INTERPOLATION TECHNIQUES FOR MCMC PARAMETER ESTIMATION ON COMPACT BINARY COALESCENCE GRAVITATIONAL-WAVE SIGNALS

DANIEL J. STEVENS
Department of Physics and Astronomy
Weinberg College of Arts and Sciences, Northwestern University, Evanston IL
May 11, 2012

ABSTRACT

With gravitational-wave detection on the horizon, astronomers look for ways of extracting useful information from a detected gravitational wave. Like its electromagnetic cousin, a gravitational wave carries important information about the characteristics of its source, and these characteristics can be recovered through numerical analysis. Using one promising technique known as a Metropolis-Hastings Markov Chain Monte Carlo (MCMC) simulation, astronomers can produce a probability distribution over an entire parameter space describing gravitational wave signals; given the gravitational wave data, the MCMC produces a sequence of parameter samples whose distribution converges to the probability density on parameter space implied by the data. Although an MCMC simulation will produce the equilibrium probability distribution in an infinite amount of time, a simulation that runs for a finite amount of time may not. This work focuses on using a kD-tree sorting structure to improve MCMC sampling. We show that a simple sampling method effectively recovers an accurate probability distribution in two dimensions but performs worse than a non-interpolated run in nine dimensions. We explain how dimensionality issues and correlations between the nine parameters—which are not taken into account by the simple sampling method—can cause the simple sampling method to yield inaccurate distributions, and we compare these results to those from an interpolated MCMC simulation with a more sophisticated sampling method which takes correlation into account. Improving the convergence of an MCMC simulation through interpolation would allow for faster, more frequent analysis of gravitational wave signals as well as higher confidence in recovered probability distributions.

1. INTRODUCTION

1.1. Gravitational Waves and Compact Binary Inspirals

The detection and analysis of gravitational waves is a top priority for the astrophysical community. A prediction of Einstein’s theory of general relativity, gravitational waves are ripples in space-time generated by the acceleration of massive objects. Of particular interest are gravitational waves created by the coalescence of a pair of compact objects, particularly neutron star-neutron star (NS-NS), neutron star-black hole (NS-BH) and black hole-black hole (BH-BH) pairs. In such a binary system, the compact objects lose orbital energy in the form of gravitational waves, and the objects spiral inwards (Peters & Mathews 1963). The amplitude and frequency of the signal increase as the objects move closer to each other, culminating in a strong chirp in the signal as the objects coalesce (Creighton 2003). Figure 1 shows a simulated inspiral waveform with the characteristic chirp at the end.

A gravitational wave from a compact binary inspiral can provide significant information about the wave’s progenitor. Using Bayesian inference techniques, we can recover the mass ratio of the compact objects; the orbital phase, polarization, inclination, declination and right-ascension of the system; the mass of the inspiraling object when the system emits the gravitational-wave chirp (chirp mass) the distance to the system; the time at which the system coalesced; and the projection of each object’s spin along each of the three spatial dimensions. Precise recovery of these parameters can help us place constraints on models of binary population synthesis (Abadie et al. 2010). Moreover, knowing the sky location of a binary merger from the detected gravitational wave can allow for detection of the merger’s electromagnetic radiation; this information can be used to calculate cosmological distances (Bloom et al. 2009); in the specific case of a NS-NS binary merger, the waveforms are simple enough to yield precise distance measurements (Creighton 2003).

The Laser Interferometer Gravitational-wave Observatory (LIGO) will be sensitive to gravitational waves from these and other sources. LIGO uses three laser interferometers—two in Hanford, Washington and a third in Livingston, Louisiana—to detect the waves. As a gravitational wave propagates through the detector, one 4km arm of an interferometer will expand while the other contracts, and the difference between the two arms reveals...
the form of the gravitational wave. Figure 2 shows the strain sensitivity in the LIGO detectors as a function of frequency in the past five runs.

