Phylogenetic Inference using Hamiltonian Monte Carlo

Matthew Bahram Edmund Robinson

Thesis submitted for the degree of
Doctor of Philosophy

School of Mathematics & Statistics
Newcastle University
Newcastle upon Tyne
United Kingdom

July 2020

I dedicate this to the memory of Prof. Richard Boys whose warmth and kindness helped make me feel at home in Newcastle and to my parents who have always supported me.

Acknowledgements

I would like to thank Prof Richard Boys, Dr Tom Nye and Dr Andrew Golightly for taking the time to muddle their way through the numerous incohorent statements I have made and providing a wonderful work environment and plenty of emotional support. I would like to thank Prof. Nick Parker and the Maths and Stats office for looking after me and always being willing to lend an ear. I would also like to thank many of the members of PhD 3, Joe Matthews, David Robertson, David Pescot, Thomas Bland, Thomas Lowe and Stephen Johnson for the many instances in which they listened to my coding woes, watched me scribble on the whiteboard and the helpful advice they have given me. Finally I would like to thank my parents for their emotional support throughout the course of this PhD. I would also like to thank the School of Mathematics & Statistics for the wonderful setting they provide for conducting research and the I.T. support Dr Michael Beaty, John Nicholson and Dr George Stagg. Finally I would like to thank EPSRC for providing the funding for this research.

Abstract

Phylogenetics is the study of evolutionary structure, aiming to reconstruct the branching structure of speciation from a common ancestor. There are many methods of infering the tree-like structure from the most basic, physical traits (morphology) to analysing the distances between genetic code based on a predefined metric. For viruses such a method is the best way to access their hereditity. Bayesian inference enables us to learn a region of possible trees and alter the distribution of trees according to prior beliefs. The most common method of conducting Bayesian inference over evolutionary trees, called Tree space (Billera et al., 2001), is by Markov Chain Monte Carlo (MCMC). Tree space is big and exploration is slow; a modern technique for speeding up MCMC is Hamiltonian Monte Carlo (HMC), developed by Duane et al. (1987). We incorporate HMC into Tree space by creating our own algorithm: Cross-Orthant HMC (COrtHMC). Many methods of increasing HMC convergence speed have been developed, such as Riemannian Manifold HMC (RM-HMC) (Girolami et al., 2011). Where applicable, we adapted such methods to COrtHMC and then compared COrtHMC to pre-existing methods of phylogenetic inference and probabilistic path HMC (Dinh et al., 2017). We found that all forms of COrtHMC perform similarly, including ppHMC, but that the increased computational cost in using such HMC methods outweighs any benefit.

Contents

I Background Material **1**

1 Introduction **2**

1.1 A Brief History of Phylogenetics 4
1.2 DNA and Bases 5
1.3 Motivation 6
1.3.1 Disease spread 6
1.3.2 Scientific Efficiency 7
1.3.3 Virology 7
1.4 Literature Review: Challenges in Phylogenetic Inference 7
1.4.1 MCMC 8
1.4.2 Metropolis-Coupled MCMC 8
1.4.3 Huelsenbeck and Ronquist 9
1.4.4 Probabilistic Path Hamiltonian Monte Carlo 10
1.4.5 Sequential Monte Carlo 10
1.4.6 Variational Bayesian Phylogenetic Inference 10
1.5 Contribution 11

2 Substitution Models **12**

2.1 Markov Models and Markov Processes 12
2.1.1 Chapman-Kolmogorov equation 13
2.1.2 The rate matrix 14
2.2 The substitution model 15
2.2.1 Examples of substitution models 15
2.2.2 General Time Reversible Model (GTR) 16
2.2.3 Hasegawa, Kishino and Yano 1985 (HKY85) 17
2.2.4 Substitution Models and Trees 19
2.3 The Alignment 20

2.4 Linear Site Rate Heterogeneity 21
2.5 The Markov Model and the Likelihood 22

3 Bayesian Inference and Tree space **25**
3.1 Tree space 26
3.1.1 Phylogenetic Tree 26
3.1.2 Newick String Representation 33
3.2 Bayesian Inference 35
3.3 Markov Chain Monte Carlo 35
3.4 MCMC in Tree space 39
3.4.1 Tree Operations 39
3.5 Assessing MCMC for phylogenetic inference 41
3.5.1 Effective Sample Size 41
3.5.2 An alternative Effective Sample Size 43
3.5.3 Convergence 43

4 Hamiltonian Monte Carlo **44**
4.1 Introduction 44
4.1.1 History 44
4.1.2 Concept 44
4.1.3 Requirements and uses 46
4.2 Theory 46
4.2.1 Manifolds and Notation 46
4.2.2 Introduction to Hamiltonian Dynamics 49
4.2.3 HMC 53
4.2.4 The Hamiltonian Monte Carlo algorithm targets the correct distribution . 55
4.2.5 Metropolis Adjusted Langevin Algorithm (MALA) 58
4.3 Numerical Integration of the Hamiltonian Flow 60
4.3.1 The Leapfrog Integrator 61
4.3.2 Appropriate Choice of Parameters 67
4.3.3 Riemannian Manifold Hamiltonian Monte Carlo (RMH-MC) 68

II Hamiltonian Monte Carlo on Tree-Space **70**

5 HMC on Tree space **71**
5.1 Flows across Tree space 71

5.1.1 Motivation . . . 74
5.2 Cross Orthant HMC . . . 78
5.3 COrtHMC targets the posterior and produces an ergodic chain . . . 80
5.4 Probabilistic Path HMC (ppHMC) . . . 83
5.5 Derivative of the Likelihood with respect to an Edge . . . 84
5.6 Other results about COrtHMC . . . 85
5.7 Methods for Crossing Codimensional Boundaries . . . 86
5.7.1 Transforming Edges: Reflecting vs Projecting across the Boundary . . . 87
5.7.2 Continuing the Integration . . . 90
5.7.3 Resetting Momenta . . . 90
5.7.4 Choosing a Topology . . . 91
5.8 Adapting COrtHMC . . . 92
5.8.1 RM-HMC . . . 92
5.8.2 Adapting RM-HMC to Tree Space . . . 92
5.9 Discussion of other methods . . . 94
5.9.1 Tubing . . . 94
5.9.2 Embedding . . . 95

6 HMC on Substitution Model Parameters and Site Rate Parameters 98
6.1 Computing the Derivative of the Likelihood with respect to Substitution Model Parameters . . . 98
6.2 HMC on the Simplex . . . 100
6.2.1 Cruising the Simplex . . . 101
6.2.2 The Farrow Transformation . . . 103
6.3 HMC on $\mathbb{R}_+$ . . . 110
6.4 Computing the Derivative with Respect to Site Rate Parameters . . . 110

III Results 114

7 Results 115
7.1 Data, Priors and Testing . . . 115
7.1.1 Alignments . . . 116
7.1.2 Priors . . . 117
7.1.3 Testing COrtHMC . . . 120
7.1.4 Prior Sensitivity . . . 121
7.1.5 Column Sensitivity . . . 123

7.2 Specifying the HMC component: J from section 5.2 . . . 123
7.2.1 Inverted vs Standard Leapfrog integration . . . 123
7.2.2 Step Number Sensitivity . . . 124
7.2.3 Step Size Sensitivity . . . 125
7.2.4 Different Choices of Kernel for the Momenta . . . 128
7.3 Specifying the Crossing component: C from section 5.2 . . . 129
7.3.1 Reflecting vs Projecting . . . 129
7.3.2 3 orthants vs 2 orthants . . . 130
7.3.3 Ratio vs Fixed Transitions . . . 130
7.3.4 Continuing the integration . . . 130
7.4 Comparison with MCMC . . . 131
7.5 Comparison with ppHMC . . . 133
7.6 Parallelising COrtHMC and Improving Efficiency . . . 133
7.7 Visualisations . . . 134

8 Conclusions and Future Work **136**
8.1 Conclusions . . . 136
8.2 Future Work . . . 137

A Algorithms **138**
A.1 The GTP algorithm for computing geodesics in polynomial time . . . 138
A.2 The Dual Averaging algorithm . . . 140
A.3 The various different NUTS algorithms . . . 142
A.3.1 Simplified NUTS . . . 142
A.3.2 Efficient NUTS . . . 144
A.4 Shadow HMC . . . 146

B Proofs **147**
B.0.1 Theorem 6.4 . . . 148

C Literature Review **153**
C.0.1 Yang and Rannala (1997) . . . 153
C.0.2 Altekar et al. (2004) . . . 153
C.0.3 Beast and MrBayes $(MC)^3$ . . . 154
C.0.4 Probabilistic Path Hamiltonian Monte Carlo . . . 154
C.0.5 PosetSMC . . . 155
C.0.6 Variational Bayes . . . 157

D Other HMC Methods **158**
D.1 Hamiltonian Monte Carlo without Detailed Balance 158
D.2 Kernel Hamiltonian Monte Carlo 158
D.3 Shadow HMC 160
D.4 Generalised HMC 160
D.5 Constrained HMC 161
D.5.1 The method of Neal (1994) 161
D.5.2 The method of (Betancourt, 2011) 162

E Results **163**
E.1 Primate Alignment from MrBayes 163
E.2 Results 166
E.2.1 5 taxa 166
E.2.2 16 taxa 170
E.2.3 32 taxa 172

List of Figures

1.1 David Hillis's Tree of Life 5

2.1 The General Rate Matrix 16
2.2 Example of an Alignment 21

3.1 $\mathcal{T}_5$, the Petersen graph. 26
3.2 Example of a Phylogenetic Tree for birds and reptiles, here we have a striped down version of Figure: 1.1 where we focus on only a few species to understand the structure of a Phylogenetic tree. We see that a Phylogenetic tree has a root representing a common ancestor, internal edges representing unknown instances of speciation for which we have no data, and pendant edges which connect the modern species to some unknown shared ancestor. We also see that by disconnecting an internal edge we form two subtrees, partitioning all species into a clade representing the appropriate toxonomic rank at which the edge was cut. 27
3.3 The Open Book 29
3.4 The two differenct strata of a cone of a manifold as described in example 4.1. We have the red stratum at 0 corresponding to M_1 and the rest of the cone corresponding to M_2. It from this we can see that the two partition the cone manifold CM into two strata. 32
3.5 $\mathcal{T}_5$ as a stratified manifold. 33
3.6 The alignment generating tree for 5 taxa. 34
3.7 Tree Operations 40

4.1 Here we can see a maninfold, the surface is curved but still has a tangent plan and a cotangent which can be applied to the tangent plane, here the sphere is the parameter space, the velocity is the tangent and the momenta, the cotangent. . . 48
4.2 The Hamiltonian of a Pendulum for small z. Here the same energy state is depicted but with two difference momenta and heights. 51

4.3 The HMC Algorithm 55
4.4 Euler vs Leapfrog 61
4.5 The leapfrog update 62
4.6 Evolution of the Leapfrog Integrator 63
4.7 Leapfrog vs Inverted Leapfrog (large) 65
4.8 Leapfrog vs Inverted Leapfrog (small) 65

5.1 The 3-spider 72
5.2 CotangentSpace 73
5.3 Flow between orthants 77
5.4 NNI transformations 87
5.5 Reflecting and Projecting Crossing 88
5.6 Crossing two orthants in a single step 89
5.7 Unordered and Ordered projection 90
5.8 Parachuting 91
5.9 Tubing 95
5.10 Embedding 96
5.11 Discontinuity 97

6.1 Constrained HMC 102

7.1 The alignment generating tree for 5 taxa. 116
7.2 The alignment generating tree for 16 taxa. 117
7.3 The alignment generating tree for 32 taxa. 117
7.4 Trace plots of the log-posterior showing different numbers of topological crossings when using a Dirichlet distribution with $\alpha = 0.01$. The first figure changes topology once, the second with $\alpha = 1$ often changes topology as can be seen by the various different colours. (We used $\mathcal{A}_5^{1000}$ to generate these trace plots.) 119
7.5 Plots for each parameter. Left: Trace plot depicted in blue, Middle: Autocorrelation in red, Right: Density in blue and red. 121
7.6 Informative vs Improper Priors. Here we see the different mixing of π_A using $\mathcal{A}_5^{100}$. The informative prior is on the left and exhibits poorer mixing than the improper prior on the right, both explore multiple topologies but the graph of the informative prior spends more time in the correct topology, as can be seen by the high percentage of black. The colours denote the topological distance from the tree used to generate $\mathcal{A}_5^{100}$, Fig: 7.1. 122

7.7 Informative vs Improper Priors. The improper prior is depicted on the left anad informative on the right. We can see that an informative prior acts to return the transition-transvertion ratio to a more confined region whereas and improper prior can cause extreme exploration around less informative areas. The informative prior effectively explores the prior for the gamma rate heterogeneity without exploring region about $\alpha = 0.5$ whereas the improper prior provides better exploration. . . 122
7.8 Two plots of the log-posterior for differing alignment lengths. On the left plot was generated from $\mathcal{A}_5^{1000}$ and the right the alignment has length $\mathcal{A}_5^{100}$. The more informative alignment focusses the chain to within a single orthant and provides a more informed path. 123
7.9 Comparison of the effective sample size for Inverted vs Standard leapfrog integrators. 124
7.10 Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator. 125
7.11 Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator. 126
7.12 Here is the tree used to generate the alignment we used to test the effect of taxa on stepsize. 127
7.13 We see the tree used for the experiment and the relationship between stepsize and the number of edges. 127
7.14 Here we see that acceptance rate has a linear relationship with the log-posterior. 128
7.16 Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator. 129
7.15 Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator. 129
7.18 Comparison between chosing via ratio of the posterior densities vs a fixed probability. 130
7.17 Comparison of the effective sample size for entering 3 orthants as opposed to 2. . 130
7.19 Comparison of the effective sample size for when we parachute the momenta as described in section 5.7.3. 131
7.20 Comparison of the tree generated by MCMC (left) and the tree generated by COrtHMC (right). 131
7.21 Here we have a relatively short chain and the Metropolis-within-Gibbs outperforms COrtHMC. This provides a perfect example as to why Intrinsic ESS is a more reliable measure of ESS than taking the minimum ESS over edges. The minimum gives an ESS of 44 samples, several edges had an ESS of over 2000, neither really provide you with information of how the tree changes as they only describe a local region of the tree, whereas Intrinsic does. 132

7.22 The added complexity of HMC and the autocorrelation significantly slows down the process and reduces the ESS. 132
7.23 ppHMC behaves as expected just like most other COrtHMC crossing kernels. . . 133
7.24 The various informative panels in our R Shiny app (RStudio, Inc, 2013). There are two plots of the internal edges of the tree. The colour either denotes distance from a given topology or merely a change of topology. We can see the current tree and it's location on the Petersen graph. We can see the acceptance rate and the logposterior, as well as a density plot. 135

Part I

Background Material

Chapter 1

Introduction

Darwin's theory of evolution revolutionised biology and is the basis of molecular biology. However, due to incompleteness of historic fossil records and the time spans involved, it is challenging to learn about the evolutionary relationships between organisms and the inferred relationships are subject to uncertainty. This means that inferring the evolutionary relationships can be posed as a statistical problem using the genetic material available from extant species.

Darwinian evolution occurs via three main processes: inheritance, variation and selection. Inheritance is the passage of traits from parent to offspring, variation is the mutation of traits and selection is the process by which favourable traits become prevalent in the population. Sometimes one species may adopt different traits within sub-populations. Once these two sub-populations are no longer able to inherit from each other they are considered different species and speciation is said to have occurred. The branching process of speciation from shared roots results in the evolutionary structure being modelled as a directed tree with extant species at the leaves and points of speciation as the internal vertices. Such trees are called phylogenetic trees. While Darwin had to use morphological data provided by observable characteristics, modern molecular biology uses genetic data from present day DNA. As a result the problem becomes how to infer evolutionary relationships from the DNA of the species in question.

The most up to date methodology involves likelihood based approaches. Likelihood based approaches involve constructing a likelihood based on a Markov model. Markov models are used to model DNA substitution. DNA substitution is a combination of mutation and fixation, the processes underlying the variation of the species and uniformity of the subspecies respectively. Markov models are used because they are forgetful, and the simplifying assumption that DNA bases do not remember what was in their position is sensible as the process of evolution occurs

over many generations. To assume the contrary would imply that more than just genetic code is passed to the offspring. As we do not fully understand the process of base substitution and the geometry of a DNA strand it is reasonable to not force such a multilevel relationship into phylogenetic modelling since we are more likely to adversely affect the results. Two approaches are commonly used to attain information from the likelihood: maximum likelihood methods and Bayesian analysis. Maximum likelihood methods attempt to find the most likely tree, and so estimate a point. Bayesian analysis utilises Bayes' theorem in order to adapt a set of prior beliefs by the data. It produces a density distribution with regions of high and low probability.

Increasing the number of species increases the number of relationships modelled and the number of different phylogenetic trees possible. There are $(2n-5)!!$ unrooted phylogenetic trees for n species as described in section 3.1. This results in a space that rapidly becomes infeasible to explore quickly. Furthermore standard Markov Chain Monte Carlo (MCMC) techniques sample trees and model parameters separately, further decreasing the efficacy of exploratory algorithms. Hamiltonian Monte Carlo (HMC) provides a coherent method of exploration that samples everything in tandem and ensures that regions of low probability are not excessively proposed.

MCMC methods produce a chain of states designed so that the stationary distribution is the required posterior. Most MCMC methods are variants of the Metropolis-Hastings algorithm. The algorithm offers a new point from a proposal distribution to an accept-reject criterion, if accepted the chain moves to the posed point otherwise it remains fixed. HMC is a variant of the Metropolis-Hastings algorithm that employs auxiliary variables to ensure the posed point is accepted more readily, enabling larger jumps around the state space. It does this by constructing curves of constant density in the augmented state space and proposing points a certain distance along these curves. For phylogenetic inference using standard Metropolis-Hastings, a separate proposal is required for every collection of parameters. These parameters include, composition parameters, transition-transversion rate, gamma rate and tree selection. There are two issues with this approach. As the tree is the underlying process, proposals for model parameters are made conditional on the tree, compounding any high rejection rates. By contrast, the proposals for composition and transition-transversion rate may commonly be a lognormal random walk. Proposals for the tree have to be able to vary the structure of the tree, and doing so requires the use of topological moves that often produce very different trees with different likelihoods. In a large space of trees this results in points in regions of low probability being proposed frequently. By following equivalent density curves HMC prevents this issue.

HMC has a few criteria that need to be satisfied, most important among which, is the need for the likelihood to be differentiable and continuous. Tree space, the space of all trees with varying edge length and structure on a fixed number of species, is a manifold stratified space. The stratified manifold for Tree space does not accept a continuous flow generated by the likelihood. During the course of this thesis I will go over how to adapt HMC to a stratified space and apply it to phylogenetic inference. This includes the effect on the likelihood, how to cross boundaries, efficiency of computation, and how to conduct HMC on the model parameters. As with most methods of MCMC, HMC has many flavours, I will discuss which methods were chosen and which were discarded while giving reasoning for the choices made.

1.1 A Brief History of Phylogenetics

Phylogenetics is the study of the evolutionary history of a group of organisms. It is primarily concerned with speciation, when two sub-populations of organisms are no longer able to reproduce. The foundations for studying phylogenetics appeared in the 19th century with the work of Bronn, Darwin, Haekel and Mendel; Haekel himself coined the term Phylogeny in 1866.

The practice started out based on physical traits or morphologies with Mendel: "*The experimental plants must necessarily ... possess constant differentiating characters*", (Mendel and Bennett, 1965). Similarly Darwin employed a trait based method resulting in a question of common ancestry. Did all life evolve from one common ancestor or many? "*The term 'variety' is almost equally hard to define; but here community of descent is almost universally implied, though it can be rarely proved.*" (Darwin, 1857) In Phylogenetics a single common ancestor was assumed until in 1977 Woese & Fox demonstrated there to be three main clades of descent. Nevertheless structural relationship or 'evolutionary proximity' between different groups of organisms remains an issue of study.

Evolution has been used by humanity since civilisation began. The first cities Uruk and Ur relied heavily on a variant of grass with shorter glumes allowing it to be threshed. Over time this was selectively bred into the various forms of modern day cereal (Campbell, 2016). We did not understand the processes behind evolution until millennia later.

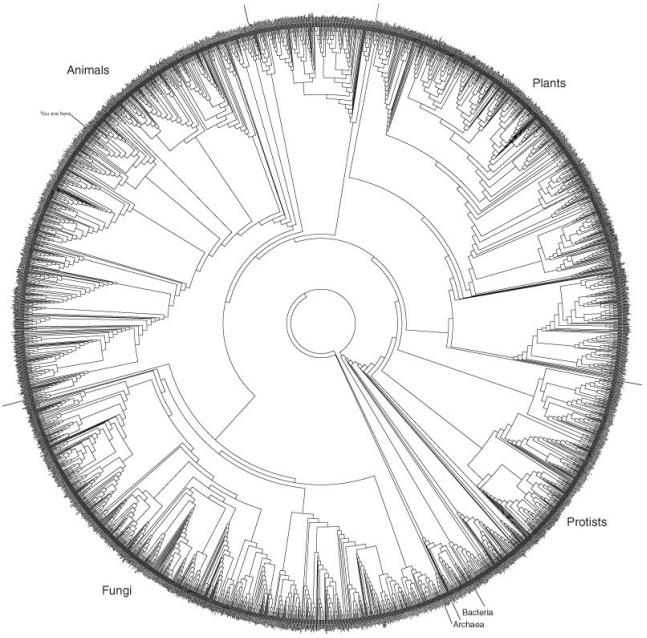


Figure 1.1: The Tree of Life, a phylogenetic tree constructed using most of the known base sequences of living creatures. In this diagram we can see the evolutionary relationship between the various kingdoms, phyla and classes of creatures living on earth. For example Animals and Fungi exist on the same subtree when compared to plants and so the phylogenetic inference used to create this tree has animals and fungi more closely related to one another than to plants. Source: David M. Hillis, Derrick Zwickl, and Robin Gutell, University of Texas

In the 1970's DNA sequencing provided genetic data. This meant that more advanced statistical techniques such as Maximum Likelihood Estimation or Bayesian Inference could be applied. This in turn led to an explosion in both theoretical models of evolution based on genetic data and computational programs such as Phylip, MrBayes (Ronquist and Huelsenbeck, 2003) and Beast (Drummond and Bouckaert, 2015).

1.2 DNA and Bases

Mendel (Abbott and Fairbanks, 2016) noticed the patterns of genetic inheritence between two sets of morphologies, through looking at seeds, if they had a round shape or angular shape, yellow or green albumen. Sutton and Boveri then independently related this pattern to chromosomes, (Sutton, 1903), (Boveri, 1904). Avery et al. (1944) demonstrated that DNA determined cell properties. Chagraff discovered two rules about the nucleotide base rates and Franklin, Watson and Crick famously discovered the double helix structure of DNA. DNA is formed of Deoxyribonucleic acid which is a compound molecule made up of two strings, each of many nucleotides connected

by a sugar phosphate backbone. These two strings are connected by hydrogen bonding between the nucleotides contained within them. It is these two strings which form the double helix structure. In humans the nucleotides are Adenine (A), Guanine (G), Cytosine (C) and Thymine (T). Adenine and Guanine and called purines because they consist of two rings, Cytosine and Thymine are called pyramidines because they consist of one. Adenine hydrogen bonds with Thymine and Guanine with Cytosine. These nucleotides are called bases. DNA supplies the code for the creation of proteins in the body, an activity that occurs in the ribosome, which transcribes the DNA into RNA and then translates it into a protein. In this way DNA determines the functionality of a living organism. Within DNA itself certain base locations have varying functional importance, for example we know that DNA consists of introns and exons where the intron is a buffer region not involved in protein construction. In 2010 approximately 90% of the genome was considered non-coding (National Human Genome Research Institute, 2010) and it is estimated more than 75% is non-coding (Graur, 2017). Srinivasan et al. (1987) demonstrates that local bases affect the local structure of the double helix and most Manhattan plots demonstrate the morphological links between nearby sites. This local correlation is quantified in many ways such as HESS. It is therefore important to maintain the order of sites in an alignment.

1.3 Motivation

The study of Phylogenetics provides two different types of categorical benefit. Benefits that arise from knowing species are distinct and benefits that arise from knowing they are similar. Such benefits can be wide ranging "*The fact that tomato and other subspecies of this genus actually are embedded within a well-marked subclade Solanum ... is a powerful statement that is important to geneticists, molecular biologists, and plant breeders*" (Soltis and Soltis, 2003).

1.3.1 Disease spread

Disease spread and infectivity is an important application of Phylogenetics. If a disease exhibits in one group of organisms, it will likely mutate to infect organisms of similar genetic makeup. To a plant breeder such information can therefore provide knowledge of what treatments to provide and how to cultivate resilience within a crop. Furthermore such information may provide possible breeding partners which have hitherto been ignored. The first example of phylogenetics being used was Holmes et al. (1995) demonstrating that HCV had existed within human populations for a long time when compared to HIV-1 and Pybus et al. (2001) demonstrated that the genetic history could lead to accurate R_0 estimation. More recently phylogenetic inference has been used to monitor and understand the diverging variants of SARS-COV-2 (Forster et al., 2020).

1.3.2 Scientific Efficiency

Cloning DNA takes several steps, one step involves cutting the DNA at certain bases by use of a restriction endonuclease. To quote an Eric Lander lecture; on the construction of ecoR1, a restriction enzyme found in Escherichia Coli strain R, "*Almost everything important that we say molecular biologists have come up with, it means molecular biologists sat at the feet of the true masters, bacteria, and learned from the true masters: this protein is found in nature*". The question is once ecoR1 is found, where to look for similar restriction enzymes but with modified cutting locations. Phylogenetic studies can provide knowledge of bacteria which are similar genetically to Ecoli and may provide similarly useful proteins. Without Phylogenetics the amount of time wasted searching in wrong clades of bacteria could be detrimental.

1.3.3 Virology

Phylogenetic analysis can be used to identify a virus at the start of an outbreak and determine its origin, with the SARS-CoV-2 virus early on three clusters could be observed, one related to the its appearance in bats (BatCoVRaTG13), an offshoot seen predominantly in eastern china and an offshoot from that seen mostly in Europe (Forster et al., 2020). Such analyses both show where the virus originated from and how it is spreading. Phylogenetic analysis can also identify the causes of inter-species transfer, such as mapping the development of HIV (Castro-Nallar et al., 2011). It can also be used to analyse cospeciation, how the virus develops with a population, branching into various new forms, for example Hepatitis C virus (HCV) has been shown to have 6 major types and 11 clusters (Simmonds et al., 1993), knowing which type of HCV you are treating can effect treatment methods as in particular type 1 may cause damage to the liver (Treatment Action Group, 2019).

1.4 Literature Review: Challenges in Phylogenetic Inference

Phylogenetic Inference using MCMC was first introduced by Mau and Newton (1997). To do so they use standard Metropolis-Hastings on the cophenetic matrix representation of a tree.

Mau and Newton (1997) use a split proposal which alternates between two different proposals depending on whether the extant species are very closely related or not. Newton et al. (1999) then extended this idea to a global proposal on trees not using the cophenetic matrix representation. Kuhner et al. (1995) also use MCMC to estimate effective population size.

Yang and Rannala (1997) also uses a Metropolis-Hastings scheme to conduct their inference

over the posterior. They use nearest neighbour interchange to change topologies. Yang and Rannala (1997) models evolution along a branch as a birth-death process as described in Nee et al. (1994). Yang (2006) then provides a clear over of the state of phylogenetic inference in 2006. More recently Yang and Rannala (2010) use a reverse jump MCMC method (rjMCMC), see Johansen and Evers (2007), to either expand or contract the number of nodes. This is in order to provide a Bayesian method of species delimination. Stephens and Donnelly (2000) compare the MCMC approach with numerical approximation, Stephens and Donnelly (2000) also documents various different proposal mechanisms used in MCMC for phylogenetics. Several software packages have since been produced to run phylogenetic inference, the most famous of these is MrBayes, (Huelsenbeck and Ronquist, 2001) based on work by Huelsenbeck and Ronquist, and Beast, (Drummond and Bouckaert, 2015) and (R. et al., 2019), based on work by Drummond, Bloomquist, Rambaut and many collaborators.

1.4.1 MCMC

A more concrete example is (Cherlin, 2016). Phylogenetic inference is conducted over a product space. This product space consists of Tree space, the substitution model parameter space and the site rate parameter space. (Cherlin, 2016) uses the Gibbs sampling algorithm to sample the tree, the substitution model parameters and the site rate parameters individually. A Metropolis-Hastings algorithm can be employed at each step (Cherlin, 2016). To sample a new tree one possibility is to propose new edge lengths according to a normal random walk. An NNI, SPR or TBR is then performed on a subset of the edges of the tree. A Metropolis-Hastings update for the new tree is then performed. New substitution model parameters can also be proposed via the Metropolis-Hastings algorithm. The acceptance probability for the proposed parameters is computed and the proposed parameters are accepted or rejected. Similarly, site rate parameters can also be found by the Metropolis-Hastings algorithm. This sequence of three different Metropolis-Hastings algorithms are then repeated forming a Markov chain of trees and substitution parameters, and repeated until the chain has apparently converged to its stationary distribution.

1.4.2 Metropolis-Coupled MCMC

Metropolis-Coupled MCMC $(MC)^3$ (Altekar et al., 2004) runs several chains with varying annealing temperatures and then combines them. Phylogenetic inference is conducted over various different topologies with radically different density gradients, so by running multiple chains with different temperatures it ensures that the posterior effectively informs the chain greatly increasing the exploration and effective sample size of a MCMC approach.

$(MC)^3$ requires two decisions to be made, the first is which temperatures to run the chains at and the second is which chains to swap between. When the posterior is continuous it makes sense to swap between neighbouring temperatures. It common for temperatures to map $\mathbb{N} \to [1, \infty)$ and there are various ways of doing this often involving a change in temperature which then becomes another parameter to be chosen.

We do not cover a Metropolis-Coupled version of COrtHMC as annealed HMC is covered in many different sources such as Neal et al. (2011), through dividing the Hamiltonian by the temperature. While running multiple different temperature chains would likely improve the result it is clearer to compare the results at a fixed temperature and any improvements would be expected to carry over to the Metropolis-Coupled version.

1.4.3 Huelsenbeck and Ronquist

As previously stated Beast and MrBayes are the two main competing tools used for phylogenetic inference MrBayes developed by Huelsenbeck and Ronquist. Huelsenbeck has a long history in the field, (Hillis et al., 1994), (Huelsenbeck, 1997) and (Hillis and Huelsenbeck, 1994). MrBayes and Ronquist and Huelsenbeck (2003) employ $(MC)^3$. In Huelsenbeck et al. (2002) problems involved in the MCMC approach are discussed, such as poor mixing, reliance on priors and whether a Bayesian approach results in overconfidence. Nylander et al. (2004) then explores including morphological in the form of the $Mk\Gamma - GTRI\Gamma$ which they found "contributed to $< 5\%$ of the characters in our data but still had significant influence on the tree" and found that a Bayesian comparison can "favor simple models over much more complicated ones". In Huelsenbeck et al. (2004) they use the reversible jump Markov Chain Monte Carlo Algorithm to sample between different time reversible Markov processes to model evolution. In the reversible jump MCMC requires a differentiable way to compare between two spaces of different dimension. Huelsenbeck et al. (2004) use one of a hierarchical likelihood-ratio test, the Akaike Informaiton Criterion or the Bayesian Informaiton Criterion, they find that partitioned model heterogeneity across sites improved fit but that it is more important to include within partition rate heterogeneity. Huelsenbeck and Andolfatto (2007) expands the number of available priors to include a dirichlet process which is useful for multiple populations. Perhaps the most useful paper with regards to HMC is Lakner et al. (2008) in which the compare a continuous change proposal from Jow et al. (2002) and the LOCAL algorithm of Larget (2008) with TBR and SPR (see subsection 3.4.1) and find that they are not very efficient when compared to TBR and SPR, as HMC is a local algorithm we might expect similar results.

1.4.4 Probabilistic Path Hamiltonian Monte Carlo

Probabilistic Path Hamiltonian Monte Carlo (ppHMC) (Dinh et al., 2017) is a method developed by Dinh, Bilge, Zhang and Matsen in Seattle, it has a similar formulation to COrtHMC, our technique. To conduct ppHMC they employ the Leap-prog integrator. The Leap-prog integrator integrates according to the hamiltonian checks for a boundary crossing and then proposes a new point in a new topology. They also developed the refractive Leap-prog algorithm which upon crossing into a new topology rescales the momenta based upon the difference between the posterior likelihood of the proposal and current topologies. Like Leap-prog COrtHMC checks repeatedly to see if the Markov chain leaves an orthant, when it does do COrtHMC proposes a new orthant and new momenta like Leap-prog. COrtHMC like the refractive Leap-prog enables momenta to be resampled. Leap-prog keeps the set of indices crossed whereas COrtHMC methods apply sequential techniques for dealing with multiple crossings. In Leap-prog it chooses a new topology using a uniform distribution, this is equivalent to option 1 in subsection 5.7.4. Leap-prog also employs a system where the state proposed in a new topology is as close to the point where the flow crosses the orthant boundary as possible, this extra level of refinement around orthant boundaries is something we did not propose in our various different crossing methods. Finally Leap-prog requires all boundary crossings are known in $FirstUpdateEvent$.

1.4.5 Sequential Monte Carlo

The idea of Sequential Monte Carlo (SMC) is to have a collection of particles in the state space with are weights. These particles are then parsed through a filter in which an intermediate set of particles are sampled from the old particles according to their weights with replacement. This intermediate set generates a new set of particles from a specified proposal distribution whic takes a particle as an input. New weights are then calculated for the set of particles and an estimate for the posterior is given by the average of the weighted averages of each collection of particles. In PosetSMC (Bouchard-Côté et al., 2012) an adaption of SMC for phylogenetic inference the particles are forests connecting the leaves representing extant species. PosetSMC therefore requires a proposal distribution for forests which depends upon an existing forest. It then produces a set of forests at each iteration and refines them by their weighting. In a way similar to parallel MCMC this effectively enables exploration over a large subsets of the state space simultaneously speeding up the convergence of the algorithm.

1.4.6 Variational Bayesian Phylogenetic Inference

A new method of phylogenetic inference has recently arisen called Variational Bayesian Phylogenetic Inference (VBPI)(Zhang and IV, 2019), (Zhang, 2020). It works by making use of Subsplit

Bayesian Networks (SBN). In Subsplit Bayesian Networks you have a total order on the clades of a tree and you can split the density into a product of the conditional density of each clade, this creates a family of distributions. VBPI takes the SBNs as a family of approximate distributions over phylogenetic tree topologies and combines it with a family for edge lengths. VBPI then rephrases the inference problem as minimizing the Kullback-Leibler divergence of this family with the posterior distribution. There are many methods of approximating the solution one such method is stochastic gradient descent. This means that the time consuming Markov chain construction is bypassed resulting in faster results.

1.5 Contribution

We have developed a novel method of applying HMC to a phylogenetic setting and have produced results agreeing with those of others who have attemped similar approaches demonstrating that HMC should not be applied to this setting. We have developed a package to conduct our approach with blackbox and user defined parameterisation in which we have developed and implemented methods of speeding up the inference. We have designed our own way of measuring ESS and ESS/s to get around the static pendant edge problem which can cause standard ESS and ESS/s to artificially lower the evaluation of ESS/s. We have also implemented and compared ppHMC with COrtHMC and found the two techniques to produce like results. Finally we have developed new ways of visualising MCMC on Tree space.

Chapter 2

Substitution Models

This chapter describes how we construct models for evolution to use in likelihood based inference schemes. We do this by using Markov processes to construct a likelihood. These Markov processes provide a probabilistic model for how a base transforms throughout the evolutionary tree from a common base ancestor. We start by considering how a single base of DNA evolves from an ancestor to a descendant. Changes in the DNA base can be modelled via a continuous-time discrete state Markov process.

2.1 Markov Models and Markov Processes

A Markov process is a particular type of probabilistic model for moving between states randomly over time. We shall only concern ourselves with discrete state continuous time Markov processes which move between discrete states randomly over continuous time. Markov processes are distinguished from more general stochastic processes by the Markov property. This states that the future state is conditional only on the current state, and that future states do not depend on past states. A Markov process consists of a family of random variables Z_t indexed by t in some indexing set T, thought of as times, taking values in a state space $\mathcal{A}$. A more in depth explanation of Markov processes can be found in Adams (2011).

Definition (The Markov Property)**.** The Markov property for the collection Z_t is:

$$\forall k \in \mathbb{N}_{\leq n-1} \qquad t_n \geq t_{n-1} \geq ... \geq t_{n-k} \qquad z, z_{n-1}, ..., z_{n-k} \in \mathcal{A}$$
$$\mathbb{P}(Z_{t_n} = z | Z_{t_{n-1}} = z_{n-1}, ..., Z_{t_{n-k}} = z_{n-k}) = \mathbb{P}(Z_{t_n} = z | Z_{t_{n-1}} = z_{n-1}).$$

A Markov process can be expressed in the form of a Markov kernel.

Definition (Kernel)**.** A kernel P is a two argument function taking in times t and s and returning the probability of Z_t given Z_s for all possible states. For discrete state-spaces the transition kernel P of the Markov Process Z_t can be represented as a matrix that corresponds to the transition probability, (Johansen and Evers, 2007), $P(t,s) = [P(t,s)]_{ij}$. Each entry of the matrix is given by the following:

$$[P(t,s)]_{ij} = \mathbb{P}(Z_t = i | Z_s = j).$$

In this project we begin by considering random changes to a single base in a DNA strand. In this case the state-space consists of the DNA bases $\mathcal{A} = \{A, G, C, T\}$ and Z_t will be indexed by $t \in T \subset \mathbb{R}$.

Definition (Discrete-state continuous time Markov process)**.** A discrete-state continuous time stochastic process is a family of random variables indexed by time $t \in T \subset \mathbb{R}$ and taking values in a discrete state space. A discrete-state continuous time Markov process is a discrete-state continuous time stochastic process satisfying the Markov property.

Definition (Time Homogeneity)**.** A Markov process is time homogeneous if $\mathbb{P}(Z_t = z_t | Z_s = z_s) = \mathbb{P}(Z_{t-s} = z_t | Z_0 = z_s)$ for all s and t in the index set such that $t \geq s$.

When the Markov process is time homogeneous, $P(t,s) = P(t-s, 0)$ and so can be represented by a function that we shall also call P with a single parameter, $P(t) = P(t, 0)$. When the kernel is discrete $P(t)$ can be represented by the matrix $[P(t,0)]_{ij}$. Most substitution models in phylogenetics are time homogeneous and all models in this thesis will be too.

A Markovian approach is appropriate to modelling evolution. The only observed influences are the current population size and the current environmental stresses and biochemical properties. These are memoryless. In phylogenetics we start by taking the state at a single site of DNA for a common ancestor of the species of interest. We take this state to be the state at time $t = 0$. As t increases Z_t evolves according to a Markov process.

2.1.1 Chapman-Kolmogorov equation

Chapman-Kolmogorov equations are important identities of Markov processes. The Chapman-Kolmogorov equation for time homogeneous Markov processes is:

$$P(t_1 + t_2) = P(t_1)P(t_2)$$

where $P(t)$ is the kernel as viewed as a matrix. This can be derived from the Markov property as follows:

$$\begin{aligned}\mathbb{P}(Z_{t_1+t_2} = i|Z_0 = j) = [P(t_1+t_2)_{ij}] &= \sum_{k\in\mathcal{A}} \mathbb{P}(Z_{t_1+t_2} = i \mid Z_{t_1} = k, Z_0 = j)\mathbb{P}(Z_{t_1} = k \mid Z_0 = j) \\ &= \sum_{k\in\mathcal{A}} \mathbb{P}(Z_{t_1+t_2} = i \mid Z_{t_1} = k)\mathbb{P}(Z_{t_1} = k \mid Z_0 = j). \\ &= \sum_{k\in\mathcal{A}} [P(t_1)]_{ik} [P(t_2)]_{kj} \\ &= [P(t_1)P(t_2)]_{ij}.\end{aligned}$$

2.1.2 The rate matrix

In this section we are using P_t to denote $P(t)$ for clarity. If we have a time homogeneous process and if the following limit exists, then P_t is differentiable in t and the rate matrix Q can be defined:

$$Q := \lim_{t\to 0} \frac{P_t - \mathbb{I}}{t} = \frac{dP_t}{dt}.$$

The time homogeneous forward and backward Kolmogorov equations $\frac{dP_t}{dt} = P_tQ$, $\frac{dP_t}{dt} = QP_t$ can be derived from the Chapman-Kolmogorov equations. By the Chapman-Kolmogorov equations

$$P_{t+dt} = P_tP_{dt} = P_t(\mathbb{I} + Qdt + \mathcal{O}(dt^2)) \text{ and } P_{(t+dt)} = P_{dt}P_t = (\mathbb{I} + Qdt + \mathcal{O}(dt^2))P_t.$$

These in turn imply

$$\frac{P_{(t+dt)} - P_t}{dt} = P_tQ + \mathcal{O}(dt) \text{ and } \frac{P_{(t+dt)} - P_t}{dt} = QP_{dt} + \mathcal{O}(dt).$$

Letting $dt \to 0$ gives

$$\frac{dP_t}{dt} = P_tQ \text{ and } \frac{dP_t}{dt} = QP_t.$$

To find a solution for P in terms of Q make an Ansatz that $P_t = P_0 \exp(Qt)$. Simple computation reveals this to be the solution. The rate matrix Q is sometimes called the Q matrix or generator. The elements of the rate matrix q_{ij} are called the transition rates and denote the rate at which the state leaves state i and moves to state j. As we can obtain the probability kernel P from Q it is often clearer to express the substitution model in terms of the rate of flow between states rather than the probability of being at a given state at time t. It is important to define a stationary distribution as many models use the stationary distribution as a key model parameter.

Definition (Stationary Distribution)**.** The stationary distribution is the vector $\Pi = (\pi_1, ..., \pi_n)$ where $n = |\mathcal{A}|$ such that for all t, $P_t\Pi = \Pi$.

If the Markov process is time homogeneous, irreducible and recurrent, then the stationary distribution exists and is unique. The stationary distribution represents the limiting distribution over the states if the process is ergodic.

Definition (Reversible Markov Process)**.** A Markov process is reversible if when stationary the transition probability going forward in time is the same as going backwards, where $\mathbb{P}$ is the transition probability when stationary.

$$\mathbb{P}(t, t-s)_{ij} = \mathbb{P}(t, t+s)_{ij}, \ \forall i, j \in \mathcal{A}.$$

If a Markov process satisfies the detailed balance equation $\pi_i P(t)_{ji} = \pi_j P(t)_{ij}$ then it is reversible.

It is important to say that we shall be using a time reversible Markov process. This means that while notionally there is a root representing the common ancestor in any given phylogenetic tree, the distribution of random variables at the leaves does not depend on the position of the root. In particular, this means that the likelihood we construct is invariant under changes of the root position. This flexibility in the location of the root will enable us to compute derivatives with respect to edge lengths faster as we shall explain in chapter 5.

2.2 The substitution model

We have outlined how a single site evolves over time by a Markov process. We now go on to explain how different substitution models give different families of kernels for that Markov process. A Substitution Model is a certain parametrised family of rate matrices Q. It defines the probability of one state moving to another and may depend on various factors. We use θ to range over the various different parameters of Q and we denote the space of substitution parameters Θ.

2.2.1 Examples of substitution models

Several standard examples of substitution model can be found in the book by Yang (1994). The generic layout for the rate matrix of a substitution model is depicted in figure 2.1. Each entry consists of base rates and conversion rates. The base rates, π_A, π_C, π_G and π_T, represent the stationary distribution. The conversion rates are q_{ij}, $i, j \in \mathcal{A}$. The asterisk is a place holder for the negation of the sum of the row, e.g. for the first line $\star = -\pi_C q_{AC} - \pi_G q_{AG} - \pi_T q_{AT}$.

$$Q = \begin{array}{c|cccc} \cdot & A & C & G & T \\ \hline A & \star & \pi_C q_{AC} & \pi_G q_{AG} & \pi_T q_{AT} \\ C & \pi_A q_{CA} & \star & \pi_G q_{CG} & \pi_T q_{CT} \\ G & \pi_A q_{GA} & \pi_C q_{GC} & \star & \pi_T q_{GT} \\ T & \pi_A q_{TA} & \pi_C q_{TC} & \pi_G q_{TG} & \star \end{array}$$

Figure 2.1: The General Rate Matrix

2.2.2 General Time Reversible Model (GTR)

Most examples in Yang (1994) are specific instances of the Generalised Time-Reversible model (GTR) (Tavaré, 1986). The GTR model assumes that there is a stationary distribution. The GTR model also assumes that there are different transition rates associated with each pair of bases. It has the same rate forwards and backwards between states, and hence it is reversible. All the models we consider are specific examples of GTR and are therefore also reversible. GTR has the most parametric freedom while preserving reversibility. This means given enough data GTR should replicate the results of any other model but can cause overfitting (Spielman, 2019). It is the instance of the general model in which $q_{ij} = q_{ji}$.

$$Q_{GTR} = \begin{array}{c|cccc} \cdot & A & C & G & T \\ \hline A & \star & \pi_C\alpha & \pi_G\beta & \pi_T\gamma \\ C & \pi_A\alpha & \star & \pi_G\delta & \pi_T\epsilon \\ G & \pi_A\beta & \pi_C\delta & \star & \pi_T\eta \\ T & \pi_A\gamma & \pi_C\epsilon & \pi_G\eta & \star \end{array}$$

In this project we need to to acquire the transition matrix. For this it is necessary to diagonalise Q and compute its eigenvalues. The transition matrix is then a function of the exponentials of the eigenvalues. Unfortunately, the eigenvalues of Q are not readily identifiable as functions of the substitution model parameters (Roch, 2012). Also the derivative of the transition matrix with respect to substitution model parameters cannot be taken in terms of matrix Q as

$$\frac{d\exp(Qt)}{d\theta} = \frac{d}{d\theta}\sum^{\infty}\frac{(Qt)^i}{i!}$$

and matrix multiplication is not commutative. Here θ is one of $\pi_A, \pi_C, \pi_G, \pi_T, \alpha, \beta, \gamma, \delta$ or ϵ. This means that the derivative of the transition kernel with respect to its parameters requires an approximation. We chose to avoid approximations where possible as we are interested in the

behaviour of HMC rather than the impact of the approximation. Other models have an algebraic transition kernel and so do not require such an approximation. It is for this reason that we decided to use HKY85 as our model in testing our algorithm. HKY85 only has two conversion parameters (or one up to a constant) which means that the transition matrix can be written down explicitly in terms of all the substitution model parameters and an exact derivative can be computed.

2.2.3 Hasegawa, Kishino and Yano 1985 (HKY85)

Other models are motivated by properties of DNA. DNA bases are composed of purines A & G, which have both an iodine and pyrimidine ring and complementary pyrimidines C & T. The Kimura 2-parameter model posits one rate, α for transitions and another rate, β for transversions.

Definition (Transitions and transversions)**.** A transition is a purine to purine or pyrimidine to pyrimidine substitution and a transversion is a purine to pyrimidine or pyrimidine to purine substitution.

HKY85 (Hasegawa et al., 1985) generalises the Kimura 2-parameter model and the Felsenstein models. The Felsenstein model has the base transition parameters all the same, $q_{ij} = 1$ for all $i, j \in \mathcal{A}$. The Kimura 2 model is the HKY85 model with fixed base rate probabilities, $(\pi_A, \pi_C, \pi_G, \pi_T) = (\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$. HKY85 has the following rate matrix:

$$Q = \begin{array}{c|cccc} \cdot & A & C & G & T \\ \hline A & \star & \pi_C\beta & \pi_G\alpha & \pi_T\beta \\ C & \pi_A\beta & \star & \pi_G\beta & \pi_T\alpha \\ G & \pi_A\alpha & \pi_C\beta & \star & \pi_T\beta \\ T & \pi_A\beta & \pi_C\alpha & \pi_G\beta & \star \end{array}$$

HKY85 employs the use of base rates or transition-transversion ratios. Such parameters result in a problem for HMC, which is normally conducted on parameters taking values on $\mathbb{R}$ without constraint. The bases $\pi_A, \pi_C, \pi_G, \pi_T$ lie on a simplex. This is an issue as it requires a form of constrained HMC, where $\pi_A + \pi_C + \pi_G + \pi_T = 1$; $\pi_A, \pi_C, \pi_G, \pi_T \geq 0$. As such it is necessary to transform the base rates to make them applicable for HMC. The transition-transversion rates also need to be transformed as they are required to be strictly non-negative.

One of the key properties for us of HKY85 is that the eigenvalues have simple closed forms and the diagonal can be computed. Following the work of Nye and Heaps (2014) we know that the eigenvalues are $1\ -\beta t\ -(\pi_Y\alpha + \pi_R\beta)t$ and $-(\pi_R\alpha + \pi_Y\beta)$. This enables the transition matrix to

be computed. Specifically if we let:

$$\pi_R = \pi_A + \pi_G \qquad \pi_Y = \pi_C + \pi_T$$
$$yr = \pi_Y/\pi_R \qquad ry = 1.0/yr$$
$$e_2 = \exp(-\beta t) \qquad e_3 = \exp(-(\pi_R\alpha + \pi_Y\beta)t)$$
$$e_4 = \exp(-(\pi_Y\alpha + \pi_R\beta)t)$$

then:

$P(t) = \exp(tQ)$

$\cdot$	A	C	G	T
A	$\pi_A + \pi_A yre_2 + \pi_G/\pi_R e_3$	$\pi_C(1-e_2)$	$\pi_G + \pi_G yre_2 - \pi_G/\pi_R e3$	$\pi_T(1-e_2)$
C	$\pi_A(1-e_2)$	$\pi_C + \pi_C rye_2 + \pi_T/\pi_Y e_4$	$\pi_G(1-e_2)$	$\pi_T + \pi_T rye_2 - \pi_T/\pi_Y e_4$
G	$\pi_A + \pi_A yre_2 - \pi_A/\pi_R e_3$	$\pi_C(1-e_2)$	$\pi_G + \pi_G yre_2 + \pi_A/\pi_R e_3$	$\pi_T(1-e_2)$
T	$\pi_A(1-e_2)$	$\pi_C + \pi_C rye_2 - \pi_C/\pi_Y e_4$	$\pi_G(1-e_2)$	$\pi_T + \pi_T rye_2 + \pi_C/\pi_Y e_4$

.

Moreover, the above formulation for HKY85 enables us to compute the derivatives $\frac{dP(t)}{d\theta}$. We give $\frac{dP(t)}{d\pi_A}$ as an example. For ease of notation the symbol $'$ will denote $\frac{d}{d\pi_A}$ applied to the preceding term.

$$\pi_R' = 1 \qquad \pi_Y' = 0$$
$$yr' = -\pi_Y/\pi_R^2 \qquad ry' = 1/\pi_Y$$
$$e_2' = 0 \qquad e_3' = -\alpha t e_3$$
$$e_4' = -\beta t e_4$$

$P'(t) =$ (*first two columns*)

.	A	C
A	$1 + yre_2 + \pi_A yr'e_2 - \pi_G/\pi_R^2 e_3 + \pi_G/\pi_R e_3'$	0
C	$1 - e_2$	$\pi_C ry'e_2 + \pi_t/\pi_Y e_4$
G	$1 + yre_2 + \pi_A yr'e_2 - 1/\pi_R e_3 + \pi_A/\pi_R^2 e_3 + \pi_A/\pi_R e_3'$	0
T	$1 - e_2$	$\pi_C ry'e_2$

(*last two columns*)

.	G	T
A	$\pi_G yr'e_2 + \pi_G/\pi_R^2 e_3 - \pi_G/\pi_R e_3'$	0
C	0	$pi_T ry'e_2 + \pi_T/\pi_Y e_4$
G	$\pi_G yr'e_2 + 1/\pi_R e_3 + \pi_A/\pi_R^2 e_3 - \pi_A/\pi_R e_3'$	0
T	0	$\pi_T ry'e_2$

Derivatives of P for all parameters were calculated algebraically and coded up in our software.

2.2.4 Substitution Models and Trees

So far we have only considered a single site, with no speciation in the evolution. This is a Markov process on $\mathbb{R}$ and can be represented by the line graph e with vertices $v(e(1))$ and $v(e(2))$, and length $\ell(e)$. The length of the edge represents the time taken multiplied by the rate of evolution, represented by μ_e, $\ell(e) = \mu_e t_e$. We cannot identify or differentiate between the effects of μ_e and t_e. Over time a species can speciate, this is a branching of the Markov process. A Markov process that branches again and again forms a Markov process on a tree, so that Z_t is indexed by points in a tree, rather than a subset of $\mathbb{R}$. The Chapman-Kolmogorov equations ensure that the Markov process can speciate at any point coherently.

To sample from the process, we start with a collection of sites in a common ancestor for extant species. This common ancestor is represented by the root of the tree. We assume that these sites can be identified with sites in extant species by common descent. We can then simulate the Markov process by simulating branch-wise. The branch-wise simulation for any edge in the tree can be treated as standard simulation on $\mathbb{R}$. Given a base at one vertex of the edge, w.l.o.g. $v(e(1))$, the base at the other, $v(e(2))$, is simulated according to the Markov process, the transition probabilities being $P(\ell(e))$. Repeating this process down the tree eventually results in bases being simulated for the leaves. Each extant species is represented by a leaf. The simulated base at the leaf gives a simulated base for the extant species identified with a particular site in

the ancestral species. We require many sites to get an accurate idea of te evolutionary distance between multiple species. Mice and humans have 85% of protein coding regions of DNA identical to one another, a single site would therefore be a poor judge of evolutionary distance (National Human Genome Research Institute, 2010). Each site produces a simulated base at each leaf referencing both the site and the extant species. This data can be collated in the form of an Alignment. The Alignment is the data structure we use for inference.

2.3 The Alignment

The data used to constructed the likelihood is called the alignment (Yang, 1994). This data consists of DNA bases arranged into an array. As seen in section 2.2.1 DNA consists of nucleotide pairings between the four bases: adenine (A), cytosine (C), guanine (G), thymine (T). DNA sequencing provides a list of bases and where they appear in the DNA strand. The set of all possible bases forms the alphabet denoted $\mathcal{A}$. In section 2.1 we described a Markov process transitioning a root base to a pendant base. The pendant base is dependent on species and initial root base. We can therefore construct an array $\mathbf{X} = \{x_{ij}\}_{i=1:m,j=1:n}$ where the rows represent different species and the columns represent the site in the DNA from the common ancestor base x_{ij} is associated with. It is standard practice to order the DNA string to maintain the order of the sites in the DNA. This can be beneficial in analysing global site heterogeneity, but it has been shown that site rate heterogeneity depends on local affects also, such as the surrounding bases and methylation, the most famous example of this is at CpG dinucleotide sites (Todorova and Danieli, 1997), which can lead to an above 10 fold increase in specific mutations(Hodgkinson et al., 2009). Hodgkinson et al. (2009) perform a comparison of SNPs between human and chimpanzee genomes, they find that there is significant global variation in mutation, not be explained by local genomic data, supporting a consistent ordering. A global approach is further supported by Matassi et al. (1999) which uses CpG sites to see the change in mutation rate. Nevertheless in adopting a global approach to the order most substitution models remove the granularity required to account for local behaviour, something that a categorised reordering might be able to account for. Combining all the strings of DNA bases for the different species of interest creates an alignment. The alignment is the data over which phylogenetic inference is conducted.

Definition (Alignment $\mathbf{X} = \{\chi_i\}_{i=1:m} = \{x_{ij}\}_{i=1:m,j=1:n}$)**.** The alignment is the name given to the $n \times m$ matrix of genetic bases where the m rows correspond to different species and the n columns to different sites of the DNA sequence.

