Steps for Running SMEE on BBH Waveforms

Nicholas Mangini

July 2013

1 Introduction

This serves as a description of the steps to follow in order to run `SMEE_BBH.m` on the three BBH waveform catalogues. The scripts discussed below are designed to run on the Q, HR, and RO3-series waveforms but it is a simple thing to change them for any new waveform catalogue. Section 3 will provide an example run from start to finish using the Q-series waveforms.

2 Methods

2.1 MDC frame Production

Burst MDC frames contain injections of gravitational wave signals. Creating them is a 5 (or 6) step process which, requires an installation of lalsuite. The easiest way of making MDC frames is to use one of the LSC clusters (e.g. Caltech or Atlas) which will have lalsuite already installed in the system. Below are the steps one needs to follow in order make these frames. This process must be repeated for each waveform and catalogue.

Step 1

To start the user will need a set of ascii files containing the waveforms accompanied by a metadata file. Metadata files contain extra information about the waveforms like the authors, mass ratio, spins, angles, etc. The only item(s) that are necessary are the lines assigning the spherical harmonic modes to the correct files. Each waveform in a catalogue will need a metadata file. As an example, below are a few lines from a metadata file for a Q-series waveform:

```
2,-2 = rStrain_FFI_l2_m-2_r75.00.asc
2,2 = rStrain_FFI_l2_m2_r75.00.asc
2,-1 = rStrain_FFI_l2_m-1_r75.00.asc
2,1 = rStrain_FFI_l2_m1_r75.00.asc
3,-2 = rStrain_FFI_l3_m-2_r75.00.asc
3,2 = rStrain_FFI_l3_m2_r75.00.asc
3,-3 = rStrain_FFI_l3_m-3_r75.00.asc
```

```
3,3 = rStrain_FFI_l3_m3_r75.00.asc
```

Step 2

Once the user haa the ascii and metadata files set up, the next step is to create a frame file of the waveforms. This is done using the lalsuite command `lalapps_fr_ninja`*. The following lines are necessary flags to run this command:

```
lalapps_fr_ninja
--verbose \
--format FORMAT \
--double-precision \
--nr-meta-file /PATH/TO/METAFILE \
--nr-data-dir /PATH/TO/DATA \
--output OUTFILE.gwf
```

The `FORMAT` option will be either `NINJA1` or `NINJA2`, depending on what type of ascii files you have. If the ascii files have the extension `.ampphase.asc` then `NINJA2` is required because `lalapps_fr_ninja` will need to read the ascii files differently. The `OUTFILE.gwf` option is the name of the frame this command will create. Be sure to choose a name that will be easily distinguishable from any other waveform frame files.

*Running this and subsequent commands work best if put into a bash script, and will thus be provided in the form that works for bash scripts. Also, for catalogues with a large number of waveforms it is useful to put the above command inside a for loop. Once the waveform frame files are made there is no need to run this command again.

Step 3

Next we need to make an xml file of the waveform that contains mass and spin parameters. This is done with the `lalapps_ninja` command using the following flags:

```
lalapps_ninja \
--datadir OUTPUTDIR \
--pattern PATTERN \
--min-mass-ratio MINRATIO \
--max-mass-ratio MAXRATIO \
--freq-lo FREQLO \
--outfile OUTFILE.xml
```

The `OUTPUTDIR` is where you want the new `xml` files to appear and `PATTERN` is a pattern in the waveform frames previously made that the command will look for. The `PATTERN` can be the exact name of a file, all the frame files in the directory (`*.gwf`), or any identifying feature in the frame file names. `FREQLO` is the frequency cutoff for the waveforms and `OUTFILE.xml`

is the output file which should be made easily identifiable from other xml files.

