icmcstat—STATISTICAL ANALYSIS OF MODE CHANGES

IVO SIEKMANN

The C tool icmcstat was developed for statistical analysis of modal gating in ion channels as described in Siekmann et al. (2013). icmcstat reads ion channel currents from a text file or stdin and splits the trace in segments based upon abrupt changes of channel activity. Ion channel activity is quantified by the average open probability within a segment.
In Section 1 we give installation instructions for Windows, Mac OS X and Linux. In Section 2 we explain how icmcstat is called from the command line and the format of the output files.

1. INSTALLING ICMCSTAT

It should be easy to compile icmcstat from source code with the supplied makefile. icmcstat depends on the GNU scientific library (gsl). First we give intructions for installing gsl on Windows, Mac and Linux (see sections 1.1.1-1.1.3). Alternatively, it should be easy to compile the library from source code (section 1.1.4). When the library has been successfully installed, icmcstat can be compiled and run via make run (section 1.2).

1.1. **Installing gsl...**

1.1.1. *... under Windows.*

(1) Download Cygwin that is freely available for download from http://www.cygwin.com.
(2) Install Cygwin and make sure that the following packages (that are usually de-selected by default) are installed: gsl, gsl-devel, make and gcc.
(3) Run a Cygwin terminal by clicking on the desktop icon.

1.1.2. *... under Mac OS X.* Install the gsl library via fink install gsl-dev if you use fink or port install gsl-dev under MacPorts. The package names can be slightly different, so it may be necessary to look for the exact name using fink list gsl or port search gsl.

Date: November 26th, 2013; please e-mail ivo.siekmann@unimelb.edu.au for help if there are problems with installing icmcstat.

1.1.3. *... under Linux.* How the gsl library can be installed depends on your Linux distribution. For Ubuntu the command would be

```
apt-get install gsl-dev
```

for Fedora, please use

```
yum install gsl-dev
```

Most other linux distributions follow one of these patterns. Note that the package name gsl-dev may be slightly different, so it might be necessary to look for the exact name using `apt-cache search gsl` or `yum search gsl`.

1.1.4. *... from source code.* Installation follows the standard pattern for installing from source code. Note that a C compiler and the make tool must be available for compiling gsl. Please refer to installation instructions specific to your system—these are provided with the gsl source code.

(1) Please download the source code from http://www.gnu.org/software/gsl/
(2) Unpack the archive of the source code.
(3) Change to the directory where the gsl source code is located and type

    ```
    ./configure
    ```

(4) After the configure script has terminated, run

    ```
    make
    ```

    for compiling the library.
(5) When the library has been compiled successfully, it needs to be installed:

    ```
    sudo make install
    ```

    You will be asked for your administrator password because this step will copy components of the library to locations that usually require root acces. This step is *essential* (as all previous steps!), if the library is not installed, icmcstat will not compile.

1.2. **Running and testing** `icmcstat`**.** These instructions assume that you have successfully installed gsl. You can test this by running the command

```
gsl-config --cflags
```

in a terminal (this should print the path to the C header files of the gsl library so that the C compiler can find it, the output is '-I/sw/include' on my machine). If this does not work, please try installing gsl before continuing.

(1) icmcstat is compiled by opening a terminal, changing to the directory containing the icmcstat source code and typing

```
make all
```

(2) icmcstat can be run on a test data set provided with the source code by

```
make run
```

2. Synopsis

2.1. **Usage.** The tool is called from the command line

```
./icmcstat datafile iterations seed [outputprefix]
```

datafile is the name of a text file or '-' for stdin that contains a single column of ion channel currents. The algorithm is run for a certain number of iterations, the random number generator is initialised with seed. The argument outputprefix is optional and allows to add a prefix to the output files generated.

2.2. **Output.** In each iteration the algorithm attempts to generate a random sample of a probability distribution for the number of changepoints, their locations and the open probabilities of each segment, see Siekmann et al. (2013) for details. Samples accepted by the algorithm are written to text files with the following format of tab-delimited columns:

- <prefix>_nK.dat: Contains the number of changepoints n for a given iteration, the number of open probabilities is the number of segments, therefore it is always $n+1$.

  Iteration # of probabilities # of changepoints

  There is always one more probability than changepoints.

- <prefix>_ipXXX.dat:
  XXX gives the number n of changepoints, it runs from 001 to the maximum number n of changepoints.