Species 1	A	G	T	G	C	...	A	G	C
Species 2	A	C	A	T	A	...	G	T	A
Species 3	G	G	T	T	G	...	A	C	C
⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮
Species m	T	G	A	T	C	...	T	T	A

Figure 2.2: Example of an Alignment

2.4 Linear Site Rate Heterogeneity

It is known that selection effects differ depending on site. This means that it is sensible to model the rate of evolution as varying over the different sites. The rate is added into the likelihood by multiplying the rate matrix by a constant k and the likelihood by the probability of that constant. That constant is often modelled as being drawn from a gamma distribution with parameters (α, α). As k is unknown we have to average over k, that is to integrate the conditional likelihood of k, multiplied by the probability of k over all possible values of k. This results in difficult integrals (Yang, 2006):

$$\prod_j \int_0^\infty L_k g_\alpha(k) dk$$

where L_k is the likelihood given k and j ranges over sites in the alignment, see section 2.5. The function $g_\alpha(k)$ is the probability of k from a $\Gamma(\alpha, \alpha)$ distribution. Yang (1994) shows that the discrete gamma distribution provides a suitable approximation for 4 or more classes. This replaces the above integral with

$$\prod_j \sum_{k_1,\ldots,k_n} L_{k_i} g_\alpha(k_i)$$

where k_i are the values at which a fraction of the density is reached. There are other models such as the log-normal model, a mixture of invariable sites (Yang, 2006) or a gamma mixture model (Mayrose et al., 2005) but we decided to adopt the simplest, the discrete gamma model.

The discrete gamma model takes the gamma distribution and splits it into K bins. Each bin represents $1/K$ of the total density. The model selects the mean of each bin to be the different rates $k_1, ..., k_K$. The rates $k_1, ..., k_K$ are dependent on α. The k_i are computed by computing the value at which the cumulative density function is equal to the appropriate density.

For example, suppose $K = 4$, then there are four partitions of $[0, 1]$, $[0, 0.25)$, $[0.25, 0.5)$, $[0.5, 0.75)$ and $[0.75, 1]$, each partition has an associated mean $c_1 = 0.125$, $c_2 = 0.375$, $c_3 = 0.625$, $c_4 = 0.875$.

$k_1, ..., k_4$ are the solutions of

$$\int_0^{k_i} \frac{t^{\alpha-1}\exp(-t)}{\Gamma(\alpha)}dt = c_i.$$

The above gives the appropriate rates k_i and the probability of k_i is $\frac{\alpha^\alpha k_i^{\alpha-1}\exp(-\alpha k_i)}{\Gamma(\alpha)}$. To conduct HMC we need to be able to compute the derivative of k_i with respect to α. This is intractable and requires a numerical approximation. To run HMC we discretised $\mathbb{R}$ so that α took values in $\{0.001 \times j\}_{j=1}^{6000}$, and computed k_i for all the values on the discretisation, $\hat{k_i^j}$. We then used the finite difference method to estimate the derivative of k_i for for every α on $\mathbb{R}$. The number 6000 was chosen as the gradient has tapered off at $\alpha = 6$ for all k_i up to 8 bins. A refinement of 0.001 was chosen because it resulted in the same derivative of the likelihood with respect to α as the numerical approximation.

$$\frac{dk_i}{d\alpha} = \frac{k_i^{\hat{\alpha_{j+1}}} - k_i^{\hat{\alpha_{j-1}}}}{\alpha_{j+1} - \alpha_{j-1}}.$$

2.5 The Markov Model and the Likelihood

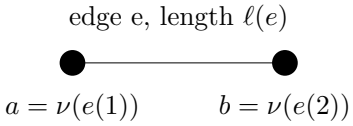
Constructing the likelihood is key to conducting inference over the posterior. The likelihood is constructed according to a Markov process on edge lengths and then as more edges are added these individual transition probabilities are multiplied together. The likelihood is defined in terms of alignment $\mathbf{X}$, model parameters θ, the site rate parameters α that generate k, tree T consisting of edge lengths ℓ and topology τ, and alphabet $\mathcal{A}$. The relationship $u \prec v$ denotes that u is a child of v. The child relationship requires the tree to be directed. We direct the tree such that the root is the parent of all its neighbouring vertices. Disconnecting these child vertices from the root generates several subtrees. Repeating the process provides a parent child relationship for every connected pair of vertices in the tree. The likelihood $\mathcal{L}_u$ is the subtree likelihood, the likelihood of the subtree created when u is disconnected from its parent v. This means that we can construct likelihood as a standard tree recursion in terms of $\mathcal{L}_u$. We write $P_{x,y;k}\{\ell, \theta\}$ for the probability of transitioning from x to y given parameters τ, θ and α. We denote the stationary distribution $\mathbb{P}(v = x)$ or simply $\underset{\sim}{\pi}(x)$.

The likelihood is then:

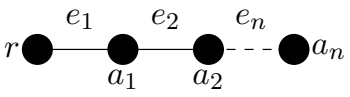
$$\mathcal{L}(\mathbf{X}|\ell, \tau, \theta, \alpha) = \prod_{i=1}^{n} \sum_{k \in \gamma} \mathbb{P}(k) \sum_{x \in \mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u \prec v_0} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(u, v_0), \theta\} \mathcal{L}_u(\chi_i|y, \ell, \tau, \theta, \alpha) \right] \right].$$

We now explain this construction. We start with a single site i. We require the transition matrix of a Markov process P modelling base substitution. We then consider the set of all possible

labellings of the vertices of the tree with members of the alphabet. Each member of the set has the requirement that for the leaves l_j the labelling matches the associated base $\omega(l_j) = x_{j,i}$, the observation of that leaf in the alignment. We let ω_r denote the labelling at the root. We can now consider a single edge in a given tree T with specific labelling ν as two bases separated by a distance.



The probability of substituting the base a with base b in evolutionary time $\ell(e)$ is given by the transition matrix $P_{a,b;k}\{\ell(u, v_0), \theta\} = P(0)\exp(Qk\ell)_{a,b}$ where θ represents all the substitution model parameters.



The probability of moving from the nucleotide at the root, r to nucleotide a_n in time t consists of two sums, one containing points $e_1, ..., e_n$ such that they lie in the simplex $\nabla = \{\ell(e_1)+...+\ell(e_n) = t|\ell(e_i) \in \mathbb{R}\}$ and the over possible bases at the internal vertices. The summand is given by the transition matrix $P_{r,a_1;k}\{\ell(e_1), \theta\}P_{a_2,a_3;k}\{\ell(e_2), \theta\}...P_{a_{n-1},a_n;k}\{\ell(e_n), \theta\}$, which can be simplified by the Chapman-Kolmogorov equations.

Consider a species S_v, with w children $S_{v,1}$,...,$S_{v,w}$. It seems sensible to take each branch of evolution as independent of each other. This results in the likelihood defined on a subtree of v being:

$$\begin{aligned}\mathcal{L}_v(\chi_i|s_v, \theta, k) =& \mathbb{P}(S_v = s_v \wedge S_{v,1} = A \vee S_{v,1} = C \vee S_{v,1} = G \vee S_{v,1} = T \\ & \wedge ... \wedge S_{v,j} = A \vee S_{v,j} = C \vee S_{v,j} = G \vee S_{v,j} = T) \\ =& \prod_{l=1}^{w} \sum_{y \in \mathcal{A}} \mathbb{P}(S_v = s_v \wedge S_{v,l} = y) \\ =& \prod_{l=1}^{w} \sum_{y \in \mathcal{A}} P_{s_v,y;k}\{t(S_{v,l}, S_v), \theta\}\mathcal{L}_{s_v}(\chi_i|y, \ell, \tau, \theta, k)\end{aligned}$$

The above is a formulation for $\mathcal{L}_v$ in terms of the children of v. Writing the likelihood in terms of $\mathcal{L}_v$ is called the Felsenstein pruning formulation.

Taking the subtree starting at the root, to define the likelihood sum over all possible rates $\sum_{k\in\gamma}$ and sum over characters the root can take $\sum_{x\in\mathcal{A}}$, the probability that the root takes that character $\mathbb{P}(\hat{\chi})(v_0) = x$ multiplied by the subtree likelihood $\mathcal{L}_{v_0}(\chi_i|x_v, \ell, \tau, \theta, \alpha)$. This gives the likelihood on a single site:

$$\mathcal{L}(\chi_i|\ell,\tau,\theta,\alpha) = \sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\pi(x)\left[\prod_{u\prec v_0}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v_0),\theta\}\mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]$$

Taking each site as independent gives the full likelihood.

The likelihood L_k in subsection 2.2.2 can now be defined as

$$\sum_{x\in\mathcal{A}}\pi(x)\left[\prod_{u\prec v_0}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v_0),\theta\}\mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right].$$

This is the recursive formulation of the likelihood. The likelihood can also be defined in terms of the Felsenstein algorithm, which takes advantage of the binary nature of a phylogenetic tree. For more details see appendix 1.

It should be clear that the location of the root does not affect the computation of the likelihood due to the reversibility of the Markov process.

Chapter 3

Bayesian Inference and Tree space

In likelihood based phylogenetic inference there are two main methods of using the likelihood to learn about the evolutionary process. These are the maximum likelihood and the Bayesian approaches. Maximum likelihood aims to find the single point in the parameter space which maximises the likelihood, it provides a best fit model. Bayesian approaches, on the other hand, allow the incorporation of prior information, summarised by a prior density over unknown parameters of interest. Upon observing data, the prior density can be combined with the likelihood function to give the posterior density, upon which we base parameter inference. Unfortunately, the posterior density is typically only available up to a constant of proportionality, except in some very simple cases. Consequently, posterior summaries such as (marginal) means and variances remain intractable. Nevertheless given samples from the posterior distribution, Monte Carlo methods can be used to approximate posterior moments as required. Of particular interest in this thesis is a commonly used technique known as Markov chain Monte Carlo (see section 3.3). In brief, a Markov chain is constructed whose equilibrium distribution is the posterior of interest. Simulations from this Markov chain are then taken as (dependent) draws from the posterior. In this chapter, we consider MCMC schemes for performing inference in Tree space. We therefore provide an introduction to Tree space before considering the inference problem and the construction of appropriate MCMC schemes. By far the most influential of this family of techniques is the Metropolis-Hastings algorithm.

3.1 Tree space

Tree space is complicated, so the MCMC methods need to be tailored to it.

Tree space $\mathcal{T}_n$ is defined to be the space of all edge-weighted phylogenetic trees including non-binary trees with n-leaves, labelled $1, ..., n$. This thesis is concerned with unrooted phylogenetic trees as we assume the Markov process is reversible. The number of different topologies, see Def: 3.1.1, of unrooted trees on n leaves is $(2n-5)!!$.

$$(2n-5)!! = (2n-5) \times (2n-3) \times ... \times 3 \times 1$$

For example there are 15 different topologies when there are 5 leaves. We will see later in figure 3.1 $\mathcal{T}_5$ modulo edge weights can be represented by the Petersen graph, a non-planar graph.

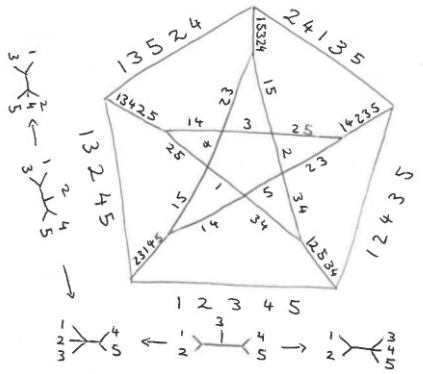


Figure 3.1: The Petersen graph encodes the different topologies in $\mathcal{T}_{\bigtriangledown}$. Up to permutation of labels there is one binary unlabelled branching tree with five leaves. This looks like an "H"-graph with an added edge in the middle. Each edge of the Petersen graph represents an orthant of Tree space as defined in subsection 3.1.1. Each edge of the Petersen graph has been given a string of numbers representing a labelling on the leaves of the tree. The first two numbers correspond to the leaves at one end of the H-graph, the middle number corresponds to the added leaf in the middle, and the last two correspond to the leaves at the other end of the H-graph. As a point travels within an edge of the Petersen graph, the internal edges of the tree represented by that point contract and expand. At corners where three edges of the Petersen graph meet one of the internal edges of the tree has contracted to a point. Around the left hand side of the diagram and bottom is depicted the tree associated with the left most and bottom edge. Also depicted are the trees at the corner. $\mathcal{T}_5$ is in fact the cone of the Petersen graph so each edge becomes an orthant with a vertex at the apex of the cone as can be seen in 3.5.

3.1.1 Phylogenetic Tree

A common assumption when conducting phylogenetic inference is that the evolutionary structure is tree-like, this model is common to both morphological and genetic data. A tree which is

designed to model evolutionary structure is called a phylogenetic tree.

A Phylogenetic Tree is an acyclic bifurcating graph in which every leaf represents an organism of interest. Each leaf has degree exactly one and the unique edge connected to a leaf is called its pendant edge. Each internal vertex has degree at least three with the sole exception of the root which, where there is one, has degree two. At every internal vertex there an act of speciation, one species divides into two. The root has degree two as it lacks a parent.

Phylogenetic Trees can either be weighted, (each edge has an associated length corresponding to evolutionary distance) or unweighted, (the tree merely provides structural information).

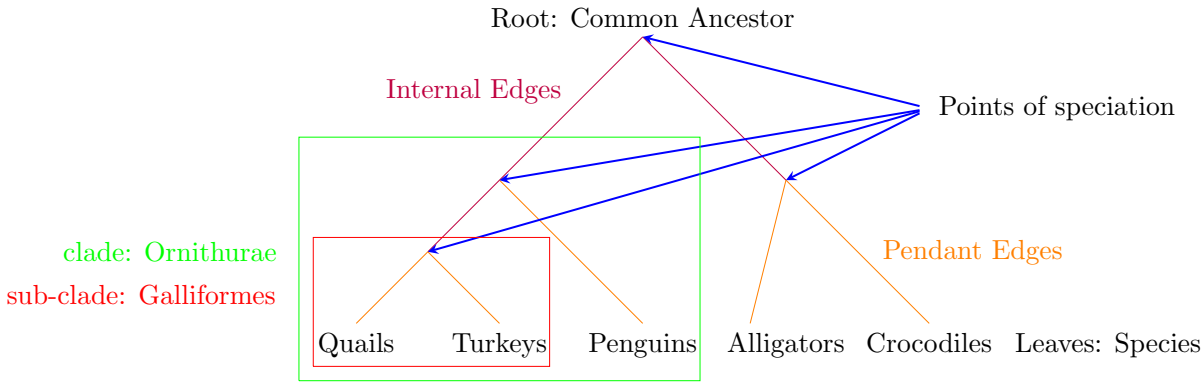


Figure 3.2: Example of a Phylogenetic Tree for birds and reptiles, here we have a striped down version of Figure: 1.1 where we focus on only a few species to understand the structure of a Phylogenetic tree. We see that a Phylogenetic tree has a root representing a common ancestor, internal edges representing unknown instances of speciation for which we have no data, and pendant edges which connect the modern species to some unknown shared ancestor. We also see that by disconnecting an internal edge we form two subtrees, partitioning all species into a clade representing the appropriate toxonomic rank at which the edge was cut.

Formally,

Definition (Phylogenetic Tree)**.** A phylogenetic tree is given by:

- a graph $T = (V, E)$ where V is a vertex set and $E \subseteq \binom{V}{2}$ a set of edges between pairs of vertices.
- a labelling $\varphi : V \to S$ where $S = S_{ant} \cup S_{inct}$ is a set of extant species for which information is known and extinct species for which information is unknown.

such that:

- $T = (V, E)$ is a tree, it is acyclic and connected,

- the labelling function φ is injective and is required to satisfy $\varphi(v) \in S_{ant}$ for all leaves v (vertices of degree 1), and $\varphi(v) \in S_{inct}$ for all internal nodes (vertices of degree greater than 1). *inct* stands for extinct as S_{inct} represents extinct species and *ant* stands for extant as S_{ant} represents extant species.

Two phylogenetic trees have the same topology when they are isomorphic to each other and the isomorphism preserves the labelling function φ.

One of the most important tools for comparing trees when they change topology is how an edge splits the extant species into two groups, this is called a tree split or split. A split is a useful tool because it can uniquely represent an edge in a way that does not depend on the topologies of the two trees formed when an edge is removed. This enables us to have a consistent notion of an edge when topology changes.

Definition (Split of a Phylogenetic Tree)**.** A split of a phylogenetic tree T is a disjoint pair of sets S_1, $S_2 \subseteq S$ obtained by removing an edge from the phylogenetic tree. As a phylogenetic tree is connected and acyclic, removing an edge results in two subtrees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$. The split is then the unordered pair of sets $S_1 = \varphi(V_1) \cap S_{ant}$ and $S_2 = \varphi(V_2) \cap S_{ant}$ and their related bases. We denote the split of edge e by $split(e) = (S_1, S_2)$.

An equivalent notion of topology is that two trees are topologically equivalent if the set of splits generated by removing edges from the trees are the same.

Definition (Topology of a Phylogenetic Tree)**.** The topology of a tree $T = (V, E)$ is the set of splits of the tree T:

$$\tau = \{split(e) | e \in E\}.$$

This definition means that two phylogenetic trees have the same topology if and only if they are isomorphic as labelled trees.

In the definition 3.1.1 a phylogenetic tree can be weighted or unweighted. If it is weighted each edge $e \in E$ has a weighting $\ell(e) \in \mathbb{R}$ associated to it, commonly called its length. As trees are acyclic, there exists a unique shortest path p between any two vertices, the length of this path, $\ell(p)$, is the sum of the length of all the edges in the path.

There are a few exceptions where DNA transfer between two different species does not occur in an acyclic branching pattern. Two species that diverged in the past can share genetic information. Certain bacteria can share DNA in the form of a plasmid ring due to processes such as recombination and conjugation. Methods of recombination and conjugation are well

understood and so the data in the alignment can be sampled to account for these effects.

Tree space is the space of all possible phylogenetic trees on n-leaves. Orthants are the subspace of phylogenetic trees restricted to particular topologies. Locations within orthants are defined by edge lengths. The orthants are connected to one another by gluing at the points where an internal edge has 0 length. When one edge is contracted this renders the topology of the tree indistinguishable between three possibilities, when more edges are contracted more orthants are connected by the codimensional point. We can observe this structure in "the open book" Figure 3.3.

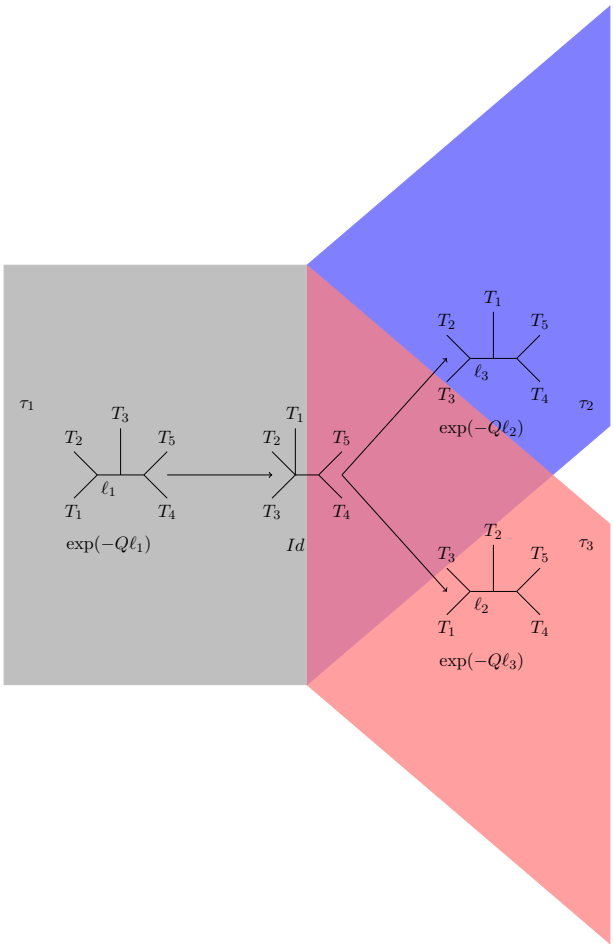


Figure 3.3: Here we can see an example of three different orthants (pages) of the open book and how they relate to one another. T_1, T_2, T_3, T_4, and T_5 are subtrees. Edges ℓ_1, ℓ_2, and ℓ_3 shrink to 0 to form the tree depicted on the spine, the intersection of the 3 orthants, labelled Id due to the associated transition probability across an edge of 0 length.

Definition (Orthant of Tree space)**.** An orthant of Tree space is the collection of trees belonging

to a single topology. The labelling on each tree is combinatorially the same but each tree in an orthant is distinct due to having different edge lengths. If an unrooted tree has n leaves, then it has $2n-3$ edges. Each orthant is isomorphic to a copy of $\mathbb{R}_+^{2n-3}$. Moving within an orthant corresponds to stretching and contracting edges.

Definition (Billera Holmes Vogtmann Tree space (BHV Tree space))**.** Billera Holmes and Vogtmann Tree space as proposed in Billera et al. (2001) consists of orthants glued together along codimensional lines. This occurs when the length of an edge is 0 in all orthants such that they are topologically indistinct from one another.

BHV Tree space can be thought of as the collection of trees, T denotes a generic tree, quotiented by an equivalence relation $\sim$. $T_1 \sim T_2$ if there is a correspondence between $e_1 \in E_1$ and $e_2 \in E_2$ preserving the length, and if the lengths of e_1 and e_2 are non-zero, the split of e_1 is the same as that of e_2. See Figure 3.3 for an example of this.

Definition (Codimensional boundaries)**.** The join between orthants consisting of n edges of 0 length has codimension n.

When dealing with HMC it is only necessary to consider codimension 1 spaces as the integrator almost surely does not pass through a codimension $n > 1$ point as can be seen in Theorem 7.2.

BHV Tree space is path connected and $CAT(0)$ when orthants are equipped with Euclidean metric on edge weights (Billera et al., 2001). A $CAT(0)$ space is curved such that the geodesic distance between points on a triangle is less than or equal to the geodesic distance in Euclidean space. This means that there exists a unique shortest path between any two trees in BHV Tree space called a geodesic (Billera et al., 2001). We will use these geodesics in computing distances and as a metric for the effective sample size on Tree space. In order to compute geodesics in Tree space we use an algorithm developed by Owen and Provan (2011) that computes the geodesic between two trees T and T', in polynomial time, see section 3.5.1.

Definition ($Cat(k)$ space)**.** A $Cat(k)$ space expresses how distance is transformed. A geodesic triangle is a collection of three points connected by geodesics. A space is $Cat(k)$ if for every geodesic triangle in it the distances between points on the triangle are less than or equal to the distances between points of a triangle with the same length sides in the simple connected surface with curvature k.

We show that BHV Tree space is a manifold stratified space or stratifold where codimensional boundaries form the different strata. There are several definitions of stratified manifold in the literature such as in Kreck (2010). We use the definition of a stratified topological space from Miller (2015).

Definition (Stratified Space)**.** A Topologically Stratified Space, Stratified Manifold or Stratifold. A topologically stratified space is a topological space, which is the disjoint union of strata $M_1, ..., M_l$

$$M = M_1 \cup M_2 \cup \cdots \cup M_l$$

where M_i are topological manifolds for all i. This satisfies two conditions:

1. $M_1 \cup \cdots \cup M_k$ is closed in M for all $k \leq l$.
2. For all points $x, y \in M_i$ there exists a homeomorphism $\phi : M \to M$ with $\phi(x) = y$ and $\phi(M_k) = M_k$ for all k.

We start with the example from Kreck (2010) as it provides intuition as to why Tree space is a stratifold.

Example 4.1:

Let $\mathcal{M}$ be an n dimensional manifold satisfying property 2. We construct the open cone of a manifold $\mathcal{M}$ in the following way. The open cone over $\mathcal{M}$ is $\mathring{CM} := \mathcal{M} \times [0, 1)/_{\mathcal{M}\times\{0\}}$.

We now show that open cone $\mathring{CM}$ is a stratifold consisting of two strata M_1 and M_2: M_1 is the point of the cone and M_2 is the rest. The strata M_1 is a point, depicted in red in Figure: 3.4, and trivially closed. $M_1 \cup M_2$ is the cone $\mathring{CM}$. To show that property 2 in definition 3.1.1 holds there we need to show it for two cases.

1. For M_1 when $k = 1$ we have $x = y$ and we can take ϕ to be the identity.
2. For M_2, $k = 2$, we write x and y as $x = x_m \times h_1$ and $y = y_m \times h_2$ in M_2. We can stretch the cone while fixing M_1 so that x overlaps y.

   Mathematically we create a map:

$$h \mapsto \begin{cases} h\frac{h_2}{h_1} \text{ if } h \in [0, h_1] \\ h_2 + \frac{(h-h_1)(1-h_2)}{(1-h_1)} \text{ if } h \in [h_1, 1) \end{cases}.$$

   This map maps 0 to 0, 1 to 1 and h_1 to h_2 in a continuous bijective manner and so is a homeomorphism. We can then pair this map with the homeomorphism that exists on the manifold mapping x_M to y_M to get a homeomorphism on $\mathring{CM}$ mapping M_1 to M_1.

This shows that the cone is a stratified manifold with two strata.

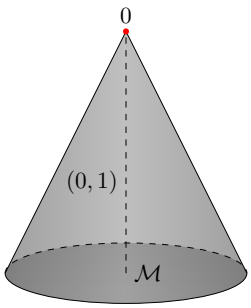


Figure 3.4: The two differenct strata of a cone of a manifold as described in example 4.1. We have the red stratum at 0 corresponding to M_1 and the rest of the cone corresponding to M_2. It from this we can see that the two partition the cone manifold CM into two strata.

From this example we can intuitively see that Tree space will be a stratified manifold with strata being orthants, and corners of codimension i, as can be seen in Figure 3.5.

Theorem 3.1. *Tree space is a stratified manifold.*

Proof. By definition there exists a partial order on Tree space. This is defined by tree $T_1 <$ tree T_2 if T_2 can be transformed into a tree T_2' in the same orthant of T_1,(isomorphic to T_1 up to edge length), by reducing 1 or more edges to length 0. We can then construct a sequence of strata based on this ordering. Starting with T_0 the tree such that all internal edges have 0 length, pick an internal edge with 0 length an extend it to get $T_1 > T_0$. Repeat this process to get $T_{n-3} > \cdots > T_1 > T_0$. Define M_i for $i = 0$ to $n-3$ be the collection of trees structurally isomorphic to T_i. Define M_{n-2} to be $\mathcal{T}_\setminus \setminus \bigcup_{i=0}^{n-3} M_i$. The union $\cup_{0\leq i\leq k} M_i$ is closed. We can see it is closed because any sequence of trees with at least $n-i$ edges of 0 length cannot tend to a tree where there are $n-i-1$ edges of 0 length in the limit. To satisfy 2. we define homeomorphisms ϕ between points x and y on M_i such that $\phi(M_k) = M_k$. The two points x and y are contained within a connected submanifold by definition. When x and y are contained within a connected submanifold of M_i, as we are operating on Tree space, this is homeomorphic to a copy of $\mathbb{R}_+^{i+n}$. Given $\mathbb{R}_+^{i+n}$ there is a trivial homeomorphic map f taking x to y, $f(m) \mapsto m\frac{y}{x}$. □

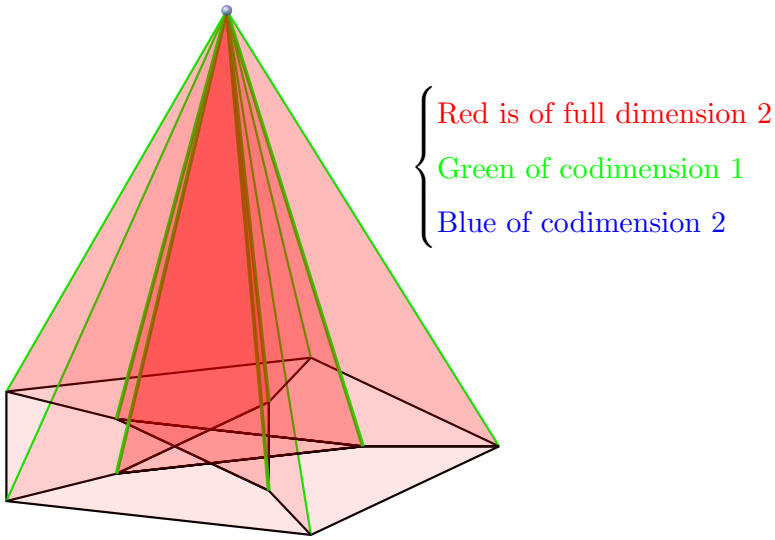


Figure 3.5: Different strata for $\mathcal{T}_5$, the strata are coloured, red represents the varying orthants and green are the boundaries over which CortHMC will cross.

In the following chapters we will see that we wish to define HMC on the cotangent space of Tree space, we do this in chapter 5.

3.1.2 Newick String Representation

It is necessary to write trees in a way that incorporates topological structure. The standard way is to use parantheses. A Newick string uses parantheses to denote subtrees, commas to denote siblings, colons to denote edge lengths and semi-colons to denote the end of the expression. We explain Newick strings through examples.

$$\text{“}(t5 : 0.73, ((t2 : 0.91, (t1 : 0.10, t3 : 0.43) : 0.18) : 0.50, t4 : 0.60) : 0.43);\text{”} \quad (2.s.f.).$$

This is the Newick string for the following tree in Figure 3.6.

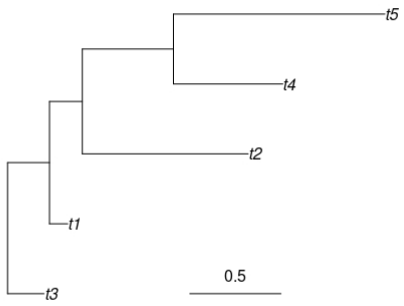


Figure 3.6: The alignment generating tree for 5 taxa.

The unrooted tree constructed from a Newick string is unique but the Newick string is dependent on the root of the tree. A Newick string represents the subtree structure by brackets, each pair enclosing a subtree. Each pair of brackets enclose two subtrees separated by a comma. The edge between two subtrees is denoted by a : followed by a weighting or length. In the above we start with two subtrees $t5 : 0.73$ and $((t2 : 0.91, (t1 : 0.10, t3 : 0.43) : 0.18) : 0.50, t4 : 0.60)$ separated by an edge of length 0.43. In Figure 3.6 the metric for computing the distance between nodes is the sum of the lengths of the horizontal edges between the two nodes. To understand how to write Newick strings let us construct a Newick string for Fig: 3.6 but with a new root 0.2 away from $t3$ on the pendent edge for $t3$. We start by opening the tree and splitting it into the subtree containing $t3$ and everything else noting that the choice of 0.2 is arbitrary.

$$(t3 : 0.2, (subtree) : 0.23);$$

We expand the subtree into two subtrees, one containing $t1$ and the other containing $t2$, $t4$ and $t5$

$$(t3 : 0.2, (t1 : 0.10, (subtree) : 0.18) : 0.23);.$$

Iterating this process along each subtree gives

$$(t3 : 0.2, (t1 : 0.10, (t2 : 0.91, (\ subtree\ ) : 0.50) : 0.18) : 0.23);$$
$$(t3 : 0.2, (t1 : 0.10, (t2 : 0.91, (t4 : 0.60, t5 : 1.16) : 0.50) : 0.18) : 0.23);.$$

3.2 Bayesian Inference

Bayesian inference uses Bayes' theorem to construct the posterior from the prior and the likelihood. We use the likelihood $\mathcal{L}$ constructed in section 2.5 with alignment $\mathbf{X}$ from subsection 2.3. We infer the substitution model parameters, site rate parameter and the tree from $\mathbf{X}$. We specify a prior density over edge lengths, substitution model parameters and the site rate parameter. Collectively denoted by π_0.

Given events Y_1 and Y_2 and probability function $\mathbb{P}$, Bayes' theorem states

$$\mathbb{P}(Y_1|Y_2) = \frac{\mathbb{P}(Y_2|Y_1)\mathbb{P}(Y_1)}{\mathbb{P}(Y_2)}.$$

Intuitively, we are often able to understand the effect of some hypothesis Y_1 on an event Y_2 (encoded by $\mathbb{P}(Y_2|Y_1)$), but wish to calculate the probability of the hypothesis having observed the event (encoded by $\mathbb{P}(Y_1|Y_2)$). Bayes' theorem gives a way of "flipping the conditional probability around". $\mathbb{P}(Y_1)$ represents our prior beliefs about Y_1, $\mathbb{P}(Y_2|Y_1)$ gives the likelihood of Y_1 and $\mathbb{P}(Y_1|Y_2)$ gives the posterior probability of Y_1. Bayes' theorem is easily extended to the inference problem in this thesis:

$$\pi(\theta, \alpha, \ell, \tau|\mathbf{X}) \propto \mathcal{L}(\mathbf{X}|\theta, \alpha, \ell, \tau)\pi_0(\theta, \alpha, \ell, \tau).$$

The above unnormalised posterior is typically straightforward to evaluate, but the full posterior remains intractable. This necessitates the use of computationally intensive sampling based approaches. We describe one such commonly used sampling approach in the next section.

3.3 Markov Chain Monte Carlo

Given an unnormalised probability density function $\hat{\pi}(z)$ where $z \in \mathbb{R}^D$, integrals involving $\hat{\pi}(z)$ are often intractable. This makes computing such things as the mean of $\hat{\pi}$ intractable. Nevertheless, key quantities of interest can be estimated, given by the ability to sample from $\pi(z)$ where $\pi(z)$ is the normalised density. For example, it should be clear that given draws $z_1, z_2, ..., z_n$ from $\pi(z)$ and some real valued function $g(\cdot)$, $n^{-1}\sum_{i=1}^{n} g(z_i)$ is a realisation of an unbiased estimator of $\mathbb{E}(g(z))$. The process of estimating integrals by random sampling is known as Monte Carlo integration. It remains that we can generate realisations from $\pi(z)$. Markov Chain Monte Carlo (MCMC) methods (see (Gamerman and Lopes, 2006) for an introduction) construct a Markov Chain $Z = \{z_i\}_{i=1}^{n}$ whose invariant density is $\pi(z)$. Hence, for n sufficiently large, we may regard draws of the Markov chain as dependent draws from $\pi(z)$. There are several

different MCMC algorithms, one of which is the Metropolis-Hastings algorithm, developed by Metropolis et al. (1953) and Hastings (1970) for a more general setting.

Metropolis-Hastings

The Metropolis-Hastings algorithm uses a carefully constructed acceptance ratio to ensure that the Markov chain induced by the algorithm is reversible and targets the distribution of interest. It also ensures aperiodicity. It requires the construction of a proposal distribution with the same support as $\pi(\cdot)$ that does not segment the space, creating a reducible sequence. We allow for proposing local moves and let $q(\hat{z}, z)$ denote the proposal density, where z denotes the current state of the chain. The Metropolis-Hastings algorithm is given below.

Algorithm 1: Metropolis-Hastings

- Input: unnormalised target density π, starting point z_0 for the Markov chain (see section 2.1), proposal distribution q and chain length n.
- for i in $1:n$
  1. Draw $\hat{z} \sim q(\cdot, z_{i-1})$.
  2. Compute $a = \min\left[1, \frac{\pi(\hat{z})q(z_{i-1},\hat{z})}{\pi(z_{i-1})q(\hat{z},z_{i-1})}\right]$.
  3. Draw $u \sim U(0,1)$
  4. If $u \leq a$ set $z_i = \hat{z}$
     else set $z_i = z_{i-1}$.
- Output: chain $\{z_i\}_{i=0}^n$ with invariant distribution π.

The proposal density $q(\cdot,\cdot)$ should be chosen so that the chain moves often and around the parameter space. A typical choice of q proposal is the random walk with normal innovations. In this case the proposal mechanism is:

$$\hat{z} = z + w_i, \qquad w_i \sim N(0, \Sigma)$$

where Σ is a matrix specified by the practitioner. Roberts and Tweedie (1996) suggest using the heuristic for $\Sigma = \frac{2.38^2}{D} \times \hat{Var}(Z)$ where $\hat{Var}(Z)$ is the estimated posterior variance from an initial pilot run for Σ, D is the dimension for Z . Σ can be further scaled to give an acceptance rate around $20\% - 30\%$.

In scenarios where the full conditional distributions of the components of z are available for sampling from, an automatic scheme is possible.

Gibbs Sampler

The Gibbs sampling algorithm, proposed by Geman and Geman (1984) and developed in Gelfand and Smith (1990) uses the full conditional distributions of z to construct a chain with the appropriate invariant distribution.

Let $\pi_j(z^1, ..., z^{j-1}, z^{j+1}, ..., z^D)$ denote the full conditional density of component j of z of $z = (z^1, ..., z^D)$.

Algorithm 2: Gibbs Sampler

- Inputs: unnormalised target density π with marginal distributions $\pi_i(\cdot|z^1, ..., z^{j-1}, z^{j+1}, ..., z^D)$, starting position $z_0 = (z_0^1, ..., z_0^D)$ and chain length n.
- for i in $1 : n$
  - For j in $1 : D$
    - Draw $\hat{z}_i^j \sim \pi_j(\cdot|z_i^1, ..., z_i^{j-1}, z_i^{j+1}, ..., z_i^n)$.
  - Set $z_{i+1} = (\hat{z}_i^0, ..., \hat{z}_i^D)$.
- Output: chain $\{z_i\}_{i=0}^n$ with invariant distribution π.

Metropolis-Hastings and Gibbs sampling are two of the most common techniques employed in phylogenetic inference. Often Metropolis-Hastings is used within Gibbs. Software such as MrBayes and Beast perform highly optimised versions of the Metropolis-Hastings and Gibbs algorithms in order to conduct phylogenetic inference.

Further details along with proofs about the mathematical properties of Metropolis-Hastings algorithm and the Gibbs sampling algorithm can be found in Johansen and Evers (2007).

Metropolis within Gibbs

Gilks et al. (1995) proposed an algorithm that applies a Metropolis-Hastings step to the full conditional distributions $\pi_i(\cdot|z^1, ..., z^{i-1}, z^{i+1}, ..., z^D)$ when they can't be sampled from directly. This means that this is an example of a rejection sampling chain.

Algorithm 3: Metropolis-Hastings within Gibbs or the Adaptive Rejection Metropolis Sampling algorithm

- Inputs: unnormalised target density π with marginal distributions $\pi_j(\cdot|z^1, ..., z^{j-1}, z^{j+1}, ..., z^D)$, starting position $z_0 = (z_0^1, ..., z_0^D)$, proposal distributions $\{q_i\}_{i=1}^D$ and chain length n.
- for i in $1 : n$
  - For j in $1 : D$
    - Draw $\hat{z}^j \sim q_j(\cdot; z_{i-1})$.
    - Compute $a = \min\left[1, \frac{\pi_j(\hat{z}^j|z_{i+1}^1,...,z_{i+1}^{j-1},z_i^{j+1},...,z_i^D)q_j(z_{i-1}^j;z_{i+1}^1,...,z_{i+1}^{j-1},z_i^{j+1},...,z_i^D)}{\pi_j(z_i^j|z_{i+1}^1,...,z_{i+1}^{j-1},z_i^{j+1},...,z_i^D)q_j(\hat{z}^j;z_{i+1}^1,...,z_{i+1}^{j-1},z_i^{j+1},...,z_i^D)}\right]$.
    - Draw $u \sim U(0, 1)$
    - If $u \leq a$ set $z_{i+1}^j = \hat{z}^j$ else set $z_{i+1}^j = z_i^j$.
- Output: chain $\{z_i\}_{i=0}^n$ with invariant distribution π.

This enables us to construct more streamlined proposals when inferring over multiple parameters.

3.4 MCMC in Tree space

3.4.1 Tree Operations

In order to conduct MCMC in Tree space, different topologies need to be proposed. To do so it is common to employ one of three different topological operations as covered in Allen and Steel (2001). These are Nearest Neighbour Interchange (NNI), Subtree Prune and Regraft (SPR) and Tree Bisection and Reconnection (TBR). In PhyloCore a method of Metropolis within Gibbs is used using NNI and SPR and a log random walk on edge lengths (Nye and Heaps, 2014).

NNI

Given an internal edge $e = (u, v)$, u and v have degree 3 and so are connected to 2 other vertices meaning that e is connected to 4 vertices. Each of these 4 vertices then define a subtree consisting of all vertices and edges beneath them if the tree is rooted at u or v. An NNI interchange swaps two of these 4 subtrees, which means that there are a total of 3 different trees possible around an edge. The new edge often has random length.

SPR

Given an edge $e = (u, v)$, cut the edge e to create two subtrees t_1, t_2. Regraft t_2 to t_1 by the cut edge to a pre-existing random edge in t_2. Then apply a forced contraction. The regrafted edge often has random length and is regrafted a random length along the random edge in t_2.

TBR

Given an edge $e = (u, v)$, cut the edge e to create two subtrees t_1, t_2. Regraft t_2 to t_1 by creating an edge between the middle of a random edge in t_1 and the middle of a random edge in t_2. The regrafted edge often has random length and is regrafted a random length along the random edge in t_1 and t_2.

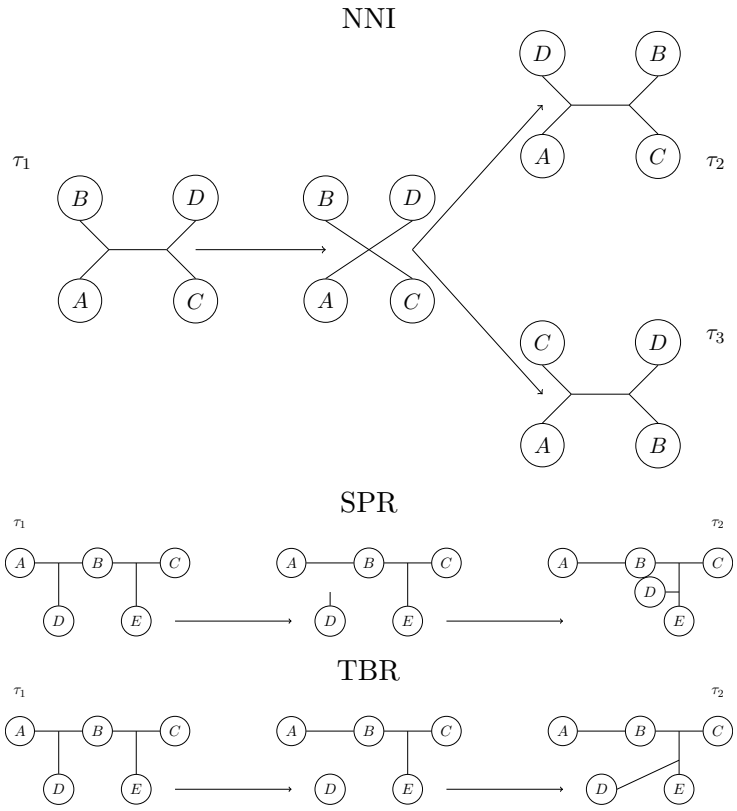


Figure 3.7: Here we can see the three different tree operations, (NNI,SPR and TBR) for switching topology from τ_1 to τ_2 or τ_3. In NNI, subtree A is interchanged with subtree C or D to produce one of two new topologies. In SPR, subtree D it cut off the tree and reattached to a new location in the tree by an edge of the same length as before and connected to the same point in subtree D as before. In TBR the edge is deleted and a new edge is introduced between subtree D and the tree connecting a random location of the tree to a random location within D.

3.5 Assessing MCMC for phylogenetic inference

Several techniques exist for analysing the effectiveness of various inference schemes. The two we shall focus on are the effective sample size and a topological measure of convergence. These will be applied to MCMC and HMC samplers to compare the behaviour of different inference algorithms.

3.5.1 Effective Sample Size

The effective sample size measures the loss of information caused by positive correlation within the Markov chain Z_t. The variance of an estimator obtained using samples from a positively correlated chain is larger than that of an i.i.d. chain. It can be shown that for a chain of length n and measurable function f that:

$$Var(\frac{1}{n}\sum_{\tau=1}^{n} f(Z_\tau)) = \frac{\sigma^2}{n}\left(1 + 2\sum_{\tau=1}^{n-1}(1 - \frac{\tau}{n})\rho(\tau)\right)$$

where $\rho(\tau)$ is the autocorrelation of Z_t with lag τ (Johansen and Evers, 2007). The standard deviation σ is the standard deviation of $f(Z_t)$. Taking the limit as $n \to \infty$ gives:

$$nVar(\frac{1}{n}\sum_{\tau=1}^{n} f(Z_\tau)) \to \sigma^2\left(1 + 2\sum_{\tau=1}^{\infty}\rho(\tau)\right)$$

under a bounded condition for the sum of the autocorrelations. We use this to give the standard form for the effective sample size. First note that if the samples are i.i.d.

$$Var(\frac{1}{n}\sum_{i=1}^{n} f(Z_i)) = \frac{\sigma^2}{n}.$$

As $Var(\frac{1}{n}\sum_{i=1}^{n} f(Z_i))$ is fixed in both we can equate the two to give

$$\frac{\sigma^2}{n_{eff}} = \frac{\sigma^2}{n}\left(1 + 2\sum_{\tau=1}^{\infty}\rho(\tau)\right).$$

This gives us the effective sample size n_{eff}.

Definition (Effective Sample Size (ESS))**.** The effective sample size is the approximate number of effective samples given by

$$n_{eff} = \frac{n}{1 + 2\sum_{\tau=1}^{n}\rho(\tau)}.$$

Definition (Autocorrelation with lag τ)**.** The autocorrelation with lag τ of chain Z_t is defined as:

$$\rho(\tau) = \frac{\mathbb{E}((Z_{t+\tau} - \mu)(Z_t - \mu))}{\sigma^2} = \frac{\mathbb{E}((Z_{t+\tau} - \mu)(Z_t - \mu))}{\mathbb{E}((Z_t - \mu)(Z_t - \mu))}$$

where $\mu = \mathbb{E}(Z_t)$.

There are many different ways of approximating the autocorrelation, some methods use spectral decomposition, others simply compute the expectation of the chain so far. We opt to use the effective sample size computed by the package Coda, which uses spectral decomposition (Plummer et al., 2006) to compute the effective sample size.

As trees are dependent on multiple parameters it is not clear which marginals of the chain to use to compute the effective sample size. A common method is to use minESS, the minimum ESS across all the different parameters. This would mean the minimum effective sample size across all edges, substitution model parameters and site rate parameters. When conducting reverse jump MCMC, as parameters are not comparable, a summary statistic is used, for example the likelihood.

We argue that using minESS is not appropriate for phylogenetic inference. Under minESS a chain of trees could explore a wide range of topologies, and have widely fluctuating edge lengths in all but one edge, and be scored the same as a chain of trees that does not exhibit such variation. To get around this issue we developed a new form of effective sample size for phylogenetic trees.

Intrinsic Effective Sample Size

The effective sample size should aim to be an estimator for every parameter. To do this we notice that the standard effective sample size for a parameter, $\mathbb{E}((Z_{t+\tau} - \mu)(Z_t - \mu))$, is $\mathbb{E}(d(Z_{t+\tau}, \mu)d(Z_t, \mu))$ where d is Euclidean distance. We posited that instead of using Euclidean distance we could use the distance intrinsic to Tree space created by the geodesics between trees. This would at least enable some collation of the data. We therefore approximate ρ by:

$$\hat{\rho}(\tau) = \frac{\mathbb{E}\left(d(Z_{t+\tau}, \mu)d(Z_t, \mu)\right)}{\mathbb{E}\left(d(Z_t, \mu)d(Z_t, \mu)\right)}$$

where d is either the Billera-Holmes-Vogtmann distance or the Robinson-Foulds metric Nye et al. (2005).

Definition (Billera-Holmes-Vogtmann distance)**.** The Billera-Holmes-Vogtmann distance is the distance defined in Billera et al. (2001) and outlined there in section 4.2. It is defined as the length of the geodesic between the two trees.

The Robinson-Foulds metric does not take into account edge-lengths so should be avoided when this is an important part of the inference.

3.5.2 An alternative Effective Sample Size

The Stan Development Team (2019) provide an alternative form of ESS estimation, requiring multiple runs of HMC. If there are M runs and W is the estimate for the within sample variance and V the estimate for the variance across all the chains and $\hat{p}_m(\tau)$ is the standard autocorrelation estimate on chain m,

$$\hat{p}_m(\tau) = \frac{1}{N-\tau} \sum_{n=1}^{N-\tau} (f(Z_n) - \mu)(f(Z_{n+\tau} - \mu)$$

calculated using a fast fourier transform.

$$\hat{p}(\tau) = 1 - \frac{W - \sum_{m=1}^{M} \hat{p}_m(\tau)}{V}$$

It should be clear to see when $M = 1$ this agrees with the standard definition. This method is useful because it attempts to account for chains that have not fully converged.

3.5.3 Convergence

To check convergence a technique from PhyloBayes is employed. It runs two chains and compares split frequences computing the maxdiff and meandiff. If the maxdiff is less than 0.1 then convergence is said to be "good" (Nye and Heaps, 2014). the maxdiff is the maximum difference between relative split frequencies.

It is also common to compute the average standard deviation of split frequencies, the closer to 0 this value the more the chain has converged (The Stan Development Team, 2019).

Chapter 4

Hamiltonian Monte Carlo

This chapter covers the theory behind Hamiltonian Monte Carlo (HMC) and describes the standard algorithm used to sample from a given target distribution. Various forms of HMC require different operational parameters and this chapter will discuss which forms to use and how to optimise the parameters involved. This chapter also covers several extensions to HMC which can improve its performance, but also add to the total computational complexity.

4.1 Introduction

4.1.1 History

Hamiltonian Monte Carlo (HMC) was introduced by Duane et al. (1987), and the theory has been further developed by Neal et al. (2011), Girolami et al. (2011) and Hoffman and Gelman (2014) among many others. A package called Stan (The Stan Development Team, 2019) has been implemented, which enables users to perform Bayesian inference using HMC in a user friendly manner on a range of models. However due to the stratified nature of tree space, HMC requires considerable adaptation. At present Stan cannot operate on phylogenetic trees.

4.1.2 Concept

Hamiltonian Monte Carlo adapts the standard Metropolis-Hastings algorithm. It generates random draws from posterior $\pi(z)$, known up to a constant of proportionality. It generates draws from an augmented posterior $\pi(z, p)$, with augmented variables p, that when marginalised over p gives $\pi(z)$:

$$\int \pi(z, p)dp = \pi(z).$$

The augmented posterior $\pi(z,p)$ is constructed to make use of properties of Hamiltonian systems. It relates the posterior with the Hamiltonian by $\pi(z,p) \propto \exp(-H(z,p))$ where $H(z,p)$ is the Hamiltonian, a smooth function of z and p. In standard HMC the parameters z are taken to lie in a manifold $\mathcal{M}$ and the augmented parameters p are taken to lie in its cotangent space at z, $T_z^*\mathcal{M}$. HMC is therefore defined on the cotangent bundle $T^*\mathcal{M} = \sqcup_{z\in\mathcal{M}} T_z^*\mathcal{M}$, which is the disjoint union of cotangent spaces over $\mathcal{M}$; see 4.2.1 for definitions of manifolds and the surrounding theory. A physical interpretation for (z,p) is a position momentum pair upon which the Hamiltonian acts. The Hamiltonian $H(z,p)$ defines a vector field on $T^*\mathcal{M}$. We will demonstrate how the vector field defines flows $\phi_H : \mathbb{R} \times T^*\mathcal{M} \to T^*\mathcal{M}$ that conserve the Hamiltonian along the flow i.e.

$$H(z,p) = H(\phi_H(t,z,p)), \text{ for all times } t \text{ and } (z,p) \in T^*\mathcal{M}.$$

As a result of the Hamiltonian construction, whenever $H(z,p) = H(z',p')$, $\pi(z,p) = \pi(z',p')$. Flows therefore conserve the augmented density: $\pi(z,p) = \pi(\phi_H(t,z,p))$. This is the property that enables large moves in the target space.

A single iteration of HMC can be summarised as follows:

1. The momenta p are randomly sampled from some distribution, which enables the algorithm to switch between different flows.
2. The position momenta pair (z,p) then becomes the starting point for the flow along which to integrate. A new pair $(z',p') = \phi_H(t,z,p)$ is proposed for some $t > 0$.
3. A Metropolis-Hastings accept-reject step is applied to (z',p') to ensure that the chain has the correct stationary distribution. It is important to note that under perfect integration there would be no integration error and so the probability of acceptance would equal 1.

Notation: Suppose $P(y_1)$ is a distribution dependent on y_1 then we shall denote the density at y, $P(y_2; y_1)$ and draw y_2 from $P(\cdot; y_1)$. For example we shall write the normal distribution $N(\mu, \sigma^2)$, draw from $N(\cdot; \mu, \sigma^2)$ and the density of y_2 is $N(y_1; \mu, \sigma^2)$.

Example 5.1.4: Consider a normal target distribution $N(0,1)$ so that $\pi(z) = N(z; 0, 1)$. Augment $\pi(x)$ with momenta $p \in \mathbb{R}$ such that

$$\pi(z,p) = \exp(-z^2/2)\exp(-p^2/2)/(2\pi).$$

It is easily checked that $\int_{-\infty}^{\infty} \exp(-z^2/2)\exp(-p^2/2)/(2\pi)\, dp = \exp(-z^2/2)/\sqrt{(2\pi)}$.

In this example the Hamiltonian is, up to additive constant, $H(z,p) = z^2/2 + p^2/2$. In the physical interpretation, H is an energy function. In this example $(z^2/2)$ is the potential energy, and $(p^2/2)$ is the kinetic energy. This Hamiltonian represents simple harmonic motion.

4.1.3 Requirements and uses

As integral curves of Hamiltonian flows need to be computed, HMC requires that the posterior is differentiable. It also requires a means of computing the integral curves. HMC is also a more computationally complex method than standard MCMC and so to be beneficial it is typically conducted on models with large numbers of parameters or posteriors with strong gradients. This is due to HMC relying heavily on the underlying geometry of the space of parameters and exploiting it to enable efficient sampling from the parameter space. It is for the above two reasons HMC might be suited to phylogenetic inference, which has an extremely large parameter space, in which the posterior density can vary very substantially from region to region.

4.2 Theory

4.2.1 Manifolds and Notation

HMC employs a lot of technical mathematics, much of which only requires a cursory understanding. For a more in-depth review of manifolds please see Lee (2013). This section is meant to refresh the reader, provide notation and highlight how manifolds will be used.

A topological manifold $\mathcal{M}$ is a Hausdorff, second-countable, locally Euclidean space. A smooth manifold is a manifold in which all transition maps between charts are diffeomorphisms. Examples of manifolds include: orthants of tree space, $\mathbb{R}^n$, $\mathbb{S}^n$ and the n-dimensional simplex

$$\Delta = \{\bar{z} | \sum_{i}^{n} z_i = 1\}.$$

A Riemannian metric g on a smooth manifold $\mathcal{M}$ is a smooth symmetric covariant 2-tensor field that is positive definite at each point. It is common to write $g = g_{ij} dz^i \otimes dz^j$. The matrix $G = \{g_{ij}\}$ is then positive definite.

Definition (Covariance Matrix)**.** The covariance matrix for a random vector $Z = (Z^1, ..., Z^D)$ is given by:

$$[Cov(Z)]_{ij} = \mathbb{E}\{(Z_i - \mathbb{E}(Z_i))(Z_j - \mathbb{E}(Z_j))\}.$$

Every positive semi-definite matrix can be thought of as the covariance matrix for some random vector.

Example 4.2.1 Euclidean Space Euclidean space of dimension n is $\mathbb{R}^n$ and has inner product $\langle y_1, y_2 \rangle = y_1 \cdot y_2 = y_1^t Id_n y_2$. This inner product creates the metric $g = \{Id_n\}_{ij} dz^i \otimes dz^j$, which means that the change $\langle y_1 + \delta, y_1 \rangle = \sum_{i=1}^n \delta_i$. We know that Id_n is positive semi-definite so it is a covariance matrix for the random vector $Z \sim N(0, Id_n)$.

Example 4.2.1 Generic Riemannian Manifold A Riemannian manifold is a a pair $(\mathcal{M}, g)$ where $\mathcal{M}$ is a smooth manifold and g is a Riemannian metric. It can be thought of as similar to Euclidean space, but where the inner product $\langle y_1, y_2 \rangle = y_1^t \{g_{ij}\} y_2$. As with the Euclidean case $\{g_{ij}\}$ is positive definite and so defines a covariance matrix for a multivariate normal random variable $Z \sim N(0, G^{-1})$.