Step 4

Now we need to make a file containing the injection parameters we want. This is done with the command `lalapps_inspinj`:

```
lalapps_inspinj
--seed SEED \
--f-lower FREQLO \
--waveform WF \
--gps-start-time GPSSTART \
--gps-end-time GPSEND \
--time-step TSTEP \
--time-interval TINT \
--l-distr LDIST \
--d-distr DDIST \
--i-distr IDIST \
--fixed-inc INC \
--longitude LONG \
--latitude LAT \
--polarization POL \
--min-distance MINDIST \
--max-distance MAXDIST \
--nr-file OUTFILE.xml \
--m-distr MDIST \
--min-mtotal MINMTOTAL \
--max-mtotal MAXMTOTAL \
--min-mass1 MINMASS1 \
--max-mass1 MAXMASS1 \
--min-mass2 MINMASS2 \
--max-mass2 MAXMASS2 \
--output OUTFILE.xml
```

This command requires a lot of inputs but most are self explanatory. `SEED` is a random number, `WF` is the name you want to give to the MDC frame set, `TSTEP` is the time between injections, `TINT` is the time you allow the injections to vary from the time step, `LDIST` is the distribution of longitude and latitude for the system, `DDIST` is the distance distribution of the injections, and `IDIST` is the inclination angle distribution for the system. `MDIST` is the mass distribution of the system. This option is where most errors appear for this command. For a list of options available for this flag, type `$lalapps_inspinj --help` in the command line.

Step 5 (Optional)

An 'optimally oriented' system is a system with inclination and polarization angles of 0° which is located above a detector. In order to move the

injections over a detector, use the `inject_overhead.py` script. To run this script, enter `$python inject_overhead.py --output-file OUTPUTFILE INPUTFILE` where the `INPUTFILE` is the file created by `lalapps_inspinj`.

Step 6

The final step is to create the MDC frame files using the injection file created by `lalapps_inspinj` (or the xml file created by `inject_overhead.py`). This command is called `lalapps_mdc_ninja`:

```
lalapps_mdc_ninja
--verbose \
--injection-type INJTYPE \
--injection-file INPUTFILE.xml \
--gps-start-time GPSSTART \
--gps-end-time GPSEND \
--ifo IFO \
--sample-rate SAMPLERATE \
--write-mdc-log \
--frame-type FRAMETYPE \
--set-name SETNAME \
--mdc-log OUT.log \
--write-frame \
--freq-low-cutoff FREQLO \
--snr-low SNRLOW \
--snr-high SNRHIGH \
--out-xml-file OUT.xml \
--fr-out-dir OUTFUT/DIRECTORY \
--double-precision
```

This command mostly requires strings for naming files and the frames. `INJTYPE` depends on how the waveforms were generated, either with numerical relativity or classical approximations. `INPUTFILE.xml` is the file created in the previous step, and `OUT.log` is a log of what the command is doing which should have a unique name, along with the `OUT.xml` file. `FRAMETYPE` sets the name of the MDC frame file and the `SETNAME` names the injection set. These MDC file names can be very sensitive to syntax so avoid hyphens and periods when giving names.

For more command options and explanations for all the listed commands, type `$COMMAND --help` at the command line.

2.2 SMEE

Before we are able to run SMEE, we need to load the waveforms into a matrix, process the waveforms, calculate the principle components, process the PCs and injections, and then run SMEE.

2.2.1 Pre-processing and Principle Component Analysis

It is important to run the following scripts in the order listed without clearing your workspace in MATLAB. Follow this process for each desired waveform catalogue.

Pre-processing

Loading the waveforms and processing them is done with the `allmodes_loadmatrix.m` script. This script creates a matrix containing all the waveforms of a given catalogue, aligns the waveforms along their peaks, zeros the data after the ringdown, and makes each waveform have a duration matching the shortest waveform in the catalogue. The final step it follows is to combine all the spherical harmonic modes. In this script you can specify the angles θ and ϕ in the combining of modes, when the data is set to zero, and the total mass of the system (used in converting to physical time units).

Principle Component Analysis

PCA is performed using the `allmodes_pca.m` script. The math follows what is explained in the Wikipedia article in PCA except there is no calculation of the deviation from the mean and the PCs are organized into descending order of eigenvalues. The β values are then calculated along with the reconstructed waveforms.

Match (Optional)

The script `allmodes_match.m` will calculate the match between the original waveform and the reconstructed waveform. The resulting plot will tell you how many PCs are needed to reconstruct the original waveform with X% accuracy.

2.2.2 Post-processing

Once you have the PCs the next step is to load the injections from the MDC frames and resample the PCs to match the sampling rate of the injections.

