NOBIAS Quick Start Guide

Thanks for your interest in using NOBIAS! The NOBIAS algorithm (NOnparametric Bayesian Inference for Anomalous diffusion in Single-molecule tracking) is a two-module algorithm for analysis of multi-diffusive state SPT datasets that predicts the anomalous diffusion type for each state.

Written by Ziyuan Chen at the University of Michigan (ziyuanc@umich.edu).

Last updated: 7/16/2021

NOBIAS was written on MATLAB R2019a on a Windows machine.

Requirements

NOBIAS requires the lightspeed toolbox (https://github.com/tminka/lightspeed)

Please put the lightspeed folder in your MATLAB path, then change the current folder to the lightspeed directory and run `install_lightspeed`

Running lightspeed requires a C/C++ compiler. For Windows, if you don't have one installed, please try the 'MATLAB Support for MinGW-w64 C/C++ Compiler'. You can get this compiler by clicking the MATLAB Add-Ons button and searching for 'MinGW', or else see the lightspeed Github page for more options. For MacOS, Xcode is the suggested compiler. (https://developer.apple.com/xcode). We test the useage of NOBIAS in Windows and in MacOS.

Running the RNN module requires the MATLAB Deep Learning toolbox:
https://www.mathworks.com/products/deep-learning.html

Due to the change of network layers in different MATLAB version, our provided net might fail on MATLAB version prior to MATLAB 2019a.

Running the RNN module needs the MATLAB Mapping toolbox:
https://www.mathworks.com/products/mapping.html

NOBIAS

There are two modules in NOBIAS: the first is the HDP-HMM module and the second is the RNN module. The second module depends on the output from the first module as well as a pre-trained classification network.

NOBIAS HDP-HMM Module

Running the HDP-HMM Module

To test the HDP-HMM module with example data, go to the main NOBIAS folder, load the example data, and run the code with the commands:

```
load('exampledata.mat')
out = NOBIAS(data);
```

To run NOBIAS with your own data, the input data variable should be a structure variable with at least two fields: ‘obs’ and ‘TrID’. ‘obs’ is a $2 \times T$ matrix of observations where T is the total amount of steps and 2 indicates the dimensionality (2D tracks). Each element of ‘obs’ is the step size in that dimension. ‘TrID’ is the trajectory ID number, which denotes the track that step is from. It should be a $1 \times T$ integer vector. To do the motion blur correction, the input data must also have a third field, ‘cor_obs’, that denotes the correlation steps.

We provide a formatting function to prepare your experimental tracks (a lists of positions) in the NOBIAS input format (a sets of displacements). Please use command:

```
data = NOBIAS_preparedata(AllTracks);
```

Here, the ‘Alltracks’ input is an $N \times 1$ cell array (a collection of N tracks). Each element of the ‘Alltracks’ cell array is a matrix with at least 4 columns: the 1st column is a placeholder; the 2nd column is a time stamp in units of frame numbers—these numbers do not need to be continuous; the 3rd and 4th columns are the y and x coordinates, respectively, of the point along the track.

HDP-HMM Parameters

The `NOBIAS` ‘Params’ structure includes two parameters, ‘Params.frametime’ and ‘Params.pixelsize’, that depend on your experimental settings. Please indicate your imaging frame time in units of seconds and your imaging pixel size in units of µm.

The `NOBIAS` ‘Params’ structure also includes the analysis hyperparameters. For standard simulated tracks, NOBIAS uses $\alpha = 1, \gamma = 0.1,\ \kappa = 5$. For tracks with motion blurring, κ should be increased accordingly if your final results have multiple states that share very similar diffusion coefficients. For example, for our simulations that average 10 steps to create motion blurred tracks, $\kappa = 20 - 100$ is used for tracks of 10 to 100 steps.

$\alpha = 1, \gamma = 0.1$ can also be changed. See (Fox et al., 2008) for details about what they mean.

Input Simulation Code

We provide our code to produce simulated collections of tracks that are a mixture of several diffusive states with input transition probabilities. `NOBIAS_step_simu_standard` and `NOBIAS_step_simu_blur` simulate a standard Brownian motion mixture or a motion blurred mixture, respectively.

The NOBIAS RNN Module

Running the RNN Module

To run the RNN module, use the following commands:

```
load('net.mat')

state_model = predict_state_model(out, data, net, minseglength);

Model_Prob = NOBIAS_difmodel_plot(state_ model, out)
```

The input 'net.mat' needs to be pre-trained. We provide two pre-trained networks: 'net_20step_class_bmfbm01091119ctrw0109lw1119_750000' and 'net_40step_class_bmfbm01091119ctrw0109lw1119_750000' are trained with 20-step and 40-step trajectories, respectively. Note that these networks will only work for MATLAB R2019 and later; we will provide pre-trained networks for older versions in the future.

The inputs to `predict_state_model` are 'out' (the output from the HDP-HMM module), 'data' (the same input that was used in the HDP-HMM module), 'net' (the pre-trained network), and 'minseglength' (the minimum segment length to be used for this prediction).

If you don't have any track segments that are at least '`minseglength`' long for any one of the diffusive states, the RNN module might give an error/NA value.

The output `Model_Prob` is an $A \times B$ matrix where A is the number of diffusive states and B is the number of diffusion types in the training network. Each element of Model_Prob therefore gives the prediction probability of the diffusion type for that diffusive state. In our pre-trained networks, the four types of diffusion are: BM, FBM, CTRW, and LW.