same detection mode for all time ranges.

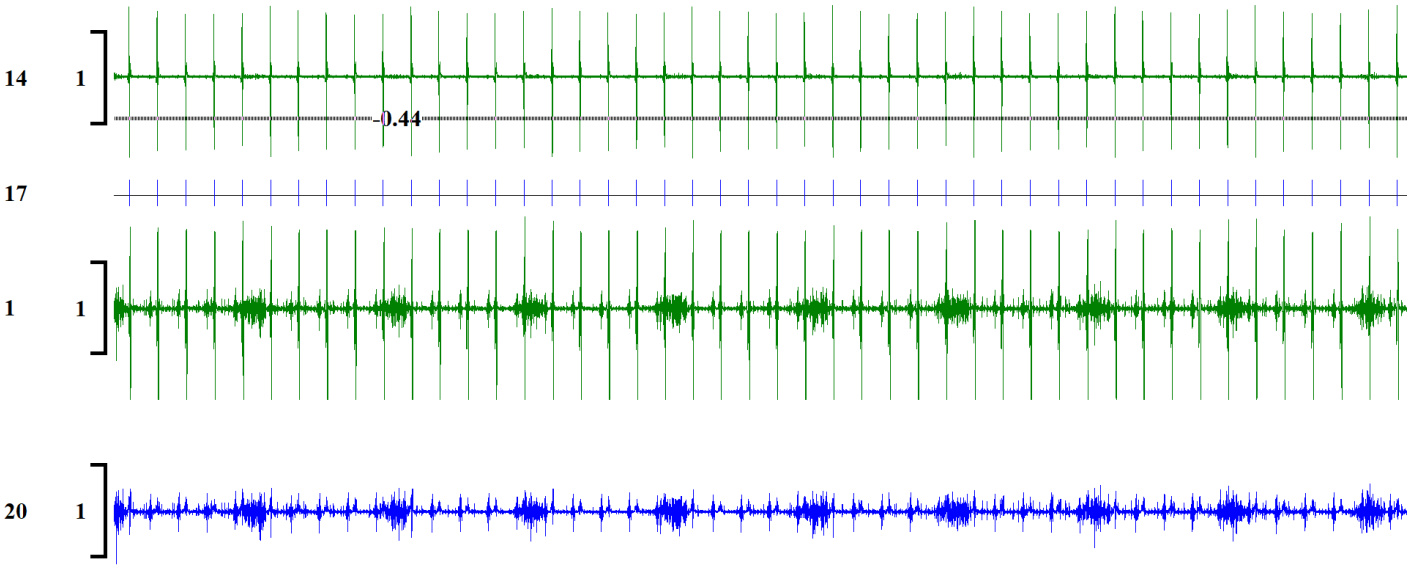


Use *Subtract ECG* when you are ready to create new data channels with the mean ECG signals subtracted. In the setup dialog, select the channel containing the triggers and the waveform channels to clean up. You can select an individual waveform channel from the drop-down list or choose multiple channels by choosing *Selected* from the list and selecting channel numbers by holding down *Ctrl* and clicking on channel numbers in the time view. Duplicate channels cannot be selected.

Select a time range to analyse and choose whether to analyse the entire time range in one go or to subdivide the time range into several epochs each with its own local mean artifact. There is also the checkbox that the scales the mean artifact to the amplitude of each individual artifact before subtraction.

Click *OK* when ready and a final dialog opens to specify the duration of the artifact to be subtracted. Do this by dragging the 'Start' and 'End' cursors in the time view relative to a Vertical Marker showing the trigger point itself. Click *Next* or *Previous* buttons in the Artifact Duration dialog to check adjacent artifacts. Your goal is to choose the minimum time range that will bracket the entire ECG artifact on all channels and in all cases. Click *OK* to generate new channels with artifacts subtracted. The channel titles will include the channel number of the source data.