We write $\frac{\partial}{\partial z_i}$ as the basis for the tangent space at point z. The tangent space $T_z\mathcal{M}$ consists of linear maps mapping smooth functions on the manifold to $\mathbb{R}$, satisfying the product rule. Members of the tangent space can be written as vectors $v = v^i \frac{\partial}{\partial z_i}|_z$. In Figure 4.1 we visualise the tangent space at a point z by a blue plane and members of the tangent space as lines within the plane. The tangent bundle $T\mathcal{M}$ is the disjoint union of all tangent spaces.

The cotangent space at a point $T_z^*\mathcal{M}$ is the dual space to $T_z\mathcal{M}$. The cotangent space consists of linear maps mapping members of the tangent space to $\mathbb{R}$. We write dz_i as a basis for the cotangent space. For $w = w_i dz_i \in T_z^*\mathcal{M}$ we write $w(v) = \left(v^i w_i\right)$. In Figure 4.1 we try to visualise the cotangent space as a red plane so that any two linear maps within the tangent space and the cotangent space can interact to get a point in $\mathbb{R}$ represented by $p(v)$. The cotangent bundle is the disjoint union of the cotagent spaces at points $z \in \mathcal{M}$.

We have stated in section 4.1.2 that HMC acts on the cotangent space, or really the cotangent bundle. We also call p the momenta. Relating these terms back to a more familiar setting we can think of the manifold as a smooth moulded piece of clay and z as a particle on the surface of the clay that cannot leave the surface. We understand the concept of velocity v for the particle at z and that is a member of the tangent plane as you can only specify direction and magnitude. We relate this to our definition by thinking how this speed acts on smooth functions of z. In time δt, z becomes $z + v\delta t$. Smooth function $f(z)$ becomes $f(z + v\delta t) = f(z) + f'(z)v\delta t + O((v\delta t)^2)$ and smooth function $f(z)g(z)$ where f and g are smooth functions becomes $f(z + v\delta t)g(z + v\delta t) = f(z)g(z) + (f'(z)g(z) + f(z)g'(z))v\delta t + O((v\delta t)^2)$.

Momenta is related to velocity by the equation $p = mv$. To understand how momenta is a covector we need to know about the Lagrangian. The Lagrangian is the addition of kinetic energy and potential energy $L = K.E. - P.E.$. We can take the derivative of the Lagrangian with respect to velocity. This gives $\frac{dL}{dv} = \frac{dK.E.}{dv} = \frac{d}{dv}\frac{1}{2}mv^2 = mv = p$. Clearly p is linear and it maps v to the derivative of L with respect to v which takes values in $\mathbb{R}$.

The smooth vector field of a function $f \in C^\infty(\mathcal{M})$, ψ_f, is defined as the inverse of an isomorphism from the tangent to cotangent, applied to the differential of f.

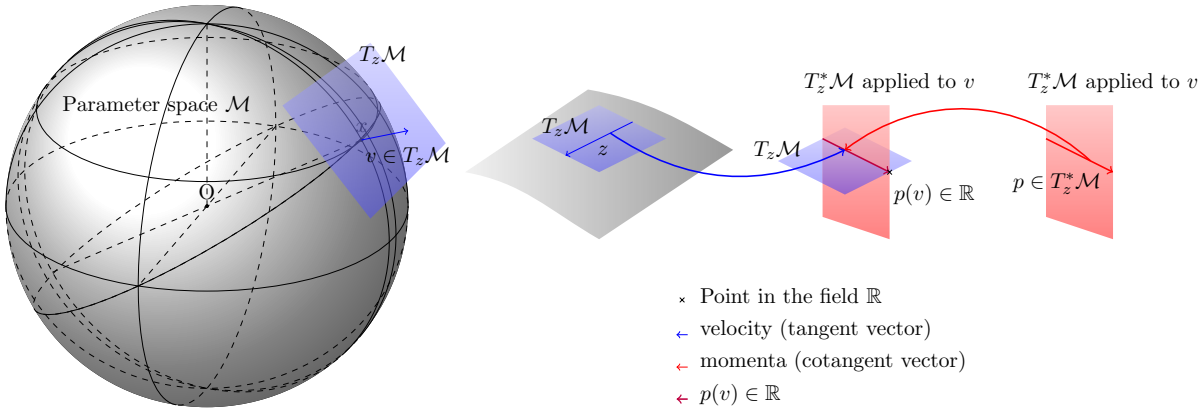


Figure 4.1: Here we can see a maninfold, the surface is curved but still has a tangent plan and a cotangent which can be applied to the tangent plane, here the sphere is the parameter space, the velocity is the tangent and the momenta, the cotangent.

4.2.2 Introduction to Hamiltonian Dynamics

A Hamiltonian system is a symplectic manifold $\mathcal{N}$ paired with a smooth function $H \in C^\infty(\mathcal{N})$. In HMC $\mathcal{N}$ will always be the cotangent bundle of a manifold $\mathcal{M}$ and so $\mathcal{N} = T^*\mathcal{M}$. The Hamiltonian H maps $T^*\mathcal{M} \to \mathbb{R}$. The Hamiltonian defines a vector field ψ_H on $T^*\mathcal{M}$ and this in turn defines flows $\phi_H(t, \cdot)$ around the field. In physics sometimes the function H can be dependent on time. This is equivalent to defining $\mathcal{M}' = \mathcal{M} \times \mathbb{R}$ and Hamiltonian $H(z', p')$ for $(z', p') \in T^*\mathcal{M}'$ such that it is constant in the time component of the momenta. However for HMC it is only necessary to allow H to be defined on z and p.

We now wish to relate the augmented posterior π to the Hamiltonian. The Hamiltonian is defined from the augmented space by:

$$H(z, p) = -\log\left(c\pi(z, p)\right) \text{ where } c \text{ is some normalising constant.}$$

so that :

$$\pi(z, p) = \frac{\exp\left(-H(z, p)\right)}{c}$$

The Hamiltonian vector field on $T^*\mathcal{M}$ is defined as:

$$\psi_H = \sum_{i=1}^{n} \left(\frac{\partial H}{\partial p^i} \frac{\partial}{\partial z^i} - \frac{\partial H}{\partial z^i} \frac{\partial}{\partial p^i} \right). \tag{1}$$

If we let $z = z(t)$ and $p = p(t)$ so that the Hamiltonian takes in values dependent on t, then taking the flow along a vector field $V_H = \left(\frac{dz(t)}{dt}, \frac{dp(t)}{dt}\right)$ evaluated at $(z(t), p(t))$ gives Hamilton's equations:

$$\left(\frac{dz}{dt}, \frac{dp}{dt}\right) = \left(\frac{\partial H}{\partial p^i}|_{(z(t),p(t))}, -\frac{\partial H}{\partial z}|_{(z(t),p(t))}\right). \tag{2}$$

The Hamiltonian flow $\phi_H : \mathbb{R} \times T^*\mathcal{M} \to T^*\mathcal{M}$ is defined as the solution to the initial value problem:

$$\frac{d\phi_H(t, z(t), p(t))}{dt} = \left(\frac{dz}{dt}, \frac{dp}{dt}\right) = \psi_H, \qquad \phi_H(0, z(0), p(0)) = (z(0), p(0)). \tag{3}$$

The object $\frac{d\phi_H(t,z(t),p(t))}{dt}$ lives in the tangent space of the cotangent space $T^*\mathcal{M}$.

A flow is a family of diffeomorphisms indexed by time t in an interval of $\mathbb{R}$. A flow has

two key properties:

1. $\phi_H(0,(z,p)) = (z,p)$, for all $(z,p) \in T^*\mathcal{M}$ and
2. $\phi_H(t,\phi_H(s,(z,p))) = \phi_H(t+s,(z,p))$ for all $s,t \in \mathbb{R}$, $(z,p) \in T^*\mathcal{M}$.

Lemma 4.1. *The Hamiltonian flow ϕ_H satisfies conditions* 1. *and* 2.

Proof. Condition 1 follows immediately from the definition. To show condition 2 first consider a flow on $\mathcal{M} = \mathbb{R}^n$:

$$\begin{aligned}
\phi_H(t+s,(z(0),p(0))) &= \int_0^{t+s} \frac{d\phi_H(u,z(u),p(u))}{du} du + (z(0),p(0)) \\
&= \int_s^{t+s} \frac{d\phi_H(u,z(u),p(u))}{du} du + \int_0^{s} \frac{d\phi_H(u,z(u),p(u))}{du} du + (z(0),p(0)) \\
&= \int_s^{t+s} \frac{d\phi_H(u,z(u),p(u))}{du} du + (z(s),p(s)) \\
&= \phi_H(t,\phi_H(s,(z,p))).
\end{aligned}$$

To prove this result on a manifold, a sequence of overlapping charts covering the flow from z,p at time zero to $\phi_H(t+s,z,p)$ is considered. The above result on $\mathbb{R}^n$ applies within each chart, and the charts are stitched together to prove the general result. □

Hamiltonian vector fields have two key properties. Equations (1) and (2) imply that H is constant along integral curves of V_H and that the Hamiltonian vector field is tangent to the level set of H, as can be seen in Figure 4.2. This means that integrating along the Hamiltonian flow stays on the same integral curve and maintains H.

Example 5.1.4 cont.: The Hamiltonian is

$$H(z,p) = -\log(\exp(-z^2)\exp(-p^2)/(2\pi)) = z^2 + p^2 + \log(2\pi)$$

resulting in a vector field

$$\psi_H = \left(2p\frac{\partial}{\partial z} - 2z\frac{\partial}{\partial p}\right).$$

We can see in Figure 4.2 the simple pendulum has this Hamiltonian. For small oscillations of z the level sets of H are circles in the z,p plane, called the phase portrait. This should not be confused with the path of the pendulum. For larger values of z, $h = l - lcos(z)$ resulting in a more complicated Hamiltonian and level sets that aren't circles and for large enough values of p can span the whole of $\mathbb{R}$ in the z dimension.

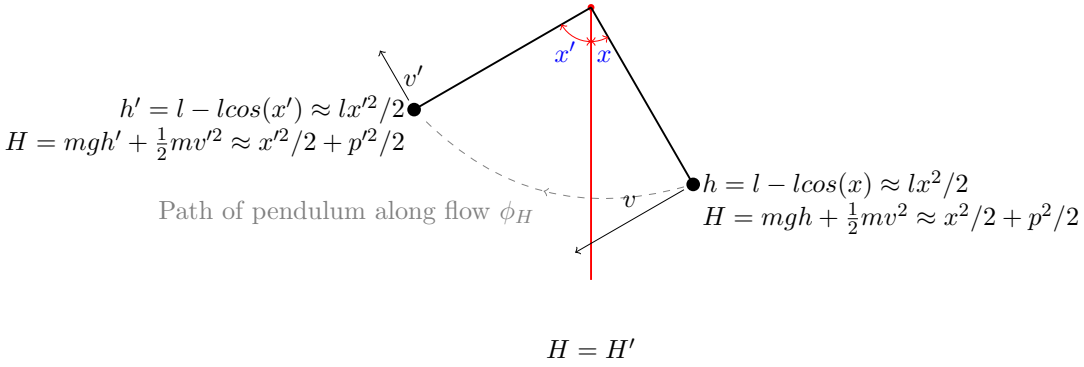


Figure 4.2: The Hamiltonian of a Pendulum for small z. Here the same energy state is depicted but with two difference momenta and heights.

In our example the posterior we wish to recover is a normal density. In subsection 4.1.2 we outline the method HMC follows. As we have seen before we can recover the density of a normal distribution from the Hamiltonian. We construct the Hamiltonian as specified below. The flows of the Hamiltonian then correspond to the points along the trajectory of the pendulum. HMC then samples from the posterior by choosing a starting point for the pendulum and giving it a little random push, this specifies z and p. The pendulum then swings for a set amount of time. This is the same as integrating along the flow for a set amount of time. This returns a new position $\hat{z}$ and momenta $\hat{p}$. These new points are then accepted according to the HMC acceptance probability. The process is then repeated creating a chain of different positions z. This chain is normally distributed for small oscillations. It should be clear to see that z will be distributed symmetrically around 0. In fact z will be distributed exactly as a normal distribution but the symmetrical properties and the decrease in probability for large values of z should be evident.

There are many different ways to augment $\pi(z)$ to $\pi(z, p)$ as in subsection 4.1.2. The pendulum gives us an intuition for how to augment. For a given z the momenta p determine which energy level and equivalently integral curve to flow along. This means that the augmentation should be constructed so that it does not exclude flows. In the pendulum example excluding flows would mean never proposing momenta large enough for the pendulum to reach a full 360° of rotation. The Hamiltonian is different for large oscillations, but the analogy holds. It also should be clear that a symmetrical augmentation is preferable, suppose positive momenta were preferred, or had a greater variance, then extreme clockwise rotations would be favoured more than extreme anticlockwise rotations.

The density function for momenta conditional on position is often written as $K(p;z)$ or $K(z,p)$, due to correspondence with kinetic energy. For consistency we shall use $K(p;x)$. It is common while working on $\mathbb{R}^n$ to choose a multivariate normal centred around the origin for $K(p;x)$. The key parameter that a multivariate normal centred around the origin requires is the covariance matrix, which we shall denote G. There are a few reasons for this. The density of the normal distribution is smooth, which both means that it has a derivative and that numerical integrators are more accurate (Dinh et al., 2017). On average it doesn't favour a particular part of the parameter space and it can propose any possible momenta, which not only means that the Markov chain produced is aperiodic but also irreducible.

It is important to give some intuition as to why we define the Hamiltonian so that $\frac{\exp(-H)}{c} = \pi$. First, $\frac{\exp(-H)}{c}$ is a function of the Hamiltonian alone so anything that preserves integral curves preserves π. Second, it is the canonical density of a Hamiltonian as written in Geman and Geman (1984). The canonical density is the density such that summing over the derivatives of the density with respect to each variable is equal to 0. i.e.

$$\sum_i \left(\frac{dP}{dz_i} + \frac{dP}{dp_i} \right) = 0$$

where P is the canonical density. The function $\frac{\exp(-H)}{c}$ is the simplest such distribution and has important applications to many physical phenomena.

It should be clear that given any function $H(z,p)$, H can be expressed in the form $H(z,p) = U(z) + V(p;z)$, where in physics U represents the potential energy and V represents the kinetic energy. H is called separable when V is a function solely dependent on p. Feeding this into the canonical density gives a density of the form:

$$\frac{\exp(-\{U(z) + V(p;z)\})}{T} = \frac{\exp(-U(z))\exp(-V(p;z))}{T}.$$

Conditioning such that

$$\int \exp(-V(p;z))dp = 1$$

demonstrates that the required Hamiltonian must have

$$\begin{aligned}\pi(z) =& \int \frac{\exp\left[-U(z) - V(p;z)\right]}{T} dp \\ =& \int \frac{\exp\left[-U(z)\right]\exp\left[-V(p;z)\right]}{T} dp \\ =& \frac{\exp\left[-U(z)\right]}{T}.\end{aligned}$$

The above implies

$$U(z) = -\log(\pi(z)).$$

Similarly we can find an expression for $V(p;z)$:

$$\pi(z)K(p;z) = \frac{\exp\left[-U(z)\right]\exp\left[-V(p;z)\right]}{T}$$

implies

$$\begin{aligned}\log(\pi(z)) + \log(K(p;z)) &= -\left[U(z) + V(p;z)\right] \\ V(p;z) &= -\log(K(p;z)).\end{aligned}$$

4.2.3 HMC

Hamiltonian Monte Carlo can be considered as a three step process consisting of a random proposal for momenta, a deterministic integration step, (which we shall adapt later to be non-deterministic) and a Metropolis-Hastings accept-reject step. As we have seen HMC constructs a Hamiltonian $H(x,p) = -\log(\pi(z)) - \log(K(p;z))$ so that the augmented density is $\pi(z,p) = \pi(z)K(p;z)$. This Hamiltonian defines flows ϕ_H that preserve $\pi(z,p)$ along them. To integrate along the flows ϕ_H we require an integrator. An integrator I is a function $t \times T^*\mathcal{M} \to T^*\mathcal{M}$ that given $z(0), p(0)$ approximates the flow at time t. Often t is fixed and so is absorbed into the integrator. For fixed t we shall express the integrator $I(t,z,p)$ as $I_t(z,p)$ or $I(z,p)$. We require that the numerical approximation of the derivative of the flow produced by an integrator converges to the true derivative. That is

$$\lim_{t\to 0} \frac{\varphi\left[I(t,(z(0),p(0)))\right] - \phi(z(0),p(0))}{t} = \left(\frac{d\phi}{dt}|_z, \frac{d\phi}{dt}|_p\right)$$

Later we shall adapt the integrator to have random components, in which case we write $I_t(z,p;z',p')$ as the probability density of integrating from (z',p') to (z,p). Often the integration length t will be fixed and so we will drop it from the expression.

In order to integrate along a flow we require the derivative of the posterior, and so require the likelihood to be differentiable. HMC reverses the direction of momenta after the integration step to ensure reversibility of the deterministic section. The integration step can be seen in Algorithm 3, step 2. For the random proposal HMC proposes new momenta, step 1. This ensures that the whole space is explored, not just circling about a single flow. It then uses these two processes combined as the proposed state for a Metropolis-Hastings accept-reject step, steps 4-5. Since the momenta are be marginalised out, it is sufficient to compare the proposal with the previous state with the momenta proposed at the end of step 1. When we use a kernel $K(p; x)$ such that it is symmetrical in p this step is not strictly necessary.

Algorithm 3: Hamiltonian Monte Carlo

- Input: unnormalised target density π, starting position z_0, starting momenta p_0; momenta proposal density K, integrator (often deterministic) acting along the Hamiltonian flows I and number of iterations n.
- For $i = 1, \dots n$:
  1. Draw $p' \sim K(\cdot; z_{i-1})$.
  2. Compute and set $(\hat{z}, \hat{p}) = I(z_{i-1}, p')$.
  3. Set $(\hat{z}, \hat{p}) = (\hat{z}, -\hat{p})$.
  4. Compute the acceptance probability
$$a = \min\left[1, \frac{\pi(\hat{z})K(\hat{p}; \hat{z})I((z_{i-1}, p'); (\hat{z}, \hat{p}))}{\pi(z_{i-1})K(p'; z_{i-1})I((\hat{z}, \hat{p}); (z_{i-1}, p'))}\right].$$
  5. With probability a, set $z_i = \hat{z}$ and $p_i = \hat{p}$ and otherwise set $z_i = x_{i-1}$ and $p_i = p_{i-1}$
- Output: chain $\{(z_i, p_i)\}_{i=0}^n$ with stationary distribution π.

It is common to simplify a into an expression involving H and further simplify a in the standard case when I is deterministic.

$$\begin{aligned} a &= \min\left[1, \exp\{H\left[(z_{i-1}, p')\right] - H\left[(\hat{z}, \hat{p})\right]\}\frac{I((z_{i-1}, p'); (\hat{z}, \hat{p}))}{I((\hat{z}, \hat{p}); (z_{i-1}, p'))}\right] \\ &= \min\left[1, \exp\{H\left[(z_{i-1}, p')\right] - H\left[(\hat{z}, \hat{p})\right]\}\right]. \end{aligned}$$

If I is a perfect integrator then $H\left[(z_{i-1}, p')\right] = H\left[(\hat{z}, \hat{p})\right]$ and a further simplifies to $a = 1$. This demonstrates that the Metropolis-Hastings accept-reject step acts as a correction step for the

integration. We include the option for non-deterministic I for clarity in later chapters.

4.2.4 The Hamiltonian Monte Carlo algorithm targets the correct distribution

Each iteration of the Hamiltonian Monte Carlo Algorithm consists of three parts: proposing the momenta, computing the integration and performing a Metropolis-Hastings accept-reject step. All three parts can be seen clearly in Figure 4.3.

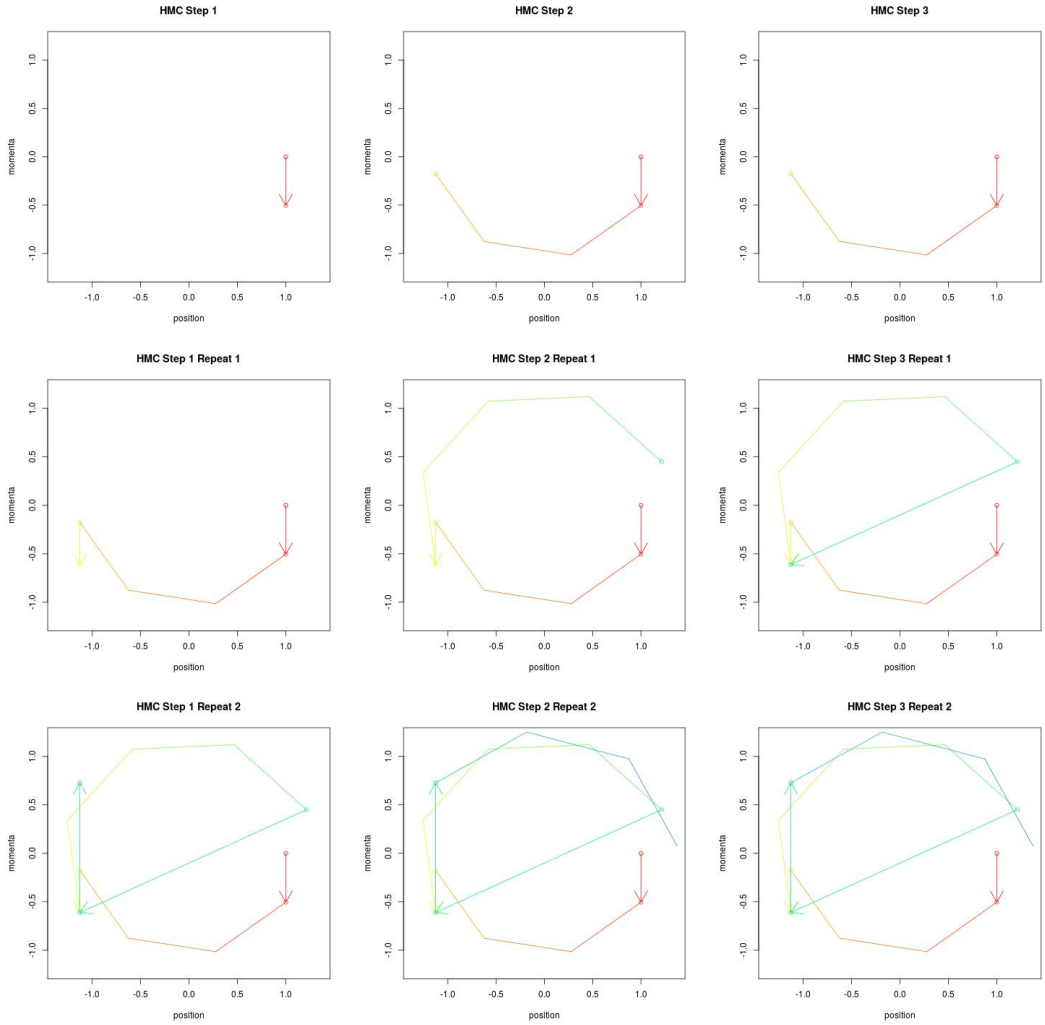


Figure 4.3: Each row contains a single iteration of HMC for the simple pendulum example, with the second row ending in a rejection step, each column a part of the process. The first column shows momenta being resampled, the second column shows a 4 step integration step and the third shows whether or not the final proposed point is accepted. The initial state is the red circle and proposed states are the yellow, green and blue coloured circles. The algorithm progresses from red to blue.

Theorem 4.1. *The HMC algorithm leaves the target distribution π invariant.*

Proof. We wish to show that the canonical distribution satisfies detailed balance:

$$\pi(z,p)a(z',p';z,p) = \pi(z',p')a(z,p;z',p')$$

We will notice that once we have sampled our cotangent vector the process is deterministic by a measure preserving transformation and so we get the following:

$$\begin{aligned}\pi(z,p)a(z',p';z,p) = \pi(z)K(p;z)a(z',p';z,p) =& e^{-H(z,p)} \times min[1, e^{-H(z',p')+H(z,p)}] \\ =& e^{-H(z',p')} \times min[e^{-H(z,p)+H(z',p')}, 1] \\ =& \pi(z')K(p';z')a(z',p';z,p) = \pi(z',p')a(z,p;z',p').\end{aligned}$$

Detailed balance implies that our update leaves the canonical distribution invariant. We can now check that the HMC kernel, which we shall denote P leaves the target distribution π invariant:

$$\int\int \pi(z,p)P(z',p';z,p)dzdp = \pi(z',p').$$

Expanding the left hand side in terms of the HMC acceptance probability gives:

$$\int\int \pi(z,p)P(z',p';z,p)dzdp = \pi(z',p')(1 - \int\int a(z,p;z',p')dzdp) + \int\int \pi(z;p)a(z',p';z,p)dzdp.$$

We can use the result from detailed balance to prove the rest:

$$\begin{aligned}&\int\int \pi(z,p)P(z',p';z,p)dzdp \\ &\quad= \pi(z',p')(1 - \int\int a(z,p;z',p')dzdp) + \int\int \pi(z',p')a(z,p;z',p')dzdp \\ &\quad= \pi(z',p')(1 - \int\int a(z,p;z',p')dzdp) + \pi(z',p')\int\int a(z,p;z',p')dzdp \\ &\quad= \pi(z',p').\end{aligned}$$

□

Theorem 4.2. *Ergodicity of HMC*

Proof. We use theorem 13.0.1.iv. in Meyn and Tweedie (2009) We know if our space is irreducible, aperiodic and Harris recurrent then the sample density of the Markov chain will converge to

the invariant distribution. The chain we have constructed is reversible, and so reversible on an intercommunicating set of size greater than two, by the Euclidean algorithm, we know that it is aperiodic. As it is reversible it is also Harris recurrent if it is irreducible, and it is irreducible by construction. □

4.2.5 Metropolis Adjusted Langevin Algorithm (MALA)

Hamiltonian Monte Carlo is very closely related to Langevin Monte Carlo (Rossky et al. (1978) and Grenander and Miller (1994)), which uses a similarly constructed Langevin diffusion, maintaining density, to enhance the proposal.

Let π be a density on $\mathbb{R}^n$, known up to a constant of proportionality. The Metropolis Adjusted Langevin diffusion algorithm (MALA) then uses a stochastic differential equation instead of a Hamiltonian. The stochastic differential equation is of the form:

$$dx(t) = \frac{1}{2}A\nabla \log(\pi(x_t))dt + \sqrt{A}db_t \tag{4}$$

where b_t is an n-dimension Brownian motion and A is a covariance matrix. A Brownian motion b_t is a stochastic process such that it is continuous almost surely. For an interval L of $\mathbb{R}$ we partition L into intervals $[t_0, t_1), [t_1, t_2), \ldots, [t_{k-1}, t_k]$ where $t_0, \ldots, t_k \in \mathbb{R}$. A Brownian motion has the property that any partition of the interval into $t_0, t_1, ..., t_k$ has $b_{t_i} - b_{t_{i+1}}$ independent to $b_{t_j} - b_{t_{j+1}}$ and $b_{t_i} - b_{t_{i+1}}$ is normally distributed according to $N(0, t_{i+1} - t_i)$.

The Langevin stochastic differential equation leaves $\pi(z)$ invariant. This means that when a set is evolved according to the s.d.e. the integral of the set w.r.t. measure π is constant almost surely. However we cannot simulate z exactly so require a discretisation, and as with HMC the error is corrected using the Metropolis-Hastings accept-reject ratio, (Girolami et al., 2011).

We discretise using the Euler-Maruyama discretisation with step size ϵ. Applying this to Equation (4) results in:

$$\begin{aligned} z_{t+1} =& z_t + \frac{1}{2}\epsilon A\nabla \log(\pi(z_t)) + \sqrt{\epsilon A}\eta \\ \eta \sim& \mathcal{N}(0, Id_n). \end{aligned} \tag{5}$$

Algorithm 4: Metropolis Adjusted Langevin Algorithm

- Input: unnormalized target density π, starting position z_0; normal proposal $\mathbf{N}(0, \Sigma)$, step size ϵ and number of iterations n.
- For $i = 1, \ldots, n$:
  1. Set $\hat{z} = z_{i-1} + \epsilon \nabla \log(\pi(z_{i-1}))$.
  2. Draw $z' = N(\cdot; \hat{z}, \epsilon A)$.
  3. Compute the acceptance probability
$$a = \min\left[1, \frac{\pi(z')N(z_{i-1}; z' + \epsilon\nabla\log(\pi(z')), \epsilon A)}{\pi(z_{i-1})N(z', z_{i-1} + \epsilon\nabla\log(\pi(z_{i-1})), \epsilon A)}\right]. \tag{4.1}$$
  4. With probability a, set $z_i = z'$ otherwise set $z_i = z_{i-1}$.
- Output: chain $\{(z_i, p_i)\}_{i=0}^n$ with stationary distribution π.

MALA is related to HMC via a single step of the Leapfrog Algorithm (Betancourt et al., 2017). Specifically, a single step of the Leapfrog algorithm produces the same proposal as MALA when the momenta is normal $V = N(0, \Sigma)$ and $\Sigma = \epsilon A$. We have seen in subsection 4.2.2 that the Hamiltonian is

$$H(z, p) = -\log(\pi(z)) - \log(\mathcal{N}(0, \Sigma)[p]).$$

This means that:

$$H(z, p) = -\log(\pi(z)) + \frac{1}{2}\log(det(2\pi\Sigma)) + \frac{1}{2}p^T\Sigma^{-1}p,$$

so the derivatives are

$$\frac{\partial H(z,p)}{\partial z} = -\nabla\log(\pi(z)) \text{ and } \frac{\partial H(z,p)}{\partial p} = \Sigma^{-1}p. \tag{6}$$

Applying the leapfrog integrator gives

$$\begin{aligned} p_{t+1/2} =& p_t + \frac{\epsilon}{2}\nabla \log\left[\pi\left(z_t\right)\right], \text{ and} \\ z_{t+1} =& z_t + \epsilon\left[\Sigma^{-1}p_{t+1/2}\right] \\ =& z_t + \epsilon\left[\Sigma^{-1}\left(p_t + \frac{\epsilon}{2}\nabla \log\left[\pi\left(z_t\right)\right]\right)\right] \\ =& z_t + \frac{\epsilon^2}{2}\Sigma^{-1}\nabla \log\left[\pi(z_t)\right] + \Sigma^{-1}\epsilon p_t \end{aligned}$$

where $p_t \sim \mathcal{N}(0, \Sigma)$. Substituting in $h = \epsilon^2\Sigma^{-1}$ gives

$$z_{t+1} = \frac{h}{2}\nabla \log\left[\pi(z_t)\right] + \sqrt{h\Sigma^{-1}}p_t$$

where $\eta_t \sim \mathcal{N}(0, Id_n)$

$$= \frac{h^2}{2}\nabla \log\left[\pi(z_t)\right] + \sqrt{h}\eta_t$$

which is the proposal produced by MALA.

The advantage of MALA is that it is less computationally intensive than HMC in producing a single proposal. This is due to needing to compute only the derivative with respect to position once rather than twice as in HMC employing a leapfrog integrator.

4.3 Numerical Integration of the Hamiltonian Flow

HMC requires the ability to integrate along flows and this requires knowledge of the derivatives of the Hamiltonian with respect to both parameters and momenta. There are many standard methods of conducting integration such as Euler and Runge-Kutta. The Euler method works by approximating the derivative in the neighbourhood of a point by the differential at the point. This leads to the approximated flow diverging over multiple iterations as can be seen in Figure 4.4. The leapfrog method or Störmer-Verlet integrator is a second order Runge-Kutta method and the standard method used during HMC (Leimkuhler and Reich, 2005). A second order method of integration is one where the error is of order $O(\epsilon^2)$ where ϵ is the distance integrated. The leapfrog method does not diverge over multiple iterations see Figure 4.4. The leapfrog integrator is also symplectic meaning that while it may have some error associated to it, the measure of a set integrated forward by leapfrog integration is preserved (Leimkuhler and Reich, 2005). Higher

order simplectic methods such as 4th order Runge-Kutta would also be effective but due to the high computational cost of differentiating the likelihood they are infeasible in this application.

The leapfrog integrator is simplectic because it employs shear transformations. A shear transformation is a transformation that transforms all points in an n-dimensional surface by an amount proportional to their distance from a $n-1$-dimensional surface. Applying the Leapfrog method l times means that the integrator integrates for time $t = l\epsilon$. Among the simplectic Runge-Kutta methods the leapfrog integrator requires the fewest computations of the derivative of the likelihood, normally a costly step.

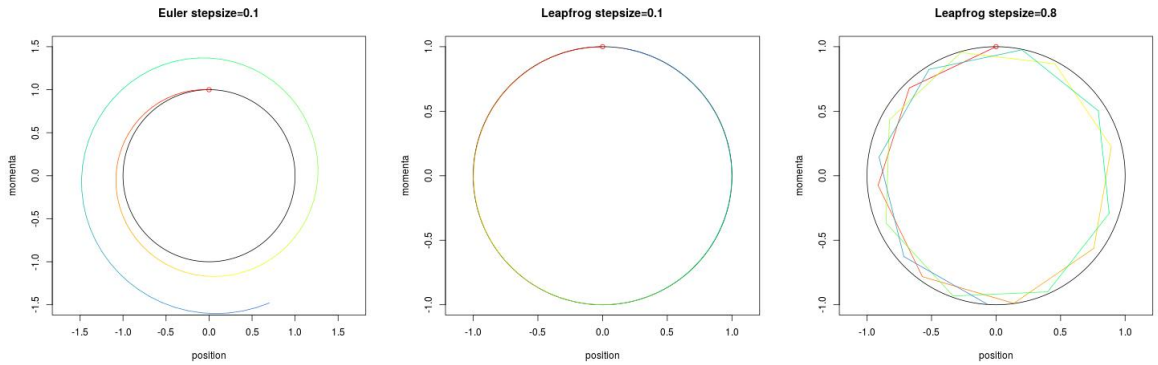


Figure 4.4: Each plot shows the phase portrait of a single flow of the simple pendulum example. The flow ϕ_t starts at the red dot and changes to blue as t increases. The first plot is the phase portrait where the flow is integrated using the Euler method with step size 0.1. The second plot shows the flow integrated using the leapfrog method and the same step size whereas the third plot uses the leapfrog method with a greater step size, 0.8.

Figure 4.4 demonstrates the phase portrait for position and momenta in Example 5.1.4 using different methods for integration. As we have seen starting at energy $H(z,p) = 1$ the phase portrait consists of circles of the form $z^2 + p^2 = 1$. The benefit of using the leapfrog integrator over the Euler integrator when integrating for a long time can clearly be seen as more proposals will be accepted due to its greater accuracy.

4.3.1 The Leapfrog Integrator

The leapfrog algorithm employs a half step update to momenta, a full step update to position conditional on the updated momenta, followed by a halfstep update to momenta from the new position momenta pair. This is illustrated in the following figure, Figure 4.5.

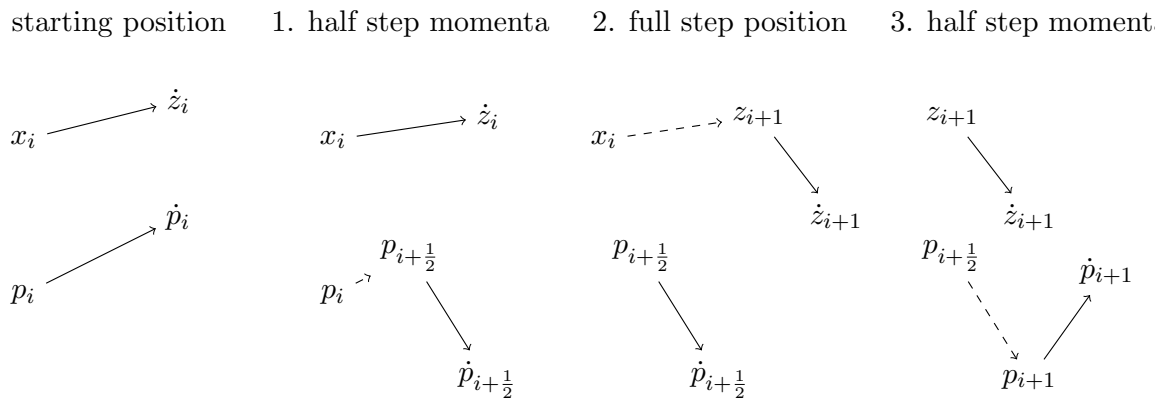


Figure 4.5: The leapfrog integrator first updates p_i to $p_{i+\frac{1}{2}}$, then x_i to x_{i+1} and finally $p_{i+\frac{1}{2}}$ to p_{i+1}. Updates are shown with dotted lines and the derivatives are shown with straight lines. The leapfrog integrator follows the flow by using the derivatives of the flow. Over small updates the behaviour is approximately linear with gradient specified by the derivative of the flow. The derivatives of the flow are given by equation (3) in subsection 4.2.2 are $-\frac{\partial H(z,p)}{\partial p}$ and $\frac{\partial H(z,p)}{\partial z}$, first performing a half update for momenta followed by a full update for position and then another half update for momenta.

Algorithm 5: The Leapfrog Integrator

- Input: starting point z_0, starting momenta p_0; step size $\epsilon > 0$; number of iterations l; $\frac{\partial H(z,p)}{\partial p}$ and $\frac{\partial H(z,p)}{\partial z}$.
- For $i = 1, \ldots, l$ set the following values:
  1. $p_{i+\frac{1}{2}} = p_i - \frac{\epsilon}{2}\frac{\partial H(z,p)}{\partial z}|_{(z_i,p_i)}$,
  2. $z_{i+1} = z_i + \epsilon\frac{\partial H(z,p)}{\partial p}|_{(z_i,p_{i+\frac{1}{2}})}$, and
  3. $p_{i+1} = p_{i+\frac{1}{2}} - \frac{\epsilon}{2}\frac{\partial H(z,p)}{\partial z}|_{(z_{i+1},p_{i+\frac{1}{2}})}$.
- Output: pair (z_l, p_l) approximately on the flow line starting at z_0, p_0.

When the Hamiltonian is separable, $H(z,p) = U(z) + V(p)$ the derivatives of H with respect to z and p simplify. The derivative $\dot{p} = -\frac{\partial H}{\partial z} = -\frac{\partial U}{\partial z}$ and the derivative $\dot{z} = \frac{\partial H}{\partial p} = \frac{\partial V}{\partial p}$. This means that $\dot{p}$ is independent of current p so that the first and and third steps concatenate to produce a single jump. This can be seen in the following figure, Figure 4.6.

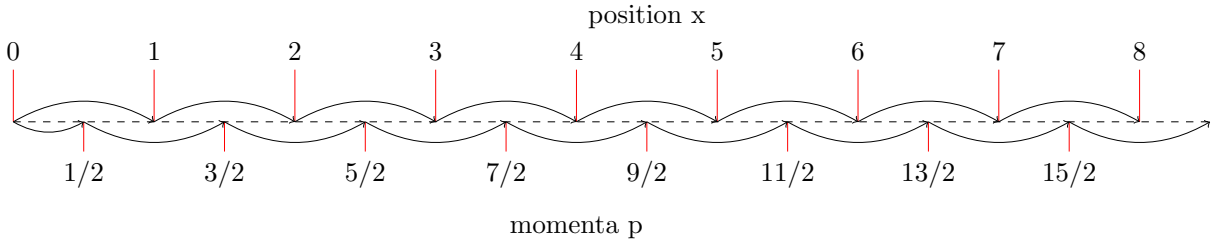


Figure 4.6: Evolution of the Leapfrog Integrator

Since calculating the derivatives of the Hamiltonian H with respect to the z is more computationally intensive than calculating derivatives of H with respect to the momenta for most choices of distribution for momenta, see (6) in subsection 4.2.5, we developed the Inverted Leapfrog Integrator.

Algorithm 6: The Inverted Leapfrog Integrator

- Input: starting point z_0, starting momenta p_0; step size $\epsilon > 0$; number of iterations l; $\frac{\partial H(x,p)}{\partial p}$ and $\frac{\partial H(z,p)}{\partial z}$.
- for $i = 1, \ldots, l$:
  1. $z_{i+\frac{1}{2}} = z_i + \frac{\epsilon}{2} \frac{\partial H(z,p)}{\partial p}|_{(z_i, p_i)}$
  2. $p_{i+1} = p_i - \epsilon \frac{\partial H(z,p)}{\partial z}|_{(z_{i+\frac{1}{2}}, p_i)}$
  3. $z_{i+1} = z_{i+\frac{1}{2}} + \frac{\epsilon}{2} \frac{\partial H(z,p)}{\partial p}|_{(z_{i+\frac{1}{2}}, p_{i+1})}$.
- output: pair (z_l, p_l) approximately on the flow.

Figure 4.6 shows how the position and momenta evolve over several leapfrog steps. As the inverted leapfrog mimics the leapfrog integrator after the first step it seems like a reasonable algorithm, as we shall now demonstrate.

Theorem 4.3. *The Inverted Leapfrog Algorithm correctly integrates along flows.*

Proof. To demonstrate this we will show that the inverted leapfrog algorithm is the same as the leapfrog algorithm with different choices for parameters z, momenta p and Hamiltonian H. Since the leapfrog algorithm is second order this demonstrates that the inverted leapfrog algorithm is too.

Suppose we perform the inverted leapfrog algorithm with the following z, p and H; $z = p'$, $p = z'$ and $H(z,p) = -H'(p,z)$ so that $\frac{\partial H}{\partial z}|_{(z,p)} = -\frac{\partial H'}{\partial p}|_{(p,z)}$ and $\frac{\partial H}{\partial p}|_{(z,p)} = -\frac{\partial H'}{\partial z}|_{(p,z)}$. This substitution converts the inverted leapfrog algorithm for z, p and H into the leapfrog algorithm for z', p' and H'. Since $H' \neq H$ we then demonstrate that integrating along flows of H' also integrates along flows of H. We start with the inverted leapfrog algorithm

$$\begin{aligned} z_{i+\frac{1}{2}} =& z_i + \frac{\epsilon}{2}\frac{\partial H(z,p)}{\partial p}|_{(z_i,p_i)} \\ p_{i+1} =& p_i - \epsilon\frac{\partial H(z,p)}{\partial z}|_{(z_{i+\frac{1}{2}},p_i)} \\ z_{i+1} =& z_{i+\frac{1}{2}} + \frac{\epsilon}{2}\frac{\partial H(z,p)}{\partial p}|_{(z_{i+\frac{1}{2}},p_{i+1})}. \end{aligned}$$

We apply the above substitutions for z and p so that the inverted leapfrog algorithm becomes

$$\begin{aligned} p'_{i+\frac{1}{2}} =& p'_i + \frac{\epsilon}{2}\frac{\partial H(p',z')}{\partial p}|_{(p'_i,z'_i)} \\ z'_{i+1} =& z'_i - \epsilon\frac{\partial H(p',z')}{\partial z}|_{(p'_{i+\frac{1}{2}},z'_i)} \\ p'_{i+1} =& p'_{i+\frac{1}{2}} + \frac{\epsilon}{2}\frac{\partial H(p',z')}{\partial p}|_{(p'_{i+\frac{1}{2}},z'_{i+1})}. \end{aligned}$$

Substituting in H' for H gives:

$$\begin{aligned} p'_{i+\frac{1}{2}} =& p'_i - \frac{\epsilon}{2}\frac{\partial H'(z',p')}{\partial z}|_{(z'_i,p'_i)} \\ z'_{i+1} =& z'_i + \epsilon\frac{\partial H'(z',p')}{\partial p}|_{(z'_i,p'_{i+\frac{1}{2}})} \\ p'_{i+1} =& p'_{i+\frac{1}{2}} - \frac{\epsilon}{2}\frac{\partial H'(z',p')}{\partial z}|_{(z'_{i+1},p'_{i+\frac{1}{2}})}. \end{aligned}$$

This is exactly the process in the iterative step of the leapfrog algorithm.

Finally, the inverted leapfrog algorithm integrates along the correct flow. It integrates along flows of H' so that by equation (3) of subsection 4.2.2 we get the first half of:

$$\frac{d\phi(t,z(t),p(t))}{dt} = \left(\frac{\partial H'}{\partial p}|_{(z(t),p(t))}, -\frac{\partial H'}{\partial z}|_{(z(t),p(t))}\right) = \left(\frac{\partial H}{\partial p}|_{(z(t),p(t))}, -\frac{\partial H}{\partial z}|_{(z(t),p(t))}\right).$$

The second half of the above equation follows by substituting back in H for H'. □

Corollary to Theorem 5.1: The Inverted Leapfrog Algorithm is a second order algorithm.

The Inverted Leapfrog Algorithm is useful to us as it requires half the number of calculations with respect to position. As the derivative calculation with respect to position is the derivative calculation involving the likelihood, it is often the more complex of the two parts to calculate. This adaptation makes HMC more competitive with MALA. To illustrate how the inverted leapfrog integrator works we use a contrived example with a forced wait for $0.001s$ at every evaluation of the derivative of the Hamiltonian with respect to position. Instead of the usual example we take $\pi(z)$ to be $N(z; 0, 0.5)$ so that we can see the different effects of the inverted leapfrog integrator. Figures 4.7 and 4.8 demonstrate the differences and similarities between the two methods.

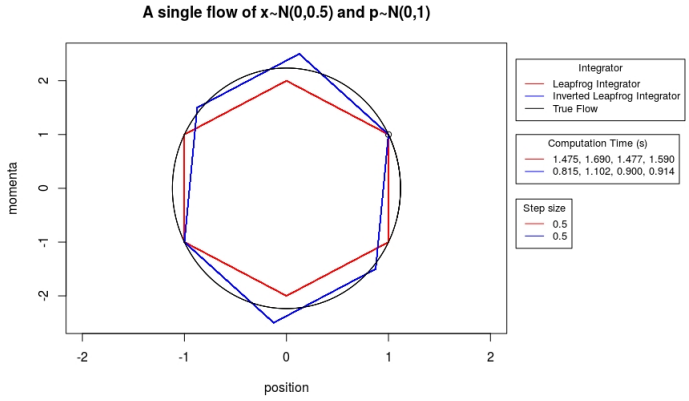


Figure 4.7: The red leapfrog integrator takes longer than than the blue inverted leapfrog integrator. Though both paths approximate the flow, they produce different paths. Since the variance of the momenta is greater than that of position, the inverted leapfrog integrator often overpredicts the magnitude of the momenta, whereas the leapfrog integrator often under predicts it.

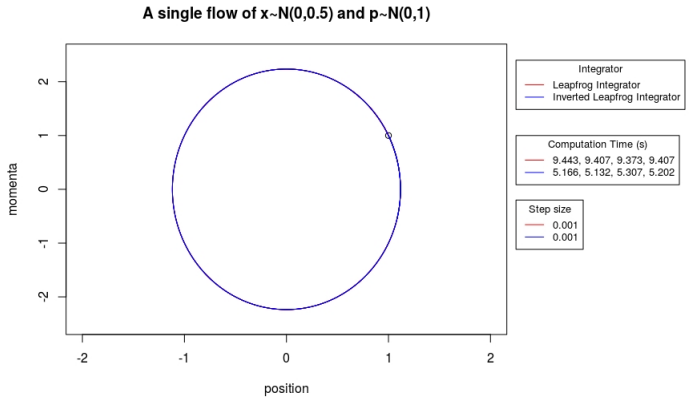


Figure 4.8: This plot is constructed the same way as Figure 4.7 but with a finer step size. The plot demonstrates how both methods converge to the true flow despite different paths taken. The difference in computational time also is more visible.

The Inverted Leapfrog Integrator is not reversible.

There is one problem with the inverted leapfrog integrator which is that it is not reversible. This is because of the dependency of $\frac{\partial H}{\partial p}$ on the state z. This would normally an issue with standard Metropolis-Hastings but we only apply the inverted leapfrog integrator within the context of COrtHMC. In chapter 5 we prove that COrtHMC leaves the target distribution invariant, see proposition 5.1, this proof involves a non-deterministic and not necessarily reversible integrator (to enable more interesting orthant crossing algorithms), and so the inverted leapfrog integrator can be applied as normal though with a slightly increased computational cost.

4.3.2 Appropriate Choice of Parameters

The leapfrog algorithm has a choice of two parameters, the step size ϵ and the number of iterations l. If you choose too small an l or ϵ, then the proposal is approximately a random walk. If l is too large then the algorithm is wasting computational time since for fixed ϵ, as l increases so too does the chance of rejecting the proposal. Increasing ϵ increases the error in the numerical integration, increasing the computational cost due to more rejections occuring. To get around this issue Hoffman and Gelman developed the No-U-Turn sampler and a method for choosing ϵ (Hoffman and Gelman, 2014). A more basic way of selecting ϵ and l is a grid search method in which many different runs are conducted (Neal et al., 2011). In Neal (1994) a method using windows of varied l is applied to average out the error in integration, resulting in higher acceptance rates. The whole window is then accepted or rejected in the Metropolis-Hastings accept-reject step. Neal (1994) demonstrates that this results in an optimum acceptance rate of approximately 0.85, rather than the standard theoretical optimum of 0.65 (Beskos et al., 2013). This means that whenever conducting HMC that employs an integrator with more than 2 steps, over a space with a likelihood which is computationally expensive to evaluate, a windowed approach may provide a good increase in efficiency. We cover both NUTS and the grid search method as well as an adaptation to Riemmanian Manifold Hamiltonian Monte Carlo, which effectively varies integration length in multiple dimensions to better correspond with the posterior. We found that with HMC it was best to restrict the number of steps and so it was relatively easy to search over them. We designed a method based on the heuristic algorithm of Hoffman and Gelman (2014) to estimate appropriate step sizes.

Algorithm: ApplyHeuristic (Hoffman and Gelman, 2014)

---

1. Parameters: Hamiltonian H, starting position z_0, starting momenta p_0, starting ϵ, integrator following Hamiltonian flows dependent on ϵ I_ϵ.
2. Set $z', p' = I_\epsilon(z_0, p_0)$.
3. Compute $p1 = H(z_0, p_0)$, $p2 = H(z', p')$
4. if $p2/p1 < 0.5$ set $a = 1$ else set $a = -1$
5. while $(p2/p1)^a \geq 2^{-a}$
   - $\epsilon' = 2^a \times \epsilon$
   - $z', p' = I_{\epsilon'}(z', p')$

- $p2 = H(z', p')$

6. return ϵ'

We enabled varying step-size using Roberts and Rosenthal (2007) and Bai et al. (2011) by applying the heuristic algorithm twice, once at the start of a run and once after a specified burn in, ϵ_{start} and $\epsilon_{burn-in}$ and setting $\epsilon_i = \epsilon_{burn-in} + (\epsilon_{start} - \epsilon_{burn-in}/(i+1))$.

4.3.3 Riemannian Manifold Hamiltonian Monte Carlo (RMH-MC)

RM-HMC is a technique designed by Girolami et al. (2011) to efficiently conduct HMC. It does so by altering or specifying the momenta kernel to be a choice that favours moves in the right direction. It approaches HMC by considering the target distribution, a specific evolutionary model, to be belonging to a space of distributions parameterised by values z and acting on data X. The idea is to consider z and by extension the evolutionary model as existing on a manifold where the metric is specified by the posterior density π. This notion is formalised in Shun-ichi (1985). In this space of distributions created by varying z there exists notions of distance such as the Kullback-Liebler divergence. The Kullback-Liebler divergence between $\pi(X|z)$ and $\pi(X|z')$ is defined as

$$KL(\pi(X|z)|\pi(X|z')) = \int_x \pi(X|z)\frac{\log(\pi(X|z))}{\log(\pi(X|z'))}.$$

When we look at an infinitesimal distance between two distributions $\pi(X|z)$ and $\pi(X|z+\delta z)$ it is a well known result that

$$KL(\pi(X|z)|\pi(X|z+\delta z)) = \delta z^t G(z)\delta z$$

where $G(z)$ is the Fisher information matrix, (Shun-ichi, 1985). The Fisher information matrix is defined by

$$G(z)_{ij} = \mathbb{E}_X\{\frac{\partial \log(\pi(X|z))}{\partial z_i}\frac{\partial \log(\pi(X|z))}{\partial z_j}\}.$$

This means that $G(z)$ encapsulates the distance between $\pi(z)$ and $\pi(z+\delta z)$. We have seen in subsection 4.3.3 example 2 that any Riemannian metric can be used as the covariance matrix for a multivariate normal distribution. By considering the Fisher information matrix as both the metric on a manifold and the covariance matrix for a multivariate normal distribution, Riemannian manifold-HMC can be conducted.

Using the Fisher information metric is a sensible idea. In Shun-ichi (1985) the Cramér-Rao

inequality is proven. The Cramér-Rao inequality is important as it tells us that the covariance matrix for an estimator in some sense dominates the inverse of the Fisher information matrix, meaning we expect our estimator to have stronger correlations than that of the Fisher information matrix. Notionally the Fisher information matrix therefore provides a greater range of exploration in dimensions which impact the change in $\pi(X|z)$ the most. It also means that we do not completely ignore any correlations occurring within the data. Consider the diagonal case, larger variance leads to larger proposed momenta and therefore better exploration. When $K(p;z) = N(p;0,G(z))$ then Hamiltonian is as follows

$$H(z,p) = -\mathcal{L}(z) - \frac{1}{2}\log((2\pi)^D|G(z)|) + \frac{1}{2}p^tG(z)^{-1}p.$$

In order to compute the flow we need to be able to compute the derivatives of $H(z,p)$ with respect to both z and p.

$$\begin{aligned}\frac{\partial H}{\partial z} =& -\frac{\partial \pi}{\partial z} + \frac{1}{2}Tr\{G(z)^{-1}\}\frac{\partial G(z)}{\partial z} - \frac{1}{2}p^tG(z)^{-1}\frac{\partial G(z)}{\partial z}G(z)^{-1}p \\ \frac{\partial H}{\partial p} =& G(z)^{-1}p.\end{aligned} \tag{7}$$

The flow therefore caries $G(z)$ along with z and p ensuring reversibility, Equation (7).

Part II

Hamiltonian Monte Carlo on Tree-Space

Chapter 5

HMC on Tree space

In the previous chapter we saw that Hamiltonian Monte Carlo (HMC) uses a measure preserving flow to make large jumps without sacrificing the acceptance probability. In this chapter we show that we have to modify HMC for Tree space because no such flow exists on the whole of it. A measure preserving flow does however exist within each orthant. In this chapter we show how to extend HMC so that it can cross orthant boundaries, while maintaining the same behaviour as HMC within every orthant. We define COrtHMC a novel method for MCMC integration over Tree space that combines HMC with a mechanism for crossing orthants. Another method has been developed by Dinh et al. (2017) called Probabilistic Path Hamiltonian Monte Carlo (ppHMC) to conduct HMC over tree space first described in subsection 1.4.4. Our method is similar to ppHMC but enables various different crossing methods, one of which coincides with ppHMC. We outline the various ways of conducting COrtHMC.

In this chapter we shall assume that the substitution parameters θ and site rate parameters α are fixed and known. HMC over substitution parameters and site rate parameters will be covered in chapter 6 .

5.1 Flows across Tree space

We first show that standard HMC is not applicable on Tree space:

Theorem 5.1. *There is no well-defined continuous flow ϕ_t defined on the cotangent bundle of $\mathcal{T}_n$ which has flow lines traversing between at least two distinct maximal orthants, where flow is defined in subsection 4.2.2.*

Proof. In order to talk about continuity we need a topology on the space of unrooted trees ($\mathcal{U}_n$). We define the topology on unrooted trees by using the Euclidean metric within each orthant

and extending it across Tree space as outlined in Nye (2015). This is done by taking the metric across tree space

$$d_{\mathcal{T}}(x,y) = inf\left[\sum_{i=0}^{j} d_{\Sigma}(x_i, x_{i+1}|x_0 = x, x_j = y, \text{ where } x_i, x_{i+1} \text{ in the same closed minimal orthant}\right]$$

d_{Σ} is the standard euclidean metric. This works out to be the infimum of lengths of paths between x and y with straight line segments in each orthant. The space of rooted trees $\mathcal{T}_n$ inherits this metric.

