THE UNIVERSITY
of EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e. g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

BAYESIAN COMPUTATION IN ASTRONOMY

Novel methods for parallel and gradient–free inference

MINAS KARAMANIS

Doctor of Philosophy
The University of Edinburgh
July 2022

Heard melodies are sweet,
but those unheard, are sweeter.

— John Keats

LAY SUMMARY

Over the past few decades, the volume of astronomical and cosmological data has increased substantially. In response to that, a variety of astrophysical models have been proposed to explain the plethora of observations. As the information provided by the data is always incomplete and uncertain, inferring the properties of a model, including the values of its parameters, given the observed data, generally requires us to reason in the face of uncertainty. In the context of *Bayesian inference*, uncertainty is represented by the notion of probability. One usually starts by quantifying their state of knowledge about the possible values of the model parameters *prior* to seeing the data, in the form of a probability distribution called the *prior*. The next step is to use the so–called *Bayes' theorem* in order to update one's degree of belief about the model parameters given the available data. The outcome of this updating process is the *posterior* probability distribution of the model parameters given the data which quantifies the plausibility of different parameter values.

Approximating the *posterior* generally requires the use of probabilistic computational methods. Standard practice in astronomy often employs conventional computational tools (e.g. *Markov chain Monte Carlo*) despite their specific theoretical limitations or narrow range of validity. The aim of this thesis is to first introduce the basic principles of Bayesian inference along with the basic methods used for Bayesian computation and then present two novel algorithms and their respective software implementations. A common element of these newly developed tools is their ability to exploit the available information about the geometry of the posterior in order to approximate it more quickly. Finally, both methods are able to benefit from the possible availability of multiple CPUs in order to accelerate their computation.

ABSTRACT

The goal of this thesis is twofold; introduce the fundamentals of *Bayesian inference and computation* focusing on astronomical and cosmological applications, and present recent advances in probabilistic computational methods developed by the author that aim to facilitate *Bayesian data analysis* for the next generation of astronomical observations and theoretical models.

The first part of this thesis familiarises the reader with the notion of probability and its relevance for science through the prism of *Bayesian reasoning*, by introducing the key constituents of the theory and discussing its best practices. The second part includes a pedagogical introduction to the principles of *Bayesian computation* motivated by the geometric characteristics of probability distributions and followed by a detailed exposition of various methods including *Markov chain Monte Carlo (MCMC)*, *Sequential Monte Carlo (SMC)* and *Nested Sampling (NS)*. Finally, the third part presents two novel computational methods and their respective software implementations.

The first such development is *Ensemble Slice Sampling (ESS)*, a new class of *MCMC* algorithms that extend the applicability of the standard *Slice Sampler* by adaptively tuning its only hyperparameter and utilising an ensemble of parallel walkers in order to efficiently handle strong correlations between parameters. The parallel, black–box and gradient–free nature of the method renders it ideal for use in combination with computationally expensive and *non–differentiable* models often met in astronomy. *ESS* is implemented in *Python* in the well–tested and open-source software package called *zeus* that is specifically designed to tackle the computational challenges posed by modern astronomical and cosmological analyses. In particular, use of the code requires minimal, if any, hand–tuning of hyperparameters while its performance is insensitive to linear correlations and it can scale up to thousands of CPUs without any extra effort.

The next contribution includes the introduction of *Preconditioned Monte Carlo (PMC)*, a novel *Monte Carlo* method for *Bayesian inference* that facilitates effective sampling of probability distributions with non–trivial geometry. *PMC* utilises a *Normalising Flow (NF)* in order to decorrelate the parameters of the distribution and then proceeds by sampling from the preconditioned target distribution using an adaptive *SMC* scheme. *PMC*, through its *Python* implementation *pocoMC*, achieves excellent sampling performance, including accurate estimation of the *model evidence*, for highly correlated, non–Gaussian, and multimodal target distributions. Finally, the code is directly parallelisable, manifesting linear scaling up to thousands of CPUs.

ACKNOWLEDGEMENTS

The completion of this journey would not have been possible without the support and guidance of a great number of people. This thesis is dedicated to all of them.

First and foremost, I would like to express my sincere gratitude to my supervisors Florian Beutler and John Peacock. Florian's constant encouragement and advice from day one of my PhD studies enabled me to explore freely and pursue whichever research avenues I found interesting. I only met John in the middle of my PhD studies but his enthusiasm and wisdom were enough to act as a huge source of inspiration ever since. My supervisors allowed me to become the researcher that I am today and also showed me the kind of researcher that I want to become. It was a pleasure and privilege to learn from and work with them.

I am also extremely grateful to Alan Heavens and Ross McLure, for being the two examiners for my viva. Their keen interest in the subject along with their insightful questions turned the examination into a fascinating and very enjoyable debate.

This thesis is but the last stage of a long journey that started more than twenty years ago. Therefore, I would like to deeply thank my family for their continuous support during all these years, and especially my father and grandfather for cultivating my love for science when I was really young. My gratitude extends to my mother, aunt, and grandmother that enabled my involvement in science at an early age, always providing access to books, lectures, telescopes, microscopes, and spare parts that I required for my "little science experiments", and of course to my brother for always being my "lab assistant".

Special thanks goes to all my friends for making this process easier and more enjoyable and especially to Jamie, Mike, Tasos and anyone else who traveled this path along with me.

A big thanks goes to my girlfriend Denia for her unconditional support and for believing in me during the past four years. Finally, I would like to thank my two cats, Poco and Gatoulis, for keeping me company and lightening my mood during this stressful period.

DECLARATION

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Parts of this work have been published in (Karamanis & Beutler, 2021; Karamanis, Beutler & Peacock, 2021; Karamanis, Beutler, Peacock, et al., 2022; Karamanis, Nabergoj, et al., 2022).

(M. Karamanis, July 2022)

CONTENTS

Lay summary v
Abstract vii
Acknowledgements ix
Declaration xi
List of Figures xix
List of Tables xxxiii
List of Algorithms xxxv

I BAYESIAN INFERENCE

1 PROBABILITY THEORY 3
- 1.1 The goal of science 3
- 1.2 The notion of probability 4
- 1.3 Bayes' theorem 7
  - 1.3.1 Rules of probability 7
  - 1.3.2 Updating degrees of belief 9
- 1.4 Representing probability distributions 12
  - 1.4.1 Function representation 12
  - 1.4.2 Sample representation 14
- 1.5 Asymptotic behaviour 14
  - 1.5.1 Bernstein–von Mises theorem 14
  - 1.5.2 Heuristic argument 15
- 1.6 Estimating parameters 17
  - 1.6.1 Coin–tossing experiment 17
  - 1.6.2 Fitting a model to data 19

2 QUANTIFYING PRIOR KNOWLEDGE 23
- 2.1 Conjugate priors 23
  - 2.1.1 Binomial likelihood function with Beta prior 24
- 2.2 Jeffreys priors 25
  - 2.2.1 One–dimensional case 25
  - 2.2.2 Multi–dimensional case 26
  - 2.2.3 Gaussian distribution with mean parameter 27
  - 2.2.4 Gaussian distribution with scale parameter 27
- 2.3 Maximum entropy priors 27
  - 2.3.1 Shannon's entropy 28
  - 2.3.2 Relative entropy 29
  - 2.3.3 Lagrange multipliers 31
  - 2.3.4 Uniform distribution 31
  - 2.3.5 Exponential distribution 33

2.3.6 Normal distribution 35
2.4 Reference priors 37
2.4.1 Mutual information 38
2.4.2 Maximising the mutual information 38
2.4.3 Asymptotic solution 40
2.4.4 Numerical solution 41
2.4.5 Many parameters 41
2.5 Weakly informative and regularisation priors 44
2.6 Informative priors 45
3 MAKING PREDICTIONS AND EVALUATING MODELS 47
3.1 Making predictions 47
3.1.1 Prior predictive checks 47
3.1.2 Posterior predictive checks 48
3.2 Evaluating and comparing models 49
3.2.1 Bayes factors 49
3.2.2 Cross–validation 53
3.2.3 Model averaging 55

II BAYESIAN COMPUTATION

4 PRINCIPLES OF BAYESIAN COMPUTATION 59
4.1 Expectation values 59
4.2 Quadrature and uniform grids 60
4.3 Geometry of high–dimensional spaces 62
4.4 Concentration of measure 63
4.5 Typical set 67
4.6 Laplace approximation 69
4.7 Monte Carlo estimators 70
4.8 Importance sampling 72
4.9 Markov chain Monte Carlo 74
4.9.1 Requirements of MCMC 76
4.9.2 Expected behaviour 76
4.9.3 Central limit theorem of MCMC 78
4.9.4 Autocorrelation 79
5 SIMPLE MCMC METHODS 83
5.1 Metropolis–Hastings 84
5.1.1 Random–walk Metropolis 85
5.1.2 Independence Metropolis 86
5.1.3 Metropolis–adjusted Langevin algorithm 86
5.1.4 Adaptive Metropolis 87
5.2 Gibbs sampling 89
5.2.1 Gibbs sampler 89
5.2.2 Metropolis–within–Gibbs sampler 90
6 AUXILIARY VARIABLE MCMC METHODS 93

- 6.1 Simulated annealing . . . 93
- 6.2 Slice sampling . . . 95
- 6.3 Hamiltonian Monte Carlo . . . 99
  - 6.3.1 Auxiliary momentum variable . . . 99
  - 6.3.2 The Hamiltonian . . . 100
  - 6.3.3 Hamilton's equations . . . 100
  - 6.3.4 Leapfrog integration . . . 100
  - 6.3.5 Metropolis acceptance criterion . . . 101
  - 6.3.6 Performance and tuning . . . 101

7 ENSEMBLE MCMC METHODS 103
- 7.1 Gaussian ensemble . . . 104
- 7.2 Affine–invariant stretch move . . . 106
- 7.3 Differential evolution . . . 108
- 7.4 Parallel tempering . . . 110

8 EVIDENCE AND BAYES FACTOR COMPUTATION 115
- 8.1 Naive Monte Carlo estimator . . . 115
- 8.2 Importance sampling estimator . . . 115
- 8.3 Harmonic mean estimator . . . 116
- 8.4 Laplace estimator . . . 116
- 8.5 Bridge sampling . . . 116
- 8.6 Thermodynamic integration . . . 119
- 8.7 Annealed importance sampling . . . 120
- 8.8 Savage–Dickey density ratio . . . 121

9 ADVANCED METHODS 125
- 9.1 Sequential Monte Carlo . . . 125
  - 9.1.1 Background . . . 125
  - 9.1.2 Bridging the prior and the posterior . . . 126
  - 9.1.3 Correction – Selection – Mutation . . . 126
  - 9.1.4 Effective sample size . . . 128
  - 9.1.5 Setting the temperature ladder . . . 129
- 9.2 Nested sampling . . . 130
  - 9.2.1 Multi–dimensional integration . . . 130
  - 9.2.2 Sampling procedure . . . 132
  - 9.2.3 Termination criterion . . . 133
  - 9.2.4 Uncertainty . . . 134
  - 9.2.5 Likelihood–constrained prior sampling . . . 135
  - 9.2.6 Parallelisation . . . 137

III NOVEL DEVELOPMENTS

10 ENSEMBLE SLICE SAMPLING 141
- 10.1 Introduction . . . 141
- 10.2 Standard Slice Sampling . . . 143
- 10.3 Ensemble Slice Sampling . . . 145

10.3.1 Adaptively tuning the length scale 146
10.3.2 The choice of direction and parallelisation 146
10.4 Empirical evaluation 155
10.4.1 Performance tests 157
10.4.2 Comparison to other ensemble methods 160
10.4.3 Convergence of the Length Scale μ 169
10.4.4 Parallel Scaling 169
10.5 Discussion 170
10.6 Conclusion 172
10.7 Appendix: Estimating the Effective Sample Size 173
11 ZEUS 175
11.1 Introduction 175
11.2 Ensemble Slice Sampling 177
11.2.1 Slice sampling 178
11.2.2 Walkers, moves and parallelism 179
11.3 Empirical Evaluation 181
11.3.1 Toy examples 182
11.3.2 Real astronomical analyses 196
11.4 Discussion 200
11.5 Conclusions 201
12 PRECONDITIONED MONTE CARLO 203
12.1 Introduction 203
12.2 Method 206
12.2.1 Sequential Monte Carlo 206
12.2.2 Normalising Flows 208
12.2.3 Preconditioning 210
12.2.4 Preconditioned Monte Carlo 211
12.2.5 Hyperparameters 213
12.2.6 Parallelisation 213
12.3 Empirical Evaluation 214
12.3.1 Rosenbrock distribution 215
12.3.2 Gaussian Mixture 215
12.3.3 Primordial Features 218
12.3.4 Gravitational Waves 219
12.4 Discussion 219
12.5 Conclusions 221
12.6 Appendix: Comparison to Independent Metropolis–Hastings Sequential Monte Carlo 222
13 POCOMC 225
13.1 Summary 225
13.2 Statement of need 225
13.3 Method 227
13.3.1 Sequential Monte Carlo 227

13.3.2 Preconditioned Monte Carlo 228
13.4 Features 228
14 CONCLUSIONS 229
BIBLIOGRAPHY 231

LIST OF FIGURES

Figure 1.1 In the Bayesian interpretation of probability, the degree of belief is distributed and the variable has a specific (unknown) fixed value. 4

Figure 1.2 In the frequentist interpretation of probability, the value of the variable itself is distributed along different experiments. 5

Figure 1.3 Illustration of posterior probability density function of Saturn's mass. M corresponds to the most probable value for the mass at the peak of the density. M_{min} and M_{max} denote the values of the mass that deviate by 1% by the mean value M. The shaded area signifies the probability that the mass of Saturn is between the values of M_{min} and M_{max}. 6

Figure 1.4 Left: Likelihood function $\mathcal{L}(\mu)$ for the family of sampling distributions as shown on the right. The marked letters indicate the points in which the observed data intersect the sampling distributions. Right: The different groups of contours illustrate the sampling distribution $p(d_1, d_2|\mu)$ for different values of μ, ranging from -2 to 2. The vertical line corresponds to the observed data $(d_1, d_2) = (0, 0)$ and the marked letters indicate the points in which the observed data intersect the sampling distributions. 10

Figure 1.5 Illustration of Bayes' theorem. The posterior probability is proportional to the product of the likelihood function and prior probability. 12

Figure 1.6 The evolution of the posterior probability distribution of a coin–tossing experiment for increasing number of trials. F is the bias parameter that we want to estimate. H is the number of times the coin landed on *heads* and N is the total number of trials. The continuous line corresponds to the case of using a uniform prior whereas the dashed line to a normal prior. The dotted line shows the true (unknown) value of F. 19

Figure 1.7 Example of data d and the straight line model $m(t|, \alpha, \beta) = \alpha + \beta t$ that was used to generate them assuming true values $\theta^* = (\alpha^*, \beta^*) = (1, 1)$ and $\epsilon = 0.1$. 20

Figure 1.8 1−D and 2−D marginal posterior contours of fitting the straight line model $m(t|,\alpha,\beta) = \alpha + \beta t$ to the data of Figure 1.7 assuming flat/uniform priors $\alpha, \beta \sim \mathcal{U}(-5, 5)$. The black lines show the true values of the parameters $\theta^{*} = (\alpha^{*}, \beta^{*}) = (1, 1)$ which were used to generate the data. 22

Figure 3.1 *Prior predictive distributions* $p(d|\mathcal{M}_i)$ and $p(d|\mathcal{M}_j)$ for models M_i and M_j respectively. The dashed line, that intersects both distributions, corresponds to the actual observed data. The Bayes factor is simply the ration between the values at the two points of intersection, in this case favouring M_j over M_i. It is clear that for other realisations of the actual observed data (e.g. on the right part of the data vector) the other model would be favoured. 50

Figure 3.2 The characteristic width $\delta\theta$ of the likelihood function $p(d|\theta,\mathcal{M}_1)$ and $\Delta\theta$ of the prior distribution. . . 51

Figure 4.1 Uniform grid approximation of 1−dimensional probability distribution. 60

Figure 4.2 Uniform grid approximation of 2−dimensional probability distribution. 61

Figure 4.3 Uniform grid approximation of 2−dimensional probability distribution with highlighted the grid−cells that actually contribute to the calculation of an expectation value. 62

Figure 4.4 The ratio of the volume of a hyper−sphere of radius R to the volume of a hyper−cube of edge size $2R$ as function of the number of dimensions D. 64

Figure 4.5 Scaling of differential volume with the number of dimensions as a function of distance. 65

Figure 4.6 Scaling of differential probability mass with the number of dimensions as a function of distance. 66

Figure 4.7 Scaling of differential probability mass with the number of dimensions as a function of distance normalised by the square root of the number of dimensions. . . 67

Figure 4.8 Illustration of the typical set as the region in parameter space that the product of probability density and differential volume is non−negligible. 68

Figure 4.9 Illustration of the typical set as a thin hyper−shell surrounding the mode of the probability distribution. 68

Figure 4.10 Illustration of the Laplace approximation to a skewed probability density. 70

Figure 4.11 Illustration of the typical set including samples generated using exact Monte Carlo sampling. 71

Figure 4.12 Illustration of the typical set including samples generated using an unsuitable importance density. In high dimensions the typical set corresponds to a very thin shell and it is difficult to achieve sufficient overlap between the typical set of the auxiliary and target distribution. Here, this is depicted by samples that do not reside in the typical set of the target and will thus have low importance weights. 74

Figure 4.13 Example of trace plot of a Markov chain for parameter θ. The chain reaches the stationary state after about 50 iterations. 75

Figure 4.14 Initial stage of exploration – no exploration has taken place. 77

Figure 4.15 Burn–in stage of exploration – the absolute difference between the estimate of f and its expectation value slowly decreases as the chain approaches the typical set. 77

Figure 4.16 Initial convergence phase – the absolute difference between the estimate of f and the expectation value decreases rapidly as the chain approaches explores the typical set for the first time. 78

Figure 4.17 Stationary or equilibrium phase – the absolute difference between the estimate of f and the expectation value has reached its minimum value as the chain samples fully populate the typical set. 79

Figure 4.18 Markov chains with different degrees of autocorrelation. 80

Figure 4.19 Autocorrelation as a function of lag ℓ for a weakly and a strongly correlated chain. 81

Figure 6.1 Illustration of the gradual annealing performed in the posterior distribution. The prior distribution corresponds to $1/T \longrightarrow 0$ and the posterior is recovered as $1/T \longrightarrow 1$. 94

Figure 6.2 Illustration of the stepping–out and shrinking procedures used in slice sampling. Given an initial state θ in the Markov chain, an auxiliary variable ϕ is sampled corresponding to the height thus defining the extended state (θ, ϕ) shown here as a *blue* point. An interval of a certain width is placed uniformly around the current point (θ, ϕ) and expanded in steps of size equal to the initial width until both of its ends, L and R, are outside the graph. A new state, shown in *red*, is then proposed uniformly along the interval (L, R). Since the proposed state lies above the graph of $f(\theta)$ (i.e. not in the slice shown as a continuous line) it is rejected. A new state, shown in *green*, is the proposed uniformly between the rejected state and R. Since the proposed state is below the graph, and thus in the slice, it is accepted and added to the Markov chain. The whole process is then repeated. 97

Figure 6.3 Illustration of Hamiltonian trajectories in parameter space. The *black* points correspond to the accepted states. 99

Figure 7.1 Illustration of the Gaussian ensemble MCMC method. A new state θ'_k is proposed in the vicinity of the position θ_k of the walker that is updated using an rescaled version of the sample covariance matrix of the rest of the walkers (i.e. excluding θ_k) for the normal proposal distribution. 104

Figure 7.2 Illustration of the affine–invariant stretch move. The selected walker θ_k is moved to its new position θ'_k along the line defined by θ_j and θ_k. θ_j is a walker that is uniformly selected from the rest of the ensemble (i.e. excluding θ_k). 106

Figure 7.3 Illustration of the differential evolution Monte Carlo. The selected walker θ_k is moved to its new position θ'_k parallel to the line defined by θ_i and θ_j. The latter are two walkers that are uniformly selected from the rest of the ensemble (i.e. excluding θ_k). 108

Figure 7.4 Illustration of the gradual tempering performed in the posterior distribution. The prior distribution corresponds to $\beta \longrightarrow 0$ and the posterior is recovered as $\beta \longrightarrow 1$. 111

Figure 7.5 Illustration of the parallel tempering swaps performed between adjacent temperature levels. 112

Figure 8.1 The *Savage–Dickey density ratio* expresses the Bayes factor BF_{01} as the ratio of marginal posterior to the prior density at the point ϕ_0 in which model M_1 reduces to M_0. 122

Figure 9.1 Illustration of the *Sequential Monte Carlo* algorithm with its three fundamental steps. During the correction step the particles are reweighted to represent the next probability distribution. Selection removes the particles with the smaller important weights and multiplies those with larger weights. Finally, mutation diversifies the particles by moving them. 127

Figure 9.2 Illustration comparing two ways which one can use to approximate the model evidence integral. The left panel shows the direct multi–dimensional integration over the parameters. The right panel shows the one–dimensional integration over the prior volume X enclosed in the iso–likelihood contours. 130

Figure 9.3 Illustration of 8 samples drawn uniformly from the prior with their respective iso–likelihood contours (*left*), along with their corresponding contributions to the evidence integral (*right*). 131

Figure 9.4 Illustration of the nested sampling procedure. Given some uniformly distributed points from the prior, we identify and remove the worst point, that is, the point with the minimum likelihood value. $\mathcal{L}_{min}$, and replace it a new point sampled from the prior subject to the likelihood constrain $\mathcal{L} > \mathcal{L}_{min}$. Finally, the volume is contracted to account for the removal of the worst point. 132

Figure 10.1 The plot shows the univariate slice sampling method. Given an initial value x_0, a value y_0 is uniformly sampled along the vertical slice $(0, f(x_0))$ (green dashed line) thus defining the initial point (blue star). An interval (L, R) is randomly positioned horizontally around the initial point, and then it is expanded in steps of size $\mu = R - L$ until both of its ends L', R' are outside the slice. The new point (green star) is generated by repeatedly sampling uniformly from the expanded interval (L', R') until a point is found inside the slice. Points outside the slice (e.g. the red star) are used to shrink the interval (L', R') by moving L' or in this case R' to that point and accelerate the sampling procedure. 144

Figure 10.2 The plot shows the differential direction move. Two walkers (red) are uniformly sampled from the complementary ensemble (blue). Their positions define the direction vector (solid black). The selected walker (magenta) then moves by Slice Sampling along the parallel direction (dashed black). 150

Figure 10.3 The plot shows the Gaussian direction move. A direction vector (solid black) is sampled from the Gaussian-approximated distribution of the walkers of the complementary ensemble (green). The selected walker (magenta) then moves by Slice Sampling along the parallel direction (dashed black). 151

Figure 10.4 The plot shows the global direction move assuming that the uniformly selected pair of walkers of the complementary ensemble belongs to different components (blue and green). A position (red) is sampled from each component (using the re-scaled by γ covariance matrix). Those two points (red) define the direction vector (black) connecting the two modes (blue and green). The selected walker (magenta) then moves by slice sampling along the parallel direction (dashed). 154

Figure 10.5 The plots compare the 1-sigma and 2-sigma contours generated by the optimised random-walk Metropolis (left), Standard Slice (centre) and Ensemble Slice Sampling (right) methods to those obtained by Independent Sampling (blue) for the AR(1) distribution. All samplers used the same number of probability density evaluations, 3×10^5. Only the first two dimensions are shown here. 158

Figure 10.6 The plots compare the 1-sigma and 2-sigma contours generated by the optimised random-walk Metropolis (left), Standard Slice (centre) and Ensemble Slice Sampling (right) methods to those obtained by Independent Sampling (blue) for the correlated funnel distribution. All samplers used the same number of probability density evaluations, 3×10^5. Only the first two dimensions are shown here. 160

Figure 10.7 The plot shows a simulated image used in the Bayesian object detection exercise. There are 8 circular objects included here. As the objects are hardly visible due to the background noise their centres are marked with red stars. 164

Figure 10.8 The plot compares the results of 6 samplers, namely Sequential Monte Carlo (SMC, red), Affine-Invariant Ensemble Sampling (AIES, yellow), Differential Evolution Markov Chain (DEMC, purple), Kernel Density Estimate Metropolis (KM, orange), Ensemble Slice Sampling using the differential move (ESS, green), and Ensemble Slice Sampling using the global move (ESS, blue). The target distribution is a 10−dimensional Gaussian Mixture. The figure shows the 1D marginal distribution for the first parameter of the 10. 167

Figure 10.9 The plot compares the results of 6 samplers, namely Sequential Monte Carlo (SMC, red), Affine-Invariant Ensemble Sampling (AIES, yellow), Differential Evolution Markov Chain (DEMC, purple), Kernel Density Estimate Metropolis (KM, orange), Ensemble Slice Sampling using the differential move (ESS, green), and Ensemble Slice Sampling using the global move (ESS, blue). The target distribution is a 50−dimensional Gaussian Mixture.The figure shows the 1D marginal distribution for the first parameter of the 50. 168

Figure 10.10 The plot shows the adaptation of the length scale μ as a function of the number of iterations and starting from a wide range of initial values. Each trace is an independent run and the y-axis shows the value of μ divided by the final value of μ. The target distribution in this example is a 20−dimensional correlated normal distribution. Starting from larger μ values leads to significantly faster adaptation. 169

Figure 10.11 The plot shows the time t_f required for ESS to complete a pre-specified number of iterations as a function of the ratio of the number of available CPUs n_{CPUs} to the total number of walkers n_{Walkers}. The results are normalised with respect to the single CPU case t_1. The method scales as $\mathcal{O}(1/n_{\text{CPUs}})$ as long as $n_{\text{CPUs}} \leq n_{\text{Walkers}}/2$ (dashed line). The shaded areas show the $2-\sigma$ intervals. 170

Figure 11.1 Illustration of the univariate slice sampling update. Given the current sample x_0, a value y_0 is uniformly sampled along the vertical slice $(0, f(x_0))$ (dashed line) thus defining the initial point (blue). An interval (L, R) is uniformly positioned horizontally around (x_0, y_0) and it is expanded in steps of size $R - L$ until both its ends are outside the slice. The new sample is generated by repeatedly sampling (uniformly) from the interval (L', R') until a sample (green star) is found inside the slice. Samples outside of the slice (red star) are rejected and they are instead used to shrink (L', R'). 178

Figure 11.2 The figure illustrates the differential move in the context of Ensemble Slice Sampling. The walker X_k to be updated is shown in red. Two walkers, X_l and X_m, (blue) are uniformly selected from the complementary ensemble (grey). The approximate slice (dotted line) is constructed parallel to the two walkers X_l and X_m using the stepping-out procedure. The new position Y (green) of X_k is sampled using the shrinking procedure along the approximate slice. . . 180

Figure 11.3 The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of a normal (Gaussian) target distribution in 10, 25 and 50 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 50-dimensional case. 183

Figure 11.4 This figure shows the distribution of step sizes of walkers for the three different samplers in the case of a normal (Gaussian) target distribution in $D = 50$. It is important to note here that both `emcee` algorithms exhibit a peak at zero separation; `zeus` on the other hand does not due to its non-rejection nature. 184

Figure 11.5 The figure shows numerical estimates of the integrated autocorrelation time (number of steps along a chain required to obtain an independent sample; left panel), the effective sample size (percentage of effectively independent samples in a chain; middle panel), and the sampling efficiency (i.e. effective sample size per model evaluation; right panel) for a normal target distribution and varying number of dimensions. The number of walkers was set to $4 \times D$ for `zeus` and $16 \times D$ for `emcee`, this was the optimal choice (i.e. the one maximising the efficiency for the given dimensionality) for each sampler. `zeus` and `emcee/DEMC` exhibit linear scaling of the autocorrelation time with the number of dimensions whereas `emcee/AIES` scales exponentially. . . . 185

Figure 11.6 The figure shows the computational cost until convergence is reached in terms of the number of model evaluations for the different ensemble samplers for a highly correlated 25−dimensional normal distribution. The left panel shows the computational cost for a single walker. From this we can see that the cost for a single walker decreases as we increase the number of walkers until it reaches a plateau. The high computational cost for low numbers of walkers can be attributed to the low variety or sparsity of possible proposals; this is significantly higher for `emcee`/AIES. The right panel takes into account the linear scaling of the total computational cost as we increase the number of walkers and shows the total computational cost for the whole ensemble until it converges. 187

Figure 11.7 The figure shows the number of possible directions along which `zeus` and `emcee`/AIES can propose new samples as a function of the number of walkers in the complementary ensemble. `emcee`/DEMC exhibits the same number of proposals as `zeus` and it is not plotted here. `zeus` has a much higher variety of possible directions compared to `emcee`/AIES for any given number of walkers, assuming that that number is greater than 2. 189

Figure 11.8 The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of the ring target distribution in $2, 10$ and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. One can notice here that in 10 and 25 dimensions both `emcee` methods mix very slowly. In the 25-dimensional case almost all of `emcee`/DEMC's walkers are unable to move and the autocorrelation time is effectively infinite. 190

Figure 11.9 This figure shows the distribution of step sizes of walkers for the three different samplers in the case of a ring target distribution in $D = 25$. It is important to note here that both `emcee` algorithms exhibit a peak at zero separation; `zeus` on the other hand does not. The existence of the zero-peak in `emcee` is due to the high number of rejected proposals (i.e. low acceptance rate). 191

Figure 11.10 The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of a two-component Gaussian mixture target distribution in $2, 10$ and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. Whereas all three samplers make valid within-mode proposals, it is only `zeus` that manages to perform between-mode jumps and thus sample correctly from the target distribution in the 10 and 25-dimensional cases. Between-mode jumps are paramount in order to distribute the probability mass correctly between different modes. . . . 192

Figure 11.11 This figure shows the distribution of step sizes of walkers for the three different samplers in the case of a two-component Gaussian mixture target distribution in $D = 25$. It is important to note here that both `emcee` algorithms exhibit a peak at zero separation; `zeus` on the other hand does not due to its non-rejection basis. 193

Figure 11.12 The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of the Student's *t*-distribution with 2 degrees of freedom in $2, 10$ and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. 194

Figure 11.13 This figure shows the distribution of step sizes of walkers for the three different samplers in the case of the Student's *t*-distribution with 2 degrees of freedom in $D = 25$. `zeus` and `emcee`/AIES exhibit similar distributions whereas `emcee`/DEMC performs shorter steps. 194

Figure 11.14 The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of the truncated normal distribution in $2, 10$ and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. `zeus` exhibits the least amount of bias near the hard boundary at zero compared to `emcee`/AIES and `emcee`/DEMC. 195

Figure 11.15 This figure shows the distribution of step sizes of walkers for the three different samplers in the case of the truncated normal distribution in $D = 25$. `zeus` and `emcee`/AIES exhibit similar distribution whereas `emcee`/DEMC performs shorter steps. 196

Figure 11.16 A corner plot showing the 1-D and 2-D marginalised posteriors for the 22-parameter Baryon Acoustic Oscillation model as produced by the three different ensemble MCMC methods. 199

Figure 11.17 A corner plot showing the 1-D and 2-D marginalised posteriors for the 14-parameter radial velocity model as produced by the three different ensemble MCMC methods. 200

Figure 12.1 Illustration of the inference scheme of a *Masked Autoregressive Flow* (MAF). The arrows show the conditional dependence of the variables as well as the action of the *Masked Autoregressive Density Estimation* (MADE) layer. The input target probability density (top) is mapped into a multivariate normal distribution (bottom). A sequence of MADE layers and permutations is repeated multiple times in order to increase the flexibility of the flow. 209

Figure 12.2 The figure illustrates the effect of preconditioning on the *Rosenbrock* distribution. The right panel shows samples (blue) from the true correlated distribution and the left panel shows samples (blue) from the preconditioned/transformed one. The orange samples in the left panel are drawn from a symmetric normal proposal distribution centred around the green point u_0 and they correspond to the respective orange points in the right panel. In other words, the transformed samples from the simple proposal in the left panel correspond to samples that capture the local geometry of the true target distribution in the right panel. 210

Figure 12.3 Parallelization of PMC compared to nested sampling. PMC (blue) exhibits linear speedup compared to the sub–linear one achieved by NS (orange). 214

Figure 12.4 Illustration of the 1–dimensional and 2–dimensional marginal posteriors for the first three out of 20 parameters of the *Rosenbrock* distribution. The figure shows the $1-\sigma$ and $2-\sigma$ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached. 216

Figure 12.5 Illustration of the 1–dimensional and 2–dimensional marginal posteriors for the first three out of 50 parameters of the two–component Gaussian mixture distribution. The figure shows the 1–σ and 2–σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached. 217

Figure 12.6 Illustration of the 1–dimensional and 2–dimensional marginal posteriors for the 12 parameters of the primordial features posterior. The figure shows the 1–σ and 2–σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached. 218

Figure 12.7 Illustration of the 1–dimensional and 2–dimensional marginal posteriors for the 13 parameters of the gravitational waves posterior. The figure shows the 1–σ and 2–σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached. 220

Figure 12.8 Comparison of the first two parameters of samples generated using PMC (*blue*) and IMH–SMC (*orange*) for the 20–D Rosenbrock target distribution. PMC produces representative samples, whereas IMH–SMC does not. 222

Figure 12.9 Comparison of the first two parameters of samples generated using PMC (*blue*) and IMH–SMC (*orange*) for the 50–D two–component Gaussian mixture target distribution. PMC produces representative samples, whereas IMH–SMC does not. 223

Figure 13.1 Logo of pocoMC. 227

LIST OF TABLES

Table 1 The typical radius r_{peak} as function of the number of dimensions D. 66

Table 2 The table shows a comparison of the optimally tuned Metropolis, Standard Slice, and Ensemble Slice Sampling with the differential move (ESS-D) and the Gaussian move (ESS-G) respectively in terms of the integrated autocorrelation time (IAT) and the number of effective samples per evaluation of the probability density (efficiency) multiplied by 10^4. These metrics are formally defined in Appendix 10.7. The target distributions are the 50−dimensional autoregressive process of order 1 and the 25−dimensional correlated funnel distribution. The total number of iterations was set to 10^7. 158

Table 3 The table shows a comparison of the Affine Invariant Ensemble Sampling (AIES), Differential Evolution Markov Chain (DEMC), and Ensemble Slice Sampling methods in terms of the integrated autocorrelation time (IAT) and the number of effective samples per evaluation of the probability density (efficiency) multiplied by 10^5. These metrics are formally defined in Appendix 10.7. The target distributions are the 16−dimensional ring distribution, the 10−dimensional Gaussian shells distribution and the 13−dimensional hierarchical Gaussian process regression distribution. In all cases the total number of iterations was set to 10^7. It should be noted that in the case of the Gaussian shells the global move was used instead of the differential move. 162

Table 4 The table shows a comparison of `emcee`/AIES, `emcee` /DEMC and `zeus` in terms of the expected squared jump distance (ESJD; higher is better) for the five toy examples i.e. 50-D normal distribution, 25-D ring distribution, 25-D Gaussian mixture, 25-D Student's t-distribution, and 25-D truncated normal distribution. 186

Table 5 The table shows a comparison of `emcee`/AIES, `emcee`/DEMC and `zeus` in terms of the inverse efficiency (i.e. reciprocal of the number of independent samples per model evaluation or the autocorrelation time estimate times the average number of model evaluations per iteration per walker), the convergence cost (i.e. number of model evaluations until convergence) and the convergence fraction (i.e. fraction of converged chains for given maximum number of model evaluations). 198
Table 6 The table shows the default values for the hyperparameters of PMC. 213
Table 7 The table shows a comparison of PMC, NS, and SMC in terms of their computational cost (i.e. total number of model evaluations until convergence). 215

LIST OF ALGORITHMS

1 Numerical reference prior 41
2 Metropolis–Hastings 85
3 Adaptive Metropolis 88
4 Gibbs sampler 90
5 Metropolis–within–Gibbs sampler 91
6 Simulated annealing 95
7 Slice sampling 98
8 Hamiltonian Monte Carlo 102
9 Gaussian ensemble 105
10 Affine–invariant stretch move 107
11 Differential evolution 109
12 Parallel tempering 113
13 Sequential Monte Carlo 129
14 Nested sampling 138
15 Function to tune the length scale μ. 147
16 Function to return a differential move direction vector. 148
17 Function to return a Gaussian Move direction vector. 150
18 Function to return a global move direction vector. 153
19 Single Iteration t of Ensemble Slice Sampling. 156
20 Preconditioned Monte Carlo 212

Part I

BAYESIAN INFERENCE

1 PROBABILITY THEORY

Science is more than a body of knowledge; it is a way of thinking.
The method of science, as stodgy and grumpy as it may seem,
is far more important than the findings of science.

— Carl Sagan

This chapter introduces the basic principles of Bayesian inference and presents its fundamental ideas and distinctive features.

1.1 THE GOAL OF SCIENCE

The key goal of science is to distil the patterns of nature into mathematical language and call them physical laws. To this end, science relies on a formal way of thinking and interrogating nature, asking the right questions, interpreting observations, and updating its beliefs and hypotheses in the light of new evidence. This way of thinking, inherent in all scientific pursuits seems to be deeply connected to the mathematical notion of probability.

Science proceeds towards this elusive target with careful steps following the scientific method. The latter is often illustrated as a loop. Hypotheses are proposed and models quantifying certain aspects of those hypotheses are developed. The hypotheses give rise to predictions, in a process called *deductive inference*, to be tested against experimental data. Unfortunately, as the information that we extract from nature in the form of data is always incomplete and uncertain, testing our hypotheses by comparing our model predictions to the experimental data requires us to reason in the presence of uncertainty. We thus rely on *plausible inference*, that is, the process of inferring the truth of our theories about the cosmos on the basis of incomplete and uncertain information.

Scientific statements about the physical world are uncertain by necessity. No amount of new information will ever be enough to validate or disprove a hypothesis. Furthermore, our models, despite our best intentions, are often simpler than the natural processes which they attempt to capture. Our best hope is thus to accept the existence of this inherent and unavoidable uncertainty and instead try to quantify the plausibility of our statements about the cosmos. Assessing the plausibility of scientific theories is the subject of probability theory.

1.2 THE NOTION OF PROBABILITY

There are few concepts in science and mathematics as controversial, with their meaning so contested during the centuries, as the notion of probability. Three centuries ago people started seriously thinking about how to best make decisions and reason in the face of uncertainty. Perhaps, the first to formally articulate this problem was *Jacob Bernoulli* in his seminal work *Ars Conjectandi* published in 1713.

The answer to Bernoulli's question was provided by *Reverend Thomas Bayes*, in an essay named *An Essay towards solving a Problem in the Doctrine of Chances*, published posthumously by his friend *Richard Price* in 1763. The paper included theorems on *conditional probability* which formed the basis of what we now call *Bayes' theorem*. The discovery of the latter is actually due to *Laplace*, who not only developed, extended and clarified probability theory, but also applied it successfully to a plethora of problems in astronomy, medicine, and economics.

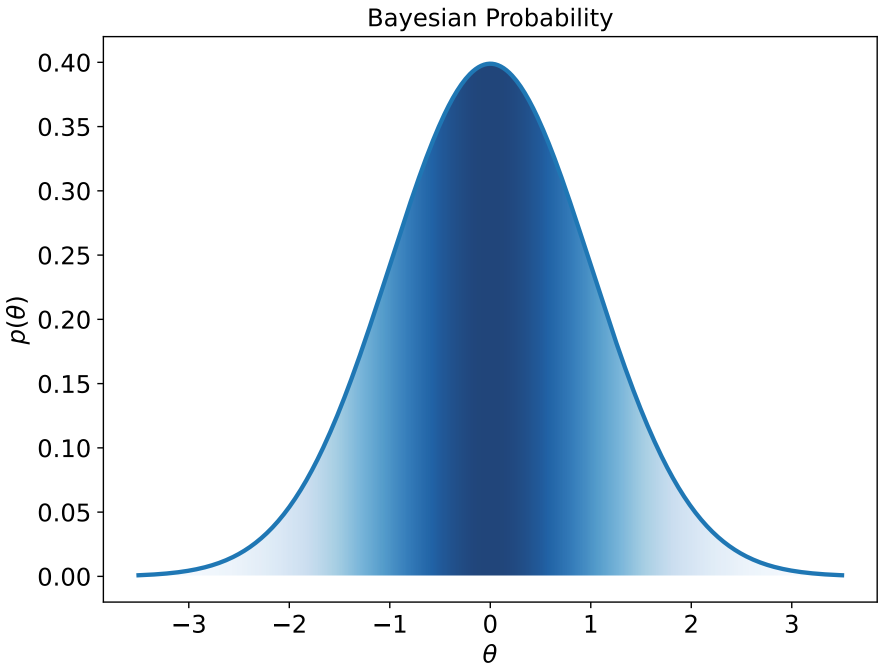

Figure 1.1: In the Bayesian interpretation of probability, the degree of belief is distributed and the variable has a specific (unknown) fixed value.

Despite Laplace's indisputable empirical success, his theory was rejected by scholars soon after his death. Their problem with Laplace's probability theory was one of interpretation. For pioneers such as Bernoulli, Bayes, and Laplace, probability represented a degree–of–belief or plausibility of various hypotheses or statements based on the available evidence and prior knowledge. To 19th century scholars though, this definition, or interpretation of probability, seemed too subjective and vague. For this reason, they redefined probability to mean the *long–run relative frequency* with which an event occurs, given infinite trials. Since frequency can be measured experimentally, probability was then seen as an objective measure for dealing with *randomness* and chance.

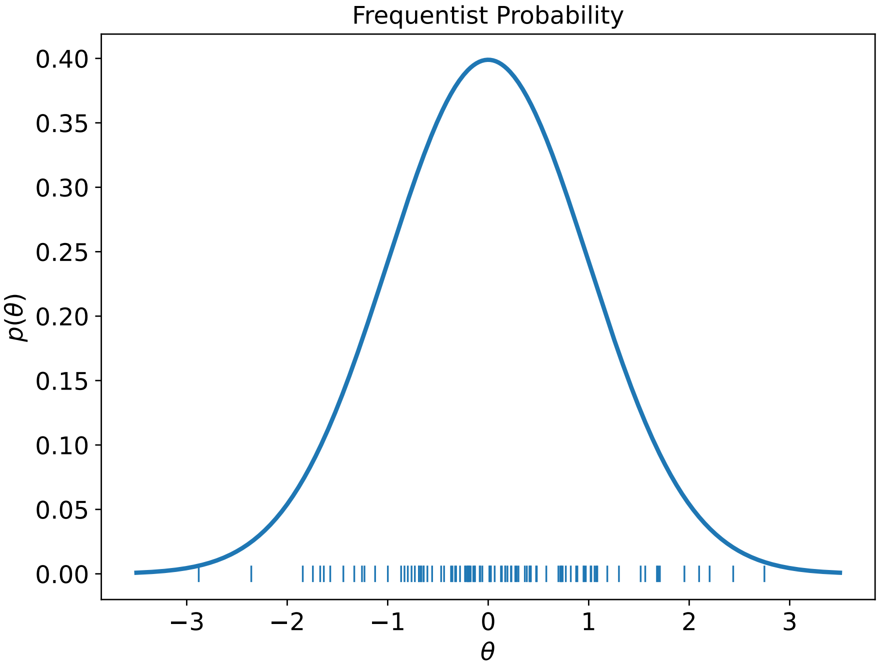

Figure 1.2: In the frequentist interpretation of probability, the value of the variable itself is distributed along different experiments.

Although the *frequentist* interpretation of probability seems more objective, its range of applicability and validity is substantially more limited. For example, Laplace used *Bayes' theorem* and his probability theory to estimate the mass of Saturn. He computed the *posterior probability density function (pdf)* $p(M|d)$, that is, the probability that Saturn has a mass M given the available data d and model assumptions (e.g. validity of celestial mechanics). An illustration of this posterior pdf is shown in Figure 1.3, in which the value M as the peak of the density corresponds to the most probable value for the mass of Saturn which also coincides with the mean value. M_{min} and M_{max} denote the values of the mass that deviate by 1% from the mean value M, and the shaded area between them is the probability that the mass of Saturn is between the values of M_{min} and M_{max}. Apart from the most probable value M, Laplace estimated that the probability (given by the area) that the real mass of Saturn is between these limits is $11327/11328 = 0.9999117$. In particular, he wrote “applying to them my formulae of probability I find that it is a bet of 11,000 against one that the error of this result is not 1/100 of its value”. Today, almost two centuries after this statement was made, Laplace would have won this bet as the current best estimate of Saturn's mass differs only by 0.5% from his.

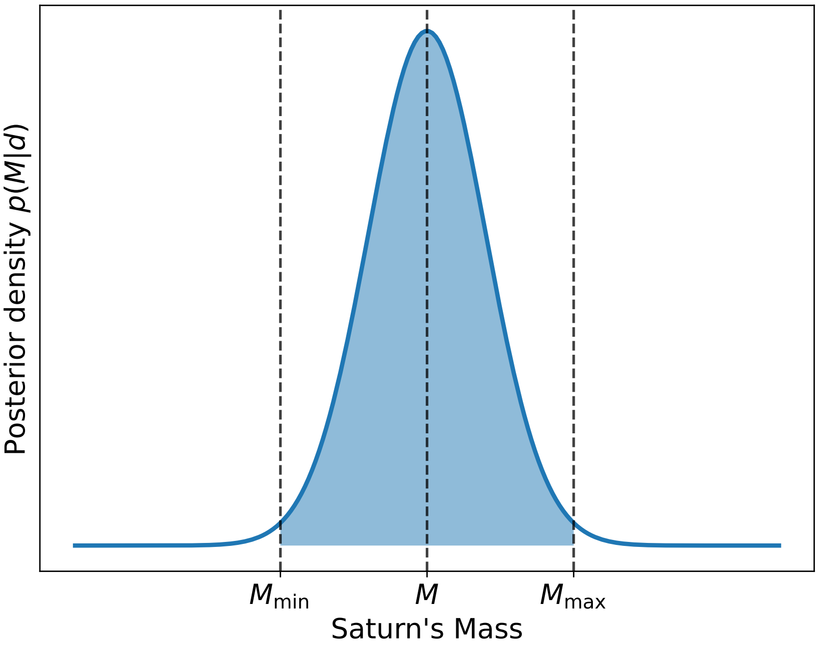

Figure 1.3: Illustration of posterior probability density function of Saturn's mass. M corresponds to the most probable value for the mass at the peak of the density. M_{min} and M_{max} denote the values of the mass that deviate by 1% by the mean value M. The shaded area signifies the probability that the mass of Saturn is between the values of M_{min} and M_{max}.

However, according to the frequentist interpretation of probability one is not allowed to use probability theory to tackle this problem, as the mass of Saturn is a fixed constant and not a *random variable* that follows a frequency distribution. If we were to interpret Laplace's results from a frequentist perspective we would have to imagine an infinitely large *ensemble* of universes in which everything remains the same but the mass of a single planet. Although one has the liberty to make any kind of syllogisms in order to find a solution, having to seek a frequency interpretation for every problem can be cumbersome and at risk of detaching any notion of intuition from the physical problem.

Faced with the realisation that the frequentist interpretation of probability does not allow one to tackle most scientific questions, the new subject of *statistics* was invented. For instance, in the problem of estimation of Saturn's mass since the mass is not a random variable, one has to create a function, called a statistic, that relates the data to the mass. Since the data are subject to random noise, so does the statistic. One is then free to apply the standard techniques to the statistic. However, the choice and construction of the statistic are often neither clear nor principled. There is no unifying principle relating the various techniques and practices used in order to choose which statistic is more appropriate for a given task. Historically, this lack of a common framework resulted in the creation of a large number of alternative schools of thought of frequentist statistics. Most notably, great statisticians such as *Neyman*, *Pearson* and *Fisher* were responsible for promoting different approaches.

Early in the 20th century something changed though, Sir *Harold Jeffreys* rediscovered Laplace's Bayesian probability theory, and in 1930*s* he explained and presented it in greater detail and more clearly than Laplace ever did (Jeffreys, 1998). Although apparently not enough to convince the most militant proponents of orthodox frequentist schools of the merits of probability theory, Jeffreys' work was the triggering event that acted as a catalyst for a change that lasted until the end of the 20th and beginning of the 21th century.

In 1946, *Richard Cox* attempted to end the debate by approaching the problem of *plausible inference* from a different perspective, that of its logical consistency (Cox, 1946). Starting by the assumption that we can order different statements based on their plausibility, by assigning a real number to each statement representing how plausible it is, proved that for a calculus of plausible inference to be consistent (i.e. in the sense that if two different methods are permitted they should give the same results), it has to obey the rules of probability theory as defined by Laplace and Jeffreys. The work of Cox is of paramount importance as he effectively showed that any system of plausible inference that is logically consistent has to reduce to Bayesian inference.

By the last decade of the 20th century, progress in computer technology and algorithms for probabilistic computation reached and surpassed the level of maturity required for the widespread application of Bayesian inference in most fields of physical science. Therefore, it is no surprise that the principles and methods of Bayesian probability theory have now become an integral and indispensable part of modern science. In the end, Laplace was right: "It is remarkable that a science which began with the consideration of games of chance should have become the most important object of human knowledge".

1.3 BAYES' THEOREM

1.3.1 Rules of probability

Any statement in probability theory can be derived by starting from the Laplace–Jeffreys sum and product rule given below. From these two formulas, expressions such as the "or" rule, the marginalisation rule and Bayes' theorem follow easily.

The “sum” rule

The “sum” rule expresses the relation between the probabilities of two mutually exclusive statements A and $\bar{A}$,

$$p(A|B) + p(\bar{A}|B) = 1\,, \tag{1.1}$$

where $p(A|B)$ represents the plausibility (probability) of A being true given that B is true and $\bar{A}$ simply means the opposite of A or that A is false.

The “product” rule

The “product” rule provides a way to compute the joint probability

$$p(A, B|C) = p(A|B, C)p(B|C) = p(B|A, C)p(A|C)\,. \tag{1.2}$$

of both A and B being true given that C is true.

The “or” rule

For instance, the “or” rule that expresses the probability that either A or B is true, given that C is true, can be written as

$$p(A \cup B|C) = p(A|C) + p(B|C) - p(A \cap B|C)\,, \tag{1.3}$$

follows easily, where $p(A \cap B|C)$ is simply another notation for the joint probability $p(A, B|C)$ for both events A and B being true given that C is also.

The marginalisation rule

Another useful probability rule is the *marginalisation rule* for discrete probability distributions,

$$p(A|C) = \sum_i p(A, B_i|C) = \sum_i p(A|B_i, C)p(B_i|C)\,, \tag{1.4}$$

and for continuous probability distributions,

$$p(A|C) = \int p(A, B|C)dB = \int p(A|B, C)p(B|C)dB\,. \tag{1.5}$$

Equation 1.4 is straightforward to prove starting from the sum rule of equation 1.1, extended to multiple mutually–exclusive events

$$\sum_i p(A_i|B) = 1\,. \tag{1.6}$$

Therefore, starting from equation 1.6 we have

$$\begin{aligned} \sum_i p(A, B_i|C) &= \sum_i p(B_i|A, C)p(A|C) \\ &= p(A|C) \sum_i p(B_i|A, C) = p(A|C)\,. \end{aligned} \tag{1.7}$$

Bayes' theorem

Arguably the most useful equation that can be derived is the so–called *Bayes' theorem* (Bayes, 1763)

$$p(A|B,C) = \frac{p(B|A,C)p(A|C)}{p(B|C)}\,, \tag{1.8}$$

that follows directly from equation 1.2.

1.3.2 Updating degrees of belief

Although Bayes' theorem is a simple identity that holds for any statements A, B, and C, it also has a special role in the context of plausible inference. In particular, if we set $A \longleftarrow \theta$ the parameters of a physical model, $B \longleftarrow d$ the experimental data, and $C \longleftarrow \mathcal{M}$ the physical model that also includes all assumptions made in an analysis, we get

$$p(\theta|d,\mathcal{M}) = \frac{p(d|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(d|\mathcal{M})}\,. \tag{1.9}$$

The importance of equation 1.9 for scientific inference is apparent if we examine each one of the constituent components individually.

Posterior probability distribution – $p(\theta|d,\mathcal{M})$

This is the probability distribution of the parameters θ, given the data d and the modelling assumptions $\mathcal{M}$. The posterior is often what we are aspiring to approximate in a *parameter estimation* analysis.

Prior probability distribution – $p(\theta|\mathcal{M})$

This probability distribution quantifies any knowledge about the possible values of the parameters θ prior to seeing the data d. We have a whole chapter dedicated to the choice of the prior distribution.

Likelihood function and sampling distribution – $p(d|\theta,\mathcal{M})$

The likelihood function is a key component of Bayes' theorem that plays a very important role, that of being the conduit that explains how the transition from prior to posterior takes place. Before we understand the role and properties of the likelihood function we first need to look into the so–called *sampling distribution.*

The *sampling distribution* $p(d|\theta)$ expresses the probability distribution of the data d given the values of the model parameters θ. In this picture, the parameters θ are known and fixed and $p(d|\theta)$ is a distribution over the data. If instead, we know the data d and fix them to a specific value of set of values, and we let θ vary as a free parameter of a set of free parameters, then $p(d|\theta)$ is called the *likelihood function*.

The likelihood function is often symbolised as $\mathcal{L}(\theta) = p(d|\theta)$ to denote that it is a function of parameters θ and not a probability distribution over the data d. This is very important as statements such as "the likelihood of the data" are meaningless and completely miss the point of the likelihood. The likelihood function shows how well the different sampling distributions $p(d|\theta)$, parameterised by θ, predict the observed data.

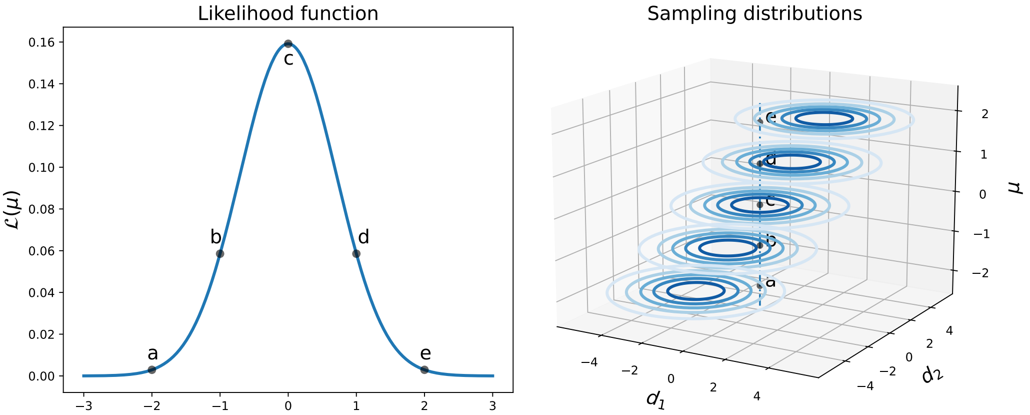

Figure 1.4: Left: Likelihood function $\mathcal{L}(\mu)$ for the family of sampling distributions as shown on the right. The marked letters indicate the points in which the observed data intersect the sampling distributions. Right: The different groups of contours illustrate the sampling distribution $p(d_1, d_2|\mu)$ for different values of μ, ranging from -2 to 2. The vertical line corresponds to the observed data $(d_1, d_2) = (0, 0)$ and the marked letters indicate the points in which the observed data intersect the sampling distributions.

To make this more apparent, let us consider a simple example. Let us assume that we have a family of sampling distributions $p(d_1, d_2|\mu)$ for the two dimensional data $d = (d_1, d_2)$, parameterised by a single parameter μ. An example of such a family of sampling distributions for $\mu \in [-2, 2]$ is shown in Figure 1.4 on the right. One can see that different values of μ correspond to different sampling distributions. In order to get a likelihood function $\mathcal{L}(\mu)$ from this family of sampling distributions, we need to specify some observed data for each member of the family. Without loss of generality, we choose the data to be simply $(d_1, d_2) = (0, 0)$. The data are indicated by a vertical line in the plot that intersects all members of the family of sampling distributions. The points of intersection, marked with letters a to e in the same plot, can either be in low or high probability regions of the respective sampling distributions. If we now monitor the value of the probability at the intersection points and plot this as a function of the parameter μ we get the likelihood function shown in the same figure on the left.

In other words, although related, the notion of likelihood is really different from that of probability in the sense that it expresses the relative capacity of different sampling distributions, belonging however to the same family, to predict and explain the observed data. *Sir Ronald Aylmer Fisher* wrote in 1922 about the difference between probability and likelihood, albeit in the frequentist tradition,

> *If we need a word to characterise this relative property of different values of p, I suggest that we may speak without confusion of the likelihood of one value of p being thrice the likelihood of another, bearing always in mind that likelihood is not here used loosely as a synonym of probability, but simply to express the relative frequencies with which such values of the hypothetical quantity p would in fact yield the observed sample. [...] Likelihood also differs from probability in that it is a differential element, and is incapable of being integrated: it is assigned to a particular point of the range of variation, not to a particular element.*

Model evidence – $p(d|\mathcal{M})$

Finally, the *model evidence* is a single real number that expresses the probability of observing the data d given the model $\mathcal{M}$ and acts as a normalisation constant for the posterior

$$p(d|\mathcal{M}) = \int p(d|\theta,\mathcal{M})p(\theta|\mathcal{M})d\theta\,, \tag{1.10}$$

such that $\int p(\theta|d,\mathcal{M})d\theta = 1$. The model evidence is often referred to as the *marginal likelihood* due to the way it is represented as an integral. Its role in the task of *model comparison* is great and it will be discussed in great length in the following chapters.

Schematically, we can summarise the above description of Bayes' theorem as

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}\,. \tag{1.11}$$

In essence, Bayes' theorem in the form of equations 1.9 and 1.11 is a recipe for updating our degree of belief when new information, in the form of data, becomes available. The factor that upweights or downweights the prior $p(\theta|\mathcal{M})$ is the likelihood–to–evidence ratio $p(d|\theta,\mathcal{M})/p(d|\mathcal{M})$, also known as the *predictive updating factor.* Keynes called this ratio the *coefficient of influence* as it is this that determines how the prior is transformed into the posterior (Keynes, 1921). To better understand this, remember that the model evidence in the denominator is simply the expectation value of the likelihood over the prior probability distribution, so intuitively it expresses some sort of mean value of the likelihood. From this perspective, the predictive updating factor is simply the ratio of the likelihood to its mean value. This means

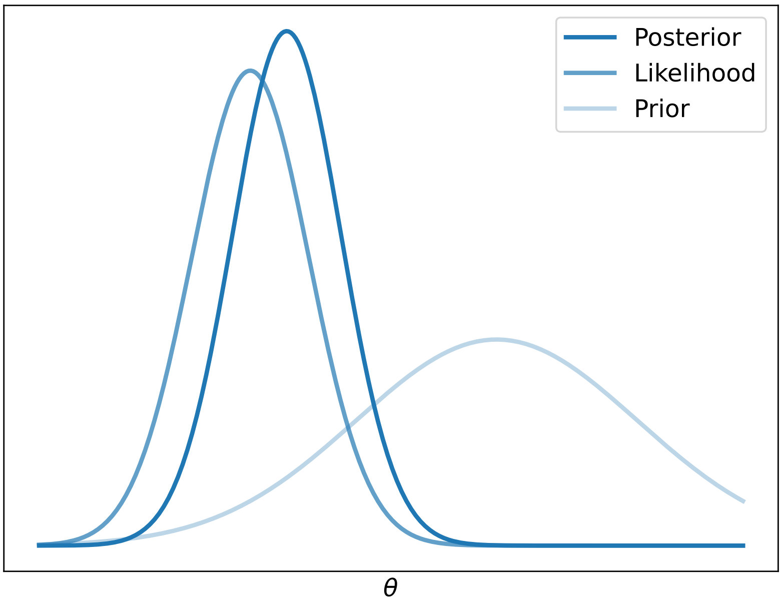

Figure 1.5: Illustration of Bayes' theorem. The posterior probability is proportional to the product of the likelihood function and prior probability.

that the prior will be upweighted for those values of θ that the likelihood is greater than its mean value and downweighted otherwise.

1.4 REPRESENTING PROBABILITY DISTRIBUTIONS

Before we move on to any examples, it is important to explain how we represent probability distributions in practice. In general, there are two ways that we can do this, each one with different advantages and disadvantages.

1.4.1 Function representation

Probability mass function

When the parameter space Θ is discrete, then a probability distribution can be represented as a *probability mass function (pmf)* $p(\theta)$ that assigns a probability value to each element of space, $p : \Theta \longrightarrow [0, 1]$. Any pmf has to obey the rule of total probability, that is

$$\sum_{\theta \in \Theta} p(\theta) = 1 \,. \tag{1.12}$$

The probability of any composite event $\Theta' \subset \Theta$ can be computed as

$$p(\Theta') = \sum_{\theta \in \Theta'} p(\theta) \,. \tag{1.13}$$

Finally, we can compute any expectation value

$$\mathbb{E}_p[f] = \sum_{\theta \subset \Theta} f(\theta) p(\theta) \tag{1.14}$$

for any function $f(\theta)$. Common examples of expectation values include the mean $\mu = \mathbb{E}_p[\theta]$ and the variance $\sigma^2 = \mathbb{E}_p[(\theta - \mu)^2]$.

Probability density function

When the parameter space $\Theta \subseteq \mathbb{R}^D$ is continuous, then a probability distribution can be represented as a *probability density function (pdf)* $p(\theta)$ that assigns a probability density value to each element of space, $p : \Theta \longrightarrow \mathbb{R}$. Any pdf has to obey the rule of total probability, that is

$$\int_{\theta \in \Theta} p(\theta) d\theta = 1 \,. \tag{1.15}$$

Unlike a pmf, a pdf expresses probability density and thus it has to be integrated first to give probabilities. The probability of $\Theta' \subseteq \Theta$ is then

$$p(\Theta') = \int_{\Theta' \subseteq \Theta} p(\theta) d\theta \,. \tag{1.16}$$

For instance, in 1–D we can compute the probability that $A \leq \theta \leq B$ as

$$p(A \leq \theta \leq B) = \int_A^B p(\theta) d\theta \,, \tag{1.17}$$

as the area below the graph of $p(\theta)$ and between A and B. Similarly, an expectation value can be computed as

$$\mathbb{E}_p[f] = \int_\Theta f(\theta) p(\theta) d\theta \,. \tag{1.18}$$

A crucial difference between probability mass functions and probability densities is that the latter do not transform quite as trivially under parameter transformations $g : \Theta \longrightarrow \Phi$. The origin of this complication is that the differential volume $d\theta$ over which we integrate will generally change under such a transformation, and density functions have to change in the opposite way to compensate and ensure that probability is conserved. This change in volume is quantified by the absolute value of the determinant of the Jacobian matrix

$$J_{ij} = \frac{\partial g_i}{\partial \theta_j} \,, \tag{1.19}$$

where $\phi = g(\theta)$ is the parameter transformation. Thus, the probability density $p(\theta)$ generally transforms as

$$p(\theta) = p(\phi) |\det J| \,. \tag{1.20}$$

1.4.2 Sample representation

One of the inherent difficulties in the density function representation of probability distributions is that the computation of expectation values is often intractable as no closed–form solution exists for most applications. An alternative way of representing probability distributions is using a collection of points $S = \{\theta_1, \theta_1, \dots, \theta_n\}$ in the parameter space Θ, called samples. The generation of samples for a given probability distribution will be the subject of discussion for most of this thesis. For now, it suffices to say that any probability distribution admits a sample representation

$$\theta_i \sim p(\theta)\,, \tag{1.21}$$

such that the empirical estimate

$$\hat{f}_p = \frac{1}{n}\sum_{i=1}^{n} f(\theta_i)\,, \tag{1.22}$$

asymptotically approaches the expectation value $\mathbb{E}_p[f]$ as $n \longrightarrow \infty$.

1.5 ASYMPTOTIC BEHAVIOUR

Let us now turn our attention to the question of the form of the posterior distribution in the limit of infinite data. Understanding the asymptotic behaviour of the posterior when the sample size is large is important for a number of reasons. First, there is practical utility as asymptotic results are often good first–order approximations. Second, as we will discuss in the following chapter, the asymptotic form of the posterior distribution can be utilised to automate the construction of prior distributions. Finally, the *Bernstein–von Mises* theorem, which describes the asymptotic behaviour of the posterior in many cases, allows us to link Bayesian inference to frequentist results.

1.5.1 Bernstein–von Mises theorem

When the number n of observations tends to infinity, the posterior distribution of a smooth finite–dimensional model approaches a normal distribution. In particular, if we denote $d^{(n)} = \{d_1, \dots, d_n\}$ the set of n observations or data, then the posterior $p(\theta|d^{(n)})$ concentrates around the *maximum likelihood estimate (MLE)*:

$$\hat{\theta}_n = \arg\max_{\theta} p(d^{(n)}|\theta)\,. \tag{1.23}$$

Moreover, MLE is a consistent estimator which means that in the limit of infinite sample size (i.e. $n \longrightarrow \infty$), $\hat{\theta}_n$ converges to θ_t, that is, the true value of the parameter vector. In other words, the asymptotic posterior is centred on the true parameter value θ_t. The precision matrix (i.e. inverse covariance matrix) is equal to $n\mathcal{I}(\hat{\theta}_n)$, where the *Fisher information matrix* is defined as

$$\begin{aligned}\mathcal{I}(\theta) &= \mathbb{E}_{d\sim p(d|\theta)}\left[-\frac{\partial^2 \log p(d|\theta)}{\partial\theta_i\partial\theta_j}\right] \\ &= -\int p(d|\theta)\frac{\partial^2 \log p(d|\theta)}{\partial\theta_i\partial\theta_j}\,dd\,.\end{aligned} \tag{1.24}$$

In more mathematical terms, we can write down that

$$p(\theta|d^{(n)}) \longrightarrow \mathcal{N}(\theta|\hat{\theta}_n, n^{-1}\mathcal{I}^{-1}(\hat{\theta}_n))\,, \tag{1.25}$$

as $n \longrightarrow \infty$, where we use the notation $\mathcal{N}(\theta|\mu,\Sigma)$ to denote the Gaussian probability density function

$$\mathcal{N}(\theta|\mu,\Sigma) = (2\pi)^{-D/2}|\Sigma|^{-1/2}\exp\left\{-\frac{1}{2}(\theta-\mu)^T\Sigma^{-1}(\theta-\mu)\right\}\,, \tag{1.26}$$

with mean μ and covariance matrix Σ, where D is the number of components in the θ vector (i.e. dimensionality of parameter space). Although this result dates back to Laplace (1810), today it is known as the *Bernstein-von Mises* theorem (Van der Vaart, 2000).

One consequence of the above theorem, combined with the fact that the MLE asymptotically follows a normal distribution, allows us to interpret Bayesian *credible intervals* as frequentist *confidence intervals* in the limit of infinite data.

1.5.2 Heuristic argument

We will now offer an intuitive heuristic argument, rather than a rigorous proof, of the *Bernstein-von Mises* theorem. Let us begin by rewriting Bayes' theorem as

$$p(\theta|d^{(n)}) = \frac{\exp\left\{\log p(\theta) + \log p(d^{(n)}|\theta)\right\}}{P(d^{(n)})}\,, \tag{1.27}$$

where $\log p(\theta)$ is the log–prior and

$$\log p(d^{(n)}|\theta) = \sum_{i=1}^{n}\log p(d_i|\theta)\,, \tag{1.28}$$

is the log–likelihood function of *identically independently distributed (iid)* data, which readily follows from the fact that their sampling distributions are conditionally independent, meaning that

$$p(d^{(n)}|\theta) = \prod_{i=1}^{n} p(d_i|\theta)\,. \tag{1.29}$$

The next step is to Taylor–expand both the log–prior and the log–likelihood around their respective maxima. Starting with the log–prior, we can write

$$\log p(\theta) = \log p(\hat{\theta}_0) - \frac{1}{2}(\theta - \hat{\theta}_0)^T \Lambda_0(\hat{\theta}_0)(\theta - \hat{\theta}_0) + R_0\,, \tag{1.30}$$

where

$$\Lambda_0(\hat{\theta}_0) = \left(-\frac{\partial^2 \log p(\theta)}{\partial\theta_i \partial\theta_j} \right)\Bigg|_{\theta=\hat{\theta}_0} \tag{1.31}$$

and R_0 denotes any higher–order terms. Notice that since the expansion takes place around the prior maximum $\hat{\theta}_0$, there is no first–order term (i.e. the first derivative is equal to zero). Similarly, we can expand the log–likelihood around the MLE as follows:

$$\log p(d^{(n)}|\theta) = \log p(d^{(n)}|\hat{\theta}_n) - \frac{1}{2}(\theta - \hat{\theta}_n)^T \Lambda_n(\hat{\theta}_n)(\theta - \hat{\theta}_n) + R_n\,, \tag{1.32}$$

where

$$\Lambda_n(\hat{\theta}_n) = \left(-\frac{\partial^2 \log p(d^{(n)}|\theta)}{\partial\theta_i \partial\theta_j} \right)\Bigg|_{\theta=\hat{\theta}_n} = \left(-\sum_{\ell=1}^{n} \frac{\partial^2 \log p(d_\ell|\theta)}{\partial\theta_i \partial\theta_j} \right)\Bigg|_{\theta=\hat{\theta}_n} \tag{1.33}$$

and R_n denotes any terms beyond the second order.

Assuming that the prior and likelihood are sufficiently smooth such that R_0 and R_n can be safely ignored we can write equation 1.27 as

$$\begin{aligned} p(\theta|d^{(n)}) \propto \exp\Bigg\{ &-\frac{1}{2}(\theta - \hat{\theta}_0)^T \Lambda_0(\hat{\theta}_0)(\theta - \hat{\theta}_0) \\ &-\frac{1}{2}(\theta - \hat{\theta}_n)^T \Lambda_n(\hat{\theta}_n)(\theta - \hat{\theta}_n) \Bigg\} \\ \propto \exp&\left\{ -\frac{1}{2}(\theta - \tilde{\theta}_n)^T \tilde{\Lambda}_n(\tilde{\theta}_n)(\theta - \tilde{\theta}_n) \right\}, \end{aligned} \tag{1.34}$$

where $\tilde{\Lambda}_n = \Lambda_n + \Lambda_0$ and $\tilde{\theta}_n = \tilde{\Lambda}_n^{-1}(\Lambda_n\hat{\theta}_n + \Lambda_0\hat{\theta}_0)$. Comparing the above expression to equation 1.26, we find that the posterior has a Gaussian probability density function

$$p(\theta|d^{(n)}) = \mathcal{N}(\theta|\tilde{\theta}_n, \tilde{\Lambda}_n^{-1})\,. \tag{1.35}$$

In the limit that $n \longrightarrow \infty$, the sum in equation 1.33 completely dominates the calculation leading to $\tilde{\Lambda}_n \longrightarrow \Lambda_n$ and $\tilde{\theta}_n \longrightarrow \hat{\theta}_n$. This means that asymptotically

$$p(\theta|d^{(n)}) \longrightarrow \mathcal{N}(\theta|\hat{\theta}_n, \Lambda_n^{-1})\,. \tag{1.36}$$

Furthermore, according to the *law of large numbers*, which states that “the average of a large number of trials approaches the expectation value“, Λ_n as given by the sum in equation 1.33 is asymptotically equal to $n\mathcal{I}(\hat{\theta}_n)$. Therefore, we can write down that

$$p(\theta|d^{(n)}) \longrightarrow \mathcal{N}(\theta|\hat{\theta}_n, n^{-1}\mathcal{I}^{-1}(\hat{\theta}_n))\,, \tag{1.37}$$

which concludes our heuristic derivation.

1.6 ESTIMATING PARAMETERS

1.6.1 Coin–tossing experiment

Let us now consider a simple example of Bayesian parameter estimation. Suppose that we have a coin and we want to determine whether the coin is fair or not. A simple way to quantify the fairness of a coin is to introduce a *bias* parameter F such that $F = 1/2$ means that the coin is fair, whereas any other value in the range $0 \leq F \leq 1$ denotes that the coin is biased. $F = 0$ corresponds to a coin which always lands on *tails* and $F = 1$ to a one that always lands on *heads*. We can then divide the continuous range of F into a discrete number of propositions (e.g. $0 \leq F \leq 0.01$, $0.01 \leq F \leq 0.02$, etc.). Our state of knowledge about the fairness of the coin is summarised by our degree of belief, quantified as a probability, of each one of those intervals (e.g. $p(0 \leq F \leq 0.01)$, $p(0.01 \leq F \leq 0.02)$, etc.).

In order to collect some data we just have to toss the coin a few times and monitor the number of times H the coin lands on heads as well as the total number of trials N. The number of times that the coin lands on tails is simply $N - H$. Furthermore, to better understand the iterative nature of Bayes' theorem for updating our degree of belief, we will keep not only the final outcome of the experiment (i.e. the total number H that the coin landed on heads in N trials) but also all the intermediate values.

Since our aim is to estimate the posterior distribution $p(F|H, N)$, that is, the probability distribution of F given the observed data in terms of the number of heads H and the number of trials N, we need to define all the components that enter Bayes's theorem. Starting with the prior probability distribution $p(F)$ we will use two choices in order to demonstrate their effect on the posterior. The first choice of prior is to be agnostic, before seeing the data, about the fairness of the coin and thus assume that intervals of the same size in the range $0 \leq F \leq 1$ are equally probable. This is quantified by the uniform probability density function

$$p(F) = \begin{cases} 1, & \text{if } 0 \leq F \leq 1 \\ 0, & \text{otherwise}\,. \end{cases} \tag{1.38}$$

The other prior that we will test is more informative than the first and assumes that it is more probable that the coin is fair, or at least close to it. To this end, we will use a normal prior with a Gaussian probability density

$$p(F|\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(F-\mu)^2}{2\sigma^2}\right)\,, \tag{1.39}$$

centred around the mean value $\mu = 1/2$ with standard deviation $\sigma = 0.1$. This kind of prior assigns most prior probability to values of F close to that of 1/2 that correspond to a fair coin. Both priors can be seen in the top–left panel of Figure 1.6 where the uniform prior corresponds to the continuous line and the normal prior to the dashed line.

To get the *likelihood function*, we start by choosing the sampling distribution $p(H, N|F)$, that is, the probability distribution of the data H and N given the value of F. For this task, we choose the *binomial* probability distribution with probability density given by

$$p(H, N|F) = \binom{N}{H} F^{H}(1 - F)^{N-H} \,. \tag{1.40}$$

The above formula can be understood as follows: R heads occur with probability F^{H} and $N - H$ tails with probability $(1 - F)^{N-H}$. The combinatorial factor that H heads can occur anywhere among the N trials, and there are $\binom{F}{H}$ of distributing H heads between N trials. If we fix H and N to their observed values then $\mathcal{L}(F) = p(H, N|F)$ is simply the likelihood function F.

According to Bayes' theorem then, the posterior distribution can be written as

$$p(F|H, N) = \frac{p(H, N|F)p(F)}{p(H, N)} \,, \tag{1.41}$$

where the model evidence is simply the normalisation factor

$$p(H, N) = \int_{0}^{1} p(H, N|F)p(F)dF \,. \tag{1.42}$$

In the case of the uniform prior of equation 1.38 the above integral can be computed analytically. This is not however true for the case of the normal prior of equation 1.39, for which numerical integration is necessary.

Figure 1.6 shows the evolution of the posterior distribution of equation 1.41, starting from the prior distribution in the top–left panel, as we gradually increase the number of data points that are included in the analysis. The posteriors with both prior choices are illustrated, also highlighting the effect of the prior choice on the posterior. As we can see from the same figure, while the number of trials N remains small (e.g. $N \leq 5$), the posterior corresponding to the informative normal prior remains unaffected. On the other hand, the posterior corresponding to the more agnostic uniform prior responds rapidly to the new data and concentrates close to the lower half of the F range. The reason for this difference is the fact that the few initial data points do not carry sufficient information compared to the normal prior, but they do so compared to the less informative uniform prior. For a higher number of trials N the behaviour is changing though. Both posteriors rapidly concentrate around the same value of F. This indicates that the prior, while important in the low–data regime, does not affect the posterior when the amount of data is substantial. This behaviour is a direct consequence of the *Bernstein–von Mises theorem* (Van der Vaart, 2000) which, under quite general conditions, states that "for sufficiently nice prior probabilities, in the

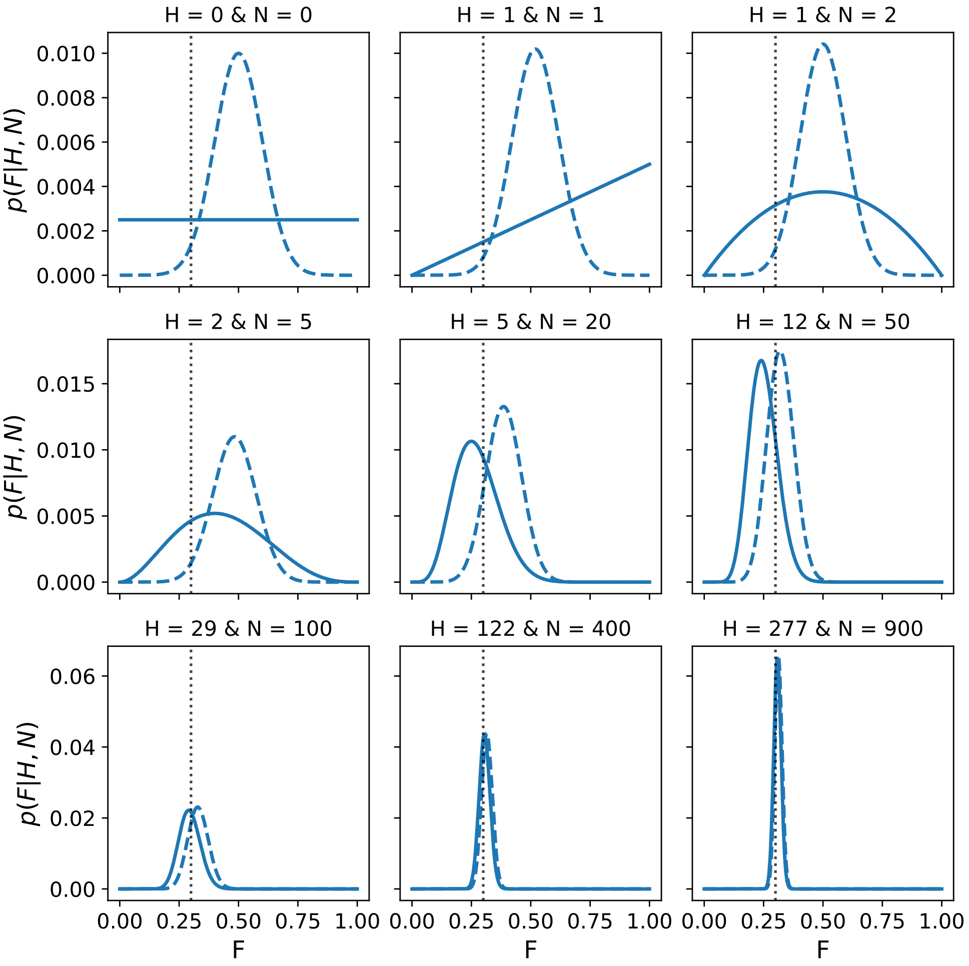

Figure 1.6: The evolution of the posterior probability distribution of a coin–tossing experiment for increasing number of trials. F is the bias parameter that we want to estimate. H is the number of times the coin landed on *heads* and N is the total number of trials. The continuous line corresponds to the case of using a uniform prior whereas the dashed line to a normal prior. The dotted line shows the true (unknown) value of F.

limit of infinite data the posterior converges to a Gaussian distribution independently of the initial prior". This also explains the symmetric form of the posterior in Figure 1.6 when the number of trials is large, as well as its reduced width.