In the above example, channel 14 has been used to create the ECG marker channel 17 by marking troughs. These markers have been used to remove the artefacts from channel 1, with the cleaned data shown in blue (channel 20).

The full help and details of the script can be found within the script itself, downloadable from our website.

Back to contents

What are the memory view types and how do I use them?

Memory views are separate views created to hold analysed data from your data view. The analysis types available are: Waveform Average, Auto-Average, Amplitude histogram, Power Spectrum and Leak Subtraction (this last type was covered in our previous newsletter and is enabled along with clamping). Each of these will perform a particular type of analysis on your data, with their own settings to apply. Navigate to the Analysis menu > New memory view and selecting an analysis to begin.

Waveform Average

Averages waveform data channels across multiple frames. Select your desired *Channels* to average, the *Width* of the new average memory view, and optionally apply an *Offset*.

Auto Average

Averages waveform channels in the same manner as standard waveform average processing, but produces a memory view with multiple frames, each frame holding a separate average. Memory view frames can be generated from a pre-set pattern of groups of source frames, or they can be generated according to the source frame state. When using a pattern of source frames, the groups of frames used can overlap, be adjacent, or can have gaps of unused data between them.

Amplitude Histogram

Creates a histogram of the amount of time a waveform channel spends within an amplitude range. One channel is processed in a defined time range and for each data point the waveform level is used to calculate the corresponding histogram bin. If this bin number exists within the histogram, the bin data is incremented by the sample interval. This process is repeated for every data point from every frame, the resulting histogram shows the amount of time the waveform spends at different levels. For example, if your data waveforms were generally at one of two levels then the amplitude histogram of the data would show two peaks with most likely Gaussian distributions.

Power spectrum

Creates a memory view that holds the power spectrum of a section, or sections, of waveform data. If multiple sections are processed the result is an averaged power spectrum. The results of the analysis are scaled to RMS power (where power is amplitude squared), so they can be converted to energy by multiplying by the time over which the transform was done. Signal uses a Fast Fourier Transform (FFT) to convert the waveform data into a power spectrum.

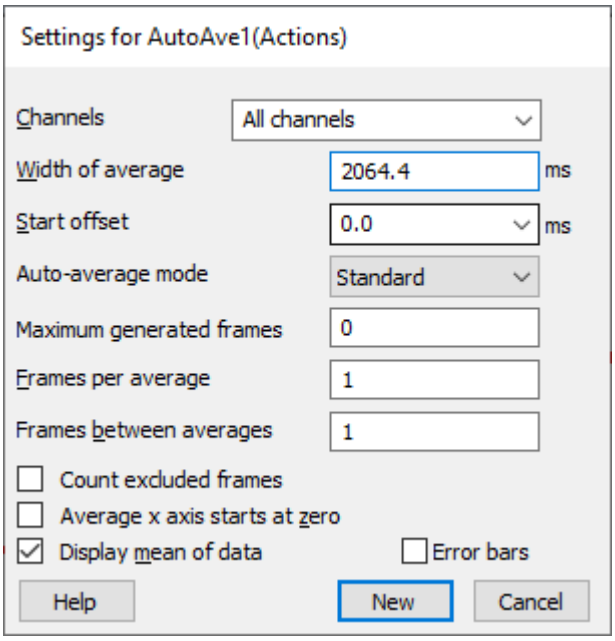
Leak subtraction

Creates a multi-frame memory view by carrying out a leak subtraction analysis on the source data. A full coverage of this analysis can be found in our previous newsletter.