Ongoing upgrades to LIGO, called Advanced LIGO, will improve the sensitivity of the gravitational-wave detectors. Advanced LIGO is predicted to have a factor of ten increase in sensitivity to most signals, yielding a factor of 1,000 increase in the volume of detectable sources (Creighton 2003). In partnership with the Virgo gravitational-wave detector in Italy, Advanced LIGO should detect an estimated 40 NS-NS inspiral events per year (Abadie et al. 2010) as well as 13 NS-BH and 500 BH-BH events per year (Creighton 2003). Such a large number of predicted detectable events makes the development and implementation of efficient parameter estimation methods a critical priority.

1.2. Markov Chain Monte Carlo

We are interested in estimating the parameters of a gravitational-wave emitting compact binary inspiral. The set of all mathematically-possible combinations of parameters form a parameter space. This space can have several dimensions, depending on the physical nature of the wave’s source (Mandel 2010). For our analysis, this paper focuses on non-spinning binary inspirals, resulting in a nine-dimensional space of parameters as discussed in the previous section. To estimate the parameters of a compact binary inspiral, we have developed a Markov Chain Monte Carlo (MCMC) algorithm to find the posterior probability-density function (PDF) of a source’s possible parameters. The MCMC algorithm uses Bayesian inference to estimate, given data $d$ from a gravitational-wave detector, the PDF over the entire parameter space. For our data, we inject a simulated waveform into simulated detector noise. We Fourier transform the data to examine it as a function of frequency rather than time, as discussed in van der Sluys et al. (2008). From Bayes’ theorem, the posterior for a proposed set of parameters $\tilde{\lambda}$ given the waveform data $d$ is given by

$$p(\tilde{\lambda}|d) = \frac{p(\tilde{\lambda}) L(d|\tilde{\lambda})}{p(d)},$$

where $p(\tilde{\lambda})$ is the prior probability distribution of the parameters — our initial guess for the relative probabilities of each set of parameters. $p(d)$ is the evidence, and $L(d|\tilde{\lambda})$ is the likelihood, or the conditional probability of the detector data given the set of parameters. We can constrain our prior based on the most extreme sources to which Advanced LIGO will be sensitive as well as physical constraints. For example, we can assign zero probability to negative values of the chirp mass, and we can also assume a uniform distribution on the sky location parameters; the latter simplification follows from the assumption that the distribution of gravitational-wave sources is isotropic (Mandel 2010). The evidence can be ignored in our calculations; $p(d)$ is constant, being a property of the model, and the term will drop out as we consider the ratio of posteriors of two parameter sets (Mandel 2010). Thus, we can re-express Equation (1) as

$$p(\tilde{\lambda}|d) \propto p(\tilde{\lambda}) L(d|\tilde{\lambda}).$$

The likelihood for a given detector follows from the assumption that the detector noise is stationary and Gaussian, with RMS magnitude given by $S(f)$ (see Figure 1.2) at frequency $f$ (van der Sluys et al. 2008). Given a data set $\tilde{d}(f)$ and waveform model $\tilde{m}(\tilde{\lambda}, f)$, is

$$L(d|\tilde{\lambda}) \propto \exp \left( -2 \int_0^\infty \frac{|\tilde{d}(f) - \tilde{m}(\tilde{\lambda}, f)|^2}{S_n(f)} df \right).$$

We need not worry about the constant of proportionality in (3), since we will be considering ratios of likelihoods. Since we assume that the noise in a given detector is independent from the noise in other detectors, Equation (4) becomes

$$p(\tilde{\lambda}|d) \propto p(\tilde{\lambda}) \prod_{i=1}^N L_i(d|\tilde{\lambda}),$$

where $N$ is the number of detectors (van der Sluys et al. 2008). For this paper, we consider data from the two Hanford detectors and the Livingston detector, so $N = 3$.

