Project 39: Bouncing Hamiltonian Monte Carlo for supernovae cosmology and beyond

Harvey Williams
Supervised by: Dr Will Handley

Easter 2025

Abstract

We introduce *Bouncing Hamiltonian Monte Carlo* (Bouncing-HMC), a novel variant of Hamiltonian Monte Carlo designed to sample efficiently from target distributions with hard constraints. By treating the Markov-chain state as a physical particle that "bounces" elastically off forbidden regions, Bouncing-HMC preserves both volume and the Hamiltonian, and hence the usual Metropolis–Hastings acceptance. We prove that on-the-spot reflections are reversible and volume-preserving, and we demonstrate empirical correctness via Kolmogorov–Smirnov tests against direct rejection sampling. We use a GPU-accelerated implementation of our algorithm written with JAX/BlackJAX and apply it to the inference of binary black-hole merger parameters, where many hard physical bounds arise. Unfortunately due to poor tuning, our samples are highly correlated as they remain in the "burn-in" region of sample space and fail to make complete orbits around the typical set of samples which dominates the true distribution. We suggest automating tuning using an initial warm-up phase.

1 Introduction

Markov chain Monte Carlo (MCMC) methods underpin modern Bayesian inference by enabling us to draw representative samples from complex posterior distributions. Among these, Hamiltonian Monte Carlo (HMC) (Duane et al., 1987) stands out for its ability to exploit gradient information, achieving far lower autocorrelation than random-walk proposals. However, many scientific applications impose *hard constraints* on parameters—regions where the prior or likelihood vanishes and standard HMC must either reject or awkwardly reparameterise. Such constraints arise naturally in supernova cosmology (e.g. positive fluxes) and in gravitational-wave inference (e.g. mass ratios, spin magnitudes), yet existing approaches (e.g. reflective boundary sampling, bounding transformations) can suffer from bias or poor mixing near the boundary.

In this work, we present *Bouncing Hamiltonian Monte Carlo* (Bouncing-HMC), which handles hard constraints by reflecting the momentum vector whenever a trajectory attempts to cross a boundary. By performing these reflections "on the spot" (i.e. without first moving up to the trajectory), we maintain the volume-preserving and time-reversible properties required for detailed balance, and we keep the acceptance high by conserving the Hamiltonian exactly at the bounce. We first review the theoretical foundations of HMC and and suggest a normalizing-flow interpretation, then derive the Jacobian of the on-the-spot reflection map to prove volume preservation. We validate Bouncing-HMC both analytically and empirically, including a Kolmogorov–Smirnov comparison to rejection sampling, and we illustrate its performance on the nontrivial problem of sampling binary black-hole merger parameters with multiple hard parameter bounds. Finally, we describe our JAX/BlackJAX implementation, which leverages GPU acceleration and automatic differentiation, and we outline directions for adaptive mass-matrix tuning in future work. "‘

2 Theoretical Background

2.1 What is Hamiltonian Monte Carlo?

Hamiltonian Monte Carlo (HMC) was first introduced as Hybrid Monte Carlo by Duane et al. (1987) in the context of lattice QCD calculations. Given a (possibly not-normalised) target probability distribution that we can evaluate pointwise, and a means of drawing random samples from a gaussian distribution, HMC allows us to draw samples from the target distribution.

HMC is a variant of the Metropolis-Hastings sampling algorithm and makes use of a Metropolis-Hastings acceptance ratio. It is therefore instructive to outline the Metropolis algorithm first:

2.1.1 The Metropolis-Hastings Algorithm

We wish to draw samples x from a target density $\pi(x)$. The Metropolis-Hastings algorithm proceeds as follows:

Algorithm 1 Metropolis Sampling

Require: Initial state $x^{(0)}$, proposal kernel $q(x' \mid x)$

1: **for** $t = 0, 1, 2, \ldots$ **do**
2: Propose $x' \sim q\big(\cdot \mid x^{(t)}\big)$.
3: Compute acceptance probability

$$\alpha = \min\!\left(1,\ \frac{\pi(x')\, q\big(x^{(t)} \mid x'\big)}{\pi\big(x^{(t)}\big)\, q\big(x' \mid x^{(t)}\big)}\right).$$

4: Draw $u \sim \mathrm{Uniform}(0, 1)$.
5: **if** $u < \alpha$ **then**
6: Accept: $x^{(t+1)} \leftarrow x'$.
7: **else**
8: Reject: $x^{(t+1)} \leftarrow x^{(t)}$.
9: **end if**
10: **end for**

The Metropolis-Hastings algorithm is an example of a Markov chain.

Markov Chain. A stochastic process $\{X_t\}_{t \geq 0}$ on a state space $\mathcal{X}$ is called a *Markov chain* if, for all t and all states $x_0, x_1, \ldots, x_{t+1} \in \mathcal{X}$,

$$\Pr\big(X_{t+1} = x_{t+1} \,\big|\, X_t = x_t, X_{t-1} = x_{t-1}, \ldots, X_0 = x_0\big) \;=\; \Pr\big(X_{t+1} = x_{t+1} \,\big|\, X_t = x_t\big).$$

Equivalently, the one-step transition probabilities are given by a kernel $T(x \to x') = \Pr(X_{t+1} = x' \mid X_t = x)$.

Statistical correctness of the Metropolis algorithm is guaranteed by the *detailed balance condition* (as well as other more subtle, but usually obeyed, properties such as irreducibility and aperiodicity). Detailed balance is physically equivalent to dynamical equillibrium whereby in any region of a large collection of particles, the probability flux of a particle leaving the region, multiplied by the local particle density, is equal to the sum of fluxes entering from its neighbours such that particle density remains everywhere constant. Formally this is expressed as follows.

Detailed Balance. A Markov chain with transition kernel $T(x \to x')$ satisfies the *detailed balance* condition with respect to a target density $\pi(x)$ if

$$\pi(x)\,T(x \to x') \;=\; \pi(x')\,T(x' \to x) \qquad \forall\, x, x'.$$

Let us show that Metropolis-Hastings obeys detailed balance. We denote by $q(x' \mid x)$ the proposal density and by

$$\alpha(x \to x') = \min\Big(1,\; \frac{\pi(x')\,q(x \mid x')}{\pi(x)\,q(x' \mid x)}\Big)$$

the Metropolis-Hastings acceptance probability. The overall transition kernel is

$$T(x \to x') = q(x' \mid x)\,\alpha(x \to x') \quad \text{for } x' \neq x,$$

Without loss of generality assume $\pi(x)\,q(x' \mid x) \leq \pi(x')\,q(x \mid x')$. Then as follows:

$$\begin{aligned}
\pi(x)\,T(x \to x') &= \pi(x)\,q(x' \mid x)\,\alpha(x \to x') \\
&= \pi(x)\;\min\big\{q(x' \mid x),\; \tfrac{\pi(x')q(x|x')}{\pi(x)}\big\} \\
&= \min\{\pi(x)\,q(x' \mid x),\; \pi(x')\,q(x \mid x')\} \\
&= \pi(x')\,q(x \mid x')\;\min\big\{1,\; \tfrac{\pi(x)\,q(x'|x)}{\pi(x')\,q(x|x')}\big\} \\
&= \pi(x')\,q(x \mid x')\,\alpha(x' \to x) \;=\; \pi(x')\,T(x' \to x).
\end{aligned}$$