Interval histogram

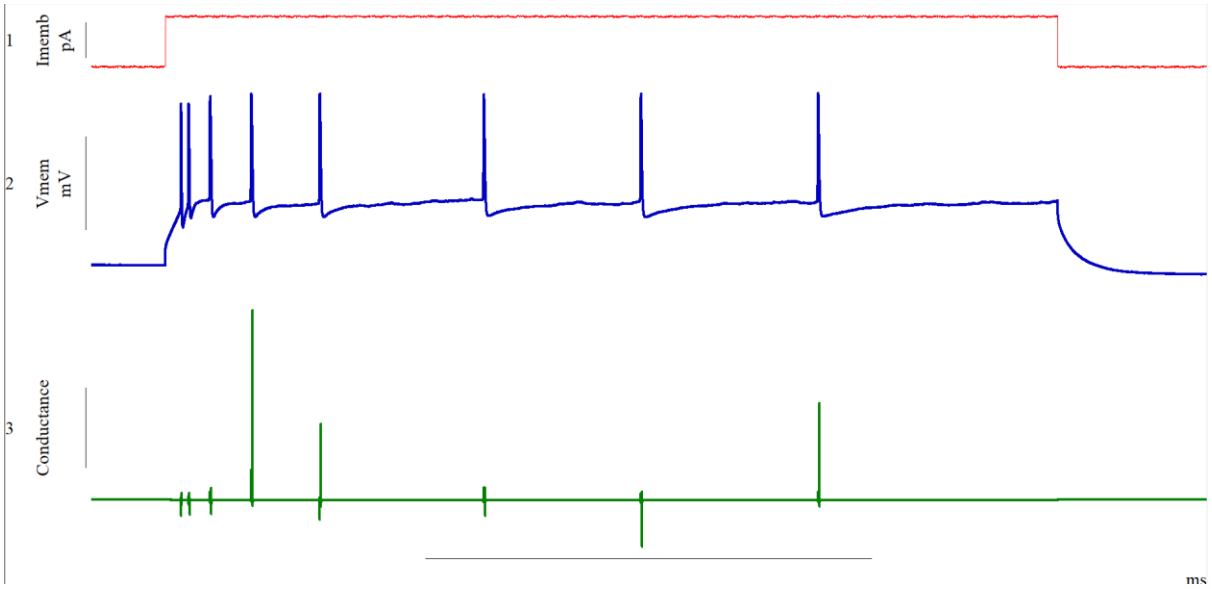
Creates a histogram showing the intervals between markers on one or more channels, either simple markers or real markers can be analysed. One channel is processed and for each marker item the interval since the previous item is calculated along with the corresponding histogram bin. If this bin number exists within the histogram the bin data is incremented. This process is repeated for every marker from every frame that is processed, the resulting histogram shows the relative frequency of occurrence of different intervals between markers. For example, if your data generally has marker intervals of 25 milliseconds histogram of the data would show a peak at about this point with most likely a Gaussian distribution. The analysis normally ignores the first marker in each frame as there is no previous marker, but you can optionally measure intervals that cross frame boundaries.

In addition to the intended uses of the memory views described above, the auto average may also perform another unique function. By entering the full frame width, 1 frame per average and 1 frame between averages into the settings, it is possible to create a complete duplicate of your data as a memory view.



Any virtual channels in your original data will also be converted to a waveform channel. This provides a unique way to save virtual channels to disk in the CFS file, instead of being stored as a calculation in the SGRX resource file, thereby creating additional waveform channels in your data after sampling.

For example, the Actions.cfs data in your example data folder (C:\Users\User\Documents\Signal7\data) contains two channels for current and voltage. It is possible to calculate the conductance in a virtual channel by dividing current by the voltage (ch(1)/ch(2)). By using the above method, this virtual channel is then converted to a waveform channel.



The full details and workings of all memory view types can be found in the software help (F1), Analysis menu > New Memory View.