The MCMC algorithm samples the posterior distribution as follows: the MCMC starts with a set of parameters $\tilde{\lambda}_i$, in the parameter space. The algorithm proposes a new set of new parameters $\tilde{\lambda}_{i+1}$ via a jump proposal – some prescribed method for exploring the parameter space which we are free to choose, so long as it can propose points anywhere in the parameter space (Metropolis et al. 1953). The MCMC calculates the probabilities $P(\tilde{\lambda}_i \rightarrow \tilde{\lambda}_{i+1})$ and $P(\tilde{\lambda}_{i+1} \rightarrow \tilde{\lambda}_i)$ of jumping from one set of parameters to the other, also known as the jump probabilities; $P(\tilde{\lambda}_i \rightarrow \tilde{\lambda}_{i+1})$ is the forward jump probability, and $P(\tilde{\lambda}_{i+1} \rightarrow \tilde{\lambda}_i)$ is the backward jump probability. To sample the posterior PDF, the jump probabilities must satisfy detailed balance (Press et al. 2007):

$$p(\tilde{\lambda}_i)P(\tilde{\lambda}_i \rightarrow \tilde{\lambda}_{i+1}) = p(\tilde{\lambda}_{i+1})P(\tilde{\lambda}_{i+1} \rightarrow \tilde{\lambda}_i).$$

Once the jump probabilities and posteriors have been calculated, the algorithm then calculates the probability $P_{\text{acc}}$ of accepting the proposed parameters, called the acceptance probability:

$$P_{\text{acc}} = \min \left\{ \frac{p(\tilde{\lambda}_{i+1})}{p(\tilde{\lambda}_i)} \right\} = \min \left\{ \frac{L(d|\tilde{\lambda}_{i+1})}{L(d|\tilde{\lambda}_i)} \right\}.$$
\[
P_{acc} = \frac{p(\tilde{x}_{i+1}) P(\tilde{x}_{i+1} \rightarrow \tilde{x}_i)}{p(\tilde{x}_i) P(\tilde{x}_i \rightarrow \tilde{x}_{i+1})} \quad (6)
\]

The proposed parameters are accepted if the acceptance probability is larger than some random number \( k \) between zero and one, with the random number being generated anew every iteration. If the proposed parameters are accepted, then the points are recorded and become the current parameters for the next iteration; otherwise, the original parameters are repeated in the chain, i.e. \( \tilde{x}_{i+1} = \tilde{x}_i \). In this way, the current state depends only on the previous state, which gives the algorithm its Markovian nature. These accepted parameters give an estimate of the equilibrium posterior PDF over the parameter space; our MCMC code records the current parameters every 100 iterations. The recovered posterior PDF will eventually converge to the true equilibrium distribution if the jump proposals satisfy detailed balance (Metropolis et al. 1953).

1.3. MCMC Convergence Issues

This guaranteed convergence theorem, however, does not guarantee convergence in finite time. The MCMC algorithm, while being able to sample the entire parameter space, will preferentially sample high-likelihood regions – regions of the parameter space in which the posterior PDF has local maxima. The parameter space for gravitational wave signals is often multi-modal, containing several separated local likelihood maxima. For an MCMC simulation that runs for a finite duration, it is possible for the simulation to spend a disproportionate amount of time sampling one maximum while under-sampling another maximum; in this case, the recovered posterior distribution will not be a reasonable approximation to the true posterior distribution, as the MCMC has not converged.

We use parallel tempering to try to sample the entire parameter space while still exploring regions of high likelihood in detail. For parallel tempering, several MCMC chains run simultaneously, each with a different temperature \( T \). For each chain, the acceptance probability becomes

\[
P_{acc} = \left( \frac{p(\tilde{x}_{i+1}) P(\tilde{x}_{i+1} \rightarrow \tilde{x}_i)}{p(\tilde{x}_i) P(\tilde{x}_i \rightarrow \tilde{x}_{i+1})} \right)^{1/T} \quad (7)
\]

In this way, higher-temperature chains still prefer jumps to high-likelihood regions, but these chains are more likely to jump around the entire parameter space and explore different modes in the probability distribution (van der Sluys et al. 2008). By swapping parameters between chains, we allow the simulation to both explore wide regions of the parameter space as well as the high-likelihood regions. We swap between chains of temperatures \( T_m \) and \( T_n \), \( T_m < T_n \), whenever

\[
\left( \frac{L_n}{L_m} \right)^{\frac{1}{T_m} - \frac{1}{T_n}} > k
\]

where \( k \) is again a random number between zero and one (van der Sluys et al. 2008). The lowest temperature is set to \( T_{min} = 1 \), and this chain acceptance probability becomes Equation (6). The remaining temperature values depend on the signal-to-noise ratio (SNR): a higher SNR leads to a higher maximum temperature (van der Sluys et al. 2008). The number of chains is a compromise: fewer chains improves MCMC simulation computation speed, whereas more chains yield more efficient temperature swaps (van der Sluys et al. 2008).

