University of Reading

Sequential Monte Carlo methods in Phylogenetics and Active Subspaces

Thesis submitted for the degree of Doctor of Philosophy

Department of Mathematics and Statistics

Leonardo Ripoli

UNIVERSITY OF READING

Date of submission: 31 May 2024

Declaration:

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Leonardo Ripoli

Acknowledgments

I would like to express my deepest gratitude to my PhD supervisor, Richard Everitt, for his patience, invaluable guidance, and support throughout the research journey. His insights and expertise have been vital in shaping this work. I also wish to thank Prof. Patrizia Gamba and Prof. Enkeleida Lushi, whose passion for Mathematics and inspiring teaching have had a profound and lasting impact on my journey. Finally, I wish to thank Andrew Meade for accepting to come onboard and for the support.

Abstract

The first part of my research concentrates on Sequential Monte Carlo (SMC) methods for phylogenetics. The aim of the research was to deliver methods that can be used in the inference of the spread of diseases, leveraging a widely used software platform, and enable researchers to easily access, validate and compare. We show the results obtained from developing and integrating an adaptive SMC algorithm in Bayesian Evolutionary Analysis by Sampling Trees version 2 (commonly known as BEAST2), a well-established software platform among researchers. Our adaptive SMC algorithm embedded in BEAST2 has comparable performances to the native Markov Chain Monte Carlo (MCMC) method, in terms of accuracy and efficiency. Our work can be seen as a first step, and future tuning is expected. It is foreseen that an integration of the adaptive SMC package into BEAST2 will be done by the owners of the platform, allowing researchers to use SMC instead of MCMC, following testing and tuning by the platforms developers.

The focus of the second part is Active Subspaces (AS). With AS we try to identify a smaller subspace informed by the data and to concentrate the algorithmic effort on this more informative part, primarily to address the *curse of dimensionality* affecting many Monte Carlo (MC) methods. Existing AS algorithms were mostly biased and targeting distributions only close by some measure to the posterior, leaving users to do substantial post-validation. We built on the foundations of an existing pseudo-marginal-based Active Subspace algorithm and developed non-biased AS algorithms that in stationarity target the correct posterior, using the structure provided by AS within a Gibbs-style MCMC, and within Particle Marginal Metropolis Hastings (PMMH), Metropolis within Particle Gibbs (MwPG), SMC-squared (SMC^2). We have run experiments that show in specific settings to outperform existing methods and provide explanations on the optimal running conditions for each algorithm. Our work sheds light on the practical applications of Active Subspaces, expanding the range of AS methods available to researchers and providing clearer guidance on which approaches are most effective in specific scenarios.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The first part of my research concentrates on Sequential Monte Carlo (SMC) methods for phylogenetics. The aim of the research was to deliver methods that can be used in the inference of the spread of diseases, leveraging a widely used software platform, and enable researchers to easily access, validate and compare. We have tried to solve a common problem in phylogenetics, the understanding the evolutionary relationships between species starting from genetic data. Two big challenges presently stand in the way. Firstly, existing algorithms are often computationally expensive and not adaptable to online inference, it has been shown for example during the recent COVID outbreak, when there was a clear need for methods that could adapt to rapid variations of the virus [Wu et al., 2020]. Secondly, when using Bayesian inference, Monte Carlo traditional methods like Markov Chain Monte Carlo (MCMC) struggle in high-dimensional spaces. In our research we have tried to address these two topics. We show the results obtained from developing and integrating an adaptive SMC algorithm in Bayesian Evolutionary Analysis by Sampling Trees version 2 (commonly known as BEAST2) [Bouckaert et al., 2019], a well-established software platform among researchers. Our adaptive SMC algorithm embedded in BEAST2 has comparable performances to the native Markov Chain Monte Carlo (MCMC) method, in terms of accuracy and efficiency. Our work can be seen as a first step, and future tuning is expected. Although we have done our work independently, our implementation can be said to integrate all the various algorithms of [Wang et al., 2019] in BEAST2. It is foreseen that an integration of the adaptive SMC package into BEAST2 will be done by the owners of the platform, allowing researchers to use SMC instead of MCMC, following testing and tuning by the platforms developers.

Following up on the aim to bring simplification in complex systems, the focus of the second part of the thesis is Active Subspaces (AS) [Constantine, 2015]. With AS we try to identify a smaller subspace informed by the data and to concentrate the algorithmic effort on this

more informative part, primarily to address the *curse of dimensionality* affecting many Monte Carlo (MC) methods. Existing AS algorithms were mostly biased and targeting distributions only close by some measure to the posterior, leaving users to do substantial post-validation [Constantine et al., 2016]. We built on the foundations of an existing pseudo-marginal-based Active Subspace algorithm [Schuster et al., 2017] and developed non-biased AS algorithms that in stationarity target the correct posterior, using the structure provided by AS within a Gibbs-style MCMC [Geman and Geman, 1984], and within Particle Marginal Metropolis Hastings (PMMH) [Andrieu et al., 2010], Metropolis within Particle Gibbs (MwPG) [Andrieu et al., 2010], SMC-squared (SMC^2) [Chopin et al., 2012]. By embedding AS into the theoretical framework of Gibbs, PMMH, MwPG, and SMC^2, we ensure that the convergence properties of the original methods are preserved. This means that convergence does not need to be re-demonstrated, as it is already guaranteed by the original algorithms. We have run experiments that show in specific settings to outperform existing methods and provide explanations on the optimal running conditions for each algorithm. Our work sheds light on the practical applications of Active Subspaces, expanding the range of AS methods available to researchers and providing clearer guidance on which approaches are most effective in specific scenarios.

1.2 Thesis Organisation

This thesis is structured into six main chapters, with a logical progression from fundamental theoretical principles to novel algorithmic contributions and their applications:

- **Chapter 2: Technical Background** This chapter introduces the main technical concepts used in this research: Bayesian statistics and Monte Carlo methods.

- **Chapter 3: Models of Genetic Evolution** This chapter explain some history of genetic evolution frameworks like the Wright-Fisher model [Wright, 1931, Fisher, 1930], coalescent theory [Kingman, 1982], substitution models, for example Jukes Cantor [Jukes and Cantor, 1969], and existing software like BEAST2 [Bouckaert et al., 2019]. We then present our algorithm an application of adaptive SMC to genetic sequences and show the results in a scenario with 10 taxa with the comparison with BEAST2 native MCMC: our adaptive SMC shows results comparable to the native BEAST2 MCMC, with far fewer samples. The early results are promising, additional future effort is advisable in the integration of the method within a complex software platform as BEAST2.

- **Chapter 4: Active Subspaces**: We provide an introduction of the mathematics behind Active Subspaces (AS) [Constantine, 2015], and application in Monte Carlo methods [Constantine et al., 2016, Schuster et al., 2017]. The chapter evaluates existing AS approaches for MCMC, highlights their limitations, to mention a few: biasedness,

non exactness, use of prior for approximations. The chapter sets the ground for the algorithmic improvements introduced in subsequent chapters.

- **Chapter 5: Active Subspaces proposed novel MCMC algorithms**: the chapter introduces new MCMC algorithms developed using the structure provided by AS within a Gibbs-style MCMC [Geman and Geman, 1984], and within Particle Marginal Metropolis Hastings (PMMH) [Andrieu et al., 2010], Metropolis within Particle Gibbs (MwPG) [Andrieu et al., 2010], we named the new algorithms AS-Gibbs, AS-PMMH, and AS-MwPG. By embedding AS into the theoretical framework of Gibbs, PMMH, MwPG, we ensure that the convergence properties of the original methods are preserved. The performance of the new algorithms is compared, in efficiency and accuracy, using two main models that we named *Gaussian* 5.3.2 and *Banana* 5.4.2. We show that As-Gibbs and AS-MwPG-i (a second variant of AS-MwPG), clearly outperform existing algorithms in specific settings. In addition, we introduced a novel method to determine the Active Subspace dimension, alternative to the traditional, eigenvalues-based method. In our tests, the two methods give identical results.

- **Chapter 6: Active Subspaces proposed novel SMC algorithms**: The chapter introduces new SMC algorithms based on Active Subspaces: named AS-SMC (AS-based SMC), AS-SMC^2, developed using the structure provided by AS within SMC^2 [Chopin et al., 2012], and an adaptive version of AS-SMC, named AS-SMC-a. This chapter provides theoretical justifications, experimental results, and comparisons with existing methods. By embedding AS into the theoretical framework of SMC^2, we ensure that the convergence properties of the original method are preserved. We show that AS-SMC-a, in particular, shows promising results in cases that would have resulted in poor approximation when using traditional AS methods.

- **Chapter 7: Conclusions and Future Work** Summarizes the contributions and findings of the thesis, discusses their broader implications, and proposes directions for future research.

Chapter 2

Technical Background

This chapter introduces the technical background for the methods and algorithms presented in this thesis. It introduces the main concepts and techniques of Bayesian statistics and Monte Carlo methods that will be used in subsequent chapters.

2.1 Bayesian Statistics

In terms of statistical modelling, the problem we are interested in is defining the distribution of some parameters of a model that we want to estimate

$$\theta \in \Theta \tag{2.1}$$

given a set of observations

$$y \in \mathcal{Y} \tag{2.2}$$

Often in the thesis we will take $\Theta = \mathbb{R}^d$ and similarly $\mathcal{Y} = \mathbb{R}^m$. In this context we express the Bayes formula as

$$\pi(\theta|y) = \frac{l(y|\theta)p(\theta)}{p(y)} = \frac{l(y|\theta)p(\theta)}{\int l(y|\theta)p(\theta)d\theta} \tag{2.3}$$

$\theta \in \Theta, y \in \mathcal{Y}$ In equation (2.3), we define:

- **marginal likelihood** the term $p(y)$ in the denominator
- **posterior** the term $\pi(\theta|y)$ on the left hand side of the equation
- **prior** the term $p(\theta)$
- **likelihood** the term $l(y|\theta)$

The prior can be considered as the a-priori information we have on the distribution of the parameters; the likelihood is the information that allows us to update the model after the data y that we observe.

Equation 2.3 is the *normalised* posterior, we will sometimes refer to the *un-normalised* version omitting the marginal likelihood

$$\tilde{\pi}(\theta|y) \propto l(y|\theta)p(\theta) \tag{2.4}$$

We may also refer to the posterior as $\pi(\theta)$ instead of $\pi(\theta|y)$, omitting the conditioning on the observations y for brevity, similarly for the likelihood $l(y|\theta)$ which we may refer to as $l(\theta)$.

2.2 The Monte Carlo method

In Bayesian statistics, we often need to calculate expectations with respect to a distribution p:

$$\mu = E_{p(\theta)}(f(\theta)) = \int f(\theta)p(\theta)d\theta \tag{2.5}$$

However, there are cases where the integral in expectation (2.5) cannot be computed in closed form. For example, in the case of the posterior π introduced in equation (2.3), either because the normalisation constant is unknown or the integral itself is intractable. Moreover, calculation may also be computationally infeasible with traditional numerical integration methods. As shown in [Caflisch, 1998], the complexity of a reference numerical integration scheme is $O(N^{-\frac{k}{d}})$, where k is the order of the scheme and d the dimension of the space. This slow convergence rate in high dimensions explains the term *curse of dimensionality*, which refers to the way increasing the dimension of the state space progressively worsens convergence rates.

We will, in this section, explain the foundations of **Monte Carlo methods** [Metropolis and Ulam, 1949] [Robert and Casella, 2004]. If we are able to draw N independent samples $\theta_1, \theta_2 ... \theta_N$ from $p(\theta)$, we can consider the approximation

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} f(\theta_i) \tag{2.6}$$

The quantity $\hat{\mu}$ of (2.6) approximates the expectation

$$\frac{1}{N}\sum_{i=1}^{N} f(\theta_i) \approx \int f(\theta)p(\theta)d\theta \tag{2.7}$$

The left-hand-side sum of equation (2.7) equals the right-hand-side integral almost surely in the limit $N \to \infty$ by the **strong law of large numbers**, that states that the average of many independent r.v. with common mean and finite variances tend to stabilize around their mean (note that in the left-hand-side the $p(\theta)$ is implicitly approximated by the random

measure since we are drawing the i samples from it):

$$\hat{\mu} \xrightarrow[N\to\infty]{a.s.} \mu \tag{2.8}$$

We will explore in the following sections of this chapter how to deal with cases when we are not able to draw the samples $\theta_1, \theta_2 ... \theta_N$ directly from the distribution $p(\theta)$, and we will see how to write an approximation conceptually similar to equation (2.6). The estimator (2.6) is unbiased, i.e.

$$E(\hat{\mu}) = \mu \tag{2.9}$$

If we define the **Monte Carlo Integration error** as

$$\epsilon_N = \frac{1}{N}\sum_{i=1}^{N} f(\theta_i) - \int f(\theta)p(\theta)d\theta \tag{2.10}$$

we can study the convergence of the MC method via the the **Central Limit Theorem (CLT)** that states that (see for example [Caflisch, 1998]) for N large, since we are considering samples from a population with mean μ and finite variance

$$\sigma^2 = E\left(\theta^2\right) \tag{2.11}$$

we have that ϵ_N converges in distribution to a Normal r.v., in particular

$$\sqrt{N}\epsilon_N \xrightarrow[N\to\infty]{\mathcal{D}} \mathcal{N}(0, \sigma_f) \tag{2.12}$$

With σ_f the standard deviation of f

$$\sigma_f^2 = E_{p(\theta)}(f(\theta)^2) - \mu^2 \tag{2.13}$$

Convergence of Monte Carlo methods

Formula (2.12) indicates that the rate of convergence of the standard Monte Carlo method is $O(N^{-\frac{1}{2}})$, with a multiplicative constant equal to σ_f, independently from the dimension d of the state space, and gives the reason for using Monte Carlo methods instead of, for example, numerical integration methods. In fact the theoretical convergence rate of nearly all MC methods is $O(N^{-\frac{1}{2}})$, as seen in formula (2.12), it is the constant in front of this rate that can make a difference among the various MC methods.

2.3 Importance Sampling

Importance sampling (IS) provides a way to estimate integrals by the use of an instrumental auxiliary distribution, named the **proposal distribution**. In detail, let's assume that we

want to calculate the expectation of a function f according to the density p in a domain $\mathbf{D} \subseteq \mathbb{R}^d$

$$\mu = E_p(f(\theta)) = \int_{\mathbf{D}} f(\theta)p(\theta)d\theta \tag{2.14}$$

we may want to approximate the integral via use of the sum as in equation (2.6), but let's assume we cannot draw easily samples from the distribution p. We can use a proposal distribution, g, easier to draw from: it is sufficient that $g(\theta) > 0$ where $f(\theta)p(\theta) \neq 0$. The validity of the process is shown by the equivalences below [Owen, 2013]

$$\begin{aligned} E_g\left(\frac{f(\theta)p(\theta)}{g(\theta)}\right) &= \int \frac{f(\theta)p(\theta)}{g(\theta)} g(\theta)d\theta \\ &= \int f(\theta)p(\theta)d\theta = E_p(f(\theta)) = \mu \end{aligned} \tag{2.15}$$

Therefore the importance sampling estimate becomes

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} \frac{p(\theta_i)}{g(\theta_i)} f(\theta_i) = \frac{1}{N}\sum_{i=1}^{N} w(\theta_i)f(\theta_i),\ \theta_i \sim g \tag{2.16}$$

Where, in equation (2.16), the so-called weights are defined as follows

$$w(\theta) = \frac{p(\theta)}{g(\theta)} \tag{2.17}$$

The weights w compensate for sampling from the proposal function g, instead of the original distribution p. Therefore, we can sample $\theta_1, \theta_2, ..., \theta_N$ independently from the proposal distribution g, and due to the LLN (as in formula (2.6)) we have

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} w(\theta_i)f(\theta_i) \xrightarrow[N\to\infty]{a.s.} E_p(f(\theta)) \tag{2.18}$$

Similarly to equation (2.9) (see also [Johansen and Evers, 2007] and [Owen, 2013]), it is easy to demonstrate that $\hat{\mu}$ of equation (2.16) is an unbiased estimator of the mean, i.e. that

$$E_g(\hat{\mu}) = \mu \tag{2.19}$$

and that [Owen, 2013]

$$Var_g(\hat{\mu}) = \frac{\sigma^2_{\text{IS}}}{N} \tag{2.20}$$

where

$$\sigma^2 = \int_{\mathbf{Q}} \frac{(f(\theta)p(\theta))^2}{g(\theta)}d\theta - \mu^2 = \int_{\mathbf{Q}} \frac{(f(\theta)p(\theta) - \mu g(\theta))^2}{g(\theta)}d\theta \tag{2.21}$$

The formulae (2.21) give us a way to analyse an optimal proposal g: from the second expression in (2.21) we see that an optimal proposal will minimise the numerator $(f(\theta)p(\theta) - \mu g(\theta))$,

therefore a function g proportional to fp, and ideally [Owen, 2013]:

$$g_{opt} = \frac{|f|p}{E_p(|f|)} \tag{2.22}$$

although the g_{opt} of equation (2.22) is not practically feasible because it would mean that we can sample directly from p (which by assumption is not the case). It is therefore advisable in a good proposal choice that g is proportional to $|f|p$ (for example it has spikes where $|f|p$ does) [Owen, 2013]. We can also see from the second expression in (2.21) that small values of g in the denominator would magnify whatever lack of proportionality in the numerator between g and $|f|p$ [Owen, 2013], therefore we want a proposal that has heavier tails than p (or at least as heavy as p) [Johansen and Evers, 2007].

2.3.1 Estimating the normalization constant through IS

In Bayesian analysis we can usually only compute an un-normalised version of p, or g or both. For example, in the case of p, we may have $p = \frac{\hat{p}}{Z}$, where p is the normalised distribution and Z is the normalising constant. Let's assume without loss of generality that only $\hat{p}$ is un-normalised, we have therefore that

$$\int_{\Theta} \hat{p}(\theta) d\theta = Z \tag{2.23}$$

We can use IS to estimate the normalising constant by considering that, from equation (2.23), we have

$$E_g(w(\theta)) = \int \frac{\hat{p}(\theta)}{g(\theta)} g(\theta) d\theta = \int \hat{p}(\theta) d\theta = Z \tag{2.24}$$

Therefore using again formulae of (2.16) and (2.18) applied to equation (2.24), we have that

$$\hat{Z} = \frac{1}{N} \sum_{i=1}^{N} w_i \tag{2.25}$$

The $\hat{Z}$ of equation (2.25) is the estimate of the normalising constant Z up to which we know the distribution p (a similar procedure can be applied if g is known up to a normalising constant). Using the result of (2.25), we can write a "self normalised" version of the estimate of equations (2.16) and (2.18), as follows

$$\hat{\mu} = \frac{\sum_{i=1}^{N} w(\theta_i) f(\theta_i)}{\sum_{i=1}^{N} w_i} = \sum_{i=1}^{N} W_i f(\theta_i) \tag{2.26}$$

The normalised weights of equation (2.26) are such that

$$W_i = \frac{w_i}{\sum_{j=1}^{N} w_j} \tag{2.27}$$

Formula (2.26) is the "self normalised" version of equations (2.16) and (2.18), with the property that the weights $\hat{w}_i$ of equation (2.26) add up to 1 (as seen from (2.24) and (2.25), this means that we are simulating drawing from normalised distributions). It is not difficult to show that (see for example [Johansen and Evers, 2007])

$$E_g(\hat{\mu}) = \mu + \frac{\mu \text{Var}(w(\theta)) - Cov_g(w(\theta), w(\theta)f(\theta))}{N} + O(N^{-2}) \tag{2.28}$$

Where Cov_g in (2.28) is the covariance

$$Cov_g(w(\theta), w(\theta) \cdot f(\theta)) = E_g\left[(w(\theta) - E_g(w(\theta)))\,(w(\theta) \cdot f(\theta) - E_g(w(\theta) \cdot f(\theta)))\right], \tag{2.29}$$

And that the variance is

$$Var_g(\hat{\mu}) = \frac{\text{Var}(w(\theta)f(\theta)) - 2\mu Cov_g(w(\theta), w(\theta)f(\theta)) + \mu Var_g(w(\theta))}{N} + O(N^{-2}) \tag{2.30}$$

Where

We can therefore see from (2.28) that $\hat{\mu}$ is biased, it has though the advantage that it can be calculated knowing the density up to a constant, in fact a normalising constant of the density would cancel out in the calculation of $\hat{\mu}$ (as shown in [Johansen and Evers, 2007]). It has to be noted that however the bias in (2.28) decreases with increasing N.

2.3.2 Effective Sample Size (ESS)

As we have seen in the previous sections, the Importance Sampling method allows us to perform calculation of integrals, for example the expectation of a function f w.r.t. a density p, like in equation (2.14), by using samples drawn from a proposal distribution g, with a sum like in equation (2.16). An obvious question to ask is how does the IS approximation compare with the usual Monte Carlo approximation [Kong, 1992] [Kong et al., 1994] of the integral that we would have by drawing samples from the distribution p, as in (2.6). The remainder of this section will be dedicated to answering this question, using the logic outlined in [Elvira et al., 2018]. For ease, we rewrite here some definitions used in the previous sections, changing slightly the notation (to make it coherent with [Elvira et al., 2018]). We start from the integral of an expectation

$$I = E_p(f) = \int f(\theta)p(\theta)d\theta \tag{2.31}$$

Then we call $\bar{I}$ the Monte Carlo approximation of (2.31)

$$\bar{I} = \frac{1}{N}\sum_{i=1}^{N} f(\theta^{(i)}),\ \theta^{(i)} \sim p \tag{2.32}$$

And we call $\tilde{I}$ the self normalised IS approximation of (2.31) (as in (2.26))

$$\tilde{I} = \sum_{i=1}^{N} W(\theta^{(i)}) f(\theta^{(i)}),\ \theta \sim g \tag{2.33}$$

Where, in equation (2.33), W are the self-normalised weights of equation (2.27). A measure that has been widely used [Owen, 2013] [Elvira et al., 2018] to compare the performance of IS estimators, is the so-called **Effective Sample Size (ESS)**, which compares the variances of the traditional Monte Carlo estimate (2.32) and the IS approximation (2.33)

$$ESS = N \frac{Var(\bar{I})}{Var(\tilde{I})} \tag{2.34}$$

We can notice that the ESS of (2.34) has some drawbacks, for example it depends on the integrand function f (as clearly seen from (2.31), (2.32) and (2.33)), and therefore an estimator that is good for an integrand function f_1 may not in general be good for another function f_2, and also in order to calculate (2.34) we will need to compute integrals that are in general intractable as the integral (2.31) that we are trying to estimate (see for example [Elvira et al., 2018] for a detailed expression of such integrals). Therefore some simplifications have been used [Elvira et al., 2018] [Kong et al., 1994] that reduce significantly the complexity of (2.34), to:

$$ESS \approx \frac{N}{1 + Var_g(W)} \tag{2.35}$$

We see from equation (2.35) that, in the ideal case where the weights are known exactly (and therefore with zero associated variance), we have $ESS = N$, i.e. we are in a situation that is as good as if we were drawing directly from the target distribution. As we see in [Elvira et al., 2018] [Kong, 1992] [Kong et al., 1994], further simplification of (2.35) bring to

$$ESS \approx \frac{NZ^2}{E_g(W^2)} \tag{2.36}$$

Where Z is the normalising constant expressed in (2.23). It is to be noted that (2.36) has, as said, approximations and that these restrict the validity of (2.36) to cases where the approximations are valid [Kong et al., 1994] [Elvira et al., 2018] (for example since there is no dependence on the integrand function f, it is assumed that the proposal g is "reasonably" close to the optimal proposal (2.22)). By using particle approximations for Z from (2.25) and $E_g(W^2) \approx \frac{1}{N} \sum_{j=i}^{N} (w^{(j)})^2$ which brings us to the final approximation of ESS in the version widely used in literature:

$$\hat{ESS} = N \frac{(\sum_{j=i}^{N} w^{(j)})^2}{\sum_{j=i}^{N} (w^{(j)})^2} = \frac{1}{\sum_{j=i}^{N} (W^{(j)})^2} \tag{2.37}$$

Where, in (2.37), in the first equation the $w^{(j)}$ are unnormalised weights of equation (2.17), whereas in the second equation the $W^{(j)}$ are the self-normalised weights of equation (2.27).

2.4 Markov Chain Monte Carlo (MCMC)

We will introduce in this section Markov Chain Monte Carlo (MCMC). Similarly to Monte Carlo methods in previous sections, MCMC provides a way to approximate drawing samples from a distribution that we cannot directly draw from. Unlike the traditional MC and Importance Sampling, MCMC samples are not independently distributed: as we can see from equation (2.38) in fact, there is conditional dependence between values. A stochastic process is defined Markov Chain Monte Carlo if, considering $X_1, X_2, ...X_N$ random variables, that are the realization of the process, defined on a common probability space $(\chi, \mathcal{A}, P)$, the following so-called **Markov property** holds:

$$P(X^{(t)} = x^{(t)}|X^{(t-1)} = x^{(t-1)}, ..., X^{(1)} = x^{(1)}) = P(X^{(t)} = x^{(t)}|X^{(t-1)} = x^{(t-1)}) \tag{2.38}$$

As we can see from (2.38), the value of the chain at a particular time t is only dependent from the value at time $t-1$. In the following parts of this section we will outline the basic concepts that will help us introduce MCMC [Robert and Casella, 2004].

2.4.1 Markov Kernel

Considering two measurable spaces (X, A), (A, B) a transition kernel is a map [Robert and Casella, 2004] $K : A \times B \to [0, 1]$, s.t.

- $\forall x \in \chi$, $k(x, \cdot)$ is a probability measure [Jiao, 2017]
- $\forall B_i \in B$, $k(\cdot, B_i)$ is measurable [Jiao, 2017]

The kernel is a conditional probability density, we will speak more extensively about the associated probability measure in the next subsections. In the continuous case we have that

$$P(X_t \in B_i|X_{t-1} = x_{t-1}) = \int_{B_i} k(x_{t-1}, x_t)dx_t \tag{2.39}$$

and in the discrete case equation (2.39) becomes

$$P(X_t \in B_i|X_{t-1} = x_{t-1}) = \sum_{x' \in B_i} k(x_{t-1}, x') \tag{2.40}$$

2.4.2 Initial distribution of the chain

The chain X_n is defined for $n \in \mathbb{N}$, therefore there is a X_0, starting point of the chain. We define the initial distribution of X_0 as μ

$$P(X_0 \in A) = \int_A \mu(x)dx \tag{2.41}$$

The obvious extension to the discrete case of (2.41) is similar to formula (2.40), we have in fact

$$P(X_0 \in A) = \sum_{x_{0i} \in A} \mu(x_{0i}) \tag{2.42}$$

2.4.3 Joint and conditional distributions of the X_i

Continuing what we have said in the previous sections, in particular using the concepts of kernel k and initial distribution μ introduced in Sections 2.4.1 and 2.4.2 respectively, the joint distribution of X_1, X_2 is obtained by $\mu(x_0)k(x_0, x_1)$, and therefore [Robert and Casella, 2004]

$$P(X_1 \in A, X_0 = x_0) = \int_A \mu(x_0)k(x_0, x_1)dx_1 \tag{2.43}$$

and, de-marginalising wrt x_0 we obtain the joint distribution

$$P(X_1 \in A_1, X_0 \in A_0) = \int_{A_0} \int_{A_1} \mu(x_0)k(x_0, x_1)dx_0dx_1 \tag{2.44}$$

By using the Markovian property (2.38) and the definition of the kernel we have that [Johansen and Evers, 2007]

$$P(X_0 = x_0, ...X_n = x_n) = \mu(x_0) \prod_{j=1}^{n} k(x_{j-1}, x_j) \tag{2.45}$$

We also introduce the notation used in literature [Robert and Casella, 2004] [Johansen and Evers, 2007] $k^1(x, A) = k(x, A)$, and

$$k^s(x_t, x_{t+s}) = \int_{A^{s-1}} \prod_{j=t+1}^{t+s} k(x_{j-1}, x_j)dx_t...dx_{t+s-1} \tag{2.46}$$

so we can express (2.45) as

$$P(X_0 = x_0, ...X_n = x_n) = \mu(x_0)k^n(x_0, x_n) \tag{2.47}$$

2.4.4 Stationary property of the MCMC and invariant distribution

We will in this section introduce the concept of **stationary/invariant distribution** of the chain, i.e. a distribution π s.t. [Robert and Casella, 2004]:

$$X_n \sim \pi \implies X_{n+1} \sim \pi \tag{2.48}$$

"A σ-finite measure π is invariant for the kernel $k(\cdot, \cdot)$ and the chain X_N" [Jiao, 2017] if

$$\pi(A) = \int_{\chi} k(x, A)\pi(x)dx \tag{2.49}$$

Equation (2.49) states the condition (2.48) (if the invariant distribution is a probability measure it is also called *stationary* due to (2.48)). A theorem states [Robert and Casella, 2004] that if X_N is a **recurrent** chain then it has an invariant measure [Jiao, 2017] (*recurrence* is a property that states that, whatever the initial condition of the chain, we will end up in a set A having positive measure an infinite number of time as $N \to \infty$ [Robert and Casella, 2004]). The stationary property of the MCMC can also be related to another property, that states that the direction of time does not matter in the dynamic of the chain, $P_{X_{n+1}}(X_{n+1}|X_n = x) = P_{X_n}(X_n|X_{n+1} = x)$. This property is called **reversibility** and is stated as follows [Johansen and Evers, 2007] [Robert and Casella, 2004]:

$$k(x, y)\pi(x) = k(y, x)\pi(y) \tag{2.50}$$

Equation (2.50) is named *detailed balance condition* and provides a sufficient condition for π to be a stationary distribution for the chain. It is easy to prove that [Robert and Casella, 2004] if k and π meet criterion (2.50), then π is the invariant density of the chain [Jiao, 2017].

2.4.5 Convergence of MCMC

We are interested in understanding the conditions of convergence for the chain X_n. We have seen in Section 2.4.4 the conditions for existence of a stationary distribution for MCMC, in this section we will define the prerequisites for the MCMC chain to converge to this particular stationary distribution. We will, in this section, state the two convergence theorems of the chain to the stationary distribution: the **convergence by LLN** and, under stronger conditions, the **convergence in total variation norm**.

Convergence by LLN

Under the following conditions [Robert and Casella, 2004]: *if the chain is Harris-recurrent, with invariant measure π, then the following convergence theorem can be proved*, with any

integrable function f where we want to estimate the expectation

$$\lim_{N\to\infty} \frac{1}{N} \sum f(X_i) = \int \pi(x) f(x) dx, \ \forall X_0, \text{ a.s. by LLN} \tag{2.51}$$

(*Harris-recurrence* is a stronger property than that of recurrence mentioned earlier in the section, the main concept anyway remains that whatever the initial condition of the chain, we will end up in a set A having positive measure an infinite number of time as $N \to \infty$ [Robert and Casella, 2004]). Formula (2.51) states that, given the conditions stated, no matter the initial condition X_0, the chain will converge by the Law of Large Numbers to the stationary distribution π, for all π-integrable functions. We can see the similarities in the results of equation (2.51) with equation (2.5) which was stating LLN in the general Monte Carlo case of samples that are independent from each other: formula (2.51) states a similar result in the setting of MCMC where samples have a dependence via the Markov property (2.38).

Ergodicity and convergence in total variation norm

We start by defining an additional property of the chain which will be auxiliary in the formulation of the convergence. We define a **periodic** chain X_n which cyclically returns in the same states: mathematically, X_n is periodic with period d if there are non empty disjoint sets $A_0 ... A_{d-1}$ s.t.

$$K(x, A_j) = 1, \text{ for } j = i + 1 (\text{mod d}), \ \forall i \text{ s.t. } x \in A_i \tag{2.52}$$

If the conditions of (2.52) are not met, then X_n is aperiodic. We can now state the following conditions of convergence [Robert and Casella, 2004]: *if the chain is Harris-recurrent, aperiodic with invariant measure π, then there is convergence of the chain to the stationary distribution π, whatever the initial distribution μ*:

$$\lim_{n\to\infty} \left|\left| \int_A k^n(x, \cdot)\mu(x) - \pi \right|\right|_{TV} \tag{2.53}$$

Where k^n is the transition kernel applied n times introduced in (2.47), and the *total variation norm* is

$$||\mu_1(A) - \mu_2(A)||_{TV} = \sup_A |\mu_1(A) - \mu_2(A)| \tag{2.54}$$

Formula (2.53) states **ergodicity**. In MCMC, ergodicity ensures that, as the number of steps $n \to \infty$, the distribution of the samples generated by the chain converges to the target distribution, regardless of the starting point: over time, the chain will explore the entire space in a way that is in accordance to the target distribution.

2.4.6 Metropolis-Hastings algorithm

We now need a way to take advantage of the properties of the chain outlined in the previous sections, and build chains that converge to a stationary distribution of our choice: the **Metropolis-Hastings algorithm** (MH) [Metropolis et al., 1953, Hastings, 1970] has been built with this purpose. Suppose we want to approximate drawing samples from a target distribution π and we want to do so using MCMC. The MH algorithm allows us to approximate drawing samples from π using an auxiliary distribution g, named *proposal distribution*. We build the transition kernel k in three steps in the following way:

1. draw samples from the proposal g: $X_n^* \sim g(\cdot|X_n)$
2. calculate the *acceptance ratio* $\alpha(X_n^*|X_n) = \frac{\pi(X_n^*)g(X_n|X_n^*)}{\pi(X_n)g(X_n^*|X_n)}$
3. we draw from the uniform distribution $u \sim \text{Unif}[0,1]$, if $u \leq \alpha$ we have $X_{n+1} = X_n^*$, otherwise $X_{n+1} = X_n$

The kernel k built with the three-step procedure outlined above can be synthesised as follows:

$$k(X_n, X_{n+1}) = \alpha(X_n^*|x_n)g(X_n^*|x_n) + \mathbf{1}_{\{x_n^*=x_n\}}\left[1 - \int \alpha(s|x_n)g(s|x_n)ds\right] \tag{2.55}$$

It can be demonstrated [Johansen and Evers, 2007, Robert and Casella, 2004] that the kernel (2.55) satisfies the reversibility condition (2.50) and therefore, as explained in Section 2.4.4, π is the invariant distribution of the chain. If the proposal g is chosen so that the whole space is covered, the chain is also recurrent, and we are therefore in the conditions for convergence by LLN of equation (2.51).

2.4.7 Gibbs Sampling

Gibbs Sampling [Geman and Geman, 1984] is a Markov Chain Monte Carlo (MCMC) method which can prove particularly useful in some high-dimensional settings, especially when the conditional distributions of the posterior are easy to draw from. The method can be mixed with others that we have already introduced producing hybrid methods. We will here be giving briefly some technical details of Gibbs sampling.

Gibbs Sampler Algorithm

If we consider a set of parameters in our state space $\theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$,the Gibbs sampler is no different from other Monte Carlo methods in that it approximates the sampling from the posterior distribution $p(\theta) = p(\theta_1, \theta_2, \ldots, \theta_n)$. And like MCMC described in Section 2.4,the Gibbs method produces samples that are not independent, in fact the algorithm iteratively samples from a sequence of conditional distributions of the components i of the state space. The Gibbs Sampling algorithm can be summarized in the following steps:

1. **Initialization**: Choose an initial state for the parameters $\theta^{(0)} = (\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_n^{(0)})$.
2. **Iterative Sampling**: For each iteration $t = 1, 2, \ldots, T$:
   (a) Sample $\theta_1^{(t)}$ from $P(\theta_1 | \theta_2^{(t-1)}, \ldots, \theta_n^{(t-1)})$
   (b) Sample $\theta_2^{(t)}$ from $P(\theta_2 | \theta_1^{(t)}, \theta_3^{(t-1)}, \ldots, \theta_n^{(t-1)})$
   (c) Continue this process for each i with $i \leq n$
3. **Samples**: At the end of each iteration t we will have a new sample $\theta^{(t)} = (\theta_1^{(t)}, \theta_2^{(t)}, \ldots, \theta_n^{(t)})$.
4. **Stop condition**: As described for MCMC in 2.4, the stop condition of the algorithm can be decided after a large enough number of iterations, for example after a predetermined ESS is reached (see Section 2.4.8 for ESS)

Theoretical Foundation

The Gibbs sampler is a special case of Metropolis-Hastings, where the proposal is chosen to be the full conditional on some subset of variables (keeping the others fixed). Plugging this into the acceptance probability gives the Gibbs sampler. It is easy to demonstrate, for example, that under mild conditions on p it is possible to have Harris recurrence on the Gibbs sampler [Tierney, 1994] and therefore the results on total variation norm convergence described in Section 2.4.5.

2.4.8 ESS for MCMC

In Section 2.3.2 we have introduced the **Effective Sample Size** (ESS) in the case of Importance Sampling, as a measure of the approximation resulting from drawing samples from a proposal distribution. In IS, the ESS gives an approximate information of how many iid samples from the target a set of points from the proposal is worth.
In MCMC there is a similar approximation (see for example chapter 11 of [Gelman et al., 2013]):

$$\text{ESS}(X) = \frac{N}{1 + 2\sum_{k=1}^{\infty} \rho_k(X)} \tag{2.56}$$

Where N is the number of posterior samples, ρ_k is the autocorrelation at lag k, and X is the vector of chain samples. The infinite sum at the denominator is usually given a truncation based on some criterion, for example at k^* where $\rho_{k^*}(X) < 0.05$. The ESS of (2.56) approximately describes how many iid samples from the posterior the chain X is equivalent. We have introduced a similar measure with the same name with the Importance Sampling ESS in Section 2.3.2. As in the case of IS, for MCMC as well we have $ESS \leq N$. It will appear obvious that in the MCMC case the ESS is linked to the autocorrelation of the chain, if we think that in the ideal case independent samples would have no correlation, and therefore we want to have a measure of "distance" from the ideal case. The ESS is therefore

a useful, if approximate, measure and can also be used as a stop condition when running the chain, i.e. for example terminating the MCMC run when a predetermined ESS has been reached.

2.4.9 MultiESS

In addition to the ESS of Section 2.4.8, we have another measure in MCMC named multiESS [Vats et al., 2019]. In case of multivariate samples in a chain, multiESS can give an idea of the overall quality of samples, like the ESS. While the ESS can give a measure of the quality of samples for a single component, the multiESS takes into consideration the covariance of the various elements, and therefore gives a measure for the overall multivariate chain. The multiESS formula is as per below [Vats et al., 2019]

$$multiESS = N \left(\frac{|\Lambda|}{|\Sigma|} \right)^{-d} \tag{2.57}$$

where

- N is the number of samples in the chain;
- Λ is the variance of the posterior;
- Σ is the variance of the N samples;
- d is the dimension of the state space.

2.5 Annealed Importance Sampling (AIS)

We will, in this section, introduce a Monte Carlo method named **Annealed importance Sampling (AIS)**. Like the other methods introduced so far, the AIS is used for approximately drawing from a target distribution, in particular the AIS can be seen as an enhancement [Neal, 1998] to the Importance Sampling and MCMC techniques introduced in Sections 2.3 and 2.4, and we will see later in this section that it combines the two (MCMC and IS). Annealed Importance Sampling allows us to move from an initial tractable distribution to a target distribution of interest, which is intractable or difficult to draw from.

2.5.1 Annealed Importance Sampling vs Importance Sampling

We have seen in Section 2.3 that an important measure for IS is the Effective Sample Size (ESS), a quantity that provides us with a measure of how close the IS process is to an ideal situation where we can draw directly from the target distribution. In a nice study on the topic of dimension and computational cost in IS [Agapiou et al., 2017] it is shown that the ESS is related to the chi square distance of the proposal from the target, named ρ in the

paper, and although the exact definition of ρ is outside the scope of this chapter (please see Section 2.2 of [Agapiou et al., 2017] for the full definition), we report one of the formulae from the paper:

$$ESS \approx \frac{N}{\rho} \tag{2.58}$$

Equation (2.58) indicates that if the distance ρ of the proposal from the target is increased, we'll have to use a larger number of samples N in order to keep the ESS constant. And comparing (2.58) with (2.35) we understand that the factor driving the variance of the importance weights is the distance between the target and the proposal. Also in [Agapiou et al., 2017] it has been shown that, in some reference scenarios, the number of importance samples N has to be increased exponentially as the dimension of the state space increases, to keep the ESS at a predefined level. Therefore the approximation of the target given by the IS method in general deteriorates exponentially with the dimension. It has to be noted that the "curse of dimensionality" for IS, does not always automatically happen, as there may be cases where the intrinsic dimension of a system is less than the nominal dimension (one example would be for over-parametrised systems), we discuss quite extensively the topic in Chapter 4 where we introduce *active subspaces* [Constantine, 2015], we expect in these scenarios that the number of particles to keep a predetermined ESS will be given mainly by the dimensionality of the subspace, which is relatively constant, and will not therefore be subject to the curse of dimensionality.

Aside from the particular cases, IS is therefore affected negatively (at an exponential rate) by high dimensions. In contrast, it has been shown that [Beskos et al., 2011] [Beskos et al., 2014] sequential Monte Carlo methods like AIS and Sequential Monte Carlo (SMC, introduced later in Section 2.6), scale polynomially as the dimension d of the state space grows large. In particular it has been shown [Beskos et al., 2011] [Beskos et al., 2014] that it is possible to find an approximation s.t. the ESS remains at a predetermined level at a cost $O(Nd^2)$, where N is the number of particles and d the dimension of the state space. AIS, as we will explain later in this section, uses intermediate distributions: to bridge the gap between proposal and target, it creates proposals that are progressively closer to the target. The $O(Nd^2)$ cost of performing AIS (or SMC) comes from[Beskos et al., 2011, 2014]:

- The number of intermediate distributions between target and proposal is $O(d)$
- For each intermediate distribution, the cost of a MCMC iteration performed in a state space of dimension d is $O(d)$ as we explain in Section 4.5

And performing the two operations above for each of the N particles brings the cost of $O(Nd^2)$.

Another reason why AIS will, in general, work better than IS is in those cases where the target distribution exhibits isolated modes, especially if some important modes are found only rarely.

2.5.2 Annealed Importance Sampling vs MCMC

AIS will also in general work better than MCMC in the case of isolated modes, in fact when MCMC is used in cases of complex multi-modal distributions it will shift very infrequently among modes, therefore showing high autocorrelation and a tendency to stabilize only after an extended time duration [Neal, 1998]. The AIS, allowing to gradually approach the desired distribution of interest by making use of interpolating distributions, is an approach to avoid this.

2.5.3 The AIS algorithm

We proceed with the description of the algorithm. We state again that the aim of the AIS algorithm [Neal, 1998] is, as all other Monte Carlo methods, to approximate the drawing of samples from a target distribution of interest: let's call this normalised target distribution π_n, and f_n the associated un-normalised version (in the rest of the section we will use either or both π and f with indexes e.g. π_j and f_j for normalised and un-normalised distributions). AIS makes use of the methods of IS and MCMC, in fact it will use intermediate proposal distributions to draw from, and will make use of MCMC to move between the intermediate steps. The algorithm starts by sampling from an initial proposal distribution that we call f_0 (or, as said, π_0), often a candidate for this proposal is the prior [Neal, 1998]. As said, the AIS moves from the initial distribution to the target f_n with intermediate target distributions f_j

$$f_j(x) = f_n(x)^{\beta(j)} f_0(x)^{1-\beta(j)}, \quad j = 1, 2, ..N, \quad 0 \leq \beta(j) \leq 1 \tag{2.59}$$

We start the algorithm with $\beta(j) = 0$ and therefore with f_0 and we arrive in the last step to the target f_n with $\beta(j) = 1$. In the particular case where we use the prior as a starting distribution, expressing the posterior f_n as prior times likelihood as in the standard Bayesian set-up (2.3) we have:

$$f_n(x) = f_0(x) l(x) \tag{2.60}$$

where, in (2.60), f_0 is the prior, and l is the likelihood. In this case, using equation (2.60) in (2.59), we have that

$$f_j(x) = f_0(x) l(x)^{\beta(j)}, \quad j = 1, 2, ..N, \quad 0 \leq \beta(j) \leq 1 \tag{2.61}$$

If we indicate with π_i the normalized probability distribution associated with each f_i, the algorithm, as described in Section 2 of [Neal, 1998], is as follows

$$\begin{aligned} &\text{step 0)} \quad x_0 \sim \pi_0 \\ &\text{step 1)} \quad \text{MCMC step targeting } \pi_1, \text{ resulting in } x_1 \\ &\text{step 2)} \quad \text{MCMC step targeting } \pi_2, \text{ resulting in } x_2 \\ &\qquad \vdots \\ &\text{step n)} \quad \text{MCMC step targeting } \pi_n, \text{ resulting in } x_n \end{aligned} \tag{2.62}$$

Where, in (2.62), the MCMC moves of the intermediate steps are performed, as explained in Section 2.4.1, using Markov kernels $k_i(x_{i-1}, x_i)$ (for example with Metropolis-Hastings 2.4.6). Explaining the algorithm more in detail:

0. in step 0 we draw x_0 from the starting proposal distribution, by assumption easy to draw from (for example the prior) π_0

1. in step 1 we apply a Markov kernel $k(x_0, x_1)$ with stationary distribution π_1 of (2.59), that allows us to move in the state space and, using the results seen in the MCMC Section 2.4, this means that we approximate drawing x_1 from the distribution π_1

2. similarly to what we did in the previous step, we move towards π_2 and we draw x_2 from the distribution π_2

... ...

n. in the last step we approximate drawing a sample x_n from the target density π_n

Repeating the procedure, say, N times and taking each time the last sample of the procedure (the x_n), the algorithm (2.62) produces samples $x_n^{(i)}$, $i = 1, 2, ...N$ that are drawn from the target distribution π_n, with approximations that we will discuss in the remainder of the section. Like Importance Sampling, each particle $x_n^{(i)}$ has a weight that accounts for not directly drawing from the target π_n, we will see that the expression of the weight of each particle is as follows (please note that the super index i indicating the particle has been omitted for brevity in the following formula for the f_j and, in addition, we are using f_j instead of π_j, this is possible because it can be shown [Neal, 1998] that normalising constants cancel out in ratios):

$$w^{(i)} = \frac{f_1(x_0)}{f_0(x_0)} \frac{f_2(x_1)}{f_1(x_1)} ... \frac{f_n(x_{n-1})}{f_{n-1}(x_{n-1})} \tag{2.63}$$

Before explaining how it is obtained mathematically, we can see from its expression that (2.63) is made up by products of importance weights: each factor $\frac{f_{j+1}(x_j)}{f_j(x_j)}$ is, as seen in (2.17), the ratio of the target over the proposal, in fact each f_j, by construction, is the proposal for the f_{j+1}, and these intermediate steps allow, compared to IS, a smoother transition from

the proposal to the target, in fact, by tuning $\beta(j)$ of equation (2.59), it is possible to have proposals that are closer to the targets, allowing for greater efficiency of the intermediate IS steps. From formula (2.63) we can also appreciate a major advantage of weighted particle systems: if we start from $w^{(i)} = \frac{f_1(x_0)}{f_0(x_0)}$ where the target is f_1, a simple reweight operation $w^{(i)}\frac{f_2(x_1)}{f_1(x_1)}$ will shift the target to f_2 [Chopin, 2002]. The validity of (2.63) can be shown [Neal, 1998] using the results we have already obtained in Section 2.3 for IS and 2.4 for MCMC. In fact, if we consider an extended state space $(x_0, ...x_n)$, with a joint distribution:

$$f(x_0, ...x_n) = f_n(x_n)\tilde{k}_{n-1}(x_n, x_{n-1})...\tilde{k}_0(x_1, x_0) \tag{2.64}$$

Where $\tilde{k}_j$ are backward transition kernels. We have that the marginal for x_n of equation (2.64) is the density we are looking to draw from (the target distribution). The $\tilde{k}_j$, as said, are backward transition kernels associated to the MCMC moves of (2.62), and can be calculated by using the detailed balance condition of MCMC of (2.50):

$$\tilde{k}_j(x, x') = \frac{k_j(x', x)p_j(x')}{p_j(x)} \tag{2.65}$$

By rewriting equation (2.64) as

$$\begin{aligned} f(x_0, ...x_n) = f_n(x_n)\frac{f_{n-1}(x_n)}{f_{n-1}(x_n)}\tilde{k}_{n-1}(x_n, x_{n-1})...\frac{f_1(x_0)}{f_1(x_0)}\tilde{k}_0(x_1, x_0) = \\ k_{n-1}(x_{n-1}, x_n)\frac{f_n(x_n)}{f_{n-1}(x_n)}...k_0(x_0, x_1)\frac{f_1(x_0)}{f_0(x_0)}f_1(x_1) \end{aligned} \tag{2.66}$$

If we take a look at the proposal distribution g of the procedure (2.62), starting from the first step $x_0 \sim p_0$ and considering all the subsequent applications of Markov kernels $k(x_j, x_{j+1})$, it has the form:

$$g(x_0, ...x_n) = f_0(x_0)k_0(x_0, x_1)...k_{n-1}(x_{n-1}, x_n) \tag{2.67}$$

Therefore, the AIS can be see as a multi-step importance sampler, and the expression of the weight for the whole importance sampling process, as seen in (2.17), is

$$w^{(i)} = \frac{f(x_0, ...x_n)}{g(x_0, ...x_n)} = \frac{f_1(x_0)}{f_0(x_0)}\frac{f_2(x_1)}{f_1(x_1)}...\frac{f_n(x_{n-1})}{f_{n-1}(x_{n-1})} \tag{2.68}$$

And (2.68) brings the result (2.63), which proves our case. Since, as we saw in the steps of (2.62), at each step $x_j \sim f_j$ and therefore the function f_j becomes the proposal for the next step $f_{j+1}(x_j)$, all the rules that apply to importance sampling choice of proposal hold (please see Section 2.3). The choice of the proposal, as in importance sampling, is critical for the success of the algorithm, and we will see in later sections for example in the application to complex posterior distributions such as for phylogenetic analysis of genetic sequences in Chapter 3 that in non-trivial cases smooth transitions between functions, i.e. small steps in

the exponent β of formula (2.59), are needed to have an acceptable ESS.

2.6 Sequential Monte Carlo (SMC)

Sequential Monte Carlo (SMC) methods are a collection of techniques used to approximate a target distribution [Del Moral and Doucet, 2003]. Like the AIS methods described in Section 2.5, SMC uses IS and a sequence of proposals to approximate intermediate target functions, with the aim to create a set of particles that approximate a distribution of interest, in general not easy to draw from.

SMC vs AIS

A difference of SMC wrt AIS lies in that SMC uses a technique called **resampling** to account for the fact that with the increase of algorithmic time the importance weights have a tendency to degenerate (see for example Section 14.3.3 of [Robert and Casella, 2004]), meaning that it is possible to end up, after a few iterations, with a significant number of particle having small weight. We'll explain in next Section 2.6.1 how resampling helps *rejuvenating* the current set of particles, although at the trade-off of impoverishing the diversity of the set.

A second difference with AIS, is that we will see in the relevant Section 2.6.2, the SMC also uses a Markov kernel, like AIS: in SMC the kernel can be a generic Markov kernel, whereas in AIS, as described in Section 2.5, there is specifically a MCMC kernel.

It can be said that AIS is a subset of SMC, that has no resampling and uses MCMC kernel.

2.6.1 Resampling

The resampling involves sampling with replacement from the current set of particles according to their weights. Mathematically, the resampling can be described as follows:

- We have a set of N particles $\{x^{(i)}\}_{i=1}^N$ with associated normalized weights $\{w^{(i)}\}_{i=1}^N$
- The resampling step will generates a new set of particles $\{x_r^{(i)}\}_{i=1}^N$ such that each $x_r^{(i)}$ is a copy of $x^{(j)}$ with probability proportional to $w^{(j)}$
- After resampling we rename $\{x_r^{(i)}\}_{i=1}^N$ as $\{x^{(i)}\}_{i=1}^N$ (so this become our current set of particles) and all weights are reset: $w^{(i)} = \frac{1}{N},\ i = 1, 2...N$.

Resampling addresses the problem of avoiding degeneracy in the particle population. At the same time, resampling introduces variance, and a technique usually employed is to resample only when the ESS drops below a given threshold (for example when the effective sample size drops below 50% of the number of particles N) [Del Moral and Doucet, 2003].

2.6.2 The SMC algorithm

As said in the introduction, the **Sequential Monte Carlo (SMC)** [Del Moral and Doucet, 2003] algorithm will have many commonalities with the AIS seen in Section 2.5. We will, similarly to Section 2.5, make use of consecutive "neighbouring" distributions, i.e. distributions that are not too different (we will give more precise definition below) one from another so that the proposals and the target distributions, at each step, are sufficiently close. The starting point is, as in the common Monte Carlo methods, that we are willing to draw samples from a target distribution π_n. We proceed through intermediate targets as in equation (2.59), and at each step the previous target becomes the proposal for the next target. We proceed in steps, similar to (2.62), we start by drawing from an initial distribution π_0 easy to draw from, it can for example be the prior (in which case the expressions simplify as in equations (2.60) and (2.61)), and we go on constructing the first steps as done in (2.62). We repeat here for simplicity equation (2.59), using the same symbolism with f_j not-normalised version of π_j

$$f_j(x) = f_n(x)^{\beta(j)} f_0(x)^{1-\beta(j)}, \quad j = 1, 2, ..N, \quad 0 \leq \beta(j) \leq 1 \tag{2.69}$$

As we know, we start the algorithm with $\beta(j) = 0$ and therefore with f_0 and we arrive in the last step to the target f_n with $\beta(j) = 1$. And, in case we use the prior as distribution f_0, we have some significant simplification in the formula which becomes (see also (2.60) and (2.61))

$$f_j(x) = f_0(x) l(x)^{\beta(j)}, \quad j = 1, 2, ..N, \quad 0 \leq \beta(j) \leq 1 \tag{2.70}$$

We present here the SMC version that makes use of resampling of the particles, we will explain further in the section what this implies. The steps of the SMC algorithm follow [Del Moral and Doucet, 2003], we use capital W for **normalised weights**, and w for the **un-normalised weights**:

$$\begin{aligned} &\text{step 0) } x_0 \sim \pi_0 \\ &\text{step 1) use kernel to move from } \pi_0 \text{ to } \pi_1, \text{ resulting in } x_1 \end{aligned} \tag{2.71}$$

0. initialise the iteration variable, say n, to $n = 0$. We draw $x_0 \sim \pi_0$ from the starting proposal distribution π_0 (for example we could choose the prior), by assumption easy to draw from, and set the weights initially to $\frac{1}{N}$

1. we use the drawn particles as an importance sampler proposal (see Section 2.3) for π_1 of equation (2.59), and we have a weight update of $w_0^{(i)} = \frac{1}{N} \frac{f_1(x_0)}{f_0(x_0)}$, this update reflects the weight of particles after the drawing process. We then normalise the weights to $W_0^{(i)}$

2. resampling step (technically resampling does not normally need to happen at every step, see further Section 2.6.3 for more details) we resample the particles according

to their normalised weight, so the bigger the normalised weight the more the particle will have a chance to be chosen in this resampling process: this resampling step allows us to eliminate particles where the proposal weakly represent the posterior, and will replicate particles where there is a strong representation of the posterior, all particles after resampling will again have weights of $W_0^{(i)} = \frac{1}{N}$

3. Update the iteration variable $n = n + 1$, so if before we were at stage 0 now we have $n = 1$. We wish to use the points of the state space obtained from the previous drawing done in step 0, for the next step. To do so, we need to move in the state space, so we apply a Markov kernel $k_1(x_0, x_1)$, that allows us to move from x_0 to x_1 in the state space. The distribution of the drawn points after the application of the Markov kernel is $\tilde{f}_1(x_1) = \int f_0(x_0)k_1(x_0, x_1)dx_0$, we update the weights with $w_1^{(i)} = W_0^{(i)} \frac{f_1(x_1)}{\tilde{f}_1(x_1)}$

we see, from the last step in the above procedure, that the weight update in the last step is

$$w_1^{(i)} = W_0^{(i)} \frac{f_1(x_1)}{\tilde{f}_1(x_1)} = w_0^{(i)} \frac{f_1(x_1)}{\int f_0(x_0)k_1(x_0, x_1)dx_0} \tag{2.72}$$

Since it is not easy, in general, to calculate $\int f_0(x_0)k_1(x_0, x_1)dx_1$, we rewrite the fraction in the RHS of (2.72) so that it can be expressed in non-integral form; we do so by writing the numerator $f_1(x_1)$, for some $L(x_1, x_0)$

$$f_1(x_1) = \int f_1(x_1)L_0(x_1, x_0)dx_0 \tag{2.73}$$

Where, in (2.73), the $L(x_1, x_0)$ is a backward kernel, built so that $f_1(x_1)$ is the x_1-marginal of the joint distribution $f_1(x_1)L(x_1, x_0)$. Equation (2.72) now becomes

$$w_1^{(i)} = W_0^{(i)} \frac{\int f_1(x_1)L_0(x_1, x_0)dx_0}{\int f_0(x_0)k_1(x_0, x_1)dx_0} \tag{2.74}$$

Instead of marginalising, we write the contribution in the RHS of (2.74) using the joint distributions

$$w_1^{(i)} = W_0^{(i)} \frac{f_1(x_1)L_0(x_1, x_0)}{f_0(x_0)k_1(x_0, x_1)} \tag{2.75}$$

Since we can choose $L_0(x_1, x_0)$ of (2.75) at will (as long as equation (2.73) holds), we can choose L s.t.

$$f_1(x_1)L_0(x_1, x_0) = f_1(x_0)k(x_0, x_1) \tag{2.76}$$

We can notice that for example the condition in (2.76) is satisfied if k is a MCMC kernel with invariant distribution π_1, since (2.76) would in that case be the expression of the detailed balance condition of equation (2.50). By substituting (2.76) in equation (2.72) we have

$$w_1^{(i)} = W_0^{(i)} \frac{f_1(x_0)}{f_0(x_0)} \tag{2.77}$$

We then normalise the weights of (2.77) to $W_1^{(i)}$

$$W_1^{(i)} = \frac{w_1^{(i)}}{\sum_i w_1^{(i)}} \tag{2.78}$$

And by repeating the steps 2 and 3 of the algorithm outlined above, we resample, if it is the case, for example we can perform the conditional resampling step, fixing α indicating the fraction of particles N to check degeneracy

$$ESS \leq \alpha \cdot N,\ \alpha \in]0, 1[\tag{2.79}$$

If condition (2.79) is true, we perform the resampling as shown in Section 2.6.1, and we end up with

$$W_1^{(i)} = \frac{1}{N},\ i = 1, 2...N \tag{2.80}$$

Therefore, at this point, $W_1^{(i)}$ will be either equal to (2.78) if no resample has taken place (condition (2.79) false) or to equation (2.80) if resample has indeed taken place (condition (2.79) true)

In the next step we move in the state space from x_1 to x_2 using a Markov kernel $k_2(x_1, x_2)$ having f_2 of (2.59) as a target, and, with a procedure similar to the one that has brought us from equation (2.72) to (2.77), we have that

$$w_2^{(i)} = W_1^{(i)} \frac{f_2(x_1)}{f_1(x_1)} \tag{2.81}$$

And, generalising (2.81), the un-normalised weight update component at each generic step j is

$$w_j^{(i)} = W_{j-1}^{(i)} \frac{f_j(x_{j-1})}{f_{j-1}(x_{j-1})} \tag{2.82}$$

Where, in equation (2.82), $W_{j-1}^{(i)}$ will either be equal to $\frac{1}{N}$, if the weight update comes after a resample, or to the normalised weight of $w_j^{(i)}$ if there has been no resample (please see equations (2.77), (2.78), (2.79), (2.80) and (2.81) where the process is described in detail with indexes $j = 1$ and $j = 2$).

It is to be noted that, if no resampling is employed in the SMC algorithm (equivalently if (2.79) was always false), the full equation of the weight update becomes

$$w_j^{(i)} = \frac{f_j(x_{j-1})}{f_{j-1}(x_{j-1})} ... \frac{f_2(x_1)}{f_1(x_1)} \frac{f_1(x_0)}{f_0(x_0)} w_0^{(i)} \tag{2.83}$$

And we see that equation (2.83) is the same of the AIS case of equation (2.68), which confirms that AIS is a subset of SMC where no resampling is employed.

2.6.3 Conditional Effective Sample Size (CESS)

We have seen, in Section 2.6, that one of the steps of the SMC algorithm consists in resampling the particles according to their weight. One naive way to apply the algorithm would be to perform the resampling at each iteration, but each resampling adds to the variance of weights [Zhou et al., 2013] and ultimately, remembering that the Effective Sample Size (ESS) we mentioned in Section 2.3.2 is a measure of performance of the estimator and depends on the variance of the weights, we understand that an increase in variance of the weights would bring a worse estimator. A better way to perform the resampling step in SMC is to do so adaptively, for example only when the ESS falls below a certain threshold, this would reduce the number of times resampling occurs.

As we have spoken (see for example in the IS Section 2.3), the better the choice of a proposal for a target distribution, the better the performance of a IS estimator. In SMC algorithm we are moving from an initial distribution, say the prior, to the posterior, through a series of intermediate distribution (2.70), where at each step of the tempering process, the current distribution acts de-facto as a IS proposal for the next, and in [Zhou et al., 2013] a quantity is shown, named **Conditional Effective Sample Size (CESS)**, that helps determine in automatic way the next best tempering exponent of (2.70), so that the ESS, calculated on the new tempering, remains high. We report below the formula of CESS (for full technical details see for example algorithm 4 in [Zhou et al., 2013])

$$\text{CESS}(W_{t-1}, w_t) = \left(\sum_{j=1}^{N} W_{t-1}^{(j)} w_t^{(j)}\right)^2 \Big/ \sum_{k=1}^{N} W_{t-1}^{(k)} \left(w_t^{(k)}\right)^2, \tag{2.84}$$

The correct value of CESS helps ensure that the convergence of the SMC algorithm from the initial distribution to the posterior happens keeping enough diversity of particles by controlling the weight update through the annealing exponent. CESS of (2.84) and ESS become equivalent if resampling is done at every iteration. Otherwise ESS will contain the information of the discrepancy of the current iteration approximation versus the target, whereas CESS will contain information on the quality of the current IS step [Amaya et al., 2022]. It is considered a good choice [Amaya et al., 2021] as a threshold of CESS so that $\frac{CESS}{N}$ is close to 1. In practical terms, fixing a predefined CESS value brings to automatic choices of the next tempering exponent. The "perfect" choice of CESS will anyway depend on implementation as choosing a high CESS threshold will normally result in more tempering steps and therefore longer runs of the algorithm, therefore a trade-off will have to be made, depending on the application [Amaya et al., 2021]. In our algorithms we have chosen $\frac{CESS}{N} = 0.9$.

2.7 Pseudo-marginals in MCMC

We have described in Section 2.4 the MCMC and the Metropolis Hastings (MH) algorithm, and in particular, we report here again the formulation of the acceptance ratio of MH

$$\alpha(X_n^*|X_n) = \frac{p(X_n^*)l(X_n^*)g(X_n|X_n^*)}{p(X_n)l(X_n)g(X_n^*|X_n)} \tag{2.85}$$

With g in (2.85) the *proposal distribution*, p as usual the prior, and l the likelihood.

There are cases where the likelihood l of (2.85) is non-tractable or not *convenient* to calculate, and it could be useful to introduce in the algorithm an estimate of the likelihood. Assuming that the state space can be partitioned into two sets of variables and expressing the likelihood as

$$l = l(a, i) \tag{2.86}$$

and the prior as

$$p = p_i(i|a)p_a(a) \tag{2.87}$$

we name $\hat{l}(a)$ the estimate of the a marginal, so that (2.85) becomes

$$\alpha(a_n^*|a_n) = \frac{p_a(a_n^*)\hat{l}(a_n^*)g_a(a_n|a_n^*)}{p_a(a_n)\hat{l}(a_n)g_a(a_n^*|a_n)} \tag{2.88}$$

Of course, having made a change in the original MCMC, we need a justification, mainly to understand if and how the change of using (2.88) instead of (2.85) as MH ratio, affects the convergence of MCMC.