1.6.2 Fitting a model to data

A general problem that scientists are often called to solve is that of fitting a mathematical model $m(t|\theta)$ to the data d, where the pairs (t, d) constitute the measured data points. A simple example of a model is the straight line $m(t|\alpha, \beta) = \alpha + \beta t$. $t = \{t_i\}$ could be a sequence of time instances, positions or any other physical quantity in which the measurements $d = \{d_i\}$

are collected. The task of *model fitting* lies within the context of Bayesian parameter estimation as the main goal is to approximate the posterior probability distribution $p(\theta|d, \mathcal{M})$, that is, the probability distribution of the parameters θ, given the data d and the model $\mathcal{M}$. The latter consists of the actual mathematical model $m(\theta)$ plus all the assumptions made during the analysis. Usually, the data d are assumed to be a noise–corrupted realisation of the model, meaning

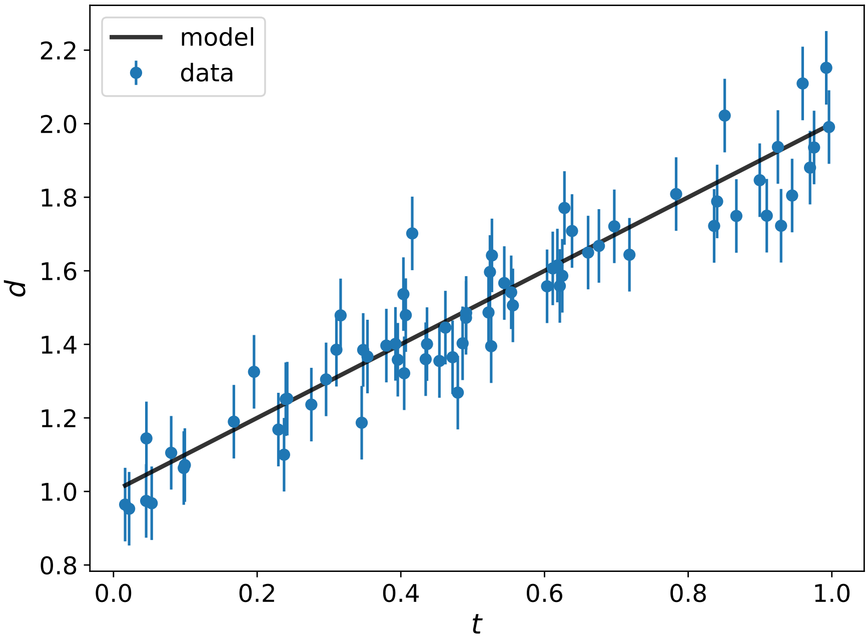

Figure 1.7: Example of data d and the straight line model $m(t|, \alpha, \beta) = \alpha + \beta t$ that was used to generate them assuming true values $\theta^* = (\alpha^*, \beta^*) = (1, 1)$ and $\epsilon = 0.1$.

$$d = m(t|\theta^*) + \epsilon\,, \tag{1.43}$$

where θ^* are the true values of the parameters that we, as scientists, are aspiring to approximate, and ϵ is the noise or uncertainty added to the model realisation $m(t|\theta^*)$ in order to generate the data d. In the absence of any noise (i.e. $\epsilon = 0$), the data are no longer corrupted and the value of θ^* can be estimated with certainty. As we have discussed already, this is an idealised scenario and in real life, our incomplete knowledge about the physical mechanism which produced the data introduces a non–zero noise contribution ϵ.

As the assumed physical model $m(\theta)$ is often *deterministic*, it follows that the sampling probability of the noise ϵ is identical to that of the data d, or in other words that

$$p(d|\theta, \mathcal{M}) = p(\epsilon|\theta, \mathcal{M})\,. \tag{1.44}$$

Furthermore, as the underlying physical mechanisms that give rise to the noise, often consist of a plethora of contributing factors one usually employs the *central limit theorem (CLT)* in order to justify the use of a zero–mean normal sampling distribution

$$\epsilon \sim \mathcal{N}(0,\Sigma)\,, \tag{1.45}$$

where Σ is the $D \times D$ positive–definite symmetric covariance matrix of the noise. The likelihood function is thus assumed to be Gaussian

$$p(d|\theta,\mathcal{M}) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left[d - m(t|\theta)\right]^T \Sigma^{-1}\left[d - m(t|\theta)\right]\right\}\,. \tag{1.46}$$

Contrary to popular opinion, and as we will discover in the next chapter where the principle of maximum entropy is discussed, a Gaussian function, or equivalently a normal sampling distribution, is quite often a very good choice. There are of course applications in which other sampling distributions will be more appropriate (e.g. *Poisson* for number counts). However, when only the (co–)variance of the noise is known, the normal distribution is the most conservative choice one can make (Gregory, 2005; Hogg et al., 2010; Sivia & Skilling, 2006). Of course, the accurate estimation of the covariance is on its own a difficult problem. Furthermore, if the covariance matrix Σ is estimated using simulated data $d_i \sim p(d)$, for instance

$$\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n}(d_i - \bar{d})(d_i - \bar{d})^T\,, \tag{1.47}$$

where $\bar{d} = n^{-1}\sum_{i=1}^{n} d_i$, the Gaussian likelihood function must to be modified to account for the uncertainty of the covariance estimate (Sellentin & Heavens, 2015).

Given the model, the data, and the likelihood, the final requirement in order to conduct Bayesian inference is the prior distribution $p(\theta|\mathcal{M})$. This will of course depend on the specific application and we will discuss the choice of prior in more detail in the next chapter. The task of approximating the posterior $p(\theta|d,\mathcal{M})$, that we have discussed so far, is in general analytically intractable for all but the simplest models and prior choices. In the rest of this thesis, we will present various methods and computational tools that will allow us to tackle problems such as this one. As an illustration, we offer Figure 1.8 which shows the 1–D and 2–D marginal posteriors of fitting the straight line model $m(t|,\alpha,\beta) = \alpha + \beta t$ to the data of Figure 1.7 assuming flat/uniform priors $\alpha,\beta \sim \mathcal{U}(-5,5)$. Although this is a relatively simple model, the same principles and techniques that were used to estimate its posterior also extend to more complicated applications.

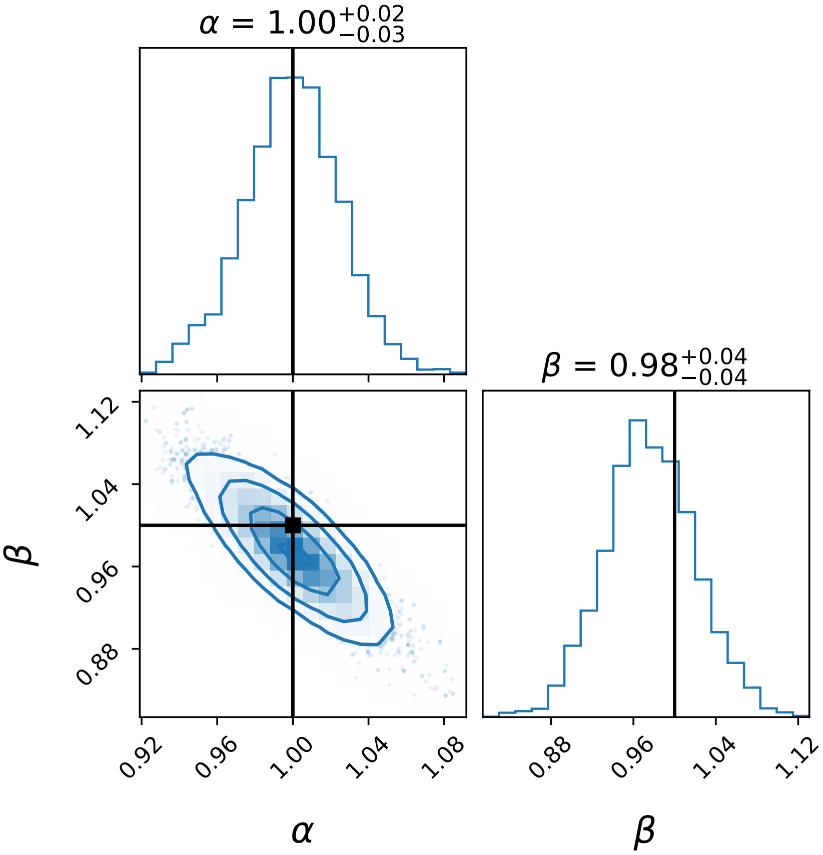

Figure 1.8: 1–D and 2–D marginal posterior contours of fitting the straight line model $m(t|, \alpha, \beta) = \alpha + \beta t$ to the data of Figure 1.7 assuming flat/uniform priors $\alpha, \beta \sim \mathcal{U}(-5, 5)$. The black lines show the true values of the parameters $\theta^* = (\alpha^*, \beta^*) = (1, 1)$ which were used to generate the data.

2

QUANTIFYING PRIOR KNOWLEDGE

Only entropy comes easy.

— Anton Chekhov

The discussion about Bayes' theorem so far explains how one can update one's prior knowledge in the light of new data. The question that naturally arises is how does one quantify their prior knowledge in the form of a probability distribution in the first place? In this chapter, we will attempt to provide a series of methods and practices that aim to do exactly that.

Ever since its initial development, many have criticised Bayesian inference for its dependence on prior knowledge (Efron, 1986; Gelman, 2008). Arguments against it mostly focus on the alleged subjectivity of its derived results. We maintain however that those claims are unfounded as all statistical analyses, Bayesian or not, employ prior information in some form or another. The difference with Bayesian inference is that this is explicitly done and taken into account. Indeed, anytime one has to perform a statistical analysis they have to assume a specific model (or a collection of them), often a specific set of parameters, a procedure of collecting data and a set of assumptions about the process that generated the data. In terms of the objectivity of its results, Bayesian inference is objective in the sense that any researcher possessing the same model assumptions, data, and prior knowledge will naturally reach exactly the same conclusions. Finally, the use of prior information can be understood as a great strength of Bayesian inference as it allows for the numerous scientific analyses which employ posterior distributions from old experiments as the priors for new ones, thus updating our knowledge of the world in a sequential and accumulative manner without discarding previous results. In this chapter, we will present both methods which employ this philosophy and those which attempt to provide a systematic procedure for generating prior distributions.

2.1 CONJUGATE PRIORS

A prior distribution is said to be conjugate to the likelihood function if it belongs to the same family of distributions as the posterior (Gelman, Carlin, et al., 2013). For instance, if the prior is a Gamma distribution and the likelihood is described by a Poisson probability mass function then the posterior is also Gamma.

From a mathematical point of view, conjugate priors are the most convenient choice as they allow us to compute the posterior analytically without the requirement of any computational method. From a scientific point of view however, conjugate priors are not well justified as they exist solely for the merit of algebraic convenience and they are not designed in order to encode the actual prior information. They are however a useful pedagogical and illustrative example of a method for choosing prior distributions.

2.1.1 Binomial likelihood function with Beta prior

Let us now consider the case of a binomial distribution

$$p(s,n|\theta) = \binom{n}{s}\theta^{s}(1-\theta)^{n-s}\,, \tag{2.1}$$

which is the sampling distribution for the number of successes s in n *Bernoulli trials* with probability of success equal to θ. Fixing the number of successes s and trials n and letting θ vary as a free parameter, the above probability mass function will be the likelihood function for this example. It is also more convenient to express it in terms of the number of failures $f = n - s$ instead of the number of trials n as

$$p(s,f|\theta) = \binom{s+f}{s}\theta^{s}(1-\theta)^{f}\,. \tag{2.2}$$

The prior distribution that is conjugate to this likelihood function turns out to be the *Beta* distribution

$$p(\theta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha,\beta)} \tag{2.3}$$

where $B(\alpha,\beta)$ is the *Beta* function

$$B(\alpha,\beta) = \int_{0}^{1}\theta^{\alpha-1}(1-\theta)^{\beta-1}d\theta\,, \tag{2.4}$$

that acts as a normalisation factor for the distribution and α and β are hyperparameters of the distribution. In the Bayesian context, a hyperparameter is a parameter of a prior distribution; the term is used to distinguish them from parameters of the model. For $\alpha = 1$ and $\beta = 1$ the *Beta* distribution reduces to the uniform distribution. We can now apply *Bayes' theorem* to produce the posterior distribution

$$p(\theta|s,f) = \frac{p(s,f|\theta)p(\theta)}{p(s,f)}\,, \tag{2.5}$$

where

$$p(s, f) = \int_0^1 p(s, f|\theta)p(\theta)d\theta\,, \tag{2.6}$$

is the *evidence*. Substituting equations 2.2 and 2.3 into 2.5 and 2.6 we have

$$\begin{aligned} p(\theta|s, f) &= \frac{\binom{s+f}{s}\theta^s(1-\theta)^f\theta^{\alpha-1}(1-\theta)^{\beta-1}/B(\alpha,\beta)}{\int_0^1 \binom{s+f}{s}\theta^s(1-\theta)^f\theta^{\alpha-1}(1-\theta)^{\beta-1}/B(\alpha,\beta)d\theta} \\ &= \frac{\theta^{s+\alpha-1}(1-\theta)^{f+\beta-1}}{B(s+\alpha, f+\beta)}\,, \end{aligned} \tag{2.7}$$

which is another *Beta* distribution with hyperparameters $\alpha' = s + \alpha$ and $\beta' = f + \beta$.

2.2 JEFFREYS PRIORS

There is often the need for priors that are invariant under reparameterisation, meaning that two different parameterisations θ and ϕ of the same model $\mathcal{M}$ yield consistent results. This type of prior was named after Sir Harold Jeffreys and it has the key feature that it is invariant under reparameterisations (C. Robert et al., 2007). One natural consequence of this approach is that a Jeffreys prior is fully determined by the choice of parameters, model and likelihood function. In that sense, it is often categorised as an objective prior as the preferences of the researcher affect it only indirectly through the choice of model and likelihood function. Although it is often characterised as an uninformative prior, this is actually far from true as all priors encode prior information. Perhaps a more appropriate name would be the *reparametersation invariant prior.*

2.2.1 One–dimensional case

Let us assume that θ and ϕ are two possible parameterisations of the same model $\mathcal{M}$, and θ is a continuously differentiable function of ϕ, then we say that the prior density $p_\theta(\theta)$ is invariant under the reparameterisation $\theta = \theta(\phi)$ if it is related to the prior density $p_\phi(\phi)$ by the usual change–of–variables theorem

$$p_\phi(\phi) = p_\theta(\theta)\left|\frac{d\theta}{d\phi}\right|\,. \tag{2.8}$$

Furthermore, the expected Fisher information is defined as

$$\mathcal{I}_\theta(\theta) = -\mathbb{E}_d\left[\frac{d^2}{d\theta^2}\log p(d|\theta)\right]\,, \tag{2.9}$$

and similarly for the ϕ parameterisation, where $\log p(d|\theta)$ is the logarithm of likelihood function, is transformed as

$$\mathcal{I}_\phi(\phi) = \mathcal{I}_\theta(\theta) \left(\frac{d\theta}{d\phi} \right)^2 , \tag{2.10}$$

under the reparametrisation $\theta = \theta(\phi)$.

Comparing the equations 2.8 and 2.10 one can see that defining the priors as

$$p_\theta(\theta) \propto \sqrt{\mathcal{I}_\theta(\theta)} , \tag{2.11}$$

and

$$p_\phi(\phi) \propto \sqrt{\mathcal{I}_\phi(\phi)} , \tag{2.12}$$

yields the desired invariance under reparameterisation.

2.2.2 Multi–dimensional case

The generalisation to multiple dimensions is straightforward. The change–of–variables formula has the general form

$$p_\phi(\phi) = p_\theta(\theta) |\det J| , \tag{2.13}$$

where θ and ϕ are now sets of parameters (i.e. vectors), and J is the Jacobian matrix of the transformation with components given by

$$J_{ij} = \frac{\partial \theta_i}{\partial \phi_i} , \tag{2.14}$$

where the indices i and j point to the i–th and j–th component of the parameter vectors θ and ϕ respectively. Similarly, the expected Fisher information matrix, defined as

$$\mathcal{I}_\theta(\theta)_{ij} = -\mathbb{E}_d \left[\frac{\partial^2}{\partial \theta_i \partial \theta_j} \log p(d|\theta) \right] , \tag{2.15}$$

is transformed as

$$\mathcal{I}_\phi(\phi) = J^T \mathcal{I}_\theta(\theta) J . \tag{2.16}$$

Computing the determinant of both parts of equation 2.16 leads to

$$\det \mathcal{I}_\phi(\phi) = \det \mathcal{I}_\theta(\theta) (\det J)^2 . \tag{2.17}$$

Comparing equations 2.13 and 2.17 one can see that defining the priors as

$$p_\theta(\theta) \propto \sqrt{\det \mathcal{I}_\theta(\theta)} , \tag{2.18}$$

and

$$p_\phi(\phi) \propto \sqrt{\det \mathcal{I}_\phi(\phi)} , \tag{2.19}$$

once again yields the desired invariance under reparameterisation.

2.2.3 Gaussian distribution with mean parameter

Assuming that the data d are Gaussian–distributed with unknown mean μ and known standard deviation σ, the probability density function of d given μ can be written as

$$p(d|\mu) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(d-\mu)^2}{2\sigma^2}} , \tag{2.20}$$

where σ is fixed. Applying equation 2.11 using equation 2.9 in this case, the Jeffreys prior for parameter $\theta \equiv \mu$ is simply

$$\begin{aligned} p(\mu) \propto \sqrt{I(\mu)} &= \sqrt{-\mathbb{E}_d \left[\frac{d^2}{d\mu^2} \log p(d|\mu) \right]} = \sqrt{\mathbb{E}_d \left[\left(\frac{d-\mu}{\sigma} \right)^2 \right]} \\ &= \sqrt{\int_{-\infty}^{\infty} p(d|\mu) \left(\frac{d-\mu}{\sigma} \right)^2 dd} = \frac{1}{\sigma} \propto 1 . \end{aligned} \tag{2.21}$$

The prior of μ in this case is independent of μ which means that is an improper (i.e. unnormalised) uniform prior.

2.2.4 Gaussian distribution with scale parameter

Assuming now that we know the mean parameter μ (i.e. μ is fixed) and the standard deviation σ is unknown, the Gaussian probability density function of d given σ is simply

$$p(d|\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(d-\mu)^2}{2\sigma^2}} . \tag{2.22}$$

Applying equation 2.11 using equation 2.9 in this case, the Jeffreys prior for parameter $\theta \equiv \sigma$ is

$$\begin{aligned} p(\sigma) \propto \sqrt{I(\sigma)} &= \sqrt{-\mathbb{E}_d \left[\frac{d^2}{d\sigma^2} \log p(d|\sigma) \right]} = \sqrt{\mathbb{E}_d \left[\left(\frac{(d-\mu)^2 - \sigma^2}{\sigma^3} \right)^2 \right]} \\ &= \sqrt{\int_{-\infty}^{\infty} p(d|\sigma) \left(\frac{(d-\mu)^2 - \sigma^2}{\sigma^3} \right)^2 dd} = \frac{\sqrt{2}}{\sigma} \propto \frac{1}{\sigma} . \end{aligned} \tag{2.23}$$

2.3 MAXIMUM ENTROPY PRIORS

In the absence of any information, one should distribute their degree of belief equally between all possible outcomes. This simple rule for assigning probabilities to discrete outcomes was considered so apparent to the fathers of probability theory, *Jacob Bernoulli* and *Pierre Simon Laplace*, that they did not even bother to give it a name. However, its importance in the context of probability theory was clear to both of them. In particular, Laplace wrote:

> *The theory of chance consists in reducing all the events of the same kind to a certain number of cases equally possible, that is to say, to such as we may be equally undecided about in regard to their existence, and in determining the number of cases favourable to the event whose probability is sought. The ratio of this number to that of all the cases possible is the measure of this probability, which is thus simply a fraction whose numerator is the number of favourable cases and whose denominator is the number of all the cases possible.*

This rule was later named the *principle of insufficient reason*, possibly as a play on Leibniz's *principle of sufficient reason* (Brading & Castellani, 2003). Finally, it was renamed to the *principle of indifference* by economist *John Maynard Keynes* that noted that it can only be applied when one has no additional information (Keynes, 1921).

But what if we have some additional information, perhaps in the form of expectation values? Can we somehow incorporate that information and minimally extend the *principle of indifference*? The answer to this question was provided by Jaynes in the form of the *principle of maximum entropy (MaxEnt)* (E. Jaynes, 1982).

Using the notion of Shannon's "entropy" that quantifies the uncertainty of a probability distribution, MaxEnt is a mathematical procedure for the derivation of the maximally agnostic (i.e. least informative) probability distribution subject to a collection of known constraints. The MaxEnt principle can be applied in the assignment of prior probabilities in cases where we know some constraints about the parameters *a priori* in the form of expectation values (e.g. mean, variance, lower or upper bounds, etc.) and we seek to find the least informative distribution that respects those constraints and still complies as much as possible to the *principle of indifference*.

The MaxEnt principle turns the problem of defining a prior distribution into a task of optimisation. In particular, one seeks the probability distribution with the maximum entropy, that is, the least informative, subject to a collection of algebraic constraints in the form of expectation values. Before we move on to discuss some explicit examples that demonstrate the application of the MaxEnt principle, let us first present a couple of definitions for the "entropy".

2.3.1 Shannon's entropy

In 1948, *Claude Shannon*'s pioneering work on *information theory* (Shannon, 1948) introduced a measure of the uncertainty of a discrete probability distribution which he termed "entropy" and defined as

$$S(p) = -\sum_{i=1}^{n} p_i \log p_i \,. \tag{2.24}$$

The entropy of a probability distribution quantifies the amount of missing information, uncertainty or "surprisal" inherent in the distribution (MacKay, 2003). In other words, entropy is the expectation value $\mathbb{E}_p[I]$ of the information

$$I_i = -\log p_i\,, \tag{2.25}$$

which quantifies the information content of an event i with probability p_i. For example, a fair coin landing "heads–tails–heads" with probability $1/2^3$ provides information of $-\log(1/2^3) = 3\log 2$ or 3 bits. Information is measured in "bits" if the logarithm has base 2 or in "nats" if it has base e. Information has a series of desired properties, namely

1. An event i with probability $p_i = 1$ is certain and offers no information (i.e. $I_i = 0$),
2. The lower the probability of an event, the more surprising it is and thus the higher its information contribution,
3. Information is additive, meaning that the total amount of information is the sum of the information of the individual events.

It turns out the form of equation 2.25 for information is the only option if we want less probable events to have more information, and information to add for independent events.

Let us consider a simple example in order to make the notions of information and entropy better understood. Suppose that according to the weather forecast there is a $p = 1/2$ chance that it rains $1\,cm$, $p = 1/4$ chance that it rains $2\,cm$ and $p = 1/4$ chance that it does not rain at all. The expected amount of rain is simply $1/2 \times 1\,cm + 1/4 \times 2\,cm + 1/4 \times 0 = 1\,cm$. The expected amount of information that you gain when you find out how much it rains is $-1/2\log(1/2) - 1/4\log(1/4) - 1/4\log(1/4) = 3/2\log 2$ or 3/2 bits, this is the Shannon entropy of the weather report.

Shannon's entropy naturally assumes that the uniform distribution $p_i = 1/n$ where n is the number of discrete events holds a very special role. In the absence of any other constraints, assigning equal probability to all outcomes (i.e. p_i is constant) corresponds to the state of complete ignorance. In other words, the distribution that maximises Shannon's entropy is the uniform distribution in accordance with the *principle of indifference*.

2.3.2 Relative entropy

Another useful, entropy–like quantity is the following

$$D_{KL}(p|q) = \sum_{i=1}^{n} p_i \log\left(\frac{p_i}{q_i}\right)\,, \tag{2.26}$$

that has been given many names, including *relative entropy*, *cross–entropy*, and *Kullback–Leibler (KL) divergence* as (Kullback & Leibler, 1951) were the first ones to demonstrate its potential for statistical applications.

The latter is a measure of information gained when one updates their beliefs, initially quantified by a distribution q to an updated distribution p. In that sense, relative entropy is a measure of statistical distance between the two distributions and it is defined as

$$D_{KL}(p|q) = \sum_{i=1}^{n} p_i \log\left(\frac{p_i}{q_i}\right) , \tag{2.27}$$

for the discrete case, and

$$D_{KL}(p|q) = \int p(\theta) \log\left(\frac{p(\theta)}{q(\theta)}\right) d\theta , \tag{2.28}$$

for the continuous case.

Relative entropy has a collection of desired properties too, including

1. It is always non–negative,

   $$D_{KL}(p|q) \geq 0 , \tag{2.29}$$

   a result commonly known as *Gibbs' inequality*. Relative entropy is zero if and only if $p = q$ almost everywhere.

2. Relative entropy, unlike Shannon's entropy, is well defined for continuous distributions.

3. Given a transformation $\theta = \theta(\phi)$ such that $p(\theta)d\theta = p(\phi)d\phi$ and $q(\theta)d\theta = q(\phi)d\phi$ the relative entropy is parameterisation invariant, meaning

   $$\begin{aligned} D_{KL}(p|q) &= \int p(\theta) \log\left(\frac{p(\theta)}{q(\theta)}\right) d\theta \\ &= \int p(\phi) \log\left(\frac{p(\phi)\frac{d\phi}{d\theta}}{q(\phi)\frac{d\phi}{d\theta}}\right) d\phi \\ &= \int p(\phi) \log\left(\frac{p(\phi)}{q(\phi)}\right) d\phi . \end{aligned} \tag{2.30}$$

4. Relative entropy reduces to the well known Shannon's entropy, up to a sign, for the case of a uniform distribution q in the discrete case.

As we mentioned in the previous sub–section, using *Shannon*'s definition of entropy places the uniform distribution into a very special place, that of the maximum entropy distribution in the absence of any other constraints that provide additional information. Although this is in accordance with the *principle of indifference*, there are cases in practice in which one requires a different prior distribution q, before taking into account any constraints. For instance, one may seek to find a distribution that minimally deviates from a Jeffreys prior subject to some constraints. In those cases, instead of maximising *Shannon*'s entropy, one can minimise the relative entropy. For simplicity, we will refer to the principle of minimum relative entropy as MaxEnt too Shore & Johnson (1980) proved that minimising the relative entropy is the uniquely correct way of updating probability distributions in the face of new information in the form of expectation values for both discrete and continuous cases. Furthermore, the *relative entropy*, unlike Shannon's entropy, is easily generalisable to the continuous case too.

2.3.3 Lagrange multipliers

The method of Lagrange multipliers (Riley et al., 1999) offers a powerful way of finding the local extrema (i.e. maxima and minima) of a function $f(p)$ subject to a constraint $g(p) = 0$. If no constraint is available then the extrema of f can be found by solving

$$df = \frac{\partial f}{\partial p_1} dp_1 + \cdots + \frac{\partial f}{\partial p_n} dp_n = 0\,. \tag{2.31}$$

Here the dp_i are independent so one concludes that the extrema are simply given by $\partial f / \partial p_i = 0$. However, the existence of the constraint means that dp_i are not actually independent since

$$dg = \frac{\partial g}{\partial p_1} dp_1 + \cdots + \frac{\partial g}{\partial p_n} dp_n = 0\,, \tag{2.32}$$

because $g(p)$ is constant. We can combine equations 2.31 and 2.32 by first multiplying the second by an unknown factor λ called *Lagrange multiplier*, thus yielding

$$d(f - \lambda g) = \left(\frac{\partial f}{\partial p_1} - \lambda \frac{\partial g}{\partial p_1} \right) dp_1 + \cdots + \left(\frac{\partial f}{\partial p_n} - \lambda \frac{\partial g}{\partial p_n} \right) dp_n = 0\,. \tag{2.33}$$

We can now choose λ such that

$$\frac{\partial f}{\partial p_i} - \lambda \frac{\partial g}{\partial p_i} = 0\,, \tag{2.34}$$

for all $i \in \{1, \ldots, n\}$. Equations 2.34 along with the constraint equation $g(p)$ are sufficient to determine the value of λ and coordinates p_i of the stationary point.

2.3.4 Uniform distribution

Assuming that the only constraint or form of information is that the sum of all probabilities is equal to one, meaning

$$\sum_{i=1}^{n} p_i = 1\,, \tag{2.35}$$

then in order to find the maximum entropy distribution, we have to solve

$$d\left[-\sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} - \lambda\left(\sum_{i=1}^{n} p_i - 1\right)\right] = 0\,, \tag{2.36}$$

where the first term is the relative entropy and the second term is the constraint multiplied with the unknown Lagrange multiplier λ. Doing some simple calculus on the expression of 2.36 we get

$$\begin{aligned}
d\left[-\sum_{i=1}^{n} p_i \log p_i + \sum_{i=1}^{n} p_i \log q_i - \lambda\left(\sum_{i=1}^{n} p_i - 1\right)\right] = 0 \Longrightarrow \\
-\sum_{i=1}^{n} dp_i\, \log p_i - \sum_{i=1}^{n} p_i\, d(\log p_i) \\
+\sum_{i=1}^{n} dp_i\, \log q_i - \lambda \sum_{i=1}^{n} dp_i = 0 \Longrightarrow \\
-\sum_{i=1}^{n} dp_i\, \log p_i - \sum_{i=1}^{n} p_i \left(\sum_{j=1}^{n} \frac{\partial \log p_i}{\partial p_j}\, dp_j\right) \\
+\sum_{i=1}^{n} dp_i\, \log q_i - \lambda \sum_{i=1}^{n} dp_i = 0 \Longrightarrow \\
-\sum_{i=1}^{n} dp_i\, \log p_i - \sum_{i=1}^{n} p_i \left(\sum_{j=1}^{n} \delta_{ij} \frac{1}{p_j}\, dp_j\right) \\
+\sum_{i=1}^{n} dp_i\, \log q_i - \lambda \sum_{i=1}^{n} dp_i = 0 \Longrightarrow \\
-\sum_{i=1}^{n} dp_i\, \log p_i - \sum_{i=1}^{n} p_i \frac{1}{p_i}\, dp_i + \sum_{i=1}^{n} dp_i\, \log q_i - \lambda \sum_{i=1}^{n} dp_i = 0 \Longrightarrow \\
\sum_{i=1}^{n} \left(-\log \frac{p_i}{q_i} - 1 - \lambda\right) dp_i = 0
\end{aligned} \tag{2.37}$$

According to our previous discussion on Lagrange multipliers, for equation 2.37 to hold, the terms in the parentheses need to vanish for every $i = 1, \dots, n$, therefore we get

$$p_i = q_i e^{-(1+\lambda)}\,. \tag{2.38}$$

We can determine the value of λ using the constraint equation 2.35

$$\sum_{i=1}^{n} q_i e^{-(1+\lambda)} = 1\,, \tag{2.39}$$

$\sum_{i=1}^{n} q_i = 1$ so $\lambda = -1$. Therefore, the distribution in the discrete case is

$$p_i = q_i\,, \tag{2.40}$$

and in the continuous case is

$$p(\theta) = q(\theta)\,, \tag{2.41}$$

Furthermore, assuming that q is a uniform (i.e. $q_i = 1/n$) in accordance with the *principle of indifference*, then p is also uniform. This means that the distribution of maximum entropy under the minimal constraint that the total probability needs to sum up to one is the uniform distribution. Let us now move on to a few more intriguing examples.

2.3.5 Exponential distribution

Suppose now that we have an additional constraint apart from equation 2.35 for the sum of probabilities,

$$\sum_{i=1}^{n} p_i \theta_i = \mu \,, \tag{2.42}$$

indicating that the mean value of θ is known and equal to μ. Having two constraints requires us to introduce two Lagrange multipliers, λ and $\tilde{\lambda}$, and solve

$$d\left[-\sum_{i=1}^{n} p_i \log\frac{p_i}{q_i} - \lambda\left(\sum_{i=1}^{n} p_i - 1\right) - \tilde{\lambda}\left(\sum_{i=1}^{n} p_i\theta_i - \mu\right)\right] = 0 \,, \tag{2.43}$$

in order to find the appropriate maximum entropy distribution. Similarly to before, after some calculus, we get

$$\sum_{i=1}^{n}\left(-\log\frac{p_i}{q_i} - 1 - \lambda - \theta_i\tilde{\lambda}\right) dp_i = 0 \,. \tag{2.44}$$

Again, the term in the parentheses needs to vanish for any value of i, thus

$$p_i = q_i e^{-(1+\lambda)} e^{-\tilde{\lambda}\theta_i} \,. \tag{2.45}$$

We can now apply the two constraints to determine the values of the Lagrange multipliers. From equation 2.35 for the first constraint, we have

$$e^{-(1+\lambda)} = \frac{1}{\sum_{i=1}^{n} q_i e^{-\tilde{\lambda}\theta_i}} \,. \tag{2.46}$$

Similarly, from equation 2.42 for the second constraint, we have

$$\sum_{i=1}^{n} q_i\theta_i e^{-\tilde{\lambda}\theta_i} - \mu \sum_{i=1}^{n} q_i e^{-\tilde{\lambda}\theta_i} = 0 \,, \tag{2.47}$$

which can only be solved numerically. Equation 2.45 can also be written for the continuous case as

$$p(\theta) = q(\theta) e^{-(1+\lambda)} e^{-\tilde{\lambda}\theta} \,. \tag{2.48}$$

Equation 2.48 is the general MaxEnt prior for an arbitrary pseudo–prior $q(\theta)$. However, the expression can be simplified more if we assume that our state of knowledge about θ prior to the information provided by the constraints 2.35 and 2.42, is that it is positive (i.e. $\theta > 0$). This means, that $q(\theta)$ is a uniform distribution, in agreement with the principle of indifference. Furthermore, to avoid issues with "infinities" and render q a proper pseudo–prior, we can set an upper limit L on the possible values of θ, therefore

$$q(\theta) = \mathcal{U}(\theta|0, L) = \begin{cases} 1/L & \text{if } 0 < \theta \leq L \\ 0 & \text{otherwise} \end{cases} . \tag{2.49}$$

Once we have derived the form of the MaxEnt prior $p(\theta)$ we can then take the limit of $L \longrightarrow \infty$ to allow θ to be any positive real number. Including this particular choice of $q(\theta)$, equation 2.48 takes the form

$$p(\theta) = \frac{1}{L} e^{-(1+\lambda)} e^{-\tilde{\lambda}\theta}, \quad \theta > 0 . \tag{2.50}$$

Using the pseudo-prior of equation 2.49, the first constraint, for the total probability, given by equation 2.35, can be expressed as

$$\begin{aligned} \int_{-\infty}^{\infty} p(\theta) d\theta &= 1 \Longrightarrow \\ \frac{1}{L} e^{-(1+\lambda)} \int_{0}^{L} e^{-\tilde{\lambda}\theta} d\theta &= 1 \Longrightarrow \\ \frac{1}{L} e^{-(1+\lambda)} \left[-\frac{e^{-\tilde{\lambda}\theta}}{\tilde{\lambda}} \right]_{\theta=0}^{\theta=L} &= 1 \Longrightarrow \\ \frac{1}{L} e^{-(1+\lambda)} &= \frac{\tilde{\lambda}}{1 - e^{-\tilde{\lambda}L}} . \end{aligned} \tag{2.51}$$

Similarly, the second constraint, for the expected or mean value of θ, given by equation 2.42, can be expressed as

$$\begin{aligned} \int_{-\infty}^{\infty} \theta p(\theta) d\theta &= \mu \Longrightarrow \\ \frac{1}{L} e^{-(1+\lambda)} \int_{0}^{L} \theta e^{-\tilde{\lambda}\theta} d\theta &= \mu \Longrightarrow \\ \frac{1}{L} e^{-(1+\lambda)} \left[-\frac{e^{-\tilde{\lambda}\theta}(\tilde{\lambda}\theta + 1)}{\tilde{\lambda}^2} \right]_{\theta=0}^{\theta=L} &= \mu \Longrightarrow \\ \frac{1}{L} e^{-(1+\lambda)} \frac{1 - e^{-\tilde{\lambda}L}(\tilde{\lambda}L + 1)}{\tilde{\lambda}^2} &= \mu \Longrightarrow \\ \frac{\tilde{\lambda}}{1 - e^{-\tilde{\lambda}L}} \times \frac{1 - e^{-\tilde{\lambda}L}(\tilde{\lambda}L + 1)}{\tilde{\lambda}^2} &= \mu \Longrightarrow \\ \mu = \frac{1}{\tilde{\lambda}} - \frac{Le^{-\tilde{\lambda}L}}{1 - e^{-\tilde{\lambda}L}} &, \end{aligned} \tag{2.52}$$

where we used equation 2.51 to simplify the result.

Taking the limit $L \longrightarrow +\infty$ and using equation 2.52, we find that

$$\mu \longrightarrow \frac{1}{\tilde{\lambda}}\,, \tag{2.53}$$

as the second term vanishes. Similarly, using equation 2.51, we find that

$$\frac{1}{L} e^{-(1+\lambda)} \longrightarrow \tilde{\lambda}\,. \tag{2.54}$$

Substituting these results into equation 2.50, we are lead to

$$p(\theta|\mu) = \frac{1}{\mu} e^{-\frac{\theta}{\mu}}\,, \tag{2.55}$$

the well known exponential distribution. What this paragraph taught us is crucial, if we only know the mean of a non−negative parameter and nothing else, then the exponential distribution is the one that best represents the current state of knowledge, by making the fewest assumptions.

2.3.6 Normal distribution

Suppose that we also know the the standard deviation σ given by

$$\sum_{i=0}^{n} p_i(\theta_i - \mu)^2 = \sigma^2\,, \tag{2.56}$$

as an additional constraint. We now have to solve

$$d\left[-\sum_{i=1}^{n} p_i \log\frac{p_i}{q_i} - \lambda\left(\sum_{i=1}^{n} p_i - 1\right) - \tilde{\lambda}\left(\sum_{i=1}^{n} p_i(\theta_i - \mu)^2 - \sigma^2\right)\right] = 0\,, \tag{2.57}$$

where the first term corresponds to the entropy, the second to the constraint that the sum of all probabilities needs to add up to one, and the last term to the standard deviation constraint that also includes that about the mean μ. Thus we have two *Lagrange* multipliers and we follow the same procedure as before, solving equation 2.57 we have

$$\sum_{i=1}^{n}\left(-\log\frac{p_i}{q_i} - 1 - \lambda - \tilde{\lambda}(\theta_i - \mu)^2\right) dp_i = 0\,. \tag{2.58}$$

The terms in the parentheses need to vanish for all values of i, thus

$$p_i = q_i e^{-(1+\lambda)} e^{-\tilde{\lambda}(\theta_i - \mu)^2}\,. \tag{2.59}$$

The corresponding continuous probability density function is simply

$$p(\theta) = q(\theta) e^{-(1+\lambda)} e^{-\tilde{\lambda}(\theta - \mu)^2}\,. \tag{2.60}$$

Furthermore, assuming a uniform prior $q(\theta) \propto 1$ in accordance with the *principle of indifference*, equation 2.48 reduces to

$$p(\theta) = e^{-(1+\lambda)} e^{-\tilde{\lambda}(\theta-\mu)^2} . \tag{2.61}$$

We can now apply the constraint equations 2.35 and 2.56 in order to uniquely determine the values of the two *Lagrange* multipliers. In the continuous limit, the first constraint given by equation 2.35, is written as

$$\int_{-\infty}^{\infty} p(\theta) d\theta = 1 . \tag{2.62}$$

Substituting equation 2.61 into equation 2.62 yields

$$e^{-(1+\lambda)} \int_{-\infty}^{\infty} e^{-\tilde{\lambda}(\theta-\mu)^2} d\theta = 1 . \tag{2.63}$$

Doing the change of variables $z = \sqrt{\tilde{\lambda}}(\theta - \mu)$ brings equation 2.63 into the simpler form

$$\frac{e^{-(1+\lambda)}}{\sqrt{\tilde{\lambda}}} \int_{-\infty}^{\infty} e^{-z^2} dz = 1 , \tag{2.64}$$

where the integral is the so called *Gaussian integral* with value equal to $\sqrt{\pi}$. Therefore,

$$e^{-(1+\lambda)} = \sqrt{\frac{\tilde{\lambda}}{\pi}} , \tag{2.65}$$

and equation 2.61 reduces to

$$p(\theta) = \sqrt{\frac{\tilde{\lambda}}{\pi}} e^{-\tilde{\lambda}(\theta-\mu)^2} . \tag{2.66}$$

We can now move on to determine the second *Lagrange* multiplier $\tilde{\lambda}$ by substituting equation 2.66 into 2.56, thus

$$\sqrt{\frac{\tilde{\lambda}}{\pi}} \int_{-\infty}^{\infty} e^{-\tilde{\lambda}(\theta-\mu)^2} (\theta - \mu)^2 d\theta = \sigma^2 . \tag{2.67}$$

Applying the same change of variables as before, $z = \sqrt{\tilde{\lambda}}(\theta - \mu)$, we have

$$\frac{1}{\tilde{\lambda}\sqrt{\pi}} \int_{-\infty}^{\infty} e^{-z^2} z^2 dz = \sigma^2 . \tag{2.68}$$

The integral can be computed using integration by parts

$$\frac{1}{\tilde{\lambda}\sqrt{\pi}} \left\{ \left[z \left(-\frac{1}{2} e^{-z^2} \right) \right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} -\frac{1}{2} e^{-z^2} dz \right\} = \sigma^2 , \tag{2.69}$$

in which the first term in the braces vanishes and the second is equal to $\sqrt{\pi}/2$, thus

$$\tilde{\lambda} = \frac{1}{2\sigma^2} . \tag{2.70}$$

Finally, substituting equation 2.70 into 2.66 leads to the usual Gaussian function

$$p(\theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\theta-\mu)^2}{2\sigma^2}} , \tag{2.71}$$

as the maximum entropy probability density function. In other words, the maximum entropy probability distribution subject to the constraints of known mean and standard deviation is the normal distribution.

2.4 REFERENCE PRIORS

The method of *reference priors*, originally proposed by Jose M Bernardo (1979) and later expanded by others (Berger et al., 2009; José M Bernardo, 2005; José M Bernardo & Smith, 2009; Kass & Wasserman, 1996), is another approach that utilises information–theoretic ideas. The main idea behind reference priors is to choose the prior $p(\theta)$ to maximise some notion of discrepancy between the prior $p(\theta)$ and the posterior $p(\theta|d)$. One reason to do this is that such a prior would allow the data d to be maximally informative and have the greatest effect on the posterior distribution. In one–dimensional cases it turns out that reference priors and Jeffreys priors are equivalent. In higher dimensional cases, however, they are generally different. The research field of reference priors has expanded substantially during the past decades. For this reason, we will cover the fundamentals in this section and direct the reader to the aforementioned references for more information.

As we discussed in the previous section regarding maximum entropy priors, a common measure of the discrepancy between two distributions is the relative entropy or KL divergence, given by equation 2.28. In the case of the prior and posterior distribution, this can be written as

$$D_{KL}\left[p(\theta|d)|p(\theta)\right] = \int p(\theta|d) \log \frac{p(\theta|d)}{p(\theta)} d\theta \, . \tag{2.72}$$

One might then wonder how can we maximise the above discrepancy measure, in order to find the prior $p(\theta)$, without knowing the posterior distribution $p(\theta|d)$. Reference priors address this point by maximising the *expectation value* of the relative entropy of equation 2.72 over the distribution of the data $p(d^{(n)})$, where $d^{(n)} = \{d_1, \dots, d_n\}$ are n conditionally independent instances of the data. At first, this appears to be a frequentist procedure as one will base the choice of the prior on unseen fictional data, such as infinite repetitions of the same experiment (e.g. in the limit that $n \longrightarrow \infty$). However, unlike frequentist approaches, once the prior is determined, the analysis proceeds in the usual Bayesian manner. Furthermore, *Bernardo* argued that taking the limit of n to infinity does not just lead to a convenient mathematical procedure but it is also, philosophically, the right thing to do. His argument is that when choosing a prior we should consider many future experiments than just a single one. In this sense, the reference prior procedure aims to maximise the *missing information* about the parameters θ which can be obtained by repeated experiments.

2.4.1 Mutual information

The expected relative entropy between the prior and posterior, also known as the *mutual information*, quantifies the missing information and can be derived as follows:

$$\begin{aligned} I(\theta, d^{(n)}) = \mathbb{E}_{d^{(n)}}\left[D_{KL}\left[p(\theta|d^{(n)})|p(\theta)\right]\right] &= \int p(d^{(n)}) D_{KL}\left[p(\theta|d^{(n)})|p(\theta)\right] dd^{(n)} \\ &= \int p(d^{(n)}) \int p(\theta|d^{(n)}) \log \frac{p(\theta|d^{(n)})}{p(\theta)} d\theta dd^{(n)} \\ &= \int\int p(\theta|d^{(n)}) p(d^{(n)}) \log \frac{p(\theta|d^{(n)})}{p(\theta)} d\theta dd^{(n)} \\ &= \int\int p(\theta, d^{(n)}) \log \frac{p(\theta, d^{(n)})}{p(\theta)p(d^{(n)})} d\theta dd^{(n)}\,, \end{aligned} \tag{2.73}$$

where we used Bayes' theorem to introduce the joint probability $p(\theta, d^{(n)})$. In order to understand the meaning and significance of the above expression, let us consider the simple case in which the parameters θ and the data $d^{(n)}$ are independent. In this case the joint probability of the two is separable, or $p(\theta, d^{(n)}) = p(\theta)p(d^{(n)})$, and the mutual information $I(\theta, d^{(n)})$ is zero. In other words, the data $d^{(n)}$ have no effect on the parameters θ. However, those two quantities are not generally independent, and the mutual information quantifies the influence or effect of the data $d^{(n)}$ on the parameters θ. Finally, defining the reference priors in terms of the mutual information, has the advantage of sharing its reparameterisation invariance.

2.4.2 Maximising the mutual information

The reference prior $p^*(\theta)$ is simply the prior which maximises the mutual information in the limit that $n \longrightarrow \infty$, or:

$$p^*(\theta) = \lim_{n\longrightarrow\infty} p_n^*(\theta)\,, \text{ where } \; p_n^*(\theta) = \underset{p(\theta)}{\arg\max}\, I(\theta, d^{(n)})\,. \tag{2.74}$$

The mutual information of equation 2.73 can be written as:

$$I(\theta, d^{(n)}) = \int p(\theta) \log \frac{f(\theta)}{p(\theta)} d\theta\,, \tag{2.75}$$

where we have introduced the function

$$f_n(\theta) = \exp\left\{\int p(d^{(n)}|\theta) \log p(d^{(n)}|\theta) dd^{(n)}\right\}\,, \tag{2.76}$$

where the product sampling distribution is defined as

$$p(d^{(n)}|\theta) = \prod_{i=1}^{n} p(d_i|\theta)\,. \tag{2.77}$$

Finding the prior distribution $p_n(\theta)$, which maximises the mutual information $I(\theta, d^{(n)})$ subject to the constraint $\int p(\theta)d\theta = 1$, is essentially a problem that can be solved via the methods of calculus of variations. It may be simpler to derive the result by working in the discrete case. This means that we have to solve

$$d\left[\sum_i p_i \log\left(\frac{f_i}{p_i}\right) + \lambda\left(\sum_i p_i - 1\right)\right] = 0 \tag{2.78}$$

where λ is a Lagrange multiplier, in order to find the prior p_i. The derivation is as follows:

$$\begin{aligned}
\sum_i dp_i \log\left(\frac{f_i}{p_i}\right) + \sum_i p_i\left[\sum_j \frac{\partial \log(f_i/p_i)}{\partial p_j} dp_j\right] + \lambda \sum_i dp_i = 0 \Longrightarrow \\
\sum_i dp_i \log\left(\frac{f_i}{p_i}\right) + \sum_i p_i\left[\sum_j \delta_{ij}\left(-\frac{1}{p_j}\right) dp_j\right] + \lambda \sum_i dp_i = 0 \Longrightarrow \\
\sum_i \left[\log\left(\frac{f_i}{p_i}\right) - 1 + \lambda\right] dp_i = 0
\end{aligned} \tag{2.79}$$

For the above equation to be true, all terms in the sum must be zero, in other words we get that $p_i = f_i \exp(\lambda - 1)$ or simply $p_i \propto f_i$. Rewriting this in the continuous case, in the limit that $n \longrightarrow \infty$ we have:

$$p(\theta) = \lim_{n\to\infty} \frac{f_n(\theta)}{f_n(\theta_0)}\,, \tag{2.80}$$

where θ_0 is an internal point in parameter space and $f_n(\theta)$ is given by equation 2.76. Alternatively, $f_n(\theta)$ can be defined as

$$f_n(\theta) = \exp\left\{\int p(d^{(n)}|\theta) \log\left[\frac{p(d^{(n)}|\theta)h(\theta)}{\int p(d^{(n)}|\theta)h(\theta)d\theta}\right] dd^{(n)}\right\}\,, \tag{2.81}$$

where we have included an arbitrary pseudo–prior $h(\theta)$. Carefully selecting the functional form of $h(\theta)$ (e.g. conjugate prior) can significantly simplify the calculations.

Intuitively, equations 2.80 and 2.81 state that the reference prior $p(\theta)$ depends only on the asymptotic behaviour of the posterior, and schematically can be written in the form

$$\begin{aligned}
p(\theta) &\propto \exp\left\{\int p(d^{(n)}|\theta) \log p^{*}(\theta|d^{(n)}) dd^{(n)}\right\} \\
&\propto \exp\left\{\mathbb{E}_{p(d^{(n)}|\theta)}\left[\log p^{*}(\theta|d^{(n)})\right]\right\}\,,
\end{aligned} \tag{2.82}$$

where $p^{*}(\theta|d^{(n)})$ is the asymptotic form of the posterior.

2.4.3 Asymptotic solution

Finding the reference prior is now reduced to computing $f_n(\theta)$ using equation 2.76 or 2.81. However, this can be quite challenging in practice. The problem can be simplified by using the *Bernstein–von Mises* theorem, which, as we discussed in Chapter 1, states that under certain conditions, as the sample size approaches infinity (i.e. $n \longrightarrow \infty$), the posterior distribution converges to a normal distribution centred on the *maximum likelihood estimate (MLE)* θ_n with variance equal to $n^{-1}I^{-1}(\theta_n)$, where $I(\theta)$ is the *Fisher information* given by

$$\mathcal{I}(\theta) = -\mathbb{E}_{p(d|\theta)}\left[\frac{\partial^2 \log p(d|\theta)}{\partial\theta^2}\right] . \tag{2.83}$$

We can use the fact that MLE is a *consistent* and *asymptotically* sufficient estimator, meaning that

$$\lim_{n\longrightarrow\infty} \hat{\theta}_n = \theta\,, \tag{2.84}$$

and

$$\lim_{n\longrightarrow\infty} \int p(d^{(n)}|\theta)\log\frac{p^*(\theta|d^{(n)})}{p^*(\theta|\hat{\theta}_n)}dd^{(n)} = 0\,, \tag{2.85}$$

respectively, in order to simplify the form of the reference prior. Starting with equation 2.81, we can write

$$\begin{aligned}
f_n^*(\theta) &= \exp\left\{\int p(d^{(n)}|\theta)\log p^*(\theta|d^{(n)})dd^{(n)}\right\} \\
&= \exp\left\{\int p(d^{(n)}|\theta)\log p^*(\theta|\hat{\theta}_n)dd^{(n)}\right\} \\
&= \exp\left\{\int p(\hat{\theta}_n|\theta)\log p^*(\theta|\hat{\theta}_n)d\hat{\theta}_n\right\} \\
&= \exp\left\{\log p^*(\theta|\hat{\theta}_n)\Big|_{\hat{\theta}_n=\theta}\right\} \\
&= p^*(\theta|\hat{\theta}_n)\Big|_{\hat{\theta}_n=\theta} .
\end{aligned} \tag{2.86}$$

Therefore, the asymptotically normal form of the posterior with mean $\hat{\theta}_n$ and variance $n^{-1}\mathcal{I}^{-1}(\theta_n)$ can be written as

$$p_n^*(\theta|\hat{\theta}_n) = (2\pi)^{-1/2}n^{1/2}I^{1/2}(\theta_n)\exp\left[-\frac{1}{2}n\mathcal{I}(\theta_n)(\theta-\theta_n)^2\right]\Bigg|_{\hat{\theta}_n=\theta} \tag{2.87}$$

Substituting this into equation 2.86 we find that

$$f_n^*(\theta) = (2\pi)^{-1/2}n^{1/2}\mathcal{I}^{1/2}(\theta)\,, \tag{2.88}$$

and using equation 2.80 we get

$$p(\theta) \propto \mathcal{I}^{1/2}(\theta)\,. \tag{2.89}$$

This means that the reference prior, in asymptotically normal models described by one parameter, is equivalent to the Jeffreys prior. As we will discuss shortly, this is not the case for models with many parameters where the two approaches generally produce different results.

2.4.4 Numerical solution

In many cases, equation 2.81 cannot be computed analytically and a numerical solution is required to derive the reference prior. This approach can be applied to one–parameter models and results in a numerical representation of the reference prior in the form of pairs $\{\theta, p(\theta)\}$ of values which can be interpolated and used to define the prior's pdf. The numerical procedure, described below, generally requires that it is computationally possible to simulate data from the sampling distribution (i.e. $d \sim p(d|\theta)$) in order to approximate the outer integral of equation 2.81, and use numerical integration (e.g. quadrature) in order to compute the inner integral in the normalisation of the asymptotic posterior.

Algorithm 1 Numerical reference prior

```
Input: Values of $\theta_t \in \{\theta_1, \dots, \theta_T\}$ for which to compute the reference prior,
    a moderate value of $n$ to simulate the asymptotic posterior, number of
    samples $m$, an arbitrary pseudo–prior (e.g. $h(\theta) = 1$)
Output: Pairs $\{\theta_t, p(\theta_t)\}$
for $t = 1$ to $T$ do
    for $j = 1$ to $m$ do
        Simulate a data set $\{d_{1j} \dots, d_{nj}\} \sim p(d|\theta_t)$,
        Compute the integral $c_j = \int \prod_{i=1}^{n} p(d_{ij}|\theta)h(\theta)d\theta$ numerically,
            where the integration takes place in the prior domain of $\theta$,
        Evaluate $r_j = \log\left[c_j^{-1} \prod_{i=1}^{n} p(d_{ij}|\theta_t)h(\theta_t)\right]$,
    end for
    Compute and store $p(\theta_t) = m^{-1} \sum_{j=1}^{m} r_j(\theta_t)$.
end for
```

2.4.5 Many parameters

So far, we have only discussed cases where the model has a single parameter θ, in which case the reference prior is identical to the Jeffreys prior under the assumption of asymptotic normality. However, the reference prior procedure can be extended to models with more than one parameter where it generally differs from the Jeffreys prior.

In the multivariate case, the reference prior can be decomposed as

$$p(\theta_1, \dots, \theta_D) = p(\theta_D|\theta_1, \dots, \theta_{D-1})p(\theta_{D-1}|\theta_1, \dots, \theta_{D-2}) \dots p(\theta_2|\theta_1)p(\theta_1)\,, \tag{2.90}$$

where D is the number of dimensions and we assumed that the parameters are $\{\theta_1, \dots, \theta_D\}$, in decreasing degree of "importance" or "relevance". The specific ordering of the parameters in terms of their importance matters as different arrangements can result in different reference priors. Given the aforementioned parameter arrangement, the reference prior procedure works by sequentially deriving the aforementioned conditional priors in reverse order, starting with $p(\theta_D|\theta_1,\dots,\theta_{D-1})$ and ending with $p(\theta_1)$. Intuitively, this means that we are seeking the reference prior that maximises the missing information about θ_1, then $\theta_2|\theta_1$, then $\theta_3|\theta_1,\theta_2$, and so on.

In practice, we first fix all parameters but θ_D and we estimate $p(\theta_D|\theta_1,\dots,\theta_{D-1})$ by treating the problem as one–dimensional. Assuming that the prior is proper, then θ_D can be marginalised, and the sampling distribution becomes

$$p(d^{(n)}|\theta_1,\dots,\theta_{D-1}) = \int p(d^{(n)}|\theta_1,\dots,\theta_D)p(\theta_D|\theta_1,\dots,\theta_{D-1})d\theta\,, \tag{2.91}$$

The process is then repeated for the next conditional prior $p(\theta_{D-1}|\theta_1,\dots,\theta_{D-2})$ using $p(d^{(n)}|\theta_1,\dots,\theta_{D-1})$ as the sampling distribution. After D iterations of the above procedure, all conditional priors are known and the reference prior can be computed as their product according to equation 2.90. Although it is possible to use numerical methods in more than one dimension, it often simpler to derive results by employing the asymptotic normality of the posterior distribution when this assumption holds.

Multivariate reference prior under asymptotic normality

To derive reference priors, using the asymptotic normality of the posterior distribution, we first need to understand its conditional structure. In particular, we want to know how we can express the variance and precision of each conditional posterior distribution in terms of the components of the covariance and precision matrices of the unconditional posterior distribution.

Let us assume that the asymptotic posterior distribution can be described as a normal distribution with covariance matrix Σ or precision matrix $P = \Sigma^{-1}$. When the conditions of the Bernstein–von Mises theorem are met, the precision matrix can be written as $P = n\mathcal{I}(\hat{\theta}_n)$, where $\mathcal{I}$ is the Fisher information matrix, n is the sample size, and $\hat{\theta}_n$ is the MLE. Following the usual conventions, we can identify the elements of those matrices using two indices, that is, Σ_{ij} is the element in the intersection of the i–th row and j–th column.

One way to decompose the asymptotic posterior into its conditionals is

$$\begin{aligned} p^*(\theta_1,\dots,\theta_D|d^{(n)}) =& p^*(\theta_D|\theta_1,\dots,\theta_{D-1},d^{(n)}) \\ & \times p^*(\theta_{D-1}|\theta_1,\dots,\theta_{D-2},d^{(n)}) \\ & \dots p^*(\theta_2|\theta_1,d^{(n)})p^*(\theta_1|d^{(n)}) \end{aligned} \tag{2.92}$$

The steps that we need to follow to compute the precision of a conditional $p^*(\theta_j|\theta_1,\dots,\theta_{j-1},d^{(n)})$ are the following:

1. Construct the matrix Σ_j from the upper $j \times j$ sub–matrix of Σ,
2. Compute the inverse matrix $P_j = \Sigma_j^{-1}$,

3. Drop the rows and columns that correspond to the conditional parameters $\theta_1, \dots, \theta_{j-1}$. For 1–D conditionals of the form $p^*(\theta_j|\theta_1, \dots, \theta_{j-1}, d^{(n)})$, this leaves only the lower right element of P_j that we denote as P_{j*} and is equal to the precision of the conditional posterior.

Using the above formula, the reference prior which corresponds to the ordered parameterisation $\{\theta_1, \dots, \theta_D$, in terms of importance or relevance, is

$$p(\theta_1, \dots, \theta_D) = p(\theta_D|\theta_1, \dots, \theta_{D-1}) \dots p(\theta_2|\theta_1)p(\theta_1)\,, \tag{2.93}$$

where

$$p(\theta_D|\theta_1, \dots, \theta_{D-1}) \propto P_{D*}^{1/2}(\theta)\,, \tag{2.94}$$

following equation 2.89, and for $j = 1, \dots, D-1$

$$\begin{aligned} p(\theta_j|\theta_1, \dots, \theta_{j-1}) \propto \exp\Bigg\{ \int \prod_{\ell=j+1}^{D} & p(\theta_\ell|\theta_1, \dots, \theta_{\ell-1}) \\ & \times \log P_{j*}^{1/2}(\theta) d\theta_{j+1} \dots d\theta_D \Bigg\}\,, \end{aligned} \tag{2.95}$$

where we used equations 2.82 and 2.91 to derive the above expression.

In the special case that the functions $P_{j*}^{1/2}(\theta)$ factorise in the form

$$P_{j*}^{1/2}(\theta) \propto f_j(\theta_j) g_j(\theta_1, \dots, \theta_{j-1}, \theta_{j+1}, \dots \theta_D)\,, \tag{2.96}$$

the reference prior is simply

$$p(\theta_1, \dots, \theta_D) = \prod_{j=1}^{D} f_j(\theta_j)\,. \tag{2.97}$$

2–D example

In this example, the joint posterior distribution $p(\theta_1, \theta_2|d^{(n)})$ is asymptotically normal with precision matrix $P = n\mathcal{I}(\theta_n)$ and covariance matrix $\Sigma = P^{-1}$. Without loss of generality, we can order the parameters in increasing importance or relevance as $\{\theta_1, \theta_2\}$ and seek to find the reference prior $p(\theta_1, \theta_2) = p(\theta_2|\theta_1)p(\theta_1)$. According to equation 2.94, the conditional prior $p(\theta_2|\theta_1)$ is given by

$$p(\theta_2|\theta_1) \propto P_{2*}^{1/2}(\theta_1, \theta_2) \propto \mathcal{I}_{22}^{1/2}(\theta_1, \theta_2)\,. \tag{2.98}$$

The marginal prior $p(\theta_1)$ can be derived using equation 2.95, and it is given by

$$p(\theta_1) \propto \exp\left\{ \int p(\theta_2|\theta_1) \log P_{1*}^{1/2}(\theta_1, \theta_2) d\theta_2 \right\}\,, \tag{2.99}$$

where $P_{1*}^{1/2}(\theta_1, \theta_2) = P_{11} - P_{12}P_{22}^{-1}P_{21} \propto \mathcal{I}_{11} - \mathcal{I}_{12}\mathcal{I}_{22}^{-1}\mathcal{I}_{21}$.

So far we have not specified any particular model for this example. In other words, the aforementioned equations hold for any 2–D likelihood function

$p(d^{(n)}|\theta_1, \theta_2)$. To make the example more specific, we choose the sampling distribution to be normal with the likelihood function parameterised by the mean $\theta_1 = \mu$ and standard deviation $\theta_2 = \sigma$, that is,

$$p(d|\mu, \sigma) = \mathcal{N}(d|\mu, \sigma)\,. \tag{2.100}$$

Substituting the above equation into the definition of the Fisher information matrix given by

$$\mathcal{I}_{ij}(\mu, \theta) = -\int p(d|\mu, \sigma)\frac{\partial^2 \log p(d|\mu, \sigma)}{\partial\theta_i \partial\theta_j} dd\,, \tag{2.101}$$

leads to

$$\mathcal{I}_{ij}(\mu, \theta) = \begin{pmatrix} \sigma^{-2} & 0 \\ 0 & 2\sigma^{-2} \end{pmatrix}\,. \tag{2.102}$$

It follows directly that the terms $P_{j*}^{1/2}$ are given by

$$P_{1*}^{1/2}(\mu, \theta) = \sigma^{-1}\,, \quad P_{2*}^{1/2}(\mu, \theta) = \sqrt{2}\sigma^{-1}\,. \tag{2.103}$$

We notice that the above terms factorise into the form of equation 2.96, thus the reference prior is simply

$$p(\mu, \sigma) = p(\sigma|\mu)p(\mu) \propto \sigma^{-1} \times 1 \propto \sigma^{-1}\,. \tag{2.104}$$

It is worth noting that, the alternative ordering of the parameters (i.e. $\theta_1 = \sigma$ and $\theta_2 = \mu$), which prioritises σ over μ, results in the same reference prior in this example. Furthermore, in this case, the bivariate reference prior $p_R(\mu, \sigma) = \sigma^{-1}$ differs markedly from the corresponding Jeffreys prior $p_J(\mu, \sigma) = \sigma^{-2}$. Indeed, even Jeffreys himself criticised his multivariate method, which is known to lead to marginalisation paradoxes (Dawid et al., 1973).

2.5 WEAKLY INFORMATIVE AND REGULARISATION PRIORS

All options that were discussed so far consist of automated methods of generating prior distributions. There is however another class of priors that is distinctly different in purpose than the ones presented above. Those are the weakly informative priors.

In most analyses, we have some limited prior information about the range and possible values that a parameter can take based on domain expertise and the model assumptions. For instance, when constraining the mass of an elementary particle we know that it must be smaller than the mass of macroscopic objects and at the same time it has to be greater than or equal to zero.

This sort of weakly informative knowledge, although not as well quantified as that in the case of *Jeffreys* and *maximum entropy* priors, can still be included in a Bayesian analysis with the hope of guiding the computation by providing regularisation without significantly affecting the outcome. Weakly informative and regularisation priors are used very often in practice, mostly in cases where the data are very informative and the posterior concentrates to a distribution approaching a multivariate normal in accordance with the *Bernstein–von Mises* theorem (Van der Vaart, 2000).

2.6 INFORMATIVE PRIORS

Finally, the last class of priors are the *informative priors* the purpose of which is, unlike *Jeffreys* and *maximum entropy* priors which attempt to minimise the amount of prior information, to include and take into account useful information for an analysis. They are often highly concentrated in parameter space and might have been the outcome (i.e. in the form of a posterior distribution) of a previous experiment or analysis of older data. Their aim is clearly to inform the analysis and often no attempt is made to restrict the amount of information provided.

3 MAKING PREDICTIONS AND EVALUATING MODELS

Tomorrow belongs to those who can hear it coming.
— David Bowie

3.1 MAKING PREDICTIONS

Making predictions is a paramount task for most scientific analyses. Often the parameters of a model are not observable quantities and we have to rely on simulated data to assess the validity of our models. In this section, we will discuss how different kinds of predictive checks can help us avoid various common pitfalls in Bayesian analyses.

3.1.1 Prior predictive checks

A very useful practice, that is always recommended, is to check the predictions of the prior distribution under the specified model (Gelman, Carlin, et al., 2013). Prior predictive checks constitute an elegant way of finding out what kind of data are compatible (i.e. can be described or explained) by our choice of prior and model. The main benefits of this approach are two. First of all, this can help diagnose priors that are either too restrictive or too wide. Furthermore, assuming that the choice of prior distribution is justified, prior predictive checks can help shield against severe cases of *model misspecification* in which no specific set of parameters corresponds to a model that describes the observed data sufficiently well.

In order to assess whether a particular choice of prior distribution is appropriate we need a way to produce simulated data that are consistent with the prior. The Bayesian way of doing this is by sampling the simulated data

$$d_{sim} \sim p(d)\,, \tag{3.1}$$

from the *prior predictive distribution*

$$p(d) = \int p(d|\theta)p(\theta)d\theta\,. \tag{3.2}$$

Generating simulated data using the prior predictive distribution in practice can be done easily by first simulating parameters from the prior distribution

$$\theta_{sim} \sim p(\theta)\,, \tag{3.3}$$

and then simulating the data according to the sampling distribution

$$d_{sim} \sim p(d|\theta_{sim}) \,, \tag{3.4}$$

given the simulated parameters. The simulated pairs (d_{sim}, θ_{sim}) constitute samples from the joint distribution

$$(d_{sim}, \theta_{sim}) \sim p(d, \theta) \,, \tag{3.5}$$

and thus

$$d_{sim} \sim p(d) \,, \tag{3.6}$$

are simulated from the prior predictive distribution.

3.1.2 Posterior predictive checks

Similarly to the prior predictive checks, but this time conditioned on the observed data d_{obs}, one can perform *posterior predictive checks* (Gelman, Carlin, et al., 2013). The latter offer a way of measuring whether a model is able to capture aspects of the data sufficiently well. Just like prior predictive checks that simulate data consistent with the prior, posterior predictive checks on the other hand simulate data that are consistent with the posterior distribution.

In practice, the process of generating simulated data

$$d_{sim} \sim p(d|d_{obs}) \,, \tag{3.7}$$

from the posterior predictive distribution

$$p(d|d_{obs}) = \int p(d|\theta)p(\theta|d_{obs})d\theta \,, \tag{3.8}$$

starts by simulating parameters from the posterior distribution

$$\theta_{sim} \sim p(\theta|d_{obs}) \,. \tag{3.9}$$

It is important to remind the reader that this last step, unless conjugate priors are used, is highly non−trivial and often requires advanced computational algorithms that are the subject of the next chapter. For now, it suffices to understand that the simulation of parameters as described by the relation 3.9 is possible although generally difficult, requiring careful steps. The last step is to generate the simulated data according to sampling distribution

$$d_{sim} \sim p(d|\theta_{sim}) \,, \tag{3.10}$$

given the simulated parameters. d_{sim} then constitute samples from the posterior predictive distribution.

3.2 EVALUATING AND COMPARING MODELS

A key goal of science is to determine which model under consideration better accounts for the observed data. As we will discover shortly, this is generally done by assessing the predictive power of different models. From a Bayesian perspective, there are two ways one can approach this subject. The first uses *Bayes factors* and compares models based on their *prior predictive performance*, meaning their capacity to explain the observed data using only the information encoded in the prior distribution. On the other hand, the second approach uses the notion of *cross–validation* in order to compare models based on their *posterior predictive performance*, meaning their ability to make predictions for out–of–sample data, meaning, future or unseen data, using what we learned from the observed data.

In the *prior predictive* approach, the main quantity that goes into the calculation of the *Bayes factor* is the *prior predictive probability* $p(d|\mathcal{M}_i)$ of the observed data d given a model $\mathcal{M}_i$, also known as the *marginal likelihood* or the model *evidence* (Gregory, 2005; E. T. Jaynes, 2003). Naturally, the *prior predictive* approach is sensitive to the choice of priors. On the other hand, in the *posterior predictive* approach, we compute the *posterior predictive probability* of some subset of the observed data given the rest of the data. Typically, *cross–validation* means that this process is repeated several times, trying to predict different subsets of data, until the entire data set is assessed as held–out data.

3.2.1 Bayes factors

The probability of a model $\mathcal{M}_i$ given the data d can be computed using Bayes' theorem

$$p(\mathcal{M}_i|d) = \frac{p(d|\mathcal{M}_i)p(\mathcal{M}_i)}{p(d)}\,, \tag{3.11}$$

where $p(d|\mathcal{M}_i)$ is the probability of the data given the model, $p(\mathcal{M}_i)$ is the prior probability of the model, and $p(d)$ is the prior predictive probability of the data. We can compare two models, $\mathcal{M}_i$ and $\mathcal{M}_j$, by computing their *odds ratio*

$$\frac{p(\mathcal{M}_i|d)}{p(\mathcal{M}_j|d)} = \frac{p(d|\mathcal{M}_i)p(\mathcal{M}_i)}{p(d|\mathcal{M}_j)p(\mathcal{M}_j)} \tag{3.12}$$

The ratio $BF_{ij} = p(d|\mathcal{M}_i)/p(d|\mathcal{M}_j)$ in the above expression is called the *Bayes factor*. Once the model priors $p(\mathcal{M}_i)$ and $p(\mathcal{M}_j)$ are specified, model comparison using Bayes factors amounts to the calculation of the model evidences $p(d|\mathcal{M}_i)$ and $p(d|\mathcal{M}_j)$.

Despite the apparent simplicity of model comparison using Bayes factors, caution must be exercised when applying the method in real analyses. There are three main reasons for this warning, all of which are sometimes overlooked in practice leading to catastrophic results.

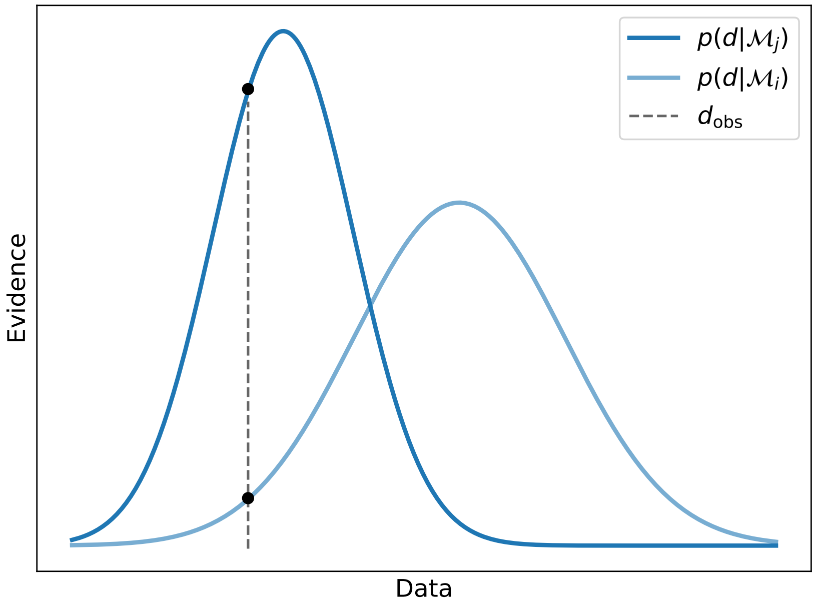

Figure 3.1: *Prior predictive distributions* $p(d|\mathcal{M}_i)$ and $p(d|\mathcal{M}_j)$ for models M_i and M_j respectively. The dashed line, that intersects both distributions, corresponds to the actual observed data. The Bayes factor is simply the ration between the values at the two points of intersection, in this case favouring M_j over M_i. It is clear that for other realisations of the actual observed data (e.g. on the right part of the data vector) the other model would be favoured.

The first reason has to do with the computational difficulty of estimating the model evidence $p(d|\mathcal{M})$, particularly in problems with many parameters. In fact, as we will see in detail in the next part of this thesis, a large collection of methods have been designed with the sole purpose of estimating the model evidence. Therefore, the practitioner has to be familiar with the range of applicability of each method as well as their intrinsic limitations when deciding which technique to use.

The second reason, equally important with the first, is the sensitivity of the model evidence to the choice of prior distribution. This sensitivity is apparent if we just notice that the model evidence is simply the *prior predictive distribution*,

$$p(d|\mathcal{M}) = \int p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) d\theta \,, \tag{3.13}$$

evaluated at the observed data d. However, we argue that this sensitivity is not a weakness of the method as it is often portrayed, but a strength that needs to be properly understood.

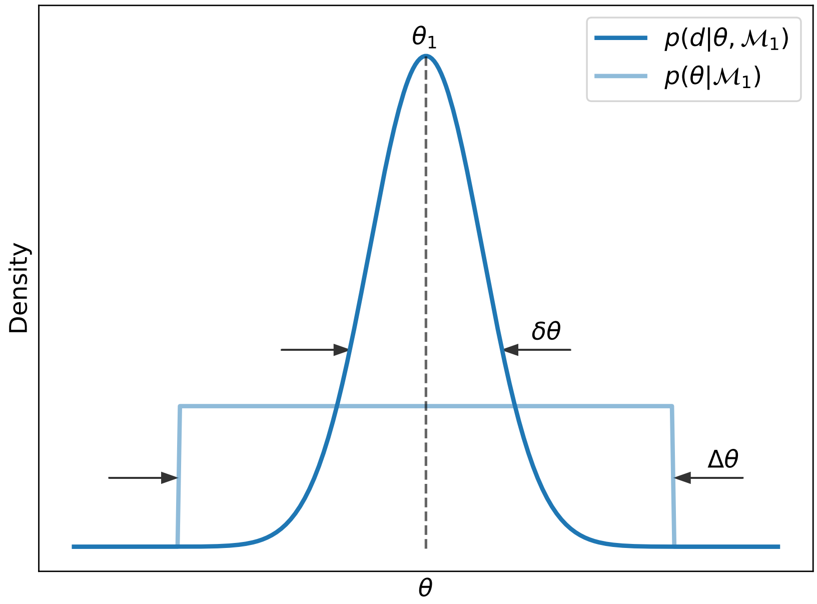

Figure 3.2: The characteristic width $\delta\theta$ of the likelihood function $p(d|\theta, \mathcal{M}_1)$ and $\Delta\theta$ of the prior distribution.

In order to understand the effects of the choice of priors on the Bayes factor, let us consider a simple example. Imagine that we have to compare two models, $\mathcal{M}_1$ with a single scalar parameter θ and $\mathcal{M}_0$ with no free parameters. Furthermore, let us assume that $\mathcal{M}_0$ is nested in $\mathcal{M}_1$, meaning that the more complex model, $\mathcal{M}_1$, reduces to the simpler one, $\mathcal{M}_0$, for a specific parameter value, $\theta = \theta_0$.

Let us now assume that the prior on parameter θ is flat or uniform, such that,

$$p(\theta|\mathcal{M}_1) = \frac{1}{\Delta\theta}, \tag{3.14}$$

and that the likelihood function is sharply peaked around a value θ_1 such that,

$$\int p(d|\theta, \mathcal{M}_1)d\theta = p(d|\theta_1, \mathcal{M}_1) \times \delta\theta, \tag{3.15}$$

where $\delta\theta$ is its characteristic width. It is easy to show that for the case of Gaussian likelihood, centred around θ_1, the characteristic width is simply $\delta\theta = \sqrt{2\pi}\sigma$, where σ is the standard deviation.

The model evidence of $\mathcal{M}_1$ is then simply,

$$\begin{aligned} p(d|\mathcal{M}_1) &= \int p(d|\theta, \mathcal{M}_1)p(\theta|\mathcal{M}_1)d\theta \\ &= p(d|\theta_1, \mathcal{M}_1) \times \frac{\delta\theta}{\Delta\theta}. \end{aligned} \tag{3.16}$$

Since the model $\mathcal{M}_0$ has no free parameters, no integration is required and its model evidence $p(d|\mathcal{M}_0)$ is simply the likelihood function of $\mathcal{M}_1$ evaluated at $\theta = \theta_0$, or,

$$p(d|\mathcal{M}_0) = p(d|\theta_0, \mathcal{M}_1)\,. \tag{3.17}$$

Therefore, the Bayes factor is,

$$B_{10} = \frac{p(d|\mathcal{M}_1)}{p(d|\mathcal{M}_0)} = \frac{p(d|\theta_1, \mathcal{M}_1)}{p(d|\theta_0, \mathcal{M}_1)} \times \frac{\delta\theta}{\Delta\theta}\,. \tag{3.18}$$

The first term in equation 3.18 is the likelihood ratio that always favours the most complex model $\mathcal{M}_1$ since it contains $\mathcal{M}_0$ as a special case. In other words, the first term is always greater than one as the most complex model can always fit the data better than the simpler one.