Tree space constists of full dimensional orthants and codimensional-1 regions ("the glue") between them $\mathcal{T}_n = \bigcup_{\tau} \mathbb{R}^{n-3} \cup \mathcal{T}_n^{(1)}$ where $\mathcal{T}_n^{(1)}$ are the codimension-1 boundaries of Tree space. The topology is the topology from the BHV metric, so open sets are unions of open sets in each orthant. We shall show that no flow traversing two maximal orthants exists on the open book. We shall then describe the cotangent space $T^*\mathcal{T}_n$ using Barden et al. (2013). By showing that the cotangent space for $\mathcal{T}_n^{(1)}$ contains a copy of the open book we demonstrate that no flow can exist on the cotangent space.

Tree space contains many copies of the 3-spider.

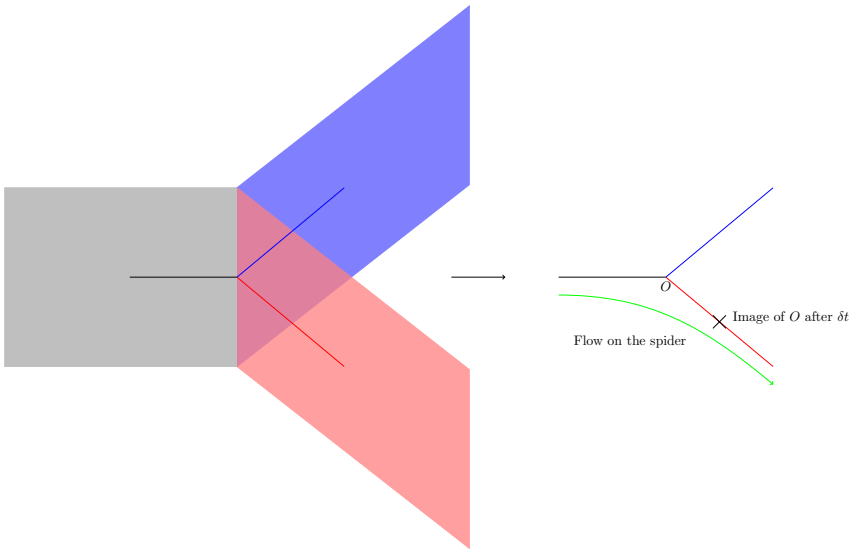


Figure 5.1: The open book and a 3-spider

Consider a flow passing through a point t on the spine of the open book from one maximal orthant to another, call them orthant A and B. In the 3-spider, $S = \vee\mathbb{R}^3_{\geq 0}$, this is shown by the

green flow in Figure 5.1. However this is discontinuous as the points on the blue orthant are not moved by the flow. By continuity all significantly close points to the t on also end up on B bounded away from the spine. By the same argument all significantly close points to the image of t come back from B. This means the flow lines on an open interval of the spine containing t all go from A to B. The flow is discontinuous because a point arbitrarily close to the origin on the third orthant, the blue orthant in 5.1, can be at most δx close to the image of the origin. δx is bounded above 0 and is only dependent on the flow and δt. Hence there cannot exist a flow on the 3-spider traversing Tree space. Since there cannot exist a flow on the 3-spider there cannot exist a flow on the open book. This is because the open book is constructed from the 3-spider, $\mathcal{T}_n^{(1)} = S \times \mathbb{R}^{n-4}$. Since pendant edges are incorporated by a product, at strictly codimension-1 tree space is locally homeomorphic to an open book. We now wish to show that the cotangent space has the same property, we do this by finding a copy of the open book in the cotangent space.

The cotangent bundle of tree space also contains regions which are locally homeomorphic to the open book. Barden et al. (2013) section 4 defines the tangent space for Tree space $\mathcal{T}_5$. The tangent bundle consists of the tangent spaces of each orthant with a co-ordinate scheme that represents each orthant or quadrant. The tangent space at a codimension-1 point under this scheme is regarded as the union of the three half planes connected by the codimensional point. It is important to note that in Barden et al. (2013) these planes are considered distinct. The cotangent spaces are the duals of the tangent spaces. It is sufficient to note that from Barden et al. (2013) there are open regions of the tangent bundle, including points in the fibres that project to codimension-1 singularities in Tree space, which are homeomorphic to open books. The same applies to the cotangent bundle. Since the cotangent bundle contains local copies of open books it follows that no continuous flow can exist.

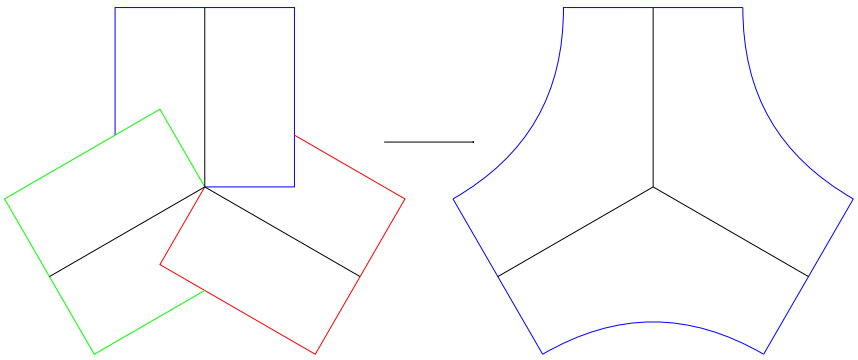


Figure 5.2: On the left we depict the cotangent space of the 3-spider. On the right we depict the cotangent space flattened to create a tubed space as in subsection 5.9.1.

□

Example 6.1:

To better get to grips with the cotangent space we describe it for $\mathcal{T}_5$ and $\mathcal{T}_4$ at a codimension-1 point. Using the same co-ordinate scheme as Barden et al. (2013) we can express the cotangent space at a codimension-1 point of $\mathcal{T}_5$ as the union of the three half planes $\mathbb{R}^2_{\geq 0} \times \mathbb{R}^2$ (the cotangent space within each half plane connected by the codimensional point). Similarly for $\mathcal{T}_4$, the three spider the cotangent space is the union of the cotangent space of the three half planes $\mathbb{R} \times \mathbb{R}$. In the first part of Figure 5.2 we depict the cotangent space at a point on $\mathcal{T}_4^{(1)}$.

There is an interesting relationship between the cotangent space and the "tubing" method outlined in 5.9.1. It is possible to extend the cotangent bundle with an auxiliary variable at codimension-1 points in Tree space. This auxiliary variable glues the disjoint $\mathbb{R}^n_{\geq 0} \times \mathbb{R}^n$ cubes that exist in the cotangent space together. Figure 5.2 demonstrates how this is done for the 3-spider. The tubing enables a flow to exists across this augmented tangent space. As this seems like a natural and intuitive thing to do here we quickly outline why it is a bad idea.

1. The auxiliary variable has no consistent meaning between maximal orthants.
2. It is a method of "tubing" but with less control over the interior of an orthant.
3. This method of tubing is guaranteed to introduce biased exploration as the momenta at this point is more likely to be negative than positive.

5.1.1 Motivation

We have now shown that HMC has to be adapted for Tree space. In this section we motivate the use of HMC in Tree space.

In section 2.5 we outlined how the likelihood was constructed in terms of $P_{x,y;k}\{\ell(u,v),\theta\}$ and in subsection 2.1.2 we saw that $P_{x,y;k}\{\ell(u,v),\theta\} = P_0 \exp(Qk\ell(u,v))$. Taking the derivative w.r.t. $\ell(u,v)$ results in $P_0 \exp(Qk(\theta)\ell(u,v))Qk(\theta)$, which is easily computable.

We use Figure 5.3 to further motivate using an adapted form of HMC. In Figure 5.3 we used the JC69 model to generate a Hamiltonian on Tree space using the likelihood construction in section 2.5 and an Alignment randomly generated from a tree in $\mathcal{T}_5$. We chose $\mathcal{T}_5$ so that there were only 2 internal edges. This enables both internal edge lengths to be put onto the different axes. The

lengths of the internal edges of this tree are highlighted by blue lines in Figure 5.3. We then used this Hamiltonian to plot flows from a grid of starting points. Flows starting from this grid are in black. We started the flow with zero momenta so that we could see how the likelihood affected the flow. Figure 5.3 demonstrates how the flow can provide information about regions of interest and correctly guide the MCMC algorithm. The flow spends less time in incorrect topologies as can be seen by the greater exploration in the correct orthant. We also notice that the flow consistently moves towards the correct values when in a neighbouring topology but if we move to a topology further away that information is lost.

When we encountered a boundary edge reaching 0, depicted in green, we deterministically specified an orthant to flow into, either the topology used to generate the alignment or a topology bordering it. In this way we can consider the flow as moving towards regions where we would expect high density or lower density. To move into the new topology the absolute value of the edge is chosen. The flow in the new topology is depicted in red. Looking closely at Figure 5.3 we see that the black flow smoothly changes to red. The first plot in Figure 5.3 shows the flow moving into the topology of the tree used to generate the alignment. The internal edge lengths of the tree are marked by blue lines and all edges taken to be of length 0.1.

It is important to notice the difference in scale of the two plots. In the first the flows explore three times further along the axis representing the internal edge of interest into the orthant than in the second, where the flow explored roughly the same region as the starting grid of points. This demonstrates two important conclusions we can make. The flow will spend more time in regions where the posterior density function takes larger values and this can be seen by the amount of red on the picture and the greater region of exploration. Tracking a given path gives a significant increase in length before returning to a boundary, notably the path that explores the furthest ends before returning to the boundary. No such path exists in the second picture. The second conclusion is that the flow responds to symmetry in the distribution. The tree used to generate the alignment was symmetrical in the sense that both internal edges are equal and all pendant edges are equal. While the alignment does not reproduce that symmetry it does hint to it in the way that choosing an orthant symmetrically equivalent in the tree results in the flow exploring roughly the same region. These plots do not only show that by randomly choosing an orthant to move into we are likely to return to the boundary with the original orthant if we choose poorly. If we choose correctly, the likelihood of reaching the boundary with another orthant is increased.

Figure 5.3 also demonstrates that the flow has some knowledge of the tree used to generate the alignment within an orthant. This is clear from the shift in the flow in the x-axis edge from

a length in the 0.1 to 0.3 region to the region 0 to 0.3 where the flows appear to spiral round 0.1, the internal edge length used to generate the alignment. This is a shift we do not see when we move to an orthant not containing the tree used to generate the alignment. This visually demonstrates the dependence of internal edge lengths on topology.

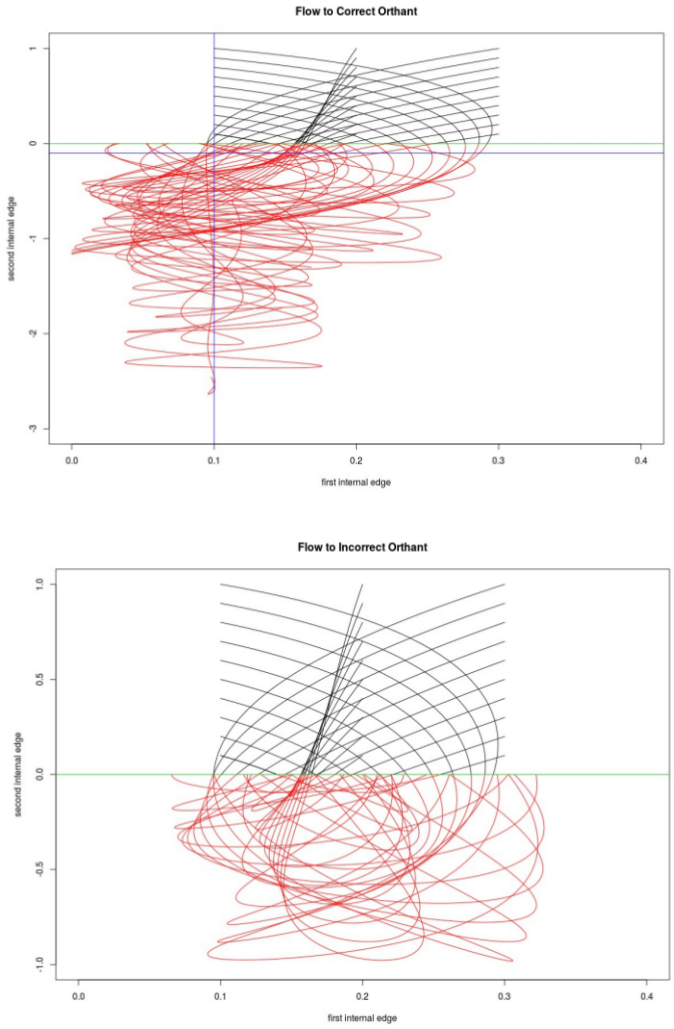


Figure 5.3: Here we create a posterior distribution from a set tree, then picked an orthant 1 nearest neighbour interchange from the topology of the set tree. We then chose a grid of points, the initial points of the black lines and ran a leapfrog integrator with low stepsize. Upon encountering a boundary we chose which topology to move into, in the first diagram we chose to move into the topology of the set tree in the second we moved to a different topology, we depict the flow in the new topology in red. Comparison between flows from an incorrect topology (black) moving to either another incorrect topology or the correct topology (red). We use a blue cross to pinpoint the correct tree in the first diagram (it is not pictured in the second). We can see that when the flow moves into the correct topology it explores further into that topology , it circles around the tree used to generate the alignment, and the total time spent in that orthant is longer. We have scaled the second plot so that the axes align.

5.2 Cross Orthant HMC

Our method applies standard HMC on each orthant $\mathbb{R}_+^{2n-3}$ and considers the integrator I as a kernel; see subsection 4.2.3. The Hamiltonian flow is defined on an orthant up to the boundary. This means that each point p within an orthant has an interval $[0, t_p)$, where t_p is when the flow hits a codimensional boundary and the flow is not defined after this point. This means that the numerical integrator along the flow J maps the orthant $\mathbb{R}_+^{2n-3}$ into $\mathbb{R}^{2n-3}$. The kernel $I = I(\cdot, \cdot; z, p)$ takes $T^*\mathcal{M} \to T^*\mathcal{M}$ where $\mathcal{M} = \mathcal{T}_n \times \Theta \times \mathbb{R}_+$, $\mathcal{T}_n$ is defined in section 3.1, Θ is as in subsection 2.2.1, and $\mathbb{R}_+$ is the space for rate heterogeneity parameter. This combines the numerical integrator for the flow J with a kernel C that dictates what happens when J maps into $\mathbb{R}^{2n-3} \setminus \mathbb{R}_+^m$.

We now explain this construction in more detail. In section 5.7 we define various different crossing methods for crossing a boundary as maps $\mathbb{R}^{2n-3} \times \Theta \times \mathbb{R}_+ \to \mathcal{T}_n \times \Theta \times \mathbb{R}_+$. The crossing method could be deterministic but in constructing a deterministic method we risk creating periodicity in the topologies we move to, creating a reducible chain, and introducing forced latency in the event that the deterministic crossing method chosen is systematically wrong. To demonstrate this point, suppose a deterministic crossing method mostly suggested moves to orthants away from regions of high density. We take the No-U-Turn Sampler (NUTS), see section A.3 in appendix A, as a motivating example for the kernel approach to integration. At every iteration of NUTS a deterministic process creates a set of possible proposals, this set is then reduced by a non-deterministic method. In the same way we think of I in COrtHMC as being built from two components, a deterministic component J and a non-deterministic component C. We take J to be a standard integrator for the Hamiltonian flow acting within an orthant and mapping to the orthant extended to include trees with "negative" edges. As J follows the flow and the flow is not defined across codimension 1 boundaries J cannot itself change topology. The non-deterministic C creates and chooses from a set of potential trees in various different topologies created from the tree with "negative" edge lengths. This means that we can express I in either of two equivalent ways depending on whether we think of J as a deterministic function $(z', p') = J(z, p)$ or a kernel

$$J(z', p'; z, p) = \begin{cases} 1 \text{ if } (z', p') = J(z, p) \\ 0 \text{ otherwise} \end{cases}.$$

We can take either:

1. I is the composition of J and C so that $I(z', p'; z, p) = C(z', p'; J(z, p))$

2. or $I(\hat{z}, \hat{p}; z, p) = C(\hat{z}, \hat{p}; z', p')J(z', p'; z, p)$.

In the algorithm we use 1. In order for our algorithm to coincide with HMC when restricted to flows contained within an orthant we require that

$$C(\hat{z}, \hat{p}; z', p') = \begin{cases} 1 \text{ if } z' \in \mathcal{T}_n \text{ and } (\hat{z}, \hat{p}) = (z', p') \\ 0 \text{ if } z' \in \mathcal{T}_n \text{ and } (\hat{z}, \hat{p}) \neq (z', p') \end{cases}.$$

We construct the MCMC algorithm as follows. Within an orthant apply standard HMC; upon hitting a boundary propose a new point, possibly in a new orthant, to move to. There are several different methods for proposing this new point. We first cover the general case before outlining some specific proposal mechanisms. We have designed COrtHMC for phylogenetic inference and therefore we shall express the algorithm in this setting. We recall that z represents all of the parameters for trees in $\mathcal{T}_n$. Since the topology τ is a parameter of interest, we henceforth denote by τ the topology and z the remaining parameters.

Algorithm 7: Cross-orthant HMC (COrtHMC)

- Input: unnormalised target density π, starting position z_0 and starting momenta p, momenta density K, deterministic component of integrator J, random component of integrator C, number of iterations n
- for i in $1 : n$
  1. Draw $p' \sim K(\cdot; z_{i-1}, \tau_{i-1})$.
  2. Compute $(\hat{z}, \hat{p}, \tau_{i-1}) = J(z_{i-1}, p', \tau_{i-1})$.
  3. Draw $(\hat{z}', \hat{p}', \tau') \sim C(\cdot; (\hat{z}, \hat{p}, \tau_{i-1}))$.
  4. Set $(\hat{z}', \hat{p}', \tau') = (\hat{z}', -\hat{p}', \tau')$.
  5. Compute
  $$a = \min\left[1, \frac{\pi(\hat{z}', \hat{p}', \tau')C(\hat{z}, \hat{p}, \tau_{i-1}; \hat{z}', \hat{p}', \tau')J(z_{i-1}, p', \tau_{i-1}; \hat{z}, \hat{p}, \tau_{i-1})}{\pi(z_{i-1}, p', \tau_{i-1})J(\hat{z}, \hat{p}, \tau_{i-1}; z_{i-1}, p', \tau_{i-1})C(\hat{z}', \hat{p}', \tau'; \hat{z}, \hat{p}, \tau_{i-1})}\right].$$
  6. With probability a, set $z_i = \hat{z}'$, $p_i = \hat{p}'$ and $\tau_i = \tau'$ and otherwise set $z_i = z_{i-1}$, $p_i = p_{i-1}$ and $\tau_i = \tau_{i-1}$.
- Output: chain $\{(z_i, p_i, \tau_i)\}_{i=0}^n$ with invariant distribution π.

5.3 COrtHMC targets the posterior and produces an ergodic chain

We first demonstrate that COrtHMC targets the posterior. This follows because it is constructed to satisfy detailed balance.

Proposition 5.1. *COrtHMC leaves $\pi(z)$ invariant where $\pi(z) = \int \frac{1}{Z} \exp(-H(z,p))dp$.*

Proof. Let $P(z', p', \tau', z, p, \tau)$ be the transition kernel for COrtHMC. A distribution π is the invariant distribution of P if

$$\int P(z', p', \tau', z, p, \tau)\pi(z, p, \tau)dz\, dp\, d\tau = \pi(z', p', \tau').$$

If P can be decomposed into kernels, $P = P_2 \cdot P_1$ where P_1 and P_2 leave π invariant then P leaves π invariant:

$$\begin{aligned}
&\int P(\hat{z},\hat{p},\hat{\tau}, z, p, \tau)\pi(z, p, \tau)dz\, dp\, d\tau \\
&\quad= \int P_1(z', p', \tau', z, p, \tau)P_2(\hat{z}, \hat{p}, \hat{\tau}, z', p', \tau')\pi(z, p, \tau)dz\, dp\, d\tau\, dz' dp'\, d\tau' \\
&\quad= \int P_2(\hat{z}, \hat{p}, \hat{\tau}, z', p', \tau') \left[\int P_1(z', p', \tau', z, p, \tau)\pi(z, p, \tau)dzdpd\tau\right] dz'\, dp'\, d\tau' \\
&\quad= \int P_2(\hat{z}, \hat{p}, \hat{\tau}, z', p', \tau')\pi(z', p', \tau')dz'\, dp'\, d\tau' \\
&\quad=\pi(\hat{z}, \hat{p}, \hat{\tau})
\end{aligned}$$

We show that the kernel for COrtHMC leaves the target density $\pi(z, \tau)$ invariant by showing that it satisfies detailed balance. We follow the same method as in Brockwell and Kadane (2018) where they validate a Hybrid Metropolis-Hastings algorithm. We shall consider the process in two steps, the first being the integration step and the second the crossing step. We shall demonstrate that each step leaves the target density invariant and therefore by the above argument the kernel for COrtHMC does also. We consider

$$P_1(z', p', \tau; z, p, \tau) = J(z', p', \tau; z, p, \tau) \min\left[1, \frac{\pi(z', p', \tau')J(z, p, \tau; z', p', \tau)}{\pi(z, p, \tau)J(z', p', \tau; z, p, \tau)}\right].$$

The kernel P_1 leaves the target π invariant because it satisfies detailed balance:

$$\begin{aligned}
P_1(z',p',\tau';z,p,\tau)\pi(z,p,\tau) =& \pi(z,p,\tau)J(z',p',\tau';z,p,\tau)\min\left[1,\frac{\pi(z',p',\tau')J(z,p,\tau;z',p',\tau')}{\pi(z,p,\tau)J(z',p',\tau';z,p,\tau)}\right]\\
=& \min\left[\pi(z,p,\tau)J(z',p',\tau';z,p,\tau),\pi(z',p',\tau')J(z,p,\tau;z',p',\tau')\right]\\
=& \pi(z',p',\tau')J(z,p,\tau;z',p',\tau')\min\left[\frac{\pi(z,p,\tau)J(z',p',\tau';z,p,\tau)}{\pi(z',p',\tau')J(z,p,\tau;z',p',\tau')},1\right]\\
=& P_1(z,p,\tau;z',p',\tau')\pi(z',p',\tau').
\end{aligned}$$

We now consider

$$P_2(z',p',\tau';z,p,\tau) = \begin{cases} C(z',p',\tau';z,p,\tau)\min\left[1,\frac{\pi(z',p',\tau')C(z,p,\tau;z',p',\tau')}{\pi(z,p,\tau)C(z',p',\tau';z,p,\tau)}\right] \\ 0 \end{cases}$$

if respectively $C(z,p,\tau;z',p',\tau') \neq 0$ and $C(z',p',\tau';z,p,\tau) \neq 0$; and if respectively $C(z,p,\tau;z',p',\tau') = 0$ and $C(z',p',\tau';z,p,\tau) = 0$. P_1 and P_2 combine to draw from J draw from C and accept with acceptance probability a. We show detailed balance which implies π invariance: When $C(z,p,\tau;z',p',\tau') \neq 0$ and $C(z',p',\tau';z,p,\tau) \neq 0$ the following holds:

$$\begin{aligned}
P_2(z',p',\tau';z,p,\tau)\pi(z,p,\tau) =& \pi(z,p,\tau)C(z',p',\tau';z,p,\tau)\min\left[1,\frac{\pi(z',p',\tau')C(z,p,\tau;z',p',\tau')}{\pi(z,p,\tau)C(z',p',\tau';z,p,\tau)}\right]\\
=& \min\left[\pi(z,p,\tau)C(z',p',\tau';z,p,\tau),\pi(z',p',\tau')C(z,p,\tau;z',p',\tau')\right]\\
=& \pi(z',p',\tau')C(z,p,\tau;z',p',\tau')\min\left[\frac{\pi(z,p,\tau)C(z',p',\tau';z,p,\tau)}{\pi(z',p',\tau')C(z,p,\tau;z',p',\tau')},1\right]\\
=& P_2(z,p,\tau;z',p',\tau')\pi(z',p',\tau').
\end{aligned}$$

When $C(z,p,\tau;z',p',\tau') = 0$ or $C(z',p',\tau';z,p,\tau) = 0$ the above does not hold and we wish to accommodate such integrators, in particular when our choice of kernel restricts the values the momenta can take. This means that $C(z',p',\tau';z,p,\tau) \neq 0$ but $C(z,p,\tau;z',p',\tau') = 0$. Let us assume that $C(z,p,\tau;z',p',\tau') = 0$ and $C(z',p',\tau';z,p,\tau) \neq 0$. We need to define the acceptance probability in this case. C having a constrained support when restricted to p we disperse p afterwards. We can disperse p by the introduction of another kernel $F \sim N(\tilde{p}, cId_{dim(p)})$ where $\tilde{p}$ is a point in the support of C. We can then define $C^c = F \circ C$ which satisfies detailed balance and never has $C^c(z,p,\tau;z',p',\tau') = 0$ or $C^c(z',p',\tau';z,p,\tau) = 0$. This adds a step to C where the momenta is resampled according to a $N(\tilde{p}, cId_{dim(p)})$ which combines with C and slightly alters the acceptance probability to a^c. It should be clear from the above that π is the invariant distribution of P in these cases. We now wish to demonstrate that $\pi(z)$ is the invariant

distribution of the limit when $c \to 0$.

We have constructed this new kernel such that the distribution π is invariant for P_2 with this choice of $C = C^c$. We can therefore write the following for $C^c a^c$:

$$\int \pi(z,p,\tau)C^c(z',p',\tau';z,p,\tau)a^c(z',p',\tau';z,p,\tau)dz\,dp\,d\tau = \pi(z',p',\tau').$$

By taking the limit of each side and integrating over p' because p' is drawn from a normal distribution $N(\tilde{p}, cId_{dim(p)})$, it follows that

$$\begin{aligned}\lim_{c\to 0}\int \pi(z,p,\tau)C^c(z',p',\tau';z,p,\tau)a^c(z',p',\tau';z,p,\tau)dz\,dp\,d\tau\,dp' \\ = \int \pi(z',p',\tau')dp' = \pi(z',\tau').\end{aligned}$$

Therefore by the dominated convergence theorem

$$\begin{aligned}\lim_{c\to 0}\int \pi(z,p,\tau)C^c(z',p',\tau';z,p,\tau)a^c(z',p',\tau';z,p,\tau)dz\,dp\,d\tau\,dp' \\ = \int \lim_{c\to 0}\pi(z,p,\tau)C^c(z',p',\tau';z,p,\tau)a^c(z',p',\tau';z,p,\tau)dz\,dp\,d\tau\,dp' \\ = \int \pi(z,p,\tau)C(z',p',\tau';z,p,\tau)a(z',p',\tau';z,p,\tau)dz\,dp\,d\tau\,dp'.\end{aligned}$$

So that:

$$\pi(z',\tau') = \int \pi(z,p,\tau)C(z',p',\tau';z,p,\tau)a(z',p',\tau';z,p,\tau)dz\,dp\,d\tau\,dp'.$$

We want to construct an appropriate acceptance probability for this case. To do this we examine C^c. As $c \to 0$, C^c selects points closer to $\tilde{p}$, this is true no matter where the chain was before the crossing. Remembering that p' is a point in the support of C, we can define P_2 to be:

$$C(z',p',\tau';z,p,\tau)\min\left[1, \frac{\pi(z',p',\tau')C(z,\tilde{p},\tau;z',p',\tau')}{\pi(z,p,\tau)C(z',p',\tau';z,p,\tau)}\right]$$

where $\tilde{p}$ is fixed and chosen by the user. This in turn defines the acceptance probability when our crossing methods present such problems. □

Theorem 5.2. *The Markov Chain defined by COrtHMC is ergodic, irreducible and aperiodic.*

COrtHMC is a generalisation of ppHMC and the arguments given in Dinh et al. (2017) generalise to COrtHMC. The arguments in Dinh et al. (2017) establish the theorem. For more details see the appendix 5.7.4Proofs.

5.4 Probabilistic Path HMC (ppHMC)

In this section we describe probabilistic path HMC in Dinh et al. (2017).

Algorithm: The Leap-prog algorithm (ppHMC)

- Input: U, starting position z and starting momenta p, stepsize ϵ
  1. $p = p - \epsilon \frac{\delta U(\tau, z)}{2}$
  2. If $FirstUpdateEvent(\tau, z, p, \epsilon) = \emptyset$, $z = z + \epsilon p$
  3. Else $t = 0$
  4. while $FirstUpdateEvent(\tau, z, p, \epsilon - t) \neq \emptyset$
     - $(z, e, I) = FirstUpdateEvent(\tau, z, p, \epsilon - t)$
     - $t = t + e$
     - $\tau \sim Z(\mathcal{N}(\tau, z))$
     - $p_I = -p_I$
  5. $z = z + (\epsilon - t)p$
  6. end if
  7. $p = p - \epsilon \frac{\delta U(\tau, z)}{2}$
- Output: chain $\{(z_i, p_i, \tau_i)\}_{i=0}^{n}$ with invariant distribution π.

They define e to be the time at which crossing occurs and I are the indices of the edges causing the cross. Looking at the above algorithm $Z(\mathcal{N}(\tau, z))$ is a randomly chosen new topology, this corresponds to choice 1 of subsection 5.7.4. Multiple crossings can occur in a single step, each crossing is mapped into a new topology according to the momenta it would have in the new topology, this choice corresponds to an ordered crossing with no dampning and no truncation asa described in subsection 5.7.1 and by the way momenta is reversed upon entering a new topology we can see that they treat the boundary as a reflecting boundary in subsection 5.7.1. So COrtHMC simplifies to the Leap-prog algorithm when we select momenta according to a $\mathcal{N}(0, \mathbb{I}_n)$ distribution, use the standard leapfrog algorithm, select an equal probability reflecting boundary with no dampning and allow multiple crossings.

5.5 Derivative of the Likelihood with respect to an Edge

In order to conduct HMC on Tree space we need to compute the derivative of the likelihood with respect to an edge to feed into the numerical computation of the flow. To do this we expand z into its components, the tree components ℓ and τ and the substitution and site rate components θ and α. We start with the Felsenstein pruning formulation of the likelihood as defined in section 2.5. Since the likelihood does not depend on the position of the root we can specify v_0 to be one of the internal vertices on the end of the edge of interest e. When the edge is a pendent edge there is only one possible choice for this vertex. Then

$$\begin{aligned}\frac{\partial}{\partial \ell(e)} \log\{\mathcal{L}(X|\ell,\tau,\theta,\alpha)\} =& \frac{\partial}{\partial \ell(e)} \sum_{i=1}^{n} \log\left(\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)\right) \\ =& \sum_{i=1}^{n} \frac{\frac{\partial}{\partial \ell(e)}\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)}{\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)}\end{aligned}$$

where $\mathbb{P}$ and $\underset{\sim}{\pi}$ are defined in section 2.5. As $\frac{\partial}{\partial \ell(e)}\mathbb{P}(k) = 0$

$$\begin{aligned}\frac{\partial}{\partial \ell(e)}\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha) =& \sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\frac{\partial}{\partial \ell(e)}\left(\underset{\sim}{\pi}(x)\mathcal{L}_{v_0}(\chi_i|x,\ell,\tau,\theta,\alpha)\right) \\ =& \sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\mathcal{L}_{v_0}(\chi_i|x,\ell,\tau,\theta,\alpha)\frac{\partial}{\partial \ell(e)}\underset{\sim}{\pi}(x) + \underset{\sim}{\pi}(x)\frac{\partial}{\partial \ell(e)}\mathcal{L}_{v_0}(\chi_i|x,\ell,\tau,\theta,\alpha).\end{aligned}$$

Since $\frac{\partial}{\partial \ell(e)}\underset{\sim}{\pi}(x) = 0$ and because we can choose v_0 such that edge e connects v_0 to a child of v_0 we shall call u, the following applies:

$$\begin{aligned}\frac{\partial}{\partial \ell(e)}\mathcal{L}_v(\chi_i|x,\ell,\tau,\theta,\alpha) =& \left[\prod_{u'\prec v}^{u'\neq u}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)|\theta\}\mathcal{L}_i^{u'}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right] \\ &\times\sum_{y\in\mathcal{A}}\frac{\partial}{\partial \ell(e)}\left(P_{x,y;k}\{\ell(u,v)|\theta\}\mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha)\right).\end{aligned}$$

We know that edge e is not contained in the subtree generated by u so $\frac{\partial}{\partial \ell(e)}\mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha)=0$. This means we can easily compute the numerator:

$$\frac{\partial}{\partial \ell(e)}\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)=\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u'\prec v}^{u'\neq u}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)|\theta\}\mathcal{L}_{u'}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right].$$
$$\times\sum_{y\in\mathcal{A}}\frac{\partial}{\partial \ell(e)}\left(P_{x,y;k}\{\ell(u,v)|\theta\}\right)\mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha)$$

$$P_{x,y;k}\{\ell(u,v)|\theta\}=\exp\{k\ell(u,v)Q\}\Rightarrow\frac{\partial}{\partial \ell(u,v)}P_{x,y;k}\{\ell(u,v)|\theta\}=kQ(\theta)P_{x,y;k}\{\ell(u,v)|\theta\}.$$

Putting everything together we get

$$\frac{\partial}{\partial \ell(e)}\log\{\mathcal{L}(X|\ell,\tau,\theta,\alpha)\}=\sum_{i=1}^{n}\frac{1}{\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)}\left[\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u'\prec v}^{u'\neq u}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)|\theta\}\right.\right.\right.$$
$$\left.\left.\left.\mathcal{L}_{u'}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\times\sum_{y\in\mathcal{A}}kQP_{x,y;k}\{\ell(u,v)|\theta\}\mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha)\right].$$

We notice that if we have previously computed the likelihood a lot of the L_u information is readily available. Our algorithms make use of this by storing the terms for each vertex.

5.6 Other results about COrtHMC

Theorem 5.3. *COrtHMC hits boundaries of codimension* 2 *with probability* 0 *for posterior* π *with a non-zero probability density function over the whole of tree space.*

Proof. Let X be a manifold of dimension n and let μ a probability measure on X such that the support of μ is X. We consider a posterior defined from the likelihood of a tree. We are interested in when a flow ϕ hits a submanifold of codimension 1. The flow ϕ is defined such that $\forall t\in\mathbb{R}, A\subset X\ \mu(\phi(t,A))=\mu(A)$. $\phi(t,\cdot)$ is a homeomorphism with respect to t. Let S be a submanifold of X such that $codim(S)=2$. We wish to show that

$$\mu(\{x:\phi([0,t],x)\cap S\neq\emptyset\})=0.$$

We use the notation $\phi([0,t],x)$ to mean $\bigcup_{s\in[0,t]}\{\phi(s,x)\}$. $\{x:\phi([0,t],x)\cap S\neq\emptyset\}$ is the set of $x\in M$ such that x hits S at some point while following flow ϕ.

We know ϕ is a measure preserving homeomorphism for each t. This implies

$$\mu(\{x : \phi([0,-t],x) \cap S \neq \emptyset\}) = \mu(\{x : x \in \phi([-t,0],S)\}).$$

If the codimension of a submanifold is greater than 0 and the measure has support across the whole of the manifold the measure of the submanifold is also 0 (Lee, 2013).

We now demonstrate that the codimension of $\phi([t,0],S)$ is 1 for all t. To show this define a homeomorphism mapping local subspaces of $\phi([0,-t],S)$ into $\mathbb{R}^{n-1}$. Define the homeomorphism in the following way: Let p be any point in S then there exists U an open neighbourhood of p such that there is a mapping $\psi_U : U \to \mathbb{R}^{n-2}$. Let τ be a point in $[-t,0]$. Define the homeomorphism φ:

$$\varphi : \phi[\tau,p] \to \mathbb{R}^{n-1} \text{ by } \phi[(\tau-\delta,\tau+\delta),U] \mapsto \tau \times \psi_U(p)$$

This is a chart and homeomorphism as open set are mapped to open sets in an invertible way:

$$\phi[(\tau-\delta,\tau+\delta),U] \mapsto (\tau-\delta,\tau+\delta) \times \psi_U(U)$$

Hence $\phi([-t,0],S)$ is a submanifold of dimension $n-1$ meaning that $codim(\phi([-t,0],S)) = 1$ and therefore $\mu(\{x : \phi([0,\infty],x) \cap S \neq \emptyset\}) = 0$. □

Theorem 5.4. *The numerical integrator for HMC lands on a boundary of codimension* 1 *with probability* 0.

This follows from the arguments above. We take S to be a submanifold of codimension 1 and apply ϕ to a point, as the numerical integrator picks out discrete points on flow lines.

Theorem 5.5. *The likelihood as defined in section 2.5 is smooth across codimension* 1 *strata.*

Proof. We outline the ideas behind the proof, for the exact details please see Appendix B. The likelihood is smooth within an orthant. At the boundaries of orthants the trees can be viewed as belonging to either orthant but are equivalent. Viewing the whole of tree space the likelihood is continuous through any codimensional space and the derivative agrees at the boundary when being approached from either orthant. We can see this by viewing the path from one orthant to another embedded in $\mathbb{R}^n$ as in 5.4. □

5.7 Methods for Crossing Codimensional Boundaries

We have designed several methods for crossing codimensional boundaries in order to produce the kernel C. Each method requires several user choices. We focussed on how to transform the

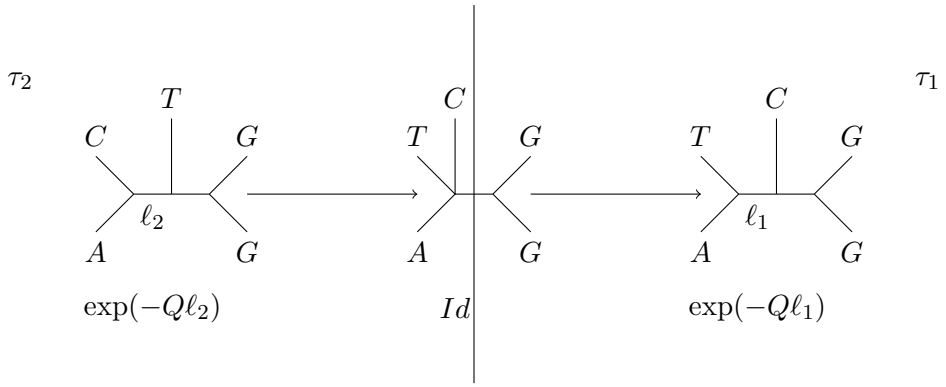


Figure 5.4: The three trees show different stages of smoothly transforming a tree across the boundary from topology τ_2 to τ_2. The edge ℓ_2 shrinks to a point and a nearest neighbour interchange occurs before it grows out to ℓ_1. Beneath each tree is the term associated with the edge expressed in the likelihood. It is clear to see that as $\ell_2 \mapsto 0 \mapsto \ell_1$ smoothly, $\exp(-Q\ell_2) \mapsto Id \mapsto \exp(-Q\ell_1)$ smoothly.

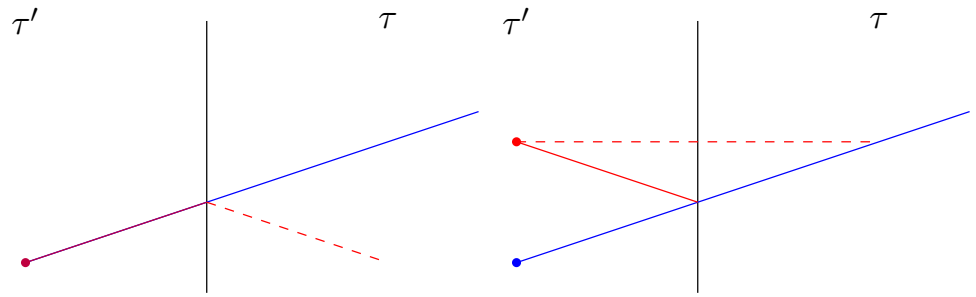
edge lengths upon crossing a boundary, how to transform the momenta, and whether or not to continue the integration upon entering a new orthant. When we generate a proposal across a boundary we perform an initial "jump" as seen in subsection 5.7.1, then transform this new state according to chosen momenta and by integrating along the flow, subsection 5.7.3 and subsection 5.7.2.

Hamiltonian Monte Carlo makes use of local information. Standard MCMC uses NNI, SPR and TBR, as defined in section 3.4, to perform topological moves. Of these NNI is guaranteed to move to a neighbouring topology, a topology sharing the spine of the open book. It therefore makes no sense to use SPR or TBR when conducting HMC. We use NNI because when conditioned on the topology you move into NNI provides a smooth flow.

5.7.1 Transforming Edges: Reflecting vs Projecting across the Boundary

We decide to transform edge lengths in one of two ways. A single step of the leapfrog algorithm for the state is a linear map. This map crosses a boundary when an edge's length becomes negative. This leads to two natural methods for transforming edge lengths. The first seen in Figure 5.5 takes the edge lengths to be the same as if the leapfrog algorithm reflected off the boundary. Upon crossing into a new topology a nearest neighbour interchange is performed. During a NNI one edge ceases to exist and a new one is created. This new edge is given the length of the old edge. We call this the reflecting boundary. The other method projects the state back along the path it came from upon hitting a boundary. The new edge lengths are then taken to be the edge lengths at this point and the new edge is given the length of the old edge. This

can be seen in Figure 3.3 and is called the projecting boundary.



Reflecting boundary Projecting boundary

Figure 5.5: The figure on the left depicts a reflecting crossing, the right a projecting crossing. The dotted red line gives the reflection or projection, the blue line and ball is the single step process of the leapfrog integrator assuming the axis is for when the edge has 0 length and the red line and ball is the proposed point.

We shall denote the transformation $\mathfrak{R}$ as the reflecting boundary and $\mathfrak{P}$ as the projecting boundary. Let $\ell = \{\ell_1, \ldots, -\ell_i, \ldots, \ell_n\}$ then $\mathfrak{R}$ is easy to compute.

$$\mathfrak{R}(\{\ell_1, \ldots, -\ell_i, \ldots, \ell_n\}) = \{\ell_1, \ldots, \ell_i, \ldots, \ell_n\}.$$

The projection is harder to compute as it requires knowledge of the gradient $\frac{\partial H}{\partial p}$.

The distance of integration to the intersection, d, is first computed $d = \frac{\ell_i}{\epsilon \frac{\partial H}{\partial p}}$ then:

$$\mathfrak{P}(\{\ell_1, \ldots, -\ell_i, \ldots, \ell_n\}) = \{\ell_1 - 2d\epsilon\frac{\partial H}{\partial p}, ..., \ell_i, ..., \ell_n - 2d\epsilon\frac{\partial H}{\partial p}\}.$$

Multiple crossing

We cover here how we design C when during a single step of the leapfrog algorithm multiple edges become negative. This is visualised in Figure 5.6 and notionally represents crossing two or more boundaries. When the edges are in two distant subtrees, crossing two boundaries in one step is motivated by the underlying flow due to the total evolutionary distance varying much more locally. When the edges are neighbouring the effect of the nearest neighbour interchange on one can have considerable impact on the direction of the flow and therefore two edges should

not be crossed at the same time.

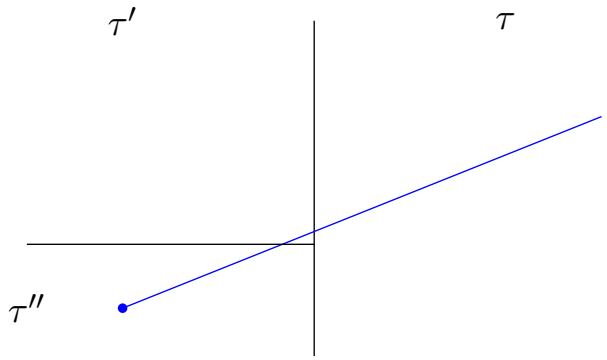


Figure 5.6: Crossing two orthants in a single step

We now discuss a few different ways to cross multiple boundaries. We can truncate the flow so that it only crosses into one orthant. If we adopt this method then we have to decide which edge to prioritise and by how much to truncate the flow. The advantage of this is method is that it simplifies the crossing algorithm and we do not skip over topologies. The disadvantage is that the next iteration will often cross the boundary anyway. If multiple edges are assigned negative values we have to decide on the order to cross them. An intuitive way is to cross in order the edges hit 0. The correct ordering is important. Suppose we choose to select orthants in an unordered manner. Unordered reflection $\mathfrak{R}$ increases the number of potential orthants it is possible to move into, the decision changes from 2^n choices to $n!2^n$ choices, where n is the number of orthants crossed. While this is not a problem, it isn't a very directed approach. Under an ordered approach the space of potential topologies reachable is greatly reduced and is informed by the direction of travel in the original topology τ, see Figure 5.7. If the order is ignored let a random negative edge be chosen. If $\mathfrak{P}$ is applied as normal ignoring all other edges, we then check to see if any edge is negative and repeat the process until there are no remaining negative edges. Figure 5.7 demonstrates that by applying unordered projection we can transform a negative edge to one that is positive, potentially undoing the first or most influential nearest neighbour interchange. When order is respected, with each edge that goes negative projection spirals towards high codimensional spaces as can be seen in Figure 5.7. This leads us to conclude that the best approach is to simply truncate the leapfrog approximation of the flow so that it only crosses one boundary in the rare cases when it crosses multiple boundaries.

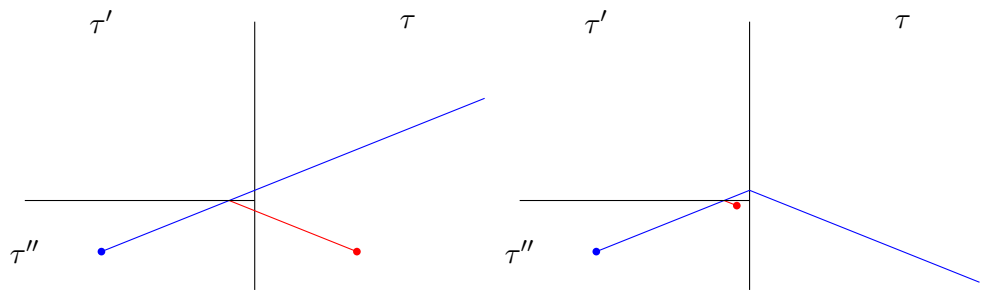


Figure 5.7: Here we see how an unordered projection will result in a negative edge becoming positive (left) and how an ordered projection will result in movement towards the origin (right).

5.7.2 Continuing the Integration

Once an NNI has been conducted from a method proposed in subsection 5.7.1 we can choose to propose the new point, or take the new point as a starting point for continuing on the leapfrog integration, the end of which we take as the proposed point. From section 5.3 and section 2.4 of Dinh et al. (2017) we can see it is possible to stop the integration during the integration process upon an edge becoming negative.

5.7.3 Resetting Momenta

The momenta can be changed upon crossing a boundary. If we maintain the density then the momenta are consistent from orthant to orthant, we can see this in Figure 5.8 in the figure titled Free-falling. In this figure the flow starts in the black orthant with no momenta and is allowed to flow according to the Hamiltonian without any outside influence to the momenta. We specify that it moves towards the orthant generating the alignment, the exact tree is specified by a purple dot. The orthant boundaries are shown in cyan. In Figure 5.8, Free-falling, we can see how if we allow the state to free fall the momenta carry the state past the region of interest. We decided to "parachute" the momenta where we abruptly stop the momenta i.e. set it to zero, upon entering a new orthant, as if we pulled a parachute upon entering a new part of the atmosphere. We can see three key effects occurring in Figure 5.8, Parachuting, the states condense, all the flows explore the region about the tree generating the alignment and the flows spend much less time in orthants that are not the generating orthant. In Figure 5.8 we can see that if we "parachute" the momenta we have more rapid convergence to the correct orthant (see the scale relative to Free-falling) and from the acceptance probability of COrtHMC we can see that moving into new orthants is favoured since if the momenta have mean 0, resetting the momenta increases the acceptance probability. Resetting the momenta, apart from affecting the acceptance probability,

only really has an effect on the proposed point if we continue the integration after crossing a codimensional boundary. This is because at the start of each HMC and COrtHMC iteration the momenta are resampled.

5.7.4 Choosing a Topology

It is possible to enter different topologies with varying probability. We therefore have three methods:

1. We choose with probability $\frac{1}{2}$ one of the two new topologies to enter.
2. We choose with probability $\frac{1}{3}$ one out of the two new topologies and the original topology to move into.
3. We choose which new topology to move into by a ratio of the posteriors.

These all explore new topologies, focussing on only the new topologies results in greater movement to new topologies and choosing by posterior results in favouring moves to regions with greater posterior density. This means that for rapid convergence a posterior new topology system will be preferable whereas once we have converged it might be better to go with a more randomized approach. In the results section we actually see that due to the continuous nature of the posterior across codimension 1 spaces that most crossing methods perform comparably.

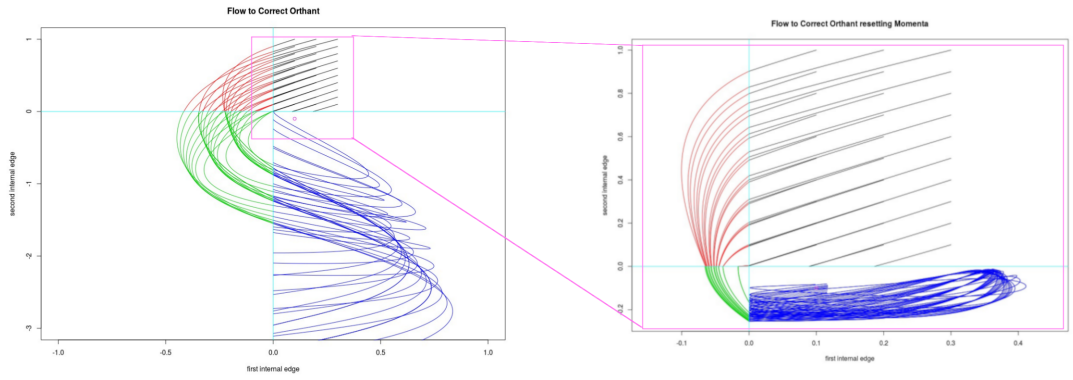


Figure 5.8: This figure demonstrates the difference between parachuting and free-falling on the movement between four topologies represented by black, red, green and blue lines in four different quadrants. The integration starts at the black points with 0 momenta, the tree used to generate the alignment is represented by a purple dot and is in the blue quadrant. The starting points are the same in both images, the scale for parachuting has been expanded as it occupies approximately an eighth of the scale of free-falling. For comparison a version to the same scale as Free-falling has been placed alongside it. The cyan lines show codimensional boundaries. We have fixed which topology we move into at every boundary so that the state converges to the generating topology.

5.8 Adapting COrtHMC

When the flow remains in an orthant COrtHMC propopses the same state as standard HMC and uses the same mechanism to propose that state. As the proposal mechanism is the same within an orthant, pre-existing adaptations can be applied to the proposal mechanism. These adapted proposal mechanisms leave π invariant, and 5.3 will hold for the new HMC component J. We implemented RM-HMC, subsection 4.3.3, in particular as this improves the direction of the flow in high dimensional spaces. We provide a brief discussion as to why we chose not to implement NUTS. In chapter 6, we cover some issues with implementing constrained HMC. A brief overview of constrained HMC can be found in subsection D.5.

5.8.1 RM-HMC

In tree space the computational bottleneck comes from computing the likelihood and derivatives of the likelihood. This is because it is a recursive algorithm of order $\mathcal{O}(|\chi||k|(2n-3)|\mathcal{A}|^2)$ where $|\chi|$ is the number of columns, $|k|$ is the number of categories for the site rate parameter, $2n-3$ is the number of edges and $|\mathcal{A}|$ is the size of the alphabet, in our case 4. RM-HMC requires not only the derivative of the likelihood to be calculated but all the second derivatives of it, see subsection 4.3.3 Equation (7). This increases the total computational time by a factor of $(2n-3)^2$ making it an $\mathcal{O}(n^3)$ order algorithm. We can reduce the order by storing parts of the computation, but it will still take more than $2n-2$ times longer than computing all the derivatives. This is impractical for HMC as it would remove all competitiveness from the algorithm. In RM-HMC the second derivatives are only necessary for integrating along the flow. We construct a kernel based on that in RM-HMC, which does not require second derivatives to integrate along the flow but produces a less exact integration.

5.8.2 Adapting RM-HMC to Tree Space

The kernel for momenta RM-HMC employs improves the exploration of HMC. Our aim is to choose the same kernel as dictated by RM-HMC but without continually transforming it around the space, the process that requires second derivatives. COrtHMC has obvious resemblances to adaptive schemes when the kernel K is dependent on state z. This is the case when using the Fisher information matrix as a covariance matrix for a normal distribution centred about the origin. Andrieu and Thoms (2008) show that a significant issue with adaptive schemes is that they subtly alter the invariant distribution to be dependent on the adaptation, resulting in the incorrect distribution being targeted. Common constructions used to ensure ergodicity and that the chain eventually targets the correct distribution require diminishing adaptation with either

the simultaneous uniform ergodicity condition or containment condition (Roberts and Rosenthal, 2007) and (Bai et al., 2011). If we were to require these conditions for the Fisher information matrix F we could adapt it to $\mathcal{F}$ by either of the following methods. We define this adapted $\mathcal{F}$ here iteratively:

$$\mathcal{F}_{n+1} = \frac{\mathcal{F}_n \times n + F_{n+1}}{n+1}$$
$$\{\mathcal{F}_{n+1}\}_{ij} = \begin{cases} \{F_{n+1}\}_{ij}^{\frac{1}{n}} & \text{if } i = j \\ \{F_{n+1}\}_{ij}^{\frac{1}{n}} - 1 & \text{otherwise} \end{cases}$$

However we have shown in section 5.3 that COrtHMC targets the correct distribution even when K is dependent on state z, and so do not require such an adaptive method to be implemented. RM-HMC also has some resemblance to adaptive methods, Girolami et al. (2011) who also avoid adapting the kernel by implicitly having the kernel as part of the state space over which they are conducting HMC.

Computing the Fisher Information Matrix

The Fisher Information matrix can be written in the following way for an alignment X with n columns:

$$F_{jk} = \sum_{i=1}^{n} \frac{\partial \log(\mathcal{L}(\chi_i|x))}{\partial x_j} \frac{\partial \log(\mathcal{L}(\chi_i|x))}{\partial x_k} \mathcal{L}(\chi_i|x)$$

where x is a parameter of interest.

When we compute the likelihood and the derivative with respect to a parameter we already compute $\frac{\partial \log(\mathcal{L}(\chi_i|x))}{\partial x_k}$. This means that we can avoid computing them again if we store the derivatives for each χ_i speeding up the computation of the Fisher Information Matrix. This is because all that is required is the final multiplication and sum, taking a minor amount of time.

Stability of the Kernel

The kernel is the covariance matrix for a normal distribution, so it is necessary for it to be invertible. Sometimes particular derivatives can become very small and on a few runs this meant that numerically the matrix became unstable and threw computational errors. We wish to have a stable kernel so we decided to try two methods, both of which prevented the error from occurring again. We call this new stabilised kernel $\mathcal{F}$. These are applied globally over the algorithm to avoid numerical issues:

1. $\mathcal{F}_{ij} = \begin{cases} F_{ij} \text{ if } i = j \\ 0 \text{ otherwise} \end{cases}$, and

2. $\mathcal{F}_{ij} = \begin{cases} 2F_{ij} \text{ if } i = j \\ F_{ij} \text{ otherwise} \end{cases}$.

5.9 Discussion of other methods

We have presented one way of constructing an algorithm to traverse Tree space using HMC. This is not the only method available. It is possible to embed Tree space into a Euclidean space. With this embedding we can then move in the super Euclidean space and project back into Tree space. Another method is to "tube". Around each orthant add an auxiliary variable creating a tube around the orthant. The value of the auxiliary variable, for example if it is positive or negative, dictates the chirality of the NNI transformation applied.

5.9.1 Tubing

Consider the following roughly designed proceedure. We adjoin an auxiliary variable to our state space and define a distribution over this auxiliary variable, for example a $N(0, 1)$ distribution. As we run the HMC algorithm this variable generates samples from the normal distribution. The value of this random variable dictates which orthant to move into. This is user defined. It is preferable that the user defines the orthant moves so that returning to the same codimensional boundary results in a new orthant being explored. The end result of this method is the longer we spend exploring an orthant the less deterministic and more random the choice of the next orthant is. We decided against such a method for several reasons.

1. As seen in subsection 5.1.1 the posterior directs us to the correct orthant.

2. The user defines the distribution, which effectively calibrates how quickly it transforms from a roughly deterministic process to a random one. This can introduce unwanted biases to the process.