It has been shown in [Beaumont, 2003, Andrieu and Roberts, 2009] that as long as the estimate $\hat{l}$ is unbiased, and so $\mathbb{E}[\hat{l}(a)] = l(a)$, the conditions of convergence for MCMC covered in Section 2.4 are still valid. It is to be noted that the variance of $\hat{l}$ will affect the efficiency of the MCMC algorithm [Beaumont, 2003, Andrieu and Roberts, 2009]. This will be clear by looking at equation (2.88), and considering, for example, that a higher variance of $\hat{l}(a)$ will normally imply a higher variance of the ratio α of (2.88). It is shown that the variance of the pseudo-marginal is greater or equal of the exact marginal [Andrieu and Vihola, 2015].

2.7.1 GIMH pseudomarginal algorithm

Different algorithms are described in [Beaumont, 2003, Andrieu and Roberts, 2009], we give here a short description of the one we will use in the sections on **Active Subspaces** (AS), i.e. Chapters 4 and subsequent, named GIMH (Grouped Independence Metropolis-Hastings). Let's suppose that we have a likelihood dependent on two set of variables a and i as in (2.86), and that we are interested in an approximation of the likelihood where the i part of (2.86) is marginalised out, and so we take N_i samples of i from a proposal distribution q and we get,

via Importance Sampling (see Section 2.3) the estimate

$$\hat{l}(a) = \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{l(a, i_n)}{q(i_n)} \tag{2.89}$$

Equation 2.89 approximates, in the Monte Carlo sense, the marginal $l(a) = \int l(a, i)di$ and this is the reason behind the name *pseudo-marginal* for the estimate of the likelihood. We know that Importance Sampling produces unbiased estimates (see Section 2.3), and therefore the estimate of likelihood $\hat{l}(a)$ of (2.89) can be used in equation (2.88), keeping the MCMC theoretical conditions for convergence intact.

2.8 Particle MCMC

With the same spirit of Section 2.7, where we have introduced an unbiased estimate of the likelihood via Importance Sampling, to be used in the Metropolis Hastings ratio of MCMC instead of the original likelihood, with **particle MCMC** we use an unbiased estimate of the likelihood generated using SMC. The idea is described in [Andrieu et al., 2010], and the demonstration uses additional auxiliary variables, and then shows that the pMCMC algorithm is actually an MCMC on an extended space, and that the intended target is obtained as a marginal of the extended one. Compared to the pseudo-marginal of Section 2.7 which uses Importance Sampling (IS), we expect particle MCMC to perform better in cases where SMC performs better than IS: for example in likelihood estimates in high-dimension, or also in cases of isolated modes: SMC will, in general, deal better than IS in those cases when there is a "distance" in some sense between the proposal and the posterior.
We will give a short description of the algorithms of [Andrieu et al., 2010] that we have used.

2.8.1 Particle Marginal Metropolis-Hastings (PMMH)

We assume again a partition of the state space like in equations (2.86) and (2.87). The central idea of the algorithm **Particle Marginal Metropolis-Hastings (PMMH)** described in [Andrieu et al., 2010] is to obtain an estimate of the likelihood by running an SMC Sampler [Del Moral and Doucet, 2003] at every step of an outer MCMC algorithm. We report below the main steps, as usual l is the likelihood, p is the prior, q is the MCMC proposal:

Alg. 1 Particle Marginal Metropolis-Hastings (PMMH)[Andrieu et al., 2010]

1: Initialize $a^{(0)}$ and estimate $\hat{l}(a^{(0)})$ using SMC
2: **for** $k = 1$ to K **do** ▷ start MCMC iteration
3: Propose a^* from $q(a^*|a^{(k-1)})$
4: Run SMC with a^* to estimate $\hat{l}(a^*)$
5: Calculate acceptance probability $\alpha(a^{(k-1)}, a^*) = \min\left(1, \frac{p_a(a^*)\hat{l}(a^*)q(a^{(k-1)}|a^*)}{p_a(a^{(k-1)})\hat{l}(a^{(k-1)})q(a^*|a^{(k-1)})}\right)$
6: With probability α, accept a^* and set $a^{(k)} = a^*$
7: Otherwise, retain $a^{(k-1)}$
8: **end for** ▷ end MCMC iteration

2.8.2 Metropolis within Particle Gibbs (MwPG)

Metropolis within Particle Gibbs (MwPG) is a version of particle MCMC that combines Gibbs sampling together with an SMC Sampler update. Considering a partition of the state space like in equations (2.86) and (2.87), unlike PMMH of Section 2.8.1 where an estimate of the likelihood was calculated with a marginalization through an SMC sampler, in MwPG we are interested in a conditional update. Gibbs sampling can be particularly useful, as we have seen in Section 2.4.7, for example when the conditional distribution of the posterior is easy to draw from. We will be using a variant of MwPG in Active Subspaces (sections 4 and subsequent).
We will give here a description of the version of PG algorithm similar to the one we have used in the Active Subspaces.

Survivor particle path in MwPG

The basic idea of PG is to use an outer MCMC performing Gibbs sampling at each iteration t, by firstly updating the $i|a$, and then obtaining an update of the *survivor* path $a_{1:T}^{(t)}$. We'll dedicate some attention to the conditioning on the path. That some conditioning appears in the SMC part of MwPG was to be expected, as the SMC estimate is part of an outer Gibbs MCMC, and Gibbs uses conditional updates: in the case of MwPG the correct theoretical framework is ensured by **keeping the SMC algorithm conditioned to a particular path** [Andrieu et al., 2010] (in Algorithm 2 it is the path of particle N of the SMC, sampled on line 16), this **special particle path** is a guaranteed **survivor** throughout the SMC algorithm, in fact the path $a_{1:T}^{(t-1)}$ is guaranteed not to be eliminated by resampling until the end of the execution of the SMC algorithm of time t (from lines 6 to 15 of Algorithm 2): the new reference path updated at time t, $a_{1:T}^{(t)}$, is only sampled at the end of the SMC algorithm execution (line 16 of Algorithm 2).

MwPG algorithm

Alg. 2 Particle Gibbs Sampler

Initialize $i^{(0)}$ and $a_{1:T}^{(0)}$ ($a_{1:T}^{(0)}$ will be path of particle N in the SMC algorithm)
for $k = 1$ to K **do** ▷ *Start MCMC iteration*
  Sample i^* from $q(\cdot|i^{(k-1)}, a_{1:T}^{(k-1)})$
  Perform an update of $i|a$
  Sample $N-1$ points $a_0^{(i)}$ and set $w_0^{(i)} = \frac{1}{N}$ for $i = 1, ...N-1$ ▷ *SMC init*
  for $t = 1$ to T **do** ▷ *Start SMC tempering loop*
    for $i = 1$ to $N-1$ **do**
      Update weight $w_t^{(i)} = w_{t-1}^{(i)} \frac{l_{1:t}(i^{(k)}, a_{t-1}^{(i)})}{l_{1:t-1}(i^{(k)}, a_{t-1}^{(i)})}$.
    end for
    Normalize weights $w_t^{(i)}$ obtaining $W_t^{(i)} = \frac{w_t^{(i)}}{\sum_i w_t^{(i)}}$
    Resample particles with replacement according to weights $W_t^{(i)}$ if degeneracy occur
    for $i = 1$ to $N-1$ **do**
      Perform MH update on $a_{t-1}^{(i)}$ to obtain $a_t^{(i)}$
    end for
  end for ▷ *End SMC tempering loop*
  Select path of particle N to $a_{1:T}^{(k)}$ according to $W_t^{(i)}$
end for ▷ *End MCMC iteration*

Some auxiliary notes on Algorithm 2: we see a reference to T, which is the final time-step of tempering in the SMC algorithm (see Section 2.6), for simplicity we will assume here that the tempering path is fixed for all the iterations, and assume N fixed as number of particles in the SMC algorithm. Comments in the algorithm are in *italics*.

2.9 SMC^2

The **SMC^2** algorithm [Chopin et al., 2012] uses a particle MCMC inside an outer SMC algorithm. In a way we can see the SMC^2 as an advancement of PMMH, where instead of an outer MCMC we have a SMC. The proofs of convergence of the algorithms can be found in [Chopin et al., 2012], and use a common technique of augmenting the number of variables and then showing that the algorithm performs an SMC on the augmented space targeting a distribution that has the original intended target as marginal. Below a summary of the algorithm in its basic form

Alg. 3 SMC^2 Algorithm

Initialize θ^m from prior $p(\theta)$ for $m = 1, \ldots, N_\theta$
Set initial weights $\omega_0^m = 1/N_\theta$ for all m
Set ESS threshold α
Set number of internal SMC particles N_x
for $t = 1$ to T **do** ▷ *Start outer SMC tempering loop*
    for all θ^m **do**
        Run internal SMC Sampler with N_x particles to estimate incremental likelihood $\hat{l}(y_t|y_{1:t-1}, \theta^m)$
    end for
    Update weights $\omega_t^m = \omega_{t-1}^m \cdot \hat{l}(y_t|y_{1:t-1}, \theta^m)$
    Normalize weights $W_t^m = \frac{\omega_t^m}{\sum_{n=1}^{N_\theta} \omega_t^n}$
    if ESS $< \alpha$ **then** ▷ *Resampling criterion*
        Resample θ^m particles based on weights W_t^m
        Set $W_t^m = 1/N_\theta$
    end if
    for all θ^m **do** ▷ *MCMC rejuvenation step*
        Perform MCMC step on θ^m to obtain θ^{m*}
        Set $\theta^m = \theta^{m*}$
    end for
end for ▷ *End outer SMC tempering loop*

While in general the number of particles N_x of the internal SMC of Algorithm 3 is fixed throughout all the internal SMC steps, it is discussed in [Chopin et al., 2012] the case for adaptation of the number of particles N_x within the steps of the algorithm. Such possibility is interesting both on the methodological side and on the practical side, we will be discussing such extension of the algorithm more in depth when discussing Active Subspaces, in Chapter 4 and subsequent.

Chapter 3

Models of genetic evolution

3.1 Introduction

The recent outbreak of COVID-19 [Wu et al., 2020] has underscored the importance of analysis in understanding the spread and evolution of infectious diseases, in particular through the study of mutations and relationships among the various strains of viruses spreading in different time periods and different geographical locations. Phylogenetic analysis, at its core, involves the study of the evolutionary relationships among biological species, typically through the analysis of genetic sequences, for example DNA or RNA, which are coded via sequences of nucleotides. By examining these genetic sequences, scientists can infer the genealogy and evolutionary history of organisms, and how the different species have diverged and evolved over time, and these relationships are often expressed via phylogenetic trees. As an introductory visually appealing example, borrowed from [Hou et al., 2022], we can see below in Figure 3.1, an instance of a phylogenetic tree:

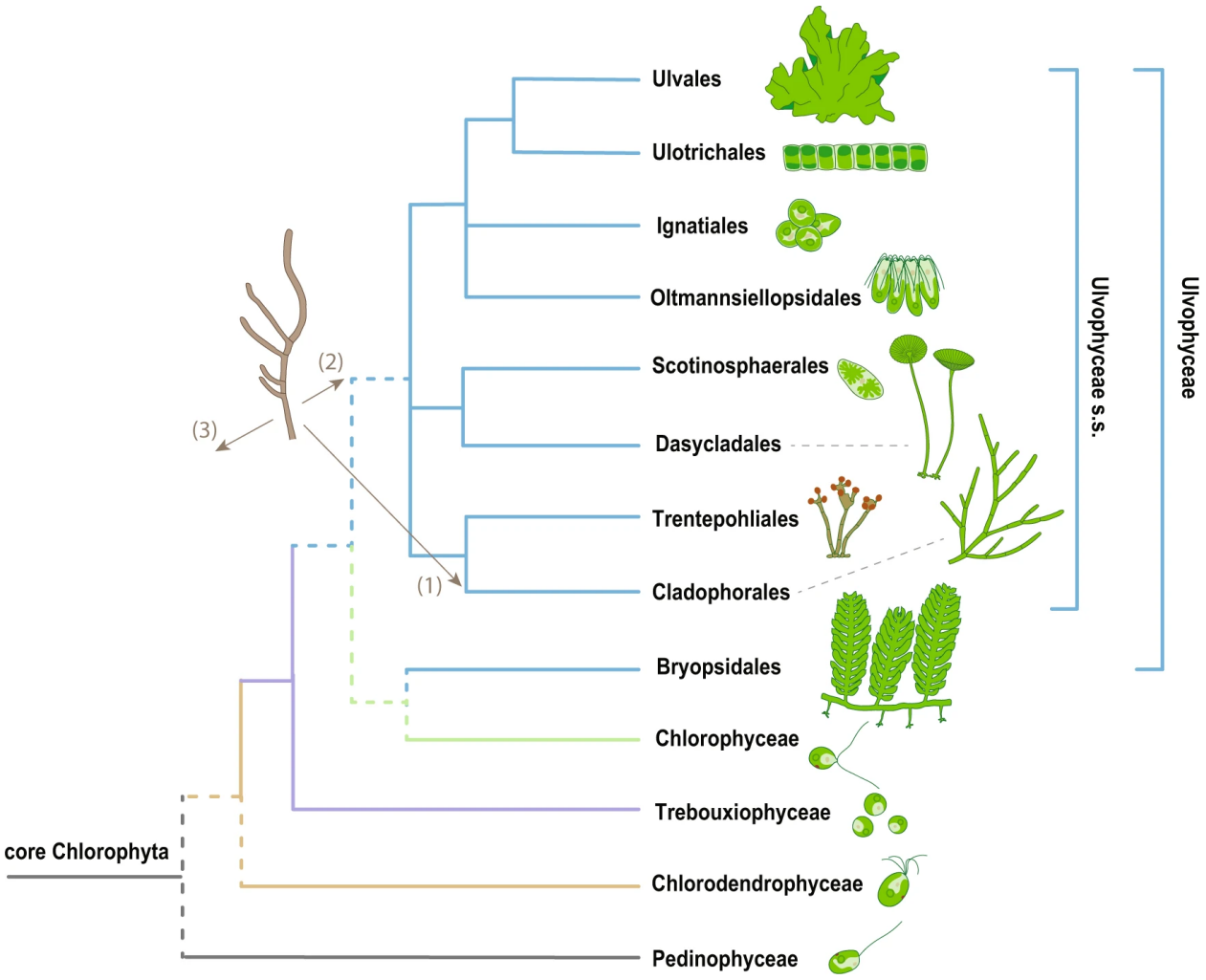

Figure 3.1: *Visually appealing example of phylogenetic tree, borrowed from [Hou et al., 2022]. The different species are the plants pictured on the RHS of the figure, the lines that combine the different species until reaching a common ancestor (core Chiorophyta on the bottom LHS), represent genetic lineage relationships. The points where the lines combine, represent the moments in past time when different species started to diverge from the same lineage.*

In Figure 3.1,

- The plants pictured on the RHS of the figure are different species
- The lines on the that combine the different species until reaching a common ancestor (*core Chiorophyta* on the bottom LHS), represent genetic lineage relationships
- The points where the lines combine, represent the moments in past time when different species started to diverge from the same lineage

Commonly, phylogenetic trees like the one pictured in Figure 3.1 are translated into a format that algorithms can act upon. Several software can be used for the representation, for example in Figure 3.2 we see the screenshot of a software tool named FigTree [Rambaut, 2023], which

is used for phylogenetic trees analysis and will be introduced later in the chapter. Although (possibly) less visually appealing than Figure 3.1, Figure 3.2 shows the reconstruction of a phylogenetic tree with 7 sequences (the 7 tips of the tree named t_0 to t_7 which can be found on the RHS of the picture), the lines representing the lineages (in Figure 3.2 we can see the estimated times reported on the lineages lines), and the points when two species merge in a single branch of the tree, these are called *coalescence times*

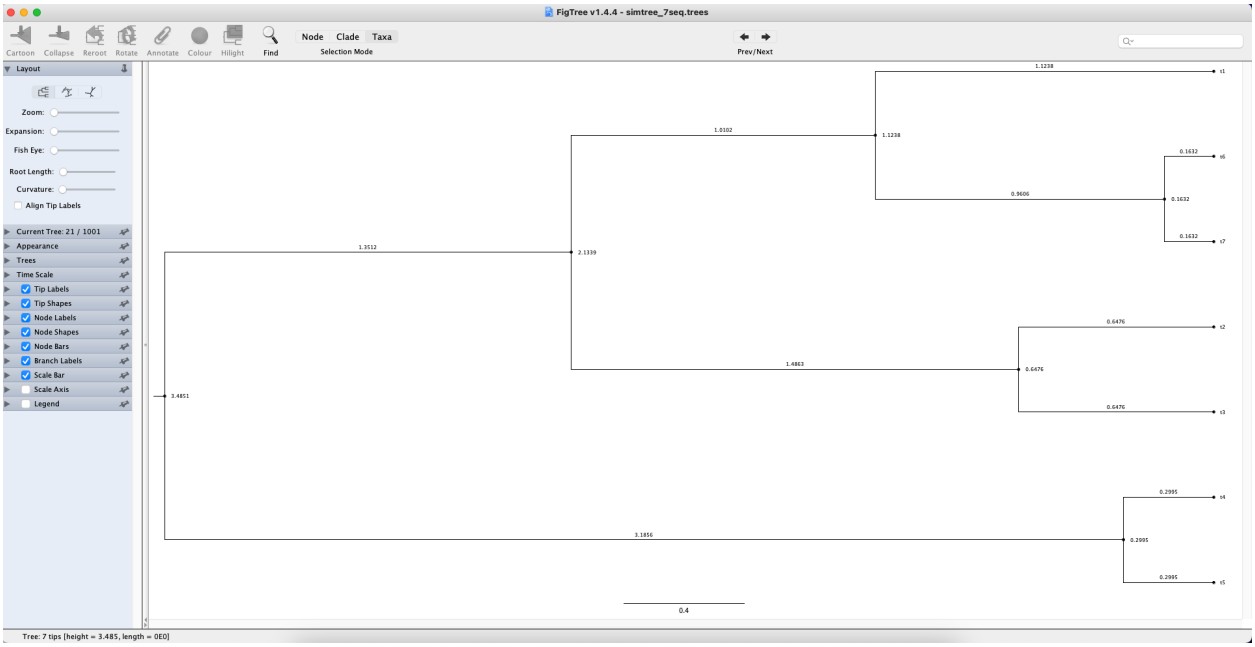

Figure 3.2: *The **Figtree** software [Rambaut, 2023], useful to visualize and get statistical information on phylogenetic trees. We can see the tips on the right hand side numbered t_0 to t_7 which represent the genetic sequences that are the starting point of the analysis (the whole tree is inferred starting from these sequences). The points where two branches merge into one are called coalescence times.*

The task of Bayesian inference in phylogenetics is, at its core, to find the posterior distribution over genealogies g, given genetic sequences data y. This chapter will give an introduction of phylogenetics, of the relevant studies on Bayesian statistics applied to phylogenetics and of some of the most commonly used software tools. A few traditional studies are considered important in the field. The Wright-Fisher model has been produced in a famous study [Wright, 1931, Fisher, 1930], it provides a basic but functional framework for describing genetic evolution, we describe the model in Section 3.2.3. The coalescent is a prior distribution on trees [Kingman, 1982] that we will use in the following sections, we describe the coalescent theory in Section 3.3. The likelihood used in our analysis is the Felsenstein's likelihood [Felsenstein, 1981], a formula which evaluates trees given the input genetic sequences, we describe the Felsenstein's likelihood in Section 3.6.

Existing Methods and Software

Probably one of the most famous statistical software tool available for Bayesian phylogenetic analysis is named Bayesian Evolutionary Analysis by Sampling Trees, in short **BEAST** [Drummond and Rambaut, 2007, Drummond et al., 2012], and, born successively as a development branch, **BEAST2** [Bouckaert et al., 2019]. Both software have been developed by researchers, and offer tools for the solution of many real-world research problems. We will in Section 3.8 give a bit of introduction to BEAST and BEAST2. For our research we have used BEAST2 only, and therefore the main focus will be about this platform. BEAST2 uses MCMC as its main MC sampling method [Bouckaert et al., 2019]. We describe in some detail BEAST2 software environment in Section 3.9.

Contribution: implementation of Annealed Adaptive SMC in BEAST2

MCMC, which is the native Monte Carlo method in BEAST2, is commonly used for the exploration of the parameter space in phylogenetics [Bouckaert et al., 2019]. We have implemented annealed importance sampling (AIS) and sequential Monte Carlo (SMC) in the BEAST2 software platform. The implementation in a platform that is widely used for phylogenetics allows for a direct comparison of algorithm performance, both from a statistical and from a computational point of view. Although we have developed our work independently, our implementation in BEAST2 can be said to integrate all the various algorithms of [Wang et al., 2019] in BEAST2 environment, in particular the results we report here are for the most advanced of the algorithm mentioned in [Wang et al., 2019], i.e. the *Annealed Adaptive SMC*. The results obtained from our implementation, discussed in detail in Section 3.12, demonstrate that our Annealed Adaptive SMC algorithm achieves performance comparable to the native MCMC method of BEAST2 in terms of both statistical accuracy and computational efficiency. To our knowledge, this is the first implementation in BEAST2 of a SMC algorithm. One of the advantages of the SMC method is that we have been able to achieve similar performances to the MCMC by using far fewer output samples, if we pick the 10 taxa example we discuss in Section 3.14, our SMC implementation has used 1000 particles, resulting in as many output samples, that can be used, for example, to compute expectations. In the native MCMC of BEAST2, in the set up to achieve similar number of likelihood evaluations, 350000 iterations have been used, resulting in as many output samples, with significant additional computational time if we want to compute expectations, compared to the SMC case, especially for information-dense objects as trees. We assume that, with growing number of leaves, the difference will probably be even more remarkable.

It has required a non trivial amount of work to integrate the algorithms in a complex platform like BEAST2, we see the equivalence in statistical results with the native BEAST2 MCMC as a first step, and future tuning of the algorithm and of the integration within BEAST2 can improve the results.

3.2 Modelling genetic evolution

The main goal of population genetics and of phylogenetics (we explain the differences between the two in Section 3.2.2) *"is to infer the past history of populations and describe the evolutionary forces that have shaped their genetic variations"* [Tataru et al., 2017]. The current section will give a short introduction.

3.2.1 DNA

DNA is packaged into **chromosomes**. Taking as example the human species, there are 46 chromosomes situated in the cells nuclei, 23 pairs, one of each chromosome is inherited by each parent. **Genes** are sections of the chromosome situated in so-called **loci**, each gene is responsible for a trait, for example hair colour. Different variations of the same gene are called **alleles**. The expression of different alleles of the same gene will result in different characteristics, for example a different colour of hair, say brown or blonde [Alberts et al., 2002].

3.2.2 Population genetics vs phylogenetics

As introduced at the beginning of Section 3.2, there can be many causes of genetic changes, just to shortlist some [Tataru et al., 2017]:

- **random drift**: changes due to chance;
- **mutations**: errors in the replication of DNA;
- **selection**: mutations that are more advantageous and become more likely to be passed to the following generations.

Population genetics has usually the time-scale of a single generation [Tataru et al., 2017], and the goal is to understand evolution of allele frequencies within the same generation, which is done for example using the **Wright-Fisher model**, introduced in Section 3.2.3.
Phylogenetics time-scale is usually longer and cross-generations [Tataru et al., 2017], and the aim is usually to infer the coalescent times of different species [Tataru et al., 2017]. Such task is accomplished for example by the **coalescent model**, introduced in Section 3.3.
Of course the above subdivision between population genetics and phylogenetics is to be taken as a reference and differences between the two can be blurred [Tataru et al., 2017]. In the rest of the work we will most of the times, for ease of notation, refer to phylogenetics, meaning either analysis related to population genetics or phylogenetics, therefore both for single and multi-generational data or a combination of the two, and the relative time-scale will be clear from the context.

3.2.3 The Wright-Fisher model

The Wright-Fisher model [Wright, 1931, Fisher, 1930] describes changes in allele frequencies. It assumes random sampling (i.e. if we consider two successive generations we can randomly assign parents from the generation before) and a constant population size [Tataru et al., 2017]. Assuming population size N (as we said it is a constant of the model), we describe below a diploid (i.e. with two sets of chromosomes, one coming from each parent) model of individuals and we see the basic laws behind changes in allele frequencies. Let's assume for the example that we have two alleles AA and AB, only subject to *random drift* (as said the model is reasonable for short timescales). We provide below two slightly different derivations of the Wright-Fisher, one with probability given by allele frequency, the other with probability given by the population size.

Mathematical derivation with probability given by allele frequency

We use the description and the same notation of [Tataru et al., 2017]. We want to express a time relationship for the frequency of the alleles [Tataru et al., 2017], so let r be the generation indicator and let $z(r)$ be the number of individuals that have, say, the allele AA in generation r. The proportion within the population N is therefore

$$x(r) = \frac{z(r)}{N} \tag{3.1}$$

Keeping the population N constant, we use a binomial for the conditional distribution of z in the following generation [Tataru et al., 2017]

$$z(r+1)|z(r) \sim Bin(N, x(r)) \tag{3.2}$$

Expressing the probability we have

$$P([z(r+1)|z(r)] = k) = \binom{N}{k} \left(\frac{z(r)}{N}\right)^k \left(1 - \frac{z(r)}{N}\right)^{N-k} \tag{3.3}$$

And, plugging (3.1) into (3.2), we have that the mean and variance of the binomial (3.2) are as follows [Tataru et al., 2017]:

$$E[x(r+1)|x(r)] = x(r) \tag{3.4}$$

$$Var[x(r+1)|x(r)] = \frac{1}{N} x(r)(1 - x(r)) \tag{3.5}$$

By iterating the two expressions we have that [Tataru et al., 2017]:

$$E[x(r+1)|x(0)] = x(0) \tag{3.6}$$

$$Var[x(r+1)|x(0)] = x(0)(1-x(0))\left(1-\left(1-\frac{1}{N}\right)^r\right) \tag{3.7}$$

And, for big N we can use the approximation [Tataru et al., 2017]

$$Var[x(r+1)|x(0)] \approx x(0)(1-x(0))\left(1-\left(1-e^{-t}\right)\right) \tag{3.8}$$

where t in (3.8) is

$$t(r,N) = \frac{r}{N} \tag{3.9}$$

We can see from (3.9) that in the Wright-Fisher we can estimate the population size N only if the generation r is known, otherwise we can only have an estimate of the combined $t(r,N)$ of (3.9), that we can name *generation time* [Tataru et al., 2017, Drummond et al., 2002]. From equations (3.4) and (3.5) we can see that there are two equilibria:

1. $x(r) = 0$, when in a generation we reach zero number of individuals with the specific allele, this causes the expected value for the following generations to be zero as well, with zero variance, so the particular allele is extinct;

2. $x(r) = N$, i.e. all the individuals have the allele, and in the future generations all individuals will have the allele as well, with zero variance, we have full spread of the allele.

The above conclusion brings us to say that, under the conditions of the model, if a certain alelle has small frequency, it is more likely to disappear after a few generations, as it is more likely to reach the equilibrium $x(r+n) = 0$ for some n, whereas if it has a frequency close to 1 (nearly all the population has the allele), it is more likely to reach the equilibrium point $x(r+n) = N$ for some n, i.e. all the population will end up having the specific allele [Tataru et al., 2017].

Mathematical derivation with probability given by the population size

We report here also a slightly different mathematical derivation of the Wright-Fisher model from [Hein et al., 2004], as we will use some of the results in the following sections on coalescent theory. In the same setting as the previous section, we consider the frequency of the allele in the generation $r+1$ distributed binomially

$$x(r+1) \sim Bin(N,p) \tag{3.10}$$

In the case of (3.10), differently from equation (3.2), the probability parameter of the binomial is inversely proportional to the population size

$$p = \frac{1}{N} \tag{3.11}$$

Expressing the binomial probability in full we have

$$P(x(r+1)=k)=\binom{N}{k}\left(\frac{1}{N}\right)^{k}\left(1-\frac{1}{N}\right)^{N-k} \tag{3.12}$$

For N large $P(x(r+1)=k)$ of equation (3.12) becomes Poisson distributed [Hein et al., 2004]

$$P\left(x(r+1)\right)\approx\frac{e^{-1}}{k!} \tag{3.13}$$

Using (3.13), we see that the probability that a particular allele has no expression in the next generation is

$$P(x(r+1)=0)\approx e^{-1}\approx 0.37 \tag{3.14}$$

And therefore, from the result of (3.14), the probability of at least one expression of the allele is approximately

$$P(x(r+1)\neq 0)\approx 1-e^{-1}\approx 0.63 \tag{3.15}$$

Extending the result of (3.15) at t generations in the future, considering the independence of the events, as per hypotheses of the Wright-Fisher model expressed at the beginning of Section 3.2.3, we see that

$$P(x(r+t)\neq 0)\approx (1-e^{-1})^{t}\approx (0.63)^{t} \tag{3.16}$$

And so, under the hypotheses of the model, after a few generations only a few lineages contribute to the current population [Hein et al., 2004], in fact, taking as an example a population size of $N=10000$, after $t=15$ generations, a number of approximately 10 lineages will have contributed to the current allele population

$$10000(0.63)^{15}\approx 10 \tag{3.17}$$

The remaining $10000-10=9990$ lineages that, in the example given, were present 15 generations ago, will not have survived [Hein et al., 2004].

3.3 Standard coalescent

From the simple version of the Wright-Fisher model described in Section 3.2.3, where we introduced probabilities concerning different versions of a gene, we move on to the problem of estimating coalescent times of genetic changes and derive the *coalescent model* [Kingman, 1982, Drummond et al., 2002]. A very clear explanation of the coalescent is in [Drummond et al., 2002] and [Drummond and Bouckaert, 2015], from where we took the derivation and symbols.

3.3.1 Coalescent of a sample of two different genes

In the same setting of the Wright-Fisher model of Section 3.2.3, i.e. a constant population size of N, discrete generation and full mixing of individuals, we want to infer the distribution of the coalescent times of two genes in a population of size N. Assuming that both genes are sampled at the same time $t = 0$ we will be going backwards in the estimation of the time when they had a common ancestor. Therefore, considering discrete generations we want to calculate the probability that two genes had the Most Recent Common Ancestor (MRCA) j generations back, we see in Figure 3.3 a graphical representation of a sample case of population size $N = 4$ and of a coalescent event happening j generations in the past

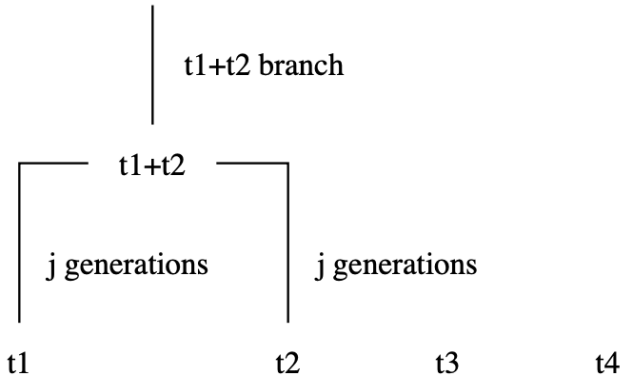

Figure 3.3: *Example of coalescent event of two genes, named in the figure t_1 and t_2, out of a population of $N = 4$. The coalescent event, as described in the paragraph, happens j generations in the past. The diagram has been created using the package graphviz.*

Therefore we have to express the probability that the two genes don't have a common ancestor in the previous $j - 1$ generations and they do have a common ancestor in the jth generation: since sampling in different generations is independent of each other, and given the probability $\frac{1}{N}$ that they have a common ancestor in any generation (and therefore $1 - \frac{1}{N}$ that they don't), the time of MRCA is distributed as follows [Hein et al., 2004] :

$$Pr\left(T_{MRCA} = j\right) = \frac{1}{N}\left(1 - \frac{1}{N}\right)^{j-1} \tag{3.18}$$

From equation (3.18) we can see that the time to the common ancestor is geometrically distributed with parameter $\frac{1}{N}$. Equation (3.18) is derived, under the same assumptions of the Wright-Fisher model, from equation (3.10): the geometric distribution of (3.18) comes from the binomial (3.10) where we focus on the number of "failures" (i.e. the number of generations where there is no coalescent event) until the first "success" (the coalescent event of two samples). Using the properties of the geometric distribution in (3.18), we can calculate

the expected T_{MRCA}

$$E\left(T_{MRCA}\right) = \frac{1}{\frac{1}{N}} = N \tag{3.19}$$

We see, in equation (3.19), that a bigger population size N means an equally bigger average time to the common ancestor.

3.3.2 Coalescent of a sample of k different genes

We can further expand the expression for two different genes found in equation (3.18), to the general case of k different genes in a population of N. Under the same conditions explained in the Section 3.3.1, if, in starting generation at $t = 0$, out of a population of N, $k \leq N$ individuals have different genes, the probability that in the previous generation they don't have a common ancestor is [Drummond and Bouckaert, 2015]:

$$\begin{aligned} Pr\left(T_{MRCA} \neq 1\right) = \left(\frac{N-1}{N}\right)\left(\frac{N-2}{N}\right)...\left(\frac{N-k+1}{N}\right) = \\ 1 - \frac{k\left(k-1\right)}{2N} + O\left(\frac{1}{N^2}\right) \end{aligned} \tag{3.20}$$

Please note in equation (3.20), as explained before, that times, in the coalescent model are counted backwards, therefore $T_{MRCA} = 1$ is one generation back. Since we assume that the population N is significantly larger than k, the term $O(\frac{1}{N^2})$ in (3.20) can be neglected and therefore the probability of no coalescent events in the previous generation becomes:

$$Pr\left(T_{MRCA} \neq 1\right) \approx 1 - \frac{k\left(k-1\right)}{2N} \tag{3.21}$$

And therefore

$$Pr\left(T_{MRCA} = 1\right) = 1 - Pr\left(T_{MRCA} \neq 1\right) \approx \frac{k\left(k-1\right)}{2N} = \binom{k}{2}\frac{1}{N} \tag{3.22}$$

And, similar to what we derived in equation (3.18), the probability that two genes out of k different genes in a population of N have a common ancestor j generations back, is given by the probability of no common ancestor for $j-1$ generations, i.e. equation (3.21) applied $j-1$ times, and then a common ancestor, i.e. equation (3.22), applied once:

$$Pr\left(T_{MRCA} = j\right) = \left(\frac{k\left(k-1\right)}{2N}\right)\left(1 - \frac{k\left(k-1\right)}{2N}\right)^{j-1} \tag{3.23}$$

3.3.3 The continuous time coalescent

Symbols and derivation have been taken from [Drummond and Bouckaert, 2015]. In the Wright-Fisher model introduced in one of its simplest versions in Section 3.2.3 and expanded with equation (3.23), the time is assumed discrete and indicates the number of generations.

Firstly we can notice that, by scaling the time by a factor of N [Hein et al., 2004] we have that:

$$t_j = \frac{j}{N} \tag{3.24}$$

Using the time as in (3.24) allows us to express the results independently from the population size N. It can be shown that [Hein et al., 2004], using the time-scale transformation of equation (3.24) and the assumption that the population size is much bigger than the number of samples $N \gg k$, the geometric distribution converges to an exponential distribution, and in fact it is shown in [Kingman, 1982] that as N grows the coalescent process converges to a continuous-time process [Drummond and Bouckaert, 2015]

$$Pr\left(T_{MRCA} = j\right) = \left(\frac{k\left(k-1\right)}{2N}\right)\left(1 - \frac{k\left(k-1\right)}{2N}\right)^{j-1} \xrightarrow{N \gg k} \lambda \exp^{-\lambda j} \tag{3.25}$$

with the rate of the exponential distribution given by

$$\lambda = \frac{k\left(k-1\right)}{2N} \tag{3.26}$$

Equation (3.25) gives us the distribution of coalescent times in continuous time. Rewriting (3.25), and using τ to express the time instead of the discrete j, we have that the density expressing the probability that two lineages out of k coalesce at time τ is given by:

$$Pr\left(T_{MRCA} = \tau\right) = exp^{-\frac{k(k-1)\tau}{2N}} = exp^{-\binom{k}{2}\frac{\tau}{N}} \tag{3.27}$$

The expected value of (3.27), and therefore the average first coalescent time when we have k different lineages and a population size of N, using the properties of the exponential distribution, is $\frac{1}{\lambda}$, i.e., using equation (3.26).

$$E\left(\tau|k,N\right) = \frac{2N}{k(k-1)} \tag{3.28}$$

So, using equation (3.27), if we want to express the density for all the times so that all the k different samples arrive to a unique common ancestor, considering that, as per hypotheses, the generations are not overlapping, there is complete mixing of the population, and that the population size is constant, due to independence, we multiply the probabilities of the occurrences [Heled and Drummond, 2008]:

$$f\left(\tau_0,\ldots\ \tau_k|N\right) \propto \prod_{i=1}^{k-1} \frac{1}{N} \exp^{-\frac{k_i(k_i-1)\tau_i}{2N}} \tag{3.29}$$

where, in equation (3.29), the k_i express the number of different samples at each coalescent event, so for example if we start the analysis with $k = 5$ samples (lineages), at the first coalescent event we will have $k_1 = 5$, then, since two of the lineages will have merged, in

the second coalescent event we have $k_2 = 5 - 2 = 3$ lineages, etc, for a number of coalescent events equal to $k - 1$ to arrive to the common ancestor of all.

3.4 Substitution model

Substitution models describe genetic variations. The simplest measure of distance between two sequences with same length l, and which differ in h sites, is the so called *Hamming distance* [Drummond and Bouckaert, 2015]:

$$d_H = \frac{h}{l} \tag{3.30}$$

The *alphabet* of possible nucleotides in a genetic sequence is relatively small with just four nucleotides $\{A, G, C, T\}$ (in RNA we have U in place of T). Bases may undergo multiple recombinations, therefore the Hamming distance usually underestimates the actual genetic distance, and more complex mathematical models have been developed to correct the formula. The Jukes-Cantor model provides the following correction to the Hamming distance

$$d_{JC} = -\frac{3}{4} ln \left(1 - \frac{4}{3} d_H\right) \tag{3.31}$$

where d_H is the Hamming distance of (3.30). Equation (3.30) assumes that the nucleotides have equally likely transitions among them, and that their equilibrium frequencies are all the same. The transitions among the four nucleotides are described as continuous time Markov process, using a transition matrix [Drummond and Bouckaert, 2015]

$$Q = \begin{bmatrix} q_{AA} & q_{AC} & q_{AG} & q_{AT} \\ q_{CA} & q_{CC} & q_{CG} & q_{CT} \\ q_{GA} & q_{GC} & q_{GG} & q_{GT} \\ q_{TA} & q_{TC} & q_{TG} & q_{TT} \end{bmatrix} \tag{3.32}$$

While all the non-diagonal elements are non negative and represent the rates of transition between two nucleotides, the element in the diagonal are negative and represent the total flow out of each state towards all other nucleotides. If we use variables i and j to indicate two generic nucleotides, the diagonal element of (3.32) are

$$q_{ii} = -\sum_{j \neq i} q_{ij} \tag{3.33}$$

therefore, the total rate of change per site per unit time is [Drummond and Bouckaert, 2015]

$$\mu = -\sum_i \pi_i q_{ii} \tag{3.34}$$

It is possible to give transitional probabilities through the following [Drummond and Bouckaert, 2015]

$$P(t) = \exp(Qt) \tag{3.35}$$

where Q is the matrix (3.32). We will see in Section 3.4.1 the expression of the substitution formulae for the Jukes Cantor.

3.4.1 Jukes Cantor

In Jukes Cantor [Jukes and Cantor, 1969] all nucleotide transitions have equal probabilities. The transition matrix (3.32) has the form below of equation (3.36) (it's a normalised version where the expected mutation rate $\mu = -\sum_i \pi_i \hat{q_{ii}} = 1$)

$$Q = \begin{bmatrix} -1 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & -1 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & -1 & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & -1 \end{bmatrix} \tag{3.36}$$

By using the Q matrix of (3.36) and the $P(t) = \exp(Qt)$ relationship, we end up with the following transition probability matrix [Drummond and Bouckaert, 2015]:

$$p_{ii}(d_{JC}) = \frac{1}{4} + \frac{3}{4}\exp(-\frac{4}{3}d_{JC}) \tag{3.37}$$

$$p_{ij}(d_{JC}) = \frac{1}{4} - \frac{1}{4}\exp(-\frac{4}{3}d_{JC}) \tag{3.38}$$

Where d_{JC} is the genetic distance of equation (3.31). All the entries of (3.37) tend to $\frac{1}{4}$ as d_{JC} grows.

3.5 Estimation of parameters in the coalescent

We will in this section derive a probability distribution of coalescent times in a phylogenetic tree, we take symbols and derivation from [Drummond et al., 2002] and [Drummond and Bouckaert, 2015]. In a tree with N leaves there will be $2N - 1$ coalescent events to arrive to a common ancestor. Considering formula (3.27) which expressed the probability of one pair coalescing from k lineages, and taking into account what we already did in equation (3.29), we can express in the more general case of having coalescent times t_i with $i \in Y$, the probability distribution of the independent coalescent events, as explained in detail for example in [Drummond et al., 2002] from which we borrow the notation

$$f(g|\theta) = \prod_{i \in Y} \frac{1}{\theta} \exp\left(-\frac{\binom{ki}{2}}{\theta} t_i\right) \tag{3.39}$$

where g in (3.39) is the tree and will be defined below in equation (3.41), θ is a quantity named **effective population size**, and is related to the population size N via a conversion factor ρ used for time conversions [Drummond et al., 2002]

$$\theta = N\rho \tag{3.40}$$

We did a similar operation to (3.40) of multiplying quantities related to population size and time in equation (3.24), and the concept is similar in (3.40) [Drummond et al., 2002]. To express what (3.39) means, let's say we have N leaf nodes (genetic sequences), with fixed ages t_i, each $i \in I$ corresponding to an individual leaf. Define t_Y as the coalescent times, and let the edge $< i, j >$ with $i > j$ be the lineage involving nodes i and j, then if E_g is the edge set [Drummond et al., 2002]

$$g = (E_g, t_Y) \tag{3.41}$$

Equation (3.41) represents a realization of a coalescent process, given the leaf nodes and the times t_I (remember each $i \in I$ represents an individual and its associated time). We define the set of trees as $\Gamma = (E_g, t_Y)$, as explained in [Drummond et al., 2002], the integration of Γ is wrt $dg = dt_{N+1}...dt_{2N-1}$, i.e., with the exception of the t_I times of the N tree leaves. The tree g of (3.41) is characterised by the following distribution (recall equation (3.39) and (3.27)) [Drummond et al., 2002]:

$$f_G(g|\theta) = \frac{1}{\theta^{N-1}} \prod_{i=2}^{2N-1} \exp\left(-\frac{\binom{ki}{2}}{\theta}(t_i - ti - 1) \right) \tag{3.42}$$

Formula (3.42) becomes the *coalescent prior* that will be used in the expression of the posterior in Section 3.7, for a detailed derivation of (3.42) please refer to [Drummond et al., 2002], we give here a brief explanation of some key parts: the product in (3.42) comes from the independence of events in forming the coalescent events at times t_i, k_i represents the branches between t_{i-1} and t_i, and the expression in the exponential comes from combinatorics considering that the number of possibilities to form a coalescent event from 2 out of the k_i branches is $\binom{k}{2}$ [Drummond et al., 2002].

3.6 Felsenstein's likelihood

As we know by now, in a Bayesian problem setting formulation, remembering Bayes formula that we expressed in equation (2.3) at the very beginning of this work, we have that the posterior distribution π of a set of parameters θ conditional to the a set of observations y is given by the prior $p(\theta)$ multiplied by the likelihood $l(y|\theta)$. In the case of phylogenetic analysis, a traditional choice for the likelihood l is the **Felsenstein's likelihood** [Felsenstein, 1981] and represents the probability of having the observed genetic data, conditioned on the parameters which include the substitution model (discussed in Section 3.4) and the

phylogenetic tree structure (as seen in Section 3.5). The Felsenstein's likelihood can be expressed as [Felsenstein, 1981]:

$$F(D|g,Q,\mu) = \sum_{D_Y \in D} \prod_{<i,j>\in E_g} \prod_{k=1}^{L} \Big[\exp\left(Q\mu(t_i - t_j)\right) \Big]_{s_{i,k}s_{j,k}} \tag{3.43}$$

Where, in (3.43), E_g is the set of edges of the tree, as expressed in (3.41), D is the data and D_Y the data relative to internal edges, whereas Q (transition matrix) and μ (mutation rate) are parameters of the substitution model, as seen in Section 3.4 (in particular equations (3.32) and (3.34) respectively).

Explanation of the terms in the expression of the Felsenstein's likelihood

Digging a bit more into the terms of (3.43), the inner exponential term represents the probability of transitioning from nucleotide $s_{i,k}$ at node i to nucleotide $s_{j,k}$ in node j, given the substitution model expressed by Q and μ, and $t_i - t_j$ represents the coalescent time between i and j (therefore the length of the section of the tree between the two nodes i and j), there is a product in k which is extended over the length L of the genetic sequence, which represents the product of probabilities of each single site, and this is due to independence, in fact one of the assumptions of the Felsenstein likelihood is that transitions in each site happen independently of each other. Then moving out in the formula (3.43) we see a product in $< i,j > \in E_g$, and as explained in the previous sentence, i and j are two nodes connected by an edge, therefore $< i,j >$ represents an edge in the set E_g (as seen in equation (3.41)), and therefore the product is over all edges of the tree. Moving finally to the outer summation over all the possible realizations D_Y of internal states, given the DNA sequences at the tips D, that is to be considered like the discrete version of an integral over the state space of all possible realizations of internal states of the tree D_Y, which ensures all the possible combinations are considered.

3.7 Full expression of the posterior

Continuing from the previous sections 3.3, 3.4, 3.5 and 3.6, we can express the full posterior for our phylogenetic analysis as

$$Posterior(\theta, g, \Omega|D) = P(\theta)P_c(g|\theta)F(D|g,\Omega) \tag{3.44}$$

Where, in equation (3.44), the following are the parameters:

- $\boldsymbol{\theta}$ is the effective population size, as seen in equation (3.40) of Section 3.5, and can be expressed as a product of effective population size and a parameter ρ that represents the conversion of coalescent times in calendar units, as explained in Section 3.5

- $\boldsymbol{g = \{E_g, t\}}$ is the tree, with the set of edges $< i, j > \in R$ and t the coalescent times (see equation (3.41))
- $\boldsymbol{\Omega = \Omega(Q, \mu)}$ includes parameters of the substitution model, and the mutation rate, as seen in equation (3.32) for the expression of Q, and in equation (3.34) for μ

And, in equation (3.44), the following is the explanation of the terms:

- $\boldsymbol{P(\theta)}$ is the prior on the population size. For our simulations we have chosen a exponential distribution
- $\boldsymbol{P_c(g|\theta)}$ is the coalescent prior, as expressed in equation (3.42)
- $\boldsymbol{F(D|g, \Omega)}$ is the Felsenstein's likelihood of equation (3.43)

Using all the above information together, and in accordance with [Bouckaert and Lockhart, 2015], we can be expressed the posterior (3.44) as

$$Post(\theta, g, \Omega|D) \propto \lambda \exp(-\lambda\theta) \frac{1}{\theta^{N-1}} \prod_{i=2}^{2N-1} \exp\left(-\frac{\binom{ki}{2}}{\theta}(t_i - ti - 1)\right) \cdot \cdot \sum_{D_Y \in D} \prod_{<ii,j> \in E_g} \prod_{k=1}^{L} \Big[\exp(Q\mu(t_{ii} - t_j))\Big]_{s_{ii,k} s_{j,k}} \tag{3.45}$$

Equations (3.43) (and consequently (3.44) and (3.45)) assumes that the mutation rate μ be constant across all sites, which is often not the case (see for example [Bouckaert and Lockhart, 2015]). As explained in [Bouckaert and Lockhart, 2015] and [Yang, 1994], good results have been obtained by assuming that the mutation rate μ varies across sites according to a gamma distribution $\Gamma(\alpha, \frac{1}{\alpha})$, in this case the Felsenstein likelihood of equation (3.43) can be expressed as

$$F(D|g, \Omega, \alpha) = \prod_{k=1}^{L} \int_0^{\infty} \Gamma(\alpha, \frac{1}{\alpha}) \Big(\sum_{D_Y \in D} \prod_{<i,j> \in E_g} \prod_{k=1}^{L} \Big[\exp(Qr(t_i - t_j))\Big]_{s_{i,k} s_{j,k}} \Big) dr \tag{3.46}$$

Although it is common to approximate the integral of equation (3.46) with a sum over K_Γ categories, and in such case equation (3.46) becomes

$$F(D|g, \Omega, \alpha) = \prod_{k=1}^{L} \sum_{c=1}^{K_\Gamma} \Big(\sum_{D_Y \in D} \prod_{<i,j> \in E_g} \Big[\exp(Qr_c(\alpha)(t_i - t_j))\Big]_{s_{i,k} s_{j,k}} \Big) \tag{3.47}$$

Therefore in this latter case of using a Γ-distributed mutation rate μ, an additional prior is needed in equation (3.44), to account for the shape α of the $\Gamma(\alpha, \frac{1}{\alpha})$ distribution, therefore, ultimately, the full expression of the posterior, formerly in equation (3.44), is updated as follows

$$Posterior(\theta, g, \Omega, \alpha|D) = P(\theta) P_\alpha(\alpha) P_c(g|N) F(D|g, \Omega, \alpha) \tag{3.48}$$

Where, in (3.48), $P_\alpha(\alpha)$ represents the prior for the parameter α which determines the shape of the additional Γ distribution

3.8 Some history of software for phylogenetics: from BEAST to BEAST2

A group of researchers in the field of phylogenetics created a software named **Bayesian Evolutionary Analysis Sampling Trees**, or with its acronym **BEAST**, [Drummond and Rambaut, 2007, Drummond et al., 2012] and a community around it. Mixing knowledge in software engineering, phylogenetics and statistics, they have been able to create a software platform to enable researchers to solve real-world phylogenetic problems, for example like the one in the format of Section 3.7, in a Bayesian framework. As the size of the software project grew over the years, some in the community felt the need to create a second branch of work, naming the new software **BEAST2** [Bouckaert et al., 2019], to indicate clearly that it was born from BEAST. One of the main developments that BEAST2 brought was in terms of application of software engineering principles, bringing more modularity, better scalability and management of packages and features additions [Bouckaert et al., 2019]. To our knowledge, both BEAST and BEAST2 continue to exist and are updated independently. We have developed our work entirely in BEAST2, and will therefore from now on only concentrate and discuss the BEAST2 platform.

3.9 Introduction to BEAST2 software: BEAUTI and BEAST2

We will in this section give an overview of **BEAST2**. For a full description of the software please refer to [Bouckaert et al., 2019], we will give here only a brief introduction of the parts useful to our work. Two of the software components that we have used in the BEAST2 package are the software **BEAUTI**, used to create configuration files, and the main software **BEAST2** which runs a MCMC analysis based on the input configuration files created by BEAUTI.

3.9.1 BEAUTI: build configuration files

The software BEAUTI takes as input **taxa** data files, i.e. files that have genetic sequences of species on which the analysis will be performed, these are to be considered the data of our Bayesian problem, the leaves of the coalescent tree. The following is an extract of a sample input file for BEAUTI with sequences from three species (the nucleotides sequences have all been cut to a length of 20 for easiness of representation): Tarsius_syrichta, Lemur_catta and Homo_sapiens

```
#NEXUS
begin data;
```

```
dimensions ntax=3;
format datatype=dna interleave=no gap=-;
matrix
Tarsius_syrichta AAGTTTCATTGGAGCCACCA...
Lemur_catta AAGCTTCATAGGAGCAACC...
Homo_sapiens AAGCTTCACCGGCGCAGTC...
;
end;
```

The software BEAUTI allows to select the Bayesian model settings such as the priors to be used, and parameters of the substitution model (see Sections 3.3 and subsequent), and also auxiliary parameters such as the desired number of MCMC iterations to be used for the analysis. The output file of BEAUTI is a *xml* configuration file which will be fed into the main software component of the package, named **BEAST**, which is to be used to perform the actual Bayesian statistical analysis.

3.9.2 BEAST2: Bayesian inference of phylogeny from sequence data

The software BEAST2 will take as input the *xml* configuration file generated by the software BEAUTI as described in the previous Section 3.9.1. The *xml* file will contain such information as the input nucleotide sequences representing the data of our problem. The software BEAST2 will perform MCMC analysis on the input data, using the settings provided in the *xml* file, and will output MCMC samples, which will approximate a sequence of parameters of the state space of the posterior. In particular, phylogenetic trees will be part of the state space (see description of the posterior in Section 3.7) and for trees the samples of the chain will contain all the internal states (coalescent times and internal sequences) that have been statistically inferred starting from the input leaves represented by the genetic sequences. Below an extract from an output file from BEAST2, in particular we can appreciate the internal representation of a coalescent tree of three taxa (internally named t_1, t_2 and t_3), in the $treeSTATE\_0$ line below we see the internal representation of the starting position of the MCMC chain for the tree, the representation format is called **Newick** (see for example [Yu, 2022]):

Listing 3.1: Coalescent tree with 3 taxa in BEAST2

```
Begin trees;
Translate
1 t1,
2 t2,
3 t3
;
```

```
tree STATE_0 = ((1:0.5665,2:0.5665):0.3993,3:0.9658):0.0;
```

Internal representation of the coalescent tree

We see in Figure 3.4 below a graphical representation of the three in listing 3.1 with internal BEAST2 representation $treeSTATE\_0 = ((1 : 0.5665, 2 : 0.5665) : 0.3993, 3 : 0.9658) : 0.0;$: 0.5665 is the time in the past of estimated coalescent between t_1 and t_2, then after an additional time of 0.3993 there is an additional coalescent event of t_3 with the branch formed by t_1 and t_2, until the **MRCA**, i.e. the estimated Most Common Recent Ancestor of all the sequences.

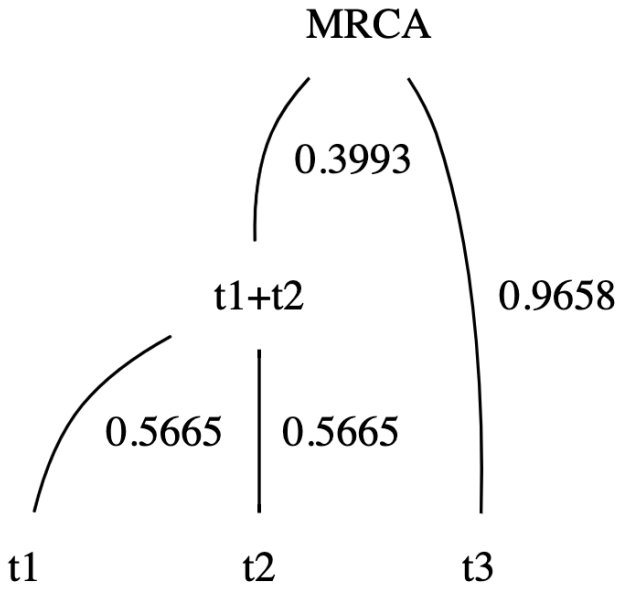

Figure 3.4: *Example of coalescent tree composed by three taxa, named t_1, t_2 and t_3 in the figure, corresponding to the internal tree state in the software BEAST2* $((1 : 0.5665, 2 : 0.5665) : 0.3993, 3 : 0.9658) : 0.0;: 0.5665)$*, as described in the paragraph. We know that coalescent times are to be intended in the past (see Section 3.3), therefore the timescale has to be intended from the bottom (time of samples) to the top (time in the past until when the three species had converged into a single ancestor, named MRCA, Most Common Recent Ancestor). The diagram has been created using the package graphviz.*

The output of BEAST2 MCMC runs will be a sequence of trees, like the one represented in Figure 3.4, plus the MCMC chain of the other components of the state space (of which we talked about when describing the posterior in Section 3.7)

3.10 MCMC example with the standard coalescent in BEAST2

We show now an example of analysis performed on a sample set of 7 taxa using BEAST2. As explained in Section 3.5, for the standard coalescent model with N sequences, we need to estimate the following parameters:

- Population size;
- Tree;
- Substitution model;
- Possibly other parameters (for example the shape of the Gamma distribution for equations (3.46) or (3.48));

Through the configuration software BEAUTI, introduced in Section 3.9.1, we set the configuration of the model, and in particular choose the priors, and for the parameters above we have chosen the priors as follows

- Population size: exponential prior with mean 0.33;
- Tree: coalescent prior with constant population size;
- Substitution model: Jukes Cantor 69, introduced in Section 3.4.1, will be used. As a reminder, the Jukes Cantor assumes all rates equal for the nucleotides (elements of the transition matrix (3.36)), and as a consequence assumes equal equilibrium frequencies;
- In addition to the parameters of the substitution model expressed in the previous point, we also want to account for variability of rates across sites; to model rate variability we will use a gamma site model with 4 categories, i.e. we use 4 groups representing 4 quantiles for the variability of the gamma shape (see for example [Bouckaert and Lockhart, 2015]): looking at the discretised equation of the full posterior (3.47), this means using $K_\Gamma = 4$.

3.10.1 Results

A run of the standard BEAST2 software has been run on the sample set of 7 taxa introduced in this section. A run with BEAST2 will produce two main outputs:

- A **log** file containing the samples of the MCMC chain for the parameters of the state space, except trees, and in addition other elements such as the values of prior, likelihood and posterior at each point;

- A **trees** file, containing the MCMC chain of the coalescent tree at each iteration of the MCMC. The internal representation of the tree is as described in Section 3.9.2 (in the listing 3.1), with nested brackets indicating coalescent events, and with the associated coalescent times.

3.10.2 Visualization of results for all parameters except coalescent trees

There are some graphical tools provided by the BEAST2 platform to analyse the results, these perform the standard set of checks for MCMC like histogram plot with the distribution of the posterior (also likelihood and prior), and with the single distributions for the parameters of the state space except the trees. One such a tool is the **Tracer**, which also provides ESS estimates of the MCMC runs (as a reminder, ESS for MCMC has been introduced in Section 2.4.8). Below a screenshot of the Tracer, with the posterior of the 7-taxa example discussed in this section

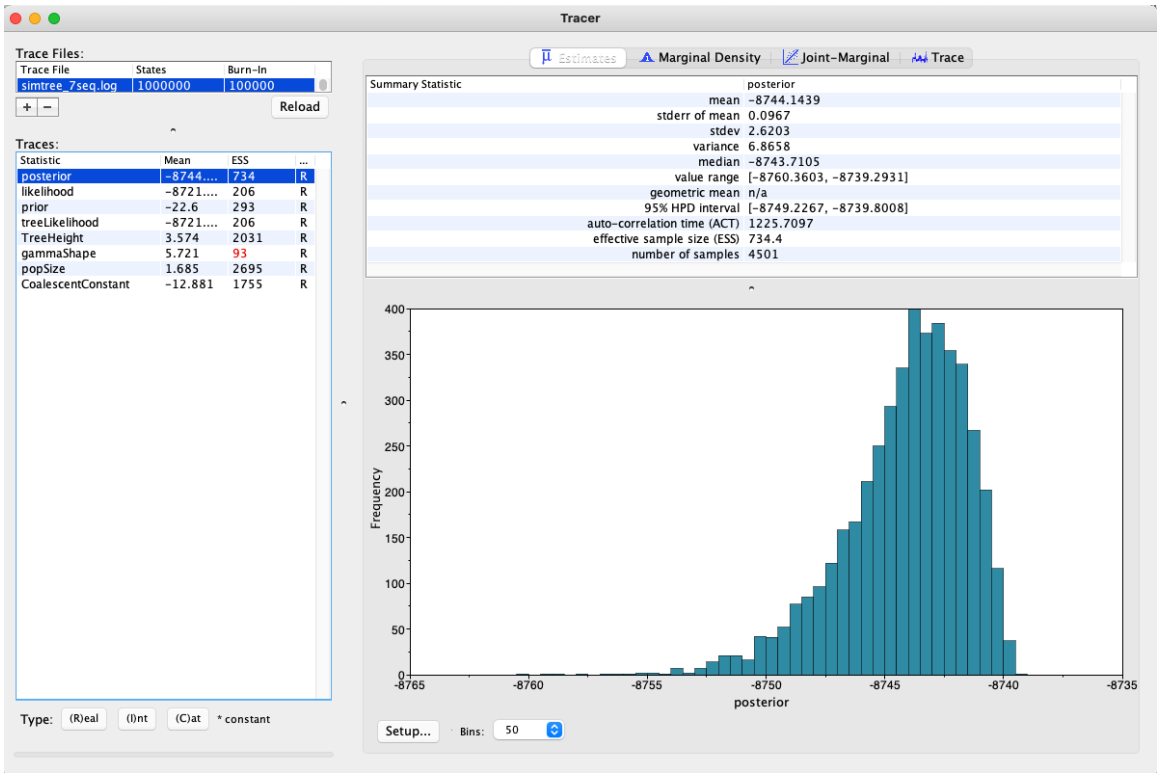

Figure 3.5: *The **Tracer**, a software part of the software package BEAST2. The software is capable of displaying parameters relative to the MCMC runs, like estimated distributions and ESS of the chains.*

3.11 Visualization of results for coalescent trees

It is not easy to visualize a complex object like a coalescent tree. And it is also not immediate to understand how to measure and estimate convergence in a MCMC chain of coalescent trees which are in the format shown in Section 3.9.2 (in the listing 3.1).

3.11.1 TreeAnnotator

Firstly, there is a BEAST2 software named **TreeAnnotator**, which performs **Consensus Tree analysis**, which gives information on the uncertainty related to the estimated value of the coalescent tree.

3.11.2 FigTree

The software **FigTree** is a freely downloadable software [Rambaut, 2023] which can take as input the trees MCMC chain and perform some visualization of the samples. Below a screenshot of the interface

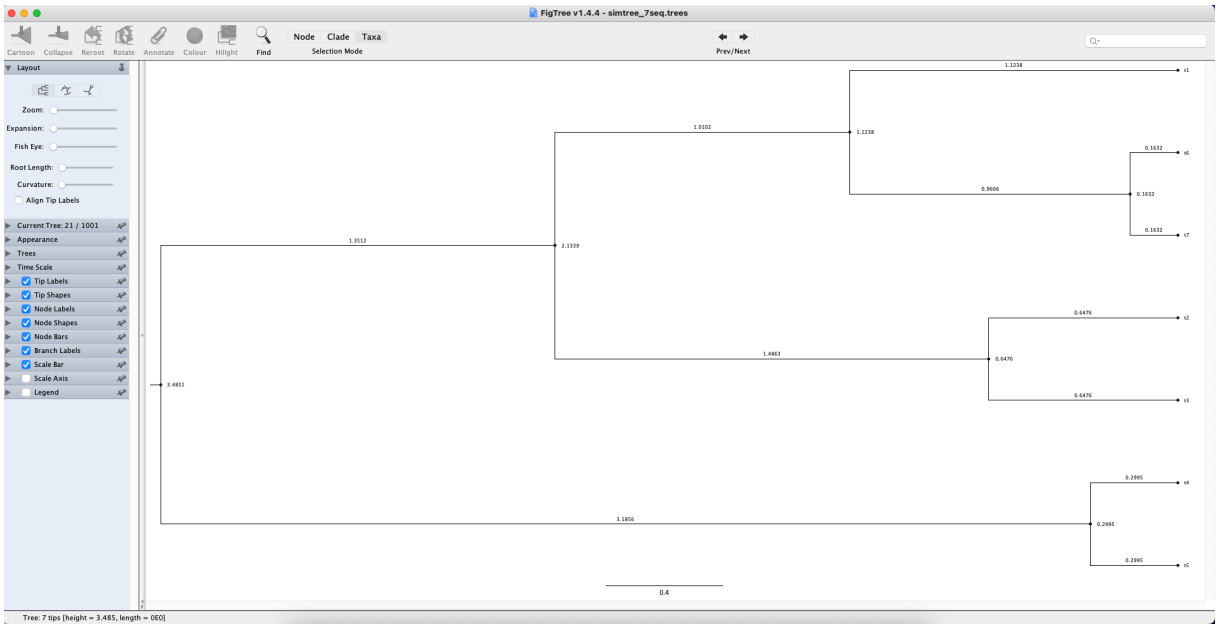

Figure 3.6: *The **Figtree** software [Rambaut, 2023], useful to visualize and get information on the MCMC chain of trees generated by BEAST2.*

3.12 Evaluating the SMC Annealed Adaptive phylogenetic vs BEAST2 MCMC

We will, starting in this section, display the results of running the Annealed Adaptive SMC algorithm we integrated in BEAST2, and compare the performances with the native BEAST2

MCMC. Firstly, we remind the format of the posterior and we rewrite here, for convenience, some parts of Section 3.7 where the full posterior was explained. The posterior is as follows

$$Posterior(\theta, g, \Omega, \alpha|D) = P(\theta)P_\alpha(\alpha)P_c(g|N)F(D|g, \Omega, \alpha) \tag{3.49}$$

Where, in (3.49)

- $P(\theta)$ is the prior on the effective population size θ, which in our simulations we have chosen exponential $\lambda \exp(-\lambda\theta)$ with $\lambda = 0.33$
- $P_\alpha(\alpha)$ is the prior on the α parameter of the shape of the Γ distributed mutation rate, which we have chosen to be $\Gamma(3, 2)$
- $P_c(g|N)$ is the coalescent prior

$$\frac{1}{\theta^{N-1}} \prod_{i=2}^{2N-1} \exp\left(-\frac{\binom{ki}{2}}{\theta}(t_i - ti - 1)\right)$$

- $F(D|g, \Omega, \alpha)$ is the Felsenstein's likelihood

$$\prod_{k=1}^{L} \int_0^\infty \Gamma(\alpha, \frac{1}{\alpha}) \left(\sum_{D_Y \in D} \prod_{<i,j> \in E_g} \prod_{k=1}^{L} \left[\exp(Qr(t_i - t_j)) \right]_{s_{i,k} s_{j,k}} \right) dr \tag{3.50}$$

  a number of 4 categories has been used to approximate the integral of (3.50)

We will therefore present the results of the simulations with respect to the ability of the algorithms to reconstruct correctly the parameters of the state space:

- α parameter for Γ shape
- population size
- coalescent tree

3.13 Generation of Synthetic Data

To assess the performance of our SMC algorithm, we designed some tests using synthetic data, so that, by knowing the underlying generation parameters, we could assess the performance of the algorithms in reconstructing the model.