On the other hand, the second term in equation 3.18 that consists of the ratio of the likelihood width $\delta\theta$ to the prior width $\Delta\theta$ penalises the most complex model $\mathcal{M}_1$, since $\delta\theta < \Delta\theta$, for any "wasted" regions of parameter space that are ruled out by the data. This term quantifies the so–called *principle of parsimony* or *Occam's razor* as it most commonly known. The principle, often attributed to *William of Ockham*, states that "entities should not be multiplied beyond necessity", meaning that between competing models or hypotheses the simplest one is often preferred. Therefore, the Bayes factor will only favour the most complex model (i.e. $\mathcal{M}_1$) only if the likelihood ratio is large enough to overcome the penalty introduced by *Occam's razor*. This intrinsic property of Bayes factors to prefer simpler models, that originates directly from the reliance to the prior distributions, is what makes them so useful in practice.

Now that we understand how sensitive the Bayes factor is to the choice of priors we can discuss some ways that we can shield our analyses against potential problems. First and foremost, Bayesian model comparison can be performed only when *proper* priors are used. By that we mean that *improper* priors such as uniform/flat priors ranging from $-\infty$ to $+\infty$ are not acceptable. Only prior distributions that can be integrated and normalised to unity are *proper* in this sense. However, the use of *proper* priors is not enough, the choice of priors needs to be well–justified too. Priors that are not defined using a principled process (e.g. MaxEnt, Jeffreys, etc.), and sometimes even those that do, can lead to significant deviations in the value of a Bayes factor. For this reason, we urge caution not to over–emphasise the significance of, and instead mostly neglect Bayes factors of $\mathcal{O}(1)$.

The third, and final in our list of reasons, has to do with the open–ended nature of the task model comparison. In particular, model comparison often takes place in the context of a finite set of possible models under investigation with no guarantee whatsoever that one of those models captures perfectly, or even sufficiently, the true data generating process. In that sense, in almost all cases, inference takes place under conditions of *model misspecification*. This brings to mind the saying by *Box*, that "all models are wrong, but some are useful". The fact that the value of a Bayes factor might seem to favour one model over another does not mean that the first model is "correct", only that it is better than the second. Both models might be far from the true data generating process and the Bayes factor will offer generally no indication of that.

3.2.2 Cross–validation

For a model to be useful in practice it must be able to make accurate predictions regarding unseen data. The generalisation uncertainty of a model is often quantified using some measure of the out–of–sample predictive accuracy. A commonly used scoring rule for the out–of–sample predictive accuracy for n data points is the *expected log–pointwise predictive density*,

$$\mathrm{ELPD} = \sum_{i=1}^{n} \int p_t(d_i) \log p(d_i|d_{obs}) dd_i \,, \tag{3.19}$$

where $p_t(d_i)$ is the probability density of the true data generative process which is in general unknown and $p(d_i|d_{obs})$ is the posterior predictive density.

Another useful quantity is the *log–pointwise predictive density*,

$$\mathrm{LPD} = \sum_{i=1}^{n} \log p(d_i|d_{obs}) = \sum_{i=1}^{n} \log \int p(d_i|\theta) p(\theta|d_{obs}) d\theta \,. \tag{3.20}$$

LPD of the observed data d_{obs} is an overestimate of ELPD. We can compute LPD in practice as,

$$\widehat{\mathrm{LPD}} = \sum_{i=1}^{n} \log \left(\frac{1}{J} \sum_{j=1}^{J} p(d_i|\theta_j) \right) \,, \tag{3.21}$$

where $\theta_j \sim p(\theta|d_{obs})$ are samples from the posterior distribution.

Leave–one–out cross–validation

The term cross–validation refers to the practice of estimating the out–of–sample predictive accuracy of a model. In general, the method requires running the analysis multiple times, each time excluding a different portion of the data. The excluded part of the data is then used in order to assess the predictive accuracy of the model. Once the whole dataset is covered, the total accuracy is computed as the average accuracy over all runs,

$$\mathrm{ELPD}_{\mathrm{LOO}} = \sum_{i=1}^{n} \log p(d_i|d_{-i}) \,, \tag{3.22}$$

where,

$$p(d_i|d_{-i}) = \int p(d_i|\theta) p(\theta|d_{-i}) d\theta \,, \tag{3.23}$$

is the leave–one–out predictive density given the data without the i–th datapoint (José M Bernardo & Smith, 2009; Geisser & Eddy, 1979; Gneiting & Raftery, 2007).

Assuming that the n datapoints are conditionally independent in the data generative model, then we can approximate equation 3.23 using draws from the posterior $\theta_j \sim p(\theta|d_{obs})$ and importance weights (Gelfand, Dey, et al., 1992),

$$w_{ij} = \frac{1}{p(d_i|\theta_j)} \propto \frac{p(\theta_j|d_{-i})}{p(\theta_j|d_{obs})}\,, \tag{3.24}$$

leading to the importance sampling leave–one–out predictive density,

$$p(d_i|d_{-i}) = \frac{\sum_{j=1}^{J} w_{ij} p(d_i|\theta_j)}{\sum_{j=1}^{J} w_{ij}} = \frac{1}{\frac{1}{J}\sum_{j=1}^{J}[p(d_i|\theta_j)]^{-1}}\,. \tag{3.25}$$

However the posterior $p(\theta|d_{obs})$ is likely to have a smaller variance than then $p(\theta|d_{-i})$ distributions leading to insufficient overlap between their typical sets and high–variance importance weights. Ionides (2008) showed that truncating the importance weights,

$$\tilde{w}_{ij} = \min\left(w_{ij}, \sqrt{J}\,\bar{w}_i\right)\,, \tag{3.26}$$

where

$$\bar{w}_i = \frac{1}{J}\sum_{j=1}^{J} w_{ij}\,, \tag{3.27}$$

leads to provable finite–variance weights at the cost of introducing bias. Vehtari et al. (2017) proposed instead to fit a generalised Pareto distribution to the upper tail of the importance weights, in order to smooth the weights, leading to improved estimates.

WAIC

The *Watanabe–Akaike* or *widely applicable information criterion (WAIC)* (Watanabe & Opper, 2010) offers a different way to approximate ELPD and is defined as,

$$\widehat{\mathrm{ELPD}}_{\mathrm{WAIC}} = \widehat{\mathrm{LPD}} - \hat{p}_{\mathrm{WAIC}}\,, \tag{3.28}$$

where,

$$\hat{p}_{\mathrm{WAIC}} = \sum_{i=1}^{n} \mathrm{Var}_{\theta\sim p(\theta|d_{obs})}[\log p(d_i|\theta)]\,, \tag{3.29}$$

is the estimated effective number of parameters expressed as the posterior variance of the log predictive density of each datapoint. Equation 3.29 can be computed using posterior samples.

3.2.3 Model averaging

Standard practice ignores model uncertainty and instead focuses on the most probable models as deduced by their Bayes factors. This approach leads to over–confident estimates and ignores the fact that often more than one model can describe the data sufficiently. There is, however, a different approach that we can follow in order to deal with the model uncertainty and properly account for the plethora of plausible models, called *Bayesian model averaging* (Madigan, Raftery, et al., 1996).

Let $\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_M$ be a set of M models with posterior model probabilities $p(\mathcal{M}_1|d)$, $p(\mathcal{M}_2|d)$, …, $p(\mathcal{M}_M|d)$ and posterior distributions $p(\theta|d, \mathcal{M}_1)$, $p(\theta|d, \mathcal{M}_2)$, …, $p(\theta|d, \mathcal{M}_M)$ respectively. Then *Bayesian model averaging* relies on the *marginal* posterior density,

$$p(\theta|d) = \sum_{i=1}^{M} p(\theta|d, \mathcal{M}_i) p(\mathcal{M}_i|d)\,, \tag{3.30}$$

which is no longer conditioned on a model.

Moreover, predictions can be made by averaging over all models, weighted proportional to their posterior model probabilities, thereby incorporating model uncertainty using the marginal posterior predictive density,

$$p(d|d_{\text{obs}}) = \sum_{i=1}^{M} p(d|d_{\text{obs}}, \mathcal{M}_i) p(\mathcal{M}_i|d_{\text{obs}})\,, \tag{3.31}$$

where d_{obs} are the available observed data and d are the new predicted data. Madigan & Raftery (1994) note that averaging over all models in this fashion leads to higher predictive accuracy than using any single model individually.

Part II

BAYESIAN COMPUTATION

4 PRINCIPLES OF BAYESIAN COMPUTATION

Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin.

— *John von Neumann*

This chapter introduces the various methods that are used in practice in order to tackle the computational challenges of Bayesian analyses. We begin this journey by discussing some fundamental ideas about the geometry of high–dimensional probability distributions, while gradually introducing the concepts and algorithms that constitute the modern mathematical machinery of Bayesian computation.

4.1 EXPECTATION VALUES

Probability theory teaches us the only well defined way to extract information from probability distributions is through expectation values. By this term, we mean high–dimensional integrals of the form

$$\mathbb{E}_p[f] = \int f(\theta)p(\theta)d\theta\,, \tag{4.1}$$

where $p(\theta)$ is the probability density function that often corresponds to the posterior density for problems of scientific inference, θ signifies the parameters of the distribution, and $f(\theta)$ is the function that we aim to integrate. In this sense, an expectation value of a function $f(\theta)$ over a probability distribution $p(\theta)$ is technically a functional of the product of the function and the probability density.

To see why expectation values hold such a central role in scientific parameter inference, let us discuss a few characteristic and common examples that a scientist often has to compute.

- **Mean value** – Perhaps the most commonly computed expectation value is the mean value. This can be calculated by choosing the function to be $f(\theta) = \theta$, the expectation value then reduces to

$$\mu \equiv \mathbb{E}_p[\theta] = \int \theta\, p(\theta)d\theta\,. \tag{4.2}$$

- **Variance** – One might also want to compute higher moments of the probability distribution. The first moment is the variance that corresponds to the following expectation value

$$\sigma^2 \equiv \mathbb{E}_p\left[(\theta-\mu)^2\right] = \int (\theta-\mu)^2\, p(\theta) d\theta\,. \tag{4.3}$$

- **Marginal distributions** – Even marginal distribution can be thought of as expectation values. In this case, the function f corresponds to a conditional distribution, for instance

$$p(\phi) \equiv \mathbb{E}_p\left[p(\phi|\theta)\right] = \int p(\phi|\theta) p(\theta) d\theta\,. \tag{4.4}$$

4.2 QUADRATURE AND UNIFORM GRIDS

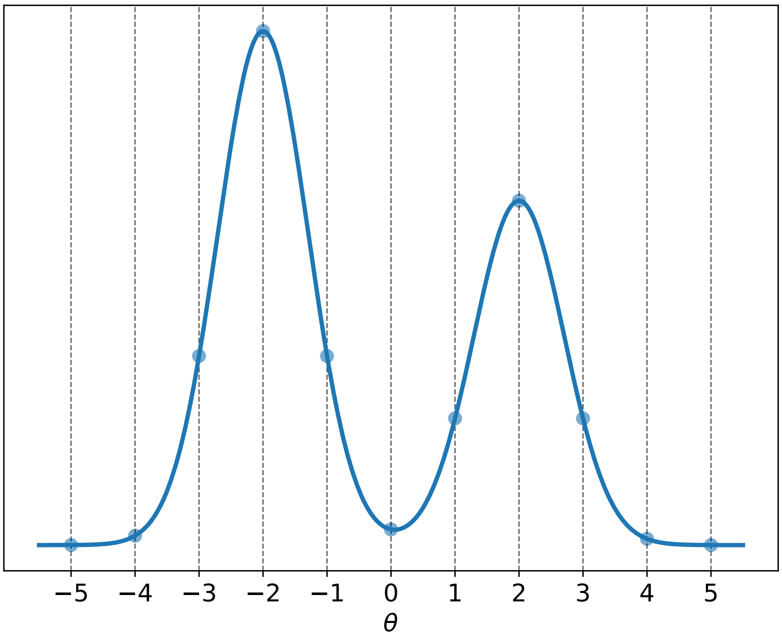

Figure 4.1: Uniform grid approximation of 1–dimensional probability distribution.

Having discussed a number of examples of commonly used expectation values, we can now turn our attention to the methods that are used for their computation. As we mentioned before, these expectation values are defined as high–dimensional integrals. As those integrals are not generally tractable analytically, one might attempt to approximate their value by means of a Riemann sum over a discreet grid of n points:

$$\mathbb{E}_p[f] = \int f(\theta) p(\theta) d\theta \approx \sum_{i=1}^{n} f(\theta_i) p(\theta_i) \Delta\theta_i\,, \tag{4.5}$$

where

$$\Delta\theta_i = \theta_{j+1} - \theta_j\,, \tag{4.6}$$

is simply the interval between two subsequent points, θ_j and θ_{j+1} on the underlying grid, and

$$\theta_i = \frac{\theta_{j+1} + \theta_j}{2}\,, \tag{4.7}$$

is just the mid–point between θ_j and θ_{j+1}.

In principle, this idea can be extended to high dimensions by replacing the 1–dimensional intervals $\Delta\theta_i$ with D–dimensional hypercubes. Figure 4.2 shows one such example for a 2–dimensional probability distribution. However, as the number of dimensions increases, one immediately has to face a significant difficulty, the *curse of dimensionality* (Bellman, 1966). Already in 2 dimensions we require n^2 grid points to approximate the distribution. As it turns out, the number of grid points required for the evaluation of the Riemann sum increases exponentially with the number of dimensions, rendering this method of computing expectation values unusable for $D > 3$. Overcoming the difficulties imposed by the *curse of dimensionality* is one of the key goals of *probabilistic computing*.

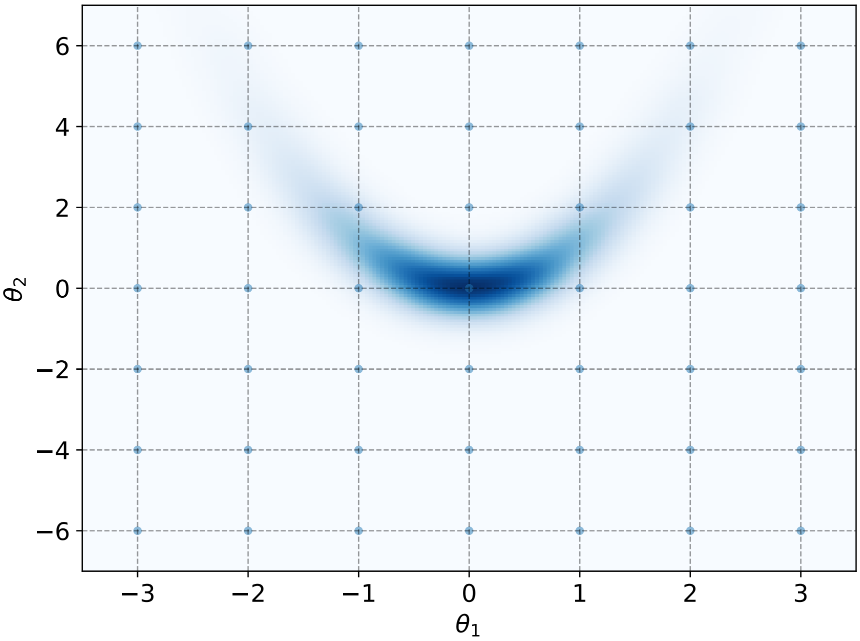

Figure 4.2: Uniform grid approximation of 2–dimensional probability distribution.

In order to reduce the computational cost of estimating expectation values in high dimensions, we need to find a way to focus our effort and computation only on those regions of parameter space that are relevant for the integral that we aim to evaluate. One simple idea would be to remove any points of the grid that the integrand $f(\theta)p(\theta)$ is very close to zero. Applying this technique would certainly reduce the total computational cost since only a few grid–cells have a non–negligible value of $f(\theta)p(\theta)$ as shown in Figure 4.3. The problem that we face however is that by focusing our attention on $f(\theta)p(\theta)$ we ignore a key factor in the estimation of any expectation value, the *volume*.

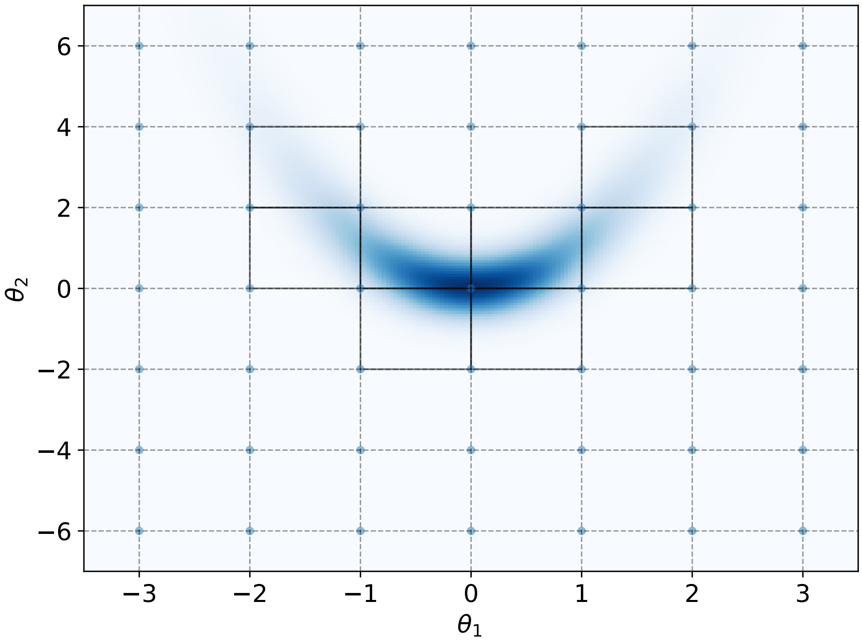

Figure 4.3: Uniform grid approximation of 2–dimensional probability distribution with highlighted the grid–cells that actually contribute to the calculation of an expectation value.

4.3 GEOMETRY OF HIGH–DIMENSIONAL SPACES

The concepts of volume and distance in high–dimensional spaces defy our everyday intuition in ways that matter for the computation of expectation values. To understand this, we will go through an example that illustrates these peculiar effects.

Let us assume that we inscribe a circle of radius R inside a square of side $2R$. We are interested in computing the area of the circle as a fraction of that of the square. We can get to the result easily using basic geometry, in particular, the ratio of the two areas is

$$\frac{A_{\text{circle}}}{A_{\text{square}}} = \frac{\pi R^2}{(2R)^2} = \frac{\pi}{4}\,. \tag{4.8}$$

We can now extend the same problem into three dimensions, in which we have a sphere of radius R inscribed in a cube of side $2R$. The ratio of the volume of the sphere to the volume of the cube is simply

$$\frac{V_{\text{sphere}}}{V_{\text{cube}}} = \frac{\frac{4}{3}\pi R^3}{(2R)^3} = \frac{\pi}{6} \,. \tag{4.9}$$

By comparing equations 4.8 and 4.9 one realises that the volume ratio has decreased going from 2 dimensions to 3. We will now show that this result in fact holds for any number of dimensions D. In D dimensions, the volume of a hyper–sphere is given by

$$V_{\text{sphere}} = \frac{\pi^{D/2}}{\Gamma\left(\frac{D}{2}+1\right)} R^D \,, \tag{4.10}$$

where Γ is Euler's gamma function which extends the factorial operation to non–integer arguments and satisfies

$$\Gamma(D) = (D-1)! \,, \tag{4.11}$$

for positive integer D, and

$$\Gamma\left(D+\frac{1}{2}\right) = \left(D-\frac{1}{2}\right) \times \left(D-\frac{3}{2}\right) \times \cdots \times \frac{1}{2} \times \pi^{1/2} \,, \tag{4.12}$$

for non–negative integer D.

The volume of a hyper–cube in D dimensions is simply

$$V_{\text{cube}} = (2R)^D \,. \tag{4.13}$$

Taking the ratio of the terms of equations 4.10 and 4.13 yields

$$\frac{V_{\text{sphere}}}{V_{\text{cube}}} = \frac{\pi^{D/2}}{2^D \Gamma\left(\frac{D}{2}+1\right)} \,. \tag{4.14}$$

Figure 4.4 shows the ratio of the volume of a hypersphere to a hypercube as a function of the number of dimensions D. As the number of dimensions D increases, the volume ratio of equation 4.14 asymptotically approaches 0. This means that in high dimensions, almost all of the volume of a hypercube is concentrated in the corners.

4.4 CONCENTRATION OF MEASURE

As we will see shortly, the strange behaviour of volume is of paramount importance in the calculation of expectation values over probability distributions. To understand this one need to think not about the probability density but instead about the probability mass. When evaluating an expectation

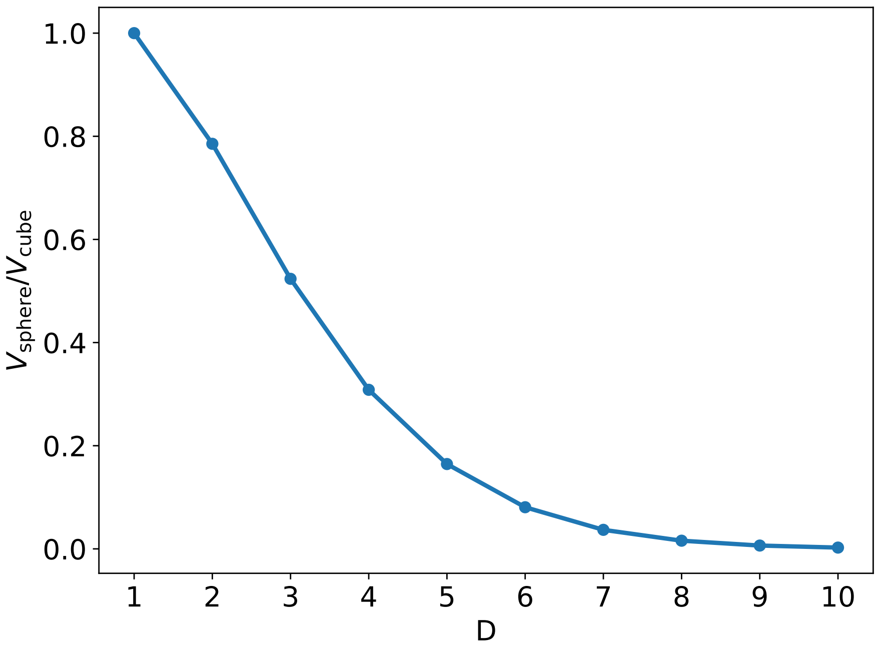

Figure 4.4: The ratio of the volume of a hyper–sphere of radius R to the volume of a hyper–cube of edge size $2R$ as function of the number of dimensions D.

value, not all regions of parameter space are contributing equally to the value of the integral. In fact, the contribution from some regions of parameter space dominates the calculation. We only need to take a look into the form of the expectation value integral to notice that is essentially the product of three terms that contributes. These terms are the function $f(\theta)$, the probability density function $p(\theta)$, as well as the differential volume element $d\theta$. In other words, it is the product of these three terms and their dependence on θ that determines the value of the integral. Assuming that the function $f(\theta)$ is well behaved, we can ignore its presence for a while.

For the sake of simplicity let us assume that the probability distribution is characterised by an D–dimensional Gaussian probability density function

$$p(\theta) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\theta-\mu)^T\Sigma^{-1}(\theta-\mu)\right], \tag{4.15}$$

where μ is the mean and Σ is the covariance matrix of the distribution. Without loss of generality let us also assume that the density is centred at zero (i.e. $\mu = 0$) and the covariance matrix diagonal with the elements of its diagonal equal to σ^2, meaning that equation 4.15 simplifies into

$$p(\theta) = \frac{1}{\sqrt{(2\pi)^D}\sigma^D} \exp\left(-\frac{|\theta|^2}{2\sigma^2}\right). \tag{4.16}$$

Assuming further spherical coordinates, the density only depends on the magnitude r of the θ parameter vector

$$p(r) = \frac{1}{(2\pi)^{\frac{D}{2}}\sigma^D} \exp\left(-\frac{r^2}{2\sigma^2}\right) . \tag{4.17}$$

Keep in mind that $p(r)$ is not a probability density function of the magnitude $r = |\theta|$, but a D–dimensional density of θ.

Let us now turn our attention to the differential volume element dV. Differentiating equation 4.10 that provides the volume of the hyper–sphere we get

$$dV = \frac{D\pi^{D/2}}{\Gamma\left(\frac{D}{2}+1\right)} r^{D-1} dr . \tag{4.18}$$

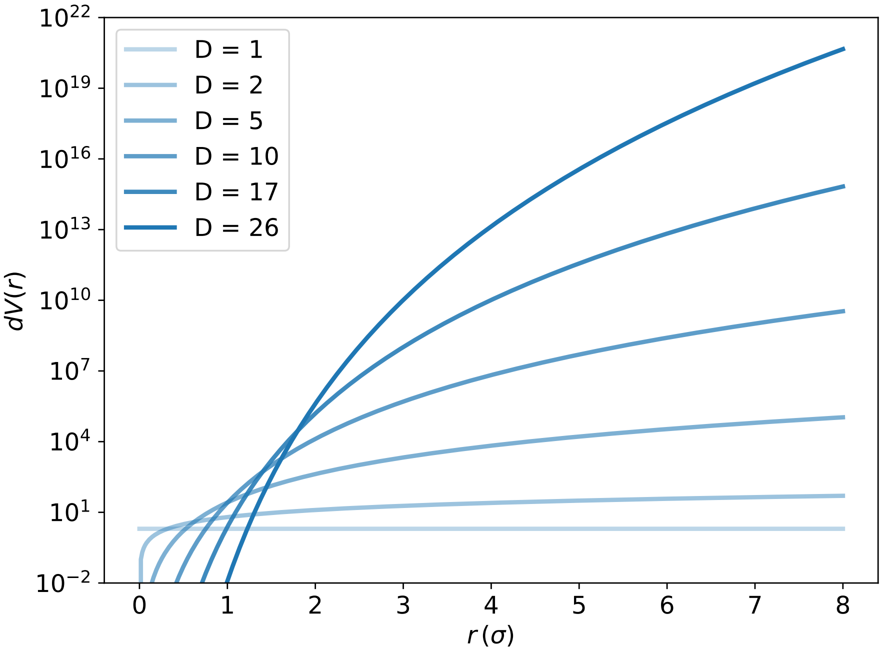

Figure 4.5: Scaling of differential volume with the number of dimensions as a function of distance.

The differential probability mass $dm(r)$ is then just the product of $p(r)$ and dV given by equations 4.17 and 4.18 respectively

$$dm(r) = \frac{Dr^{D-1}}{\Gamma\left(\frac{D}{2}+1\right) 2^{\frac{D}{2}} \sigma^D} \exp\left(-\frac{r^2}{2\sigma^2}\right) dr . \tag{4.19}$$

The differential mass $dm(r)$ has a clear physical meaning, that of the probability mass enclosed in a hyper–spherical shell of radius r and width dr. The probability mass differential $dm(r)$ peaks (i.e. is maximised) at the typical radius

$$r_{\text{peak}} = \sqrt{D-1}\sigma . \tag{4.20}$$

Equation 4.20 indicates that while in 1−D the probability mass peaks at $r_{\text{peak}} = 0$, in higher dimensions this is not the case. As the number of dimensions increases the radius in which the probability mass peaks moves to greater distances. Table 1 shows the typical radius of the probability mass for a different number of dimensions for our problem. This is a direct consequence of the rapid increase of the differential volume for large r values.

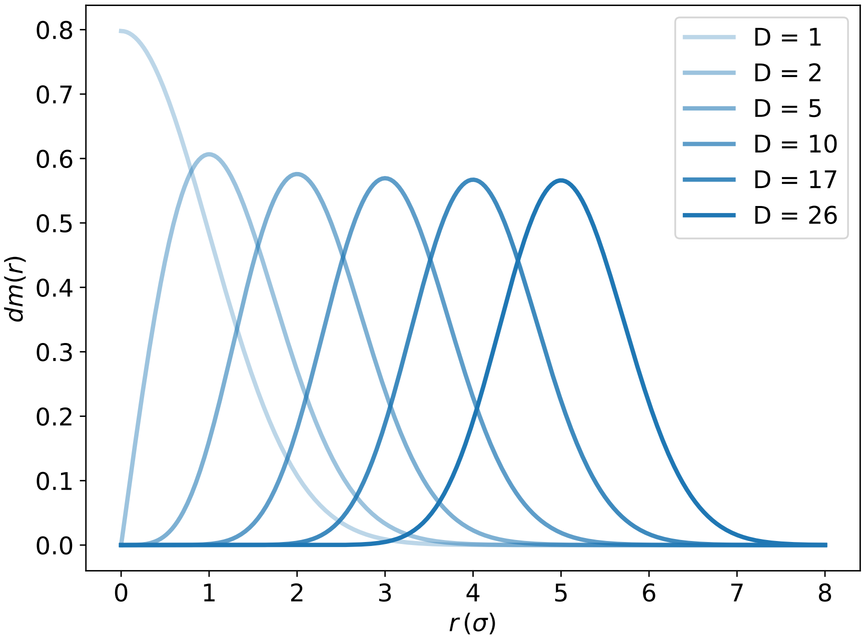

Figure 4.6: Scaling of differential probability mass with the number of dimensions as a function of distance.

Table 1: The typical radius r_{peak} as function of the number of dimensions D.

Number of dimensions D	Typical radius r_{peak}
1	0
2	1σ
5	2σ
10	3σ
17	4σ
26	5σ

In general, we expect the probability mass to form a hyper−shell of mean radius

$$r_{\text{mean}} \equiv \mathbb{E}_p[r] = \int_0^{+\infty} r\,dm(r) \tag{4.21}$$

and width (i.e. standard deviation)

$$\Delta r \equiv \sqrt{\mathbb{E}_p[(r - r_{\text{mean}})^2]} = \sqrt{\int_0^{+\infty} (r - r_{\text{mean}})^2 \, dm(r)} \tag{4.22}$$

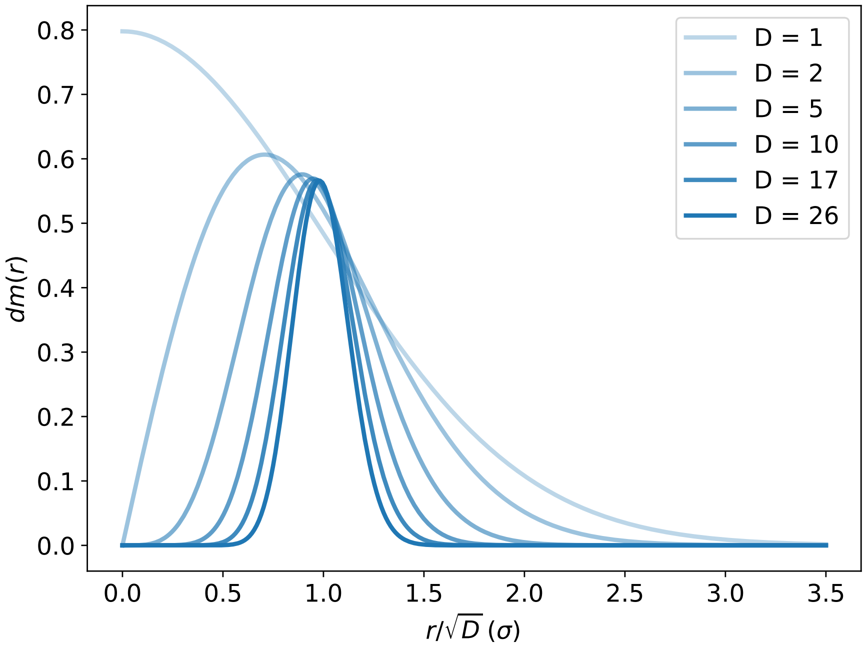

Figure 4.7: Scaling of differential probability mass with the number of dimensions as a function of distance normalised by the square root of the number of dimensions.

4.5 TYPICAL SET

The qualitative conclusions of the previous section are general and hold for any continuous probability distribution. The probability mass does not concentrate close to the mode where the probability density is high as there is not sufficient volume there. On the other hand, it does not concentrate on large distances because the density vanishes. Instead, it compromises on some region of intermediate distance surrounding the mode, as shown in Figure 4.8. This region is called the *typical set*, and has the form of a high–dimensional thin hyper–shell surrounding the mode as shown in Figure 4.9. In high dimensions, the typical set exhibits the effect of *concentration of measure* (Ledoux, 2001) illustrated in Figure 4.7.

The concept of the typical set is not only important for properly understanding probability distributions, but also for developing new computational methods. The notion of the typical set is originally borrowed from the field

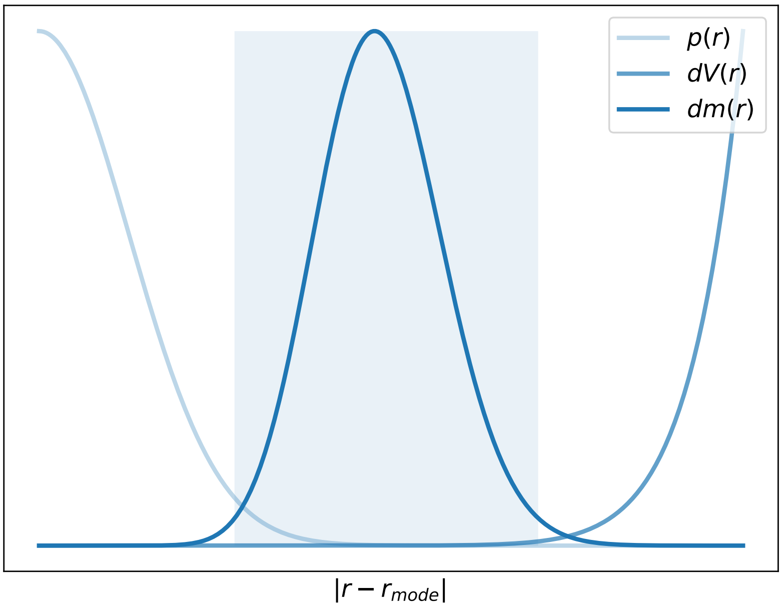

Figure 4.8: Illustration of the typical set as the region in parameter space that the product of probability density and differential volume is non–negligible.

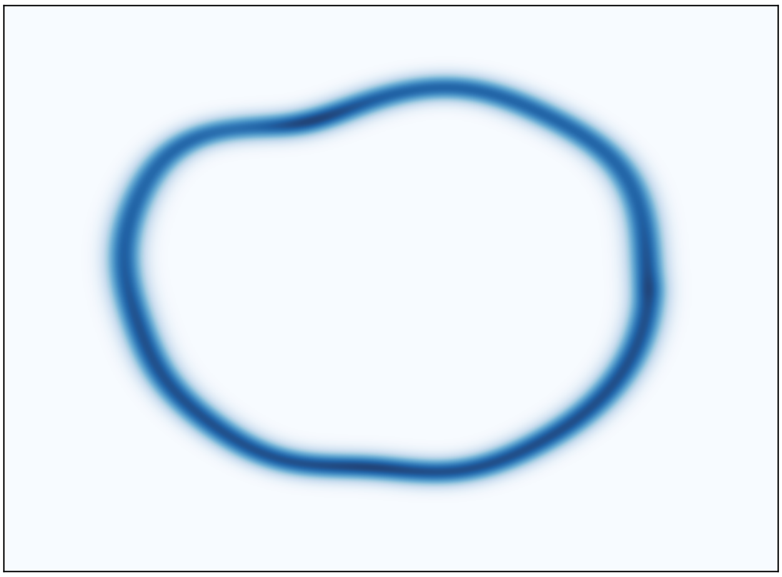

Figure 4.9: Illustration of the typical set as a thin hyper–shell surrounding the mode of the probability distribution.

of information theory, in which one of the main tasks is to compress and encode a *message* with as few *words* as possible. In probability theory, the typical set defines the most efficient way to compress a probability distribution by focusing on a limited region of parameter space. As we will see in the next sections, the task of developing powerful and effective computational methods comes down to how efficiently we can locate and approximate the typical set of a probability distribution.

4.6 LAPLACE APPROXIMATION

Before we move on to stochastic estimators of expectation values let us first discuss another simple deterministic method, called *Laplace approximation*, that, unlike *quadrature in a uniform grid*, can extend to higher dimensions (Tierney & Kadane, 1986). The *Laplace approximation* makes a very strong assumption about the target probability distribution. In particular, it assumes that it can be sufficiently described by a Gaussian probability density, similar to equation 4.15. The mean of the Gaussian density is determined at the point of the mode of the target density

$$\mu = \underset{\theta}{\arg\max}\; p(\theta)\,, \tag{4.23}$$

and the precision matrix Σ^{-1} (i.e. inverse of the covariance matrix Σ) is given by the second-order derivatives of the negative logarithm of the target probability density function evaluated at the mode,

$$\left(\Sigma^{-1}\right)_{ij} = -\frac{\partial^2}{\partial\theta_i\partial\theta_j}\log p(\theta)\bigg|_{\theta=\mu}\,. \tag{4.24}$$

The reasoning behind this approach is quite simple, one effectively performs a Taylor expansion of the logarithm of the density, up to second order, around the *maximum a posteriori* point μ,

$$\log p_L(\theta) = \log p(\theta=\mu) - \frac{1}{2}(\theta-\mu)^T\Sigma^{-1}(\theta-\mu) + \dots\,, \tag{4.25}$$

where the first order term simply vanishes because we evaluate the expansion around the maximum. For this reason, this very common method is often called the saddle–point approximation. Expectation values can then be determined using the Gaussian density $p_L(\theta) = \mathcal{N}(\theta|\mu,\Sigma)$ in place of the target density $p(\theta)$ in the formula for the expectation value 4.1,

$$\mathbb{E}_{p_L}[f] = \int f(\theta)\mathcal{N}(\theta|\mu,\Sigma)d\theta\,. \tag{4.26}$$

The quality of the *Laplace approximation* is determined by the overlap of the typical set of the target distribution with that of the Gaussian approximation. The greater the overlap, the more accurate the approximation will be.

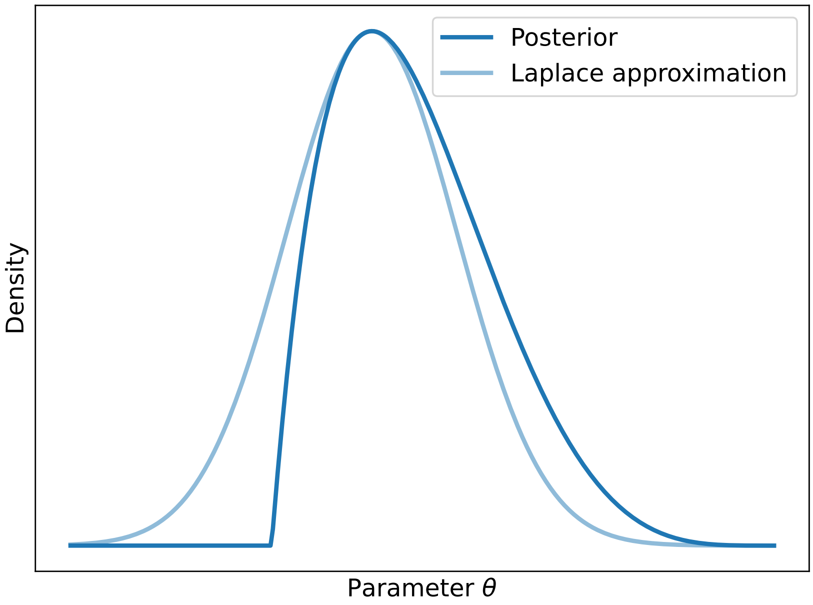

Figure 4.10: Illustration of the Laplace approximation to a skewed probability density.

4.7 MONTE CARLO ESTIMATORS

Another type of estimators is *stochastic* estimators, and in particular *Monte Carlo* estimators (Brooks et al., 2011) that rely on a collection of independent points or samples,

$$\{\theta_1, \ldots, \theta_n\} \in \Theta\,, \tag{4.27}$$

from the distribution $p(\theta)$, such that the ensemble average of a function $f(\theta)$,

$$\hat{f}_n^{MC} = \frac{1}{n} \sum_{i=1}^{n} f(\theta_n)\,, \tag{4.28}$$

asymptotically converges to the corresponding expectation value

$$\lim_{n \longrightarrow \infty} \hat{f}_n^{MC} = \mathbb{E}_p[f(\theta)]\,. \tag{4.29}$$

The asymptotic result of equation 4.29 is not particularly useful as a computational algorithm will never be able to produce infinite samples. Fortunately, the behaviour of Monte Carlo estimators can be quantified even for finite samples.

For any square–integrable (i.e. both $\mathbb{E}_p[f]$ and $\mathbb{E}_p[f^2]$ exist and are finite) real–valued function $f(\theta)$, the Monte Carlo estimator satisfies the *central limit theorem*,

$$\frac{\hat{f}_n^{MC} - \mathbb{E}_p[f(\theta)]}{\text{MC–SE}_n[f]} \sim \mathcal{N}(0, 1)\,, \tag{4.30}$$

where $\text{MC–SE}_n[f]$ is the *Monte Carlo Standard Error* defined as,

$$\text{MC–SE}_n[f] = \sqrt{\frac{\text{Var}_p[f]}{n}}\,. \tag{4.31}$$

This means that we can estimate the expected number of samples that is required to reach a certain level of precision for our estimates.

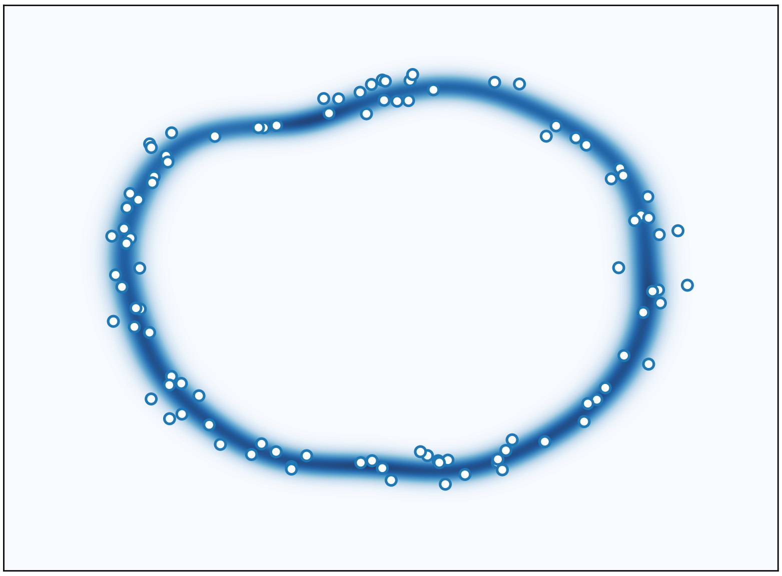

Figure 4.11: Illustration of the typical set including samples generated using exact Monte Carlo sampling.

Another interesting property of Monte Carlo estimators is that their precision, as quantified by the Monte Carlo Standard Error of equation 4.31, does not depend on the dimensionality of the problem but relies only on the number n of samples instead. This means that Monte Carlo estimators can be applied even in high–dimensional problems. This insensitivity to the *curse of dimensionality* is directly related to the fact that the Monte Carlo samples are already distributed in the typical set as shown in Figure 4.11. As we will discover shortly, once we discuss more advanced methods, the difficult part is to get the samples to the typical set in the first place.

We can also think of the Monte Carlo samples as a stochastic grid where the computation is mostly focused in the regions of parameter space that contribute to the computation of the expectation value. Starting from the Monte Carlo estimator,

$$\hat{f}_n^{MC} = \frac{1}{n}\sum_{i=1}^{n} f(\theta_n)\,, \tag{4.32}$$

and manipulate it into the quadrature form

$$\hat{f}_n^{MC} = \sum_{i=1}^{n} f(\theta_i)p(\theta_i)\frac{1}{np(\theta_i)}\,, \tag{4.33}$$

where $\Delta\theta_i = 1/np(\theta_i)$ is the effective volume of each sample.

Monte Carlo estimators are very powerful methods assuming that one can generate independent samples from the target distribution. However, in most interesting and realistic cases, this is not feasible. In that case, one has to rely to alternative methods.

4.8 IMPORTANCE SAMPLING

One alternative method to exact Monte Carlo sampling, that does not rely on exact samples from the target distribution but instead requires an *auxiliary distribution* is *importance sampling*. Importance sampling estimators use samples from the auxiliary distribution and correct for any deviation from the typical set of the target distribution using *weighting factors*. Although Kloek & Van Dijk (1978) is typically credited with introducing importance sampling to statistics, there are references to it in statistical physics as early as 1949 (Goertzel, 1949; Kahn & Harris, 1951).

In order to derive the *importance weights* necessary for the computation of the expectation values we start with the definition of the expectation value and do some re–arrangements,

$$\begin{aligned}\mathbb{E}_p[f(\theta)] &= \int_{\Theta} f(\theta)p(\theta)d\theta \\ &= \int_{\Theta} f(\theta)\frac{p(\theta)}{q(\theta)}q(\theta)d\theta \\ &= \mathbb{E}_q\left[f(\theta)\times\frac{p(\theta)}{q(\theta)}\right]\,.\end{aligned} \tag{4.34}$$

We can now estimate the expectation value,

$$\hat{f}_n^{IS} = \frac{1}{n}\sum_{i=1}^{n} w(\tilde{\theta}_i)f(\tilde{\theta}_i)\,, \tag{4.35}$$

using samples from the *auxiliary* distribution,

$$\{\tilde{\theta}_1, \dots, \tilde{\theta}_n\} \sim q(\tilde{\theta})\,, \tag{4.36}$$

and importance weights given by,

$$w(\tilde{\theta}) = \frac{p(\tilde{\theta})}{q(\tilde{\theta})}\,. \tag{4.37}$$

For any square–integrable real–valued function $f(\theta)$, the importance sampling estimator satisfies the *central limit theorem*,

$$\frac{\hat{f}_n^{IS} - \mathbb{E}_p[f(\theta)]}{\text{IS–SE}_n[f]} \sim \mathcal{N}(0,1)\,, \tag{4.38}$$

where MC–SE$_n[f]$ is the *Importance Sampling Standard Error* defined as,

$$\text{IS–SE}_n[f] = \sqrt{\frac{\text{Var}_q[wf]}{n}}\,. \tag{4.39}$$

By comparing the expression 4.39 for IS–SE to the respective expression 4.31 for the *Monte Carlo Standard Error* we can define the *Effective Sample Size (ESS)*,

$$\text{ESS}_n[f] = \frac{\text{Var}_q[wf]}{\text{Var}_p[f]}n\,, \tag{4.40}$$

as the effective number of exact samples that contain the same amount of information as the n samples and their importance weights.

It is important to mention here that if the target or auxiliary distribution is known only up to a normalisation factor, for instance if the computation of the normalisation constant is very costly, then the importance weights have to be normalised such that,

$$\sum_{i=1}^{n} w(\tilde{\theta}_i) = 1\,, \tag{4.41}$$

for the aforementioned estimators to be valid.

The quality of the importance sampling estimator is determined by the amount of overlap between the auxiliary and target distribution. Samples from the auxiliary distribution residing in regions of high overlap will receive large importance weights and those residing in regions of little or no overlap will receive small importance weights.

As we showed in previous sections, in low dimensions the typical set is broad so we should expect that the construction of importance sampling estimators for low dimensional cases to be a feasible procedure. In higher dimensions however, the typical shell is very thin thus complicating the choice of auxiliary distributions with significant overlap with the target distribution.

It is common in practice to assume that an auxiliary distribution with broader density tails than the target distribution would be sufficient to construct an importance sampling estimator. Although the reasoning of this idea is appealing, it can however be misleading as it does not extend to higher dimensions in which the typical set becomes the central object of interest and not the probability density.

It is useful to define a measure of the quality of an importance sampling estimator. A straightforward choice would be to define the *importance sampling effective sample size*,

$$N_{\text{eff}} = \frac{\left(\sum_{i=1}^{n} w(\tilde{\theta}_i)\right)^2}{\sum_{i=1}^{n} w(\tilde{\theta}_i)^2}\,. \tag{4.42}$$

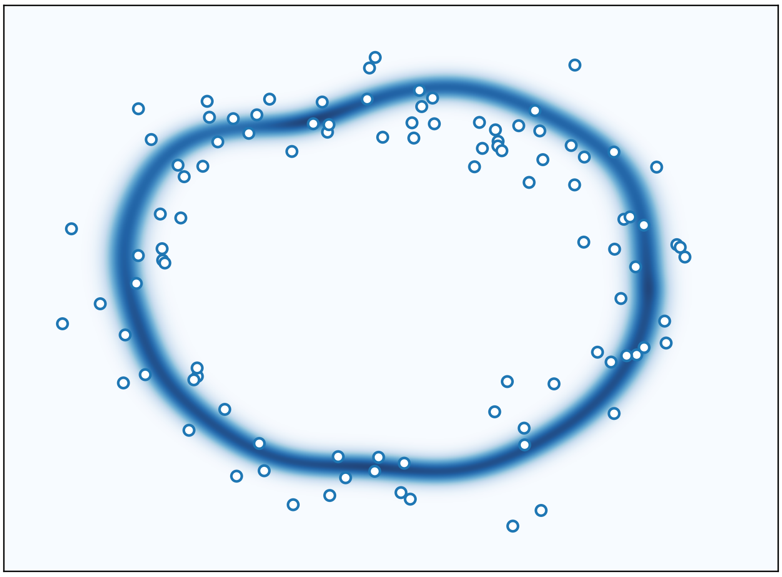

Figure 4.12: Illustration of the typical set including samples generated using an unsuitable importance density. In high dimensions the typical set corresponds to a very thin shell and it is difficult to achieve sufficient overlap between the typical set of the auxiliary and target distribution. Here, this is depicted by samples that do not reside in the typical set of the target and will thus have low importance weights.

Equation 4.42 is just a heuristic diagnostic and it should not be confused with equation 4.40.

4.9 MARKOV CHAIN MONTE CARLO

Importance sampling estimators trade the ability to produce exact samples from the target distribution with weighted samples from an auxiliary distribution. On the other hand, *Markov chain Monte Carlo (MCMC)* estimators replace the exact samples with *correlated* samples generated by a *Markov chain* (Brooks et al., 2011; Gilks, Richardson, et al., 1995).

Therefore, the key idea in MCMC is to explore the typical set using a sequence of local steps. Starting a point θ_1 in parameter space Θ, the next point θ_2 is chosen stochastically in the neighbourhood of θ_1. Then the process is repeated for the next point θ_3 in the neighbourhood of θ_2 and so on. At the end, we have generated a chain of n samples that is *Markov*, meaning that each sample conditionally depends only on the previous one,

$$P(\theta_n|\theta_1, \ldots, \theta_{n-1}) = P(\theta_n|\theta_{n-1}) \,. \tag{4.43}$$

More formally, the Markov chain can be generated by repeatedly sampling from a conditional probability distribution on the product space $\Theta \times \Theta$, known as *Markov transition probability* $T(\theta'|\theta)$. Given an initial point θ_1, sampling from the *Markov transition probability* $T(\theta'|\theta_1)$ returns sample θ_1. We can thus construct a sequence of transitions,

$$\begin{aligned} \theta_2 &\sim T(\theta_2|\theta_1) \\ \theta_3 &\sim T(\theta_3|\theta_2) \\ &\ldots \\ \theta_n &\sim T(\theta_n|\theta_{n-1}) \,, \end{aligned} \tag{4.44}$$

that constitute the Markov chain $\{\theta_1, \theta_2, \ldots, \theta_n\}$. The samples of the Markov chain are not independent, but they are *correlated*. The reason for this is their sequential origin i.e. θ_n depends on θ_{n-1} which depends on θ_{n-2} so even samples that are not right next to each other in the Markov chain can be correlated.

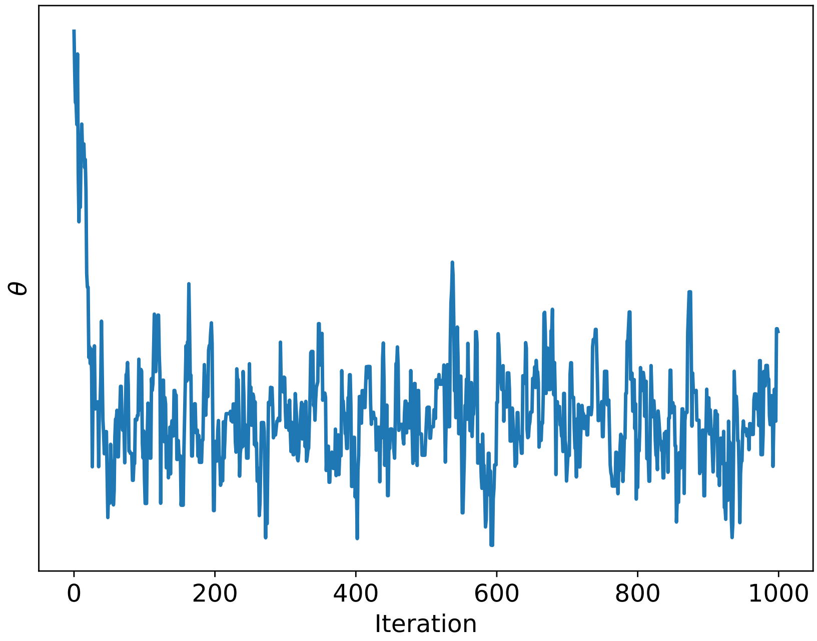

Figure 4.13: Example of trace plot of a Markov chain for parameter θ. The chain reaches the stationary state after about 50 iterations.

4.9.1 Requirements of MCMC

In general, the possible values θ of the Markov chain are called the *states* of the Markov chain (MacKay, 2003; Tierney, 1994). For a Markov chain Monte Carlo estimator to generate samples from the target distribution, the Markov chain must satisfy a couple of conditions:

1. First of all, the Markov chain has to leave the target distribution p *invariant* or *stationary*,

   $$p(\theta') = \int T(\theta'|\theta)p(\theta)d\theta \,. \tag{4.45}$$

   This means that if we start from a state θ of p, the next state θ' is also a state of p. In practice, a sufficient but not necessary condition is *detailed balance*, which requires that each transition $\theta \longrightarrow \theta'$ is reversible. More formally, for any pair of states θ and θ' the following relation must hold,

   $$T(\theta'|\theta)p(\theta) = T(\theta|\theta')p(\theta') \,, \tag{4.46}$$

   meaning that the probability of being at state θ and transitioning to state θ' is equal to the probability of being at state θ' and transitioning to state θ.

2. Furthermore, we need to make sure that the *stationary* distribution is unique and that the distribution of states is able to converge to it regardless the starting point θ_1. In other words, we need to make sure that the *stationary* distribution is also the *limiting* distribution. This requires two properties, *irreducibility*, that is the ability to visit any state θ for which $p(\theta) > 0$ in a finite number of steps, and *aperiodicity*, meaning that no states are only accessible at certain regularly spaced times. These two properties combined, when met, render the Markov chain *ergodic*.

4.9.2 Expected behaviour

When all of the aforementioned conditions are obeyed, the Markov chain samples from the target distribution. The behaviour of the Markov chain, in terms of the computed expectation values, passes through four stages that characterise its normal behaviour (Betancourt, 2017b). The first stage, shown in Figure 4.14 consists of the initialisation of the Markov chain. Often we do not know where the typical set resides and thus we set the first state of the Markov chain to some arbitrary point in parameter space. During the second stage, the Markov chain moves towards the typical set as shown in Figure 4.15. At the same time the absolute difference of the estimated value

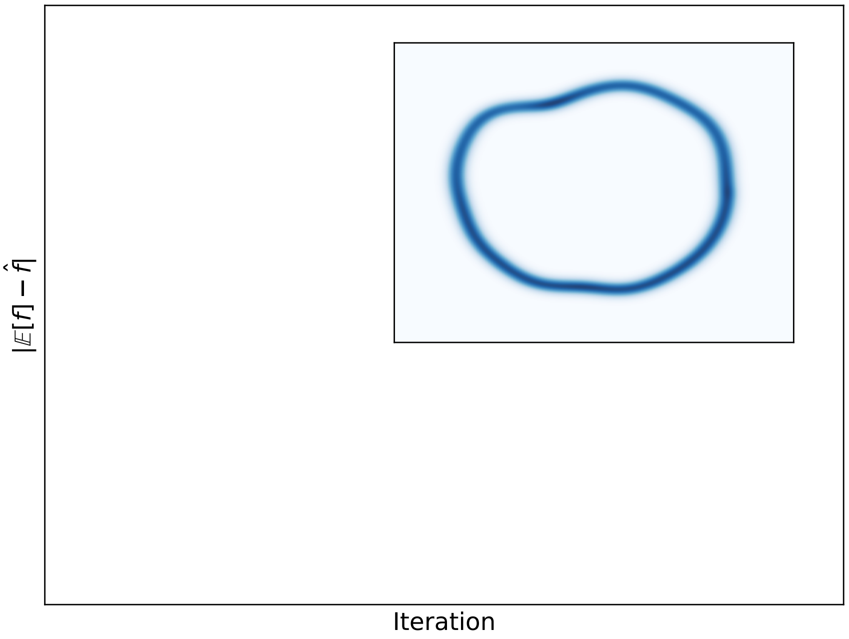

Figure 4.14: Initial stage of exploration – no exploration has taken place.

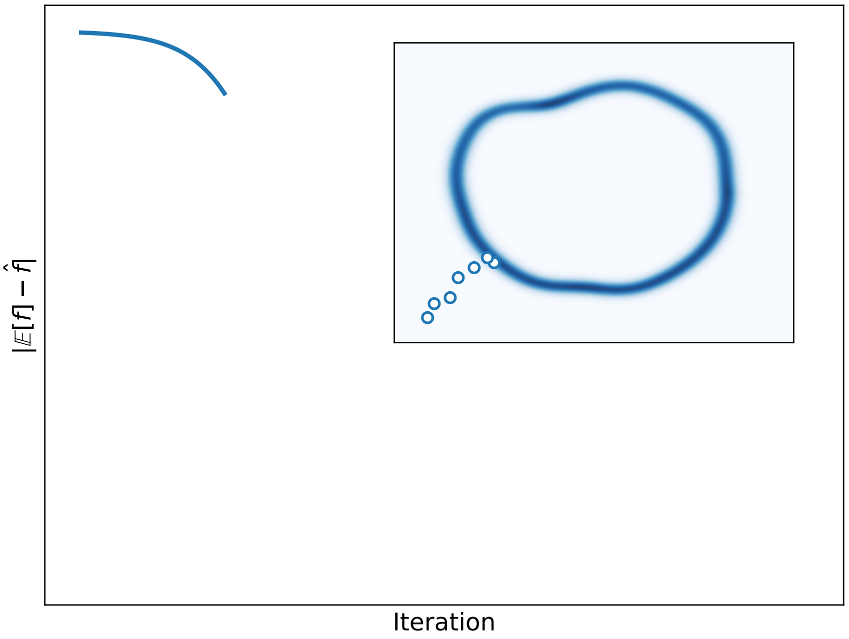

Figure 4.15: Burn–in stage of exploration – the absolute difference between the estimate of f and its expectation value slowly decreases as the chain approaches the typical set.

$\hat{f}$ from the expectation value $\mathbb{E}[f]$ slowly decreases. In the third state shown in Figure 4.16, the Markov chain starts to explore the typical set. The absolute difference between the estimate of f and its expectation value decreases very rapidly. Finally, in the fourth stage shown in Figure 4.17 the Markov chain wanders inside the typical set and the standard error of the estimate asymptotically decreases as prescribed by the central limit theorem.

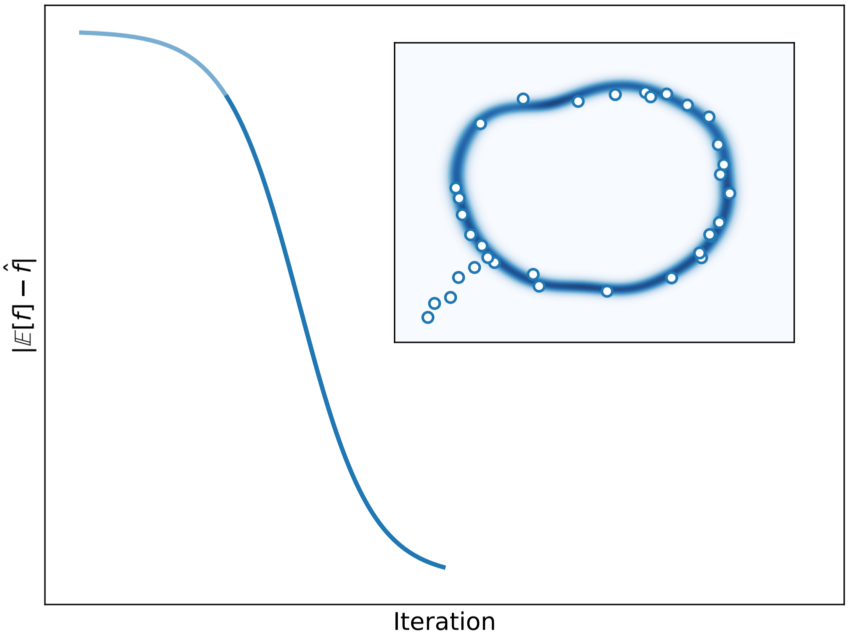

Figure 4.16: Initial convergence phase – the absolute difference between the estimate of f and the expectation value decreases rapidly as the chain approaches explores the typical set for the first time.

4.9.3 Central limit theorem of MCMC

Markov chain Monte Carlo estimators are particularly useful for many analyses since they obey a *central limit theorem* that allows us to quantify their precision (Geyer, 1992; Kipnis & Varadhan, 1986; Tierney, 1994). In particular, given a square–integrable real–valued function $f(\theta)$ and a long enough Markov chain, the following is true,

$$\frac{\hat{f}_n^{\mathrm{MCMC}} - \mathbb{E}_p[f(\theta)]}{\mathrm{MCMC\text{–}SE}_n[f]} \sim \mathcal{N}(0,1)\,, \tag{4.47}$$

where $\mathrm{MCMC\text{–}SE}_n[f]$ is the *Markov chain Monte Carlo Standard Error* given by,

$$\mathrm{MCMC\text{–}SE}_n[f] = \sqrt{\frac{\mathrm{Var}_p[f]}{\mathrm{ESS}_n[f]}}\,. \tag{4.48}$$

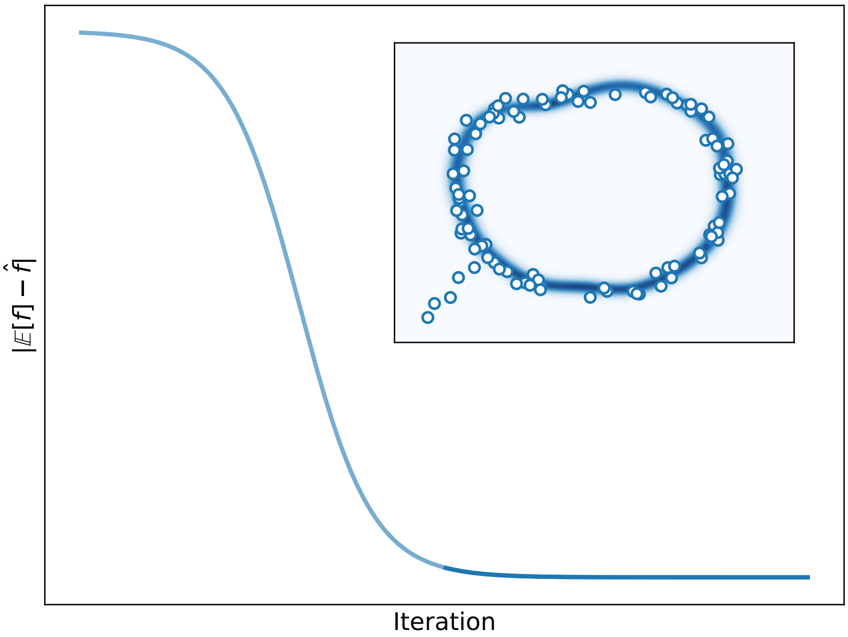

Figure 4.17: Stationary or equilibrium phase – the absolute difference between the estimate of f and the expectation value has reached its minimum value as the chain samples fully populate the typical set.

Comparing MCMC–SE$_n[f]$ with the standard error MC–SE$_n[f]$ of the exact Monte Carlo estimator given by equation 4.31, one immediately notices that the number of samples n has been replaced by the term ESS$_n[f]$. This term, called the *Effective Sample Size* accounts for the loss of information due to the correlation between samples due to the Markov property of the chain. The effective sample size is given by,

$$\mathrm{ESS}_n[f] = \frac{n}{\tau[f]}, \tag{4.49}$$

where $\tau[f]$ is the *relaxation* or *autocorrelation time* of the Markov chain. Less formally, $\tau[f]$ describes the number of steps required for the Markov chain to “forget” where it started, meaning that only one out of $\tau[f]$ is actually independent. A method for estimating the autocorrelation time of a Markov chain will be discussed in the next subsection.

4.9.4 Autocorrelation

The autocorrelation of the Markov chains is a necessary evil of MCMC methods and must be properly understood before making any inference (Geyer, 1992). Figure 4.18 shows two Markov chains with different levels of autocorrelation. In the weakly correlated chain, large jumps take place from one iteration to the next. On the other hand, the strongly correlated chain is characterised by very short jumps and more rigid trajectories.

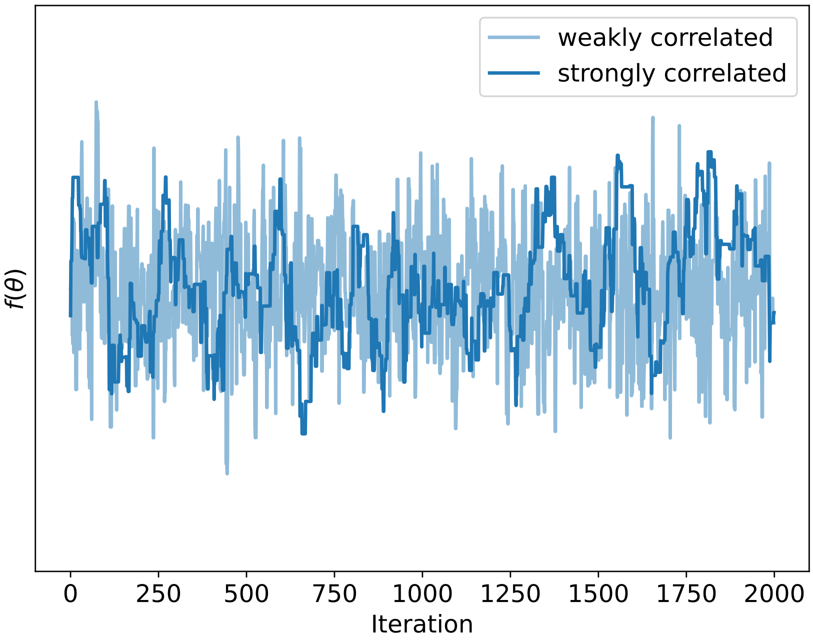

Figure 4.18: Markov chains with different degrees of autocorrelation.

In order to quantify and measure the degree of autocorrelation of a Markov chain we need to compare the states of the chain after fixed number of iterations called *lags*. Given an arbitrary function $f(\theta)$ of the states,

$$\mu_f = \mathbb{E}_p[f] \,, \tag{4.50}$$

is the mean value of the function expressed as the expectation value over the stationary distribution $p(\theta)$. The value $f(\theta_i) - \mu_f$ then quantifies the deviation of the i–th state of the chain from the mean value μ_f. The expectation value of the product of two such deviations defines the *autocovariance* of the chain,

$$c_{ij} = \mathbb{E}_p \left[(f_i - \mu_f)(f_j - \mu_f) \right] \,, \tag{4.51}$$

where $f_i = f(\theta_i)$ and $f_j = f(\theta_j)$. Once the Markov chain has reached the stationary phase, the autocovariance will no longer depend on the particular states, θ_i and θ_j, that we are comparing but on the number of iterations, called lag $\ell = j - i$, between them,

$$c_{ij} = c_{i,i+\ell} = c_\ell \,. \tag{4.52}$$

Finally, if we normalise the autocovariance by the variance,

$$\mathrm{Var}_p[f] = \mathbb{E}_p \left[(f - \mu_f)^2 \right] \,, \tag{4.53}$$

we get the lag$-\ell$ *autocorrelation function*,

$$\rho_\ell[f] = \frac{\mathbb{E}_p\left[(f_{i+\ell}-\mu_f)(f_i-\mu_f)\right]}{\mathrm{Var}_p[f]}\,. \tag{4.54}$$

The normalisation ensures that the maximum possible value of $\rho_\ell[f]$ is $+1$ for fully correlated states and -1 for the completely anti–correlated states. The value of 0 corresponds to uncorrelated samples. The lag$-\ell$ autocorrelation is always unity as any state is perfectly correlated with itself. Furthermore, the autocorrelation function depends only on the absolute lag and is invariant under changes of the sign, that is, $\rho_\ell[f] = \rho_{-\ell}[f]$. For this reason, only the non–negative part of the autocorrelation function is often plotted.

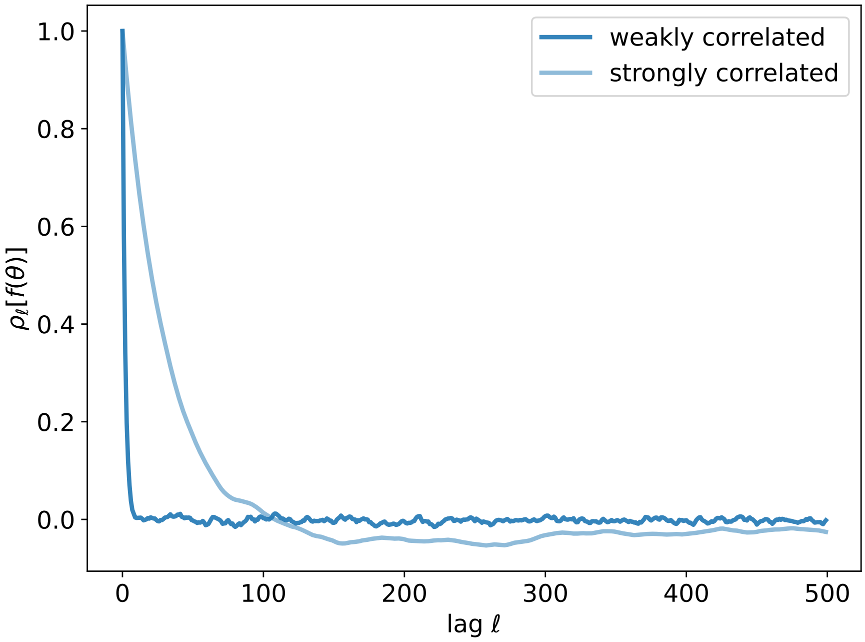

Figure 4.19: Autocorrelation as a function of lag ℓ for a weakly and a strongly correlated chain.

Figure 4.19 shows the autocorrelation function for the weakly and strongly correlated chains of Figure 4.18. We notice that although both functions begin at the value of 1 for lag $\ell = 0$, they approach the value of 0 at different rates. In particular, the autocorrelation function of the weakly correlated chain goes to 0 after only a few lags, whereas the one corresponding to the strongly correlated chain takes much longer.

The asymptotic variance of an infinitely long chain is defined as,

$$\lim_{n\longrightarrow\infty} n\times\mathrm{Var}_p\left[\hat{f}_n^{\mathrm{MCMC}}\right] = \mathrm{Var}_p[f]\times\tau_I[f]\,, \tag{4.55}$$

where,

$$\tau_I[f] = \sum_{\ell=-\infty}^{\ell=+\infty}\rho_\ell[f] = 1 + 2\times\sum_{\ell=1}^{\ell=+\infty}\rho_\ell[f]\,, \tag{4.56}$$

is the *integrated autocorrelation time* and the last equality hold due to the lag–sign invariance of the autocorrelation function. Equation 4.55 implies that the standard error of MCMC is,

$$\text{MCMC–SE}_n[f] = \sqrt{\frac{\text{Var}_p[f]}{\text{ESS}_n[f]}}\,, \tag{4.57}$$

where we have defined the *effective sample size* as,

$$\text{ESS}_n[f] = \frac{n}{1 + 2 \times \sum_{\ell=1}^{\ell=+\infty} \rho_\ell[f]}\,. \tag{4.58}$$

Estimating the integrated autocorrelation time, and thus the effective sample size, in not trivial in practice. The autocorrelation function can be very noisy in large lags, as shown in Figure 4.19. This means that the sum in equation 4.56 needs to be truncated in practice in order avoid adding noise.

5 SIMPLE MCMC METHODS

Not all those who wander are lost.

— J.R.R. Tolkien

During the first half of the twentieth century, research efforts were focused on the task of understanding the equilibrium behaviour of thermodynamic systems. Furthermore, it was well understood that this behaviour was described by specific probability distributions (e.g. *canonical* distribution for a system in constant temperature). The physicists of that time showed great interest in methods that produced *exact samples* from such probability distributions. Enrico Fermi, for instance, would exploit such methods to make amazingly–quick predictions of experimental outcomes as early as 1930s (Metropolis, 1987). During the next two decades, *Stan Ulam* and *John von Neumann* developed various such algorithms which, collectively, were anointed with the name "Monte Carlo" after the infamous casino.

After the war, *Nicholas Metropolis* lead the group in *Los Alamos* in applying Monte Carlo methods to increasingly complex thermodynamic systems. As *exact sampling* was possible only for a limited number of distributions, Metropolis, along with *Arianna Rosenbluth, Marshall Rosenbluth, Edward Teller*, and *Augusta Teller* introduced the so–called *Metropolis* algorithm that produced correlated samples from a wider range of probability distributions (Metropolis et al., 1953). Arianna implemented the algorithm on the *MANIAC* computer (Gubernatis, 2005) and thus she is considered the first person in history to implement a MCMC method.

After decades of empirical success in physics and chemistry, the statistician Hastings (Hastings, 1970) generalised the method by realising that by introducing a small modification he could allow for any *proposal* distribution, not just a limited family of symmetric ones. The method is known today as *Metropolis–Hastings*. Despite *Hasting*'s seminal contribution, it was not until 1984 that S. Geman & D. Geman (1984) introduced the *Gibbs sampler* for the task of image reconstruction, for the broader statistical community to realise the potential of MCMC methods for parameter inference (Gelfand & Smith, 1990).

5.1 METROPOLIS–HASTINGS

The key idea of the *Metropolis–Hastings* algorithm is to separate the *Markov transition probability* into two steps, a *proposal* and an *acceptance* step. During the *proposal* step, a new state θ' is generated conditional on the current state θ,

$$\theta' \sim q(\theta'|\theta)\,, \tag{5.1}$$

by sampling from a proposal distribution $q(\theta'|\theta)$. The aim of this step is to produce a new state that is likely, but not necessary, to reside in the typical set of the target distribution. The form of the conditional proposal distribution can be chosen based on the particular target distribution. As we will see shortly, many of the developments in the field of MCMC focus explicitly on the choice of the proposal distribution.

Once the new state θ' is generated, its validity (i.e. whether or not it belongs to the typical set) is assessed in the *acceptance* step. In particular, the new state θ' is accepted with probability,

$$\alpha(\theta', \theta) = \min\left(1, \frac{p(\theta')q(\theta|\theta')}{p(\theta)q(\theta'|\theta)}\right)\,. \tag{5.2}$$

Equation 5.2 is often called the *Metropolis acceptance probability.* If accepted, the new state θ' is added to the Markov chain and the process is repeated with that as the current state (i.e. $\theta \longleftarrow \theta'$). On the other hand, if the state θ' is rejected, the current state θ is added (i.e. repeated) on the chain. This acceptance/rejection procedure based on equation 5.2 is often referred to as the *Metropolis acceptance criterion.*

It is important to mention here that the *Metropolis acceptance criterion* can be evaluated even if we are only able to compute $p(\theta)$ up to a normalisation constant, as any such factor would cancel out in the ratio $p(\theta')/p(\theta)$ that appears in equation 5.2. This is a very important feature of the algorithm and one of the reasons for its widespread success. In practice, it is very difficult to know the exact value for the *model evidence* $\mathcal{Z} = p(d)$ that acts as the normalisation factor for the posterior distribution $p(\theta|d)$ that might be the target distribution.

It is straightforward to show that the *Metropolis–Hastings* algorithm leaves the target distribution $p(\theta)$ *stationary* by first proving that it preserves *detailed balance.* The *Markov transition probability* is simply,

$$T(\theta'|\theta) = q(\theta'|\theta)\alpha(\theta', \theta)\,, \tag{5.3}$$

that is, the probability of proposing the new state θ' given the old state, times the probability of accepting it. Therefore, the *Markov transition probability* for the *Metropolis–Hastings* algorithm preserves detailed balance,

$$\begin{aligned} T(\theta'|\theta)p(\theta) &= q(\theta'|\theta)\alpha(\theta',\theta)p(\theta) \\ &= q(\theta'|\theta)\min\left(1, \frac{p(\theta')q(\theta|\theta')}{p(\theta)q(\theta'|\theta)}\right)p(\theta) \\ &= \min\big(p(\theta)q(\theta'|\theta), p(\theta')q(\theta|\theta')\big) \\ &= q(\theta|\theta')\alpha(\theta,\theta')p(\theta') \\ &= T(\theta|\theta')p(\theta'), \end{aligned} \tag{5.4}$$

Algorithm 2 Metropolis–Hastings

```
Input: initial state $\theta_1$, (unnormalised) target density $f(\theta) \propto p(\theta)$, proposal
    density $q(\theta'|\theta)$, and number of iterations $N$
Output: Markov chain $\theta_1, \theta_2, \ldots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution
for $t = 1$ to $N$ do
    Draw new state from proposal distribution $\theta' \sim q(\theta'|\theta_t)$
    Compute acceptance probability $\alpha = \min\left(1, \frac{f(\theta')q(\theta_t|\theta')}{f(\theta_t)q(\theta'|\theta_t)}\right)$
    Draw uniform random number $u \sim \mathcal{U}(0,1)$
    if $u < \alpha$ then
        Accept proposed state and set $\theta_{t+1} \longleftarrow \theta'$
    else
        Reject proposed state and set current state as the next $\theta_{t+1} \longleftarrow \theta_t$
    end if
end for
```

5.1.1 Random–walk Metropolis

A very common, and simplifying in practice, choice for the proposal distribution is the conditional normal distribution $q(\theta'|\theta) = \mathcal{N}(\theta'|\theta,\Sigma)$ centred around the current state θ with covariance matrix Σ (Metropolis et al., 1953; Tierney, 1994). The probability density has the usual Gaussian functional form,

$$q(\theta'|\theta) = \det(2\pi\Sigma)^{-\frac{1}{2}}\exp\left[-\frac{1}{2}(\theta'-\theta)^T\Sigma^{-1}(\theta'-\theta)\right]. \tag{5.5}$$

The symmetry of this proposal distribution,

$$q(\theta'|\theta) = q(\theta|\theta'), \tag{5.6}$$

means that the *Metropolis acceptance probability* of equation 5.2 is simplified and the q terms drop out,

$$\alpha(\theta',\theta) = \min\left(1, \frac{p(\theta')}{p(\theta)}\right). \tag{5.7}$$

5.1.2 Independence Metropolis

Another simple choice of proposal distribution is to make it independent of the current state θ. For instance, one can choose a normal distribution $q(\theta) = \mathcal{N}(\theta|\mu, \Sigma)$ with mean μ and covariance matrix Σ, both of which must be known *a priori* and can not depend on the current state. In this case, the *Metropolis acceptance probability* reduces to,

$$\alpha(\theta', \theta) = \min\left(1, \frac{p(\theta')q(\theta)}{p(\theta)q(\theta')}\right) . \tag{5.8}$$

One of the benefits of *Independence Metropolis* (Hastings, 1970; Tierney, 1994), as this approach is called, is that any states produced are independent samples from the target distribution $p(\theta)$. However, it suffers from similar problems to *Importance sampling*. Instead of vanishingly small importance weights, in *Independence Metropolis* we might experience vanishingly small *acceptance probability* when the overlap of the typical set of the proposal distribution q with the target p is small. For this reason, the use of *Independence Metropolis* is wise only when we have good reasons to believe that the proposal distribution is sufficiently close to the target distribution or the dimensionality is low.