3. It adds another parameter.

We summarise that tubing adds extra hassle to the process, increases computation time, and introduces a variable which can have significantly negative impact if mishandled. To make best use of this method, we would need a rough understanding of how long the chain spends in each orthant, which is one of the original aims of performing the inference.

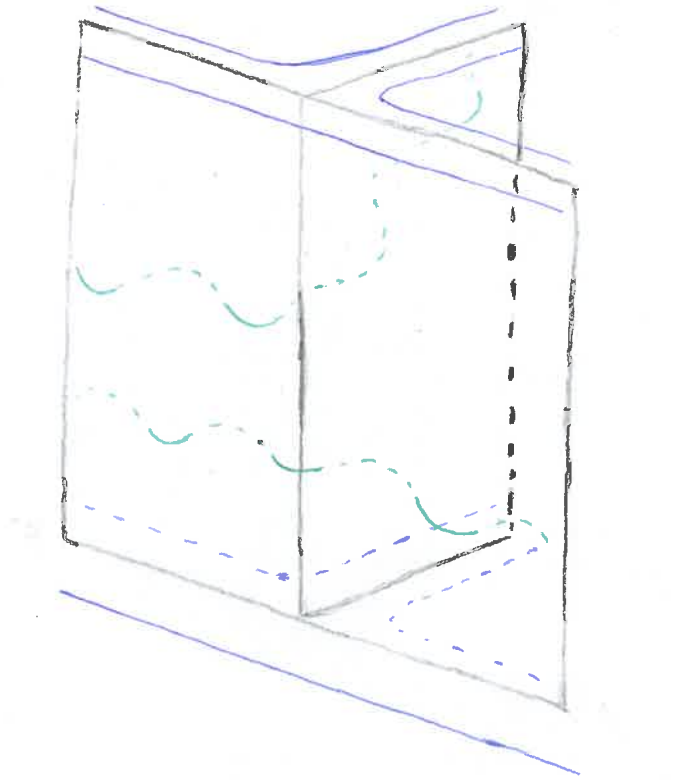


Figure 5.9: Here we depict the open book in black. The tubes around the pages of the open book representing the added auxiliary variable are pictured in blue. We can see two flows in green oscillating from side to side. The side at the codimension 1 space dictates which orthant of the open book the flow enters.

5.9.2 Embedding

We can embed Tree space into Euclidean space. We can show that the dimension of this Euclidean space is bounded between $2n-3$, the number of edges contained in a tree in $\mathcal{T}_n$ and n^2. To see that Tree space $\mathcal{T}_n$ can be embedded into a Euclidean space of dimension $\frac{n(n-1)}{2}$, we can construct any tree from all the path lengths between taxa contained in the tree. We don't know of any embedding of order less than this.

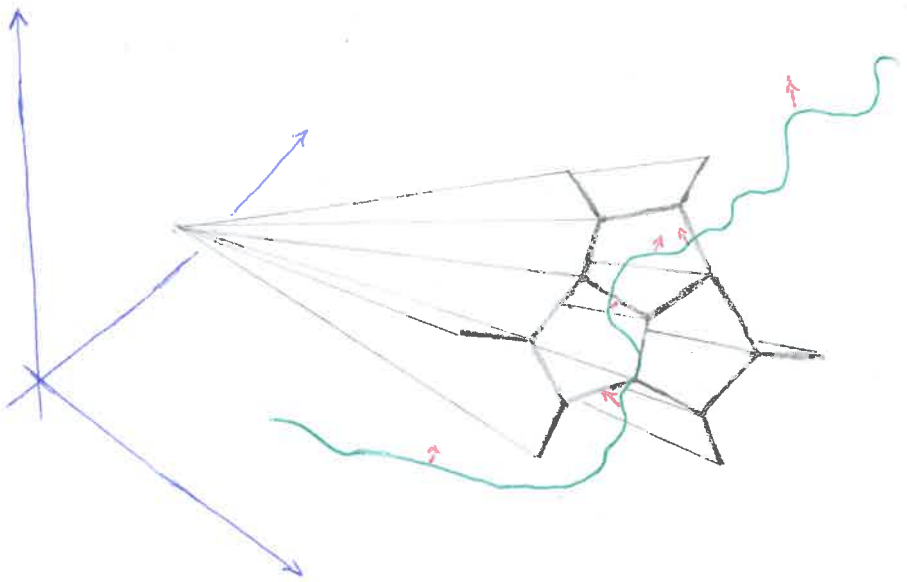


Figure 5.10: Here we depict Tree space in black. The Euclidean space we embed it in is pictured in blue. We can see the flow in green is travelling in the Euclidean space. The projection from the flow to Tree space is in red.

This method has one benefit but several reasons why it is not feasible. The benefit is that it enables geometrically motivated topological moves that are greater than nearest neighbour interchanges. The problem is that it requires every iteration to be projected back onto Tree space, (this is costly and would give discontinuous flows see Figure 5.11). It requires us to define a consistent and useful cotangent space, (this is impossible to do everywhere). On the almost everywhere region where this is possible, the "honeycomb" structure of Tree space has to have the distribution extended onto it. This results in a posterior function that is greatly more complex, involving the posteriors of projections to neighbouring topologies, if it wished to be informative. It also converts an $\mathcal{O}(n)$ dimensional problem into an $\mathcal{O}(n^2)$ dimensional one. It should therefore be clear that such a method could never hope to be competitive. In the case where the structure is filled in by an arbitrary posterior this method would then act as a costly form of a two step HMC within Gibbs sampler, sampling a HMC update and then a topological update.

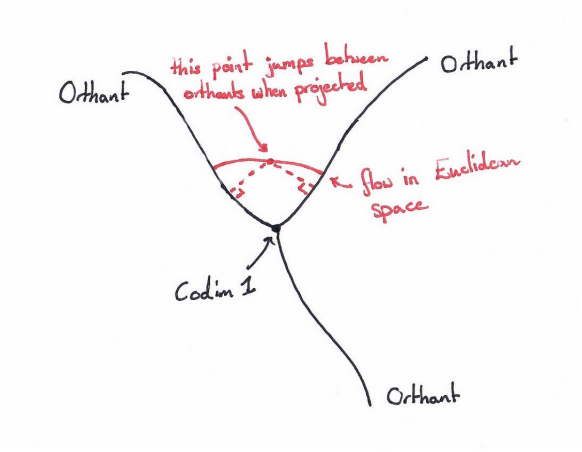


Figure 5.11: Here by imbedding Tree space in Euclidean space we encounter a discontinuity in the flow when projecting from Euclidean space to Tree space..

Chapter 6

HMC on Substitution Model Parameters and Site Rate Parameters

In chapter 5 we covered how we adapt HMC to work on Tree space, and in chapter 2 we covered various parametric schemes for carrying out inference. One of the key benefits in conducting HMC over standard Metropolis within Gibbs is that we can lift the work in chapter 5 to enable simultaneous updates of the model parameters and the tree incorporating the dependency between the two. The model parameters and tree are dependent on one another through the likelihood defined in section 2.5 and so HMC provides an update which does not fall into the standard problems Gibbs encounters with highly correlated parameters. Some adaptation is necessary to convert the substitution model parameters and site rate parameters into a form applicable to COrtHMC. In chapter 2 we saw that some substitution model parameters lie on the simplex and that some lie on $\mathbb{R}_+$. HMC, in its basic form, acts on copies of $\mathbb{R}$. This means that we have to transform the parameters into something that HMC and CortHMC can handle.

6.1 Computing the Derivative of the Likelihood with respect to Substitution Model Parameters

Before outlining exactly how to transform the parameters, we describe how to compute the derivative of the loglikelihood with respect to a general substitution model parameter. Unlike the computation in subsection 5.5 the choice of root v_0 does not matter so long as it is an internal vertex. In our code we specify a root to the tree which is used in computing the

likelihood. We choose the root used for calculating the derivative with respect to substitution model parameters v_0 to be consistent with the root of the tree as this saves minor computational time and stops unnecessary confusion. The time save comes from subtree likelihoods having been already computed based on a recursion from v_0 and can be reused in the derivative calculation. We use the same notation as in 5.5. We use ϕ to denote a general substitution model parameter. We discriminate between composition parameters and transition/transversion parameters later on. Using the definition of the likelihood found in section 2.5 we compute the derivative of the loglikelihood as follows:

$$\begin{aligned}\frac{\partial}{\partial\phi}\log\{\mathcal{L}(X|\ell,\tau,\theta,\alpha)\} =& \frac{\partial}{\partial\phi}\sum_{i=1}^{n}\log\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)\\ =& \sum_{i=1}^{n}\frac{\frac{\partial}{\partial\phi}\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)}{\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)}.\end{aligned}$$

Since $\frac{\partial}{\partial\phi}\mathbb{P}(k)=0$ and from the definition of $\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha)$ in section 2.5 we have that

$$\begin{aligned}\frac{\partial}{\partial\phi}\mathbb{P}(\chi_i|\ell,\tau,\theta,\alpha) =& \frac{\partial}{\partial\phi}\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\mathcal{L}_{v_0}(\chi_i|x,\theta,k)\\ =& \sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\left[\mathcal{L}_{v_0}(\chi_i|x,\theta,k)\frac{\partial}{\partial\phi}\underset{\sim}{\pi}(x)+\underset{\sim}{\pi}(x)\frac{\partial}{\partial\phi}\mathcal{L}_{v_0}(\chi_i|x,\theta,k)\right].\end{aligned}$$

There are two derivatives to compute in this calculation, $\frac{\partial}{\partial\phi}\mathcal{L}_{v_0}(\chi_i|x,\theta,k)$ and $\frac{\partial}{\partial\phi}\underset{\sim}{\pi}(x)$. Since $\underset{\sim}{\pi}(x)$ is a substitution model parameter its derivative is easy to compute. $\mathcal{L}_{v_0}(\chi_i|x,\theta,k)$ is defined in section 2.5 as $\prod_{u\prec v}\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)$. We define $u\prec v$, $P_{x,y;k}\{\ell(u,v)\}$ and $\mathcal{L}_u(\chi_i|y,\theta,k)$ at the start of 2.5. The product rule applied to $\frac{\partial}{\partial\phi}\mathcal{L}_{v_0}(\chi_i|x,\theta,k)$ gives:

$$\begin{aligned}\frac{\partial}{\partial\phi}\mathcal{L}_v(\chi_i|x,\theta,k) =& \sum_{u\prec v}\left[\prod_{u'\prec v}^{u'\neq u}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)\}\mathcal{L}_i^{u'}(\chi_i|y,\theta,k)\right]\right]\\ &\times\sum_{y\in\mathcal{A}}\frac{\partial}{\partial\phi}\left(P_{x,y;k}\{\ell(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)\right).\end{aligned}$$

We are now left with the derivative $\frac{\partial}{\partial\phi}\left(P_{x,y;k}\{\ell(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)\right)$ to compute. The product rule gives:

$$\frac{\partial}{\partial\phi}\left(P_{x,y;k}\{\ell(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)\right)=\frac{\partial}{\partial\phi}P_{x,y;k}\{\ell(u,v)\}\times\mathcal{L}_u(\chi_i|y,\theta,k)+P_{x,y;k}\{\ell(u,v)\}\frac{\partial}{\partial\phi}\mathcal{L}_u(\chi_i|y,\theta,k).$$

We wish to compute $\frac{\partial}{\partial\phi}P_{x,y;k}\{\ell(u,v)\}$. This is not necessarily algebraically possible. In subsection 2.2.2 we noted that GTR does not admit $\frac{\partial}{\partial\phi}P_{x,y;k}\{\ell(u,v)\}$ in algebraic form. We chose to use HKY85 and we covered the algebraic form of the transition matrix for HKY85 in subsection 2.2.3. We need an algebraic form for the transition matrix rather than the rate matrix (as needed for edges) because differentiating with respect to substitution model parameters does not have the same nice form as differentiating with respect to edge lengths:

$$P_{x,y;k}\{\ell(u,v)\} = \exp\{k\ell(u,v)Q\} \not\Rightarrow \frac{\partial}{\partial\phi}P_{x,y;k}\{\ell(u,v)\} = k\ell(u,v)P_{x,y;k}\{\ell(u,v)\}\frac{\partial}{\partial\phi}Q.$$

The analysis above tells us that $\frac{\partial}{\partial\phi}\log\{\mathcal{L}(X|\theta)\}$ depends on information we have previously computed and two derivatives, $\frac{\partial}{\partial\phi}\mathbb{P}(\hat{\chi}(v_0) = x)$ and $\frac{\partial}{\partial\phi}P_{x,y;k}\{t(u,v)\}$. When we choose the reparameterisation of the model parameters it is important that it enables us to compute these two derivatives effectively.

We also notice that

$$\left[\prod_{u'\neq u}^{u'\prec v_0}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)\}\mathcal{L}_i^{u'}(\chi_i|y,\theta,k)\right]\right] = \frac{\mathcal{L}_{v_0}(\chi_i|x,\theta,k)}{\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v)\}\mathcal{L}_i^{u}(\chi_i|y,\theta,k)}$$

since

$$\mathcal{L}_{v_0}(\chi_i|x,\theta,k) = \prod_{u'\prec v_0}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)\}\mathcal{L}_i^{u'}(\chi_i|y,\theta,k)\right].$$

When calculating the derivative for substitution parameters we do not vary v_0. This means that the numerator and the summands in the denominator have already been computed. This leads to considerable time saving over the course of a full run of HMC.

6.2 HMC on the Simplex

In subsection 2.2.1 the simplex is defined as $\Delta_n = \{(\pi_1,\ldots,\pi_n)|\sum_{i=1}^{n}\pi_i = 1\}$. For phylogenetic inference the base rates are positive and represent the proportion of the base occurring at stationarity. It follows that each parameter is bounded between 0 and 1. We work with the DNA base rate parameters π_A, π_C, π_G and π_T. These lie on Δ_4. Working on the simplex complicates HMC in two ways; the parameters do not exist on copies of $\mathbb{R}$ and are related by a constraint. This constraint affects the derivative of the log-likelihood as its calculation depends on how we decide to formulate the dependency between π_A, π_C, π_G and π_T. We tried two methods of

transforming the parameters; a method involving trigonometric functions and constrained HMC from Betancourt (2012) and an exponential method from Heaps et al. (2014). The method of Heaps et al. (2014), although requiring more operations, was easier to implement and avoided some issues that occurred.

6.2.1 Cruising the Simplex

In Betancourt (2012) the parameters π_i are transformed so that $y_i^2 = \pi_i$ for $i = 1, \ldots, n$, where n is the number of parameters lying in the simplex. The y_i are then transformed using a hypersphere transformation

$$y_i = \prod_{j=1}^{n} \sin(\theta_j) \times \begin{cases} \cos(\theta_i) & \text{if } i < n \\ 1 & \text{otherwise} \end{cases} .$$

In order to simplify the mathematics the θ_i are further transformed to $z_i = sin^2(\theta_i)$. Following everything through we get:

$$\begin{aligned} \pi_i = y_i^2 &= \prod_{j=1}^{n} \sin(\theta_j)^2 \times \begin{cases} \cos(\theta_i)^2 & \text{if } i < n \\ 1 & \text{otherwise} \end{cases} \\ &= \prod_{j=1}^{n} z_k \times \begin{cases} 1 - z_i \text{ if } i < n \\ 1 \text{ otherwise} \end{cases} \\ \Rightarrow z_i &= \frac{1 - \sum_{j=1}^{i-1} \pi_j}{1 - \sum_{j=1}^{i} \pi_j} \text{ for } i < n \text{ so that there is one less } z_i. \end{aligned}$$

The chain rule implies the derivatives with respect to the transformed variables z_i are

$$\frac{\partial f}{\partial z_i} = \left(\frac{\pi_i}{z_i - 1} \right) \frac{\partial f}{\partial \pi_i} + \sum_{j=i+1}^{n} \frac{\pi_j}{z_i} \frac{\partial f}{\partial \pi_j}$$

for further details see (Betancourt, 2012).

The transformation above means that z_i, like π_i are bounded between 0 and 1. Hence, when conducting HMC we require the use of constrained methods such as those described in subsection D.5. The use of constrained HMC can cause problems when there are constraints in multiple dimensions. When we deal with Δ_4 the space for the transformed base rates is $[0, 1]^3$. We shall describe the problem in terms of $[0, 1]^2$ to allow illustration.

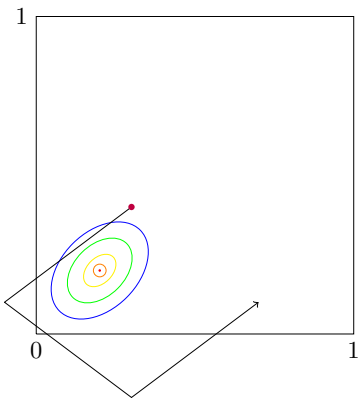


Figure 6.1: This figure exaggerates the flaw of constrained HMC that can result in the chain getting stuck. The density is depicted by the ellipses, the various arrows depict possible paths under constrained HMC from the original state in purple. We can see here that even if we propose moving towards regions of high density the dynamics involved can result in catapulting the point to a region of low density.

When conducting HMC using a numerical integrator it is important to choose the correct stepsize, too small and exploration is slow, too large and the integration error will continuously reject proposals. Constrained HMC can increase the integration error and therefore aggravate any issue caused by too large a stepsize. The large and mostly unknown nature of the posterior over tree space means that appropriately adapting the stepsize to deal with this issue is difficult. We have found that the optimum stepsize does vary over Tree space. Constrained HMC is particularly problematic because of a dissociation from the underlying principals of HMC. In constrained HMC the gradient informs the integration along a flow faithfully up until the boundary is hit. Once the boundary is hit, unlike when hitting a codimension-1 boundary in Tree space, most methods of constrained HMC switch from one integral curve to another. Different integral curves hae different values for the Hamiltonian and different derivates of the Hamiltonian. This means that constrained HMC can cause us to continuously land in regions of low density. At this point the algorithm gets stuck for a period of time before moving on. Such stepsize problems appear throughout HMC. The issues depicted in Figure 6.1 only become more extreme in higher dimensions.

Constrained HMC has some similarities to COrtHMC in the above. We also allow the flow to switch from one integral curve to another in COrtHMC. This is done in the crossing kernel C of Algorithm 7. As a consequence we can create choices of kernel that produce poor results. We notice that in the choices of crossing kernel we provide the proposed point always attemps to

maintain the marginal density $\pi(z)$ and never to decrease the density of $\pi(z, p)$. This avoids the above problem and may even improve mixing. We preserve the density by maintaining the gradient of the parameters from one step to the next.

The transformation in Betancourt (2012) is computationally cheap. The method requires the parameters to be constrained to a cube. As a form of constrained HMC it can encounter the problems depicted in Figure 6.1. In phylogenetics the substitution model parameters are dependent on the tree. After each step the distribution changes and may vary substantially between topologies, which makes accepting new states in different topologies particularly difficult. Hence constrained HMC schemes such as that of Betancourt (2012) may require locally very different stepsizes to avoid a lack of mixing. This problem is further aggravated in the case of small alignments as it is common for the likelihood to dictate that a lot of the density is situated close to the boundary of the simplex and therefore close to the boundary of the transformed space in Betancourt (2012). This means in situations where the alignment is small the transformation in Betancourt (2012) loses efficiency. We note that in inference on Tree space the bottleneck in the computation is in the likelihood. This means the marginal gains from performing the transformation in Betancourt (2012) are not worth the time it spends stuck. To avoid the above issue we decided to use a different transformation.

It may be of interest that Lan et al. (2013) covers a method that removes the constraints by transitioning in a similar way to a spherical system. We decided against this approach as we had doubts about the possibility of using it in conjunction with our form of RM-HMC.

6.2.2 The Farrow Transformation

Instead of the transformation found in Betancourt (2012) we chose to use the transformation found in Heaps et al. (2014). We call this transformation the Farrow transformation, after Malcolm Farrow the famous Bayesian Statistician.

Definition (The Farrow Transformation)**.** It is $\mathcal{F}$ is a map from $\mathbb{R}^{n-1}$ to the n-dimensional simplex

$$\Delta^n = \{\pi : \sum_{i=1}^{n} \pi_i = 1\}.$$

We define the Farrow transformation $\mathcal{F}$ as the composition of a linear map H sending $\mathbb{R}^{n-1}$ to the hyperplane $\{(a_1, ..., a_n) | a_1 + ... + a_n = 0\} \subset \mathbb{R}^n$ and another transformation $\mathfrak{F}$ taking

$\{(a_1, ..., a_n)|a_1 + ... + a_n = 0\}$ to Δ^n so that:

$$\mathcal{F}(\beta) = \mathfrak{F}(H\beta).$$

The matrix for H is the $n \times n-1$ dimensional matrix

$$h_{ij} = \begin{cases} 0 \text{ if } i < j \\ d_j \text{ if } i = j \\ -d_j/(n-j) \text{ if } i > j \end{cases}$$

where

$$d_1 = 1 \qquad \& \qquad d_j = d_{j-1}\sqrt{1 - \frac{1}{(n-j+1)^2}}, \;\; j = 2, \ldots, n-1.$$

It can be shown that $\sum_{i=1}^{n} h_{ij} = 0$ and therefore $H\beta$ is an n dimensional vector which sums to 0. We now define the transformation $\mathfrak{F}$ to be:

$$\mathfrak{F}(\mathbf{a}) = \left(\frac{\exp(a_1)}{\sum_{i=1}^{n} \exp(a_i)}, \ldots, \frac{\exp(a_n)}{\sum_{i=1}^{n} \exp(a_i)} \right).$$

We notice that for each i, $0 < \frac{\exp(a_i)}{\sum_{j=1}^{n} \exp(a_j)} < 1$. This means that this transformation never maps to the boundary case $\pi_i = 0$ or $\pi_i = 1$. This in turn means that the transformed derivative is not inclined to map to the boundary.

Theorem 6.1. *The Farrow transformation is bijective.*

Proof. (Injectivity)

We are going to show that if $\mathcal{F}(\beta) = \mathcal{F}(\beta')$ then $\beta = \beta'$. First we show that $\mathfrak{F}(a) = \mathfrak{F}(a')$ implies $a = a'$ for $a \in \{(a_1, ..., a_n)|a_1 + ... + a_n = 0\}$. We then show that $H\beta = H\beta'$ implies $\beta = \beta'$.

Suppose $\mathfrak{F}(a) = \mathfrak{F}(a')$, where a and a' are in $\{(a_1, ..., a_n)|a_1 + ... + a_n = 0\}$. This means that:

$$\frac{\exp(a_i)}{\sum_{j=1}^{n} \exp(a_j)} = \frac{\exp(a'_i)}{\sum_{j=1}^{n} \exp(a'_j)}$$

for all $i = 1, \ldots, n$, which implies $\exp(a_i) = c\exp(a'_i)$ for all i and hence $a_i = a'_i + c'$ for some constant c'.

As $0 = \sum_{i=1}^{n} a_i = \sum_{i=1}^{n} a'_i$ by nature of a_i and a'_i being in $\{(a_1, ..., a_n) | a_1 + ... + a_n = 0\}$ and as $a_i = a'_i + c'$ we can conclude $c' = 0$.

$$\begin{aligned}
\sum_{i=1}^{n} a_i &= \sum_{i=1}^{n} a'_i + c' \\
0 &= \sum_{i=1}^{n} a'_i + c' \\
0 &= nc' + \sum_{i=1}^{n} a'_i \\
0 &= nc'.
\end{aligned}$$

If $c' = 0$ then $a = a'$ and $\mathfrak{F}$ is injective.

We now need to show that H is injective. Suppose $H\beta = H\beta'$, then we get the following equations:

$$\begin{aligned}
d_1\beta'_1 &= d_1\beta_1 \\
-\frac{d_1}{n-1}\beta'_1 + d_2\beta'_2 &= -\frac{d_1}{n-1}\beta_1 + d_2\beta_2 \\
&\vdots \\
-\frac{d_1}{n-1}\beta'_1 - \frac{d_2}{n-2}\beta'_2 \ldots - \frac{d_{n-2}}{2}\beta'_{n-2} + d_{n-1}\beta'_{n-1} &= -\frac{d_1}{n-1}\beta_1 - \frac{d_2}{n-2}\beta_2 \ldots - \frac{d_{n-2}}{2}\beta_{n-2} + d_{n-1}\beta_{n-1} \\
-\frac{d_1}{n-1}\beta'_1 - \frac{d_2}{n-2}\beta'_2 \ldots - \frac{d_{n-2}}{2}\beta'_{n-2} - d_{n-1}\beta'_{n-1} &= -\frac{d_1}{n-1}\beta_1 - \frac{d_2}{n-2}\beta_2 \ldots - \frac{d_{n-2}}{2}\beta_{n-2} - d_{n-1}\beta_{n-1}.
\end{aligned}$$

As d_i is non-zero for all i, $\beta_1 = \beta'_1$. We now assume $\beta_i = \beta'_i$ for all $i < j$ and show that $\beta_j = \beta'_j$.

$$-\frac{d_1}{n-1}\beta'_1 - \frac{d_2}{n-2}\beta'_2 \ldots - \frac{d_{j-1}}{n-j+1}\beta'_{j-1} + d_j\beta'_j = -\frac{d_1}{n-1}\beta_1 - \frac{d_2}{n-2}\beta_2 \ldots - \frac{d_{j-1}}{n-j+1}\beta_{j-1} + d_j\beta'_j$$

which by supposition is equal to

$$-\frac{d_1}{n-1}\beta_1 - \frac{d_2}{n-2}\beta_2 \ldots - \frac{d_{j-1}}{n-j+1}\beta_{j-1} + d_j\beta_j$$

and as d_j is non-zero

$$\beta'_j = \beta_j.$$

Hence by induction $\beta = \beta'$, meaning H is injective. The composition of two injective functions is also injective so $\mathcal{F} = \mathfrak{F} \circ H$ is injective.

(Surjectivity)

We show that $\mathcal{F}$ is surjective by construction. First we construct a such that $a = \pi$. Let

$$a_i = \log\left(\frac{\pi_i}{\left(\prod_{i=1}^{n}\pi_i\right)^{1/n}}\right) \text{ for } 1 \leq i \leq n$$

Applying $\mathfrak{F}(a)_i$ gives

$$\pi_i = \frac{\pi_i}{\left(\prod_{i=1}^{n}\pi_i\right)^{1/n}} \Bigg/ \sum_{j=1}^{n} \frac{\pi_j}{\left(\prod_{i=1}^{n}\pi_i\right)^{1/n}} .$$

Pulling out the product from the denominator gives:

$$\mathfrak{F}(a)_i = \frac{\pi_i}{\sum_{j=1}^{n}\pi_j}.$$

As $\sum_{j=1}^{n}\pi_j = 1$ this gives π_i showing that $\mathfrak{F}$ is surjective. We now show that H is surjective onto $\{(a_1, ..., a_n)|a_1 + ... + a_n = 0\}$.

We wish to find β so that $a = H\beta$. The system of n linear equations $a = H\beta$, is in $n-1$ variables, the d_i are non-zero. This means that if we think of the first $n-1$ linear equations, Gaussian elimination gives β_i for $i = 1$ to $n-1$. As H maps from $\mathbb{R}^{n-1}$ to $\{(a_1, ..., a_n)|a_1 + ... + a_n = 0\}$ this is sufficient and H is surjective.

To show that H maps $\mathbb{R}^{n-1}$ to $\{(a_1, ..., a_n)|a_1 + ... + a_n = 0\}$ all we have to show is $a_n = -\sum_{i=1}^{n-1} a_i$

where $a_n = -d_{n-1}\beta_{n-1} - \sum_{j=1}^{n-2} \frac{d_j}{n-j}\beta_j$ and $a_i = d_i\beta_i - \sum_{j=1}^{i-1} \frac{d_j}{n-j}\beta_j$ for all $i < n$.

$$\begin{aligned}
-\sum_{i=1}^{n-1} a_i &= -\sum_{i=1}^{n-1} d_i\beta_i - \sum_{j=1}^{i-1} \frac{d_j}{n-j}\beta_j \\
&= -\sum_{i=1}^{n-1} d_i\beta_i - \sum_{j<n-i} \frac{d_i}{n-i}\beta_i \\
&= -\sum_{i=1}^{n-1} d_i\beta_i - \frac{(n-i-1)d_i}{n-i}\beta_i \\
&= -\sum_{i=1}^{n-1} \frac{d_i}{n-i}\beta_i \\
&= a_n
\end{aligned}$$

Since H and $\mathfrak{F}$ are surjective $\mathcal{F}$ is.

As $\mathcal{F}$ is both injective and surjective it is bijective. □

Composition parameters and the Farrow transformation

In this subsection we cover how to transform the base rates by the Farrow transformation. The base rates π_A, π_G, π_C, and π_T exist on the 4 dimensional simplex Δ_4 such that $\pi_A + \pi_G + \pi_C + \pi_T = 1$. The Farrow transformation provides a method to transform the space that generates a space where HMC can easily be run.

In this case:

$$H = \begin{bmatrix} 1 & 0 & 0 \\ -1/3 & \sqrt{8/9} & 0 \\ -1/3 & -\sqrt{2/9} & \sqrt{2/3} \\ -1/3 & -\sqrt{2/9} & -\sqrt{2/3} \end{bmatrix}$$

The columns sum to 0.

We now have the following relationships between α and β.

$$\begin{aligned}
\alpha_1 =& \beta_1 & \beta_1 =& \alpha_1 \\
\alpha_2 =& -\frac{\beta_1}{3} + \sqrt{\frac{8}{9}}\beta_2 & \beta_2 =& \frac{\alpha_2 + \frac{\alpha_1}{3}}{\sqrt{8/9}} \\
\alpha_3 =& -\frac{\beta_1}{3} - \sqrt{\frac{2}{9}}\beta_2 + \sqrt{\frac{2}{3}}\beta_3 & \beta_3 =& \frac{\alpha_3 + \frac{\alpha_1}{3} + \frac{\alpha_2 + \frac{\alpha_1}{3}}{2}}{\sqrt{2/3}} \\
\alpha_4 =& -\frac{\beta_1}{3} - \sqrt{\frac{2}{9}}\beta_2 - \sqrt{\frac{2}{3}}\beta_3 & =& -\frac{\alpha_4 + \frac{\alpha_1}{3} + \frac{\alpha_2 + \frac{\alpha_1}{3}}{2}}{\sqrt{2/3}}
\end{aligned}$$

Hence we can express π_A, π_G, π_C, and π_T in terms of β: Let:

$$\mathcal{S} = \exp(\beta_1) + \exp(-\frac{\beta_1}{3} + \sqrt{\frac{8}{9}}\beta_2) + \exp(-\frac{\beta_1}{3} - \sqrt{\frac{2}{9}}\beta_2 + \sqrt{\frac{2}{3}}\beta_3), \\ + \exp(-\frac{\beta_1}{3} - \sqrt{\frac{2}{9}}\beta_2 - \sqrt{\frac{2}{3}}\beta_3)$$

then:

$$\begin{aligned}
\pi_A =& \frac{\exp(\beta_1)}{\mathcal{S}} & \pi_G =& \frac{\exp(-\frac{\beta_1}{3} + \sqrt{\frac{8}{9}}\beta_2)}{\mathcal{S}} \\
\pi_C =& \frac{\exp(-\frac{\beta_1}{3} - \sqrt{\frac{2}{9}}\beta_2 + \sqrt{\frac{2}{3}}\beta_3)}{\mathcal{S}} & \pi_T =& \frac{\exp(-\frac{\beta_1}{3} - \sqrt{\frac{2}{9}}\beta_2 - \sqrt{\frac{2}{3}}\beta_3)}{\mathcal{S}}.
\end{aligned}$$

Hence we can compute the derivatives of π_A, π_G, π_C, and π_T with repect to β_1, β_2, and β_3.

$$\begin{aligned}
\frac{\partial \pi_A}{\partial \beta_1} &= \pi_A \left(1 - \pi_A + \frac{1}{3}(\pi_G + \pi_C + \pi_T)\right) & \frac{\partial \pi_A}{\partial \beta_2} &= -\sqrt{\frac{2}{9}}\pi_A (2\pi_G - \pi_C - \pi_T) \\
\frac{\partial \pi_A}{\partial \beta_3} &= -\sqrt{\frac{2}{3}}\pi_A (\pi_C - \pi_T) & & \\
\frac{\partial \pi_G}{\partial \beta_1} &= -\frac{\pi_G}{3} - \pi_G \left(\pi_A - \frac{\pi_G}{3} + \frac{\pi_C}{3} + \frac{\pi_T}{3}\right) & \frac{\partial \pi_G}{\partial \beta_2} &= \sqrt{\frac{8}{9}}\pi_G - \sqrt{\frac{2}{9}}\pi_G (2\pi_G - \pi_C - \pi_T) \\
\frac{\partial \pi_G}{\partial \beta_3} &= -\sqrt{\frac{2}{3}}\pi_G (\pi_C - \pi_T) & & \\
\frac{\partial \pi_C}{\partial \beta_1} &= -\frac{\pi_C}{3} - \pi_C \left(\pi_A - \frac{\pi_G}{3} + \frac{\pi_C}{3} + \frac{\pi_T}{3}\right) & \frac{\partial \pi_C}{\partial \beta_2} &= -\sqrt{\frac{2}{9}}\pi_C - \sqrt{\frac{2}{9}}\pi_A (2\pi_G - \pi_C - \pi_T) \\
\frac{\partial \pi_C}{\partial \beta_3} &= \sqrt{\frac{2}{3}}\pi_C - \sqrt{\frac{2}{3}}\pi_C (\pi_C - \pi_T) & & \\
\frac{\partial \pi_T}{\partial \beta_1} &= -\frac{\pi_T}{3} - \pi_T \left(\pi_A - \frac{\pi_G}{3} + \frac{\pi_C}{3} + \frac{\pi_T}{3}\right) & \frac{\partial \pi_T}{\partial \beta_2} &= -\sqrt{\frac{2}{9}}\pi_T - \sqrt{\frac{2}{9}}\pi_A (2\pi_G - \pi_C - \pi_T) \\
\frac{\partial \pi_T}{\partial \beta_3} &= -\sqrt{\frac{2}{3}}\pi_T - \sqrt{\frac{2}{3}}\pi_T (\pi_C - \pi_T). & &
\end{aligned}$$

In subsection 2.2.3 we see that for HKY85 model the transition matrix takes the following form:

Let $\pi_R = \pi_A + \pi_G$ $\quad \pi_Y = \pi_C + \pi_T$

$yr = \pi_Y / \pi_R$ $\quad ry = 1.0/yr$

$e_2 = \exp(-\beta t)$ $\quad e_3 = \exp(-(\pi_R \alpha + \pi_Y \beta)t)$

$e_4 = \exp(-(\pi_Y \alpha + \pi_R \beta)t)$

then:

$P(t) = \exp(tQ)$ has the form

$\cdot$	A	C	G	T
A	$\pi_A + \pi_A yr e_2 + \pi_G/\pi_R e_3$	$\pi_C(1-e_2)$	$\pi_G + \pi_G yr e_2 - \pi_G/\pi_R e3$	$\pi_T(1-e_2)$
C	$\pi_A(1-e_2)$	$\pi_C + \pi_C ry e_2 + \pi_T/\pi_Y e_4$	$\pi_G(1-e_2)$	$\pi_T + \pi_T rye_2 - \pi_T/\pi_Y e_4$.
G	$\pi_A + \pi_A yr e_2 - \pi_A/\pi_R e_3$	$\pi_C(1-e_2)$	$\pi_G + \pi_G yr e_2 + \pi_A/\pi_R e_3$	$\pi_T(1-e_2)$
T	$\pi_A(1-e_2)$	$\pi_C + \pi_C rye_2 - \pi_C/\pi_Y e_4$	$\pi_G(1-e_2)$	$\pi_T + \pi_T rye_2 + \pi_C/\pi_Y e_4$

This means we can compute the derivative of P with respect to β_i, and so we have everything needed for the derivative calculation in section 6.1.

$$\frac{\partial \pi_R}{\partial \beta_i} = \frac{\partial \pi_A}{\partial \beta_i} + \frac{\partial \pi_G}{\partial \beta_i} \qquad \frac{\partial \pi_Y}{\partial \beta_i} = \frac{\partial \pi_C}{\partial \beta_i} + \frac{\partial \pi_T}{\partial \beta_i}$$

$$\frac{\partial e_1}{\partial \beta_i} = 0 \qquad \frac{\partial e_2}{\partial \beta_i} = -(\frac{\partial \pi_R}{\partial \beta_i}\lambda + \frac{\partial \pi_Y}{\partial \beta_i})t\exp(-(\pi_R\lambda + \pi_Y)t)$$

$$\frac{\partial e_3}{\partial \beta_i} = -(\frac{\partial \pi_R}{\partial \beta_i} + \frac{\partial \pi_Y}{\partial \beta_i}\lambda)t\exp(-(\pi_R + \pi_Y\lambda)t)$$

We can also apply the chain rule to get the derivative of any function f with respect to β_j instead of π_i

$$\frac{\partial f}{\partial \beta_j} = \sum_j \frac{\partial f}{\partial \pi_i}\frac{\partial \pi_i}{\partial \beta_j}.$$

This reparameterisation affects the derivative, we now infer over β_1, β_2 and β_3. The choice of reparameterisation affects how the derivative with respect to β_1, β_2 and β_3 interacts with the parameters π_A, π_C, π_G and π_T. Since the Farrow transformation does not map to the boundary, the derivative avoids the boundary. This is a good feature for HMC as we do not wish to propose points on the boundary.

6.3 HMC on $\mathbb{R}_+$

The transition-transversion parameters and α are positive real parameters. We choose to conduct HMC on $\mathbb{R}_+$ by means of the log transformation, $\phi \in \mathbb{R}_+ \Rightarrow \log(\phi) \in \mathbb{R}$. This means that

$$\frac{\partial f}{\partial \log(\phi)} = \phi\frac{\partial f}{\partial \phi}.$$

6.4 Computing the Derivative with Respect to Site Rate Parameters

To compute the derivative with respect to site rate parameters we start with the same formulation of the likelihood as in section 6.1. We differentiate with respect to the site rate parameter α. Taking the derivative of the Likelihood as defined in section 2.5 with respect to α we get:

$$\begin{aligned}\frac{\partial}{\partial \alpha}\log\{\mathcal{L}(X|\theta)\} =& \frac{\partial}{\partial \alpha}\sum_{i=1}^{n}\log \mathbb{P}(\chi_i,|\theta) \\ =& \sum_{i=1}^{n}\frac{\frac{\partial}{\partial \alpha}\mathbb{P}(\chi_i,|\theta)}{\mathbb{P}(\chi_i,|\theta)}\end{aligned}$$

We now expand $\mathbb{P}(\chi_i, |\theta) = \sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\mathcal{L}_{v_0}(\chi_i|x,\theta k)\mathbb{P}(\hat{\chi}(v_0)=x)$:

$$\begin{aligned}\frac{\partial}{\partial\alpha}\mathbb{P}(\chi_i,|\theta) =& \sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\mathcal{L}_{v_0}(\chi_i|x,\theta k)\frac{\partial}{\partial\alpha}\mathbb{P}(\hat{\chi}(v_0)=x)\\ &+\mathbb{P}(\hat{\chi}(v_0)=x)\frac{\partial}{\partial\alpha}\mathcal{L}_{v_0}(\chi_i|x,\theta,k)\\ &+\sum_{k\in\gamma}\frac{\partial\mathbb{P}(k)}{\partial\alpha}\sum_{x\in\mathcal{A}}\mathcal{L}_{v_0}(\chi_i|x,\theta k)\mathbb{P}(\hat{\chi}(v_0)=x). \qquad (1)\end{aligned}$$

We know that $\frac{\partial}{\partial\alpha}\mathbb{P}(\hat{\chi}(v_0)=x)=0$ and hence that the first term can be removed:

$$\begin{aligned}&\frac{\partial}{\partial\alpha}\mathbb{P}(\chi_i,|\theta)\\ &\quad=\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\mathbb{P}(\hat{\chi}(v_0)=x)\frac{\partial}{\partial\alpha}\mathcal{L}_{v_0}(\chi_i|x,\theta,k)+\sum_{k\in\gamma}\frac{\partial\mathbb{P}(k)}{\partial\alpha}\sum_{x\in\mathcal{A}}\mathcal{L}_{v_0}(\chi_i|x,\theta k)\mathbb{P}(\hat{\chi}(v_0)=x)\end{aligned}$$

As with the substitution model parameter derivative calculation we expand and differentiate $\mathcal{L}_v(\chi_i|x,\theta,k)$.

$$\begin{aligned}\frac{\partial}{\partial\alpha}\mathcal{L}_v(\chi_i|x,\theta,k) =& \sum_{u\prec v}\left[\prod_{u'\prec v}^{u'\neq u}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{t(u',v)\}\mathcal{L}_i^{u'}(\chi_i|y,\theta,k)\right]\right]\\ &\times\sum_{y\in\mathcal{A}}\frac{\partial}{\partial\alpha}\left(P_{x,y;k}\{t(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)\right)\end{aligned}$$

We expand $\frac{\partial}{\partial\alpha}\left(P_{x,y;k}\{t(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)\right)$ by the product rule:

$$\begin{aligned}&\frac{\partial}{\partial\alpha}\left(P_{x,y;k}\{t(u,v)\}\mathcal{L}_u(\chi_i|y,\theta,k)\right)\\ &\quad=\frac{\partial}{\partial\alpha}P_{x,y;k}\{t(u,v)\}\times\mathcal{L}_u(\chi_i|y,\theta,k)+P_{x,y;k}\{t(u,v)\}\frac{\partial}{\partial\alpha}\mathcal{L}_u(\chi_i|y,\theta,k)\end{aligned}$$

With substitution model parameters we noticed that the nature of Q meant that we could not trivially compute $\frac{\partial}{\partial\phi}P_{x,y;k}\{t(u,v)\}$ whereas with the site rate parameter this is no longer a problem and so the derivative behaves like an edge length.

$$P_{x,y;k}\{t(u,v)\}=\exp\{kt(u,v)Q\}\Rightarrow\frac{\partial}{\partial\alpha}P_{x,y;k}\{t(u,v)\}=\frac{\partial k}{\partial\alpha}t(u,v)P_{x,y;k}\{t(u,v)\}Q. \qquad (2)$$

The above analysis shows that two derivatives must be calculated, $\frac{\partial\mathbb{P}(k)}{\partial\alpha}$ from (1) and $\frac{\partial k}{\partial\alpha}$ from (2). In section 2.4 we have seen that k is fixed at various levels and α generates a fixed gamma

distribution with shape and rate α. We use this information to compute $\frac{\partial k}{\partial \alpha}$ and $\frac{\partial \mathbb{P}(k)}{\partial \alpha}$ from it.

We first show how to compute $\frac{\partial \mathbb{P}(k)}{\partial \alpha}$ if we have $\frac{\partial k}{\partial \alpha}$ available.

$$\mathbb{P}(k) = \frac{\alpha^\alpha k^{\alpha-1} \exp(-\alpha k)}{\Gamma(\alpha)}$$

We can take the derivative:

$$\frac{\partial \mathbb{P}(k)}{\partial \alpha} = \frac{\partial}{\partial \alpha} \frac{\alpha^\alpha k^{\alpha-1} \exp(-\alpha k)}{\Gamma(\alpha)}$$

We now use the chain rule:

$$\frac{\mathbb{P}(k)}{\partial \alpha} = \frac{\partial \mathbb{P}(k)}{\partial k} \frac{\partial k}{\partial \alpha}.$$

$$\begin{aligned}\frac{\partial \mathbb{P}(k)}{\partial \alpha} &= \frac{\partial k}{\partial \alpha} \frac{\partial}{\partial k} \frac{\alpha^\alpha k^{\alpha-1} \exp(-k\alpha)}{\Gamma(\alpha)} \\ &= \frac{\partial k}{\partial \alpha} \frac{\alpha^\alpha (\alpha-1) k^{\alpha-2} \exp(-\alpha k) + \alpha^\alpha k^{\alpha-1} \exp(-\alpha k)}{\Gamma(\alpha)}.\end{aligned}$$

This means that once we have $\frac{\partial k}{\partial \alpha}$ we can compute $\frac{\partial P(k)}{\partial \alpha}$.

We now focus on computing $\frac{\partial k}{\partial \alpha}$. We first explain the difficulty in calculating $\frac{\partial k}{\partial \alpha}$ and then explain our solution. k is the solution to the cumulative density function at a particular point c.

$$\int_0^k \frac{\alpha^\alpha x^{\alpha-1} \exp(-\alpha x)}{\Gamma(a)} dx = c.$$

We apply the fundamental theorem of calculus to get $F(k) - F(0) = c$ for some F and that $k = F^{-1}(c + F(0))$. Standard theory tells us that F is

$$F(x) = \frac{\int_0^{\alpha x} t^{\alpha-1} e^{-t} dt}{\Gamma(\alpha)}$$

so that $F(0) = 0$. The integral is continuously differentiable since $t^{\alpha-1}e^{-t}$ is. We can apply the I.V.T. to find that the derivative of $\frac{\partial F^{-1}(c)}{\partial \alpha}$ in terms of the derivative of F with respect to α. This requires a numerical solution, hence if we want to incorporate gamma rate heterogeneity we need to employ a numerical approximation.

We now describe the numerical approximation. In Phylogenetics there are characteristic choices

for c which depend on how many partitions of the interval $(0, 1)$ are required, these are found by approximating the value across the interval by the value in the middle. It is common to partition the interval into powers of 2 and rarely are more than 8 partitions put into practice. We used a standard approximation,

$$\frac{\partial k(\alpha)}{\partial \alpha} \approx \frac{k(\alpha + h) - k(\alpha - h)}{2h}$$

for small h. As stated in subsection 2.4 we chose $h = 0.001$. We decided to take values between 0 and 6 as the derivative became fairly consistent, changing in the 4th significant figure, we approximated greater values of α by the final value at 6.

We decided not to go with a linear approximation to compute the derivative with respect to α for values greater than 6. If we were to go with a decreasing scheme in the tail rather than a flat approximation the gradient would be smaller in the tail, resulting in proposals staying in this region of low density as the gradient along with the momenta dictate the change in the proposal. Furthermore it is in a region where the CortHMC should rarely visit due to the Metropolis-Hastings acceptance step rejecting most of the moves to such a region. It can be seen when the Alignment is not very informative causing α to reach values far above something realistic. At this point α is effecting the gamma-distribution in such a way that all columns of the alignment have approximately the same evolutionary rate.

Part III

Results

Chapter 7

Results

We wish to compare and contrast COrtHMC with existing Metropolis within Gibbs techniques (see chapter 3). We implemented COrtHMC in our project COrtHMC dependent on PhyloCore. All runs are conducted either in COrtHMC or in PhyloCore for comparison, we do this as to maintain a comparitive level of optimisation and to run code using the same language. We split every run into lots of short runs stitched together, for ease of storage and to manually perform garbage collection. In this section we use several artificial alignments over which we design an optimal version of COrtHMC. We then compare COrtHMC with Metropolis within Gibbs for our artificial alignments and an alignment from MrBayes (Huelsenbeck and Ronquist, 2001). We first specify the alignments and priors we use. We then move on to demonstrate that COrtHMC does in fact target the correct distribution as a means to inspire confidence in our technique. COrtHMC requires specification of lots of tuning parameters and various crossing methods are available to the practitioner. To simplify things for the reader we shall split the design of COrtHMC into two parts. The first parts deals with the parameters required for J, the "HMC" component of COrtHMC, see section 5.2. These are the choice of integrator and choice of momenta kernel. Since we use the leapfrog integrator the tuning parameters involved in the choice of integrator are the step-size and number of steps. The second part covers the different choice of crossing kernels.

7.1 Data, Priors and Testing

We consider COrtHMC on several alignments constructed from synthetic trees with 5, 16, and 32 taxa, constructed using the package "ape" (Paradis and Schliep, 2018). We also specify priors for the edges, both internal and pendant, substitution model parameters and site rate parameters. Where possible, priors commonly adopted in the literature are used.

7.1.1 Alignments

We denote the alignments generated with n sites for 5, 16 and 32 taxa as $\mathcal{A}_5^n, \mathcal{A}_{16}^n$, and $\mathcal{A}_{32}^n$ respectively. We also use the primate alignment from MrBayes, denoted $\mathcal{A}_p$, as this alignment is not generated from a tree we include the alignment in Appendix E. Each alignment is fixed throughout the testing process. The alignments were all generated using a HKY85 substitution model with base rates $\pi_A = 0.4$, $\pi_C = 0.1$, $\pi_G = 0.2$ and $\pi_T = 0.3$ and a transition transversion ratio of 1.5 and a discrete gamma site rate heterogeneity with $\alpha = 0.5$. We assume a HKY85 model and a discrete gamma model throughout this section.

We used 3 different trees to construct the alignments for the 5, 16 and 32 taxa trees. All alignments $\mathcal{A}_5$ were constructed from the random tree

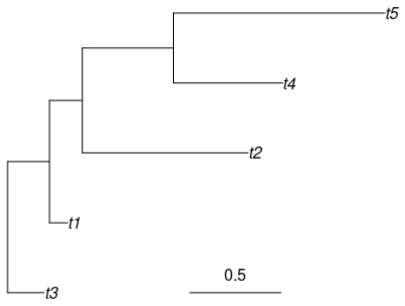


Figure 7.1: The alignment generating tree for 5 taxa.

We generated two alignments from this tree denoted $\mathcal{A}_5^{100}$ and $\mathcal{A}_5^{1000}$. For the $\mathcal{A}_{16}$ alignments we used the random tree

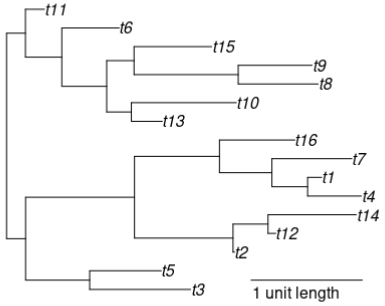


Figure 7.2: The alignment generating tree for 16 taxa.

Finally, all $\mathcal{A}_{32}$ alignments were constructed from the random tree

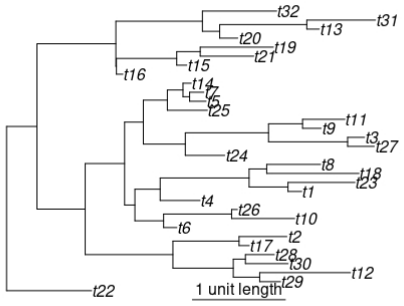


Figure 7.3: The alignment generating tree for 32 taxa.

7.1.2 Priors

We adopted an independent prior specification across internal and pendant edges. For the former we specified $\Gamma(0.01, 10)$ distributions and for the latter, $\Gamma(0.1, 1)$ distributions.

We chose this specification because we observed very different behaviour on internal edges when compared to pendant edges. COrtHMC is dependant on the derivative of the posterior. The derivative of the prior influences this derivative according to the product rule. For the Gamma

distribution the derivative is

$$\frac{\partial \Gamma_{\alpha,\beta}(\ell)}{\partial l} = \frac{\alpha - 1}{\ell} - \frac{1}{\beta}.$$

Looking back at the leapfrog integrator, (see section 4.3.1) we can see that the momenta is transformed by subtracting a small step along the derivative of the Hamiltonian with respect to the parameters and then the parameters add on a small step of the derivative of the Hamiltonian with respect to the momenta. For a normal distribution this derivative is $\Sigma^{-1}p$ where Σ is the covariance matrix. Taking $\Sigma = I$ allows us to easily understand how problems occur. For small values of ℓ

$$\frac{\partial \Gamma_{\alpha,\beta}(\ell)}{\partial l} \approx \frac{\alpha - 1}{\ell}$$

As α gets smaller $\frac{\partial \Gamma_{\alpha,\beta}(\ell)}{\partial l}$ decreases meaning that the change in momenta p increases. The choice of $\alpha = 1$ is convenient because this term disappears but still results in $\frac{\partial \Gamma_{\alpha,\beta}(\ell)}{\partial l} = -\frac{1}{\beta} < 0$. This means that the prior is informing us that we should be moving away from any codimensional boundary. By choosing $\alpha > 1$ the prior now favours crossing codimensional boundaries for small edge lengths. As we wish to explore tree space by crossing boundaries we opt to use a prior that says such movement is possible rather than one that says it ought not occur. This is also consistent with the continuous nature of the likelihood. The likelihood states that two points infinitesimally close to each other but either side of the boundary are effectively the same and so evolution from one to the other should not be much harder than large jumps within an orthant. Such a poorly designed prior is demonstrated in Fig 7.4 where we can see changes in topology by changes in colour.

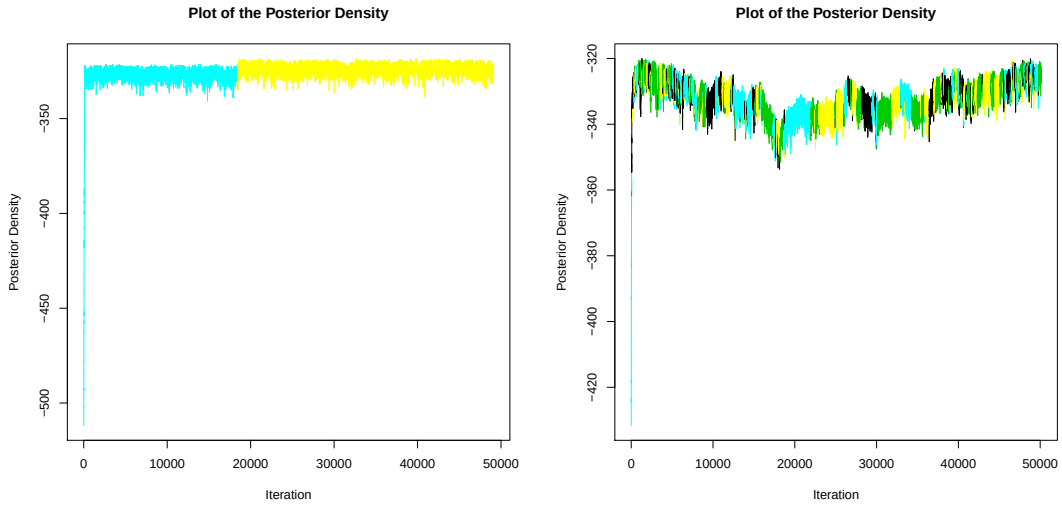


Figure 7.4: Trace plots of the log-posterior showing different numbers of topological crossings when using a Dirichlet distribution with $\alpha = 0.01$. The first figure changes topology once, the second with $\alpha = 1$ often changes topology as can be seen by the various different colours. (We used $\mathcal{A}_5^{1000}$ to generate these trace plots.)

We opt to use a Dirichlet prior for the base rates with all the concentration parameters fixed at 0.25. A Dirichlet prior is a prior on the simplex given by a Dirichlet distribution. The choice of 0.25 makes sense as we have no reason to presume that any base occurs more often than any other base. We used an inverse gamma prior for the transition transversion ratio with $\alpha = \beta = 2$ to have a mean of 2. While we decided to use this prior for the TT ratio we find that we are actually giving a lot of information to COrtHMC, for example that the most likely value for the TT ratio is $\frac{2}{3}$. To get around this abundance of information we provide an alternative prior.
Let $c = \frac{1}{2(b-a)} / (2\frac{1}{\sqrt{2\pi}}) + 1/2 + \frac{1}{2(b-a)} / (2\frac{(a+1)^a a^{-a-1} \exp(-\frac{a+1}{a})}{\Gamma(a)})$ then

$$\mathbb{P}(x|a) = \begin{cases} c\frac{1}{2(b-a)a^{-a-1}\exp(-\frac{a+1}{a})} x^{-a-1} \exp(-\frac{a+1}{x}) & \text{if } x < a \\ c\frac{1}{2(b-a)} \qquad \text{if } a < x < b \\ c\frac{1}{2(b-a)} \exp\left(-(x-b)^2\right) \quad \text{otherwise} \end{cases}.$$

This prior is constructed piecewise by first using an inverted gamma distribution, the a uniform distribution and finally a normal distribution, such that they are all smoothly connected. The middle uniform distribution means that the user can abdicate all knowledge of the transition transversion ratio for a region, (a, b), while allowing the ratio to lie anyway on $\mathbb{R}_+$. This prior is also similar to the inverse gamma prior in that high values and low values, corresponding to all transitions or all transversions, are increasingly unlikely. We chose not to use this prior as it is

complicated and while the mean is at $(a + b)/2$ it is an unknown distribution and therefore we avoid it.