3.13.1 Synthetic Data Configuration

We generated synthetic data with variable number of sequences using a two-step approach:

1. **Tree Generation:** We employed the R library *'ape'* to simulate random coalescent trees with a specified number of leaves, the output of this process was random coalescent trees having specified number of leaves.

2. **Sequence Evolution:** We used the 'seq-gen' software [Rambaut and Grassly, 1997] to simulate DNA sequences evolving along these generated trees, according to the evolution model that we have chosen in advance (we worked therefore *backwards*, i.e. knowing what priors and models we would use in BEAST2, we generated data accordingly), the specific parameters used in 'seq-gen' were as follows:

    - Substitution model: GTR (General Time Reversible)
    - Nucleotide frequencies: 0.25 for each base
    - Substitution rates: Equal for all possible substitutions
    - Gamma shape parameter: 1.0 (moderate among-site rate variation)
    - Number of gamma categories: 4
    - Sequence length: 1000 nucleotides

The GTR model mentioned above [Tavaré, 1986] is a more general substitution model than the Jukes Cantor we described earlier in Section 3.4.1, but with the settings above (equal mutation rates and equal equilibrium frequencies) the GTR reduces to Jukes Cantor, therefore, from the setting above, the sequences have been generated according to the Jukes Cantor substitution model.

BEAST2 XML Configuration

Accordingly to the settings used for the synthetic data, we generated a BEAST2 XML file with consistent substitution model and priors. In particular, by looking at the posterior expression that we outlined in equation (3.49) in Section 3.12 we can spot in the code snippet below from the BEAST2 xml configuration file some of the parameters configurations

```
<substModel id=\textcolor{black}{``}JC69.s:SimTree" spec=\textcolor{black
<frequencies>0.25 0.25 0.25 0.25</frequencies>
</substModel>
<siteModel id=\textcolor{black}{``}SiteModel.s:SimTree" spec=\textcolor{b
gammaCategoryCount="4"
  shape="@gammaShape.s:SimTree">
<substModel>JC69.s:SimTree</substModel>
</siteModel>
```

3.14 Data with 10 Taxa

Following the procedure outlined in Section 3.13.1, we have generated a synthetic model with 10 taxa. Additional results for 5 and 20 taxa cases are reported in Appendix A.

Generator tree

The first step has been as described in 3.13.1 to generate a coalescent tree with 10 leaves, and the tree has been randomly generated as below in Figure 3.7

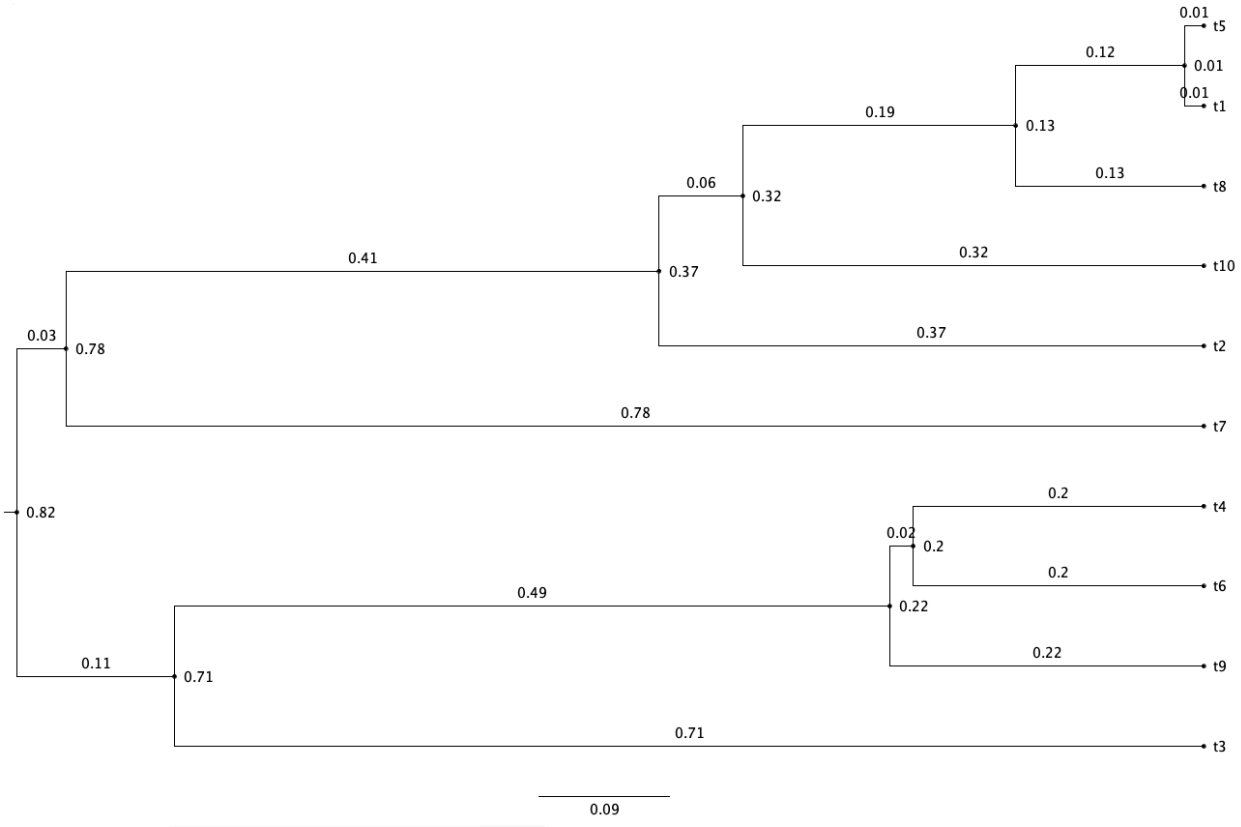

Figure 3.7: *Random coalescent tree with 10 leaves generated using the procedure outlined in the first part of Section 3.13.1. This has been the generating tree for the synthetic data of the test described in this section. Visualization via FigTree [Rambaut, 2023]*

Generation of synthetic sequences

Using the tree generated in the previous step, synthetic sequences have been generated using 'seq-gen' program, as explained in Section 3.13.1

3.15 Annealed Adaptive SMC vs MCMC in BEAST2, problem set up with 10 Taxa

We have run BEAST2 both with the traditional MCMC algorithm and with our Annealed Adaptive SMC embedded in BEAST2, and we report here the comparison. A fair comparison in terms of likelihood evaluations has been kept between the two methods. For the comparison of results we have used a similar set-up and metrics of [Wang et al., 2019], in fact we have a number of iterations of MCMC which is comparable with the likelihood evaluations of the SMC algorithm, given by the number of particle times number of intermediate tempering steps of the annealing procedure, times the number of MCMC moves per annealing step. So, considering the comparison fair we report below the results for the various parameters of the state space.

3.15.1 SMC set up

The SMC has been set up with 1000 particles, and 5 MCMC moves per each annealing step. The number of annealing steps adaptively determined by the CESS (see Section 2.6.3 for details on CESS) has been 55, as can be seen in Figure 3.8

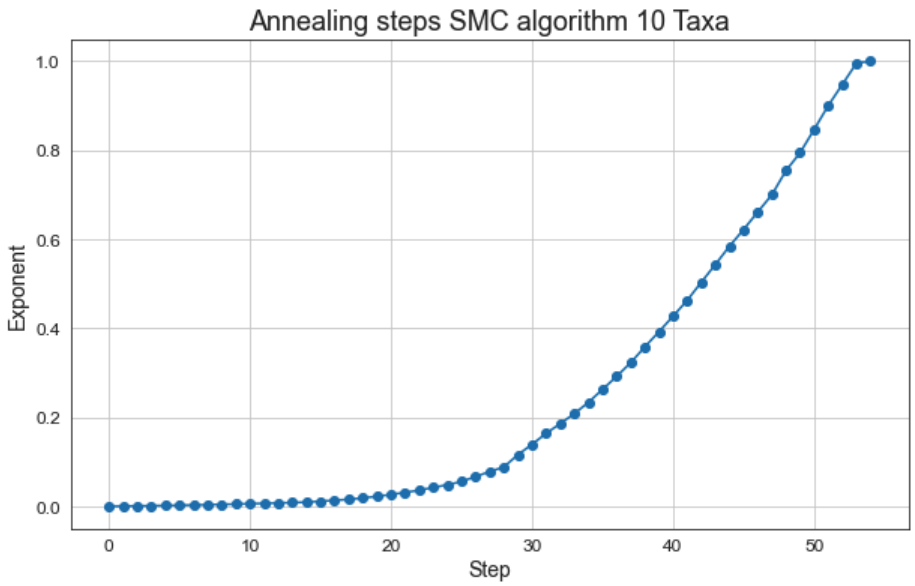

Figure 3.8: *Annealing steps in the SMC run for the* 10*-taxa example studied in this section.*

Therefore the total number of likelihood evaluation for the algorithm has been $1000 \times 55 \times 5 = 275000$. The adaptive annealing steps have been determined using CESS with a threshold of 90%, and resampling of particles is done when ESS falls below 50% of particles, we can see below in Figure 3.9 the ESS chart related to the SMC run

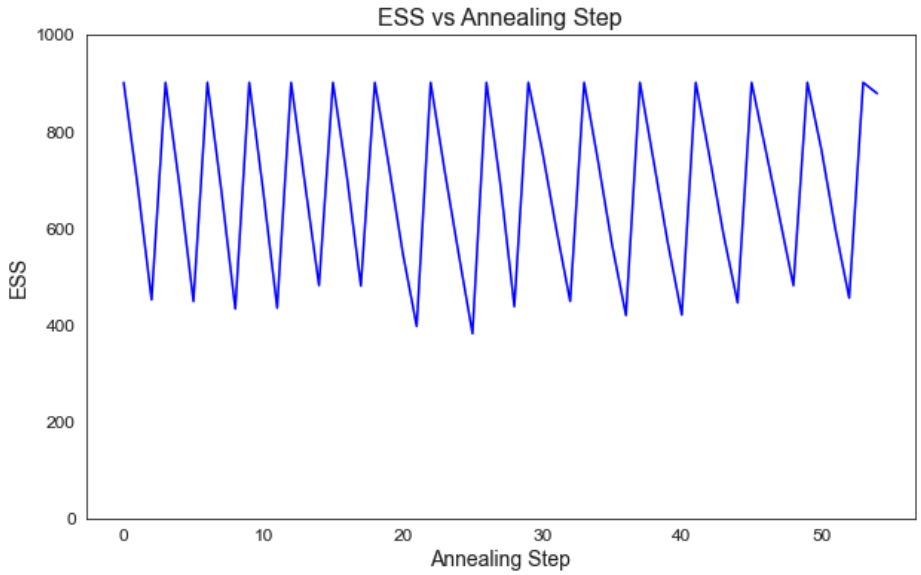

Figure 3.9: *ESS versus annealing SMC step for the* 10*-taxa example studied in this section. Resampling is performed whenever the ESS falls below* 50% *of particles (*1000 *particles are used for the simulation).*

3.15.2 MCMC set up

Considering that the total number of likelihood evaluations from the Annealed Adaptive SMC was 275000, we have used a comparison similar to [Wang et al., 2019] and therefore we have used a number of MCMC iterations roughly 20% greater than the number of SMC likelihood evaluations, in our MCMC simulation we have used 350000 iterations.

3.16 Annealed Adaptive SMC vs MCMC in BEAST2: Results with 10 Taxa

We will report in this section the results for the runs of traditional MCMC and Annealed Adaptive SMC using BEAST2, the parameters analysed are those composing the state space of our problem (see Sections 3.7 and 3.13.1):

- Gamma shape parameter
- Effective population size
- Coalescent Tree

3.16.1 Gamma shape

The true value of the gamma shape parameter (i.e. the value with which the data has been generated) is 1. In both the MCMC and the SMC runs the empirical distributions are rather

scattered as we can appreciate from Figures 3.10 for MCMC and 3.11 for SMC, and this is due to the fact that the probability of MCMC moves on the gamma shape parameter has been kept to the default value that the configuration software BEAUTi (see Section 3.9.1) gives, and MCMC moves are less likely to happen than moves on effective population size and trees (as an example, an MCMC move on the gamma shape is 30 times less likely than a move on the Effective Population Size parameter), therefore the low ESS and the scattered distributions are a result of this. A better tuning of the frequency of moves should improve the results.

MCMC results for Gamma shape

The mean of the MCMC run is close to the true value of 1, we can see the full statistics in the following table. The acronym HPD in table 3.16.1 stands for Highest Posterior Density Interval, which represents the range within which the parameter falls with 95% probability given the data.

Statistic	Value
Mean	1.1159
Standard Deviation	0.1208
Value Range	[0.9516, 1.6006]
95% HPD Interval	[0.9516, 1.3179]
Effective Sample Size (ESS)	54

Table 3.1: Statistics for the parameter Gamma shape

And the distribution of the Gamma shape values is in Figure 3.10

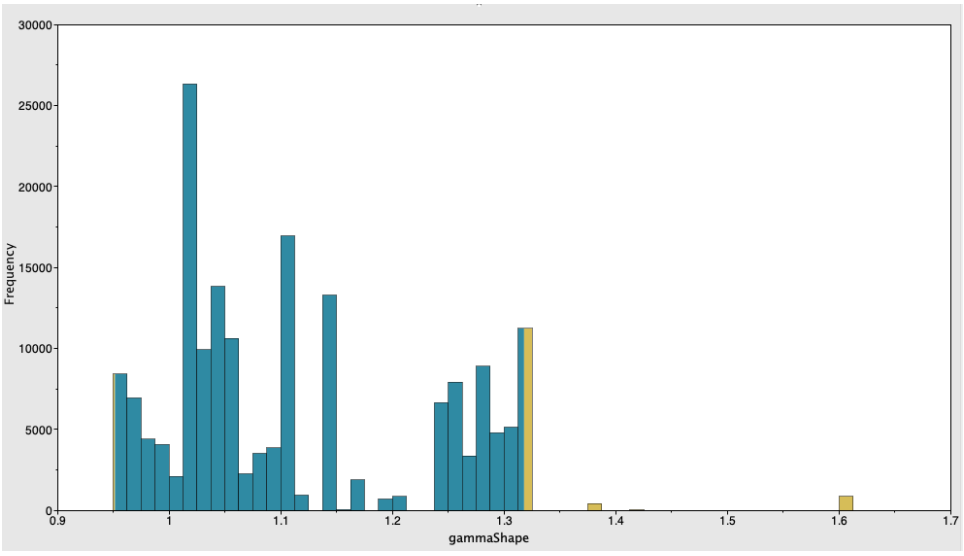

Figure 3.10: *Frequency distribution using the native MCMC run with BEAST2 for the parameter Gamma shape with 10 taxa. The values inside the 95% HPD interval shown in blue and those outside the 95% HPD interval highlighted in gold. Visualization with the software Tracer.*

Ammealed Adaptive SMC results for Gamma shape

We can see in the table below the statistics for the SMC run, and although the mean is very close to the true value of 1, from Figure 3.11 we see that these results should be taken with a pinch of salt, since the particle diversity is not good, future tuning of the frequency of the moves on this parameter should improve the particle diversity

Statistic	Value
Mean	0.9871
Standard Deviation	0.066
Value Range	[0.8282, 1.3063]
95% HPD Interval	[0.9743, 1.2840]

Table 3.2: Statistics for the parameter Gamma shape

And the distribution of the Gamma shape values is in Figure 3.11

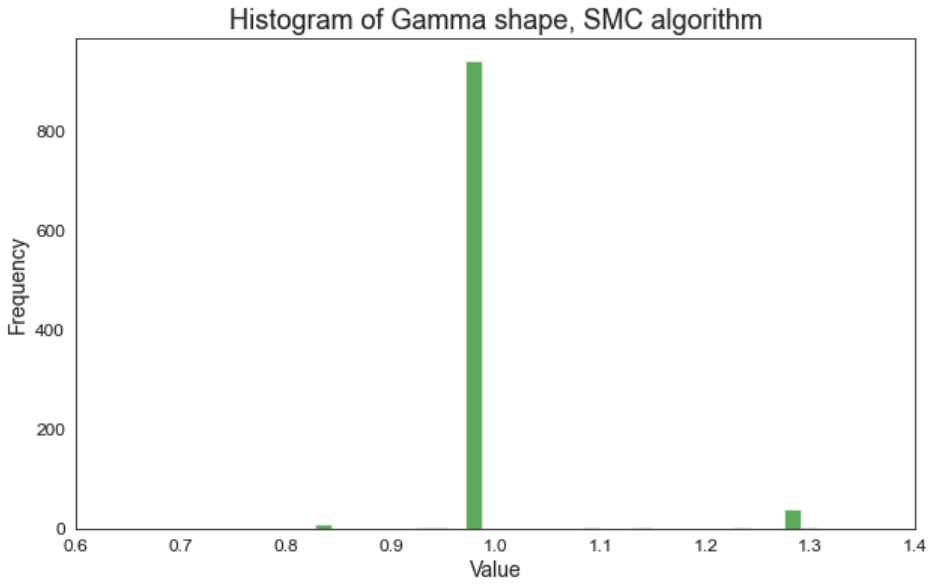

Figure 3.11: *Frequency distribution for the parameter Gamma shape with 5 taxa, using Annealed Adaptive SMC algorithm that we have embedded into BEAST2. Visualization with python matplotlib.*

3.16.2 Effective Population Size

Unlike the Gamma shape parameter, seen in Section 3.16.1, where we had at hand the true value of the parameter, and that was useful in checking how well MCMC and SMC could retrieve it, for the Effective Population Size, even if we generated synthetic data, it is not straightforward to have the true value. In fact, as explained in Section 3.5, we are usually only able to estimate the product of two terms, the coalescent constant and the Effective

Population Size. Anyway, keeping this constraint in mind, we report the results of the analysis in this section.

MCMC results for Effective Population Size

The mean of the MCMC run is 1.73, we can see the full statistics in the following table

Statistic	Value
Mean	1.73
Standard Deviation	0.74
Value Range	[0.488, 8.205]
95% HPD Interval	[0.721, 3.078]
Effective Sample Size (ESS)	1027

Table 3.3: Statistics for the Effective Population Size

And the distribution of the Effective Population Size is in Figure 3.12

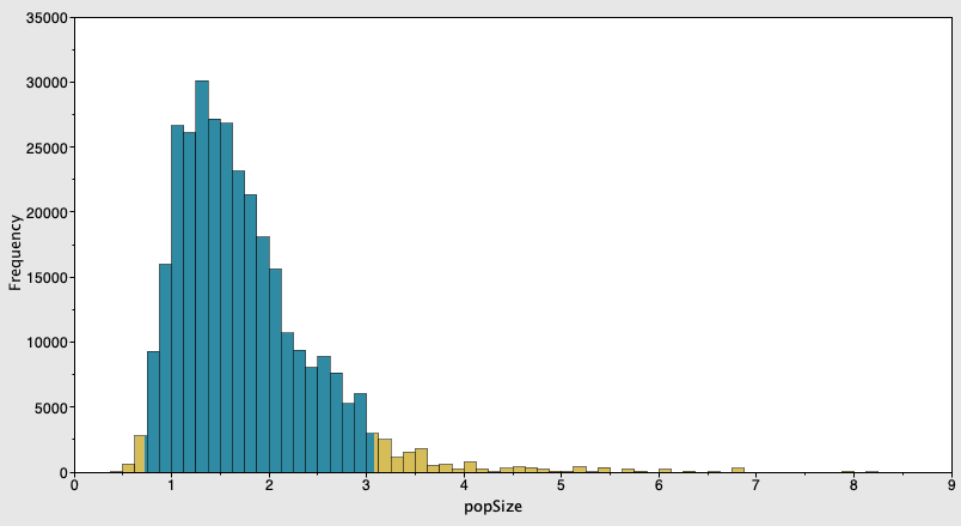

Figure 3.12: *Frequency distribution using the native MCMC run with BEAST2 for the parameter Effective Population Size with 10 taxa.The values inside the* 95% *HPD interval shown in blue and those outside the* 95% *HPD interval highlighted in gold. Visualization with the software Tracer.*

Ammealed Adaptive SMC results for Effective Population Size

The statistics for the SMC run are in general better than the MCMC run, we can see a lower variance for example. And we can see from Figure 3.13, that it has the same peak of the correspective MCMC Figure 3.12, but in the MCMC case the bigger variance and right-skewness is causing a slightly higher value of the mean:

Statistic	Value
Mean	1.683
Standard Deviation	0.556
Value Range	[0.61, 4.29]
95% HPD Interval	[0.81, 2.96]

Table 3.4: Statistics for the Effective Population Size

And the distribution of the Effective Population Size is in Figure 3.13

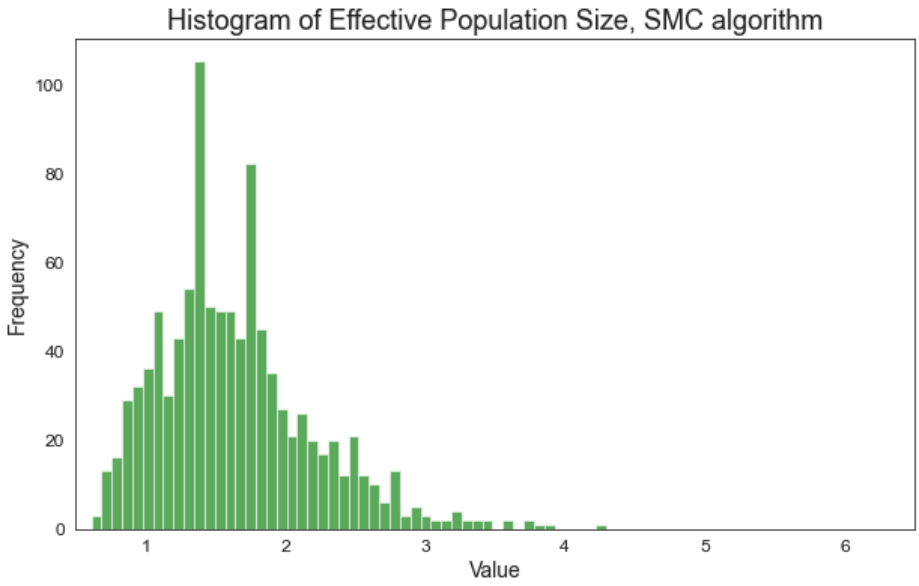

Figure 3.13: *Frequency distribution for the parameter Effective Population Size with 10 taxa, using Annealed Adaptive SMC algorithm that we have embedded into BEAST2. Visualization with python matplotlib.*

We can also appreciate the SMC algorithm at work by looking at Figure 3.14 below, showing the evolution of the estimated standard deviation of the *population size* parameter vs the annealing step for the parameter Effective Population Size, and we see how the standard deviation drops significantly through the annealing journey

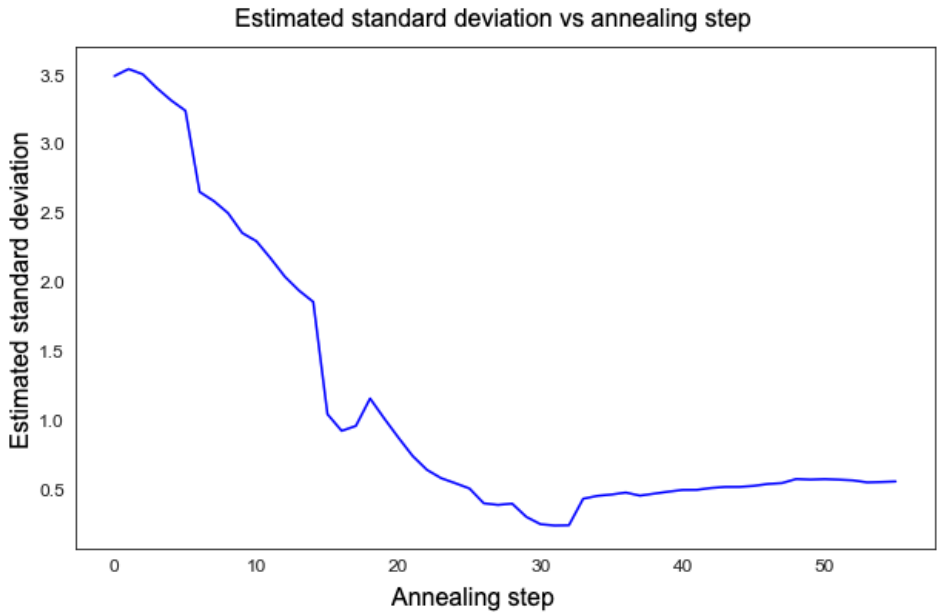

Figure 3.14: *Evolution of the estimated standard deviation of the population size parameter vs the annealing step for the parameter Effective Population Size in the SMC algorithm: we see how the standard deviation drops significantly through the annealing journey.*

3.16.3 Tree

For the tree analysis we use a methodology similar to [Wang et al., 2019] and we compare trees using the *majority-rule consensus*. So we will have a the *consensus-tree*, which is a "summary" tree for the MCMC run and one for the SMC run, and we will compare them to the generating tree shown in Section 3.14 to assess how each of the two algorithms has performed. In addition to visualizing the "summary" trees for the two runs, we will also give a basic topological metric of performance, the Robinson-Foulds (RF) "symmetric difference" metric [Robinson and Foulds, 1981], which will identify possible topology mismatch with the reference tree. The consensus tree has been generated using **TreeAnnotator** and then the visualization using **FigTree**. For the SMC algorithm, the particles have been resampled in order to be able to compare SMC tree samples without the need to consider the particle weights when building the consensus, for ease of calculation.

MCMC results for Tree

The RF metric result for the run is 0, meaning a match from a topological point of view with the reference tree of Section 3.14, and we can see from the picture below 3.15 the consensus tree created with visualization of the 95% confidence range in the coalescent times (see the comparison with the generator tree of Figure 3.7)

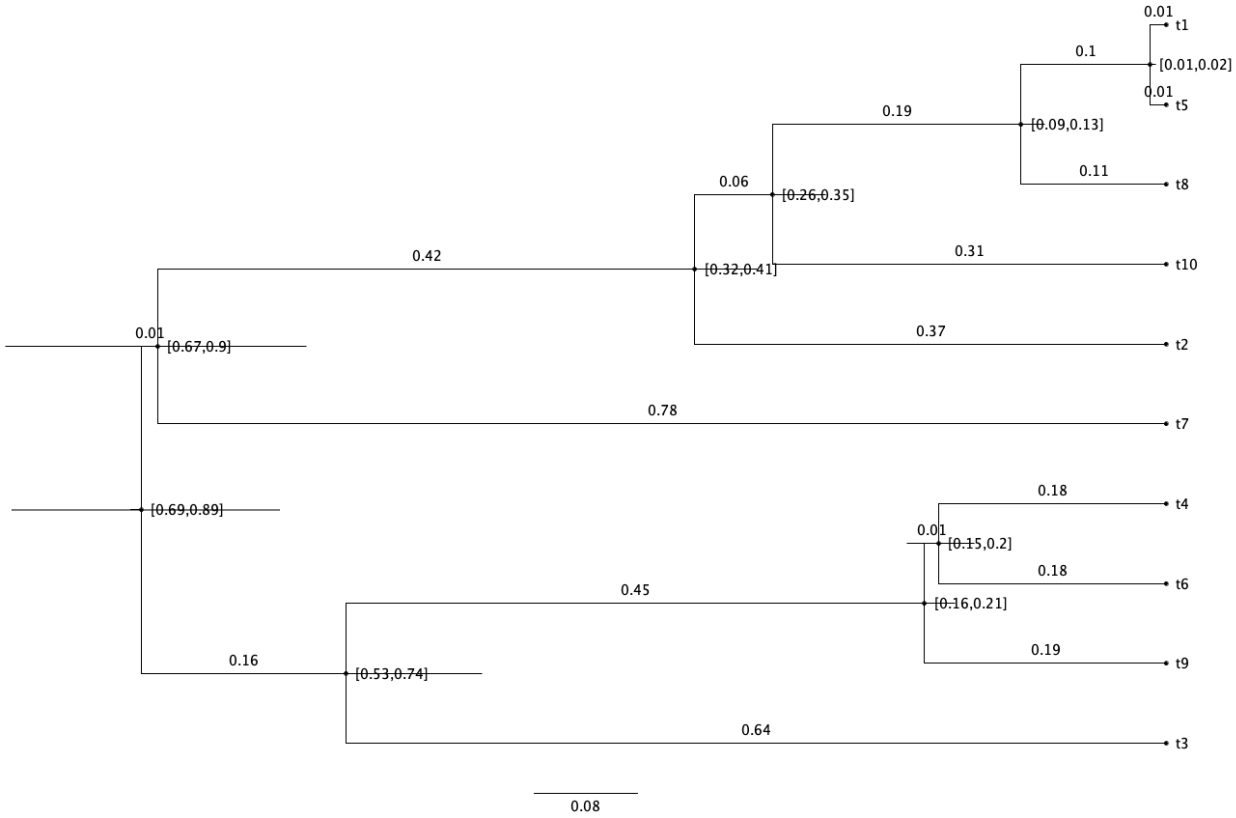

Figure 3.15: *Consensus tree for the MCMC run with visualization of* 95% *range for coalescent times. See comparison with the generating tree (which the MCMC run tries to reconstruct) in Figure 3.7. Consensus tree has been generated with TreeAnnotator and the visualization is with FigTree (both softwasre from BEAST2 package).*

Annealed Adaptive SMC results for Tree

The RF metric result for the run is 0, meaning a match from a topological point of view with the reference tree of Section 3.14, and we can see from the picture below 3.16 the consensus tree created with visualization of the 95% confidence range in the coalescent times (see the comparison with the generator tree of Figure 3.7, and with the MCMC-generated consensus tree of Figure 3.15)

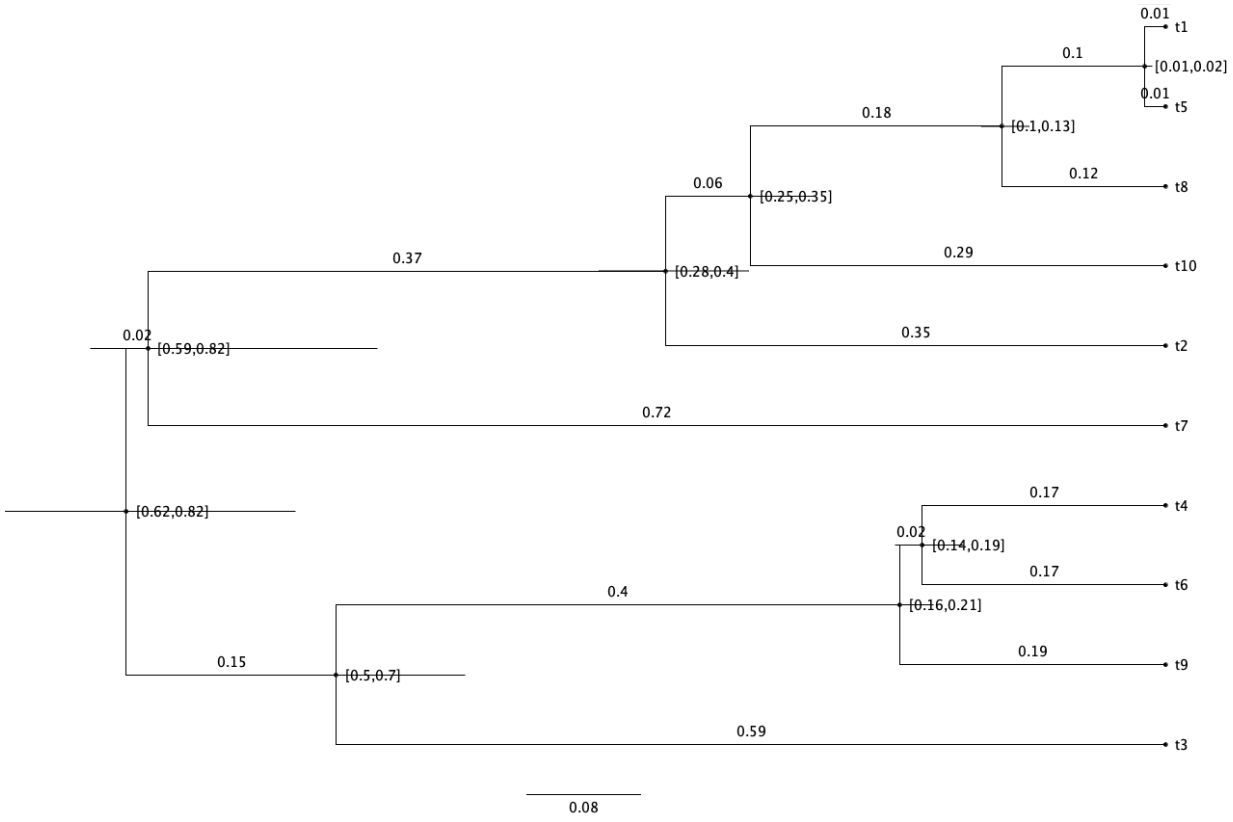

Figure 3.16: *Consensus tree for the Annealed Adaptive SMC run with visualization of* 95% *range for coalescent times. See comparison with the generating tree (which the SMC run tries to reconstruct) in Figure 3.7, and also with the tree reconstructed using MCMC in Figure 3.15: we see that the SMC is able to reconstruct the generating tree well and with a smaller uncertainty (the* 95% *uncertainty ranges in the coalescent times are in general smaller compared to the MCMC of Figure 3.15). Consensus tree has been generated with TreeAnnotator and the visualization is with FigTree (both softwasre from BEAST2 package).*

By comparing Figure 3.16 with Figure 3.15 we can see that the SMC algorithm has been able to reconstruct the generating tree with similar performances to the MCMC algorithm. We may suppose that as the complexity of the posterior increases (the current examples have been produced synthetically), the Annealed Adaptive SMC will outperform the MCMC for the reasons outlined in sections 2.5 and 2.6, namely that Annealed SMC navigates better than MCMC in complex distributions. On the other hand MCMC is often simpler and more efficient for exploring posteriors that are not highly multimodal or complex.

3.17 Conclusion

In this section we have shown how we successfully integrated a Sequential Monte Carlo algorithm into the BEAST2 platform. Although we have developed our work independently from [Wang et al., 2019], our implementation in BEAST2 can be said to implement all the various algorithms of [Wang et al., 2019] in BEAST2 environment, in particular the results we

have reported in Section 3.12 are for the most advanced of the algorithm mentioned in [Wang et al., 2019], i.e. the *Annealed Adaptive SMC*. Our implementation is integrated within the BEAST2 platform, complementing the Markov Chain Monte Carlo (MCMC) method used natively in phylogenetic analyses by BEAST2.

To our knowledge, this is the first implementation in BEAST2 of a SMC algorithm. The results obtained from our implementation, discussed in detail in Section 3.12, demonstrate that, for the particular cases analysed, the annealed adaptive SMC algorithm achieves performance comparable to the native MCMC method of BEAST2 in terms of both statistical accuracy and computational efficiency. One of the advantages of the SMC method is that we have been able to achieve similar performances to the MCMC by using far fewer output samples. We may suppose that as the complexity of the posterior increases (the current examples have been produced synthetically), the Annealed Adaptive SMC will outperform the MCMC for the reasons outlined in sections 2.5 and 2.6, namely that Annealed SMC navigates better than MCMC in complex distributions. On the other hand MCMC is often simpler and more efficient for exploring posteriors that are not highly multimodal or complex.. If we pick the 10 taxa example we discussed in Section 3.14: as explained in Section 3.15.1, our SMC implementation has used 1000 particles, resulting in as many output samples, that can be used, for example, to compute expectations. In the native MCMC of BEAST2, in the set up to achieve similar number of likelihood evaluations of the SMC, 350000 iterations have been used, resulting in as many output samples. Even if we were to discard, say, the first 20% as burn-in, in order to calculate an expectation we would still have to use an impressive 280000 MCMC samples versus the 1000 of the SMC, with significant additional computational time. And in cases of many leaves, things are likely to get worse quickly, especially for information-dense objects as trees: for example, in the case of 20 taxa of Appendix A.4, the software TreeAnnotator, described in Section 3.11.1, needs to be run in a low memory configuration when processing the tree output samples from MCMC (480000 samples as explained in Appendix A.5.2) in order not to hang up, whereas it runs smoothly on the 1000 SMC samples. We assume that, with growing number of leaves, the difference will probably be even more remarkable.

It has required a non trivial amount of work to integrate the algorithms in a complex platform like BEAST2, as we had to dig deep into the software details of the platform. This involved understanding and modifying core components to ensure compatibility and efficiency. We see the equivalence in results with the native BEAST2 MCMC as a first step, and future tuning of the algorithm and of the integration within BEAST2 can improve the results.

Our analysis has so far been limited to synthetic data. Future research should focus on more complex scenarios, particularly those involving multimodal distributions. Such distributions are likely to present a more useful test for the Annealed Adaptive SMC algorithm. We may suppose that in these more complex scenarios, the Annealed Adaptive SMC will out-

perform the MCMC for the reasons outlined in sections 2.5 and 2.6, namely that Annealed SMC navigates better than MCMC in complex distributions.

Moreover, we see a significant scope for advancement in refining the tuning of parameters within the BEAST2 platform to optimize the performance and the integration of the Annealed Adaptive SMC algorithm.

In summary, we see our work as a step forward in the application of Monte Carlo methods in phylogenetic analysis. By embedding the Annealed Adaptive SMC algorithm into the BEAST2 platform, we have opened to the possibility of using additional algorithms in a widely used statistical software platform.

Chapter 4

Active Subspaces

4.1 Introduction

As we have explained in previous chapters, Monte Carlo (MC) methods have been widely used in the exploration of complex and intractable distributions. However, their effectiveness diminishes in some scenarios, for example in high-dimensional settings with a phenomenon known as the *curse of dimensionality* (we have discussed it in Section 2.2). The dimensionality problem is particularly visible for MC methods such as Importance Sampling (IS) (see Section 2.3 and 2.5.1), Markov Chain Monte Carlo (MCMC) (sections 2.4 and 2.5.2) and Sequential Monte Carlo (SMC) (Section 2.6): all these methods are impacted in general in a worse-than-linear way [Agapiou et al., 2017, Beskos et al., 2011, 2014]. Active Subspaces (AS) have been introduced primarily as a method to mitigate the high-dimensional challenges in generic mathematical systems [Constantine, 2015], with some later applications to Monte Carlo algorithms (see for example [Constantine et al., 2016, Schuster et al., 2017, Parente, 2020]).

AS represent a smaller dimension of the system that is less than the nominal dimension of the state space, this intrinsic lower-dimensional subspace is present for example in over-parametrised systems [Agapiou et al., 2017]. Giving a more precise definition, we call **Active Subspace** a subspace informed by the data identified by the directions of biggest change of the negative log-likelihood; the remaining part of the state space is called **inactive**, is orthogonal to the Active Subspace and will in the ideal case only be informed by the prior. The exploitation of the Active Subspaces can significantly improve the efficiency of MC algorithms [Constantine et al., 2016]. However there are some challenges that come with the application of AS and there are limitations in the cases it can be used.

We will in this chapter give a mathematical background of the general theory behind AS 4.2, and then we'll examine its estimation with Monte Carlo methods in Section 4.3.

In Section 4.4 we will review select literature and give an example on how existing AS methods have been applied to MCMC with the aim to improve efficiency. The main part of the chapter is the description and results obtained by using two algorithms. The first

algorithm is from [Constantine et al., 2016], and is historically the first AS-based MCMC algorithm. The algorithm approximately integrates out inactive variables, estimating the marginal likelihood of the active variables. The algorithm is biased and does not target the intended posterior but rather a distribution close in Hellinger distance, we call this algorithm AS-MH-Bias.

In Section 4.7 we discuss some unresolved questions that we spotted in current AS, with a list of open points and some indications on possible mitigations.

In Section 4.8 we discuss one of the open points, i.e. the bias in AS-based MC algorithms, and we introduce the second AS-based MCMC algorithm [Schuster et al., 2017], that we name AS-MH. The AS-MH, like AS-MH-Bias produces an estimate of the marginal likelihood where inactive variables are approximately integrated out. But the AS-MH, unlike AS-MH-Bias, is unbiased and in stationarity samples from the desired posterior. We will outline some of the advantages and disadvantages of using either algorithm compared to standard MCMC, and show some results in a toy example in 4.8.4.

We conclude the chapter with Section 4.9 where we discuss another open point, namely that in some cases samples from the prior are used in order to build the Active Subspace structure, rather than posterior points [Constantine et al., 2016]. We discuss the issue in the section and propose a possible solution.

4.2 Active Subspaces: general Mathematical formulation

4.2.1 Structural matrix

As introduced, AS are a tool for dimension reduction [Constantine, 2015]. They are defined for scalar-valued, multivariate functions. In particular, given a function

$$f : \mathbb{R}^m \to \mathbb{R} \tag{4.1}$$

the active subspace can be found through eigenvalues analysis. Formally, by considering the following matrix [Constantine, 2015]

$$C = \int \nabla f(\theta) \nabla f(\theta)^T \rho(\theta) d\theta \tag{4.2}$$

with ρ a probability density function. By construction, the matrix C is symmetric and positive semi-definite, therefore the eigenvalues of C are guaranteed to be real and non-negative.

4.2.2 Estimating the dimension of Active Subspace through the spectral gap

Assuming the matrix C of (4.2) is $m \times m$, after sorting the eigenvalues λ_i in decreasing order, so that $\lambda_i \geq \lambda_{i+1}, i = 0, ...m-1$ we look for a *spectral gap*, i.e. a significant gap in the difference of two consecutive eigenvalues λ_i, λ_{i+1}, see for example Figure 4.1 which represents a system with $m = 4$ and the spectral gap is found between eigenvalue 2 and 3 where there is a difference of 5 orders of magnitude between them (note the y-axis is in log-scale)

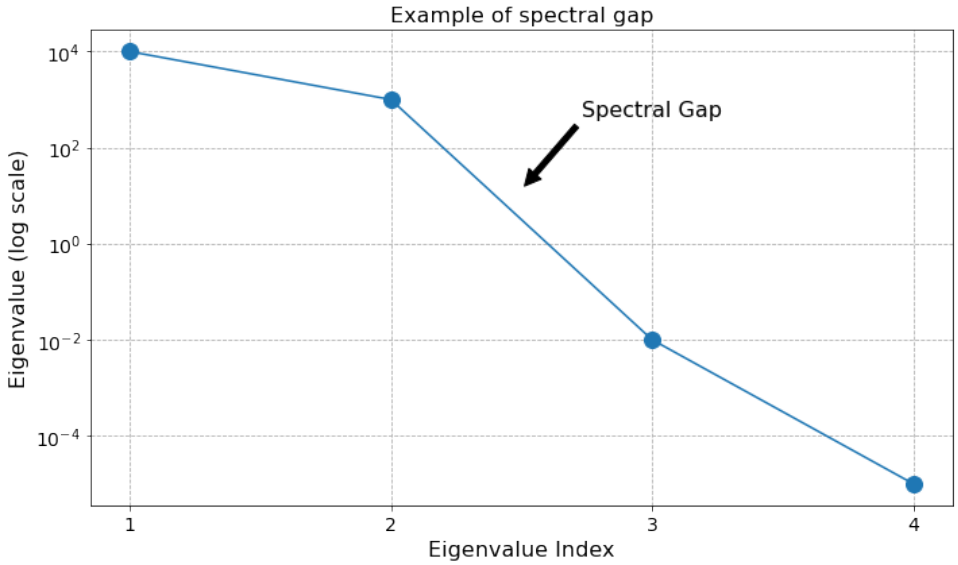

Figure 4.1: *Example of spectral gap, i.e. a significant difference between two consecutive eigenvalues arranged in descending order. In the figure the order of the system is 4, and the spectral gap is found between eigenvalue 2 and 3 since there is a difference of 5 orders of magnitude between them (note the y-axis is in log-scale).*

Say that the spectral gap is found at $n < m$ (n would be 2 in the example of Figure 4.1), that gives an indication that the dimension of the active subspace is n. Each eigenvalue of (4.2) is, in fact, "the average squared directional derivative of f along the corresponding eigenvector" [Constantine et al., 2016], and therefore a null eigenvalue will identify directions where f does not change.

4.2.3 Approximation of a function with the Active Subspace

Assuming the spectral gap obtained in Section 4.2.2 is at n, we partition the eigenvalues in two disjoint sets, one with $[\lambda_1, ..., \lambda_n]$, and the other with $[\lambda_{n+1}, ...\lambda_m]$, we can construct an orthonormal basis of $\mathcal{R}^{m \times m}$, using the corresponding eigenvalues as columns in the following matrix W

$$W = \begin{bmatrix} B_a & B_i \end{bmatrix} \tag{4.3}$$

where W is the orthonormal basis composed of eigenvectors, and B_a and B_i are the *active* and *inactive* matrices respectively, whose columns are composed by the subsets of eigenvectors corresponding to eigenvalues $\lambda_i \leq \lambda_n$ and $\lambda_i > \lambda_n$. With (4.3) we identify a partition of the space, so that

$$\theta = B_a a + B_i i \tag{4.4}$$

where a is the *active variable*, i the *inactive variable*. The variables a and i are defined once the matrices B_a and B_i are built with (4.3), since by construction B_a and B_i are orthonormal, we have that

$$a = B_a^T \theta \tag{4.5}$$

and

$$i = B_i^T \theta \tag{4.6}$$

It can be shown [Constantine, 2015] that the function f of (4.1) can be approximated with its conditional average given the active variables, we name the approximate conditional average g

$$f \approx g(B_a^T \theta) \tag{4.7}$$

Formally, the function $g : \mathbb{R}^n \to \mathbb{R}$ which approximates f is defined by

$$g(a) = \int f(B_a a + B_i i)\rho_i(i|a)di \tag{4.8}$$

with $\rho_i(i|a)$ the conditional density of i given a. As for the quality of the approximation, we have that the difference of $f(\theta)$ of (4.1) and $g(a)$ of (4.8) is bounded in Hellinger distance by a quantity that depends on the size of the "inactive" eigenvalues [Constantine et al., 2016], in the following way

$$\sqrt{\int (f(\theta) - g(B_a^T \theta))^2 \rho(\theta) d\theta} \leq C\sqrt{\lambda_{n+1} + ... + \lambda_m} \tag{4.9}$$

where in (4.9) we have used that since B_a and B_i are orthonormal, it is $a = B_a^T \theta$ (and similarly $i = B_i^T \theta$). The Hellinger distance is used in (4.9) as it provides an upper bound on the posterior mean and covariance [Constantine et al., 2016]. It is stated in (4.9) that, via some constant C, the size of the *inactive* eigenvalues bounds the Hellinger distance of the *Active Subspace* approximation $g(a)$ with the exact function $f(\theta)$ It is therefore important the eigenvalues after the spectral gap are as small as possible: in the ideal case where $\lambda_j = 0, j > n$, $g(a)$ will be an exact approximation of the function $f(\theta)$, in the other cases we have a biased approximation, controlled by the size of the $\lambda_j = 0, j > n$, via (4.9).

4.2.4 Active Subspaces directions by principal components

Finding an Active Subspace, as we have seen in the previous sections, corresponds to finding the eigenvalues of the structural matrix of (4.2), this corresponds, as we can appreciate by simply taking a look at the equation (4.2) itself, to performing the principal components analysis (PCA) of the uncentered covariance matrix of the gradient of the function (see also for example [Cui et al., 2019]), for points taken from a distribution ρ. In fact, the formula for the covariance used in the unscaled PCA for generic matrix data Z is as follows

$$\Sigma = \int \mathbf{Z}\mathbf{Z}^{\top} \rho(\theta) d\theta \tag{4.10}$$

And we can appreciate that equation (4.10) is the same as (4.2) when we substitute the generic X with the gradient data $\nabla f(\theta)$. Therefore we will, in the remainder, often make use of principal components as a tool to find the Active Subspace directions as an alternative to the traditional eigenvalue method described previously in Sections 4.2 and 4.2.3: the two methods are equivalent.

4.3 Active Subspaces using Monte Carlo approximations

Equations in Section 4.2 are given in integral form, we will see in this section how to extend them when using Monte Carlo approximations of equations, and an explanation of how the general AS theory extends to MC approximations is given in Section 4.3.4.

4.3.1 Monte Carlo approximation of the conditional expectation

Practical versions of the equation (4.8) will use the Monte Carlo estimate defined by

$$\hat{g}(a) = \frac{1}{M} \sum_{j=1}^{M} f(B_a a + B_i i_j), i_j \sim \rho_i(i|a) \tag{4.11}$$

4.3.2 Monte Carlo approximation of the structural matrix

Likewise, to obtain a Monte Carlo approximation of the matrix C of (4.2), we can draw independent samples from the distribution ρ and estimate the quantity in (4.2). The estimated matrix $\hat{C}$ is given by

$$\hat{C} = \frac{1}{N} \sum_{j=1}^{N} \nabla f(\theta_j) \nabla f(\theta_j)^T, \theta_j \sim \rho \tag{4.12}$$

where θ_j for $j = 1, \ldots, N$ are drawn independently from the the density ρ. Accordingly, matrix approximations $\hat{B}_a$ and $\hat{B}_i$ will be created from the eigenvectors of (4.12), similarly

to what has been shown in Section 4.2.3.

4.3.3 Monte Carlo approximation of the Active Subspace

Putting together the approximations (4.11) and (4.12), we obtain a Monte Carlo approximation of (4.8) that uses the matrices $\hat{B}_a$ and $\hat{B}_i$, we call it g_ϵ as per notation in [Constantine et al., 2016]

$$g_\epsilon(a) = \frac{1}{M}\sum_{j=1}^{M} f(\hat{B}_a a + \hat{B}_i i_j), i_j \sim \rho_i(i|a) \tag{4.13}$$

4.3.4 Validity of Monte Carlo approximations

It is shown in [Constantine et al., 2016] that similar bounds in Hellinger distance to (4.9) can be calculated for the approximations $\hat{g}(a)$ of (4.11) and $g_\epsilon(a)$ of (4.13), and in general for all MC approximations of integral quantities. The bounds will be dependent both on the size of the inactive eigenvalues, similar to the non-MC version (4.9), and from the quality of the respective Monte Carlo approximations (see for example formulae (2.16) and (2.17) in [Constantine et al., 2016]).

4.4 Active Subspaces: literary review on existing methods for MCMC

4.4.1 Introduction

This section has so far provided an introduction to the topic of AS and an explanation of the mathematics behind. As a summary, we have seen how a generic real-valued function f like in (4.1), once we have performed the *spectral gap* analysis explained in Section 4.2.2 can be approximated by its conditional average g of (4.8) where the inactive part is marginalised out, and the quality of the approximation is given in Hellinger distance by (4.9). We have seen the analysis both with the original integral formulae in Section 4.2, and with the practical MC approximations in Section 4.3. We will see now applications of AS to MCMC, historically the first MC method to have been considered.

Monte Carlo methods such as MCMC (described in Section 2.4), can enable the exploration of complex and intractable distributions. However, as the dimensionality of the state space increases, these methods face significant challenges that can affect their performance. We have mentioned in 4.1 that Active Subspaces (AS) have been introduced primarily as a method to mitigate the high-dimensional challenges in mathematical systems [Constantine, 2015]. We will in this section explore the application of AS to Monte Carlo methods, specifically to MCMC, in existing literature and compare them through results from application to a simple but illustrative toy model.

In Section 4.5, considering that AS was introduced in MCMC setting first [Constantine et al., 2016] in order to decrease the effect of high-dimensions, we will quantify the problems of standard MCMC in high dimensions and see that the complexity of the algorithm scales with d^2 with d the dimension of the state space.

In Section 4.6 we introduce historically the first AS algorithm to be used in MCMC [Constantine et al., 2016]. The algorithm is biased and targets a distribution that is not the intended posterior, but is only close in Hellinger distance. The closeness to the intended target distribution depends on the quality of the AS approximation.

In Section 4.8.2 we introduce a second AS-based MCMC algorithm, from [Schuster et al., 2017], that is unbiased and in addition it targets the intended posterior. It can therefore be considered a theoretical advancement from the biased algorithm mentioned earlier.

4.5 MCMC problems in high dimensions

Considering that AS was introduced in the MCMC setting [Constantine et al., 2016] primarily in order to decrease the effect of high-dimensions, we examine what is the effect that increasing dimensionality brings to MCMC. We will mention some studies performed in reference settings, and we will focus on one of the most used MCMC algorithms, Metropolis Hastings with random Walk (RWMH). In [Gelman et al., 1997, Roberts and Rosenthal, 2001] an analysis is provided of the optimal scaling of the proposal density parameters in the Metropolis-Hastings RWMH algorithm for achieving optimal performance and the behaviour when the dimension d of the state space becomes large is examined.

For convenience of the reader we report below in Algorithm 4 a standard generic version of MCMC MH

Alg. 4 Standard MCMC with Metropolis-Hastings

1: Input: posterior $\pi(\theta) \propto p(\theta)l(\theta)$ and proposal q
2: Initialize $\theta^{(0)}$
3: **for** $t = 1$ to T **do**
4: $\quad \theta^* \sim q(\theta^*|\theta^{(t-1)})$
5: $\quad$ Set $\theta^{(t)} = \theta^*$ with probability $1 \wedge \frac{p(\theta^*)\cdot l(\theta^*)\cdot q(\theta^{(t-1)}|\theta^*)}{p(\theta^{(t-1)})\cdot l(\theta^{(t-1)})\cdot q(\theta^*|\theta^{(t-1)})}$
6: $\quad$ Else let $\theta^{(t)} = \theta^{(t-1)}$
7: **end for**

For particularization of MH to RWMH with Gaussian proposal (which we have mostly used in this thesis), substitute the proposal on line 4 of Algorithm 4 with

$$\theta^* = \theta^{(t-1)} + \epsilon, \; \epsilon \sim \mathcal{N}(0, \sigma_q^2) \tag{4.14}$$

It is shown in [Gelman et al., 1997, Roberts and Rosenthal, 2001] firstly that (in reference settings having d independent components) with an optimal covariance of the proposal that

is proportional to $1/d$ it is possible to achieve a constant acceptance rate of 0.234. With this optimal proposal, as d grows, it requires a number of iterations that is linear in d to keep constant the mean squared error of an expectation of a function h

$$MSE = \mathbb{E}\left[\left(\frac{1}{n}\sum_{j=1}^{n} h(\theta_j) - \mathbb{E}(h)\right)^2\right] \quad j = 1, ...n \tag{4.15}$$

where in equation (4.15) $\frac{1}{n}\sum_{j=1}^{n} h(\theta_j)$ is the approximation of the expectation of h using MCMC samples θ_j, whereas $\mathbb{E}(h)$ is the true expectation value. In addition the operation of proposing and evaluating a new point $\theta^* = (\theta_1^*, \theta_2^*, \ldots, \theta_d^*)$ (lines 4 and 5 of Algorithm 4) with RWMH, has a cost of d, given by the d evaluations of the marginals $\pi(\theta^*) = \prod_j \pi_i(\theta_j^*)$ (with π the posterior and π_i its marginals) due to the independence of the components. Therefore the complexity of RWMH can be estimated in $O(d^2)$, which comes from the $O(d)$ iterations to keep the error constant, times $O(d)$ operations per iteration to evaluate each new proposed point.

4.6 Biased Active Subspace MCMC algorithm

The paper titled "Accelerating MCMC with Active Subspaces" [Constantine et al., 2016], focuses on the application of AS to improve the performance of MCMC, providing a way to mitigate dimensionality-related problem [Constantine, 2015, Constantine et al., 2016]. We recap that the central idea behind Active Subspaces is to be able to identify a part of the space, informed by the data, from its perpendicular subspace, which is, in the ideal case, not influenced at all by the data and therefore not influenced by the likelihood, and so this *inactive* subspace will only be informed by the prior [Constantine et al., 2016]. This separation will allow the AS-MCMC algorithms to operate on the active variables, i.e. on a subspace characterised by a reduced dimension, while the inactive variables can be sampled directly from the prior. When the AS is applied to MCMC, the function of interest to approximate, named f in equation (4.8) and subsequent, becomes the negative log-likelihood

$$f = -\log(l(\theta)) \tag{4.16}$$

Therefore this section will explain the particularization of AS in MCMC. We have seen how the use of gradients allows in Active Subspaces to determine the directions of maximum change of the negative log-likelihood, as we explained in Section 4.2.3, specifically by using equation (4.2). The integration of (4.2) is performed with respect to a distribution that can be for example prior or posterior (the use of prior or posterior for integration will be discussed in Section 4.9). The eigenvectors of equation (4.2) give the directions of the data-informed subspace, calculated using the gradient of the log-likelihood. We call **Active Subspace** the subspace generated by the eigenvectors associated to the biggest eigenvalues, these will

represent the directions of maximum variations of the likelihood: it makes sense that, in order to maximise efficiency, we look for an enhancement of the algorithms by concentrating the AS algorithmic power on the Active Subspace. The subspace of vectors which is orthogonal to the Active Subspace is informed by the prior mainly, and is called **inactive subspace**.

We report below the algorithms from [Constantine et al., 2016] which performs the MCMC on the Active Subspace. There is a main MCMC loop in Algorithm 6 which acts on the approximate active marginal created from the conditional expectation of equation (4.11): see row 6 of Algorithm 6 where the MH ratio shows the use of the estimate of the negative log-likelihood $\hat{g}_\epsilon(a)$ and so we have $\exp(-\hat{g}_\epsilon(a))$ that we use in place of the likelihood. Algorithm 5 is the algorithm that calculates the approximate conditional marginal $\hat{g}_\epsilon(a)$ of (4.11).

Alg. 5 Calculate the marginal $\hat{g}_\epsilon(a)$ of (4.11)

1: **function** ComputeMarginal(a, N_i, B_a, B_i, f)
2: $\quad \{i\}_{j=1}^{N_i} \sim q_i(\cdot|a)$.
3: $\quad$ Compute $\hat{g}_\epsilon(a) = \frac{1}{N_i} \sum_{n=1}^{N_i} f(B_a a + B_i i^n)$.
4: $\quad$ **return** $\hat{g}_\epsilon(a)$.
5: **end function**

Alg. 6 AS-MH-Bias

1: Compute the AS and using the procedure outlined in Section 4.2.3 estimate matrices B_a and B_i.
2: Initialize the algorithm by choosing an initial value a^1 and use Algorithm 5 to estimate $\hat{g}_\epsilon(a^1)$.
3: **for** $k = 2$ to T **do**
4: $\quad a^* \sim q_a(\cdot|a^{k-1})$.
5: $\quad$ Use Algorithm 5 to compute $\hat{g}_\epsilon(a^*)$
6: $\quad$ Set $a^k = a^*$ with probability $1 \wedge \frac{p(a^*)\exp(-\hat{g}_\epsilon(a^*))q_a(a^*|a^{t-1})}{p(a^{k-1})\exp(-\hat{g}_\epsilon(a^{k-1}))q_a(a^{t-1}|a^*)}$
7: $\quad$ Else let $a^k = a^{k-1}$.
8: **end for**
9: To map back to the original space, for each $k = 1, ...T$ draw $\{i_j^k\}_{j=1}^{N_i} \sim q_i(\cdot|a^k)$ and compute $\theta_j^k = B_a a^k + B_i i_j^k$.

As we mentioned already in the general AS Section 4.2.3 equation (4.9), it is to be noted that Algorithm 6 **does not target the exact posterior**, it provides MCMC samples from a distribution close in Hellinger distance from the intended posterior, and the quality of the approximation depends on the size of the "inactive" eigenvalues, as per formula (4.9): the smaller the inactive eigenvalues, the better the approximation. In addition Algorithm 6 is also **biased** as we explain in more detail in Section 4.8.3 (see also the note in section 4 of [Schuster et al., 2017]).

4.7 Discussion on open points in current AS methods

Now that we have introduced the concept of AS and the first AS-based MCMC algorithm AS-MH-Bias (algorithm 6), we report here a list of some Open Points (OP), which identify some unresolved questions in current literature.

OP1: Bias

The AS-MH-Bias algorithm is biased and in stationarity samples from a distribution that is only close in Hellinger distance to the intended posterior [Constantine et al., 2016]. This problem is acknowledged in the paper [Constantine et al., 2016] itself when the authors have to revert to some empirical methods to understand if the samples collected through the AS-MH-Bias are close enough to the intended posterior.

OP2: Noisiness of likelihood estimate

We have seen that in AS-based algorithms in MCMC we may use estimates of the marginal likelihood, obtained by approximately marginalising out inactive (or in some cases active) variables, one such example of approximate marginal likelihood is Algorithm 5. In some cases the estimates that we obtain may be very noisy, to the point that the benefit of using AS becomes questionable.

OP3: weaknesses of MCMC

The current AS Monte Carlo applications mainly employ marginalisation of the inactive variables and perform MCMC on the active part [Constantine et al., 2016, Schuster et al., 2017]: if the Active Subspace marginal is complex (for instance high-dimensional or multi-modal), the MCMC may not perform well.

OP4: Prior vs Posterior

Some decisions on the structure of the AS, needs to be done preliminarily by using information from the Bayesian prior and the resulting structure is maintained throughout the MC algorithm, but ideally the structure should be defined on the posterior, and posterior-defined may be quite different from prior-defined.

OP5: Dimensionality

Although AS aims to decrease the impact of the *curse of dimensionality* some AS-algorithms may still suffer from it and overall risk to produce a worse outcome than the non-AS counterparts, this happens for example when methods like Importance Sampling are used to create likelihood marginal estimates.

OP6: Linearity

Existing literature focuses mainly on partition of the state space in active/inactive through linearization, but real-world cases are likely to require non-linear transformations.

4.7.1 Open points mitigations

In Section 4.8 we will discuss the bias in AS, which is open point **OP1**, and we will introduce the second AS-based MCMC algorithm, which we name AS-MH [Constantine et al., 2016]. We will see that AS-MH uses unbiased marginals and in stationarity targets the intended posterior. In addition, in Chapters 5 and 6 we'll introduce additional MC applications to AS, with algorithms that build on the foundations of AS-MH, and are likewise unbiased and target the intended posterior. In Chapter 5 we introduce additional applications in AS to MCMC, with algorithms that are based on Gibbs sampling [Geman and Geman, 1984], PMMH and MwPG [Andrieu et al., 2010] and aim to address a few of the *Open Points*, and we named the algorithms: AS-Gibbs is discussed in Section 5.6, AS-PMMH in Section 5.2, AS-MwPG in Section 5.7. For AS-PMMH and AS-MwPG we also devised a second version, named AS-PMMH-i (Section 5.5) and AS-MwPG-i (Section 5.7), where the additional i in the acronym stands for *inverted* and means that we switch methods applied to active and inactive variables.