5.1.3 Metropolis–adjusted Langevin algorithm

As we have seen, the normal proposal distribution of *Random–walk Metropolis* can utilise only *global* information about the target distribution (i.e. the covariance matrix) in order to achieve efficient sampling. Although sufficient in low to moderate dimensional problems, this strategy can become inefficient as the number of parameters of the target distribution increases. In practice, *Random–walk Metropolis* proposes new states indiscriminately along directions of great covariance without taking into account the local structure of the typical set. This results in either low acceptance probabilities or small proposed steps being accepted as the typical set become thinner in high dimensions.

One way to circumvent this effect and achieve better sampling performance is to capitalise on the knowledge of the gradient of the target distribution in order to “bias” the proposed states towards directions that are more likely to lead to higher acceptance probability. *Metropolis–adjusted Langevin algorithm (MALA)* (Roberts & Stramer, 2002) achieves this by using a conditional normal distribution,

$$q(\theta'|\theta) = \mathcal{N}\left(\theta + \tau\Sigma\nabla\log p(\theta), 2\tau\Sigma\right) , \tag{5.9}$$

where its mean $\theta + \tau\Sigma\nabla\log p(\theta)$ is shifted from the current state θ along the direction of the gradient of the logarithm of the target distribution $\nabla\log p(\theta)$.

If known, Σ can be an approximate covariance matrix that characterises the target distribution, otherwise, a unit–diagonal matrix can be used. τ is the step size of the method and determines the amount of shift of the proposal distribution. In the limit that $\tau \longrightarrow 0$, MALA reduces to RWM. The step size τ can be modified in order to achieve the theoretically optimal acceptance probability of 0.574. Despite the fact that the aforementioned acceptance rate has only been proven to be optimal for certain types of target distributions (Roberts & Rosenthal, 1998), we expect that values in the range between 0.4 and 0.8 would result in a high performance for most applications.

In terms of the typical set, we can think of the gradient of the log probability as a guide that allows for better–informed proposals that are more likely to belong to the typical shell.

5.1.4 Adaptive Metropolis

Hyperparameters of the proposal distribution, such as the covariance matrix Σ of RWM or the step size τ of MALA, do not have to be chosen *a priori* or based on preliminary MCMC runs but they can instead be adaptively tuned during the run. Haario et al. (2001) presented a prototype adaptive version of RWM in which the proposal distribution is continuously adapted during the run using all of the collected samples in order to estimate its covariance matrix. The estimation of the covariance matrix is efficient as only incremental updates are required using simple recursive formulas.

In order to achieve this kind of proposal adaptation in practice we need to abandon the Markov property of the chain. In general, this is not a problem as there is nothing special about the Markov property apart from its simplicity. However, continuous tuning of the proposal distribution during the run requires the adaptation to be *diminishing*, with very specific characteristics, in order to preserve the *ergodicity* of the method (Brooks et al., 2011).

One of the most commonly used algorithms for *diminishing adaptation* is the *stochastic approximation* algorithm of Robbins & Monro (1951). Suppose that we have a function $f(\lambda_i) = f_i$, which encodes some aspect of the behaviour of the i–th state of chain (e.g. the acceptance probability) as a function of some tunable property λ (e.g. the proposal scale), that has expectation value,

$$\mathbb{E}[f(\lambda)] = \frac{1}{n}\sum_{i=1}^{n} f(\lambda_i)\,. \tag{5.10}$$

The solution to the equation $\mathbb{E}[f(\lambda)] = f^{*}$ can be found iteratively, using the recursive formula,

$$\lambda_{i+1} = \lambda_i - \gamma_i\,(f_i - f^{*})\ , \tag{5.11}$$

assuming that f is a non–decreasing function of λ that is uniformly bounded (Andrieu & Thoms, 2008). The parameter γ_i determines the *learning rate* or the rate of convergence of the approximation and has to obey two conditions,

$$\sum_{i=1}^{n} \gamma_i = \infty \,, \qquad \sum_{i=1}^{n} \gamma_i^2 < \infty \,. \tag{5.12}$$

The former condition ensures that any point θ can eventually be reached, and the latter condition ensures that the fluctuations introduced by new iterations is contained and does not prevent convergence. A commonly used schedule for the learning rate that satisfies the above conditions has the form $\gamma_i = i^{-\kappa}$ for $\kappa \in (0.5, 1]$.

Algorithm 3 Adaptive Metropolis

```
Input: initial state $\theta_1$, (unnormalised) target density $f(\theta) \propto p(\theta)$, the target
    acceptance rate $\alpha^*$, learning rate schedule (e.g. $g_t = 1/t$), and number of
    iterations $N$
Output: Markov chain $\theta_1, \theta_2, \ldots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution
Initialise $\mu_1 = 0, \Sigma_1 = 1, \log \lambda_1 = 0$
for $t = 1$ to $N$ do
    Draw new state from proposal distribution $\theta' \sim \mathcal{N}(\theta' | \theta_t, \lambda_t \Sigma_t)$
    Compute acceptance probability $\alpha_t = \min\left(1, f(\theta')/f(\theta_t)\right)$
    Draw uniform random number $u \sim \mathcal{U}(0, 1)$
    if $u < \alpha_t$ then
        Accept proposed state and set $\theta_{t+1} \longleftarrow \theta'$
    else
        Reject proposed state and set current state as the next $\theta_{t+1} \longleftarrow \theta_t$
    end if
    Update mean estimate $\mu_{t+1} \longleftarrow \mu_t - \gamma_t(\mu_t - \theta_{t+1})$
    Update covariance estimate
        $\Sigma_{t+1} \longleftarrow \Sigma_t - \gamma_t \left[\Sigma_t - (\mu_t - \theta_{t+1})(\mu_t - \theta_{t+1})^T\right]$
    Update proposal scale estimate $\log \lambda_{t+1} \longleftarrow \log \lambda_t - \gamma_t(\alpha_t - \alpha^*)$
end for
```

Let us now go through an example of developing an *adaptive* version of the commonly used RWM, in which we tune the covariance matrix Σ of the Gaussian proposal distribution, $\theta' \sim \mathcal{N}(\theta' | \theta, \lambda\Sigma)$, using the following *diminishing adaptation* scheme,

$$\begin{aligned} \mu_{i+1} &= \mu_i - \gamma_i \left(\mu_i - \theta_{i+1}\right) \,, \\ \Sigma_{i+1} &= \Sigma_i - \gamma_i \left[\Sigma_i - (\mu_i - \theta_{i+1})(\mu_i - \theta_{i+1})^T\right] \,, \end{aligned} \tag{5.13}$$

where the μ is the mean value used for the estimation of the covariance Σ, and $\gamma_i = 1/i$ is the *learning rate*. At the same time we can also tune the *magnitude* of the proposal scale, λ_i, by attempting to match the acceptance probability α to the theoretically optimal value of $\alpha^* = 0.234$,

$$\log \lambda_{i+1} = \log \lambda_i - \gamma_i (\alpha_i - \alpha^*) \; . \tag{5.14}$$

Understanding equation 5.14 is straightforward, if the observed acceptance rate is greater than the target (i.e. $\alpha_i < \alpha^*$) then the logarithm of the magnitude of the proposal scale $\log \lambda$ is reduced and *vice versa*. The *Adaptive Metropolis* method presented in this paragraph constitutes a generalisation of the method presented by Haario et al. (2001). With the inclusion of the adaptation of the proposal scale using equation 5.14 the algorithm resembles that of Andrieu & Thoms (2008).

5.2 GIBBS SAMPLING

Another very popular Markov chain Monte Carlo method, to which we partly owe the widespread use of Bayesian inference today, is *Gibbs sampling*. Initially known as the *heat bath* algorithm in the *statistical physics* literature, the *Gibbs sampler* enjoyed great success in the *statistical community* following the seminal paper by S. Geman & D. Geman (1984) that demonstrated its benefits for analysing *Gibbs* distributions on lattices in the context of image processing.

5.2.1 Gibbs sampler

Gibbs sampler attempts to overcome the *curse of dimensionality* using *conditioning* (Casella & George, 1992). In particular, assuming that exact sampling from the conditional distributions of the target distribution $p(\theta)$ is possible, we can generate samples from the target distribution by sequentially sampling from its full set of conditionals instead. Given an initial state $\theta = (\theta_1, \dots, \theta_D)$, the next state in the Markov chain can be generated as,

$$\begin{aligned}
\theta_1' &\sim p(\theta_1 | \theta_2, \dots, \theta_D) \\
&\vdots \\
\theta_k' &\sim p(\theta_k' | \theta_1, \dots, \theta_{k-1}', \theta_{k+1}, \dots \theta_D) \\
&\vdots \\
\theta_D' &\sim p(\theta_D' | \theta_1, \dots, \dots \theta_{D-1}') \; .
\end{aligned} \tag{5.15}$$

The current state θ is then replaced by the new state $\theta' = (\theta_1', \dots, \theta_D')$ and the process is repeated until enough states are collected in the Markov chain. The order of the state updates of equation 5.15 can be either fixed (with a possible reversal after every iteration), as shown above, or randomised to ensure detailed balance.

Algorithm 4 Gibbs sampler

```
Input: initial state $\theta^{(1)} = (\theta_1^{(0)}, \theta_2^{(0)}, \dots, \theta_D^{(0)})$, all the conditional distributions $p(\theta_k|\theta_1, \dots, \theta_{k-1}, \theta_{k+1}, \dots, \theta_D)$ of target $p(\theta)$, and number of iterations $N$
Output: Markov chain $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(N)}$ that has $p(\theta)$ as its equilibrium distribution
for $t = 1$ to $N$ do
    for $k = 1$ to $D$ do
        Draw from conditional $\theta'_k \sim p(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})$
    end for
    Set $\theta^{(t+1)} \longleftarrow \theta' = (\theta'_1, \dots, \theta'_D)$
end for
```

5.2.2 Metropolis–within–Gibbs sampler

The *Gibbs sampler* relies on our ability to produce samples from each one of the conditional distributions. This however is not always feasible as some of the components of the full conditional set might not admit an exact sampling solution. Instead of abandoning *Gibbs sampler* altogether, Müller (1991, 1992) suggested the use of a compromise between the *Gibbs sampler* and the *Metropolis–Hastings* algorithm.

The key idea behind the *Metropolis–within–Gibbs* sampler is to use *Gibbs sampling* for as many of the conditional distributions as possible in order to produce exact samples, and rely on correlated samples generated using *Metropolis–Hastings* for any conditional distributions that *exact sampling* is not possible.

Suppose that we have a *partial* state $(\theta'_1, \dots, \theta'_{k-1}, \theta_k, \dots, \theta_D)$ and we have difficulty generating exact samples from the conditional distribution $\theta'_k \sim p(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)$. We can then treat $p(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)$ as the target distribution for a *Metropolis–Hastings* estimator as follows, in order to proceed with the computation,

1. First, we have to propose a new sample $\theta^*_k \sim q(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)$ from an arbitrary proposal distribution,

2. Then, compute the *Metropolis acceptance probability*

$$\begin{aligned} \alpha(\theta^*_k, \theta_k) = \min \Bigg(1, & \frac{p(\theta^*_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)}{p(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)} \\ & \times \frac{q(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)}{q(\theta^*_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}, \dots, \theta_D)} \Bigg), \end{aligned} \tag{5.16}$$

3. Finally, accept the new state $\theta'_k \longleftarrow \theta^*_k$ with probability $\alpha(\theta^*_k, \theta_k)$, otherwise reject and keep the previous state $\theta'_k \longleftarrow \theta_k$.

Using the algorithm presented above we can replace *exact sampling* from conditional distributions where it is not feasible with *Metropolis–Hastings* estimates.

Algorithm 5 Metropolis–within–Gibbs sampler

```
Input: initial state $\theta^{(1)} = (\theta_1^{(0)}, \theta_2^{(0)}, \dots, \theta_D^{(0)})$, proposal distributions
    $q(\theta_k|\theta_1, \dots, \theta_{k-1}, \theta_{k+1}, \dots, \theta_D)$, (unnormalised) target distribution $f(\theta) \propto$
    $p(\theta)$, the conditional distributions $p(\theta_\ell|\theta_1, \dots, \theta_{\ell-1}, \theta_{\ell+1}, \dots, \theta_D)$ of target
    $p(\theta)$ where $\ell \in L$ and $L$ the set of indices for which exact sampling of the
    conditional is possible, and number of iterations $N$
Output: Markov chain $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(N)}$ that has $p(\theta)$ as its equilibrium distribution
for $t = 1$ to $N$ do
    for $k = 1$ to $D$ do
        if $k \in L$ then
            Draw from conditional $\theta'_k \sim p(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})$
        else
            Draw from proposal $\theta^*_k \sim q(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})$
            Compute acceptance probability
                $$\alpha = \min\left(1, \frac{p(\theta^*_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})}{p(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})} \times \frac{q(\theta_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})}{q(\theta^*_k|\theta'_1, \dots, \theta'_{k-1}, \theta_{k+1}^{(t)}, \dots, \theta_D^{(t)})}\right)$$
            Draw uniform number $u \sim \mathcal{U}(0, 1)$
            if $u < \alpha$ then
                accept new partial state and set $\theta'_k \longleftarrow \theta^*_k$
            else
                reject new partial state and set $\theta'_k \longleftarrow \theta_k^{(t)}$
            end if
        end if
    end for
    Set $\theta^{(t+1)} \longleftarrow \theta' = (\theta'_1, \dots, \theta'_D)$
end for
```

6 AUXILIARY VARIABLE MCMC METHODS

> *Natura non facit saltus.*
>
> — *Gottfried Leibniz*

Auxiliary variable MCMC methods rely on the introduction of one or more additional variables in order to make sampling from the target distribution more efficient.

6.1 SIMULATED ANNEALING

In metallurgy, *annealing* refers to the thermal process used to harden steel. Initially, the metal is heated to a high temperature and then it is cooled down slowly enough for the atoms to *self–arrange* in an ordered pattern that corresponds to the minimum energy (Cahn & Haasen, 1996). The slow rate of cooling ensures that the energy of the system will reach its global minimum instead of getting trapped in local minima.

Realising that the *Metropolis–Hastings* method can be used to simulate the process of gradually cooling a solid towards a low–temperature equilibrium state, Kirkpatrick et al. (1983) suggested that we should construct a sequence of *Boltzmann* distributions,

$$p_i(\theta) \propto e^{-\frac{H(\theta)}{T_i}}, \tag{6.1}$$

for a series of temperatures $T_1 > T_2 > \cdots > T_m$ and simulate from each one in succession by performing a number of MCMC steps in each temperature before moving on to the next. The result is a non–homogeneous Markov chain, that is, a Markov chain with time–varying target density. Assuming that T_1 is high enough and $T_m \approx 0$ we can find the global minimum of the energy $E(\theta)$ by simulating the cooling process of a solid.

Simulated annealing can be used for sampling too, not just for optimisation. By stopping the cooling process earlier at $T_m = 1$ we can sample from any target distribution $p(\theta)$, not just *Boltzmann* distributions, simply setting $H(\theta) = -\log p(\theta)$. Furthermore, for applications in which the target distribution is the posterior distribution we can construct the following sequence of densities,

$$p_i(\theta) \propto p(\theta) e^{\frac{\log p(d|\theta)}{T_i}}, \tag{6.2}$$

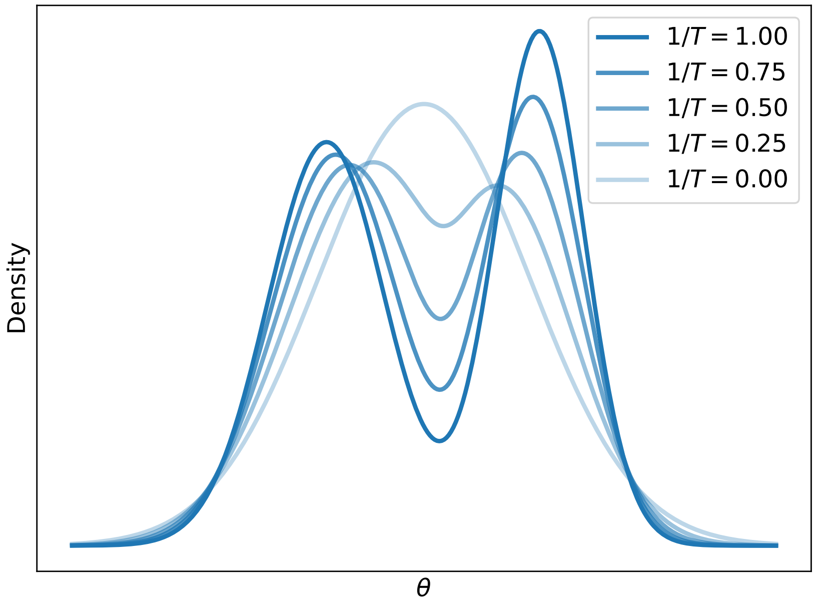

Figure 6.1: Illustration of the gradual annealing performed in the posterior distribution. The prior distribution corresponds to $1/T \longrightarrow 0$ and the posterior is recovered as $1/T \longrightarrow 1$.

where $T_1 > T_2 > \cdots > T_m = 1$. In this case, for $T_1 >> 1$ we effectively sample from the prior distribution,

$$p_i(\theta) \propto p(\theta)\,, \tag{6.3}$$

whereas in the limit that $T_m = 1$ we acquire samples from the posterior,

$$p_i(\theta) \propto p(\theta)p(d|\theta)\,. \tag{6.4}$$

The number of MCMC steps to perform in each temperature, before moving on to the next one, is arbitrary and different mixing criteria can be utilised (e.g. Gelman–Rubin, autocorrelation thresholds, etc.). The benefit of using simulated annealing for sampling is that by simulating multiple Markov chains, possibly in parallel, through this sequence of densities, the risk of the chains getting trapped in isolated modes of the posterior distribution is minimised. This means that this approach can be used when the probability distribution is strongly multimodal. Furthermore, if the number of temperature levels is large enough and the spacing between them small enough, then the Markov chain is approximately always in equilibrium, meaning that no, or minor, burn–in is required to be discarded.

Algorithm 6 Simulated annealing

```
Input: initial state $\theta_1^{(1)}$, temperature schedule $T_1 > T_2 > \cdots > T_m = 1$, prior
    density $\pi(\theta) \equiv p(\theta|\mathcal{M})$, likelihood function $\mathcal{L}(\theta) \equiv p(d|\theta,\mathcal{M})$, and num-
    ber of MCMC iterations $N$ per temperature
Output: Multiple Markov chains $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(m)}$ with the last one having
    $p(\theta)$ as its equilibrium distribution
for $i = 1$ to $m$ do
    Set annealed density $p_i(\theta) \propto \pi(\theta)\mathcal{L}(\theta)^{1/T_i}$
    Generate Markov chain $\theta_1^{(i)}, \dots, \theta_N^{(i)}$ targeting $p_i(\theta)$ (e.g. using
    Metropolis–Hastings)
    Set last state as the first state for the next annealed density $\theta_1^{(i+1)} \longleftarrow$
    $\theta_N^{(i)}$
end for
```

6.2 SLICE SAMPLING

Slice sampling is another MCMC method that relies on an auxiliary variable in order to make sampling easier (Besag & Green, 1993; Neal, 1997, 2003). The method is based on the realisation that sampling from the target distribution with density $p(\theta)$ is equivalent to uniform sampling from the area or volume below the curve or surface of $f(\theta) \propto p(\theta)$. This is equivalent to the introduction of an auxiliary variable ϕ, called *height*, such that the joint distribution $p(\theta, \phi)$ is uniform over the region,

$$U = \{(\theta, \phi) \;:\; 0 < \phi < f(\theta)\} \,. \tag{6.5}$$

In other words, the joint distribution can be written as,

$$p(\theta, \phi) = \begin{cases} 1/\mathcal{Z} & \text{if } 0 < \phi < f(\theta)\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{6.6}$$

where

$$\mathcal{Z} = \int f(\theta) d\theta \,. \tag{6.7}$$

To sample from the target distribution $p(\theta)$ we first sample uniformly from $p(\theta, \phi)$ and then marginalise over ϕ by dropping the ϕ–value of each sample and keeping the θ–value. The proof that this results in the marginal density for θ is straightforward,

$$p(\theta) = \int p(\theta, \phi) d\phi = \int_0^{f(\theta)} \frac{1}{\mathcal{Z}} d\phi = \frac{f(\theta)}{\mathcal{Z}} \,. \tag{6.8}$$

Generating independent samples from the uniform joint density $p(\theta, \phi)$ is rarely possible in practice. Instead, one might prefer to construct a Markov chain that leaves the distribution $p(\theta, \phi)$ invariant. One such option is to use

Gibbs sampling, that is, to sample alternately from the conditional distribution $p(\phi|\theta)$, which is uniform over the interval $(0, f(\theta))$, and then from the conditional distribution $p(\theta|\phi)$, which is uniform over the region,

$$S = \{\theta \ : \ \phi < f(\theta)\} \ , \tag{6.9}$$

called the *slice*. Applying this procedure repeatedly will produce a Markov chain that has the joint distribution $p(\theta, \phi)$ as its stationary distribution.

Sampling uniformly from the aforementioned slice is not trivial either. However, the fact that the conditional density $p(\theta|\phi)$ is uniform allows us to construct procedures to sample from it which would otherwise would not have worked. Neal (2003) proposed the following sequence of steps for univariate probability distributions,

1. Uniformly sample a real value ϕ in the interval $(0, f(\theta_0))$, therefore defining the horizontal slice $S = \{\theta \ : \ \phi < f(\theta_0)\}$ that always includes θ_0,

2. Find an interval $I = (L, R)$ around θ_0 along the slice that contains all, or much of, the slice,

3. Sample a new value θ_1 from the part of the slice within the interval, that is, from $I \cap S$.

It is important to mention that as we often work with $g(\theta) = \log f(\theta)$, to avoid numerical issues, one can use the variable $\psi = \log(\phi) = g(\theta_0) - e$, where e is exponentially distributed with mean one, to define the slice as $S = \{\theta \ : \ \psi < g(\theta_0)\}$.

The first step in the above procedure is trivial, yet steps two and three require more serious consideration. Neal (2003) suggested to use the so–called *stepping–out* and *shrinking* procedures for those steps respectively. *Stepping–out* works by uniformly positioning an interval of width w around θ_0 such that it includes θ_0, and then expanding the interval in steps of size w until both its ends (L, R) are outside the slice S. This effectively constructs the interval $I = (L, R)$. It is worth noting that the algorithm is valid even if only a pre–specified number of expansions take place and the interval ends up not covering the entirety of the slice S. The *shrinking* procedure that follows functions by uniformly sampling points in the interval $I = (L, R)$ until one of them lies in the slice S. Every time a point is rejected, being outside of the slice, the interval I shrinks such that the rejected point now defines one of its two boundaries, determined by whether the rejected point lies left or right of θ_0.

The fact that the three–step procedure presented so far describes a slice sampling update from a univariate probability distribution $p(\theta)$ does not prohibit its use in multivariate cases. In particular, there are many ways the aforementioned recipe can be generalised and used in target distribution

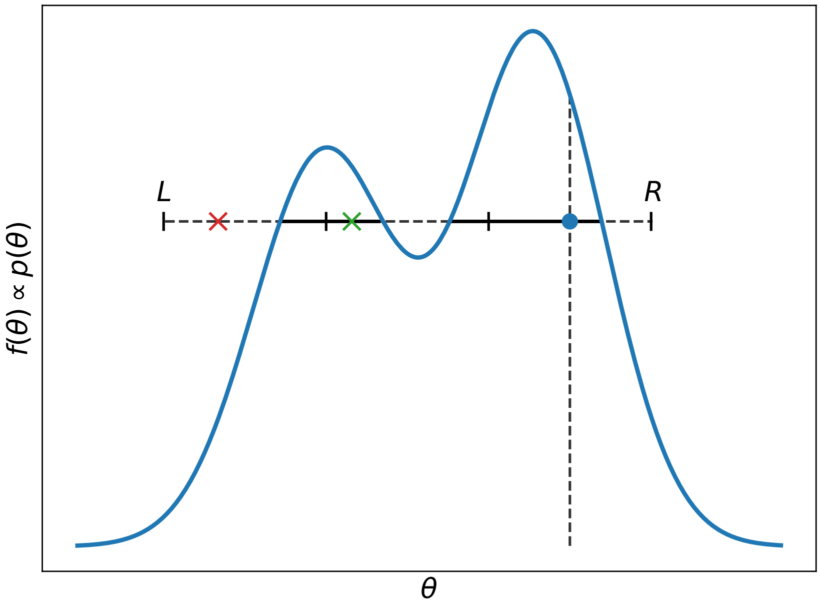

Figure 6.2: Illustration of the stepping–out and shrinking procedures used in slice sampling. Given an initial state θ in the Markov chain, an auxiliary variable ϕ is sampled corresponding to the height thus defining the extended state (θ, ϕ) shown here as a *blue* point. An interval of a certain width is placed uniformly around the current point (θ, ϕ) and expanded in steps of size equal to the initial width until both of its ends, L and R, are outside the graph. A new state, shown in *red*, is then proposed uniformly along the interval (L, R). Since the proposed state lies above the graph of $f(\theta)$ (i.e. not in the slice shown as a continuous line) it is rejected. A new state, shown in *green*, is the proposed uniformly between the rejected state and R. Since the proposed state is below the graph, and thus in the slice, it is accepted and added to the Markov chain. The whole process is then repeated.

with more than one parameter. Perhaps the simplest one is to apply this univariate scheme along each coordinate axis in turn, updating one parameter at a time. This corresponds to a *Metropolis–within–Gibbs* scheme. Another option is to apply 1–D updates in random directions. This is more general than the previous one, and there is freedom to choose the distribution of the random directions. The directions can be drawn from a multivariate zero–mean normal distribution with unit–diagonal covariance matrix or a more appropriate non–diagonal covariance matrix that encodes some of the correlations of the parameters of the target distribution. Such a covariance can be configured *a priori*, estimated during a short preliminary run from samples from the target, or adaptively tuned using an appropriate algorithm for *diminishing adaptation*.

One of the great benefits of slice sampling is the fact that it has a single hyper–parameter, the initial width w of the interval I. Furthermore, the value of w is adapted continuously by the *stepping–out* and *shrinking* procedures. This sort of *local adaptation* is absent from many MCMC that assume a global proposal scale. Another characteristic of slice sampling is the lack of rejected samples in the Markov chain. Unlike methods that include a Metropolis acceptance criterion, slice sampling always moves to a new state in every iteration.

Algorithm 7 Slice sampling

Input: initial state θ_1, (unnormalised) target density $f(\theta) \propto p(\theta)$, number of maximum expansions m, and number of iterations N

Output: Markov chain $\theta_1, \dots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution

```
Draw “height” auxiliary variable $\phi \sim \mathcal{U}(0, f(\theta_1))$
for $t = 1$ to $N$ do
    Draw left bound of the $I$ interval $L \sim \mathcal{U}(\theta_t - w, \theta_t)$
    Set right bound $R \longleftarrow L + w$
    Draw uniform variable $u \sim \mathcal{U}(0, 1)$
    Set maximum number of interval expansions to the left $J \longleftarrow \text{Floor}(m \times u)$
    Set maximum number of interval expansions to the right $K \longleftarrow (m - 1) - J$
    while $J > 0$ and $\phi < f(L)$ do
        Expand left boundary $L \longleftarrow L - w$ in steps of $w$
        Reduce count $J \longleftarrow J - 1$ by one
    end while
    while $K > 0$ and $\phi < f(R)$ do
        Expand right boundary $L \longleftarrow R + w$ in steps of $w$
        Reduce count $K \longleftarrow K - 1$ by one
    end while
    repeat
        Draw state within interval $\theta' \sim \mathcal{U}(L, R)$
        if $\theta' < \theta_t$ then
            Contract interval $L \longleftarrow \theta'$
        else
            Contract interval $R \longleftarrow \theta'$
        end if
    until $\phi < f(\theta')$
    Accept new state $\theta_{t+1} \longleftarrow \theta'$
end for
```

6.3 HAMILTONIAN MONTE CARLO

Hamiltonian Monte Carlo (HMC) introduces a *momentum* auxiliary variable and uses the gradient of the target probability density to efficiently explore the typical set. HMC turns the problem of sampling from the target distribution into the approximate simulation of Hamiltonian dynamics with a subsequent *Metropolis* correction step (Neal et al., 2011). In the statistical physics literature HMC was suggested as a method of efficiently simulating states from a physical system (Duane et al., 1987), which was then employed to statistical inference problems (Liu & Liu, 2001; Neal, 1992, 1993, 1996).

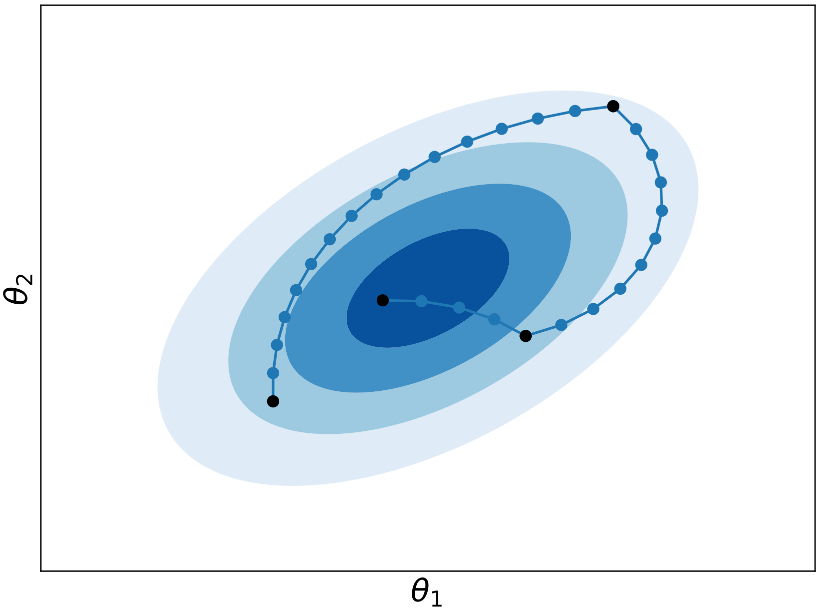

Figure 6.3: Illustration of Hamiltonian trajectories in parameter space. The *black* points correspond to the accepted states.

6.3.1 Auxiliary momentum variable

HMC introduces an auxiliary variable ρ and samples from the joint probability density,

$$p(\rho, \theta) = p(\rho|\theta)p(\theta)\,. \tag{6.10}$$

In most applications of HMC, the momentum variable ρ chosen to be Gaussian–distributed,

$$\rho \sim \mathcal{N}(0, M)\,, \tag{6.11}$$

and its probability density function to be independent of the state variable θ (i.e. $p(\rho|\theta) = p(\theta)$). M is the symmetric, *positive definite mass matrix* that has the role of the *Euclidean metric*, that is to define the relative *length scales* between parameters. In practice, M can be chosen to be Σ^{-1}, meaning the inverse of the sample covariance matrix that characterises the target distribution assuming that it is known or easy to estimate.

6.3.2 The Hamiltonian

HMC treats sampling from the joint distribution $p(\rho, \theta)$ as a problem of solving the *Hamiltonian dynamics* given the *Hamiltonian*,

$$\begin{aligned} \mathcal{H}(\rho, \theta) &= -\log p(\rho, \theta) \\ &= -\log p(\rho|\theta) - \log p(\theta) \\ &= T(\rho|\theta) + V(\theta)\,, \end{aligned} \tag{6.12}$$

where

$$T(\rho|\theta) = -\log p(\rho|\theta)\,, \tag{6.13}$$

is the *kinetic energy*, and,

$$V(\theta) = -\log p(\theta)\,, \tag{6.14}$$

is the *potential energy*.

6.3.3 Hamilton's equations

The dynamics of a system (i.e. its evolution in time) that is characterised by the *Hamiltonian* of equation 6.12 are given by solving *Hamilton's equations*,

$$\begin{aligned} \frac{d\theta}{dt} &= \frac{\partial \mathcal{H}}{\partial \rho} = \frac{\partial T}{\partial \rho}\,, \\ \frac{d\rho}{dt} &= -\frac{\partial \mathcal{H}}{\partial \theta} = -\frac{\partial T}{\partial \theta} - \frac{\partial V}{\partial \theta}\,, \end{aligned} \tag{6.15}$$

or, in the case that the momentum variable ρ is independent of the state variable θ, that is $p(\rho|\theta) = p(\rho)$,

$$\begin{aligned} \frac{d\theta}{dt} &= \frac{\partial \mathcal{H}}{\partial \rho} = \frac{\partial T}{\partial \rho}\,, \\ \frac{d\rho}{dt} &= -\frac{\partial \mathcal{H}}{\partial \theta} = -\frac{\partial V}{\partial \theta}\,. \end{aligned} \tag{6.16}$$

Therefore, given an initial state (ρ, θ), the system's evolution in time is completely determined by equations 6.16.

6.3.4 Leapfrog integration

Solving *Hamilton's equations* analytically is only feasible for very simple systems that correspond to simple target probability distributions. In practice, however, we aim to solve equations 6.16 for systems of arbitrary complexity. To this end, we turn to numerical methods for integrating this system of differential equations.

The most commonly used numerical method is the *leapfrog integration algorithm* (Leimkuhler & Reich, 2004) that begins by sampling a value for the momentum variable ρ according to equation 6.11 and then proceeds by applying L times the following steps,

$$\begin{aligned} \rho &\longleftarrow \rho - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta}\,, \\ \theta &\longleftarrow \theta + \epsilon M^{-1}\rho\,, \\ \rho &\longleftarrow \rho - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta}\,, \end{aligned} \tag{6.17}$$

where ϵ is the *integration step size* that determines the smallest time interval. The length of the trajectory will then be ϵL and the new state of the system is denoted as (ρ', θ'). The numerical error introduced into the calculation by the *leapfrog* algorithm is of the order of ϵ^3 per step and ϵ^2 globally (Leimkuhler & Reich, 2004).

6.3.5 Metropolis acceptance criterion

If the *leapfrog algorithm* were perfect and did not introduce any numerical error, we would not have to do anything more than re–sample the *momentum* variable every L integration steps. However, the *leapfrog integrator* is far from this which means that we need to account for the numerical error that it introduces before it accumulates. To this end, we only accept and add the new state (ρ', θ') into the Markov chain with probability

$$\alpha(\theta', \theta) = \min\left(1, \frac{p(\rho', \theta')}{p(\rho, \theta)}\right)\,, \tag{6.18}$$

and reject it otherwise by adding (ρ, θ) into the chain. Equation 6.18 is simply the *Metropolis acceptance probability* for HMC. Therefore, we see that HMC is essentially a case of *Metropolis–Hastings* with symmetric proposal distribution in the augmented state space of (ρ, θ).

6.3.6 Performance and tuning

The sampling performance of HMC is very sensitive to its tuning (M. D. Hoffman, Gelman, et al., 2014; Neal et al., 2011) and many efforts have been made to develop heuristics and automated tuning procedures for the two hyperparameters, ϵ and L, that the method relies upon. The step size ϵ can be adaptively tuned by trying to match the observed acceptance rate to the theoretically optimal value of 0.65. Tuning the number of steps L is more

Algorithm 8 Hamiltonian Monte Carlo

```
Input: initial state $\theta_1$, potential energy $V(\theta) = -\log p(\theta)$ up to an additive constant, kinetic energy definition $T(\rho) = \rho^T M^{-1} \rho/2$, number of leapfrog steps $L$, integration step size $\epsilon$, and number of iterations $N$
Output: Markov chain $\theta_1, \ldots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution

Draw momentum variable $\rho_t \sim \mathcal{N}(0, M)$
Set proposed state $\theta' \longleftarrow \theta_t$
Set proposed momentum $\rho' \longleftarrow \rho$
for = 1 to $L$ do
    Update momentum $\rho' \longleftarrow \rho' - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta}$
    Update position $\theta' \longleftarrow \theta' + \epsilon M^{-1} \rho'$
    Update momentum $\rho' \longleftarrow \rho' - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta}$
end for
Reverse momentum $\rho' \longleftarrow -\rho'$
Compute acceptance probability
    $\alpha = \min\left(1, \exp\left[V(\theta) - V(\theta') + T(\theta) - T(\theta')\right]\right)$
Draw uniform number $u \sim \mathcal{U}(0, 1)$
if $u < \alpha$ then
    Accept proposed state and set $\theta_{t+1} \longleftarrow \theta'$
else
    Reject proposed state and set $\theta_{t+1} \longleftarrow \theta$
end if
```

cumbersome in practice. In principle, L can be tuned by minimising the autocorrelation time of the Markov chain. In practice this requires running multiple preliminary runs with different values of L in order to determine the most efficient one. For this reason, other approaches, such as *Empirical HMC* (Wu et al., 2018) and the *No U-Turn Sampler (NUTS)* (M. D. Hoffman, Gelman, et al., 2014), have been proposed that automate the use of HMC for many applications.

7 ENSEMBLE MCMC METHODS

As for me, I am tormented with an everlasting itch for things remote.
I love to sail forbidden seas, and land on barbarous coasts.

— *Herman Melville, Moby–Dick or, the Whale*

In order to avoid issues caused by multimodality or the need for tuning the proposal distribution, ensemble MCMC methods rely on an ensemble of parallel samplers, often called *walkers*, that sample from an extended probability distribution. A common way to construct such an extended probability distribution is using the product density,

$$\pi(\{\theta_k\}_{k=1}^{K}) = \prod_{k=1}^{K} p_k(\theta_k)\,, \tag{7.1}$$

where $p_k(\theta_k)$ are the individual densities, one of which can correspond to the target distribution of interest (e.g. the posterior), and K is the number of walkers. It is important to note here that θ_k is not the k–th component of a vector, but a D–dimensional vector itself.

The simplest product density that we can construct based on equation 7.1 is to assume that $p_k(\theta_k) = p(\theta_k)$ for all k, meaning that the product density is just the product of K identical copies of the target distribution $p(\theta_k)$. A natural question to ask is then why would anyone want to do this? Why sample K copies of the same distribution instead of just one? The answer is that the walkers sampling each copy do not have to be independent of each other and instead are allowed to exchange information about their current state. For instance, the proposal distribution for a single walker can depend on the current positions of the rest of the walkers in the ensemble. This allows for effective proposals that take into account the relative length–scales and positions of the modes of the target distribution.

Of course, other product densities, that do not rely on the simplifying assumption that $p_k(\theta_k) = p(\theta_k)$ for all k, can also be defined as we will see in the case of the *parallel tempering* algorithm in Section 7.4. In those cases, the goal is not usually to construct effective proposal distribution but rather to deal with the challenge of multimodality.

7.1 GAUSSIAN ENSEMBLE

Perhaps the simplest way to construct an ensemble MCMC method that limits the requirement for tuning, to some extent, its proposal distribution is the *Gaussian ensemble (GE)* algorithm. GE uses an ensemble of K walkers that target a product density of the form of equation 7.1, where all copies $p_k(\theta_k)$ are identical and correspond to the target distribution of interest (e.g. posterior), and the proposal distribution of each walker is simply a normal distribution informed by the positions of the rest of the walkers in the ensemble (Speagle, 2019).

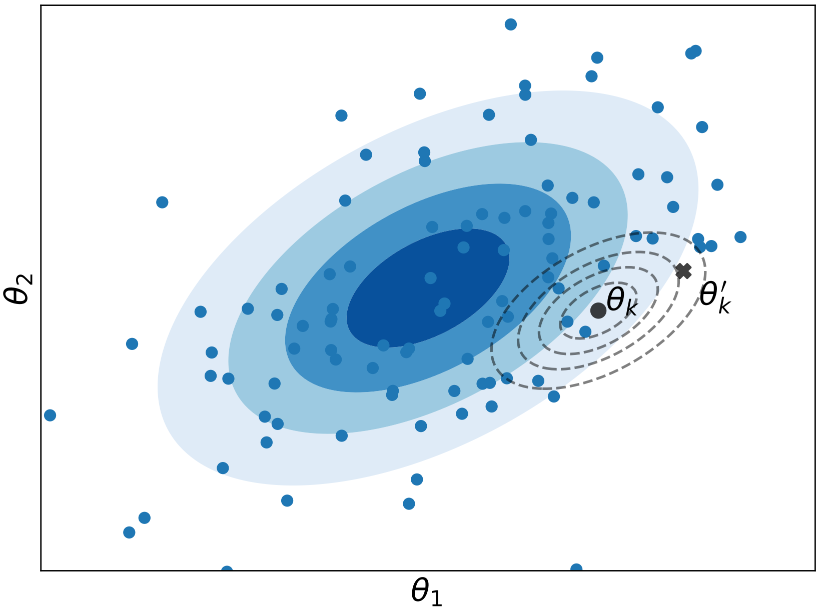

Figure 7.1: Illustration of the Gaussian ensemble MCMC method. A new state θ_k' is proposed in the vicinity of the position θ_k of the walker that is updated using an rescaled version of the sample covariance matrix of the rest of the walkers (i.e. excluding θ_k) for the normal proposal distribution.

In particular, in a given iteration t of the method, the algorithm performs a loop over the K walkers updating each walker in turn. A new position $\tilde{\theta}_k$ is proposed from a normal distribution,

$$\theta_k' \sim \mathcal{N}\left(\theta|\theta_k^{(t-1)}, \gamma\Sigma_{-k}\right) , \tag{7.2}$$

centred on the current state $\theta_k^{(t-1)}$ of the k−th walker and γ is a multiplying factor used to scale the covariance matrix in order to achieve the optimal acceptance rate (e.g. $\gamma = 2.38^2/D$). The covariance matrix Σ_{-k} of the proposal distribution is simply the sample covariance estimated using the positions of the ensemble $\{\theta_1^{(t)}, \dots, \theta_{k-1}^{(t)}, \theta_{k+1}^{(t-1)}, \dots, \theta_K^{(t-1)}\}$ which excludes the k–th walker. It is important to notice also that all the walkers up to and excluding the k–th have already been updated and it is their updated positions that are used to compute the proposal covariance. This is essentially a *Metropolis–within–Gibbs* scheme in disguise. The new point $\tilde{\theta}_k$ is then accepted or rejected based on the usual Metropolis criterion and the process continuous with the next walker until all of them have been updated.

Algorithm 9 Gaussian ensemble

Input: initial state for the ensemble $\theta^{(1)} = (\theta_1^{(1)}, \dots, \theta_K^{(1)})$, (unnormalised) target density $f(\theta) \propto p(\theta)$, covariance scaling factor (e.g. $\gamma = 2.38^2/D$), and number of iterations N

Output: Markov chain $\theta_1, \dots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution

```
for $t = 1$ to $N$ do
    for $k = 1$ to $K$ do
        Compute ensemble mean $\mu_{-k}^{(t)} = \frac{1}{K-1}\sum_{i \neq k} \theta_i^{(t)}$ excluding the $k$th state
        Compute ensemble covariance matrix $\Sigma_{-k}^{(t)} = \frac{1}{K-1}\sum_{i \neq k}(\theta_i^{(t)} - \mu_{-k}^{(t)})(\theta_i^{(t)} - \mu_{-k}^{(t)})^T$ excluding the $k$th state
        Draw proposal $\theta_k' \sim \mathcal{N}(\theta_k^{(t)}, \gamma\Sigma_{-k}^{(t)})$
        Compute acceptance probability $\alpha_k = \min\left(1, f(\theta_k')/f(\theta_k^{(t)})\right)$
        Draw uniform number $u \sim \mathcal{U}(0, 1)$
        if $u < \alpha_k$ then
            Accept proposed state and set $\theta_k^{(t+1)} \longleftarrow \theta_k'$
        else
            Reject proposed state and set $\theta_k^{(t+1)} \longleftarrow \theta_k^{(t)}$
        end if
    end for
end for
```

GE solves the problem of tuning the proposal, up to the scaling factor γ of the covariance matrix, but still assumes a Gaussian proposal. This means that we do not expect that GE will perform better than K parallel well–tuned *Random–walk Metropolis* samplers. As we will discuss in the next couple of sections, there are ways to relax this limitation and allow for more flexible proposals. Last but not least, the estimation of the proposal covariance matrix requires that the absolute minimum size of the ensemble to be $D + 1$ for the covariance to be non–singular.

7.2 AFFINE–INVARIANT STRETCH MOVE

The *affine–invariant ensemble sampler* and in particular the *stretch move* introduced by Goodman & Weare (2010) is perhaps the most popular ensemble MCMC method in the astronomical literature, made available in the *Python* implementation *emcee* (Foreman-Mackey, Hogg, et al., 2013). The *stretch move* algorithm relaxes the limitation of the Gaussian proposal and instead updates each walker in turn along the direction of a different uniformly selected walker sampled from the rest of the ensemble. As we will discuss this change introduces both benefits and challenges.

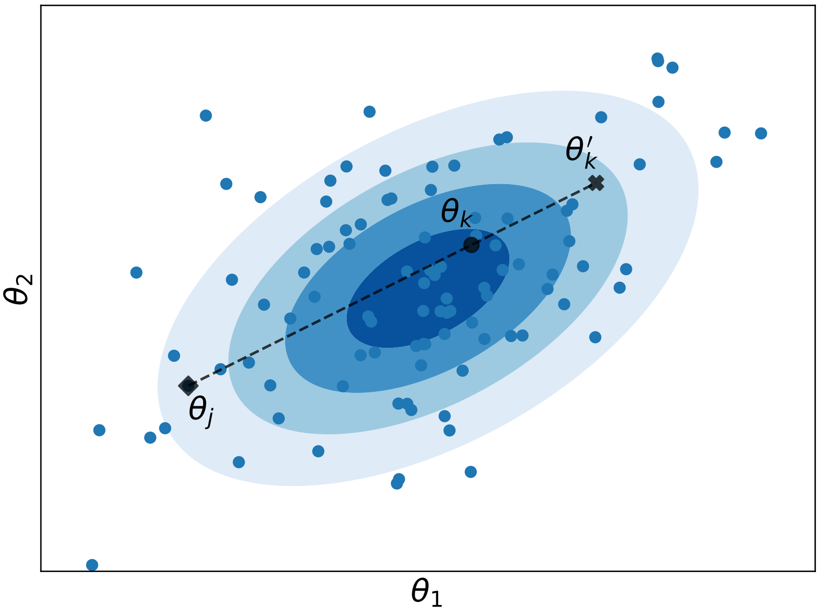

Figure 7.2: Illustration of the affine–invariant stretch move. The selected walker θ_k is moved to its new position θ_k' along the line defined by θ_j and θ_k. θ_j is a walker that is uniformly selected from the rest of the ensemble (i.e. excluding θ_k).

In particular, in a given iteration t of the method, the algorithm performs a loop over the K walkers updating each one in turn. In the so–called *stretch move*, we move a walker θ_k using a uniformly selected walker θ_j from the complementary ensemble $S_{-k} = \{\theta_1^{(t)}, \dots, \theta_{k-1}^{(t)}, \theta_{k+1}^{(t-1)}, \dots, \theta_K^{(t-1)}\}$ that excludes θ_k. The θ_j walker acts as an *anchor point* for the move that consists of a proposal of the form

$$\theta_k' = \zeta\theta_k + (1-\zeta)\theta_j\,, \tag{7.3}$$

where ζ is a scaling variable with a probability density g that satisfies the symmetry condition,

$$g\left(\frac{1}{\zeta}\right) = \zeta g(\zeta)\,, \tag{7.4}$$

such that the move expressed by equation 7.3 is symmetric in the *Metropolis* sense. A particular density that obeys this condition is

$$g(\zeta) \propto \begin{cases} \frac{1}{\sqrt{\zeta}} & \text{if } \zeta \in \left[\frac{1}{\alpha}, \alpha\right]\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{7.5}$$

where $\alpha > 1$ is a parameter that can be tuned to enhance the performance. The default value is usually set to $\alpha = 2$. The new state θ_k' is then accepted with *Metropolis* probability,

$$\alpha(\theta_k', \theta_k) = \min\left(1, \zeta^{D-1}\frac{p(\theta_k')}{p(\theta_k)}\right)\,, \tag{7.6}$$

where the ζ^{D-1} comes from the fact that the update takes place along a straight line. The process is then repeated for the next walker, until all the walkers are updated for the current iteration t before the algorithm moves to its next iteration.

Algorithm 10 Affine–invariant stretch move

Input: initial state for the ensemble $\theta^{(1)} = (\theta_1^{(1)}, \dots, \theta_K^{(1)})$, (unnormalised) target density $f(\theta) \propto p(\theta)$, and number of iterations N

Output: Markov chain $\theta_1, \dots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution

```
for $t = 1$ to $N$ do
    for $k = 1$ to $K$ do
        Draw a walker $\theta_j$ from the complementary ensemble $S_{-k} = \left\{\theta_1^{(t+1)}, \dots, \theta_{k-1}^{(t+1)}, \theta_{k+1}^{(t)}, \dots, \theta_K^{(t)}\right\}$
        Draw random number $\zeta \sim g(\zeta)$
        Compute proposed state $\theta_k' \longleftarrow \zeta\theta_k + (1-\zeta)\theta_j$
        Compute acceptance probability $\alpha_k = \min\left(1, \zeta^{D-1} f(\theta_k')/f(\theta_k^{(t)})\right)$
        Draw uniform number $u \sim \mathcal{U}(0,1)$
        if $u < \alpha_k$ then
            Accept proposed state and set $\theta_k^{(t+1)} \longleftarrow \theta_k'$
        else
            Reject proposed state and set $\theta_k^{(t+1)} \longleftarrow \theta_k^{(t)}$
        end if
    end for
end for
```

One of the strict requirements of this method is the minimum number of walkers to be $D + 1$ for it to be *ergodic* and avoid the risk of walkers getting *trapped* in some hyper–plane of lower than D dimensions. Practically, the actual number of walkers required is much larger as it determines the plethora of possible new directions along which updates take place in each iteration. In this sense, the initial positions of the walkers and the number of them are the only free hyperparameters of this method. A great benefit of this method is that it is *affine–invariant*, that is, its performance is insensitive to any linear correlations between the parameters of the target distribution. As the astronomical community has witnessed during the past few years, this offers a great advantage over other methods.

7.3 DIFFERENTIAL EVOLUTION

Another ensemble method in the spirit of the *stretch move* is the *differential evolution* MCMC (Ter Braak, 2006; Ter Braak & Vrugt, 2008). Unlike the *stretch move* that requires another single walker to act as an *anchor point* for a proposal, *differential evolution* involves two. We will discuss shortly how this difference can affect the performance and alter the characteristics of the method.

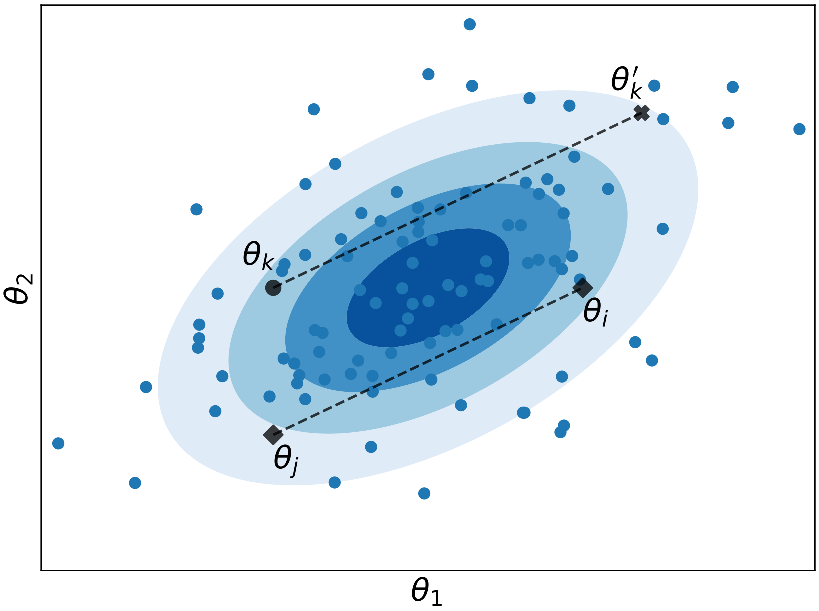

Figure 7.3: Illustration of the differential evolution Monte Carlo. The selected walker θ_k is moved to its new position θ'_k parallel to the line defined by θ_i and θ_j. The latter are two walkers that are uniformly selected from the rest of the ensemble (i.e. excluding θ_k).

An update of the ensemble works as follows: the algorithm performs a loop over the K walkers updating each one in turn. Assuming that the current walker to be updated is θ_k, the algorithm uniformly selects two walkers (without replacement), θ_i and θ_j, from the complementary ensemble $S_{-k} = \{\theta_1^{(t)}, \ldots, \theta_{k-1}^{(t)}, \theta_{k+1}^{(t-1)}, \ldots, \theta_K^{(t-1)}\}$ that excludes θ_k. The vector $\theta_i - \theta_j$ connecting the two auxiliary walkers defines the direction along which a move is proposed. The move consists of a proposal of the form

$$\theta_k' = \theta_k + \gamma \times \left(\theta_i - \theta_j\right) + \epsilon\,, \tag{7.7}$$

where γ is a non–zero scaling factor and $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is some optional Gaussian noise. The value of γ determines the scale of the proposal. Its default value is often set to $\gamma = 2.38/\sqrt{2D}$ which results in the optimal acceptance rate (i.e. 23.4%) for normal target distributions. In practice, one can adapt γ using some diminishing adaptation scheme during the run. The proposed update of equation 7.7 is then accepted with Metropolis acceptance probability

$$\alpha(\theta', \theta) = \min\left(1, \frac{p(\theta')}{p(\theta)}\right)\,. \tag{7.8}$$

Algorithm 11 Differential evolution

Input: initial state for the ensemble $\theta^{(1)} = (\theta_1^{(1)}, \ldots, \theta_K^{(1)})$, (unnormalised) target density $f(\theta) \propto p(\theta)$, proposal scale parameter (e.g. $\gamma = 2.38/\sqrt{2D}$), optional Gaussian noise standard deviation (e.g. $\sigma = 10^{-3}$), and number of iterations N

Output: Markov chain $\theta_1, \ldots, \theta_N$ that has $p(\theta)$ as its equilibrium distribution

```
for $t = 1$ to $N$ do
    for $k = 1$ to $K$ do
        Draw walkers $\theta_i$ and $\theta_j$ without replacement from the complementary ensemble $S_{-k} = \left\{\theta_1^{(t+1)}, \ldots, \theta_{k-1}^{(t+1)}, \theta_{k+1}^{(t)}, \ldots, \theta_K^{(t)}\right\}$
        Draw random noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$
        Compute proposed state $\theta_k' \longleftarrow \theta_k + \gamma(\theta_i - \theta_j) + \epsilon$
        Compute acceptance probability $\alpha_k = \min\left(1, f(\theta_k')/f(\theta_k^{(t)})\right)$
        Draw uniform number $u \sim \mathcal{U}(0, 1)$
        if $u < \alpha_k$ then
            Accept proposed state and set $\theta_k^{(t+1)} \longleftarrow \theta_k'$
        else
            Reject proposed state and set $\theta_k^{(t+1)} \longleftarrow \theta_k^{(t)}$
        end if
    end for
end for
```

The advantage of *differential evolution* over the *stretch moves* comes down to the flexibility of their proposals. The direction along which a walker moves in the context of the *stretch moves* is determined by a single walker. This means that at any given iteration, the number of equally possible directions is $K-1$. On the other hand, *differential evolution* moves each walker along a direction defined by two walkers. This implies that the total number of possible directions is given by the binomial combination $\binom{K-1}{2}$. The latter increases much faster with the number size of the ensemble K than the former, offering a larger variety of possible trajectories for the walkers. In other words, *differential evolution* is expected to perform better even with a lower number of walkers.

7.4 PARALLEL TEMPERING

So far we have only discussed ensemble methods that target a trivial product density given by the product of K copies of the target distribution as shown in equation 7.1. The main rationale for attempting to do this was to reduce the tuning requirements of MCMC. If we focus on addressing the challenge of multimodality, that is, the existence of multiple peaks in the target distribution, then we have to introduce a different product density as the extended target distribution.

One such choice is,

$$p^{*}\left(\{\theta_k\}_{k=1}^{K}\right) = \prod_{k=1}^{K} p_k(\theta_k)\,, \tag{7.9}$$

where

$$p_k(\theta_k) \propto p^{\beta_k}(d|\theta,\mathcal{M})p(\theta|\mathcal{M})\,, \tag{7.10}$$

is the *annealed or tempered posterior* that offers a simple interpolation between the prior $p(\theta|\mathcal{M})$ and the unnormalised posterior density $p(d|\theta,\mathcal{M})$ $p(\theta|\mathcal{M})$ for different monotonically–increasing values of $\beta_k \in [0,1]$. In the limit that $\beta_k = 1$ for all values of k, equation 7.9 reduces to the usual product density of equation 7.1.

The method of *parallel tempering (PT)* (Earl & Deem, 2005), also known as *replica exchange Monte Carlo (REMC)* (Hukushima & Nemoto, 1996; Swendsen & J.-S. Wang, 1986) or *Metropolis–coupled Markov chain Monte Carlo (MC^3)* (Geyer, 1991), relies on K parallel Markov chains, each one targeting a different tempered density. The β_k values are usually chosen *a priori* using a heuristic rule (e.g. $\beta_k = (k-1)^3/(K-1)^3$), or are set adaptively during the run using some diminishing adaptation scheme. The choice of MCMC method used for each different β_k is completely arbitrary and it can be anything from simple *Random–walk Metropolis* to *Hamiltonian Monte Carlo* or even an ensemble MCMC method.

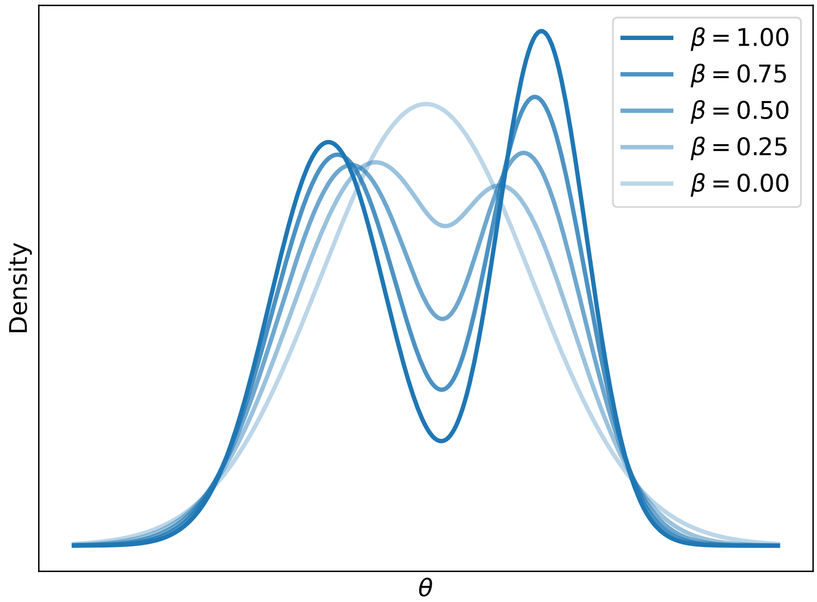

Figure 7.4: Illustration of the gradual tempering performed in the posterior distribution. The prior distribution corresponds to $\beta \longrightarrow 0$ and the posterior is recovered as $\beta \longrightarrow 1$.

So far, PT might look very similar to a parallel version of the sequential *simulated annealing* method in which $\beta_k = 1/T_k$ has the role of the inverse temperature. The crucial difference that makes PT so powerful is the fact that one can also perform *between–chain exchange moves*. Either periodically (e.g. once every 10 steps) or randomly (e.g. with probability 10%) a swap can take place between two states θ_i and θ_j that belong to different tempered posteriors (i.e. $\beta_i \neq \beta_j$). The reason that exchange/swap moves are desirable is that they enable the transfer of information from states of low β to those of higher β.

To understand how to perform a swap in practice let us consider the extended state,

$$\{\theta_k\}_{k=1}^{K} = \{\theta_1, \dots, \theta_i, \dots, \theta_j, \dots, \theta_K\}, \tag{7.11}$$

prior to the swap, where θ_i and θ_j are the two states that we want to exchange. This means that the proposed new state will be,

$$\{\tilde{\theta}_k\}_{k=1}^{K} = \{\theta_1, \dots, \theta_j, \dots, \theta_i, \dots, \theta_K\}. \tag{7.12}$$

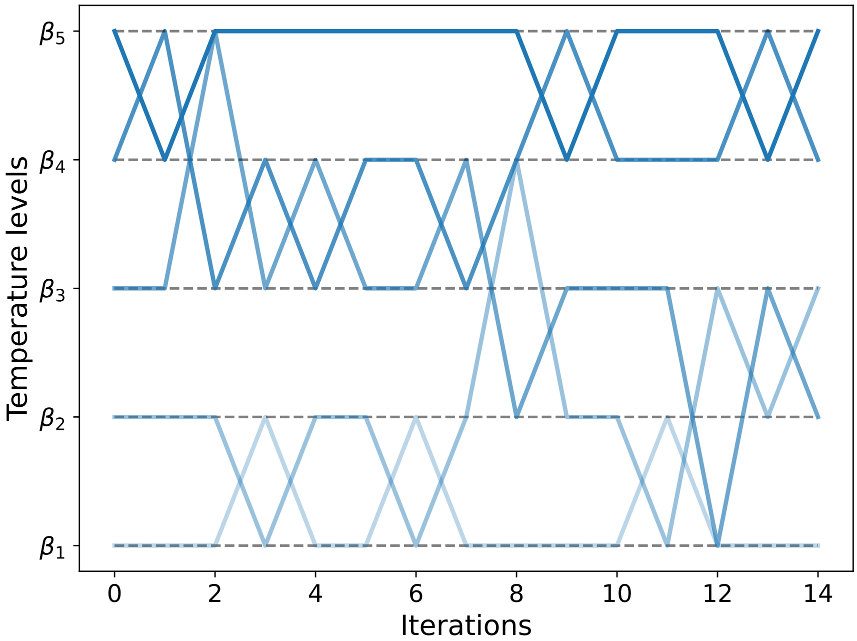

Figure 7.5: Illustration of the parallel tempering swaps performed between adjacent temperature levels.

Notice that the rest of the states, with the exception of θ_i and θ_j, are left unaffected by this exchange proposal. The *Metropolis acceptance probability* for this proposal is,

$$\begin{aligned}
\alpha_{ij} &= \min\left(1, \frac{p^*\left(\{\tilde{\theta}_k\}_{k=1}^K\right)}{p^*\left(\{\theta_k\}_{k=1}^K\right)}\right) \\
&= \min\left(1, \frac{p_i(\theta_j)p_j(\theta_i)}{p_i(\theta_i)p_j(\theta_j)}\right) \\
&= \min\left(1, \frac{p^{\beta_i}(d|\theta_j,\mathcal{M})p(\theta_j|\mathcal{M})p^{\beta_j}(d|\theta_i,\mathcal{M})p(\theta_i|\mathcal{M})}{p^{\beta_i}(d|\theta_i,\mathcal{M})p(\theta_i|\mathcal{M})p^{\beta_j}(d|\theta_j,\mathcal{M})p(\theta_j|\mathcal{M})}\right) \\
&= \min\left(1, \frac{p^{\beta_i}(d|\theta_j,\mathcal{M})p^{\beta_j}(d|\theta_i,\mathcal{M})}{p^{\beta_i}(d|\theta_i,\mathcal{M})p^{\beta_j}(d|\theta_j,\mathcal{M})}\right) \\
&= \min\left[1, \left(\frac{p(d|\theta_i,\mathcal{M})}{p(d|\theta_j,\mathcal{M})}\right)^{(\beta_j-\beta_i)}\right],
\end{aligned} \tag{7.13}$$

The chains are usually chosen to be in adjacent β_k levels (i.e. $i = j - 1$) such that the overlap between the typical sets of $p_i(\theta_i)$ and $p_j(\theta_j)$ is large, leading to high acceptance probabilities. Atchadé et al. (2011) estimated that the optimal acceptance rate is 23.4%. The exchange updates are typically performed after the local MCMC updates are completed in all β levels for a given iteration. Furthermore, there are different strategies for proposing swaps between adjacent temperature levels (Lingenheil et al., 2009). One option is to randomly select a pair of adjacent temperature levels per iteration. Another strategy involves proposing to swap all adjacent pairs starting from the lowest or highest β level and progressively moving towards the other end of the ladder. Finally, strategies that involve two steps, for instance, proposing to swap all even pairs in one iteration and all odd pairs in the next, have also been suggested in the literature (Lingenheil et al., 2009).

Algorithm 12 Parallel tempering

```
Input: initial state for the ensemble $\theta^{(1)} = (\theta_1^{(1)}, \dots, \theta_K^{(1)})$, prior probability density $\pi(\theta) \equiv p(\theta|\mathcal{M})$, likelihood function $\mathcal{L}(\theta) \equiv p(d|\theta, \mathcal{M})$, temperature ladder (e.g. $\beta_k = (k-1)^3/(K-1)^3$), local MCMC kernel $\theta' \longleftarrow \mathcal{K}(\theta; f(\theta))$ (e.g. a single random–walk Metropolis update), and number of iterations $N$
Output: $K$ Markov chains that each has $p_t(\theta) \propto \pi(\theta)\mathcal{L}(\theta)^{\beta_k}$ as its equilibrium distribution
for $t = 1$ to $N$ do
    for $k = 1$ to $K$ do
        Update state using local MCMC update $\theta'_k \longleftarrow \mathcal{K}(\theta_k^{(t)}; \pi(\theta)\mathcal{L}(\theta)^{\beta_k})$
    end for
    Draw random value of $k$ uniformly $k \sim \mathcal{U}(1, K-1)$
    Compute acceptance probability $\alpha_k = \min\left(1, \left[\frac{\mathcal{L}\left(\theta_k^{(t+1)}\right)}{\mathcal{L}\left(\theta_{k+1}^{(t+1)}\right)}\right]^{\beta_{k+1}-\beta_k}\right)$
    Draw uniform number $u \sim \mathcal{U}(0, 1)$
    if $u < \alpha_k$ then
        Accept proposed swap and set $\theta_k^{(t+1)} \longleftarrow \theta'_{k+1}$ and $\theta_{k+1}^{(t+1)} \longleftarrow \theta'_k$
    else
        Reject proposed swap and set $\theta_k^{(t+1)} \longleftarrow \theta'_k$ and $\theta_{k+1}^{(t+1)} \longleftarrow \theta'_{k+1}$
    end if
end for
```

8 EVIDENCE AND BAYES FACTOR COMPUTATION

> *There is nothing more deceptive than an obvious fact.*
>
> *— Arthur Conan Doyle*

8.1 NAIVE MONTE CARLO ESTIMATOR

The simplest estimator for the evidence we can construct is just the expectation value of the likelihood function with respect to the prior distribution (Hammersley & Handscomb, 1964; Raftery et al., 1991). The, so–called, *Naive Monte Carlo (NMC)* estimator can be computed as the sum

$$\hat{p}_{\mathrm{NMC}}(d|\mathcal{M}) = \frac{1}{n}\sum_{i=1}^{n} p(d|\theta_i, \mathcal{M})\,, \quad \text{with} \quad \theta_i \sim p(\theta|\mathcal{M})\,. \tag{8.1}$$

Although simple and unbiased, this approach can become extremely inefficient and result in a high variance in higher dimensions as the probability mass concentrates in the typical set that occupies a negligible fraction of the prior volume (Newton & Raftery, 1994). For this reason, this technique is only recommended for low–dimensional problems (i.e. $D \leq 3$).

8.2 IMPORTANCE SAMPLING ESTIMATOR

A more general strategy for the unbiased estimation of the evidence is importance sampling using samples from an auxiliary distribution $q(\theta)$. A simple estimator can then be constructed as,

$$\hat{p}_{\mathrm{IS}}(d|\mathcal{M}) = \frac{1}{n}\sum_{i=1}^{n} \frac{p(\theta_i|d, \mathcal{M})}{q(\theta_i)}\,, \quad \text{with} \quad \theta_i \sim q(\theta)\,. \tag{8.2}$$

These estimators share the same difficulty as most methods based upon importance sampling, that is, a large overlap between the typical set of the proposal and posterior distribution must be achieved for the method to be effective. Constructing effective proposal distributions becomes increasingly unmanageable as the number of dimensions increases and thus the application of this method on its own is limited to low dimensions. Finally, the importance sampling estimator reduces to the NMC one when the proposal distribution is chosen to be the prior.

8.3 HARMONIC MEAN ESTIMATOR

The *harmonic mean (HM)* estimator is another variation of the importance sampling estimator in which the posterior is used as the proposal and the prior as the target distribution (Newton & Raftery, 1994). This suggests the following estimator,

$$\hat{p}_{\mathrm{HM}}(d|\mathcal{M}) = \frac{1}{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{p(d|\theta_i\mathcal{M})}}\,, \quad \text{with} \quad \theta_i \sim p(\theta|d,\mathcal{M})\,. \tag{8.3}$$

The possible occurrence of samples with small likelihood value renders the variance of this estimator infinite (Neal, 2008). This pathology can be addressed by using a mixture $q(\theta) = \delta p(\theta|\mathcal{M}) + (1-\delta)p(\theta|d,\mathcal{M})$ between the prior and the posterior as the proposal distribution, where δ is very small (e.g. $\delta = 0.05$). The resulting method is then called the *stabilised harmonic mean (SHM)* estimator (Newton & Raftery, 1994).

8.4 LAPLACE ESTIMATOR

As discussed in detail in Section 4.6, for a sufficiently Gaussian target distribution $p(\theta)$ we can use the *Laplace approximation*, that is, a second order expansion around the mode, to estimate expectation values (Tierney & Kadane, 1986). Assuming that the target distribution is the unnormalised posterior $p(d|\theta\mathcal{M})p(\theta|\mathcal{M})$, the Gaussian approximation's mean is given by,

$$\mu = \arg\max_{\theta}\,\left[p(d|\theta,\mathcal{M})p(\theta|\mathcal{M})\right]\,, \tag{8.4}$$

following equation 4.23, and the inverse covariance is given by,

$$\left(\Sigma^{-1}\right)_{ij} = -\frac{\partial^2}{\partial\theta_i\partial\theta_j}\left[\log p(d|\theta,\mathcal{M}) + \log p(\theta|\mathcal{M})\right]\,, \tag{8.5}$$

following equation 4.24. Then, the model evidence is approximated by the normalising constant of the Gaussian, or in other words,

$$\begin{aligned}\hat{p}_L(d|\mathcal{M}) &= \int e^{-\frac{1}{2}(\theta-\mu)^T\Sigma^{-1}(\theta-\mu)}d\theta \\ &= (2\pi)^{D/2}\det(\Sigma)^{1/2}p(d|\theta=\mu,\mathcal{M})p(\theta=\mu|\mathcal{M})\,.\end{aligned} \tag{8.6}$$

As with any method, this result is only as good as the assumptions entering its calculation. The closer the posterior resembles a normal distribution, the better the outcome of the *Laplace* estimator will be.

8.5 BRIDGE SAMPLING

Originally, Meng & Wong (1996) introduced *bridge sampling (BS)* as a way to directly estimate the *Bayes factor* of two models, $\mathcal{M}_1$ and $\mathcal{M}_2$. However, in this section we present a version of BS that targets the model evidence of a single model $\mathcal{M}$. BS follows from the basic identity,

$$1 = \frac{\int p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) \alpha(\theta) q(\theta)}{\int p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) \alpha(\theta) q(\theta)}, \tag{8.7}$$

where $q(\theta)$ is the *proposal distribution* and $\alpha(\theta)$ is the so–called *bridge function* the *support* of which encompasses that of both the target posterior and of the proposal distribution.

Multiplying both sides of equation 8.7 with the model evidence $p(d|\mathcal{M})$ results in

$$\begin{aligned} p(d|\mathcal{M}) &= \frac{\int p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) \alpha(\theta) q(\theta)}{\int \frac{p(d|\theta,\mathcal{M}) p(\theta|\mathcal{M})}{p(d|\mathcal{M})} \alpha(\theta) q(\theta)} \\ &= \frac{\int p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) \alpha(\theta) q(\theta)}{\int p(\theta|d, \mathcal{M}) \alpha(\theta) q(\theta)}, \end{aligned} \tag{8.8}$$

which can be written as,

$$p(d|\mathcal{M}) = \frac{\mathbb{E}_{q(\theta)} \left[p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) \alpha(\theta) \right]}{\mathbb{E}_{p(\theta|d,\mathcal{M})} \left[\alpha(\theta) q(\theta) \right]}, \tag{8.9}$$

in terms of expectation values. The model evidence can then be approximated as,

$$\hat{p}(d|\mathcal{M}) = \frac{\frac{1}{n_2} \sum_{i=1}^{n_2} p(d|\tilde{\theta}_i, \mathcal{M}) p(\tilde{\theta}_i|\mathcal{M}) \alpha(\tilde{\theta}_i)}{\frac{1}{n_1} \sum_{j=1}^{n_1} \alpha(\theta_j^*) q(\theta_j^*)}, \tag{8.10}$$

where $\tilde{\theta}_i$ are samples from the proposal distribution,

$$\tilde{\theta}_i \sim q(\theta), \tag{8.11}$$

and θ_j^* are samples from the posterior distribution,

$$\theta_j^* \sim p(\theta|d, \mathcal{M}). \tag{8.12}$$

It is clear from the above discussion that BS relies on samples from both the proposal distribution $q(\theta)$, which plays the role of an *importance density*, and the posterior distribution $p(\theta|d, \mathcal{M})$. Often, the proposal distribution is some distribution that is easy to sample from and its *typical set* has a large overlap with the one of the posterior distribution. A common proposal used in practice is a normal distribution with its first two moments matching those of the posterior distribution.

Although highly arbitrary, the choice of the *bridge function* $\alpha(\theta)$ can have a significant impact on the precision of the method for a given proposal distribution. For instance, setting $\alpha(\theta) = [q(\theta)]^{-1}$ the BS estimator reduces to the naive Monte Carlo estimator, whereas setting $\alpha(\theta) = [p(d|\theta, \mathcal{M}) p(\theta|\mathcal{M}) q(\theta)]^{-1}$ leads to the *harmonic mean estimator*. Meng & Wong (1996) showed that the *optimal bridge function*, that is, the one that minimises the mean–square–error, is,

$$\alpha(\theta) = \frac{C}{s_1 p(d|\theta,\mathcal{M})p(\theta|\mathcal{M}) + s_2 p(d|\mathcal{M})q(\theta)}\,, \tag{8.13}$$

where $s_1 = n_1/(n_1+n_2)$ and $s_2 = n_2/(n_1+n_2)$ and C is a constant that cancels out and its value does not affect the outcome in any way. The *bridge function* of equation 8.13 depends on the model evidence $p(d|\mathcal{M})$, the same quantity that we are trying to approximate. We can resolve this issue by employing an iterative scheme,

$$\hat{p}_{\mathrm{BS}}^{(t+1)}(d|\mathcal{M}) = \frac{\frac{1}{n_2}\sum_{i=1}^{n_2}\frac{p(d|\tilde{\theta}_i,\mathcal{M})p(\tilde{\theta}_i|\mathcal{M})}{s_1 p(d|\tilde{\theta}_i,\mathcal{M})p(\tilde{\theta}_i|\mathcal{M})+s_2\hat{p}_{\mathrm{BS}}^{(t)}(d|\mathcal{M})q(\tilde{\theta}_i)}}{\frac{1}{n_1}\sum_{j=1}^{n_1}\frac{q(\theta_j^*)}{s_1 p(d|\theta_j^*,\mathcal{M})p(\theta_j^*|\mathcal{M})+s_2\hat{p}_{\mathrm{BS}}^{(t)}(d|\mathcal{M})q(\theta_j^*)}}\,, \tag{8.14}$$

starting from some initial guess of the value of the model evidence $\hat{p}_{\mathrm{BS}}^{(0)}(d|\mathcal{M})$ and keep updating it until the estimate has converged for some arbitrary tolerance level. Rearranging the terms on the right hand side, the aforementioned estimator can be written in the simpler form

$$\hat{p}_{\mathrm{BS}}^{(t+1)}(d|\mathcal{M}) = \frac{\frac{1}{n_2}\sum_{i=1}^{n_2}\frac{\ell_{2,i}}{s_1\ell_{2,i}+s_2\hat{p}_{\mathrm{BS}}^{(t)}(d|\mathcal{M})}}{\frac{1}{n_1}\sum_{j=1}^{n_1}\frac{1}{s_1\ell_{1,j}+s_2\hat{p}_{\mathrm{BS}}^{(t)}(d|\mathcal{M})}}\,, \tag{8.15}$$

where we have defined

$$\ell_{1,j} = \frac{p(d|\theta_j^*,\mathcal{M})p(\theta_j^*|\mathcal{M})}{q(\theta_j^*)}\,, \tag{8.16}$$

and

$$\ell_{2,i} = \frac{p(d|\tilde{\theta}_i,\mathcal{M})p(\tilde{\theta}_i|\mathcal{M})}{q(\tilde{\theta}_i)}\,. \tag{8.17}$$

Furthermore, the numerical stability of equation 8.15 can be improved and overflow issues avoided if we define

$$\hat{p}_{\mathrm{BS}}^{(t)}(d|\mathcal{M}) = \hat{r}^{(t)}\exp(\ell^*)\,, \tag{8.18}$$

and use the iterative formula

$$\hat{r}^{(t+1)} = \frac{\frac{1}{n_2}\sum_{i=1}^{n_2}\frac{\exp[\log(\ell_{2,i})-\ell^*]}{s_1\exp[\log(\ell_{2,i})-\ell^*]+s_2\hat{r}^{(t)}}}{\frac{1}{n_1}\sum_{j=1}^{n_1}\frac{1}{s_1\exp[\log(\ell_{1,j})-\ell^*]+s_2\hat{r}^{(t)}}}\,, \tag{8.19}$$

where ℓ^* is a constant that we can choose in order to make the sums numerically tractable, for instance $\ell^* = \mathrm{median}[\log(\ell_{1,j})]$.