Finally, we used an exponential $Exp(0.5)$ prior for the site rate parameter α.

We note that unlike standard Metropolis within Gibbs, the statistical efficiency of COrtHMC is likely to be particular sensitive to the choice of prior. When the shape parameter in the gamma distribution is greater than or equal to the scale parameter the probability density tends to 0 as the random variable does. This dominates the posterior causing the posterior density to also tend to as the random variable does. The positive nature of the gradient near 0 values also is inherited by the posterior from the prior in this case. The resulting effect is as the proposal gets closer to crossing a codimensional boundary the prior moves the proposal away from the boundary. This means that to overcome this effect and propose a point in a new topology large momenta is required which will often compound the error in the integration meaning that new topologies are proposed rarely, limiting exploration and convergence.

7.1.3 Testing COrtHMC

To test COrtHMC we ran it on a known target distribution. We targeted the distribution comprising of the priors outlined in subsection 7.1.2. To do this we turned off the likelihood so that the posterior is completely dictated by the prior. Below we compare the results of COrtHMC, in blue, with the theoretical result depicted, in red. We do this by looking at the three graphs in Fig: 7.5. The first graph is diagnostic and depicts the trace plot, which gives an idea of how the parameter mixes and whether it gets stuck at particular values. The second plot plots the autocorrelation against lag which shows how correlated one sample is to the next, the faster the autocorrelation function (ACF) drops to 0, the less correlated the samples are and the more effective samples are produced, a number related to the number of independent samples. The final plot in Fig: 7.5 is the most important in demonstrating the correctness of COrtHMC. In each plot the theoretical density plot for the prior is depicted in red and the estimated density is drawn on it in blue. We can see that the prior for edges here is our standard prior for internal edges $\Gamma(10, 0.01)$ and the prior for piG is our standard Dirichlet prior. When these align we know that COrtHMC correctly samples from the prior. Here we provide the three plots for a couple of different parameters as all parameters produced similar results. The ACF plots for the substitution model parameters demonstrate that the samples are more correlated from one iteration to the next. The ACF here is greater than that expected in MCMC due to the correlated nature of HMC but also could be due to a lack of sufficient tuning. The density

plots demonstrate that the algorithm correctly targets the posterior as the inferred density (blue) agrees with the true density (red).

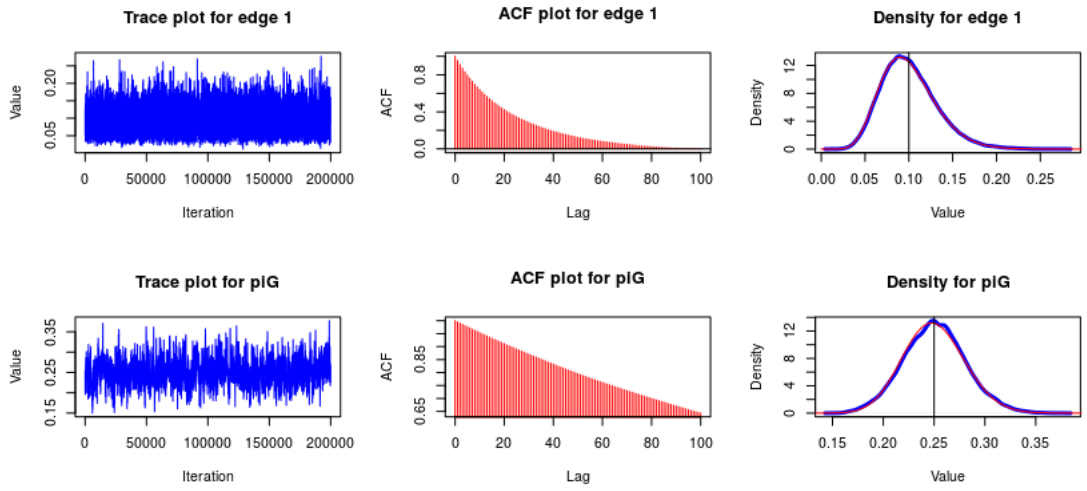


Figure 7.5: Plots for each parameter. Left: Trace plot depicted in blue, Middle: Autocorrelation in red, Right: Density in blue and red.

7.1.4 Prior Sensitivity

In this section we compare the effect of different priors on mixing performance. We consider the prior specification discussed in section 7.1.2 and an improper prior that is globally uniform. We find that priors can alter the mixing of COrtHMC as can be seen in Figure 7.6 and can override the information in the likelihood as in Figure 7.7.

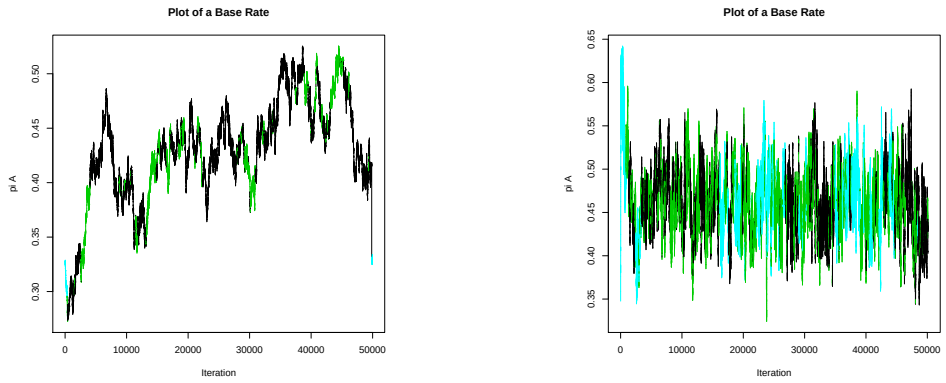


Figure 7.6: Informative vs Improper Priors. Here we see the different mixing of π_A using $\mathcal{A}_5^{100}$. The informative prior is on the left and exhibits poorer mixing than the improper prior on the right, both explore multiple topologies but the graph of the informative prior spends more time in the correct topology, as can be seen by the high percentage of black. The colours denote the topological distance from the tree used to generate $\mathcal{A}_5^{100}$, Fig: 7.1.

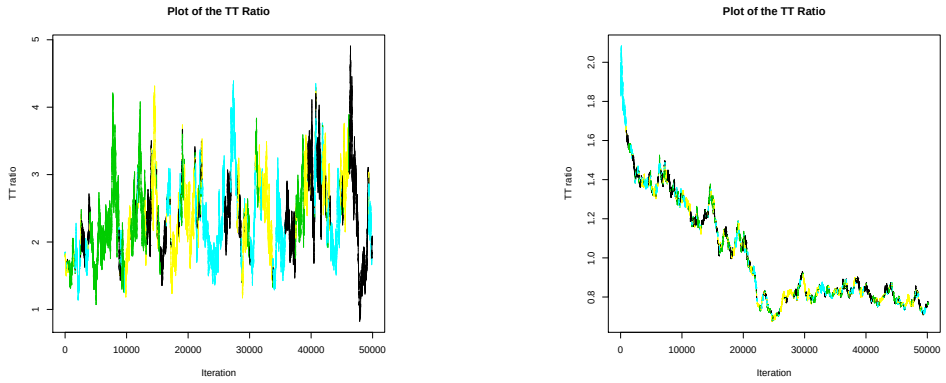


Figure 7.7: Informative vs Improper Priors. The improper prior is depicted on the left anad informative on the right. We can see that an informative prior acts to return the transition-transvertion ratio to a more confined region whereas and improper prior can cause extreme exploration around less informative areas. The informative prior effectively explores the prior for the gamma rate heterogeneity without exploring region about $\alpha = 0.5$ whereas the improper prior provides better exploration.

As two different posteriors are being compared it makes no sense to talk about comparative effective sample sizes. As the alignment length increased the importance of the prior decreased as can be seen in column sensitivity.

7.1.5 Column Sensitivity

The alignment length is not user defined but has an effect on the posterior via the likelihood. As the alignment length is the amount of data available, increasing the length increases how informative the likelihood is. This affects some parameters to a greater degree than others in particular the site rate heterogeneity. For smaller alignments the site rate heterogeneity parameter α is unbounded as there is not enough information available to specify how the site rate changes over sites. To see this consider the extreme example with only 1 site, in such a situation α could be anything. Referring back to section 2.4 this is equivalent to site rate homogeneity as the gamma distribution becomes more diffuse.

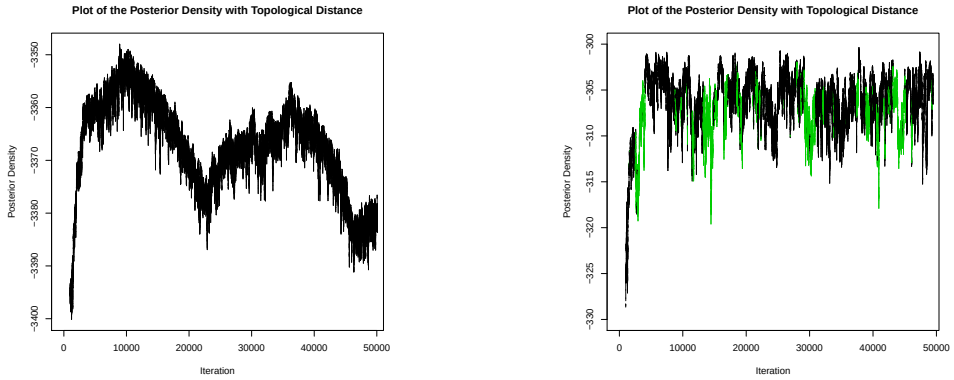


Figure 7.8: Two plots of the log-posterior for differing alignment lengths. On the left plot was generated from $\mathcal{A}_5^{1000}$ and the right the alignment has length $\mathcal{A}_5^{100}$. The more informative alignment focusses the chain to within a single orthant and provides a more informed path.

In Figure 7.8 there are two things to notice. First the alignment of length 1000 has a posterior varying by approximately ± 20 whereas in an alignment of length 100 the posterior varies by approximately ± 10 but as a percentage of the posterior it is 0.6% and 3%. The second thing to notice is that the chain does not leave the correct topology (depicted in black) when the alignment is length 1000 but does for the shorter alignment.

7.2 Specifying the HMC component: J from section 5.2

7.2.1 Inverted vs Standard Leapfrog integration

The first thing we want to compare is the inverted leapfrog integrator in section 4.3.1 to the standard leapfrog integrator. We found that the inverted leapfrog integrator provided significant speed up when we ran COrtHMC over more complicated trees. Although we expected a factor of

(*5 taxa*)

	Standard			*Inverted*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	570	8.9	540	2000	7.6	740
ESS / s	0.11	0.0017	0.10	0.37	0.0014	0.13

(*16 taxa*)

	Standard			*Inverted*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	370	58	102	730	120	170
ESS / s	0.012	0.0019	0.000042	0.019	0.0031	0.0044

Figure 7.9: Comparison of the effective sample size for Inverted vs Standard leapfrog integrators.

2 increase in CPU time, some of the time saved in not needing to compute as many derivatives was spent updating the parameters in storage. For less complicated trees, when the likelihood calculation has a less expensive time cost than inverting the momenta kernel, the inverted leapfrog integrator can be detrimental, for the simplest 5 taxa tree we found that the time taken for the inverted compared to the standard leapfrog was comparable. As most trees will have significantly more than 5 taxa, increasing the likelihood time cost above that of the momenta kernel, we would recommend always using the inverted leapfrog integrator, but do use the standard leapfrog integrator for testing, we decided to do so as to maintain a consistent base for our testing only varying one part at a time. We can see the benefit as the inverted leapfrog integrator on 16 taxa outperformed the leapfrog integrator we saw a $3:4$ ratio in the time taken or a speed up of roughly 25%. For most trees it provides effective speed up. In Figure 7.9 we see that both the leapfrog and inverted leapfrog produce an approximately equivalent effective sample size. We conducted the experiments with 16 and 32 taxa and received similar results to that of the 5 taxa tree, see appendix E. For ease of reading we only reproduce the results of the 5 taxa tree when this is the case but the data is available if required.

7.2.2 Step Number Sensitivity

We tested several different step numbers, l in section 4.3.1. As can be expected the time cost increased linearly as we increased the number of steps. Due to this we stopped testing after $l = 8$ as it became apparent that the cost was not being outweighed by the gain in ESS.

(*5 taxa*)

	One Step			*Two Steps*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	570	8.9	540	240	6.1	19
ESS / s	0.11	0.0017	0.10	0.026	0.00066	0.0020

(*5 taxa*)

	Three Steps			*Four Steps*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	120	15	660	760	7.3	130
ESS / s	0.090	0.0011	0.050	0.043	0.00041	0.071

(*5 taxa*)

	Eight Steps		
	Edges	*Parameters*	*Posterior*
ESS	100	11	900
ESS / s	0.033	0.00033	0.025

Figure 7.10: Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator.

The above tables show that the one step MALA process produces the best ESS / s followed either by the 3 step process for a 5 taxa tree. We found for the 16 and 32 taxa case that 3 steps produced the best results. We did not run 8 steps for the 32 taxa case as this would have taken a significantly longer run time when it was clear from the 16 taxa case that 8 steps would not be the best performing example. As 3 steps performed comparatively to 1 in the 5 taxa case we opt to try both for our final analysis.

7.2.3 Step Size Sensitivity

While we found that there were many different ways of generating ϵ. The best approaches were had with a manual grid approach similar to that suggested in Neal et al. (2011). The problem with this approach is that it required a lot of user input as the granularity of the grid to

(*5 taxa*)

	Fixed Blackbox			*Adaptive Blackbox*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	960	15	58	790	11	74
ESS / s	0.30	0.0047	0.018	0.17	0.0024	0.016

Figure 7.11: Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator.

search over differs depending on the number of taxa in the tree. We therefore constructed two blackbox methods based on the heuristic algorithm in Hoffman and Gelman (2014) as described in subsection 4.3.2. The two blackbox methods differed in that one enabled an adaptive stepsize. The two blackbox methods use the same method to choose an appropriate ϵ but one uses an adaptive scheme to alter ϵ throughout the run to slightly improve the results. To construct the adaptive scheme we make sure that we satisfy the simultaneous uniform ergodicity condition and diminishing adaptation condition found in Roberts and Rosenthal (2007). To construct the blackbox scheme we use the method found in Hoffman and Gelman (2014). We can see that both blackbox methods produce results that work.

We also tested how stepsize changes with the number of taxa by simulating some data for a random 20 taxa tree, Figure 7.12. We then removed data from the alignment associated with each pendant as we pruned the tree. We then used the specifically constructed tree in Figure 7.13 as the initial tree around which we found the optimum stepsize. We choose this tree because we could prune off pendant edges while maintaining a similar structure, and all edge lengths are the same which removes the impact of different edge lengths different topological importance in the pendant edges. We found that as the number of taxa increases the stepsize has to decrease which makes sense as the logposterior decreases.

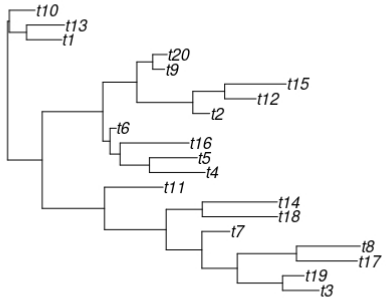


Figure 7.12: Here is the tree used to generate the alignment we used to test the effect of taxa on stepsize.

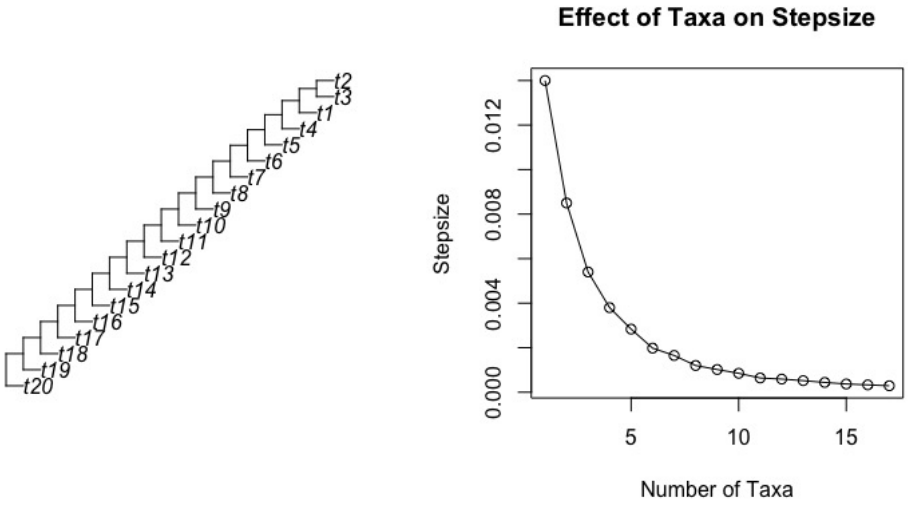


Figure 7.13: We see the tree used for the experiment and the relationship between stepsize and the number of edges.

While more taxa results ina a lower optimum stepsize as seen in Fig: 7.13 when we compare the acceptance rate to the logposterior we find a linear relationship, Fig: 7.14, the acceptance rate is inversely related to the stepsize and the and the logposterior is inversely related to the number of taxa.

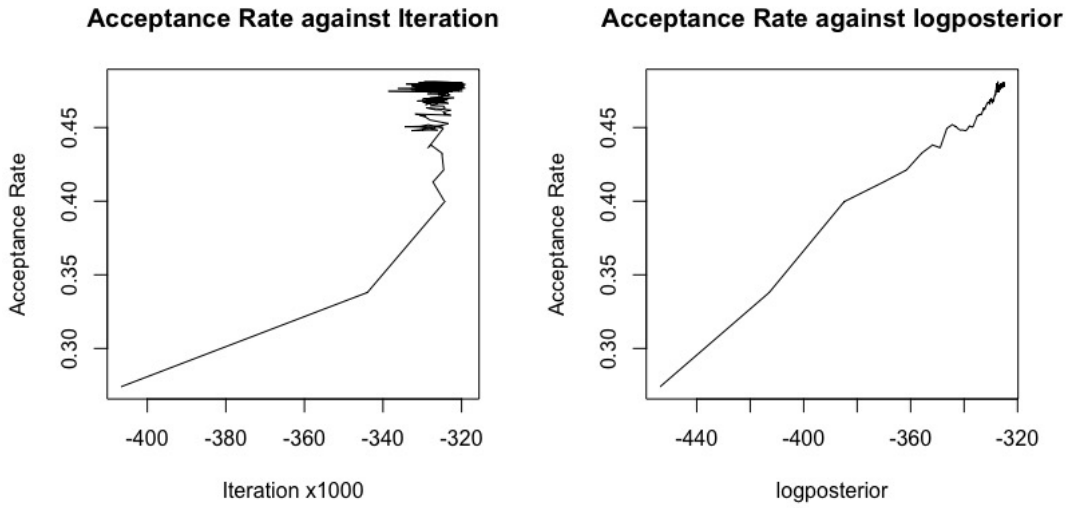


Figure 7.14: Here we see that acceptance rate has a linear relationship with the log-posterior.

7.2.4 Different Choices of Kernel for the Momenta

In Girolami et al. (2011) they adapt HMC for a position dependent momenta kernel, we produced a similar scheme that loses some accuracy in the integration in order to avoid the computational cost in computer the second order derivatives. As with Girolami et al. (2011) and Betancourt (2013) we tried using the Fisher information matrix, G. We encountered problems with this approach due to numerically errors during the run resulting in eigenvalues too close to 0. To counteract the issue we decided upon trying two different forms of the Fisher information matrix, the diagonal Fisher and the stabilised Fisher

$$\begin{aligned} &\textit{Diagonal} \quad \{G_d\}_{ij} = \delta_{ij} G_{ij} \\ &\textit{Stabilised} \quad \{G_s\}_{ij} = \{G\}_{ij} + \{G_d\}_{ij}. \end{aligned}$$

We then compared both of these with a standard fixed normal kernel.

(*5 taxa*)

	Projecting			*Reflecting*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	1100	152	1200	1000	4.4	100
ESS / s	0.22	0.032	0.25	0.24	0.0010	0.024

Figure 7.16: Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator.

(*5 taxa*)

	Stabilised Fisher			*Diagonal Fisher*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	1200	280	1300	1100	80	720
ESS / s	0.22	0.054	0.24	0.22	0.017	0.15

Figure 7.15: Comparison of the effective sample size for various different numbers of steps in the leapfrog integrator.

Both choices, provide significant improvements to the ESS / s when compared with the standard approach. This is particularly true for the worst performing parameters which improve by a factor of approximately 30 for the stabilised Fisher or 10 for the diagonal Fisher.

7.3 Specifying the Crossing component: C from section 5.2

7.3.1 Reflecting vs Projecting

We found that reflecting and projecting across a boundary, as described in subsection 5.7.1, only provided marginal differences in ESS/s. We found that projecting produced slighlty better ESS and ESS/s. This is counter-intuitive as the likelihood is differentiable across codimension 1 boundaries, this means we might expect the reflecting boundary to produce better results as it is linear estimate of the derivative at the boundary Dinh et al. (2017). The projecting boundary is simpler so we expect it to be quicker, the improvement in results could be due to the dependent nature of substitution model parameters and tree parameters.

(*5 taxa*)

	Fixed Crossing			*Ratio Crossing*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	1100	152	1200	2900	25	2200
ESS / s	0.22	0.032	0.25	0.63	0.0053	0.48

Figure 7.18: Comparison between chosing via ratio of the posterior densities vs a fixed probability.

7.3.2 3 orthants vs 2 orthants

We found that there was little practical benefit to using all 3 orthants when compared with 2 orthants. We in fact found that 3 orthants performed slightly poorer than 2 orthants.

(*5 taxa*)

	2 *Orthants*			3 *Orthants*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	1100	152	1200	360	55	540
ESS / s	0.22	0.032	0.25	0.081	0.012	0.12

Figure 7.17: Comparison of the effective sample size for entering 3 orthants as opposed to 2.

7.3.3 Ratio vs Fixed Transitions

When choosing a crossing kernel we can choose to either force entry into a new orthant, or have some chance of remaining in the current orthant. We found that there was little practical benefit to using all 3 orthants as this artificially forces the markov chain back on itself, slightly slowing exploration when compared with 2 orthants. In Fig: 7.17 we see that 3 orthants performed slightly poorer than 2 orthants.

7.3.4 Continuing the integration

Parachuting resets the momenta after changing topology. This is to enable faster convergence. We found that parachuting does not improve the ESS / s.

(*5 taxa*)

	Standard			*Continuing and Parachuting*		
	Edges	*Parameters*	*Posterior*	*Edges*	*Parameters*	*Posterior*
ESS	1100	152	1200	630	0.84	2.3
ESS / s	0.22	0.032	0.25	0.13	0.00017	0.00048

Figure 7.19: Comparison of the effective sample size for when we parachute the momenta as described in section 5.7.3.

7.4 Comparison with MCMC

Here we compare how COrtHMC runs against standard Metropolis-within-Gibbs. We compare the two using the alignment from MrBayes, $\mathcal{A}_p$. We do this by conducting several runs each pairwise, making sure their maxdiff is < 0.1. We can see in Fig: 7.20 that both COrtHMC and Metropolis-within-Gibbs produce the same tree but with a different root, one separating Tarsius syrichta subclade, the other separating the Homo sapiens subclade.

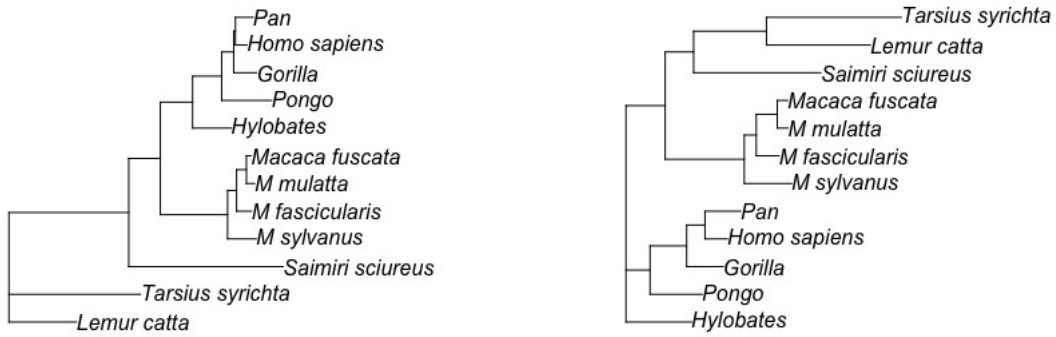


Figure 7.20: Comparison of the tree generated by MCMC (left) and the tree generated by COrtHMC (right).

We ran this comparison several times and consistently found that Metropolis-within-Gibbs

outperformed COrtHMC. In Fig: 7.21 and Fig: 7.22 we see that Metropolis-within-Gibbs always has the substitution model parameters converge faster than edges whereas, COrtHMC varies.

(*COrtHMC*)

	Edges	*Parameters*	*Posterior*	*Intrinsic ESS*
ESS	12	9.7	11	8.8
ESS / s	0.0021	0.0017	0.0018	0.0015

(*Metropolis-within-Gibbs*)

	Edges	*Parameters*	*Posterior*	*IntrinsicESS*
ESS	44	880	680	1200
ESS / s	0.38	7.7	5.9	13

Figure 7.21: Here we have a relatively short chain and the Metropolis-within-Gibbs outperforms COrtHMC. This provides a perfect example as to why Intrinsic ESS is a more reliable measure of ESS than taking the minimum ESS over edges. The minimum gives an ESS of 44 samples, several edges had an ESS of over 2000, neither really provide you with information of how the tree changes as they only describe a local region of the tree, whereas Intrinsic does.

(*COrtHMC*)

	Edges	*Parameters*	*Posterior*
ESS	540	1200	1700
ESS / s	0.014	0.031	0.044

(*Metropolis-within-Gibbs*)

	Edges	*Parameters*	*Posterior*
ESS	53000	66000	1200000
ESS / s	2.9	3.6	6.5

Figure 7.22: The added complexity of HMC and the autocorrelation significantly slows down the process and reduces the ESS.

7.5 Comparison with ppHMC

We used the primate alignment $\mathcal{A}_p$ to compare ppHMC's performance with that of COrtHMC. We found that both behaved similarly because the behaviour of HMC approaches are dominated by the behaviour inside and orthant.

(*ppHMC*)

	Edges	*Parameters*	*Posterior*
ESS	1.53	0.96	1.65
ESS / s	$6.85e-13$	$4.28e-13$	$7.37e-13$

(*COrtHMC*)

	Edges	*Parameters*	*Posterior*
ESS	1.32	1.38	1.57
ESS / s	$5.91e-13$	$6.15e-13$	$7.02e-13$

Figure 7.23: ppHMC behaves as expected just like most other COrtHMC crossing kernels.

7.6 Parallelising COrtHMC and Improving Efficiency

There are two key ways of improving the efficiency of COrtHMC. In sections 5.5,6.1 and 6.4 we have seen that computing the derivative is trivially parallelisable. This means that we can parallelise the computation of the derivative. Sections 5.5,6.1 and 6.4 also demonstrate that computing the derivative uses a lot of shared information due to the recursive nature of the algorithm. We also implemented both approaches, sharing as much information as possible within the recursion or evaluating the derivatives in parallel. We were unable to do both as evaluating the derivative would alter some of the shared information in the recursion. We quickly noticed by use of Netbeans's (Apache Software Foundation Oracle Corporation and Roman Staněk, 2021) internal profiler that implementing in parallel produced exactly the same results but ran slower. It is for this reason we do not go over how exactly to run COrtHMC in parallel. We include a link to our code where one can see how we efficiently stored the information in the likelihood in one object, called the "StoredLikelihoodInfomationHMC", and stored information specific to the derivatives in various different objects the constructors of which require a "StoredLikelihoodInformationHMC" object. All of the above referenced objects are all

found in the folder "DerivativeCalculators".

https://github.com/matthewberobinson/COrtHMC

7.7 Visualisations

During this project we found that we had to visualise and describe the Markov Chain and how the algorithm operates on Tree space. To do this we created two visualisations, a video of the tree as it alters throughout the chain combined with videos of the posterior densities of the substitution model parameters and site rate parameters and an rshiny for viewing the tree and the posterior density particular targeted at 5 taxa trees as they provide a good starting test. As we cannot include a video in this thesis and the rshiny is interactive we provide an example and the code below.

https://github.com/matthewberobinson/Visualisations
https://github.com/matthewberobinson/Rshiny

At 1 minute in we can observe a second peak forming in the base rate corresponding to a change in topology.

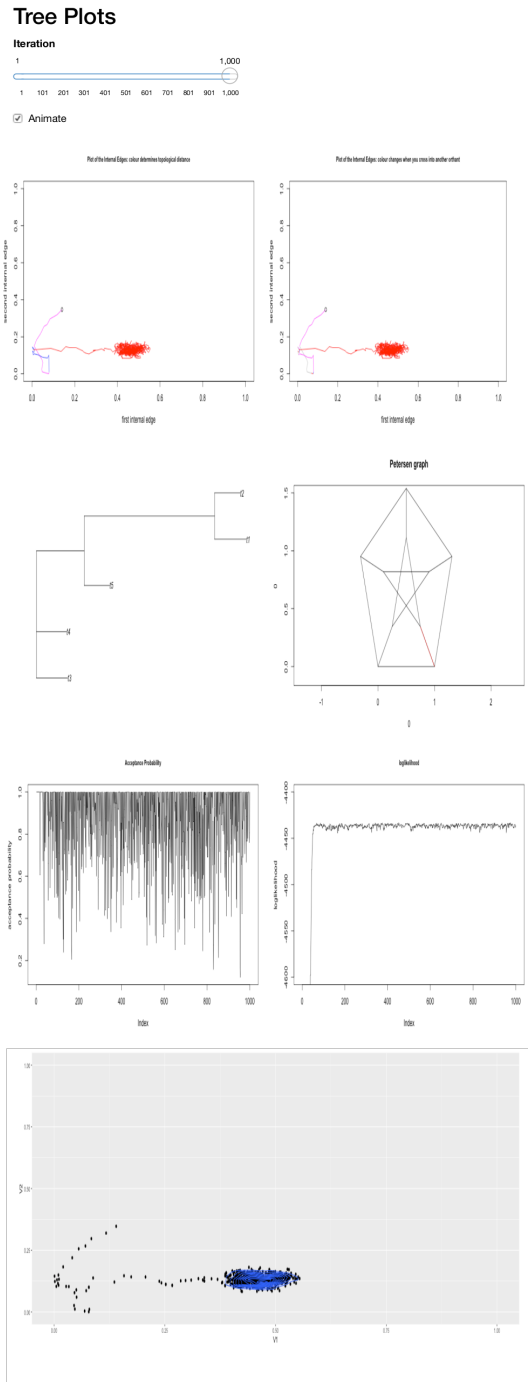


Figure 7.24: The various informative panels in our R Shiny app (RStudio, Inc, 2013). There are two plots of the internal edges of the tree. The colour either denotes distance from a given topology or merely a change of topology. We can see the current tree and it's location on the Petersen graph. We can see the acceptance rate and the logposterior, as well as a density plot.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

We have constructed a method for conducting HMC traversing Tree-space. We then streamlined the process as much as possible. Many of the techniques we developed to improve COrtHMC can be used to improve other forms of HMC. These include such examples as inverting the leapfrog integrator, stabilising and fixing the momenta kernel, parachuting the momenta and using the Farrow transformation to conduct HMC on the simplex. We have justified certain choices of priors and invalidated gamma priors with small α. Through our visualisation we can also see the coupling the tree in Tree space and the substitution model parameters. We have also employed the use of our own measure of ESS for tree parameters, which encompasses how the tree changes as a whole.

However HMC greatly increases the complexity of the computations involved in MCMC, and when put to the test COrtHMC does not seem to yield a significant improvement on Metropolis-within-Gibbs.

We also looked into some work that ended up being unrelated, in particular Windowed HMC (Neal et al., 2011), which we adapted and the relationship between Operads and trees which we didn't manage to use Baez and Otter (2015).

8.2 Future Work

It is important that COrtHMC can adapt itself to different alignments without a lot of user interaction. Currently it requires either a lot of tuning or uses the Dual Averaging algorithm from Hoffman and Gelman (2014), see section A.2. In Fig: 7.14 we saw that there is a strong relationship between the optimum stepsize and the logposterior. With a little bit of extra exploration it may be possible to come up with an algebraic form for the stepsize dependent on the logposterior. We might even be able to show that due to convergence of the logposterior, the stepsize satisfies the simultaneous uniform ergodicity and diminishing adaptation requirements to produce an ergodic chain (Roberts and Rosenthal, 2007).

We also noticed a significant improvement from applying the stabilised version of the Fisher information matrix. The Fisher information metric is not the only geometry we can use, in particular Critchley et al. (2002) recommends various different preferred point geometries. We have also in no way exhausted the various different crossing methods, in particular a perpendicular crossing method resulting in proposing the midpoint between the reflecting and projecting crossing methods might produce a better result than either. In section 5.9 we discussed two other ways of performing HMC on Tree space and reasoned why they were inappropriate, coding them up would provide a practical validation. Much of my time spent on this project has been in optimising code, during which I have noticed that small changes such as storing ordered lists of edges and vertices preventing them from being recomputed at each iteration can provide significant improvement in the speed of the algorithm. It is possible that I may have missed something. Furthermore, while the time cost of parallelisation is too great for the trivial approach, a non-trivial approach might produce better results. Finally while I have conducted many runs over a lot of synthetic data I have not tested how COrtHMC runs with large numbers of taxa for real data.

Appendix A

Algorithms

In this appendix we outline several algorithms for the sake of completeness that would either be time consuming for the reader or distracting from the overall outline of the thesis.

A.1 The GTP algorithm for computing geodesics in polynomial time

1. If $\mathcal{E} = (E_1, ..., E_k)$ and $\mathcal{E}' = (E'_1, ..., E'_k)$ are partitions of E and E', for each $i > j$ E_i and E_j are compatible

   Definition. Compatible

   Two splits $S = \mathcal{E}|\bar{\mathcal{E}}$ and $S' = \mathcal{E}'|\bar{\mathcal{E}}'$ are compatible if the intersection of one of the sets they generate is empty. i.e. if one of the following is true $\mathcal{E}|\bar{\mathcal{E}}' = \emptyset$, $\mathcal{E}|\bar{\mathcal{E}} = \emptyset$, $\bar{\mathcal{E}}|\bar{\mathcal{E}}' = \emptyset$, $\mathcal{E}|\mathcal{E}' = \emptyset$

2. $\frac{|A_1|}{|B_1|} \leq \frac{|A_2|}{|B_2|} \leq ... \leq \frac{|A_k|}{|B_k|}$

3. For each pair A_i, B_i there is a non-trivial partition of A_i, C_1, C_2 and B_i D_1, D_2 such that C_2 is compatible with D_1 and $\frac{|C_1|}{|D_1|} < \frac{|C_2|}{|D_2|}$

Algorithm A.1: The GTP algorithm

- Inputs: Trees T and T' and proper path between them Γ^0 with support
- Initialise: form the incompatibility graph between T and T'. Set Γ^0 to be the cone path between the two.
- At each stage Γ^i is a proper path and has support $\mathcal{E}^i$ and $\mathcal{E}'^i$ satisfying 1 and 2.
- For each support pair solve the extension problem.
- If every min weight cover found has weight ≥ 1 then Γ^i satisfies 3
- Choose any min weight cover $C_1 \cup D_2$ having weight $\frac{|C_1|^2}{|A_j|^2} + \frac{|D_2|^2}{|B_j|^2}$ replace A_j and B_j with C_1, C_2 and D_1, D_2 and replace Γ^i with the associated proper path Γ^{i+1}
- output: Geodesic Γ^n where Γ^n is the geodesic between T and T'

A.2 The Dual Averaging algorithm

Algorithm A.2.1: Heuristic for choosing an initial value of ϵ (Hoffman and Gelman, 2014)

1. Parameters: Hamiltonian H, starting position x_0, starting momenta p_0 tuning parameters a, integrator following Hamiltonian flows I, number of iterations k.
2. Initialise $\epsilon = 1$ and $p \sim N(\cdot; 0, Id)$.
3. Calculate $(x', p') = I(x, p, \epsilon)$.
4. Set $a \leftarrow 2\mathbb{I}\left[\frac{\pi(x',p')}{\pi(x_0,p_0)} > 0.5\right] - 1$.
5. While $\left[\frac{\pi(x',p')}{\pi(x_0,p_0)} > 0.5\right]^a > 2^{-a}$
   (a) $\epsilon = 2^a \epsilon$
   (b) Set $x', p' \leftarrow I(x, p, \epsilon)$.
6. output: ϵ.

Algorithm A.2.2: Hamiltonian Monte Carlo with Dual Averaging (Hoffman and Gelman, 2014)

1. Parameters: Hamiltonian H, starting position x_0, starting momenta p_0, optimum δ, λ, integrator following Hamiltonian flows I, number of iterations M, number of adaptations M^{adapt}.

2. Initialise $\epsilon_0 = ApplyHeuristic(x_0)$, $\mu = \log(10\epsilon_0)$, $\bar{\epsilon} = 1$, $\bar{H}_0 = 0$, $\gamma = 0.05$, $t_0 = 10$, $\kappa = 0.75$.

3. for $i = 1, ..., M$

   (a) $p \sim N(\cdot; 0, \mathbb{I})$

   (b) $x^i = x^{i-1}$, $\tilde{x} = x^{i-1}$, $\tilde{p} = p$.

   (c) Calculate $(x', p') = I(x, p, \epsilon)$.

   (d) With probability $a = \min\{1, \frac{\pi((\tilde{x},\tilde{p}))}{\pi((x^{m-1},p))}\}$ set $x^i = \tilde{x}$ and $p^i = -p$.

   (e) if $i \leq M^{adapt}$:

   - Set $\bar{H}_i = (1 - \frac{1}{i+t_0})\bar{H}_{i-1} + \frac{1}{i+t_0}(\delta - a)$.
   - Set $\log \epsilon_i = \mu - \frac{\sqrt{i}}{\gamma}\bar{H}_i$ and $\log \bar{\epsilon}_i = i^{-\kappa} \log \epsilon_i + (1 - i^{-\kappa}) \log \bar{\epsilon}_{m-1}$.

   (f) Otherwise set $\epsilon_i = \bar{\epsilon}_{M^{adapt}}$.

A.3 The various different NUTS algorithms

A.3.1 Simplified NUTS

Algorithm A.3.1.1: BuildTree for Simplified NUTS Hoffman and Gelman (2014)

- input: starting position x, starting momenta p, tuning parameters $\epsilon > 0$ and $\Delta_{max} > 0$, integrator I, $u \in \mathbb{R}_{\geq 0}$, direction $v \in \{-1, 1\}$ and counter $j \in \mathbb{N}$.
- If $j = 0$:
  1. Calculate $(x', p') = I(x, p, v\epsilon)$.
  2. Set $\mathcal{C}' = \begin{cases} \{(x', p')\} \text{ if } u < \exp\left[-H(x', p')\right]. \\ \emptyset \text{ o/w} \end{cases}$
  3. Set $s' = \mathbb{I}\left[\exp\left[-H(x', p')\right] > \log u - \Delta_{max}\right]$.
  4. Set $x^+ = x^- = x'$ and $p^+ = p^- = p'$.
- Otherwise:
  1. Set $x^-, p^-, x^+, p^+, \mathcal{C}', s' \leftarrow BuildTree(x, p, u, v, j-1, \epsilon)$.
  2. If $v_j = 1$ $x^-, p^-, \mathcal{C}'', s'' \leftarrow BuildTree(x^-, p^-, u, v, j-1, \epsilon)$ otherwise set $x^+, p^+, \mathcal{C}'', s'' \leftarrow BuildTree(x^+, p^+, u, v, j-1, \epsilon)$.
  3. Set $s' = s's''\mathbb{I}\left[(x^+ - x^-) \cdot p^- \geq 0\right]\mathbb{I}\left[(x^+ - x^-) \cdot p^+ \geq 0\right]$ and $\mathcal{C}' = \mathcal{C}' \cup \mathcal{C}''$.
- Output: $x^-, p^-, x^+, p^+, \mathcal{C}', s$.

Algorithm A.3.1.1: The Simplified NUT sampler (Hoffman and Gelman, 2014)

- Input: starting position x_0, tuning parameters $\epsilon > 0$ and $\Delta_{max} > 0$, integrator I integrating along Hamiltonian flows, number of iterations n
- For $i = 1, \ldots, n$:
  1. Draw $p^0 \sim N(\cdot; 0, Id)$.
  2. Draw $u \sim U(\cdot; [0, \exp(-H(x, p))])$.
  3. Set $x^- = x^{i-1}$, $x^+ = x^{i-1}$, $p^- = p^0$, $p^+ = p^0$, $j = 0$, $\mathcal{C} = \{(x^{i-1}, p^0)\}$, $s = 1$.
  4. While $s = 1$:
     (a) Draw $v_j \sim U(\cdot; \{-1, 1\})$.
     (b) If $v_j = -1$ then $x^-, p^-, \mathcal{C}', s' \leftarrow BuildTree(x^-, p^-, u, v_j, j, \epsilon)$ otherwise $x^+, p^+, \mathcal{C}', s' \leftarrow BuildTree(x^+, p^+, u, v_j, j, \epsilon)$.
     (c) If $s' = 1$ set $\mathcal{C} = \mathcal{C} \cup \mathcal{C}'$.
     (d) Set $s = s' \mathbb{I}\left[(x^+ - x^-) \cdot r^- \geq 0\right] \left[(x^+ - x^-) \cdot r^+ \geq 0\right]$ and $j = j + 1$.
- Draw $(x, p) \sim U(\cdot; \mathcal{C})$.
- Output: (x, p) where (x, p) is a single output from $\mathcal{C}$ found by integrating along the intergral curve of the Hamiltonian.

A.3.2 Efficient NUTS

Algorithm A.3.2.1: BuildTree for Efficient NUTS(Hoffman and Gelman, 2014)

- Input: starting position x, starting momenta p, tuning parameters $\epsilon > 0$ and $\Delta_{max} > 0$, integrator I, $u \in \mathbb{R}_{\geq 0}$, direction $v \in \{-1, 1\}$, counter $j \in \mathbb{N}$ and tree set size n.
- If $j = 0$:
  1. Calculate $(x', p') = I(x, p, v\epsilon)$.
  2. Set $n' = \mathbb{I}\left[u < \exp\left[-H(x', p')\right]\right]$, $s' = \mathbb{I}\left[\exp\left[-H(x', p')\right] > \log u - \Delta_{max}\right]$, $x^+ = x^- = x'$ and $p^+ = p^- = p'$.
- Otherwise set $x^-, p^-, x^+, p^+, x', n', s' = BuildTree(x, p, u, v, j - 1, \epsilon)$.
- If $v_j = 1$ set $x^-, p^-, x'', n'', s'' = BuildTree(x^-, p^-, u, v_j, j, \epsilon)$ otherwise set $x^+, p^+, x'', n'', s'' = BuildTree(x^+, p^+, u, v_j, j, \epsilon)$.
- With probability $\frac{n''}{n'+n''}$ set $x' = x''$, $s' = s''\mathbb{I}\left[(x^+ - x^-) \cdot p^- \geq 0\right]\left[(x^+ - x^-) \cdot p^+ \geq 0\right]$ and $n' = n' + n''$.
- Output: $x^-, p^-, x^+, p^+, x', p', n', s$

Algorithm A.3.2.2: Efficient NUTS Hoffman and Gelman (2014)

- Parameters: starting position x_0, tuning parameters $\epsilon > 0$ and $\Delta_{max} > 0$, integrator following Hamiltonian flows I, number of iterations k
- for $i = 1, \ldots, n$:
  1. Draw $p^0 \sim N(\cdot; 0, Id)$
  2. Draw $u' \sim U(\cdot; [0, \exp(-H(x, p))])$
  3. Set $x^- = x^{i-1}$, $x^+ = x^{i-1}$, $p^- = p^0$, $p^+ = p^0$, $j = 0$, $n = 1$ and $s = 1$
  4. While $s = 1$:
     (a) Draw $v_j \sim U(\cdot; \{-1, 1\})$
     (b) If $v_j = -1$ then set $x^-, p^-, x', n', s' = BuildTree(x^-, p^-, u, v_j, j, \epsilon)$ otherwise set $x^+, p^+, x', n', s' \leftarrow BuildTree(x^+, p^+, u, v_j, j, \epsilon)$.
  5. If $s' = 1$
     (a) Draw $u' \sim U[0, 1]$ if $u' \leq \frac{n'}{n}$ set $x^i = x'$
     (b) Set $s' = s'' \mathbb{I}\left[(x^+ - x^-) \cdot r^- \geq 0\right]\left[(x^+ - x^-) \cdot r^+ \geq 0\right]$, $j = j + 1$ and $n = n + n'$
- output: (x_i, p_i) is the result of integrating along the intergral curve of the Hamiltonian for a set amount of time.

A.4 Shadow HMC

All that is left to do is formulate the shadow Hamiltonian based on an approximation. There is a general formula but for our purposes we only show the 4th order shadow Hamiltonian.

$$H(x,p) = \frac{1}{2\epsilon}(A_{10} - \frac{1}{6}A_{12})$$

where

$$\begin{aligned} A_{10} =& y_1 + mass \times 0.5 \times (x_{i+1} - x_{i-1})p_i - z_1 + mass \times 0.5 \times (p_{i+1} - p_{i-1})x_i - 0.5 \times (\beta_{i+1} - \beta_{i-1}) \\ A_{12} =& y_2 - z_2 \end{aligned}$$

where

$$\begin{aligned} y_1 =& y_1 + mass \times 0.5 \times (x_{i+1} - x_{i-1})p_i \\ z_1 =& z_1 + mass \times 0.5 \times (p_{i+1} - p_{i-1})x_i \\ y_2 =& y_2 + mass \times 0.5 \times (x_{i+1} - x_{i-1})(p_{i+1} - 2 \times p_i + p_{i-1}) \\ z_2 =& z_2 + mass \times 0.5 \times (p_{i+1} - p_{i-1})(x_{i+1} - 2 \times x_i + x_{i-1}) \end{aligned}$$

and β_i is a constant in $\exp(-\beta H(x,p))$ often called the temperature, here propogated along with the momenta.

The *mass* is a value corresponding to the covariance in $K(p;x)$ when $K(p;x)$ is normal.

Appendix B

Proofs

We follow the general layout of in Probabilistic Path Hamiltonian Monte Carlo.

Proposition B.1. *The Hamiltonian H is not conserved along any system dynamics.*

Proof. Clearly the Hamiltonian is not preserved as it will depend on the choice of C. It is true that the Hamiltonian is preserved within an orthant. CortHMC preserves the hamiltonian on an orthant but may change it upon crossing into a new orthant. □

Proposition B.2. *The chain generated by CortHMC is Markov.*

Proof. CortHMC is defined by several kernels each of which are Markov, hence the composition is. □

We now follow the same arguments as outlined in (Dinh et al., 2017). This is because CortHMC is deeply related to ppHMC and is designed to carry out the same function. We include the full proof for completion.

Proposition B.3. *Lemma 3.4 (Dinh et al., 2017)*
For every sequence of topologies $\omega = \{\tau^{(0)}, ..., \tau^{(n_\omega)}\}$ and every set with positive measure $B \subset \mathcal{M}$, let B_ω the set of all $(\tau', q') \in B$ such that (τ', q') can be reached from $(\tau^{(0)}, q^{(0)})$ in k CortHMC steps and such that the sequence of topologies crossed by the topologies crossed by trajectory is ω. Denote by $I_{B,\omega}$ the set of all sequences of initial momenta for each CortHMC step $\{p^{(0)}, ..., p^{(k)}\}$ that make such a path possible.
Then, if $\mu(I_{B,\omega}) = 0$, then $\mu(B_\omega) = 0$.

Proof. For a path Σ of k leapfrog steps and n boundary steps denote by

$$F_\Sigma = \{(\ell^{(0)}, \tau^{(0)}), ..., (\ell^{(n+k)}, \tau^{(n+k)})\}$$

the states which are visited.

If the sequence of topologies crossed by Σ has been prespecified then the whole path is deterministic based on the initial momenta $p^{(0)}$. Hence the functions:

$$\phi_{i,\omega}(p) := (\ell^i, \tau^i) \qquad \forall p \in I_{B,\omega}$$

are well-defined. Show that $\phi_{1,\omega}$ is Lipschitz. i.e. given p and p' $d(\phi_{1,\omega}(p), \phi_{1,\omega}(p)) \leq Cd(p, p')$. d is defined as the Euclidean metric on the cotangent space and d is defined as the Euclidean metric defined by Euclidean distance where two orthants are joined via the negative plane as the algorithm cannot explore further than an orthant in a single jump, this is a well-defined metric. There are two cases:

1. If topology does not alter, $\phi_{1,\omega}(p)$ is the result of the leapfrog algorithm which from standard results about HMC on Euclidean spaces results in $\phi_{1,\omega}(p)$ being Lipschitz.

2. If topology does alter $\phi_{1,\omega}(p)$ is a composition of the leapfrog algorithm on Euclidean spaces, a linear map and a projection. Linear maps and projections are Lipschitz $\phi_{1,\omega}(p)$ is Lipschitz.

As $\phi_{n,\omega}(p) = \underbrace{\phi_{1,\omega} \cdot ... \cdot \phi_{1,\omega}}_{n \text{ times}}$ and compositions of finite Lipschitz functions are Lipschitz, $\phi_{n,\omega}$ is Lipschitz on $I_{B,\omega}$. As Lipschitz functions preserve zero measure if $\mu(I_{B,\omega}) = 0$ then $\mu(B) = 0$. □

Theorem B.1. *Theorem 3.1 (Dinh et al., 2017)*
The Markov chain generated by CortHMC is ergodic.

Proof. By exactly the same arguments as in (Dinh et al., 2017) the chain is irreducible. It preserves π due to proposition 6.3. So all that is required is aperiodicity. Suppose it is periodic with period d, then $\mathcal{M}$ must be able to be partitioned into at most d orbits and at least two. i.e. k disjoint subsets $X_1, ..., X_k$ of $\mathcal{M}$ with positive measure. Around any point $x \in \mathcal{M}$ there is a neighbourhood U_x reachable from x by Hamiltonian dynamics. Hence $\mu(U_x \cap X_1) = 0$ as U_x exists for almost every $x \in X_1$, $\mu(X_1) = 0$ a contradiction. □

B.0.1 Theorem 6.4

Proof. The likelihood is defined as:

$$\mathcal{L}(X|\ell, \tau, \theta, \alpha) = \prod_{i=1}^{n} \sum_{k \in \gamma} \mathbb{P}(k) \sum_{x \in \mathcal{A}} \underline{\pi}(x) \left[\prod_{u \prec v_0} \left[\sum_{y \in \mathcal{A}} P_{x,y;k} \{\ell(u, v_0), \theta\} \mathcal{L}_u(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right]$$

The process of travelling across codimension 1 strata involves a particular $\ell(u,v) \to 0$ followed by a nearest neighbour interchange and an increase in the length of a new edge. First we show that the process is continuous upto and including $\ell(u,v) = 0$ and that the same is true for all derivatives with respect to $\ell(u,v)$. We then demonstrate that a nearest neighbour interchange does not alter the likelihood, nor the derivative and finally by the initial argument the likelihood and derivatives are continuous in any other orthant.

Since we can specify the root of the tree let us take v_0 to be the root and u_0 to be a vertex connected with v_0 such that $e(u_0, v_0)$ is the edge via which we are going to cross the codimension 1 boundary and $\ell(u_0, v_0)$ is the length of that edge. Apart from $P_{x,y;k}\{\ell(u_0,v_0),\theta\}$ everything else is independent of the edge $e(u_0,v_0)$ and so constant with respect to $\ell(u_0,v_0)$ and the derivatives are 0 with respect to $\ell(u_0,v_0)$. In subsection 2.1.2 we show that $P_{x,y;k}\{\ell(u_0,v_0),\theta\} = P_0 \exp(Q\ell(u_0,v_0))$ for some P_0 constant and Q a rate matrix specified by θ, we know multiplying by a constant is continuous and exp is a continuous function, since compositions of continuous functions are continuous so is $P_0 \exp(Q\ell(u_0,v_0))$. If we take the nth derivative of $P_{x,y;k}\{\ell(u_0,v_0),\theta\}$ we get $P_0 \exp(Q\ell(u_0,v_0))Q^n$. By the same reasoning the nth derivative on an orthant is continuous.

When $\ell(u,v_0) = 0$, $P_{x,y;k}\{\ell(u,v_0),\theta\} = \begin{cases} 1 \text{ if } x = y \\ 0 \text{ if } x \neq y \end{cases}$.