In Chapter 6 we focus on the use of AS on SMC-based algorithms: so while Chapter 5 presents methods where the structure has an outer MCMC, in Chapter 6 the structure has an outer SMC sampler. The first algorithm to be presented is a SMC version of the AS-MH, where the outer MCMC has been replaced by a SMC sampler, we call the new algorithm AS-SMC. We then introduce AS-SMC^2, based on the SMC^2 algorithm of [Chopin et al., 2012]. When comparing AS-SMC and AS-SMC^2 performances to the standard SMC, we see that AS-SMC seems to outperform standard SMC in cases of perfect Active Subspaces, whereas in non-perfect Active Subspaces the case to use either AS-SMC or AS-SMC^2 becomes less clear, as the additional algorithmic complexity brings a *trade-off* and probably further study on applications are required. We finally introduce an adaptive version of AS-SMC in Section 6.4, which we name AS-SMC-a, which has the addition of re-calculating the structure of the AS at each tempering step of the SMC sampler, ensuring a more reliable reproduction of the directions of the posterior Active Subspaces (in existing literature traditionally algorithms rely on Active Subspaces directions determined by using the prior [Constantine et al., 2016, Schuster et al., 2017]).

As we understand from the above description, Chapters 5 and 6, in addition to the bias in AS, which is **OP1**, we will also address the case of noisy estimates of the likelihood, which is **OP2**, and will in fact see examples of how a noisy marginal estimate can cause problems, see for example Figures 5.10 and 5.11 in Section 5.4 on the "sticky" behaviour of algorithms when the marginal estimate is noisy.

Also open point **OP3**, on the weaknesses of MCMC, i.e. in cases when the Active Subspace is complex to explore and the MCMC may encounter problems in navigating the active marginal, will be addressed in Chapter 6, in fact we will introduce algorithms where the "backbone" is given by an SMC sampler, not MCMC.

Additionally, in Sections 5 and 6 we will address **OP5**, on possible dimensionality problems faced by some AS algorithms, for example by the AS-MH that, by using Importance Sampling (IS) on the inactive subspace, may suffer in high dimensions [Agapiou et al., 2017], in fact we will be introducing algorithms where the marginal estimate is produced not by IS but, for example, by a SMC sampler.

In Section 4.9 we discuss the **OP4** and we present a toy model which highlights the problems of using prior samples to build the matrices B_a and B_i, instead of the posterior samples. We then give a possible solution to the issue in Section 6.4, with AS-SMC-a which does not need prior samples to build the AS structure, but rather adaptively builds the structure at each step of a SMC algorithm.

We have not analysed **OP6** which deals with linearity of the transformation (4.4) in the present thesis. But considering that real-world scenarios are likely to require greater complexity of the transformation, future work should be foreseen in that direction.

4.8 Bias in Active Subspaces MCMC

4.8.1 Introduction

The issue **OP1** in the open points list in 4.7, is about the biasedness and non exactness of Algorithm 6. Algorithm 6 is, historically, the first AS-enhanced MCMC algorithm introduced by [Constantine et al., 2016]. In general, a lot of existing scientific literature and works on **Active Subspaces** are based on [Constantine, 2015] and, for Monte Carlo (specifically MCMC) in [Constantine et al., 2016]. Both papers can be said to be foundational, precisely because they made the way to a lot of subsequent studies (for example just to mention two amongst many [Cui et al., 2019] and [Parente, 2020]).

The theoretical distinction between *active* and *inactive* subspaces in the original formulation [Constantine, 2015, Constantine et al., 2016] is clear and we have explained in the previous sections (for example 4.1 and 4.6). However, its practical application can be quite complicated, and this is mainly due to the Hellinger distance bound between the true and approximated posteriors (4.9): in many real-world scenarios, the so-called 'inactive' variables might have even small influences of the likelihood, and in these cases there is no clear separation between active and inactive variables and the bound (4.9) may become too loose, due to the lack of spectral gap or to the size of inactive eigenvalues being not small.

The paper [Constantine et al., 2016] itself gives a clue of the problems, in fact the two examples cited in the paper show firstly a toy 2D example (the same that we will discuss

in an application in Section 4.8.4) and then a more real-world 100D problem. While the 2D example is created *ad-hoc* with a perfect scenario having the inactive variables that by construction do not influence the likelihood, allowing for a direct and smooth application of Active Subspace theory, the 100D example highlights the challenges. The application of theory becomes less straightforward in this case. Although a spectral gap is found, the inactive eigenvalues are not small in size, and the authors have to acknowledge this and have to resort to empirical methods to assess convergence, stressing the need of hands-on validations because the theoretical part becomes weak [Constantine et al., 2016].

4.8.2 Exact Active Subspace MCMC algorithm: AS-MH

We introduce in this section the second AS-MCMC algorithm in existing literature. The paper "Exact active subspace Metropolis-Hastings, with applications to the Lorenz-96 system" [Schuster et al., 2017] shows an application of AS to MCMC. In this the aim is similar to what discussed in Section 4.6. In fact the authors start by discussing [Constantine et al., 2016] of Section 4.6, and demonstrate that the formulation of the algorithm used in [Constantine et al., 2016], which as we said targets $g_\epsilon(a)$ of (4.13), is biased (as we also remarked in Section 4.6), and in addition, and this comes from using the method of Section 4.2.3, it targets a posterior that is only close by some distance to the intended posterior. In [Schuster et al., 2017] pseudo-marginals [Beaumont, 2003, Andrieu and Roberts, 2009] are used to create an unbiased estimate of the likelihood (as we have shown in Section 2.7). We see below the main idea: starting from the posterior

$$\pi(a, i) \propto p(a, i)l(a, i) \tag{4.17}$$

we look for the active marginal

$$l_a(a) = \int p(B_a a + B_i i)l(B_a a + B_i i)di = \int p(a, i)l(a, i)di \tag{4.18}$$

then we make use of Bayes' theorem for the prior

$$p(a, i) = p_i(i|a)p_a(a) \tag{4.19}$$

where in equation (4.19) p_a is the marginal prior for active variables, and p_i is the conditional prior of inactive given the active. Substituting equation (4.19) in (4.18)

$$l_a(a) = p_a(a) \int p_i(i|a)l(a, i)di \tag{4.20}$$

The final step is to approximate (4.20) with the Monte Carlo integration performed with Importance Sampling via pseudo-marginal [Beaumont, 2003, Andrieu and Roberts, 2009]:

$$\hat{l}_a(a) = p_a(a)\frac{1}{N_i}\sum_{n=1}^{N_i}\frac{p_i(i|a)l(a,i)}{q_i(i)} \tag{4.21}$$

where, in (4.21), N_i is the number of samples of inactive variables chosen per each active variable, and q_i is an Importance proposal. Compare the estimate (4.20) with (4.8) of the non-exact and biased version, and also the Monte Carlo approximations (4.21) with (4.11) to appreciate the conceptual differences.

By looking at Importance Sampling in (4.21), and remembering from Section 2.3 that the ideal importance proposal is proportional to the posterior, if the likelihood remains constant on the inactive subspace (as in the case of perfect Active Subspace), it is clear that an ideal proposal of the inactive variables is the conditional prior of inactive given the active

$$q_i(i) = p_i(i|a) \tag{4.22}$$

Alg. 7 Calculate the pseudo-marginal $\hat{l}_a(a)$ of (4.21)

---

function ComputePseudoMarginal(a, N_i, B_a, B_i)
  for $n = 1 : N_i$ **do**
    $i^n \sim q_i(\cdot \mid a)$;
    $$\tilde{w}^n = \frac{p_i(i^n \mid a)\, l(B_a a + B_i i^n)}{q_i(i^n \mid a)};$$
  end for
  $u \sim \mathcal{M}\left(\left(w^1, ..., w^{N_i}\right)\right)$, where for $n = 1 : N_i$
    $$w^n = \frac{\tilde{w}^n}{\sum_{p=1}^{N_i}\tilde{w}^p};$$
  Set $\hat{l}_a(a) = \frac{1}{N_i}\sum_{n=1}^{N_i}\tilde{w}^n$
  return $\hat{l}_a(a), i^u, \{i\}_{j=1}^{N_i}$.
end function

Alg. 8 AS-MH

1: Compute the AS and using the procedure outlined in Section 4.2.3 estimate matrices B_a and B_i.
2: Initialize the algorithm by choosing an initial value a^1 and use Algorithm 7 to estimate $\hat{l}_a(a^1)$.
3: **for** $k = 2$ to N_a **do**
4: $a^* \sim q_a(\cdot|a^{k-1})$.
5: Use Algorithm 7 to get $\hat{l}_a(a^*)$, i^{*u} and $\{i^*\}_{j=1}^{N_i}$
6: Set $a^k = a^*$ and $\theta^k = a^* + i^{*u}$ with probability $1 \wedge \frac{p_a(a^*)\hat{l}_a(a^*)q_a(a^*|a^{t-1})}{p_a(a^{k-1})\hat{l}_a(a^{k-1})q_a(a^{t-1}|a^*)}$
7: Else let $a^k = a^{k-1}$ and $\theta^k = \theta^{k-1}$.
8: **end for**

We see that Algorithm 8 generates one i point for each a point, the index u which is drawn on line 6 of the auxiliary pseudo-marginal Algorithm 7 ensures that one inactive particle is drawn according to the importance weights. There is a way to reuse all the inactive points, following [Andrieu et al., 2010]. In fact the output from Algorithm 8 may be used to estimate an integral

$$\mathbb{E}_\pi \left[h\left(\theta\right) \right] = \int_\theta h\left(\theta\right) \pi\left(\theta\right) d\theta$$

with respect to the posterior distribution π, for some function h in two possible ways [Andrieu et al., 2010].

1. **Using one i-point for each a-point** (the i-point which is drawn, for each active particle, using the u index, based on weights on line 6 of the auxiliary pseudo-marginal Algorithm 7):

$$\hat{\mathbb{E}}_\pi \left[h\left(\theta\right) \right]_1 = \frac{1}{N_a} \sum_{k=0}^{N_a} h\left(B_a a^k + B_i i^{u^k,k} \right) \tag{4.23}$$

2. **Using all of the accepted i-points for each a-point:**

$$\hat{\mathbb{E}}_\pi \left[h\left(\theta\right) \right]_2 = \frac{1}{N_a} \sum_{k=0}^{N_a} \sum_{n=1}^{N_i} w^{n,k} h\left(B_a a^k + B_i i^{n,k} \right). \tag{4.24}$$

AS-MH: simplification in cases of perfect Active Subspaces

In case of perfect Active Subspaces the AS-MH Algorithm 8 simplifies. Let's remember that, in case of perfect Active Subspaces, the inactive variables do not affect at all the likelihood. In this case, if we use the prior as a proposal for inactive variables, we see that the estimate of the likelihood of equation (4.21) can be simplified as

$$\hat{l}_a(a) = p_a(a) \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_i(i|a) l(a,i)}{q_i(i)} = p_a(a) l(a) \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_i(i)}{p_i(i)} = p_a(a) l(a) \tag{4.25}$$

where, in equation (4.25) we have used that $p_i(i|a) = p_i(i)$ considering that in case of perfect Active Subspaces, active and inactive components are independent, and we have also used that since the likelihood is not affected by the inactive variables, we have that $l(a, i) = l(a)$. So in case of perfectly inactive variables the integral itself of the active marginal can be calculated exactly, and in this case Algorithm 8 simplifies in Algorithm 21, reported in Appendix C, which becomes a MCMC targeting the exact marginal posterior $\pi_a = p_a l_a$.

4.8.3 Biasedness or unbiasedness of AS-MCMC algorithms: Mathematics behind

We will explain in this section why **AS-MH** algorithm 8 is unbiased, whereas **AS-MH-Bias** algorithm 6 possesses asymptotic bias.

In Algorithm 8 the values of the pseudo-marginal $\hat{l}_a(a^*)$ from line 5, if accepted on line 6, are stored and reused in the following iteration on the denominator of the MH in line 6: algorithm 8 is part of a class of algorithm named "grouped independence Metropolis Hastings" (GIMH) in [Beaumont, 2003, Andrieu and Roberts, 2009], in such case it is proven that if the estimate of the likelihood $\hat{l}_a(a)$ coming from Algorithm 7 is unbiased (plus some additional relatively mild conditions), then Algorithm 8 is unbiased and targets the intended posterior (so even if instead of the likelihood we are using an estimate with a pseudo-marginal, the MCMC algorithm is still converging to the correct posterior).

On the other hand, algorithm 6 produces new estimates of the quantity $\hat{g}_\epsilon(a)$ both for the new value (numerator) and the old value (denominator) of the MH on line 6 (for further explanations see sections 3 and 4 of [Schuster et al., 2017]), so Algorithm 6 is part of a class of algorithms named "Monte Carlo within Metropolis" (MCwM) in [Beaumont, 2003, Andrieu and Roberts, 2009], and as a result Algorithm 6 is likely to be biased. In addition to being biased, we have to remember that in the original formulation seen in equation 4.2.3, equation (4.9), algorithm 6 will also produce MCMC samples from a distribution only close in Hellinger distance to the intended posterior.

In conclusion, we can see the AS-MH Algorithm 8 from [Schuster et al., 2017] as a theoretical advancement to AS-MH-Bias Algorithm 6 from [Constantine et al., 2016], considering that it provides an unbiased formulation of the AS algorithm, targeting the correct posterior and therefore giving a possible solution to the issue of biasedness **OP1** in the open points list in 4.7.

4.8.4 Comparison of results: standard MCMC vs biased AS MCMC vs exact AS MCMC

Introduction

We'll demonstrate in this section some results on AS MCMC algorithms:

- The standard MCMC Algorithm 4
- The AS-MH-Bias Algorithm 6 [Constantine et al., 2016]
- The AS-MH Algorithm 8 [Schuster et al., 2017]

We'll do the comparisons using a simple but illustrative toy example from [Constantine et al., 2016]. For generality on Bayesian inverse models applied in the case of Active Subspaces, please refer to Appendix B. We use here a two-parameter quadratic model: $\theta = [\theta_1,\ \theta_2]^T$, in the equation below the negative log-likelihood f of equation (4.16), is

$$f(\theta) = \frac{1}{2}\theta^T \mathbf{A}\theta, \quad \mathbf{A} = \mathbf{Q}\begin{bmatrix} 1 & \\ & \epsilon \end{bmatrix}\mathbf{Q}^T, \quad \mathbf{Q} = \frac{1}{2}\begin{bmatrix} \sqrt{2} & \sqrt{2} \\ -\sqrt{2} & \sqrt{2} \end{bmatrix} \tag{4.26}$$

The noise parameter in the likelihood is $\sigma = .1$, and it is assumed a single data point $\mathbf{d} = .9$ [Constantine et al., 2016] (for the meaning of d and σ see equation (B.1) and subsequent). The parameter ϵ of (4.26) can be tuned to determine the characteristics of the system: in [Constantine et al., 2016] the value $\epsilon = 0.01$ is used. We can see from the pictures below, for example Figure 4.2 and 4.4, the effect of choosing progressively higher values of ϵ: in Figure 4.2, with $\epsilon = 0.01$, we see a spectral gap (differences between the two eigenvalues) of 4 orders of magnitude, in Figure 4.4, with$\epsilon = 0.2$, we see a spectral gap of 2 orders of magnitude and we can also notice that the smaller eigenvalue is bigger than 100: it is worth mentioning again that the upper bound formulae (4.9) and similar, indicate that the spectral gap is an important parameter, and also the size of *inactive* eigenvalues is important, in fact the Active Subspace will be closer to ideal the more the inactive eigenvalues are close to zero, whereas for relatively big inactive eigenvalues the exsistence of Active Subspaces is more uncertain, as the upper bound in Hellinger distance becomes too loose. We can notice in fact that in the case $\epsilon = 0.01$ where the inactive eigenvalue has order of magnitude roughly 0, the AS approximation of Figure 4.3 is noticeably better than the case of $\epsilon = 0.2$, see Figure 4.5, where the inactive eigenvalue has order of magnitude 2.

case $\epsilon = 0.01$

The case of model of equation (4.26) when using $\epsilon = 0.01$ is ideal for Active Subspaces, and even in the traditional biased and non-exact formulation of [Constantine et al., 2016] will give near-exact results: in fact, firstly there is a spectral gap of nearly 4 orders of magnitude, as we see from Figure 4.2, and in addition the Hellinger bound of equation (4.9) is dominated by a number that is relatively small, close to 0 (the size of the smallest eigenvalue).

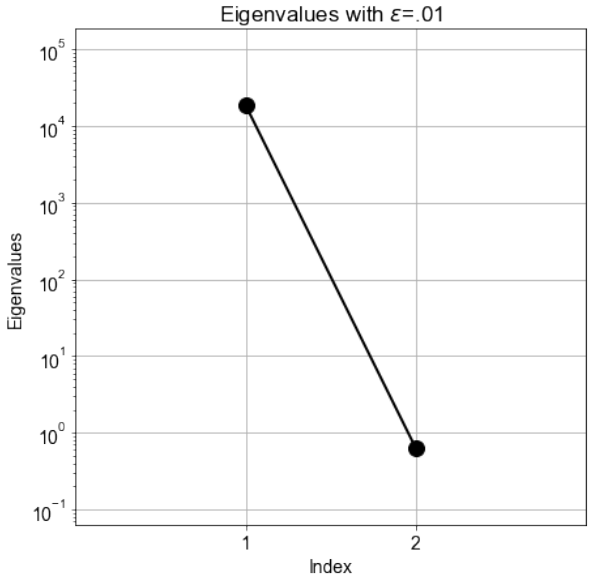

Figure 4.2: *Spectral gap for the model of* (4.26) *with* $\epsilon = 0.01$*: the eigenvalues difffer by 4 orders of magnitude and the inactive eigenvalue* 2 *is less than* 1

	Std MCMC	AS-MH-Bias	AS-MH
MSE Mean (θ_1)	0.07	0.38	0.09
MSE Mean (θ_2)	0.07	0.38	0.09
MSE SD (θ_{11})	0.01	0.18	0.01
MSE SD (θ_{22})	0.02	0.18	0.01
MSE SD (θ_{12})	0.16	0.58	0.17

Table 4.1: Comparison of MSEs for the three MCMC methods discussed in the paragraph, using ϵ=0.01

We see in table 4.1 results obtained by averaging 100 runs each of standard MCMC, AS-MH-Bias and AS-MH on the model (4.26) and calculating the mean squared error (MSE). We see that MCMC and AS-MH have comparable errors, and AS-MH has the advantage of performing MCMC on a space having half dimension. We report below charts of the posterior marginals composed by the active component $a = B_a^T\theta$ and inactive $i = B_i^T\theta$ (both red-dotted) versus the prior components θ_1 and θ_2 (continuous line in both cases) in chart 4.3

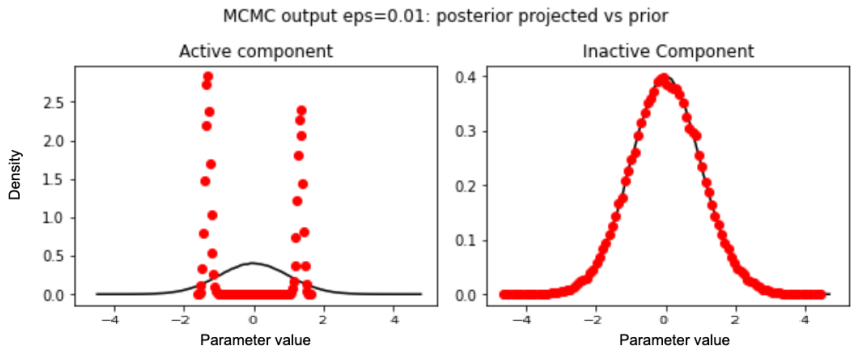

Figure 4.3: *Posterior active and inactive marginals versus prior. Active marginal* $a = B_a^T\theta$ *in LHS and inactive marginal* $i = B_i^T\theta$ *in RHS (both red-dotted) versus* ***prior*** *components* 1 *and* 2 *(continuous line) of* (4.26) *with* $\epsilon = 0.01$*: we see that the inactive marginal RHS is almost identical to the prior second component, indicating that we are in near-perfect Active Subspace and the likelihood is very little informative on this component. See comparison with LHS chart where the active marginal is very different from the first component of the prior indicating that the first component is active and the differences are due to the effect of the likelihood.*

case $\epsilon = 0.2$

The second case of equation (4.26) when using $\epsilon = 0.2$ is less ideal for the traditional biased and non-exact algorithm of Active Subspaces than the one discussed in Section 4.8.4: in fact, firstly there is a spectral gap of nearly 2 orders of magnitude (compared to 4 of Section 4.8.4), as we see from Figure 4.4, but the real weak point is that the Hellinger bound of equation (4.9) is too loose, in fact it is dominated by a number that is around 200 (the size of the smallest eigenvalue). See for example the difference in the chart right hand side of Figures 4.5 (where we can spot differences between the red-dotted projected posterior and the black solid prior) with the same item in Figure 4.3 (where the dotted-red and black solid shapes are nearly identical).

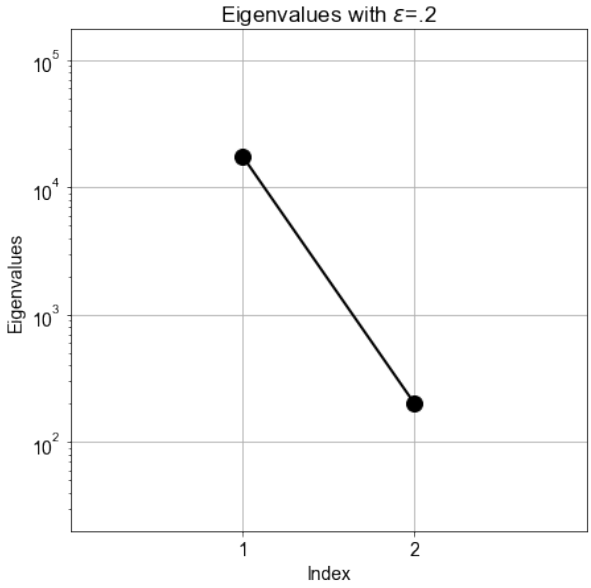

Figure 4.4: *Spectral gap for the model of* (4.26) *with* $\epsilon = 0.2$*: the eigenvalues difffer by* 2 *orders of magnitude and the smallest eigenvalue is around* 200*.*

	Std MCMC	AS-MH-Bias	AS-MH
MSE Mean (θ_1)	0.27	0.68	0.36
MSE Mean (θ_2)	0.27	0.68	0.35
MSE SD (θ_{11})	0.06	0.28	0.08
MSE SD (θ_{22})	0.05	0.29	0.09
MSE SD (θ_{12})	0.15	0.82	0.27

Table 4.2: Comparison of MSEs for the three MCMC methods discussed in the paragraph, using ϵ=0.2

We see in table 4.2 results obtained by averaging 100 runs each of standard MCMC, AS-MH-Bias and AS-MH on the model (4.26) and calculating the mean squared error (MSE). As the approximation of the AS becomes worse with $\epsilon = 0.2$, we see that AS-MH performance worsen compared to standard MCMC (see table 4.1 for comparison). We report below charts of the posterior marginals composed by the active component $a = B_a^T\theta$ and inactive $i = B_i^T\theta$ (both red-dotted) versus the prior components θ_1 and θ_2 (continuous line in both cases) in chart 4.5: we see that, compared to the case $\epsilon = 0.01$ of Figure 4.3, the likelihood is slightly more informative in the RHS part of the graph and there is a worse fit

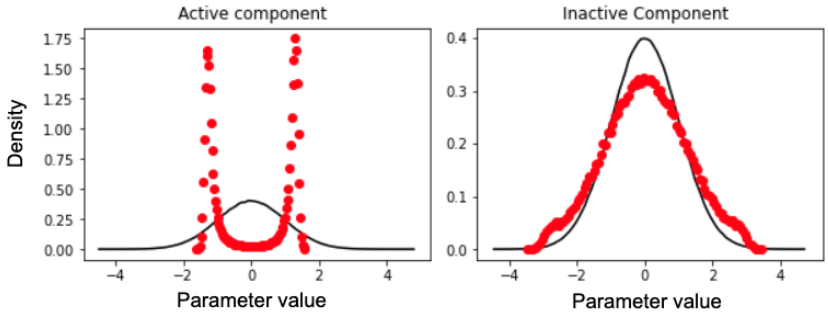

Figure 4.5: *Posterior active and inactive marginals versus prior. Active marginal* $a = B_a^T\theta$ *in LHS and inactive marginal* $i = B_i^T\theta$ *in RHS (both red-dotted) versus* ***prior*** *components* 1 *and* 2 *(continuous line) of* (4.26) *with* $\epsilon = 0.2$*: we see that the inactive marginal RHS is very similar to the prior second component, although, compared to Figure 4.3 RHS (case* $\epsilon = 0.01$*), we can see some differences: in the current case the fit is not perfect, indicating that the likelihood is, although slightly, more informative in this case.*

4.8.5 Conclusion

The toy example shown in Section 4.8.4 is simple but illustrative. We have seen how both cases of $\epsilon = 0.01$ and $\epsilon = 0.2$ are *fairly* good candidates for using AS, since both have a significant *spectral gap*, which is one of the pre-requirements [Constantine, 2015, Constantine et al., 2016]. But there is a significant problem, among the others, that is clear in the example: while in the case $\epsilon = 0.001$ we can see from Figure 4.2 that the inactive eigenvalue is *fairly* close to 0 making it a close-to-ideal case, in the second example $\epsilon = 0.2$ we see from Figure 4.4 that, although there is order of magnitude 2 between the two eigenvalues, which can be considered fairly good as a spectral gap, the inactive eigenvalue is bigger in size than 100: not being 0 or even close to 0 forces the users of the biased Algorithm 6 (second column results in Table 4.2) to have to use additional empirical methods to check the quality of approximation. This in particular refers to OP1 in the list of Section 4.7.

4.9 Prior AS vs Posterior AS

This section will discuss **OP4** of the *open points* list of section.4.7. We have explained in Chapter 4 the formulae for the Active Subspaces, and we have seen both the integral formulae in Section 4.2 and the Monte Carlo approximations in Section 4.3. If, as an example, we take into consideration equation (4.12) (which is the MC version of (4.2)), it requires samples to be drawn from a distribution ρ that is supposed to be the posterior. The AS MCMC algorithms all require that the structural matrices B_a and B_i of equation (4.3) are set *before* the start of the algorithm. Therefore the problem is clear: we can only have AS-MCMC samples that approximate the posterior *after* having run Algorithm 6 or 8, but *before* running the algorithms we need the posterior samples to build the matrices B_a and B_i. The authors of

[Constantine et al., 2016] come up with a solution: when it is not possible or convenient to draw from the true posterior π in (4.2) or its MC version (4.12), approximations of matrices C and $\hat{C}$ can be built drawing samples from the prior distribution instead of the full posterior π. If we use the familiar notation

$$\pi(\theta) \propto p(\theta) l(\theta) \tag{4.27}$$

where p is the prior and l is the likelihood, we can then build a new variant of equation (4.12), as follows

$$\hat{C}_{pri} = \sum_{i=1}^{N} \nabla f(\theta_i) \nabla f(\theta_i)^T, \theta_i \sim p \tag{4.28}$$

The matrix $\hat{C}_{pri}$ is built approximating integration against the prior p and not the full posterior π. Drawing from the prior is assumed to be easy and will bring an approximation which is again bounded in Hellinger distance, and is quantified in [Constantine et al., 2016] (see equations (3.9) and subsequent in [Constantine et al., 2016]). But there are cases where the Active Subspace generated by the prior, that we have called **prior Active Subspace** may be different from that generated by the posterior, the **posterior Active Subspace**, in such cases the approximation obtained using the prior will be poor. We show one example of significant differences between the **prior Active Subspace** and the **posterior Active Subspace** in the following Section 4.9.1.

4.9.1 Toy example to demonstrate prior vs posterior Active Subspace differences

Let's consider a model with a likelihood of the form, for $\theta \in \mathbb{R}^d$:

$$l(\theta) \propto \prod_{j=1}^{d} \left[\frac{\exp\left(-\frac{\theta_j^2}{\sigma_j^2}\right)}{1 + \left(\frac{\theta_j}{\gamma_j}\right)^2} \right]. \tag{4.29}$$

The gradient of the log-likelihood of equation (4.29) is:

$$\nabla \log l(\theta) = \nabla \sum_{j=1}^{d} \left[-\frac{\theta_j^2}{\sigma_j^2} - \log\left(1 + \left(\frac{\theta_j}{\gamma_j}\right)^2\right)\right] \tag{4.30}$$

$$= \nabla \log \prod_{j=1}^{d} \left[\frac{\exp\left(-\frac{\theta_j^2}{\sigma_j^2}\right)}{1 + \left(\frac{\theta_j}{\gamma_j}\right)^2}\right] \tag{4.31}$$

$$= \nabla \sum_{j=1}^{d} \left[-\frac{\theta_j^2}{\sigma_j^2} - \log\left(1 + \left(\frac{\theta_j}{\gamma_j}\right)^2\right)\right] \tag{4.32}$$

$$= \begin{bmatrix} -2\theta_1 \left(\frac{1}{\sigma_1^2} + \frac{1}{\theta_1^2+\gamma_1^2}\right) \\ \vdots \\ -2\theta_d \left(\frac{1}{\sigma_d^2} + \frac{1}{\theta_d^2+\gamma_d^2}\right) \end{bmatrix}. \tag{4.33}$$

We see from the expression of the likelihood in equation (4.29) that it is composed of two terms: the Gaussian-like term given by the exponential at the numerator, and the Cauchy-like term given at the denominator.
If we fix the dimension $d = 2$ so that $\theta \in \mathbb{R}^2$ and we choose a prior p of the form

$$p(\theta) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 5000 & 0 \\ 0 & 5000 \end{bmatrix}\right) \tag{4.34}$$

We will therefore have a posterior in $2D$

$$\rho(\theta) = p(\theta) l(\theta) \tag{4.35}$$

And we then set, as an example, the following values for the parameters in 4.29

$$\begin{aligned} \sigma_1 &= 10 \\ \gamma_1 &= 10^{12} \\ \sigma_2 &= 50 \\ \gamma_2 &= 0.1 \end{aligned} \tag{4.36}$$

The behaviour of the likelihood will be different in regions where θ is close to 0, where the Cauchy term will dominate, whereas far from the origin the Gaussian term will be predominant.

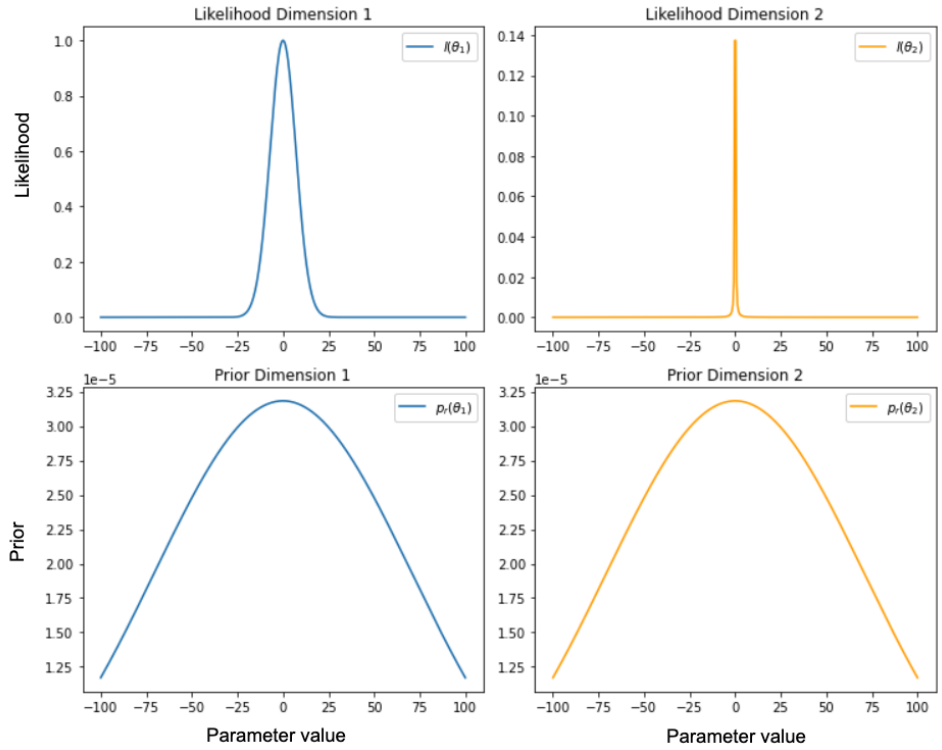

Figure 4.6: *Charts representing prior (bottom two charts) and likelihood (upper two) of the posterior* (4.35)*, with parameter values* (4.36)*. Note: "Dimension 1" in the chart titles is the first component* θ_1 *and "Dimension 2" is* θ_2*.*

The prior (4.34), as we can appreciate in Figure 4.6 in the bottom two charts, is very wide relative to the likelihood, therefore the posterior behaviour of (4.35) will be constrained by the likelihood, which is more narrowly condensed around 0, as we can appreciate by looking at Figure 4.6 in the top two charts.

Firstly, we can ask ourselves, by just having a look at the charts in 4.6, which one will be, if any, the predominant direction in the Active Subspaces. Recalling what we said in Section 4.8.2, that when the variables are inactive the likelihood is little or not informative at all and the prior is a good Importance Sampling proposal for the Monte Carlo integration of the LHS of (4.21), we see from Figure 4.6 that clearly the first dimension (top and bottom charts left hand side) will be inactive, whereas the second dimension (top and bottom charts right hand side), will be the active variable, as we can appreciate from the very small variance of the likelihood on the second dimension (top chart right hand side).

To prove even further that θ_2 would be the right choice as direction of the Active Subspace, we can also use the ESS, in fact by drawing 1000 importance points from the prior and measuring the ESS in each of the two dimensions, we have the following results

ESS θ_1	**ESS** θ_2
140	5

Table 4.3: ESS using prior as importance proposal in the system of Figure 4.6, we see clearly that the first variable θ_1 is inactive since the high ESS shows that it is little informed by the likelihood, compared to the very low ESS of θ_2

We see from Table 4.3 that, as we would expect from the visual inspection of Figure 4.6, the first state space dimension θ_1 is little informed by the likelihood, and therefore, by definition, inactive: this is shown by the relatively high ESS, which shows that the prior is a good Importance Sampling proposal for the posterior on that dimension. Compare with the very low level of ESS on θ_2, and we can see that on the second dimension the likelihood is very informative, the direction is therefore, by definition, active.

We can give a graphical representation of the directions of the Active Subspace in Figures 4.7 and 4.8, where we make use of the principal components to determine the Active Subspace directions, as described in Section 4.2.4. We see in Figure 4.7 the directions of the Active Subspace generated using the prior (4.34), and we see that the predominant direction of the gradient will be horizontal and, wrongly, the first dimension θ_1 would be chosen as active variable.

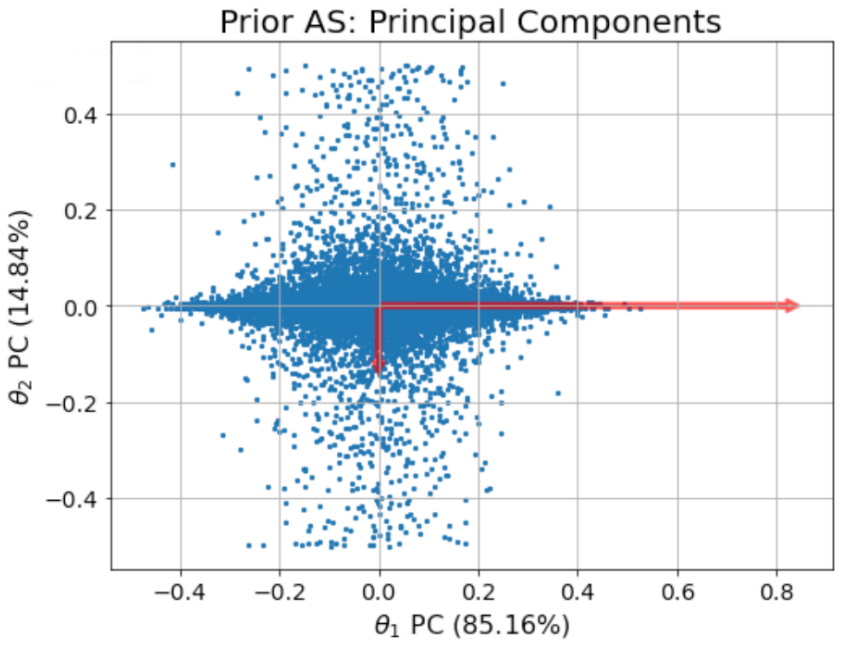

Figure 4.7: *Directions of the principal components of the covariance of gradients using* ***prior*** *samples: we can see that the main direction is horizontal. See difference with Figure 4.8 where the main direction is vertical*

If instead we plot the directions of the principal components estimated using the posterior points (a MCMC run with 100000 iterations has been used to sample the points), we obtain

Figure 4.8, where the main direction is instead vertical, and in this case, correctly, the second dimension of the state space θ_2 is chosen as active component.

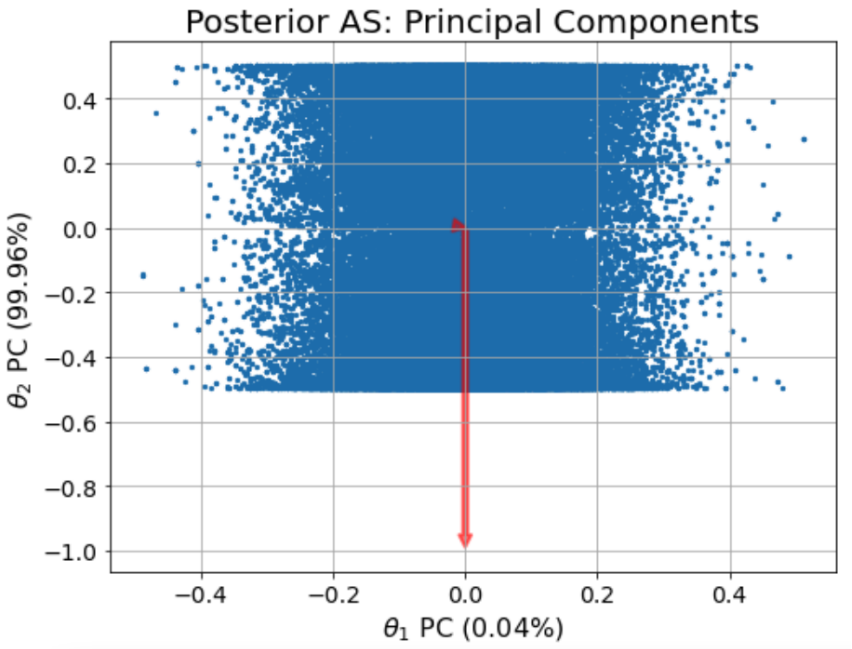

Figure 4.8: *Directions of the principal components of the covariance of gradients using* ***posterior*** *samples: we can see that the main direction is vertical. See difference with Figure 4.7 where the main direction is horizontal*

Comparing therefore Figures 4.7 for the prior, and 4.8 for the posterior, we understand that in this model the estimate of the Active Subspace using a prior AS will pose a challenge, in fact the true predominant direction of the Active Subspace in the posterior will be perpendicular to the one which would be wrongly estimated using the prior.

4.10 Conclusion

Active Subspaces are a way to address high-dimensionality challenges of MCMC, by identifying a sub-dimension of the state space which is less than the nominal dimension and by concentrating the main efforts of the algorithms on this Active Subspace. However there are some limitations that can show up in real-world situations, one for all is represented by the Hellinger bound equation (4.9), when the bound is too loose, empirical validations may become necessary to check the quality of the AS approximation. We have presented algorithm AS-MH-Bias in Section 4.6, historically the first AS-based algorithm to be applied to MCMC. Then we have shown in Section 4.8.2, AS-MH, an unbiased version, which uses pseudo-marginals, and targets the intended posterior, and we have compared the performances on a toy model in Section 4.8.4. In addition, we have seen in Section 4.9 that the prior distribution is used to generate samples for the Monte Carlo integration of (4.2) (or its approximation (4.12)) [Constantine et al., 2016], but there are cases where the Active Sub-

space generated by the prior, that we have called **prior Active Subspace** may be different from that generated by the posterior, the **posterior Active Subspace**, in such cases the approximation obtained using the prior will be poor. We will address this problem in Section 6.4 when a method to have progressive approximations of the Active Subspace matrices will be presented. There are alternative methods, such as using a Laplace approximation, which could potentially provide an efficient way to approximate the posterior Active Subspace instead of drawing from the prior in MCMC settings. Although we do not investigate these methods in this thesis, they may offer a valid alternative for future work [Tierney and Kadane, 1986, Rue et al., 2009].

Chapter 5

Active Subspaces proposed novel MCMC algorithms

5.1 Introduction

A lot of existing scientific literature on **Active Subspaces** (AS) start their work based on the concepts and methods that can be found both in [Constantine, 2015] and, for Monte Carlo (specifically MCMC) in [Constantine et al., 2016] (we have discussed the first AS algorithm to be applied to MCMC [Constantine et al., 2016], the AS-MH-Bias algorithm in Section 4.6). Both papers [Constantine, 2015] and [Constantine et al., 2016] can be said to be foundational, precisely because they made the way to a lot of subsequent studies (for example just to mention two among many [Cui et al., 2019] and [Parente, 2020]).
In our work we have chosen a different path, opting instead to build on the foundations laid by the AS-MH exact algorithm that uses an unbiased estimate of the likelihood in [Schuster et al., 2017] (we recap the algorithm in 4.8.2). And this decision, as already mentioned in Section 4.8, starts from the consideration that while both papers provide a full theoretical support for AS, in real-world cases their practical use seems to be difficult. Some weak points of the theory are evident in the very same paper [Constantine et al., 2016] which introduced it, when, after a $2D$ toy example is discussed and AS is successfully applied, with a more realistic 100D model the authors have to use empirical methods to assess convergence to the true posterior.
Fo this reason, from this section forward we will start building upon the AS-MH, trying to address some of the open points outlined in Section 4.7.

In Section 5.2 we introduce our first proposed novel AS algorithm, named AS-PMMH, that is based on the AS integration of the particle MCMC [Andrieu et al., 2010]. The AS-PMMH can be considered an equivalent of AS-MH, where the inactive variables are marginalised out by using a SMC sampler rather than Importance Sampling (IS). The reason for introducing AS-PMMH is that, in cases of high dimension of the inactive subspace, we know that the performances of IS worsen significantly as the number of points required to keep the ESS at

a predetermined level scales exponentially with the dimension d of the space [Agapiou et al., 2017], whereas we know that in SMC the cost is polynomial [Beskos et al., 2011, 2014]. We compare the performances in a Gaussian model in Section 5.3, the comparison with AS-MH and standard MCMC seem to show worse performances of AS-PMMH compared to the other methods, in terms of distribution of RMSE between the true posterior mean and the mean estimated with the method. This is possibly due to the fact that having kept the number of likelihood evaluations equal between the algorithms, for a fair comparison, and by using 100000 likelihood evaluations while we have as many iterations in standard MCMC, for AS-PMMH 100000 likelihood evaluation result in a mere 1666 final samples. The small number of samples highlights one of the challenges of AS-PMMH: the biggest share of computational effort is spent on the marginalisation of the inactive subspace, i.e. on the part of the space we are interested the least. We then introduced the more complex *Banana* model with some non linearities in Section 5.4, and the behaviour of the AS-PMMH shows long tails in the distribution of RMSE, indicating that sometimes the algorithm may get stuck in one of the tails. This is a behaviour that can happen when estimates of the likelihood are used in lieu of the actual likelihood, due to noise in the estimate [Andrieu and Vihola, 2015].

The consideration that AS-PMMH uses the computationally more intensive SMC sampler on the part of the state space we are interested the least, the inactive variable i, which may lead to a sub-optimal estimate of the active part a, brought us to devise an *inverted* algorithm, AS-PMMH-i in Section 5.5, where we switch roles and we use the internal SMC sampler on the active component whereas the outer MCMC is on the inactive part. But comparison of results with standard MCMC and AS-MH still shows long tails in the Banana model, indicating that the noise in the estimate of the likelihood still bring out issues of *sticky* behaviour, i.e. the algorithm getting stuck in one of the modes.

So we moved to a version that has no estimate of the likelihood. the AS-Gibbs of Section 5.6 integrates Gibbs sampling into AS. We show that, in the case of perfect or near-perfect Active Subspaces, when active and inactive variables are independent, the AS-Gibbs proves a winning strategy, outperforming all other algorithms.

Continuing on the strategy employed with AS-Gibbs, we have shown that, in the case of perfect or near-perfect Active Subspaces, by integrating MwPG [Chopin, 2002] into AS, we have devised AS-MwPG and AS-MwPG-i where sampling of particles in the inactive or active case respectively is done via an internal SMC sampler [Chopin, 2002]. We have shown in Section 5.8 that by using a "challenging" proposal in a bimodal posterior, the AS-MwPG-i has been the only algorithm capable of reconstructing correctly both modes, while the other algorithms remained stuck in one of the modes.

We finally introduced, in Section 5.9, a novel, alternative way to the traditional "eigen-based" method of [Constantine, 2015, Constantine et al., 2016] to determine the size of the Active Subspace. The new approach determines the dimension of the inactive subspace as the largest dimension that yields an ESS that does not drop below some threshold, by using

the prior as importance proposal and, on the models used, brings identical results to the traditional method.

5.2 Active Subspace particle MCMC algorithm: AS-PMMH

We have seen how the AS-MH algorithm by [Schuster et al., 2017], introduced in Section 4.8.2, produces an unbiased estimate of the likelihood where the inactive variables are marginalised out using an importance sampler.

Building on the pseudo-marginal approach, we now look for an enhancement of the AS-MH: we propose the use of AS in an algorithm that will sample from the marginal distribution of the inactive variables using a SMC sampler, instead of using Importance Sampling like for AS-MH. The proposed new algorithm consists in the application of the PMMH algorithm [Andrieu et al., 2010] (discussed in Section 2.8) to AS, we call therefore this new algorithm AS-PMMH. The main difference between AS-MH and AS-PMMH is that in AS-PMMH we will use an internal SMC to obtain an unbiased estimate of the likelihood where the inactive variables have been marginalised out, whereas in AS-MH we use a pseudomarginal (based on Importance Sampling) to get the likelihood estimate. Therefore: in AS-PMMH the internal SMC acts on the inactive variables, in similar way that in AS-MH the Importance Sampler was acting on the inactive variables. We expect AS-PMMH to outperform AS-MH in cases where SMC outperforms IS, so for example when the inactive subspace has high dimensions: let's remember that IS as a reference will require an exponential number of samples to keep the ESS constant as the dimension d_i of the inactive space grows [Agapiou et al., 2017], whereas the cost of constructing an SMC sampler that stabilises the variance of the marginal likelihood estimator grows in $O(d_i^2)$, therefore the introduction of AS-PMMH will also address Open Point **OP5** in the list that we introduced in Section 4.7.

As an expected drawback, the AS-PMMH brings an overhead considering that a SMC sampler is run on the inactive variables at each outer MCMC iteration. There will therefore be a trade-off between algorithmic complexity and the need for accuracy.

Alg. 9 SMC on inactive variables for a given a and t.

1: Simulate N_i points, $\{i_0^n\}_{n=1}^{N_i} \sim p_i(\cdot \mid a)$ and set each weight $w_0^n = 1/N_i$;
2: **for** $s = 1 : t$ **do**
3: **for** $n = 1 : N_i$ **do** ▷ reweight
4: **if** $s = 1$ **then**

$$\tilde{w}_s^n = w_{s-1}^n l_{1:s}\left(B_a a + B_i i^n\right);$$

5: **else**

$$\tilde{w}_s^n = w_{s-1}^n \frac{l_{1:s}\left(B_a a + B_i i_{s-1}^n\right)}{l_{1:s-1}\left(B_a a + B_i i_{s-1}^n\right)};$$

6: **end if**
7: **end for**
8: $\{w_s^n\}_{n=1}^{N_i} \leftarrow$ normalise $\left(\{\tilde{w}_s^n\}_{n=1}^{N_i}\right)$;
9: If $s = t$, go to line 32;
10: **for** $n = 1 : N_i$ **do** Simulate the index $v_{s-1}^n \sim \mathcal{M}\left(\left(w_s^1, ..., w_s^{N_i}\right)\right)$ of the ancestor of particle n;
11: **end for**
12: **if** *some degeneracy condition is met* **then** ▷ resample
13: **for** $n = 1 : N_i$ **do** Set $i_s^n = i_{s-1}^{v_{s-1}^n}$;
14: **end for**
15: $w_s^n = 1/N_i$ for $n = 1 : N_i$;
16: **else**
17: **for** $n = 1 : N_i$ **do**
18: Set $i_s^n = i_{s-1}^n$;
19: **end for**
20: **end if**
21: **for** $n = 1 : N_i$ **do** ▷ move
22: Simulate $i_s^{n*} \sim q_{t,i}(\cdot \mid i_s^n, a)$;
23: Set $i_s^n = i_s^{n*}$ with probability
24:

$$1 \wedge \frac{l_{1:s}\left(B_a a + B_i i_s^{n*}\right) p_i\left(i_s^{n*} \mid a\right) q_{t,i}\left(i_s^n \mid a\right)}{l_{1:s}\left(B_a a + B_i i_s^n\right) p_i\left(i_s^n \mid a\right) q_{t,i}\left(i_s^{n*} \mid a\right)},$$

25: **end for**
26: **end for**
27: Estimate $l_{t,a}(a)$ using

$$\bar{l}_{t,a}(a) = \prod_{s=1}^{t} \sum_{n=1}^{N_i} \tilde{w}_s^n.$$

Alg. 10 Active subspace particle marginal Metropolis-Hastings

Initialise a^0;
Run Algorithm 9 for $a = a^0$ and $t = T$, obtaining $\bar{l}_{T,a}(a^0)$ and weights, denoted $\left(w_T^{1,0}, ..., w_T^{N_i,0}\right)$;
$u^0 \sim \mathcal{M}\left(\left(w_T^{1,0}, ..., w_T^{N_i,0}\right)\right)$;
Let $\bar{l}_a^0 = \bar{l}_{T,a}(a^0)$;
for $m = 1 : N_a$ **do**
    $a^{*m} \sim q_a(\cdot \mid a^{m-1})$;
    Run Algorithm 9 for $a = a^{*m}$ and $t = T$, obtaining $\bar{l}_{T,a}(a^{*m})$ and weights, denoted $\left(w_T^{*1,m}, ..., w_T^{*N_i,m}\right)$;
    $u^{*m} \sim \mathcal{M}\left(\left(w_T^{*1,m}, ..., w_T^{*N_i,m}\right)\right)$;
    Set $\left(a^m, \{i^{n,m}, w^{n,m}\}_{n=1}^{N_i}, u^m, \bar{l}_a^m\right) = \left(a^{*m}, \{i^{*n,m}, w^{*n,m}\}_{n=1}^{N_i}, u^{*m}, \bar{l}_{T,a}(a^{*m})\right)$ with probability

$$\alpha_a^m = 1 \wedge \frac{p_a(a^{*m})\,\bar{l}_{T,a}(a^{*m})}{p_a(a^{m-1})\,\bar{l}_a^{m-1}} \frac{q_a(a^{m-1} \mid a^{*m})}{q_a(a^{*m} \mid a^{m-1})};$$

    Else let $\left(a^m, \{i^{n,m}, w^{n,m}\}_{n=1}^{N_i}, u^m, \bar{l}_a^m\right) = \left(a^{m-1}, \{i^{n,m-1}, w^{n,m-1}\}_{n=1}^{N_i}, u^{m-1}, \bar{l}_a^{m-1}\right)$;
end for

5.3 Comparison of performances of AS-PMMH with other algorithms in a Gaussian model

5.3.1 Introduction

We will draw in this section some scenarios to understand better the relative strength and weaknesses of AS-PMMH Algorithm 10 compared to the others. We start by introducing in Section 5.3.2 a toy model that we will use for later tests.

5.3.2 Gaussian Model

In this section, we explore some of the features of **Active Subspaces** by using an ad-hoc system obtained by overparameterization of a Gaussian model. In particular, we consider a system that has an underlying Gaussian distribution, and we focus on the inference of the true mean. This system is intentionally overparameterized, with parts of the system that do not influence the likelihood.

Likelihood

In the following, the vector of parameters is referred to as $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$, and the sum of the parameters represent the estimated mean of the Gaussians that compose the likelihood

$$\hat{\mu} = \sum_{j=1}^{n} \theta_j \tag{5.1}$$

The likelihood function of our model:

$$l(y|\theta) = \prod_{i=1}^{P} \mathcal{N}(y_i|\hat{\mu}, 1) \tag{5.2}$$

Here, $\hat{\mu}$ is from equation (5.1), P represents the number of independent observations, and n is the size of the state space. The normal distribution, for each independent observation, will model the probability of observing the data y_i given the array of parameters θ.

Prior

We will use a prior given by a Gaussian centered at 0 and with variance 5000 in all directions:

$$p(\theta) = \prod_{i=1}^{n} \mathcal{N}(0, \sigma^2) \tag{5.3}$$

where the variance $\sigma^2 = 5000$.

Gradient of log-likelihood

With the likelihood of equation (5.2), with the explicitation of the normal terms we have

$$l(y|\theta) = \prod_{i=1}^{P} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(y_i - \sum_{j=1}^{n} \theta_j)^2} \tag{5.4}$$

The log-likelihood, from (5.4) is

$$\begin{aligned} \log l(y|\theta) &= \sum_{i=1}^{P} \log \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(y_i - \sum_{j=1}^{n} \theta_j)^2} \right) \\ &= \sum_{i=1}^{P} \left[-\frac{1}{2} \log(2\pi) - \frac{1}{2}(y_i - \sum_{j=1}^{n} \theta_j)^2 \right] \end{aligned} \tag{5.5}$$

Differentiating the negative log likelihood w.r.t. a generic parameter θ_k, *k=1,...n* (by

looking at (5.5), all the $\partial\theta_k$ will have the same expression)

$$\frac{\partial}{\partial\theta_k}(-\log l(y|\theta)) = \sum_{i=1}^{P}(\sum_{j=1}^{n}\theta_j - y_i),\ k{=}1,{\ldots}n \tag{5.6}$$

We take a look at level surfaces for this model in Section 5.3.2, Figures 5.1 and 5.2.

Overparametrization

The overparameterization of the system is evident in the fact that we only need one parameter to infer the mean of a Gaussian of equation (5.2), therefore our model, for $n > 1$ will include more parameters than strictly necessary. For instance, in the case where $n = 2$, the level surface for the likelihood can be represented as:

$$\theta_1 + \theta_2 = \text{true mean} \tag{5.7}$$

Which means that any value of (θ_1, θ_2) satisfying (5.7) will leave the likelihood invariant. Similarly, for $n = 3$, the level surface is defined by:

$$\theta_1 + \theta_2 + \theta_3 = \text{true mean} \tag{5.8}$$

The overparameterization can be further extended by adding more components to the system. This increases the dimensionality of the state space, yet the actual dimension required to represent the system remains one: by what we said in the previous sections, our expectations when analysing the system are to find an **Active Subspace** having dimension 1, and an **Inactive Subspace** having dimension $n - 1$ (increasing the number of parameters will, therefore, increase the dimension of the **Inactive Subspace**). We further expect that the data will show that the direction of the **Active Subspace** will be perpendicular to, in the case of $n = 2$ or $n = 3$, the line represented by (5.7) or the plane represented by (5.8) respectively, in fact along the above mentioned lines the likelihood will remain constant.

Visual representation

We can see what the level surface of the likelihood will look like, projected on the same 3D chart of the prior, for the particular case where $\theta_1 + \theta_2 + \theta_3 = 0$, we give two snapshots of the 3D graph done using python library plotly [Plotly, 2015]. We can appreciate from Figures 5.1 and 5.2 a cross-section showing the prior and the level surface of the log-likelihood

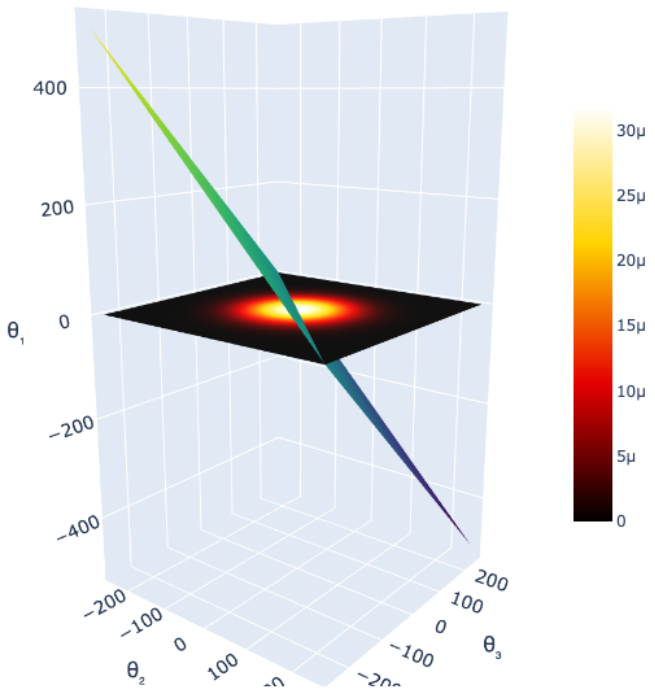

Figure 5.1: *Graphical representation of the model described in 5.3.2: we see a 2D slice of the Gaussian prior on the horizontal plane $\theta_1 = 0$ (black color indicates low probability zones, whereas progressively warmer color towards the center indicate zones of higher probability, as indicated by the colorbar) together with the level surface of the likelihood in the particular case $\theta_1 + \theta_2 + \theta_3 = 0$ (green plane), created using python library plotly [Plotly, 2015]*

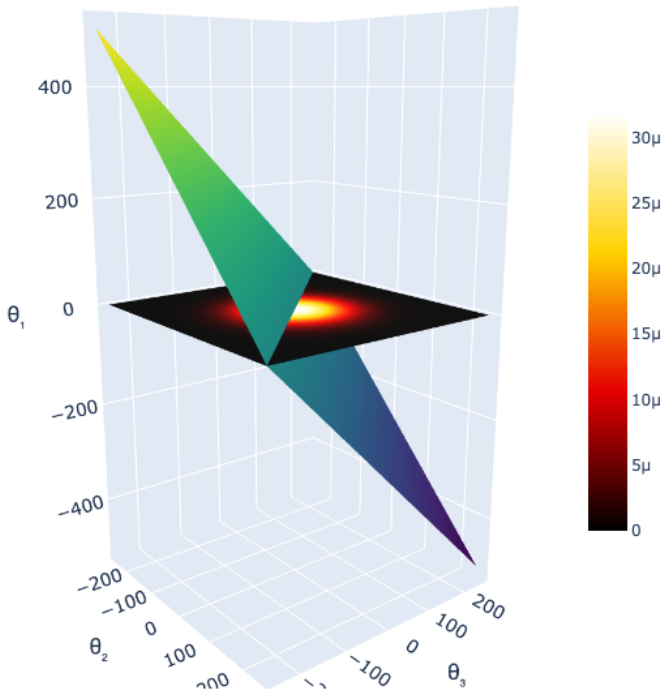

Figure 5.2: *Different viewpoint of Figure 5.1, created using python library plotly [Plotly, 2015]*

Active Subspace dimension

By performing the eigenvalue analysis as described in Section 4.2.2, using the gradient of log likelihood (5.6) we see that for a 10D system we have the eigenvalues in Figure 5.3

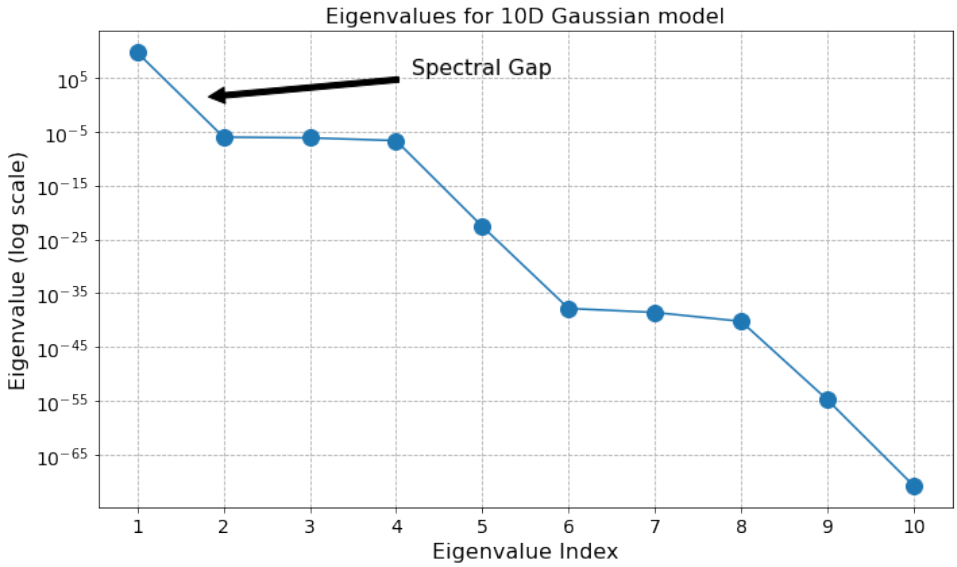

Figure 5.3: *Eigenvalues of 10D Gaussian model, we see that the estimate AS size is* 1*, considering the spectral gap between eigenvalues* 1 *and* 2*. The dimension of the Active Subspaces is* $n_a = 1$*.*

For a 25D system the chart is in Figure 5.4

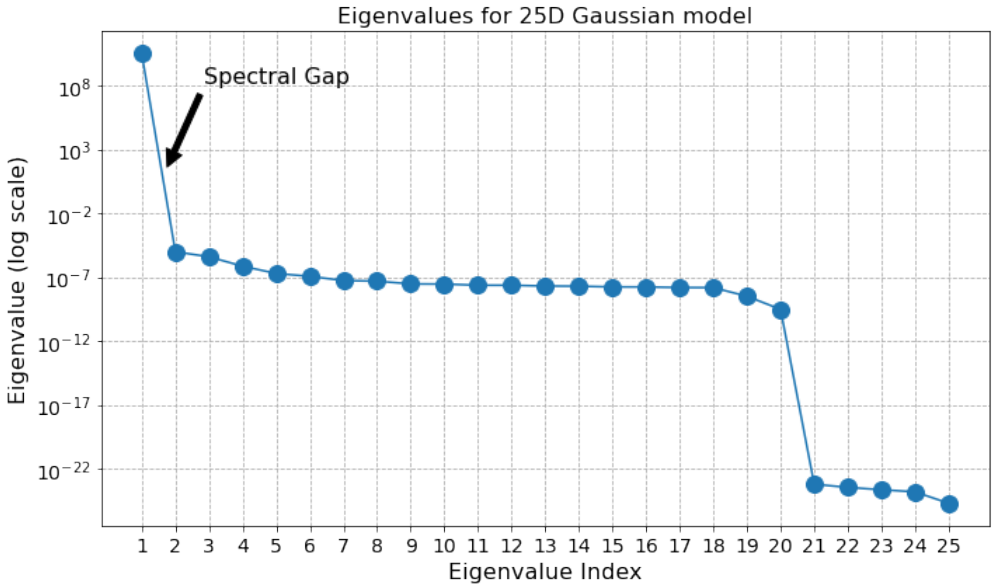

Figure 5.4: *Eigenvalues of 25D Gaussian model, we see that the estimate AS size is* 1*, considering the spectral gap between eigenvalues* 1 *and* 2*. The dimension of the Active Subspaces is* $n_a = 1$.

We see that for both 10D and 25D the size of the Active Subspace is 1, according to the procedure that we described in Section 4.2.2.