2.1.2 HMC as a Metropolis-Hastings Sampler

Algorithm 2 Hamiltonian Monte Carlo

Require: Current position $q^{(t)}$, potential $U(q)$, mass matrix M, Hamiltonian $H(q,p) = U(q) + \frac{1}{2}\,p^T M^{-1} p$, simulation time τ

1: **Momentum refresh:** Sample a momentum

$$p^{(t)} \;\sim\; \mathcal{N}(0, M).$$

2: **Deterministic proposal:** Evolve the state according to Hamilton's equations of motion

$$\frac{dq}{dt} \;=\; \frac{\partial H}{\partial p}, \qquad \frac{dp}{dt} \;=\; -\,\frac{\partial H}{\partial q},$$

for time τ to obtain (q', p').

3: **Acceptance probability:**

$$\alpha \;=\; \min\Big(1,\; \exp\big(-H(q',p') + H(q^{(t)},p^{(t)})\big)\Big).$$

4: Draw $u \sim \text{Uniform}(0,1)$.
5: **if** $u < \alpha$ **then**
6: Accept: $q^{(t+1)} \leftarrow q'$
7: **else**
8: Reject: $q^{(t+1)} \leftarrow q^{(t)}$
9: **end if**

Starting from the above HMC acceptance probability

$$\alpha \;=\; \min\big(1,\; \exp[-H(q',p') + H(q,p)]\big),$$

with

$$H(q,p) = U(q) + K(p), \quad U(q) = -\log \pi(q), \quad K(p) = \tfrac{1}{2}\,p^T M^{-1} p,$$

we can factor the exponential as

$$\exp[-H(q',p') + H(q,p)] = \frac{e^{-U(q')}e^{-K(p')}}{e^{-U(q)}e^{-K(p)}} = \frac{\pi(q')\,\exp\!\big(-\frac{1}{2}\,p'^T M^{-1} p'\big)}{\pi(q)\,\exp\!\big(-\frac{1}{2}\,p^T M^{-1} p\big)}.$$

Hence the acceptance ratio can be written in product form as

$$\alpha = \min\!\Big(1,\ \frac{\pi(q')\,\mathcal{N}(p' \mid 0, M)}{\pi(q)\,\mathcal{N}(p \mid 0, M)}\Big),$$

where $\mathcal{N}(p \mid 0, M) \propto \exp(-K(p))$ is the Gaussian density of the momentum which fulfills the role of a proposal density.

> **Nota Bene:** there are some important subtleties in how our momentum draws act as a proposal for position. Understanding this is crucial to the correctness of our novel *Bouncing* Hamiltonian Monte Carlo algorithm and will be addressed when discussing this.

2.1.3 HMC Viewed as a Normalising Flow

The effectiveness of HMC is often presented in terms of exploration of the "typical set" of a probability distribution (e.g. Betancourt (2017)). Whilst this is in practice the reason that HMC is efficient (loosely meaning that it is able to rapidly generate a set of samples representative of the target distribution), it does not give a full picture of why HMC works.

The "typical set" of a probability distribution is the set of states which dominates any expectation values. It arises because most practical distributions have modes in probability density (as a function of values) which smoothly decay. Since the probability density only takes on high values for a small volume around the mode, it is the lower values surrounding the mode and occupying a much greater volume that dominate. The effect becomes exaggerated with increasing dimensions. This is much the same way that the radial distribution of electron density around an atom is shifted outward due to increasingly large volume shells as the radius increases. However, the multidimensional uniform distribution with finite support is a counter example which has no typical set. Here all non-zero values contribute equally.

Betancourt (2017) explains the motion of HMC as being drawn into the typical set. Whilst this is true in practice, as most practical distributions possess a typical set that a correct sampler must necessarily take the majority of its samples from, it is not a natural way to think of the HMC motion. **Instead HMC should be viewed as a normalising flow.**

> **Normalizing flows** rely on finding an invertible, differentiable map $\Phi\colon \mathcal{Z} \to \mathcal{X}$ that sends latent samples $z \in \mathcal{Z}$ from a simple base distribution to observations $x = \Phi(z) \in \mathcal{X}$, rearranging the base density into the target distribution by accounting for the volume change under Φ.

Because Hamilton's equations of motion conserve the Hamilonian H, the Metropolis-Hastings ratio α will always be one (ignoring discretisation error). The interpretation is that at every position, the Hamiltonian update function $\Phi_\tau(q,p) = (q',p')$ is a flow map which takes momentum draws from a gaussian into position draws in our target distribution (after marginalisation). Ignoring discretisation errors, this is a perfect mapping and the covariance between any two (even adjacent) steps in our Markov chain will be zero. The intuition for this is that the target probability acts as a potential landscape. If step t the state is far away from the mode, the landscape will be flat and sloping towards the mode: thus a little momentum goes a long way and the gaussian tails allow state $t+1$ to reach any point with appropriate probability. Likewise if at step t the state is near the mode, the gaussian tails in the momentum draw still allow the $t+1$ state to reach any point but it is unlikely to reach far away from the mode as the particle is sitting inside a potential well. This is analogous to normal coordinates in general relativity, which map the tangent space into the manifold.

2.2 What is Bouncing Hamiltonian Monte Carlo?

Bouncing Hamiltonian Monte Carlo allows us to apply the HMC algorithm to target probabilities with hard constraints (ie boundaries at which the probability discontinuously drops to 0). If we interpret vanilla Hamiltonian Monte Carlo as treating the Markov chain state as a physical particle which is given momentum kicks and then evolved according to Hamiltons equations, *Bouncing* Hamiltonian Monte Carlo simply reflects the state off of any hard boundaries in the usual physical sense. That is, if we have the surface normal $\hat{n}$ to the boundary, momentum p is instantaneously mapped to $p - 2\hat{n}(\hat{n}^T \cdot p)$ whenever the particle attempts to cross the boundary. Note that when the mass matrix M used, our dot product becomes an inner product with metric M^{-1}

2.3 Proof of Correctness

In practice, the Hamiltonian update step is performed by a Stormer–Verlet integrator with finite step size ϵ for a number of steps n.

