(Some) Waveform Simulation [MDC] Infrastructure

All-Sky Burst Call Oct 21st 2014

James A. Clark

Georgia Institute Of Technology

Bursts In LALSimulation

`LALSimulation`: simulation engine for well-defined, analytic waveforms. Can make the following *right now*:

- sine-Gaussians
- white noise bursts
- Gaussians (?)
- Cosmic strings
- ...

Astrophysical waveforms will require a little more work:

- Supernovae
- Binary coalescence with matter
- NR BBH (!)

MDC Generation With LALSimulation

Utilities in excess-power for e.g.,:

- MDC frames with user-specified distributions on optimal-SNR h_{rss}, frequencies, quality factors, bandwidths, durations etc
- Distributions created by `lalapps_binj` - creates XML file with a `sim_burst` which is readable by `LAL` codes.
- Already deployed in ETG comparison study, cosmic string search

Useful links:

- ETG trigger comparison wiki: https://wiki.ligo.org/viewauth/DetChar/ETGperformanceStudy
- Excess-power utilities: https://github.com/cpankow/excesspower-utils

Astrophysical Waveforms: NINJA Codes

`LALSimulation` and friends: parameterised waveforms. No current support for waveforms directly from numerical simulation (SNe, NR BBH, NSBH, BNS).

- Obvious candidate: `NINJA` codes[1]
- NINJA infrastructure: tools and standards for handling NR BBH data in a standard format, project signals with random extrinsic params onto detectors.
- Currently only support binary mergers (i.e., `sim_inspiral` tables); would be convenient to add support to `sim_burst` for e.g., SNe & injection-finding / characterisation using burst codes
- `NINJA` codes were used in IMR study, S6 IMBH analysis, post-BNS merger studies (and NINJA!)

[1] Currently broken, unmaintained...

LALSimulation & Swig

Worth noting injections easily performed in python using swig-wrapped LAL routines:

1. Construct LAL TimeSeries objects for h_+, $h_\times$ (e.g., read from file or generated with `LALSimulation`)
2. Generate sky-location, polarisation, detector site (can e.g., be read from `sim_burst`, `sim_inspiral` tables)
3. `SimDetectorStrainREAL8TimeSeries()` projects $h_{+,\times}$ onto this detector with these angles
4. Can then pass the TimeSeries or python arrays directly on for further analysis or write to frame with e.g. pylal frame library

LALSim & Swig

Currently using LALSim/Swig in LIB post-merger studies:

1. Read quadrupole moments $\ddot{I}_{xx}, \ddot{I}_{xy}, \ldots$ from file
2. Construct expansion parameters H_{lm} (see LIGO-T1000553)
3. Generate random $(\theta,\ \phi)$ and construct

$$h_+ - ih_\times = \frac{1}{D}\sum_{l=2}^{\infty}\sum_{m=-2}^{m=2} H_{lm}(t)^{-2}\, Y_{lm}(\theta,\phi) \qquad (1)$$

4. Choose detector, sky-location & project $h_{+,\times}$ onto detector
5. Write to frame, generate cache file
6. Call `lalinference_nest` with `subprocess.call()`
7. Delete frame, cache
8. repeat (3–7)...

Summary

- `LALSimulation` + `LALBurst` + `GSTLAL` = well documented, largely reviewed and easy way to generate MDCs
- Can be run by *anyone* with LDG access (i.e., software is installed system-wide)
- Used in ETG trigger study & cosmic string analysis
- Only standard ad hoc burst waveforms (+strings) currently supported
- NINJA can be used for unparameterized, astrophysical waveforms (e.g., NR mergers, SNe) but some maintenance required, details TBD for SNe
- Swig-wrapped `LAL`: high-level routines for waveform generation & injection - useful for development & plotting
- Wiki to collate info & examples coming soon!