Mean and covariance values used in tests

For the tests, we calculated both the estimated posterior mean μ_π and the estimated posterior covariance Σ_π of the 10D and 25D Gaussian model by running a test SMC with $N = 50$M particles each, so if we name w_j the weights and x_j the particles we have

$$\mu_\pi = \sum_{j=1}^{N} w_j x_j \tag{5.9}$$

$$\Sigma_\pi = \sum_{j=1}^{n} w_j (x_j - \mu_\pi)(x_j - \mu_\pi)^T \tag{5.10}$$

We then used both μ_π of equation (5.9) and Σ_π of equation (5.10) as the "true" posterior values for reference. We used μ_π to calculate the RMSE of the difference between the mean estimated by each of the algorithm runs and the true mean μ_π (see for example Figure 5.5).

We then used Σ_π to build both the optimal coviariance of the proposal in standard MCMC

$$\frac{2.38^2}{d}\Sigma_\pi \tag{5.11}$$

with d the dimension of the state space [Roberts and Rosenthal, 2001], and by remembering that $\theta = B_a a + B_i i$, following the same concept of optimal scaling [Roberts and Rosenthal, 2001] on the active subspace, we used

$$\frac{2.38^2}{d_a} B_a^T \Sigma_\pi B_a \tag{5.12}$$

as optimal covariance for proposal of MCMC moves on **active** marginals, whereas we used

$$\frac{2.38^2}{d_i} B_i^T \Sigma_\pi B_i \tag{5.13}$$

as optimal covariance for proposal of MCMC moves on **inactive** marginals. In equations (5.12) and (5.13), d_a and d_i are the dimensions of the active and inactive subspaces respectively.

5.3.3 Comparison of performances in the Gaussian model

We will, in this section, compare the performances of standard MCMC, AS-MH and AS-PMMH, both for the 10D and 25D Gaussian model. In the case, for example, of the 25D Gaussian model, we will compare:

- standard MCMC Algorithm 4 performed on the full 25D space;
- AS-MH Algorithm 8;
- AS-PMMH Algorithm 10, which will perform an outer MCMC and several inner SMC samplers: for each step the algorithm will perform an SMC sampler on the 24D inactive subspace to obtain an unbiased estimate of the likelihood to be used in the outer *active* MCMC as estimate of the marginal likelihood.

Likelihood evaluations

When comparing the performances of the algorithms, we have tried to keep the number of likelihood evaluations constant in each run, to ensure a fair comparison. Due to the structure of the algorithms, the same number of likelihood evaluations may result in a different number of output samples. Taking a reference figure of 100000 likelihood evaluations, it will result in:

- Standard MCMC: 100000 iterations, and therefore as many samples;
- AS-MH: if we use 10 inactive variables in the pseudo-marginal calculation, we will have 10000 outer MCMC iterations ($10000 \times 10 = 100000$);
- AS-PMMH: if we use 10 inner inactive variables and 6 tempering steps of the inner SMC sampler,then we have 1666 outer MCMC steps and therefore as many output

samples ($1666 \times 10 \times 6 = 99960$ which is the closest integer to 100000), we summarise in Table 5.1

Method	Nr of output samples
MCMC	100000
AS-MH*	10000
AS-PMMH	1666

Table 5.1: Comparison of number of output samples when performing 100000 likelihood evaluations in different MCMC methods. **For AS-MH the figure in the table indicates the number of samples when one inactive sample is used per active variable, like in formula (4.23). If instead all inactive particles are used, like in formula (4.24), the relative figure must be multiplied by the number of inactive variables used,* 10 *in this case.*

In Table 5.1, for AS-MH the figure in the table indicates number of samples when one inactive sample is used per active variable, like in formula (4.23). If instead all inactive particles are used, like in formula (4.24), the relative figure must be multiplied by 10 (number of inactive variables used) to consider all the samples.

From the numbers above, we understand that AS-PMMH carries a problem, since 100000 likelihood evaluations result in a mere 1666 output samples. Structurally, in AS-PMMH, a lot of computational effort is spent in the calculation of the inactive marginal, therefore it is inefficient since we spend a lot of computational effort on the part of the space, the inactive, that we are interested the least.

For ease of reference in the rest of the sections, we also report the table for 200000 likelihood evaluations (it is the above Table 5.1 with numbers $\times 2$)

Method	Nr of output samples
MCMC	200000
AS-MH*	20000
AS-PMMH	3332

Table 5.2: Comparison of number of output samples when performing 200000 likelihood evaluations in different MCMC methods, this is the equivalent of Table 5.1, adapted for 200000. **For AS-MH the figure in the table indicates number of samples when one inactive sample is used per active variable, like in formula (4.23). If instead all inactive particles are used, like in formula (4.24), the relative figure must be multiplied by the number of inactive variables used,* 10 *in this case.*

We have reported additional data for other algorithms in Appendix D, tables D.1 and D.2.

Comparison using MultiESS

The first comparison is using the multiESS [Vats et al., 2019] that we described in Section 2.4.8, we use the *R* software implementation of it. We have run the algorithms in one test

run with 200000 likelihood evaluations for each algorithm, using no burn-in (the choice of not having burn-in, here and in the rest of the experiments, has been made considering that we start already in the posterior set), and we used optimal covariances as per equations (5.11) in the MCMC proposal, and (5.12) for AS-MH and AS-PMMH in the active marginal MCMC proposal. As we explained 200000 likelihood evaluations will give a different number of output samples in each of the three algorithms, look at Table 5.2 for reference.

The acceptance rate of the MCMC parts has been around 25% for all algorithms. Results are in Tables 5.3 and 5.4.

Algorithm	**MultiESS**/N	**MultiESS**
MCMC	1.6	3200
AS-MH	22.5	4500
AS-PMMH	20.2	673

Table 5.3: Gaussian 10D multiESS out of 200000 likelihood evaluations (please see Table 5.2 for the number of corresponding output samples N per algorithm).

Algorithm	**MultiESS**/N	**MultiESS**
MCMC	1.3	2600
AS-MH	21.9	4380
AS-PMMH	26.7	890

Table 5.4: Gaussian 25D multiESS out of 200000 likelihood evaluations (please see Table 5.2 for the number of corresponding output samples N per algorithm).

We see in Table 5.3 for 10D and Table 5.4 for the 25D case, that the performance of AS-MH seem to remain fairly constant, accounting for some random variability, between the 10D and the 25D runs, that is possibly due to the Gaussian system being a *near-perfect* Active Subspace, and so the AS-MH pseudo-marginal becomes close to equation (4.25), and performance may remain constant since the dimension of the Active Subspace remains constant to 1 from 10D to 25D. We also have to remember that, by construction, AS-MH and AS-PMMH output samples are more likely to show less correlation than the standard MCMC, which in turn will cause higher multiESS. The reason of lower correlation is that even when using small steps in the active marginal MCMC, points in the state space may contain inactive parts that may be very different. In fact we see significantly higher multiESS per sample in Tables 5.3 and 5.4.

We also see from Table 5.4 that the overhead of running AS-PMMH with its internal SMC samplers, seems not to pay off compared to AS-MH in terms of multiESS, this may be due to the fact that the Gaussian system is fairly simple to explore, and the AS-MH may be, at least in the dimensions up to 25D explored here, a better choice.

Comparison of estimate of expectation

To compare the estimates of the mean coming from the three algorithms, we have performed 50 runs of each algorithm, with 100000 likelihood evaluations in each algorithm, on both the 10D and 25D models, and we have measured the Root Mean Squared Error (RMSE) of the difference of the true mean and the mean estimated by each of the three algorithms. As "true" mean we used μ_p as explained in equation (5.9), and we used optimal covariances as per equations (5.11) in MCMC, and (5.12) for AS-MH and AS-PMMH in the active marginal MCMC proposal.

We report the chart with the violin plots showing mean and upper and lower quartile of the distribution of the RMSE.

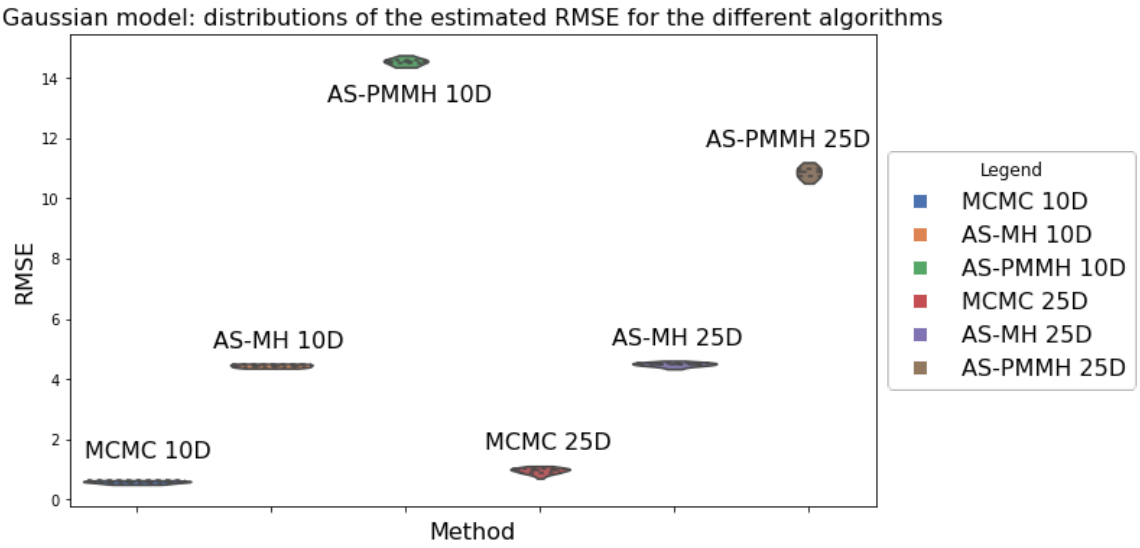

Figure 5.5: *Distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs. We see that the standard MCMC in both* 10D *and* 25D *has a lower error. Second best performer is AS-MH and third best is AS-PMMH. The algorithms are in order from the LHS: MCMC, AS-MH and AS-PMMH (*10D *first then* 25D*)*

5.3.4 Review

In the analysis of performance using the Gaussian model we see that there is a trade-off: using the AS-MH algorithm instead of the standard MCMC brings the advantage of having better multiESS, as we see in Tables 5.3 and 5.4, which indicates that the samples have lower autocorrelation and better mixing, but seems to do so at the cost of accuracy (see Figure 5.5), in fact the RMSE of the expectation is higher than standard MCMC. The AS-MH in this case seems not to suffer the *curse of dimensionality* when increasing from 10D to 25D system, and this is possibly because we are in a *near perfect* Active Subspace, and in this case the AS-MH outer MCMC targets a relatively constant space dimension of 1.

The AS-PMMH algorithm has worse multiESS (see Tables 5.3 and 5.4), and worse RMSE than AS-MH in Figure 5.5: this is possibly due to the lower number of output samples which makes the estimate poorer (see Table 5.1 for reference, AS-PMMH has 1666 output samples per every 100000 likelihood evaluations).

5.4 Comparison of performances of AS-PMMH with other algorithms in the "Banana" model

5.4.1 Introduction

We now introduce a second toy model, it will be similar to the model previously introduced in Section 5.3.2, but it adds some non-linearity in the expression of equation (5.1). The aim of the toy model is to set a more challenging scenario for the algorithms.

5.4.2 Banana model

The model we are going to introduce in this section is similar to the one discussed in Section 5.3.2, it will add a bit of non-linearity and because the non linearity is quadratic and there is some curvature, we call this model "banana". The difference with the model of Section 5.3.2 will be in having quadratic terms in the expression of the estimated mean of the likelihood Gaussians. The Bayesian inverse model

$$l(\theta) = \prod_{i=1}^{P} \mathcal{N}\left(y_i \mid \hat{\mu}, 1\right) \tag{5.14}$$

Where in (5.14), the term $\hat{\mu}$ represents the estimate of the mean of the Gaussians using the parameters of the state space model $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$

$$\hat{\mu} = \theta_1 + \theta_2 + \ldots + \theta_n + c + b\left(\theta_n^2 + \theta_{n-1}^2 + \ldots + \theta_{n-H+1}^2\right) \tag{5.15}$$

We see that the case where contemporarily $b = 0$ and $c = 0$ in (5.15) brings again the Gaussian model of Section 5.3.2, and also, for the model to make sense, we need to have

$H \leq n$. The log likelihood of (5.14) is

$$\begin{aligned}
\log l(\theta) &= \log\left(\prod_{i=1}^{P} \mathcal{N}\left(y_i \mid \hat{\mu}, 1\right)\right) \\
&= \sum_{i=1}^{P} \log\left(\mathcal{N}\left(y_i \mid \hat{\mu}, 1\right)\right) \\
&= \sum_{i=1}^{P} \log\left(\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y_i - \hat{\mu})^2}{2}\right)\right) \\
&= \sum_{i=1}^{P} \left(-\frac{1}{2}\log(2\pi) - \frac{(y_i - \hat{\mu})^2}{2}\right)
\end{aligned} \tag{5.16}$$

The gradient of the negative log likelihood for $j \leq n - H$ is:

$$\frac{\partial}{\partial \theta_j}(-\log l(\theta)) = \sum_{i=1}^{P}(\hat{\mu} - y_i) \tag{5.17}$$

And for $j > n - H$:

$$\frac{\partial}{\partial \theta_j}(-\log l(\theta)) = \sum_{i=1}^{P}(\hat{\mu} - y_i)\left(1 + 2b\theta_j\right) \tag{5.18}$$

The components of the state space that will be affected by the quadratic part will be $(\theta_{n-H+1}, \theta_{n-H+2}, ..., \theta_n)$.

Prior

We will use the same prior of the previous model of Section 5.3.2: a Gaussian centered at 0 and with variance 5000 in all directions:

$$p(\theta) = \prod_{i=1}^{n} \mathcal{N}(0, \sigma^2) \tag{5.19}$$

where the variance $\sigma^2 = 5000$.

Visual representation

We can have an idea of what the level surface of the log-likelihood will look like in the banana family of models models. For the particular case where the model has $\theta_1 + \theta_2 + \theta_3 + b\left(\theta_2^2 + \theta_3^2\right) = 0$, using the value $\mathbf{b = 0.001}$, so that the curvature is mild but still visible, we give a couple of snapshots of the 3D graph of the level surface projected on the same chart of the prior. We can appreciate from Figures 5.6 and 5.7 a cross-section showing the prior and the level surface of the log-likelihood, we can appreciate the curvature of the level surface (green oblique plane) if compared to Figures 5.1 and 5.2

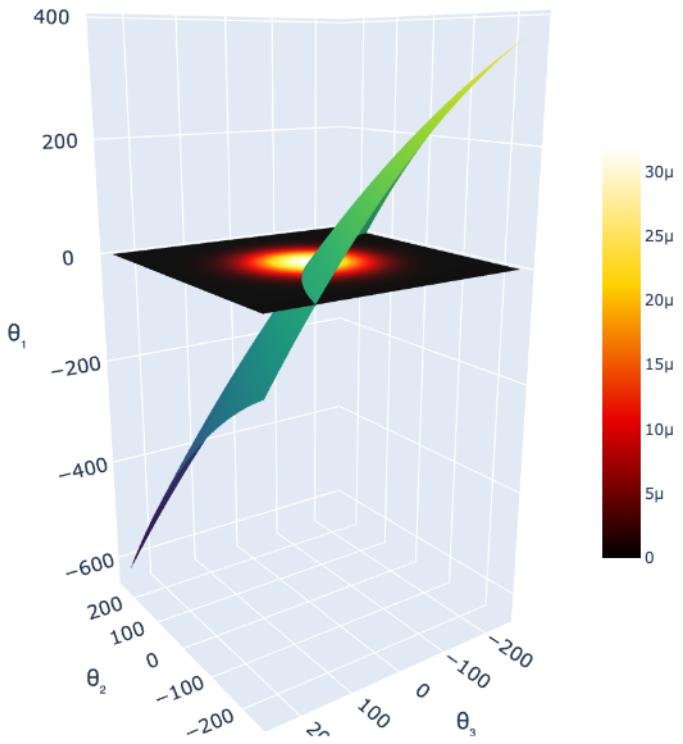

Figure 5.6: *Graphical representation of the model described in Section 5.4.2: we see a 2D slice of the Gaussian prior on the plane $\theta_1 = 0$ (black color indicates low probability zones, whereas progressively warmer color towards the center indicate zones of higher probability, as indicated by the colorbar) together with the level surface of the likelihood in the particular case $\theta_1 + \theta_2 + \theta_3 + b\,(\theta_2^2 + \theta_3^2) = 0$ (green plane), with $\mathbf{b} = \mathbf{0.001}$, so that the curvature is mild but still visible. The curvature can be appreciated by comparing with Figures 5.1 and 5.2, where the curvature was absent. Created using python library plotly [Plotly, 2015]*

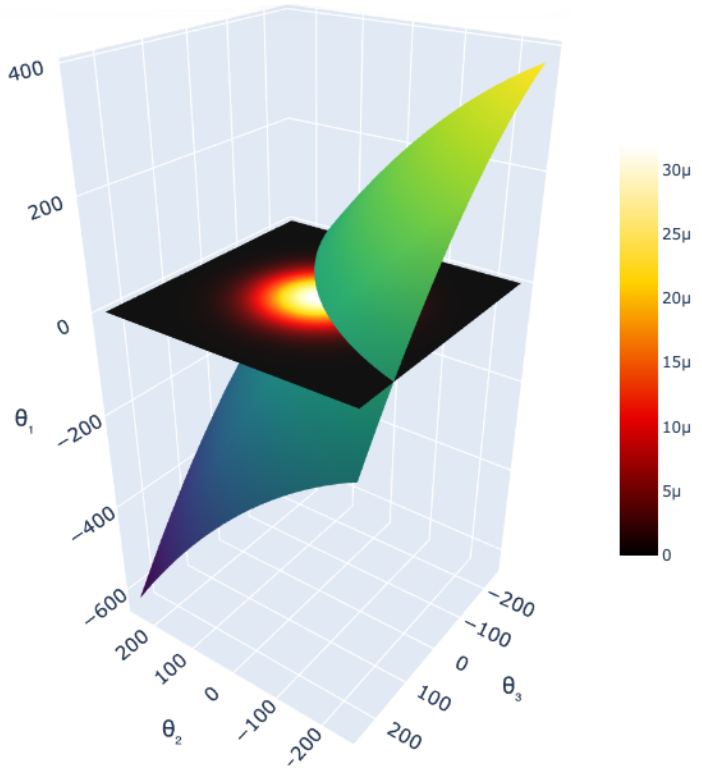

Figure 5.7: *Different viewpoint of Figure 5.6, created using python library plotly [Plotly, 2015]*

Active Subspace dimension

By performing the eigenvalue analysis as described in Section 4.2.2, using the gradient of log likelihood (5.6) we see that for a 10D system we have the eigenvalues in Figure 5.8

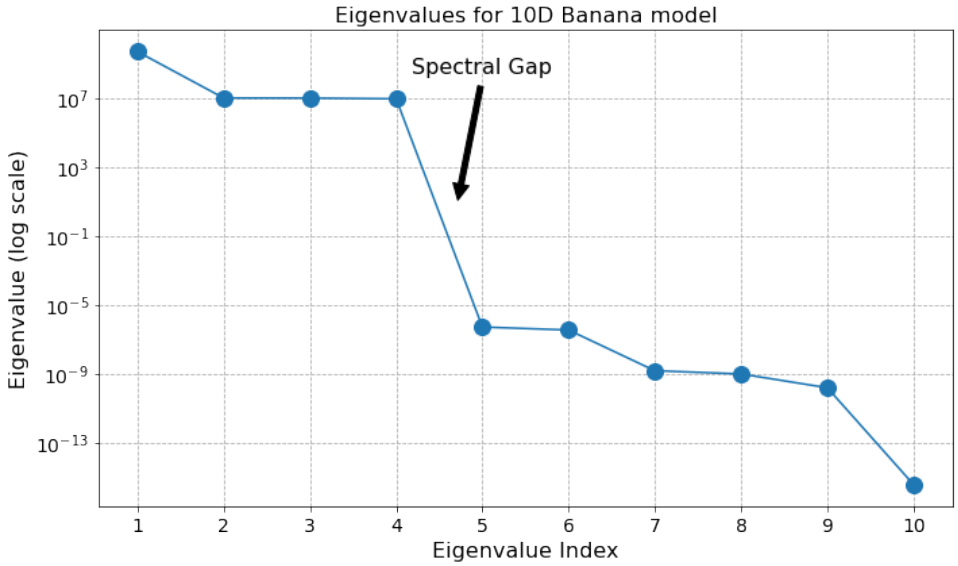

Figure 5.8: *Eigenvalues of 10D Banana model, we see that the estimate AS size is* 4*, considering the spectral gap between eigenvalues* 4 *and* 5*. The dimension of the Active Subspaces is* $n_a = 4$*.*

For a 25D system the chart is in Figure 5.9

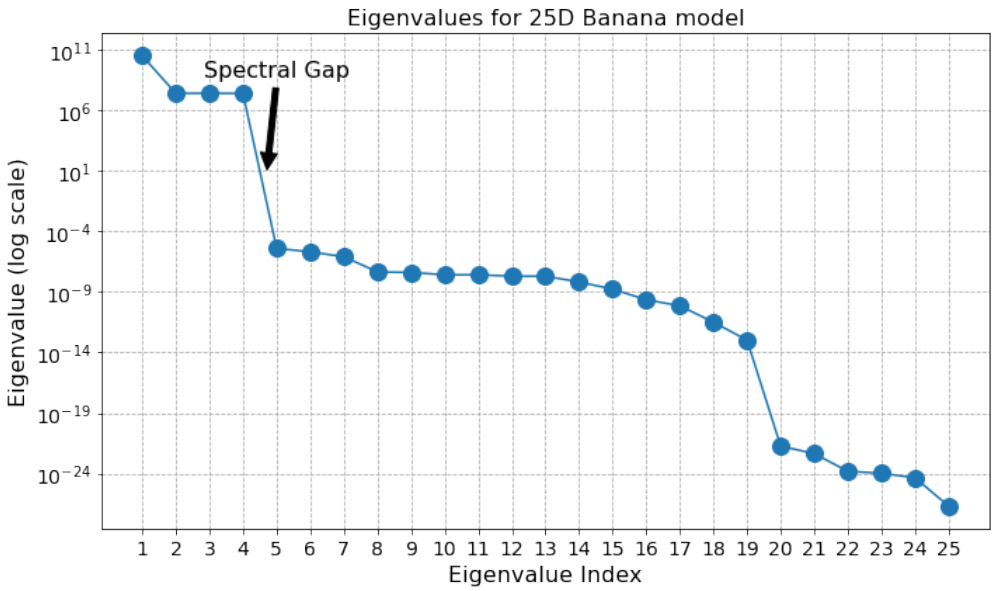

Figure 5.9: *Eigenvalues of 25D Banana model, we see that the estimate AS size is* 4*, considering the spectral gap between eigenvalues* 4 *and* 5*. The dimension of the Active Subspaces is* $n_a = 4$*.*

Mean and covariance values used in tests

For the tests, we calculated both the estimated posterior mean μ_π and the estimated posterior covariance Σ_π of the 10D and 25D Banana model by running a test SMC with $N = 50$M

particles each, and calculating μ_π as in equation 5.9 and Σ_π as in equation (5.10). We then used both μ_π and Σ_π as the "true" posterior values for reference. We used μ_π to calculate the RMSE of the difference between the mean estimated by each of the algorithm runs and the true mean μ_π (see for example Figure 5.5).

We then used Σ_π to build both the optimal covariance of the proposal in standard MCMC using equation (5.11) and (5.12) as optimal covariance for proposal of MCMC moves on **active** marginals, whereas we used (5.13) as optimal covariance for proposal of MCMC moves on **inactive** marginals. In equations (5.12) and (5.13), d_a and d_i are the dimensions of the active and inactive subspaces respectively.

5.4.3 Comparison of performances in the Banana model

Introduction

We will, in this section, compare the performances of standard MCMC, AS-MH and AS-PMMH. For the rest of the chapter we will be using the 25D Banana model. We have used the following parameters in the model of equation (5.15): $b = 0.001$ (mild curvature), $c = 0$ (absence of the constant term), $H = 3$ (the last 3 components will be affected by the quadratic part of (5.15)).

Similarly to what we have done in Section 5.3.3, we will compare:

- standard MCMC Algorithm 4 performed on the full 25D space;
- AS-MH Algorithm 8;
- AS-PMMH Algorithm 10, which will perform an outer MCMC and several inner SMC samplers: for each step the algorithm will perform an SMC sampler on the 21D inactive subspace to obtain an unbiased estimate of the likelihood to be used in the outer *active* MCMC as an estimate of the marginal likelihood.

Comparison using MultiESS

The first comparison is using the multiESS [Vats et al., 2019] that we described in Section 2.4.8, we use the R software implementation of it. We have run the algorithms in one test run with 200000 likelihood evaluations for each algorithm, using no burn-in, and we used optimal covariances as per equations (5.11) in the MCMC proposal, and (5.12) for AS-MH and AS-PMMH in the active marginal MCMC proposal. As we explained 200000 likelihood evaluations will give a different number of output samples in each of the three algorithms, look at Table 5.2 for reference. All algorithms have shown an acceptance rate of around 10% in their main MCMC. Results are in Table 5.5.

Algorithm	MultiESS/N	MultiESS
MCMC	0.1	200
AS-MH	5.7	1140
AS-PMMH	15.5	516

Table 5.5: Banana 25D multiESS out of 200000 likelihood evaluations (please see Table 5.2 for the number of corresponding output samples N per algorithm).

We see in Table 5.5 that as the posterior becomes more complex, the multiESS drops: look for comparison with Table 5.4 in the simpler posterior of the Gaussian case where all percentages were higher. We remind that, by construction, AS-MH and AS-PMMH output samples are more likely to show less correlation than the standard MCMC, which in turn will cause higher multiESS. The reason of lower correlation is that even when using small steps in the active marginal MCMC, point in the state space may contain inactive parts that may be very different. We see, in fact, significantly higher multiESS per sample for the two methods in Table 5.5, with AS-MH having the highest absolute multiESS. We also remind that the output samples N produced by the 200000 likelihood evaluations are different for each algorithm, with numbers reported in Table 5.2: in the AS-PMMH case the higher relative multiESS compared to AS-MH results in a lower absolute multiESS, due to the low number of output samples which, for AS-PMMH.

Comparison of estimate of expectation

To compare the estimates of the mean coming from the three algorithms, we have performed 50 runs of each algorithm, with 100000 likelihood evaluations in each algorithm in the 25D Banana model, and we have measured the Root Mean Squared Error (RMSE) of the difference of the true mean and the mean estimated by each of the three algorithms. As "true" mean we used μ_p as explained in equation (5.9), and we used optimal covariances as per equations (5.11) in MCMC, and (5.12) for AS-MH and AS-PMMH in the active marginal MCMC proposal.

We report the chart with the violin plots showing mean and upper and lower quartile of the distribution of the RMSE.

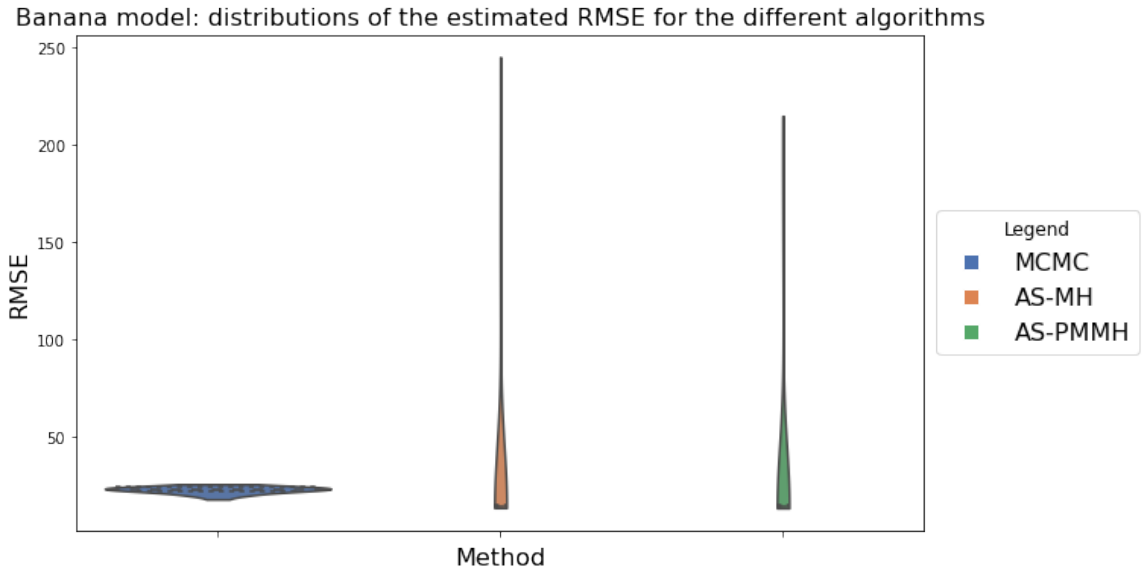

Figure 5.10: *Distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs. Starting from LHS: MCMC, AS-MH and AS-PMMH. We can see from the chart that the distributions for AS-MH and AS-PMMH have lower mean and upper quartile of MCMC, but longer tails. This may indicate very noisy estimates of the likelihood in some of the runs of both algorithms which cause the distributions to have long tails. The MCMC has comparatively smaller tails.*

We see in Figure 5.10 that although AS-MH and AS-PMMH both have lower mean and upper quartile of the standard MCMC, their distribution is characterised by long tails, although fairly thin. It is not uncommon that MCMC algorithms that use estimates of the likelihood will show *sticky* behaviour (i.e. the MCMC getting stuck), like the one shown by AS-MH and AS-PMMH in Figure 5.10. This is due to the chain getting stuck, possibly due to noisiness of the likelihood estimates that we use in the AS-MH and AS-PMMH, and also to the very few effective samples of the two algorithms.

An example taken from one of the runs of the AS-PMMH shows the *stickiness* in action: Figure 5.11 shows the trace-plot of one of the components during a AS-PMMH run, we see how a noisy estimate of the likelihood causes the outer MCMC to become stuck. One potential fix for the observed sticky behavior could be to increase the number of internal SMC tempering steps or to increase the number of outer MCMC iterations. This approach may help reduce the noise in likelihood estimates, allowing the MCMC chain to explore the posterior more effectively. However, this improvement would come at the cost of a significant increase in the number of likelihood evaluations.

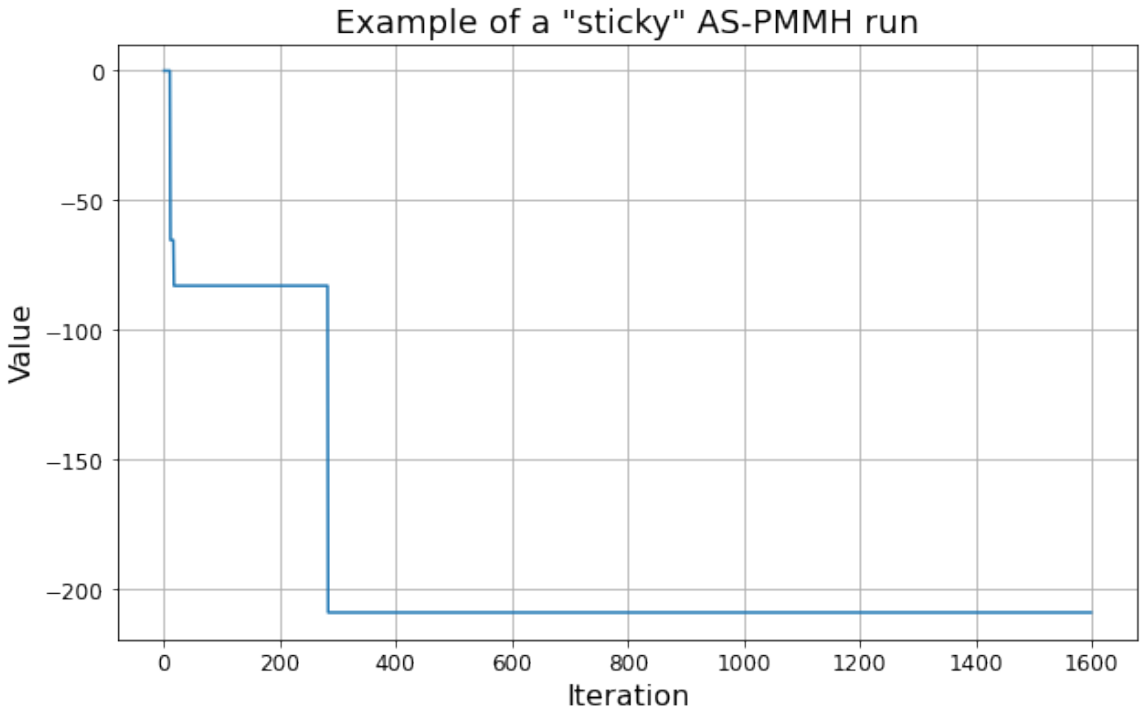

Figure 5.11: *Example of 'sticky' trace-plot in AS-PMMH, taken from one of the runs in Figure 5.10: a noisy estimate of the likelihood causes the outer MCMC to remain stuck for a long time, and this causes the very long tails in the distribution of RMSE seen in Figure 5.10 for the AS-PMMH.*

Review

Although on the theoretical level AS-PMMH may be seen as an advancement of the AS-MH algorithm, there are challenges that the AS-PMMH faces that are common to the AS-MH. We saw previously in Section 5.3.2 how, in a simpler model, all algorithms were not showing noisy likelihood behaviours, see for example Figure 5.5 where distributions in the simpler Gaussian model case show relatively short tails, and compare with Figure 5.10 of the more complex Banana model. On the positive side, AS-PMMH shows the highest multiESS per sample, as seen in Table 5.5.

When performing tests on the accuracy of the algorithm, we have run the three algorithms using 100000 likelihood evaluations for each run, and for the AS-PMMH this has meant to have 1666 outer MCMC step, 10 inner inactive variables, 6 tempering step of the SMC sampler ($1666 \times 10 \times 6 = 99960$ which is the closest integer to 100000). One way to decrease the chance of having a noisy likelihood estimate coming from the SMC sampler could be for example to increase the number of tempering steps, which is likely to reduce the variance of the estimate of the likelihood, but adding tempering steps comes at the cost of increasing the number of likelihood evaluations: if we keep the 1666 outer MCMC iterations and the 10 inactive variables, each extra tempering step will add $1666 \times 10 = 16660$ likelihood evaluations.

Another consideration is that we have relatively few MCMC iterations (1666 in this

example) to explore the Active Subspace, while reserving the bigger computational effort of the SMC sampler on the inactive part. We have devised therefore a revised version of the AS-PMMH algorithm that we have named AS-PMMH-i, which stands for *inverted* AS-PMMH, introduced in Section 5.5, it is an algorithm where we will switch roles: it will use the MCMC for the inactive and the SMC sampler for the active part.

5.5 AS-PMMH-i algorithm: giving more relevance to the Active component

We have introduce the AS-PMMH algorithm in Section 5.2. We have seen how, if we consider the state space variable

$$\theta = B_a a + B_i i \tag{5.20}$$

in AS-PMMH we get an estimate of the likelihood by marginalising out the inactive variables i through an SMC sampler, and we use the estimate in an outer MCMC on the active variable a. One of the problems of this approach is that it uses the computationally more intensive SMC sampler on the part of the state space we are interested the least, the inactive variable i. This may lead to a sub-optimal estimate of a. We have therefore devised an *inverted* algorithm, we name it AS-PMMH-i, that on the theoretical level has exactly the same framework of AS-PMMH, the difference is that in AS-PMMH-i we switch roles and we use the internal SMC sampler on the active component whereas the outer MCMC is on the inactive part: the aim is clear, and is to use the most computationally intensive part on the component of the state space we are interested in the most, i.e. the active one. The algorithms are reported below: 12 is the outer MCMC on the inactive, whereas 11 is the SMC sampler on the active variables (they are conceptually the same as the algorithms introduced in Section 5, only the roles are inverted).

Alg. 11 SMC on active variables for a given i and t.

1: Simulate N_a points, $\{a_0^n\}_{n=1}^{N_a} \sim p_a(\cdot \mid i)$ and set each weight $w_0^n = 1/N_a$;
2: **for** $s = 1 : t$ **do**
3: **for** $n = 1 : N_a$ **do** ▷ reweight
4: **if** $s = 1$ **then**
5:
$$\tilde{w}_s^n = w_{s-1}^n l_{1:s}\left(B_i i + B_a a^n\right);$$
6: **else**
7:
$$\tilde{w}_s^n = w_{s-1}^n \frac{l_{1:s}\left(B_i i + B_a a_{s-1}^n\right)}{l_{1:s-1}\left(B_i i + B_a a_{s-1}^n\right)};$$
8: **end if**
9: **end for**
10: $\{w_s^n\}_{n=1}^{N_a} \leftarrow$ normalise $\left(\{\tilde{w}_s^n\}_{n=1}^{N_a}\right)$;
11: If $s = t$, go to line 30;
12: **for** $n = 1 : N_a$ **do**
13: Simulate the index $v_{s-1}^n \sim \mathcal{M}\left(\left(w_s^1, ..., w_s^{N_a}\right)\right)$ of the ancestor of particle n;
14: **end for**
15: **if** some degeneracy condition is met **then** ▷ resample
16: **for** $n = 1 : N_a$ **do**
17: Set $a_s^n = a_{s-1}^{v_{s-1}^n}$;
18: **end for**
19: $w_s^n = 1/N_a$ for $n = 1 : N_a$;
20: **else**
21: **for** $n = 1 : N_a$ **do**
22: Set $a_s^n = a_{s-1}^n$;
23: **end for**
24: **end if**
25: **for** $n = 1 : N_a$ **do** ▷ move
26: Simulate $a_s^{n*} \sim q_{t,a}(\cdot \mid a_s^n, i)$;
27: Set $a_s^n = a_s^{n*}$ with probability
$$1 \wedge \frac{l_{1:s}\left(B_i i + B_a a_s^{n*}\right) p_a\left(a_s^{n*} \mid i\right) q_{t,a}\left(a_s^n \mid i\right)}{l_{1:s}\left(B_i i + B_a a_s^n\right) p_a\left(a_s^n \mid i\right) q_{t,a}\left(a_s^{n*} \mid i\right)},$$
28: **end for**
29: **end for**
30: Estimate $l_{t,i}(i)$ using
$$\bar{l}_{t,i}(i) = \prod_{s=1}^{t} \sum_{n=1}^{N_a} \tilde{w}_s^n.$$

Alg. 12 Inactive subspace particle marginal Metropolis-Hastings: AS-PMMH-i

1: Initialise i^0;
2: Run Algorithm 11 for $i = i^0$ and $t = T$, obtaining $\bar{l}_{T,i}(i^0)$ and weights, denoted $\left(w_T^{1,0}, ..., w_T^{N_a,0}\right)$;
3: $u^0 \sim \mathcal{M}\left(\left(w_T^{1,0}, ..., w_T^{N_i,0}\right)\right)$;
4: Let $\bar{l}_i^0 = \bar{l}_{T,i}(i^0)$;
5: **for** $m = 1 : N_i$ **do**
6: $\quad i^{*m} \sim q_i(\cdot \mid i^{m-1})$;
7: $\quad$ Run Algorithm 11 for $i = i^{*m}$ and $t = T$, obtaining $\bar{l}_{T,i}(i^{*m})$ and weights, denoted $\left(w_T^{*1,m}, ..., w_T^{*N_a,m}\right)$;
8: $\quad u^{*m} \sim \mathcal{M}\left(\left(w_T^{*1,m}, ..., w_T^{*N_a,m}\right)\right)$;
9: $\quad$ Set $\left(i^m, \{a^{n,m}, w^{n,m}\}_{n=1}^{N_a}, u^m, \bar{l}_i^m\right) = \left(i^{*m}, \{a^{*n,m}, w^{*n,m}\}_{n=1}^{N_a}, u^{*m}, \bar{l}_{T,i}(i^{*m})\right)$ with probability

$$\alpha_i^m = 1 \wedge \frac{p_i(i^{*m})\,\bar{l}_{T,i}(i^{*m})}{p_i(i^{m-1})\,\bar{l}_i^{m-1}} \frac{q_i(i^{m-1} \mid i^{*m})}{q_i(i^{*m} \mid i^{m-1})};$$

10: $\quad$ Else let $\left(i^m, \{a^{n,m}, w^{n,m}\}_{n=1}^{N_a}, u^m, \bar{l}_i^m\right) = \left(i^{m-1}, \{a^{n,m-1}, w^{n,m-1}\}_{n=1}^{N_a}, u^{m-1}, \bar{l}_i^{m-1}\right)$;
11: **end for**

5.5.1 Comparison of performances with other algorithms

We will, in this section, compare the performances of standard MCMC, AS-MH, AS-PMMH and the newly introduced AS-PMMH-i, in the 25D Banana model. We have used the following parameters in the model of equation (5.15): $b = 0.001$ (mild curvature), $c = 0$ (absence of the constant term), $H = 3$ (the last 3 components will be affected by the quadratic part of (5.15)).

Similarly to what we have done in Section 5.3.3, we will compare:

- standard MCMC Algorithm 4 performed on the full 25D space;
- AS-MH Algorithm 8;
- AS-PMMH Algorithm 10, which will perform an outer MCMC and several inner SMC samplers: for each step the algorithm will perform an SMC sampler on the 21D inactive subspace to obtain an unbiased estimate of the likelihood to be used in the outer *active* MCMC as estimate of the marginal likelihood.
- AS-PMMH-i Algorithm 12, which is similar to the AS-PMMH above, only with inverted roles: for each step the algorithm will perform an SMC sampler on the 4D active subspace to obtain an unbiased estimate of the likelihood to be used in the outer *inactive* MCMC as estimate of the marginal likelihood.

Comparison using MultiESS

The first comparison is using the multiESS [Vats et al., 2019] that we described in Section 2.4.8, we use the R software implementation of it. We have run the algorithms in one test run with 200000 likelihood evaluations for each algorithm, using no burn-in, and we used optimal covariances as per equations (5.11) in the MCMC proposal, (5.12) for AS-MH and AS-PMMH in the active marginal MCMC proposal, (5.13) for AS-PMMH-i in the inactive marginal MCMC proposal. As we explained 200000 likelihood evaluations will give a different number of output samples in each of the algorithms, look at Table D.2 for reference. All algorithms have shown an acceptance rate of around 10% in their main MCMC, except AS-PMMH-i which has shown a very high acceptance rate of around 60% in the outer inactive MCMC. Results are in Table 5.5.

Algorithm	**MultiESS**/N	**MultiESS**
MCMC	0.1	200
AS-MH	5.7	1140
AS-PMMH	15.5	516
AS-PMMH-i	73.6	2452

Table 5.6: Banana 25D multiESS out of 200000 likelihood evaluations (please see Table D.2 for the number of corresponding output samples N per algorithm).

We remind again that, by construction, AS-MH and AS-PMMH output samples are more likely to show less correlation than the standard MCMC, which in turn will cause higher multiESS. The reason of lower correlation is that even when using small steps in the active marginal MCMC, points in the state space may contain inactive parts that may be very different. The same applies to AS-PMMH-i, only with inverted roles active/inactive. Therefore the relatively high multiESS figures for AS-PMMH-i in Table 5.6 are likely to be due to the much higher acceptance rate of around 60% for AS-PMMH-i in the test run, the more the samples are accepted the higher the multiESS, as the samples will have little correlation.

Comparison of estimate of expectation

We perform the RMSE test on the 25D Banana model of Section 5.4.2. In this analysis, we concentrate on the estimates of individual components, as our primary interest lies in understanding the 'stickiness' behavior of the algorithms shown by some of the components. Focusing on component-wise RMSE allows us to better highlight how each algorithm performs across different parts of the state space. However, future work could incorporate additional distributional metrics for a more comprehensive analysis. We have run the algorithms with 100000 likelihood evaluations, which means for both AS-PMMH and AS-PMMH-i 1666 outer MCMC steps, 10 inner particles, 6 tempering step of the SMC sampler ($1666 \times 10 \times 6 = 99960$ which is the closest integer to 100000), as explained in Table 5.1 (AS-PMMH-i figures are

the same as AS-PMMH). The results in terms of distribution of the RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over 50 runs can be seen in Figure 5.12

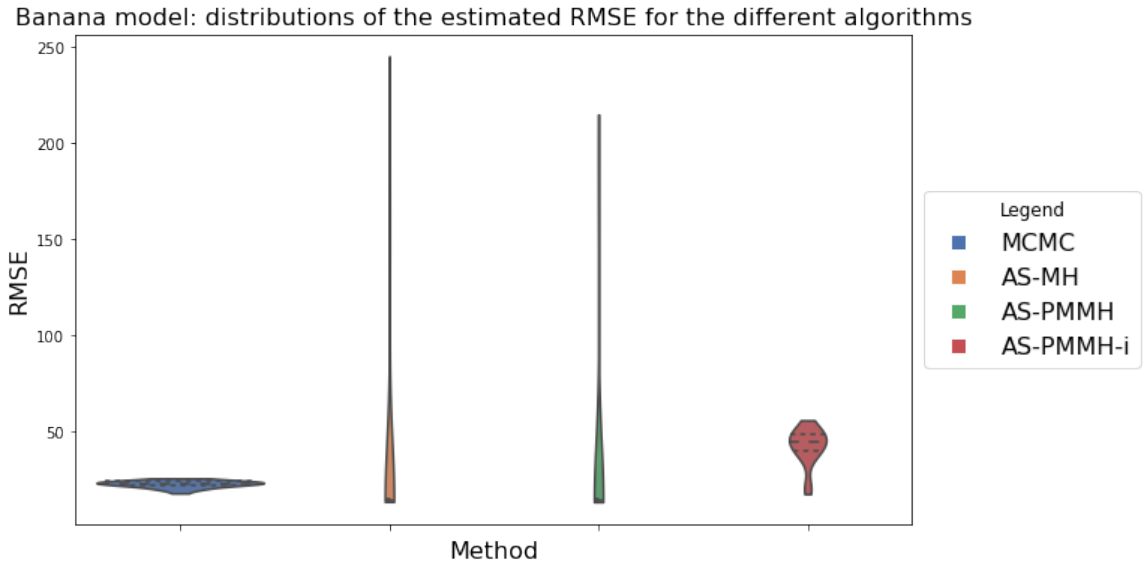

Figure 5.12: *Distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs. We see the tails of the AS-PMMH-i distribution are smaller than the AS-PMMH, probably because, keeping constant the number of tempering of the SMC sampler between the two* (6)*, the size of the space the SMC has to act upon is much smaller:* 4*D in case of AS-PMMH-i vs* 21*D in the case of AS-PMMH. By contrast, the MCMC part has to act on a much bigger space:* 21*D vs* 4*D, this probably explains why the AS-PMMH-i has a bigger average error.*

5.5.2 Review

We see in Figure 5.12 that by switching roles and concentrating the efforts on the smaller subspace the tail of the RMSE distribution for AS-PMMH-i become smaller, compared to the AS-PMMH. One of the possible reasons is that the estimate of the likelihood is probably less noisy when running on the active 4D space rather than on the inactive 21D space, but we also see that the average RMSE is higher than the AS-PMMH, one possible reason is that the outer MCMC runs on a bigger space, 21D in the AS-PMMH-i compared to the 4D in the case of AS-PMMH.

We believe that using a marginal estimate brings a *trade-off*, where by increasing the number of likelihood evaluations the quality of the approximation should generally improve, but that comes at the cost of increasing, sometimes considerably, the algorithmic cost: as said in Section 5.4.3, in the example of Section 5.5.1 if we keep the 1666 outer MCMC iterations and the 10 active variables, each additional tempering step will add $1666 \times 10 = 16660$ likelihood evaluations.

The noisiness of marginal estimates, especially when we want to keep down the number

of likelihood evaluations (and therefore the algorithmic complexity) is what triggered our research into alternative ways to estimate the Active Subspace, for example with AS-Gibbs in Section 5.6 and AS-MwPG in 5.7.

5.6 Active Subspaces Gibbs algorithm: AS-Gibbs

We will, in this section, introduce a novel approach that uses AS to construct an effective Metropolis-within-Gibbs sampler. We start by reminding what is the problem we try to address, we named it **OP2** in the list of open points in Section 4.7: we know that estimates of marginal likelihoods, like the ones used in the AS-MH Algorithm 7 or AS-PMMH Algorithm 10 or AS-PMMH-i Algorithm 12 will have a variance that will be in general greater, and also have a higher acceptance rate, than that of a marginal algorithm (i.e. where no estimate is used) [Andrieu and Vihola, 2015]. We have seen this issue in Figure 5.10, for example, where we can appreciate the long tails of the distributions of the error of algorithms using estimates of marginals. Therefore it easy to understand that an AS-based MCMC algorithm where an exact marginal was used instead of an estimated one would be an enhancement of Algorithm 7, exactly because it is likely to show lower variance and higher acceptance rate.

We propose in this section the use of a Gibbs sampler algorithm that we will apply to the AS setting. By reminding the AS partition of the state space $\theta = B_a a + B_i i$ coming from (4.4), we will apply a Gibbs algorithm firstly to the inactive variables i, then to the active a. We report below the full Gibbs AS Algorithm 13

Alg. 13 AS-Gibbs

1: Initialize $\theta^{(0)} = B_a a^{(0)} + B_i i^{(0)}$
2: **for** $t = 1$ to T **do**
3: $\quad i^* \sim p_i(\cdot|a^{(t-1)})$ ▷ propose inactive
4: $\quad$ Set $i^{(t)} = i^*$ with probability $1 \wedge \frac{p_i\left(i^*|a^{(t-1)}\right) l\left(a^{(t-1)},i^*\right) p_i\left(i|a^{(t-1)}\right)}{p_i\left(i|a^{(t-1)}\right) l\left(a^{(t-1)},i\right) p_i\left(i^*|a^{(t-1)}\right)} = \frac{l\left(a^{(t-1)},i^*\right)}{l\left(a^{(t-1)},i\right)}$
5: $\quad$ Else let $i^{(t)} = i^{(t-1)}$
6: $\quad a^* \sim q_a(\cdot|a^{(t-1)})$ ▷ propose active
7: $\quad$ Set $a^{(t)} = a^*$ with probability $1 \wedge \frac{p_a(a^*) p_i\left(i^{(t)}|a^*\right) l\left(a^*,i^t\right) q_a\left(a^{(t-1)}|a^*\right)}{p_a(a^{(t-1)}) p_i\left(i^t|a^{(t-1)}\right) l\left(a^{(t-1)},i^t\right) q_a\left(a^*|a^{(t-1)}\right)}$
8: $\quad$ Else let $a^{(t)} = a^{(t-1)}$
9: **end for**

The core idea into using the Gibbs method in AS is that if we have perfectly inactive variables then we will accept all moves on the inactive variables, in the MH ratio of line 3 in Algorithm 13. In that case the acceptance probability for the inactive part in the MH ratio would in fact be:

$$1 \wedge \frac{p_i\left(i^*\right) l\left(a^{(t-1)}, i^*\right) p_i\left(i\right)}{p_i\left(i\right) l\left(a^{(t-1)}, i\right) p_i\left(i^*\right)} = \frac{l\left(a^{(t-1)}, i^*\right)}{l\left(a^{(t-1)}, i\right)} = 1 \tag{5.21}$$

where the final step in (5.21) follows from the assumption that the likelihood function $l(a^{(t-1)}, i)$ remains invariant with respect to changes in i. Specifically, we assume that $l(a^{(t-1)}, i) = l(a^{(t-1)}, i^*)$, which allows us to simplify the expression:

$$\frac{l(a^{(t-1)}, i^*)}{l(a^{(t-1)}, i)} = 1.$$

This simplification comes from the likelihood remaining constant over the inactive variable i, enabling the cancellation of terms.

5.6.1 Comparison of performances with other algorithms

We use to compare the performances, the Banana model of Section 5.4.2 in 25D.

Comparison using MultiESS

The first comparison is using the multiESS [Vats et al., 2019] that we described in Section 2.4.8, we use the R software implementation of it. We have run the algorithms in one test run with 200000 likelihood evaluations for each algorithm, using no burn-in, and we used optimal covariances as per equations (5.11) in the MCMC proposal, (5.12) for AS-MH, AS-PMMH and AS-Gibbs in the active marginal MCMC proposal, (5.13) for AS-PMMH-i in the inactive marginal MCMC proposal. As we explained 200000 likelihood evaluations will give a different number of output samples in each of the algorithms, look at Table D.2 for reference. All algorithms have shown an acceptance rate of around 10% in their main MCMC, except AS-PMMH-i which has shown a very high acceptance rate of around 60% in the outer inactive MCMC. Results are in Table 5.5.

Algorithm	**MultiESS/N**	**MultiESS**
MCMC	0.1	200
AS-MH	5.7	1140
AS-PMMH	15.5	516
AS-PMMH-i	73.6	2452
AS-Gibbs	63.7	63700

Table 5.7: Banana 25D multiESS out of 200000 likelihood evaluations (please see Table D.2 for the number of corresponding output samples N per algorithm).

We have explained earlier, for example in the multiESS section of 5.5.1, the reason why, by construction the samples in AS-MH, AS-PMMH and AS-PMMH-i are likely to be less correlated, and therefore to show higher MultiESS/N, please refer to the section for explanation.

In AS-Gibbs too, by construction, chain samples are likely to show little correlation, and therefore have high MultiESS/N, as we can appreciate in Table 5.7, especially as we get near perfect Active Subspaces. One of the reasons is that, by looking at line 4 of AS-Gibbs,

in case of near perfect Active Subspaces it becomes like in equation (5.21), so it is almost always accepted, which means that consecutive θ samples of the state space in the chain will in general have different inactive part, even if small steps are taken in the active MH of line 7, and will therefore tend to have little correlation and show high multiESS.

We can appreciate in Table 5.7 that the AS-Gibbs is a clear winner as it shows a much bigger multiESS: the AS-Gibbs, in terms of multiESS is a much more efficient algorithm than the other listed, since with the same number of likelihood evaluations it brings a much higher effective sample size.

Comparison of estimate of expectation

To compare the estimates of the mean coming from the algorithm, we have performed 50 runs of each algorithm, with 100000 likelihood evaluation in each run, and we have measured the Root Mean Squared Error (RMSE) of the difference of the true mean and the mean estimated by each of the algorithms. As a reminder, 100000 likelihood evaluations will result in different number of output samples per each algorithm, please refer to Table D.1. We report the chart with the violin plots showing mean and upper and lower quartile of the distribution of the RMSE.

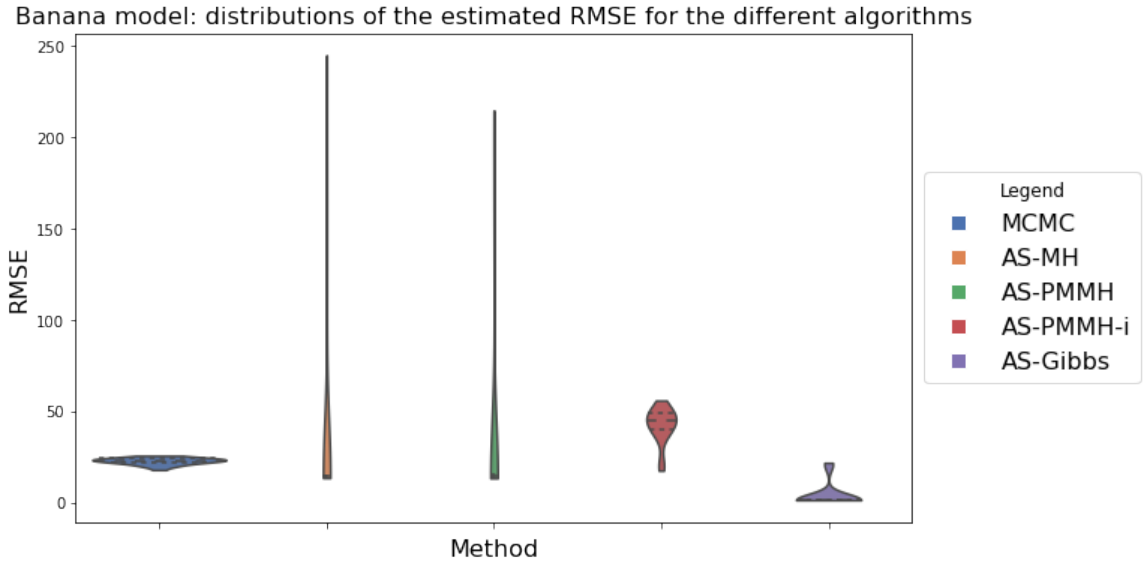

Figure 5.13: *Distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs. We can see from the chart that AS-Gibbs has lower mean RMSE.*

5.6.2 Review

We have introduced a novel way of performing AS-MCMC on Active Subspaces that uses Gibbs sampling, we named it AS-Gibbs. We have shown on the Banana model that it performs better than the algorithms discussed in this section: the standard MCMC, the AS-

MH Algorithm 8, the AS-PMMH Algorithm 10 and the AS-PMMH-i Algorithm 12. It has in fact a much higher multiESS (see Table 5.7), and the distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms shows lower mean and lower upper quartile than all the others (see Figure 5.13).

We expect the Gibbs Algorithm 13 to perform better than the standard MCMC Algorithm 4 in case of perfectly Active Subspace (so eigenvalues of equation (4.9) all equal to zero). In fact, considering that the inactive proposal is always accepted (see equation 5.21), algorithm 13 is *de facto* an MCMC acting on a marginal of the posterior on a sub-dimension of the space that is $d_a < d$, with d_a the dimension of the active component and d the dimension of the full space.

We also expect AS-Gibbs to outperform, in general, algorithms that use an estimate of the marginal likelihood, like AS-MH, AS-PMMH and AS-PMMH-i, in cases of perfect Active Subspaces: the reason is that MCMC chains that use estimates of marginal likelihoods (like AS-MH, AS-PMMH and AS-PMMH-i) will in general be noisier, have higher variance and lower acceptance rate than those that use exact marginals [Andrieu and Vihola, 2015]: in summary, we expect Algorithm 13 to perform better because it uses an exact marginal, whereas Algorithm 8 uses an estimate of the marginal likelihood [Andrieu and Vihola, 2015].

If applied on the earlier examples, such as the Gaussian model discussed in Section 5.3.2, AS-Gibbs is expected to perform similarly to traditional MCMC. The reason is that, in a $25D$ Gaussian model, the reduction brought by AS-Gibbs is *de facto* a MCMC performed on a $24D$ space instead of the original $25D$ space. However, this reduction comes with the additional computational overhead for the calculation of the Active Subspace and of the structural matrices, described in section 4.2. The advantage of using AS-Gibbs compared to, for example, traditional MCMC becomes more apparent as the dimension of the Active Subspace increases. For instance, in the Banana model analysed in this section (with the Active Subspace having dimension 4).

5.7 Active Subspace Metropolis within particle Gibbs algorithms: AS-MwPG and AS-MwPG-i

We have introduced in Section 5.6 with AS-Gibbs a way of bringing improvements to AS-based MCMC methods in cases where either we are in the presence of perfect active subspaces and we have independence between active and inactive components. We have seen how the AS-Gibbs brings improvements compared to the standard MCMC, where the potential *curse of dimensionality* problem is reduced by exploiting the separation of active and inactive subspaces in the Gibbs update, and we have seen how AS-Gibbs performs better than algorithms where an estimate of the marginal likelihood is use [Andrieu and Vihola, 2015].

We now propose an additional algorithm based on the application of the Metropolis within Particle Gibbs (MwPG) [Andrieu et al., 2010] method to Active Subspaces and we name the

algorithm AS-MwPG. We explain the rationale behind: in cases where the active and inactive are independent (and so the right case to use AS-Gibbs), but the inactive subspace is complex to explore, it could be difficult to find a good proposal for the inactive part in the AS-Gibbs (line 4 of the AS-Gibbs algorithm): the inactive subspace could be for example multimodal, in which case using an SMC sampler can perform better since the SMC transitions smoothly from a starting distribution to the posterior.

With this in mind, we have devised the AS-MwPG. We have discussed the MwPG in Section 2.8.2. The fundamental idea takes the root from what we have done in the AS-Gibbs, i.e. exploit the cases where there is independence between active and inactive component: in addition, the AS-MwPG will ease the problems in cases where the inactive part is challenging to explore. The AS-MwPG will draw from the inactive part using a inner SMC sampler embedded in an outer MCMC performed on the active component. The *inactive* SMC will be conditioned on a path of the inactives that will "survive" all resampling, as we explained in 2.8.2, and that is the part that plays the role of the conditioning in this *extended* Gibbs algorithm. The AS-MwPG algorithm is performed by using Algorithms 15 (outer active MCMC) and 14 (inner inactive conditioned SMC). As a note, in the below Algorithms 15 and 14, $l(\theta)$ is the likelihood, $l_s(\theta)$ is as per equation (5.22)

$$l_s(\theta) = l(\theta)^{\eta_t},\ \eta_t \in [0, 1] \tag{5.22}$$

where η_t is the tempering exponent (see Section 2.6 for details on the tempering)

$$l_{1:t}(\theta) = \prod_{s=1}^{t} l_s(\theta) = l(\theta)^{\eta_t} \tag{5.23}$$

Alg. 14 Conditional SMC on active variables for a given a, $i_{0:T}^1$, $\tilde{w}_{0:T}^1$ and t.

Simulate $N_i - 1$ points, $\{i_0^n\}_{n=2}^{N_i} \sim p_i(\cdot \mid a)$ and set each weight $w_0^n = 1/N_a$;
for $s = 1 : t$ **do**
  for (**do**$n = 2 : N_i$) ▷ reweight
    if $s = 1$ **then**

$$\tilde{w}_s^n = w_{s-1}^n l_{1:s}\left(B_a a + B_i i^n\right);$$

    else

$$\tilde{w}_s^n = w_{s-1}^n \frac{l_{1:s}\left(B_a a + B_i i_{s-1}^n\right)}{l_{1:s-1}\left(B_a i + B_i i_{s-1}^n\right)};$$

    end if
  end for
  $\{w_s^n\}_{n=1}^{N_i} \leftarrow$ normalise $\left(\{\tilde{w}_s^n\}_{n=1}^{N_i}\right)$;
  If $s = t$, terminate the algorithm;
  for $n = 2 : N_i$ **do** Simulate the index $v_{s-1}^n \sim \mathcal{M}\left(\left(w_s^1, ..., w_s^{N_i}\right)\right)$ of the ancestor of particle n;
  end for
  if (**then** some degeneracy condition is met) ▷ resample
    for $n = 2 : N_i$ **do** Set $i_s^n = i_{s-1}^{v_{s-1}^n}$;
    end for$w_s^n = 1/N_a$ for $n = 1 : N_i$;
  else
    for $n = 2 : N_i$ **do** Set $i_s^n = i_{s-1}^n$;
    end for
  end if
  for (**do**$n = 2 : N_i$) ▷ move
    Simulate $i_s^{n*} \sim q_{t,i}(\cdot \mid i_s^n, a)$;
    Set $i_s^n = i_s^{n*}$ with probability

$$1 \wedge \frac{l_{1:s}\left(B_a a + B_i i_s^{n*}\right) p_i\left(i_s^{n*} \mid a\right) q_{t,i}\left(i_s^n \mid a\right)}{l_{1:s}\left(B_a a + B_i i_s^n\right) p_i\left(i_s^n \mid a\right) q_{t,i}\left(i_s^{n*} \mid a\right)},$$

  end for
end for

Alg. 15 Active Subspace Metropolis within particle Gibbs

Initialise a^0;
Initialise $i_t^{1,0}$ for $t = 0 : T$;
for $m = 1 : N_i$ **do**
    $a^{*m} \sim q_a(\cdot \mid a^{m-1})$;
    Set $a^m = a^{*m}$ with probability

$$\alpha_a^m = 1 \wedge \frac{p_a(a^{*m})\, l_{1:T}\left(B_a a^{*m} + B_i i_T^{m-1,1}\right)}{p_a(a^m)\, l_{1:T}\left(B_a a^{m-1} + B_i i_T^{m-1,1}\right)};$$

    Else let $a^m = a^{m-1}$;
    Run Algorithm 14 for $a = a^m$, $i_{0:T}^1 = i_{0:T}^{1,m-1}$, $\tilde{w}_{0:T}^{1,m-1}$ and $t = T$, obtaining points $\left(i_{0:T}^{1,m}, ..., i_{0:T}^{N_i,m}\right)$ and unnormalised and normalised weights $\left(\tilde{w}_T^{1,m}, ..., \tilde{w}_T^{N_i,m}\right)$ and $\left(w_T^{1,m}, ..., w_T^{N_i,m}\right)$;
    $u^m \sim \mathcal{M}\left(\left(w_T^{1,m}, ..., w_T^{N_i,m}\right)\right)$;
    Set $i_{0:T}^{1,m} = i_{0:T}^{u^m,m}$;
    Set $\tilde{w}_{0:T}^{1,m} = \tilde{w}_{0:T}^{u^m,m}$;
    Set $w_{0:T}^{1,m} = w_{0:T}^{u^m,m}$;
end for

5.7.1 Inverted Active Subspace Metropolis within particle Gibbs: AS-MwPG-i

The *inverted* version of the AS-MwPG algorithm, which we have named AS-MwPG-i shares the very same theoretical framework of Algorithms 14 and 15 (which together make up the AS-MwPG) therefore no further explanation is necessary on the theory. The aim of the AS-MwPG-i is to use the inner SMC sampler on the Active Subspace, and therefore on the part of the space we are interested the most, whereas the AS-MwPG was using the SMC on the inactive part. The AS-MwPG-i algorithms are below

Alg. 16 Conditional SMC on active variables for a given i, $a_{0:T}^1$, $\tilde{w}_{0:T}^1$ and t.