Algorithm 3 Stormer–Verlet Integrator for HMC

Require: Initial state (q, p), potential $U(q)$, mass matrix M, step size ϵ, steps n

1: **for** $i = 1$ **to** n **do**
2: $\quad p \leftarrow p - \frac{\epsilon}{2} \nabla U(q)$ ▷ $\partial H/\partial q = \nabla U(q)$
3: $\quad q \leftarrow q + \epsilon M^{-1} p$ ▷ $\partial H/\partial p = M^{-1} p$
4: $\quad p \leftarrow p - \frac{\epsilon}{2} \nabla U(q)$
5: **end for**
6: **return** (q, p)

Whilst there is a discretisation error associated with the finite step size, so that $H(p, q) \neq H(p', q')$, this is counteracted by the Metropolis-Hastings ratio so that this does not break detailed balance. However, there are two key properties of the Stormer-Verlet integrator that are also necessary to ensure detailed balance is obeyed. Namely it is both volume-preserving and reversible: Steps 2, 3, and 4 of the algorithm are skew maps which trivially conserve volume. If we label the input and output of a single Stormer-Verlet integration iteration as (q, p) mapped to (q', p') then one can readily show that $(q', -p')$ is mapped to exactly $(q, -p)$. See figure 1 for diagram of this. If we instead want to ensure that (q', p') maps to the original (q, p) then we add a final step to the integrator which reverses the sign of p. This is the reversibility condition. In practice the Hamiltonian H is invariant under a sign change of momentum and it is immediately discarded after calculating the acceptance ratio so there is little need to bother actually reversing the sign.

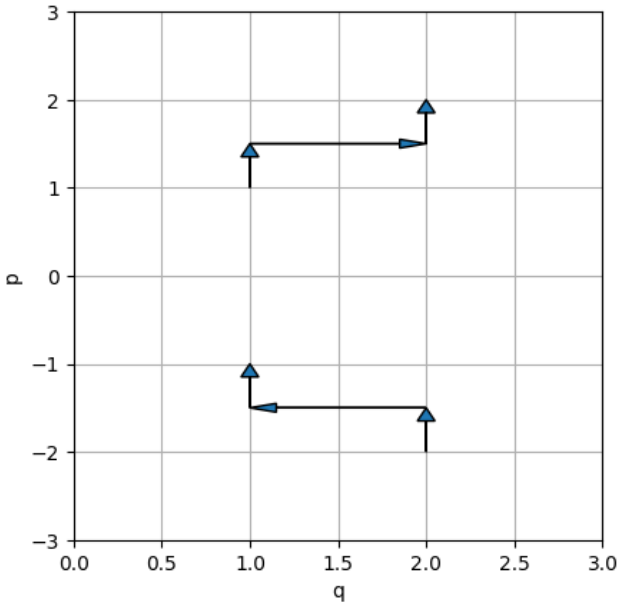

Figure 1: Here we see a pair of Stormer-Verlet integration iterations from (q, p) to (q', p') and $(q', -p')$ to $(q, -p)$

It is not immediately obvious that *volume conservation* of the joint probability space in position and momentum is necessary for correctness. Indeed following the approach of Betancourt (2017) and considering a collection of momentum proposals as promoting a position q to an m-dimensional surface with fixed q in the $2m$-dimensional joint space (where the position and momentum states both have dimension m) one is tempted to investigate correctness in terms of conservation of area as the Hamiltonian update flows that surface into one which is a mixed collection of q' and p'. See figure 2 for a sketch of this surface mapping.

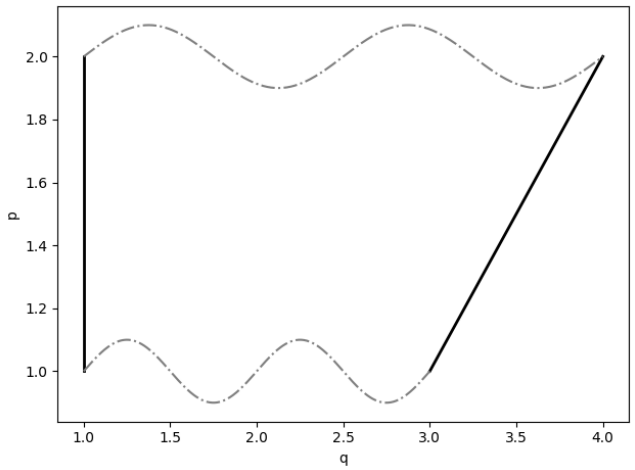

Figure 2: Here we consider the initial collection of possible momentum proposals at $q = 1.0$ become mapped under a Hamiltonian update (the dashed grey lines) to a collection of proposal points along a surface (in this low dimensional case, a line) which is mixed in q and p. One might be tempted to investigate the correctness of a HMC algorithm in terms of "surface area" conservation.

Mohasel Afshar and Domke (2015) show that it *is* in fact volume conservation of the joint probability space which is necessary to ensure correctness. In a later paper citepAfshar2021 it is shown that when the update map is not volume conserving, this can be counteracted by including the corresponding Jacobian factor into the Metropolis-Hastings acceptance ratio so that detailed balance is still obeyed.

Now we investigate whether our "bouncing" map is also volume conserving. We follow Skilling

(2019) and perform any reflections "on the spot". That is, if any Stormer-Verlet iteration would take our particle/sample through a boundary, we instead reflect the momentum without updating position, as if the boundary were at the location the particle currently is. See figure 3 for a sketch of bouncing on the spot. The magnitude of the momentum is not changed under bouncing so that the Hamiltonian is conserved (keeping the acceptance ratio high). Bouncing on the spot does not destroy irreducibility near the boundary (irreducability requires there to be a path between any two states) - we can pick a spot arbitrarily close to the boundary and simply trace a path backwards from it into the bulk. Bouncing on the spot is in contrast to Mohasel Afshar and Domke (2015) who also propose a bouncing variant of HMC which calculates the intersection of the trajectory with the (affinely parameterised - non curving) boundary and performs a full position and momentum update.

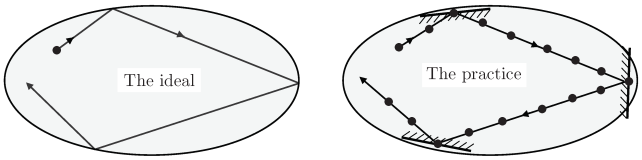

Figure 3: Taken from Skilling (2019) - this shows bouncing "on the spot" as an approximation to bouncing off the hard boundary.