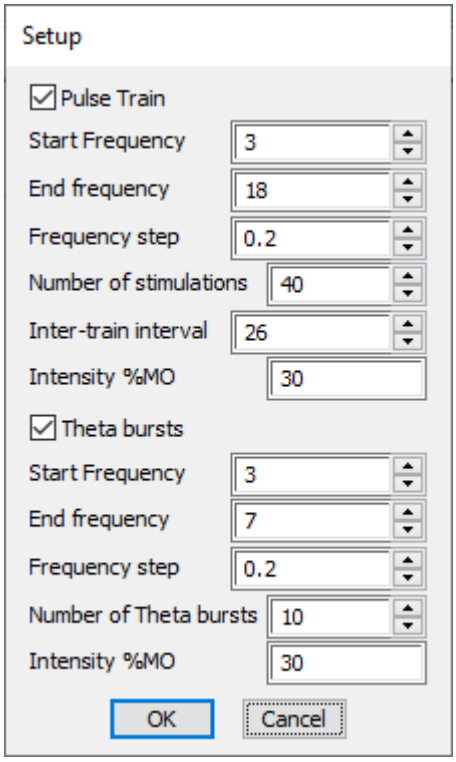
Back to contents

Scripts: Signal

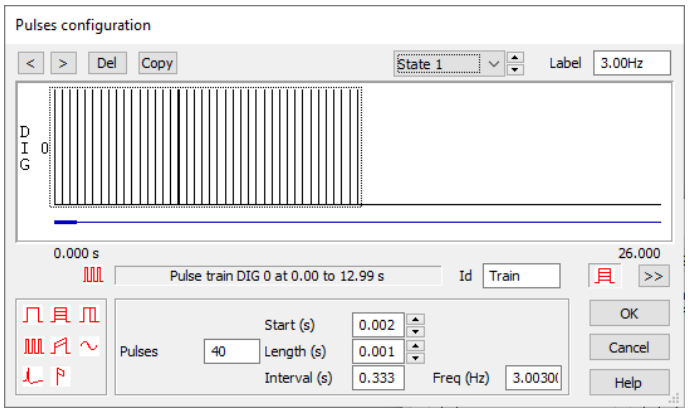
We have recently created a script for a researcher performing transcranial magnetic stimulation (TMS) studies. The researcher is using a Magstim Rapid and needed to create pulse trains and theta bursts in Signal of increasing frequency. Due to the large number of multiple frame states and pulse trains to set up, our script writers developed the attached script to automate the creation. Whilst this script is designed with the Magstim Rapid in mind, it is perfectly feasible to adjust the script for use with other TMS stimulators.

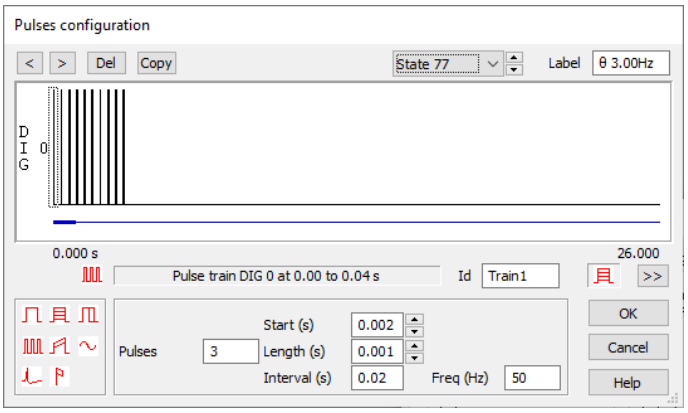
The script operates by first asking the user to select the type of pulse trains to create, by ticking the Pulse Train and Thera bursts options. The starting frequency, end frequency and frequency step size must be then defined for the script to then calculate the number of states required and begin creating the pulse trains of each frequency. The following options are available:

- Simple pulse trains of increasing frequency
    - Start frequency of pulse trains.
    - End frequency of pulse trains.
    - Frequency step size, used to calculate the required number of states.
    - Number of stimulations/pulses per train/state.
    - Inter-train interval, which is equivalent to the output length (the experiment therefore must operate on fixed interval sweep mode).
    - Intensity for Magstim Rapid as a % of maximum output.
- Theta bursts of increasing frequency
    - Start frequency of theta bursts.
    - End frequency of theta bursts.
    - Frequency step size, used to calculate the required number of states.
    - Number of theta bursts per train/state
    - Intensity for Magstim Rapid as a % of maximum output.