Simulate $N_a - 1$ points, $\{a_0^n\}_{n=2}^{N_a} \sim p_a(\cdot \mid i)$ and set each weight $w_0^n = 1/N_a$;
for $s = 1 : t$ **do**
  for (**do**$n = 2 : N_a$) ▷ reweight
    if $s = 1$ **then**

$$\tilde{w}_s^n = w_{s-1}^n l_{1:s}\left(B_i i + B_a a^n\right);$$

    else

$$\tilde{w}_s^n = w_{s-1}^n \frac{l_{1:s}\left(B_i i + B_a a_{s-1}^n\right)}{l_{1:s-1}\left(B_i i + B_a a_{s-1}^n\right)};$$

    end if
  end for
  $\{w_s^n\}_{n=1}^{N_a} \leftarrow$ normalise $\left(\{\tilde{w}_s^n\}_{n=1}^{N_a}\right)$;
  If $s = t$, terminate the algorithm;
  for $n = 2 : N_a$ **do** Simulate the index $v_{s-1}^n \sim \mathcal{M}\left(\left(w_s^1, ..., w_s^{N_a}\right)\right)$ of the ancestor of particle n;
  end for
  if (**then** some degeneracy condition is met) ▷ resample
    for $n = 2 : N_a$ **do** Set $a_s^n = a_{s-1}^{v_{s-1}^n}$;
    end for$w_s^n = 1/N_a$ for $n = 1 : N_a$;
  else
    for $n = 2 : N_a$ **do** Set $a_s^n = a_{s-1}^n$;
    end for
  end if
  for (**do**$n = 2 : N_a$) ▷ move
    Simulate $a_s^{n*} \sim q_{t,a}(\cdot \mid a_s^n, i)$;
    Set $a_s^n = a_s^{n*}$ with probability

$$1 \wedge \frac{l_{1:s}\left(B_i i + B_a a_s^{n*}\right) p_a\left(a_s^{n*} \mid i\right) q_{t,a}\left(a_s^n \mid i\right)}{l_{1:s}\left(B_i i + B_a a_s^n\right) p_a\left(a_s^n \mid i\right) q_{t,a}\left(a_s^{n*} \mid i\right)},$$

  end for
end for

Alg. 17 Inactive subspace Metropolis within particle Gibbs

1: Initialise i^0;
2: Initialise $a_t^{1,0}$ for $t = 0 : T$;
3: **for** $m = 1 : N_i$ **do**
4: $i^{*m} \sim q_i\left(\cdot \mid i^{m-1}\right)$;
5: Set $i^m = i^{*m}$ with probability

$$\alpha_i^m = 1 \wedge \frac{p_i\left(i^{*m}\right) l_{1:T}\left(B_i i^{*m} + B_a a_T^{m-1,1}\right)}{p_i\left(i^m\right) l_{1:T}\left(B_i i^{m-1} + B_a a_T^{m-1,1}\right)};$$

6: Else let $i^m = i^{m-1}$;
7: Run Algorithm 16 for $i = i^m$, $a_{0:T}^1 = a_{0:T}^{1,m-1}$, $\tilde{w}_{0:T}^{1,m-1}$ and $t = T$, obtaining points $\left(a_{0:T}^{1,m}, ..., a_{0:T}^{N_a,m}\right)$ and unnormalised and normalised weights $\left(\tilde{w}_T^{1,m}, ..., \tilde{w}_T^{N_a,m}\right)$ and $\left(w_T^{1,m}, ..., w_T^{N_a,m}\right)$;
8: $u^m \sim \mathcal{M}\left(\left(w_T^{1,m}, ..., w_T^{N_a,m}\right)\right)$;
9: Set $a_{0:T}^{1,m} = a_{0:T}^{u^m,m}$;
10: Set $\tilde{w}_{0:T}^{1,m} = \tilde{w}_{0:T}^{u^m,m}$;
11: Set $w_{0:T}^{1,m} = w_{0:T}^{u^m,m}$;
12: **end for**

5.7.2 Review

We have introduced two versions of a novel method that are based on the application of the Metropolis within Particle Gibbs (MwPG) [Andrieu et al., 2010] to Active Subspaces and we named the algorithms AS-MwPG (when the internal SMC sampler is used to sample the *inactive* variables) and AS-MwPG-i (when the internal SMC sampler is used to sample the *active* variables). Some of the conditions where we expect the novel algorithms to perform at their best are similar to those of AS-Gibbs introduced in Section 5.6 (i.e. that the conditional distributions are easy to draw from, or in case for perfect AS).

We expect the use of AS-MwPG-i to be prominent with respect to AS-MwPG, as it dedicates most of the computational power to the Active Subspace. We expect the AS-MwPG-i to perform better than AS-Gibbs in those cases where the Active Subspace is complex to explore (for example multimodal). We will see the AS-MwPG-i in action on one such example in Section 5.8.

5.8 Comparison of performances of AS-based MCMC algorithms in a multi-modal distribution

We show how the AS-MwPG-i will allow us to perform correctly the MC analysis of a multimodal posterior where other algorithms presented in this chapter so far would fail. We use the Gaussian mixture model of Section 5.8.1.

5.8.1 Gaussian mixture model

Gaussian mixture models are a class of multimodal distributions that are often used for example in approximating complex distributions [Reynolds, 2009]. We use here a mixture of two Gaussians, in a 4D space. For a single observation y, the likelihood of the model has the form:

$$l(\theta) = \frac{1}{2}\mathcal{N}(y|\theta_1 + \theta_2, 1) + \frac{1}{2}\mathcal{N}(y|\theta_3 + \theta_4, 1) \tag{5.24}$$

We see in (5.24) that with the state space parameter $\theta = (\theta_1, \theta_2, \theta_3, \theta_4)$, the sum of the first two elements $\theta_1 + \theta_2$ is used to estimate the mean of the first Gaussian, and the sum $\theta_3 + \theta_4$ is used to estimate the mean of the second Gaussian. The system is clearly over-parametrised since we would actually strictly only need two parameters in the state space, one for the estimate of the mean of each Gaussian. When we have multiple observations, say P, in a vector $y = (y_1, y_2, ..., y_n)$, considering independence of observations, the likelihood (5.24) becomes:

$$l(\theta) = \prod_{i=1}^{P} \left(\frac{1}{2}\mathcal{N}(y_i|\theta_1 + \theta_2, 1) + \frac{1}{2}\mathcal{N}(y_i|\theta_3 + \theta_4, 1) \right) \tag{5.25}$$

Prior

We will use a prior given by a Gaussian centered at 0 and with variance 25 in all directions:

$$p(\theta) = \prod_{i=1}^{n} \mathcal{N}(0, \sigma^2) \tag{5.26}$$

where the variance $\sigma^2 = 25$.

Gradient of log-likelihood

The log-likelihood becomes:

$$\log l(\theta) = \sum_{i=1}^{P} \log \left(\frac{1}{2}\mathcal{N}(y_i|\theta_1 + \theta_2, 1) + \frac{1}{2}\mathcal{N}(y_i|\theta_3 + \theta_4, 1) \right) \tag{5.27}$$

The derivative of the log likelihood with respect to θ_1:

$$\frac{\delta(\log l(\theta))}{\delta\theta_1} = \sum_{i=1}^{P} \frac{1}{\left(\frac{1}{2}\mathcal{N}(y_i|\theta_1 + \theta_2, 1) + \frac{1}{2}\mathcal{N}(y_i|\theta_3 + \theta_4, 1)\right)} \frac{1}{2} \frac{\delta}{\delta\theta_1} \mathcal{N}(y_i|\theta_1 + \theta_2, 1) \tag{5.28}$$

We have

$$\begin{aligned} \frac{\delta}{\delta\theta_1}\mathcal{N}(y_i|\theta_1 + \theta_2, 1) &= \frac{\partial}{\partial\theta_1}\left[\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y_i - (\theta_1 + \theta_2))^2}{2}\right)\right] \\ &= (y_i - (\theta_1 + \theta_2))\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y_i - (\theta_1 + \theta_2))^2}{2}\right) \\ &= (y_i - (\theta_1 + \theta_2))\mathcal{N}(y_i|\theta_1 + \theta_2, 1) \end{aligned} \tag{5.29}$$

Considering that we are interested in the gradient of the negative log-likelihood, combining (5.28) and (5.29) we have

$$\frac{\delta(-\log l(\theta))}{\delta\theta_1} = \sum_{i=1}^{P} \frac{\frac{1}{2}\mathcal{N}(y_i|\theta_1+\theta_2,1)}{\left(\frac{1}{2}\mathcal{N}(y_i|\theta_1+\theta_2,1)+\frac{1}{2}\mathcal{N}(y_i|\theta_3+\theta_4,1)\right)}(\theta_1+\theta_2-y_i) \tag{5.30}$$

To generalise equation (5.30), it easy to check that, with likelihood given in (5.25), for $\mathbf{i} = \mathbf{1}, \mathbf{2}$ the components of the gradient of the negative log-likelihood are

$$\frac{\delta(-\log l(\theta))}{\delta\theta_i} = \sum_{i=1}^{P} \frac{\frac{1}{2}\mathcal{N}(y_i|\theta_1+\theta_2,1)}{\left(\frac{1}{2}\mathcal{N}(y_i|\theta_1+\theta_2,1)+\frac{1}{2}\mathcal{N}(y_i|\theta_3+\theta_4,1)\right)}(\theta_1+\theta_2-y_i),\ \text{i=1,2} \tag{5.31}$$

Whereas for $\mathbf{i} = \mathbf{3}, \mathbf{4}$ the components of the gradient are

$$\frac{\delta(-\log l(\theta))}{\delta\theta_i} = \sum_{i=1}^{P} \frac{\frac{1}{2}\mathcal{N}(y_i|\theta_3+\theta_4,1)}{\left(\frac{1}{2}\mathcal{N}(y_i|\theta_1+\theta_2,1)+\frac{1}{2}\mathcal{N}(y_i|\theta_3+\theta_4,1)\right)}(\theta_3+\theta_4-y_i),\ \text{i=3,4} \tag{5.32}$$

Considering that the y_i are noisy observations of the mean of either of the two Gaussians, we understand from equations (5.31) and (5.32) that the surface level of the log likelihood are those that have either

$$\theta_1+\theta_2 = \mu_1 \text{ and } \theta_3+\theta_4 = \mu_2 \tag{5.33}$$

or

$$\theta_1+\theta_2 = \mu_2 \text{ and } \theta_3+\theta_4 = \mu_1 \tag{5.34}$$

Where μ_1 and μ_2 are respectively the means of each of the two Gaussians. We show this visually in next subsection.

Visual representation

In a realization of the Gaussian mixture model obtained by generating synthetic data from the underlying model

$$\frac{1}{2}\mathcal{N}(-5,1)+\frac{1}{2}\mathcal{N}(5,1) \tag{5.35}$$

We show what we said earlier at the beginning of Section 5.8.1, specifically in equations (5.33) and (5.34) we expect the following combinations to leave the likelihood invariant:

- $\theta_1+\theta_2 = -5$ and $\theta_3+\theta_4 = 5$
- $\theta_1+\theta_2 = 5$ and $\theta_3+\theta_4 = -5$

After generating noisy synthetic data from (5.35), in order to visualize the posterior, we have run a SMC with 1M particles with likelihood of the form of equation (5.25) and Figure 5.14 shows the output combinations that leave the likelihood invariant, which, as expected, are $\theta_1+\theta_2 = -5$ and $\theta_3+\theta_4 = 5$ or vice-versa

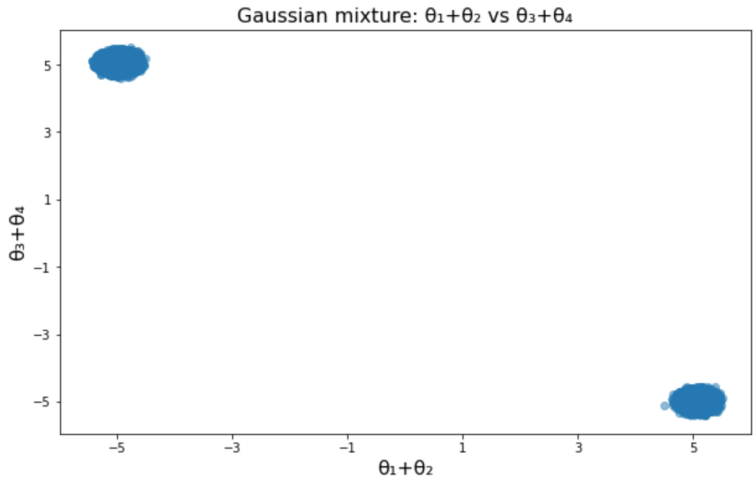

Figure 5.14: *Level surface of the system of equation* (5.35)*, the combinations* $\theta_1 + \theta_2 = -5$ *and* $\theta_3 + \theta_4 = 5$ *or* $\theta_1 + \theta_2 = 5$ *and* $\theta_3 + \theta_4 = -5$ *are the ones that leave the likelihood* (5.25) *invariant*

In the following scatter-plot, as a confirmation of what we said, we see that for the couple of parameter θ_1 and θ_2, the expected combinations are the ones that bring $\theta_1 + \theta_2 = \pm 5$ (similar results hold for the other couple θ_3 and θ_4).

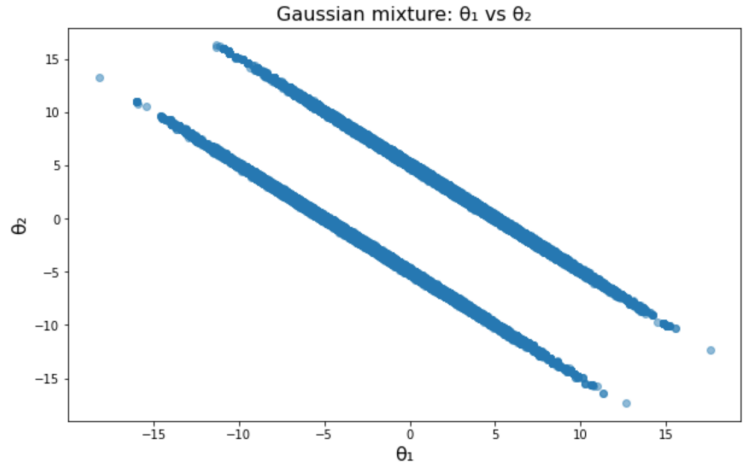

Figure 5.15: *Level surface of the system of equation* (5.35)*: the combination of parameters* $\theta_1 + \theta_2 = \pm 5$ *are the ones that leave the likelihood* (5.25) *invariant*

5.8.2 Results

We have used for the likelihood a realization of the Gaussian mixture model obtained by generating synthetic data from the same underlying model of equation (5.35), we know therefore that the following combinations will leave it invariant (see also Figure 5.14):

- $\theta_1 + \theta_2 = -5$ and $\theta_3 + \theta_4 = 5$

- $\theta_1 + \theta_2 = 5$ and $\theta_3 + \theta_4 = -5$

To illustrate that the SMC is able to explore both modes of the target without a carefully chosen proposal, we have used for all algorithms a covariance matrix that is considerably smaller than the optimal covariance [Roberts and Rosenthal, 2001], for the proposals both of the standard MCMC and of the MCMC part in AS-Gibbs, AS-MH, AS-MwPG-i. This setting will allow us to see if any method, even with a covariance of the proposal that is technically very challenging, is capable of reconstructing the bi-modal posterior correctly without remaining stuck in one of the modes.

We have run standard MCMC Algorithm 4, AS-MH Algorithm 8, AS-Gibbs Algorithm 13 and AS-MwPG-i Algorithm 15, for each method we have used 440000 likelihood evaluations to have fair comparisons. Metropolis within Particle Gibbs AS-MwPG-i is the only algorithm that has been able to correctly reconstruct the posterior, as we can appreciate from Figure 5.16 and 5.17. Which highlights the strengths of the AS-MwPG-i algorithm: even with a covariance of the proposal that is technically "challenging", it is the only algorithm that is capable of reconstructing the bi-modal posterior correctly.

Results AS-MwPG-i

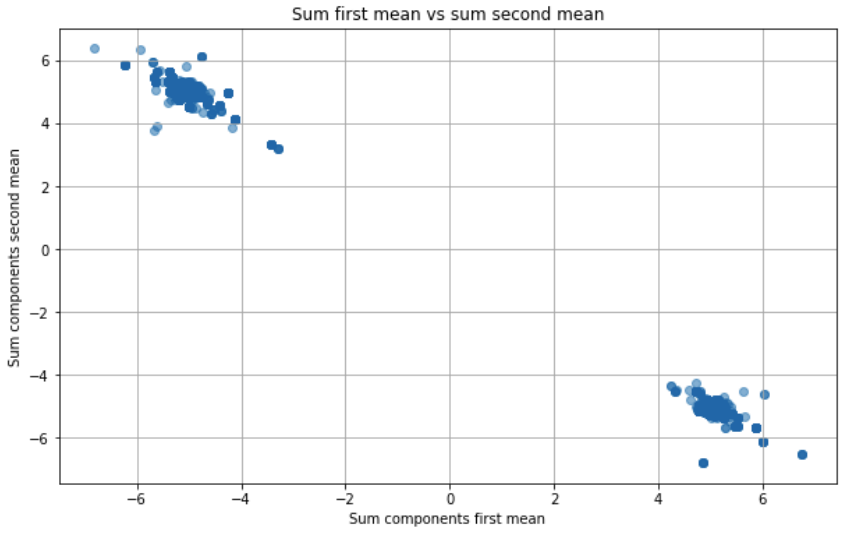

Figure 5.16: ***AS-MwPG-i**: reconstruction of the posterior for the system having likelihood (5.35). We see that AS-MwPG-i correctly reconstructs the bimodal posterior, both modes ($\theta_1 + \theta_2 = -5$ and $\theta_3 + \theta_4 = 5$) and ($\theta_1 + \theta_2 = 5$ and $\theta_3 + \theta_4 = -5$) are found (Note: in the figure "Sum components first mean" on the x-axis is the sum $\mu_1 = \theta_1 + \theta_2$, whereas "Sum components second mean" on the y-axis is the sum $\mu_2 = \theta_3 + \theta_4$).*

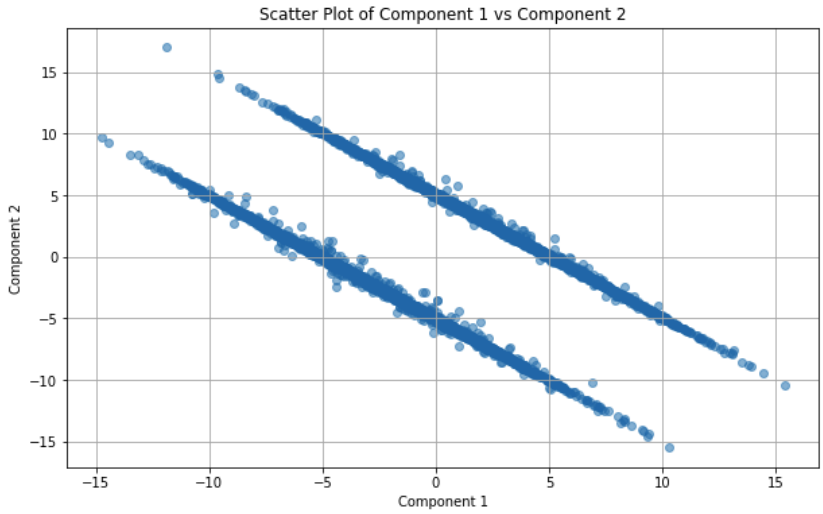

Figure 5.17: ***AS-MwPG-i****: reconstruction of the posterior for the system having likelihood (5.35). We see in the figure θ_1 (named Component 1 in the figure) vs θ_2 (named Component 2): we can appreciate that MwPG correctly reconstructs the bimodal posterior, in fact both combinations $\theta_1 + \theta_2 = -5$ and $\theta_1 + \theta_2 = 5$ are found (Note: in the figure "Component 1" on the x-axis is θ_1, whereas "Component 2" on the y-axis is θ_2).*

Results standard MCMC

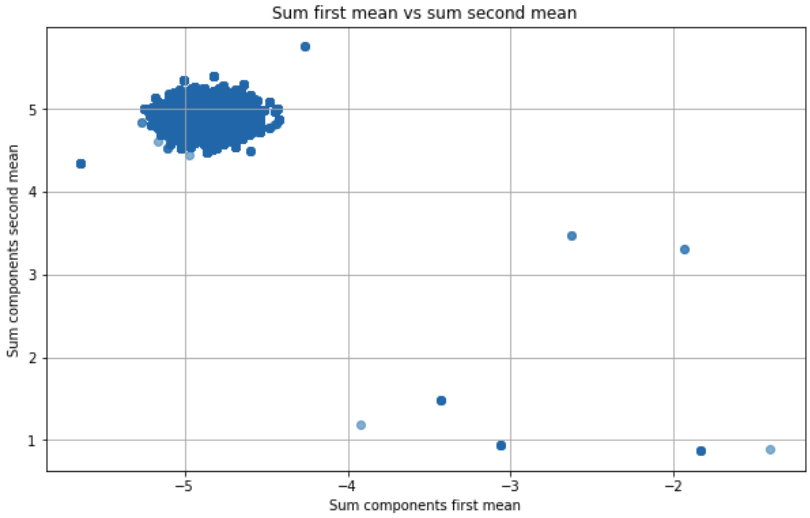

Figure 5.18: ***Standard MCMC****: incorrect reconstruction of the posterior for the system having likelihood (5.35). We see that MCMC incorrectly reconstructs the bimodal posterior: only the mode ($\theta_1 + \theta_2 = -5$ and $\theta_3 + \theta_4 = 5$) is found, whereas the mode ($\theta_1 + \theta_2 = 5$ and $\theta_3 + \theta_4 = -5$) is missing, see comparison with Figure 5.16. (Note: in the figure "Sum components first mean" on the x-axis is the sum $\mu_1 = \theta_1 + \theta_2$, whereas "Sum components second mean" on the y-axis is the sum $\mu_2 = \theta_3 + \theta_4$).*

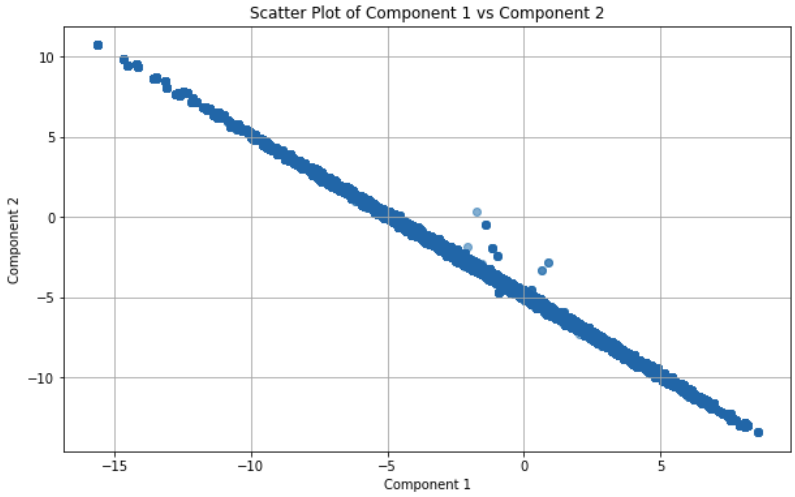

Figure 5.19: ***Standard MCMC**: incorrect reconstruction of the posterior for the system having likelihood (5.35). We see in the figure θ_1 (named Component 1 in the figure) vs θ_2 (named Component 2): we can appreciate that MCMC gets stuck in one mode and only the combination $\theta_1+\theta_2=-5$ is found, while the mode $\theta_1+\theta_2=5$ is missing, see comparison with Figure 5.17 (Note: in the figure "Component 1" on the x-axis is θ_1, whereas "Component 2" on the y-axis is θ_2).*

Results AS-MH

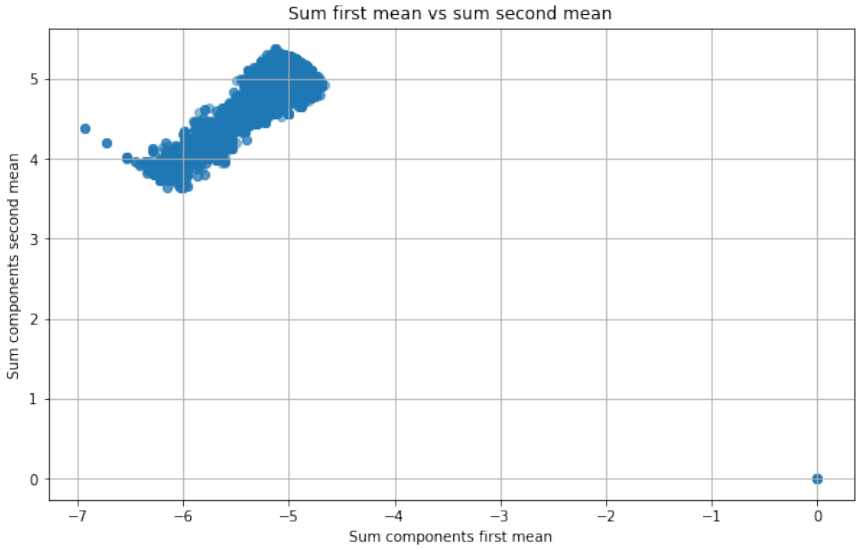

Figure 5.20: ***AS-MH** algorithm 8 of Section 4.8.2: incorrect reconstruction of the posterior for the system having likelihood (5.35). We see that AS-MH algorithm incorrectly reconstructs the bimodal posterior: only the mode ($\theta_1+\theta_2=-5$ and $\theta_3+\theta_4=5$) is found, whereas the mode ($\theta_1+\theta_2=5$ and $\theta_3+\theta_4=-5$) is missing, see comparison with Figure 5.16. (Note: in the figure "Sum components first mean" on the x-axis is the sum $\mu_1=\theta_1+\theta_2$, whereas "Sum components second mean" on the y-axis is the sum $\mu_2=\theta_3+\theta_4$).*

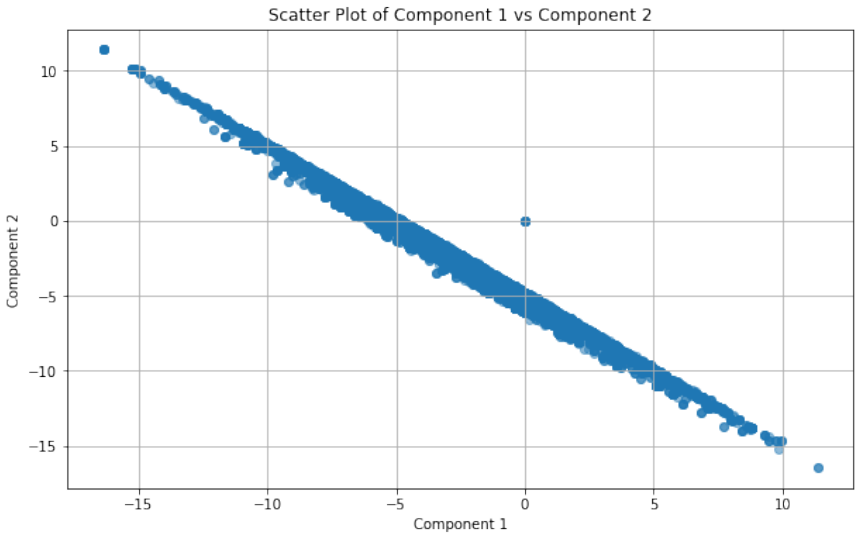

Figure 5.21: ***AS-MH*** *algorithm 8 of Section 4.8.2: incorrect reconstruction of the posterior for the system having likelihood (5.35). We see in the figure θ_1 (named Component 1 in the figure) vs θ_2 (named Component 2): we can appreciate that AS-Gibbs gets stuck in one mode and only the combination $\theta_1 + \theta_2 = -5$ is found, while the mode $\theta_1 + \theta_2 = 5$ is missing, see comparison with Figure 5.17 (Note: in the figure "Component 1" on the x-axis is θ_1, whereas "Component 2" on the y-axis is θ_2).*

Results AS-Gibbs

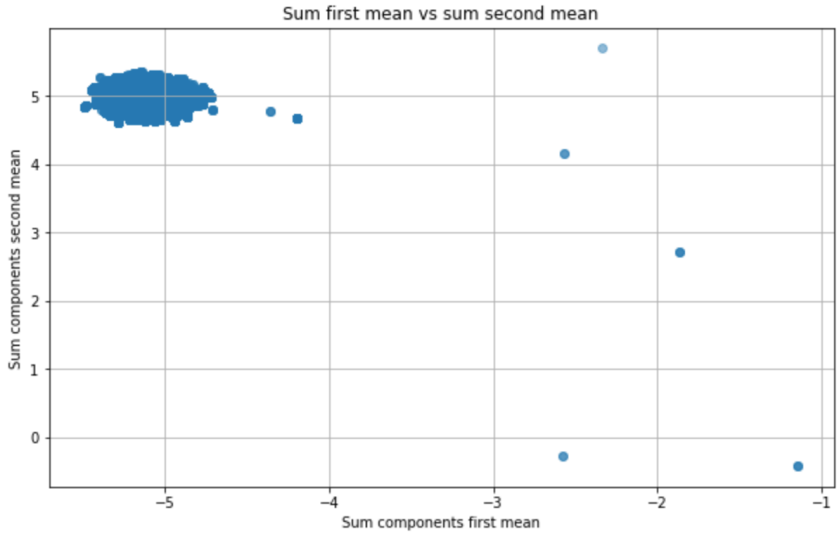

Figure 5.22: ***AS-Gibbs*** *algorithm 13 of Section 5.6: incorrect reconstruction of the posterior for the system having likelihood (5.35). We see that AS-Gibbs algorithm incorrectly reconstructs the bimodal posterior: only the mode ($\theta_1 + \theta_2 = -5$ and $\theta_3 + \theta_4 = 5$) is found, whereas the mode ($\theta_1 + \theta_2 = 5$ and $\theta_3 + \theta_4 = -5$) is missing, see comparison with Figure 5.16. (Note: in the figure "Sum components first mean" on the x-axis is the sum $\mu_1 = \theta_1 + \theta_2$, whereas "Sum components second mean" on the y-axis is the sum $\mu_2 = \theta_3 + \theta_4$).*

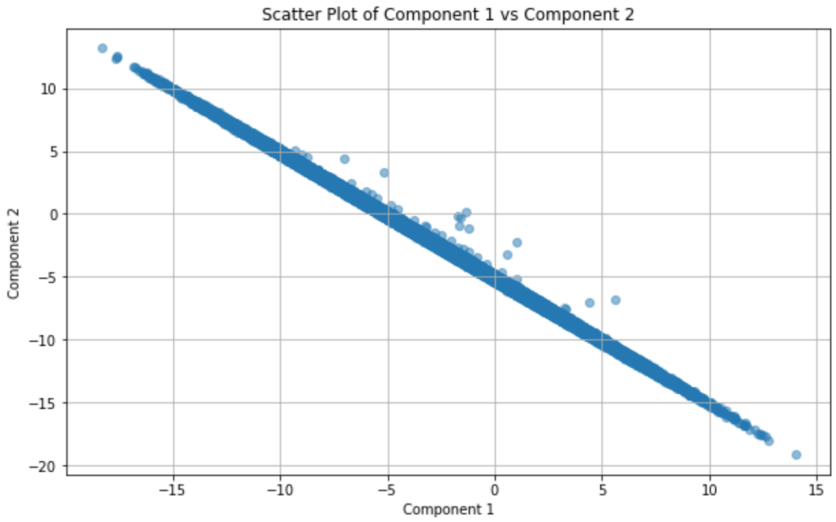

Figure 5.23: ***AS-Gibbs*** *algorithm 13 of Section 5.6: incorrect reconstruction of the posterior for the system having likelihood (5.35). We see in the figure θ_1 (named Component 1 in the figure) vs θ_2 (named Component 2): we can appreciate that AS-Gibbs gets stuck in one mode and only the combination $\theta_1 + \theta_2 = -5$ is found, while the mode $\theta_1 + \theta_2 = 5$ is missing, see comparison with Figure 5.17 (Note: in the figure "Component 1" on the x-axis is θ_1, whereas "Component 2" on the y-axis is θ_2).*

5.9 Determining the dimension of Active Subspaces with ESS

We propose a new, alternative method to determine the size of Active Subspaces that uses the ESS (we spoke about ESS in Section 2.3.2). The basic idea is that since the inactive subspace in ideal (or close-to-ideal) cases is not informed (or very little informed) by the likelihood, the prior will be a good importance proposal for the inactive subspace. Remembering that, in Section 2.3, we spoke about the optimal proposal for Importance Sampling being proportional to the posterior we see that, fixing an active variable a^1 arbitrarily within the posterior set, we have that the importance ratio of (5.36) brings a constant

$$IS = \frac{p_a(a^1)p_i(i|a^1)l(a^1,i)}{p_i(i|a^1)} = \frac{p_a(a^1)p_i(i)l(a^1)}{p_i(i)} = p_a(a^1)l(a^1)\ \forall i \sim p_i(\cdot) \tag{5.36}$$

the simplifications in equation (5.36) come from the fact that in case of inactive variables the likelihood does not depend on i $l(a,i) = l(a)$ and active and inactive are independent $p_i(i|a) = p_i(i)$.

The novel proposed approach determines the dimension of the inactive subspace *as the largest dimension that yields an ESS that does not drop below some threshold*, by using the prior as importance proposal (consequently the dimension of the Active Subspace is fixed as the complement to n, where n is the dimension of the full space). We compare in the sections

below the novel ESS method with the traditional eigenvalues method explained in Section 4.2.2 and we see that they bring the same results, in case of the Gaussian model of Section 5.3.2 and for the Banana model of Section 5.4.2.

Gaussian model

As a recap, we report below in Figure 5.24 a copy of the spectral gap in the Gaussian 10D model, identified using the traditional method which identifies the size of the Active Subspace as $n_a = 1$

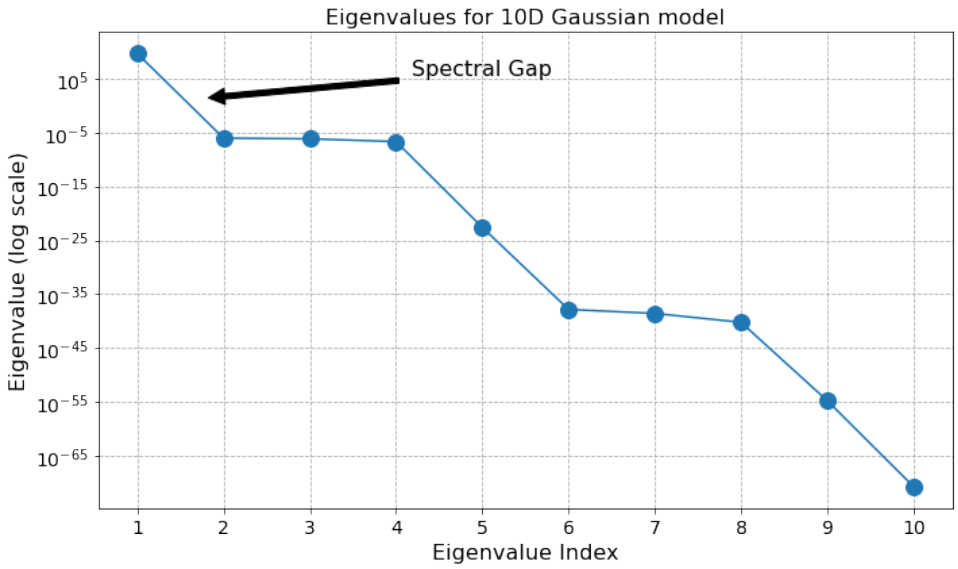

Figure 5.24: *Eigenvalues of 10D Gaussian model, we see that the estimate AS size is* 1*, considering the spectral gap between eigenvalues* 1 *and* 2*. The dimension of the Active Subspaces is* $n_a = 1$*.*

We report below in Figure 5.25 a visual representation of the novel ESS method: recollecting that the method consists in finding *the largest dimension that yields an ESS that does not drop below some threshold*, we see in the figure that the dimension of the inactive subspace in the Gaussian 10D model is $n_i = 9$, in fact ESS is close to 100% until $n_i = 9$ (the number of full bars), and therefore the Active Subspace dimension is $n_a = 10 - 9 = 1$, compare with Figure 5.3 where with the traditional eigenvalue method the active dimension result is $n_a = 1$ as well, so the two methods show identical results.

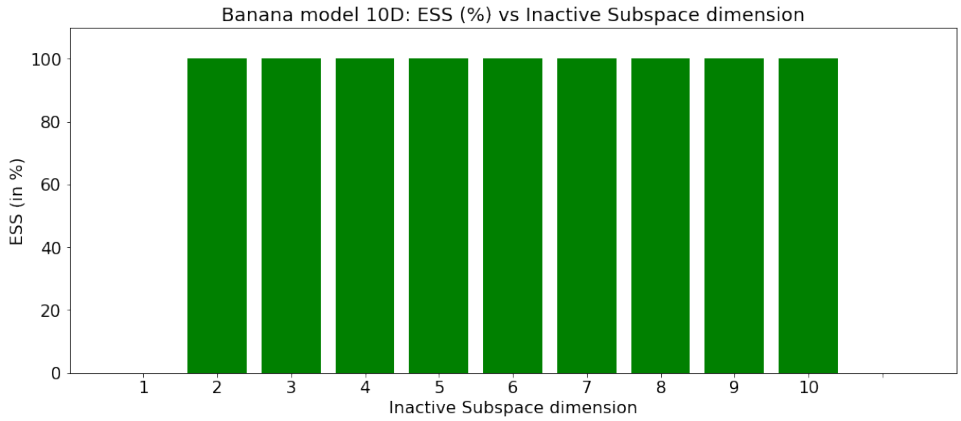

Figure 5.25: *ESS (%) using the prior as importance proposal for different dimensions of inactive subspace in the Gaussian* $10D$ *model. We see from the figure that the largest dimension* n_i *of the inactive subspace that brings a high ESS is* 9 *(the number of full bars) and we therefore set* $n_i = 9$ *and the dimension of Active Subspace is therefore* $n_a = 10 - n_i = 1$*. See comparison with Figure 5.24 where with the traditional eigenvalue method the active dimension result is* $n_a = 1$ *as well.*

With similar reasoning, we see that in the 25D Gaussian model both the traditional spectral gap in Figure 5.26 and the novel ESS method in Figure 5.27 bring identical results of dimension of the Active Subspace $n_a = 1$.

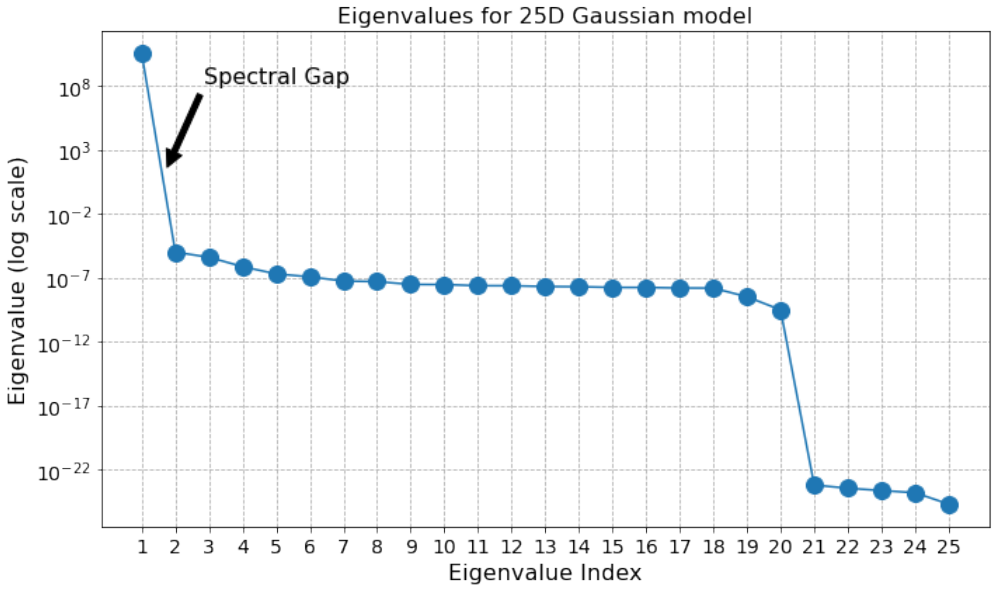

Figure 5.26: *Eigenvalues of 25D Gaussian model, we see that the estimate AS size is* 1*, considering the spectral gap between eigenvalues* 1 *and* 2*. The dimension of the Active Subspaces is* $n_a = 1$*.*

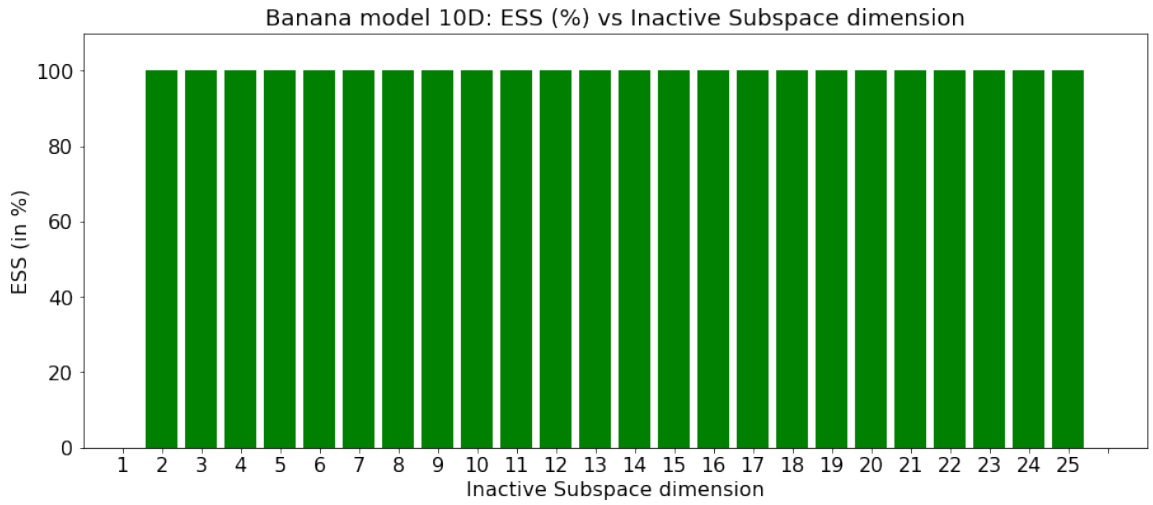

Figure 5.27: *ESS (%) using the prior as importance proposal for different dimensions of inactive subspace in the Gaussian* $25D$ *model. We see from figure that the largest dimension* n_i *of the inactive subspace that brings a high ESS is* 24 *(the number of full bars) and we therefore set* $n_i = 24$ *and the dimension of Active Subspace is therefore* $n_a = 25 - n_i = 1$*. See comparison with Figure 5.26 where with the traditional eigenvalue method the active dimension result is identically* $n_a = 1$*.*

Banana model

Repeating the same exercise of the previous section for the Banana model, we see that identical results are in the Banana 10D model with dimension of Active Subspace $n_a = 4$ identically in both the traditional spectral gap method of Figure 5.28 and in the novel ESS of Figure 5.29

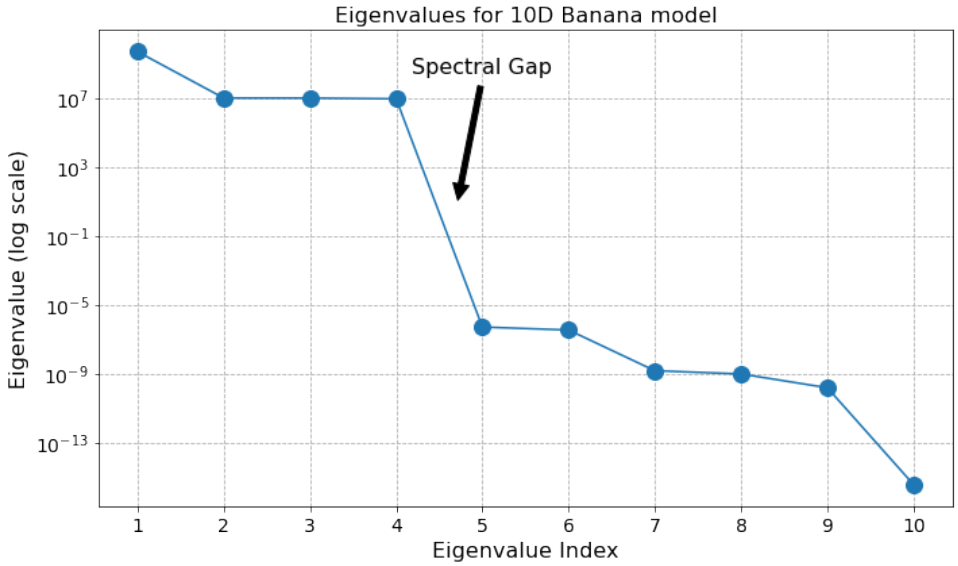

Figure 5.28: *Eigenvalues of* $10D$ *Banana model, we see that the estimate AS size is* 4*, considering the spectral gap between eigenvalues* 4 *and* 5*. The dimension of the Active Subspaces is* $n_a = 4$*.*

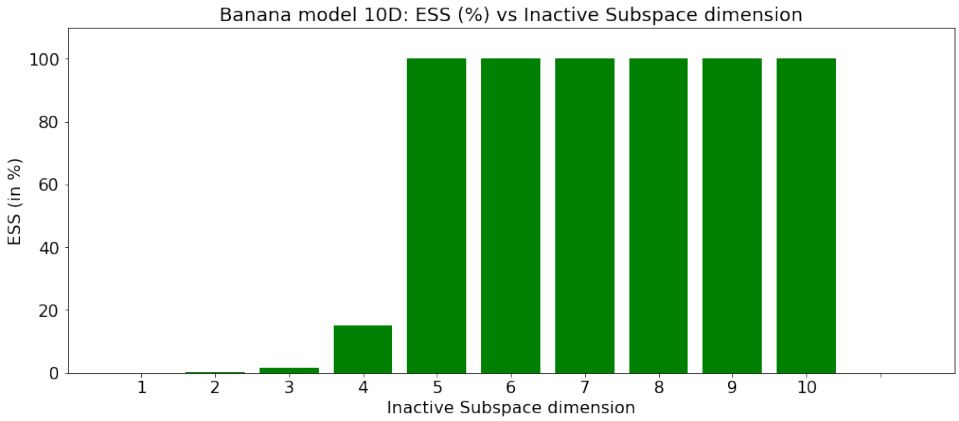

Figure 5.29: *ESS (%) using the prior as importance proposal for different dimensions of inactive subspace in the Banana* $10D$ *model. We see from figure that the largest dimension* n_i *of the inactive subspace that brings a high ESS is* 6 *(the number of full bars) and we therefore set* $n_i = 6$ *and the dimension of Active Subspace is therefore* $n_a = 10 - n_i = 4$*. See comparison with Figure 5.28 where with the traditional eigenvalue method the active dimension result is* $n_a = 4$ *as well.*

For a 25D system the charts are in Figure 5.30 and 5.31, both identically indicating a dimension of the Active Subspace of $n_a = 4$.

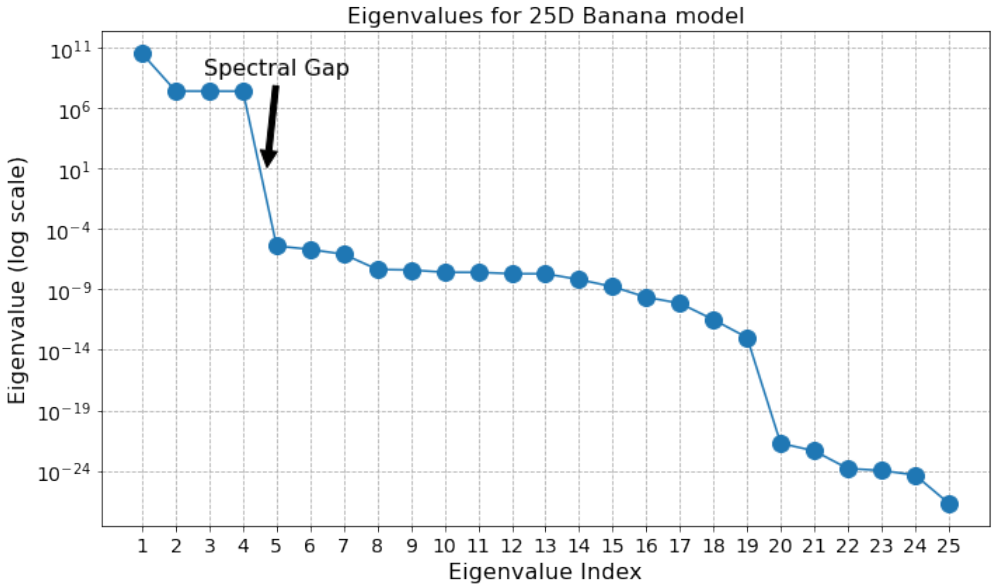

Figure 5.30: *Eigenvalues of 25D Banana model, we see that the estimate AS size is* 4*, considering the spectral gap between eigenvalues* 4 *and* 5*. The dimension of the Active Subspaces is* $n_a = 4$*.*

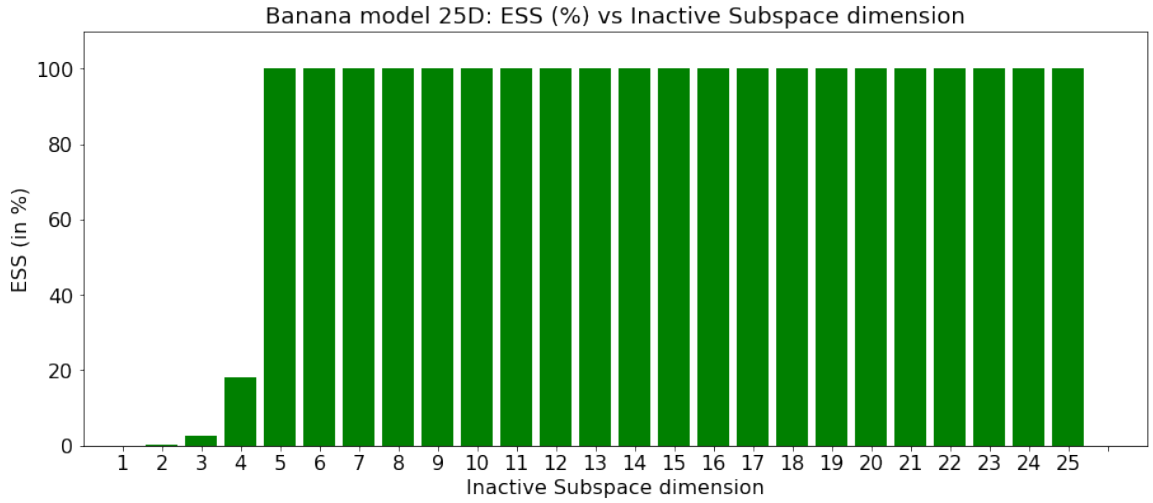

Figure 5.31: *ESS (%) using the prior as importance proposal for different dimensions of inactive subspace in the Banana* 25*D model. We see from figure that the largest dimension* n_i *of the inactive subspace that brings a high ESS is* 21 *(the number of full bars) and we therefore set* $n_i = 21$ *and the dimension of Active Subspace is therefore* $n_a = 25 - n_i = 4$*. See comparison with Figure 5.30 where with the traditional eigenvalue method the active dimension result is* $n_a = 4$ *as well.*

5.9.1 Review

We have introduced an alternative method to determine the dimension of the Active Subspace using the ESS (instead of the traditional spectral gap method explained in Section 4.2.2) by

finding the largest dimension of the inactive subspace that yields an ESS that does not drop below some threshold, and we have shown that it the Gaussian model of Section 5.3.2 and for the Banana model of Section 5.4.2 it brings identical results in determining the dimension of the Active Subspace. We consider the use of the ESS a relevant method, considering that, by definition, the inactive subspace should not affect the likelihood, and therefore should only influence the prior. There is indeed a relationship between the Effective Sample Size (ESS) method and the eigenvalues method. As Constantine notes in his work [Constantine et al., 2016], the eigenvalues represent 'the average squared directional derivative of the negative log-likelihood along the corresponding eigenvector.' Consequently, a small eigenvalue indicates minimal variation in that direction, suggesting that the likelihood provides little additional information along that dimension. Similarly, if the ESS method suggests that the prior is an effective proposal for the posterior, it implies that the likelihood is not particularly informative. Together, both a low ESS and small eigenvalues suggest that certain directions contribute minimally to the posterior, highlighting *inactive* areas.

5.10 Conclusion

We have explored a few methods in this chapter of Active Subspaces (AS) in MCMC. We started highlighting the main problem we are trying to solve, which is the *curse of dimensionality* affecting MCMC, and we have explored a few algorithms that have the aim to improve the performances. We started by describing a proposed novel Algorithm 10 named AS-PMMH, after the application of PMMH [Andrieu et al., 2010] to AS, which can be seen as a possible theoretical advancement compared to AS-MH Algorithm 8 as it uses a SMC sampler instead of Importance Sampling to obtain an estimate of the likelihood, we expect therefore the AS-PMMH to perform better in cases where the inactive subspace is complex to explore. We have seen that AS-PMMH brings an additional computational cost that can make it not always the best choice, and that it also shares with AS-MH the possibility that a noisy likelihood estimate may cause "sticky" behaviour (see Section 5.4.3).

We then introduced Algorithm 12, named AS-PMMH-i, where the additional i in the acronym means *inverted*, because, compared to AS-PMMH, we switch roles between active and inactive parts and we apply the outer MCMC to the inactive space and the inner SMC sampler to the active part, with the aim to spend the biggest computational effort of the SMC on the part we consider most valuable, the inactive, and we have concluded that, specifically on the models we used to test the algorithm, using estimate of marginals with SMC brings a *trade-off*: there is a bigger computational cost needed to have a more accurate estimate of the likelihood, which adds to the algorithmic complexity, so the potential advantages come at the cost of adding algorithmic complexity which can scale considerably depending on the setting (see for example the conclusions in Section 5.4).

We therefore then devised a proposed novel application of Gibbs sampler [Geman and

Geman, 1984] to AS, named AS-Gibbs, which has the advantage of not having marginals estimate and in cases of *near-perfect* AS we expect AS-Gibbs to outperform algorithms that use an estimate of marginal because in AS-Gibbs we would have the exact marginal [Andrieu and Vihola, 2015]. Also in general in cases of independence of active and inactive parts we expect AS-Gibbs to outperform standard MCMC as we have explained in Section 5.6.2.

We then proposed a natural extension of AS-Gibbs by introducing AS-MwPG Algorithm 15 and AS-MwPG-i Algorithm 17, from the application to AS of MwPG [Andrieu et al., 2010]. We consider the algorithms an extension of AS-Gibbs since the conditions where we expect to use them are similar to As-Gibbs, namely when active and inactive parts are independent. Both algorithms use an SMC sampler to draw particles either from the inactive (AS-MwPG) or the active (AS-MwPG-i) subspace and are expected to outperform AS-Gibbs in cases where the inactive or active subspaces respectively are complex, for example multi-modal. We have shown in Section 5.8 a toy example of a multi-modal distribution where the AS-MwPG-i was the only algorithm capable of correctly reconstructing both modes, whereas the other algorithms were stuck in one of the modes.

We also introduced in Section 5.9 a novel alternative method to determine the dimension of the Active Subspace that uses the ESS. The idea behind this, is measuring how good the prior will be as importance proposal for the inactive subspace, and in picking as inactive dimension the largest dimension of inactive subspace that yields an ESS that does not drop below some threshold (consequently the dimension of the Active Subspace is fixed as the complement to n, where n is the dimension of the full space). We have seen that the novel ESS method brings identical results to the traditional spectral gap of Section 4.2.2 in the Gaussian model and in the Banana model.

We proceed in the next Chapter 6 with the exploration of AS methods in SMC.

Chapter 6

Active Subspaces proposed novel SMC algorithms

6.1 Introduction

We have explored in the two Chapters 4 and 5 AS applications to MCMC in order, mainly, to decrease the problems caused by the *curse of dimensionality*. While MCMC remained the central part, we had additions from other MC methods like Importance Sampling, PMMH, MwPG and Gibbs sampling. We now shift the focus to algorithms where the core is SMC, with additional components from other algorithms.

We'll start the narrative of this chapter with the introduction of AS-SMC, an algorithm that was born with the intention to create the SMC-variant of the AS-MH Algorithm 8: while AS-MH uses Importance Sampling to marginalise out the inactive variables and an outer MCMC to sample from the active subspace, the AS-SMC still uses the IS on the inactive part, but will have a SMC sampler to generate samples from the active subspace. AS-SMC addresses the *Open Point* **OP3**, we expect AS-SMC to perform better than AS-MH in cases where the active subspace is complex to explore using MCMC, for example in case of multimodal distributions, where it is difficult to find a good proposal for the Active Subspace, and SMC overcomes this limitation by using tempering and intermediate distributions which ensure that at every step good proposals are available for the successive step.

6.2 AS-SMC

We introduce the AS-SMC Algorithm 18. The AS-SMC can be considered as the SMC counterpart of the AS-MH Algorithm 8, with the difference that the AS-SMC samples from the active marginal using SMC, whereas the AS-MH uses MCMC. And we see that actually an AS-MH move is embedded in the SMC sampler in the rejuvenation step lines 21-32, the pseudo-marginal in particular is calculated in lines 24-27.

We show the algorithm below, and give formal justification in Section 6.2.1.

Alg. 18 AS-SMC

Simulate N_a points, $\left\{\theta_0^m\right\}_{m=1}^{N_a} \sim p$ and set each weight $\omega_0^m = 1/N_a$; ▷ Draw from prior N_a particles

for $m = 1 : N_a$ **do** ▷ For each active particle, draw N_i inactive particles

  Set $a_0^m = B_a^T\left(\theta_0^m\right)$, $i_0^{1,m} = B_i^T\left(\theta_0^m\right)$ and $u_0^m = 1$;

  for $n = 2 : N_i$ **do** $i_0^{n,m} \sim q_{0,i}\left(\cdot \mid a_0^m\right) := p_i\left(\cdot \mid a_0^m\right)$;

  end for

  Set $w_0^{n,m} = 1/N_i$ for $n = 1 : N_i$;

end for

for $t = 1 : T$ **do**

  for $m = 1 : N_a$ **do** ▷ reweight

    for $n = 1 : N_i$ **do**

      $$\tilde{w}_t^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right) = \frac{p_i\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) l_{1:t}\left(B_a a_{t-1}^m + B_i i_{t-1}^{n,m}\right)}{q_{t,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right)};$$

    end for

$$\tilde{\omega}_t^m = \omega_{t-1}^m \frac{\prod_{j=1}^{N_i} q_{t,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right)}{\prod_{j=1}^{N_i} q_{t-1,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) \sum_{n=1}^{N_i} \tilde{w}_{t-1}^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right)};$$

  end for

  $\left\{\omega_t^m\right\}_{m=1}^{N_a} \leftarrow$ normalise $\left(\left\{\tilde{\omega}_t^m\right\}_{m=1}^{N_a}\right)$; ▷ Normalise outer weights

  for $m = 1 : N_a$ **do** $\left\{w_t^{n,m}\right\}_{n=1}^{N_i} \leftarrow$ normalise $\left(\left\{\tilde{w}_t^{n,m}\right\}_{n=1}^{N_i}\right)$;

  end for ▷ Normalise inner weights

  for $m = 1 : N_a$ **do**

    $u_t^m \sim \mathcal{M}\left(\left(w_t^{1,m}, ..., w_t^{N_i,m}\right)\right)$;

  end for

  if some degeneracy condition is met **then** resample and move

    for $m = 1 : N_a$ **do**

      Simulate $\left(a_t^m, i_t^{1:N_i,m}\right)$ from the mixture distribution

$$\sum_{j=1}^{N_a} \omega_t^j K_{t,a}\left\{\cdot \mid \left(a_{t-1}^j, i_{t-1}^{1:N_i,j}\right)\right\},$$

where $K_{t,a}$ is an AS-MH move, i.e.: $j^* \sim \mathcal{M}\left(\left\{\omega_t^j\right\}_{j=1}^{N_a}\right)$; $a_t^{*m} \sim q_a\left(\cdot \mid a_t^{j^*}\right)$;

      for $n = 1 : N_i$ **do** ▷ Calculate pseudo-marginal

        $i_t^{*n,m} \sim q_{t,i}\left(\cdot \mid a_t^{*m}\right)$;

$$\tilde{w}_t^{n,m}\left(a_t^{*m}, i_{t,i}^{*n,m}\right) = \frac{p_i\left(i_t^{*n,m} \mid a_t^{*m}\right) l_{1:t}\left(B_a a_t^{*m} + B_i i_t^{*n,m}\right)}{q_{t,i}\left(i_t^{*n,m} \mid a_t^{*m}\right)};$$

      end for

      $u_t^{*m} \sim \mathcal{M}\left(\left(w_t^{*1,m}, ..., w_t^{*N_i,m}\right)\right)$, where for $n = 1 : N_i$

$$w_t^{*n,m} = \frac{\tilde{w}_t^{n,m}\left(a_t^{*m}, i_t^{*n,m}\right)}{\sum_{p=1}^{N_i} \tilde{w}_t^{p,m}\left(a_t^{*m}, i_t^{*p,m}\right)};$$

      Set $\left(a_t^m, \left\{i_t^{n,m}, \tilde{w}_t^{n,m}\right\}_{n=1}^{N_i}, u_t^m\right) = \left(a_t^{*m}, \left\{i_t^{*n,m}, \tilde{w}_t^{*n,m}\right\}_{n=1}^{N_i}, u_t^{*m}\right)$ with probability

$$\alpha_{t,a}^m = 1 \wedge \frac{p_a\left(a_a^{*m}\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,m}\left(a_t^{*m}, i_t^{*n,m}\right)}{p_a\left(a_t^{j^*}\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,j^*}\left(a_{t-1}^{j^*}, i_{t-1}^{n,j^*}\right)} \frac{q_{t,a}\left(a_t^{j^*} \mid a_t^{*m}\right)}{q_{t,a}\left(a_t^{*m} \mid a_t^{j^*}\right)};$$

      Else let $\left(a_t^m, \left\{i_t^{n,m}, \tilde{w}_t^{n,m}\right\}_{n=1}^{N_i}, u_t^m\right) = \left(a_t^{j^*}, \left\{i_t^{n,j^*}, \tilde{w}_t^{n,j^*}\right\}_{n=1}^{N_i}, u_t^{j^*}\right)$;

    end for

    $\omega_t^m = 1/N_a$ for $m = 1 : N_a$;

  end if

end for

6.2.1 Formal justification

Our aim is to use an SMC sampler to simulate from the target distribution $\pi(a, i)$ of equation (4.17). We borrow the idea from [Chopin et al., 2012] and we follow the same derivation. The target distribution π_t used at the tth iteration is chosen to be

$$\pi_t\left(a, \{i^n\}_{n=1}^{N_i}\right) = \frac{1}{Z_t} p_a(a) \prod_{j=1}^{N_i} q_{t,i}\left(i^j \mid a\right) \left(\frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_i\left(i^n \mid a\right) l_{1:t}\left(B_a a + B_i i^n\right)}{q_{t,i}\left(i^n \mid a\right)}\right) \tag{6.1}$$

where Z_t is a normalising constant. Following the derivation in [Chopin et al., 2012], we may rearrange this target as follows:

$$\begin{aligned}
\pi_t\left(a, \{i^n\}_{n=1}^{N_i}\right) &= \frac{1}{Z_t} p_a(a) \prod_{j=1}^{N_i} q_{t,i}\left(i^j \mid a\right) \left(\frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_i\left(i^n \mid a\right) l_{1:t}\left(B_a a + B_i i^n\right)}{q_{t,i}\left(i^n \mid a\right)}\right) \\
&= \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_a(a)\, p_i\left(i^n \mid a\right) l_{1:t}\left(B_a a + B_i i_j^n\right)}{Z_t} \left(\prod_{\substack{j=1 \\ j \neq n}}^{N_i} q_{t,i}\left(i^j \mid a\right)\right) \\
&= \frac{\pi_{t,a}(a)}{N_i} \sum_{n=1}^{N_i} \pi_{t,i}\left(i^n \mid a\right) \left(\prod_{\substack{j=1 \\ j \neq n}}^{N_i} q_{t,i}\left(i^j \mid a\right)\right)
\end{aligned} \tag{6.2}$$

where we used the result

$$p_a(a)\, p_i\left(i \mid a\right) l_{1:t}\left(B_a a + B_i i^n\right) = Z_t \pi_{t,a}(a)\, \pi_{t,i}\left(i \mid a\right).$$

Equation (6.2) includes in its marginals the target at iteration t of $\pi_t(a, i) = \pi_{t,a}(a)\, \pi_{t,i}\left(i \mid a\right)$.