This means that the likelihood can be expressed as follows

$$\mathcal{L}(X|\ell,\tau,\theta,\alpha) = \prod_{i=1}^{n} \sum_{k\in\gamma} \mathbb{P}(k) \sum_{x\in\mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u\prec v_0}^{u\neq u_0} \left[\sum_{y\in\mathcal{A}} P_{x,y;k}\{\ell(u,v_0),\theta\} \mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha) \right] \right] \cdot \\ \times \left[P_{x,x;k}\{\ell(u_0,v_0),\theta\} \mathcal{L}_{u_0}(\chi_i|x,\ell,\tau,\theta,\alpha) \right]$$

A nearest neighbour interchange across $e(u_0,v_0)$ consists of a swap of two subtrees, one connected to a child v' of v_0, the other to a child u' of u_0 so that v' is a child of u_0 with length $\ell(v',u_0) = \ell(v',v_0)$ and u' is a child of v_0 with length $\ell(u',v_0) = \ell(u',u_0)$ but nothing else is altered in any lower subtrees. We now write the likelihood of the same tree having experienced a nearest neighbour interchange.

$$\mathcal{L}(X|\ell,\tau,\theta,\alpha) = \prod_{i=1}^{n} \sum_{k\in\gamma} \mathbb{P}(k) \sum_{x\in\mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u\prec v_0\cup u'}^{u\neq u_0, u\neq v'} \left[\sum_{y\in\mathcal{A}} P_{x,y;k}\{\ell(u,v_0),\theta\} \mathcal{L}_u(\chi_i|y,\ell,\tau,\theta,\alpha) \right] \right]$$

$$
\begin{aligned}
&\times \left[\prod_{w \prec u_0 \cup v'}^{w \neq u'} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(w, u_0), \theta\} \mathcal{L}_{u_0,w}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
=& \prod_{i=1}^{n} \sum_{k \in \gamma} \mathbb{P}(k) \sum_{x \in \mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u \prec v_0 \cup u'}^{u \neq u_0, u \neq v'} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(u, v_0), \theta\} \mathcal{L}_{u}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
&\times \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(v', u_0), \theta\} \mathcal{L}_{u_0,v'}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \\
&\times \left[\prod_{w \prec u_0}^{w \neq u'} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(w, u_0), \theta\} \mathcal{L}_{u_0,w}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
=& \prod_{i=1}^{n} \sum_{k \in \gamma} \mathbb{P}(k) \sum_{x \in \mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u \prec v_0 \cup u'}^{u \neq u_0} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(u, v_0), \theta\} \mathcal{L}_{u}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
&\times \left[\prod_{w \prec u_0}^{w \neq u'} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(w, u_0), \theta\} \mathcal{L}_{u_0,w}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
=& \prod_{i=1}^{n} \sum_{k \in \gamma} \mathbb{P}(k) \sum_{x \in \mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u \prec v_0 \cup u'}^{u \neq u_0} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(u, v_0), \theta\} \mathcal{L}_{u}(\chi_i | y, \theta, k) \right] \right] \\
&\times \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(u', v_0), \theta\} \mathcal{L}_{v_0,u'}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \\
&\times \left[\prod_{w \prec u_0}^{w \neq u'} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(w, u_0), \theta\} \mathcal{L}_{u_0,w}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
=& \prod_{i=1}^{n} \sum_{k \in \gamma} \mathbb{P}(k) \sum_{x \in \mathcal{A}} \underset{\sim}{\pi}(x) \left[\prod_{u \prec v_0 \cup u'}^{u \neq u_0} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(u, v_0), \theta\} \mathcal{L}_{u}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right] \\
&\times \left[\prod_{w \prec u_0}^{w} \left[\sum_{y \in \mathcal{A}} P_{x,y;k}\{\ell(w, u_0), \theta\} \mathcal{L}_{u_0,w}(\chi_i | y, \ell, \tau, \theta, \alpha) \right] \right].
\end{aligned}
$$

This demonstrates that the likelihood $\mathcal{L}(X|\theta)$ is continuous, to show that it is smooth we take the derivative of the likelihood with respect to an edge. In subsection 5.5 we show that the derivative of the likelihood is

$$\prod_{i=1}^{n}\left[\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u'\prec v}^{u'\neq u}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u',v)\}\mathcal{L}_{u'}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\times\sum_{y\in\mathcal{A}}kQP_{x,y;k}\{\ell(u,v)\}\mathcal{L}_{u}(\chi_i|y,\ell,\tau,\theta,\alpha)\right].$$

It is not hard to see that further derivatives involve higher powers of k and Q. We now need to show that this derivative agrees when crossing codimension 1 strata, which is equivalent to saying the limit from one orthant is the same as from the other and so the derivative exists. We apply the same argument as used for the likelihood.

$$\begin{aligned}
\mathcal{L}(X|\ell,\tau,\theta,\alpha)=&\prod_{i=1}^{n}\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u\prec v_0\cup u'}^{u\neq u_0,u\neq v'}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v_0),\theta\}\mathcal{L}_{u}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\\
&\times kQ\left[\prod_{w\prec u_0\cup v'}^{w\neq u'}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(w,u_0),\theta\}\mathcal{L}_{u_0,w}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\\
=&\prod_{i=1}^{n}\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u\prec v_0\cup u'}^{u\neq u_0,u\neq v'}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v_0),\theta\}\mathcal{L}_{u}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\\
&kQ\times\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(v',u_0),\theta\}\mathcal{L}_{u_0,v'}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\\
&\times\left[\prod_{w\prec u_0}^{w\neq u'}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(w,u_0),\theta\}\mathcal{L}_{u_0,w}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\\
=&\prod_{i=1}^{n}\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u\prec v_0\cup u'}^{u\neq u_0}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v_0),\theta\}\mathcal{L}_{u}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\\
&\times kQ\left[\prod_{w\prec u_0}^{w\neq u'}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(w,u_0),\theta\}\mathcal{L}_{u_0,w}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]\\
=&\prod_{i=1}^{n}\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u\prec v_0\cup u'}^{u\neq u_0}\left[\sum_{y\in\mathcal{A}}P_{x,y;k}\{\ell(u,v_0),\theta\}\mathcal{L}_{u}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right]
\end{aligned}$$

$$
\begin{aligned}
&\times kQ\left[\sum_{y\in\mathcal{A}} P_{x,y;k}\{\ell(u',v_0),\theta\}\mathcal{L}_{v_0,u'}(\chi_i|y,\ell,\tau,\theta,\alpha)\right] \\
&\times\left[\prod_{w\prec u_0}^{w\neq u'}\left[\sum_{y\in\mathcal{A}} P_{x,y;k}\{\ell(w,u_0),\theta\}\mathcal{L}_{u_0,w}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right] \\
=&\prod_{i=1}^{n}\sum_{k\in\gamma}\mathbb{P}(k)\sum_{x\in\mathcal{A}}\underset{\sim}{\pi}(x)\left[\prod_{u\prec v_0\cup u'}^{u\neq u_0}\left[\sum_{y\in\mathcal{A}} P_{x,y;k}\{t(u,v_0),\theta\}\mathcal{L}_{u}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right] \\
&\times kQ\left[\prod_{w\prec u_0}^{w}\left[\sum_{y\in\mathcal{A}} P_{x,y;k}\{\ell(w,u_0),\theta\}\mathcal{L}_{u_0,w}(\chi_i|y,\ell,\tau,\theta,\alpha)\right]\right].
\end{aligned}
$$

It should be clear that the same argument holds for higher order derivatives. This therefore shows that the likelihood is smooth with respect to edge lengths and the nearest neighbour substitution algorithm when applied to edges of zero length. □

Appendix C

Literature Review

Here we outline definitions and techniques that the reader may be interested in but would break the immersion for someone who already knows the papers referenced.

C.0.1 Yang and Rannala (1997)

Definition (Cophenetic Matrix Representation of a tree)**.** The cophenetic matrix representation of a tree is a symmetric matrix $A = \{a_{ij}\}$ such that $a_{ij} = d(n_i, n_j)$ the time to coalesce / total edge length between each extant pair. As such each matrix is unique up to ordering of extant vertices.

Yang and Rannala (1997) uses a transition probability of

$$q_{ij} = \begin{cases} \frac{\beta}{h_j - 1} & \text{no topological change} \\ \frac{1-\beta}{2(s-2)h_j} & \text{topological change} \end{cases}$$

$h_j =$ distinct labelled histories $\quad s =$number of extant species $\quad \beta =$ model parameter

There are $s!(s-1)!/2^{s-1}$ different labelled histories, that is $s-1$ speciation events occur splitting up s extant species.

C.0.2 Altekar et al. (2004)

$(MC)^3$ transitions between the different temperatures by employing a Metropolis-Hastings acceptance probability, 3.3:

$$a = \min\left[1, \frac{\pi(\hat{z})q(z_{i-1}, \hat{z})}{\pi(z_{i-1})q(\hat{z}, z_{i-1})}\right]$$

A chain heated by temperature β uses acceptance probability:

$$a_\beta = \min\left[1, \frac{\pi(\hat{z})^\beta q(z_{i-1}, \hat{z})}{\pi(z_{i-1})^\beta q(\hat{z}, z_{i-1})}\right]$$

In Metropolis-Coupled MCMC then proposes to swap the states of two chains with probability:

$$a_{\text{swap }\beta_1,\beta_2} = \min\left[1, \frac{\pi(z_{\beta_1})^{\beta_2}\pi_(z_{\beta_2})^{\beta_1}}{\pi(z_{\beta_1})^{\beta_1}\pi_(z_{\beta_2})^{\beta_2}}\right]$$

where $\beta_1 < \beta_2$ and z_β is the state sampled from the MCMC process with temperature β.

C.0.3 Beast and MrBayes $(MC)^3$

Beast and MrBayes use temperature scalars $\beta_i = \frac{1}{1+(i-1)\delta t}$ and only propose swapping states between β_i and β_{i+1}. Beast also can use temperature scalar $\beta_i = 1 - cdf\frac{i-1}{\text{number of chains}}$ where cdf is the cumulative density function of the beta distribution with $\alpha = 1$. Best also implements an adaptive method for choosing an updating δt after a burn in period

$$\delta t_{new} = \max\left[0, \delta t_{current} + \frac{p_{global} - p_{target}}{\text{number of exchanges}}\right]$$

where p_{global} is the average acceptance probability and p_{target} is the acceptance probability we want.

C.0.4 Probabilistic Path Hamiltonian Monte Carlo

Probabilistic Path Hamiltonian Monte Carlo employs the Leap-prog algorithm to cross orthants.

Leap-prog Algorithm

- Input: starting point z_0, starting momenta p_0; step size $\epsilon > 0$; number of iterations l; $\frac{\partial H(z,p)}{\partial p}$ and $\frac{\partial H(z,p)}{\partial z}$.
- 1. $p_{i+\frac{1}{2}} = p_i - \frac{\epsilon}{2}\frac{\partial H(z,p)}{\partial z}|_{(z_i,p_i)}$,
  2. if $FirstUpdateEvent(z, p, \epsilon) = \emptyset$

$$z_{i+1} = z_i + \epsilon p_{i+\frac{1}{2}}$$

otherwise set $t = 0$.

while $FirstUpdateEvent(z, p, \epsilon - t) \neq \emptyset$:

let $\hat{z}, e, I$ be the first update event $FirstUpdateEvent(z, p, \epsilon - t)$, $\hat{z}$ is the state where an orthant boundary is crossed, e, and I the set of indices where it crossed. Then:

$$t = t + e$$

Sample new topology τ at random from neighbouring orthants $\tau \sim \mathcal{Z}$.

$$p_{i+\frac{1}{2}} = -p_{i+\frac{1}{2}}$$

  3. $z_{i+1} = \hat{z} + (\epsilon - t)p_{i+\frac{1}{2}}$
  4. $p_{i+1} = p_{i+\frac{1}{2}} - \frac{\epsilon}{2}\frac{\partial H(z,p)}{\partial z}|_{(z_{i+1},p_{i+\frac{1}{2}})}$.
- Output: pair (z_l, p_l) approximately on the flow line starting at z_0, p_0.

C.0.5 PosetSMC

The weighted particle measure is

$$\pi_{r,K}(\cdot) = \mathbb{E}_w(\cdot) = \sum_{k=1}^{K} w_{r,k}\delta_{s_{r,k}}(\cdot)$$

where $w_{r,k}$ the are weighted according to

$$w_{r,k} = \frac{\gamma_*(s_{r,k})}{\gamma_*(\tilde{s}_{r-1,k})q(\tilde{s}_{r-1,k} \to s_{r,k})}.$$

The forest proposal distribution is q and γ_* is the posterior when extended to forests. The estimate for the posterior is then:

$$\pi_{est}(\cdot) = \mathbb{E}_{unif}(\mathbb{E}_w(\cdot)) = \prod_{r=1}^{R} \frac{1}{K} \sum_{k=1}^{K} w_{r,k} \delta_{s_{r,k}}(\cdot).$$

PosetSMC and HMC

There are two ways of combining PosetSMC, one is boring the other is more interesting.

Boring PosetSMC HMC
Boring PosetSMC HMC uses a proposal distribution between forests and trees to just use PosetSMC as a kernel within a HMC scheme.
Interesting PosetSMC HMC
Interesting PosetSMC uses HMC in the construction of q. This requires a form of HMC for forests. For this we include an adaption of COrtHMC for forests.

To conduct HMC on forests we remember that the crossing kernel occurs at every step of COrtHMC, whenever the flow is contained within an orthant the crossing kernel is the identity kernel. We replace this identity to conduct HMC on forests and provide an example for clarity.

A tree is a graph and so a collection of vertices and edges. The vertices can be split into two sets internal and external, for a phylogenetic tree these represent extinct and extant species. In order to conduct phylogenetic inference on a forest with a set number of extant leaves we need a way of connecting trees within the forest and a way of splitting trees.

For every internal edge of length l_e in the tree propose cutting the edge with probability:

$$\frac{(\frac{l_e}{2})}{(1 + (\frac{l_e}{2})^2)^2}$$

so that low edge lengths are not cut and large edge lengths have a greater chance. This gives a probability of:

$$\prod_{e\in T:e \text{ is cut}} \frac{(\frac{l_e}{2})}{(1+(\frac{l_e}{2})^2)^2} \prod_{e\in T:e \text{ is not cut}} 1 - \frac{(\frac{l_e}{2})}{(1+(\frac{l_e}{2})^2)^2}$$

For the internal vertices $V_{<3}$ of degree less than 3 draw a permutation uniformly:

$$\mathbb{P}(\sigma) = |V_{<3}|!$$

Then propose joining v and u if v follows u in the permutation with probability $1/2$ so that the total probability of joining v and u is

$$\mathbb{P}((v,u)|\sigma) = \frac{1}{2}$$

$$\mathbb{P}((v,u),\sigma) = \frac{1}{2}\frac{2\times\frac{|V_{<3}|!}{2}}{|V_{<3}|!} = \frac{1}{2(|V_{<3}|-1)!}$$

This gives a total density of

$$\prod_{e\in T:e \text{ is cut}} \frac{(\frac{l_e}{2})}{(1+(\frac{l_e}{2})^2)^2} \prod_{e\in T:e \text{ is not cut}} 1 - \frac{(\frac{l_e}{2})}{(1+(\frac{l_e}{2})^2)^2} \frac{1}{2}^{|V_{<3}|/2} \frac{1}{(|V_{<3}|-1)!}$$

We do not conduct this kind of inference for three reasons:

1. If COrtHMC performs poorly in comparison with standard MCMC then PosetSMC HMC will perform poorly in comparison with PosetSMC.
2. It is more complicated.
3. Slight tuning error has the potential to radically reduce the efficiency of the algorithm.

C.0.6 Variational Bayes

In Variational Bayesian Phylogenetic Inference the total order is such that if clade W includes clade X then $W \succ X$. Ordered by increasing depth and such that clades that are subclades of the same clade are ordered next to one another. Given this decomposition the SBN tree probability is:

$$p_{sbn}(T) = \underbrace{p(S_r = s_r)}_{\text{prob. of root base}} \prod_{i>1} \underbrace{p(S_i = s_i | S_{\sigma(i)} = s_{\sigma(i)})}_{\text{prob. of next clade base given parent clade base}} .$$

We can see the similarities to the likelihood in section 2.5.

Appendix D

Other HMC Methods

D.1 Hamiltonian Monte Carlo without Detailed Balance

Hamiltonian Monte Carlo without Detailed Balance is carried out through an algorithm called Look Ahead HMC developed by Sohl-Dickstein et al. (2014). The aim of Look Ahead HMC is to avoid efficiency loss caused by forward and reverse transitions happening with equal probability. In (Sohl-Dickstein et al., 2014) it is shown that this can provide moderately improved mixing. It has similar ideas to NUTS. It works by first transitioning to a new state via the leapfrog integrator, applying between 1 and K times with set probability, or flip the momenta. Finally corrupt the momentum with noise. While Sohl-Dickstein et al. (2014) demonstrates that this algorithm will target the correct distribution. We can think of this algorithm as being a chained form of NUTS with the turning criterion being K leapfrog iterations. It requires a lot more evaluations of the likelihood than standard HMC and would add extra factors muddying the water as to the effectiveness of HMC. As it would likely not provide any significant gains we decided that it was not to be explored during this project.

D.2 Kernel Hamiltonian Monte Carlo

Kernel Hamiltonian Monte Carlo (KMC) seeks to adapt HMC to situations where no derivative of the likelihood exists. Similar to Shadow HMC it does this by replacing the Hamiltonian with an approximation. Strathmann et al. (2015) does this in two ways that they call *lite* and *finite*.

Letting $\mathcal{H}$ be a reproducing kernel Hilbert space on $\mathcal{M}$. A Hibert space is a reproducing kernel Hilbert space if the evaluation functional L_x is a bounded operator i.e. for any point x $|L_x(f \in \mathcal{H}) \leq M||f||_{\mathcal{H}}$.

Model $\pi(x)$ with an exponential family model so that:

$$C\pi(x) \approx \exp\left(\langle f, k(x,\cdot)\rangle_{\mathcal{H}} - A(f)\right).$$

The function k is a unique function that is symmetric positive definite and has the property $\langle f, k(x,\cdot)\rangle_{\mathcal{H}} = f(x)$. The function $A(f) = \log \int_{\mathcal{M}} \exp\left(\langle f, k(x,\cdot)\rangle_{\mathcal{H}}\right) dx$. This family is dense in the space of continuous densities on compact domains with respect to total variation.

This leads to the problem of constructing suitable f. This is done from the data. Both approaches *lite* and *finite* use score matching.

In score-matching the aim is to construct some $\tilde{\pi}(x, f)$ such that

$$J(f) = \frac{1}{2}\int_{\mathcal{M}} \pi(x)||\Delta \log(\tilde{\pi}(x,f)) - \Delta \log(\pi(x))||_2^2 dx.$$

Given samples $x \in \mathcal{D}$ this can be computed without π by

$$\hat{J}(f) = \frac{1}{n}\sum_{x\in\mathcal{D}}\sum_{l=1}^{d}\left[\frac{\partial^2 \log(\tilde{\pi}(x,f))}{\partial x_l^2} + \frac{1}{2}\left(\frac{\partial \log(\tilde{\pi}(x,f))}{\partial x_l}\right)\right]. \tag{9}$$

It is this that enables the derivative of π to be avoided.

The *lite* model takes a solution in the form of the span $(\{k(z_i,\cdot)\}_{i=1}^h)$ so that $f(x) = \sum_{i=1}^n \alpha_i k(z_i, x)$ where α_i are obtained by minimising (1). Strathmann et al. (2015) show that this can be done by taking $\alpha_\lambda = -\frac{\sigma}{2}(C - \lambda I)^{-1} b$ where

$$b = \sum_{l=1}^{d}\left(\frac{2}{\sigma}(K_{s_l} + D_{s_l}K\mathbf{1} - 2D_{x_l}K_{x_l}) - K\mathbf{1}\right) \qquad C = \sum_{l=1}^{d}[D_{x_l}K - KD_{x_l}][KD_{x_l} - D_{x_l}K]$$

where s_l is the entry wise product of x_l with itself and D_x is a diagonal matrix composed of x, $diag(x)$.

The *finite* model attempts to fit a finite dimensional model. To do so Strathmann et al. (2015) define an approximate feature space $\mathcal{H}_m = \mathbb{R}^m$ and define ϕ_x to be the embedding of $x \in \mathcal{M}$ in $\mathcal{H}_m$ when $\mathcal{M} = \mathbb{R}^d$. Strathmann et al. (2015) then assume the model can take the form $f(x) = \langle \theta, \phi_x\rangle_{H_m} = \theta^T \phi_x$. The function $f(x)$ is then required to minimise the score function for

given data. Strathmann et al. (2015) prove that this is the case when $\theta_\lambda = (C - \lambda I)^{-1} b$ where:

$$b = -\frac{1}{n} \sum_{i=1}^{t} \sum_{l=1}^{d} \frac{\partial^2 \phi_x}{dx_l^2} \qquad C = \frac{1}{n} \sum_{i=1}^{t} \sum_{l=1}^{d} \frac{\partial \phi_{x_i}}{\partial x_l} \frac{\partial \phi_{x_i}}{\partial x_l}^T$$

KMC in both forms calculates this approximation at each iteration and then proceeds by standard HMC using this approximation. We do not use KHMC as we have access to an analytic form of the derivative. We include it here as it is a possible solution to a problem that arises later. Experimentally it appears to carry over the explorative properties of HMC as demonstrated on a synthetic example (Strathmann et al., 2015).

D.3 Shadow HMC

Shadow HMC developed by Izaguirre and Hampton (2004) is a technique that attempts to increase the acceptance rate of HMC by using high order approximations to the Hamiltonian. It is primarily done for an ensemble of particles. Shadow HMC requires a reweighting at the end of the process. It works by constructing $\mathcal{H}(x,p) = \max(H(x,p), H_{[2k]}(x,p) - c)$ where $H_{[2k]}$ is a Shadow Hamiltonian and c is an arbitrary constant. Shadow HMC proposes momenta $p \sim K(p;x)$ and accepts it with probability $\min(1, \frac{\exp(-H_{[2k]}(x,p)-c)}{\exp(-H(x,p))})$, but if p is rejected the process is repeated until a proposed p is accepted. A new position an momenta is then proposed based on the accepted momenta and an integrator that integrates flows of $H_{[2k]}$. This point is accepted with probability $\min(1, \frac{\pi(x',p')}{\pi(x,p)}$. Once a sequence is obtained it is reweighted by $\frac{\sum_{i=1}^{n} w_i A((x_i,p_i))}{\sum_{i=1}^{n} w_i}$. For proper canonical distributions $w_i = \frac{\exp(-H((x_i,p_i))}{-\mathcal{H}((x,p))}$.

We do not use Shadow HMC as we have the derivative, are not operating on an ensemble and still requires a large number of derivative calculations in computing x_{i+1}, x_i and x_{i-1}.

D.4 Generalised HMC

Generalised HMC is another method used primarily in biomolecular simulations and therefore on an ensemble of particles. It attempts to maintain momenta to only partially refresh momenta at each step of the Hamiltonian Monte Carlo algorithm. This is motivated by the fact that the behaviour local to the sampled point should remain consistent. In (Elena Akhmatskayaa, 2008) they find that they must minimise the rejection rate and encounter Zitterbewegung and resort to the use of a shadow Hamiltonian. While interesting, we do not outline the method further as we have already explained above we it would be impractical for our purposes. The full method can

be found in (Elena Akhmatskayaa, 2008). It may be fruitful to combine GHMC with Windowed HMC but we have not done this nor do we outline the method.

D.5 Constrained HMC

HMC occurs on a manifold. Often it is necessary to have boundaries on the manifold to constrain the state space. Constrained HMC is a method for doing this. There are several methods developed but most are similar in design. Here we go over the method developed by Michael Betancourt in (Betancourt, 2011) which is a generalisation of the method of Radford Neal in (Neal, 1994).

D.5.1 The method of Neal (1994)

Suppose we wish to constrain on $\mathbb{R}$ and sample $x \leq u$.

Radford Neal alters the potential energy function to achieve constrained HMC

$$\hat{U}(x) = U(x) + C_r(x,u) \text{ where } C_r(x,u) = \begin{cases} 0 \text{ if } q \leq u \\ r^{r+1}(x-u)^r \text{ otherwise} \end{cases} .$$

This results in the formulation for the Hamiltonian being:

$$H(x,p) = \frac{U(x)}{2} + K(p;x) + C_r(x,u) + \frac{U(x)}{2}$$

For leapfrog the only step that needs to be updated is the full step update for position. This is done by computing

$$x_{i+1} = x_i + \epsilon \frac{\partial H(x,p)}{\partial p}|_{(x_i, p_{i+\frac{1}{2}})}.$$

Then checking if $x_{i+1} \leq u$ if it is, the above is linear $C_r(x,u)$ must be zero along the whole path and x_{i+1} is left unchanged. If not the particle can be envisioned as climbing a hill until $p = 0$ then rolling down until the position $x = u$ and p is the negation of the original momenta. This leads to the following addition to step 2 of algorithm 5.

- Repeat until $x_{i+1} \leq u$,
  $x_{i+1} = u + (u - x_{i+1})$ and $p_{i+\frac{1}{2}} = -p_{i+\frac{1}{2}}$.

Intuitively the trajectory bounces off the walls given by the dynamics (Neal, 1994).

To generalise to multiple dimensions conduct the one dimensional case in each dimension.

D.5.2 The method of (Betancourt, 2011)

Michael Betancourt adapts this to multiple dimensions and general constraint $C(x) \geq 0$. This is done by altering the last two steps of algorithm 5 (Betancourt, 2011).

1. $x_{i+1} = x_i + \epsilon \frac{\partial H(x,p)}{\partial p}|_{(x_i, p_{i+\frac{1}{2}})}$
2. If $C(x_{i+1} \geq 0)$, $p_{i+1} = p_{i+\frac{1}{2}} - \frac{\epsilon}{2} \frac{\partial H(x,p)}{\partial x}|_{(x_{i+1}, p_{i+\frac{1}{2}})}$ otherwise
   (a) $n = \frac{\nabla C(x_{i+1}}{|\nabla C(x_{i+1}|}$
   (b) $p_{i+\frac{1}{2}} = p_{i+\frac{1}{2}} - 2(p_{i+\frac{1}{2}} \cdot n)n$
3. $x_{i+1} = x_{i+1} + \epsilon \frac{\partial H(x,p)}{\partial p}|_{(x_{i+1}, p_{i+\frac{1}{2}})}$
4. $p_{i+1} = p_{i+\frac{1}{2}} - \frac{\epsilon}{2} \frac{\partial H(x,p)}{\partial x}|_{(x_{i+1}, p_{i+\frac{1}{2}})}$.

This is not quite bouncing off the wall, the particle bounces off a wall parallel to the original wall but set at x_{i+1} and re-enters the space.

Appendix E

Results

E.1 Primate Alignment from MrBayes

NEXUS

Data from: Hayasaka, K., T. Gojobori, and S. Horai. 1988. Molecular phylogeny and evolution of primate mitochondrial DNA. Mol. Biol. Evol. 5:626-644.

12 taxa, 898 columns;

Tarsius syrichta: AAGTTTCATTGGAGCCACCACTCTTATAATTGCCCATGGCCTCACCTCCTCCCTATTATTTTGCCTAGCAAATACAAACTACGAACGAGTCCACAGTCGAACAATAGCACTAGCCCGTGGCCTTCAAACCCTATTACCTCTTGCAGCAACATGATGACTCCT
CGCCAGCTTAACCAACCTGGCCCTTCCCCCAACAATTAATTTAATCGGTGAACTGTCCGTAATAATAGCAGCATTTTCATGGTCACACCTAACTATTATCTTAGTAGGCCTTAACACCCTTATCACCGCCCTATATTCCCTATATATACTAATCATAACTCAACGAGGAAAATACACATATCAT
ATCAACAATATCATGCCCCCTTTCACCCGAGAAAATACATTAATAATCATACACCTATTTCCCTTAATCCTACTATCTACCAACCCCAAAGTAATTATAGGAACCATGTACTGTAAATATAGTTTAAACAAAACATTAGATTGTGAGTCTAATAATAGAAGCCCAAAGATTTCTTATTTACCAA
GAAAGTATGCAAGAACTGCTAACTCATGCCTCCATATATAACAATGTGGCTTTCTTACTTTTAAAGGATAGAAGTAATCCATCGGTCTTAGGAACCGAAAAATTGGTGCAACTCCAAATAAAAGTAATAAATTTATTTTCATCCTCCATTTTACTATCACTTACACTCTTAATTACCCCATTTA
TTATTACAACAACTAAAAAATATGAAACACATGCATACCCTTACTACGTAAAAAACTCTATCGCCTGCGCATTTATAACAAGCCTAGTCCCAATGCTCATATTTCTATACACAAATCAAGAAATAATCATTTCCAACTGACATTGAATAACGATTCATACTATCAAATTATGCCTAAGCTT
Lemur catta: AAGCTTCATAGGAGCAACCATTCTAATAATCGCACATGGCCTTACATCATCCATATTATTCTGTCTAGCCAACTCTAACTACGAACGAATCCATAGCCGTACAATACTACTAGCACGAGGGATCCAAACCATTCTCCCTCTTATAGCCACCTGATTCCCTCTTATAGCC
ACCTGATGACTACTCGCCAGCCTAACTAACCTAGCCCTACCCACCTCTATCAATTTAATTGGCGAACTATTCGTCACTATAGCATCCTTCTCATGATCAAACATTACAATTATCTTAATAGGCTTAAATATGCTCATCACCGCTCTCTATTCCCTCTATATATTAACTACTACACAACGAGGAA
AACTCACATATCATTCGCACAACCTAAACCCATCCTTTACACGAGAAAACACCCTTATATCCATACACATACTCCCCCTTCTCCTATTTACCTTAAACCCCAAAATTATTCTAGGACCCACGTACTGTAAATATAGTTTAAAAAAACACTAGATTGTGAATCCAGAAATAGAAGCTCAAACCTT
CTTATTTACCGAGAAAGTAATGTATGAACTGCTAACTCTGCACTCCGTATATAAAAATACGGCTATCTCAACTTTTAAAGGATAGAAGTAATCCATTGGCCTTAGGAGCCAAAAAATTGGTGCAACTCCAAATAAAAGTAATAAATCTATTATCCTCTTTCACCCTTGTCACACTGATTATCCT
AACTTTACCTATCATTATAAACGTTACAAACATATACAAAAACTACCCCTATGCACCATACGTAAAATCTTCTATTGCATGTGCCTTCATCACTAGCCTCATCCCAACTATATTATTTATCTCCTCAGGACAAGAAACAATCATTTCCAACTGACATTGAATAACAATCCAAACCCTAAAACTA
TCTATTAGCTT
Homo sapiens: AAGCTTCACCGGCGCAGTCATTCTCATAATCGCCCACGGGCTTACATCCTCATTACTATTCTGCCTAGCAAACTCAAACTACGAACGCACTCACAGTCGCATCATAATCCTCTCTCAAGGACTTCAAACTCTACTCCCACTAATAGCTTTTTGATGACTTCT
AGCAAGCCTCGCTAACCTCGCCTTACCCCCCACTATTAACCTACTGGGAGAACTCTCTGTGCTAGTAACCACGTTCTCCTGATCAAATATCACTCTCCTACTTACAGGACTCAACATACTAGTCACAGCCCTATACTCCCTCTACATATTTACCACAACACAATGGGGCTCACTCACCCACCAC
ATTAACAACATAAAACCCTCATTCACACGAGAAAACACCCTCATGTTCATACACCTATCCCCCATTCTCCTCCTATCCCTCAACCCCGACATCATTACCGGGTTTTCCTCTTGTAAATATAGTTTAACCAAAACATCAGATTGTGAATCTGACAACAGAGGCTTACGACCCCTTATTTACCGAG
AAAGCTCACAAGAACTGCTAACTCATGCCCCCATGTCTAACAACATGGCTTTCTCAACTTTTAAAGGATAACAGCTATCCATTGGTCTTAGGCCCCAAAAATTTTGGTGCAACTCCAAATAAAAGTAATAACCATGCACACTACTATAACCACCCTAACCCTGACTTCCCTAATTCCCCCCATC
CTTACCACCCTCGTTAACCCTAACAAAAAAAACTCATACCCCCATTATGTAAAATCCATTGTCGCATCCACCTTTATTATCAGTCTCTTCCCCACAACAATATTCATGTGCCTAGACCAAGAAGTTATTATCTCGAACTGACACTGAGCCACAACCCAAACAACCCAGCTCTCCCTAAGCTT
Pan: AAGCTTCACCGGCGCAATTATCCTCATAATCGCCCACGGACTTACATCCTCATTATTATTCTGCCTAGCAAACTCAAATTATGAACGCACCCACAGTCGCATCATAATTCTCTCCCAAGGACTTCAAACTCTACTCCCACTAATAGCCTTTTGATGACTCCT
AGCAAGCCTCGCTAACCTCGCCCTACCCCCTACCATTAATCTCCTAGGGGAACTCTCCGTGCTAGTAACCTCATTCTCCTGATCAAATACCACTCTCCTACTCACAGGATTCAACATACTAATCACAGCCCTGTACTCCCTCTACATGTTTACCACAACACAATGAGGCTCACTCACCCACCAC
ATTAATAACATAAAGCCCTCATTCACACGAGAAAATACTCTCATATTTTTACACCTATCCCCCATCCTCCTTCTATCCCTCAATCCTGATATCATCACTGGATTCACCTCCTGTAAATATAGTTTAACCAAAACATCAGATTGTGAATCTGACAACAGAGGCTCACGACCCCTTATTTACCGAG
AAAGCTTATAAGAACTGCTAATTCATATCCCCATGCCTGACAACATGGCTTTCTCAACTTTTAAAGGATAACAGCCATCCGTTGGTCTTAGGCCCCAAAAATTTTGGTGCAACTCCAAATAAAAGTAATAACCATGTATACTACCATAACCACCTTAACCCTAACTCCCTTAATTCTCCCCATC
CTCACCACCCTCATTAACCCTAACAAAAAAAACTCATATCCCCATTATGTGAAATCCATTATCGCGTCCACCTTTATCATTAGCCTTTTCCCCACAACAATATTCATATGCCTAGACCAAGAAGCTATTATCTCAAACTGGCACTGAGCAACAACCCAAACAACCCAGCTCTCCCTAAGCTT
Gorilla: AAGCTTCACCGGCGCAGTTGTTCTTATAATTGCCCACGGACTTACATCATCATTATTATTCTGCCTAGCAAACTCAAACTACGAACGAACCCACAGCCGCATCATAATTCTCTCTCAAGGACTCCAAACCCTACTCCCACTAATAGCCCTTTGATGACTTCT
GGCAAGCCTCGCCAACCTCGCCTTACCCCCCACCATTAACCTACTAGGAGAGCTCTCCGTACTAGTAACCACATTCTCCTGATCAAACACCACCCTTTTACTTACAGGATCTAACATACTAATTACAGCCCTGTACTCCCTTTATATATTTACCACAACACAATGAGGCCCACTCACACACCAC
ATCACCAACATAAAACCCTCATTTACACGAGAAAACATCCTCATATTCATGCACCTATCCCCCATCCTCCTCCTATCCCTCAACCCCGATATTATCACCGGGTTCACCTCCTGTAAATATAGTTTAACCAAAACATCAGATTGTGAATCTGATAACAGAGGCTCACAACCCCTTATTTACCGAG
AAAGCTCGTAAGAGCTGCTAACTCATACCCCCGTGCTTGACAACATGGCTTTCTCAACTTTTAAAGGATAACAGCTATCCATTGGTCTTAGGACCCAAAAATTTTGGTGCAACTCCAAATAAAAGTAATAACTATGTACGCTACCATAACCACCTTAGCCCTAACTTCCTTAATTCCCCCTATC
CTTACCACCTTCATCAATCCTAACAAAAAAAGCTCATACCCCCATTACGTAAAATCTATCGTCGCATCCACCTTTATCATCAGCCTCTTCCCCACAACAATATTTCTATGCCTAGACCAAGAAGCTATTATCTCAAGCTGACACTGAGCAACAACCCAAACAATTCAACTCTCCCTAAGCTT
Pongo: AAGCTTCACCGGCGCAACCACCCTCATGATTGCCCATGGACTCACATCCTCCCTACTGTTCTGCCTAGCAAACTCAAACTACGAACGAACCCACAGCCGCATCATAATCCTCTCTCAAGGCCTTCAAACTCTACTCCCCCTAATAGCCCTCTGATGACTTCT
AGCAAGCCTCACTAACCTTGCCCTACCACCCACCATCAACCTTCTAGGAGAACTCTCCGTACTAATAGCCATATTCTCTTGATCTAACATCACCATCCTACTAACAGGACTCAACATACTAATCACAACCCTATACTCTCTCTATATATTCACCACAACACAACGAGGTACACCCACACACCAC
ATCAACAACATAAAACCTTCTTTCACACGCGAAAATACCCTCATGCTCATACACCTATCCCCCATCCTCCTCTTATCCCTCAACCCCAGCATCATCGCTGGGTTCGCCTACTGTAAATATAGTTTAACCAAAACATTAGATTGTGAATCTAATAATAGGGCCCCACAACCCCTTATTTACCGAG
AAAGCTCACAAGAACTGCTAACTCTCACTCCATGTGTGACAACATGGCTTTCTCAGCTTTTAAAGGATAACAGCTATCCCTTGGTCTTAGGATCCAAAAATTTTGGTGCAACTCCAAATAAAAGTAACAGCCATGTTTACCACCATAACTGCCCTCACCTTAACTTCCCTAATCCCCCCATTA
CCGCTACCCTCATTAACCCCAACAAAAAAAACCCATACCCCCACTATGTAAAAACGGCCATCGCATCCGCCTTTACTATCAGCCTTATCCCAACAACAATATTTATCTGCCTAGGACAAGAAACCATCGTCACAAACTGATGCTGAACAACCACCCAGACACTACAACTCTCACTAAGCTT
Hylobates: AAGCTTTACAGGTGCAACCGTCCTCATAATCGCCCACGGACTAACCTCTTCCCTGCTATTCTGCCTTGCAAACTCAAACTACGAACGAACTCACAGCCGCATCATAATCCTATCTCGAGGGCTCCAAGCCTTACTCCCACTGATAGCCTTCTGATGACTCGC
AGCAAGCCTCGCTAACCTCGCCCTACCCCCCACTATTAACCTCCTAGGTGAACTCTTCGTACTAATGGCCTCCTTCTCCTGGGCAAACACTACTATTACACTCACCGGGCTCAACGTACTAATCACGGCCCTATACTCCCTTTACATATTTATCATAACACAACGAGGCACACTTACACACCAC
ATTAAAAACATAAAACCCTCACTCACACGAGAAAACATATTAATACTTATGCACCTCTTCCCCCTCCTCCTCCTAACCCTCAACCCTAACATCATTACTGGCTTTACTCCCTGTAAACATAGTTTAATCAAAACATTAGATTGTGAATCTAACAATAGAGGCTCGAAACCTCTTGCTTACCGAG
AAAGCCCACAAGAACTGCTAACTCACTATCCCATGTATGACAACATGGCTTTCTCAACTTTTAAAGGATAACAGCTATCCATTGGTCTTAGGACCCAAAAATTTTGGTGCAACTCCAAATAAAAGTAATAGCAATGTACACCACCATAGCCATTCTAACGCTAACCTCCCTAATTCCCCCCATT
ACAGCCACCCTTATTAACCCCAATAAAAAGAACTTATACCCGCACTACGTAAAAATGACCATTGCCTCTACCTTTATAATCAGCCTATTTCCCACAATAATATTCATGTGCACAGACCAAGAAACCATTATTTCAAACTGACACTGAACTGCAACCCAAACGCTAGAACTCTCCCTAAGCTT
Macaca fuscata: AAGCTTTTCCGGCGCAACCATCCTTATGATCGCTCACGGACTCACCTCTTCCATATATTTCTGCCTAGCCAATTCAAACTATGAACGCACTCACAACCGTACCATACTACTGTCCCGAGGACTTCAAATCCTACTTCCACTAACAGCCTTTTGATGATTAAC
AGCAAGCCTTACTAACCTTGCCCTACCCCCCACTATCAATCTACTAGGTGAACTCTTTGTAATCGCAACCTCATTCTCCTGATCCCATATCACCATTATGCTAACAGGACTTAACATATTAATTACGGCCCTCTACTCTCTCCACATATTCACTACAACACAACGAGGAACACTCACACATCAC
ATAATCAACATAAAGCCCCCCTTCACACGAGAAAACACATTAATATTCATACACCTCGCTCCAATTATCCTTCTATCCCTCAACCCCAACATCATCCTGGGGTTTACCTCCTGTAGATATAGTTTAACTAAAACACTAGATTGTGAATCTAACCATAGAGACTCACCACCTCTTATTTACCGAG
AAAACTCGCAAGGACTGCTAACCCATGTACCCGTACCTAAAATTACGGTTTTCTCAACTTTTAAAGGATAACAGCTATCCATTGACCTTAGGAGTCAAAAACATTGGTGCAACTCCAAATAAAAGTAATAATCATGCACACCCCCATCATTATAACAACCCTTATCTCCCTAACTCTCCCAATT
TTTGCCACCCTCATCAACCCTTACAAAAAACGTCCATACCCAGATTACGTAAAAACAACCGTAATATATGCTTTCATCATCAGCCTCCCCTCAACAACTTTATTCATCTTCTCAAACCAAGAAACAACCATTTGGAGCTGACATTGAATAATGACCCAAACACTAGACCTAACGCTAAGCTT
M mulatta: AAGCTTTTCTGGCGCAACCATCCTCATGATTGCTCACGGACTCACCTCTTCCATATATTTCTGCCTAGCCAATTCAAACTATGAACGCACTCACAACCGTACCATACTACTGTCCCGGGGACTTCAAATCCTACTTCCACTAACAGCTTTCTGATGATTAAC
AGCAAGCCTTACTAACCTTGCCCTACCCCCCACTATCAACCTACTAGGTGAACTCTTTGTAATCGCGACCTCATTCTCCTGGTCCCATATCACCATTATATTAACAGGATTTAACATACTAATTACGGCCCTCTACTCCCTCCACATATTCACCACAACACAACGAGGAGCACTCACACATCAC
ATAATCAACATAAAACCCCCCTTCACACGAGAAAACATATTAATATTCATACACCTCGCTCCAATCATCCTCCTATCTCTCAACCCCAACATCATCCTGGGGTTTACTTCCTGTAGATATAGTTTAACTAAAACATTAGATTGTGAATCTAACCATAGAGACTTACCACCTCTTATTTACCGAG
AAAACTCGCGAGGACTGCTAACCCATGTATCCGTACCTAAAATTACGGTTTTCTCAACTTTTAAAGGATAACAGCTATCCATTGACCTTAGGAGTCAAAAATATTGGTGCAACTCCAAATAAAAGTAATAATCATGCACACCCCTATCATAATAACAACCCTTATCTCCCTAACTCTCCCAATT
TTTGCCACCCTCATCAACCCTTACAAAAAACGTCCATACCCAGATTACGTAAAAACAACCGTAATATATGCTTTCATCATCAGCCTCCCCTCAACAACTTTATTCATCTTCTCAAACCAAGAAACAACCATTTGAAGCTGACATTGAATAATAACCCAAACACTAGACCTAACACTAAGCTT
M fascicularis: AAGCTTCTCCGGCGCAACCACCCTTATAATCGCCCACGGGCTCACCTCTTCCATGTATTTCTGCTTGGCCAATTCAAACTATGAGCGCACTCATAACCGTACCATACTACTATCCCGAGGACTTCAAATTCTACTTCCATTGACAGCCTTCTGATGACTCAC
AGCAAGCCTTACTAACCTTGCCCTACCCCCCACTATTAATCTACTAGGCGAACTCTTTGTAATCACAACTTCATTTTCCTGATCCCATATCACCATTGTGTTAACGGGCCTTAATATACTAATCACAGCCCTCTACTCTCTCCACATGTTCATTACAGTACAACGAGGAACACTCACACACCAC
ATAATCAATATAAAACCCCCCTTCACACGAGAAAACATATTAATATTCATACACCTCGCTCCAATTATCCTTCTATCTCTCAACCCCAACATCATCCTGGGGTTTACCTCCTGTAAATATAGTTTAACTAAAACATTAGATTGTGAATCTAACTATAGAGGCCTACCACTTCTTATTTACCGAG
AAAACTCGCAAGGACTGCTAATCCATGCCTCCGTACTTAAAACTACGGTTTCCTCAACTTTTAAAGGATAACAGCTATCCATTGACCTTAGGAGTCAAAAACATTGGTGCAACTCCAAATAAAAGTAATAATCATGCACACCCCCATCATAATAACAACCCTCATCTCCCTGACCCTTCCAATT
TTTGCCACCCTCACCAACCCCTATAAAAAACGTTCATACCCAGACTACGTAAAAACAACCGTAATATATGCTTTTATTACCAGTCTCCCCTCAACAACCCTATTCATCCTCTCAAACCAAGAAACAACCATTTGGAGTTGACATTGAATAACAACCCAAACATTAGACCTAACACTAAGCTT

```
M sylvanus:          AAGCTTCTCCGGTGCAACTATCCTTATAGTTGCCCATGGACTCACGTCTTCCATATACTTCTGCTTGGCCAACTCAAACTACGAACGCACCCACAGCGGCATCATACTATCCCGAGGACTCCAAATCCTACTCCCACTAACAGCCTTCTGATGATTCAC
AGCAAGCCTTACTAATCTTGCTCTACCCTCCACTATTAATCTACTGGGCGAACCTCATTTTCCTGATCCCACATCACCATCAACTAATTACAGCCCTCTACTCTCTTCACATATTCACCACAACACACAACGAGGAGCGCTCACACACCAC
ATAATTAACATAAAACCACCTTTCACACGAGAAAACATATTAATACTCATACACCCAACACATCATTCTTGTATCTTCTTAACCCCAACACATCATTCTTGTAAATATAGTTTAATTAAAACATTAGACTGTGAATCTAACTATAGAAGCTTACCACTTCTTATTTACCGAG
AAAACTTGCAAGGACCGCTAATCCACACCTCCGTACTTAAAACTACGGTTTTCTCAACTTTTAAAGGATAACAGCTATCCATTGGTGCAACTCCAAATAAAAGTAATAATCATGTATACCCCCATCATAATAACAACTCTCATCTCCCTAACTCTTCCAATT
TTCGCTACCCTTATCAACCCCCAACAAAAAACACCTATATCCAAACTACGTAAAAACAGCCGTAATATATGCTTTCATTACCAGCCTCTCTCAACAACTTTATATATATTCTTAAACCAAGAAACAATCATCTGAAGCTGGCACTGGACATAATAACCCAAACACTACTAAGCCTAACATTAAGCTT
Saimiri sciureus:    AAGCTTCACCGGCGCAATGATCTCTAATAATCGCTCACGGGTTTACTTCGTCTATGCTATTCTGCCTAGCAAACTCAAATTACGAACGCAATTCACAGCCGAACAATAACATTTACTCGAGGGCTCCAAACACTATTCCCGCTTATAGGCCTCTGATGACTCGT
AGCAAATCTCGCTAACCTCGCCCTACCCACAGCTATTAATCTAGTAGGAGAATTACTCACAATCGTATCTTCCTCTCTTGATCGAACTTTATTATTCACAGGACTTAATTATACTCAATTACAGCACACTCTACTCACTTCATATGTATGCCTCTACACAGCGAGGTCCACTTACATACAGC
ACCAGCAATATAAAACCAATATTTACACGAGAAAATACGCTAATATTTATACATATACCAATCCTCCTCGTTACCTTGAGCCCCAAGGTAATTATAGGACCCTCACCTTGTAATTATAGTTTAGCTAAAACATTAGATTGTGAATCTAATAATAGAAGAATATAACTTCTTAATTACCGAG
AAAGTGCGCAAGAACTGCTAATTCATGCTCCCAAGACTAACAACTTGGCTTCCATTGGTCTTAGGAGCCAAAACATTGGTGCAACTCCAAATAAAAGTAATAATACACTTCTCCATCACTCTATAATTAGCCTACTAGCGCCAATCCTA
GCTACCCTCATTAACCCTAACAAAAGCACACTATACCCGTACTACGTAAAACTAGCCATCATCTACGCCCTCATTACCAGTACGCCTCATTATCTTATCCTTACAGGCCAAGAATCAATAATTTCAAACTGACACTGAATAACTATCCATAACCATATCCCTAAGCTT
```

E.2 Results

Here we put the effective sample size output of our processing of the results.

E.2.1 5 taxa

Base HMC

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	837.433576449256	1345.68707530281	2726.75538782086	574.269082804174	1109.24359195978	NA	1406.64770629525	1312.34380864157	NA	6541.78363703753
Effective Sample Size / time	0.159270395386449	0.255934462836647	0.518598036851793	0.109219484922309	0.210965586333909	NA	0.267528485424495	0.249592950609339	NA	1.24417326425018

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	285.662619121297	1702.86476415458	3398.73311742712	8.86498581413694	13.7427937703116	538.154997534773
Effective Sample Size / time	0.0543297994898758	0.323865619798189	0.646400601371682	0.00168602004435 92	0.00261372395264186	0.102350994331967

Inverted Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	2761.33417022253	2228.49918659539	5067.86653303925	2847.57657836106	2018.90871048583	NA	4328.92138883957	3399.57120887336	NA	4838.83513654021
Effective Sample Size / time	0.509848976100393	0.411467051245976	0.935723966605346	0.525772656747244	0.372768551514506	NA	0.799286142734793	0.627692192678286	NA	0.893435921840932

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	251.218996317148	1397.0022523923	2419.60182546715	16.1385121735835	7.64260891428671	739.075107983382
Effective Sample Size / time	0.0463847329419548	0.257940591063904	0.446751981918092	0.00297979288828086	0.00141112088920799	0.136461820206371

1000 Alignment

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1057.62362273103	706.704403924221	859.040450710534	273.024408203681	1130.14563883574	NA	1096.02976298244	626.532074712111	NA	2044.97648599975
Effective Sample Size / time	0.0166957274031337	0.0111560897742103	0.0135608782605414	0.00430998419078412	0.0178405654963709	NA	0.0173020096707218	0.00989048325310853	NA	0.0322821552225797

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	227.340048213531	679.54640905349	1140.05825061046	9.11563042167249	4.10793174507917	149.907347882963
Effective Sample Size / time	0.00358880738970944	0.0107273715899411	0.0179970467440347	0.000143900039065846	6.484812473214 15e−05	0.00236644885967831

1000 Alignment Inverted Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	529.600716695151	246.272235352824	366.000684362169	137.961273721093	237.486146264071	NA	372.882747386529	215.139194312904	NA	800.486474716049
Effective Sample Size / time	0.00821639380518557	0.00382074571493999	0.0056782509178854	0.00214037503920145	0.00368443553694189	NA	0.00578502143049555	0.0033377378233849	NA	0.0124190015320121

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	175.010231115372	269.68961690163	430.394333000242	11.4638718295219	1.92351722268019	87.7619424420151
Effective Sample Size / time	0.00271516433692469	0.00418405041341537	0.00667727444463736	0.000177854150332927	2.984205741111394e−05	0.00136156666235536

Diagonal Fisher

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1050.67973828561	2144.86101736775	3101.42737161499	1917.23907364395	1759.59909066879	NA	2336.43243491307	1815.07885224303	NA	80.0683871104358
Effective Sample Size / time	0.21963996415639	0.44837373350682	0.648339709916427	0.40079037033825	0.367836427334863	NA	0.488420893216686	0.379434226739897	NA	0.0167379430992848

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	240.947826352185	126.268693618343	195.120019746183	163.434233856164	799.251091555751	719.981638779096
Effective Sample Size / time	0.0503690801441607	0.0263959133595361	0.0407889793450987	0.0341652055884007	0.167079914748999	0.150508985347647

Stabilised Fisher

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1556.30072169558	2216.23824218071	3063.87738041825	1218.46697405489	1532.97186029404	NA	1570.74165324425	1154.60018856277	NA	283.8315992563
Effective Sample Size / time	0.29653786922447	0.422282504189832	0.583791845167566	0.2321669553767	0.292092783030647	NA	0.299289447381116	0.219997764538431	NA	0.0540813330538956

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	1317.13999763469	624.937610346833	567.147893309979	642.681473857783	1375.45756638983	1278.23646416113
Effective Sample Size / time	0.250968134194127	0.119075744672649	0.108064479745313	0.12245666418518	0.262079976099695	0.243555446684104

Projecting Boundary 2 with Parachuting

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	5204.40420259926	3162.87629556644	4867.58913436652	1335.50831721714	629.690259727069	NA	1847.01374625384	422.146255944592	NA	79.6117623471034
Effective Sample Size / time	1.0828850453586	0.658102504620516	1.01280363233951	0.277880412121755	0.131020216516977	NA	0.384309805028429	0.0878363496993979	NA	0.016564890720299

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	87.6266061285565	92.9130044536958	69.5570766236059	0.839523467554749	1.1815292614671	2.3392752213355
Effective Sample Size / time	0.0182325464468635	0.0193324920827622	0.0144728032532421	0.000174680400071 27	0.00024584185203319	0.000486735091193915

Projecting Boundary 2 using Ratios

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	2870.68048566374	3867.5283078048	7574.45959637121	4483.56974682887	3430.75144052833	NA	8143.32592454528	6503.84619869041	NA	10496.6843488381
Effective Sample Size / time	0.632450055222035	0.852069223328552	1.66875673343856	0.987791552607476	0.755841323634294	NA	1.79408574252348	1.43288600318273	NA	2.31256269348834

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	766.728738085789	3776.20352324489	4703.62218539347	63.7201933432461	24.2235110584715	2185.81304201494
Effective Sample Size / time	0.168920796014872	0.831949127996973	1.03627210543004	0.0140384274738876	0.00533676979486614	0.481564418621726

Projecting Boundary 3

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1983.71783445492	741.048634589212	359.441560707997	3390.67952123542	1508.77888107417	NA	839.356949934695	1740.03412578761	NA	8913.44371942063
Effective Sample Size / time	0.455525830476544	0.170168755269328	0.0825394179582433	0.778609779054078	0.346464472350542	NA	0.192743526848577	0.39956816258837	NA	2.0468152184611

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	863.879374770633	3132.31856777985	3957.12552669129	7.49642796199099	15.8478217839073	40.2015674851922
Effective Sample Size / time	0.198374669415664	0.719281740639835	0.908684118546172	0.00172142252979838	0.00363917289746114	0.00923158127485557

Reflecting Boundary 2

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	2727.04024419512	3681.28304294173	7252.15115959254	4425.55139381485	3258.16340624445	NA	8448.26172050785	5738.55491751594	NA	11775.2737327512
Effective Sample Size / time	0.623897498430496	0.842210996480482	1.65916105431184	1.01248613754566	0.745408874313719	NA	1.93280952297301	1.31287760247225	NA	2.69396971344151

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	742.21868412334	3743.04431154019	4565.29388709965	69.9678909325688	6.94060621814797	2206.39052570958
Effective Sample Size / time	0.169806214374222	0.856340857988942	1.04445669323159	0.0160073883090682	0.0015878852049621	0.504782257057318

Reflecting Boundary 2 with Parachuting

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	2517.80653827699	1485.29556379664	2735.12716977845	1885.39049133345	1030.98695714039	NA	2576.18360086019	2451.09160212398	NA	4771.90128055743
Effective Sample Size / time	0.503561307655398	0.297059112759328	0.54702543395569	0.377078098266689	0.206197391428077	NA	0.515236720172039	0.490218320424795	NA	0.954380256111486

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	374.479995929677	1532.96992976339	1888.98029367174	33.4394765083421	4.41169176722769	103.418989249445
Effective Sample Size / time	0.0748959991859353	0.306593985952678	0.377796058734348	0.00668789530166843	0.000882338353445537	0.0206837978498889

Reflecting Boundary 2 using Ratios

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	3133.19446907494	3868.4974116784 9	7512.00947888186	4107.81228301955	3641.46796600894	NA	8394.47331200556	6197.02182634557	NA	11331.2233697838
Effective Sample Size / time	0.697381168441912	0.861043663806599	1.67201046709241	0.914309966376904	0.810511830671297	NA	1.86842778659193	1.37932272151395	NA	2.52208468752633

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	795.401422980578	3730.48871060611	4633.94411432125	53.611584205629	32.1559379072257	1998.51567066659
Effective Sample Size / time	0.177039114301235	0.830325918655777	1.03141550670935	0.011932776469272	0.00715721471214888	0.444826265115433

Reflecting Boundary 3

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1161.15645851908	622.412285983523	360.375129646541	2106.59893863199	564.103896261877	NA	9320.53616086623	5528.95470918162	NA	54.5385962937642
Effective Sample Size / time	0.259419124934703	0.139055895003491	0.0805130093324111	0.470644631254322	0.126028958515334	NA	2.08234240703182	1.23524834956	NA	0.0121847101843105

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	132.969365453851	90.7124297043 77	141.847862877 84	201.864544645929	2981.8819980678 6	544.2186181139 95
Effective Sample Size / time	0.0297117451817477	0.0202664670742333	0.0316901741064745	0.0450994556756 09	0.6661955095904 08	0.1216019391590 94

Adaptive Stepsize

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1608.53860650 76	1134.4263672261	2744.1563777807 7	1167.2744178326 3	787.7670605733 95	NA	1383.6133688280 2	1102.2504543751 7	NA	3815.8072776943 6
Effective Sample Size / time	0.349446158812809	0.246447884358725	0.596152869143233	0.253583942556211	0.171138058012 99	NA	0.300582388920 46	0.239457844389174	NA	0.828963128746614

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	236.328332532186	1059.93840737615	1942.15001522308	22.6040103584466	11.1602648005548	73.5547846027552
Effective Sample Size / time	0.0513410294834713	0.230265785065577	0.421921401147756	0.004910596837133 99	0.002424506101 4451	0.015978372106923

Fixed Black Box

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	2177.33506918083	2183.99227559862	1344.45078882791	1178.31242601419	961.621323838196	NA	2116.26377473854	2267.32859518207	NA	5122.41962525813
Effective Sample Size / time	0.678155231433961	0.680228692438133	0.418744155868902	0.366990514503237	0.299507659914131	NA	0.659133897325718	0.706184716338135	NA	1.595434582227 95

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	156.522531563552	1023.65914979916	1248.88209284 68	55.2939163864305	15.0094044003 21	58.4319497482515
Effective Sample Size / time	0.0487506838610031	0.318830031017 1	0.388978222367443	0.017221905436 0051	0.004674845988 93399	0.018192808390173

1 Step Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	837.433576449256	1345.68707530281	2726.75538782086	574.269082804174	1109.24359195978	NA	1406.64770629525	1312.34380864157	NA	6541.78363703753
Effective Sample Size / time	0.159270395386449	0.255934462836647	0.518598036851793	0.109219484922309	0.210966586333909	NA	0.267528485424495	0.249592950609339	NA	1.24417326425018

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	285.662619121297	1702.86476415458	3398.73311742712	8.86498581413694	13.7427937703116	538.154997534773
Effective Sample Size / time	0.0543297994898758	0.323865619798189	0.646400601371682	0.001686020044359 2	0.002613723952641 86	0.102350994331967

2 Step Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	361.211116690085	507.900694816999	962.961958861727	237.882305057445	282.400712335908	NA	458.324475505 09	354.662148025072	NA	3259.758514683 52
Effective Sample Size / time	0.0394679507909407	0.0554960761269178	0.105218619953101	0.025992353712906 9	0.030856684367 18	NA	0.050079100586 04	0.038752373775 0099	NA	0.356179482588474

