Analyse des séquences de potentiels d'action tutorial

Christophe Pouzat

Data used

We are going to use spike trains obtained from the antennal lobe–first olfactory relay–of locust, *Schistocerca americana*. These spike trains can be found on the zenodo-locust-datasets-analysis GitHub repository. You can also find there a complete description of the sorting procedure used to go from the raw data, that are available on zenodo, to the spike trains. We will mostly use the spike trains from experiment `locust20010214` that can be found at the following address: https://github.com/christophe-pouzat/zenodo-locust-datasets-analysis/tree/master/Locust_Analysis_with_R/locust20010214/locust20010214_spike_trains.

Getting a spike train

We will start by downloading the spike train from unit 1 from `group` `Spontaneous_1`. This is done by typing in the shell:

```
wget https://raw.githubusercontent.com/christophe-pouzat/\
zenodo-locust-datasets-analysis/master/Locust_Analysis_with_R/\
locust20010214/locust20010214_spike_trains/\
locust20010214_Spontaneous_1_tetB_u1.txt
```

This "spike train" contains in fact the result of 30 consecutive continuous acquisitions, each 29 s long with a 1 s gap in between, as is made clear in the detailed sorting description of this data set.

Making the "observed counting process" plot

We first get a gnuplot script, `aspa_ocp.gp`, from the github repository:

```
wget https://raw.githubusercontent.com/christophe-pouzat/\
aspa/master/gnuplot/aspa_ocp.gp
```

We then get a graph showing the observed counting process associated with the train by typing in the shell:

```
cat aspa_ocp.gp locust20010214_Spontaneous_1_tetB_u1.txt | \
gnuplot -persist
```

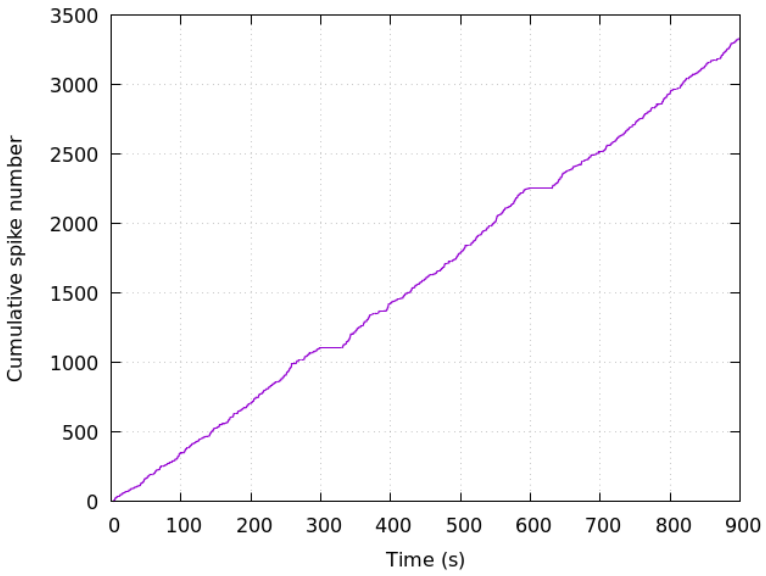

Figure 1: The observed counting process associted with `locust20010214_Spontaneous_1_tetB_u1.txt` spike train

Notice the two "long" horizontal sections, thess are due to the presence of recording noise during trials 11 and 21 (these trials were skipped during spike sorting).

`C` code

The "heavy duty" work will be done by `C` codes. In addition to a `C` compiler like gcc, you will need the GSL (Gnu Scientific Library) to compile the codes.

First test

To test that everything is properly set, we start by downloading the `Makefile`

```
wget https://raw.githubusercontent.com/christophe-pouzat/\
aspa/master/code/Makefile
```

as well as the header file `aspa.h` and the two `C` source files, `aspa_single.c` and `aspa_single_test.c`:

```
wget https://raw.githubusercontent.com/christophe-pouzat/\
aspa/master/code/aspa.h
wget https://raw.githubusercontent.com/christophe-pouzat/\
aspa/master/code/aspa_single.c
wget https://raw.githubusercontent.com/christophe-pouzat/\
aspa/master/code/aspa_single_test.c
```

We can then compile the test file (`aspa_single_test.c`) with:

```
make aspa_single_test
```

And we run it on the spike train we downloaded previously with:

```
cat locust20010214_Spontaneous_1_tetB_u1.txt | \
aspa_single_test
```

We should see the number of spikes (3331), the time of the first spike (0.290975 s) and the time of the last one (898.15 s) printed.