The weight update of the SMC sampler, on line 11 of the algorithm, is obtained from equation (6.1) and by considering that

$$\tilde{\omega}_t^m = \omega_{t-1}^m \frac{\pi_t\left(a_{t-1}^m, \left\{i_{t-1}^{n,m}\right\}_{n=1}^{N_i}\right)}{\pi_{t-1}\left(a_{t-1}^m, \left\{i_{t-1}^{n,m}\right\}_{n=1}^{N_i}\right)} \tag{6.3}$$

$$= \omega_{t-1}^m \frac{p_a\left(a_{t-1}^m\right) \prod_{j=1}^{N_i} q_{t,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) \frac{1}{N_i} \sum_{n=1}^{N_i} \tilde{w}_t^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right)}{p_a\left(a_{t-1}^m\right) \prod_{j=1}^{N_i} q_{t-1,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) \frac{1}{N_i} \sum_{n=1}^{N_i} \tilde{w}_{t-1}^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right)} \tag{6.4}$$

$$= \omega_{t-1}^m \frac{\prod_{j=1}^{N_i} q_{t,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right)}{\prod_{j=1}^{N_i} q_{t-1,i}\left(i_{t-1}^{n,m} \mid a_{t-1}^m\right) \sum_{n=1}^{N_i} \tilde{w}_{t-1}^{n,m}\left(a_{t-1}^m, i_{t-1}^{n,m}\right)}, \tag{6.5}$$

Where, in (6.3), ω^m represents the *outer* weight of particle m, whereas $w^{n,m}$ is the *inner* weight of inactive particle n belonging to particle m. The upper ˜is for un-normalised quantities.

Following also [Chopin et al., 2012], in similar fashion to what was done in equations (4.23) and (4.23) for AS-MH, for estimating the expectation of a function h with respect to π_t, we may use

1. **Using one i-point for each a-point:**

$$\sum_{m=1}^{N_a} \omega_t^m h\left(B_a a_t^m + B_i i_t^{u_t^m, m}\right). \tag{6.6}$$

2. **Using all of the accepted i-points for each a-point:**

$$\sum_{m=1}^{N_a} \omega_t^m \sum_{n=1}^{N_i} w_t^{n,m} h\left(B_a a_t^m + B_i i_t^{n,m}\right). \tag{6.7}$$

6.2.2 Comparison of performances with other algorithms

We see how the AS-SMC compares to standard SMC, both in the Gaussian model introduced in Section 5.3.2, and in the Banana model of Section 5.4.2. The two models will allow us to evaluate the algorithms in different conditions: we have seen in the previous chapter that the Gaussian model seems to be a posterior that is easier to explore for the As-related algorithms, whereas the exploration of the Banana model presents some challenges.

Comparison in the Gaussian model

We start with comparisons with the 25D Gaussian model introduced in Section 5.3.2, which, as a reminder, has dimension of the Active Subspace of 1. We have performed 50 runs each of the standard SMC algorithm and of the AS-SMC Algorithm 18, in order to have the same number of likelihood evaluations for a fair comparison, we have used 10000 particles for the standard SMC and $N_a = 1000$ particles in the AS-SMC with number of inactive variables $N_i = 10$, so that $N_a \times N_i = 10000$, and we have also used the same tempering path for both algorithms. Figure 6.1 shows the distribution of the RMSE of the difference between the true mean and the mean estimated by each of the algorithms across the runs:

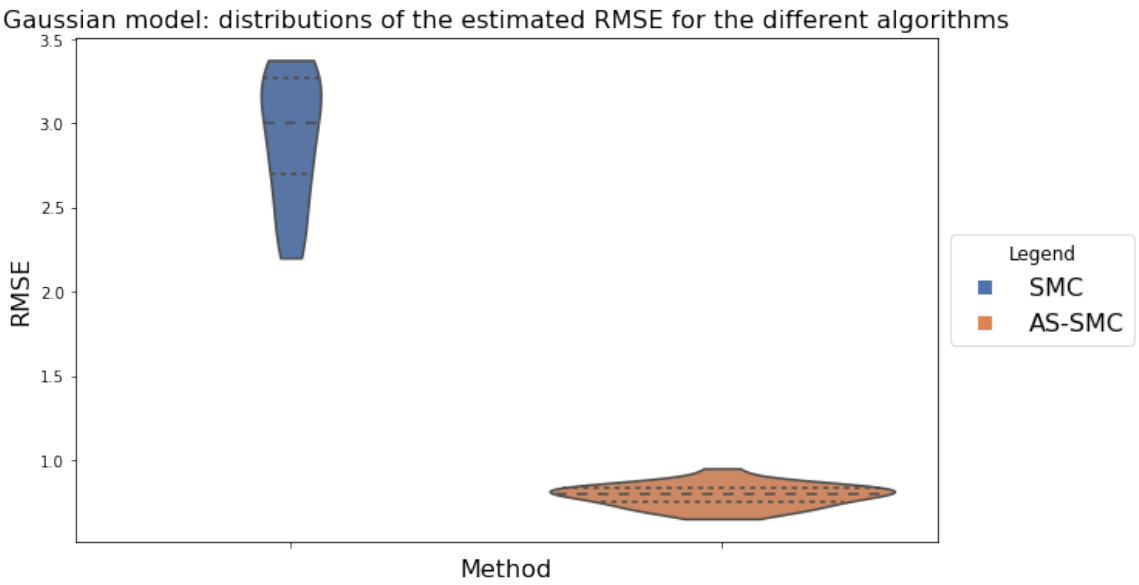

Figure 6.1: *Violin plots with the distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs in the* 25D *Gaussian model. We see that the performances of the standard SMC appears to be worse than the AS-SMC, this is probably due to the fact that we have a good estimate of the likelihood in the AS-SMC (in the Gaussian model the Importance Sampler seems to behave well on the inactive subspace even in high dimensions), coupled with the fact that the in the AS version the SMC operates on a* 1D *subspace instead of the full* 25D *space as the non-AS SMC.*

We see in Figure 6.1 that the performances of the standard SMC appear to be worse than the AS-SMC, which may be expected if we remember that in the Gaussian model the estimate of the likelihood via pseudo-marginal seems to behave well even in fairly high dimensions of the inactive subspace (we see small tails in the distribution of the AS-SMC in Figure 6.1), coupled with the fact that the SMC sampler in AS-SMC acts on a 1D space, whereas the SMC sampler in the standard SMC algorithm acts on the full 25D space.

Comparison in the Banana model

We proceed with comparisons using the 25D Banana model introduced in Section 5.4.2, which, as a reminder, has dimension of the Active Subspace of 4, and we have also seen in Chapter 5 that the Banana model poses some challenges to the algorithms (we had some cases of long tails in the distributions of the RMSE, for example Figure 5.13, indicating poor estimate of the likelihood). We have performed 50 runs each of the standard SMC algorithm and of the AS-SMC Algorithm 18, with the same conditions of the previous paragraph. Figure 6.2 shows the distribution of the RMSE of the difference between the true mean and the mean estimated by each of the algorithms across the runs:

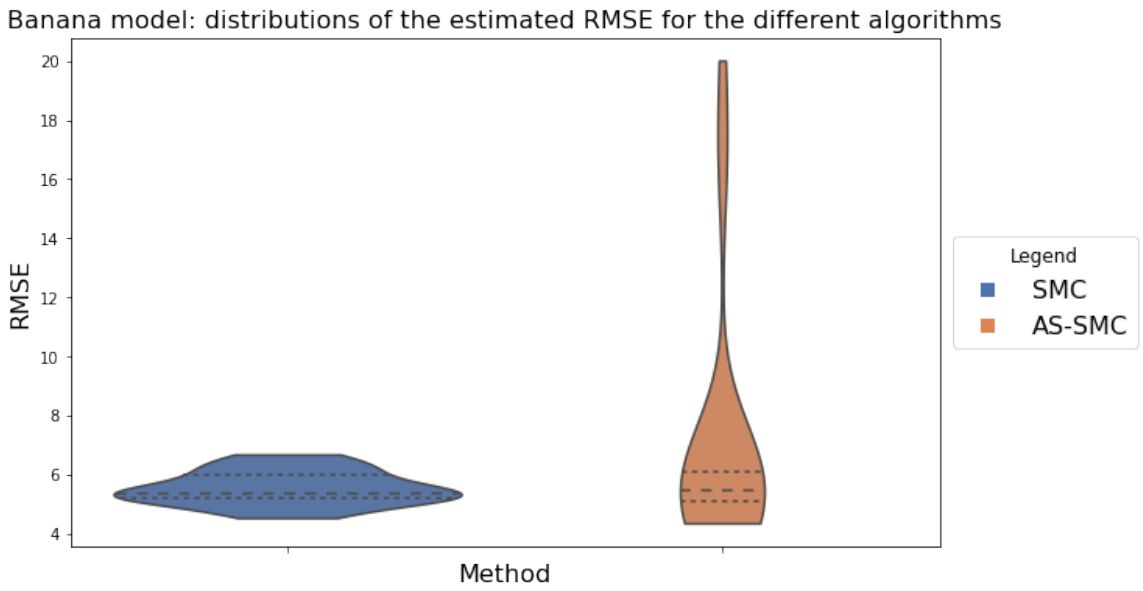

Figure 6.2: *Violin plots with the distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs in the* 25D *Banana model. We see that the performances of the standard SMC and AS-SMC appear to be approximately equal in terms of mean and upper and lower quartile, but the AS-SMC is showing some tails which suggest again that the estimate of the likelihood may be poor in some cases and the algorithm can get stuck in the tail of the distribution, this is probably due to the fact that using* 10 *inactive variables is too little for the exploration of the* 21D *inactive subspace of the Banana model, and this causes the noise in the importance sampler estimate of the likelihood.*

We see in Figure 6.2 that the performances of the standard SMC and AS-SMC appear to be approximately equal in terms of mean and upper and lower quartile, but the AS-SMC is showing some tails which suggest again that the estimate of the likelihood may be poor in some cases and the algorithm can get stuck in the tail of the distribution, this is probably due to the fact that using 10 inactive variables is too little for the exploration of the 21D inactive subspace of the Banana model, and this causes the noise in the importance sampler estimate of the likelihood.

6.2.3 Review

We have introduced the AS-SMC algorithm, which is the SMC equivalent of the AS-MH. We have seen that it outperforms the standard SMC in cases where the Importance Sampling is a good tool for the exploration of the inactive subspace, see for example Figure 6.1; but in cases of more complex inactive subspace the AS-SMC can give problems particularly if the estimate of the likelihood is noisy, see for example Figure 6.2 where the use of 10 particles to explore the inactive subspace is probably creating long tails in the RMSE distribution due to the noisy likelihood estimate. This potentially indicates the need for a tool that is more refined than Importance Sampling in cases where the inactive subspace is challenging

to explore, with this aim in mind, we have introduced the AS-SMC^2 algorithm in Section 6.3.

6.3 AS-SMC^2

We have introduced the SMC^2 [Chopin et al., 2012] in Section 2.9, as a recap of the algorithm: at each step of the tempering of an outer SMC, a particle MCMC is run to obtain an estimate of the likelihood (so the name SMC^2), in our case the outer SMC will be on the Active Subspace, and the inner on the inactive one: it can be though of as the SMC version of the AS-PMMH introduced in Section 5.2. The purpose of introducing the SMC^2 is that we aim to have an algorithm that can deal more effectively than the AS-SMC with a complex inactive subspace: we have seen in Section 6.2 that the performances of the internal Importance Sampler degrade as we go from a relatively simple inactive subspace like the Gaussian model, see Figure 6.1, to the more complex Banana model, see results in Figure 6.2. We present the Algorithm 19 below and give the formal justification in Section 6.3.1. We then show some results in Section 6.3.2.

Alg. 19 AS-SMC^2

Simulate N_a points, $\{\theta_0^m\}_{m=1}^{N_a} \sim p$ and set each weight $\omega_0^m = 1/N_a$;
for $m = 1 : N_a$ **do**
  Set $a_0^m = B_a^T(\theta_0^m)$, $i_0^{1,m} = B_i^T(\theta_0^m)$ and $u_0^m = 1$;
  for $n = 2 : N_i$ **do**
    $i_0^{n,m} \sim q_{0,i}(\cdot \mid a_0^m) := p_i(\cdot \mid a_0^m)$;
  end for
  Set $w_0^{n,m} = 1/N_i$ for $n = 1 : N_i$;
end for
for $t = 1 : T$ **do**
  for $m = 1 : N_a$ **do** ▷ reweight
    if $t = 1$ **then**
      Simulate $\left(i_t^{1:N_i,m}, v_{t-1}^{1:N_i,m}\right)$ using lines 3-30 (ignoring line 11) of Algorithm 9 taking s in these lines equal to t, then compute

$$\widehat{l_{t,a}\left(a_{t-1}^m\right)} = \sum_{n=1}^{N_i} \tilde{w}_t^{n,m};$$

$$\tilde{\omega}_t^m = \omega_{t-1}^m \widehat{l_{t,a}\left(a_{t-1}^m\right)};$$

    else
      Simulate $\left(i_t^{1:N_i,m}, v_{t-1}^{1:N_i,m}\right)$ using lines 3-30 (ignoring line 11) of Algorithm 9 taking s in these lines equal to t, then compute:

$$\frac{\widehat{l_{t,a}\left(a_{t-1}^m\right)}}{l_{t-1,a}\left(a_{t-1}^m\right)} = \sum_{n=1}^{N_i} \tilde{w}_t^{n,m};$$

$$\tilde{\omega}_t^m = \omega_{t-1}^m \frac{\widehat{l_{t,a}\left(a_{t-1}^m\right)}}{l_{t-1,a}\left(a_{t-1}^m\right)};$$

    end if
  end for
  $\{\omega_t^m\}_{m=1}^{N_a} \leftarrow$ normalise $\left(\{\tilde{\omega}_t^m\}_{m=1}^{N_a}\right)$;
  if some degeneracy condition is met **then**resample and move
    for $m = 1 : N_a$ **do**
      Simulate $\left(a_t^m, i_{1:t}^{1:N_i,m}, v_{1:t-1}^{1:N_i,m}\right)$ from the mixture distribution

$$\sum_{j=1}^{N_a} \omega_t^j K_{t,a}\left\{\cdot \mid \left(a_{t-1}^j, i_t^{1:N_u,j}, v_{t-1}^{1:N_i,j}\right)\right\},$$

      where $K_{t,a}$ is an AS-PMMH move, i.e.:
        $j^* \sim \mathcal{M}\left(\left\{\omega_t^j\right\}_{j=1}^{N_\theta}\right)$, then $a^* \sim q_{t,a}\left(\cdot \mid a_{t-1}^{j^*}\right)$, then run Algorithm 9 up to target t conditional on a^*.
      Set $a_t^m = a^*$ and $i_{1:t}^{n,m}, v_{1:t-1}^{n,m}$ and $\tilde{w}_{1:t}^{n,m}$ to be the variables and unnormalised weights generated when running Algorithm 9 with probability

$$1 \wedge \frac{p_a(a^*)}{p_a\left(a_{t-1}^{j^*}\right)} \frac{q_{t,a}\left(a_{t-1}^{j^*} \mid a^*\right)}{q_{t,a}\left(a^* \mid a_{t-1}^{j^*}\right)} \frac{\bar{l}_{t,a}(a^*)}{\prod_{t=1}^T \sum_{n=1}^{N_i} \tilde{w}_t^{n,j^*}},$$

      where $\bar{l}_{t,a}(a^*)$ given by Algorithm 9 run conditional on θ_a^*;
      Else set $a_t^m = a_{t-1}^{j^*}$, $\tilde{w}_{1:t}^{n,m} = \tilde{w}_{1:t}^{n,j^*}$, $i_{1:t}^{n,m} = i_{1:t}^{n,j^*}$ and $v_{1:t-1}^{n,m} = v_{1:t-1}^{n,j^*}$.
    end for
    $\omega_t^m = 1/N_a$ for $m = 1 : N_a$;
  end if
end for

6.3.1 Formal justification

We borrowed the idea from [Chopin et al., 2012], here we present the main ideas, with the full argument being found in the paper.

At iteration t we require our algorithm to have as one of its marginals the target distri-

bution

$$\pi_t(a,i) = \frac{p_a(a)\, p_i(i \mid a)\, l_{1:t}(B_a a + B_i i)}{Z_t},$$

with marginal distribution

$$\pi_{t,a}(a) = \frac{p(a)\, l_{t,a}(a)}{Z_t},$$

where

$$l_{t,a}(a) = \int_i p_i(i \mid a)\, l_{1:t}(B_a a + B_i i)\, di.$$

At $t = 0$, $\{a_0^m\}_{m=1}^{N_a} \sim p_a$ and $\{i_0^{n,m}\}_{n=1}^{N_i} \sim p_i(\cdot \mid a_0^m)$ for $m = 1 : N_u$. The target at iteration 0 is

$$p(a) \prod_{n=1}^{N_i} p_i(i_0^n \mid a),$$

from which we simulate N_a a-points and N_i i-points for every a. At $t = 1$ each particle is assigned the weight

$$\hat{l}_{1,a}(a) = \frac{1}{N_i} \sum_{n=1}^{N_i} l_1(B_a a + B_i i_0^n). \tag{6.8}$$

We introduce notation for the distribution of the i-variables generated at iteration 0 used to estimate $l_{1,a}$: $\psi_0\left(\{i_0^n\}_{n=1}^{N_i} \mid a\right) = \prod_{n=1}^{N_i} p_i(i_0^n \mid a)$. The target distribution at $t = 1$ being

$$\pi_1\left(a, \{i_0^n\}_{n=1}^{N_i}\right) = p_a(a)\, \psi_0\left(\{i_0^n\}_{n=1}^{N_i} \mid a\right) \frac{\hat{l}_{1,a}(a)}{Z_1}. \tag{6.9}$$

This results in the weight update in equation (6.8). We can rewrite the target as

$$\begin{aligned}
\pi_1\left(a, \{i_0^n\}_{n=1}^{N_i}\right) &= \frac{p_a(a)}{Z_1} \prod_{n=1}^{N_i} p_i(i_0^n \mid a) \left(\frac{1}{N_i} \sum_{n=1}^{N_i} l_1(B_a a + B_i i_0^n)\right) \\
&= \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_a(a)}{Z_1} p_i(i_0^n \mid a)\, l_1(B_a a + B_i i_0^n) \left(\prod_{\substack{j=1 \\ j \neq n}}^{N_i} p_i\left(i_0^j \mid a\right)\right) \\
&= \frac{\pi_{1,a}(a)}{N_i} \sum_{n=1}^{N_i} \pi_{t,i}(i_0^n \mid a) \left(\prod_{\substack{j=1 \\ j \neq n}}^{N_i} p_i\left(i_0^j \mid a\right)\right)
\end{aligned} \tag{6.10}$$

where we used the result

$$p_a(a)\, p_i(i_0^n \mid a)\, l_1(B_a a + B_i i_0^n) = Z_1 \pi_{1,a}(a)\, \pi_{t,i}(i_0^n \mid a).$$

In the marginals of equation (6.10) we have the target at iteration 1 of $\pi_1(a, i \mid y) =$

$\pi_{1,a}(a)\,\pi_{1,i}(i_0^n \mid a)$.

For $t \geq 2$, similarly to equation 6.9, we again have that our target distribution is defined to be the prior, multiplied by the likelihood estimate, multipled by the distribution of the variables used in the likelihood estimator, multiplied by the normalising constant (which again follows from the unbiasedness of the likelihood estimator).

Let ψ_{t-1} be the distribution of all of the random variables generated by the internal SMC up to time t.

$$\pi_t\left(a, \left\{i_{0:t-1}^n, v_{0:t-1}^n\right\}_{n=1}^{N_i}\right) = p_a(a)\,\psi_{t-1}\left(\left\{i_{0:t-1}^n, v_{0:t-1}^n\right\}_{n=1}^{N_i} \mid a\right)\frac{\hat{l}_{t,a}(a)}{Z_t}, \tag{6.11}$$

Similarly to the $t = 1$ case, we may rearrange equation (6.11) to see that $\pi_t(a, i)$ is included in its marginals:

$$\pi_t\left(a, \left\{i_{0:t-1}^n, v_{0:t-1}^n\right\}_{n=1}^{N_i}\right) = \frac{\pi_{t,a}(a)}{N_i} \times \sum_{n=1}^{N_i} \frac{\pi_{t,i}\left(\mathbf{i}_{1:t-1}^n \mid a\right)}{N_u^{t-1}} \left(\prod_{\substack{j=1 \\ j \neq \mathbf{h}_t^n(0)}}^{N_i} p_i\left(i_0^j \mid a\right)\right)\left(\prod_{s=2}^{t}\prod_{\substack{j=1 \\ j \neq \mathbf{h}_t^n(s-1)}}^{N_i} w_{s-1}^{v_{s-1}^j} K_{s-1,i}\left(i_{s-1}^j \mid i_{s-2}^{v_{s-2}^j}, a\right)\right)$$

where $\mathbf{i}_{1:t-1}^n$ and $\mathbf{h}_t^n$ are deterministic functions of $\left\{i_{0:t-1}^n\right\}_{n=1}^{N_i}$ and

$$\left\{v_{0:t-1}^n\right\}_{n=1}^{N_i} : \mathbf{h}_t^n = \left(\mathbf{h}_t^n(0), \ldots, \mathbf{h}_t^n(t-1)\right)$$

denotes the index history of v_{t-1}^n, i.e. $\mathbf{h}_t^n(t-1) = n$ and $\mathbf{h}_t^n(s) = v^{\mathbf{h}_t^n(s+1)}$, recursively for $s = t-2, ..., 0$, and $\mathbf{i}_{1:t-1}^n = \left(\mathbf{i}_{1:t-1}^n(0), ..., \mathbf{i}_{1:t-1}^n(t-1)\right)$ denotes the state trajectory of particle i_{t-1}^n, i.e. $\mathbf{i}_{1:t-1}^n(s) = i_s^{h_t^n(s)}$, for $s = 0, ..., t-1$.

For the remainder of the proof we follow [Chopin et al., 2012], with the one difference in the notation from that paper that here the index of the i variable is one fewer: i.e. equation (6.11) uses $\psi_{t-1}\left(\left\{i_{0:t-1}^n, v_{0:t-1}^n\right\}_{n=1}^{N_i} \mid a\right)$, whereas the equivalent in [Chopin et al., 2012] would be $\psi_t\left(\left\{i_{1:t}^n, v_{1:t-1}^n\right\}_{n=1}^{N_i} \mid a\right)$. The reason is that here the weight update in the internal SMC only involves the values of the particles from the previous iteration.

6.3.2 Comparison of performances with other algorithms

We compare the performances of the SMC^2 using the 25D Banana model introduced in Section 5.4.2, characterised by a dimension of the Active Subspace of 4. We have performed 50 runs each of the standard SMC algorithm, the AS-SMC Algorithm 18 and the SMC^2. For SMC and AS-SMC we have used the same run conditions of the previous paragraph: the standard SMC has been run with 10000 particles, the AS-SMC with 1000 particles and

10 inactive variables, so that it has the same number of likelihood evaluations (the same tempering sequence has been used). For the SMC^2, we have used a significantly higher number of likelihood evaluations of approximately 300000, the move has been necessary for the structure of the algorithm with the 2 SMC parts: the SMC^2 will require future tuning to reduce the significant additional operational complexity, without compromising the positive parts. Figure 6.3 shows the distribution of the RMSE of the difference between the true mean and the mean estimated by each of the algorithms across the runs:

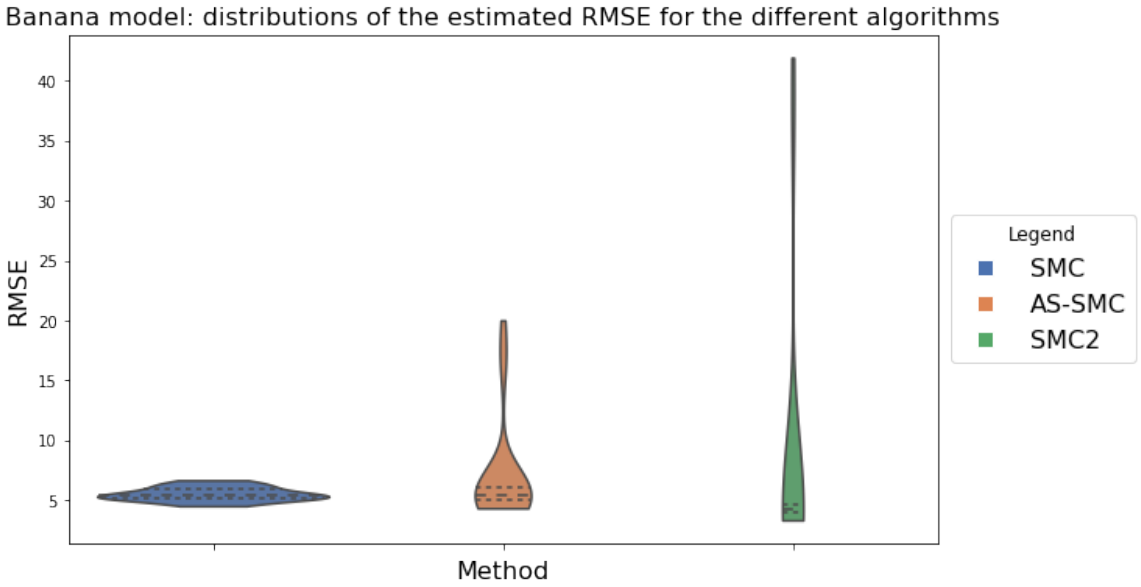

Figure 6.3: *Violin plots with the distribution of RMSE of the differences between the true posterior mean and the mean estimated by each of the algorithms over* 50 *runs in the* $25D$ *Banana model. The SMC^2 has lower mean and upper quartile than all the algorithms, but it shows longer tails, it is a sign that, although currently the algorithm is using significantly more likelihood evaluations than the others, still the algorithm may get stuck in one of tails, probably indicating that additional tuning of the algorithm is necessary.*

We see in Figure 6.3 the SMC^2 has lower mean and upper quartile than all the algorithms, but it shows longer tails, it is a sign that, although currently the algorithm is using significantly more likelihood evaluations than the others, and still suffers from the problem of the AS-PMMH, i.e. we are spending a considerable amount of computational effort on the inactive part, which is the least interesting to us. Still the algorithm may get stuck in one of tails, future tuning of the algorithm is necessary.

6.3.3 Review

We have compared the performances of the SMC^2 to standard SMC and AS-SMC algorithms. While for SMC and AS-SMC the same number of likelihood evaluation was used, each of the SMC^2 runs has a significant addition of approximately 300000 likelihood evaluations, future tuning of the algorithm is expected to bring the additional complexity down, while

still keeping the benefits. We have seen that there is a *trade-off* in the current setting, where the additional complexity allows for better mean and upper quartile in the estimate of the expectation (figure 6.3), but showing significantly longer tails of the RMSE distribution than the other two methods, probably indicating that the additional algorithmic complexity is still not enough and that future tuning is needed. Anyway, with the models explored so far, the use of SMC^2 would not seem justified as the additional overhead is significant and does not seem to bring comparable benefits, and we are still spending, like in AS-PMMH case, a considerable amount of computational effort on the inactive part, which is the least interesting to us.

6.4 Adaptive Active Subspaces SMC: AS-SMC-a

6.4.1 Introduction

We mentioned in Section 4.9 (and we reported in the list of *open points* as **OP4**) that one of the problems in current Active Subspace formulation is that, before running the MC algorithms we need to have *a priori* information on the structure of the Active Subspaces, through the matrices B_a and B_i which from equation (4.2) are generated using the posterior. So we have a potential deadlock because we would need posterior information beforehand but we can only have samples that approximate the posterior *after* having run the MC algorithms. The authors of [Constantine et al., 2016] came up with a solution that is now commonly used in AS: when it is not possible or convenient to draw from the posterior in the generating equation (4.2), approximations of matrices B_a and B_i can be built drawing samples from the prior instead of the full posterior. But there are cases where the prior AS can be very different from the posterior AS, and we have shown an example of this in Section 4.9 where the prior AS is orthogonal to the posterior AS, in such cases the prior AS approximation would be very poor (see also [Parente, 2020]). So we have devised a method that allows us to leverage the condition that, by using tempering and SMC in our algorithms, we *de facto* have progressive approximations of the posterior: the tempering usually starts from the prior and goes up to the posterior, and the SMC algorithm generates a smooth transition where at each step an intermediate distribution becomes the Importance proposal for the next step (we have explained the full process in Sections 2.5 and 2.6). The fundamental idea is to build a variant of the AS-SMC Algorithm 18 that at each step of the tempering will generate new matrices B_a and B_i which will therefore at each step be closer to the actual posterior directions. The advantages are clear: instead of approximating the directions using prior samples and keep them throughout the whole algorithm, with the novel method at each step we have an updated version of the matrices that are relevant for the current approximation of the posterior.

One main obstacle that we faced when thinking about such adaptive Active Subspaces

SMC algorithm, is that, in general, the dimension of the Active Subspace may change at every tempering step, and so we had to figure a way that could ensure theoretical consistency of the algorithm (mainly that the algorithm was still approximating the intended posterior). We found a solution in [Chopin et al., 2012], and we borrowed an idea that is used in the paper for automatic calibration of the number of particles in the particle filter, where they use a move of drawing from the current posterior approximation one particle and then using it to generate conditioned samples in the following approximation step. We will explain mathematically this move and how we have used it, in the next section.

6.4.2 Theoretical justification

We will be using a slightly different notation than in the previous chapters, as this will help us with indexes. We will be indicating with θ_a the active variable (previously a) and θ_i the inactive variable (previously i), with the following templates of equation (6.12) for the active variables

$$\theta^m_{t,a_{t-1}} \tag{6.12}$$

The meaning of symbols in (6.12) is for θ_a active particle m, at iteration t, for the active subspace determined by $B_{a_{t-1}}$. For the inactive variables we use the template (6.13)

$$\theta^{m,n}_{t,i_{t-1}} \tag{6.13}$$

The meaning of symbols in (6.13) is for θ_i, the inactive sub-particle number n for the active particle m, at iteration t, for the active subspace determined by $B_{i_{t-1}}$.

For ease of reference, we rewrite Algorithm 18 using the notation of 6.12 and 6.13, the new notation is below in Algorithm 20.

Alg. 20 AS-SMC (same Algorithm as 18, with different notation)

Simulate N_a points, $\{\theta_0^m\}_{m=1}^{N_a} \sim p$ and set each weight $\omega_0^m = 1/N_a$;
for $m = 1 : N_a$ **do**
  Set $\theta_{0,a}^m = B_a^T\left(\theta_0^m\right)$, $\theta_{0,i}^{1,m} = B_i^T\left(\theta_0^m\right)$ and $u_0^m = 1$;
  for $n = 2 : N_i$ **do**
    $\theta_{0,i}^{n,m} \sim q_{0,i}\left(\cdot \mid \theta_{0,a}^m\right) = p_i\left(\cdot \mid \theta_{0,a}^m\right)$;
  end for
  Set $w_0^{n,m} = 1/N_i$ for $n = 1 : N_i$;
end for
for $t = 1 : T$ **do**
  for $m = 1 : N_\theta$ **do** ▷ reweight
    for $n = 1 : N_i$ **do**
      $\tilde{w}_t^{n,m}\left(\theta_{t-1,a}^m, \theta_{t-1,i}^{n,m}\right) = \frac{p_i\left(\theta_{t-1,i}^{n,m} \mid \theta_{t-1,a}^m\right) l_{1:t}\left(B_a\theta_{t-1,a}^m + B_i\theta_{t-1.i}^n\right)}{q_{t,i}\left(\theta_{t-1,i}^{n,m} \mid \theta_{t-1,a}^m\right)}$;
    end for

$$\tilde{\omega}_t^m = \omega_{t-1}^m \frac{\prod_{j=1}^{N_i} q_{t,i}\left(\theta_{t-1,i}^{n,m} \mid \theta_{t-1,a}^m\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,m}\left(\theta_{t-1,a}^m, \theta_{t-1,i}^{n,m}\right)}{\prod_{j=1}^{N_i} q_{t-1,i}\left(\theta_{t-1,i}^{n,m} \mid \theta_{t-1,a}^m\right) \sum_{n=1}^{N_i} \tilde{w}_{t-1}^{n,m}\left(\theta_{t-1,a}^m, \theta_{t-1,i}^{n,m}\right)};$$

  end for
  $\{\omega_t^m\}_{m=1}^{N_a} \leftarrow$ normalise $\left(\{\tilde{\omega}_t^m\}_{m=1}^{N_a}\right)$;
  for $m = 1 : N_a$ **do**
    $\{w_t^{n,m}\}_{n=1}^{N_i} \leftarrow$ normalise $\left(\{\tilde{w}_t^{n,m}\}_{m=1}^{N_i}\right)$;
  end for
  if some degeneracy condition is met **then** resample and move
    for $m = 1 : N_\theta$ **do**
      Simulate $\left(\theta_{t,a}^m, \theta_{t,i}^{1:N_i,m}\right)$ from the mixture distribution

$$\sum_{j=1}^{N_a} \omega_t^j K_{t,\theta}\left\{\cdot \mid \left(\theta_{t-1,a}^j, \theta_{t-1,i}^{1:N_i,j}\right)\right\},$$

where $K_{t,\theta}$ is an ASMH move, i.e.:

$$j^* \sim \mathcal{M}\left(\left\{\omega_t^j\right\}_{j=1}^{N_a}\right);$$

$$\theta_{t,a}^{*m} \sim q_a\left(\cdot \mid \theta_{t,a}^{j^*}\right);$$

      for $n = 1 : N_i$ **do**

$$\theta_{t,i}^{*n,m} \sim q_{t,i}\left(\cdot \mid \theta_{t,a}^{*m}\right);$$

$$\tilde{w}_t^{*n,m} = \frac{p_i\left(\theta_{t,i}^{*n,m} \mid \theta_{t,a}^{*m}\right) l_{1:t}\left(B_a\theta_{t,a}^{*m} + B_i\theta_{t,i}^{*n,m}\right)}{q_{t,i}\left(\theta_{t,i}^{*n,m} \mid \theta_{t,a}^{*m}\right)};$$

      end for
      $u_t^{*m} \sim \mathcal{M}\left(\left(w_t^{*1,m}, ..., w_t^{*N_i,m}\right)\right)$, where for $n = 1 : N_i$

$$w_t^{*n,m} = \frac{\tilde{w}_t^{*n,m}}{\sum_{p=1}^{N_i} \tilde{w}_t^{*p,m}};$$

      Set $\left(\theta_{t,a}^m, \left\{\theta_{t,i}^{n,m}, \tilde{w}_t^{n,m}\right\}_{n=1}^{N_i}, u_t^m\right) = \left(\theta_{t,a}^{*m}, \left\{\theta_{t,i}^{*n,m}, \tilde{w}_t^{*n,m}\right\}_{n=1}^{N_i}, u_t^{*m}\right)$ with probability

$$\alpha_{t,a}^m = 1 \wedge \frac{p_a\left(\theta_{t,a}^{*m}\right) \sum_{n=1}^{N_i} \tilde{w}_t^{*n,m}\left(\theta_{t-1,a}^m, \theta_{t-1,i}^{n,m}\right)}{p_a\left(\theta_{t,a}^{j^*}\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,j^*}\left(\theta_{t-1,a}^{j^*}, \theta_{t-1,i}^{n,j^*}\right)} \frac{q_{t,a}\left(\theta_{t,a}^{j^*} \mid \theta_{t,a}^{*m}\right)}{q_{t,a}\left(\theta_{t,a}^{*m} \mid \theta_{t,a}^{j^*}\right)};$$

      Else let $\left(\theta_{t,a}^m, \left\{\theta_{t,i}^{n,m}, \tilde{w}_t^{n,m}\right\}_{n=1}^{N_i}, u_t^m\right) = \left(\theta_{t,a}^{j^*}, \left\{\theta_{t,i}^{n,j^*}, \tilde{w}_t^{n,j^*}\right\}_{n=1}^{N_i}, u_t^{j^*}\right)$;
    end for
    $\omega_t^m = 1/N_a$ for $m = 1 : N_a$;
  end if
end for

As said, we borrowed the idea from [Chopin et al., 2012]. At each iteration (we use t

as iteration variable) we start by having the adaptive version of equation (4.12), which is equation (6.14) where we can see the tempered likelihood $\log l_{1:t}$ featuring

$$\hat{C}_t = \sum_{m=1}^{N_a} \omega_{t-1}^m \sum_{n=1}^{N_i} w_{t-1}^{n,m} \nabla \log l_{1:t}\left(B_{a_{t-1}}\theta_{t-1,a_{t-1}}^m + B_i \theta_{t-1,i_{t-1}}^{n,m}\right) \nabla \log l_{1:t}\left(B_{a_{t-1}}\theta_{t-1,a_{t-1}}^m + B_{i_{t-1}}\theta_{t-1,i_{t-1}}^{n,m}\right)^T \tag{6.14}$$

Where we have used formula (6.7) to calculate the expectation. By using the approximation $\hat{C}_t$, instead of $\hat{C}$ of (4.12), following the procedure described in Sections 4.2 and 4.3, we find the matrices current to iteration t, B_{a_t} and B_{i_t}, which will also give a dimension of the current active and inactive subspace, $d_{t,a}$ and $d_{t,i}$, so that $d_{t,a} + d_{t,i} = d$ with d the dimension of the state space.

The target at time t is in equation (6.15) [Chopin et al., 2012], we can spot the terms of the pseudo-marginal part, and the proposal of the inactive variables

$$\begin{aligned}
\pi_{t,a_t,i_t}\left(\theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right) &= \frac{1}{Z_t} p_{a_t}\left(\theta_{a_t}\right) \prod_{j=1}^{N_i} q_{t,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right) \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_{i_t}\left(\theta_{i_t}^n \mid \theta_{a_t}\right) l_{1:t}\left(B_{a_t}\theta_{a_t} + B_{i_t}\theta_{i_t}^n\right)}{q_{t,i_t}\left(\theta_{i_t}^n \mid \theta_{a_t}\right)} \\
&= \frac{\pi_{t,a_t}\left(\theta_{a_t}\right)}{N_i} \sum_{n=1}^{N_i} \pi_{t,i_t}\left(\theta_{i_t}^n \mid \theta_{a_t}\right) \left(\prod_{\substack{j=1 \\ j\neq n}}^{N_i} q_{t,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right)\right)
\end{aligned} \tag{6.15}$$

at iteration t, where the first subscript in π_{t,a_t,i_t} denotes that this is the target for iteration t, and the second and third denote that the spaces of active and inactive variables are those determined by the adaptive procedure to be used at iteration t. Since the active subspace has changed between iterations, we additionally need to define the target from the previous iteration for the current active subspace. This is given by

$$\begin{aligned}
\pi_{t-1,a_t,i_t}\left(\theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right) &= \frac{1}{Z_t} p_{a_t}\left(\theta_{a_t}\right) \prod_{j=1}^{N_i} q_{t-1,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right) \frac{1}{N_i} \sum_{n=1}^{N_i} \frac{p_{i_t}\left(\theta_{i_t}^n \mid \theta_{a_t}\right) l_{1:t-1}\left(B_{a_t}\theta_{a_t} + B_{i_t}\theta_{i_t}^n\right)}{q_{t-1,i_t}\left(\theta_{i_t}^n \mid \theta_{a_t}\right)} \\
&= \frac{\pi_{t-1,a_t}\left(\theta_{a_t}\right)}{N_i} \sum_{n=1}^{N_i} \pi_{t-1,i_t}\left(\theta_{i_t}^n \mid \theta_{a_t}\right) \left(\prod_{\substack{j=1 \\ j\neq n}}^{N_i} q_{t-1,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right)\right).
\end{aligned} \tag{6.16}$$

This target is not the same as $\pi_{t-1,a_{t-1},i_{t-1}}$: in particular q_{t-1,i_t} and $q_{t-1,i_{t-1}}$ can be chosen independently of each other. This additional change in target between iterations $t-1$ and t requires an additional IS step, which is detailed at the end of this section.

The weight update at iteration t is then similar to that in section 6.2

$$\begin{aligned}\tilde{\omega}_t^m &= \frac{\pi_{t,a_t,i_t}\left(\theta_{t-1,a_t}^m, \left\{\theta_{t-1,i_t}^{n,m}\right\}_{n=1}^{N_i}\right)}{\pi_{t-1,a_t,i_t}\left(\theta_{t-1,a_t}^m, \left\{\theta_{t-1,i_t}^{n,m}\right\}_{n=1}^{N_i}\right)} \\ &= \omega_{t-1}^m \frac{\prod_{j=1}^{N_i} q_{t,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right) \sum_{n=1}^{N_i} \tilde{w}_t^{n,m}\left(\theta_{t-1,a_t}^m, \theta_{t-1,i_t}^{n,m}\right)}{\prod_{j=1}^{N_i} q_{t-1,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right) \sum_{n=1}^{N_i} \tilde{w}_{t-1}^{n,m}\left(\theta_{t-1,a_t}^m, \theta_{t-1,i_t}^{n,m}\right)},\end{aligned}$$

where

$$\tilde{w}_t^{n,m}\left(\theta_{t-1,a_t}^m, \theta_{t-1,i_t}^{n,m}\right) = \frac{p_{i_t}\left(\theta_{t-1,i_t}^{n,m} \mid \theta_{t-1,a_t}^m\right) l_{1:t}\left(B_{a_t}\theta_{t-1,a_t}^m + B_{i_t}\theta_{t-1.i_t}^n\right)}{q_{t,i_t}\left(\theta_{t-1,i_t}^{n,m} \mid \theta_{t-1,a_t}^m\right)},$$

and similarly to Section 6.2 and, for clarity,

$$\tilde{w}_{t-1}^{n,m}\left(\theta_{t-1,a_t}^m, \theta_{t-1,i_t}^{n,m}\right) = \frac{p_{i_t}\left(\theta_{t-1,i_t}^{n,m} \mid \theta_{t-1,a_t}^m\right) l_{1:t-1}\left(B_{a_t}\theta_{t-1,a_t}^m + B_{i_t}\theta_{t-1.i_t}^n\right)}{q_{t-1,i_t}\left(\theta_{t-1,i_t}^{n,m} \mid \theta_{t-1,a_t}^m\right)}.$$

At iteration t, for each particle we use an MCMC move with invariant distribution

$$\pi_{t,a_t,i_t}\left(\theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right) \tag{6.17}$$

running AS-MH Algorithm 8 with likelihood $l_{1:t}$ in place of l.

At the beginning of each iteration, an eigendecomposition of equation (6.14) is used to determine B_{a_t} and B_{i_t}. We then perform the following conditional SMC step on each particle to obtain a particle in the new active and inactive subspaces, moving particle $\left(\theta_{a_{t-1}}^m, \left\{\theta_{i_{t-1}}^{n,m}\right\}_{n=1}^{N_i}\right)$ to $\left(\theta_{a_t}^m, \left\{\theta_{i_t}^{n,m}\right\}_{n=1}^{N_i}\right)$. We use the observation [Chopin et al., 2012] that the target in equation (6.15) can be seen as a marginalisation of an extended distribution π_{t,a_t,i_t}^* over a uniformly distributed particle index variable u_t

$$\pi_{t,a_t,i_t}^*\left(u_t, \theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right) = \frac{\pi_{t,a_t}\left(\theta_{a_t}\right)}{N_i} \pi_{t,i_t}\left(\theta_{i_t}^{u_t} \mid \theta_{a_t}\right) \left(\prod_{\substack{j=1 \\ j \neq u_t}}^{N_i} q_{t,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right)\right). \tag{6.18}$$

[Chopin et al., 2012] notes that it is simple to extend a set of weighted particles from π_{t,a_t,i_t} so that they are from π_{t,a_t,i_t}^*: for each particle we simulate from the conditional distribution of u_t, which is given by $u_t \mid \theta_{t,a_t}, \left\{\theta_{t,i_t}^n\right\}_{n=1}^{N_i} \sim \mathcal{M}\left(\left(w_t^{1,m}, ..., w_t^{N_i,m}\right)\right)$, where $w_t^{n,m}$ is the normalised version of $\tilde{w}_t^{n,m}\left(\theta_{t,a_t}^m, \theta_{t,i_t}^{n,m}\right)$. At the beginning of iteration t, our method performs this simulation of u_{t-1} for each particle, then makes use of the following transformation of the extended state

$$\theta_{t-1,a_t} = G_{t-1\to t,a}\left(u_{t-1}, \theta_{t-1,a_{t-1}}, \left\{\theta_{t-1,i_{t-1}}^n\right\}_{n=1}^{N_i}\right) = B_{a_t}^T\left(B_{t-1,a_{t-1}}\theta_{t-1,a_{t-1}} + B_{t-1,i_{t-1}}\theta_{t-1,i_{t-1}}^{u_{t-1}}\right)$$

$$\theta_{t-1,i_t}^{u_{t-1}} = G_{t-1\to t,i}\left(u_{t-1}, \theta_{t-1,a_{t-1}}, \left\{\theta_{t-1,i_{t-1}}^n\right\}_{n=1}^{N_i}\right) = B_{i_t}^T\left(B_{t-1,a_{t-1}}\theta_{t-1,a_{t-1}} + B_{t-1,i_{t-1}}\theta_{t-1,i_{t-1}}^{u_{t-1}}\right)$$

$$u_{t-1} = u_{t-1}$$

The conditional IS move makes use of this transformation, along with a proposal from [Chopin et al., 2012]. Our desired target distribution for the new point is $\pi^*_{t-1,a_t,i_t}\left(u_{t-1}, \theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right)$, the extension of the target in equation (6.16), just as equation (6.18) is an extension of (6.15). Our proposal uses the current particle with values $\left(u_{t-1}, \theta_{t-1,a_t-1}, \left\{\theta_{t-1,i_{t-1}}^n\right\}_{n=1}^{N_i}\right)$ passed through the transformation above to give the point $\left(u_{t-1}, \theta_{t-1,a_t}, \theta_{t-1,i_t}^{u_{t-1}}\right)$. We additionally require the variables $\left\{\theta_{t-1,i_t}^n\right\}_{n=1,n\neq u_{t-1}}^{N_i}$ and will propose them from the conditional distribution $\left\{\theta_{t-1,i_t}^n\right\}_{n=1,n\neq u_{t-1}}^{N_i} \mid u_{t-1}, \theta_{t-1,a_t}, \theta_{t-1,i_t}^{u_{t-1}}$. Using equation (6.18), this conditional distribution is given by

$$\frac{\pi^*_{t-1,a_t,i_t}\left(u, \theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right)}{\frac{\pi_{t-1,a_t}(\theta_{a_t})}{N_i}\pi_{t-1,i_t}\left(\theta_{i_t}^u \mid \theta_{a_t}\right)} = \prod_{\substack{j=1\\ j\neq u}}^{N_i} q_{t-1,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right),$$

For the IS to be valid, we need to artificially extend the target distribution using a backwards kernel L (as in [Del Moral and Doucet, 2003]) to form a joint distribution over all of the variables involved in the proposal, such that the desired target π^*_{t,a_t,i_t} is a marginal distribution. The IS target is then

$$\pi^*_{t-1,a_t,i_t}\left(u, \theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right) L\left(\left\{\theta_{i_{t-1}}^n\right\}_{n=1,n\neq u}^{N_i} \mid u, \theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right)$$

and the proposal is

$$\pi^*_{t-1,a_{t-1},i_{t-1}}\left(u, \theta_{a_{t-1}}, \left\{\theta_{i_{t-1}}^n\right\}_{n=1}^{N_i}\right) \prod_{\substack{j=1\\ j\neq u}}^{N_i} q_{t-1,i_t}\left(\theta_{i_t}^j \mid \theta_{a_t}\right).$$

We choose the backwards kernel to be

$$L\left(\left\{\theta_{i_{t-1}}^n\right\}_{n=1,n\neq u}^{N_i} \mid u, \theta_{a_t}, \left\{\theta_{i_t}^n\right\}_{n=1}^{N_i}\right) = \frac{N_i \pi^*_{t-1,a_{t-1},i_{t-1}}\left(u, \theta_{a_{t-1}}, \left\{\theta_{i_{t-1}}^n\right\}_{n=1}^{N_i}\right)}{\pi_{t-1,a_t}\left(\theta_{a_t}\right)\pi_{t-1,i_t}\left(\theta_{i_t}^u \mid \theta_{a_t}\right)},$$

which gives an importance weight of 1 (as in [Chopin et al., 2012]). To proceed with the next iteration of the SMC, we then discard the value u_{t-1}, so that our particle is from the marginal distribution π_{t,a_t,i_t}, rather than the extended target π^*_{t,a_t,i_t}.

In summary:

- this additional step is run after determining the active subspace for the next iteration, for each of the N_a particles;
- we sample one of the inactive particles, using the weights of the particles in the inactive

space;

- we project the active variable and sampled inactive variable into the new active and inactive subspaces;
- we sample $N_i - 1$ additional inactive variables from the proposal q_{t-1,i_t}.

6.4.3 Early results for AS-SMC-a

We report below some results from early experiments that we have done by running the AS-SMC-a on the 2D model of Section 4.9.1. As a reminder, the model of Section 4.9.1 is quite interesting because the prior Active Subspace is different (orthogonal) from the posterior Active Subspace, see Figures 4.7 and 4.8.

Firstly we show the estimated mean across 10 runs of the model: both components have true mean of 0 and we see that the algorithm, at least across the few runs done, seems to correctly do the estimation of the two components in Figures 6.4 and 6.5 (the true mean of the model had been previously estimated with a system similar to what described for 5.9)

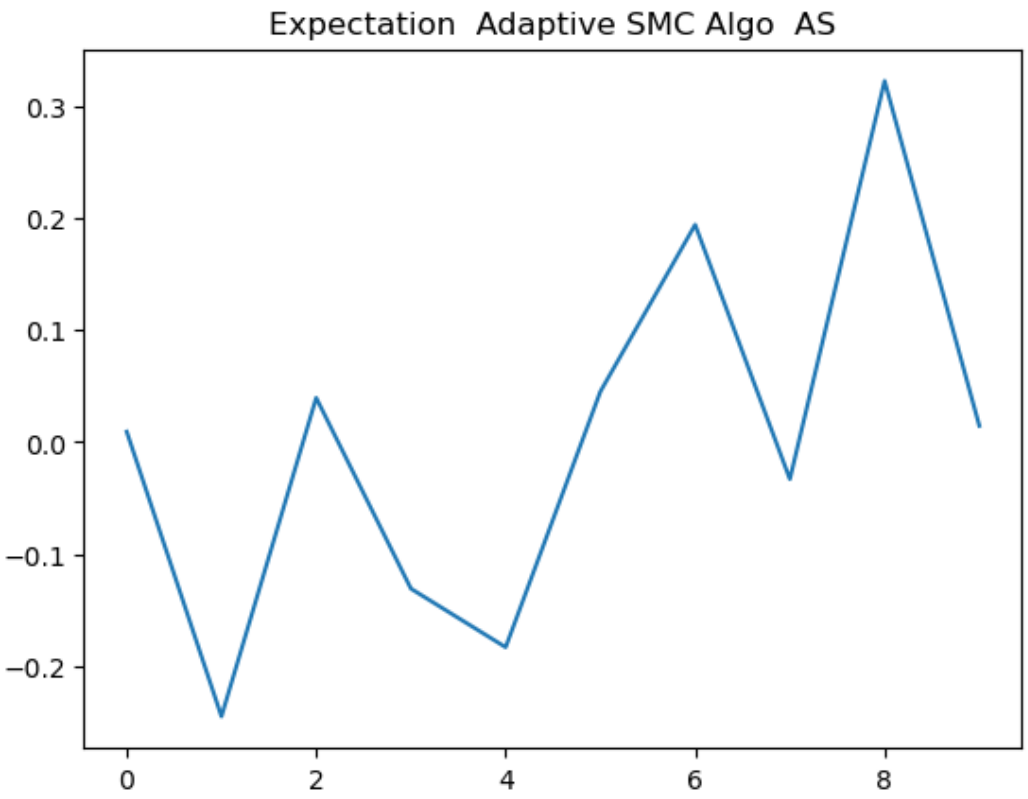

Figure 6.4: *Estimate of the mean of component* θ_1 *of the model of Section 4.9.1 across* 10 *runs of AS-SMC-a. The average across runs is* 0.0 *with a standard deviation of the measure of* 0.2.

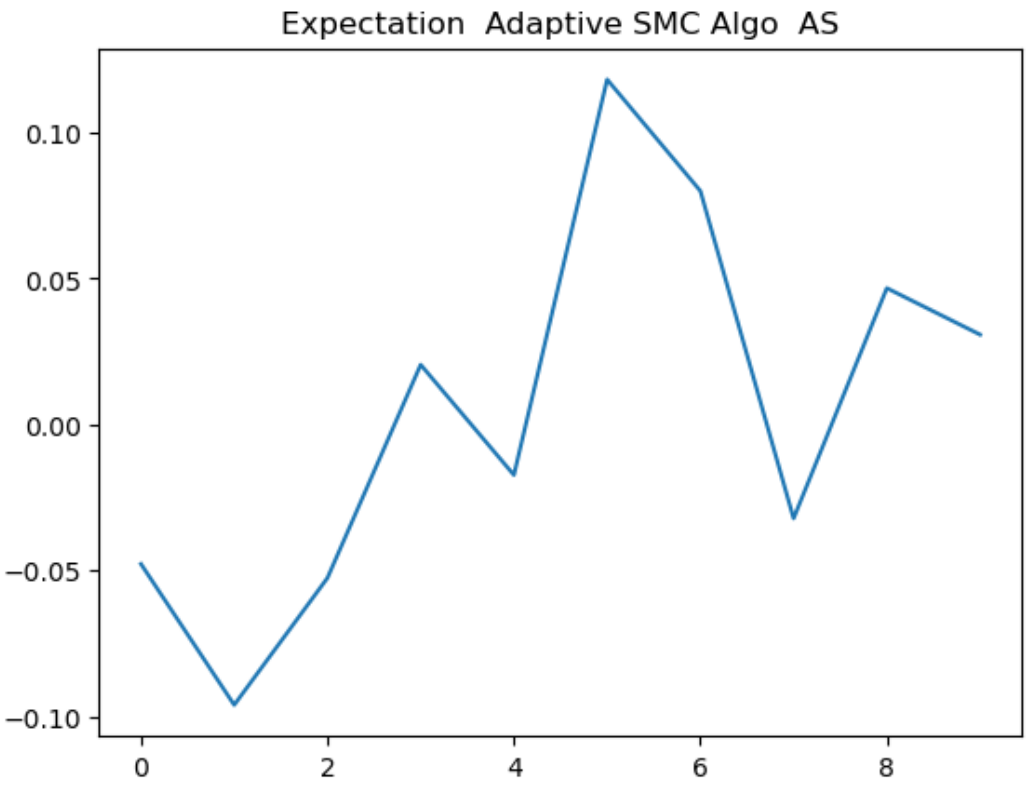

Figure 6.5: *Estimate of the mean of component θ_2 of the model of Section 4.9.1 across* 10 *runs of AS-SMC-a. The average across runs is* 0.0 *with a standard deviation of the measure of* 0.1.

We then visualize in Figure 6.6 below, how, during the adaptation in the tempering steps of the AS-SMC-a algorithm, the direction of the Active Subspace changes from the prior AS direction seen in Figure 4.7 to being 100% aligned along the posterior AS direction of Figure 4.8 in the final tempering steps. The same tempering steps have been used across the 10 runs.

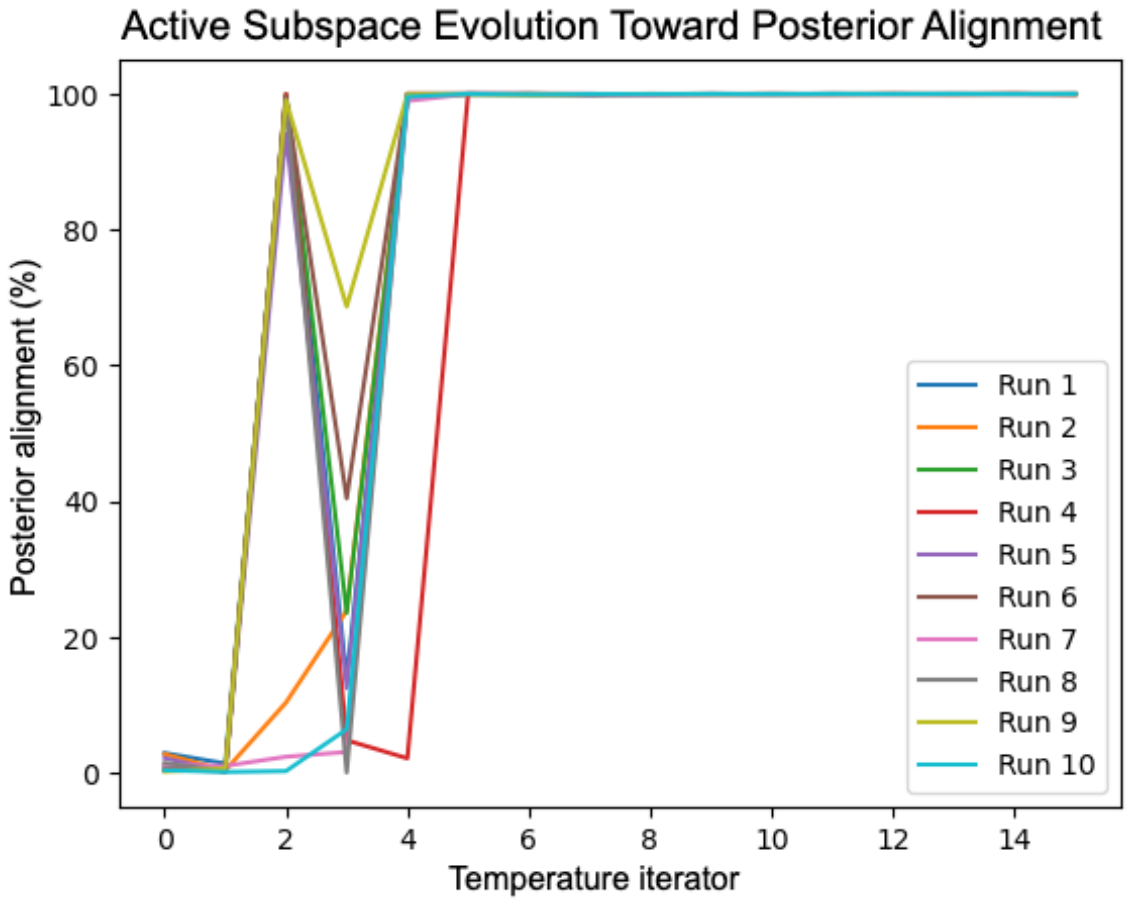

Figure 6.6: *Adaptation of the direction of the Active Subspaces in AS-SMC-a algorithm, measured across* 10 *different runs: the direction during the adaptation goes from prior AS of Figure 4.7 at tempering step* 0*, to posterior AS of Figure 4.8 in the final tempering steps. The same tempering steps have been used across the* 10 *runs.*

6.4.4 Review

We have introduced AS-SMC-a which brings adaptation of the Active Subspaces structure to the AS-SMC algorithm that we introduced in Section 6.2. We have seen that while in traditional AS methods the direction of the Active Subspace for the model of Section 4.9.1 would remain wrongly fixed along the direction of the prior AS of Figure 4.7 for the whole duration, the AS-SMC-a allows for the direction to adapt in the tempering steps, until it aligns correctly with the posterior AS of Figure 4.8. We expect the AS-SMC-a to possibly bring improvements into cases where the prior Active Subspaces directions are very different from the posterior Active Subspaces. It is to be noted that, as is the case for other algorithms, there is a *trade-off*, since the adaptation will bring additional computations, for example the calculation of the structure of the AS at each tempering step. Future experiments are needed to better understand the performances of the algorithm and how it stands compared to traditional methods and other AS-based algorithms that we have introduced throughout. One final note to underline the difference between our method and Spike-and-Slab [Mitchell

and Beauchamp, 1988, George and McCulloch, 1997, Ishwaran and Rao, 2005] and to explain why we did not use such a method in Active Subspaces. Spike-and-Slab is designed primarily for selecting individual variables by assigning them either a probability of exclusion (spike) or of inclusion (slab) [Mitchell and Beauchamp, 1988, George and McCulloch, 1997, Ishwaran and Rao, 2005] which could potentially be very useful in active-inactive setting, considering that we may want to eliminate inactive variables. However, applying spike-and-slab directly to the models we have used for our AS-SMC-a method may not always be feasible. For instance, both in the Gaussian model of section 5.3.2 and in the Banana model of section 5.4.2, AS-SMC-a will operate on linear combinations of variables rather than on each variable independently, see for example equations (5.1) and (5.15). Spike-and-slab, in contrast, operates separately for each variable from the other. This difference makes Spike-and-slab not suitable in general to be applied to the models we have considered.

6.5 Conclusion

We have started the chapter with the exploration of the AS-SMC algorithm in Section 6.2 which can be considered as the SMC counterpart of AS-MH of Section 4.8.2, we have seen that it performs better than standard SMC in cases where the Importance Sampler behaves well even in high dimension of the inactive space, for example the Gaussian model (see Figure 6.1), whereas when the inactive subspace becomes more challenging, for example in the Banana model, the advantage becomes less clear and long tails in the distribution of the RMSE appear (see Figure 6.2). We then introduced AS-SMC^2 in Section 6.3, which can be seen as the SMC counterpart of AS-PMMH of Section 5.2. We have seen that in the AS-SMC^2 as well, when applied to the Banana model, the extra computation cost does not seem to bring additional benefits and that more tuning of the algorithm may be needed to understand better the conditions of optimal performance of the algorithm. The AS-SMC^2 has a similar drawback to AS-PMMH, on spending a considerable amount of computational effort on the inactive part, which is the least interesting to us. Overall, while the performances on the Gaussian model are encouraging in saying that in case of perfect Active Subspaces the AS-SMC seems a clear winner, the results on the Banana model seem to show that in more complex scenarios when we diverge from cases of perfect Active Subspaces, the case for using either AS-SMC or AS-SMC^2 may be less clear, as the additional complexity brings some advantages in results but also disadvantages for examples longer tails in the distribution of the RMSE. We finally introduced in Section 6.4 a version of the AS-SMC that performs an adaptation of the structure of the AS at each tempering step, and we named the algorithm AS-SMC-a. We have shown how the adaptation has allowed the Active Subspace direction of the model of Section 4.9.1 to be correctly identified, while traditional AS methods would have kept the wrong AS direction throughout the algorithm. The adaptation may produce benefits in cases where prior and posterior AS are significantly different, however the extra

computations carry a *trade-off*, and the case of using the AS-SMC-a will have to be explored further.