Upon clicking ok, the script will clear the existing sample configuration, enable multiple frame states, and begin creating the pulse trains based on your chosen parameters:





Each pulse output is labelled with the train's frequency size and is routed to digital output 0 on the 1401. The Magstim configuration is also set to trigger from digital output zero with each state set to the chosen stimulation power %, however you will need to double check the com port for communication is correct. We also recommend using the Test button to ensure Signal is communicating to the stimulator correctly, with no errors.

The user is then free to finalise the sampling configuration for their experiment, check the equipment, and ultimately begin the experiment much quicker than if they had needed to set up the pulse trains themselves.

Back to contents

Scripters Corner – Toolbars

Toolbars are a convenient way to structure your scripts. They provide a simple way for the user to access multiple functions in a script and operate by linking a button to a user-defined function. For example, we could supply a button for opening a data file, additional buttons for performing different types of analysis and a final button to Quit out of the script. A toolbar like this allows the user to pick and choose which functions to use for any given analysis.

To get started with toolbars use the ToolMake script in your Spike2 or Signal scripts folder (usually C:\Users\User\Documents\Spike10 or Signal7\Scripts). This script allows you to build toolbars by adding labelled buttons and linking them to functions in your script. Run ToolMake and follow these steps:

1. Click *Add button* and then *OK* (to add button 1). Enter the function name “Quit” and leave the *This button closes toolbar* box checked.

2. Click *Add button* and edit the button number to 3 and click *OK*. By not using button 2 we create a gap on the toolbar to provide a visual break between buttons. In the next dialog set the label to "Action1" and the function name to “DoAction1” and leave the checkbox unchecked as this button does not close the toolbar.

3. Click *Test* to check that the result is as intended. Click *Quit* to exit the test. You can edit buttons if you have made any errors.

4. Click *Write* and then set the checkboxes to determine what can be done while the toolbar is active. One would usually set *Can swap to other applications*, *Can move and resize* and *Can use View menu* unless they have reasons for using the other checkboxes. By allowing more permissions the user has more of a chance of performing an action that will cause the script to error, and you will inevitably need to write in more error-proofing to your script.

5. Click *Quit* to exit Toolmake and view your new script.

Whilst this script will help you quickly generate the code for a toolbar it is still worthwhile understanding how the script commands operate. The toolbar buttons are created with the `ToolbarSet()` function. The toolbar itself is displayed with the `Toolbar()` function. These commands are combined into a user-defined function to effectively create the toolbar when called, which in your example is `DoToolbar()`.

ToolbarSet(item%, label$ {,func ff%()})

This script command has two required arguments and one optional (the last one enclosed in { } brackets).

`Item%` sets the button number 1 to 40, or the idle function if set to 0. Idle functions are incredibly useful user-defined functions that are repeatedly called while the toolbar is active and the script is not performing any other action. We will cover more on these in a later issue.

`Label$` sets the button label but can also be expanded to link the button to a key and provide a tooltip that appears when the mouse hovers over the toolbar button. Adding ampersand before a character in the label name links the button to that key, e.g. `"&Quit"` links to key Q, “E&xit” links to the x key. Do note that keys are case sensitive. Alternatively, a hexadecimal key code can be suppled after the label, separated by |, e.g. `"label"|0x2e` links to the End key. A full list of hexadecimal codes is available in the help text for `ToolbarSet()`, but be careful not to use a key you will need to use regularly whilst the toolbar is active. To add a tooltip, supply the text after the code separated by another |. The key code may also be omitted, e.g. `"label"||"this button has no associated key"`.

`func ff% ()` is the name of the function with no arguments, e.g. `DoAction1%()` in your script, for example: `ToolbarSet(3, "Action&1||Do action 1", DoAction1%);` If the function is omitted then there is no function linked to the button. Omitting the argument will cause the `Toolbar()` function to return the button number and leave the toolbar; a typical use is to set a Quit command with no linked function.