Compared to other methods such as importance sampling or the harmonic mean estimator, BS estimates are more robust in cases in which the overlap between the *typical sets* of the proposal and posterior distribution is far from perfect.

8.6 THERMODYNAMIC INTEGRATION

A large body of work in *statistical physics* is concerned with methods for the estimation of normalising constants and *partition functions* in particular. The method of *thermodynamic integration (TI)* was developed for exactly this purpose (Gelman & Meng, 1998). Friel & Pettitt (2008) studied the particular case in which the normalising constant that is estimated using TI is the model evidence. To this end, they introduced the notion of the *power posterior*,

$$p(\theta|d,\beta,\mathcal{M}) \propto p^{\beta}(d|\theta,\mathcal{M})p(\theta|\mathcal{M})\,, \tag{8.20}$$

in which β is an auxiliary variable in the interval $[0,1]$. By construction, the normalising constant of the *power posterior* is simply,

$$p(d|\beta,\mathcal{M}) = \int p^{\beta}(d|\theta,\mathcal{M})p(\theta|\mathcal{M})d\theta\,, \tag{8.21}$$

where $p(d|\beta=1,\mathcal{M})$ is the model evidence and $p(d|\beta=0,\mathcal{M})$ is the integral over the prior which is simply equal to 1. Furthermore, the logarithm of the model evidence is,

$$\begin{aligned}\log p(d|\mathcal{M}) &= \log\left[\frac{p(d|\beta=1,\mathcal{M})}{p(d|\beta=0,\mathcal{M})}\right]\\ &= \int_0^1 \mathbb{E}_{p(\theta|d,\beta,\mathcal{M})}[\log p(d|\theta,\beta,\mathcal{M})]d\beta\,,\end{aligned} \tag{8.22}$$

that is, the integral over β of the expectation value of the likelihood with respect to the posterior for each value of β. To prove the above identity we first need to notice that,

$$\begin{aligned}\frac{d\log p(d|\beta,\mathcal{M})}{d\beta} &= \frac{1}{p(d|\beta,\mathcal{M})}\times\frac{dp(d|\beta,\mathcal{M})}{d\beta}\\ &= \frac{1}{p(d|\beta,\mathcal{M})}\int\frac{d}{d\beta}p^{\beta}(d|\theta,\mathcal{M})p(\theta|\mathcal{M})d\theta\\ &= \int\log p(d|\theta,\mathcal{M})\frac{p^{\beta}(d|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(d|\beta,\mathcal{M})}d\theta\\ &= \int\log p(d|\theta,\mathcal{M})p(\theta|d,\beta,\mathcal{M})d\theta\\ &= \mathbb{E}_{p(\theta|d,\beta,\mathcal{M})}\left[\log p(d|\theta,\mathcal{M})\right]\,,\end{aligned} \tag{8.23}$$

Integrating both sides with respect to β leads to equation 8.22 and completes the proof.

Using equation 8.22 to estimate the model evidence often requires the discretisation of the integral. A sequence $0=\beta_1<\beta_2<\cdots<\beta_m=1$ must be chosen *a priori* or based on *diminishing adaptation* scheme. The model evidence can then be approximated using the *trapezoidal rule*,

$$\hat{p}_{\mathrm{TI}}(d|\mathcal{M}) = \sum_{i=1}^{m}(\beta_{i+1}-\beta_i)\frac{E_i+E_{i+1}}{2}\,, \tag{8.24}$$

where,

$$E_i = \mathbb{E}_{p(\theta|d,\beta_i,\mathcal{M})}\left[\log p(d|\theta,\mathcal{M})\right] \,, \tag{8.25}$$

is the expected likelihood at β_i.

There are two sources of error in the above approximation. The first one is the Monte Carlo error that originates from the estimation of equation 8.25 using a finite number of samples. The second type has to do with the choice of the discretisation of β. Calderhead & Girolami (2009) showed that the discretisation error depends on the *Kullback–Leibler (KL) divergence* between subsequent densities $p(\theta|d,\beta_i,\mathcal{M})$ and $p(\theta|d,\beta_{i+1},\mathcal{M})$. This means that the optimal discretisation sequence of β values is the one minimising the KL divergence between subsequent power posteriors. Of course, knowing the optimal scheme is *a priori* hardly ever possible and thus we must rely on *ad hoc* choices (e.g. $\beta_i = (i-1)^3/(m-1)^3$) or *diminishing adaptation* strategies.

Thermodynamic integration can be combined with many different MCMC methods in order to estimate the model evidence. Perhaps the simplest one is to run m independent chains, either in parallel or serially, and then estimate the evidence using equation 8.24 where the expected likelihood of each discrete β value is computed with equation 8.25 for each chain. Furthermore, the chains do not even have to be independent for this method to work. Lastly, a parallel tempering approach can be followed as it is often done in practice.

8.7 ANNEALED IMPORTANCE SAMPLING

Annealed importance sampling (AIS) is another method that relies on a sequence of annealed or tempered distributions in order to construct an importance sampling estimator for the model evidence (Neal, 2001), similarly to *simulated annealing* and *parallel tempering*.

The basic idea is to use MCMC transitions in order to push a collection of n particles through a series of m *intermediate distributions*

$$p(\theta|d,\beta,\mathcal{M}) \propto p^{\beta}(d|\theta,\mathcal{M})p(\theta|\mathcal{M}) \,, \tag{8.26}$$

where $0 = \beta_1 < \beta_2 < \cdots < \beta_m = 1$, connecting the prior for $\beta_1 = 0$ to the posterior for $\beta_m = 1$. The particles are initialised by drawing samples from the prior

$$\theta_i^{(1)} \sim p(\theta|\mathcal{M}) \,, \tag{8.27}$$

and assigned (unnormalised) importance weights

$$w_i^{(1)} = 1 \,, \tag{8.28}$$

for $i \in \{1,\dots,n\}$ the particle index.

A number of N MCMC steps is then performed for each particle before the value of β is updated to the next value in the predefined sequence. The number N of MCMC steps is chosen such that the Markov chains defined by the particle trajectories have enough time to reach the stationary distribution. The critical difference between AIS and simulated annealing is that the associated importance weights w_i are updated during the run every time we move from one intermediate distribution to the next,

$$w_i^{(t+1)} = w_i^{(t)} \times \frac{p(\theta_i|d,\beta_{t+1},\mathcal{M})}{p(\theta_i|d,\beta_t,\mathcal{M})} = w_i^{(t)} \times \frac{p^{\beta_{t+1}}(d|\theta_i,\mathcal{M})}{p^{\beta_t}(d|\theta_i,\mathcal{M})} \,. \tag{8.29}$$

In practice, the logarithm of the weights is used in order to avoid numerical issues. Once the final distribution (i.e. the posterior) is reached and the particle weights are updated accordingly, the model evidence can be estimated as

$$\hat{p}_{\mathrm{AIS}}(d|\mathcal{M}) = \frac{1}{n}\sum_{i=1}^{n} w_i^{(m)} \,. \tag{8.30}$$

Furthermore, the samples $\theta_i^{(m)}$ combined with their respective weights $w_i^{(m)}$ can be used to compute arbitrary expectation values

$$\mathbb{E}_{p(\theta|d,\mathcal{M})}\left[f(\theta)\right] = \frac{\sum_{i=1}^{n} w_i^{(m)} f\left(\theta_i^{(m)}\right)}{\sum_{i=1}^{n} w_i^{(m)}} \,. \tag{8.31}$$

Assuming that the annealing process is slow enough (i.e. large number m of β levels and number N of MCMC steps) and a large enough collection of particles is used, then AIS yields unbiased estimates of the model evidence and weighted posterior samples, even in high dimensions. In the limit that the number of MCMC steps N goes to zero, the AIS estimator reduces to the usual importance sampling estimator.

8.8 SAVAGE–DICKEY DENSITY RATIO

Suppose now that we have two models or hypotheses and their respective parameters, $\mathcal{M}_0 : \theta$ and $\mathcal{M}_1 : \theta, \phi$, such that $\mathcal{M}_0$ is nested inside $\mathcal{M}_1$. This means that the more complex model, $\mathcal{M}_1$, is reduced to the simpler one, $\mathcal{M}_0$, for some specific choice of one or more of its parameters, $\phi = \phi_0$. This specific parameter choice is often called a *point–null hypothesis* as it is associated with zero probability mass in the context of the $\mathcal{M}_1$ model.

Evaluating the plausibility of this hypothesis can be done by computing the *Bayes factor* between the two models. The *Savage–Dickey density ratio (SDDR)* is a method that aims to do exactly this that was introduced by J. M. Dickey & Lientz (1970), J. M. Dickey (1971), Gunel & J. Dickey (1974), and J. M. Dickey (1976) who in turn attributed the origin of the method to *Leonard Jimmie Savage*.

Although the SDDR can be only applied to nested models, it has the advantage that it is simple to compute, given some posterior samples, without making any assumptions about the Gaussianity of the posterior distribution. In particular, the *Bayes factor* of $\mathcal{M}_0$ over $\mathcal{M}_1$ is simply,

$$\mathrm{BF}_{01} = \frac{p(d|\mathcal{M}_0)}{p(d|\mathcal{M}_1)} = \frac{p(\phi = \phi_0|d, \mathcal{M}_1)}{p(\phi = \phi_0|\mathcal{M}_1)}\,, \tag{8.32}$$

where the numerator of the right–hand–side ratio is just the *marginal* posterior of ϕ for $\mathcal{M}_1$ evaluated at $\phi = \phi_0$, and the denominator is the prior of ϕ for $\mathcal{M}_1$ evaluated at $\phi = \phi_0$. In other words, the *Bayes factor* is simply the marginal posterior to prior ratio for $\mathcal{M}_1$ evaluated at $\phi = \phi_0$. This means that only the parameters ϕ determine the value of the *Bayes factor*, and the nuisance parameters θ, that are common among the two models, are irrelevant. A schematic representation of SDDR is depicted in Figure 8.1.

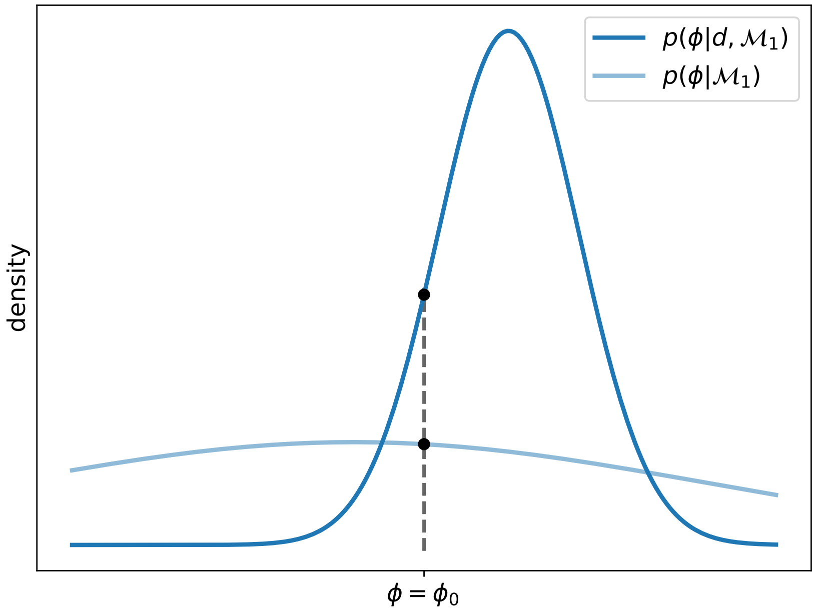

Figure 8.1: The *Savage–Dickey density ratio* expresses the Bayes factor BF_{01} as the ratio of marginal posterior to the prior density at the point ϕ_0 in which model M_1 reduces to M_0.

The validity of this method relies on two conditions. First, that the likelihood function of $\mathcal{M}_1$ has to reduce to that of $\mathcal{M}_0$

$$p(d|\theta, \phi = \phi_0, \mathcal{M}_1) = p(d|\theta, \mathcal{M}_0)\,, \tag{8.33}$$

and the same must be true for the prior

$$p(\theta|\phi = \phi_0, \mathcal{M}_1) = p(\theta|\mathcal{M}_0)\,. \tag{8.34}$$

The condition of equation 8.34 is satisfied by separable priors,

$$p(\theta, \phi|\mathcal{M}_1) = p(\theta|\mathcal{M}_1)p(\phi|\mathcal{M}_1)\,. \tag{8.35}$$

The proof of equation 8.32 is straightforward, starting with,

$$\begin{aligned} p(d|\mathcal{M}_0) &= \int p(d|\theta, \mathcal{M}_0) p(\theta|\mathcal{M}_0) d\theta \\ &= \int p(d|\theta, \phi = \phi_0, \mathcal{M}_1) p(\theta|\phi = \phi_0 \mathcal{M}_1) d\theta \\ &= p(d|\phi = \phi_0 \mathcal{M}_1)\,, \end{aligned} \tag{8.36}$$

in which we used the fact that $\mathcal{M}_0$ is nested in $\mathcal{M}_1$ for $\phi = \phi_0$. The next step is simply to employ Bayes' theorem

$$p(\phi = \phi_0|d, \mathcal{M}_1) = \frac{p(d|\phi = \phi_0 \mathcal{M}_1) p(\phi = \phi_0|\mathcal{M}_1)}{p(d|\mathcal{M}_1)}\,, \tag{8.37}$$

and solve for the ratio of model evidences by first substituting equation 8.36 into it to compete the proof.

Practical use of equation 8.32 requires the evaluation the marginal posterior of $\mathcal{M}_1$ at $\phi = \phi_0$. As the closed–form expression for the marginal posterior is rarely available, one can use samples from posterior (e.g. generated using MCMC) to create a density histogram for ϕ. Even better, *Kernel Density Estimation (KDE)* (Silverman, 2018) can be used to approximate the marginal posterior from samples as,

$$\hat{p}_{\mathrm{KDE}}(\phi|d, \mathcal{M}_1) = \frac{1}{n} \sum_{i=1}^{n} K_h\left(\phi - \phi_i\right)\,, \tag{8.38}$$

where K_h is the *kernel* and h is the *bandwidth*, a parameter that controls the smoothing. The *kernel* is generally a non–negative function, and most commonly it is chosen to be a simple Gaussian,

$$K_h(\phi) = \frac{1}{(2\pi)^{D/2} h^D} e^{-\frac{\phi^2}{2h^2}}\,, \tag{8.39}$$

where D is the dimensionality of ϕ (i.e. the number of elements of the ϕ vector). Finally, the value of h can either be determined on the basis of trial–and–error, or heuristics such as,

$$h = 1.06 \hat{\sigma} n^{-1/5}\,, \tag{8.40}$$

for the 1–D case where $\hat{\sigma}$ is the standard deviation of the samples (Silverman, 2018).

9 ADVANCED METHODS

Look on my works, ye Mighty, and despair!

— Percy Shelley, Ozymandias

This chapter introduces two advanced Monte Carlo methods, *Sequential Monte Carlo* and *Nested sampling*, which combine different previously introduced methods, such as MCMC and importance sampling, in order to provide samples from posterior distributions and estimate the model evidence. What distinguishes those two methods from all the previous ones introduced in this thesis, is their level of complexity and their reliance on multiple individual algorithms as their constituent parts.

9.1 SEQUENTIAL MONTE CARLO

Sequential Monte Carlo (SMC) is, from a physics point of view, conceptually related to the notion of *thermodynamic reversibility.* For a physical process starting from a state A and ending in a state B, to be thermodynamically reversible, the transition has to be slow enough such that each intermediate state of the system is approximately in equilibrium.

9.1.1 Background

The basic idea of SMC is to slowly guide a population of n particles $\{\theta_i^{(t)}\}_{i=1}^n$, drawn from a known probability distribution $\rho(\theta)$, through a series of intermediate distributions which create a path from $\rho(\theta)$ to the target distribution of interest $p(\theta)$ (Liu & R. Chen, 1998). In the context of SMC, the rate of this transition is governed by the number of intermediate distributions bridging $\rho(\theta)$ to $p(\theta)$. Just like in *annealed importance sampling (AIS)*, SMC relies on a number of MCMC steps performed in each intermediate step by every particle. This aims to equilibrate the particles by letting them reach the equilibrium distribution of each step. Furthermore, when transitioning from an intermediate distribution to the next, the particle distribution is adjusted using importance sampling. This guarantees that the particle distribution at any stage is the correct equilibrium distribution.

The main difference between SMC and AIS is the use of resampling in the case of SMC. During the run, the particle distribution might experience *weight degeneracy*, that is, only a few of the particles have non−negligible importance weights with the rest of them being vanishingly small. This high weight−variance can substantially affect any expectation values. In order to address this issue, SMC performs regular resampling steps, in which the particle distribution is resampled according to their weights, and the importance weights are re-initialised to be equal.

SMC methods are particularly suited for challenging target distributions which exhibit multiple modes. Furthermore, modifications of the main algorithm that we will present here can also be used for tasks of *online learning* in which the data arrive sequentially. These algorithms are most often called by the name of *particle filters* (Naesseth et al., 2019).

9.1.2 Bridging the prior and the posterior

A common way to construct such a sequence of intermediate distributions that bridge a known density $\rho(\theta)$ to the target density $p(\theta)$ is to interpolate between the two densities

$$p_t(\theta) \propto \rho^{1-\beta_t}(\theta)p^{\beta_t}(\theta)\,, \quad t = 1, \dots, m\,, \tag{9.1}$$

where β_t is a temperature annealing ladder, such that

$$0 = \beta_1 < \beta_2 < \cdots < \beta_m = 1\,. \tag{9.2}$$

In the Bayesian context, a natural choice is to set the prior as the auxiliary density $\rho(\theta) = p(\theta|\mathcal{M})$ and the posterior as the target density $p(\theta) = p(\theta|d, \mathcal{M})$. Equation 9.1 then reduces to the usual annealed or tempered interpolation

$$p_t(\theta) \propto p^{\beta_t}(d|\theta\mathcal{M})p(\theta|\mathcal{M})\,. \tag{9.3}$$

Although we will focus on this case, the algorithm is valid for any pair of distributions as long as the support of the auxiliary density encompasses that of the target.

9.1.3 Correction – Selection – Mutation

Given the initial positions of the particles $\{\theta_i^{(1)}\}_{i=1}^n$ drawn from the prior distribution, as well as the initial weights $\{W_i^{(1)}\}_{i=1}^n = 1/n$, SMC proceeds by the sequential application of the following three steps, selection, mutation, and correction, until the posterior density is reached. The procedure that takes place in a single iteration t is illustrated in Figure 9.1 and involves the steps:

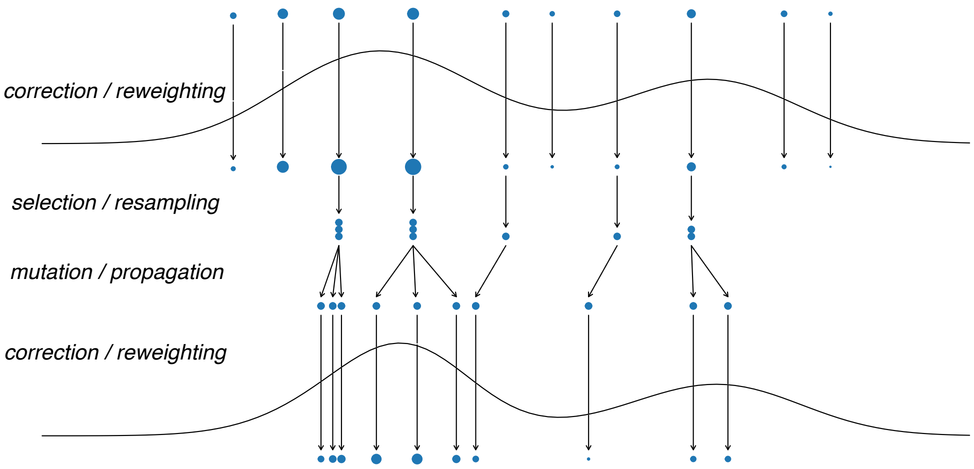

Figure 9.1: Illustration of the *Sequential Monte Carlo* algorithm with its three fundamental steps. During the correction step the particles are reweighted to represent the next probability distribution. Selection removes the particles with the smaller important weights and multiplies those with larger weights. Finally, mutation diversifies the particles by moving them.

1. **Correction / reweighting** – During this stage, the weights of the particles are updated according to

$$w_i^{(t)} = W_i^{(t-1)} \times \frac{p_t(\theta_i^{(t-1)})}{p_{t-1}(\theta_i^{(t-1)})} = W_i^{(t-1)} \times \left[p(d|\theta_i^{(t-1)}, \mathcal{M})\right]^{\beta_t - \beta_{t-1}} , \quad (9.4)$$

   where with w_i we denote the unnormalised weights and with $W_i = w_i / \sum_{i=1}^{n} w_i$ the normalised ones.

   The reweighing step accounts and corrects for any deviations of the particle distribution from the typical set of the target p_t. The ratio of the normalisation constants is estimated as

$$\frac{\mathcal{Z}_t}{\mathcal{Z}_{t-1}} = \sum_{i=1}^{n} w_i^{(t)} . \quad (9.5)$$

   Assuming the density for $t = 1$ corresponds to the prior for which $\mathcal{Z}_1 = 1$, equation 9.5 will eventually lead to the estimation of the model evidence $\mathcal{Z}_m = p(d|\mathcal{M})$.

2. **Selection / resampling** – The particle positions $\{\theta_i^{(t-1)}\}_{i=1}^{n}$ are resampled according to their weights $\{W_i^{(t-1)}\}_{i=1}^{n}$. The weights are then set again to be equal, $W_i^{(t-1)} = 1/n$. Their new, resampled, positions are denoted as $\{\tilde{\theta}_i^{(t-1)}\}_{i=1}^{n}$. Particles with small weight values are removed and those with large importance weights are multiplied.

   Resampling can be done using simple *multinomial resampling*, in which we draw n new particles, with replacement, with probabilities given by their weights, or using more advanced schemes characterised by lower variance (Li et al., 2015). This process can be performed in each iteration, or only when some criterion is triggered (e.g. when the effective sample size of the weights drops below a threshold).

Finally, caution must be taken when applying resampling too frequently. This could lead to the phenomenon of *weight impoverishment* in which there is no diversity between the particle positions. Fortunately, weight impoverishment is also reduced by the next step.

3. **Mutation / propagation** – Finally, the population of particles $\{\tilde{\theta}_i^{(t-1)}\}_{i=1}^n$ is updated and the particles move to their new positions $\{\theta_i^{(t)}\}_{i=1}^n$ by performing a number of MCMC steps targeting the density $p_t(\theta)$.

   The purpose of this step is to diversify the particles and allow their distribution to approach the stationary distribution. An advantage of SMC is that the particle distribution from the previous iteration can be used to construct efficient proposal distributions for MCMC for the current density. Furthermore, as n particles are updated at once, this step can be done in parallel. Any MCMC method can be used in this step and there is no requirement for the final/new positions to be uncorrelated from the initial ones, although in practice this helps reduce the variance of the estimates.

   A common approach is to use the particle covariance $\Sigma^{(t-1)}$ to construct a normal proposal distribution $q(\theta'|\theta) = \mathcal{N}(\theta'|\theta, \Sigma^{(t-1)})$.

Once all three steps are completed, the value of β is updated and the process is repeated again until β reaches the value of one. The names of those three steps are inspired by natural selection and evolutionary programming. The reason is the apparent analogy with genetic algorithms (Koza et al., 1994). More specifically, reweighting, resampling and propagation have the roles of correction, selection and mutation in genetic algorithms, in which the particle positions are the *genes* and the importance weights play the role of the so–called *fitness*. A critical difference with most genetic algorithms is the fact that SMC solves a sampling task, not an optimisation one, and thus the solution is represented by the distribution of the particles and not by any particle individually.

9.1.4 Effective sample size

A common measure of the quality of the importance weights of the particles, at any iteration of the SMC run, is the *effective sample size (ESS)*

$$\mathrm{ESS}_t = \frac{\mathbb{E}_{p_t}\left[w^{(t)}\right]^2}{\mathbb{E}_{p_t}\left[\left(w^{(t)}\right)^2\right]}. \tag{9.6}$$

Algorithm 13 Sequential Monte Carlo

```
Input: initial state for the ensemble $\theta^{(1)} = (\theta_1^{(1)}, \dots, \theta_K^{(1)})$, prior probability
    density $\pi(\theta) \equiv p(\theta|\mathcal{M})$, likelihood function $\mathcal{L}(\theta) \equiv p(d|\theta, \mathcal{M})$, and a local
    MCMC kernel $\theta' \longleftarrow \mathcal{K}(\theta; f(\theta))$ (e.g. $N$ steps of random−walk Metropolis
    update)
Output: Posterior samples and estimate of the model evidence $\mathcal{Z}$
Initialise temperature parameter $\beta_1 = 1$
Initialise estimate of evidence $\mathcal{Z} = 1$
for $i = 1$ to $n$ do
    Draw particle positions from the prior $\theta_i^{(1)} \sim \pi(\theta)$
    Initialise particle weights $W_i^{(1)} = 1/n$
end for
while $\beta_t \neq 1$ do
    Update iteration index $t \longleftarrow t + 1$
    Set temperature $\beta_t$ solving $\left(\sum_{i=1}^n w_i^{(t)}(\beta_t)\right)^2 / \sum_{i=1}^n \left(w_i^{(t)}(\beta_t)\right)^2 =$
    $\alpha \times n$ where the importance weights are computed as $w_i^{(t)} \longleftarrow$
    $W_i^{(t-1)} \mathcal{L}(\theta_i^{(t-1)})^{\beta_t - \beta_{t-1}}$
    Update evidence estimate $\mathcal{Z} \longleftarrow \mathcal{Z} \times n^{-1} \sum_{i=1}^n w_i^{(t)}$
    $\left\{ \tilde{\theta}_i^{(t-1)} \right\}_{i=1}^n \longleftarrow$ resample $\left\{ \theta_i^{(t-1)} \right\}_{i=1}^n$ according to $\left\{ w_i^{(t)} \right\}_{i=1}^n$
    for $i = 1$ to $n$ do
        Reset weights $W_i^{(t)} \longleftarrow 1/n$
    end for
    Update particles using MCMC
        $\left\{ \theta_i^{(t)} \right\}_{i=1}^n \longleftarrow \mathcal{K}\left( \left\{ \tilde{\theta}_i^{(t-1)} \right\}_{i=1}^n ; \pi(\theta)\mathcal{L}(\theta)^{\beta_t} \right)$
end while
```

which can be estimated as:

$$\hat{\text{ESS}}_t = \frac{\left(\sum_{i=1}^n w_i^{(t)}\right)^2}{\sum_{i=1}^n (w_i^{(t)})^2} = \frac{1}{\sum_{i=1}^n (W_i^{(t)})^2} \,. \tag{9.7}$$

9.1.5 Setting the temperature ladder

A β ladder can be specified *a priori* or determined adaptively during the run (Gelman & Meng, 1998). In the first case, the resampling step is usually triggered whenever the ESS drops below a prespecified threshold value (e.g. 50% – 95%). In the latter case, the next value of β is chosen adaptively such that the ESS has an approximately constant fraction α (e.g. 50% – 95%) of the number of particles n throughout the duration of the SMC run. Numerically, this can be done by solving

$$\frac{\left[\sum_{i=1}^{n} w_i^{(t)}(\beta_t)\right]^2}{\sum_{i=1}^{n} \left[w_i^{(t)}(\beta_t)\right]^2} = \alpha \times n\,, \tag{9.8}$$

for the next β_t such that $\beta_{t-1} < \beta_t \leq 1$ using, for instance, the *bisection method* (Burden et al., 2015).

9.2 NESTED SAMPLING

Nested sampling (NS), originally developed by Skilling (2004, 2006), is a method for estimating the model evidence $\mathcal{Z} = p(d|\mathcal{M})$. The basic idea is to approximate the evidence by integrating the prior in *nested* shells of constant likelihood. Despite its original purpose to estimate the model evidence, NS can also provide weighted samples from the posterior distribution as an optional byproduct. Therefore, the method is suitable for both tasks of parameter estimation and model comparison (Greg Ashton et al., 2022). Over the years, many variants of NS have emerged, with each one aiming to improve a different aspect of the original version (Brewer et al., 2011; Feroz, Michael P Hobson, et al., 2013; Higson et al., 2019).

9.2.1 Multi–dimensional integration

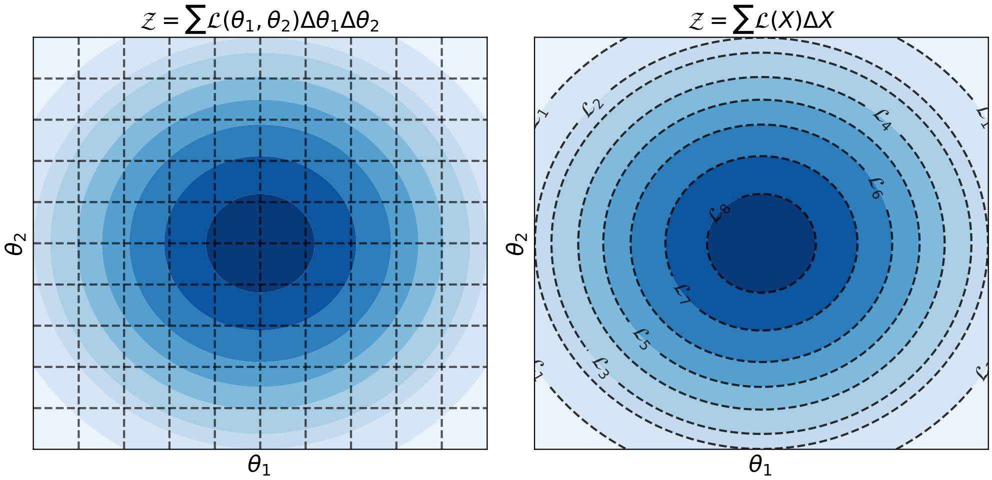
$\mathcal{Z} = \sum \mathcal{L}(\theta_1, \theta_2)\Delta\theta_1\Delta\theta_2$
θ_2
θ_1
$\mathcal{Z} = \sum \mathcal{L}(X)\Delta X$
θ_2
θ_1
$\mathcal{L}_1$
$\mathcal{L}_2$
$\mathcal{L}_3$
$\mathcal{L}_4$
$\mathcal{L}_5$
$\mathcal{L}_6$
$\mathcal{L}_7$
$\mathcal{L}_8$

Figure 9.2: Illustration comparing two ways which one can use to approximate the model evidence integral. The left panel shows the direct multi–dimensional integration over the parameters. The right panel shows the one–dimensional integration over the prior volume X enclosed in the iso–likelihood contours.

NS attempts to compute the evidence integral,

$$\mathcal{Z} = \int \mathcal{L}(\theta)\pi(\theta)d\theta\,, \tag{9.9}$$

where $\mathcal{L}(\theta) = p(d|\theta,\mathcal{M})$ is the likelihood function and $\pi(\theta) = p(\theta|\mathcal{M})$ is the prior, by transforming it into a one–dimensional integral over the prior volume

$$X(\lambda) = \int_{\mathcal{L}(\theta)>\lambda} \pi(\theta)d\theta, \tag{9.10}$$

enclosed in the iso–likelihood contour defined by $\mathcal{L}(\theta) = \lambda$. Equation 9.9 can then be written as

$$\mathcal{Z} = \int_0^{+\infty} X(\lambda)d\lambda = \int_0^1 \mathcal{L}(X)dX\,, \tag{9.11}$$

assuming that $\mathcal{L}(X(\lambda)) = \lambda$ exists. Figure 9.2 illustrates the equivalency between the integrals of equations 9.9 and 9.11. Unlike equation 9.9, the above integral is now 1–dimensional and can be approximated using standard numerical integration techniques (e.g. quadrature),

$$\mathcal{Z} = \sum_{i=1}^{m} \mathcal{L}_i w_i\,, \tag{9.12}$$

where,

$$w_i = \frac{X_{i-1} - X_{i+1}}{2}\,. \tag{9.13}$$

This of course assumes that we are able to evaluate the iso–likelihood contours $\mathcal{L}_i = \mathcal{L}(X_i)$ associated with an ordered collection of samples with prior volume $1 > X_1 > X_2 > \cdots > X_m > 0$. This is illustrated in Figure 9.3 for a collection of 8 samples that are uniformly distributed in the prior volume.

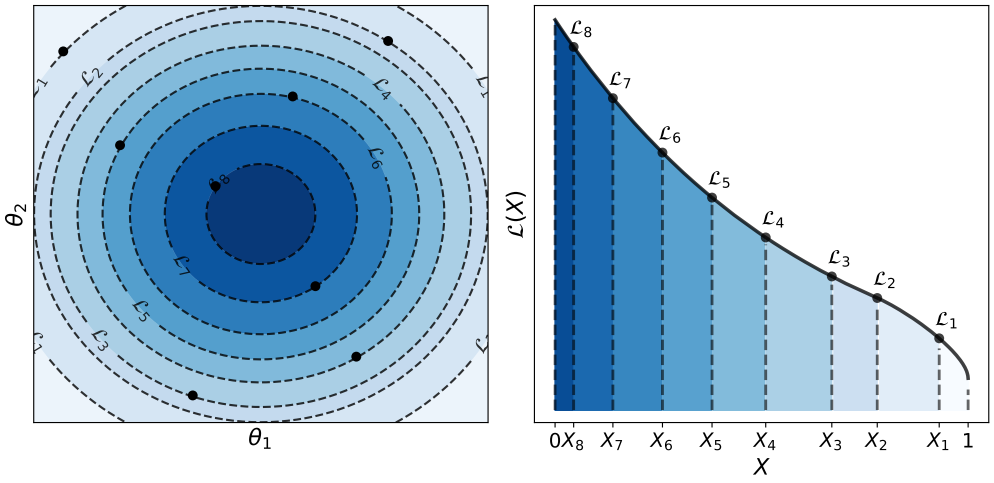

Figure 9.3: Illustration of 8 samples drawn uniformly from the prior with their respective iso–likelihood contours (*left*), along with their corresponding contributions to the evidence integral (*right*).

Using the simpler weights $w_i = X_i - X_{i+1}$ in equation 9.12 a lower bound on the evidence can be estimated as

$$\mathcal{Z} \geq \sum_{i=1}^{m} \mathcal{L}_i \left(X_i - X_{i+1}\right) . \tag{9.14}$$

Similarly, an upper bound also exists, using $w_i = X_{i-1} - X_i$, which can be written as

$$\mathcal{Z} \leq \sum_{i=1}^{m} \mathcal{L}_i \left(X_{i-1} - X_i\right) + X_m \mathcal{L}_{\max} , \tag{9.15}$$

where $\mathcal{L}_{\max}$ is the maximum likelihood value to be found as $X \longrightarrow 0$.

Soon after its original inception, it was realised that a NS run can also be used for the task of parameter estimation without any additional computation. In particular, the collected samples combined with their normalised weights

$$p_i = \frac{\mathcal{L}_i w_i}{\mathcal{Z}} , \tag{9.16}$$

correspond to weighted samples from the posterior distribution and thus can be used to compute expectation values

$$\mathbb{E}_p \left[f(\theta)\right] = \sum_{i}^{n} p_i f(\theta_i) . \tag{9.17}$$

9.2.2 Sampling procedure

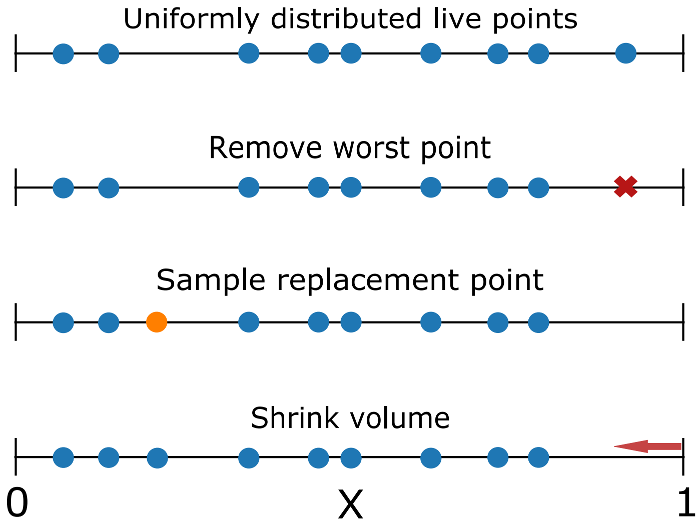

Figure 9.4: Illustration of the nested sampling procedure. Given some uniformly distributed points from the prior, we identify and remove the worst point, that is, the point with the minimum likelihood value. $\mathcal{L}_{min}$, and replace it a new point sampled from the prior subject to the likelihood constrain $\mathcal{L} > \mathcal{L}_{min}$. Finally, the volume is contracted to account for the removal of the worst point.

The NS algorithm begins by drawing a collection of points $\{\theta_i\}_{i=1}^n$ uniformly from the prior, often called *live points*. We can associate each live point θ_i with a prior volume X–value, namely the volume that would be enclosed by the iso–likelihood contour $\mathcal{L}_i = \mathcal{L}(\theta_i)$. On average, we expect roughly half of the live points to fall inside the iso–likelihood contour corresponding to half prior volume $X = 1/2$, one quarter to $X = 1/4$, one eighth to $X = 1/8$ and so on. In other words, since the live points are uniformly distributed under the prior, the corresponding X–values are uniformly distributed between 0 and 1. This is illustrated in Figure 9.3 and the top panel of Figure 9.4.

What we described so far is only the first step of the algorithm, and one still needs a way to propagate the live points into regions of smaller prior volume (i.e. lower X) in order to probe iso–likelihood contours corresponding to higher likelihood values. NS achieves this by first identifying the live point with the lowest likelihood value $\mathcal{L}^* = \mathcal{L}_1$, corresponding to volume X_1 and removing it. The remaining live points are now distributed over a compressed volume X_1. On average, the volume compression factor is

$$t = e^{-1/n}\,, \tag{9.18}$$

such that the compressed volume is $X_1 = t \times X_0$, where $X_0 = 1$ is the initial total volume. Finally, we sample a new live point to replace the one that we removed. The new point is sampled uniformly from the prior subject to the constrain $\mathcal{L} > \mathcal{L}^*$, that is, from the likelihood–constrained prior

$$\pi^*(\theta) \propto \begin{cases} \pi(\theta) & \text{if} \mathcal{L}(\theta) > \mathcal{L}^*\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{9.19}$$

This whole process, that is shown in Figure 9.4, is repeated multiple times until a criterion for termination is met. In each iteration, the volume shrinks on average by the compression factor of equation 9.18.

9.2.3 Termination criterion

During an NS run, the remaining prior volume X asymptotically approaches 0. The fact that we can only perform a finite number of steps means that it is unavoidable to introduce a truncation error into the evidence estimate of equation 9.11. A common way of determining when to stop is to approximately estimate the amount of remaining evidence and terminate the run when this can be considered negligible for the purpose of the analysis.

Perhaps the simplest way to roughly estimate the remaining evidence is by utilising the upper bound of equation 9.15. In this case, the remaining evidence is approximated as $\Delta\mathcal{Z} = \mathcal{L}_{\max} X_m$, where $\mathcal{L}_{\max}$ is the maximum likelihood value, estimated from the current population of live points, and X_m is simply the estimate of the remaining prior volume. An alternative would be to use the mean likelihood of the live points and get $\Delta\mathcal{Z} = \bar{\mathcal{L}}X_m$. The run then terminates when $\Delta\mathcal{Z}/\mathcal{Z}$ drops below a prespecified threshold.

Of course, neither of these approaches guarantees that the run will terminate early and that beyond lies a "spike" of huge likelihood. Upon deciding to stop, however, the current estimate of the model evidence is approximately corrected by either adding $\Delta\mathcal{Z}$ or removing the live points one–by–one in accordance with the NS procedure and adding their respective evidence contributions $\mathcal{L}_i X_i$, but without replacing them with new ones.

9.2.4 Uncertainty

So far we have assumed that the compression factor is given by equation 9.18, however, this is simply the mean compression factor associated with the removal of the outermost or lowest–likelihood live point. In truth, the prior volume bounded by the iso–likelihood contour of that point can be slightly different from what the mean compression factor predicts. The compression in volume t associated with the removal of the outermost of n live points follows a Beta$(n, 1)$ probability distribution with density,

$$p(t) = n t^{n-1}\,, \tag{9.20}$$

where the first n factor comes from the fact that any live point could be the outermost, and the second factor from the fact that the remaining $n - 1$ live points lie uniformly distributed above the outermost at t.

The compression factors can therefore be sampled from the probability distribution of equation 9.20 instead of just assuming their expected value of equation 9.18. Furthermore, we can use equation 9.20 to compute the expectation values

$$\mathbb{E}[\log t] = -\frac{1}{n}\,, \qquad \mathrm{Var}[\log t] = \frac{1}{n^2}\,. \tag{9.21}$$

Since the individual $\log t$ are independent, we expect that after i steps, the prior volume to have shrunk to

$$\log X_i = -\frac{i \pm \sqrt{i}}{n}\,. \tag{9.22}$$

What the above expression means is that there is uncertainty in the estimates of the compression factor which enter into the prior volume estimates too. In other words, there is uncertainty in the number of steps i required for the prior volume to shrink to a certain value X_i.

The uncertainty originating from the noisy estimates of the compression factor t also propagates into the estimate of the model evidence. To quantify this we need to consider the information gained when transitioning from the prior $\pi(\theta)$ to the posterior $p(\theta)$, given by the *Kullback–Leibler (KL) divergence*,

$$H = \int p(\theta) \log \left(\frac{p(\theta)}{\pi(\theta)} \right) d\theta \,. \tag{9.23}$$

We can rewrite the above equation in terms of the prior volume X, as

$$\begin{aligned} H &= \int p(X) \log p(X) dX \\ &= -\int p(X) \log X dX + \int p(\log X) \log p(\log X) d \log X \,, \end{aligned} \tag{9.24}$$

where $p(X) = \mathcal{L}(X)/\mathcal{Z}$ is the volume posterior density. Ignoring the second term on the right hand size, which is subdominant, we thus get that the KL divergence provides a measure of the compression we require to reach the bulk of the posterior mass,

$$H = -\log X \,. \tag{9.25}$$

Comparing equation 9.22 and 9.25 we roughly expect $nH \pm \sqrt{nH}$ steps to reach the bulk of the posterior mass. Equivalently, the uncertainty introduced into the estimate of the model evidence is

$$\Delta \log \mathcal{Z} = \sqrt{\frac{H}{n}} \,. \tag{9.26}$$

Of course, the above expression does not include any sources of numerical error such as truncation error.

9.2.5 Likelihood–constrained prior sampling

The efficient application of the NS algorithm requires sampling from the prior distribution $\pi(\theta)$ subject to the likelihood constrain $\mathcal{L} > \mathcal{L}^{*}$. Unfortunately, drawing points from the prior until the likelihood criterion is met is not feasible in practice, as the volume contained in the constrained prior shrinks exponentially with each iteration. For this reason, two different approaches, *region* and *step* samplers are often employed in order to produce samples from the likelihood–constrained prior.

For the sake of simplicity, both samplers usually operate in the *latent* parameter space that the prior is uniform over the unit hypercube. In this case, the practitioner specifies their prior preference, not by providing a (log–) probability density function, but by defining the *inverse–cumulative density function* Φ^{-1} that transforms points ϕ in the hypercube to points θ in the original parameter space. For instance, let us assume that we require a normal prior on a parameter $\theta \sim \mathcal{N}(\mu, \sigma^2)$. We can transform a unit hypercube parameter $\phi \sim \mathcal{U}(0, 1)$, using the standard normal distribution's *inverse–cumulative density function* Φ^{-1}, such that,

$$\theta = \mu + \Phi^{-1}(\phi)\sigma \,. \tag{9.27}$$

Region samplers

The basic idea behind *region* samplers is to construct a hypersurface that bounds a given iso–likelihood contour. In practice, this is done using simple geometric shapes (e.g. spheres, ellipses, etc.). The hypersurface must encompass the current distribution of live points and at least contain the currently estimated volume. One can then sample uniformly from within the volume enclosed by the hypersurface until the likelihood–constraint is satisfied. In order to reduce the risk of missing parts of the currently estimated volume, the bounding region is usually expanded by a prespecified factor or using cross–validation of the live points.

Most region samplers attempt to construct such a bounding region by wrapping the current generation of live points with one or multiple ellipsoids. Using multiple ellipsoids offers some flexibility in the case of multimodal posterior distributions. The most popular such sampler is the *MultiNest* algorithm that determines the shape and location of the ellipsoids based on the mean and covariance of the live points, by first estimating the number of distinct modes, and thus required ellipsoids, using a clustering algorithm (Feroz, M. Hobson, et al., 2009).

Region samplers have to face serious challenges when the complexity of the posterior or the dimensionality of the parameter space increases. In the first case, the ability to accurately bound the current volume depends on the number of live points, with a higher number often resulting in better bounding regions. The second limitation arises from the curse of dimensionality. As the number of dimensions increases, most of the volume of the bounding shape concentrates near its edges, and given that the bounding region is often chosen to be significantly broader than the encompassing likelihood–constrained volume to guarantee that no parts are encroached, the total number of samples until one is found to lie within the iso–likelihood contour increases exponentially. As a consequence, region samplers are more efficient and appropriate for low–dimensional problems, that is, $D \lesssim 10 - 20$.

Step samplers

On the other hand, *step* samplers do not rely on a bounding region and thus bypass some of the pathologies of *region* samplers. Instead, they evolve a randomly chosen live point through a sequence of local steps to an approximately independent position. This is usually achieved using some MCMC method targeting the likelihood–constrained prior of equation 9.19 as the target distribution. The advantage of using MCMC methods in the context of NS is that, in each iteration, one can use the distribution of the live points to construct effective proposal distributions for the MCMC sampler.

Although *step* samplers enjoy a better scaling with the number of dimensions than region samplers, there are still challenges in their use. First of all, determining the minimum number of steps to perform for the new point to be independent of its starting position (i.e. the randomly chosen live point) is not trivial. Although small correlations can be effectively ignored, larger violations can have catastrophic results and lead to substantial bias in the final estimates of NS. Furthermore, the step sampler must be tuned appropriately to achieve good sampling performance. Adaptation during a given iteration has to be diminishing in order to avoid spurious effects and biases.

Although any MCMC method (e.g. random walk Metropolis, slice sampling, etc.) can in principle be used as a step sampler, there are also methods that are naturally suited and have been developed for use in the context of sampling from the likelihood–constrained prior. One such example is *Galilean Monte Carlo (GMC)* (Feroz & Skilling, 2013; Skilling, 2012, 2019) that samples by moving consistently along a direction until a proposed point is rejected, by being outside the iso–likelihood contour. In this case, the sampler reflects off the current iso–likelihood boundary.

9.2.6 Parallelisation

Parallelising NS is not as straightforward as with other Monte Carlo algorithms (e.g. Sequential Monte Carlo), as the method relies on updating a single (worst) point at a time. In general, we would like to generate as many candidate points per step as the number of available CPUs (i.e. n_{CPU}) and evaluate their likelihoods in parallel. In this case, there are three strategies that one can follow:

1. Replace a single live point and discard as many as $n_{\mathrm{CPU}} - 1$ acceptable live points. This scheme is quite wasteful, particularly in cases in which it is likely that more than one candidate point satisfies the likelihood constraint.

2. Replace the worst (i.e. lowest likelihood) n_{CPU} live points in a single step. This results in linear speed–up with respect to the number of CPUs but increases the variance of the evidence estimate by a factor of $\sqrt{n_{\mathrm{CPU}}}$.

3. Replace a single live point and consider the other $n_{\mathrm{CPU}} - 1$ candidates for subsequent iterations. This results in a speed–up of $n \log(1 + n_{\mathrm{CPU}}/n)$ which is approximately linear for $n_{\mathrm{CPU}} << n$. The "diminishing returns" represented by the logarithmic factor in this expression originate from the fact that the likelihood threshold increases as the run progresses, and thus the points might not be valid for a subsequent iteration. This strategy is the most widely employed in practice.

Finally, it is important to note that apart from parallelising a single NS run, it is also possible to combine different, possibly parallel, independent

NS runs into a joint one, thus achieving linear scaling. In order to combine two or more runs together, we collect the points from all runs as live points and begin by removing the worst point, which with no loss of generality we assume that it belongs to run A. Then, as a replacement that satisfies the likelihood constraint, we simply take the replacement that was originally used in run A. We then proceed with the next worst point until all points are accounted for.

Algorithm 14 Nested sampling

```
Input: termination criterion (e.g. $\Delta \log \mathcal{Z} \leq \epsilon$), number of live points $n$, an
    estimate of the compression factor e.g. $t = \exp(-1/n)$, prior distribution
    $\pi(\theta)$, likelihood function $p_i = \mathcal{L}(\theta)$)
Output: Estimate of model evidence $\mathcal{Z}$, posterior samples $\tilde{\theta}_i$ with weights
    $\mathcal{L}_i w_i / \mathcal{Z}$
Initialise volume $X = 1$ and evidence $\mathcal{Z} = 0$
Draw $n$ live points from the prior $\theta_1, \theta_2, \ldots, \theta_n \sim \pi(\theta)$
repeat
    Find the minimum likelihood value of the live points $\mathcal{L}^* \longleftarrow \min(\mathcal{L}(\theta_1), \ldots, \mathcal{L}(\theta_n))$
    Replace live point $\theta^*$ corresponding to $\mathcal{L}^*$ with a new point from the prior satisfying $\mathcal{L} > \mathcal{L}^*$
    Set $w^* \longleftarrow \Delta X$ where $\Delta X = (1 - t)X$
    Update estimate of the evidence $\mathcal{Z} \longleftarrow \mathcal{Z} + \mathcal{L}^* w^*$
    Store values of $w_i \longleftarrow w^*$, $\mathcal{L}_i \longleftarrow \mathcal{L}^*$, and $\tilde{\theta}_i \longleftarrow \theta^*$
    Contract volume $X \longleftarrow tX$
until termination criteria satisfied
Update evidence $\mathcal{Z} \longleftarrow \mathcal{Z} + \frac{1}{n} \sum_{j=1}^{n} \mathcal{L}(\theta_j)\, X$
```

Part III

NOVEL DEVELOPMENTS

10 ENSEMBLE SLICE SAMPLING

This chapter presents *Ensemble Slice Sampling* which is the main contribution introduced in the paper titled *Ensemble Slice Sampling: Parallel, black–box, and gradient–free inference* that was published in the journal *Statistics and Computing* in 2021 (Karamanis & Beutler, 2021). The content of the chapter is almost identical to that included in the aforementioned publication with the exception of minor text and figure formatting differences.

Slice Sampling has emerged as a powerful Markov Chain Monte Carlo algorithm that adapts to the characteristics of the target distribution with minimal hand-tuning. However, Slice Sampling's performance is highly sensitive to the user-specified initial length scale hyperparameter and the method generally struggles with poorly scaled or strongly correlated distributions. This paper introduces Ensemble Slice Sampling (ESS), a new class of algorithms that bypasses such difficulties by adaptively tuning the initial length scale and utilising an ensemble of parallel walkers in order to efficiently handle strong correlations between parameters. These affine–invariant algorithms are trivial to construct, require no hand-tuning, and can easily be implemented in parallel computing environments. Empirical tests show that Ensemble Slice Sampling can improve efficiency by more than an order of magnitude compared to conventional MCMC methods on a broad range of highly correlated target distributions. In cases of strongly multimodal target distributions, Ensemble Slice Sampling can sample efficiently even in high dimensions. We argue that the parallel, black-box and gradient-free nature of the method renders it ideal for use in scientific fields such as physics, astrophysics and cosmology which are dominated by a wide variety of computationally expensive and non-differentiable models.

10.1 INTRODUCTION

Bayesian inference and data analysis has become an integral part of modern science. This is partly due to the ability of Markov Chain Monte Carlo (MCMC) algorithms to generate samples from intractable probability distributions. MCMC methods produce a sequence of samples, called a *Markov chain*, that has the target distribution as its equilibrium distribution. The more samples are included, the more closely the distribution of the samples approaches the target distribution. The Markov chain can then be used to numerically approximate expectation values (e.g. parameter uncertainties, marginalised distributions).

Common MCMC methods entail a significant amount of time spent hand-tuning the hyperparameters of the algorithm to optimize its efficiency with respect to a target distribution. The emerging and routine use of such mathematical tools in science calls for the development of black-box MCMC algorithms that require no hand-tuning at all. This need led to the development of adaptive MCMC methods like the Adaptive Metropolis algorithm (Haario et al., 2001) which tunes its proposal scale based on the sample covariance matrix. Unfortunately, most of those algorithms still include a significant number of hyperparameters (e.g. components of the covariance matrix) rendering the adaptation noisy. Furthermore, the tuning is usually performed on the basis of prior knowledge, such as one or more long preliminary runs which further slow down the sampling. Last but not least, there is no reason to believe that a single Metropolis proposal scale is optimal for the whole distribution (i.e. the appropriate scale could vary from one part of the distribution to another). Another approach to deal with those issues would be to develop methods that by construction require no or minimal hand-tuning. An archetypal such method is the Slice Sampler (Neal, 2003), which has only one hyperparameter, the initial length scale.

It should be noted that powerful adaptive methods that require no hand-tuning (although they do require preliminary runs) already exist. Most notable of them is the No U-Turn Sampler (NUTS) (M. D. Hoffman, Gelman, et al., 2014), an adaptive extension of Hamiltonian Monte Carlo (HMC) (Neal et al., 2011). However, such methods rely on the gradient of the log probability density function. This requirement is the reason why these methods are limited in their application in quantitative fields such as physics, astrophysics and cosmology, which are dominated by computationally costly non–differentiable models. Thus, our objective in this paper is to introduce a parallel, black-box and gradient–free method that can be used in the aforementioned scientific fields.

This paper presents Ensemble Slice Sampling (ESS), an extension of the Standard Slice Sampling method. ESS naturally inherits most of the benefits of Standard Slice Sampling, such as the acceptance rate of 1, and most importantly the ability to adapt to the characteristics of a target distribution without any hand-tuning at all. Furthermore, we will show that ESS's performance is insensitive to linear correlations between the parameters, thus enabling efficient sampling even in highly demanding scenarios. We will also demonstrate ESS's performance in strongly multimodal target distributions and show that the method samples efficiently even in high dimensions. Finally, the method can easily be implemented in parallel taking advantage of multiple CPUs thus facilitating Bayesian inference in cases of computationally expensive models.

Our implementation of ESS is inspired by Tran & Ninness (2015). However, our method improves upon that by extending the direction choices (e.g. Gaussian and global move), adaptively tuning the initial proposal scale, and parallelising the algorithm. Nishihara et al. (2014) developed a general algorithm based on the elliptical slice sampling method (Murray et al., 2010) and a Gaussian mixture approximation to the target distribution. ESS utilises an ensemble of parallel and interacting chains, called walkers. Other methods that are based on the ensemble paradigm include the Affine Invariant Ensemble Sampler (Goodman & Weare, 2010) and the Differential Evolution MCMC (Ter Braak, 2006) along with its various extensions (Ter Braak & Vrugt, 2008; Vrugt et al., 2009), as well as more recent approaches that are based on langevin diffusion dynamics (Garbuno-Inigo, Hoffmann, et al., 2020; Garbuno-Inigo, Nüsken, et al., 2020) and the time discretization of stochastic differential equations (Leimkuhler, Matthews, et al., 2018) in order to achieve substantial speedups.

In Section 10.2, we will briefly discuss the Standard Slice Sampling algorithm. In Section 10.3, we will introduce the Ensemble Slice Sampling method. In Section 10.4 we will investigate the empirical evaluation of the algorithm. We reserve Sections 10.5 and 10.6 for discussion and conclusion, respectively.

10.2 STANDARD SLICE SAMPLING

Slice Sampling is based on the idea that sampling from a distribution $p(x)$ whose density is proportional to $f(x)$ is equivalent to uniformly sampling from the region underneath the graph of $f(x)$. More formally, in the univariate case, we introduce an auxiliary variable, the height y, thus defining the joint distribution $p(x, y)$, which is uniform over the region $U = \{(x, y) : 0 < y < f(x)\}$. To sample from the marginal density for x, $p(x)$, we sample from $p(x, y)$ and then we ignore the y values.

Generating samples from $p(x, y)$ is not trivial, so we might consider defining a Markov chain that will converge to that distribution. The simplest, in principle, way to construct such a Markov chain is via Gibbs sampling. Given the current x, we sample y from the conditional distribution of y given x, which is uniform over the range $(0, f(x))$. Then we sample the new x from the *slice* $S = \{x : y < f(x)\}$.

Generating a sample from the slice S may still be difficult, since we generally do not know the exact form of S. In that case, we can update x based on a procedure that leaves the uniform distribution of S invariant. Neal (2003) proposed the following method:

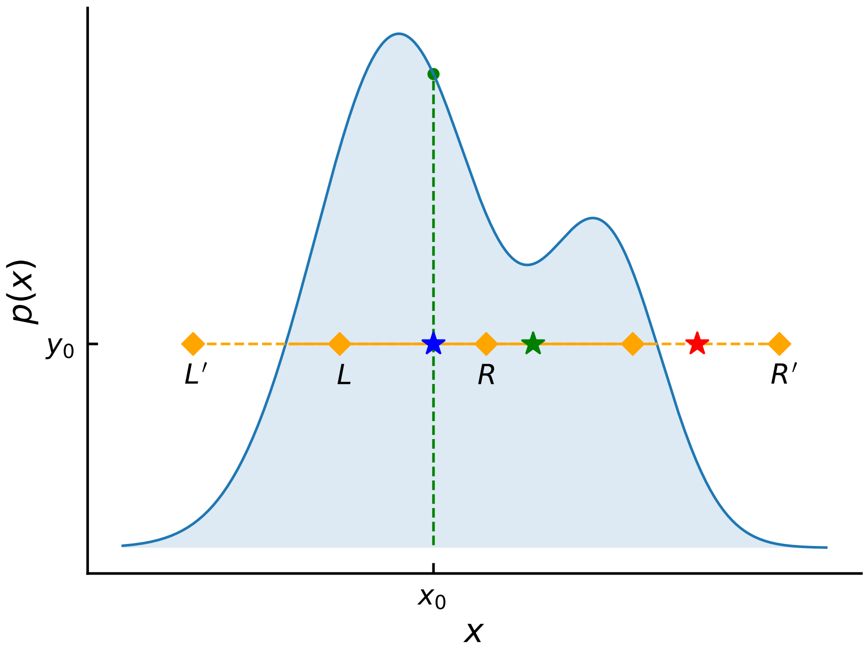

Figure 10.1: The plot shows the univariate slice sampling method. Given an initial value x_0, a value y_0 is uniformly sampled along the vertical slice $(0, f(x_0))$ (green dashed line) thus defining the initial point (blue star). An interval (L, R) is randomly positioned horizontally around the initial point, and then it is expanded in steps of size $\mu = R - L$ until both of its ends L', R' are outside the slice. The new point (green star) is generated by repeatedly sampling uniformly from the expanded interval (L', R') until a point is found inside the slice. Points outside the slice (e.g. the red star) are used to shrink the interval (L', R') by moving L' or in this case R' to that point and accelerate the sampling procedure.

Given the current state x_0, the next one is generated as:

1. Draw y_0 uniformly from $(0, f(x_0))$, thus defining the horizontal slice $S = \{x : y_0 < f(x)\}$,
2. Find an interval $I = (L, R)$ that contains all, or much, of S (e.g. using the stepping-out procedure defined below),
3. Draw the new point x_1 uniformly from $I \cap S$.

In order to find the interval I, Neal (2003) proposed to use the *stepping-out* procedure that works by randomly positioning an interval of length μ around the point x_0 and then expanding it in steps of size μ until both ends are outside of the slice. The new point x_1 is found using the *shrinking* procedure, in which points are uniformly sampled from I until a point inside S is found. Points outside S are used to shrink the interval I. The stepping-out and shrinking procedures are illustrated in Figure 10.1. By construction, the stepping-out and shrinking procedures can adaptively tune a poor estimate of the length scale μ of the initial interval. The length scale μ is the only free hyperparameter of the algorithm. For a detailed review of the method we direct the reader to Neal, 2003 and MacKay, 2003 (also Exercise 30.12 in that text).

It is important to mention here that for multimodal distributions there is no guarantee that the slice would cross any of the other modes, especially if the length scale is underestimated initially. Ideally, in order to provide a large enough initial value of the scale factor μ, prior knowledge of the distance between the modes is required. As we will show in the next section, Ensemble Slice Sampling does not suffer from this complication and can handle strongly multimodal distributions efficiently.

10.3 ENSEMBLE SLICE SAMPLING

The univariate slice sampling scheme can be used to sample from multivariate distributions by sampling repeatedly along each coordinate axis in turn (one parameter at a time) or by sampling along randomly selected directions (MacKay, 2003). Using either of those choices, the Standard Slice Sampler performs acceptably in cases with no strong correlations in parameter space. The overall performance of the algorithm generally depends on the number of expansions and contractions during the stepping-out and shrinking procedures, respectively. Ideally we would like to minimize that number. A reasonable initial estimate of the length scale is still required in order to reduce the amount of time spent expanding or contracting the initial interval.

However, when strong correlations are present two issues arise. First, there is no single value of the initial length scale that minimizes the computational cost of the stepping-out and shrinking procedures along all directions in parameter space. The second problem concerns the choice of direction. In particular, neither the component-wise choice (one parameter at a time) nor the random choice is suitable in strongly correlated cases. Using such choices results in highly autocorrelated samples.

Our approach would be to target each of those two issues individually. The resulting algorithm, Ensemble Slice Sampling (ESS), is invariant under affine transformations of the parameter space, meaning that its performance is not sensitive to linear correlations. Furthermore, ESS minimizes the computational cost of finding the slice by adaptively tuning the initial length scale. Last but not least, unlike most MCMC methods, ESS is trivially parallelizable, thus enabling the data analyst to take advantage of modern high performance computing facilities with multiple CPUs.

10.3.1 Adaptively tuning the length scale

Let us first consider the effect of the initial length scale on the performance of the univariate slice sampling method. For instance, if the initial length scale is λ times smaller than the actual size of the slice, then the stepping-out procedure would require $\mathcal{O}(\lambda)$ steps in order to fix this. However, in this case, since the final interval is an accurate approximation of the slice there would probably be no contractions during the shrinking phase. On the other hand, when the initial length scale is larger than the actual slice then the number of expansions would be either one or zero. In this case though, there would be a number of contractions.

Stochastic approximation

As the task is to minimize the total number of expansions and contractions we employ and adapt the *Robbins–Monro* stochastic approximation algorithm (Robbins & Monro, 1951) of Tibbits et al. (2014). Ideally, based on the reasoning of the previous paragraph, only one expansion and one contraction will take place. Therefore, the target ratio of number of expansions to total number of expansions and contractions is 1/2. To achieve this, we update the length scale μ based on the following recursive formula:

$$\mu^{(t+1)} = 2\mu^{(t)} \frac{N_e^{(t)}}{N_e^{(t)} + N_c^{(t)}} \,, \tag{10.1}$$

where $N_e^{(t)}$ and $N_c^{(t)}$ are the number of expansions and contractions during iteration t. It is easy to see that when the fraction $N_e^{(t)}/(N_e^{(t)} + N_c^{(t)})$ is larger than 1/2 the length scale μ will be increased. In the case where the fraction is smaller than 1/2 the length scale μ will be decreased accordingly. The optimization can stop either when the fraction is close to 1/2 within a threshold or when a maximum number of tuning steps has been completed. The pseudocode for the first case is shown in Algorithm 15. In order to preserve detailed balance it is important to be sure that the adaptation stops after a finite number of iterations. In practice this happens after $\mathcal{O}(10)$ iterations. An alternative would be to use diminishing adaptation (Roberts & Rosenthal, 2007) but we found that our method is sufficient in practice (see Section 4.3 for more details).

10.3.2 The choice of direction and parallelisation

In cases where the parameters are correlated we can accelerate mixing by moving more frequently along certain directions in parameter space. One way of achieving this is to exploit some prior knowledge about the covari-

Algorithm 15 Function to tune the length scale μ.

```
function TuneLengthScale($t$, $\mu^{(t)}$, $N_e^{(t)}$, $N_c^{(t)}$, $M^{\text{adapt}}$)
if $t \leq M^{\text{adapt}}$ then
    Compute $\mu^{(t+1)}$ using Equation 10.1,
    return $\mu^{(t+1)}$
else
    return $\mu^{(t)}$
end if
```

ance of the target distribution. However, such an approach would either require significant hand-tuning or noisy estimations of the sample covariance matrix during an initial run of the sampler. For that reason we employ a different approach to exploit the covariance structure of the target distribution and preserve the hand-tuning-free nature of the algorithm.

Ensemble of walkers

Following the example of Goodman & Weare (2010) we define an ensemble of parallel chains, called walkers. In our case though, each walker is an individual slice sampler. The sampling proceeds by moving one walker at a time by slice sampling along a direction defined by a subset of the rest of the walkers of the ensemble. As long as the aforementioned direction does not depend on the position of the current walker, the resulting algorithm preserves the detailed balance of the chain. Moreover, assuming that the distribution of the walkers resembles the correlated target distribution, the chosen direction will prefer directions of correlated parameters.

We define an ensemble of N parallel walkers as the collection $S = \{\mathbf{X_1}, \dots, \mathbf{X_N}\}$. The position of each individual walker $\mathbf{X_k}$ is a vector $\mathbf{X_k} \in \mathbb{R}^D$ and therefore we can think of the ensemble S as being in $\mathbb{R}^{ND}$. Assuming that each walker is drawn independently from the target distribution with density p, then the target distribution for the ensemble would be the product

$$P(\mathbf{X_1}, \dots, \mathbf{X_N}) = \prod_{k=1}^{N} p(\mathbf{X_k}) \,. \tag{10.2}$$

The Markov chain of the ensemble would preserve the product density of equation 10.2 without the individual walker trajectories being Markov. Indeed, the position of $\mathbf{X_k}$ at iteration $t+1$ can depend on $\mathbf{X_j}$ at iteration t with $j \neq k$.

Given the walker $\mathbf{X_k}$ that is to be updated there are arbitrary many ways to define a direction vector from the complementary ensemble $S_{[k]} = \{\mathbf{X_j}, \forall j \neq k\}$. Here we will discuss a few of them. Following the convention in the ensemble MCMC literature we call those recipes of defining direction vectors, *moves*. Although the use of the ensemble might seem equivalent to that of a sample covariance matrix in the Adaptive Metropolis algorithm (Haario et al., 2001) the first has a higher information content as it encodes both linear and non-linear correlations. Indeed, having an ensemble of walkers allows for arbitrary many policies for choosing the appropriate directions along which the walkers move in parameter space. As we will shortly see, one of the choices (i.e. the Gaussian move, introduced later in this Section) is indeed the slice sampling analogue of a covariance matrix. However, other choices (i.e. Differential move or Global move) can take advantage of the non-Gaussian nature of the ensemble distribution and thus propose more informative moves. As will be discussed later in this section, those advanced moves make no assumption of Gaussianity for the target distribution. Furthermore, as we will show in the last part of this section, the ensemble can also be easily parallelised.

Algorithm 16 Function to return a differential move direction vector.

```
function DifferentialMove($k$, $\mu$, $S$)
Draw two walkers $\mathbf{X_l}$, and $\mathbf{X_m}$ uniformly and without replacement from the complementary ensemble $S$,
Compute direction vector $\boldsymbol{\eta}_k$ using Equation 10.6,
return $\boldsymbol{\eta}_k$
```

Affine transformations and invariance

Affine invariance is a property of certain MCMC samplers first introduced in the MCMC literature by Goodman & Weare (2010). An MCMC algorithm is said to be affine invariant if its performance is invariant under the bijective mapping $g : \mathbb{R}^D \longrightarrow \mathbb{R}^D$ of the form $\mathrm{Y} = A\mathrm{X} + b$ where $A \in \mathbb{R}^{D\times D}$ is a matrix and $b \in \mathbb{R}^D$ is a vector. Linear transformations of this form are called affine transformations and describe rotations, rescaling along specific axes as well as translations in parameter space. Assuming that X has the probability density $p(\mathrm{X})$, then $\mathrm{Y} = A\mathrm{X} + b$ has the probability density

$$p_{A,b}(\mathrm{Y}) = p(A\mathrm{X} + b) \propto p(\mathrm{X})\,. \tag{10.3}$$

Given a density p as well as an MCMC transition operator $\mathcal{T}$ such that $\mathrm{X}(t+1) = \mathcal{T}\left(\mathrm{X}(t); p\right)$ for any iteration t we call the operator $\mathcal{T}$ affine invariant if

$$\mathcal{T}\left(A\mathrm{X} + b; p_{A,b}\right) = A\,\mathcal{T}\left(\mathrm{X}; p\right) + b \tag{10.4}$$

for $\forall A \in \mathbb{R}^{D\times D}$ and $\forall b \in \mathbb{R}^D$. In case of an ensemble of walkers we define an affine transformation from $\mathbb{R}^{ND}$ to $\mathbb{R}^{ND}$ as

$$S = \{\mathrm{X}_1, \dots, \mathrm{X}_\mathrm{N}\} \xrightarrow{A,b} \{A\mathrm{X}_1 + b, \dots, A\mathrm{X}_\mathrm{N} + b\}\,. \tag{10.5}$$

The property of affine invariance is of paramount importance for the development of efficient MCMC methods. As we have discussed already, proposing samples more frequently along certain directions can accelerate sampling by moving further away in parameter space. Given that most realistic applications are highly skewed or anisotropic and are characterised by some degree of correlation between their parameters, affine invariant methods are an obvious choice of a tool that can be used in order to achieve high levels of efficiency.

Differential move

The differential direction choice works by moving the walker $\mathbf{X}_k$ based on two randomly chosen walkers $\mathbf{X}_l$ and $\mathbf{X}_m$ of the complementary ensemble $S_{[k]} = \{\mathbf{X}_\mathrm{j}, \forall j \neq k\}$ (Gilks, Roberts, et al., 1994), see Figure 10.2 for a graphical explanation. In particular, we move the walker $\mathbf{X}_k$ by slice sampling along the vector $\boldsymbol{\eta}_k$ defined by the difference between the walkers $\mathbf{X}_l$ and $\mathbf{X}_m$. It is important to notice here that the vector $\boldsymbol{\eta}_k$ is not a unit vector and thus carries information about both the length scale and the optimal direction of movement. It will also prove to be more intuitive to include the initial length scale μ in the definition of the direction vector in the following way:

$$\boldsymbol{\eta}_k = \mu\left(\mathbf{X}_l - \mathbf{X}_m\right). \tag{10.6}$$

The pseudocode for a function that, given the value of μ and the complementary ensemble S, returns a differential direction vector $\boldsymbol{\eta}_k$ is shown in Algorithm 16. Furthermore, the Differential move is clearly affine invariant. Assuming that the distribution of the ensemble of walkers follows the target distribution and the latter is highly elongated or stretched along a certain direction then the proposed direction given by equation 10.6 will share the same directional asymmetry.

Gaussian move

The direction vector $\boldsymbol{\eta}_k$ can also be drawn from a normal distribution with the zero mean and the covariance matrix equal to the sample covariance of the complementary ensemble $S_{[k]}$,

$$\mathbf{C}_S = \frac{1}{|S|}\sum_{j\in S}\left(\mathbf{X}_j - \bar{\mathbf{X}}_S\right)\left(\mathbf{X}_j - \bar{\mathbf{X}}_S\right)^t. \tag{10.7}$$

We chose to include the initial length scale μ in this definition as well:

$$\frac{\boldsymbol{\eta}_k}{2\mu} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{C}_S\right). \tag{10.8}$$

The factor of 2 is used so that the magnitude of the direction vectors are consistent with those sampled using the differential direction choice in the case of Gaussian-distributed walkers.

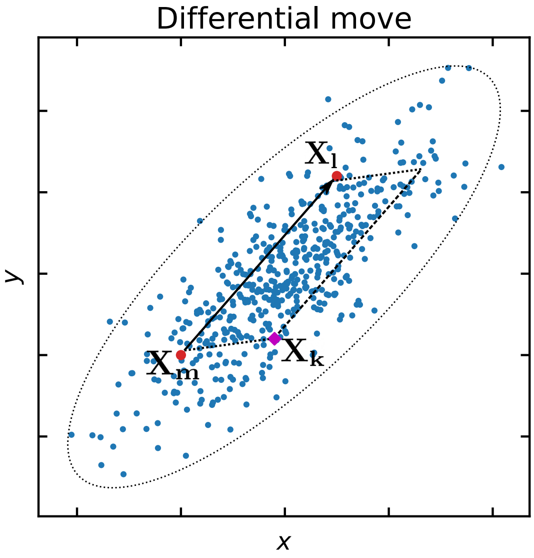

Figure 10.2: The plot shows the differential direction move. Two walkers (red) are uniformly sampled from the complementary ensemble (blue). Their positions define the direction vector (solid black). The selected walker (magenta) then moves by Slice Sampling along the parallel direction (dashed black).

The pseudocode for a function that, given the value of μ and the complementary ensemble S, returns a Gaussian direction vector $\boldsymbol{\eta}_k$ is shown in Algorithm 17. See Figure 10.3 for a graphical explanation of the method. Moreover, just like the Differential move, the Gaussian move is also affine invariant. In the limit in which the number of walkers is very large and the target distribution is normal, the first reduces to the second. Alternatively, assuming that the distribution of walkers follows the target distribution then the covariance matrix of the ensemble would be the same as that of independently drawn samples from the target density. Therefore any anisotropy characterising the target density would also be present in the distribution of proposed directions given by equation 10.8.

Algorithm 17 Function to return a Gaussian Move direction vector.

```
function GaussianMove($k$, $\mu$, $S$)
Estimate sample covariance $\mathbf{C}_S$ of the walkers in the complementary ensemble $S$ using Equation 10.7,
Sample $\boldsymbol{\eta}_k/(2\mu) \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_S)$,
return $\boldsymbol{\eta}_k$
```

Global move

ESS and its variations described so far (i.e. differential move, Gaussian move) have as much difficulty traversing the low probability regions between mod-

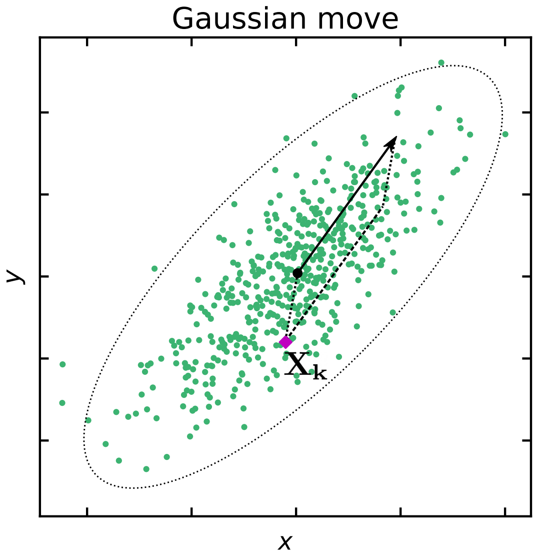

Figure 10.3: The plot shows the Gaussian direction move. A direction vector (solid black) is sampled from the Gaussian-approximated distribution of the walkers of the complementary ensemble (green). The selected walker (magenta) then moves by Slice Sampling along the parallel direction (dashed black).

es/peaks in multimodal distributions as most local MCMC methods (e.g. Metropolis, Hamiltonian Monte Carlo, Slice Sampling, etc.). Indeed, multimodal distributions are often the most challenging cases to sample from. Fortunately, Ensemble Slice Sampling's flexibility allows to construct advanced moves which are specifically designed to handle multimodal cases even in moderate to high dimensional parameter spaces. The *global move* is such an example.

We first fit a *Gaussian Mixture* to the distribution of the walkers of the complementary ensemble $S_{[k]}$ using *Variational Inference*. To avoid defining the number of components of the Gaussian Mixture we use a *Dirichlet process* as the prior distribution for the Gaussian Mixture weights[1] (Gorur & Rasmussen, 2010). The exact details of the construction of the Dirchlet process Gaussian mixture (DPGM) are beyond the scope of this work and we direct the reader to Gorur & Rasmussen (2010) and Bishop (2006) for more details. One of the major benefits of fitting the DPGM using variational inference compared to the expectation–maximisation (EM) algorithm (Dempster et al., 1977) that is often used is the improved stability. In particular, the use of priors in the variational Bayesian treatment guarantees that Gaussian components do not collapse into specific data points. This regularisation due to the priors leads to component covariance matrices that do not diverge even when the number of data points (i.e. walkers in our case) in a component is lower than the number of dimensions. In our case, this means that even if the number of walkers located in a mode of the target distribution is small DPGM would still identify that mode correctly. In such cases, the covariance of the component that corresponds to that mode would be over–estimated. This however does not affect the performance of the Global move as the latter does not rely on exact estimates of the component covariance matrices.[2]

---

[1] To this end we use the Scikit-Learn implementation of the Dirichlet process Gaussian mixture.

In practice, we recommend using more than the minimum number of walkers in cases of multimodal distributions (e.g. at least two times as many in bimodal cases). We found that the computational overhead introduced by the variational fitting of the DPGM is negligible compared to the computational cost of the evaluation of the model and posterior distribution in common problems in physics, astrophysics and cosmology. Indeed the cost is comparable, and only a few times higher than the Differential or Gaussian move. The reason for that is the relatively small number of walkers (i.e. $\mathcal{O}(10-10^3)$) that simplifies the fitting procedure.

Once fitting is done, we have a list of the means and covariance matrices of the components of the Gaussian Mixture. As the ensemble of walkers traces the structure of the target distribution, we can use the knowledge of the means and covariance matrices of the Gaussian Mixture to construct efficient direction vectors. Ideally, we prefer direction vectors that connect different modes. This way, the walkers will be encouraged to move along those directions that would otherwise be very unlikely to be chosen.

We uniformly select two walkers of the complementary ensemble and identify the Gaussian components to which they belong, say i and j. There are two distinct cases and we will treat them as such. In case A, $i = j$, meaning that the selected walkers originate from the same component. In case B, $i \neq j$, meaning that the two walkers belong to different components and thus probably different peaks of the target distribution.

As we will show next, only in case B, we can define a direction vector that favors mode-jumping behaviour. In case A, we can sample a direction vector from the Gaussian component that the two select walkers belong to[3]:

$$\frac{\boldsymbol{\eta}_k}{2\mu} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}_{i=j}\right), \tag{10.9}$$

where $\boldsymbol{C}_{i=j}$ is the covariance matrix of the i_{th} (or equivalently j_{th}) component. Just as in the Gaussian move, the mean of the proposal distribution is zero so that we can interpret $\boldsymbol{\eta}$ as a direction vector.