Parallel tempering, however, is computationally expensive. Each temperature chain in a parallel tempering setup is an individual MCMC; our code runs eight chains, which considerably increases the hardware and time needed to produce results. A less-expensive method, which is the focus of this paper, is to interpolate between the MCMC simulation data. This method proposes jumps based on accepted points from a previous or ongoing MCMC run. Two variations of this approach are discussed below.

2. INTERPOLATIVE MCMC JUMP PROPOSALS

2.1. kD-Tree Jump Proposal

2.1.1. Overview

The first attempt at interpolating the data from an MCMC simulation uses a jump proposal based on a kD-tree. A kD-tree is a sorting algorithm that partitions an entire k-dimensional space into cells (or leaves). In the first step, the algorithm places a set of k-dimensional data points into a single cell that contains the entire parameter space. The algorithm then further partitions the space according to the following recursion (Farr & Mandel 2011):

1. If the given cell contains a single point, stop.
2. If the given cell contains more than one point, partition the current cell into two sub-cells by choosing a dimension \( x \) and dividing the points in half along this dimension. In this way, every point on one side of this division has a smaller \( x \)-coordinate than every point on the other side of the division.
3. Repeat the previous two steps for each sub-cell and store the resulting kD-trees as sub-trees of the current cell.

See Figure 3 for a two-dimensional illustration of this kD-tree sorting method.

To use kD-tree interpolation as the basis for a jump proposal, we store accepted parameters from a previous MCMC, or from the past history of an ongoing MCMC, and sort them into a kD-tree. To select parameters, the algorithm uniformly chooses a set of parameters from this sorted data set. The proposal then finds the nearest cell containing this point and at most \( N \) total points. The choice of \( N \) is a compromise: smaller \( N \) allows us to explore the local structure of the posterior with better precision, whereas larger \( N \) enables us to smooth out statistical fluctuations. For our simulations, we set \( N = 64 \) (Weinberg 2009). From within this cell, the proposal selects a set of proposed parameters \( \tilde{x}_{i+1} \) by generating a point from anywhere within this cells bounds. The MCMC performs this search to establish the initial parameters \( \tilde{x}_0 \), and then the jump proposal uses this
method to select proposed parameters at every iteration of the simulation. For this proposal, the forward jump probability is

$$P(\lambda_i \rightarrow \lambda_i + 1) = \frac{N_{\text{cell}}}{NV},$$  \hspace{1cm} (9)$$

where $N_{\text{cell}}$ is the total number of points in the tree (i.e. the total number of points from the original run) and $V$ is the volume of the cell containing $\lambda_i + 1$.

The backward jump probability is calculated by finding the cell whose bounds would contain $\lambda_i$. This probability is

$$P(\lambda_i + 1 \rightarrow \lambda_i) = \frac{N'_{\text{cell}}}{NV'},$$  \hspace{1cm} (10)$$

where $N'_{\text{cell}}$ is the number of previously-accepted points which lie within the bounds of the cell containing $\lambda_i$ and $V'$ is the volume of that cell. The acceptance probability is still given by Equation (6), but accepted parameters are not added to the tree; they are simply recorded and become the current parameters for the next iteration.

This interpolated proposal can still sample the entire parameter space, but should better explore high-likelihood regions, as most of the original points should be in high-likelihood region. Moreover, this proposal still satisfies detailed balance and retains the Markovian property.

For this jump proposal, we inject a simple waveform that simulates a non-spinning compact binary inspiral. The non-interpolating MCMC produces a posterior that converges quickly to the true posterior, making it ideal for testing the convergence of the first kD-tree jump proposal.

