Testing spike detection and sorting algorithms using synthesized noisy spike trains

Leslie Smith, Dept of Computing Science and Maths, University of Stirling, Stirling FK9 4LA, Scotland
email lss@cs.stir.ac.uk

UNIVERSITY OF STIRLING

Abstract: *Spike detection and spike sorting techniques are often difficult to assess because of the lack of ground truth data (i.e. spike timings for each neuron). This is particularly important for* in vitro *recordings where the signal to noise ratio is poor (as is the case for multi-electrode arrays at the bottom of a cell culture dish). We present an analysis of the transmission of intracellular signals from neurons to an extracellular electrode, and a set of MATLAB functions based on this analysis. These produce realistic signals from neighboring neurons as well as interference from more distant neurons, and Gaussian noise. They thus generate realistic but controllable synthetic signals (for which the ground truth is known) for assessing spike detection and spike sorting techniques. They can also be used to generate realistic (non Gaussian) background noise. We use signals generated in this way to compare two automated spike sorting techniques. The software is available freely on the web at* http://www.cs.stir.ac.uk/~lss/noisyspikes.

Aims:

1. To develop a model for the transfer of charge from spiking neurons to electrodes
2. To develop a tool based on this model, to provide realistic data with realistic interference for which the ground truth is known to be used to assess spike detection and sorting techniques
3. To use this tool to assess some spike sorting techniques.

The model

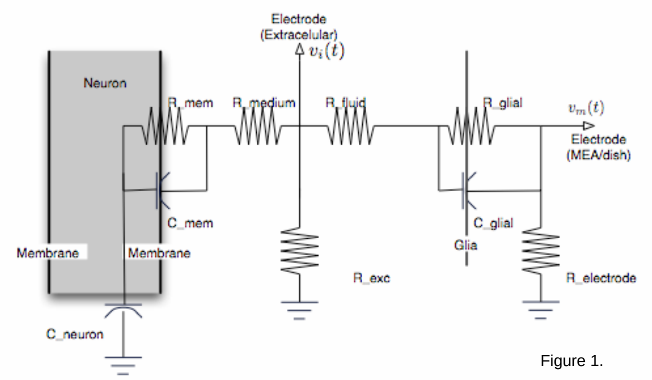


Figure 1.

Figure 1 shows equivalent circuit description for transfer of charge from a point on a neuron to an electrode for an extracellular and a dish based electrode. The extracellular electrode is assumed to be near the neuron, but the dish electrode is may have a layer of glia between it and the neural culture. There are a number of simplifications in this circuit: distributed resistances and capacitances have been lumped together, for example.
This circuit is used to develop an expression for the voltage at the extracellular electrode resulting from the intracellular spike voltage in some neighbourhood of a small area of the membrane. This expression is in terms of the intracellular voltage and its time derivative, as well as the lumped components in the circuit above.

What is the effect of the integrating this over the extent of the neuron?
1. Because the time taken for spike movement from the spike initiation point on the soma through the axon is comparable to or larger than the spike duration, this integration will have a major effect on the shape of the voltage recorded at the electrode.
2. Because the different parts of the spiking neural surface will be at differing distances, the shape of the voltage at the electrode corresponding to a spike will depend on the precise geometry of the electrode and neuron.

We have lumped these together. We use

$$v_k^{\mathrm{spike}}(N_j, n\delta t) = \sum_{i=1\ldots k} V'_{\mathrm{spike}}(t - i\delta t) b_2(i) + \sum_{i=1\ldots k} V_{\mathrm{spike}}(t - i\delta t) b_1(i)$$

where $v_k^{\mathrm{spike}}(N_j, n\delta t)$ is the voltage detected at the electrode from neuron *Nj* at time $n\delta t$, $V_{\mathrm{spike}}(t)$ is the intracellular voltage inside Neuron *j* (assumed constant over its extent)

The *b*'s are constants reflecting (1) the strength of the intracellular signal (at the different parts of the spiking surface of the neuron) and (2) the response function (coupling) between those parts of the neuron and the electrode. For an electrode on an MEA, there is likely to be an additional complexity: these are often (partially) covered in insulating glial cells. This adds on another term, this time in the second derivative of *V(t).*
A much more detailed derivation is included in the paper on the website (available from the author)[1].