Toolbar(text$, allow% {,help%|help$});

This function has two required arguments and one optional. The function displays the toolbar and causes the script to wait for the user to click a button, press a linked key, or perform a linked mouse action (more advanced than what we will cover today). As explained above, it returns the number of the button that was pressed to leave the toolbar, so is added to the return line of your user-defined function DoToolbar().

`Text$` this is used to display text in the left-hand message area of the toolbar, eg. `"To begin, press Open..."`

`Allow%` is a code that corresponds to the checkboxes you used to enable permissions when creating your script. The full list is available in the help text for `Toolbar()`. For example, to select the three checkboxes mentioned above you could write the code as `1+4+32` or simply the total `37`.

`Help%` is the number of a help item, mostly for CED internal use. This is used to set the help information that is presented when the user presses the F1 key and could be set 0 to accept the default help. Using the string `help$` instead allows you to set a string to select a help topic. The help topic string should be as displayed in the *Help Index* tab. For example, if your toolbar involved a lot of cursor placement you could put `"Cursors: Adding"` to select this topic.

```
var DataVH%;
DoToolbar();   'Try it out
Halt;

Func DoToolbar()
ToolbarClear();
ToolbarSet(1, "&Quit");   'Link to function
ToolbarSet(2, "Open", Open%);   'Link to function
ToolbarSet(3, "DoAnalysis", Analysis%);   'Link to function
ToolbarEnable(3, 0);       'Disable button 3
return Toolbar("First Open a data file to enable Analsys", 1023);
end;

Func Open%()   'Button 2 routine
DataVH%:=FileOpen("", 0, 1);
If DataVH% < 0 then
    return 1;       'return early
endif;
ToolbarEnable(3, 1);       'Enable button 3
return 1;  'This leaves toolbar active
end;

Func Analysis%()   'Button 3 routine
'Your code in here...
return 1;  'This leaves toolbar active
end;
```

In addition to these functions, you can also enable and disable toolbar buttons with `ToolbarEnable()` in response to function calls. In my example above, my `DoToolbar()` function has 3 buttons Quit (1), Open a data file (2), Analysis (3), and an extra line with `ToolbarEnable(3, 0)` to disable the Analysis button. Within my linked Open function, I then include the line `ToolbarEnable(3, 1)` to enable the analysis button only once the user has opened a data file. This is a measure of error-proofing to ensure the user cannot perform the Analysis function on the wrong data file. Further error-proofing has been added for my `FileOpen("", 0, 1);` line. This command opens the file open dialog and returns the view handle of the opened file, but if the user clicks cancel the command returns a negative error

code and the script continues. Therefore, if variable `DataVH% < 0`, I return early from the function to prevent the user from going any further.

Back to contents

Recent Questions – *How do I decontaminate my equipment from COVID-19?*

We suggest you disconnect all cables from the CED1401 and use an alcohol-based sanitiser, sprayed very lightly on to the 1401 black outer casing. The liquid in the deposited spray will evaporate and not need wiping off. This should be followed by a light application of the spray on to a cloth, which then could then be used on the front and rear panel connectors, followed by a careful application on the power supply and cables. Do not oversaturate the cloth with liquid, nor should the equipment be submerged.

Back to contents

Contact Us

If you have any comments about the newsletter format and content, please get in touch: Marjorie@ced.co.uk.

To adjust your subscription preferences, please visit our website: www.ced.co.uk/upgrades/subscribeenews.

Back to contents

Contact us:

In the UK:
Technical Centre, 139 Cambridge Road,
Milton, Cambridge, CB24 6AZ, UK
Telephone: (01223) 420186
Fax: (01223) 420488

Email: info@ced.co.uk
International Tel: [44] 1223 420186
International Fax: [44] 1223 420488
USA and Canada Toll Free: 1 800 345 7794
Website: www.ced.co.uk