If we calculate the jacobian of our "on the spot" bouncing update $q \to q$ and $p \to p - 2\hat{n}(\hat{n}.p)$, we find a matrix in block form:

$$\text{Jacobian} \;=\; \begin{pmatrix} I & 0 \\ \dfrac{\partial p_i}{\partial q'_j} & R_{ij} \end{pmatrix}$$

where

$$R_{ij} \;=\; \delta_{ij} \;-\; \hat{n}_i\,\hat{n}_j$$

The volume change is given by the determinant of the jacobian. Due to the zero in the upper right block, it becomes the product $\det I \times \det R$. Since determinants are invariant under rotations, we can choose to rotate R such that the unit vector n is aligned with one axis. Since I is isotropic, we see that $\det R$ is simply -1 so that the determinant of our whole Jacobian is $1 \times -1 = -1$ and it is volume preserving. The physical interpretation is that on-the-spot bouncing is also a skew map! See figure 4 for a 2-dimensional analogue of our on-the-spot bouncing map as a skew.

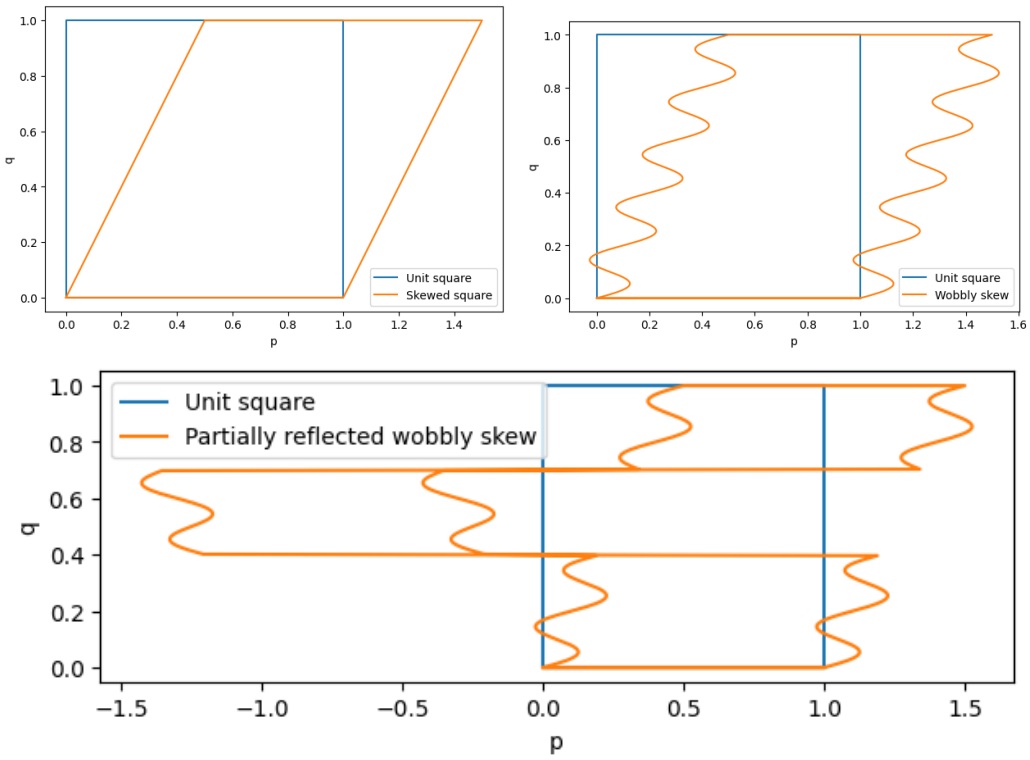

Figure 4: Here we see three examples of skews, which are all volume preserving by nature.

2.4 Empirical Correctness

As a crude empirical test of correctness, we compare 12_000 sample draws from the Python library numpy, with any falling outside the boundary discarded, against 10_000 draws from our bouncing HMC algorithm. Visually the cumulative distribution function of the marginalised Y-variable from each set appears identical (see figure 7) and this is quantified by the KS-test giving a p-value of 0.957 suggesting our algorithm is correct. A plot of the gaussian draws from numpy is shown in figure 5 and a plot of the bouncing HMC draws is shown in figure 6. We chose a curved boundary centred at the mode of the underlying distribution as this was most likely to show any bias introduced by the boundary. The boundary curve can be seen in figure 8, along with the reflecting trajectories of our samples.

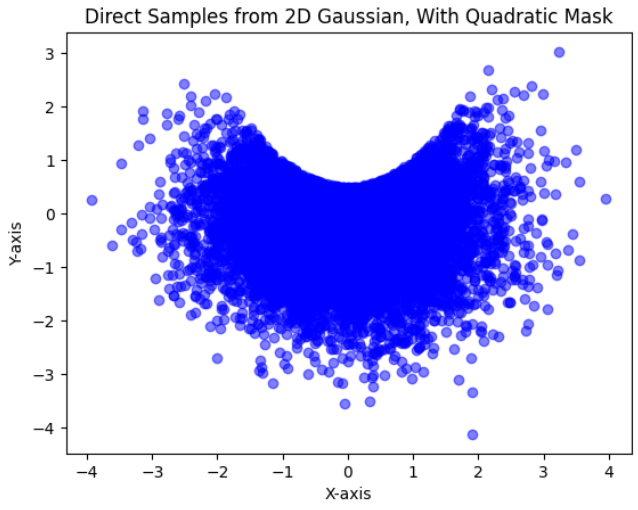

Figure 5: Direct gaussian draws from the numpy library, with samples outside the boundary discarded.

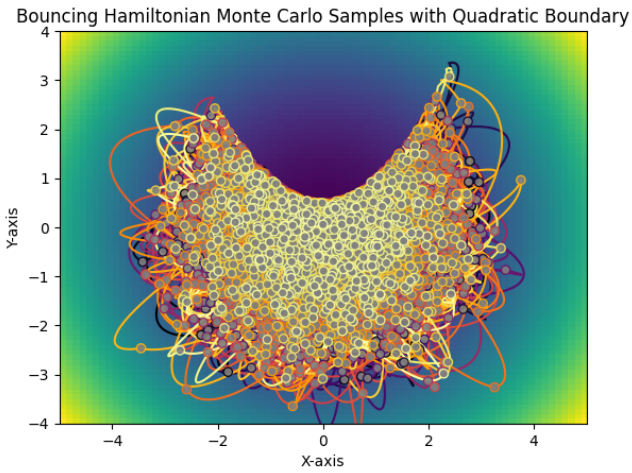

Figure 6: Samples drawn using our bouncing HMC algorithm. The lines show the Hamiltonian update steps between each successive sample. Darker colours were draw early in the chain and lighter colours drawn later. Underneath we display the probability density function of the underlying gaussian distribution.