2.1.2. Two-Dimensional Test Simulation and Results

We first tested the kD-tree jump proposal in two-dimensional space, covering the chirp mass (mc) and the mass ratio (eta), but by limiting the MCMC to estimating only two parameters, we improve the computational speed of the MCMC. We ran one MCMC simulation without the kD-tree jump proposal and took the data from the lowest-temperature chain; we then sorted these parameters into a two-dimensional kD-tree for the interpolating simulation, which ran for a comparable number of iterations. Figure 4 shows the mass ratio posterior PDFs for the 2D original and interpolated simulations, while Figure 5 shows the corresponding chirp mass posterior PDFs. For both the mass ratio and the chirp mass, the posterior recovered with the interpolated jump proposal appears to have the same shape as the posterior recovered with the original jump proposal. However, for both parameters, the posterior appears to be shifted to the right. By inspection, the interpolated posteriors match the original posteriors closely enough to warrant a more rigorous run.

2.1.3. Full Non-spinning Simulation and Results

After the two-dimensional simulation, we tested the kD-tree jump proposal in a full nine-dimensional simulation. Using the same injected waveform as before, we took the accepted parameters from the lowest-temperature chain of a non-interpolating, nine-dimensional run, sorted them into a kD-tree, and ran the MCMC using only the kD-tree jump proposal. Figure 6 shows compares the recovered mass ratio and chirp mass posteriors for the original and interpolated runs. In higher dimensions, the kD-tree proposal yields considerably imprecise posteriors. From Figure 6, the kD-tree proposal recovers maxima for values of the chirp mass and mass ratio which are significantly different from the values which yield local maxima in the non-interpolated runs.
The PDF volume is $d \sim \log_2(N)$. From Equation (11), the kD-tree should have at least $4 = 2^2$ points to subdivide the two-dimensional parameter space once along each dimension; the kD-tree will split the points along one dimension such that two points lie on either side of the split, and then the tree will subdivide these cells by splitting along the second dimension such that exactly one point lies in each bottom-level cell.

Similarly, a single partitioning of the nine-dimensional parameter space along each dimension requires that the kD-tree contain about $2^9 = 512$ points. Similarly, the approximate number of points needed to subdivide the parameter space along each dimension $n$ times is

$$nd \sim \log_2(N).$$

A single division along each dimension does not constitute a "reasonable" subdivision of the parameter spaces under investigation, however. Given that the posterior PDFs are roughly a factor of 10 smaller in most dimensions, being bounded by physical constraints, the total volume of the PDF is $\sim 10^{-d}$ of the total volume of the $d$-dimensional parameter space. In two-dimensional space, the PDF volume is $\frac{1}{100}$ of the total volume. Assuming that each division from a kD-tree cuts the current cell in half, splitting along each of the two dimensions yields cells that have $\left(\frac{1}{2}\right)^2 = \frac{1}{4}$ the volume of the top-level cell. Thus, to achieve a reasonable subdivision of the space, we would need to split along each dimension three times to yield cells whose volumes are roughly $\frac{1}{64}$ of the total volume. From Equation (12), this procedure would require approximately $N = 2^{3 \cdot 9} = 64$ points.

Similarly, the total volume of the PDF in nine-dimensional space is $10^{-9}$ of the total volume. Creating cells of comparable size would require approximately three splits along each dimension, yielding a relative cell volume of $\left(\frac{1}{2}\right)^{3 \cdot 9}$. This subdivision requires, from Equation (12), approximately $2^{3 \cdot 9} \approx 134,000,000$ points. The computational cost of running an MCMC long enough to acquire so many data points makes this kD-tree jump proposal inefficient, if not impractical, for simulations in high-dimensional parameter spaces.

Additionally, neither the kD-tree nor the associated jump proposal take into account orientation of the points in a given cell. Figure 7 shows the density of accepted parameters in the inclination-distance plane from the non-interpolated simulation. This cross-section shows that these parameters do not align with the distance and iota coordinate axes. However, the kD-tree cells do align with the coordinate axes by construction. This can lead to the jump proposal proposing parameters which lie outside of the neighborhood of the sorted parameters, thereby inaccurately sampling the posterior along these dimensions.