Load Injections

Use the `load_frame.m` script to create a matrix of injections. This script is a function and requires the user to supply the catalogue of which you want the injections from. Within the script itself the user can specify when to start reading the frame and for how long to read the frame. This script makes use of the command `frgetvect` from the LIGO frame library for MATLAB. For a download and installation instructions visit:

`http://lappweb.in2p3.fr/virgo/FrameL/FrDoc.html`.

This command is also useful for plotting the MDC frame files. Here is the syntax for pulling the injection out of the MDC frame:

```
[data,timestamp] = frgetvect(FRAMENAME,CHANNELNAME,GPSSTART,DURATION)
```

To get the channel name use the command `$FrDump -i FRAMENAME` on the cluster where the MDC frames were created. From here it is simple to plot the injection using MATLAB.

Resampling

The PCs must be resampled to match the sampling rate of the injections. Run the `resample.m` script inside the directory which contains the PC matrix, time matrix, and injection matrix. Once the resampling is done the injections are aligned along their peaks and cut to the same duration as the PCs.

Shorten PCs and Injections

Because the three catalogues considered in this project are different lengths, the PCs and injections need to be cut to be the same length and their peaks must occur at the same time (for now). This is done with the `align_MDCs.m` script.

2.2.3 SMEE

SMEE is run with the script `SMEE_BBH.m`. The are a lot of arguments to this function but the minimum is six: `run_name`, `noisetype`, `model`, `catalogue`, `wv`, and `seed`. All the inputs are listed and described at the top of the script. The only thing that needs changes in the script are the minimum and maximum beta values. These values are obtainable from the `minbeta` and `maxbeta` vectors in the `betas_<catalogue>.mat` save file. For my project I adjusted the minimum and maximum values in the script to include all the values from each catalogue, meaning I could run `SMEE_BBH.m` on all the catalogues without editing the script in-between each run.

Some interesting options for the script are the `doPlot` and `typeofscaling`. The `doPlot` option, when set to 1, shows the nested sampling plot,this will however increase the runtime of the script. It is possible to scale the injections to a specific SNR or distance using the `typeofscaling` option.

SMEE will output many values and write them to a text file. In this text file is the calculated Bayes factor. Using the script `plot_bayes.m` the user can make plots of the difference in Bayes factor for each combination of PCs for every catalogue.

3 Example End-to-End Run

This section serves as an example run of SMEE, from start to finish. I will be using the Q-series catalogue for most of these steps but this process can be easily adapted to fit other BBH catalogues. The following steps are run on a cluster (e.g. Caltech or Atlas)

1. ```
   lalapps_fr_ninja
   --verbose
   --format NINJA1
   --double-precision
   --nr-meta-file ~/GATech_Studies/Waveforms/Q-series/D10_a0.0_q1.00_m103_Qs/*.bbh
   ```
   ```

```
--nr-data-dir ~/GATech_Studies/Waveforms/Q-series/D10_a0.0_q1.00_m103_Qs/
--output D10_a0.0_q1.00_m103_Qs.gwf
```

2. lalapps_ninja
```
--datadir ./
--pattern ./D10_a0.0_q1.00_m103_Qs.gwf
--min-mass-ratio 1.00
--max-mass-ratio 1.00
--freq-lo 10
--outfile sim-D10_a0.0_q1.00_m103_Qs.xml
```

3. lalapps_inspinj
```
--seed 76
--f-lower 10
--waveform GATech
--gps-start-time 900000005
--gps-end-time 900000110
--time-step 10
--time-interval 0
--l-distr fixed
--d-distr uniform
--i-distr fixed
--fixed-inc 0.0
--longitude 0.0
--latitude 0.0
--polarization 0.0
--min-distance 1000000
--max-distance 1000000
--nr-file ~/GATech_Studies/MDCframe_generation/Q-series/sim-D10_a0.0_q1.00_m103_Qs.xml
--m-distr nrwaves
--min-mtotal 250
--max-mtotal 250
--min-mass1 0.5
--max-mass1 0.5
--min-mass2 0.5
--max-mass2 0.5
--disable-spin
--output GHLTV-GATech-D10_a0.0_q1.00_m103_Qs-900000005-105.xml
```

