A High-Throughput Computing Solution to Calculating Uncertainties Associated with Compact Binary Coalescence Searches in Gravitational Wave Observations

M. Brett Deaton
August 4, 2010

In gravitational wave astrophysics massive interferometers (like LIGO and VIRGO) are deployed to measure the stretch and compression expected from a gravitational wave passing through the earth at the speed of light. The task of finding a true gravitational strain in the blaring stochastic background of an interferometer is difficult. Expected strains are on the order of 10^{-21}; that is to say if we could build a detector stretching from here to Alpha Centauri we would expect it to stretch and compress by 10 μm, the width of a human hair. Fortunately there are astrophysical events (specifically compact binary coalescences, or CBCs) which are expected to produce measurable gravity waves and are well-enough understood that we can more or less precisely predict the structure of the wave. Binary systems of black holes or neutron stars, as they approach final coalescence, produce a sinusoid of rapidly increasing frequency and amplitude called an inspiral or chirp, followed by a complex merger, and finally a rapidly decaying ringdown. By employing matched filtering, CBC analysts can search through noisy detector data with a very specific waveform template, or more precisely a whole bank of templates representing a complete spectrum of parameters (mass ratio of the black holes, total mass of the system, orbital phase at coalescence, etc).

Various waveform families are used in compact binary coalescence searches. Different waveform families might correspond to different portions of an expected signal (the inspiral phase, the merger phase, the ringdown phase), different orders of approximation, or different resolutions of a numerical simulation. In order to speak with any confidence about a gravitational signal detected by a waveform family, and about the physical parameters characterizing the detected wave, it is important to understand the errors associated with each family. The magnitude of these errors depend upon incomplete or incorrect waveform templates and detector noise.

One of the primary methods used to characterize these errors is to mimic an actual search with a fake signal. The technique is loosely the following: inject a full target waveform into existing data (literally adding the gravitational wave signal on top of the time series of noise); search for this target with some bank of waveform templates (the family of templates whose errors we wish to characterize); examine the best match for effectiveness (that is signal to noise ratio, or how well the template matches the target waveform) and faithfulness (that is how closely the parameters of the template waveform match those of the target).

This is an interesting study, but as it stands it has limited scope. It answers the question, “How well does template family A perform at this very specific point in parameter space - the point represented by the target waveform?” A more comprehensive study would examine a whole spectrum of target waveforms against the family of templates. Such a study answers the question, “How well does template family A perform across the whole of the expected parameter space of incoming signals - the space represented by the spectrum of target waveforms?”

This is a computationally expensive project due to the volume of parameter space being explored and the fine granularity needed to fully characterize the space. For example, exploring waveforms due to compact binaries composed of some combination of 10-110 $M_\odot$-objects at a granularity of 1 $M_\odot$ requires us to run the search 10^4 times. Moreover, each of these 10^4 searches is independent of the others. Clearly, high-throughput methods provide an excellent solution to this computing bottleneck.

Before defining a solution, it is important to examine the size of the computational demands. An efficient maximization algorithm, often referred to as the amoeba method, is adopted from Numerical Recipes to search across the template bank for the best match to a given target waveform. For an exploration of a two dimensional parameter space (for instance the two binary masses), and at a reasonable tolerance of fit, the amoeba method compares approximately 100 templates to the target before finding the best match. Each comparison involves a fast Fourier transform and additionally several multiplications across the N vector members, where N is the number of sampled points in the target waveform (also the number of points in the template waveform). The lengths of the waveforms vary enormously, but we pad them all to the maximum length by the appropriate number of zeroes; so for an average search, the number of points in a waveform, N, is $\sim 10^5$ (~1.6 MB). Since 2 waveforms are stored in memory at a time, the approximate size of the program is 3-5 MB. In the current implementation matching a best-fit template to a target waveform takes between 10 and 100 sec. (I am uncertain about the cause of the variability in run time, but believe it has to do with the following: for some families of template waveforms there are reasonably good fits to a target in disparate regions of parameter space, and so the amoeba algorithm finds a maximum more cautiously.) If the code was run in serial, it could take anywhere from 2 hrs to 10 days of wall-clock time. The high-throughput computing solution breaks this into approximately 30min jobs. An initial script generates the condor submission file by splitting the work into chunks of approximately 30 target waveforms across some subdomain of the parameter space being explored. A final clean-up script gleans the important data from the output of these jobs (the effectiveness and faithfulness of the template family at each point in parameter space).