In case B, where the two selected walkers belong to different components, $i \neq j$, we will follow a different procedure to facilitate long jumps in parameter space. We will sample two vectors, one from each component:

$$\boldsymbol{\eta}_{k,n} \sim \mathcal{N}\left(\boldsymbol{\mu}_n, \gamma \boldsymbol{C}_n\right), \tag{10.10}$$

---

[2] Indeed the covariance matrix of a component only enters through equation 10.10 but then it is re–scaled by the factor γ.

[3] In practice we use uniformly sample two walkers from the list of walkers that DPGM identified in that mode. This step removes any dependency on covariance matrix estimates.

for $n = i$ or $n = j$. Here, $\boldsymbol{\mu}_n$ is the mean of the nth component and $\boldsymbol{C}_n$ is its covariance matrix. In practice, we also re-scale the covariance by a factor of $\gamma = 0.001$, which results in direction vectors with lower variance in their orientation. $\gamma < 1$ ensures that the chosen direction vector is close to the vector connecting the two peaks of the distribution. Finally, the direction vector will be defined as:

$$\boldsymbol{\eta}_k = 2\left(\boldsymbol{\eta}_{k,i} - \boldsymbol{\eta}_{k,j}\right) . \tag{10.11}$$

The factor of 2 here is chosen to better facilitate mode-jumping. There is also no factor of μ in the aforementioned expression since in this case there is no need for the scale factor to be tuned.

The pseudocode for a function that, given the complementary ensemble S, returns a Global direction vector $\boldsymbol{\eta}_k$ is shown in Algorithm 18. See Figure 10.4 for a graphical explanation of the method. It should be noted that for the global move to work at least one walker needs to be present on each well separated mode.

Algorithm 18 Function to return a global move direction vector.

```
function GlobalMove($k$, $\mu$, $S$)
Fit Dirichlet process Gaussian mixture (DPGM) to the complementary ensemble $S_{[k]}$,
If $N$ is the number of components of the DPGM then select two components $i, j$ uniformly such that $i \neq j$,
if $i = j$ then
    Sample $\boldsymbol{\eta}_k/(2\mu) \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{C}_{i=j}\right)$,
else
    Sample $\boldsymbol{\eta}_{k,n} \sim \mathcal{N}\left(\boldsymbol{\mu}_n, \gamma \boldsymbol{C}_n\right)$ for $n = i, j$,
    Compute direction vector $\boldsymbol{\eta}_k$ using Equation 10.11,
end if
return $\boldsymbol{\eta}_k$
```

Here we introduced three general and distinct moves that can be used in a broad range of cases. In general, the global move requires a higher number of walkers than the differential or Gaussian move in order to perform well. We found that the differential and Gaussian moves are good choices for most target distributions whereas the global move is only necessary in highly dimensional and multimodal cases. One can use the information in the complementary ensemble to construct more moves tailor-made for specific problems. Such additional moves might include Kernel Density Estimation or Clustering methods and as long as the information used comes from the complementary ensemble (and not from the walker that would be updated) the detailed balance is preserved.

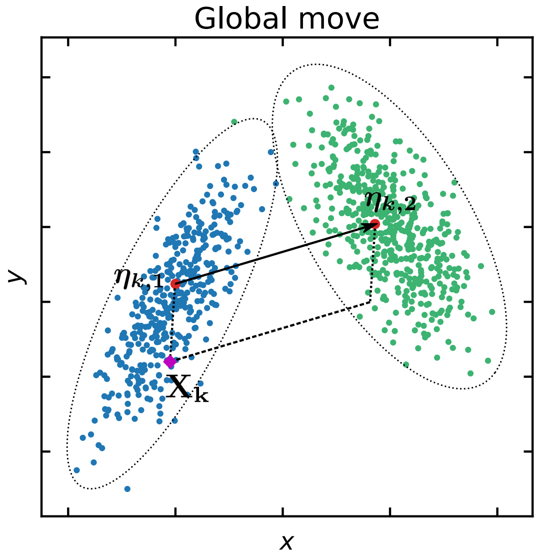

Figure 10.4: The plot shows the global direction move assuming that the uniformly selected pair of walkers of the complementary ensemble belongs to different components (blue and green). A position (red) is sampled from each component (using the re-scaled by γ covariance matrix). Those two points (red) define the direction vector (black) connecting the two modes (blue and green). The selected walker (magenta) then moves by slice sampling along the parallel direction (dashed).

Parallelizing the ensemble

Instead of evolving the ensemble by moving each walker in turn we can do this in parallel. A naive implementation of this would result in a subtle violation of detailed balance. We can avoid this by splitting the ensemble into two sets of walkers (Foreman-Mackey, Hogg, et al., 2013) of $n_{\mathrm{Walkers}}/2$ each. We can now update the positions of all the walkers in the one set in parallel along directions defined by the walkers of the other set (the complementary ensemble). Then we can perform the same procedure for the other set. In accordance with equation 10.2, the stationary distribution of the split ensemble would be

$$P(\mathrm{X_1},\dots,\mathrm{X_N}) = \prod_{k=1}^{N/2} p(\mathrm{X_k}) \prod_{k=1+N/2}^{N} p(\mathrm{X_k})\,. \tag{10.12}$$

The method generates samples from the target distribution by simulating a Markov chain which leaves this product distribution invariant. The transition operator $\mathcal{T}_1$ that updates the walkers of the first set (i.e. $k = 1,\dots,N/2$) uses the walkers of the complementary ensemble (i.e. $k = 1 + N/2,\dots,N$) and vice versa for the transition operator $\mathcal{T}_2$ that acts on the second set. In the context of ESS the aforementioned transition operators correspond to a single iteration of Algorithm 19 coupled with one of the moves (e.g. Differential move).

It follows from the ensemble splitting technique that the maximum number of CPUs used without any of them being idle is equal to the total number of walkers updated concurrently, that is $n_{\text{Walkers}}/2$. We will also verify this empirically in Section 10.4. Of course, this does not mean that if there are more CPUs available they cannot be used as we can always increase the size of the ensemble to match the available CPUs.

Combining this technique with the stochastic approximation solution of Subsection 10.3.1 and the choices (moves) of direction and ensemble-splitting technique of this subsection leads to the Ensemble Slice Sampling method of Algorithm 19[4]. Of course, another move (e.g. Gaussian, global) can be used instead of the differential move in Algorithm 19. Finally, the minimum number of walkers used should be twice the number of parameters. Using fewer walkers than that could lead to erroneous sampling from a lower dimensional parameter space (Ter Braak, 2006).

In general, parallelizing a slice sampler is not trivial (e.g. as it is for Metropolis) because each update requires an unknown number of probability density evaluations. However, because of the affine invariance (i.e. performance unaffected by linear correlations) induced by the existence of the ensemble, all iterations require on average the same number of probability density evaluations (i.e. usually 5 if the stochastic approximation for the length scale μ is used). Therefore, the parallelization of Ensemble Slice Sampling is very effective in practice. Furthermore, the benefit of having parallel walkers instead of parallel independent chains (e.g. such as in Metropolis sampling) is clear, the walkers share information about the covariance structure of the distribution thus accelerating mixing.

10.4 EMPIRICAL EVALUATION

To empirically evaluate the sampling performance of the Ensemble Slice Sampling algorithm we perform a series of tests. In particular, we compare its ability to sample from two demanding target distributions, namely the *autoregressive process of order 1* and the *correlated funnel*, against the Metropolis and Standard Slice Sampling algorithms. The Metropolis' proposal scale was tuned to achieve the optimal acceptance rate, whereas the initial length scale of Standard Slice Sampling was tuned using the stochastic scheme of Algorithm 15. Ensemble Slice Sampling significantly outperforms both of them. These tests help establish the characteristics and advantages of Ensemble Slice Sampling. Since our objective was to develop a gradient-free black-box method we then proceed to compare Ensemble Slice Sampling with a list of gradient-free ensemble methods such as *Affine Invariant Ensemble Sam-*

---

[4] Perhaps a small detail, but we have included the length scale in the definition of the direction vector η and therefore it does not appear in the definition of the (L, R) interval.

Algorithm 19 Single Iteration t of Ensemble Slice Sampling.

```
Given $t, f, \mu^{(t)}, S_{[0]}, S_{[1]}, M^{\text{adapt}}$:
Initialise $N_e^{(t)} = 0$ and $N_c^{(t)} = 0$,
for $i = 0, 1$ do
    for $k = 1, ..., N/2$ do
        $k \longleftarrow k + iN/2$
        Compute direction vector $\boldsymbol{\eta}_k \longleftarrow$ DifferentialMove($k$, $\mu^{(t)}$, $S_{[i]}$)
        Sample $Y \sim$ Uniform($0, f(\mathbf{X_k}^{(t)})$)
        Sample $U \sim$ Uniform($0, 1$)
        Set $L \longleftarrow -U$, and $R \longleftarrow L + 1$
        while $Y < f(L)$ do
            $L \longleftarrow L - 1$
            $N_e^{(t)} \longleftarrow N_e^{(t)} + 1$
        end while
        while $Y < f(R)$ do
            $R \longleftarrow R + 1$
            $N_e^{(t)} \longleftarrow N_e^{(t)} + 1$
        end while
        while True do
            Sample $X' \sim$ Uniform($L, R$)
            Set $Y' \longleftarrow f(X'\boldsymbol{\eta}_k + \mathbf{X_k}^{(t)})$
            if $Y < Y'$ then
                break
            end if
            if $X' < 0$ then
                $L \longleftarrow X'$
                $N_c^{(t)} \longleftarrow N_c^{(t)} + 1$
            else
                $R \longleftarrow X'$
                $N_c^{(t)} \longleftarrow N_c^{(t)} + 1$
            end if
        end while
        Set $\mathbf{X_k}^{(t+1)} \longleftarrow X'\boldsymbol{\eta}_k + \mathbf{X_k}^{(t)}$
    end for
end for
$\mu^{(t+1)} \longleftarrow$ TuneLengthScale($t$, $\mu^{(t)}$, $N_e^{(t)}$, $N_c^{(t)}$, $M^{adapt}$),
```

pling (AIES), *Differential Evolution Markov Chain* (DEMC) and *Kernel Density Estimate Metropolis* (KM) on a variety of challenging target distributions. Moreover, we are also interested in assessing the convergence rate of the length scale μ during the first iterations as well as the parallel scaling of the method in the presence of multiple CPUs. Unless otherwise specified we use the differential move for the tests. Unlike ESS that has an acceptance rate of 1, AIES's and DEMC's acceptance rate is related to the number of walkers. For that reason, and for the sake of a fair comparison, we made sure the selected number of walkers in all examples would yield the optimal acceptance rate for AIES and DEMC. As we will discuss further in Section 10.5 it makes sense to increase the number of walkers in cases of multimodal distributions or strong non-linear correlations. In general though, we recommend using the minimum number of walkers (i.e. twice the number of dimensions) as the default choice and increase it only if it is required by a specific application. For more rules and heuristics about the initialisation and number of walkers we direct the interested reader to Section 10.5.

10.4.1 Performance tests

Autoregressive process of order 1

In order to investigate the performance of ESS. in high dimensional and correlated scenarios we chose a highly correlated Gaussian as the target distribution. More specifically, the target density is a discrete-time *autoregressive process of order 1*, also known as AR(1). This particular target density is ideally suited for benchmarking MCMC algorithms since the posterior density in many scientific studies often approximates a correlated Gaussian. Apart from that, the AR(1) is commonly used as a prior for time-series analysis.

The AR(1) distribution of a random vector $\boldsymbol{X} = (X_1, ..., X_N)$ is defined recursively as follows:

$$\begin{aligned} X_1 &\sim \mathcal{N}(0,1)\,, \\ X_2|X_1 &\sim \mathcal{N}(\alpha X_1, \beta^2)\,, \\ &\vdots \\ X_N|X_{N-1} &\sim \mathcal{N}(\alpha X_{N-1}, \beta^2)\,, \end{aligned} \tag{10.13}$$

where the parameter α controls the degree of correlation between parameters and we chose it to be $\alpha = 0.95$. We set $\beta = \sqrt{1-\alpha^2}$ so that the marginal distribution of all parameters is $\mathcal{N}(0,1)$. We also set the number of dimensions to $N = 50$.

For each method, we measured the mean *integrated autocorrelation time* (IAT), and the number of effective samples per evaluation of the probability density function, also termed *efficiency* (see Appendix 10.7 for details). For this test we ran the samplers for 10^7 iterations. In this example we used the minimum number of walkers (i.e. 100 walkers) for ESS and the equivalent number of probability evaluations for Metropolis and Slice Sampling with each walker initialised at a position sampled from the distribution $\mathcal{N}(0,1)$. The results are presented in Table 2. The chain produced by Ensemble Slice Sampling has a significantly shorter IAT (20 – 40 times) compared to either of

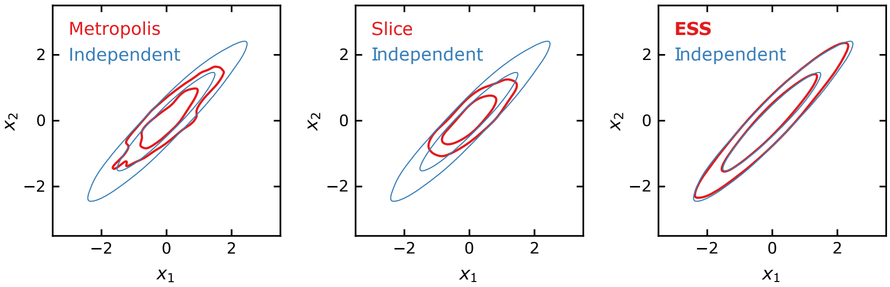

Figure 10.5: The plots compare the 1-sigma and 2-sigma contours generated by the optimised random-walk Metropolis (left), Standard Slice (centre) and Ensemble Slice Sampling (right) methods to those obtained by Independent Sampling (blue) for the AR(1) distribution. All samplers used the same number of probability density evaluations, 3×10^5. Only the first two dimensions are shown here.

Table 2: The table shows a comparison of the optimally tuned Metropolis, Standard Slice, and Ensemble Slice Sampling with the differential move (ESS-D) and the Gaussian move (ESS-G) respectively in terms of the integrated autocorrelation time (IAT) and the number of effective samples per evaluation of the probability density (efficiency) multiplied by 10^4. These metrics are formally defined in Appendix 10.7. The target distributions are the 50−dimensional autoregressive process of order 1 and the 25−dimensional correlated funnel distribution. The total number of iterations was set to 10^7.

	Metropolis	Slice	**ESS-D**	**ESS-G**
Autoregressive process of order 1				
IAT	4341	2075	**111**	**107**
efficiency	2.3	1.0	**17.5**	**17.8**
Correlated funnel distribution				
IAT	-	3905	**129**	**141**
efficiency	-	0.5	**15.3**	**14.0**

the other two methods. Furthermore, Ensemble Slice Sampling, with either Differential or Gaussian move, generates an order of magnitude greater number of independent samples per evaluation of the probability density. In this example the Differential and Gaussian moves have achieved almost identical IAT values and efficiencies.

To assess the mixing rate of Ensemble Slice Sampling, we set the maximum number of probability density evaluations to 3×10^5 and show the results in Figure 10.5. We compare the results of Ensemble Slice Sampling with those obtained via the optimally tuned Metropolis and Standard Slice Sampling methods. Ensemble Slice Sampling significantly outperforms both of them, being the only one with a chain resembling the target distribution in the chosen number of probability evaluations.

Correlated funnel

The second test involves a more challenging distribution, namely the correlated funnel distribution adapted from Neal (2003). The funnel, tornado like, structure is common in Bayesian hierarchical models and possesses characteristics that render it a particularly difficult case. The main difficulty originates from the fact that there is a region of the parameter space where the volume of the region is low but the probability density is high, and another region where the opposite holds.

Suppose we want to sample an N−dimensional vector $\boldsymbol{X} = (X_1, ..., X_N)$ from the correlated funnel distribution. The marginal distribution of X_1 is Gaussian with mean zero and unit variance. Conditional on a value of X_1, the vector $\boldsymbol{X}_{2-N} = (X_2, ..., X_N)$ is drawn from a Gaussian with mean zero and a covariance matrix in which the diagonal elements are $\exp(X_1)$, and the non-diagonal equal to $\gamma \exp(X_1)$. If $\gamma = 0$, the parameters X_2 to X_N conditional on X_1 are independent and the funnel distribution resembles the one proposed by Neal (2003). The value of γ controls the degree of correlation between those parameters. When $\gamma = 0$ the parameters are uncorrelated. For the following test we chose this to be $\gamma = 0.95$. We set the number of parameters N to 25.

Using 10^7 iterations, we estimated the IAT and the efficiency of the algorithms for this distribution as shown in Table 2. Just like in the AR(1) case we used the minimum number (i.e. 50) of walkers for ESS with each walker initialised at a position sampled from the distribution $\mathcal{N}(0, 1)$. Since the optimally-tuned Metropolis fails to sample from this particular distribution, we do not quote any results. The Metropolis sampler is unable to successfully explore the region of parameter space with negative X_1 values. The presence of strong correlations renders the Ensemble Slice Sampler 30 times more efficient than the Standard Slice Sampling algorithm on this particular example. In this example, the Differential move outperforms the Gaussian move in terms of efficiency, albeit by a small margin. In general, we expect the former to be more flexible than the latter since it makes no assumption about the Gaussianity of the target-distribution and recommend it as the default configuration of the algorithm.

To assess the mixing rate of the algorithm on this demanding case, we set the maximum number of evaluations of the probability density function to 3×10^5. As shown in Figure 10.6, the Ensemble Slice Sampling is the only algorithm out of the three whose outcome closely resembles the target distribution. The results of Metropolis were incorrect for both, the limited run with 3×10^5 iterations and the long run with 10^7 iterations. In particular, the chain produced using the Metropolis method resemble a converged chain

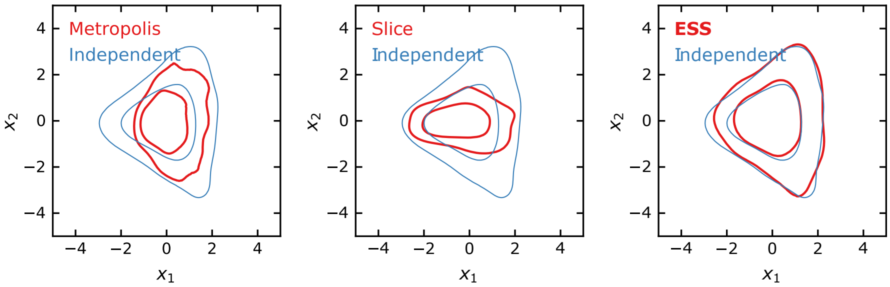

Figure 10.6: The plots compare the 1-sigma and 2-sigma contours generated by the optimised random-walk Metropolis (left), Standard Slice (centre) and Ensemble Slice Sampling (right) methods to those obtained by Independent Sampling (blue) for the correlated funnel distribution. All samplers used the same number of probability density evaluations, 3×10^5. Only the first two dimensions are shown here.

but in fact it is biased in favour of positive values of x_1. The problem arises because of the vanishing low probability of accepting a point with highly negative value of x_1. This indicates the inability of Metropolis to handle this challenging case. For a more detailed discussion of this problem we direct the reader to Section 8 of Neal, 2003. In general, the correlated funnel is a clear example of a distribution in which a single Metropolis proposal scale is not sufficient for all the sampled regions of parameter space. The locally adaptive nature of ESS solves this issue.

10.4.2 Comparison to other ensemble methods

So far we have demonstrated Ensemble Slice Sampling's performance in simple, yet challenging, target distributions. The tests performed so far demonstrate ESS's capacity to sample efficiently from highly correlated distributions compared with standard methods such as Metropolis and Slice Sampling. Although the use of Metropolis and Slice Sampling is common, these methods are not considered to be state-of-the-art. For this reason, we will now compare ESS with state-of-the-art gradient-free ensemble MCMC methods.

By far, the two most popular choices[5] of gradient-free ensemble methods are the Affine-Invariant Ensemble Sampling (AIES) (Goodman & Weare, 2010) method and the Differential Evolution Monte Carlo (DEMC) (Ter Braak, 2006) algorithm supplemented with a Snooker update (Ter Braak & Vrugt, 2008).

---

[5] For instance, in the fields of Astrophysics and Cosmology where most models are not differentiable and gradient methods (e.g. Hamiltonian Monte Carlo or NUTS) are not applicable the default choice is the Affine-Invariant Ensemble Sampler (AIES) (Goodman & Weare, 2010) as implemented in emcee.

In cases of strongly multimodal target distributions we will also test our method against Sequential Monte Carlo[6] (SMC) (Del Moral et al., 2006; Liu & R. Chen, 1998) and Kernel Density Estimate Metropolis (KM) (B. Farr & W. M. Farr, 2015) which are particle methods specifically designed to handle strongly multimodal densities.

Ring distribution

Although, all three of the compared methods (i.e. ESS, AIES, DEMC) are affine invariant and thus unaffected by linear correlations, they do however differ significantly in the way they handle non-linear correlations. In particular, only Ensemble Slice Sampling (ESS) is locally adaptive because of its stepping-out procedure and therefore able to handle non-linear correlations efficiently.

To illustrate ESS's performance in a case of strong non-linear correlations we will use the 16−dimensional ring distribution defined by:

$$
\begin{aligned}
\ln \mathcal{L} = & -\left[\frac{(x_n^2 + x_1^2 - a)^2}{b}\right]^2 \\
& - \sum_{i=1}^{n-1} \left[\frac{(x_i^2 + x_{i+1}^2 - a)^2}{b}\right]^2 ,
\end{aligned} \tag{10.14}
$$

where $a = 2$, $b = 1$ and $n = 16$ is the total number of parameters. We also set the number of walkers to be 64 and run the samplers for 10^7 steps discarding the first half of the chains. Here we followed the heuristics discussed at the beginning of this section and increased the number of walkers from the minimum of 2×16 to 4×16 due to the presence of strong non-linear correlations in order to achieve the optimal acceptance rate for AIES and DEMC. The number of iterations is large enough for all samplers to converge and provide accurate estimates of the autocorrelation time.

The results are shown in Table 3 and verify that ESS' performance is an order of magnitude better than that of the other methods.

Gaussian shells distribution

Another example that demonstrates ESS's performance in cases of non-linear correlations is the Gaussian Shells distribution defined as:

$$
\mathcal{L}(\Theta) = \mathrm{circ}(\Theta|\mathbf{c}_1, r_1, w_1) + \mathrm{circ}(\Theta|\mathbf{c}_2, r_2, w_2), \tag{10.15}
$$

where

$$
\mathrm{circ}(\Theta|\mathbf{c}, r, w) = \frac{1}{\sqrt{2\pi} w} \exp\left[-\frac{1}{2}\frac{(|\Theta - \mathbf{c}| - r)^2}{w^2}\right]. \tag{10.16}
$$

---

[6] As there are many different flavours of SMC, we decided to use the one implemented in PyMC3 which utilises importance sampling, simulated annealing and Metropolis sampling.

Table 3: The table shows a comparison of the Affine Invariant Ensemble Sampling (AIES), Differential Evolution Markov Chain (DEMC), and Ensemble Slice Sampling methods in terms of the integrated autocorrelation time (IAT) and the number of effective samples per evaluation of the probability density (efficiency) multiplied by 10^5. These metrics are formally defined in Appendix 10.7. The target distributions are the 16–dimensional ring distribution, the 10–dimensional Gaussian shells distribution and the 13–dimensional hierarchical Gaussian process regression distribution. In all cases the total number of iterations was set to 10^7. It should be noted that in the case of the Gaussian shells the global move was used instead of the differential move.

	AIES	DEMC	**ESS**
Ring distribution			
IAT	49470	91128	**1675**
efficiency	2.0	1.1	**12.2**
Gaussian shells distribution			
IAT	33046	2760	**89**
efficiency	3.0	36.0	**731.0**
Hierarchical Gaussian process regression			
IAT	55236	30990	**547**
efficiency	1.8	3.2	**38.0**

We choose the centres, $\mathbf{c}_1$ and $\mathbf{c}_2$ to be -3.5 and 3.5 in the first dimension respectively and zero in all others. We take the radius to be $r = 2.0$ and the width $w = 0.1$. In two dimensions, the aforementioned distribution corresponds to two equal-sized Gaussian Shells. In higher dimensions the geometry of the distribution becomes more complicated and the density becomes multimodal.

For our test, we set the number of dimensions to 10 and the number of walkers to 40 due to the existence of two modes. Since this target distribution exhibits some mild multimodal behaviour we opt for the global move instead of the default differential move although the latter also performs acceptably in this case. The total number of iterations was set to 10^7 and the first half of the chains was discarded. The results are presented in Table 3. ESS's autocorrelation time is $2-3$ orders of magnitude lower than that of the other methods and the efficiency is higher by $1-2$ orders of magnitude respectively.

Hierarchical Gaussian process regression

To illustrate ESS's performance in a real-world example we will use a modelling problem concerning the concentration of CO_2 in the atmosphere adapted from Chapter 5 of Rasmussen, 2003. The data consist of monthly measurements of the mean CO_2 concentration in the atmosphere measured at the *Mauna Loa Observatory* (Keeling & Whorf, 2004) in *Hawaii* since 1958. Our goal is to model the concentration of CO_2 as a function of time. To this end, we will employ a *hierarchical Gaussian process* model with a composite covariance function designed to take care of the properties of the data. In particular, the covariance function (kernel) is the sum of following four distinct terms:

$$k_1(r) = \theta_1^2 \exp\left(-\frac{r^2}{2\theta_2}\right) , \tag{10.17}$$

where $r = x - x'$ that describes the smooth trend of the data,

$$k_2(r) = \theta_3^2 \exp\left[-\frac{r^2}{2\theta_4} - \theta_5 \sin^2\left(\frac{\pi r}{\theta_6}\right)\right] , \tag{10.18}$$

that describes the seasonal component,

$$k_3(r) = \theta_7^2 \left[1 + \frac{r^2}{2\theta_8\theta_9}\right]^{-\theta_8} , \tag{10.19}$$

which encodes medium-term irregularities, and finally:

$$k_4(r) = \theta_{10}^2 \exp\left(-\frac{r^2}{2\theta_{11}}\right) + \theta_{12}^2\delta_{ij} , \tag{10.20}$$

that describes the noise. We also fit the mean of the data, having in total 13 parameters to sample.

We sample this target distribution using 36 walkers for 10^7 iterations and we discard the first half of the chains. The number of walkers that was used corresponds to 1.5 times the minimum number. We found that this value results in the optimal acceptance rate for AIES and DEMC. For this example we use the differential move of ESS. The results are presented in Table 3. The integrated autocorrelation time of ESS is 2 orders of magnitude lower than that of the other methods and its efficiency is more than an order of magnitude higher. The performance is weakly sensitive to the choice of the number of walkers.

Bayesian object detection

Another real world example with many applications in the field of *astronomy* is *Bayesian object detection.* The following model adapted from Feroz & Mike P Hobson, 2008 can be used with a few adjustments to detect astronomical objects in telescope images often hidden in background noise.

We assume that the 2D circular objects present in the image are described by the Gaussian profile:

$$\mathbf{G}(x, y; \boldsymbol{\theta}) = A \exp\left[- \frac{(x - X)^2 + (y - Y)^2}{2R^2} \right], \tag{10.21}$$

where $\theta = (X, Y, A, R)$ are parameters that define the coordinate position, the amplitude and the size of the object, respectively. Then the data can be described as:

$$\mathbf{D} = \mathbf{N} + \sum_{i=1}^{n_{\mathrm{Obj}}} \mathbf{G}(\boldsymbol{\theta_i}), \tag{10.22}$$

where n_{Obj} is the number of objects in the image and $\mathbf{N}$ is an additive Gaussian noise term.

Assuming a 200 × 200 pixel-wide image, we can create a simulated dataset by sampling the coordinate positions (X, Y) of the objects from $\mathcal{U}(0, 200)$ and their amplitude A and size R from $\mathcal{U}(1, 2)$ and $\mathcal{U}(3, 7)$, respectively. We sample $n_{\mathrm{Obj}} = 8$ objects in total. Finally, we sample the noise $\mathbf{N}$ from $\mathcal{N}(0, 4)$. In practice we create a dataset of 100 such images and one such example is shown in Figure 10.7. Notice that the objects are hardly visible as they are obscured by the background noise, this makes the task of identifying those objects very challenging.

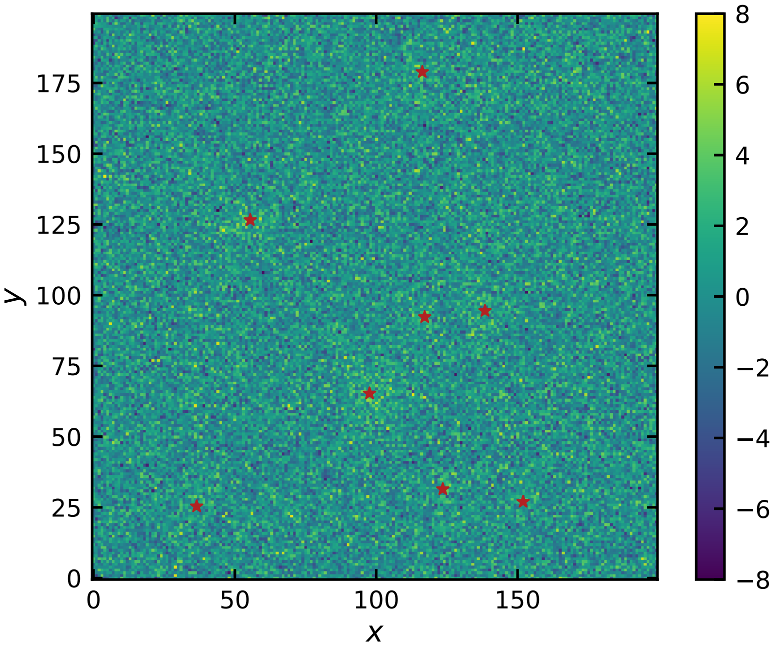

Figure 10.7: The plot shows a simulated image used in the Bayesian object detection exercise. There are 8 circular objects included here. As the objects are hardly visible due to the background noise their centres are marked with red stars.

Following the construction of the simulated dataset, the posterior probability density function is defined as:

$$P(\boldsymbol{\theta}|\mathbf{D}) \propto \exp\left\{\frac{[\mathbf{G}(\boldsymbol{\theta}) - \mathbf{D}]^2}{2\sigma^2}\right\} P(\boldsymbol{\theta}) \,, \tag{10.23}$$

where $\sigma = 2$ is the standard deviation of the $\mathbf{N}$ noise term. The prior $P(\boldsymbol{\theta})$ can be decomposed as the product of prior distributions of X, Y, A, and R. We used uniform priors for all of these parameters with limits $(0, 200)$ for X and Y, $(1, 2)$ for A, and $(2, 9)$ for R. It is important to mention here that the posterior does not include any prior information about the exact or maximum number of objects in the data. In that sense, the sampler is agnostic about the exact number, positions and characteristics (i.e. amplitude and size) of the objects that it seeks to detect.

We sampled the posterior distribution using 200 walkers (initialised from the prior distribution) for each image in our dataset (i.e. 100 images in total) using Ensemble Slice Sampling (ESS), Affine Invariant Ensemble Sampling (AIES), and Differential Evolution Markov Chain (DEMC). Although the posterior distribution is multimodal (i.e. 8 modes) we used the differential move since the number of dimensions is low and there is no reason to use more sophisticated moves like the global move. We used a large enough ensemble of walkers due to the potential presence of multiple modes so that all three samplers are able to resolve them.

We ran each sampler for 10^4 iterations in total and we discarded the first half of the chains. We found that, on average for the 100 images, ESS identifies correctly 7 out of 8 objects in the image, whereas AIES and DEMC identify 4 and 5, respectively.

In cases where the objects are well-separated ESS often identifies correctly 8 out of 8. Its accuracy falls to 7/8 in cases where two of the objects are very close to each other or overlap. In those cases ESS identifies the merged object as a single object. In this context, by identification of an object, we mean that at least one walker has sampled the posterior mode which corresponds to that object.

Gaussian Mixture

One strengths of ESS is its ability to sample from strongly multimodal distributions in high dimensions. To demonstrate this, we will utilise a Gaussian Mixture of two components centred at $-\mathbf{0.5}$ and $+\mathbf{0.5}$ with standard deviation of $\mathbf{0.1}$. We also put 1/3 of the probability mass in one mode and 2/3 in the other.

We first set this distribution at 10 dimensions and we sample this using 80 walkers for 10^5 steps. The distance between the two modes in this case is approximately 32 standard deviations. We then increase the number of dimensions to 50 and we sample it using 400 walkers for 10^5 iterations. In this case, the actual distance between the two modes is approximately 71 standard deviations. The total number of iterations was set to 10^7 for all methods but the SMC.

This problem consists of two, well separated, modes and thus requires using at least twice the minimum number of walkers (i.e. at least 40 for the 10−dimensional case and 200 for the 50−dimensional one). Although the aforementioned configuration was sufficient for ESS to provide accurate estimates, we opted instead for twice that number (i.e. 80 walkers for the 10−dimensional cases and 400 for the 50−dimensional one) in order to satisfy the requirements of the other samplers, mainly the Kernel Density Estimate Metropolis (KM), but also AIES and DEMC. For the Sequential Monte Carlo (SMC) sampler we used 2000 and 20000 independent chains for the low and high dimensional case respectively. The temperature ladder that interpolates between the prior and posterior distribution was chosen adaptively guaranteeing an effective sample size of 90% the physical size of the ensemble. Our implementation of SMC was based on that of `PyMC3` using an independent Metropolis mutation kernel.

The results for the 10−dimensional and 50−dimensional cases are plotted in Figures 10.8 and 10.9, respectively. In the 10−dimensional case, both ESS (differential and global move) and SMC managed to sample from the target whereas AIES, DEMC and KM failed to do so. In the 50−dimensional case, only the Ensemble Slice Sampling with the global move manages to sample correctly from this challenging target distribution. In practice ESS_G is able to handle similar cases in even higher number of dimensions and with more than 2 modes.

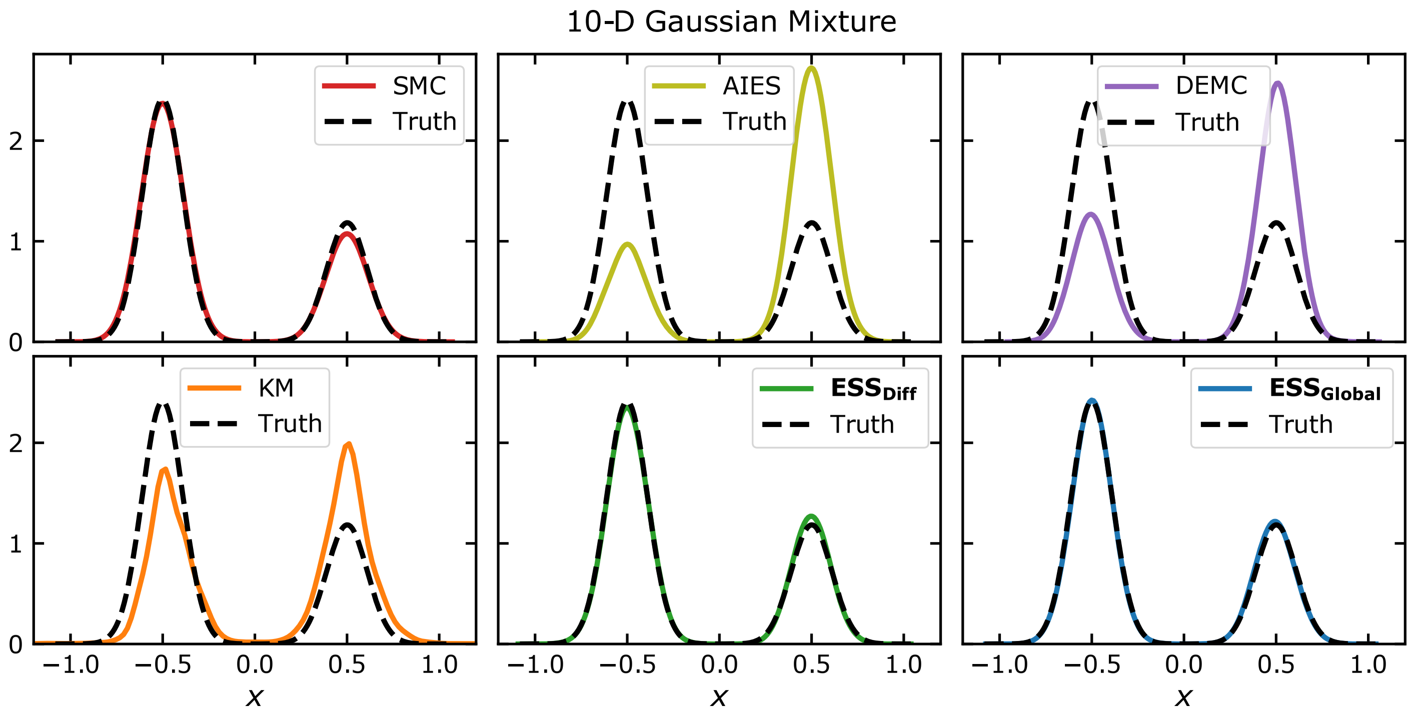

Figure 10.8: The plot compares the results of 6 samplers, namely Sequential Monte Carlo (SMC, red), Affine-Invariant Ensemble Sampling (AIES, yellow), Differential Evolution Markov Chain (DEMC, purple), Kernel Density Estimate Metropolis (KM, orange), Ensemble Slice Sampling using the differential move (ESS, green), and Ensemble Slice Sampling using the global move (ESS, blue). The target distribution is a 10−dimensional Gaussian Mixture. The figure shows the 1D marginal distribution for the first parameter of the 10.

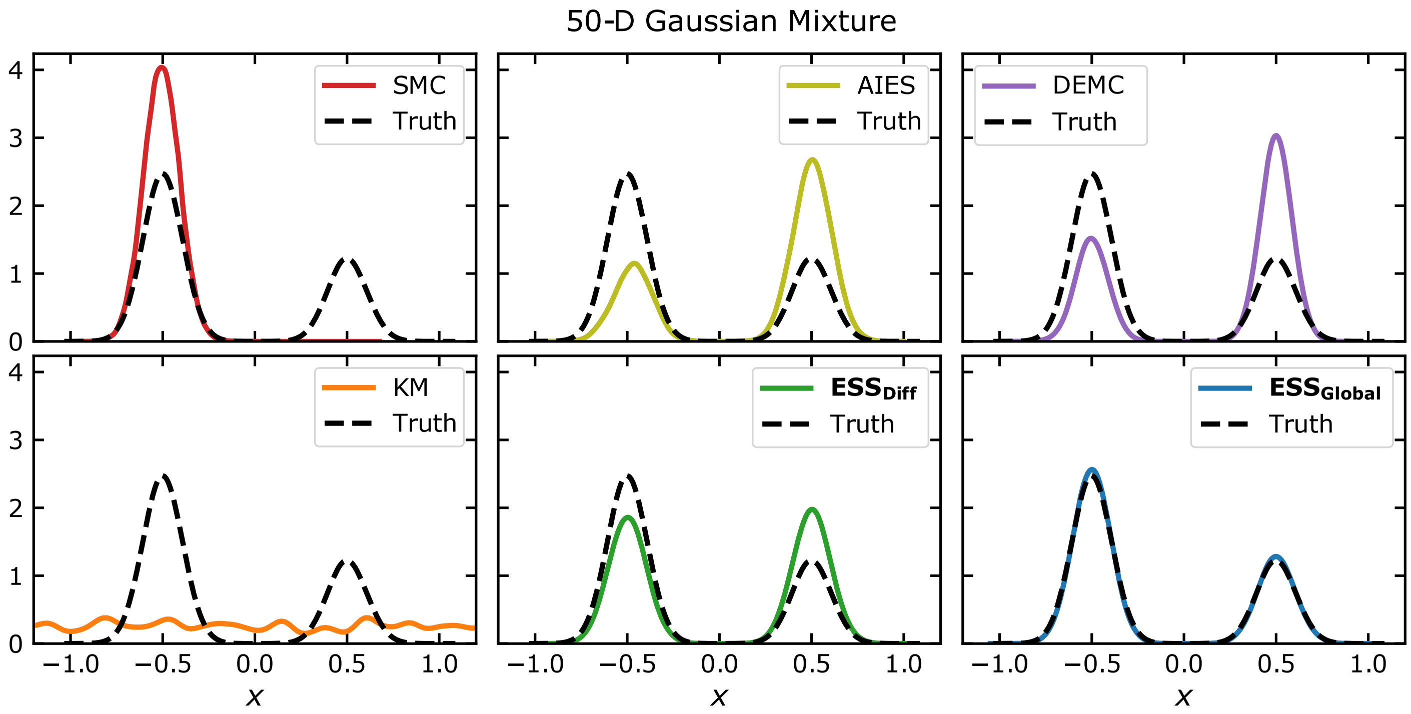

Figure 10.9: The plot compares the results of 6 samplers, namely Sequential Monte Carlo (SMC, red), Affine-Invariant Ensemble Sampling (AIES, yellow), Differential Evolution Markov Chain (DEMC, purple), Kernel Density Estimate Metropolis (KM, orange), Ensemble Slice Sampling using the differential move (ESS, green), and Ensemble Slice Sampling using the global move (ESS, blue). The target distribution is a 50−dimensional Gaussian Mixture.The figure shows the 1D marginal distribution for the first parameter of the 50.

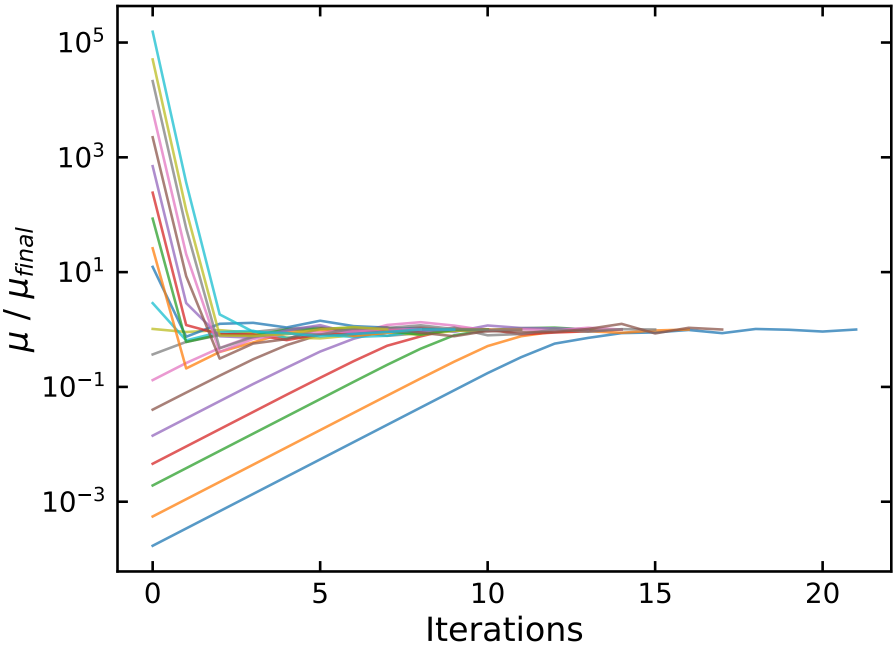

Figure 10.10: The plot shows the adaptation of the length scale μ as a function of the number of iterations and starting from a wide range of initial values. Each trace is an independent run and the y-axis shows the value of μ divided by the final value of μ. The target distribution in this example is a 20−dimensional correlated normal distribution. Starting from larger μ values leads to significantly faster adaptation.

10.4.3 Convergence of the Length Scale μ

Figure 10.10 plots the convergence of the length scale during the first 20 iterations. The target distribution in this example is a 20−dimensional correlated normal distribution. The length scale μ was initialised from a wide range of possible values. Adaptation is significantly faster when the initial length scale is larger than the optimal one rather than smaller. Another benefit of using a larger initial estimate would be the reduced number of probability evaluations during the first iterations. This is due to the fact that the shrinking procedure is generally faster than the stepping-out procedure.

10.4.4 Parallel Scaling

By construction, Ensemble Slice Sampling can be used in parallel computing environments by parallelising the ensemble of walkers as discussed in Section 10.3.2. The maximum number of CPUs used without any of them being idle is equal to the size of complementary ensemble, $n_{\text{Walkers}}/2$. In order to verify this empirically and investigate the scaling of the method for any number of CPUs, we sampled a 10−dimensional Normal distribution for 10^5 iterations with varying number of walkers. The results are plotted in Figure

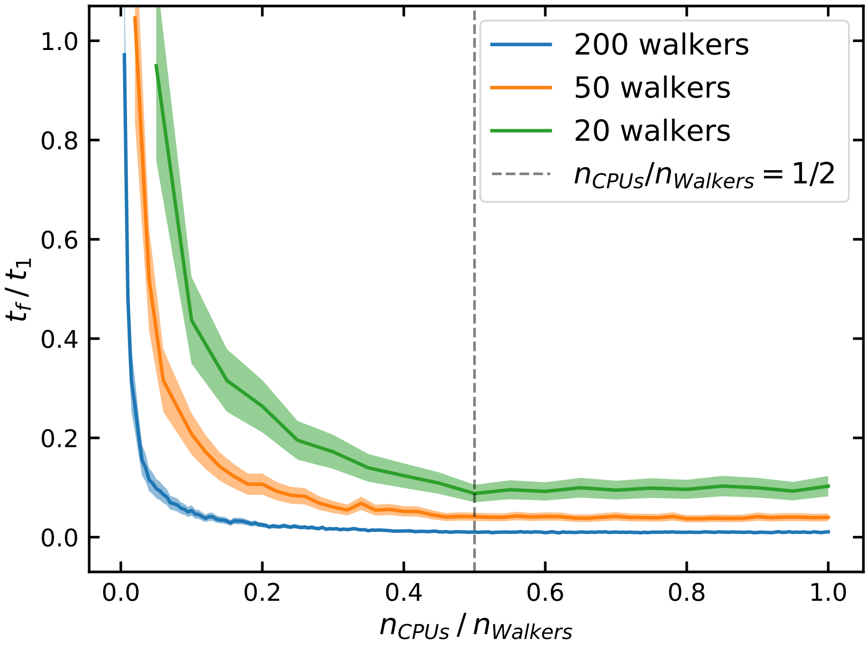

Figure 10.11: The plot shows the time t_f required for ESS to complete a pre-specified number of iterations as a function of the ratio of the number of available CPUs n_{CPUs} to the total number of walkers n_{Walkers}. The results are normalised with respect to the single CPU case t_1. The method scales as $\mathcal{O}(1/n_{\text{CPUs}})$ as long as $n_{\text{CPUs}} \leq n_{\text{Walkers}}/2$ (dashed line). The shaded areas show the $2-\sigma$ intervals.

10.11. We sampled the aforementioned distribution multiple times in order to get estimates of the confidence integrals shown in Figure 10.11. The required time to do the pre-specified number of iterations scales as $\mathcal{O}(1/n_{\text{CPUs}})$ as long as $n_{\text{CPUs}} \leq n_{\text{Walkers}}/2$. This result does not depend on the specific distribution. We can always use all the available CPUs by matching the size of the complementary ensemble (i.e. half the number of walkers) to the number of CPUs.

10.5 DISCUSSION

In Section 10.4 we provided a quantitative comparison of the efficiency of Ensemble Slice Sampling compared to other methods. In this Section we will provide some qualitative arguments to informally demonstrate the advantages of Ensemble Slice Sampling over other methods. Furthermore, we will briefly discuss some general aspects of the algorithm and place our work in the context of other related algorithms.

After the brief adaptation period is over and the length scale μ is fixed, the Ensemble Slice Sampling algorithm performs on average 5 evaluations of the probability density per walker per iteration, assuming that either the differential or Gaussian move is used. This is in stark contrast with Metropolis-based MCMC methods that perform 1 evaluation of the probability density per iteration. However, the non-rejection nature of Ensemble Slice Sampling more than compensates for the higher number of evaluations as shown in Section 10.4, thus yielding a very efficient scheme.

One could think of the number of walkers as the only free hyperparameter of Ensemble Slice Sampling. However, choosing the number of walkers is usually trivial. As we mentioned briefly at the end of Section 10.3, there is a minimum limit to that number. In particular, in order for the method to be ergodic, the ensemble should be made of at least $2 \times D$ walkers[7], where D is the number of dimensions of the problem. Assuming that the initial relative displacements of the walkers span the parameter space (i.e. they do not belong to a lower-than-D-dimensional space) the resulting algorithm would be ergodic. As shown in Section 10.4, using a value close to the minimum number of walkers, meaning twice the number of parameters, is generally a good choice. Furthermore, we suggest to increase the number of walkers by a multiplicative factor equal to the number of well separated modes (e.g. four times the number of dimensions in a bimodal density). Other cases in which increasing the number of walkers can improve the sampling efficiency include target distributions with strong non-linear correlations between their parameters.

Regarding the initial positions of the walkers, we found that we can reduce the length of the burn-in phase by initialising the walkers from a tight sphere (i.e. Normal distribution with a very small variance) close to the *Maximum a Posteriori* (MAP) estimate. In high dimensional problems, the MAP estimate will not reside in the typical set and the burn-in phase might be longer. We found that the tight sphere initialisation is still an efficient strategy compared to a more dispersed initialisation (Foreman-Mackey, Hogg, et al., 2013). Other approaches include initialising the walkers by sampling from the prior distribution or the *Laplace approximation* of the posterior distribution. In multimodal cases, a prior initialisation is usually a better choice. A brief simulated annealing phase can also be very efficient, particularly in cases with many well separated modes.

Recent work on the No U-Turn Sampler (M. D. Hoffman, Gelman, et al., 2014) has attempted to reduce the hand-tuning requirements of Hamiltonian Monte Carlo (Betancourt, 2017b) using the dual averaging scheme of Nesterov (2009). In order to achieve a similar result, we employed the much simpler stochastic approximation method of Robbins & Monro (1951) to tune the initial length scale μ. The Affine Invariant Ensemble Sampler (Goodman

---

[7] The reason that the minimum limit is $2 \times D$ instead of $D + 1$ has to do with the ensemble splitting procedure that we introduced in order to make the method parallel. Splitting the ensemble into two equal parts means that each walker is updated based on the relative displacements of half the ensemble.

& Weare, 2010) and the Differential Evolution MCMC (Ter Braak, 2006) use an ensemble of walkers to perform Metropolis updates. Our method differs by using the information from the ensemble to perform Slice Sampling updates. So why does ESS perform better, as demonstrated, compared to those other methods? The answer lies in the locally adaptive and non-rejection nature of the algorithm (i.e. stepping out and shrinking) that enables both efficient exploration of non-linear correlations and large steps in parameter space (e.g. using the global move)[8].

For all numerical benchmarks in this paper we used the publicly available, open source `Python` implementation of Ensemble Slice Sampling called `zeus`[9] (Karamanis, Beutler & Peacock, 2021).

10.6 CONCLUSION

We have presented Ensemble Slice Sampling (ESS), an extension of Standard Slice Sampling that eliminates the latter's dependence on the initial value of the length scale hyperparameter and augments its capacity to sample efficiently and in parallel from highly correlated and strongly multimodal distributions.

In this paper we have compared Ensemble Slice Sampling with the optimally-tuned Metropolis and Standard Slice Sampling algorithms. We found that, due to its affine invariance, Ensemble Slice Sampling generally converges faster to the target distribution and generates chains of significantly lower autocorrelation. In particular, we found that in the case of AR(1), Ensemble Slice Sampling generates an order of magnitude more independent samples per evaluation of the probability density than Metropolis and Standard Slice Sampling. Similarly, in the case of the correlated funnel distribution, Ensemble Slice Sampling outperforms Standard Slice Sampling by an order of magnitude in terms of efficiency. Furthermore, in this case, Metropolis-based proposals fail to converge at all, demonstrating that a single Metropolis proposal scale is often not sufficient.

When compared to state-of-the-art ensemble methods (i.e. AIES, DEMC) Ensemble Slice Sampling outperforms them by $1-2$ orders of magnitude in terms of efficiency for target distributions with non-linear correlations (e.g. the Ring and Gaussian shells distributions). In the real world example of hierarchical Gaussian process regression, ESS's efficiency is again superior by $1-2$ orders of magnitude. Furthermore, in the Bayesian object detection example ESS achieved higher accuracy compared to AIES and DEMC. Finally, in the strongly multimodal case of the Gaussian Mixture, ESS outperformed all other methods (i.e. SMC, AIES, DEMC, KM) and was the only sampler able to produce reliable results in 50 dimensions.

---

[8] Indeed, large steps like the ones in the 50−dimensional Gaussian Mixture example would not have been possible without the non-rejection aspect of the method as most attempts to jump to the other mode would have missed it using Metropolis updates.

[9] The code is available at `https://github.com/minaskar/zeus`.

The consistent high efficiency of the algorithm across a broad range of different problems along with its parallel, black-box and gradient-free nature, renders Ensemble Slice Sampling ideal for use in scientific fields such as physics, astrophysics and cosmology, which are dominated by a wide range of computationally expensive and almost always non-differentiable models. The method is flexible and can be extended further using for example tempered transitions (Iba, 2001) or subspace sampling (Vrugt et al., 2009).

10.7 APPENDIX: ESTIMATING THE EFFECTIVE SAMPLE SIZE

Assuming that the computational bottleneck of a MCMC analysis is the evaluation of the probability density function, which is usually a valid assumption in scientific applications, the *efficiency* can be formally defined as the ratio of the *Effective Sample Size* N_{Eff} to the total number of probability evaluations for a given chain.

The N_{Eff} quantifies the number of effectively independent samples of a chain, and it is defined as

$$N_{\text{Eff}} = \frac{n}{\text{IAT}}\,, \tag{10.24}$$

where n is the actual number of samples in the chain, and IAT is the *integrated autocorrelation time*. The latter describes the number of steps that the sampler needs to do in order to forget where it started and it is defined as

$$\text{IAT} = 1 + 2\sum_{k=1}^{\infty}\rho(k)\,, \tag{10.25}$$

where $\rho(k)$ is the *normalised autocorrelation function* at lag k. In practice, we truncate the above summation in order to remove noise from the estimate (Sokal, 1997).

Given a chain $X(k)$ with $k = 1, 2, ..., n$ the normalised autocorrelation function $\hat{\rho}(k)$ at lag k is estimated as

$$\hat{\rho}(k) = \frac{\hat{c}(k)}{\hat{c}(0)}\,, \tag{10.26}$$

where

$$\hat{c}(k) = \frac{1}{n-k}\sum_{m=1}^{n-k}\left[X(k+m) - \bar{X}\right]\left[X(m) - \bar{X}\right]\,, \tag{10.27}$$

and $\bar{X}$ is the mean of the samples.

In the case of ensemble methods, the IAT of an ensemble of chains is computed by first concatenating the chain from each walker into a single long chain. We found this estimator has lower variance than the Goodman & Weare (2010) estimator and the Foreman-Mackey (2019) estimator.

11 ZEUS

This chapter presents *zeus* which is the main contribution introduced in the paper titled *zeus: A Python implementation of Ensemble Slice Sampling for efficient Bayesian parameter inference* that was published in the journal *Monthly Notices of the Royal Astronomical Society* in 2021 (Karamanis, Beutler & Peacock, 2021). The content of the chapter is almost identical to that included in the aforementioned publication with the exception of minor text and figure formatting differences.

We introduce `zeus`, a well-tested `Python` implementation of the Ensemble Slice Sampling (ESS) method for Bayesian parameter inference. ESS is a novel Markov chain Monte Carlo (MCMC) algorithm specifically designed to tackle the computational challenges posed by modern astronomical and cosmological analyses. In particular, the method requires only minimal hand–tuning of 1 – 2 hyper-parameters that are often trivial to set; its performance is insensitive to linear correlations and it can scale up to 1000s of CPUs without any extra effort. Furthermore, its locally adaptive nature allows to sample efficiently even when strong non-linear correlations are present. Lastly, the method achieves a high performance even in strongly multimodal distributions in high dimensions. Compared to `emcee`, a popular MCMC sampler, `zeus` performs 9 and 29 times better in a cosmological and an exoplanet application respectively.

11.1 INTRODUCTION

Over the past few decades the volume of astronomical and cosmological data has increased substantially. In response to that, a variety of astrophysical models have been developed to explain the plethora of observations. Markov chain Monte Carlo (MCMC) has been established as the standard procedure of inferring the model parameters subject to the available data in a Bayesian framework. Within the Bayesian context, the object that quantifies the probability distribution of the model parameters θ given the data D and model $\mathcal{M}$ is the posterior distribution $\mathcal{P}(\theta) \equiv P(\theta|D,\mathcal{M})$ which is defined using Bayes's theorem:

$$\mathcal{P}(\theta) = \frac{\mathcal{L}(\theta)\pi(\theta)}{\mathcal{Z}}, \tag{11.1}$$

where $\mathcal{L}(\theta) \equiv P(D|\theta, \mathcal{M})$ is the likelihood function, $\pi(\theta) \equiv P(\theta|\mathcal{M})$ is the prior distribution of the model parameters θ, and $\mathcal{Z} \equiv P(D|\mathcal{M})$ is the, so called, Bayesian model evidence or marginal likelihood and in this context can be treated as a simple normalisation constant.

MCMC does not in general require knowing the value of the model evidence and it only depends on the ability to evaluate the unnormalised posterior distribution for arbitrary values of θ. MCMC methods can then be used to generate (Markov) chains of samples from the posterior distribution. Those samples can be used to calculate integrals (e.g. parameter uncertainties, marginal distributions etc.) that are paramount for modern astronomical and cosmological analyses.

The most commonly used MCMC methods are variants of the Metropolis-Hastings (MH) algorithm (Hastings, 1970; Metropolis et al., 1953). MH consists of two steps. First, given the last sample in the chain, a new sample is proposed and then the Metropolis criterion determines whether or not that new sample should be accepted and thus added to the chain. The resulting chain is Markovian in the sense that each sample is proposed based only on the previous sample. The purpose of the Metropolis acceptance criterion is to bias the chain so that the time spent in a region of the parameter space would be proportional to the posterior probability in that region. In other words, the stationary distribution of the Markov chain is the target distribution i.e. the posterior distribution. For a detailed introduction to MCMC methods we direct the reader to MacKay (2003) and for an intuitive introduction to Bayesian inference to E. T. Jaynes (2003).

Arguably, the most difficult part of the MH algorithm is the proposal step. There are many ways of choosing a new sample and the efficiency of the method depends on this choice. By far the simplest one is the use of a normal (Gaussian) distribution, centred around the previous sample to generate the new proposed sample. The resulting method is often called Random Walk Metropolis algorithm and its performance is highly sensitive to the $n(n+1)/2$ elements that form its covariance matrix. Those elements generally need to be chosen *a priori* or be adaptively tuned. More efficient methods utilise the gradient of the target distribution (Betancourt, 2017a) or an ensemble of parallel and communicating chains (Gilks, Roberts, et al., 1994; Goodman & Weare, 2010; Ter Braak, 2006; Ter Braak & Vrugt, 2008).

Out of the methods mentioned in the previous paragraph we will focus our attention on the last one, the ensemble or population MCMC variety. The reason is simple: the Random Walk Metropolis algorithm requires a great amount of tuning (or *a priori* knowledge) for it to perform efficiently and even then there is no guarantee that the proposal covariance matrix is optimal for the whole parameter space. On the other hand, gradient based methods, although very powerful, are in general unsuitable for astronomical applications in which the models that are used are almost always not differentiable.

One benefit of ensemble MCMC over its alternatives is that the ensemble of parallel chains (also known as walkers) collectively sample the posterior, thus information about their distribution can be shared and used to make better educated proposals. Other advantages include the lack of hand-tuning of hyper-parameters and their capacity for parallel implementation. For the aforementioned reasons, ensemble MCMC methods have dominated astronomical analyses. The most common ones are affine–invariant ensemble sampling (AIES) (Goodman & Weare, 2010) and differential evolution MCMC (DEMC) (Ter Braak, 2006; Ter Braak & Vrugt, 2008), both implemented in the popular `Python` package `emcee` (Foreman-Mackey, Will M Farr, et al., 2019; Foreman-Mackey, Hogg, et al., 2013).

In this paper we introduce `zeus`, a stable and well-tested `Python` implementation of Ensemble Slice Sampling (ESS) (Karamanis & Beutler, 2021). ESS is a method based on the ensemble MCMC paradigm, with the crucial difference being that its proposals are performed via Slice Sampling updates (Neal, 2003) instead of Metropolis-Hastings ones. As we will thoroughly demonstrate in Section 11.3, this subtle difference leads to substantial improvements in terms of sampling efficiency and robustness. `zeus` is a user-friendly tool that does not require any hand-tuning or preliminary runs and can scale up to 1000s of CPUs without any extra effort from the user.

`zeus` has been used in various astronomical and cosmological analyses, including cosmological tests of gravity (Tamosiunas, 2020), relativistic effects and primordial non-Gaussianity (M. Wang et al., 2020), 21cm intensity mapping (Umeh et al., 2021), and has been implemented as part of the `CosmoSIS` package (J. Zuntz et al., 2015).

`zeus` is open source software that is publicly available at `https://github.com/minaskar/zeus` under the `GPL-3 Licence`. Detailed documentation and examples on how to get started are available at `https://zeus-mcmc.readthedocs.io`.

11.2 ENSEMBLE SLICE SAMPLING

`zeus` is a `Python` implementation of the Ensemble Slice Sampling (ESS) method presented in Karamanis & Beutler (2021). Here we will provide a high-level description of the method and will refer to the accompanying paper for more details about the underlying algorithmic structure and mathematics.

ESS combines the ensemble MCMC paradigm with slice sampling. Since the use of slice sampling in astronomical parameter inference is rare we will start by explaining its function and how it differs from Metropolis updates.

Then we will move on to discuss how it can be efficiently combined with ensemble MCMC.

11.2.1 Slice sampling

Slice sampling is based on the idea that sampling from a distribution with density $P(x)$ is equivalent to uniform sampling from the area under the plot of $f(x) \propto P(x)$. To this end, we introduce an auxiliary variable y, called height, such that the joint distribution $P(x, y)$ is uniform over the region $U = \{(x, y) : 0 < y < f(x)\}$. To sample from the marginal distribution $P(x)$, we first sample from $P(x, y)$ and then we marginalise by dropping the y value of each sample.

In order to generate samples from $P(x, y)$ we utilise the following scheme (Neal, 2003):

1. Given the current state x_0, draw y_0 uniformly from $(0, f(x_0))$.
2. Find an interval $I = (L, R)$ that contains all, or at least part, of the slice $s = \{x : y_0 < f(x)\}$.
3. Draw the new sample x_1 uniformly from $I \cap S$.

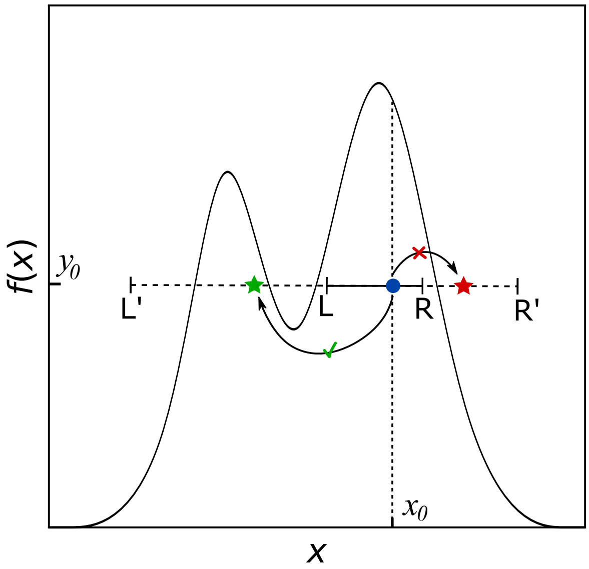

Figure 11.1: Illustration of the univariate slice sampling update. Given the current sample x_0, a value y_0 is uniformly sampled along the vertical slice $(0, f(x_0))$ (dashed line) thus defining the initial point (blue). An interval (L, R) is uniformly positioned horizontally around (x_0, y_0) and it is expanded in steps of size $R - L$ until both its ends are outside the slice. The new sample is generated by repeatedly sampling (uniformly) from the interval (L', R') until a sample (green star) is found inside the slice. Samples outside of the slice (red star) are rejected and they are instead used to shrink (L', R').

To construct the interval I (step ii), Neal (2003) introduced the stepping-out procedure that works by randomly positioning an interval of length μ around the sample x_0 (i.e. blue dot in Figure 11.1) and then expanding it in steps of size μ until both its ends (i.e. L' and R') are outside the slice. To obtain x_1 (i.e. green star in Figure 11.1) we then use the shrinking procedure in which candidates are sampled uniformly from I until a point inside the slice S is found. Samples outside of the slice are used to shrink the interval I. The two procedures are shown in Figure 11.1.

The length scale μ is the only free hyperparameter of slice sampling and although its choice can reduce or increase the computational cost of the method it generally does not affect its mixing properties (e.g. convergence rate, autocorrelation time, etc.). `zeus` utilises a stochastic optimization algorithm similar to Tibbits et al. (2014) and based on the Robbins & Monro (1951) optimisation scheme in order to tune μ to its optimal value (see Section 3.1 of Karamanis & Beutler (2021) for more details).

It is important to note here that for multimodal target distributions there is no guarantee that the approximate slice would cross any of the other modes. In particular, if the initial estimate of the length scale μ is low then the probability of missing the other peaks, assuming that they are located far away, is also low. As we will show in Section 11.3, unlike simple slice sampling, ESS and thus `zeus` does not suffer from this effect.

11.2.2 Walkers, moves and parallelism

The slice sampling update described in the previous paragraphs is a univariate update scheme. For it to be used to sample from multivariate target distributions it needs to be generalised accordingly. Perhaps the simplest such generalisation in a multivariate setting is the use of slice sampling to sample along each coordinate axis in turn (i.e. component-wise slice sampling) or to sample along randomly selected directions in parameter space (MacKay, 2003). Although valid, both of these approaches are unsuitable in cases of correlated parameters in which the proper choice of direction can substantially accelerate mixing.

To address this issue, Tibbits et al. (2014) proposed to orthogonalise the parameter space using the sample covariance, thus getting rid of linear correlations between parameters. We will instead follow a different, perhaps more flexible, approach to construct an efficient slice sampler. Our aim is to utilise an ensemble of parallel chains/walkers that can exchange information about the covariance structure of the target distribution and thus by-pass the difficulties posed by correlations.

As hinted in the introduction, the ensemble of walkers collectively sample the target distribution and thus their positions encode information about the correlations between the parameters. One way to take advantage of this

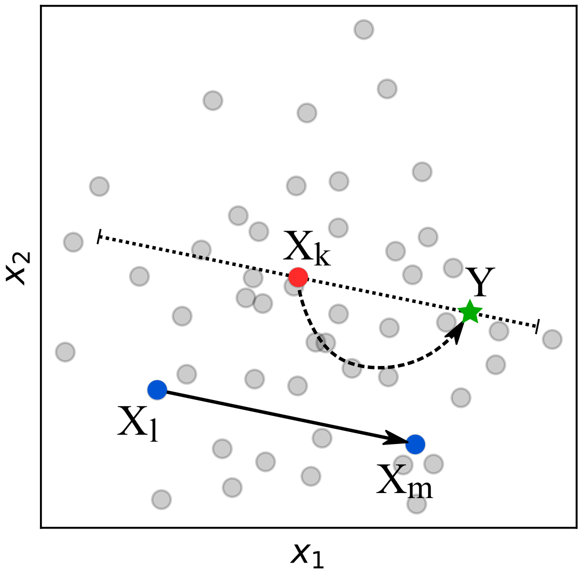

Figure 11.2: The figure illustrates the differential move in the context of Ensemble Slice Sampling. The walker X_k to be updated is shown in red. Two walkers, X_l and X_m, (blue) are uniformly selected from the complementary ensemble (grey). The approximate slice (dotted line) is constructed parallel to the two walkers X_l and X_m using the stepping-out procedure. The new position Y (green) of X_k is sampled using the shrinking procedure along the approximate slice.

information is to use it to construct direction vectors along which slice sampling can take place. Many moves that generate direction vectors from the complementary ensemble are possible. zeus offers a collection of them, including some that utilise clustering algorithms and density estimation methods. As we will show in Section 11.3, such moves can help accelerate sampling in difficult cases such as strongly multimodal distributions. Any distribution of the complementary ensemble can be used as a valid proposal to generate such direction vectors and zeus offers a highly flexible interface for the user to define such a move or choose one (or a mixture) from the ones that are already implemented and tested. Here is a list of the currently implemented moves in zeus:

- **Differential move:** This is the default move used by zeus and shown in Figure 11.2. Using the differential move, Ensemble Slice Sampling updates the position of each walker in the ensemble by slice sampling along a direction defined by the difference between two uniformly selected walkers from the rest of the ensemble (i.e. the complementary ensemble).

- **Gaussian move:** The Gaussian move samples the direction vectors along which slice sampling is performed from a normal distribution that shares the same covariance structure as the complementary ensemble. This approach is very efficient in cases in which the target distribution is close to normal.

- **Global move:** The Global move utilises a Dirichlet Process Gaussian Mixture to fit the complementary ensemble and proposes directions along different peaks of the target distribution in cases of strong multimodality.

- **KDE move:** The KDE move samples the direction vectors from a Gaussian Kernel Density Estimate of the complementary ensemble. This can be useful in cases of highly non-Gaussian target distributions.

- **Random move:** The Random move performs slice sampling along isotropic directions. This is equivalent of standard multivariate slice sampling and it is mostly offered for testing purposes as it cannot handle correlations efficiently.

For more information on how those moves work as well as a comparison of the Differential, Gaussian and Global moves we direct the interested reader to Karamanis & Beutler (2021). Unless stated otherwise the Differential move will be used for the following examples.

To parallelise this process and capitalise on the availability of multiple CPUs we randomly split the ensemble into two sets of walkers (i.e. active and passive sets) (Foreman-Mackey, Hogg, et al., 2013) and choose to update the positions of the active walkers along direction vectors defined by passive walkers. Then the passive becomes active and *vice versa* and the process is repeated. The ensemble splitting technique is required in order to parallelise the algorithm without violating detailed balance. Parallelisation is achieved in practice using either `multiprocessing` or `MPI` using the implemented `ChainManager` utility that can distribute both multiple ensembles and multiple chains in parallel computing environments at the same time. Heuristics to determine the number of required walkers per application are discussed in Section 11.4.

11.3 EMPIRICAL EVALUATION

For the empirical evaluation of `zeus` we use five toy examples that manifest significant aspects of real astronomical applications[1] (i.e. linear and non-linear correlations, multimodality, heavy tails, hard boundaries) and two real-world astronomical examples characteristic of modern astronomical analyses.

---

[1] For additional demonstrations on similarly common structures (e.g. the funnel) we direct the reader to the accompanying paper (Karamanis & Beutler, 2021).

11.3.1 Toy examples

In order to understand the behaviour of `zeus` in various sampling scenarios, it is important to study its performance in different toy examples that demonstrate different characteristics of common target distributions that arise in astronomical applications. For that reason, we chose five such toy examples. The first one is a normal (Gaussian) distribution which by definition is characterised only by the linear correlation between its parameters. The second toy problem is the ring distribution, a characteristic example of strong non-linear correlations. The third example is a Gaussian mixture with two components. While the purpose of the first two examples is to study the behaviour of the algorithm in the presence of linear and non-linear correlations respectively, the goal of the third example is to demonstrate the ability of `zeus` to sample efficiently from multimodal target distributions. The fourth toy example investigates the effect that heavy tails have on the sampling efficiency and the fifth shows the effects that hard boundaries have on sampling.

We compare `zeus` with two popular alternatives offered by `emcee`, namely affine–invariant ensemble sampling with the *stretch* move (`emcee`/AIES) and the differential evolution move (`emcee`/DEMC). The main goal of this analysis is to justify our choice of slice sampling as the basis of `zeus` instead of Metropolis updates through the use of simple yet instructive toy examples.

For all three toy examples discussed below we adopt the same analysis procedure, where we initialise the walkers by sampling from a normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ where $\mathbf{I}$ is the identity covariance matrix and we discarded 10^4 iterations as burn–in.

The main metric that we use to investigate the behaviour of the samplers in those toy examples and to compare their performance is the distribution of steps performed by the walkers. As a step, we define the distance spanned in parameter space by a single walker in a single iteration. This is a fundamental measure of the efficiency of an MCMC method and it is directly related to the expected squared jump distance (ESJD) (Pasarica & Gelman, 2010) given by:

$$\mathrm{ESJD} = \mathbf{E}\left[|\theta_{t+1} - \theta_t|^2\right] = 2\,(1 - \rho_1) \cdot \mathrm{Var}_{(\pi)}(\theta_t)\,, \tag{11.2}$$

where θ_t are the chain samples, ρ_1 is the first-order autocorrelation, and $\mathrm{Var}_{(\pi)}(\theta_t)$ is a function of the stationary distribution only. Assuming that the higher-order autocorrelations $\rho_2, \rho_3, \ldots$ are monotonically decreasing with respect to ρ_1, then maximising the ESJD leads to minimisation of the autocorrelation between chain elements and thus maximisation of the sampling efficiency. In other words, the further away (i.e. the greater the ESJD) the walkers jump per iteration, the higher the sampling efficiency of the method. A benefit of using ESJD instead of the autocorrelation time as a metric is that the former, as an expectation value, is more accurate when computed using short chains.

In order to account for the different computational costs (i.e. different number of model evaluations per iteration) between `zeus` and `emcee` we thinned the chains of the latter method according to the average number of model evaluations of `zeus`. This allowed us to compare the distribution of steps of the three samplers as shown in Figures 11.4, 11.9, 11.11, 11.13, and 11.15 for the five toy examples respectively.

The correlated normal distribution

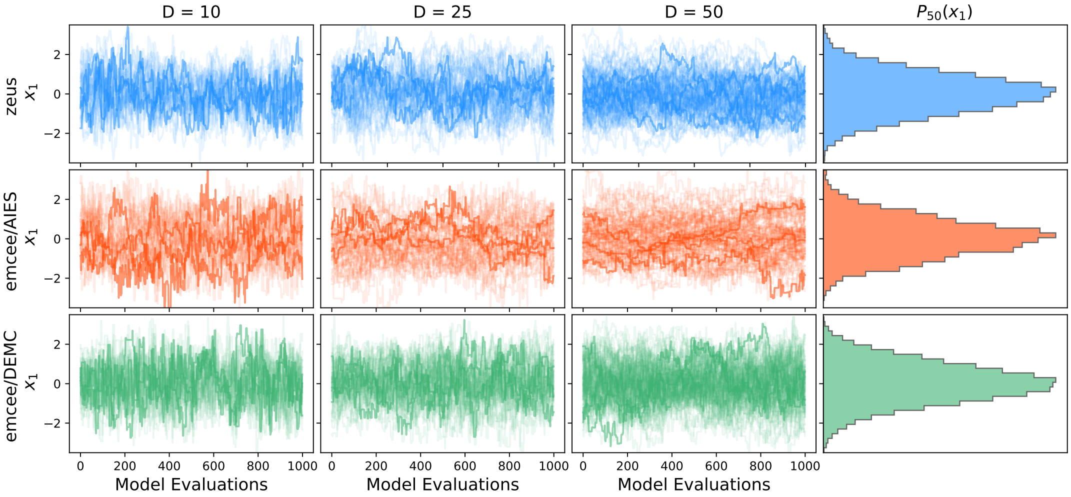

Figure 11.3: The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of a normal (Gaussian) target distribution in $10, 25$ and 50 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 50-dimensional case.

Starting with the normal target distribution it is important to note here that all three of the methods used in the comparison are affine–invariant[2], meaning that their performance is immune to any linear correlations between the parameters. Since the normal distribution incorporates, by construction, only linear correlations (i.e. the 2D marginal distribution contours look like ellipses), it is the perfect testing ground to assess the effect that high dimensionality has on the three methods independently of other complications. For our example, we used a zero-mean normal distribution with a covariance matrix in which the diagonal elements are set to 1 and the off-diagonal ones are equal to 0.95. We then proceed by sampling the aforementioned distribution in 10, 25 and 50 dimensions. Based on Figure 11.3 one can see that the walkers of `emcee`/AIES dissolve into an inefficient random walk characterised by low step size and high autocorrelation time as the number of parameters increases. `zeus` and `emcee`/DEMC are not so severely affected by the high number of parameters exhibiting a substantially lower autocorrelation.

---

[2] Differential evolution Metropolis is only approximately affine–invariant due to the jitter that it is often added to its proposal. This however has a negligible effect.

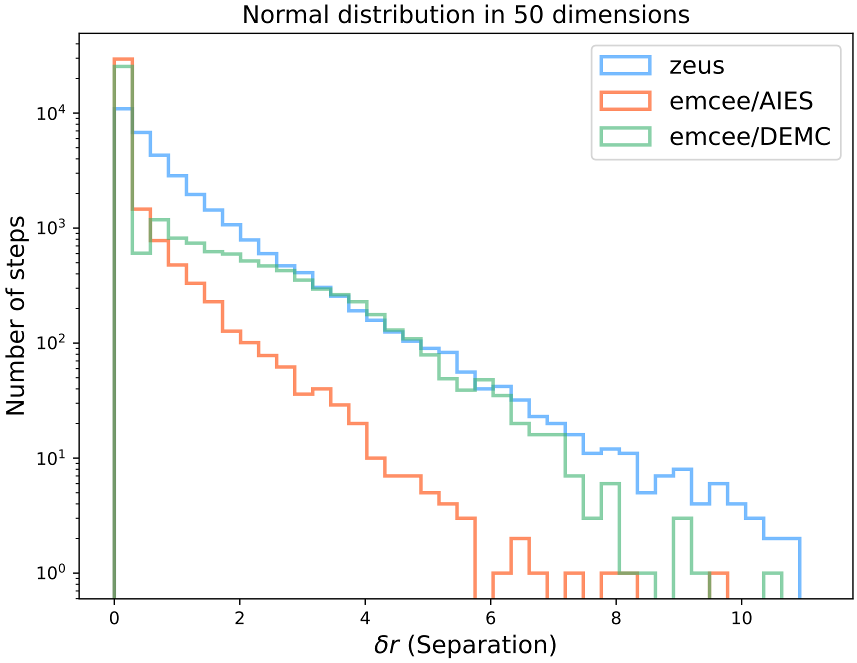

Figure 11.4: This figure shows the distribution of step sizes of walkers for the three different samplers in the case of a normal (Gaussian) target distribution in $D = 50$. It is important to note here that both `emcee` algorithms exhibit a peak at zero separation; `zeus` on the other hand does not due to its non-rejection nature.

