VLSEE: Very Large Scale Electrophysiology-Experimental

Contents

1 Introduction

VLSEE is a collection of scripts used in and with a large scale spike sorting process. A section is dedicated to each of the scripts detailing behavior and usage. [Deprecated] proceeds the titles of scripts that are no longer actively maintained or improved.

1.1 Note on `custom/`

Scripts in the `custom/` directory, such as `plot_penultimate`, are modified versions of original core subroutines with either performance enhancements or changes needed to better run scripts in VLSEE.

2 `simple_snr`

`simple_snr` is a function that calculates the mean signal-to-noise ratio (SNR) and simply returns an array of units whose mean SNR falls below a given threshold.

2.1 Usage

Example:

```
badunits = simple_snr(dounits, bestchannel, wavedir, TL)
```

Parameters	Meaning
`dounits`	Vector of indices of units to do
`bestchannel`	Vector of best channel for each unit
`wavedir`	Path to waveforms files
`TL`	Low SNR threshold (not in dB)

3 `get_sane`

`get_sane` is function that returns an array of indices of bad units based on some simple criteria. Note that bad units are considered to be those that fall *outside* the valid range of these parameters. It should function as a basic sanity check before any sorting process to prevent garbage units from causing a "garbage–in–garbage–out" effect.

Value	Minimum	Maximum
Peak-to-peak Voltage (V_{pp})	60μV	350μV
Absolute Voltage	-350μV	350μV
Wave Count	200[1]	∞
Coefficient of Variation (Magnitude)	0	0.25

3.0.1 Coefficient of Variation

The calculation of this parameter is motivated by the observation that most units that are considered to be artifacts are very noisy around the peaks. Here, the mean of the lowest point of the unit is considered to be the peak and the coefficient of variation is defined as:

$$c_v = \sigma/\mu \tag{1}$$

were σ is the standard deviation of the minimum of the waveforms and μ is the minimum of the mean waveform. The absolute value of the coefficient of variation is used in the function to simplify the process.

3.1 Usage

Example:

```
badunits = get_sane(do_units, spiketimes, bestchannel, wavedir, sampling_rate, ispen)
```

Paramters	Meaning
`do_units`	The indices of units to check
`spiketimes`	The cell array of spike times
`wavedir`	The director where the waveforms are store
`sampling_rate`	The sampling frequency[2]
`ispen` (1=True, 0=False)	Whether this function is being run just after `get_penultimate_units`

4 `pca_merge`

`pca_merge` is a script that attempts to use MATLAB's Principal Component Analysis (`pca`) tool to guide an algorithm in merging units. Two units are first compared by concatenating their waveform data and taking the z-scores of this data (to compensate for the fact that `pca` is not a scale-invariant process). Note that each column of the concatenated waveforms represents values of a given variable. The number of variables is determined by the number of sample points per unit. `pca` is then used to transform the data such that the first few components or variables contain most of the information of the changes in the data. It is expected then, for two distinguishable units that plotting the first two components of the transformed data will yield two clusters in 2-D space. Currently `pca_merge` considers the first three dimensions of the transformed data and takes the distance between the centers of the two clusters as the basis for merge decisions.

[1]This minimum is 0 if this function is run at the penultimate step

[2]currently unused

Additionally, `pca_merge` performs the additional basic sanity check of only comparing units that share the same best channel.

4.0.1 Note on Z-score

Given a value of a variable x, the z-score of this value is defined as:

$$z = \frac{x - \mu}{\sigma} \tag{2}$$

where μ is the mean of x and σ is the standard deviation of x.

4.1 Usage

Example: `samemeans = pca_merge(dounits,bestchannel,wavedir)`

Parameter	Meaning
dounits	Vector of indices of units to do
bestchannel	Vector of of best channels of units
wavedir	Path to waveforms

5 [Deprecated] `density_distance_area_merge.m`

`density_disance_area_merge.m` was an experimental script that attempted to establish a criteria from the merging of two units based on the similarity of of the distribution of their interspike intervals (ISI)s. Unfortunately, while this idea works well in theory, units that are "incomplete" or "immature," meaning that they do not have most of the waveforms supposed to be associated with them actually included, tend to have ISIs that deviate significantly from the ideal case. Or equivalently, that a unit is incomplete often has a significant impact on its ISI distribution, meaning that this idea works well only with units that have enough spike times such that the distribution is well-defined and will not change significantly with the addition of more spikes from a unit it should be merged with. To compare actual distributions, this script separated the ISI data into bins of predetermined size and performed a discrete summation of absolute value of the difference of the distributions of two units and compared the result to an empirically defined parameter.

6 [Deprecated] `quasi_pdf_merge`

`quasi_pdf_merge` was an experimental script that attempted to also use the distribution of the ISIs of two units as a merge criteria, but in addition to the problems encountered with `density_distance_area_merge`, `quasi_pdf_merge` encounters additional problems. Namely, as this script relies on first fitting a distribution to the ISIs of a given unit, the validity of the comparison of the distribution is dependent not only on the number of spike times of the unit, but the "goodness of fit." In practice, the "goodness of fit" was usually not good enough to justify use of this function. Additionally, even with a "good fit," it remains troublesome to identify a good metric for comparing different probability density functions. The approach of simply looking at the different in the parameters of some probability density function was used. Note that if two discrete probability distributions are to be compared, the approach of `density_distance_area_merge` is probably better.