The software

The software (available from http://www.cs.stir.ac.uk/~lss/noisyspikes) is a set of MATLAB m-files. It needs the statistics and signal processing toolboxes. As well as the m-files, the website also contains a reasonably comprehensive user manual, and an extended paper[1].

The software allows

- user selectable (and user definable) intracellular spike shapes[2]
- a user-selectable number of target neurons (including 0, allowing pure interference generation)
- a user selectable number of interference neurons, whose spike times are correlated with one of the target neurons
- a user-selectable number of uncorrelated interference neurons

Neural spikes may have Gaussian or Poisson distributions. Spike times may be re-used so that the same experiment may be run with different amounts of interference.
Virtually all the parameters in the simulation can be set (and are detailed in the user manual). Unfortunately, there's no GUI. So using the software means typing command lines like

```
[signal info rinfo]  = generatenoisysamples('N_Jitter', 5, 'N_Uncorr',
50, …
'ShowSNR', 1,  'Duration', 30, 'SameTargetSizes', 1, 'ReuseTargets',
old_info) ;
```

which creates 30 seconds of simulated data (into the 1 dimensional array `signal`), keeping lots of information about this run in `info`, with two target neurons (default), 5 correlated (jittered) neurons and 50 uncorrelated neurons, reusing target times from a structure `old_info`.

Generating 30 seconds of data as above (at 24 Ksamples/second) takes 94 seconds on a 2 GHz Macintosh G5, with 4Gbytes, 82 seconds on a 2.8 GHz Windows XP PC with 1GB and about 5 minutes on an 867 MHz G4 laptop (with 640Mbytes).

Assessing Wave_clus and KlustaKwik

Wave_clus[3] and KlustaKwik[4] are spike sorting software packages. Wave_clus detects spikes using a threshold based technique, and can generate either three PCA components or 10 wavelet components from the 64 samples surrounding each spike it detects (64 samples at 24 Ksamples/second = 2.67 ms).

Test 1: Dissimilar spike shapes.

Spikes with very different spike shapes were generated (see figure 2). These were then input to four different spike sorting systems.

The preprocessed input was either three PCA components, or 10 Wavelet coefficients,
and
the spike sorting technique was either KlustaKwik (which uses CEM[5]), or Wave_clus (which uses SPC[3]).

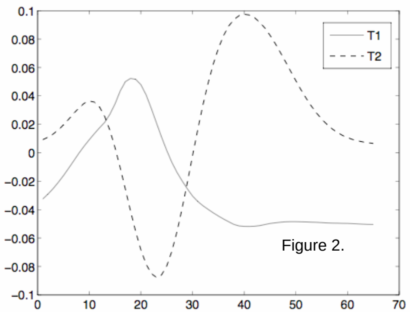


Figure 2.

Zero noise results.

There are 482 T1 spikes, and 608 T2 spikes. SPC applied to wavelets fails to classify 85 of them, but outperforms the other techniques. The other techniques all discover more than the (correct) two clusters.

Wavelets

Target	SPC				KlustaKwik				
Class	0	1	2	FoM	0	1	2	3	FoM
T1	50	0	432	0.92	482	0	0	0	0.74
T2	35	573	0		51	99	360	98	

PCA

Target	SPC						KlustaKwik					
Class	0	1	2	3	4	FoM	0	1	2	3	4	FoM
T1	66	0	253	86	77	0.68	212	0	220	0	50	0.54
T2	106	502	0	0	0		161	373	0	74	0	

In this case, preprocessing using Wavelets followed by clustering using SPC is best.

Figure of Merit

We introduce a *figure of merit* in an attempt to summarise the effectiveness of spike sorting where the ground truth is known:
The figure of merit is calculated as below:

$$M = \frac{1}{n}\sum_{j=1}^{n}((\max_i(T_j(i) - \sum_{k=1 k\neq j}^{n} T_k(i))/N_j)$$

where *n* is the number of target neurons, and *Tj(i)* is the number of spikes from target neuron *j* found in cluster *i*. It measures the degree to which different clusters detected follow the spikes from different target neurons: its maximum value is 1. It can cope with the number of clusters found being different from the actual number of classes.

