Configuring the Two-Stage BP Test

Hannes Helgason

This document is a short description on how to configure the Two-Stage BP Test in ChirpLabDataStream. It focuses on the settings for the simulations done for the thesis "Nonparametric Detection and Estimation of Highly Oscillatory Signals" (see Chapter 4). See also the file `Demo.m`, which comes with ChirpLabDataStream. For further details about each function, see the documentation in the corresponding file (or type `help <FUNCTIONNAME>` in the Matlab command prompt).

1 Stage I

1.1 Configurations

Two main things need to be decided for the first stage of the test:

- Configurations for the monoscale chirplet graph. Amongst them are:
  - chirplet scale (i.e., the length of time-support for each chirplet in the graph)
  - range of frequency offsets and chirplet slopes
  - connectivities in the chirplet graph
- Which base intervals to consider for extension. In the simulations for the thesis we considered all dyadic intervals of lengths $L_1 = 2^7 = 128$, $L_2 = 2^8 = 256$, and $L_3 = 2^9 = 512$ in a data stream 65,536 samples long.

Sample configuration for the step I statistic as it would appear in an `m`-file. The first lines are for configuring the chirplet graph and the last ones for choosing lengths of base intervals:

```
% CONFIGURATIONS FOR THE STEP I STATISTIC
% Fs: sampling frequency  in Hz;  freqmin: minimum frequency in Hz
J = 8
N = 2^J;
sldf = 8;     % slope discretization factor
fmin = floor(freqmin/Fs*N);    % minimum discrete frequency in chirplet graph
fmax = N/2-1; % maximum frequency in chirplet graph
csc = 1;      % coarsest scale in chirplet graph
fsc = 1;      % finest scale in chirplet graph
graphParamStepI = GetChirpletGraphParam(N,csc,fsc,sldf,[],fmin,fmax,'COLOREDNOISE');
sbase = csc;      % scale index for the base time interval in search
smin = 7;    % scale index for the smallest time interval to consider, length 2^smin
smax = 9;   % scale index for the largest time interval to consider, length 2^smax
```

Once we have stored the chirplet graph configuration in a variable `graphParamStepI`, by using the function `GetChirpletGraphParam`, the chirplet norms for noise with the power spectrum `P` can be calculated with function `CTNormsDataStream` as follows:

```
% calculate chirp norms for STEP I
% P is a vector with the power spectrum of the noise
CnormStepI = CTNormsDataStream(P,graphParamStepI);
```

1.2 Data processing

Data processing in the first step of the procedure consists of a monoscale chirplet transform followed by the calculation of the BP statistic for each base interval under consideration. This can be done as follows, where `y` is the stream of data samples to process:

```
% Step I: Mono-scale statistic

% do mono-scale chirplet transform
D = 1./P;      % inverse of power spectrum
yy = ifft(D.*fft(y));
overwhitened = 1;
C = CTDataStream(yy,graphParamStepI,P,CnormStepI,overwhitened);

% find BP on all dyadic intervals of lengths 2^s, where s=smin,...,smax
% time-support of chirplets is 2^(graphParamStepI{1}(1)-sbase)
T = MonoScaleBP(C{1},M,smin,smax,sbase,graphParamStepI);
```

2 Stage II

2.1 Tagging promising intervals

For each scale of base intervals we order the values of the BP statistics in Stage I. Then we tag and extend the intervals corresponding to a desired proportion of the most extreme statistics. In the simulations for the thesis the tagged intervals corresponded to the 50, 30, and 12 biggest statistics for the base-interval lengths $L_1 = 128$, $L_2 = 256$, and $L_3 = 512$, respectively. This amounts to about 10% of the total number of dyadic intervals of each length since the data stream processed was 65,536 samples long. Assume we have calculated the mono-scale BP statistics as above and stored them in a variable `T`. Then we can tag "promising" intervals as follows:

```
% configurations for tagging and extension
numTagged = [50 30 12];

% Find promising intervals and extend them
taggedIntervals = TagIntervals2(T,smin,smax,numTagged);
```

The variable `taggedIntervals` stores a list of tagged intervals that can be processed further by the function `ExtIntAndFindBP`, as will be described shortly.

2.2 Configuring the chirplet graphs

An example of how to configure the chirplet graphs to use for the second stage can be found in the file `InitGraphParam.m`. This file can serve as a base for other configurations. Here is how the function could be called, where `P` is the same power spectrum as used in the mono-scale analysis in the first stage:

```
fname = 'graphParamAndNormStepII';
[graphParamExtInt, CnormExtInt, maxLength] = InitGraphParam(P,fname);
```

The chirplet graph configuration is stored in a variable `graphParamExtInt` and the chirplet graph norms in a variable `CnormExtInt`. These variables can of course be precomputed and stored for repeated use; in the example above these variables will be stored in a file with the name set in the variable `fname`.

2.3 Extending intervals and calculating the BP statistic

The function `ExtIntAndFindBP` is used for extending intervals and calculating the BP statistic. Using the parameters and variables set in the previous code examples, this can be done as follows, where the maximum extension level is set to 1:

```
lmax = 1; % extension-level

% Extend promising intervals and calculate BP on each of them
TT = ExtIntAndFindBP(yy,overwhitened,P,lmax,smin,smax,...
          taggedIntervals,graphParamExtInt,CnormExtInt,maxLength);

[Tmax, TmaxInd] = FindMaximumStats(TT);
```

The function `FindMaximumStats` is tailored for a configuration where the maximum extension level for the second stage is $l_{max} = 1$. This is a post-processing step which stores the most extreme statistic for each type of extended interval and chirplet path length. See the function's documentation for further information.