Tutorial for the HBBSD algorithm

Michal Natora[*]

September 15, 2010

1 Getting started

In this section a step-by-step instruction is given, how to run the HBBSD algorithm. More details about individual functions can be found in the subsequent sections 2 and 3.

1. The provided version of HBBSD was developed in MATLAB, version 7.6.0.324 (2008a), Windows edition. The compatibility with different versions of MATLAB has not been tested. The Signal Processing Toolbox and the Statistics Toolbox are required.

2. Download the HBBSD algorithm from
`http://user.cs.tu-berlin.de/~natora/`

3. Extract the .zip or the .rar to a directory of your choice, e.g. C:\

4. Open the `MainFile2CallHBBSD.m` file with the editor. In line 22 enter the path of the directory where you extracted HBBSD.rar and add '\data \', e.g. C:\HBBSD\data\
Run `MainFile2CallHBBSD.m`

5. Open the `plotROC.m` file with the editor. In line 6 enter the path of the directory where you extracted HBBSD.rar and add '\data \', e.g. C:\HBBSD\data\
Run `plotROC.m`

If everything went as it should, a graphics like Fig. 1 should have appeared.

[*]M. Natora is with the Institute for Software Engineering and Theoretical Computer Science, Berlin Institute of Technology, 10623 Berlin, Germany. Email: natora@cs.tu-berlin.de

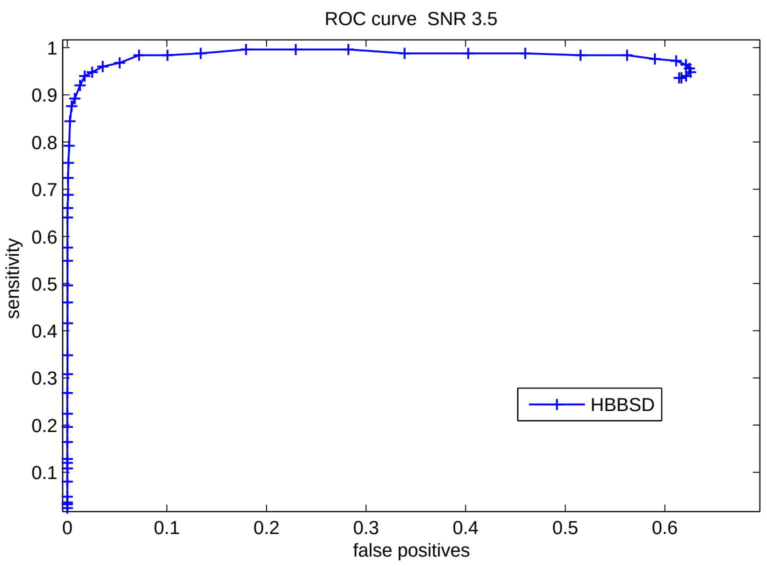

Figure 1:

2 More details

In this section the working principles behind the steps of the instruction given in Sec. 1 are explained. The scope is to explain how to use the algorithm on datasets provided by the user, and not what the theoretic background of the HBBSD algorithm is. For latter, please study [1].

In the '...\data\ ' directory the file `simData_template_1_snr_3.5_run8XX.mat` contains a simulated single channel electrode recording. Assuming a sampling frequency of 10kHz the length of the recording is 6s, and it contains waveforms from two distinctive neurons. The ground truth information is stored in the `simData_template_1_snr_3.5_run8c.mat` file. The script `MainFile2CallHBBSD.m` loads this recording and passes it to the

`HBBSDi.m`

function (line 31). This function is the main function of the whole algorithm, as it contains all the essential procedures like the super-exponential algorithm (SEA), noise peak and variance estimation, sparse deflation and the calculation of the MVDR filters. The required input parameters for this function are

1. the recording, which must be single channel data
2. the spike length (in samples) of the spikes encountered in that recording
3. the desired length (in samples) of the MVDR filters (usually this value is equal or greater than the spike length)

4. the minimum firing frequency of the neurons encountered in that recording (a rough estimate is sufficient, for example 2Hz)

5. the maximum firing frequency of the neurons encountered in that recording (e.g. 50Hz)

6. the sampling frequency of the recording

7. the maximum number of filters the algorithm should calculate (a reasonable value for single channel recordings is 3 to 5)

8. the operating mode of the algorithm (in case of 'skew', which is the default mode, SEA uses third order cross cumulants for filter estimation, whereas in case of 'kurt' fourth order cross cumulants are used)

Usually the operating mode 'skew' is not only computationally more efficient, but also provides better results in terms of detection performance. However, when SEA does not converge in one mode, the algorithm automatically switches to the other one. The `HBBSDi.m` function returns the following objects

1. the MVDR filters

2. the filters estimated by SEA

3. an object containing several quantities that were calculated (this object is mostly for debugging/checking purposes)

Once the MVDR filters are calculated the actual spike detection procedure can be performed. This is implemented in the `HBBSDfiltering.m` function (line 49). This function performs a convolution between the MVDR filter and the recording. The spikes times are determined by performing a maximum search on the filter output data which are above a certain threshold.

3 Notes

In this section some remarks are given which might be of interest, in particular if it is intended to compare the performance of HBBSD with various other spike detection algorithms.

- The estimation of the MVDR filters should not be done on the entire recording, as this part of the algorithm is quite computationally demanding. I.e. given a recording of 10min duration, `HBBSDi.m` should be applied only to the first 5-15s of data. Assuming approximatively stationary data, the learned filters can then be used for spike detection in the entire 10min long recording.

- As with any other estimation method, the length of the data on which the filters should be learned, depends on the data itself. When many neurons are present, the signal-to-noise ratio is low and/or the firing frequencies are very low, a longer data segment for `HBBSDi.m` should be used. On the other hand, given few neurons, a good signal-to-noise ratio and decent firing frequencies, a shorter data segment is already sufficient for robust filter estimation.

- The filters are designed such that the spike is detected in the "middle"of its waveform. For example, if the waveform has a length of 9 samples, the filters will detect the spike at sample 5.
- Sometimes the MVDR filters cannot be calculated (the most often reason being that no bump the SEA filter output was found). In such a case, spike detection should be done with the SEA filter.

References

[1] M. Natora et al., "Spike detection in extracellular recordings by hybrid blind beamforming", Proceedings of *32nd Annual International Conference of the IEEE EMBS*, p. 4636 - 4641, 2010