Let us now try to explain this difference in behaviour by looking into the distribution of the steps of the walkers in Figure 11.4. One thing to notice here is that the distribution of the steps of `zeus`'s walkers extends significantly further away than those of `emcee`/AIES and `emcee`/DEMC. This should come as no surprise since the construction of the approximate slice allows for larger steps than Metropolis updates as shown in Table 4. This is because when a proposal is rejected in slice sampling the approximate slice shrinks and another sample is proposed instead. In this way, `zeus`'s walkers always move and the chance of staying fixed is zero – unlike MH-based updates in which frequent rejection of samples is a necessity. This aforementioned procedure leads to greater steps in parameter space. The difference between `emcee`/AIES and `emcee`/DEMC is attributed to the fact that DEMC uses a proposal scale[3] $\gamma = 2.38/\sqrt{D}$ that guarantees a constant acceptance rate accounting for the number of dimensions D. This proposal scale is however optimal only in the case of a normal target distribution such as the one that we are studying here and there is no guarantee that it would return acceptable results in non-Gaussian distributions. For the case of `emcee`/AIES, the relevant proposal scale γ is allowed to vary in the range between $1/\alpha$ and α where $\alpha = 2$ is often taken as the typical value. It is clear that in the latter case γ does not possess the desired scaling $\gamma \propto 1/\sqrt{D}$ and thus, although the method generates proposals in the right overall direction, most of the samples do not reside in the typical set (Speagle, 2019). In other words, the lack of proper scaling of the proposal scale with the number of dimensions leads to `emcee`/AIES "overshooting" the typical set where most of the posterior mass is located.

---

[3] The proposal scale γ is similar to μ used in ESS in the sense that its value determines the length scale of the proposed jumps in parameter space. A high value would lead to large

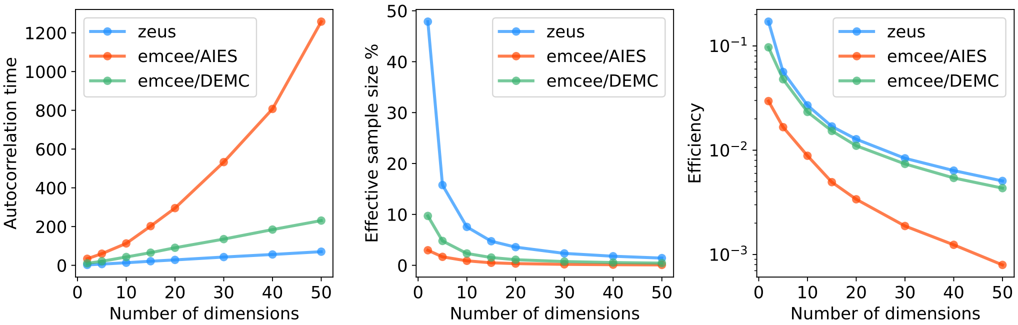

Figure 11.5: The figure shows numerical estimates of the integrated autocorrelation time (number of steps along a chain required to obtain an independent sample; left panel), the effective sample size (percentage of effectively independent samples in a chain; middle panel), and the sampling efficiency (i.e. effective sample size per model evaluation; right panel) for a normal target distribution and varying number of dimensions. The number of walkers was set to $4 \times D$ for `zeus` and $16 \times D$ for `emcee`, this was the optimal choice (i.e. the one maximising the efficiency for the given dimensionality) for each sampler. `zeus` and `emcee/`DEMC exhibit linear scaling of the autocorrelation time with the number of dimensions whereas `emcee/AIES` scales exponentially.

We can also draw some useful insights about the sampling efficiency of those samplers and their scaling with the number of dimensions by estimating the integrated autocorrelation time of the chains. Given the autocorrelation time, we can also estimate the effective sample size as the percentage of effectively independent samples in a chain. By dividing the effective sample size by the computational cost of each method we can then estimate the sampling efficiency. The results of such a comparison are shown in Figure 11.5. We immediately notice here that the autocorrelation times of `zeus` and `emcee`/DEMC scale linearly with the number of dimensions, whereas

---

steps that are often rejected and a low value would lead to small steps that are often accepted but do not carry the walkers far. For such methods, a balance must be found.

the autocorrelation time of `emcee`/AIES scales exponentially. The computational cost of `zeus` per iteration per walker, although somewhat higher than that of `emcee`, does not vary with the number of dimensions. This means that in high dimensions, `zeus` dominates over `emcee`/AIES in terms of sampling efficiency.

Table 4: The table shows a comparison of `emcee`/AIES, `emcee`/DEMC and `zeus` in terms of the expected squared jump distance (ESJD; higher is better) for the five toy examples i.e. 50-D normal distribution, 25-D ring distribution, 25-D Gaussian mixture, 25-D Student's t-distribution, and 25-D truncated normal distribution.

	`emcee`/AIES	`emcee`/DEMC	**zeus**
Normal	0.5288	1.1162	**2.1354**
Ring	0.0043	0.0006	**0.1257**
Mixture	0.0037	0.0056	**0.1015**
Student	12.9124	2.4137	**23.5720**
Truncated	0.0940	0.3501	**0.5882**

The above discussion allows us to clearly state a crucial distinction between the three methods, which is their response to the curse of dimensionality. As the number of dimensions increases, the probability mass of a distribution is concentrated into a thin shell within the tails of the distribution (i.e. the typical set). To account for this and maintain its efficiency, a sampling method has to adjust its proposal scale – otherwise, the proposals will not be located in the typical set and thus they will not be accepted. The three methods that we mentioned so far deal with this in different ways. `emcee`/AIES's proposal scale is not adjusted and thus its proposals become increasingly inefficient in high dimensions. `emcee`/DEMC's proposal scale is adjusted based on the theoretical expectation for the case of the normal target distribution. Although both `emcee` methods perform well in this example, their sub-optimal scaling will degrade their performance in non-Gaussian target distributions as we will demonstrate in the next toy example. Finally, `zeus`'s proposal scale is continuously adapted, as the slice expands and contracts in every iteration, thus guaranteeing optimal scaling. Huijser et al. (2017) found that the suboptimal scaling of `emcee`/AIES with the number of dimensions can introduce biases into the expectation values derived from the chains in high dimensions that are hard to diagnose. The locally adaptive nature of `zeus` allows it to avoid this problem by adjusting its proposals accordingly.

Another kind of analysis we can perform is to use the highly correlated 25–dimensional normal distribution as the target distribution and estimate the convergence rate of the three samplers. Although simple, the normal distribution is a valid approximation of many realistic astronomical posterior distributions and as such we expect the results presented in this paragraph to

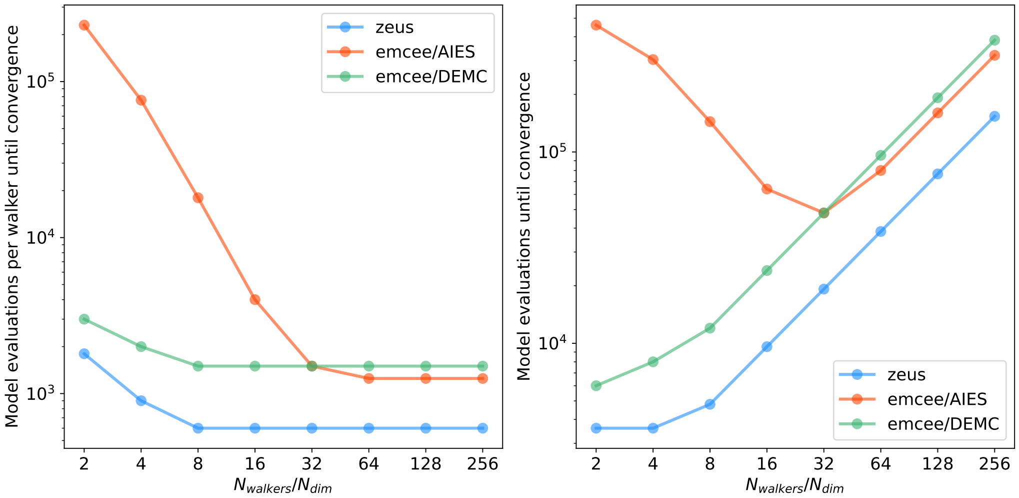

Figure 11.6: The figure shows the computational cost until convergence is reached in terms of the number of model evaluations for the different ensemble samplers for a highly correlated 25–dimensional normal distribution. The left panel shows the computational cost for a single walker. From this we can see that the cost for a single walker decreases as we increase the number of walkers until it reaches a plateau. The high computational cost for low numbers of walkers can be attributed to the low variety or sparsity of possible proposals; this is significantly higher for `emcee`/AIES. The right panel takes into account the linear scaling of the total computational cost as we increase the number of walkers and shows the total computational cost for the whole ensemble until it converges.

be applicable to a wide range of other distributions that resemble the normal distribution to some extent. We acknowledge however that the *no free lunch* theorem also applies to this case, and there are bound to be cases in which the results would be qualitatively different. That being said, we initialised the walkers from a compact normal distribution (i.e. standard deviation equal to 10^{-4} times that of the target distribution) centred around a point along the first axis of the parameter space at a distance of 100 standard deviations from the mode. We then measured the number of model evaluations required until the samplers have converged to the target distribution. The results for varying number of walkers are presented in Figure 11.6.

In general, walkers move along directions defined by the walkers of the complementary ensemble. Thus, increasing the number of walkers offers a wider variety of available directions along which the walkers of `zeus` or `emcee` can move via slice sampling or Metropolis updates respectively. This is demonstrated in the left panel of Figure 11.6 in which the computational cost until convergence (i.e. number of model evaluations) for a single walker diminishes and then reaches a plateau as the number of walkers is increased. We notice however that, at the level of a single walker, the computational cost of `emcee`/AIES is significantly higher compared to that of either `zeus` or `emcee`/DEMC. This is due to the way that different samplers choose the directions along which walkers move. In particular, both `zeus` and `emcee`/DEMC define a direction vector as the difference between two walkers from the complementary ensemble, thus two walkers are required to define a direction. On the other hand, `emcee`/AIES requires only a single walker from the complementary ensemble as the direction is defined by the difference between the updated walker and the complementary one. This stark contrast between the way those samplers choose their direction vectors lies at the heart of the difference in the computational cost of `emcee`/AIES as compared to `zeus` and `emcee`/DEMC in the limit of low number of walkers. In order to dive a little deeper into this, we can compute the exact number of possible directions for all three methods. Since `emcee`/AIES requires only a single walker from the complementary ensemble the number of available directions is equal to the size of the complementary ensemble. On the other hand, `zeus`'s and `emcee`/DEMC's requirement for a pair of walkers means that the number of available directions is equal to $\binom{n}{2}$, meaning the 2−combination from a set of n walkers that comprise the complementary ensemble. Clearly, as shown in Figure 11.7, the latter increases faster with the size of the complementary ensemble, thus explaining the larger variety of possible directions available in the case of `zeus` and `emcee`/DEMC compared to `emcee`/AIES.

The discussion so far was about the computational cost of convergence in terms of the number of model evaluations for a single walker. Of course, the ensemble of walkers consists by definition of more than a single walker. Therefore, in order to compute the total number of model evaluations required until the ensemble converges we need to multiply the results of the single walker with the total number of walkers. Those results are presented in the right panel of Figure 11.6. From this plot we can see that both `zeus` and `emcee`/DEMC converge faster when the number of walkers is close to its minimum value i.e. $2 \times D$. `emcee`/AIES on the other hand prefers a higher number of walkers (i.e. $32 \times D$) in order to overcome the sparsity of available directions in the limit of low number of walkers. This, however, means that even if we choose the optimal number of walkers for `emcee`/AIES it would still converge slower than either `zeus` or `emcee`/DEMC. Furthermore, we cannot know *a priori* the optimal number of walkers for `emcee`/AIES unlike for `zeus` and `emcee`/DEMC in which the optimal size of the ensemble is close to $2 \times D$. Finally, the faster convergence of `zeus` compared to `emcee`/DEMC can be attributed to the local adaptation that the former performs by extending the length of the slice and thus allowing larger steps in parameter space.

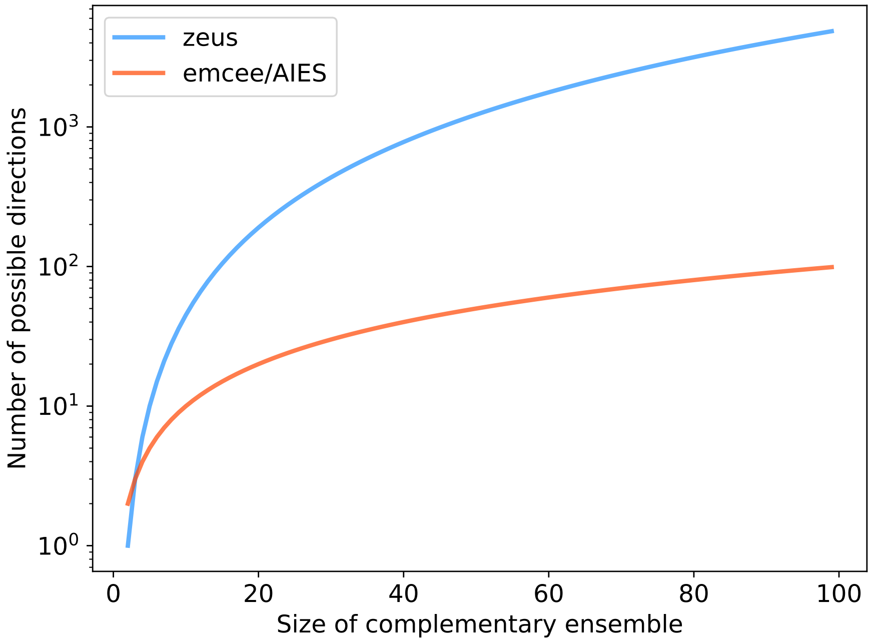

Figure 11.7: The figure shows the number of possible directions along which `zeus` and `emcee`/AIES can propose new samples as a function of the number of walkers in the complementary ensemble. `emcee`/DEMC exhibits the same number of proposals as `zeus` and it is not plotted here. `zeus` has a much higher variety of possible directions compared to `emcee`/AIES for any given number of walkers, assuming that that number is greater than 2.

The ring distribution

The ring distribution defined as

$$\ln P(x) = -\left[\frac{(x_n^2 + x_1^2 - a)^2}{b}\right]^2 - \sum_{i=1}^{n-1}\left[\frac{(x_i^2 + x_{i+1}^2 - a)^2}{b}\right]^2, \tag{11.3}$$

where $a = 2$, $b = 1$ and n is the total number of parameters; this is an artificial target distribution that exhibits strong non-linear correlations between its parameters. This aspect of the ring distribution allows us to demonstrate the locally adaptive nature of `zeus`. Whereas `emcee`/AIES and `emcee`/DEMC use a single global proposal scale for all regions of the parameter space, `zeus` has the ability to adjust its proposal scale locally by expanding the slice appropriately. As expected, this will allow `zeus` to sample efficiently even in cases in which strong non-linear correlations are present. Looking at Figure 11.8 one can see that `zeus` manages to generate multiple samples efficiently even in high dimensions. On the other hand, `emcee`/AIES and `emcee`/DEMC do not efficiently produce valid proposals: for `emcee`/AIES this leads to an inefficient random walk, characterised by small steps; for `emcee`/DEMC the acceptance rate almost vanishes beyond $D = 2$. The expected squared jump distance of each method for the case of $D = 25$ is shown in Table 4. It is important to note here that out of the three samplers only `zeus` manages to converge in all three cases (i.e. in 2, 10 and 25 dimensions). `emcee`/AIES and `emcee`/DEMC on the other hand converge successfully only in 2 dimensions.

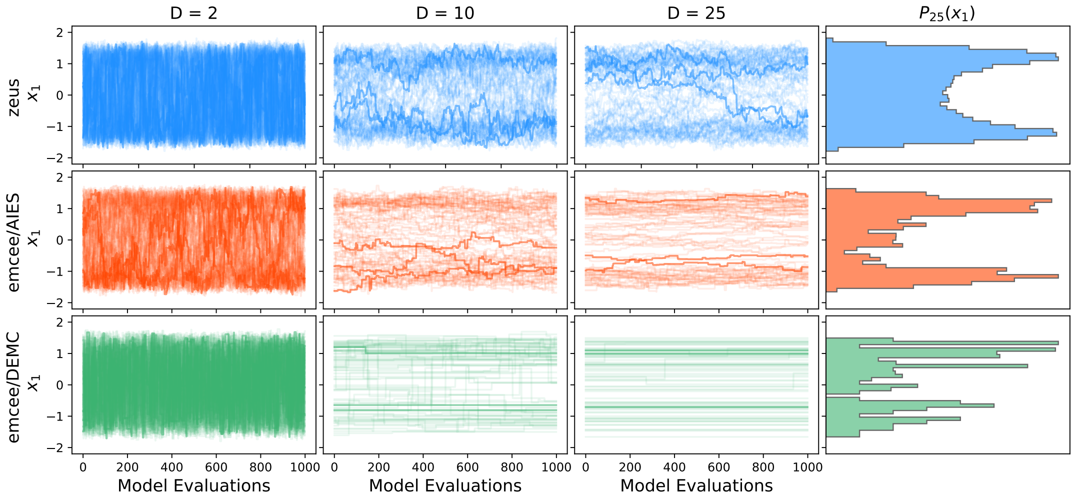

Figure 11.8: The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of the ring target distribution in 2, 10 and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. One can notice here that in 10 and 25 dimensions both `emcee` methods mix very slowly. In the 25-dimensional case almost all of `emcee`/DEMC's walkers are unable to move and the autocorrelation time is effectively infinite.

To explain this result one only has to look at the distribution of walker steps of the different methods at Figure 11.9. `zeus`'s steps extend to large distances in parameter space whereas most of `emcee`/AIES's and `emcee`/DEMC's steps are rejected (i.e. shown as zero in the histogram). We can see that `emcee`/DEMC manages to perform some long distance steps but those are few and there is almost nothing in between. It is clear from this and the previous toy examples that the $\gamma = 2.38/\sqrt{D}$ scaling of `emcee`/DEMC's scale factor does not generalise well beyond the Gaussian case.

The two-component Gaussian mixture distribution

One other important aspect of astronomical posterior distributions is the fact that many of them exhibit multiple peaks. Multimodality can arise either from non-linear models or sparse and uninformative data. In either case, multimodal target distributions present a formidable challenge for most MCMC methods. Perhaps the simplest example of such a distribution is the two-component Gaussian mixture. In this example we will position the two,

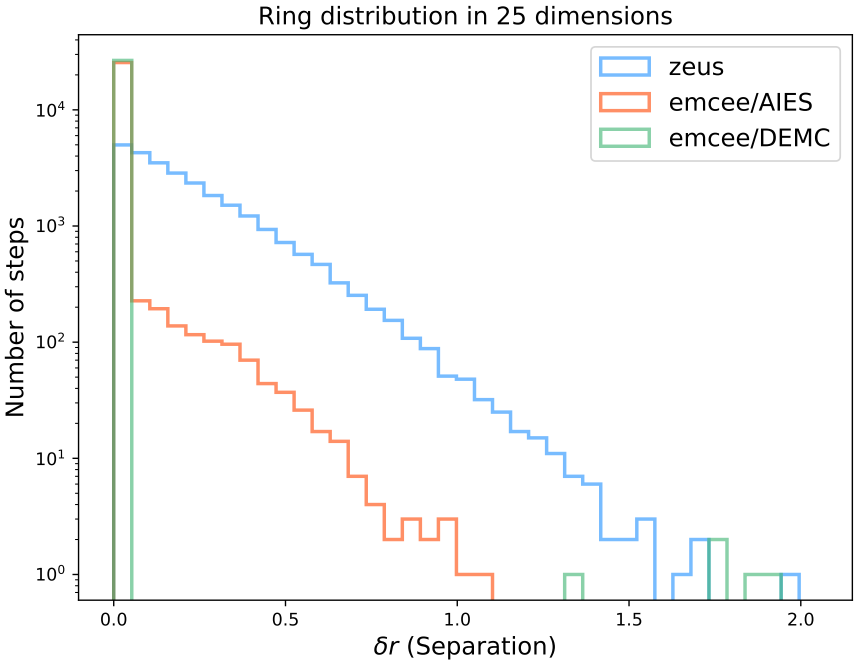

Figure 11.9: This figure shows the distribution of step sizes of walkers for the three different samplers in the case of a ring target distribution in $D = 25$. It is important to note here that both `emcee` algorithms exhibit a peak at zero separation; `zeus` on the other hand does not. The existence of the zero-peak in `emcee` is due to the high number of rejected proposals (i.e. low acceptance rate).

equal-mass, components at $-\mathbf{0.5}$ and $+\mathbf{0.5}$ respectively with standard deviation of 0.1. Sampling from multimodal distributions requires two types of proposals, local proposals that sample different modes individually and global proposals that transfer walkers from one mode to the other. For this reason we will make use of `zeus`'s `GlobalMove` that uses a Dirichlet Process Gaussian Mixture model of the ensemble to efficiently propose between-mode and within-mode steps.

As seen in Figure 11.10, `zeus`'s walkers manage to move from one mode to the other frequently enough for mixing to be efficient even in the $D = 25$ case. Out of `emcee`/AIES and `emcee`/DEMC, only the latter proposes valid steps from one mode to the other in the $D = 2$ case. As for the $D = 25$ case, one can see in Figure 11.11 that `zeus`'s walkers perform numerous jumps whereas `emcee`'s walkers are unable to do so. The ability of the walkers to jump from mode to mode is of paramount importance if we want to sample correctly from the target distribution. Lack of such proposals will lead to an improper probability mass ratio between the two modes and thus biased inference. The expected squared jump distance of each method for the case of $D = 25$ is shown in Table 4.

Clustering-based proposals have also been applied to MH-type ensemble MCMC methods but as shown in Karamanis & Beutler (2021), they fail to

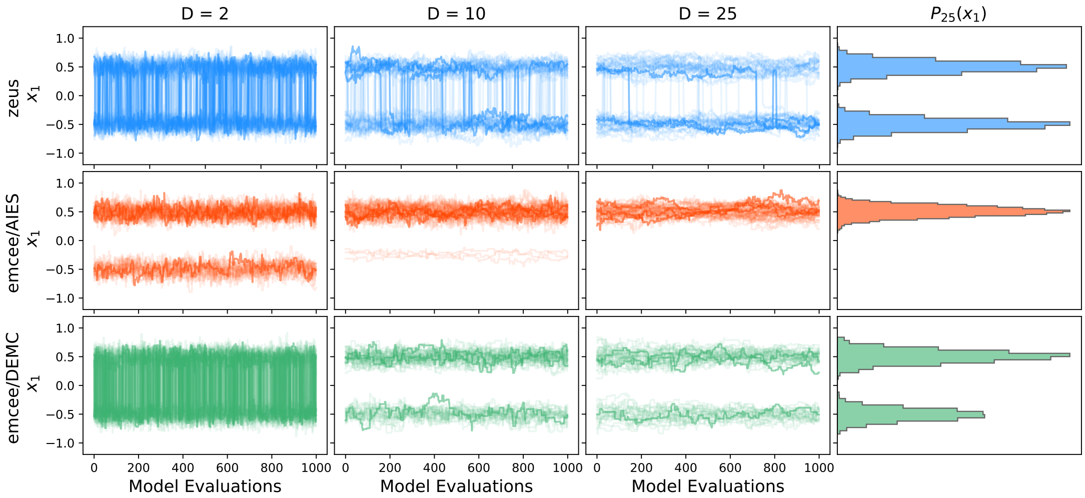

Figure 11.10: The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of a two-component Gaussian mixture target distribution in $2, 10$ and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. Whereas all three samplers make valid within-mode proposals, it is only zeus that manages to perform between-mode jumps and thus sample correctly from the target distribution in the 10 and 25-dimensional cases. Between-mode jumps are paramount in order to distribute the probability mass correctly between different modes.

generate valid proposals in problems with moderate number of dimensions. The reason is, as discussed in Section 11.3, that MH has to propose a valid point in the other mode. In other words, whereas Ensemble Slice Sampling only needs to determine the direction of the other mode relative to the chosen walker correctly, MH needs to guess both the direction and the distance, a task that rapidly becomes very hard as the number of dimensions rises.

The Student's t-distribution

The fourth toy example tests the case in which the target distribution is characterised by heavy-tails. In order to demonstrate zeus's ability to sample efficiency is such cases we chose to use the multivariate Student's t-distribution with 2 degrees of freedom. The aforementioned density exhibits heavier tails than a normal distribution which means that it is more likely to produce samples that are far away from the mean. The t-distribution arises when estimating the mean of a normally distributed sample with unknown

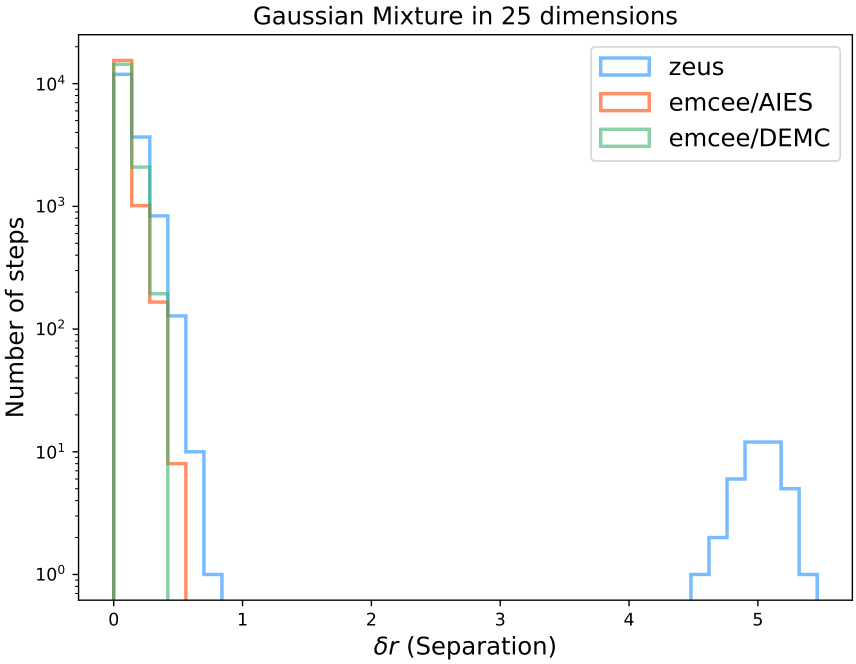

Figure 11.11: This figure shows the distribution of step sizes of walkers for the three different samplers in the case of a two-component Gaussian mixture target distribution in $D = 25$. It is important to note here that both `emcee` algorithms exhibit a peak at zero separation; `zeus` on the other hand does not due to its non-rejection basis.

standard deviation and small size. The probability density function of a p–dimensional Student's t-distribution with ν degrees of freedom is given by:

$$P(x) = \frac{\Gamma[(\nu+p)/2]}{\Gamma(\nu/2)\nu^{p/2}\pi^{p/2}|\Sigma|^{1/2}} \exp\left[1 + \frac{1}{\nu}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]^{-\frac{(\nu+p)}{2}}, \qquad (11.4)$$

where Σ is the $p \times p$ positive semi-definite shape matrix and $\boldsymbol{\mu}$ is the mean vector.

We sampled the above distribution using the three samplers in 2, 10 and 25 dimensions respectively as shown in Figure 11.12. The diagonal elements of shape matrix Σ were set to 1 and the off-diagonal elements to 0.95. The mean vector $\boldsymbol{\mu}$ was set to $\mathbf{0}$. All three samplers managed to sample efficiently in 2, 10 and 25 dimensions as shown in Figure 11.12 and Table 4. Overall, `zeus` was the most efficient method with `emcee`/AIES being second and `emcee`/DEMC last. One can see from Figure 11.13 that the distributions of steps of `zeus` and `emcee`/AIES are very similar whereas that of `emcee`/DEMC is substantially shorter. Unlike the previous toy examples in which the proposal strategy of `emcee`/AIES was causing it to overshoot the bulk of posterior mass, in the case of the heavy-tailed t-distribution more proposals are accepted. On the other hand, `emcee`/DEMC's proposals which are optimised for Gaussian targets are more conservative in the case of the t-distribution and they do not extend far away. As also demonstrated in the

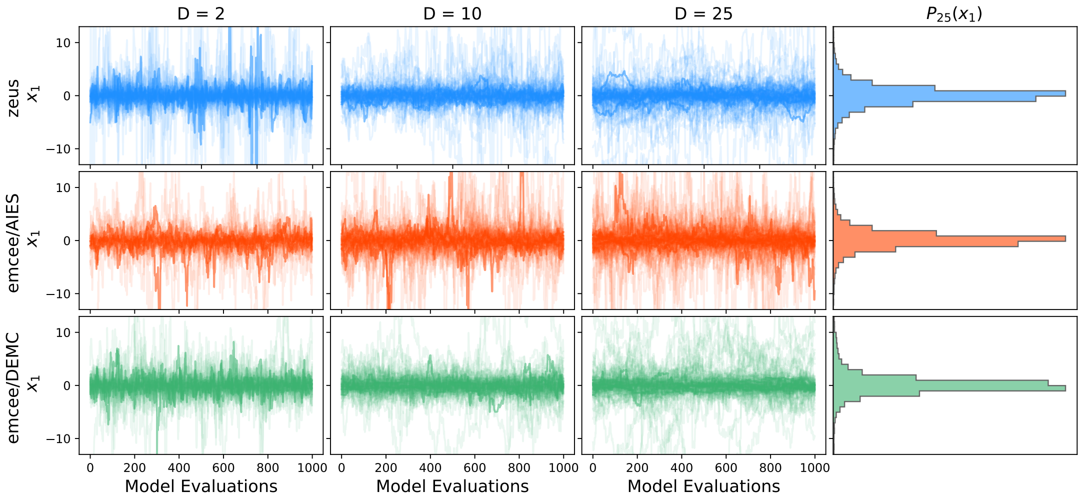

Figure 11.12: The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of the Student's *t*-distribution with 2 degrees of freedom in 2, 10 and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case.

previous toy examples, the locally adaptive nature of `zeus` allows it to perform efficient proposals that span large distances in parameter space.

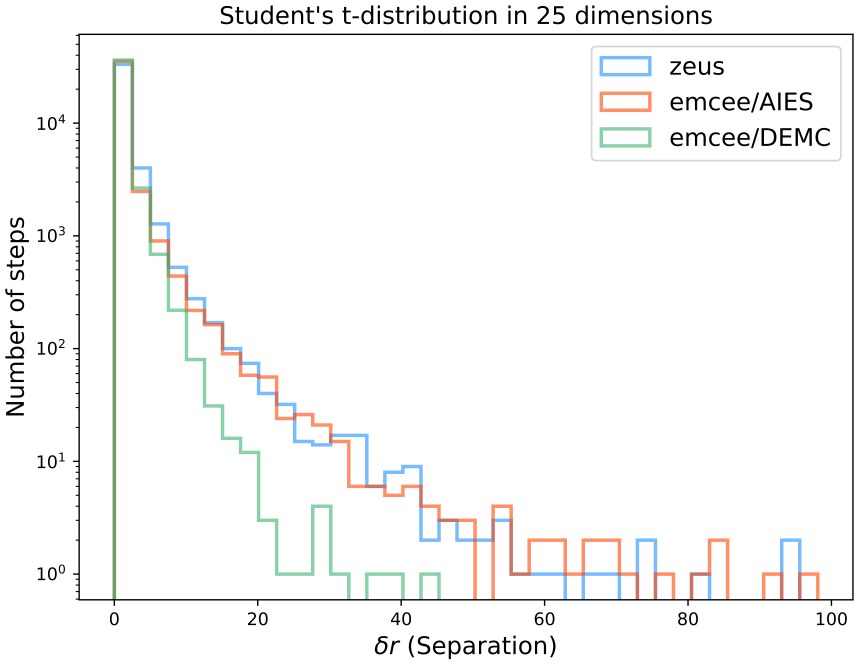

Figure 11.13: This figure shows the distribution of step sizes of walkers for the three different samplers in the case of the Student's *t*-distribution with 2 degrees of freedom in $D = 25$. `zeus` and `emcee`/AIES exhibit similar distributions whereas `emcee`/DEMC performs shorter steps.

The truncated normal distribution

The fifth and final toy example tests the case in which the target distribution is bounded from below or above. We chose to employ a truncated normal distribution similar to the one used in the first toy example, with the additional constraint being that $x > 0$. This effectively introduces a hard boundary along all dimensions. One of the reasons that we study this distribution is to assess the bias introduced by the presence of the hard boundary.

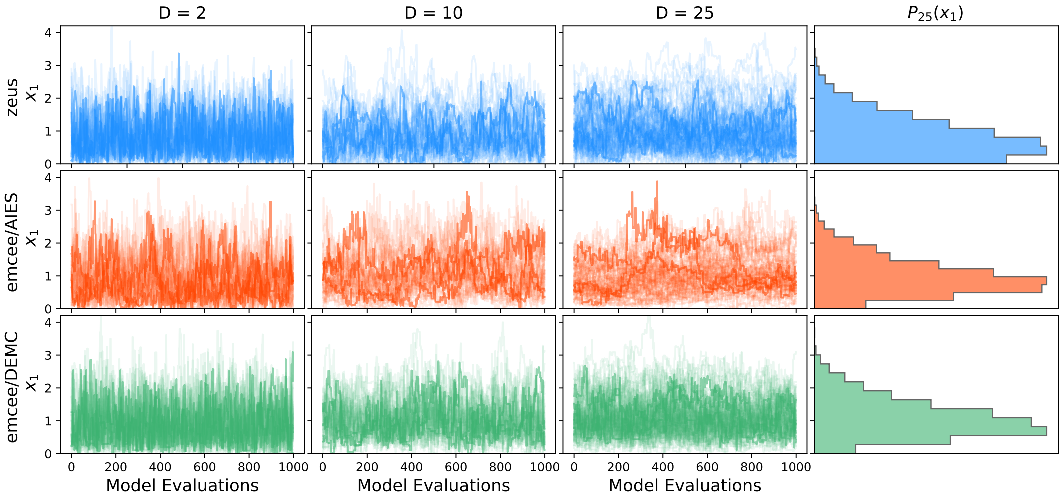

Figure 11.14: The figure shows numerical results (i.e. walker trajectories/chains for the first parameter) demonstrating the performance of the three ensemble MCMC methods in the case of the truncated normal distribution in 2, 10 and 25 dimensions respectively. The last column illustrates the 1-D marginal posterior corresponding to the first parameter x_1 estimated directly from the samples for the 25-dimensional case. `zeus` exhibits the least amount of bias near the hard boundary at zero compared to `emcee`/AIES and `emcee`/DEMC.

We sampled the above distribution using the three samplers in 2, 10 and 25 dimensions respectively as shown in Figure 11.14. The diagonal elements of the covariance matrix were set to 1 and the off-diagonal to 0.95. The mean vector $\boldsymbol{\mu}$ was set to $\mathbf{0}$. All three samplers managed to sample efficiently in 2, 10 and 25 dimensions as shown in Figure 11.14 and Table 4. Overall, `zeus` was the most efficient method with `emcee`/DEMC being second and `emcee`/AIES last. One can see from Figure 11.15 that the distributions of steps of `zeus` and `emcee`/AIES are very similar whereas that of `emcee`/AIES is slightly shorter. As shown in the right panels of Figure 11.14 `zeus` exhibits the least amount of bias compared to `emcee`/AIES and `emcee`/DEMC. In practical astronomical examples however, only one or two parameters would usually be bounded (e.g. the neutrino mass in galaxy clustering analyses) and thus unbiased sampling would be easier to perform by either of the three samplers.

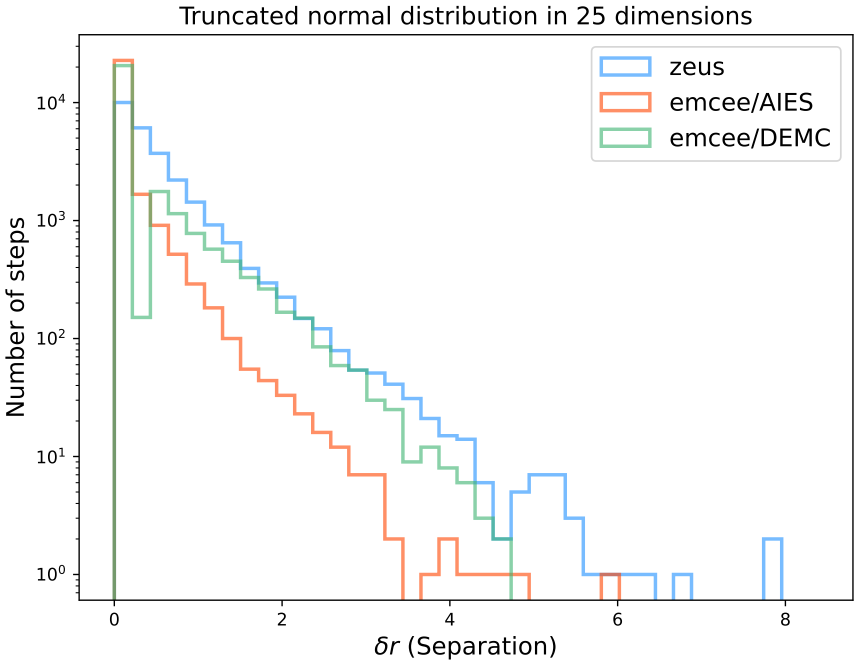

Figure 11.15: This figure shows the distribution of step sizes of walkers for the three different samplers in the case of the truncated normal distribution in $D = 25$. `zeus` and `emcee`/AIES exhibit similar distribution whereas `emcee`/DEMC performs shorter steps.

11.3.2 Real astronomical analyses

The previous section employs toy examples in order to exhibit various scenarios that might emerge during sampling, and shows how `zeus` is better equipped to handle them. To demonstrate the efficiency of `zeus` compared to other samplers in realistic target distributions, we chose two common astronomical inference problems as the testing ground. Those are the cases of baryon acoustic oscillation (BAO) parameter inference and exoplanet parameter estimation.

We used the same three samplers in our comparison, namely `emcee` with AIES and DEMC, and of course `zeus`. We performed three distinct tests:

- The first test was to estimate the efficiency for each sampler, defined as the number of independent samples produced per log-likelihood evaluation. To this end, we ran the MCMC procedure 5 times for each sampler and computed the mean efficiency using the estimated autocorrelation time of the chains. The autocorrelation time was estimated using the method presented in Karamanis & Beutler (2021).

- The second test relates to the convergence rate of the three algorithms. As a measure of convergence rate, we adopt the inverse of the number of iterations required until all the convergence criteria specified below are met. In order to estimate the mean convergence rate we ran the sampling procedure 40 times for each sampler initialising the walkers close to the *Maximum a Posteriori* (MAP) estimate.

- Finally, we tested the sensitivity of the samplers to the initial conditions by running 40 realisations with the walkers initialised from a small sphere (of radius 10^{-4}) around a randomly chosen point in the prior volume, counting how many of those attempts led to converged chains before a predetermined number of likelihood evaluations.

To determine whether a chain has converged we used four different metrics: the Gelman-Rubin split-R statistic (Gelman, Carlin, et al., 2013; Gelman, Rubin, et al., 1992) using four independent ensembles of walkers; the Geweke test (Geweke, 1992); a minimum length of the chain as a multiple of the integrated autocorrelation time (IAT); as well as an upper bound on the rate of change of the IAT. Only the second half of the chains was used to evaluate the aforementioned criteria. The number of walkers used in both examples was close to the minimum value of $2 \times D$ as specified below. As we will discuss in Section 11.4 this often leads to faster convergence.

Cosmological inference

The particular inference problem that we consider here is that of the anisotropic BAO parameter inference using estimates of the galaxy power spectrum. The data we used comes from the 12th data release (DR12) of the high-redshift North Galactic Cap (NGC) sample as observed by the Sloan Digital Sky Survey (SDSS) (Daniel J. Eisenstein et al., 2011) Baryon Oscillation Spectroscopic Survey (BOSS) (Dawson et al., 2013). Our analysis follows closely that of Beutler, Seo, et al. (2017) with the difference that we chose not to fix any parameters and fit the hexadecapole multipole of the power spectrum as well as the monopole and quadrupole. Those choices were made solely to render the problem more challenging. Indeed the inclusion of the hexadecapole does not contribute any additional constraining power for the data that we used. However, such extended models will prove useful when analysing data from larger galaxy surveys such as DESI (DESI Collaboration et al., 2016). In terms of Bayesian inference, the problem has 22 free parameters. The results of our analysis are consistent with those of Beutler, Seo, et al. (2017). We used weakly informative flat (uniform) priors for all parameters except for the two scaling parameters, $\alpha_{\parallel}$ and $\alpha_{\perp}$ for which we used normal (Gaussian) priors. We used 50 walkers in total.

In terms of efficiency, `zeus` generates at least 5 effectively independent samples for each one generated by `emcee`/DEMC and at least 9 for each one generated by `emcee`/AIES factoring in the different computational costs of the methods. As for the convergence rate, `zeus` converges more than 3 times faster than either `emcee` variant. Finally, we found that `zeus` is less sensitive to the initialisation than either of the other two methods. In particular, out of the 40 tests conducted with different initialisation, `zeus` converged 36 times, `emcee`/DEMC 14 times and `emcee`/AIES 7 times prior to the predetermined maximum number of likelihood evaluations (i.e. 5×10^6 in this case). The aforementioned results are presented in detail in Table 5. The 1-D and 2-D marginal posterior distributions are shown in Figure 11.16 demonstrating the agreement between the three methods[4].

Table 5: The table shows a comparison of `emcee`/AIES, `emcee`/DEMC and `zeus` in terms of the inverse efficiency (i.e. reciprocal of the number of independent samples per model evaluation or the autocorrelation time estimate times the average number of model evaluations per iteration per walker), the convergence cost (i.e. number of model evaluations until convergence) and the convergence fraction (i.e. fraction of converged chains for given maximum number of model evaluations).

	`emcee`/AIES	`emcee`/DEMC	**zeus**
Cosmological inference			
efficiency^{-1}	12140	6750	**1320**
convergence cost	24×10^5	22×10^5	$\mathbf{6.6 \times 10^5}$
convergence fraction	7/40	14/40	**36/40**
Exoplanet inference			
efficiency^{-1}	1386	338	**47**
convergence cost	36.0×10^2	17.1×10^2	$\mathbf{4.8 \times 10^2}$
convergence fraction	23/40	29/40	**38/40**

Exoplanet inference

Another common application of MCMC methods in astronomy is the problem of exoplanet parameter inference through modelling of Keplerian orbits and radial velocity time series data. In this section we demonstrate the performance of `zeus` using a two-planet model with 14 free parameters and real data from the K2-24 (EPIC-203771098) extrasolar system (Petigura et al., 2016) that is known to host two exoplanets. We used the popular `Python` package `RadVel` (Fulton et al., 2018) for the Keplerian modelling of the planetary orbits. The results of our analysis are consistent with published constraints for the aforementioned extrasolar system (Petigura et al., 2016). We used 30 walkers in total for sampling.

We performed the same suite of tests as in the cosmological inference case. In terms of efficiency, `zeus` generates more than 7 independent samples

---

[4] No upper limit on the number of likelihood evaluations or iterations was used for this run and convergence was diagnosed using all the metrics that we introduced.

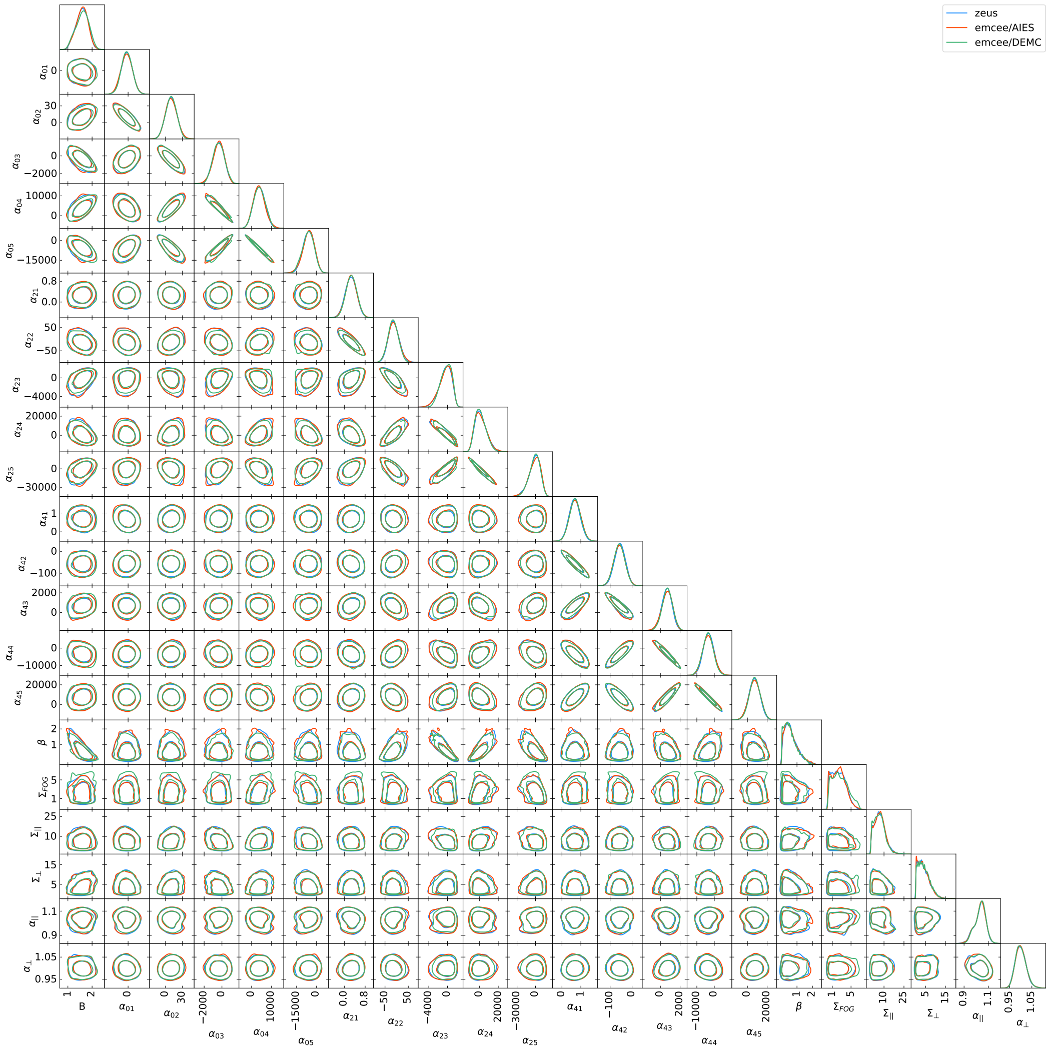

Figure 11.16: A corner plot showing the 1-D and 2-D marginalised posteriors for the 22-parameter Baryon Acoustic Oscillation model as produced by the three different ensemble MCMC methods.

per each one generated by `emcee`/DEMC and more than 29 independent samples per each one generated by `emcee`/AIES. As for the convergence rate, `zeus` converges 7.5 times faster than `emcee`/AIES and 3.5 faster than `emcee`/DEMC on average. Finally, we found again that `zeus` is less sensitive to the specific initialisation of the walkers. In particular, out of the 40 tests conducted with different initialisation, `zeus` converged 38 times, `emcee`/DEMC 29 times and `emcee`/AIES 23 times prior to the predetermined maximum number of likelihood evaluations (i.e. 5×10^3 in this case). Detailed results about the values of the used metrics are shown in Table 5. The 1-D and 2-D marginal posterior distributions are shown in Figure 11.17, demonstrating the agreement between the three methods.

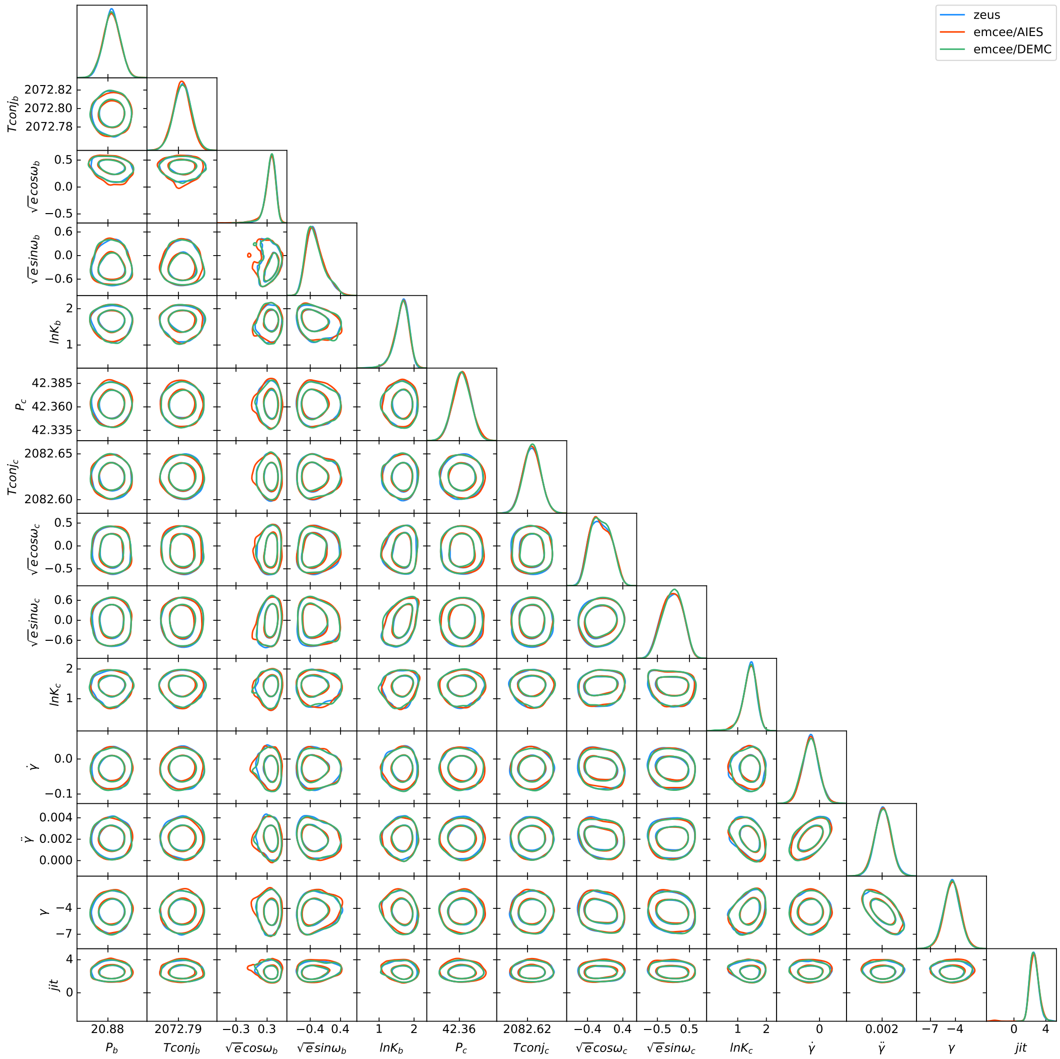

Figure 11.17: A corner plot showing the 1-D and 2-D marginalised posteriors for the 14-parameter radial velocity model as produced by the three different ensemble MCMC methods.

11.4 DISCUSSION

Following the analysis we conducted in Section 11.3 using the normal distribution there are two important questions that need to be answered about the initialisation of the walkers. First, how many walkers are necessary and, second, how to choose the initial positions of the walkers. Although there are many ways of answering those questions and there is no consistent solution that works for all target distributions, we will try to provide some general rules and heuristics to help ease the task of choosing the number and initial positions of the walkers for most cases.

Let us first discuss the effect of the number of the walkers on the general performance of `zeus`. Naively, one might expect that the minimum number of walkers should be $D + 1$, where D is the number of dimensions. However, the ensemble splitting technique, which was introduced in Section 11.3 to render the algorithm parallelisable, requires at least $2 \times D$ walkers in order to produce 2 linearly independent samples. If a smaller number is chosen then the walkers can be trapped in a lower–dimensional hyper–plane of the parameter space, being unable to sample properly and leading to erroneous results. Although there is no upper bound on the number of walkers, we recommend to use between two to four times the number of dimensions. The reason is that increasing the number of dimensions can increase the cost of the burn-in period as we explained in detail in Section 11.3. Ideally, one wants to use the minimum number (or close to that) of walkers until the burn-in period is over and then increase the number of walkers to rapidly produce a great number of independent samples. It is also worth noting that in cases in which either non-linear correlations or multiple modes are present it is recommended to use more walkers (e.g. 4-8 times the number of parameters for a bimodal target distribution).

As for the initialisation of the walkers, there are many ways to choose their starting positions ranging from prior sampling to more localised initial positions. Empirical tests indicate that the latter often outperforms the former (i.e. leads to shorter burn-in periods). That is not surprising since the total probability of a prior-sampled initialisation can be very small when the number of parameters is high. In particular we found that initialising the walkers from a tight region in parameter space (i.e. normal distribution with small variance) consistently leads to good performance. For low to moderate dimensional problems initialising the walkers from a tight ball around the *Maximum A Posteriori* (MAP) estimate can substantially reduce the burn-in period (Foreman-Mackey, Hogg, et al., 2013).

Finally, while `emcee`/AIES and `emcee`/DEMC can sample acceptably from most target distributions with $D \lesssim 20$, the efficient scaling of `zeus` with the number of parameters allows us to extend this range and efficiently test more complicated models (Karamanis & Beutler, 2021). Like most gradient-free methods, `zeus` will fail to sample efficiently in very high dimensional problems in which $D = \mathcal{O}(10^2)$. In such cases, more sophisticated algorithms (e.g. tempering, block updating, Hamiltonian dynamics etc.) need to be used (C. P. Robert et al., 2018).

11.5 CONCLUSIONS

The aim of this project was to develop a tool that could facilitate Bayesian parameter inference in computationally demanding astronomical analyses and tackle the challenges posed by the complexity of the models and data that are often used by astronomers. To this end, we introduced `zeus`, a parallel, general-purpose and gradient-free `Python` implementation of Ensemble Slice Sampling.

After introducing the method in Section 11.2, we thoroughly demonstrated its performance compared to two popular alternatives (i.e. `emcee` with affine-invariant ensemble sampling and differential evolution Metropolis) using a variety of artificial and realistic target distributions in Section 11.3. The artificial toy examples helped to shed light on the general behaviour of the samplers in target distributions characterised by linear and non-linear correlations as well as multimodal densities. When compared to `emcee`/AIES and `emcee`/DEMC in the problems of Baryon Acoustic Oscillation parameter inference and exoplanet radial velocity fitting, `zeus` consistently converges faster (i.e. its burn-in is shorter by a factor of at least 3), it is less sensitive to the initialisation of the walkers and generates substantially more independent samples per likelihood evaluation (i.e. approximately ×9 and ×29 speed-up compared to `emcee`/AIES in the cosmological and exoplanet examples, respectively).

We have shown that `zeus` performs similarly or better than existing MCMC methods in a range of problems. We hope that `zeus` will prove useful to the astronomical and cosmological community by complementing existing approaches and facilitating the study of novel models and data over the coming years. `zeus` is publicly available at `https://github.com/minaskar/zeus` with detailed documentation and examples that can be found at `https://zeus-mcmc.readthedocs.io`.

12 PRECONDITIONED MONTE CARLO

This chapter presents *Preconditioned Monte Carlo* which is the main contribution introduced in the paper titled *Accelerating astronomical and cosmological inference with Preconditioned Monte Carlo* that was submitted for publication in the journal *Monthly Notices of the Royal Astronomical Society* in July 2022 (Karamanis, Beutler, Peacock, et al., 2022). The content of the chapter is almost identical to that included in the aforementioned publication with the exception of minor text and figure formatting differences.

We introduce *Preconditioned Monte Carlo* (PMC), a novel Monte Carlo method for Bayesian inference that facilitates efficient sampling of probability distributions with non–trivial geometry. PMC utilises a *Normalising Flow* (NF) in order to decorrelate the parameters of the distribution and then proceeds by sampling from the preconditioned target distribution using an adaptive *Sequential Monte Carlo* (SMC) scheme. The results produced by PMC include samples from the posterior distribution and an estimate of the model evidence that can be used for parameter inference and model comparison respectively. The aforementioned framework has been thoroughly tested in a variety of challenging target distributions achieving state–of–the–art sampling performance. In the cases of *primordial feature* analysis and *gravitational wave* inference, PMC is approximately 50 and 25 times faster respectively than *Nested Sampling* (NS). We found that in higher dimensional applications the acceleration is even greater. Finally, PMC is directly parallelisable, manifesting linear scaling up to thousands of CPUs. An open–source `Python` implementation of PMC, called `pocoMC`, is publicly available at `https://github.com/minaskar/pocomc`.

12.1 INTRODUCTION

Modern astronomical and cosmological analyses have largely adopted the framework of *Bayesian probability* for tasks of parameter inference and model comparison. In the Bayesian context, the *posterior probability distribution* $\mathcal{P}(\theta) = P(\theta|\mathcal{D},\mathcal{M})$, meaning the probability distribution of the parameters

θ of a model $\mathcal{M}$, given some data $\mathcal{D}$ and the model $\mathcal{M}$ is given by Bayes' theorem:

$$\mathcal{P}(\theta) = \frac{\mathcal{L}(\theta)\pi(\theta)}{\mathcal{Z}}\,, \tag{12.1}$$

where $\mathcal{L}(\theta) = P(\mathcal{D}|\theta, \mathcal{M})$ is the *likelihood function*, $\pi(\theta) = P(\theta|\mathcal{M})$ is the *prior* probability distribution, and $\mathcal{Z} = P(\mathcal{D}|\mathcal{M})$ is the *model evidence* or *marginal likelihood* that acts as a normalisation constant for the posterior probability distribution. For a detailed introduction to Bayesian probability theory we refer the reader to Gregory (2005), E. T. Jaynes (2003), and MacKay (2003) and the reviews Sharma (2017) and Trotta (2017) for its use in astronomy and cosmology.

In tasks of parameter inference, the goal is to infer the values of physical and nuisance parameters from the data along with the respective uncertainties. Mathematically, this is formulated as the problem of estimating expectation values (e.g. mean values, standard deviations, 1–D and 2–D marginal posterior distributions, etc.) that correspond to high–dimensional integrals over the posterior probability density. During the past two decades, *Markov chain Monte Carlo* (MCMC) has been established as the standard computational tool for the calculation of such integrals (see e.g. (Speagle, 2019) for a review). MCMC methods generate a sequence of correlated samples, called a Markov chain, that are distributed according to the posterior probability distribution. Those samples can then be used in order to numerically estimate expectation values. Examples of MCMC software implementations in the astronomical and cosmological community are *emcee* (Foreman-Mackey, Hogg, et al., 2013) and *zeus* (Karamanis, Beutler & Peacock, 2021).

Most modern MCMC methods are based upon the *Metropolis–Hastings* (MH) paradigm that consists of two steps (Hastings, 1970; Metropolis et al., 1953). In the first step, known as the *proposal step*, a new sample is drawn from a known proposal distribution that depends only on the position of the current sample/state. The validity of the new sample, and thus the decision on whether to add it or not to the Markov chain, is determined in the second step, known as the *acceptance step*, which takes into account the new sample, the old sample (i.e. current state) and the proposal distribution that was used in order to generate it. Arguably, the most important element of an efficient MCMC method is the choice of the proposal distribution. The degree to which the proposal distribution characterises the local geometry of the target distribution determines the sampling efficiency (i.e. rate of effectively independent samples) of the method. Unfortunately, choosing or tuning the optimal proposal distribution for a given target distribution is not an easy task. However, certain optimal proposal distributions are known for specific classes of target distributions. For instance, in the case of a normal or Gaussian target distribution, using a normal proposal distribution of the form $\mathcal{N}(\theta, 2.38^2\Sigma/D)$, where Σ is the covariance matrix of the target density, θ is the current state of the chain, and D is the number of dimensions yields the maximum sampling efficiency scheme with acceptance rate of 23.4% in the acceptance step of MH (Gelman, Gilks, et al., 1997). Alternatively, one can use a simpler proposal distribution of the form $\mathcal{N}(u, 1)$ where $u = f(\theta)$ and f is a suitable transformation. In this case, $f(\theta)$ is proportional to $L^{-1}\theta$ where L is the lower triangular matrix of the *Cholesky decomposition* of the covariance matrix $\Sigma = LL^T$. In other words, assuming that a suitable transformation can be found, one can increase the sampling efficiency of an MCMC method. This notion of preconditioning is central for the discussion that will follow in the next section.

In recent years, the need for higher sampling efficiency when the correlations between parameters are strong enough or the posterior exhibits multiple modes, as well as the required computation of the model evidence $\mathcal{Z}$ for model comparison tasks, motivated the development of more advanced sampling methodologies and algorithms. One very popular approach is the *Sequential Monte Carlo* (SMC) algorithm (Del Moral et al., 2006), which evolves a set of particles through a series of intermediate steps that bridge the gap between the prior distribution and the posterior distribution by geometrically interpolating between them. Another class of algorithms called *Nested Sampling* (NS) (Skilling, 2004) attempts to approach the problem of Bayesian computation from a slightly different perspective. Instead of evolving a set of particles though a series of geometrically–interpolated steps between prior and posterior distribution, NS splits the posterior distribution into many slices and attempts to sample each slice individually with an appropriate weighting scheme. Many popular versions and implementations of NS exist in the astronomical literature (Buchner, 2021; Feroz, M. Hobson, et al., 2009; WJ Handley et al., 2015; Speagle, 2020). Whereas both SMC and NS largely addressed the problem of multimodality, the performance of both methods is still very sensitive to the geometry of the target distribution, meaning the presence of strong non–linear correlations.

In this paper, we introduce *Preconditioned Monte Carlo* (PMC), a novel Monte Carlo method for Bayesian inference that extends the range of applications of SMC to target distributions with non–trivial geometry, strong non–linear correlations between parameters, and severe multimodality. PMC achieves this by first preconditioning, or transforming the geometry of the target distribution into a more manageable one using a generative model known as a *Normalising Flow* (NF) (Papamakarios, Nalisnick, et al., 2021), before sampling using a SMC scheme. M. Hoffman et al. (2019) used a NF to neutralise the bad geometry in Hamiltonian Monte Carlo (HMC) (Betancourt, 2017b) achieving great results in terms of sampling speed but unreliable estimates for unknown target distributions. Moss (2020) used a NF in order to parameterise efficient MCMC proposals and used it in the context of NS achieving a substantial speedup on several challenging distributions. Both of the aforementioned works used NFs as preconditioning transformations, the first in the context of HMC and the second in NS. In the context of NS and SMC, NFs have also been used as a sampling component of the algorithm (Albergo et al., 2019; Arbel et al., 2021; M. J. Williams et al., 2021), albeit not as a preconditioner but as a density from which new samples can be generated independently. The novelty of our work lies in the use of NFs as preconditioning transformations in the context of SMC, thus achieving both robustness and high sampling efficiency.

The structure of the rest of the paper is the following: Section 12.2 consists of a detailed presentation of the method, Section 12.3 includes a wide range of empirical tests that act as a demonstration of PMC's sampling performance, and Section 12.5 is reserved for the conclusions.

We also release a `Python` implementation of PMC, called `pocoMC`, which is publically available at `https://github.com/minaskar/pocomc` and detailed documentation with installation instructions and examples at `https://pocomc.readthedocs.io`. The code implementation is described in the accompanying paper (Karamanis, Nabergoj, et al., 2022).

12.2 METHOD

12.2.1 Sequential Monte Carlo

In this subsection, we will present a brief introduction to SMC algorithms. For a more detailed exposition, we refer the reader to Naesseth et al. (2019). We begin by first introducing the concept of *importance sampling*, which is crucial for understanding the function of SMC. Assuming that we have a target probability density $p(\theta)$ that we are able to evaluate up to an unknown multiplicative constant, then if we define another density $\rho(\theta)$, called the *importance sampling density*, such that $\rho(\theta) = 0 \Rightarrow p(\theta) = 0$ then the following relation holds for any expectation value:

$$\begin{aligned} \mathrm{E}_p[f(\theta)] &= \int f(\theta) w(\theta) \rho(\theta) d\theta \Big/ \int w(\theta)\rho(\theta) d\theta \\ &= \mathrm{E}_\rho[f(\theta) w(\theta)] / \mathrm{E}_\rho[w(\theta)] \,, \end{aligned} \tag{12.2}$$

for any function $f(\theta)$ where $w(\theta) = p(\theta)/\rho(\theta)$ are called importance weights. What is important here is that one can use samples from the importance density $\rho(\theta)$ in order to estimate the aforementioned expectation value without explicitly sampling from the target density $p(\theta)$.

A common measure of the quality of using the importance sampling density $\rho(\theta)$ to approximate $p(\theta)$ is the *Effective Sample Size*, defined as:

$$\mathrm{ESS} = \mathrm{E}_\rho[w(\theta)]^2 / \mathrm{E}_\rho[w(\theta)^2] \,. \tag{12.3}$$

Unfortunately, in high–dimensional scenarios it is difficult to find an appropriate importance sampling density that ensures that the ESS is high enough for the variance of the expectation value to be low. This is exactly the problem that SMC methods address.

SMC samplers extend the importance sampling procedure from the setting of two densities (i.e. importance sampling density and target density) to a sequence of T probability distribution densities $\{p_t\}_{t=1}^{T}$ in which each individual density p_t acts as the importance density for the next one in the series. The method proceeds by pushing a collection of N particles $\{\theta_t^k\}_{k=1}^{N}$ through this sequence of densities until the last one is reached. Each iteration of a SMC algorithm consists of three main steps:

1. **Mutation** – The population of particles is moved from $\{\theta_{t-1}^k\}_{k=1}^{N}$ to $\{\theta_t^k\}_{k=1}^{N}$ using a *Markov transition kernel* $K_t(\theta'|\theta)$ that defines the next importance sampling density
$$p_t(\theta') = \int p_{t-1}(\theta)K_t(\theta'|\theta)d\theta\,. \tag{12.4}$$
In practice, this step consists of running multiple short MCMC chains (i.e. one for each particle) to get the new states θ' starting from the old ones θ.

2. **Correction** – The particles are reweighted according to the next density in the sequence. This step consists of multiplying the current normalised weight W_t^k of each particle by the appropriate importance weight:
$$w_t(\theta_t) = p_t(\theta_{t-1})/p_{t-1}(\theta_{t-1})\,. \tag{12.5}$$

3. **Selection** – The particles are resampled according to their normalised weights W_t^k which are then set to $1/N$. This can be done using *multinomial resampling* or more advanced schemes. The purpose of this step is to eliminate particles with low weight and multiply the ones with high weights.

An important feature of the SMC method is that it allows for the unbiased estimation of the ratios of normalising constants

$$\mathcal{Z}_t/\mathcal{Z}_{t-1} = \sum_{k=1}^{N} W_{t-1}^k w_t(\theta_{t-1}^k)\,, \tag{12.6}$$

between subsequent densities, where $W_0^k = 1/N$. This is of paramount importance in cases in which the first density in the series corresponds to the prior distribution (i.e. with $\mathcal{Z} = 1$) and the last to the posterior distribution. Then, SMC methods can be used in order to compute the model evidence $\mathcal{Z}$ for tasks of model comparison.

In principle, there are arbitrary many ways to construct the sequence of densities $\{p_t\}_{t=1}^{T}$. A very common way to do so is to geometrically interpolate between two densities $\rho(\theta)$ and $p(\theta)$:

$$p_t(\theta) \propto \rho(\theta)^{1-\beta_t}p(\theta)^{\beta_t}, \quad t = 1, \dots, T \tag{12.7}$$

parameterised by a *temperature annealing* ladder:

$$\beta_1 = 0 < \beta_2 < \cdots < \beta_T = 1\,. \tag{12.8}$$

In the Bayesian context, a natural choice of geometric interpolation is from the prior $\pi(\theta)$ to the posterior:

$$p_t(\theta) \propto \pi(\theta)\mathcal{L}(\theta)^{\beta_t}, \qquad t = 1, \ldots, T \tag{12.9}$$

where $\mathcal{L}(\theta)$ is the likelihood function. In practice, it can still be difficult to choose a good temperature schedule. However, this can be done adaptively by selecting the next value of β_t such that the ESS is a constant α fraction of the number of particles N. Numerically, this can be done by solving

$$\left(\sum_{k=1}^{N} w_{t+1}^{k}(\beta_{t+1})\right)^2 \Big/ \sum_{k=1}^{N} w_{t+1}^{k}(\beta_{t+1})^2 = \alpha N\,, \tag{12.10}$$

the next β_{t+1} such that $\beta_t < \beta_{t+1} \leq 1$ using, for instance, the *bisection method.*

12.2.2 Normalising Flows

Normalising flows (NF) are generative models, which can facilitate efficient and exact density estimation (Papamakarios, Nalisnick, et al., 2021). They are based on the formula of change–of–variables $\theta = f(u)$ where u is sampled from a base distribution $u\, p_u(u)$ (i.e. usually a normal distribution). The NF is a bijective mapping between the base distribution $p_u(u)$ and the often more complex target distribution $p_\theta(\theta)$ that can be evaluated exactly using

$$p_\theta(\theta) = p_u(f^{-1}(\theta))\left|\det\left(\frac{\partial f^{-1}}{\partial \theta}\right)\right|, \tag{12.11}$$

where the Jacobian determinant is tractable.

NFs are usually parameterised by neural networks. However, neural networks are not in general invertible, and the Jacobian is not generally tractable. Thus special care needs to be taken when choosing the architecture of the neural network to ensure the invertability of the transformation and the tractability of the Jacobian. For instance, if the forward transformation is $\theta_i = u_i \exp(\alpha_i) + \mu_i$ and inverse transformation is $u_i = (\theta_i - \mu_i)\exp(-\alpha_i)$, where μ_i and α_i are constants, then it is straightforward to show that the Jacobian satisfies

$$\left|\det\left(\frac{\partial f^{-1}}{\partial \theta}\right)\right| = \exp\left(-\sum_i \alpha_i\right). \tag{12.12}$$

To this end, we chose to use the *Masked Autoregressive Flow* (MAF), which has been used many times successfully for density estimation tasks due to its superior performance and high flexibility compared to alternative models (Papamakarios, Pavlakou, et al., 2017). A MAF consists of many stacked

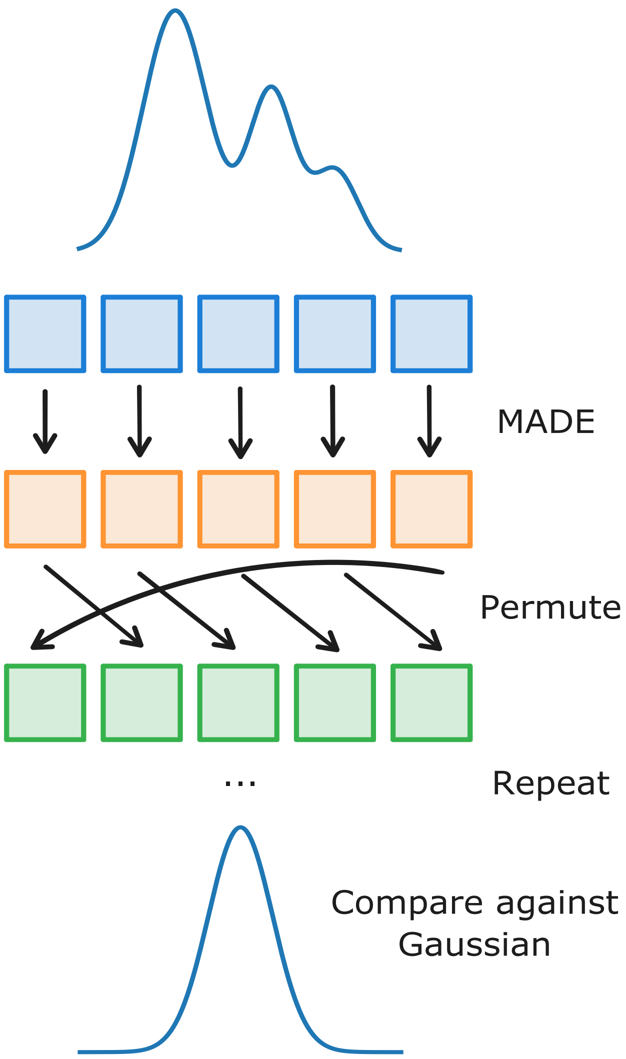

Figure 12.1: Illustration of the inference scheme of a *Masked Autoregressive Flow* (MAF). The arrows show the conditional dependence of the variables as well as the action of the *Masked Autoregressive Density Estimation* (MADE) layer. The input target probability density (top) is mapped into a multivariate normal distribution (bottom). A sequence of MADE layers and permutations is repeated multiple times in order to increase the flexibility of the flow.

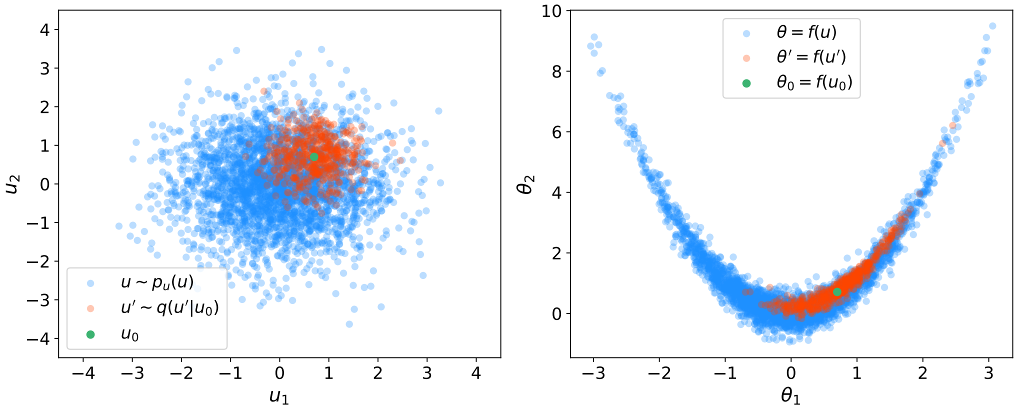

Figure 12.2: The figure illustrates the effect of preconditioning on the *Rosenbrock* distribution. The right panel shows samples (blue) from the true correlated distribution and the left panel shows samples (blue) from the preconditioned/transformed one. The orange samples in the left panel are drawn from a symmetric normal proposal distribution centred around the green point u_0 and they correspond to the respective orange points in the right panel. In other words, the transformed samples from the simple proposal in the left panel correspond to samples that capture the local geometry of the true target distribution in the right panel.

layers of a simpler generative model, called *Masked Autoregressive Density Estimator* (MADE) (Germain et al., 2015), with subsequent permutations of its outputs as shown in Figure 12.1. A MADE model decomposes a joint density $p(\theta)$ as a product of conditionals $p(\theta) = \prod_i p(\theta_i|\theta_{1:i-1})$ that ensures that any given value θ_i is only a function of the previous values thus maintaining the *autoregressive property*. When the MADE is based on an *autoencoder*, then *masking* is required in order to remove connections between different units in different layers, so as to preserve the aforementioned autoregressive property.

12.2.3 Preconditioning

Most *Markov chain Monte Carlo* (MCMC) methods struggle to sample efficiently from highly correlated or skewed target distributions. Often, transforming the parameters of the distribution before sampling, a process also known as *preconditioning*, using appropriate change–of–variable transformations, can help ameliorate this effect by disentangling the dependence between parameters. This is equivalent to choosing an appropriate proposal distribution in the context of *Metropolis–Hastings* (MH) methods. However, finding a valid transformation and selecting an appropriate proposal distribution is often difficult a priori; and there is no obvious way of making this joint choice in an optimal way. For instance, a linear transformation $\theta \longleftarrow L^{-1}\theta$

where L is the lower triangular matrix of the *Cholesky decomposition* of the *sample covariance* matrix $\Sigma = LL^T$ can remove only linear correlations and is not effective against non–linear ones. More sophisticated transformations, such as the use of the *chirp mass* and *mass ratio* instead of the individual black–hole masses in gravitational wave astronomy requires expert knowledge that is problem–specific.

The *Metropolis acceptance criterion* employed by MH methods in order to maintain detailed balance is

$$\alpha = \min\left(1, \frac{p_\theta(\theta')q(\theta|\theta')}{p_\theta(\theta)q(\theta'|\theta)}\right), \tag{12.13}$$

where $p_\theta(\theta)$ is the target distribution and $q(\theta'|\theta)$ is the proposal distribution. For a general transformation $\theta = f(u)$ and its inverse $u = f^{-1}(\theta)$ the modified *Metropolis acceptance criterion* takes the following form

$$\alpha = \min\left(1, \frac{p_\theta(f^{-1}(u'))q(u|u')\left|\det \frac{\partial f^{-1}(u')}{\partial u'}\right|}{p_\theta(f^{-1}(u))q(u'|u)\left|\det \frac{\partial f^{-1}(u)}{\partial u}\right|}\right), \tag{12.14}$$

where the Jacobian determinant also appears. In this formulation of MH, the sampler samples the distribution in the transformed space and then samples are pushed through the $\theta = f(u)$ transformation to the original space. Assuming that the transformation $\theta = f(u)$ induces a simpler geometry onto the transformed space, sampling using the above acceptance criterion can be substantially more efficient.

Figure 12.2 shows one such transformation that transforms the banana–shaped *Rosenbrock* distribution into a unit–variance normal distribution and *vice versa*. The same figure also demonstrates the effectiveness of simple proposal distributions $q(u'|u)$ in the transformed/latent space. A symmetric normal proposal distribution $q(u'|u_0)$ centred around a point u_0 corresponds to a highly effective proposal distribution in the original space, which captures the local geometry of the target distribution around that point.

12.2.4 Preconditioned Monte Carlo

Preconditioned Monte Carlo (PMC) is the result of the amalgamation of SMC, NFs and preconditioning as they were introduced in the previous paragraphs. In particular, we suggest the use of the transformation $\theta = f(u)$ of a NF in order to precondition the *Mutation* step of SMC. A pseudocode of the algorithm is presented at Algorithm 20. The *Mutation* step in this case consists of N *Random–Walk Metropolis* (RWM) steps, meaning MH with an isotropic Gaussian proposal distribution centred around the current state of the Markov chain, in which the algorithm targets the preconditioned density. We fix the acceptance rate of MH to its optimal value 23.4% between temperature steps by adapting the proposal scale (Gelman, Gilks, et al., 1997). As the optimal proposal scale of MH for a Gaussian target distribution is

$$\sigma_{\text{opt}} = \frac{2.38}{\sqrt{D}}\,, \tag{12.15}$$

where D is the number of dimensions/parameters, we can assess the performance of the NF preconditioner by estimating the ratio of the true scale σ to the optimal one σ_{opt}. Assuming that the NF preconditions perfectly the target density and maps it into a unit–variance Gaussian distribution, this ratio should be equal to one. In practice, this ratio can deviate slightly from the optimal value of unity, and one can utilise this ratio as a metric of the preconditioning quality. The number N of the MCMC steps performed in each iteration is determined adaptively during the run. The process we used is based on the mean correlation coefficient between the initial positions of the particles in the beginning of an iteration and their current positions. In particular, the particles are updated, using MCMC, until their mean correlation coefficient drops below a prespecified threshold value. The lower the value of this threshold, the higher the number N of MCMC steps. It is important to note that the correlation coefficient is computed in the preconditioned u space.