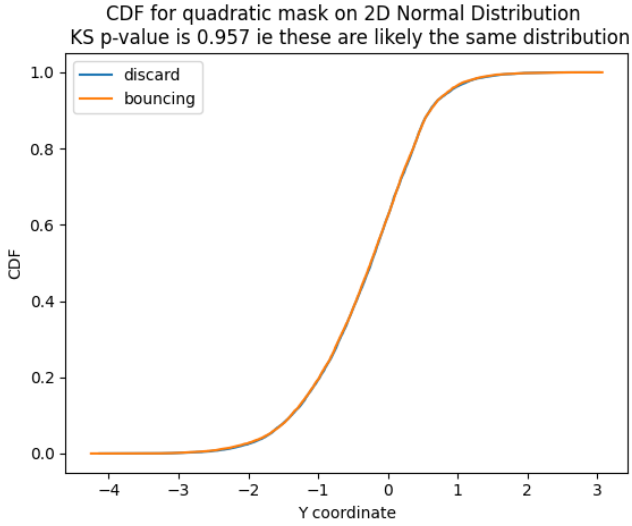

Figure 7: We plot the cumulative density function of one of the sample variables for both the direct draws using discard for the boundary and for the bouncing algorithm. This is the basis of the KS-test which quantifies how likely two sets of samples are to have been sampled from the same distribution.

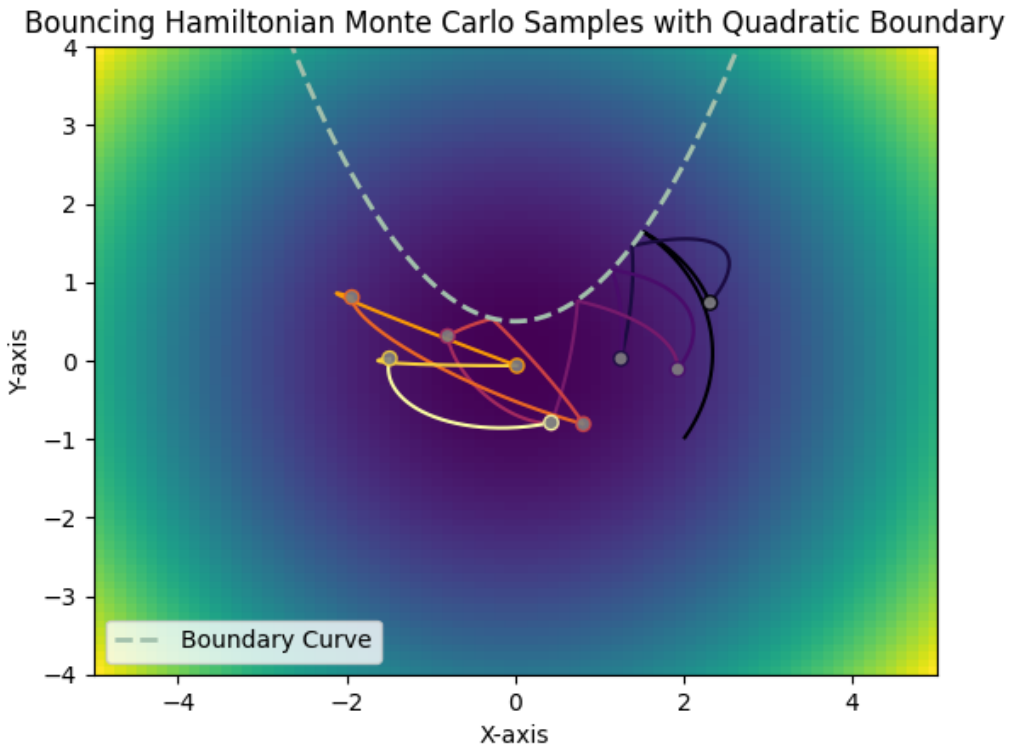

Figure 8: For clarity, here is the boundary explicitly drawn in the light-blue dashed line. We can see the bouncing HMC trajectories reflecting off of it. Samples are represented by the open circles, trajectories by the trailing lines.

3 Relevance to Astronomy

3.1 Why Sample?

Physics is fitting models to data (and ideally making quantifiable predictions of future data). It is usually straightforward to calculate the *likelihood* of observing a set of data, assuming a given model. That is $P(D|\theta)$.

However, the quantity we really care about is the *posterior* $P(\theta|D)$. Which is a direct measure of how "good" the model is given data we have actually observed already. We can use Bayes' theorem to relate the two:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \tag{1}$$

We refer to $P(\theta)$ as the *prior* as it comes from any pre-existing beliefs we have about the likelihood of the candidate models. This is often taken to be uniform in lieu of any more reasoned judgment. The denominator $P(D)$ is referred to as the *evidence* and may be calculated from the likelihood and prior via the law of total probability/marginalisation:

$$P(D) = \int P(D|\theta)P(\theta)\, d\theta \tag{2}$$

Here we provide a sketch of such a situation. Consider that we are observing a supernova, which is of interest as a "standard candle". One measurement we might take is its brightness over time. See figure 9.

We can't make truly continuous measurement, and instead must take a discrete number of observations over time. Each of these has some uncertainty associated, in part due to stochastic processes in the supernova and in part due to error in our measuring devices. We appeal to the central limit theorem and assume that the total uncertainty σ_t on each measurement follows a Gaussian distribution. For simplicity and depending on our goals, we might also assume that our knowledge of the physics of