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	219.867664940367	1018.56228773272	1721.25990595841	6.05847504018304	17.7838960335924	18.5095025080077
Effective Sample Size / time	0.0240239731819518	0.111293823451825	0.188074503047682	0.000661982933817747	0.001943161393328 9	0.002022451968934 36

3 Step Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	771.953924464232	1183.33275993439	3256.08192995698	704.975789069798	1872.0871799403	NA	1489.60376020099	1565.58728727117	NA	7767.17428252299
Effective Sample Size / time	0.0584454404235048	0.0895913630709999	0.246521374424555	0.0533744556175273	0.141737681842818	NA	0.112779461393216	0.118532253838219	NA	0.588060902861839

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	325.478524464478	2079.34508750753	4042.97771782244	15.2415027422531	20.0709908044085	655.962744783673
Effective Sample Size / time	0.0246423201021215	0.157429395175592	0.306098078981264	0.00115395014165553	0.00151959574285981	0.0496636266819953

4 Step Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	1098.26592712274	1832.41135608158	3982.77866439791	759.545470490141	1618.76489638086	NA	2105.47165870585	1954.89809166731	NA	7858.27005154002
Effective Sample Size / time	0.0624092910456427	0.104127325461728	0.226322593337042	0.0431613994020606	0.0919886012452804	NA	0.119644059314037	0.111087670986084	NA	0.446548554999477

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	340.530063969962	1981.35776365518	3942.66661183758	27.1117865411451	7.28795544208517	1255.89945567786
Effective Sample Size / time	0.0193507231238144	0.112591249790689	0.224043213940672	0.00154063541008385	0.000414140256089306	0.0713668636327501

8 Step Leapfrog

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	piA
Effective Sample Size	825.598227870956	1112.74823910085	3148.21129089329	590.845306229372	1844.02939678033	NA	1547.79063578065	1628.52147565824	NA	8355.68460237324
Effective Sample Size / time	0.0243050648189429	0.0327585708950164	0.0926812545203878	0.0173940943440562	0.0542870036583793	NA	0.0455659308109387	0.0479425932477907	NA	0.245985818539169

	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	359.852611888987	2105.88849464034	4302.10103594331	16.3324675656213	11.2908338655962	859.137998517791
Effective Sample Size / time	0.0105938224695351	0.0619959620016284	0.126651004091771	0.000480817023868122	0.000332394668131634	0.0252924534446258

E.2.2 16 taxa

Base HMC

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	32.7289463273649	72.5354167792284	91.4049711137737	81.2828524061704	36.9298089122199	32.6282425201278	36.2100053552119	57.1793297293358	207.829050126225	227.428532351895
Effective Sample Size / time	0.00116822818429148	0.00258908176857419	0.00326261231789726	0.00290131304963822	0.00131817392409246	0.00116463365898342	0.00129248123010716	0.00204096104654781	0.0074182575706381	0.00811784219229162

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	228.685942500444	4.80907791070642	299.972733664828	42.2467814589489	20.2271177681141	104.821395113727	NA	32.9410818356655	64.3906484282276	151.206152261803
Effective Sample Size / time	0.00816272423524088	0.00017165539946037	0.010707237516327	0.00150795813291067	0.000721987467220623	0.00374149863743774	NA	0.001175800015673468	0.00229836211487703	0.00539715782303735

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	65.0244392043574	70.2462220854164	92.3203396988965	3.93148541436552	3.47355888581262	8.94491990625094	23.2542880623074	240.321222739688	51.6344272173024	337.071702235534
Effective Sample Size / time	0.00232098466557613	0.00250737117105319	0.00329528551701156	0.00014033058141418 2	0.000123985335476858	0.000319280292014019	0.000830039392294962	0.00857803434549179	0.00184304109737 15	0.0120314494313377

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	1.74798167001361	1.0528812918249	0.582525676339205	1.21431846873252	5.34735776795965	2.60264088506495	1.18700934002555
Effective Sample Size / time	6.23925204346533e−05	3.75815826003128e−05	2.07926923881422e−05	4.33439269840744e−05	0.000190868779994831	9.289875711580 81e−05	4.236915396435e−05

Inverted Leapfrog

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	769.789449417596	1122.61995472667	1130.31433049845	1192.95943662342	984.030053554402	1428.73018542671	1456.06486884088	1214.3408650968	1211.80756862026	733.883475666894
Effective Sample Size / time	0.0196339466011258	0.028633102546596	0.0288293522654592	0.0304271536764946	0.0250982830954716	0.0364406295634576	0.0371378172359029	0.0309724999724457	0.0309078867099712	0.0187181429722035

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	1700.07359615866	1227.2539414873	1104.75050854835	1076.37083645305	1110.6808544555	953.627914863963	NA	879.470613425165	1080.58207471236	1057.2314557838
Effective Sample Size / time	0.0433614077592475	0.0313018558144869	0.0281773314882601	0.027453490745966	0.0283285885559626	0.0243228581165248	NA	0.0224314311846092	0.0275609009308816	0.0269653292385265

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	1370.35381589756	1099.0833098366	1328.26545534418	1420.11687580052	1603.9454291856	1502.02421074942	1301.0685851633	1343.84525062642	882.362339583072	538.848513841587
Effective Sample Size / time	0.0349517048672694	0.0280327861493123	0.0338782157148016	0.0362209418795232	0.04090960058179	0.0383100381146611	0.0331845430524221	0.0342755878389057	0.0225051864134089	0.0137436580286545

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	12390.3562706192	1225.11931026622	5475.09906506019	9324.47199628072	123.322481542423	2139.64123014414	174.157729436397
Effective Sample Size / time	0.316023548478502	0.0312474107510484	0.1396457209963	0.237826311333806	0.00314541465741593	0.0545728467569939	0.00444199847440696

Stabilised and Normalisesd Fisher

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	1115.063185186	1472.55015134705	2018.75297043981	2009.45310891147	1978.99452616118	2686.99851610631	2456.86270479795	2189.44253336665	2127.55086033454	2011.55815139569
Effective Sample Size / time	0.0369796372290532	0.0488352329462998	0.0669493473497514	0.0666409293962511	0.0656308096509482	0.0891108519057071	0.0814786935413562	0.0726100472990257	0.0705574895187733	0.0667107404244086

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	1942.65430623517	1783.1533350477	1506.76231602294	1362.21671681937	984.465486655153	994.553474746757	NA	758.416955322669	1593.89410665311	2005.0181796511
Effective Sample Size / time	0.0644256329689975	0.059135988282889	0.0499698354112928	0.0451761664133225	0.0326485324281944	0.0329830875861035	NA	0.025151923450434	0.0528594492479149	0.0664938506680107

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	1637.7529485448	2073.81151327722	1670.51122366392	1592.35682296405	1596.67081083455	1569.93365356514	1778.0501809506	700.391462362994	1065.27766512809	646.947498821167
Effective Sample Size / time	0.0543139713628844	0.0687752931504355	0.0554003574495637	0.0528084672103992	0.0529515350729034	0.0520648316201355	0.0589667487368778	0.0232275825626772	0.0353285644508 82	0.0214551822089431

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	234.887604804433	221.237531177744	254.548622855344	269.452037285365	54.9472842171186	58.0891645954121	25.5769414400778
Effective Sample Size / time	0.00778974548766961	0.00733705834171147	0.00844177787895985	0.00893603046161013	0.00182225605155441	0.00192645247571 22	0.000848226386135278

Adaptive Stepsize

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	18.5050529660239	43.5630414084242	24.9884684358472	21.3121658417112	29.7388590860066	11.9601141601113	18.5342907836233	21.8598253889358	11.0976189031161	16.2559819211556
Effective Sample Size / time	0.000864749969976438	0.00203572174687547	0.00116772307376427	0.000995927696375167	0.00138971109939403	0.0005589018512868	0.000866116267167682	0.00102152008878222	0.000518597035679903	0.000759649805059557

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	18.4297894972236	20.8550783753014	15.7823952358909	23.7167483965369	17.7858168996451	16.3125676844388	NA	24.2816316102512	10.0854046672738	16.3510927143952
Effective Sample Size / time	0.000861232872105664	0.000974567780595393	0.000737518872896553	0.00110829498848231	0.000831139724820752	0.000762294084824122	NA	0.00113469225105684	0.000471295780630191	0.000764094377887821

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	21.1160091992284	13.255996309138	23.6713312448373	21.8004038149622	13.7285431612926	22.0171960213346	21.7792536656589	19.7156099127564	25.5308001700248	24.6575134781036
Effective Sample Size / time	0.000986761202714809	0.000619459043504585	0.00110617262327532	0.00101874329022866	0.000641541383769346	0.00102887409365278	0.00101775493364256	0.000921319874708066	0.001193066494921 1	0.00115225739040212

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	13.4983506975692	7.24707734949853	9.07853653247067	6.48585282875603	23.5015020334466	3.0439634347365	2.79637261485006
Effective Sample Size / time	0.000630784379914286	0.0003386593884331	0.000424244351437317	0.000303087002735744	0.00109823642305364	0.000142245866230738	0.000130675828876274

Fixed Black Box

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	297.023138493585	366.863296065665	1229.00656372234	1678.24703504433	1514.34816483921	1388.32093662795	1381.09285449967	1398.68551151536	1456.3411737128	1140.29808482102
Effective Sample Size / time	0.0094777122224445	0.0117062420211512	0.0392163741502829	0.0535511897052574	0.0483213401613253	0.0442999369560514	0.0440692960616096	0.0446306602074082	0.0464703949064395	0.0363857750293472

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	1299.01801436495	1182.72088025395	1319.82092198328	1127.80150647784	1121.87778467189	980.358826370631	NA	213.474746181604	1000.9047594504	1118.68801802611
Effective Sample Size / time	0.0414503697400941	0.0377394441368252	0.0421141697820591	0.0359870216732868	0.0357980016163062	0.031282272748845	NA	0.00681176632005834	0.0319378729894836	0.0356962193427811

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	1163.34921192544	1581.29143438918	1466.81489477932	1742.36395034116	1091.61969883989	1120.61418772752	1155.94630091334	778.034601377082	915.293064852842	1189.96988490772
Effective Sample Size / time	0.0371213135136778	0.0504574331513934	0.046804600903555	0.0555970965489337	0.0348324962641907	0.0357576814975956	0.036885094003855	0.0248263084430224	0.0292060891682411	0.037970752648196

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	298.196498616634	201.358625183869	269.330586016732	340.988237853388	73.248219483445	58.1823636549166	102.010512321567
Effective Sample Size / time	0.00951515297415143	0.00642515297858182	0.0085940704818997	0.010880594710579	0.00233727765652014	0.00185654121742715	0.00325505374548057

1 Step Leapfrog

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	32.7289463273649	72.5354167792284	91.4049711137737	81.2828524061704	36.9298089122199	32.6282425201278	36.2100053552119	57.1793297293358	207.829050126225	227.428532351895
Effective Sample Size / time	0.00116822818429148	0.00258908176857419	0.00326261231789726	0.00290131304963822	0.00131817392409246	0.00116463365898342	0.00129248123010716	0.00204096104654781	0.0074182575706381	0.00811784219229162

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	228.685942500444	4.80907791070642	299.972733664828	42.2467814589489	20.2271177681141	104.821395113727	NA	32.9410818356655	64.3906484282276	151.206152261803
Effective Sample Size / time	0.00816272423524088	0.00017165539946037	0.010707237516327	0.00150795813291067	0.000721987467220623	0.00374149863743774	NA	0.00117580015673468	0.00229836211487703	0.00539715782303735

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	65.0244392043574	70.2462220854164	92.3203396988965	3.93148541436552	3.47355888581262	8.94491990625094	23.2542880623074	240.321222739688	51.6344272173024	337.071702235534
Effective Sample Size / time	0.00232098466557613	0.00250737117105319	0.00329528551701156	0.00014033058141418 2	0.000123985335476858	0.000319280292014019	0.000830039392294962	0.00857803434549179	0.00184030410973715	0.0120314494313377

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	1.74798167001361	1.0528812918249	0.582525676339205	1.21431846873252	5.34735776795965	2.60264088506495	1.18700934002555
Effective Sample Size / time	6.23925204346533e−05	3.75815826003128e−05	2.07926923881422e−05	4.33439269840744e−05	0.000190868779994831	9.28987571158081e−05	4.236915396435e−05

2 Step Leapfrog

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	34.465029565144	60.8872876610094	68.4225339607576	397.139256639653	291.132733102617	44.4441553616088	12.7361341853428	16.1132165488978	1645.94077432061	1397.03378107213
Effective Sample Size / time	0.000878180390363151	0.00155142829473663	0.00174342887098705	0.0101192400478037	0.00741815865035889	0.00113245182717879	0.000324520925463338	0.000410570104754179	0.0419391171266927	0.0355968843402123

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	298.200868270917	85.9591444317582	325.595097448912	244.671030377604	220.714759013374	434.465624360661	NA	51.0862676284305	43.3430405023471	34.714659883947
Effective Sample Size / time	0.00759825707997157	0.00219026752522238	0.00829627113006134	0.00623429904684783	0.00562388530272146	0.0110703282838986	NA	0.00130169505188829	0.00110439505516548	0.000884541053724794

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	92.8192371261282	14.20507880337	38.7486653055539	16.6436274851658	43.3343664935887	8.17039858219027	76.614349905645	370.8358007275 25	264.849319544991	362.645619510522
Effective Sample Size / time	0.00236506496356 15	0.000361950121792407	0.000987328850531346	0.000424085151424437	0.00110417403854389	0.00020818446717907	0.00195215905967488	0.00944901926249571	0.00674844855089461	0.00924033072721686

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	11.9513297747498	21.2874221824693	4.9963865080008	12.3299998308049	30.2869607201256	13.1375283375987	20.2216602479393
Effective Sample Size / time	0.000304523848648109	0.00054241058133118	0.000127309586249096	0.00031417248733611	0.000771721809721367	0.00033474858166341	0.000515254613571725

3 Step Leapfrog

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	287.741762234803	410.470834721481	434.686334333342	514.959005238312	129.816693362533	51.9938151120136	122.49986570428	142.353415808307	182.039481388134	229.633259787131
Effective Sample Size / time	0.00823315619859338	0.0117448036426191	0.0124376818302794	0.0147345240853136	0.00371444556861534	0.00148769924064299	0.00350508914944034	0.0040731588583124	0.00520869641215454	0.00657049738465441

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	39.5564436954481	56.4896931917911	30.6429849870444	141.435838831432	32.896658888267	68.4829083909089	NA	196.992915580329	327.624528807437	402.63775480173
Effective Sample Size / time	0.00113182868234376	0.00161633981819821	0.00087678785251761 8	0.00404690422458743	0.00094127223290560 8	0.00195950173286357	NA	0.00563655908475908	0.00937432195873281	0.0115206756954122

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	281.862926006823	126.091885739614	264.370786257221	8.70372053398112	282.255927967937	270.355327483571	19.1112132149391	41.9027744870882	20.9310205302165	47.7538349094536
Effective Sample Size / time	0.00806494503398874	0.00360786763313937	0.007564442369072 9	0.00024903959059893 4	0.00807619000068067	0.00773567807122048	0.00054682922048283 8	0.00119896425470944	0.00059889937450446 9	0.00136638067007646

	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	4.0466523308434	4.95096844617099	3.32634372746024	4.70237182632129	6.79070513417099	9.36742662925616	6.36919160108351
Effective Sample Size / time	0.000115786879396563	0.000141662080036788	9.51766864347057e−05	0.000134548983954493	0.000194302473280 18	0.00026803021576893 9	0.000182241704864 86

E.2.3 32 taxa

Base HMC

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	3.89354180130485	2.2265164917109	20.2227133651373	9.87806150531131	9.81462181392696	3.55751648297122	2.96450019648838	2.85722101766513	108.708027103705	72.5062056018182
Effective Sample Size / time	4.09367220771145e−11	2.34096078770068e−11	2.12621730784124e−10	1.03857998485608e−10	1.0319099318622e−10	3.74037498452797e−11	3.11687730180591e−11	3.00408390822589e−11	1.14295685527363e−09	7.62330684774728e−10

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	2.17256710568741	1.99191995075884	1.9291981530969	1.15681369310864	7.11324631362992	4.20379928209122	1.69553304762704	2.21020388264985	4.95668086769654	6.99748959985572
Effective Sample Size / time	2.28423837056535e−11	2.09430584247861e−11	2.02836010643428e−11	1.21627461746829e−11	7.47887148173262e−11	4.41987711089345e−11	1.78268447304225e−11	2.32380970065543e−11	5.21145726592282e−11	7.35716480277142e−11

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	2.79414075525909	5.6230665223632	9.9724911789696	1.92595056236721	1.79941996499244	9.87446008563784	8.1588997273861	4.56522842915	7.1968369297761	5.09578892962819
Effective Sample Size / time	2.93776128213255e−11	5.91209554678378e−11	1.04850832646291e−10	2.02494558757444e−11	1.89191124087123e−11	1.03820133137352e−10	8.5782721141742 1e−11	4.79988393498239e−11	7.56675870616885e−11	5.3577155664777e−11

	Edge31	Edge32	Edge33	Edge34	Edge35	Edge36	Edge37	Edge38	Edge39	Edge40
Effective Sample Size	3.42416462035932	4.11459666438093	NA	3.58733891832389	3.83259968496747	1.66863816921583	0.899959933903361	1.19201055167077	3.53380025412826	4.2637844376554
Effective Sample Size / time	3.60016875542368e−11	4.32608942461426e−11	NA	3.77173031111741e−11	4.02959762968949e−11	1.75440718159392e−11	9.46218419496446e−12	1.25328061587475e−11	3.71543972716061e−11	4.48294553977895e−11

	Edge41	Edge42	Edge43	Edge44	Edge45	Edge46	Edge47	Edge48	Edge49	Edge50
Effective Sample Size	8.34666402776194	2.72950143353414	4.60445416951062	3.35718183396156	3.43115582428874	1.65437056712789	1.62737837487181	3.61368714854504	1.07198219715282	6.81615870498315
Effective Sample Size / time	8.77568761329415e−11	2.86979946012718e−11	4.84112590215155e−11	3.52974301324218e−11	3.60751931146875e−11	1.7394062161186e−11	1.71102660883485e−11	3.79943285632767e−11	1.12708273124878e−11	7.1665133900935e−11

	Edge51	Edge52	Edge53	Edge54	Edge55	Edge56	Edge57	Edge58	Edge59	Edge60
Effective Sample Size	1.61231425093867	1.87736157333634	2.19771694105951	3.57797208930273	2.59853396749363	4.05641949147608	1.97667024279383	2.47094618556109	2.24555753753833	1.98508604275265
Effective Sample Size / time	1.69518817980913e−11	1.97385909508325e−11	2.31068092270561e−11	3.76188202141227e−11	2.73210018702197e−11	4.26492190979163e−11	2.0782722902894e−11	2.59795431583452e−11	2.36098055481436e−11	2.08712066746248e−11

	Edge61	Edge62	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	1.73457959705123	3.65916341885131	4.81554662274252	4.81554662274252	4.81554662274252	4.81554662274252	1.94748677752418	6.17447745686817	6.33766971532883
Effective Sample Size / time	1.82373803875234e−11	3.84724663446683e−11	5.06306863531123e−11	1.56408586813067e−11	6.28924570486102e−11	6.20772429527851e−11	2.04758877723221e−11	6.49184933724128e−11	6.66342977661147e−11

Inverted Leapfrog

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	1.78117427874994	3.69421942427302	5.48317418795969	8.17617558899076	5.46859911719657	4.01659248855659	3.11785623062959	3.71618354269744	15.5071479283806	3.69675360908213
Effective Sample Size / time	2.27002112514917e−11	4.70810534043956e−11	6.98804232019151e−11	1.04201433466487e−10	6.96946709208073e−11	5.1189543375497e−11	3.97355811452337e−11	4.7360975551338 5e−11	1.97631157198959e−10	4.71133503734253e−11

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	7.01901521730278	10.2747565244883	5.69934277457848	4.9403660588557	2.01021065247555	3.70672254666894	1.03983525290804	3.18908190294281	14.7275170958519	21.5643701921153
Effective Sample Size / time	8.94539799452 03e−11	1.30946840208 82e−10	7.26353880813915e−11	6.29625941344892e−11	2.56191699013415e−11	4.72403996980608e−11	1.32522011963517e−11	4.0643318151841 7e−11	1.87695136446965e−10	2.74827547594018e−10

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	19.735548593474	17.4753945205588	30.4175772497104	21.4714227248021	29.5594081423319	7.14284382917996	3.61976130652986	1.64910853913334	3.98878949936118	5.94617993076156
Effective Sample Size / time	2.51520094120355e−10	2.2271551529 3795e−10	3.87657422165107e−10	2.736429767362e−10	3.76720468797092e−10	9.10321161681018e−11	4.61321204322979e−11	2.10170967892251e−11	5.08352076231292e−11	7.57812091596107e−11

	Edge31	Edge32	Edge33	Edge34	Edge35	Edge36	Edge37	Edge38	Edge39	Edge40
Effective Sample Size	4.20544008616142	3.2109099758482	NA	3.40897294197489	6.75231303898971	4.7572237990705	3.58441528627935	5.53172919849494	5.5729331965497	7.46427615631356
Effective Sample Size / time	5.35963153635671e−11	4.0921506463944e−11	NA	4.34457239006163e−11	8.60549886947854e−11	6.06285339384723e−11	4.56816523696504e−11	7.04992335056671e−11	7.10243586837073e−11	9.512861689576 33e−11

	Edge41	Edge42	Edge43	Edge44	Edge45	Edge46	Edge47	Edge48	Edge49	Edge50
Effective Sample Size	2.97030950791223	11.2538523569633	3.78916470285986	6.34291205683038	4.03859688193954	1.20633360856261	4.92857631575121	6.275253219049	2.86497520571238	2.95423632010359
Effective Sample Size / time	3.78551689839375e−11	1.4342494666455 66e−10	4.82910854079824e−11	8.08373696821713e−11	5.14699788074 55e−11	1.53741428229044e−11	6.28123395174842e−11	7.99750902696 56e−11	3.65127338609449e−11	3.76503239201535e−11

	Edge51	Edge52	Edge53	Edge54	Edge55	Edge56	Edge57	Edge58	Edge59	Edge60
Effective Sample Size	28.1620171657125	4.50587471080095	9.9166134712239	15.3987659366236	5.82638695039624	12.235793739132	5.57160194089095	4.18433427826303	8.98419790417398	2.70164468948057
Effective Sample Size / time	3.58911391522266e−10	5.74252104514561e−11	1.263824788971e−10	1.96249880735397e−10	7.42545051233 01e−11	1.55939318247347e−10	7.10073924695 15e−11	5.33273319723 05e−11	1.14499290037519e−10	3.44311648272402e−11

	Edge61	Edge62	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	4.11489142609445	3.01159335854708	2.20717963916518	2.20717963916518	2.20717963916518	2.20717963916518	4.6596853472857	2.51674091235492	2.76652501847764
Effective Sample Size / time	5.24423161527193e−11	3.83813118447832e−11	2.81294450951786e−11	5.34061945198303e−11	1.14247200326155e−10	4.18235589379417e−11	5.93854531871533e−11	3.20746549382152e−11	3.52580334789329e−11

Parallelised HMC

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	9.78191768558831	4.38334399521145	4.1800748540065	6.52716157080784	4.63919475827711	1.34460936744353	5.61755423570077	4.63379020536532	4.4122013682825	5.07868650588706
Effective Sample Size / time	1.410364109002 88e−10	6.319937712589 44e−11	6.026862765081 27e−11	9.410909710161 37e−11	6.688825230443 36e−11	1.938667706502 35e−11	8.099431143324 91e−11	6.681032906180 37e−11	6.361544485992 85e−11	7.322487674715 65e−11

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	6.10665558306311	2.97849301856301	4.11389246915051	3.20899941599757	3.48024883515801	5.64068741646075	3.65402501322187	2.75190481478846	6.29240542031746	6.17165288766156
Effective Sample Size / time	8.804621074539 61e−11	4.294413209473 12e−11	5.931440514302 55e−11	4.626759033969 49e−11	5.017848447792 1e−11	8.132784734021 16e−11	5.268400223444 48e−11	3.967716665504 16e−11	9.072436560354 45e−11	8.898334667859 49e−11

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	3.19474527619181	9.79978562606632	6.60292903441123	3.86335115437165	7.5944629674646	4.68265406461609	25.7776793986248	6.91795497156711	13.9773678217338	5.82618195625796
Effective Sample Size / time	4.606207310030 57e−11	1.412940321843 96e−10	9.520151798196 77e−11	5.570208198161 62e−11	1.094975273840 63e−10	6.751485178965 48e−11	3.716644833445 474e−10	9.974358518648 16e−11	2.015267205033 69e−10	8.400232130078 71e−11

	Edge31	Edge32	Edge33	Edge34	Edge35	Edge36	Edge37	Edge38	Edge39	Edge40
Effective Sample Size	11.6032950587152	18.4037052468459	NA	2.5932907494162	3.24882431283139	3.10790311478641	4.55408531690997	2.81047216452656	3.12488722627392	3.23532514250233
Effective Sample Size / time	1.672971642477 7e−10	2.653459800736 23e−10	NA	3.739025735795 3e−11	4.684178864052 42e−11	4.480997640995 16e−11	6.566113809943 12e−11	4.052159502466 21e−11	4.505485458246 56e−11	4.664715660675 16e−11

	Edge41	Edge42	Edge43	Edge44	Edge45	Edge46	Edge47	Edge48	Edge49	Edge50
Effective Sample Size	4.65977937462575	3.98702607440997	1.8122393941149	5.68002112860427	6.35207680467068	2.43477253000721	2.09509309468153	3.80377069153741	8.43741683609641	6.24744663362035
Effective Sample Size / time	6.718504282168 04e−11	5.748523610345 91e−11	2.612900129129 49e−11	8.189496370393 64e−11	9.158471202570 15e−11	3.510472997505 89e−11	3.020720681499 88e−11	5.484304647262 81e−11	1.216512984550 97e−10	9.007614649988 81e−11

	Edge51	Edge52	Edge53	Edge54	Edge55	Edge56	Edge57	Edge58	Edge59	Edge60
Effective Sample Size	9.98007151576974	8.12509154477833	98.3496679498929	6.23783229825146	5.49599073875383	12.447710682615	29.4367178816561	6.54255574128118	10.4925574388864	44.4631995613317
Effective Sample Size / time	1.438934074436 25e−10	1.171481693615 27e−10	1.418012768735 92e−09	8.993752630319 43e−11	7.924159996531 84e−11	1.794720110135 13e−10	4.244207702575 26e−10	9.433105138727 97e−11	1.512824672943 11e−10	6.410736917682 75e−10

	Edge61	Edge62	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	10.7142681470105	11.8863073066623	2.18871012253717	2.18871012253717	2.18871012253717	2.18871012253717	9.06503498520495	5.7808497773659	5.87613835059464
Effective Sample Size / time	1.544791086418 51e−10	1.713776557187 1e−10	3.155698402968 18e−11	3.491106167570 36e−11	3.923060909749 8e−11	8.118801059595 34e−11	1.307003432345 85e−10	8.334871860091 5e−11	8.472259628011 4e−11

Stabilised Fisher

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	5.04868651934203	7.94045929729852	1.41505077997194	3.95940886566261	5.15038548455517	3.18910571890996	1.51919316343704	3.64119387862533	4.39335971469305	35.4095205721965
Effective Sample Size / time	6.189890569565 9e−11	9.735319064487 2e−11	1.734908563811 31e−11	4.854392821722 4e−11	6.314577547712 74e−11	3.909970511974 29e−11	1.862591269963 31e−11	4.464248585234 28e−11	5.386433830364 53e−11	4.341348123377 91e−10

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	24.383765971918	47.8173980045358	13.9955081157649	16.121554316592	73.6448983280197	4.33867577475949	17.2294405261045	40.881177230949	2.14622534823203	6.22057685199651
Effective Sample Size / time	2.989546735806 19e−10	5.862603269890 27e−10	1.715904985782 2e−10	1.976566709949 4e−10	9.029157582092 42e−10	5.319389144027 86e−11	2.112397967725 86e−10	5.012195002509 98e−10	2.631357679326 88e−11	7.626674749148 67e−11

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	16.1925559057128	12.1667871108337	2.7240564103359	2.79548466012859	12.6850697785985	20.2116665663424	7.55677864046735	12.3639571798498	11.3658715227725	8.62934778647739
Effective Sample Size / time	1.985271787304 45e−10	1.491696513751 47e−10	3.339801554464 75e−11	3.427375430976 69e−11	1.555240031164 88e−10	2.478030747104 45e−10	9.264911311828 08e−11	1.515870348790 1e−10	1.393501075659 49e−10	1.057992376440 56e−10

	Edge31	Edge32	Edge33	Edge34	Edge35	Edge36	Edge37	Edge38	Edge39	Edge40
Effective Sample Size	9.2910017792 9725	2.99523047194927	NA	3.13616643457254	4.7819032115737	2.28360143332108	5.30062545498865	2.28067033594189	2.15865607522	2.1812013481749
Effective Sample Size / time	1.139113788807 53e−10	3.672271744535 15e−11	NA	3.845064842821 66e−11	5.862803618425 99e−11	2.799786226102 13e−11	6.498777729798 76e−11	2.796192584081 3e−11	2.646598245256 44e−11	2.674239647018 49e−11

	Edge41	Edge42	Edge43	Edge44	Edge45	Edge46	Edge47	Edge48	Edge49	Edge50
Effective Sample Size	5.7122286479418	42.2924589948165	32.5992578776839	24.9224940796599	55.5109560534054	4.7393937533731	30.602100102839	15.4847973262425	47.6840576114615	35.2723507579731
Effective Sample Size / time	7.003419622834 33e−11	5.185223762514 06e−10	3.996798734460 15e−10	3.055596945517 11e−10	6.805864100814 51e−10	5.810685331139 75e−11	3.751939244192 99e−10	1.898497769155 2e−10	5.846255207112 55e−10	4.324530558329 31e−10

	Edge51	Edge52	Edge53	Edge54	Edge55	Edge56	Edge57	Edge58	Edge59	Edge60
Effective Sample Size	8.2226740637466	2.17100844805563	6.07030417820194	8.42913184443819	9.97371024376576	10.997457417046	12.9069976403485	9.84183002797094	10.6799479901931	12.0027279635105
Effective Sample Size / time	1.008132559801 56e−10	2.661742745877 36e−11	7.442434471440 62e−11	1.033445105249 18e−10	1.222816563178 38e−10	1.348332040307 52e−10	1.582449270108 67e−10	1.206647523945 44e−10	1.309404121144 88e−10	1.471582209467 05e−10

	Edge61	Edge62	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	3.899928731491	16.0490696083648	2.32545472935409	2.32545472935409	2.32545472935409	2.32545472935409	4.11991506623239	2.0015240668761	1.516586120972
Effective Sample Size / time	4.781467810400 19e−11	1.967679796290 28e−10	2.851100032460 97e−11	3.116422352547 02e−11	3.673686270016 85e−11	4.111700695028 66e−11	5.051179810468 39e−11	2.453948150444 91e−11	1.859394932161 98e−11

Reflecting 2 Boundary

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	5.048686519342 03	7.94045929729852	1.41505077997194	3.95940886566261	5.15038548455517	3.189105718909 96	1.51919316343704	3.64119387862533	4.39335971469305	35.4095205721965
Effective Sample Size / time	6.47294091910772e−11	1.01804942146995e−10	1.81424219174721e−11	5.07637373876116e−11	6.60332956392396e−11	4.08876503308489e−11	1.94776355275112e−11	4.66838904753986e−11	5.63274383557929e−11	4.53986861255209e−10

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	24.383765971918	47.8173980045358	13.9955081157649	16.121554316592	73.6448983280197	4.33867577475949	17.2294405261045	40.881177230949	2.14622534823203	6.220576851996 51
Effective Sample Size / time	3.12625226218529e−10	6.13068747689166e−10	1.79436962107216e−10	2.06695084386198e−10	9.442041490290 27e−10	5.56263334029345e−11	2.20899337218681e−10	5.24139187302902e−10	2.7516839924551e−11	7.97542614133034e−11

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	16.1925559057128	12.1667871108337	2.7240564103359	2.79548466012859	12.6850697785985	20.2116665663424	7.55677864046735	12.3639571798498	11.3658715227725	8.629347786477 39
Effective Sample Size / time	2.07605398563521e−10	1.5599085790347 6e−10	3.49252347850643e−11	3.58410191957066e−11	1.62635780449145e−10	2.59134575021483e−10	9.68857572977122e−11	1.5851878314278e−10	1.4572228752809e−10	1.10637208664682e−10

	Edge31	Edge32	Edge33	Edge34	Edge35	Edge36	Edge37	Edge38	Edge39	Edge40
Effective Sample Size	9.29100177929725	2.99523047194927	NA	3.13616643457254	4.7819032115737	2.28360143332108	5.30062545498865	2.28067033594189	2.15865607522	2.1812013481749
Effective Sample Size / time	1.19120300629308e−10	3.84019681352003e−11	NA	4.02089136762692e−11	6.13089698693393e−11	2.92781441352066e−11	6.79595282317737e−11	2.92405644199857e−11	2.76762147660399e−11	2.79652686007018e−11

	Edge41	Edge42	Edge43	Edge44	Edge45	Edge46	Edge47	Edge48	Edge49	Edge50
Effective Sample Size	5.7122286479418	42.2924589948165	32.5992578776839	24.9224940796599	55.5109560534054	4.7393937533731	30.602100102839	15.4847973262425	47.6840576114615	35.2723507579731
Effective Sample Size / time	7.32367090190828e−11	5.42233286516094e−10	4.17956372297215e−10	3.1953228055735e−10	7.11708159182175e−10	6.07639544274386e−11	3.92350733616379e−10	1.985311978732e−10	6.11359185245165e−10	4.52228201651466e−10

	Edge51	Edge52	Edge53	Edge54	Edge55	Edge56	Edge57	Edge58	Edge59	Edge60
Effective Sample Size	8.2226740637466	2.17100844805563	6.07030417820194	8.42913184443819	9.97371024376576	10.997457417046	12.9069976403485	9.84183002797094	10.6799479901931	12.0027279635105
Effective Sample Size / time	1.05423228809714e−10	2.7834585025848e−11	7.78276095296856e−11	1.08070232167096e−10	1.27873332806191e−10	1.40998835733257e−10	1.65481126326 55e−10	1.26182491352738e−10	1.36928034836024e−10	1.53887449098445e−10

	Edge61	Edge62	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	3.899928731491	16.0490696083648	2.32545472935409	2.32545472935409	2.32545472935409	2.32545472935409	4.11991506623239	2.0015240668761	1.516586120972
Effective Sample Size / time	5.00011402390694e−11	2.05765755080257e−10	2.98147468960509e−11	3.25892962731945e−11	3.84167602220 21e−11	4.29971990245398e−11	5.28215937220431e−11	2.56616190833525e−11	1.94442105331385e−11

3 Step Leapfrog

	Edge1	Edge2	Edge3	Edge4	Edge5	Edge6	Edge7	Edge8	Edge9	Edge10
Effective Sample Size	5.84437508824641	4.80729081543756	1.30544092439141	14.0573824811713	14.6948945064625	15.1881329088785	15.7398034552998	12.4181936508156	3.22605600418543	32.6484059841324
Effective Sample Size / time	5.03898291388 38e−11	4.14481547048149e−11	1.12554283627726e−11	1.21201854889511e−10	1.26698442898207e−10	1.30951129355153e−10	1.35707581189733e−10	1.07068873374682e−10	2.78148490452214e−11	2.81492473421905e−10

	Edge11	Edge12	Edge13	Edge14	Edge15	Edge16	Edge17	Edge18	Edge19	Edge20
Effective Sample Size	36.2241576068018	18.7674983344004	25.5625805477988	25.6069540097294	20.6225050025378	26.5961766423451	16.1676236916694	14.344479845269	7.48750063875998	11.9579118848906
Effective Sample Size / time	3.12322375779062e−10	1.61812173269942e−10	2.20398938586581e−10	2.20781523744313e−10	1.77805922412914e−10	2.29310538170171e−10	1.39396219972715e−10	1.23677190045906e−10	6.45567528037051e−11	1.03100353354923e−10

	Edge21	Edge22	Edge23	Edge24	Edge25	Edge26	Edge27	Edge28	Edge29	Edge30
Effective Sample Size	16.1153449613572	27.8472725423828	32.7890993073188	15.5442041283082	21.7736866563538	39.7831722936668	4.97988488606718	20.3419370794226	30.2223788221256	17.0845470913938
Effective Sample Size / time	1.38945476095354e−10	2.40097407200187e−10	2.82705522278163e−10	1.34021136271678e−10	1.87731337186551e−10	3.43007973343531e−10	4.29362497702555e−11	1.75386883680091e−10	2.60575422011986e−10	1.47301875025286e−10

	Edge31	Edge32	Edge33	Edge34	Edge35	Edge36	Edge37	Edge38	Edge39	Edge40
Effective Sample Size	53.7110067358197	37.6824352134933	NA	12.083328892308	5.37332070296461	9.93988034977585	7.13284874892606	29.1980046881762	17.553173485248	20.6980537225826
Effective Sample Size / time	4.63092873305811e−10	3.24895552265621e−10	NA	1.04181690791252e−10	4.63284282822 8e−11	8.57010142098842e−11	6.14989668364017e−11	2.5174333358 43e−10	1.51342341894661e−10	1.78457298657121e−10

	Edge41	Edge42	Edge43	Edge44	Edge45	Edge46	Edge47	Edge48	Edge49	Edge50
Effective Sample Size	8.01108032501281	14.2083580925172	11.848477536043	29.3612520571816	41.0926266420601	35.4502034094483	11.4639439734474	16.4246590758599	15.3632569340045	19.9967312320216
Effective Sample Size / time	6.90710234541132e−11	1.22503556978268e−10	1.02156817381087e−10	2.53150841984658e−10	3.54298005191991e−10	3.05649392067875e−10	9.88413935376885e−11	1.41612362656308e−10	1.32461021106 85e−10	1.72410540887986e−10

	Edge51	Edge52	Edge53	Edge54	Edge55	Edge56	Edge57	Edge58	Edge59	Edge60
Effective Sample Size	53.2881863067136	13.0239532730307	9.4991894498786	8.43003597970263	10.6848188196603	21.741031826765	12.79905134038	20.4503775528634	12.7358994079673	8.68015679268902
Effective Sample Size / time	4.59447342542066e−10	1.12291694189864e−10	8.19014053871167e−11	7.26832323794141e−11	9.21238261700395e−11	1.87449789329219e−10	1.10352604075327e−10	1.76321850523579e−10	1.09808112143168e−10	7.48397580712366e−11

	Edge61	Edge62	piA	piG	piC	piT	TT	Alpha	Posterior
Effective Sample Size	20.6539496692188	8.57095955666565	2.55356820870637	2.55356820870637	2.55356820870637	2.55356820870637	9.04177261338542	5.16618246529309	4.06807708037208
Effective Sample Size / time	1.78077036322864e−10	7.38982664690435e−11	2.20167021774249e−11	2.8994746024714e−11	2.04614240480881e−11	1.76054827728202e−11	7.79575866061365e−11	4.45424956125273e−11	3.50747010430295e−11

Bibliography

Abbott, S. and Fairbanks, D. (2016). Experiments on Plant Hybrids by Gregor Mendel. *Genetics*, 204:407–422.

Adams, S. (2011). Markov Processes and Percolation theory. `https://warwick.ac.uk/fac/sci/maths/people/staff/stefan_adams/ma3h2notes.pdf`. Date accessed: 03.01.2020.

Allen, B. L. and Steel, M. (2001). Subtree Transfer Operations and their Induced Metrics on Evolutionary Trees. *Annals of combinatorics*, 5(1):1–15.

Altekar, G., Dwarkadas, S., Huelsenbeck, J. P., and Ronquist, F. (2004). Parallel Metropolis coupled Markov chain Monte Carlo for Bayesian phylogenetic inference. *Bioinformatics*, 20(3):407–415.

Andrieu, C. and Thoms, J. (2008). A tutorial on adaptive MCMC. *Statistics and Computing*, 18, issue 4:343 – 373. Publisher: Springer Netherlands.

Apache Software Foundation Oracle Corporation and Roman Staněk (2021). Apache Netbeans. `https://netbeans.apache.org/`.

Avery, O. T., MacLeod, C. M., and McCarty, M. (1944). Studies on the Chemical Nature of the Substance Inducing Transformation of Pneumococcal Types. *Journal of Experimental Medicine*, 79(2):137–158.

Baez, J. and Otter, N. (2015). Operads and Phylogenetic Trees. *Theory and Applications of Categories*, 32.

Bai, Y., Roberts, G., and Rosenthal, J. (2011). On the Containment Condition for Adaptive Markov Chain Monte Carlo Algorithms. *Adv. Appl. Stat.*, 21.

Barden, D., Le, H., and Owen, M. (2013). Central limit theorems for Fréchet means in the space of phylogenetic trees. *Electronic Journal of Probability [electronic only]*, 18.

Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J.-M., and Stuart, A. (2013). Optimal tuning of the hybrid Monte Carlo algorithm. *Bernoulli*, 19(5A):1501–1534.

Betancourt, M. (2011). Nested Sampling with Constrained Hamiltonian Monte Carlo. *AIP Conference Proceedings*, 1305:165–172.

Betancourt, M. (2012). Cruising the simplex: Hamiltonian Monte Carlo and the Dirichlet distribution. *AIP Conference Proceedings*, 1443(1):157–164.

Betancourt, M. (2013). A General Metric for Riemannian Manifold Hamiltonian Monte Carlo. In *International Conference on Geometric Science of Information*, pages 327–334. Springer.

Betancourt, M., Byrne, S., Livingstone, S., Girolami, M., et al. (2017). The geometric foundations of Hamiltonian Monte Carlo. *Bernoulli*, 4A(23):2257–2298.

Billera, L. J., Holmes, S. P., and Vogtmann, K. (2001). Geometry of the space of phylogenetic trees. *Advances in Applied Mathematics*, 27(4):733–767.

Bouchard-Côté, A., Sankararaman, S., and Jordan, M. I. (2012). Phylogenetic Inference via Sequential Monte Carlo. *Systematic Biology*, 61(4):579–593.

Boveri, T. (1904). *Ergebnisse über die Konstitution der chromatischen Substanz des Zellkerns. Von Theodor Boveri.* G. Fischer,.

Brockwell, A. and Kadane, J. B. (2018). Identification of Regeneration Times in MCMC Simulation, with Application to Adaptive Schemes.

Campbell, C. S. (2016). Encyclopædia Britannica. `https://www.britannica.com/plant/Poaceae/Economic-and-ecological-importance`. Date Published: October 07, 2016; Access Date: January 08, 2019.

Castro-Nallar, E., Pérez-Losada, M., Burton, G., and Crandall, K. (2011). The evolution of HIV: Inferences using phylogenetics. *Molecular phylogenetics and evolution*, 62:777–92.

Cherlin, S. (2016). Rooting Major Cellular Radiations using Statistical Phylogenetics. `http://theses.ncl.ac.uk/jspui/handle/10443/3492`.

Critchley, F., Marriott, P., and Salmon, M. (2002). On preferred point geometry in statistics. *Journal of Statistical Planning and Inference*, 102(2):229 – 245.

Darwin, C. (1857). *On the Origin of Species.* New York :D. Appleton and Co.,. http://www.biodiversitylibrary.org/bibliography/28875.

Dinh, V., Bilge, A., Zhang, C., Matsen, I., and Frederick, A. (2017). Probabilistic Path Hamiltonian Monte Carlo. *arXiv preprint arXiv:1702.07814*.

Drummond, A. J. and Bouckaert, R. R. (2015). *Bayesian evolutionary analysis by sampling trees*, page 79–96. Cambridge University Press.

Duane, S., Kennedy, A., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, 195(2):216 – 222.

Elena Akhmatskayaa, S. R. (2008). GSHMC: An efficient method for molecular simulation. *Journal of Computational Physics*, 227(10):4934–4954.

Forster, P., Forster, L., Renfrew, C., and Forster, M. (2020). Phylogenetic network analysis of SARS-CoV-2 genomes. *Proceedings of the National Academy of Sciences*, 117(17):9241–9243.

Gamerman, D. and Lopes, H. (2006). *Markov Chain Monte Carlo Stochastic Simulation for Bayesian Inference*. Taylor & Francis.

Gelfand, A. E. and Smith, A. F. M. (1990). Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association*, 85(410):398–409.

Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741.

Gilks, W. R., Best, N. G., and Tan, K. K. C. (1995). Adaptive Rejection Metropolis Sampling Within Gibbs Sampling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 44(4):455–472.

Girolami, M., Calderhead, B., and Chin, S. (2011). Riemann Manifold Langevin and Hamiltonian Monte Carlo Methods by Girolami and Calderhead. *Journal of the Royal Statistical Society, Series B*.

Graur, D. (2017). An Upper Limit on the Functional Fraction of the Human Genome. *Genome Biology and Evolution*, 9(7):1880–1885.

Grenander, U. and Miller, M. I. (1994). Representations of Knowledge in Complex Systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581.

Hasegawa, M., Kishino, H., and Yano, T.-a. (1985). Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution*, 22(2):160–174.

Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109.

Heaps, S. E., Nye, T. M., Boys, R. J., Williams, T. A., and Embley, T. M. (2014). Bayesian modelling of compositional heterogeneity in molecular phylogenetics. *Statistical applications in genetics and molecular biology*, 13(5):589–609.

Hillis, D. and Huelsenbeck, J. (1994). To tree the truth: biological and numerical simulation of phylogeny. *Society of General Physiologists series*, 49:55—67.

Hillis, D. M., Huelsenbeck, J., and Swofford, D. L. (1994). Hobgoblin of phylogenetics? *Nature*, 369:363–364.

Hodgkinson, A., Ladoukakis, M., and Eyre-Walker, A. (2009). Cryptic Variation in the Human Mutation Rate. *PLoS biology*, 7:e1000027.

Hoffman, M. and Gelman, A. (2014). The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623.

Holmes, E. C., Nee, S., Rambaut, A., Garnett, G. P., Harvey, P. H., Harvey, P. H., Leigh Brown, A. J., and Smith, J. M. (1995). Revealing the history of infectious disease epidemics through phylogenetic trees. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 349(1327):33–40.

Huelsenbeck, J. P. (1997). Is the Felsenstein Zone a Fly Trap? *Systematic Biology*, 46(1):69–74.

Huelsenbeck, J. P. and Andolfatto, P. (2007). Inference of Population Structure Under a Dirichlet Process Model. *Genetics*, 175(4):1787–1802.

Huelsenbeck, J. P., Larget, B., and Alfaro, M. E. (2004). Bayesian Phylogenetic Model Selection Using Reversible Jump Markov Chain Monte Carlo. *Molecular Biology and Evolution*, 21(6):1123–1133.

Huelsenbeck, J. P., Larget, B., Miller, R. E., and Ronquist, F. (2002). Potential Applications and Pitfalls of Bayesian Inference of Phylogeny. *Systematic Biology*, 51(5):673–688.

Huelsenbeck, J. P. and Ronquist, F. (2001). MRBAYES: Bayesian inference of phylogenetic trees . *Bioinformatics*, 17(8):754–755.

Izaguirre, J. A. and Hampton, S. S. (2004). Shadow hybrid Monte Carlo: an efficient propagator in phase space of macromolecules. *Journal of Computational Physics*, 200(2):581 – 604.

Johansen, A. M. and Evers, L. (2007). Monte Carlo Methods. `https://warwick.ac.uk/fac/sci/statistics/staff/academic-research/johansen/teaching/mcm-2007.pdf`. Date accessed: 03,01,2020.

Jow, H., Hudelot, C., Rattray, M., and Higgs, P. G. (2002). Bayesian Phylogenetics Using an RNA Substitution Model Applied to Early Mammalian Evolution. *Molecular Biology and Evolution*, 19(9):1591–1601.

Kreck, M. (2010). *Differential algebraic topology. From stratifolds to exotic spheres*. Providence, R.I.: American Mathematical Society.

Kuhner, M., Yamato, J., and Felsenstein, J. (1995). Estimating Effective Population Size and Mutation Rate From Sequence Data Using Metropolis-Hastings Sampling. *Genetics*, 140:1421–30.

Lakner, C., van der Mark, P., Huelsenbeck, J. P., Larget, B., and Ronquist, F. (2008). Efficiency of Markov Chain Monte Carlo Tree Proposals in Bayesian Phylogenetics. *Systematic Biology*, 57(1):86–103.

Lan, S., Zhou, B., and Shahbaba, B. (2013). Spherical hamiltonian monte carlo for constrained target distributions. *JMLR workshop and conference proceedings*, 32.

Larget, B. (2008). Markov chain Monte Carlo algorithms for the Bayesian analysis of phylogenetic trees. *Society for Molecular Biology and Evolution*.

Lee, J. M. (2013). *Introduction to Smooth Manifolds*. Springer.

Leimkuhler, B. and Reich, S. (2005). *Simulating Hamiltonian Dynamics*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press.

Matassi, G., Sharp, P. M., and Gautier, C. (1999). Chromosomal location effects on gene sequence evolution in mammals. *Current Biology*, 9(15):786–791.

Mau, B. and Newton, M. (1997). Phylogenetic Inference for Binary Data on Dendograms Using Markov Chain Monte Carlo. *Journal of Computational and Graphical Statistics - J COMPUT GRAPH STAT*, 6:122–131.

Mayrose, I., Friedman, N., and Pupko, T. (2005). A Gamma mixture model better accounts for among site rate heterogeneity. *Bioinformatics (Oxford, England)*, 21 Suppl 2:ii151–8.

Mendel, G. and Bennett, J. (1965). *Experiments in plant hybridization*, volume 254. Oliver & Boyd.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *jcp*, 21(6):1087–1092.

Meyn, S. and Tweedie, R. L. (2009). *Markov Chains and Stochastic Stability*. Cambridge University Press, USA, 2nd edition.

Miller, E. (2015). Fruit Flies and Moduli: Interactions between Biology and Mathematics. *Notices of the American Mathematical Society*, 62.

National Human Genome Research Institute (2010). NHGRI. *Why Mouse Matters*.

Neal, R. M. (1994). An improved acceptance procedure for the hybrid Monte Carlo algorithm. *Journal of Computational Physics*, 111(1):194–203.

Neal, R. M. et al. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2:113–162.

Nee, S., May, R. M., and Harvey, P. H. (1994). The Reconstructed Evolutionary Process. *Philosophical Transactions: Biological Sciences*, 344(1309):305–311.

Newton, M. A., Mau, B., and Larget, B. (1999). Markov Chain Monte Carlo for the Bayesian Analysis of Evolutionary Trees from Aligned Molecular Sequences. *Lecture Notes-Monograph Series*, 33:143–162.

Nye, T. (2015). Convergence of random walks to Brownian motion in phylogenetic tree-space. *arXiv*.

Nye, T. M., Liò, P., and Gilks, W. R. (2005). A novel algorithm and web-based tool for comparing two alternative phylogenetic trees. *Bioinformatics*, 22(1):117–119.

Nye, T. M. W. and Heaps, S. E. (2014). PhyloCore MCMC Manual.

Nylander, J. A. A., Ronquist, F., Huelsenbeck, J. P., and Nieves-Aldrey, J. (2004). Bayesian Phylogenetic Analysis of Combined Data. *Systematic Biology*, 53(1):47–67.

Owen, M. and Provan, J. S. (2011). A Fast Algorithm for Computing Geodesic Distances in Tree Space. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 8(1):2–13.

Paradis, E. and Schliep, K. (2018). ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. *Bioinformatics*, 35:526–528.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2006). CODA: Convergence Diagnosis and Output Analysis for MCMC. *R News*, 6(1):7–11.

Pybus, O. G., Charleston, M. A., Gupta, S., Rambaut, A., Holmes, E. C., and Harvey, P. H. (2001). The Epidemic Behavior of the Hepatitis C Virus. *Science*, 292(5525):2323–2325.

R., B., T.G., V., J., B.-S., S., D., M., F., and Gavryushkina, A. (2019). BEAST 2.5: An advanced software platform for Bayesian evolutionary analysis. *PLoS computational biology*, 15(4):754–755.

Roberts, G. and Rosenthal, J. (2007). Coupling and Ergodicity of Adaptive Markov Chain Monte Carlo Algorithms. *Journal of Applied Probability*, 44:458–475.

Roberts, G. O. and Tweedie, R. L. (1996). Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms. *Biometrika*, 83(1):95–110.

Roch, S. (2012). Notes 13 : Eigenvector-based estimation. `https://www.math.wisc.edu/~roch/evol-gen/roch-evolgen-notes13.pdf`. Date accessed: 03.01.2020.

Ronquist, F. and Huelsenbeck, J. P. (2003). MrBayes 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, 19(12):1572–1574.

Rossky, P., Doll, J., and Friedman, H. (1978). Brownian dynamics as smart Monte Carlo simulation. *The Journal of Chemical Physics*, 69.

RStudio, Inc (2013). Easy web applications in R. URL: `http://www.rstudio.com/shiny/`.

Shun-ichi, A. (1985). *Differential-Geometrical Methods in Statistics*, volume 28. Springer-Verlag New York.

Simmonds, P., Holmes, E. C., Cha, T.-A., Chan, S.-W., McOmish, F., Irvine, B., Beall, E., Yap, P. L., Kolberg, J., and Urdea, M. S. (1993). Classification of hepatitis C virus into six major genotypes and a series of subtypes by phylogenetic analysis of the NS-5 region. *Journal of General Virology*, 74(11):2391–2399.

Sohl-Dickstein, J., Mudigonda, M., and DeWeese, M. R. (2014). Hamiltonian Monte Carlo without detailed balance. *arXiv preprint arXiv:1409.5191*.

Soltis, D. E. and Soltis, P. S. (2003). The Role of Phylogenetics in Comparative Genetics. *Plant Physiology*, 132(4):1790–1800.

Spielman, S. J. (2019). Model fit does not predict accuracy in single-gene protein phylogenetics. *bioRxiv*.

Srinivasan, A. R., Torres, R., Clark, W., and Olson, W. K. (1987). Base Sequence Effects in Double Helical DNA. I. Potential Energy Estimates of Local Base Morphology. *Journal of Biomolecular Structure and Dynamics*, 5(3):459–496. PMID: 3271482.

Stephens, M. and Donnelly, P. (2000). Inference in molecular population genetics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(4):605–635.

Strathmann, H., Sejdinovic, D., Livingstone, S., Szabo, Z., and Gretton, A. (2015). Gradient-free Hamiltonian Monte Carlo with efficient kernel exponential families. *Advances in Neural Information Processing Systems*, pages 955–963.

Sutton, W. S. (1903). The Chromosomes in Heredity. *Biological Bulletin*, 4(5):231–251.

Tavaré, S. (1986). *Some probabilistic and statistical problems in the analysis of DNA sequences*. Providence, R.I. American Mathematical Society.

The Stan Development Team (2011-2019). Stan Modeling Language Users Guide and Reference Manual. `http://mc-stan.org/docs/2_19/reference-manual/index.html#overview`.

Todorova, A. and Danieli, G. A. (1997). Large majority of single-nucleotide mutations along the dystrophin gene can be explained by more than one mechanism of mutagenesis. *Human Mutation*, 9(6):537–547.

Treatment Action Group (2019). HCV Genotypes. `https://www.treatmentactiongroup.org/publication/hcv-genotypes/`.

Yang, Z. (1994). Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: Approximate methods. *Journal of Molecular Evolution*, 39(3):306–314.

Yang, Z. (2006). *Computational molecular evolution*. Oxford University Press.

Yang, Z. and Rannala, B. (1997). Bayesian Phylogenetic Inference Using DNA Sequences: A Markov Chain Monte Carlo Method. *Molecular biology and evolution*, 14:717–24.

Yang, Z. and Rannala, B. (2010). Bayesian species delimitation using multilocus sequence data. *Proceedings of the National Academy of Sciences*, 107(20):9264–9269.

Zhang, C. (2020). Improved Variational Bayesian Phylogenetic Inference with Normalizing Flows.

Zhang, C. and IV, F. A. M. (2019). Variational Bayesian Phylogenetic Inference. In *International Conference on Learning Representations*.

(Darwin, 1857)