4. python inject_overhead.py --output-file ...
GHLTV-GATech-D10_a0.0_q1.00_m103_Qs-900000005-105.xml ...
GHLTV-GATech-D10_a0.0_q1.00_m103_Qs-900000005-105.xml

5. lalapps_mdc_ninja
```
--verbose
--injection-type NR
--injection-file GHLTV-GATech-D10_a0.0_q1.00_m103_Qs-900000005-105.xml
```

```
--gps-start-time 900000005
--gps-end-time 900000110
--ifo H1
--sample-rate 16384
--write-mdc-log
--frame-type Q-series_q100
--set-name Q-series
--mdc-log D10_a0.0_q_1.00_m103_Qs.log
--write-frame
--freq-low-cutoff 10
--snr-low 0
--snr-high 1e6
--out-xml-file out-D10_a0.0_q1.00_m103_Qs.xml
--fr-out-dir ./MDC/frames/
--double-precision
```

After the previous steps, you will have a directory of MDC frames on a cluster. The next step is to copy the frames to a computer where SMEE will be run. To copy files from a cluster (e.g. Caltech) to a computer, run:

```
$gsiscp ldas-grid.ligo.caltech.edu:<filename> .
```

The following scripts are written to run in the same directory. The path to the ascii files will need to be changed based on the users folder setup. Run the scripts in the order provided and don't clear the workspace until finished.

6. `Q_allmodes_loadmatrix.m` - This script creates two matrices and saves them: the full waveforms after processing and mode combination (`data_Q-sereis.mat`) and a time matrix in physical units (`time_Q-series.mat`). Also created are matrices of the raw waveforms and raw time vector in NR units.

7. `Q_allmodes_pca.m` - This script creates a save file for the PC vectors (`vectors_Q-series.mat`) and β values (`betas_Q-series.mat`). Also created but not saved are the reconstructed waveforms.

8. `Q_allmodes_match.m` - This script is only meant to plot the match between the original and reconstructed waveforms.

9. `load_frame.m` - This script loads an MDC injection from the frame files and save them in a mtrix called `injection_Q-series.mat`. Also saved is the time vector for the injections called `Tinjection_Q-series.mat`. The times read from the frames files are set up to work for the later scripts and should only be touched if the user is trying to read in a new catalogue. The syntax for running this function is `$load_frame('Q')` for the Q-series.

10. `resampleQ.m` - This script creates three new matrices: resampled PC vectors (`RvectorsPC_Q-series.mat`), aligned MDC injections (`MDC_Q-series.mat`), and a resampled time vector which is for both the PC vectors and MDC injections (`resampledtime_Q-series.mat`).

11. `align_MDCs.m` - This script aligns all the MDC frames and PC vectors from all the catalogues at the same time and saves the new matrices with the same file names as in step 10 but prefixed with the word 'final.'

12. `SMEE_BBH.m` - To run SMEE the user needs to provide it with the PC vector matrix and the injection matrix along with a noise file (in this case `ZERO_DET_high_P.txt`). SMEE gives a lot of screen print outs and makes new save files each time it is run (see the bottom of `SMEE_BBH.m` for the outputs). The most important is the log of the Bayes factor which when printed to the screen is called `log(nested sampling Bayes factor vs Noise)`. If the user wants to reconstruct the 5^th waveform of the Q-series using eight PCs from the HR-series with one detector; the function would be run by:

    ```
    SMEE_BBH('Q-series','aligo','HR','Q',5,742,1,8)
    ```

    By setting `doPlot` to 1 a live updating plot while show the progress of the nested sampling process. To fix the SNR the user would put 'SNR' for the `typeofscaling` followed by the number for `scaling`.

13. `plot_bayesQ.m` - This script is set up to read the text files created by `SMEE_BBH.m` and plot the difference in Bayes factors between different models for a catalogue. The inputs to the function are used to read the text file the user wants. For example, if I want to plot the difference in Bayes factors for the Q-series with an SNR of 30 using 4 PCs, I would enter `plot_bayesQ(4,30)`