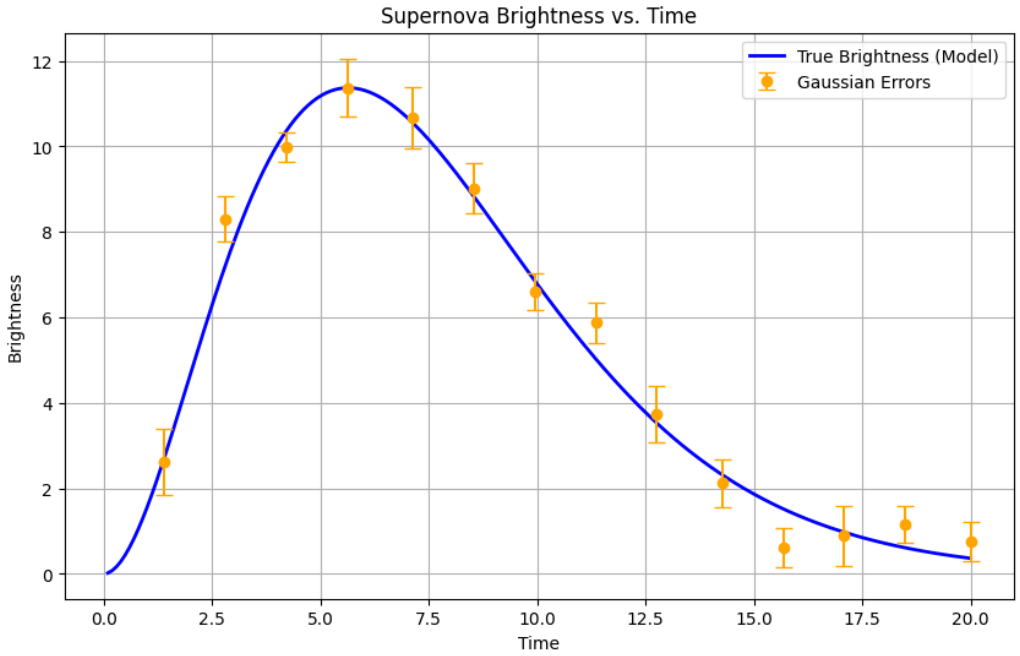

Figure 9: A rough sketch of supernova brightness over time, with errors. Data is fictitious.

supernovae is correct and that the unknowns are simply some set of parameters (e.g. progenitor mass, temperature, etc.) which we shall label with the vector $\boldsymbol{\theta}$ (with one component for each parameter).

Thus we have a set of observational data D_t for the brightness at different times t and can calculate expected brightness $I_t(\boldsymbol{\theta}_i)$ as a function of parameters for each time t. Making the Gaussian approximation (valid by central limit theorem) we can calculate the likelihood as

$$P(D|\theta) = \text{normalisation} \times \prod_t \exp\left(\frac{(D_t - I_t(\boldsymbol{\theta}))^2}{2\sigma_t^2}\right) \tag{3}$$

We can then plug this into Bayes' theorem to calculate the posterior and judge how probable any set of parameters might be, given the data we have observed. If we calculate the evidence via marginalisation, then we do not need to know the normalisation factor as it will cancel.

The "obvious" thing to do would be to incrementally adjust the parameters (via gradient descent) until we find a maximum in the posterior and declare that this is our best guess for the properties of the supernova progenitor. In this case, we would not need to evaluate the normalisation nor the evidence — which is good because this is very difficult!

The drawback of the gradient descent approach is that it does not give us any information about the uncertainty on our parameter estimation (it might also fail to hit the global maximum). This is where sampling becomes advantageous. In general $I(\boldsymbol{\theta})$ will be some complicated function that does not have an analytic integral—so we cannot directly calculate the evidence nor higher order moments such as variance.

If we can obtain a (large) sample of points from our probability distribution, then we can appeal to the law of large numbers and approximate any property of the distribution. E.g., as the number of samples becomes large, the sample mean and variance converge to the mean and variance of the underlying distribution.

A great deal of astrophysical research can be summarised as follows.

Astrophysics

1. Define a forward model.
2. Define a likelihood which compares observed data to the forward model.
3. Perform statistical inference.

Sampling is a robust and efficient way to carry out the 3rd step of astrophysics.

3.2 Why Bouncing HMC?

Prior knowledge of physical problems is a natural source of hard boundaries on the posterior - this can be intrinsic such as flux being greater than zero, or based on physical insight such as the mass of a star which must be sufficiently massive for fusion to occur but less massive than the Eddington limit. Indeed, the most widely used statistical packages in astronomy all make some attempt to implement hard boundary conditions. However this is not yet done well.

For example the MultiNest algorithm (Feroz et al., 2009) is used in many cosmological packages. It uses a complicated arrangement of ellipsoids as an attempt to approximate hard boundaries. PyMC3 offers the "Bound" method which automatically reparameterises the variable into an unbounded one (Stan implements bounds similarly) - the issue with this is that if the posterior is concentrated near the bound, this is sent to plus or minus infinity and any sampling will diverge.

Hard constraints are often dealt with in a bespoke way: for example the SALT model for type Ia Supernova light curves places the colour law term inside an exponential $Flux \propto \exp(-cCL(\lambda))$ so that positive colour (reddening eg from dust) dims the flux but can never make it negative (negative colour brightens the flux according to the empirical intrinsic bluer-brighter law).

3.3 Gravitational Wave Inference

Since the project title describes supernovae, the author spent a good deal of time attempting to understand various light-curve models, including terms such as K-corrections which account for the fact that colour filters have a broad transmission function and this needs standardising after relativistic frequency shifts, in the hope of finding examples of unconstrained hard boundaries. This turned out to be fruitless.

Instead a more striking example of a hard boundary problem is in gravitational wave inference. Specifically we investigate binary black hole mergers, since these are the most well studied sources of gravitational waves.

3.4 Binary Black Hole Mergers

Binary black hole mergers observed via gravitational waves consist of three main phases:

1. **Inspiral:** Quasi-circular orbital decay described by post-Newtonian expansions.
2. **Merger:** Highly non-linear plunge and coalescence into a common horizon.
3. **Ringdown:** Damped quasi-normal mode oscillations of the remnant black hole.

Inferred Parameters

- *Mass parameters:*

$$\begin{aligned} & m_1,\, m_2 && \text{(component masses)} \\ & \mathcal{M} = \frac{(m_1 m_2)^{3/5}}{(m_1+m_2)^{1/5}} && \text{(chirp mass)} \\ & M = m_1 + m_2 && \text{(total mass)} \\ & q = \frac{m_2}{m_1} \leq 1 && \text{(mass ratio)} \end{aligned}$$

- *Spin parameters:*

$$\begin{aligned} & \vec{\chi}_i = \frac{\vec{S}_i}{m_i^2}, \quad \chi_i = |\vec{\chi}_i| && \text{(dimensionless spin vectors)} \\ & \chi_{\text{eff}} = \frac{m_1\chi_1\cos\theta_1 + m_2\chi_2\cos\theta_2}{M} && \text{(effective aligned spin)} \\ & \chi_p && \text{(effective precession spin)} \end{aligned}$$

- *Extrinsic (geometric) parameters:*

$$D_L,\ (\alpha,\delta),\ \iota,\ \psi,\ t_c,\ \phi_c$$

where

  - D_L = luminosity distance
  - (α, δ) = right ascension & declination
  - ι = orbital inclination
  - ψ = polarization angle
  - t_c, ϕ_c = coalescence time & phase

Typical Parameter Bounds for Binary Black Hole Mergers

Masses

$$0 < m_2 \leq m_1, \quad m_1 + m_2 \lesssim 200\, M_\odot$$

Chirp Mass and Mass Ratio

$$0 < \mathcal{M} = \frac{(m_1\, m_2)^{3/5}}{\big(m_1 + m_2\big)^{1/5}} \leq \frac{M}{2^{1/5}}, \quad 0 < q = \frac{m_2}{m_1} \leq 1$$