  Iteration j_1 j_2 $\dots$ j_n

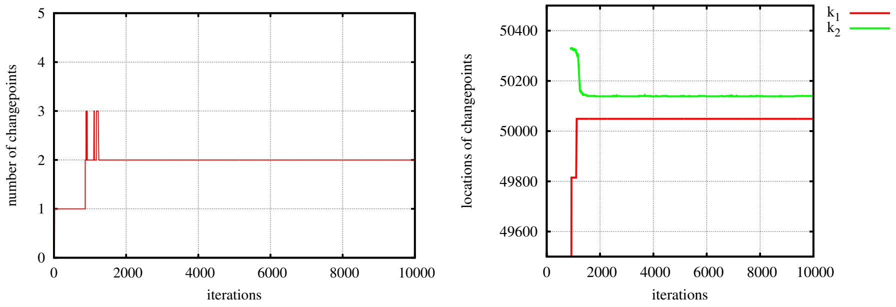
- <prefix>_pXXX.dat:
  XXX runs from 000 to the maximum number of changepoints.

  Iteration p_0 p_1 $\dots$ p_n

  Here, p_i is the open probability of the segment *after* the ith changepoint and p_0 is the open probability of the segment *before* the first changepoint.

- `<prefix>_likelihood.dat`: Contains values for log-prior p, log-likelihood L and log-posterior π for the current iteration in the following order (10 columns):
  Iteration, p_{old}, p_{new}, $p_{\text{new}} - p_{\text{old}}$, L_{old}, L_{new} , $L_{\text{new}} - L_{\text{old}}$, π_{old}, π_{new} , $\pi_{\text{new}} - \pi_{\text{old}}$

2.3. **Results.** For the test data set provided the MCMC algorithm converges to a model with two change points (Figures 1, 2).



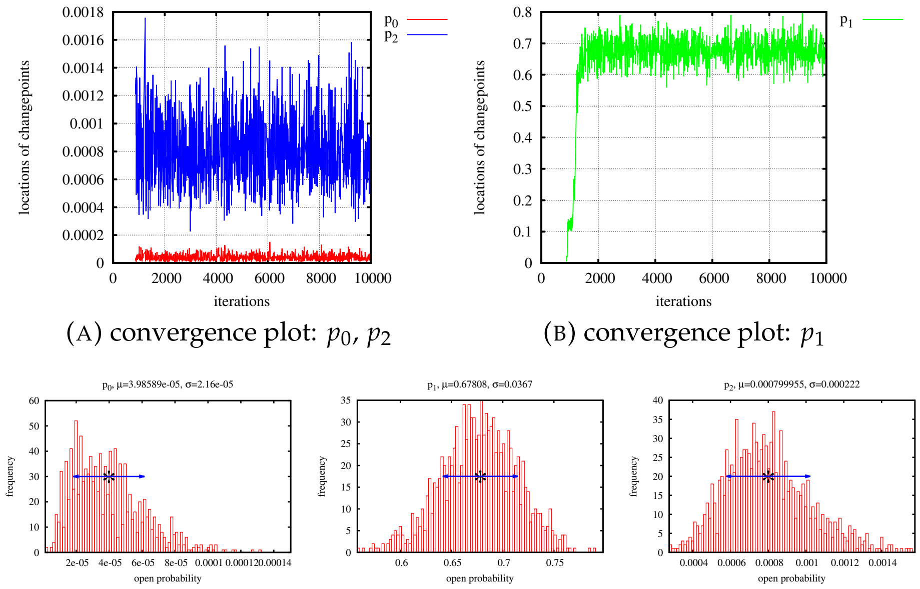
(A) number of changepoints

(B) locations of changepoints

FIGURE 1. Results for the test data set provided with the `icmcstat` code. (A): convergence plot of the number of changepoints, (B): convergence plot of their locations.

REFERENCES

Siekmann, I., Sneyd, J., and Crampin, E. J. (2013). Statistical analysis of modal gating in ion channels. submitted.



(A) convergence plot: p_0, p_2

(B) convergence plot: p_1

(C) p_0

(D) p_1

(E) p_2

FIGURE 2. Results for the test data set provided with the `icmcstat` code. (A), (B): convergence plots for open probabilities p_0, p_2 and p_1, (C)-(E): histograms for these open probabilities (burn-in: 3,000 iterations).