Correlated and Uncorrelated Noise results:

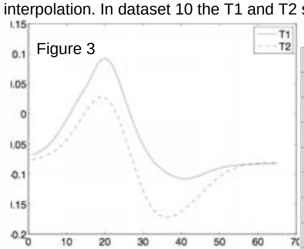
Table rows show the figure of merit, the number misclassified, and the number not classified. P is preprocessing type, FoM is figure of merit, MC is misclassified (i.e. target 1 spikes classed as target 2 or vice versa, UC is unclassified spikes (either in the catch-all group, or in some other cluster apart from the one selected in the figure of merit), M is spikes missed, and I is spikes inserted.
Using KlustaKwik on wavelet data has some wrinkles: if the maximum number of clusters permitted is not set, far too many are produced. Setting a limit (of 5) sometimes results in only one cluster being produced (line 2). Best results are obtained using PCA: SPC and KK applied to PCA perform equally.

			SPC			KK				
Data	SNR	P	FoM	MC	UC	FoM	MC	UC	M	I
UC	4.6	PCA	0.92	0	80	0.93	0	69	5	7
UC	4.6	WAV	0.67	0	358	0.11	477	0	5	7
UC	3.2	PCA	0.90	5	85	0.91	4	75	13	1
UC	3.2	WAV	0.74	0	250	0.86	0	127	13	1
UC	2.3	PCA	0.65	1	125	0.67	1	107	232	1
UC	2.3	WAV	0.67	1	107	0.62	1	149	232	1
UC	1.6	PCA	0.21	51	8	0.21	51	10	737	9
UC	1.6	WAV	0.21	46	21	0.24	1	69	737	9
C,UC	4.6	PCA	0.91	2	87	0.91	3	79	10	46
C,UC	4.6	WAV	0.70	1	294	0.68	0	353	10	46
C,UC	3.2	PCA	0.83	41	56	0.86	8	94	39	38
C,UC	3.2	WAV	0.86	1	111	0.83	0	138	39	38
C,UC	2.3	PCA	0.52	91	20	0.56	5	163	284	27
C,UC	2.3	WAV	0.58	5	136	0.43	3	317	284	27
C,UC	1.6	PCA	0.19	59	7	0.19	59	3	741	35
C,UC	1.6	WAV	0.19	51	16	0.21	12	83	741	35

Test2: Using similar spike shapes.

The spike shapes used in dataset 1 are shown in figure 3. For the other datasets, the T1 spike shape was made more and more like the T2 spike shape in nine steps using simple linear interpolation. In dataset 10 the T1 and T2 spike shapes are identical.

Figure 3



Even for dataset 1, PCA preprocessed data failed. The table thus shows only wavelet preprocessing. KK is better at separating the two clusters particularly when they are very similar.

	SPC			KK				
Dataset	FoM	MC	UC	FoM	MC	UC	M	I
1	0.58	1	401	0.79	3	174	61	72
2	0.64	1	354	0.62	0	357	49	53
3	0.55	16	410	0.70	0	275	64	62
4	0.47	14	500	0.72	1	251	57	55
5	0.20	43	822	0.68	0	297	55	47
6	0.05	454	88	0.69	1	282	63	34
7	0.05	444	109	0.71	0	259	58	28
8	0.036	445	122	0.69	0	272	70	45
9	0.06	445	93	0.69	0	284	64	48
10	0.08	447	63	0.056	414	154	66	34

Conclusions

The synthetic data enables informed experimentation with spike sorting techniques. For very similar spike recording shapes, it is clear that wavelet preprocessing followed by KlustaKwik is the best technique. For other data, no clear winner emerges from these experiments. Perhaps this is not unexpected: the best method of separating the different clusters depends on the actual shapes of the clusters. The synthetic data emphasizes the importance of trying out different techniques, and different parameters to these techniques.

References: [1] Smith L.S., Mtetwa N., http://www.cs.stir.ac.uk/~lss/noisyspikes/webV1_1/spikegen_paper_revised1.pdf (submitted to J. Neur. Methods), [2] Uses Naundorf et al, Nature, 440 (7087), 1060-3, 2006, or Touretzky et al, HHsim, http://www.cs.smu.edu/~dst/Hhsim, [3] Quiroga R., Neural Computation 16, 1661-87, 2004, [4] Harris, K. KlustaKwik, http://klustakwik.sourceforge.net, [5] Celeux G., Govaert G, Comp. Stats and Data Analysis 14 (3), 315-22, 1992.

*Fifteenth Annual Computational Neuroscience Meeting CNS*2006, July 16-30, Edinburgh*