Dimensionless Spins

$$-1 \leq \chi_i = \frac{|\vec{S}_i|}{m_i^2} \leq 1, \quad i = 1, 2; \quad -1 \leq \chi_{\text{eff}} \leq 1, \quad 0 \leq \chi_p \leq 1$$

Extrinsic Parameters

$$0 \leq \iota \leq \pi, \quad 0 \leq \psi < 2\pi, \quad 0 \leq \alpha < 2\pi, \quad -\tfrac{\pi}{2} \leq \delta \leq \tfrac{\pi}{2},$$
$$D_L > 0, \quad z \geq 0, \quad t_c \in \mathbb{R}, \quad 0 \leq \phi_c < 2\pi$$

As is apparent, there is a high number of hard boundary constraints, making gravitational wave inference a striking example of the usefulness of Bouncing HMC. In particular the mass ratio $q := \frac{m_2}{m_1}$, asserted to be ≤ 1 to break an ambiguity in which black hole is which, is expected to lay close to the boundary - ie we expect binary black holes to have a similar mass. This would cause samplers using reparamterisation to diverge.

3.5 JAX and jim

Full forward modeling by integration of the einstein equations is painfully slow. However there is an impetus to speed this up allow electromagnetic (EM) follow-up immediately after detection of gravitational waves from sources such as binary neutron star mergers. Gravitational waves travel at the speed of light but they precede detection of EM radiation as the processes that generate substantial radiation occur after the merger. Two ways of accelerating inference are to use neural network approaches to "learn" the einstein equations or to use phenomenological models.

One such phenomenological model is IMRPhenomD. This is written essentially as a polynomial in frequency (which is how gravitational waves are analysed) with coefficients composed of general, physically motivated functions of the physical parameters. These functions are then tuned using the output of many numerical integrations of the full Einstein equations. An additional speedup is gained from the use of heterodyning which allows us to approximate our likelihood inner products using a nearby reference inner product. jimGW is a Python package implementing IMRPhenmD using the JAX library. This gives further speed up as JAX traces complex composed operations and compiles them into highly optimised linear algebra operations that can be readily executed by a GPU. Crucially, JAX compatible code also allows us to obtain gradients via automatic differentiation - when JAX traces the composed operations it can also construct the differentials associated with each composition. Since HMC relies on gradients, this is essential for the viability of the sampler.

4 Results

The sampler was implemented by wrapping the existing implementation of HMC within the Blackjax library. With some judicious hand-tuning of the mass-matrix and choosing a starting point near the known mode of GW150914's parameters, 10_000 samples were generated taking about an hour on an RTX A2000 laptop GPU. A full corner plot is given in figure 10.

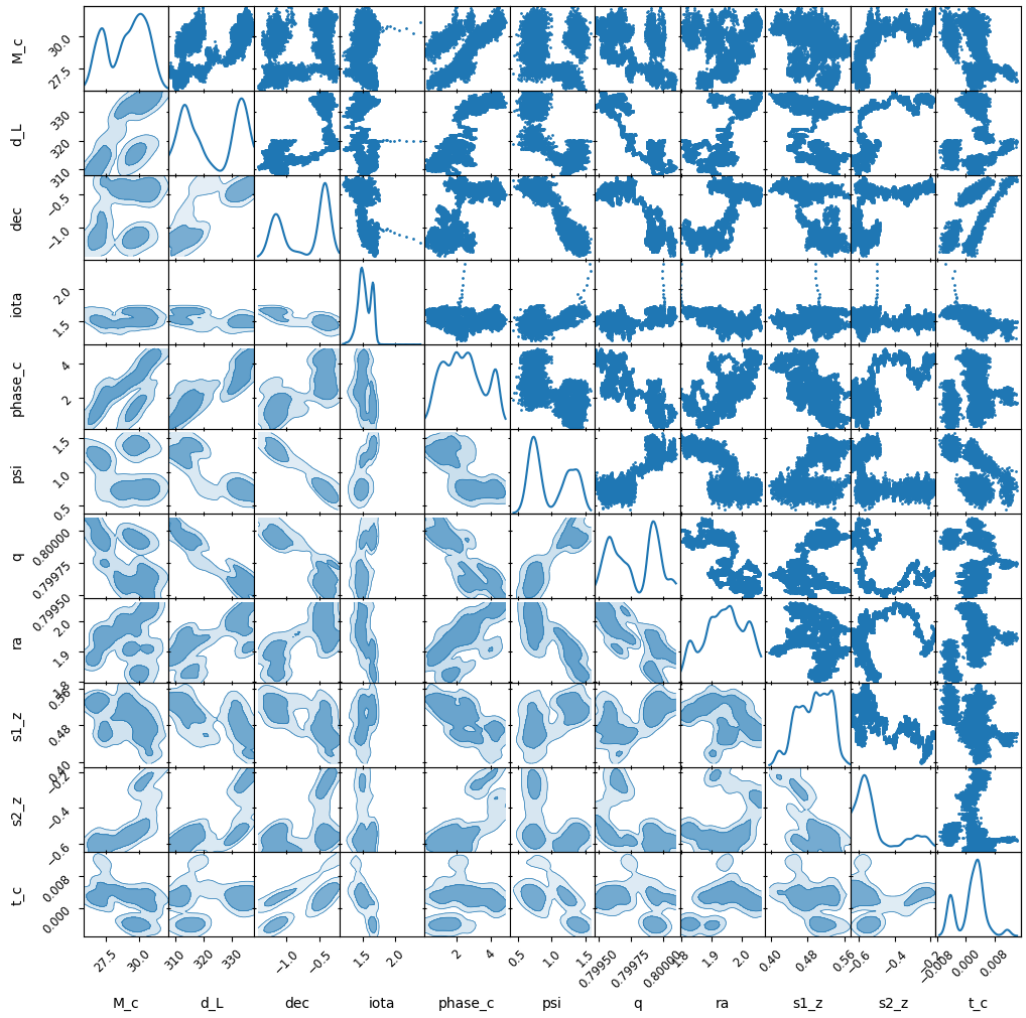

Figure 10: A triangle plot of our samples after 10_000 draws. Burn-in tails are clearly visible as they were not removed - observing the trace plots we see that we essentially never left the burn-in phase through the whole 10_000 samples.

Despite manually tuning the mass matrix by eye, and starting the chain near the "correct" values for parameters, we see from the trace plots (figs. 11 - 14) that the mass matrix chosen was not optimal and prevented the chain from mixing fully for the full 10_000 samples. This is noticible in the trace of the luminosity distance. We also observe other unfortunate behaviour such as a random walk in the right ascension, again cause by too small an inverse mass matrix entry. A much better approach would be to implement an adaptive tuning phase for the mass matrix.