**Figure 7.** A non-interpolated point density cross-section in the inclination-distance (iota-dist) plane. The parameters are not aligned orthogonally with respect to the distance and inclination axes; hence, no matter how small we make the cells, the kD-tree jump proposal will likely propose parameters away from the neighborhood of points.

### 2.2. Principal Component Cell Jump Proposal

#### 2.2.1. Overview

A more sophisticated jump proposal built upon the kD-tree sorting structure is the Principal-component Cell (PCC) jump proposal. As in the aforementioned kD-tree jump proposal, the PCC proposal will draw uniformly from the existing points. Once a point is selected, the proposal finds the largest cell containing this point and containing no more than $N$ total points; as with the previous proposal, $N = 64$ for our simulations. The proposal then shifts the cell, translating its center to the mean of all the points in the cell, and aligns the cell boundaries along the principal axes of the cloud of points in the cell. The principal axes are determined by calcu-
This jump proposal, like the simpler kD-tree proposal discussed in Section 2.1, should better explore high-likelihood regions. Currently, the intersection of the PCC cell and the original cell volumes is not calculated in computing the forward and backward jump probabilities. Instead, only the PCC cell volume is used for the calculation, although the algorithm does verify that the proposed parameters lie inside the original cell. If the proposed parameters lie outside of the original cell, then the calculated proposal probability is incorrect and the proposed jump is rejected. Since the regions of $V_{PCC}$ which lie outside of $V_{cell}$ are likely a small fraction of the total volume, we do not expect this to be a significant concern.

2.2.2. Markovian and Asymptotically-markovian Jump Proposals

In the PCC jump proposal, accepted parameters are added to the kD-tree, altering the structure of the tree as well as the means and covariance matrices for the cells to which the new parameter set belongs. At first glance, this seems to violate the requirement that a jump proposal be Markovian – the requirement that the current state of the system depends only on the most-recent previous state. However, a jump proposal need only be asymptotically Markovian to be a valid MCMC jump proposal. If each change to a jump proposals state modifies the proposal by successively smaller amounts, then, after sufficiently many iterations, the system will behave in a nearly-Markovian manner (ter Braak & Frugt 2008). To see how this jump proposal is asymptotically Markovian, examine the mean point and covariance matrix of a given cell. As we add a new point to a cell, the jump proposal updates the cells mean point $\bar{\mu}$ and covariance matrix $\mathbf{S}$ according to the recursive relations

$$\bar{\mu}_{i+1} = \bar{\mu}_i + \frac{(\bar{\lambda}_i - \bar{\mu}_i)}{i+1}, \quad (15)$$

and

$$S_{mn}^{i+1} = \frac{i}{i+1} S_{mn}^i + \frac{1}{i+1} (\bar{\lambda}_i \bar{\lambda}_i - \bar{\mu}_i \bar{\mu}_i), \quad (16)$$

where the superscripts indicate the corresponding entry in the matrix or vector. For the mean vector, as $i \to \infty$, the second term on the right-hand side of Equation (15) becomes arbitrarily small; hence, for sufficiently large $i$, $\bar{\mu}_{i+1} \approx \bar{\mu}_i$. For the covariance matrix, as $i \to \infty$, the second term on the right-hand side of Equation (16) approaches zero, while the first term approaches $S_{ii}^i$. Hence, changes to the PCC jump proposal state become successively smaller, so the PCC jump proposal is asymptotically Markovian.

2.2.3. Non-spinning Simulation and Results

We tested the PCC jump proposal on a new nine-dimensional, non-spinning simulation. For comparison, we ran the simulation with the same waveform without the PCC jump proposal. Comparison plots for the PCC-proposal simulation and the non-interpolated simulation are given in Figure 9. Not only does the PCC proposal perform much better in sampling the chirp mass and mass ratio posteriors’ maxima than the simpler kD-tree jump proposal, but it also recovers peaks in the other
parameters’ posteriors with considerable accuracy.