Algorithm 20 Preconditioned Monte Carlo

```
input Number of particles $N$
$t \leftarrow 1, \beta_1 \leftarrow 0, \mathcal{Z} \leftarrow 1$
for $k = 1$ to N do sample $\theta_1^k \sim \pi(\theta)$ and set $W_1^k = 1/N$
train $\theta = f(u)$ using $\{\theta_1^k\}_{k=1}^N$
while $\beta_t \neq 1$ do
    $t \leftarrow t + 1$
    $\beta_t \leftarrow$ solution to Eq. 12.10
    for $k = 1$ to N do $w_t^k \leftarrow W_{t-1}^k \mathcal{L}(\theta)^{\beta_t - \beta_{t-1}}$
    $\mathcal{Z} \leftarrow \mathcal{Z} N^{-1} \sum_{k=1}^N w_t^k$
    $\{\tilde{\theta}_{t-1}^k\}_{k=1}^N \leftarrow$ resample $\{\theta_{t-1}^k\}_{k=1}^N$ according to $\{W_t^k\}_{k=1}^N$ where $W_t^k = w_t^k / \sum_{k'=1}^N w_t^{k'}$
    for $k = 1$ to N do $W_t^k \leftarrow 1/N$
    $\{\theta_t^k\}_{k=1}^N \leftarrow$ move $\{\tilde{\theta}_{t-1}^k\}_{k=1}^N$ according to $K_t\left(\{\theta_t^k\}_{k=1}^N \leftarrow \{\tilde{\theta}_{t-1}^k\}_{k=1}^N ; f\right)$

    train $\theta = f(u)$ using $\{\theta_t^k\}_{k=1}^N$
end while
return samples $\{\theta_t^k\}_{k=1}^N$ and estimate of the marginal likelihood $\mathcal{Z}$
```

Table 6: The table shows the default values for the hyperparameters of PMC.

NF hyperparameters		SMC hyperparameters	
`blocks`	6	`particles`	1000 – 4000
`neurons`	$3 \times D$	`ESS`	95%
`batch`	1000	`threshold`	75%
`epochs`	500		
`tolerance`	30		
`lr`	$10^{-2} - 10^{-5}$		
`l1`	0.2		

12.2.5 Hyperparameters

We can classify the hyperparameters of PMC into two groups, those that have to do with the normalising flow and those that have to do with the SMC algorithm. The first group consists of structure and training hyperparamaters for the NF. The NF structure parameters include the number of MADE layers (`blocks`), as well as the number of neurons per hidden layer (`neurons`). The NF training hyperparameters include the learning rate (`lr`) of the Adam optimiser (Kingma & Ba, 2014), the maximum number of epochs (`epochs`), the training batch size (`batch`), the tolerance for early stopping (`tolerance`), and the scale of L_1 regularisation (`l1`). On the other hand, the SMC hyperparameters include the number of particles (`particles`), the desired effective sample size (`ESS`), and the correlation coefficient threshold (`threshold`). The default values for those hyperparameters are shown in Table 6. We found that this configuration was robust and efficient for a wide range of applications and thus decided to recommend this as the default choice.

12.2.6 Parallelisation

An important property of PMC, and indeed of any SMC algorithm, is its ideal scaling with the available number of CPUs. In particular, the mutation step of PMC is exactly parallelisable, meaning that that the speedup gained by using more than one CPU scales linearly with the number of CPUs as long as $n_{\mathrm{CPUs}} \leq n_{\mathrm{particles}}$. Similar methods that also use a large collection of particles scale less favourably. For instance, *Nested Sampling* (NS) exhibits sub–linear scaling as shown in Figure 12.3 of WJ Handley et al., 2015. The aforementioned characteristic of PMC renders it ideal for computationally costly applications that are often encountered in astronomy and cosmology.

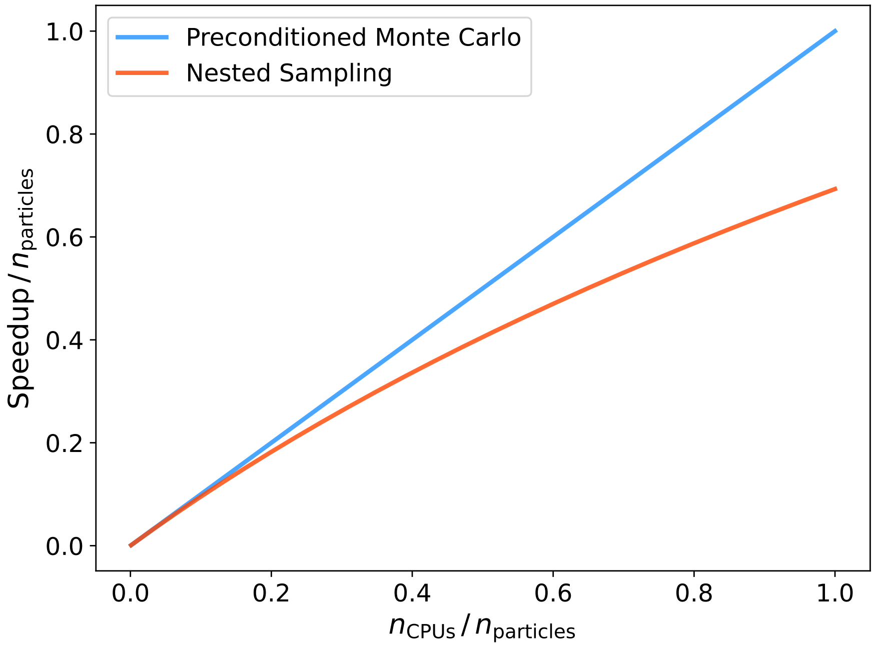

Figure 12.3: Parallelization of PMC compared to nested sampling. PMC (blue) exhibits linear speedup compared to the sub–linear one achieved by NS (orange).

12.3 EMPIRICAL EVALUATION

In this section we present two toy examples and two realistic parameter inference examples that reproduce common astronomical and cosmological analyses. In all cases, the hyperparameters of PMC were set to their default values as shown in Table 6. In both analyses, the performance of PMC is compared to that of SMC without preconditioning but otherwise using the same settings (e.g. number of particles, ESS, etc.) as PMC, as well as *Nested Sampling* (NS), a popular particle Monte Carlo alternative [1]. The metric that we use in order to evaluate the performance of each method is the total number of model evaluations performed until convergence. Convergence in all methods is well–defined: in PMC and SMC the algorithm converges when $\beta = 1$, whereas in NS the run stops when less than 1% of the model evidence is left unaccounted. All other computational costs are negligible, including the training and evaluation of the normalising flow in the case of PMC that only required a few seconds for the whole inference procedure. All methods used 1000 particles.

---

[1] We used the popular `Python` implementation `dynesty` (Speagle, 2020) for NS.

Table 7: The table shows a comparison of PMC, NS, and SMC in terms of their computational cost (i.e. total number of model evaluations until convergence).

	Model evaluations ($\times 10^6$)		
Distribution	**PMC**	NS	SMC
Rosenbrock	**1.5**	136.1	118.0
Gaussian Mixture	**1.6**	222.1	9.6
Primordial Features	**0.4**	21.3	19.5
Gravitational Waves	**0.4**	10.2	4.6

12.3.1 Rosenbrock distribution

The first toy example that we used is the *Rosenbrock* distribution, which exhibits strong non–linear correlation between its parameters. For this reason, the *Rosenbrock* distribution has often been used as a benchmark target for optimization and sampling tasks. Here we use a 20–dimensional generalisation of the distribution which is defined through the probability density function given by:

$$\log P(\theta) = - \sum_{i=1}^{N/2} \left[10 \left(\theta_{2i-1}^{2} - \theta_{2i} \right)^{2} + (\theta_{2i-1} - 1)^{2} \right] . \tag{12.16}$$

Furthermore, we use flat priors $\mathcal{U}(-10, 10)$ for all parameters. Figure 12.4 shows the 2–dimensional marginal posterior for the first two parameters as generated by the three methods. The total computational cost of PMC, NS, and SMC is 1.5×10^6, 136.1×10^6, and 118.0×10^6 model evaluations, respectively. PMC requires approximately 1/91 of the number of model evaluations that NS does, and approximately 1/79 of those that SMC does.

12.3.2 Gaussian Mixture

The second toy example that we used is a 50–dimensional Gaussian Mixture with two components, one of them being twice as massive as the other. This is a highly multimodal problem as the target distribution exhibits two distinct modes that are well separated. Just as in the *Rosenbrock* case, we use flat priors $\mathcal{U}(-10, 10)$ for all parameters. Figure 12.4 shows the 1–dimensional and 2–dimensional marginal posteriors for the first three parameters as generated by the three methods. The total computational cost of PMC, NS, and SMC is 1.6×10^6, 222.1×10^6, and 9.6×10^6 model evaluations respectively. PMC requires approximately 1/139 of the number of model evaluations that NS does, and 1/6 of those that SMC does.

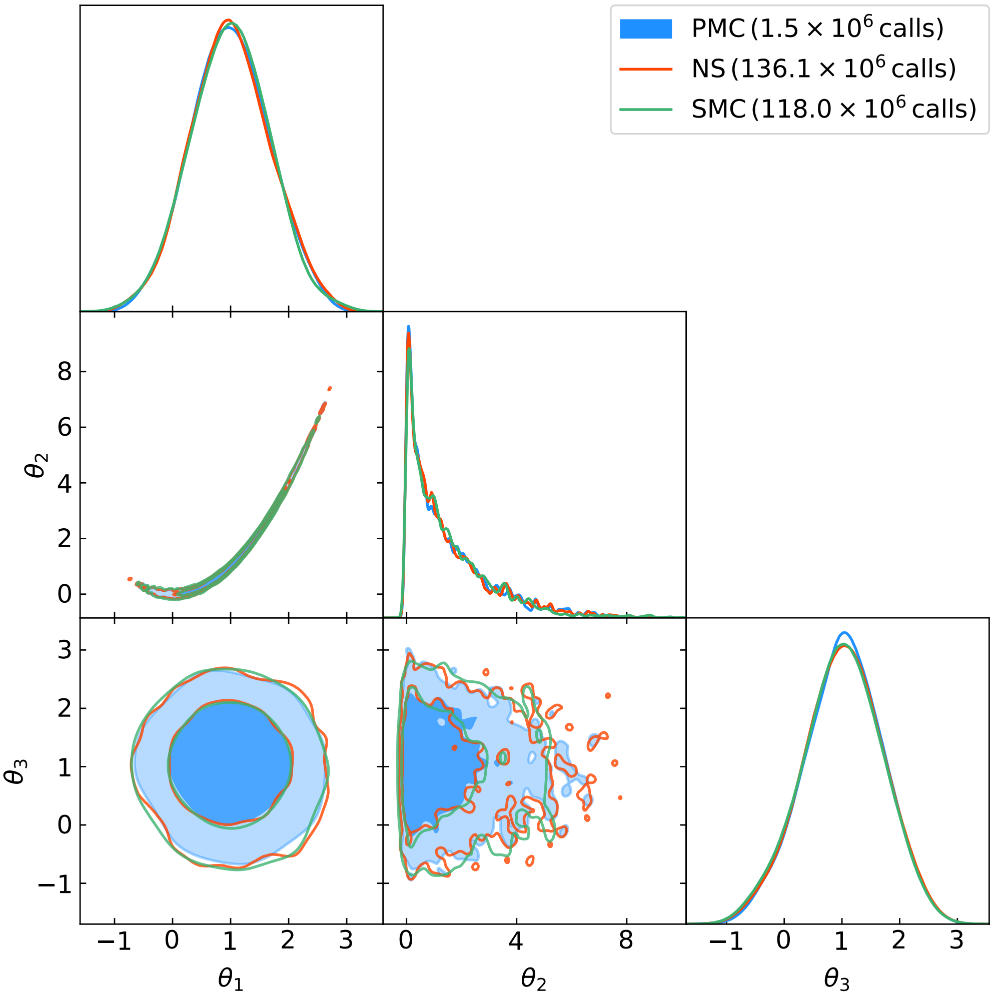

Figure 12.4: Illustration of the 1−dimensional and 2−dimensional marginal posteriors for the first three out of 20 parameters of the *Rosenbrock* distribution. The figure shows the 1−σ and 2−σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green.* The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached.

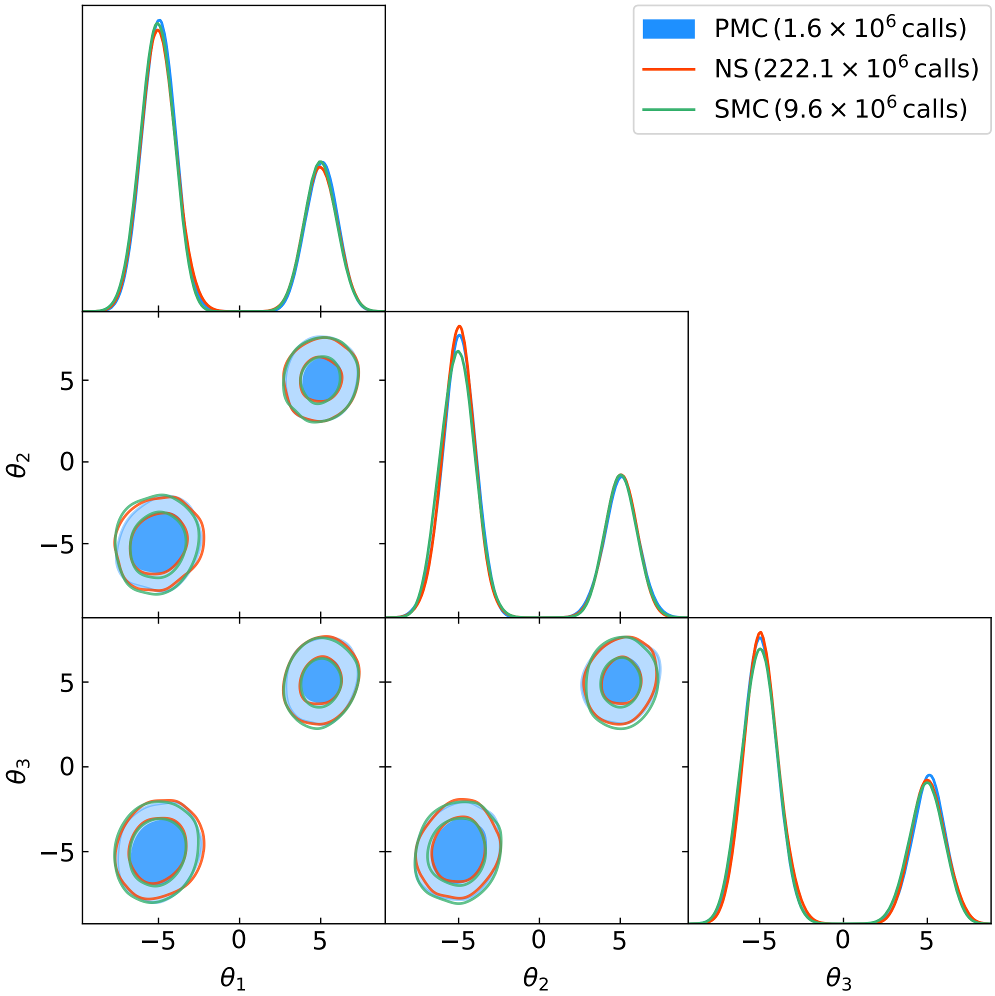

Figure 12.5: Illustration of the 1–dimensional and 2–dimensional marginal posteriors for the first three out of 50 parameters of the two–component Gaussian mixture distribution. The figure shows the 1–σ and 2–σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached.

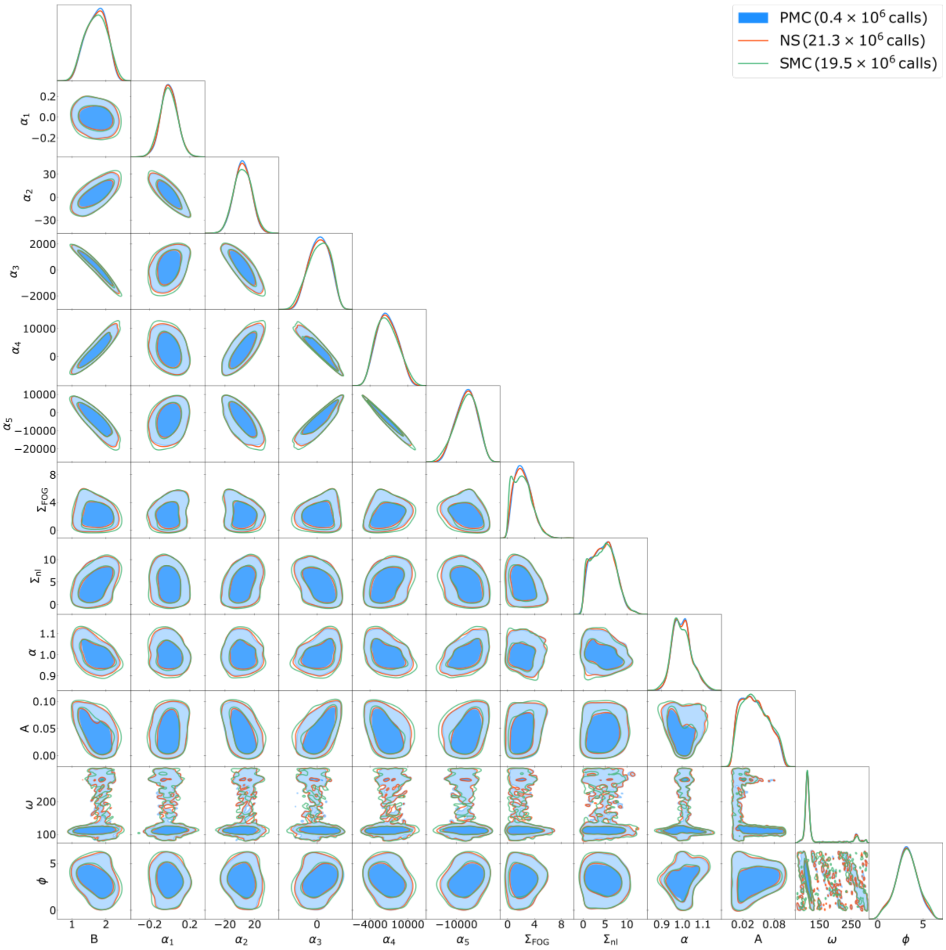

Figure 12.6: Illustration of the 1−dimensional and 2−dimensional marginal posteriors for the 12 parameters of the primordial features posterior. The figure shows the 1−σ and 2−σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached.

12.3.3 Primordial Features

The first realistic application that we study is the the search for primordial features along the Baryon Accoustic Oscillation (BAO) signature in the distribution of galaxies observed by the Sloan Digital Sky Survey (SDSS) (Daniel J. Eisenstein et al., 2011). In particular, the data that we analysed come from the 12th data release (DR12) of the high−redshift North Galactic Cap (NGC) sample of the Baryon Oscillation Spectroscopic Survey (BOSS) (Dawson et al., 2013). Our analysis follows closely that of Beutler, Biagetti, et al. (2019) for the linear oscillation model. The inference problem includes 12 free parameters with either flat/uniform or normal priors. Figure 12.6 shows the 1−dimensional and 2−dimensional marginal posteriors of the aforementioned analysis. The posterior distribution exhibits a highly non−Gaussian geometry that can hinder the sampling performance of conventional methods. The total computational cost of PMC, NS, and SMC is 0.4×10^6, 21.3×10^6, and 19.5×10^6 model evaluations respectively. PMC requires approximately 1/53 of the number of model evaluations that NS does, and 1/49 of those that SMC does.

12.3.4 Gravitational Waves

The second realistic application is the simulated gravitational wave analysis of an injected signal. For this, we used the standard CBC injected signal configuration provided by BILBY (Gregory Ashton et al., 2019). The inference problem includes 13 free parameters with a variety of common priors. Figure 12.7 shows the 1–dimensional and 2–dimensional marginal posteriors of the aforementioned analysis. The posterior distribution exhibits a highly non–Gaussian geometry that can hinder the sampling performance of conventional methods. The total computational cost of PMC, NS, and SMC is 0.4×10^6, 10.2×10^6, and 4.6×10^6 model evaluations respectively. PMC requires approximately 1/25 of the number of model evaluations that NS does, and 1/11 of those that SMC does.

12.4 DISCUSSION

While we have demonstrated PMC's superior sampling performance for a number of target distributions, including two real–world applications, the real test is based on researchers applying the method to their analyses. Different applications pose different computational challenges and there is no one single sampler to rule them all. Sometimes, certain kinds of distributions will be better handled by other, perhaps simpler, approaches.

In general, we expect PMC to be a useful tool when dealing with computationally expensive likelihood functions and highly correlated or multimodal posteriors. There two main reasons for this. First, training of the normalising flow takes about $\mathcal{O}(1\,\mathrm{s})$ per iteration, whereas the actual vectorised evaluation of the bijective mapping takes almost $\mathcal{O}(10\,\mathrm{ms})$ per MCMC step for the whole population of particles. This means that if the cost of evaluating the likelihood is low enough to be comparable to that of the normalising flow, as discussed above, the chances are that there are simpler methods (e.g. MCMC) that can obtain the results more quickly. The second reason has to do with the geometry of the posterior distribution. If the latter is trivial enough, for instance, if the target is approximately Gaussian with no non–linear correlation or multiple modes, then the use of the normalising flow as a preconditioner would offer no benefit and instead only help delay the run.

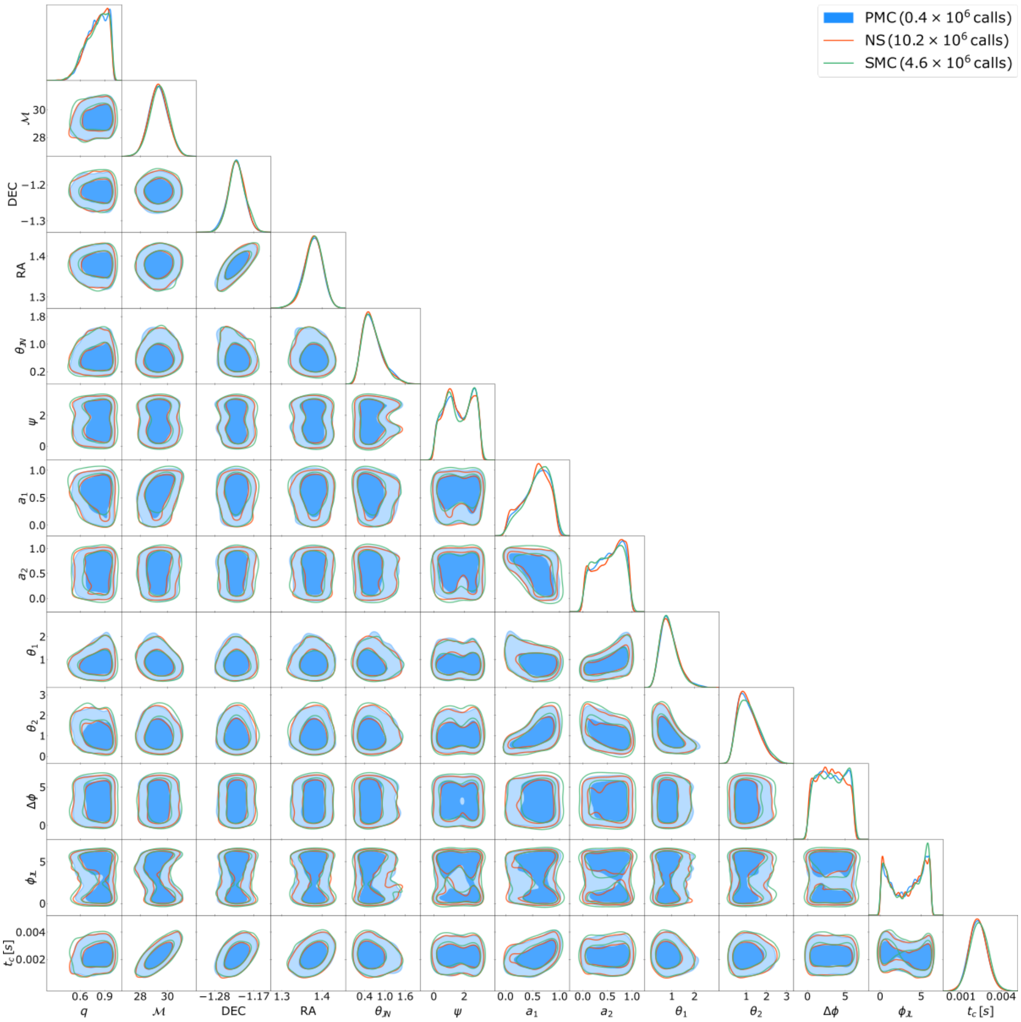

Figure 12.7: Illustration of the 1−dimensional and 2−dimensional marginal posteriors for the 13 parameters of the gravitational waves posterior. The figure shows the 1−σ and 2−σ contours generated by *Preconditioned Monte Carlo* (PMC) in *blue*, *Nested Sampling* (NS) in *orange*, and *Sequential Monte Carlo* (SMC) in *green*. The legend also shows the computational cost of each method in terms of the total number of required model evaluations until convergence is reached.

On the other hand, if both of these conditions are met, that is, the likelihood function is computationally expensive, as it is often the case in cosmology, and the posterior is non–Gaussian, then PMC can be a valuable asset in the astronomer's toolkit. Furthermore, when the cost of evaluating the likelihood function is large enough to dominate both the normalising flow evaluation and any potential *MPI* communication overhead, one can capitalise on the availability of multiple CPUs in order to accelerate PMC. In particular, if the evaluation of the likelihood function takes $\mathcal{O}(1\,\mathrm{s})$, one should be able to use up to thousands of CPUs, potentially parallelising all or a substantial fraction of the particles simultaneously.

12.5 CONCLUSIONS

The goal of this work was to develop a novel sampling method that can accelerate Bayesian parameter inference and model comparison in computationally challenging astronomical and cosmological analyses. To this end, we introduced PMC, a preconditioned generalisation of the standard SMC algorithm.

After introducing the method in Section 12.2, we presented a thorough demonstration of *Preconditioned Monte Carlo*'s sampling capabilities by comparing its sampling performance to that of *Nested Sampling* and *Sequential Monte Carlo* in a range of target distributions characterised by non–trivial geometry. The results are presented in Table 7. In general, we found that *Preconditioned Monte Carlo* is one to two orders of magnitude faster than either *Nested Sampling* or *Sequential Monte Carlo*, both of which performed similarly to each other. Furthermore, in the realistic analyses of primordial features and gravitational waves, *Preconditioned Monte Carlo* required approximately 50 and 25 times fewer model evaluations compared to NS in order to converge. The reduced computational cost, combined with the superior parallisation scaling, renders *Preconditioned Monte Carlo* ideal for astronomical and cosmological Bayesian analyses with computationally expensive, strongly correlated, multimodal and high–dimensional posteriors.

We hope that *Preconditioned Monte Carlo* will prove useful to the astronomical community by facilitating challenging Bayesian data analyses and enabling the investigation of complex models and sparse datasets. We also release a `Python` implementation of *Preconditioned Monte Carlo*, called `poco`MC, which is publically available at `https://github.com/minaskar/pocomc` and detailed documentation with installation instructions and examples at `https://pocomc.readthedocs.io`.

12.6 APPENDIX: COMPARISON TO INDEPENDENT METROPOLIS–HASTINGS SEQUENTIAL MONTE CARLO

Recent practice in the literature (Albergo et al., 2019; Arbel et al., 2021; M. J. Williams et al., 2021) is to use normalising flows as auxiliary densities for *Importance Sampling (IS)* and *Independent Metropolis–Hastings (IMH)* estimators. The latter approach can also be accommodated in the context of *Sequential Monte Carlo (SMC)* as an alternative to PMC. For this reason, we will offer an experimental comparison of PMC to IMH–SMC.

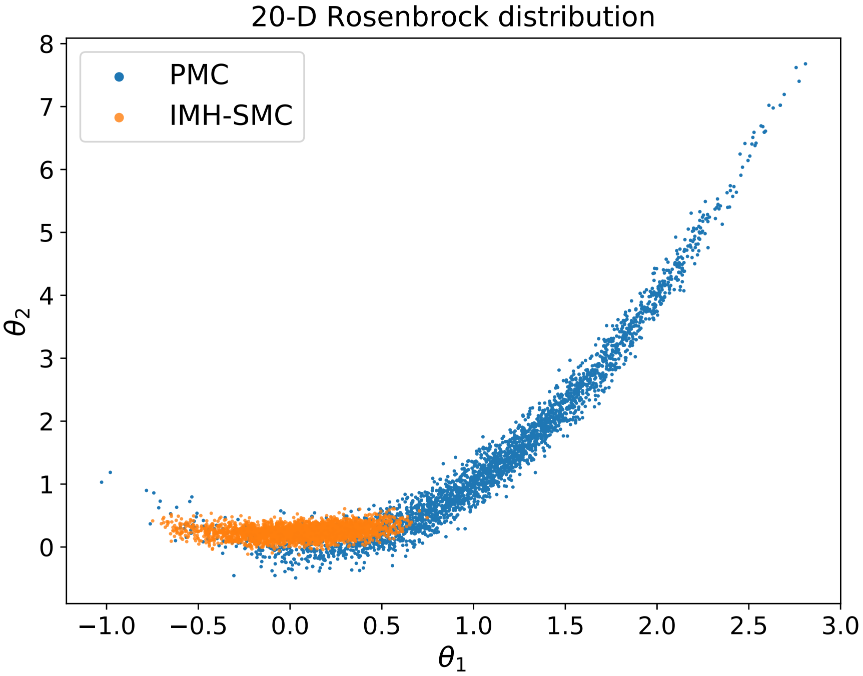

Figure 12.8: Comparison of the first two parameters of samples generated using PMC (*blue*) and IMH–SMC (*orange*) for the 20–D Rosenbrock target distribution. PMC produces representative samples, whereas IMH–SMC does not.

The IMH–SMC allgorithm is identical to Algorithm 20 with the exception that the mutation step of line 12 takes place using the modified *Metropolis acceptance criterion*

$$\alpha = \min\left(1, \frac{p_\theta(f^{-1}(u'))q(u)\left|\det \frac{\partial f^{-1}(u')}{\partial u'}\right|}{p_\theta(f^{-1}(u))q(u')\left|\det \frac{\partial f^{-1}(u)}{\partial u}\right|}\right), \tag{12.17}$$

instead of that of equation 12.14. The difference between the two criteria is that the proposal distribution $q(u) = \mathcal{N}(u|0,1)$ is no longer conditional on the previous state of the Markov chain.

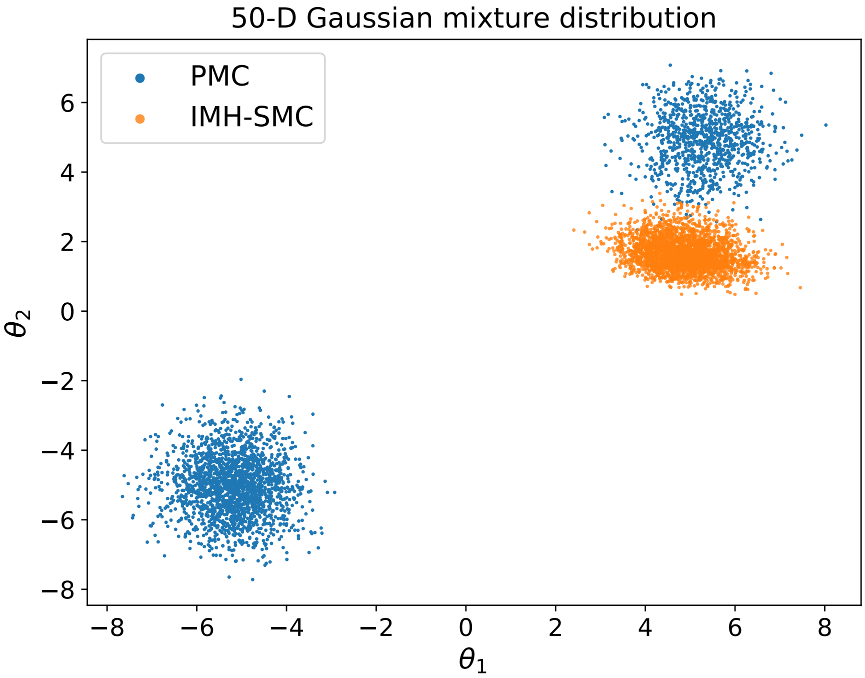

Figure 12.9: Comparison of the first two parameters of samples generated using PMC (*blue*) and IMH–SMC (*orange*) for the 50–D two–component Gaussian mixture target distribution. PMC produces representative samples, whereas IMH–SMC does not.

The number M of IMH steps performed in each iteration of IMH–SMC is determined adaptively during the run, based on the observed acceptance rate α, as

$$M = \frac{\log(1 - p)}{\log(1 - \alpha)}\,, \tag{12.18}$$

where p is the target probability of generating a new independent sample. In our examples below, the value of p is chosen such that the computational cost of IMH–SMC is similar to that of PMC for the same example. This results in $p > 0.99$ which corresponds to very conservative sampling.

Despite this, as shown in Figures 12.8 and 12.9, for the 20–dimensional Rosenbrock and the 50–dimensional two–component Gaussian mixture studied in the main text respectively, IMH–SMC does not manage to produce typical samples from the posterior distribution. It is important to note here that the acceptance rate of IMH–SMC was high throughout both runs, and as such offered no indication on its own that NF is not correct.

The origin of this discrepancy between IMH–SMC and PMC in both cases, and the ultimate inability of IMH–SMC to compete with PMC, originates in the substantial mismatch between the importance/NF distribution and target distribution in high dimensions and the subsequent over–fitting of the NF to the particle distribution leading to a narrower distribution. The high acceptance rate does not imply the high quality of NF solution, and other tests of the quality of solution are needed, such as comparing expectation of $\log p$ between samples from NF and true MCMC samples. On the other hand, PMC does not suffer from this pathology as the local exploration offered by MCMC helps diversify the particles in order to avoid over–fitting. Furthermore, local MCMC methods generally scale better with the number of dimensions compared to IMH and IS.

13 POCOMC

This chapter presents *pocoMC* which is the main contribution introduced in the paper titled *pocoMC: A Python package for accelerated Bayesian inference in astronomy and cosmology* that was submitted for publication in the *Journal of Open Source Software* in July 2022 (Karamanis, Nabergoj, et al., 2022). The content of the chapter is almost identical to that included in the aforementioned publication with the exception of minor text and figure formatting differences.

13.1 SUMMARY

pocoMC is a Python package for accelerated Bayesian inference in astronomy and cosmology. The code is designed to sample efficiently from posterior distributions with non–trivial geometry, including strong multimodality and non–linearity. To this end, pocoMC relies on the Preconditioned Monte Carlo algorithm which utilises a Normalising Flow in order to decorrelate the parameters of the posterior. It facilitates both tasks of parameter estimation and model comparison, focusing especially on computationally expensive applications. It allows fitting arbitrary models defined as a log–likelihood function and a log–prior probability density function in Python. Compared to popular alternatives (e.g. nested sampling) pocoMC can speed up the sampling procedure by orders of magnitude, cutting down the computational cost substantially. Finally, parallelisation to computing clusters manifests linear scaling.

13.2 STATEMENT OF NEED

Over the past few decades the volume of astronomical and cosmological data has increased substantially. At the same time, theoretical and phenomenological models in these fields have grown even more complex. As a response to that, a number of methods aiming at efficient Bayesian computation have been developed with the sole task of comparing those models to the available data (Sharma, 2017; Trotta, 2017). In the Bayesian context, scientific inference proceeds though the use of Bayes' theorem:

$$\mathcal{P}(\theta) = \frac{\mathcal{L}(\theta)\pi(\theta)}{\mathcal{Z}} \tag{13.1}$$

where the posterior $\mathcal{P}(\theta) \equiv p(\theta|d, \mathcal{M})$ is the probability of the parameters θ given the data d and the model $\mathcal{M}$. The other components of this equation are: the likelihood function $\mathcal{L}(\theta) \equiv p(d|\theta, \mathcal{M})$, the prior $\pi(\theta) \equiv p(\theta|\mathcal{M})$, and the model evidence $\mathcal{Z} = p(d|\mathcal{M})$. The prior and the likelihood are usually provided as input in this equation and one seeks to estimate the posterior and the evidence. Knowledge of the posterior, in the form of samples, is paramount for the task of parameter estimation whereas the ratio of model evidences yields the Bayes factor which is the cornerstone of Bayesian model comparison.

Markov chain Monte Carlo (MCMC) has been established as the standard tool for Bayesian computation in astronomy and cosmology, either as a standalone algorithm or as part of another method (e.g. nested sampling (Skilling, 2006)). However, as MCMC relies on the local exploration of the posterior, the presence of non-linear correlation between parameters and multimodality can at best hinder its performance and at worst violate its theoretical guarantees of convergence (i.e. ergodicity). Usually those challenges are partially addressed by reparameterising the model using a common change–of–variables parameter transformation. However, guessing the right kind of reparameterisation a priori is not trivial as it often requires a deep knowledge of the physical model and its symmetries. These problems are usually complicated further by the substantial computational cost of evaluating astronomical and cosmological models. pocoMC is designed to tackle exactly these kinds of difficulties by automatically reparameterising the model such that the parameters of the model are approximately uncorrelated and standard techniques can be applied. As a result, pocoMC produces both samples from the posterior distribution and an unbiased estimate of the model evidence thus facilitating both scientific tasks with excellent efficiency and robustness. Compared to popular alternatives such as nested sampling, pocoMC can reduce the computational cost, and thus, the total run time of the analysis by orders of magnitude, in both artificial and realistic applications (Karamanis, Beutler, Peacock, et al., 2022). Finally, the code is well-tested and is currently used for research work in the field of gravitational wave parameter estimation (Vretinaris et al., 2022).

Figure 13.1: Logo of pocoMC.

13.3 METHOD

pocoMC implements the Preconditioned Monte Carlo (PMC) algorithm. PMC combines the popular Sequential Monte Carlo (SMC) (Del Moral et al., 2006) method with a Normalising Flow (NF) (Papamakarios, Nalisnick, et al., 2021). The latter works as a preconditioner for the target distribution of the former. As SMC evolves a population of particles, starting from the prior distribution and gradually approaching the posterior distribution, the NF transforms the parameters of the target distribution such that any correlation between parameters or presence of multimodality is removed. The effect of this bijective transformation is the substantial rise in the sampling efficiency of the algorithm as the particles are allowed to sample freely from the target without being hindered by its locally–curved geometry. The method is explained in detail in the accompanying publication (Karamanis, Beutler, Peacock, et al., 2022) and we provide only a short summary here.

13.3.1 Sequential Monte Carlo

The basic idea of basic SMC is to sample from the posterior distribution $\mathcal{P}(\theta)$ by first defining a path of intermediate distributions starting from the prior $\pi(\theta)$. In the case of pocoMC the path has the form:

$$p_t(\theta) = \pi(\theta)^{1-\beta_t} \mathcal{P}(\theta)^{\beta_t} \tag{13.2}$$

where $0 = \beta_1 < \beta_2 < \cdots < \beta_T = 1$. Starting from the prior, each distribution with density $p_t(\theta)$ is sampled in turn using a collection of particles propagated by a number of MCMC steps. Prior to MCMC sampling, the particles are re-weighted using importance sampling and then re-sampled to account for the transition from $p_t(\theta)$ to $p_{t+1}(\theta)$. pocoMC utilises the importance weights of this step to define an estimator for the effective sample size (ESS) of the population of particles. Maintaining a fixed value of ESS during the run allows pocoMC to adaptively specify the β_t schedule.

13.3.2 Preconditioned Monte Carlo

In vanilla SMC, standard MCMC methods (e.g. Metropolis-Hastings) are used to update the positions of the particles during each iteration. This however can become highly inefficient if the distribution $p_t(\theta)$ is characterised by a non–trivial geometry. pocoMC, which is based on PMC, utilises a NF to learn an invertible transformation that simplifies the geometry of the distribution by mapping $p_t(\theta)$ into a zero-mean unit-variance normal distribution. Sampling then proceeds in the latent space in which correlations are substantially reduced. The positions of the particles are transformed back to the original parameter space at the end of each iteration. This way, PMC and pocoMC are able to sample from very challenging posteriors very efficiently using simple Metropolis-Hastings updates in the preconditioned/uncorrelated latent space.

13.4 FEATURES

- User–friendly black-box API (only the log-likelihood, log-prior and some prior samples required from the user)
- Default configuration sufficient for most applications (no tuning is required but is possible for experienced users)
- Posterior corner, trace, and run plotting tools
- Support for both MAF and RealNVP normalising flows with added regularisation (Dinh et al., 2016; Papamakarios, Pavlakou, et al., 2017)
- Straightforward parallelisation using MPI or multiprocessing
- Continuous integration, unit tests and wide range of examples available
- Extensive documentation available online `http://pocomc.readthedocs.io`

14 CONCLUSIONS

So long, and thanks for all the fish.

— Douglas Adams, The Hitchhiker's Guide to the Galaxy

Over the past couple of decades, Bayesian inference has been established as the standard mathematical framework for conducting scientific inference in the physical sciences. This progress has been largely facilitated by the recent advances in computer technology and probabilistic computational methods. However, the specific characteristics of the mathematical models and available data used in astronomy and cosmology still pose significant challenges for existing computational tools.

From the perspective of theoretical modelling, many astrophysical models involve computationally expensive operations which are almost always non–differentiable. This limits the potential range of application of a plethora of MCMC methods, particularly those that rely on the use of the gradient of the posterior density function or are unable to scale to a large number of parallel CPUs. On the other hand, the commonly sparse nature of the available data often induces a level of multimodality in the studied posterior distributions. The existence of multiple modes in the posterior distribution can hinder the sampling procedure of most computational tools and in the case of most MCMC methods, make the results unreliable. This thesis has introduced two methods and their software implementations that were specifically designed with this kind of challenge in mind.

In Chapter 10 we introduced *Ensemble Slice Sampling (ESS)*, a method that extends the applicability of the univariate slice sampler to multivariate target distributions, by utilising an ensemble of parallel walkers. The method requires minimal tuning and no gradient information, demonstrates affine–invariant sampling performance, and is trivially parallelisable to a large number of CPUs. Chapter 11 presents `zeus`, an open–source `Python` implementation of ESS. Compared to the popular MCMC sampler `emcee`, the sampling efficiency of `zeus` scales more favourably with the total number of dimensions. Furthermore, the generated Markov chains exhibit substantially lower autocorrelation levels for a wide range of target distributions and the method generally requires significantly fewer walkers than `emcee`. Finally, in the problems of BAO and exoplanet parameter estimation, `zeus` is 9 and 29 times more efficient than the competition, respectively.

Chapter 12 is devoted to *Preconditioned Monte Carlo (PMC)*, a novel Monte Carlo method for sampling from posteriors with non–trivial geometry (i.e.

non–linear correlations, multimodality). PMC utilises a Normalising Flow (NF) transformation in order to precondition the target distribution by approximately removing the correlations between its parameters. PMC then relies on a *Sequential Monte Carlo (SMC)* in order to produce posterior samples and an estimate of the model evidence. Empirical tests validate the high sampling efficiency of PMC. In the cases of primordial feature analysis and gravitational wave inference, PMC is approximately 50 and 25 times faster respectively than nested sampling. Finally, Chapter 13 offers a short overview of `poco`MC, an open–source `Python` implementation of PMC. The basic principles of PMC are presented along with the various options and features provided in the package. In terms of parallelisation, `poco`MC manifests linear scaling up to thousands of CPUs.

The methods introduced in the aforementioned chapters aim to address the various computational challenges currently presented by modern astrophysical models and data. Despite their empirical success, as demonstrated by the provided tests and their adoption by the astronomical community, their application in higher dimensions (e.g. $D > 100$) is still hindered by the *curse of dimensionality.* In the future, in order to accommodate for subtle effects present in the data, astrophysical models will necessarily become increasingly complicated. As a response, sampling methods such as the ones presented in this thesis will have to evolve in order to cope with the additional computational challenges. A possible avenue of future research could be the self–supervised construction of surrogate models (e.g. emulators) for either the likelihood function, posterior density, or model, thus enabling the use of gradient–based MCMC methods in the context of advanced schemes such as PMC. We sincerely hope that, in the meantime, methods and packages such as ESS & PMC, and `zeus` & `poco`MC will prove useful to the astronomical community by facilitating the next generation of Bayesian data analyses.

BIBLIOGRAPHY

Albergo, MS, G Kanwar & PE Shanahan

2019 "Flow-based generative models for Markov chain Monte Carlo in lattice field theory", *Physical Review D*, 100, 3, p. 034515.

Andrieu, Christophe & Johannes Thoms

2008 "A tutorial on adaptive MCMC", *Statistics and Computing*, 18, 4, pp. 343-373.

Arbel, Michael, Alex Matthews & Arnaud Doucet

2021 "Annealed flow transport monte carlo", in *International Conference on Machine Learning*, PMLR, pp. 318-330.

Ashton, Greg, Noam Bernstein, Johannes Buchner, Xi Chen, Gábor Csányi, Andrew Fowlie, Farhan Feroz, Matthew Griffiths, Will Handley, Michael Habeck, et al.

2022 "Nested sampling for physical scientists", *Nature Reviews Methods Primers*, 2, 1, pp. 1-22.

Ashton, Gregory, Moritz Hübner, Paul D Lasky, Colm Talbot, Kendall Ackley, Sylvia Biscoveanu, Qi Chu, Atul Divakarla, Paul J Easter, Boris Goncharov, et al.

2019 "BILBY: A user-friendly Bayesian inference library for gravitational-wave astronomy", *The Astrophysical Journal Supplement Series*, 241, 2, p. 27.

Atchadé, Yves F, Gareth O Roberts & Jeffrey S Rosenthal

2011 "Towards optimal scaling of Metropolis-coupled Markov chain Monte Carlo", *Statistics and Computing*, 21, 4, pp. 555-568.

Bayes, Thomas

1763 "An Essay towards Solving a Problem in the Doctrine of Chances", *Philosophical Transactions (1683-1775)*, 53, pp. 370-418.

Bellman, Richard

1966 "Dynamic programming", *Science*, 153, 3731, pp. 34-37.

Berger, James O, José M Bernardo & Dongchu Sun

2009 "The formal definition of reference priors", *The Annals of Statistics*, 37, 2, pp. 905-938.

Bernardo, José M

2005 “Reference analysis”, *Handbook of statistics*, 25, pp. 17-90.

Bernardo, Jose M

1979 “Reference posterior distributions for Bayesian inference”, *Journal of the Royal Statistical Society: Series B (Methodological)*, 41, 2, pp. 113-128.

Bernardo, José M & Adrian FM Smith

2009 *Bayesian theory*, John Wiley & Sons, vol. 405.

Besag, Julian & Peter J Green

1993 “Spatial statistics and Bayesian computation”, *Journal of the Royal Statistical Society: Series B (Methodological)*, 55, 1, pp. 25-37.

Betancourt, Michael

2017a “A Conceptual Introduction to Hamiltonian Monte Carlo”, *ArXiv e-prints*, eprint: `1701.02434` (stat.ME).

2017b “A conceptual introduction to Hamiltonian Monte Carlo”, *arXiv preprint arXiv:1701.02434*.

Beutler, Florian, Matteo Biagetti, Daniel Green, Anže Slosar & Benjamin Wallisch

2019 “Primordial features from linear to nonlinear scales”, *Physical Review Research*, 1, 3, p. 033209.

Beutler, Florian, Hee-Jong Seo, Ashley J Ross, Patrick McDonald, Shun Saito, Adam S Bolton, Joel R Brownstein, Chia-Hsun Chuang, Antonio J Cuesta, Daniel J Eisenstein, et al.

2017 “The clustering of galaxies in the completed SDSS-III Baryon Oscillation Spectroscopic Survey: baryon acoustic oscillations in the Fourier space”, *MNRAS*, 464, 3, pp. 3409-3430, DOI: `10.1093/mnras/stw2373`.

Bishop, Christopher M

2006 “Pattern recognition”, *Machine Learning*, 128, 9.

Brading, Katherine & Elena Castellani

2003 *Symmetries in physics: philosophical reflections*, Cambridge University Press.

Brewer, Brendon J, Livia B Pártay & Gábor Csányi

2011 “Diffusive nested sampling”, *Statistics and Computing*, 21, 4, pp. 649-656.

Brooks, Steve, Andrew Gelman, Galin Jones & Xiao-Li Meng

2011 *Handbook of Markov chain Monte Carlo*, CRC Press.

Buchner, Johannes

2021 "UltraNest–a robust, general purpose Bayesian inference engine", *arXiv preprint arXiv:2101.09604*.

Burden, Richard L, J Douglas Faires & Annette M Burden

2015 *Numerical analysis*, Cengage Learning.

Cahn, Robert W & Peter Haasen

1996 *Physical metallurgy*, Elsevier, vol. 1.

Calderhead, Ben & Mark Girolami

2009 "Estimating Bayes factors via thermodynamic integration and population MCMC", *Computational Statistics & Data Analysis*, 53, 12, pp. 4028-4045.

Casella, George & Edward I George

1992 "Explaining the Gibbs sampler", *The American Statistician*, 46, 3, pp. 167-174.

Cox, Richard T

1946 "Probability, frequency and reasonable expectation", *American Journal of Physics*, 14, 1, pp. 1-13.

Da Costa-Luis, Casper O

2019 "tqdm: A fast, extensible progress meter for Python and CLI", *Journal of Open Source Software*, 4, 37, p. 1277, DOI: `10.21105/joss.01277`.

Dalcin, Lisandro D, Rodrigo R Paz, Pablo A Kler & Alejandro Cosimo

2011 "Parallel distributed computing using Python", *Advances in Water Resources*, 34, 9, pp. 1124-1139, DOI: `10.1016/j.advwatres.2011.04.013`.

Dawid, A Philip, Mervyn Stone & James V Zidek

1973 "Marginalization paradoxes in Bayesian and structural inference", *Journal of the Royal Statistical Society: Series B (Methodological)*, 35, 2, pp. 189-213.

Dawson, Kyle S., David J. Schlegel, Christopher P. Ahn, Scott F. Anderson, Éric Aubourg, Stephen Bailey, Robert H. Barkhouser, et al.

2013 "The Baryon Oscillation Spectroscopic Survey of SDSS-III", *AJ*, 145, 1, p. 10, DOI: `10.1088/0004-6256/145/1/10`, arXiv: `1208.0022`.

Del Moral, Pierre, Arnaud Doucet & Ajay Jasra

2006 "Sequential monte carlo samplers", *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68, 3, pp. 411-436.

Dempster, Arthur P, Nan M Laird & Donald B Rubin

1977 "Maximum likelihood from incomplete data via the EM algorithm", *Journal of the Royal Statistical Society: Series B (Methodological)*, 39, 1, pp. 1-22.

DESI Collaboration, Amir Aghamousa, Jessica Aguilar, Steve Ahlen, Shadab Alam, Lori E. Allen, Carlos Allende Prieto, James Annis, et al.

2016 "The DESI Experiment Part I: Science,Targeting, and Survey Design", *ArXiv e-prints*, eprint: 1611.00036.

Dickey, James M

1971 "The weighted likelihood ratio, linear hypotheses on normal location parameters", *The Annals of Mathematical Statistics*, pp. 204-223.

1976 "Approximate posterior distributions", *Journal of the American Statistical Association*, 71, 355, pp. 680-689.

Dickey, James M & BP Lientz

1970 "The weighted likelihood ratio, sharp hypotheses about chances, the order of a Markov chain", *The Annals of Mathematical Statistics*, pp. 214-226.

Dinh, Laurent, Jascha Sohl-Dickstein & Samy Bengio

2016 "Density estimation using real nvp", *arXiv preprint arXiv:1605.08803*.

Duane, Simon, Anthony D Kennedy, Brian J Pendleton & Duncan Roweth

1987 "Hybrid monte carlo", *Physics Letters B*, 195, 2, pp. 216-222.

Earl, David J & Michael W Deem

2005 "Parallel tempering: Theory, applications, and new perspectives", *Physical Chemistry Chemical Physics*, 7, 23, pp. 3910-3916.

Efron, Bradley

1986 "Why isn't everyone a Bayesian?", *The American Statistician*, 40, 1, pp. 1-5.

Eisenstein, Daniel J., David H. Weinberg, Eric Agol, Hiroaki Aihara, Carlos Allende Prieto, Scott F. Anderson, James A. Arns, et al.

2011 "SDSS-III: Massive Spectroscopic Surveys of the Distant Universe, the Milky Way, and Extra-Solar Planetary Systems", *The Astronomical Journal*, 142, 3, p. 72, DOI: 10.1088/0004-6256/142/3/72, eprint: 1101.1529.

Farr, B. & W. M. Farr

2015 *Kombine: a kernel-density-based, embarrassingly parallel ensemble sampler*, https://github.com/bfarr/kombine (visited on 11/02/2020).

Feroz, Farhan, Michael P Hobson, Ewan Cameron & Anthony N Pettitt

2013 “Importance nested sampling and the MultiNest algorithm”, *arXiv preprint arXiv:1306.2144*.

Feroz, Farhan & Mike P Hobson

2008 “Multimodal nested sampling: an efficient and robust alternative to Markov Chain Monte Carlo methods for astronomical data analyses”, *Monthly Notices of the Royal Astronomical Society*, 384, 2, pp. 449-463.

Feroz, Farhan, MP Hobson & Michael Bridges

2009 “MultiNest: an efficient and robust Bayesian inference tool for cosmology and particle physics”, *Monthly Notices of the Royal Astronomical Society*, 398, 4, pp. 1601-1614.

Feroz, Farhan & John Skilling

2013 “Exploring multi-modal distributions with nested sampling”, in *AIP Conference Proceedings*, 1, American Institute of Physics, vol. 1553, pp. 106-113.

Foreman-Mackey, Daniel

2019 *Autocorrelation analysis & convergence — emcee 3.0.2 documentation*, `https://emcee.readthedocs.io/en/stable/tutorials/autocorr/` (visited on 02/04/2020).

Foreman-Mackey, Daniel, Will M Farr, Manodeep Sinha, Anne M Archibald, David W Hogg, Jeremy S Sanders, Joe Zuntz, Peter KG Williams, Andrew RJ Nelson, Miguel de Val-Borro, et al.

2019 “emcee v3: A Python ensemble sampling toolkit for affine-invariant MCMC”, *ArXiv e-prints*, eprint: `1911.07688`.

Foreman-Mackey, Daniel, David W Hogg, Dustin Lang & Jonathan Goodman

2013 “emcee: the MCMC hammer”, *Publications of the Astronomical Society of the Pacific*, 125, 925, p. 306.

Friel, Nial & Anthony N Pettitt

2008 “Marginal likelihood estimation via power posteriors”, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70, 3, pp. 589-607.

Fulton, Benjamin J., Erik A. Petigura, Sarah Blunt & Evan Sinukoff

2018 “RadVel: The Radial Velocity Modeling Toolkit”, *Publications of the Astronomical Society of the Pacific*, 130, 986, p. 044504, DOI: `10.1088/1538-3873/aaaaa8`, eprint: `1801.01947`.

Garbuno-Inigo, Alfredo, Franca Hoffmann, Wuchen Li & Andrew M Stuart

2020 "Interacting Langevin diffusions: Gradient structure and ensemble Kalman sampler", *SIAM Journal on Applied Dynamical Systems*, 19, 1, pp. 412-441.

Garbuno-Inigo, Alfredo, Nikolas Nüsken & Sebastian Reich

2020 "Affine invariant interacting Langevin dynamics for Bayesian inference", *SIAM Journal on Applied Dynamical Systems*, 19, 3, pp. 1633-1658.

Geisser, Seymour & William F Eddy

1979 "A predictive approach to model selection", *Journal of the American Statistical Association*, 74, 365, pp. 153-160.

Gelfand, Alan E, Dipak K Dey & Hong Chang

1992 *Model determination using predictive distributions with implementation via sampling-based methods*, tech. rep., Stanford University, Department of Statistics.

Gelfand, Alan E & Adrian FM Smith

1990 "Sampling-based approaches to calculating marginal densities", *Journal of the American Statistical Association*, 85, 410, pp. 398-409.

Gelman, Andrew

2008 "Objections to Bayesian statistics", *Bayesian Analysis*, 3, 3, pp. 445-449.

Gelman, Andrew, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari & Donald B Rubin

2013 *Bayesian Data Analysis*, CRC Press.

Gelman, Andrew, Walter R Gilks & Gareth O Roberts

1997 "Weak convergence and optimal scaling of random walk Metropolis algorithms", *The Annals of Applied Probability*, 7, 1, pp. 110-120.

Gelman, Andrew & Xiao-Li Meng

1998 "Simulating normalizing constants: From importance sampling to bridge sampling to path sampling", *Statistical Science*, pp. 163-185.

Gelman, Andrew, Donald B Rubin, et al.

1992 "Inference from iterative simulation using multiple sequences", *Statistical Science*, 7, 4, pp. 457-472, DOI: `10.1214/ss/1177011136`.

Geman, Stuart & Donald Geman

1984 "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6, pp. 721-741.

Germain, Mathieu, Karol Gregor, Iain Murray & Hugo Larochelle

2015 "Made: Masked autoencoder for distribution estimation", in *International Conference on Machine Learning*, PMLR, pp. 881-889.

Geweke, John

1992 "Evaluating the accuracy of sampling-based approaches to the calculations of posterior moments", *Bayesian Statistics*, 4, pp. 641-649, DOI: 10.21034/sr.148.

Geyer, Charles J

1991 "Markov chain Monte Carlo maximum likelihood".

1992 "Practical Markov chain Monte Carlo", *Statistical Science*, pp. 473-483.

Gilks, Walter R, Sylvia Richardson & David Spiegelhalter

1995 *Markov chain Monte Carlo in practice*, CRC press.

Gilks, Walter R, Gareth O Roberts & Edward I George

1994 "Adaptive direction sampling", *Journal of the Royal Statistical Society Series D (The Statistician)*, 43, 1, pp. 179-189, DOI: 10.2307/2348942.

Gneiting, Tilmann & Adrian E Raftery

2007 "Strictly proper scoring rules, prediction, and estimation", *Journal of the American Statistical Association*, 102, 477, pp. 359-378.

Goertzel, Gerald

1949 *Quota sampling and importance functions in stochastic solution of particle problems*, tech. rep.

Goodman, Jonathan & Jonathan Weare

2010 "Ensemble samplers with affine invariance", *Communications in Applied Mathematics and Computational Science*, 5, 1, pp. 65-80.

Gorur, Dilan & Carl Edward Rasmussen

2010 "Dirichlet process gaussian mixture models: Choice of the base distribution", *Journal of Computer Science and Technology*, 25, 4, pp. 653-664.

Gregory, Phil

2005 *Bayesian logical data analysis for the physical sciences: a comparative approach with mathematica® support*, Cambridge University Press.

Gubernatis, James E

2005 "Marshall Rosenbluth and the Metropolis algorithm", *Physics of Plasmas*, 12, 5, p. 057303.

Gunel, Erdogan & James Dickey

1974 “Bayes factors for independence in contingency tables”, *Biometrika*, 61, 3, pp. 545-557.

Haario, Heikki, Eero Saksman, Johanna Tamminen, et al.

2001 “An adaptive Metropolis algorithm”, *Bernoulli*, 7, 2, pp. 223-242.

Hammersley, John M & DC Handscomb

1964 “Percolation processes”, in *Monte Carlo Methods*, Springer, pp. 134-141.

Handley, WJ, MP Hobson & AN Lasenby

2015 “POLYCHORD: next-generation nested sampling”, *Monthly Notices of the Royal Astronomical Society*, 453, 4, pp. 4384-4398.

Hastings, W Keith

1970 “Monte Carlo Sampling Methods using Markov Chains and their Applications”, *Biometrika*, 57, 1, pp. 97-109, DOI: `10.1093/biomet/57.1.97`.

Higson, Edward, Will Handley, Michael Hobson & Anthony Lasenby

2019 “Dynamic nested sampling: an improved algorithm for parameter estimation and evidence calculation”, *Statistics and Computing*, 29, 5, pp. 891-913.

Hobson, Michael P & Charles McLachlan

2003 “A Bayesian approach to discrete object detection in astronomical data sets”, *Monthly Notices of the Royal Astronomical Society*, 338, 3, pp. 765-784.

Hoffman, Matthew, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran & Srinivas Vasudevan

2019 “Neutra-lizing bad geometry in hamiltonian monte carlo using neural transport”, *arXiv preprint arXiv:1903.03704.*

Hoffman, Matthew D, Andrew Gelman, et al.

2014 “The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo.” *Journal of Machine Learning Research*, 15, 1, pp. 1593-1623.

Hogg, David W, Jo Bovy & Dustin Lang

2010 “Data analysis recipes: Fitting a model to data”, *arXiv preprint arXiv:1008.4686.*

Huijser, David, Jesse Goodman & Brendon J. Brewer

2017 “Properties of the Affine Invariant Ensemble Sampler in high dimensions”, *ArXiv e-prints*, eprint: `1509.02230`.

Hukushima, Koji & Koji Nemoto

1996 “Exchange Monte Carlo method and application to spin glass simulations”, *Journal of the Physical Society of Japan*, 65, 6, pp. 1604-1608.

Hunter, John D

2007 “Matplotlib: A 2D graphics environment”, *IEEE Annals of the History of Computing*, 9, 03, pp. 90-95, DOI: 10.1109/MCSE.2007.55.

Iba, Yukito

2001 “Extended ensemble monte carlo”, *International Journal of Modern Physics C*, 12, 05, pp. 623-656.

Ionides, Edward L

2008 “Truncated importance sampling”, *Journal of Computational and Graphical Statistics*, 17, 2, pp. 295-311.

Jaynes, Edwin

1982 “On the rationale of maximum-entropy methods”, *Proceedings of the IEEE*, 70, 9, pp. 939-952.

Jaynes, Edwin T

2003 *Probability theory: The logic of science*, Cambridge University Press.

Jeffreys, Harold

1998 *The theory of probability*, OUP Oxford.

Kahn, Herman & Theodore E Harris

1951 “Estimation of particle transmission by random sampling”, *National Bureau of Standards applied mathematics series*, 12, pp. 27-30.

Karamanis, Minas & Florian Beutler

2021 “Ensemble slice sampling”, *Statistics and Computing*, 31, 5, pp. 1-18.

Karamanis, Minas, Florian Beutler & John A Peacock

2021 “zeus: A Python implementation of Ensemble Slice Sampling for efficient Bayesian parameter inference”, *Monthly Notices of the Royal Astronomical Society*, 508, 3, pp. 3589-3603.

Karamanis, Minas, Florian Beutler, John A Peacock, David Nabergoj & Uroš Seljak

2022 “Accelerating astronomical and cosmological inference with preconditioned Monte Carlo”, *Monthly Notices of the Royal Astronomical Society*, 516, 2, pp. 1644-1653.

Karamanis, Minas, David Nabergoj, Florian Beutler, John A Peacock & Uroš Seljak

2022 "pocoMC: A Python package for accelerated Bayesian inference in astronomy and cosmology", *arXiv:2207.05660.*

Kass, Robert E & Larry Wasserman

1996 "The selection of prior distributions by formal rules", *Journal of the American statistical Association*, 91, 435, pp. 1343-1370.

Keeling, Charles David & Timothy P Whorf

2004 "Atmospheric CO2 concentrations derived from flask air samples at sites in the SIO network", *Trends: A Compendium of Data on Global Change.*

Keynes, John Maynard

1921 *A treatise on probability*, Macmillan and Company, Limited.

Kingma, Diederik P & Jimmy Ba

2014 "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980.*

Kipnis, Claude & SR Srinivasa Varadhan

1986 "Central limit theorem for additive functionals of reversible Markov processes and applications to simple exclusions", *Communications in Mathematical Physics*, 104, 1, pp. 1-19.

Kirkpatrick, Scott, C Daniel Gelatt Jr & Mario P Vecchi

1983 "Optimization by simulated annealing", *Science*, 220, 4598, pp. 671-680.

Kloek, Tuen & Herman K Van Dijk

1978 "Bayesian estimates of equation system parameters: an application of integration by Monte Carlo", *Econometrica: Journal of the Econometric Society*, pp. 1-19.

Koza, John R et al.

1994 *Genetic programming II*, MIT Press Cambridge, vol. 17.

Kullback, Solomon & Richard A Leibler

1951 "On information and sufficiency", *The Annals of Mathematical Statistics*, 22, 1, pp. 79-86.

Laplace, Pierre-Simon

1810 "Mémoire sur les intégrales définies et leur application aux probabilités, et spécialementa la recherche du milieu qu'il faut choisir entre les résultats des observations", *Mem. Acad. Sci.(I), XI, Section V*, pp. 375-387.

Ledoux, Michel

2001 *The concentration of measure phenomenon*, 89, American Mathematical Society.

Leimkuhler, Benedict, Charles Matthews & Jonathan Weare

2018 “Ensemble preconditioning for Markov chain Monte Carlo simulation”, *Statistics and Computing*, 28, 2, pp. 277-290.

Leimkuhler, Benedict & Sebastian Reich

2004 *Simulating hamiltonian dynamics*, 14, Cambridge University Press.

Lewis, Antony

2019 “GetDist: a Python package for analysing Monte Carlo samples”, *ArXiv e-prints*, eprint: `1910.13970`.

Li, Tiancheng, Miodrag Bolic & Petar M Djuric

2015 “Resampling methods for particle filtering: classification, implementation, and strategies”, *IEEE Signal Processing Magazine*, 32, 3, pp. 70-86.

Lingenheil, Martin, Robert Denschlag, Gerald Mathias & Paul Tavan

2009 “Efficiency of exchange schemes in replica exchange”, *Chemical Physics Letters*, 478, 1-3, pp. 80-84.

Liu, Jun S & Rong Chen

1998 “Sequential Monte Carlo methods for dynamic systems”, *Journal of the American Statistical Association*, 93, 443, pp. 1032-1044.

Liu, Jun S & Jun S Liu

2001 *Monte Carlo strategies in scientific computing*, Springer, vol. 10.

MacKay, David JC

2003 *Information theory, inference and learning algorithms*, Cambridge University Press.

Madigan, David & Adrian E Raftery

1994 “Model selection and accounting for model uncertainty in graphical models using Occam’s window”, *Journal of the American Statistical Association*, 89, 428, pp. 1535-1546.

Madigan, David, Adrian E Raftery, C Volinsky & Jennifer Hoeting

1996 “Bayesian model averaging”, in *Proceedings of the AAAI Workshop on Integrating Multiple Learned Models, Portland*, pp. 77-83.

Meng, Xiao-Li & Wing Hung Wong

1996 “Simulating ratios of normalizing constants via a simple identity: a theoretical exploration”, *Statistica Sinica*, pp. 831-860.

Metropolis, N

1987 "The beginning", *Los Alamos Science*, 15, pp. 125-130.

Metropolis, N, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller & Edward Teller

1953 "Equation of state calculations by fast computing machines", *The Journal of Chemical Physics*, 21, 6, pp. 1087-1092, DOI: `10.1063/1.1699114`.

Moss, Adam

2020 "Accelerated Bayesian inference using deep learning", *Monthly Notices of the Royal Astronomical Society*, 496, 1, pp. 328-338.

Müller, Peter

1991 *A generic approach to posterior integration and Gibbs sampling*, Purdue University, Department of Statistics.

1992 "Alternatives to the Gibbs sampling scheme".

Murray, Iain, Ryan Adams & David MacKay

2010 "Elliptical slice sampling", in *Proceedings of the thirteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, pp. 541-548.

Naesseth, Christian A, Fredrik Lindsten, Thomas B Schön, et al.

2019 "Elements of sequential monte carlo", *Foundations and Trends in Machine Learning*, 12, 3, pp. 307-392.

Neal, Radford M

1992 "Bayesian learning via stochastic dynamics", *Advances in Neural Information Processing Systems*, 5.

1993 *Probabilistic inference using Markov chain Monte Carlo methods*, Department of Computer Science, University of Toronto.

1996 *Bayesian Learning for Neural Networks*, Springer.

1997 "Markov chain Monte Carlo methods based onslicing'the density function", *Preprint*.

2001 "Annealed importance sampling", *Statistics and computing*, 11, 2, pp. 125-139.

2003 "Slice sampling", *The Annals of Statistics*, 31, 3, pp. 705-767.

2008 *The harmonic mean of the likelihood: worst Monte Carlo method ever*, `http://radfordneal.wordpress.com/2008/08/17/the-harmonic-mean-of-the-likelihood-worstmonte-carlo-method-ever`.

Neal, Radford M et al.

2011 "MCMC using Hamiltonian dynamics", *Handbook of Markov chain Monte Carlo*, 2, 11, p. 2.

Nesterov, Yurii

2009 "Primal-dual subgradient methods for convex problems", *Mathematical Programming*, 120, 1, pp. 221-259.

Newton, Michael A & Adrian E Raftery

1994 "Approximate Bayesian inference with the weighted likelihood bootstrap", *Journal of the Royal Statistical Society: Series B (Methodological)*, 56, 1, pp. 3-26.

Nishihara, Robert, Iain Murray & Ryan P Adams

2014 "Parallel MCMC with generalized elliptical slice sampling", *The Journal of Machine Learning Research*, 15, 1, pp. 2087-2112.

Papamakarios, George, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed & Balaji Lakshminarayanan

2021 "Normalizing flows for probabilistic modeling and inference", *Journal of Machine Learning Research*, 22, 57, pp. 1-64.

Papamakarios, George, Theo Pavlakou & Iain Murray

2017 "Masked autoregressive flow for density estimation", *Advances in Neural Information Processing Systems*, 30.

Pasarica, Cristian & Andrew Gelman

2010 "Adaptively scaling the Metropolis algorithm using expected squared jumped distance", *Statistica Sinica*, pp. 343-364, DOI: 10.2139/ssrn.1010403.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al.

2019 "Pytorch: An imperative style, high-performance deep learning library", *Advances in Neural Information Processing Systems*, 32.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al.

2011 "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research*, 12, pp. 2825-2830, DOI: 10.5555/1953048.

Petigura, Erik A., Andrew W. Howard, Eric D. Lopez, Katherine M. Deck, Benjamin J. Fulton, Ian J. M. Crossfield, David R. Ciardi, Eugene Chiang, Eve J. Lee, Howard Isaacson, Charles A. Beichman, Brad M. S. Hansen, Joshua E. Schlieder & Evan Sinukoff

2016 "Two Transiting Low Density Sub-Saturns from K2", *The Astrophysical Journal*, 818, 1, p. 36, DOI: 10.3847/0004-637X/818/1/36, eprint: 1511.04497.

Raftery, Adrian E, Steven Lewis & Jeffrey D Banfield

1991 *Three Short Papers on Sampling-Based Inference: 1. How Many Iterations in the Gibbs Sampler? 2. Model Determination. 3. Spatial Statistics*, tech. rep., Washington University, Department of Statistics.

Rasmussen, Carl Edward

2003 "Gaussian processes in machine learning", in *Summer School on Machine Learning*, Springer, pp. 63-71.

Riley, Kenneth Franklin, Michael Paul Hobson & Stephen John Bence

1999 *Mathematical methods for physics and engineering.*

Robbins, Herbert & Sutton Monro

1951 "A Stochastic Approximation Method", *The Annals of Mathematical Statistics*, 22, 3, pp. 400-407, DOI: `10.1214/aoms/1177729586`, `https://doi.org/10.1214/aoms/1177729586`.

Robert, Christian et al.

2007 *The Bayesian choice: from decision-theoretic foundations to computational implementation*, Springer, vol. 2.

Robert, Christian P., Victor Elvira, Nick Tawn & Changye Wu

2018 "Accelerating MCMC Algorithms", *ArXiv e-prints*, eprint: `1804.02719`.

Roberts, Gareth O & Jeffrey S Rosenthal

1998 "Optimal scaling of discrete approximations to Langevin diffusions", *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60, 1, pp. 255-268.

2007 "Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms", *Journal of Applied Probability*, 44, 2, pp. 458-475.

Roberts, Gareth O & Osnat Stramer

2002 "Langevin diffusions and Metropolis-Hastings algorithms", *Methodology and Computing in Applied Probability*, 4, 4, pp. 337-357.

Sellentin, Elena & Alan F Heavens

2015 "Parameter inference with estimated covariance matrices", *Monthly Notices of the Royal Astronomical Society: Letters*, 456, 1, pp. L132-L136.

Shannon, Claude Elwood

1948 "A mathematical theory of communication", *The Bell System Technical Journal*, 27, 3, pp. 379-423.

Sharma, Sanjib

2017 “Markov chain Monte Carlo methods for Bayesian data analysis in astronomy”, *Annual Review of Astronomy and Astrophysics*, 55, pp. 213-259.

Shore, John & Rodney Johnson

1980 “Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy”, *IEEE Transactions on Information Theory*, 26, 1, pp. 26-37.

Silverman, Bernard W

2018 *Density estimation for statistics and data analysis*, Routledge.

Sivia, Devinderjit & John Skilling

2006 *Data analysis: a Bayesian tutorial*, OUP Oxford.

Skilling, John

2004 “Nested sampling”, in *AIP Conference Proceedings*, 1, American Institute of Physics, vol. 735, pp. 395-405.

2006 “Nested sampling for general Bayesian computation”, *Bayesian Analysis*, 1, 4, pp. 833-859.

2012 “Bayesian computation in big spaces-nested sampling and Galilean Monte Carlo”, in *AIP Conference Proceedings 31st*, 1, American Institute of Physics, vol. 1443, pp. 145-156.

2019 “Galilean and Hamiltonian Monte Carlo”, *Multidisciplinary Digital Publishing Institute Proceedings*, 33, 1, p. 19.

Sokal, Alan

1997 “Monte Carlo methods in statistical mechanics: foundations and new algorithms”, in *Functional integration*, Springer, pp. 131-192.

Speagle, Joshua S

2019 “A conceptual introduction to Markov chain Monte Carlo methods”, *arXiv preprint arXiv:1909.12313*.

2020 “dynesty: a dynamic nested sampling package for estimating Bayesian posteriors and evidences”, *Monthly Notices of the Royal Astronomical Society*, 493, 3, pp. 3132-3158.

Swendsen, Robert H & Jian-Sheng Wang

1986 “Replica Monte Carlo simulation of spin-glasses”, *Physical Review Letters*, 57, 21, p. 2607.

Tamosiunas, Andrius

2020 “Testing and Emulating Modified Gravity on Cosmological Scales”, *ArXiv e-prints*, eprint: `2011.08786`.

Ter Braak, Cajo JF

2006 "A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: easy Bayesian computing for real parameter spaces", *Statistics and Computing*, 16, 3, pp. 239-249, DOI: `10.1007/s11222-006-8769-1`.

Ter Braak, Cajo JF & Jasper A Vrugt

2008 "Differential evolution Markov chain with snooker updater and fewer chains", *Statistics and Computing*, 18, 4, pp. 435-446.

Tibbits, Matthew M, Chris Groendyke, Murali Haran & John C Liechty

2014 "Automated factor slice sampling", *Journal of Computational and Graphical Statistics*, 23, 2, pp. 543-563.

Tierney, Luke

1994 "Markov chains for exploring posterior distributions", *The Annals of Statistics*, pp. 1701-1728.

Tierney, Luke & Joseph B Kadane

1986 "Accurate approximations for posterior moments and marginal densities", *Journal of the American Statistical Association*, 81, 393, pp. 82-86.

Tran, Khoa T & Brett Ninness

2015 "Reunderstanding slice sampling as parallel MCMC", in *2015 IEEE Conference on Control Applications (CCA)*, IEEE, pp. 1197-1202.

Trotta, Roberto

2017 "Bayesian methods in cosmology", *arXiv preprint arXiv:1701.01467*.

Umeh, Obinna, Roy Maartens, Hamsa Padmanabhan & Stefano Camera

2021 "The effect of finite halo size on the clustering of neutral hydrogen", *ArXiv e-prints*, eprint: `2102.06116`.

Van der Vaart, Aad W

2000 *Asymptotic statistics*, Cambridge University Press, vol. 3.

Van Der Walt, Stefan, S Chris Colbert & Gael Varoquaux

2011 "The NumPy array: a structure for efficient numerical computation", *Computing in Science and Engineering*, 13, 2, pp. 22-30, DOI: `10.1109/MCSE.2011.37`.

Vehtari, Aki, Andrew Gelman & Jonah Gabry

2017 "Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC", *Statistics and Computing*, 27, 5, pp. 1413-1432.

Virtanen, Pauli, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al.

2020 "SciPy 1.0: fundamental algorithms for scientific computing in Python", *Nature Methods*, 17, 3, pp. 261-272, DOI: 10.1038/s41592-019-0686-2.

Vretinaris, George, Stamatis Vretinaris, Christos Mermigkas, Minas Karamanis & Nikolaos Stergioulas

2022 "Robust and fast parameter estimation of gravitational waves from neutron star merger remnants", *in prep.*

Vrugt, Jasper A, CJF Ter Braak, CGH Diks, Bruce A Robinson, James M Hyman & Dave Higdon

2009 "Accelerating Markov chain Monte Carlo simulation by differential evolution with self-adaptive randomized subspace sampling", *International Journal of Nonlinear Sciences and Numerical Simulation*, 10, 3, pp. 273-290.

Wang, Mike, Florian Beutler & David Bacon

2020 "Impact of relativistic effects on the primordial non-Gaussianity signature in the large-scale clustering of quasars", *MNRAS*, 499, 2 (Dec. 2020), pp. 2598-2607, DOI: 10.1093/mnras/staa2998, eprint: 2007.01802.

Waskom, Michael L.

2021 "seaborn: statistical data visualization", *Journal of Open Source Software*, 6, 60, p. 3021, DOI: 10.21105/joss.03021, https://doi.org/10.21105/joss.03021.

Watanabe, Sumio & Manfred Opper

2010 "Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory." *Journal of Machine Learning Research*, 11, 12.

Williams, Michael J, John Veitch & Chris Messenger

2021 "Nested sampling with normalizing flows for gravitational-wave inference", *Physical Review D*, 103, 10, p. 103006.

Wu, Changye, Julien Stoehr & Christian P Robert

2018 "Faster Hamiltonian Monte Carlo by learning leapfrog scale", *arXiv preprint arXiv:1810.04449*.

Zuntz, J., M. Paterno, E. Jennings, D. Rudd, A. Manzotti, S. Dodelson, S. Bridle, S. Sehrish & J. Kowalkowski

2015 "CosmoSIS: Modular cosmological parameter estimation", *Astronomy and Computing*, 12, pp. 45-59, DOI: `10.1016/j.ascom.2015.05.005`, eprint: `1409.3409`.