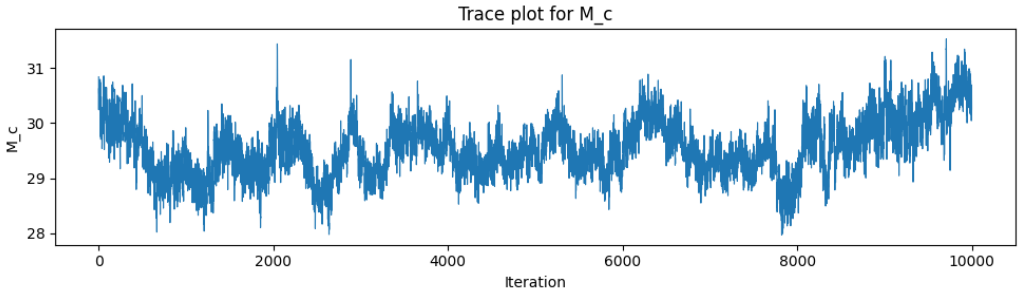

Figure 11: This is what the trace of a well-mixed variable should look like.

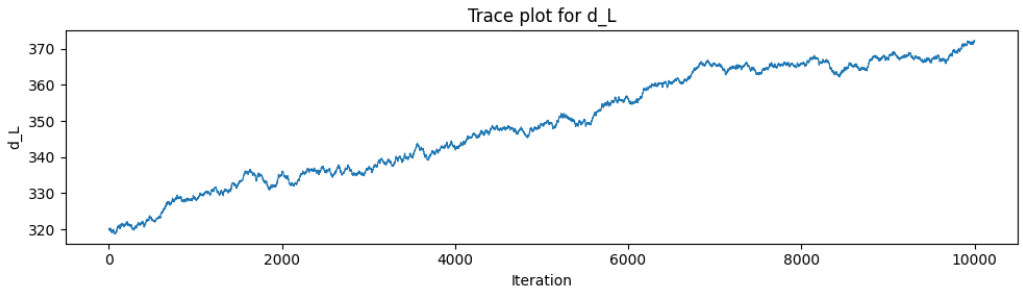

Figure 12: Here we see a clear drift in the luminosity distance. This tells us that we suffer from some combination of starting at a value too far away from the max posterior, we did not sample for long enough to escape the burn-in period or we did not use a high enough value for the inverse mass matrix.

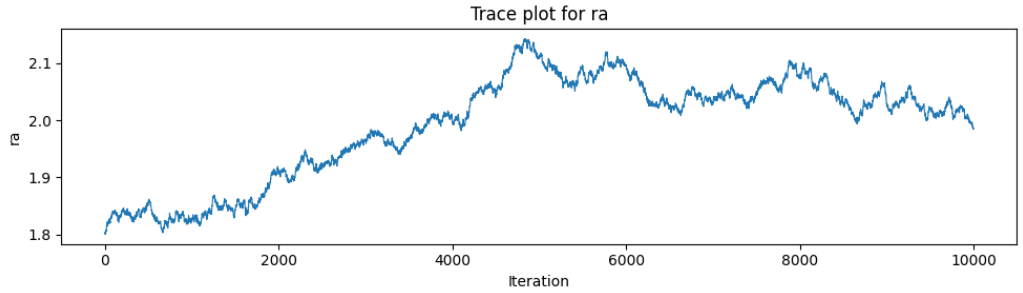

Figure 13: Here we see that right ascension exhibits random-walk like behaviour. This tells us that the corresponding component of the inverse mass matrix was far too low and we are not able to fully orbit the typical set.

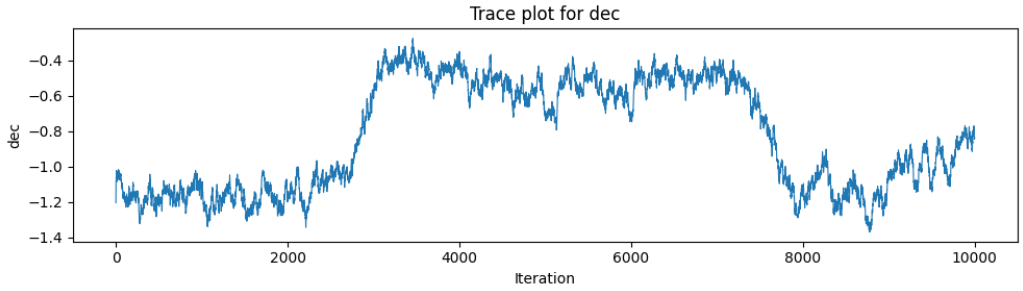

Figure 14: Here we see that the declination is well mixed and multimodal.

A collection of 5 total chains were run but statistics such as the Gelman-Rubin ratio will be misleading on account of the fact that some of the variables are still within the burn-in phase. Indeed the calculated Effective Sample Size (ESS) of the chain displayed in the trace plots is only about 10 samples, despite 10_000 having been drawn.

5 Conclusion and Further Work

We have developed a novel variant of the HMC sampling algorithm and proved it obeys theoretical correctness criteria. We have implemented it using a JAX based framework which compiles the total markov chain (both the sampling algorithm and the posterior combined) into GPU-optimised linear algebra operations. Unfortunately real world distributions are messy and require tuning of the mass matrix to ensure fast convergence.

With the aim to develop a truly generic sampler, which does not require (potentially biased) human tuning, future development of the Bouncing HMC algorithm should include an adaptive mass matrix

which is automatically tuned during the burn in phase.

References

Michael Betancourt. A conceptual introduction to hamiltonian monte carlo. 2017.

Simon Duane, A D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Phys. Lett. B*, 195(2):216–222, September 1987.

F Feroz, M P Hobson, and M Bridges. MultiNest: an efficient and robust bayesian inference tool for cosmology and particle physics. *Mon. Not. R. Astron. Soc.*, 398(4):1601–1614, October 2009.

Hadi Mohasel Afshar and Justin Domke. Reflection, refraction, and hamiltonian monte carlo. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL `https://proceedings.neurips.cc/paper`$_{f}iles/paper/2015/file/8303a79b1e19a194f1875981be5bdb6f-Paper.pdf$.

John Skilling. Galilean and hamiltonian monte carlo. *Proc. (MPDI)*, 33(1):19, December 2019.

A Statement on AI use

Various generative AI tools have been used to produce "boilerplate" code such as for plotting figures and formatting LaTeXetc. It has also been used extensively to perform literature searches. However I confirm that all analysis is my own original work, unless otherwise referenced.

B Full Reflection off Curved Boundaries is also Volume Conserving

This section has temporarily been removed for correction.