Figure 10 shows the cumulative parameter acceptance ratio (the number of parameters accepted over the total number of proposed parameters) versus the number of proposed parameters, in hundreds. The MCMC rejects a significant amount of parameters proposed by the PCC jump proposal. We can tolerate such a low acceptance

---

**Figure 9.** Plots of non-spinning inspiral simulations using the interpolating PCC jump proposal (yellow) and non-interpolating jump proposal (purple). These plots compare the recovered posteriors for chirp mass (mchirp), mass ratio (eta), declination (dec), right ascension (ra), inclination (iota), orbital phase (phi orb), polarization (psi) and distance (dist).
ratio: the goal of the PCC jump proposal is to explore high-likelihood regions in detail, so we do not want to jump around the parameter space too frequently. Overall, the principal-component cells seem to solve the high-dimensionality problem, at least in the nine-dimensional parameter space of non-spinning inspirals.

3. CONCLUSIONS AND FUTURE WORK

From the trial MCMC runs, we note that the original kD-tree jump proposal is insufficient for interpolating high-likelihood regions of a high-dimensional parameter space. The kD-tree proposal's inability to recover an accurate posterior can be explained by the exceptionally large number of points that need to be sorted into the tree to fill in a reasonable subdivision of nine-dimensional parameter space. Moreover, the cells used for this proposal are not tight enough, leading to a lower parameter acceptance rate as "bad" parameters are proposed too frequently. The PCC jump proposal, on the other hand, more accurately recovers the posterior and explores posterior maxima in nine-dimensional parameter space. This contrast is likely due to the PCC proposal drawing points from cells that are tighter and better aligned with the points in the cell as opposed to the standard kD-tree cells. Additionally, the PCC jump proposal's ability to hop between modes in the nine-dimensional posterior is vastly improved over that of the original kD-tree jump proposal.

The PCC jump proposal appears to work sufficiently well for non-spinning inspirals. Pending satisfactory results from additional testing and more robust statistical analysis, the PCC proposal should be an efficient complement to, or replacement for, the computationally-expensive parallel tempering method currently employed by our group. Given the results in nine dimensions, this proposal should be tested on inspirals in which one or both of the compact objects spin. Estimating the spin parameters along with the other nine parameters yields a 15-dimensional parameter space. Posterior in this space can be even more multi-modal, and the potential for high-dimensionality issues increases significantly.

Additionally, the kD-tree data structure itself is a powerful sorting method, and it could form the basis of other data-analysis techniques. The kD-tree structure could be used for clustering analysis, nearest-neighbor analysis, and other geometric comparisons of MCMC simulation data.

4. ACKNOWLEDGMENTS

First and foremost, I would like to thank Professor Vicky Kalogera. Vicky welcomed me into the LIGO group at Northwestern while I was only a sophomore, giving me my first undergraduate research project as well as funding to pursue it during the summer; that project became the foundation for this senior thesis. Her mentorship as a teacher, a major adviser and a research adviser have shaped, in no small way, this thesis and my post-undergraduate plans.

I would also like to extend my sincerest gratitude to Will Farr, who has played a significant part in the inception and development of this project. Will's guidance and encouragement have been invaluable throughout my undergraduate research career. This project was my first real exposure to numerical research in astrophysics, and Will's patience and willingness to help me learn as I went cannot be understated.

Additionally, I wish to thank the other members of the Northwestern LIGO group for welcoming me into the collaboration and for their assistance with this project. Specifically, I want to thank Ilya Mandel for his help in the beginning stages of the project and his continued investment and insight throughout. I would also like to thank Vivien Raymond for his assistance with data analysis following the nine-dimensional MCMC with the kD-tree jump proposal.

Finally, I must thank my family, friends and fellow undergraduate physics and astronomy researchers their encouragement and support, both direct and indirect.

This project was funded by Vicky Kalogera during the summer of 2010 and by a Northwestern University Summer Undergraduate Research Grant during the summer of 2011.

REFERENCES

Creighton, T. 2003, Classical and Quantum Gravity, 20, 853
Mandel, I. 2010, Phys. Rev. D, 81, 084029, 0912.5531

Figure 10. The cumulative parameter acceptance ratio for the PCC jump proposal as a function of the number of steps (in hundreds).