Chapter 7

Conclusions and Future Work

This thesis contains two major threads: the integration of a Sequential Monte Carlo (SMC) algorithm into the BEAST2 platform for phylogenetic analysis and the development of novel Active Subspace (AS) methods for Monte Carlo methods. Both threads are linked in the aim to improve accuracy and efficiency of existing algorithms in the fields, and to reduce complexity.

7.1 Phylogenetics and SMC Integration

We have successfully integrated an Annealed Adaptive SMC algorithm into the BEAST2 platform, in this way we have complemented the native BEAST2 MCMC method. To our knowledge, this is the first integration of SMC within BEAST2. Our results show that, for the synthetic cases analyzed, the Annealed Adaptive SMC achieves performance comparable to BEAST2's native MCMC in terms of accuracy and efficiency. We have to note that the SMC method requires far fewer output samples to achieve the same aim of the MCMC: in a 10-taxa example shown, SMC required 1000 particles, compared to the $350,000$ iterations required by MCMC to achieve comparable likelihood evaluations. Future work is required to test the performance of SMC in more challenging scenarios, for example in multimodal distributions, where SMC is likely to outperform MCMC. Additionally, some fine-tuning of parameters within the BEAST2 platform could further optimize the SMC algorithm's performance. Our work represents a first step in making SMC methods more accessible for phylogenetic analysis, by the mean of the platform BEAST2, and future tuning is expected.

7.2 Active Subspaces for MCMC-based methods

We had the initial aim to address the *curse of dimensionality* in Monte Carlo methods, and so we explored Active Subspaces [Constantine, 2015, Constantine et al., 2016], a promising field in mathematics for the reduction of complexity in systems, by identifying an active and inactive subsystem. The first algorithm developed was AS-PMMH, which combines AS with

Particle Marginal Metropolis Hastings (PMMH) Andrieu et al. [2010]. Our AS-PMMH algorithm showed significant trade-offs between accuracy and computational costs, and sensitivity to noisy likelihood estimates. Also a second version of the algorithm, AS-PMMH-i, obtained by swapping the roles of active and inactive subspaces, still demonstrated trade-offs in computational complexity. We introduced AS-Gibbs, which embeds Gibbs sampling [Geman and Geman, 1984] into AS, avoiding marginal likelihood estimation. We have seen that AS-Gibbs proves particularly effective in cases of near-perfect Active Subspaces or independence between active and inactive parts. Extending this idea, we embedded Metropolis Within Particle Gibbs MwPG algorithms [Andrieu et al., 2010] in AS and AS-MwPG and AS-MwPG-i were proposed, using SMC samplers for either the inactive or active subspace. AS-MwPG-i in particular, showed superior performance in multimodal distributions, as demonstrated by a toy example where AS-MwPG-i was the only method capable of correctly reconstructing both modes. In determining the dimension of the Active Subspace, we proposed a novel ESS-based method. This method can be seen as an alternative to the spectral gap method, traditionally used in AS, and has shown to give identical results in the examples where we have tested it.

7.3 Active Subspaces for SMC-based methods

We have developed SMC based AS algorithms, named AS-SMC (AS-based SMC), AS-SMC^2, developed using the structure provided by AS within SMC^2 [Chopin et al., 2012], and an adaptive version of AS-SMC, named AS-SMC-a. AS-SMC, showed improved performance compared to traditional SMC in scenarios where Importance Sampling behaves well, even in high dimensions, such as in the Gaussian model of section 5.3.2. However, when applied to more complex cases, such as the Banana model of 5.4.2, performances declined. The metric used, RMSE of estimate of the mean, showed longer tails, and question the applicability of the method in practical application. We then introduced AS-SMC^2 which has two nested SMC within the AS framework, borrowing from the SMC^2 algorithm of [Chopin et al., 2012]. Experiments have shown that the additional computational costs often outweighed the benefits in practical scenarios. To address some of the limitations of traditional, fixed Active Subspace directions, namely that some pre-analysis needs to be done using prior samples and Active Subspace directions may be set wrongly, we proposed AS-SMC-a, an adaptive version of AS-SMC that updates the AS structure at each tempering step. In our tests, AS-SMC-a successfully adapted the direction of the Active Space in a case where the prior and posterior Active Subspace directions differed significantly. In a case where traditional AS algorithms would fail, AS-SMC-a successfully identified the correct AS direction dynamically. However, the adaptation brings additional trade-offs, since Active Subspace calculations need to be done at each tempering step for their practical implementation.

7.4 Potential future research directions

While the two research threads of phylogenetics and Active Subspaces address different challenges, they share a unique underlying theme: enhancing the computational efficiency and reducing complexity of Bayesian methods. Both approaches take on high-dimensional state spaces and aim to address the *curse of dimensionality*. A potential thread of future research lies in combining these threads. Incorporating Active Subspaces into phylogenetic analysis could bring more improvements in the efficiency of SMC methods in this field. By identifying Active Subspaces in high-dimensional phylogenetics models, AS could focus computational efforts on the most informative parts, potentially enabling SMC to handle even larger and more complex datasets.

Appendix A

SMC Annealed Adaptive phylogenetic algorithm: additional results with 5 and 20 taxa

Following the procedure and the structure of Section 3.14 where we presented the results for 10 taxa, we report in this section the results of comparison of the Annealed Adaptive SMC vs BEAST2 MCMC for 5 and 20 taxa. Please refer to Section 3.12 for full details of implementation.

A.1 Data with 5 Taxa

Following the procedure outlined in Section 3.13, we have generated a synthetic model with 5 taxa.

Generator tree

The first step has been as described in 3.13.1 to generate a coalescent tree with 5 leaves, and the tree has been randomly generated as below in Figure A.1

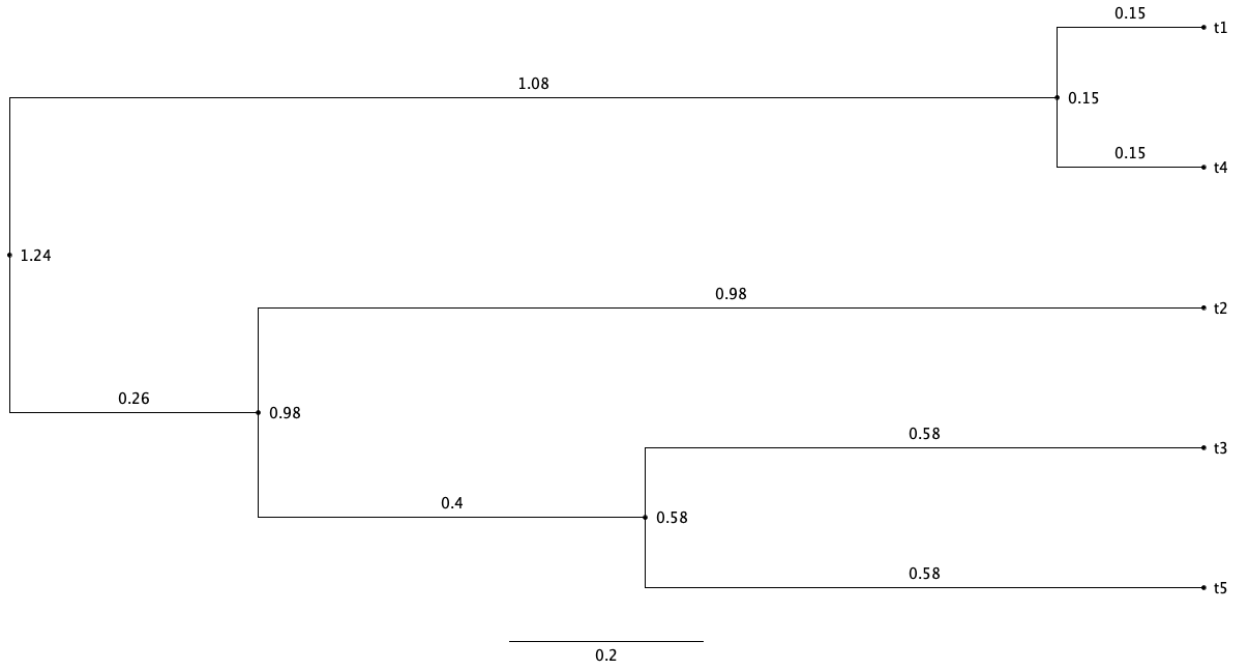

Figure A.1: *Random coalescent tree with 5 leaves generated using the procedure outlined in the first part of Section 3.13.1. This has been the generating tree for the synthetic data of the test described in this section. Visualization via FigTree [Rambaut, 2023]*

Generation of synthetic sequences

Using the tree generated in the previous step, synthetic sequences have been generated using 'seq-gen' program, as explained in Section 3.13.1

A.2 Annealed Adaptive SMC vs MCMC in BEAST2, problem set up with 5 Taxa

We have run BEAST2 both with the traditional MCMC algorithm and with our Annealed Adaptive SMC embedded in BEAST2, and we report here the comparison. A fair comparison in terms of likelihood evaluations has been kept between the two methods. For the comparison of results we have used a similar set-up and metrics of [Wang et al., 2019], in fact we have a number of iterations of MCMC which is comparable with the likelihood evaluations of the SMC algorithm, given by the number of particle times number of intermediate tempering steps of the annealing procedure, times the number of MCMC moves per annealing step. So, considering the comparison fair we report below the results for the various parameters of the state space.

A.2.1 SMC set up

The SMC has been set up with 1000 particles, and 5 MCMC moves per each annealing step. The number of annealing steps adaptively determined by the CESS (see Section 2.6.3 for details on CESS) has been 33, as can be seen in Figure A.2

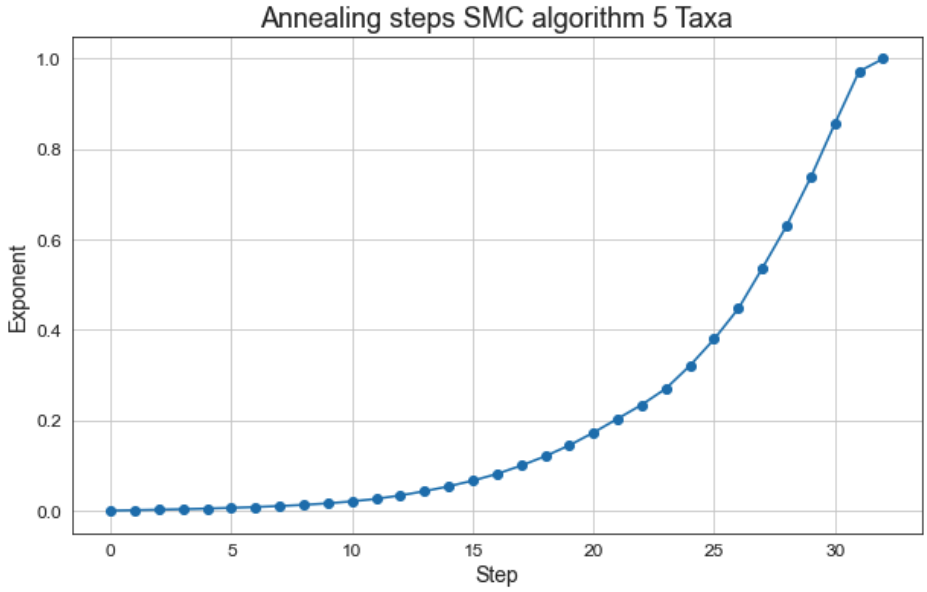

Figure A.2: *Annealing steps in the SMC run for the 5-taxa example studied in this section.*

Therefore the total number of likelihood evaluation for the algorithm has been $1000 \times 33 \times 5 = 165000$. The adaptive annealing steps have been determined using CESS with a threshold of 90%, and resampling of particles is done when ESS falls below 50% of particles, we can see below in Figure A.3 the ESS chart related to the SMC run

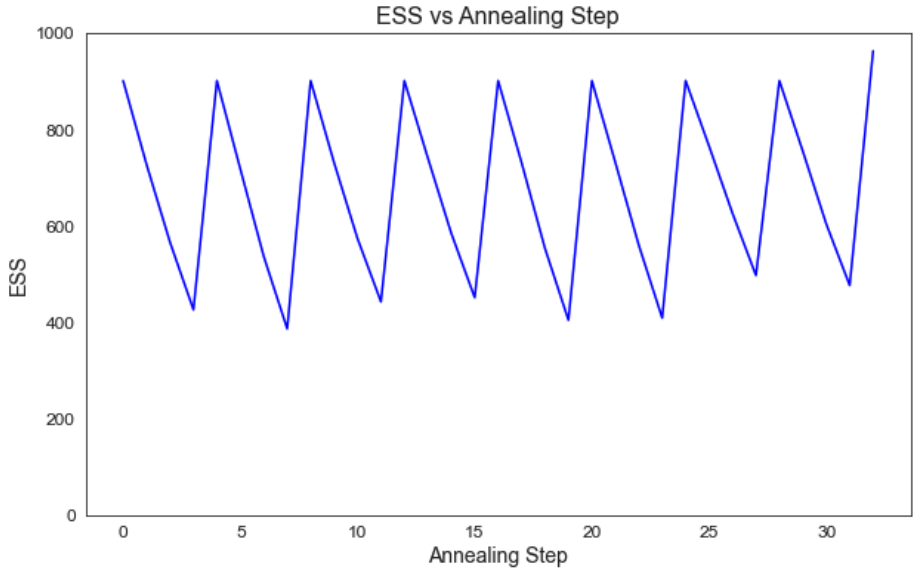

Figure A.3: *ESS versus annealing SMC step for the 5-taxa example studied in this section. Resampling is performed whenever the ESS falls below* 50% *of particles (1000 particles are used for the simulation).*

A.2.2 MCMC set up

Considering that the total number of likelihood evaluations from the Annealed Adaptive SMC was 165000, we have used a comparison similar to [Wang et al., 2019] and therefore we have used a number of MCMC iterations greater than +20% compared to the SMC run, in our MCMC simulation we have used 210000 iterations.

A.3 Annealed Adaptive SMC vs MCMC in BEAST2: Results with 5 Taxa

We will report in this section the results for the runs of traditional MCMC and Annealed Adaptive SMC using BEAST2, the parameters analysed are those composing the state space of our problem (see Sections 3.7 and 3.13.1):

- Gamma shape
- Effective population size
- Coalescent Tree

A.3.1 Gamma shape

The true value of the gamma shape parameter (i.e. the value with which the data has been generated) is 1. In both the MCMC and the SMC runs the empirical distributions are rather

scattered as we can appreciate from Figures A.4 for MCMC and A.5 for SMC, and this is due to the fact that the probability of MCMC moves on the gamma shape parameter has been kept to the default value that the configuration software BEAUTi (see Section 3.9.1) gives, and MCMC moves are less likely to happen than moves on effective population size and trees (as an example, an MCMC move on the gamma shape is 30 times less likely than a move on the Effective Population Size parameter), therefore the low ESS and the scattered distributions are a result of this.

MCMC results for Gamma shape

The mean of the MCMC run is close to the true value of 1, we can see the full statistics in the following table. Like in the 10 taxa case for gamma shape (see the MCMC part of Section 3.16.1), the ESS is very low and the distribution, as we can appreciate in the following Figure A.4, is rather scattered:

Statistic	Value
Mean	0.9
Standard Deviation	0.0656
Value Range	[0.7356, 1.0667]
95% HPD Interval	[0.7729, 1.0024]
Effective Sample Size (ESS)	99

And the distribution of the Gamma shape values is in Figure A.4

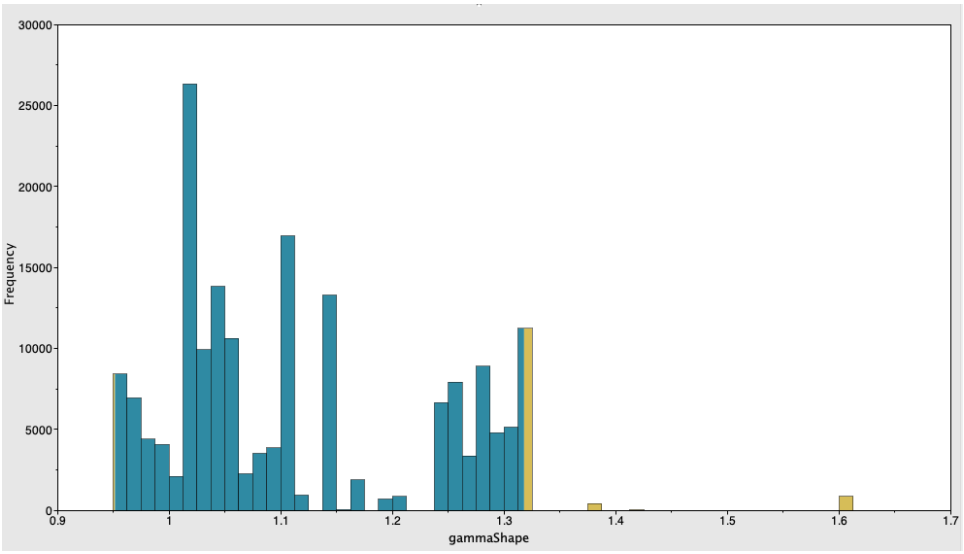

Figure A.4: *Frequency distribution using the native MCMC run with BEAST2 for the parameter Gamma shape with* 5 *taxa. Visualization with the software Tracer.*

Ammealed Adaptive SMC results for Gamma shape

The Statistics for the SMC run are similar to the MCMC run:

Statistic	Value
Mean	1.17
Standard Deviation	0.132
Value Range	[0.697, 1.256]
95% HPD Interval	[0.798, 1.25]

And the distribution of the Gamma shape values is in Figure A.5

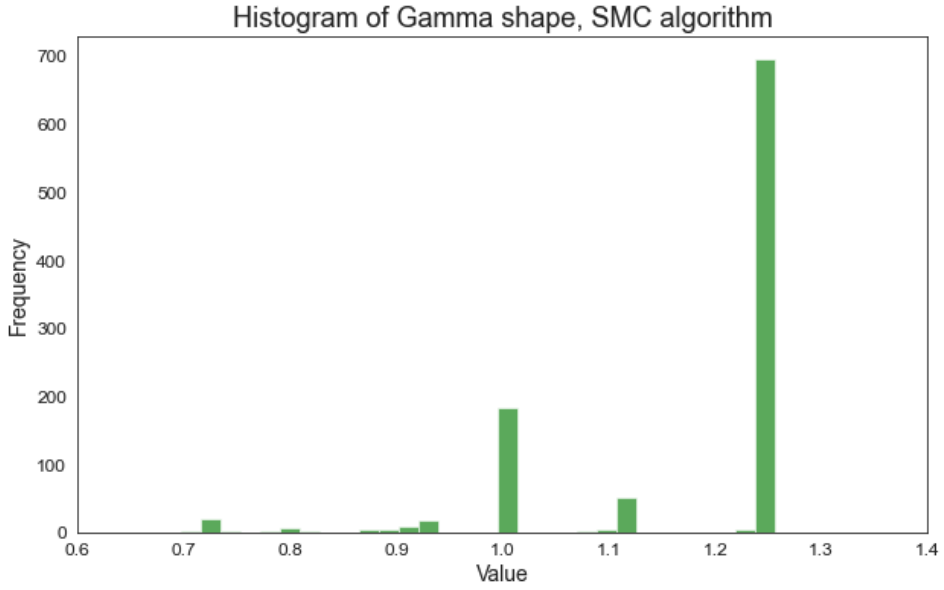

Figure A.5: *Frequency distribution for the parameter Gamma shape with 10 taxa, using Annealed Adaptive SMC algorithm that we have embedded into BEAST2. Visualization with python matplotlib.*

A.3.2 Effective Population Size

Unlike the Gamma shape parameter, seen in Section A.3.1, where we had at hand the true value of the parameter, and that was useful in checking how well MCMC and SMC could retrieve it, for the Effective Population Size, even if we generated synthetic data, it is not straightforward to have the true value. In fact, as explained in Section 3.5, we are usually only able to estimate the product of two terms, the coalescent constant and the Effective Population Size. Anyway, keeping this constraint in mind, we report the results of the analysis in this section.

MCMC results for Effective Population Size

The MCMC run has given a mean value of 2.7633

Statistic	Value
Mean	2.7633
Standard Deviation	1.4387
Value Range	[0.4307, 13.3307]
95% HPD Interval	[0.6802, 5.4841]
Effective Sample Size (ESS)	519

And the distribution of the Effective Population Size is in Figure A.6

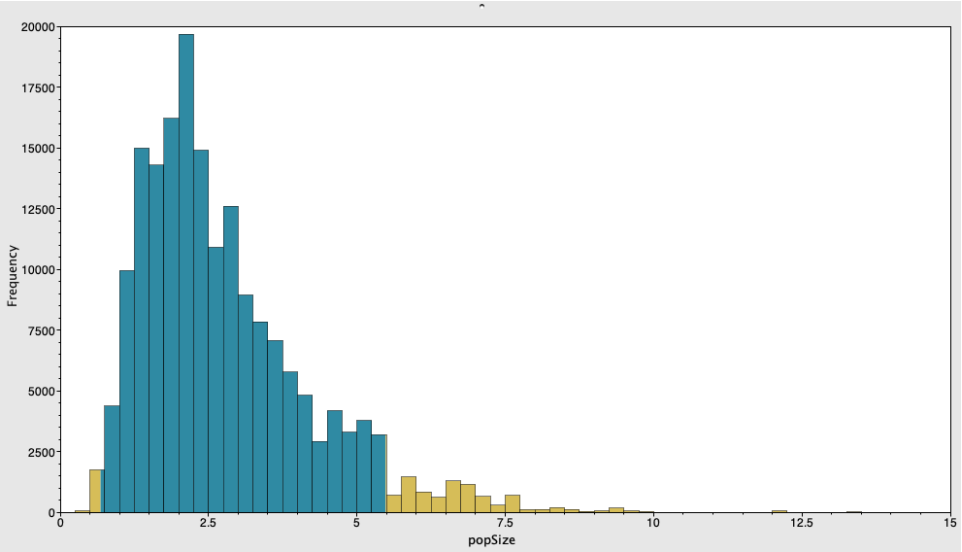

Figure A.6: *Frequency distribution using the native MCMC run with BEAST2 for the parameter Effective Population Size with 5 taxa. Visualization with the software Tracer.*

Ammealed Adaptive SMC results for Effective Population Size

The Statistics for the SMC run are in general better than the MCMC run, we can see a lower variance for example. And we can see from Figure A.7, that it has the same peak of the correspective MCMC Figure A.6, but in the MCMC case the bigger variance and right-skewness is causing a slightly higher value of the mean:

Statistic	Value
Mean	2.223
Standard Deviation	0.759
Value Range	[0.602, 6.272]
95% HPD Interval	[1.063, 3.903]

And the distribution of the Effective Population Size is in Figure A.7

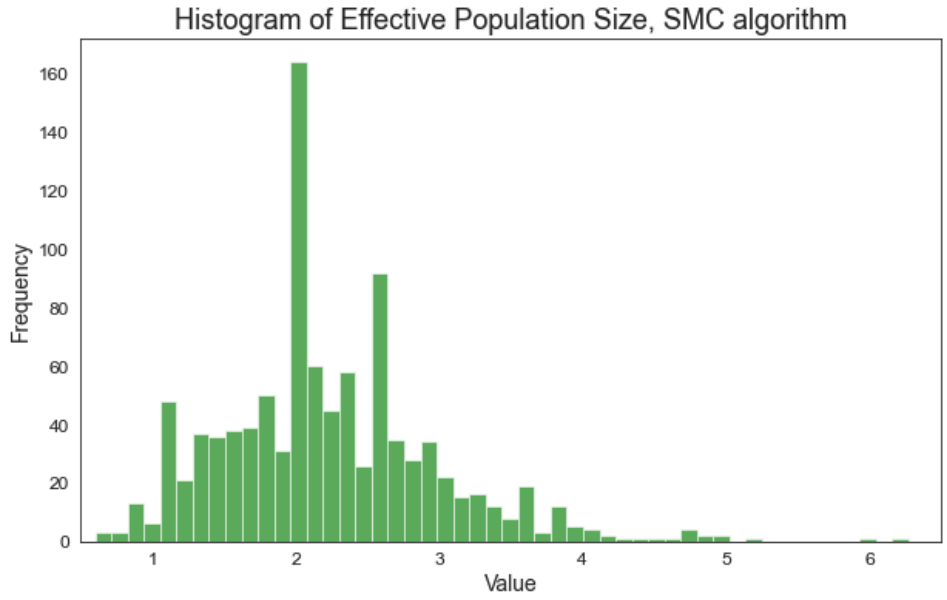

Figure A.7: *Frequency distribution for the parameter Effective Population Size with 5 taxa, using Annealed Adaptive SMC algorithm that we have embedded into BEAST2. Visualization with python matplotlib.*

We can also appreciate the SMC algorithm at work by looking at Figure A.8 below, showing the evolution of the estimated standard deviation of particles vs the annealing step for the parameter Effective Population Size, and we see how the standard deviation drops significantly through the annealing journey

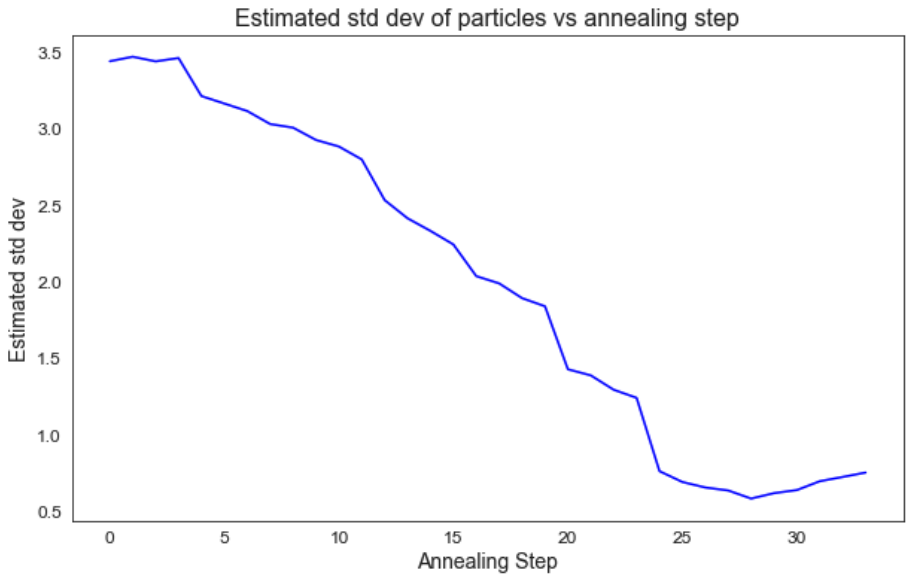

Figure A.8: *Evolution of the estimated standard deviation of particles vs the annealing step for the parameter Effective Population Size in the SMC algorithm: we see how the standard deviation drops significantly through the annealing journey.*

A.3.3 Tree

For the tree analysis we use a methodology similar to [Wang et al., 2019] and we compare trees using the *majority-rule consensus*. So we will have a the *consensus-tree*, which is a "summary" tree for the MCMC run and one for the SMC run, and we will compare them to the generating tree shown in Section A.1 to assess how each of the two algorithms has performed. In addition to visualizing the "summary" trees for the two runs, we will also give a basic topological metric of performance, the Robinson-Foulds (RF) "symmetric difference" metric [Robinson and Foulds, 1981], which will identify possible topology mismatch with the reference tree. The consensus tree has been generated using **TreeAnnotator** and then the visualization using **FigTree**. For the SMC algorithm, the particles have been resampled in order to be able to compare SMC tree samples without the need to consider the particle weights when building the consensus, for ease of calculation.

MCMC results for Tree

The RF metric result for the run is 0, meaning a match from a topological point of view with the reference tree of Section A.1, and we can see from the picture below A.9 the consensus tree created with visualization of the 95% confidence range in the coalescent times (see the comparison with the generator tree of Figure A.1)

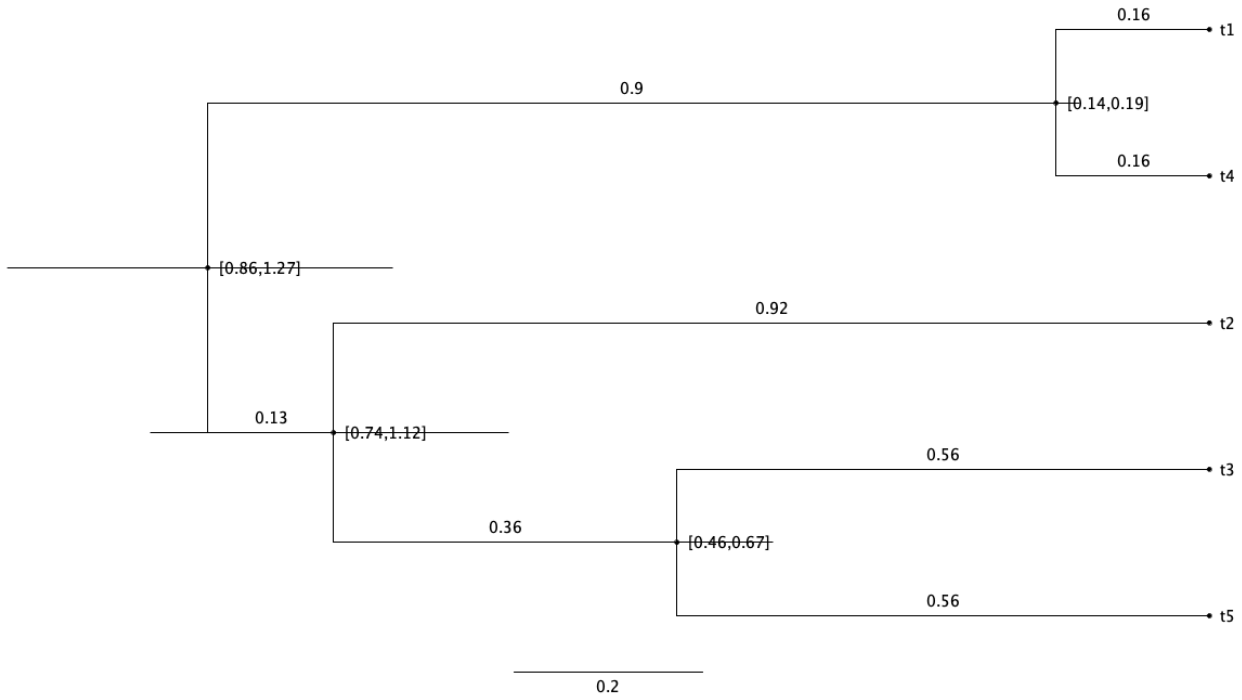

Figure A.9: *Consensus tree for the MCMC run with visualization of* 95% *range for coalescent times. See comparison with the generating tree (which the MCMC run tries to reconstruct) in Figure A.1. Consensus tree has been generated with TreeAnnotator and the visualization is with FigTree (both softwasre from BEAST2 package).*

Ammealed Adaptive SMC results for Tree

The RF metric result for the run is 0, meaning a match from a topological point of view with the reference tree of Section A.1, and we can see from the picture below A.10 the consensus tree created with visualization of the 95% confidence range in the coalescent times (see the comparison with the generator tree of Figure A.1, and with the MCMC-generated consensus tree of Figure A.9)

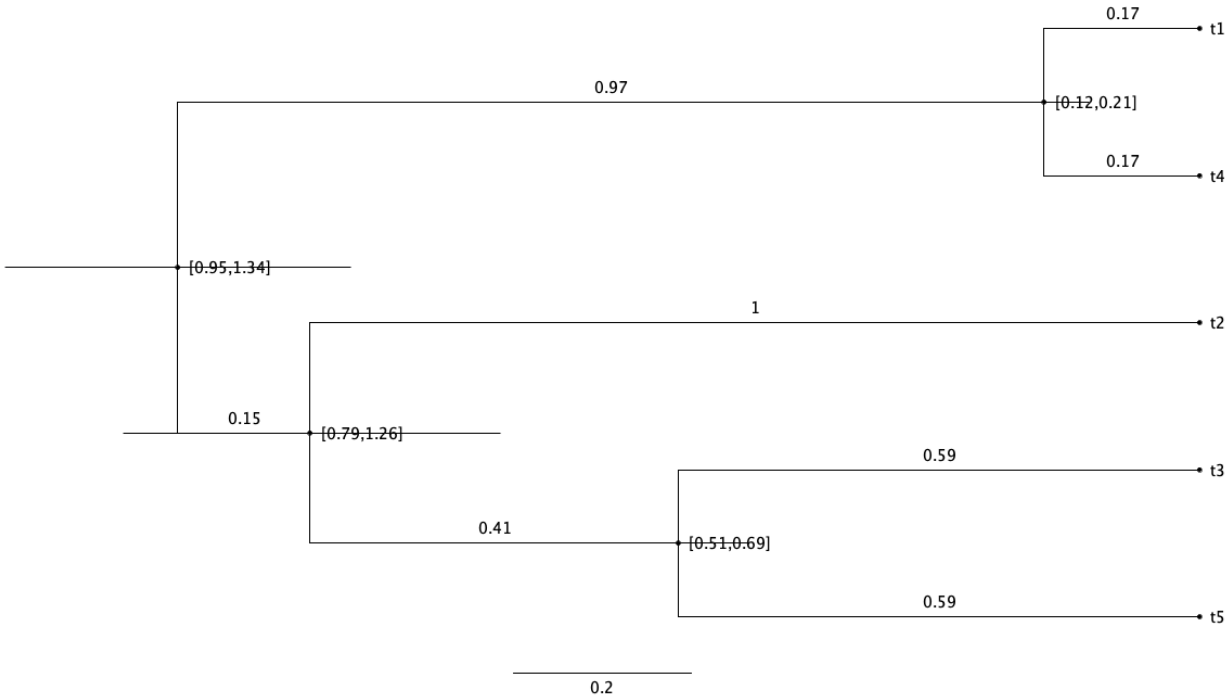

Figure A.10: *Consensus tree for the Annealed Adaptive SMC run with visualization of* 95% *range for coalescent times. See comparison with the generating tree (which the SMC run tries to reconstruct) in Figure A.1, and also with the tree reconstructed using MCMC in Figure A.9: we see that the SMC is able to reconstruct the generating tree well and with a smaller uncertainty (the* 95% *uncertainty ranges in the coalescent times are in general smaller compared to the MCMC of Figure A.9). Consensus tree has been generated with TreeAnnotator and the visualization is with FigTree (both softwasre from BEAST2 package).*

By comparing Figure A.10 with Figure A.9 we can see that the SMC algorithm has been able to reconstruct the generating tree with smaller uncertainty than the MCMC algorithm, in fact the 95% uncertainty ranges in the coalescent times are in general smaller in SMC compared to MCMC.

A.4 Data with 20 Taxa

Following the procedure outlined in Section 3.13, we have generated a synthetic model with 20 taxa.

Generator tree

The first step has been as described in 3.13.1 to generate a coalescent tree with 20 leaves, and the tree has been randomly generated as below in Figure A.11

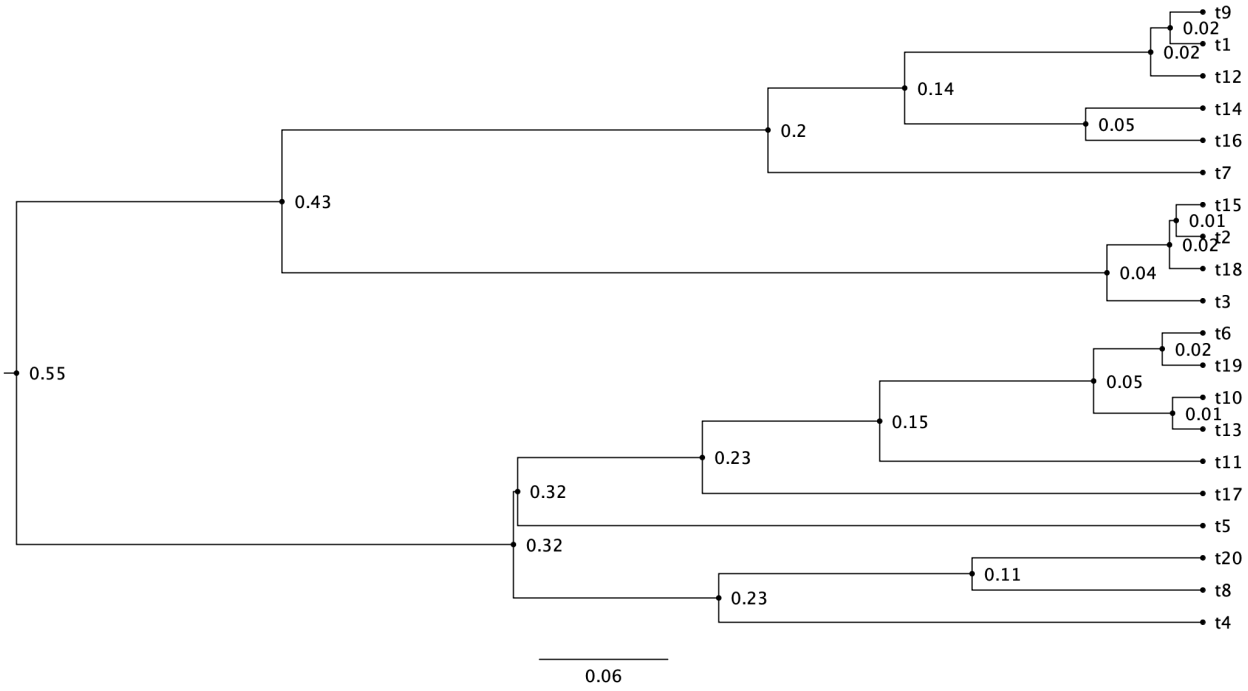

Figure A.11: *Random coalescent tree with 20 leaves generated using the procedure outlined in the first part of Section 3.13.1. This has been the generating tree for the synthetic data of the test described in this section. Visualization via FigTree [Rambaut, 2023]*

Generation of synthetic sequences

Using the tree generated in the previous step, synthetic sequences have been generated using 'seq-gen' program, as explained in Section 3.13.1

A.5 Annealed Adaptive SMC vs MCMC in BEAST2, problem set up with 20 Taxa

The set up is similar to what done in Section A.2 for 5 taxa, where it is fully explained, therefore please refer to that section for the details.

A.5.1 SMC set up

The SMC has been set up with 1000 particles, and 5 MCMC moves per each annealing step. The number of annealing steps adaptively determined by the CESS (see Section 2.6.3 for details on CESS) has been 96, as can be seen in Figure A.12

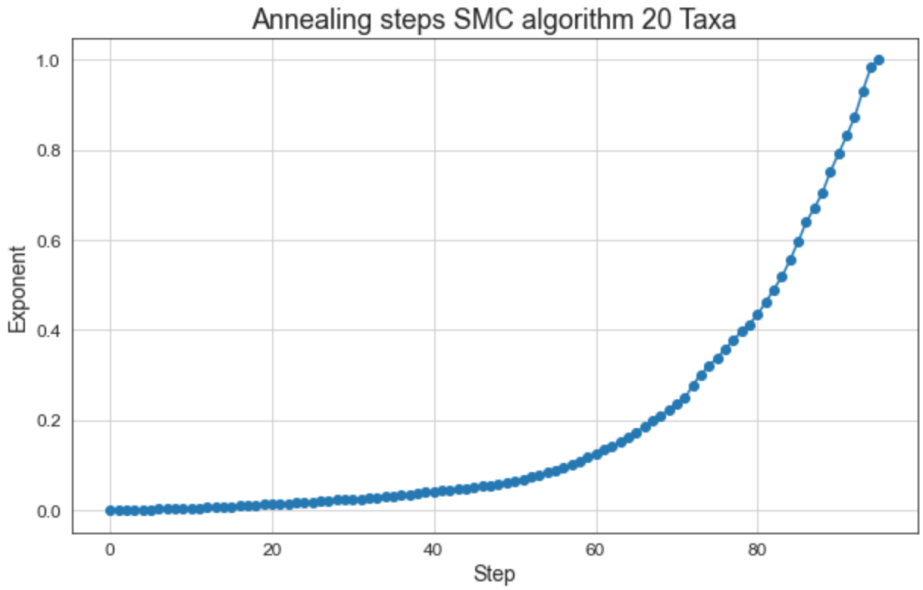

Figure A.12: *Annealing steps in the SMC run for the 20-taxa example studied in this section.*

Therefore the total number of likelihood evaluation for the algorithm has been $1000 \times 96 \times 5 = 480000$. The adaptive annealing steps have been determined using CESS with a threshold of 90%, and resampling of particles is done when ESS falls below 50% of particles, we can see below in Figure A.13 the ESS chart related to the SMC run

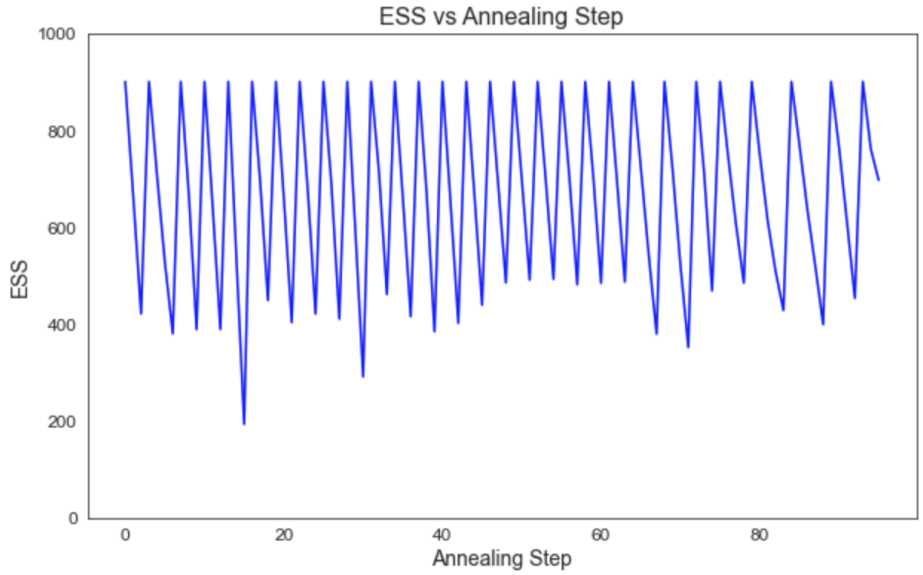

Figure A.13: *ESS versus annealing SMC step for the 20-taxa example studied in this section. Resampling is performed whenever the ESS falls below* 50% *of particles (1000 particles are used for the simulation).*

A.5.2 MCMC set up

Considering that the total number of likelihood evaluations from the Annealed Adaptive SMC was 480000, we have used a comparison similar to [Wang et al., 2019] and therefore we have used a number of MCMC iterations greater than +20% compared to the SMC run, in our MCMC simulation we have used 600000 iterations.

A.6 Annealed Adaptive SMC vs MCMC in BEAST2: Results with 20 Taxa

As we have done previously for 5 and 10 taxa in Sections A.3 and 3.16 respectively, we'll report here the results for 20 taxa in the analysis of the following components of the state space:

- Gamma shape
- Effective population size
- Coalescent Tree

A.6.1 Gamma shape

The true value of the gamma shape parameter (i.e. the value with which the data has been generated) is 1. As explained already already in Sections A.3.1 and 3.16.1, due to the fact that the probability of MCMC moves on the gamma shape parameter has been kept to the default value that the configuration software BEAUTi (see cection 3.9.1) uses, and MCMC moves are less likely to happen than moves on effective population size and trees, we see a a low ESS in the MCMC run, and also the empirical distributions are rather scattered (see following Figures A.14 and A.15).

MCMC results for Gamma shape

The mean of the MCMC run is close to the true value of 1, we can see the full statistics in the following table. Like in the 5 and 10 taxa cases for gamma shape (see the MCMC part of Section A.3.1 and 3.16.1), the ESS is very low and the distribution, as we can appreciate in the following Figure A.14, is rather scattered:

Statistic	Value
Mean	0.92
Standard Deviation	0.066
Value Range	[0.776, 1.107]
95% HPD Interval	[0.7829, 1.0403]
Effective Sample Size (ESS)	142

And the distribution of the Gamma shape values is in Figure A.14

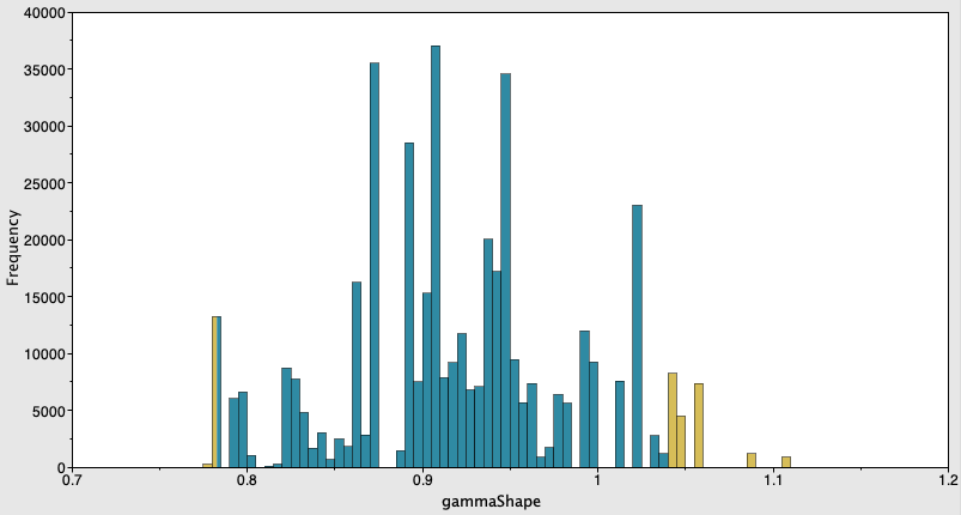

Figure A.14: *Frequency distribution using the native MCMC run with BEAST2 for the parameter Gamma shape with 20 taxa. Visualization with the software Tracer.*

Ammealed Adaptive SMC results for Gamma shape

The Statistics for the SMC run are similar to the MCMC:

Statistic	Value
Mean	1.04
Standard Deviation	0.038
Value Range	[0.709, 1.096]
95% HPD Interval	[0.95, 1.08]

And the distribution of the Gamma shape values is in Figure A.15

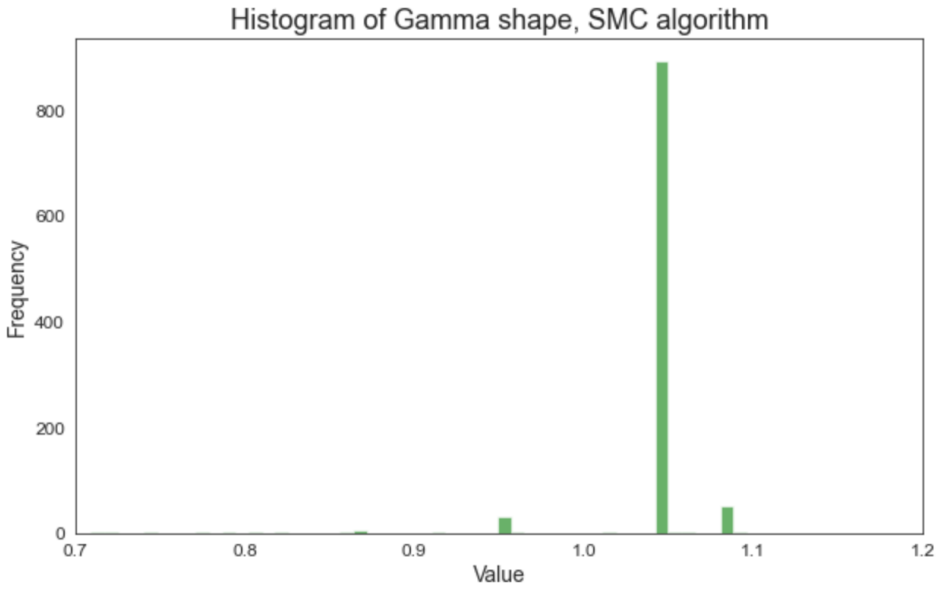

Figure A.15: *Frequency distribution for the parameter Gamma shape with 20 taxa, using Annealed Adaptive SMC algorithm that we have embedded into BEAST2. Visualization with python matplotlib.*

A.6.2 Effective Population Size

See Section A.3.2 for the technical details on the parameter, below we can see the statistics and the visualization of the distributions in MCMC and SMC.

MCMC results for Effective Population Size

The mean of the MCMC run is 1.73, we can see the full statistics in the following table

Statistic	Value
Mean	0.93
Standard Deviation	0.244
Value Range	[0.344, 2.383]
95% HPD Interval	[0.52, 1.395]
Effective Sample Size (ESS)	2109

And the distribution of the Effective Population Size is in Figure A.16

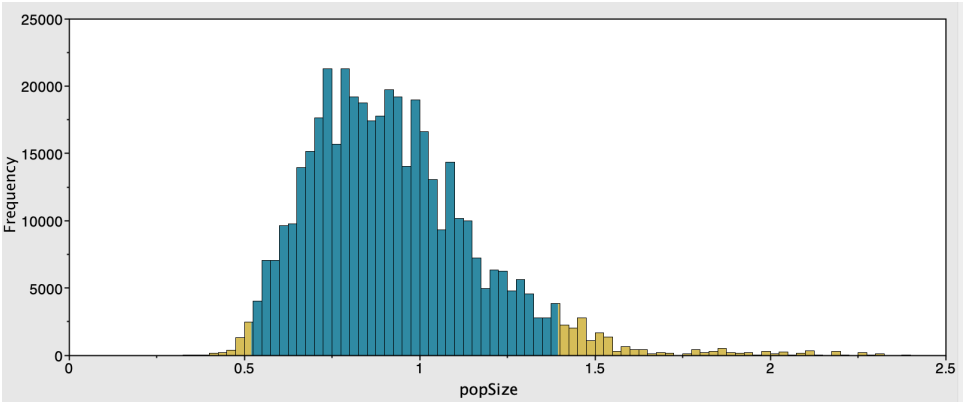

Figure A.16: *Frequency distribution using the native MCMC run with BEAST2 for the parameter Effective Population Size with 20 taxa. Visualization with the software Tracer.*

Ammealed Adaptive SMC results for Effective Population Size

The Statistics for the SMC run are comparable to the MCMC run, we can notice a peak in the distribution of figure , it may be due to a mixture of not enough MCMC moves on the parameter and some degree of degeneracy of particles:

Statistic	Value
Mean	1.075
Standard Deviation	0.21
Value Range	[0.44, 1.91]
95% HPD Interval	[0.62, 1.52]

And the distribution of the Effective Population Size is in Figure A.17

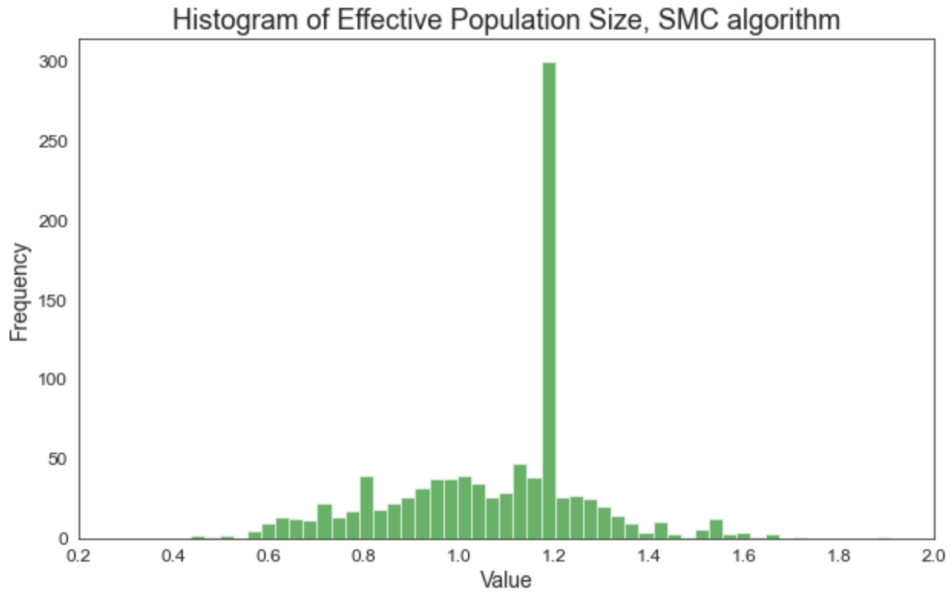

Figure A.17: *Frequency distribution for the parameter Effective Population Size with 20 taxa, using Annealed Adaptive SMC algorithm that we have embedded into BEAST2. Visualization with python matplotlib.*

We can also appreciate the SMC algorithm at work by looking at Figure A.18 below, showing the evolution of the estimated standard deviation of particles vs the annealing step for the parameter Effective Population Size, and we see how the standard deviation drops significantly through the annealing journey

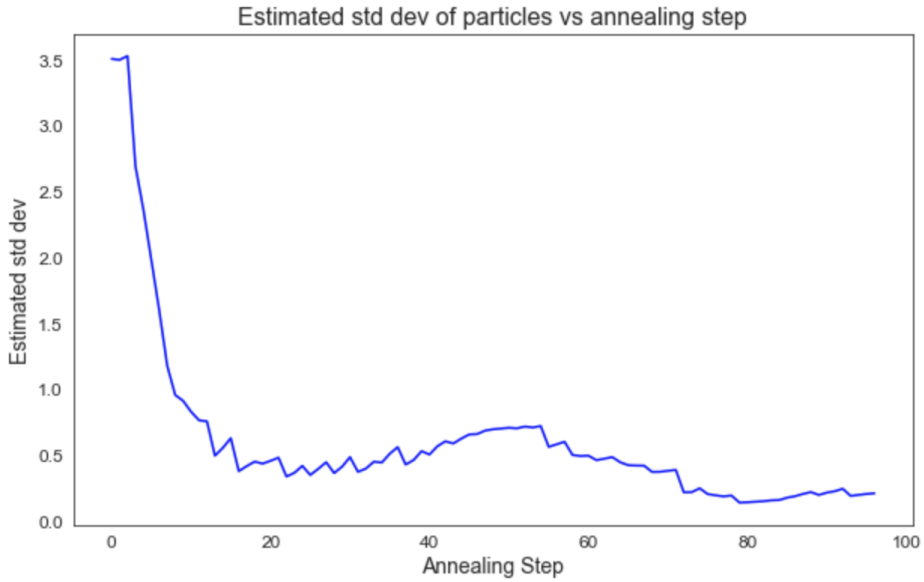

Figure A.18: *Evolution of the estimated standard deviation of particles vs the annealing step for the parameter Effective Population Size in the SMC algorithm: we see how the standard deviation drops significantly through the annealing journey.*

A.6.3 Tree

For tree analysis please refer to the 5-taxa Section A.3.3 where full technical details are given. For the SMC algorithm, the particles have been resampled in order to be able to compare SMC tree samples without the need to consider the particle weights when building the consensus, for ease of calculation (unlike what has been done in gamma shape and Effective Population Size where particles have been weighted to calculate the Statistics). In addition, differently from the 5 and 10 taxa cases of Sections A.3.3 and 3.16.3 respectively, we have omitted the 95% ranges in Figures A.19 for MCMC and A.20 for SMC for ease of visualization as it would have been difficult to visualize with many nodes and branches, anyway similar conclusions to the 5 and 10 taxa cases can be drawn for the 20 taxa case as well.

MCMC results for Tree

The RF metric result for the run is 0, meaning a match from a topological point of view with the reference tree of Section A.4, and we can see from the picture below A.19 the consensus tree (see the comparison with the generator tree of Figure A.11)

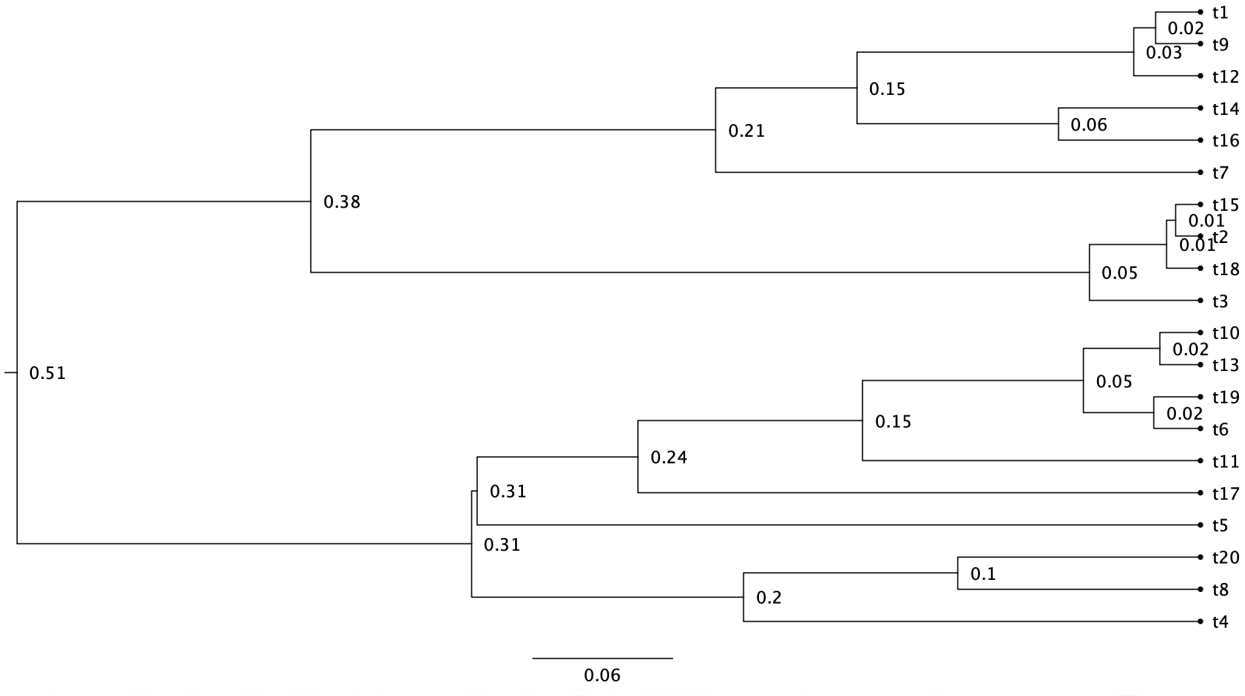

Figure A.19: *Consensus tree for the MCMC run. See comparison with the generating tree (which the MCMC run tries to reconstruct) in Figure A.11. Consensus tree has been generated with TreeAnnotator and the visualization is with FigTree (both softwasre from BEAST2 package).*

Ammealed Adaptive SMC results for Tree

The RF metric result for the run is 0, meaning a match from a topological point of view with the reference tree of Section A.4, and we can see from the picture below A.20 the consensus tree (see the comparison with the generator tree of Figure A.11, and with the MCMC-generated consensus tree of Figure A.19)

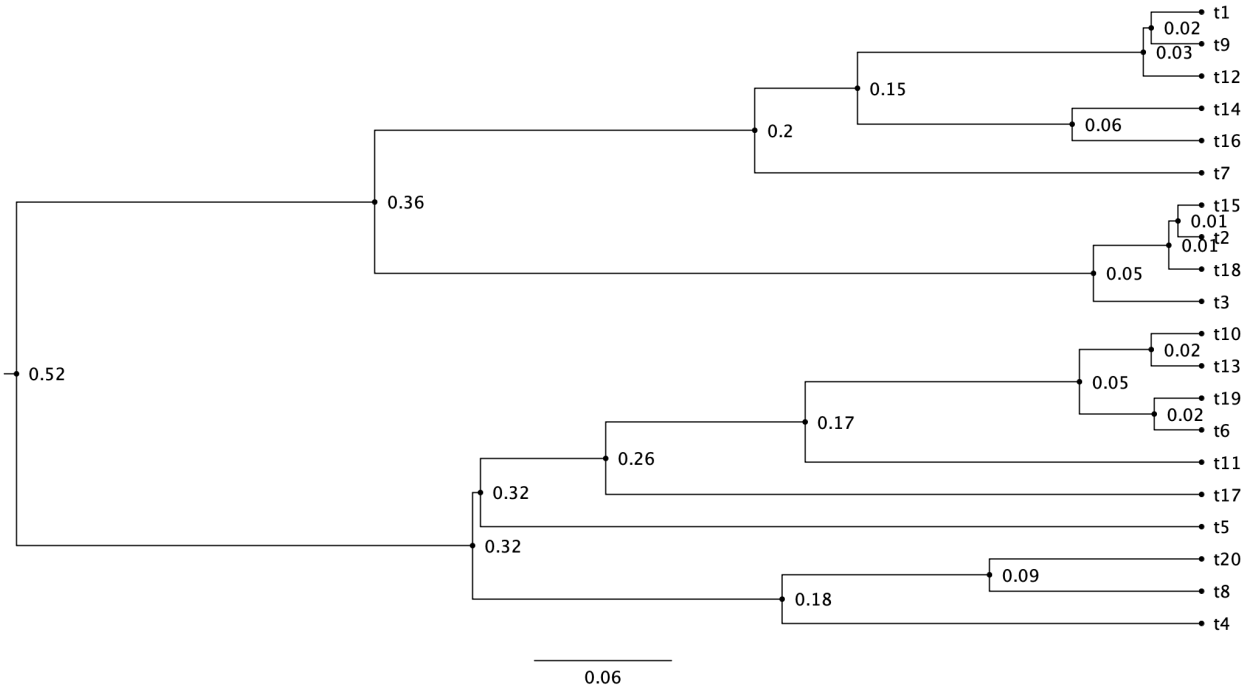

Figure A.20: *Consensus tree for the Annealed Adaptive SMC run. See comparison with the generating tree (which the SMC run tries to reconstruct) in Figure A.11, and also with the tree reconstructed using MCMC in Figure A.19: we see that the SMC is able to reconstruct the generating tree with similar level of accuracy as the MCMC. Consensus tree has been generated with TreeAnnotator and the visualization is with FigTree (both softwasre from BEAST2 package).*

By comparing Figure A.20 with Figure A.19 we can see that the SMC algorithm has been able to reconstruct the generating tree with similar level of accuracy as the MCMC algorithm.

Appendix B

Bayesian inverse problem in Active Subspace

We use quite a few examples of Bayesian inverse problems in the Active Subspace Chapters 4 and subsequent, we give here the basic concepts. We consider a classical Bayesian inverse problem reported in [Constantine et al., 2016], and we follow the same notation of the paper: we have a model with additive noise, as follows

$$d = m(x) + e, e \sim \mathcal{N}(0, \sigma^2 \mathbb{I}) \tag{B.1}$$

It is assumed for simplicity that the Gaussian noise e has uncorrelated components and therefore diagonal covariance matrix.
Given N independent measurements of (B.1)

$$d_i = m(x) + e_i \tag{B.2}$$

The likelihood of the system is (see for example [Najm, 2018] for a nice and clear derivation)

$$\rho_{lik}(d, x) = \prod_{i=1}^{N} p(d_i|x) \tag{B.3}$$

We show the one-dimensional case below of the the density $p(d_i|x)$, where we recognize the familiar Gaussian form

$$p(d_i|x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(d_i - m(x))^2}{2\sigma^2} \right) \tag{B.4}$$

The product (B.3) is a result of the independence of the measurements, and that it results in a product of Gaussians with members as in (B.4) is a consequence of the Gaussian noise assumption and of the components of noise being uncorrelated.
In similar fashion and for a more general multi-dimensional parameter space, the likelihood

can be expressed in the compact form (neglecting for brevity the multiplying constants)

$$\rho_{lik}(d, x) \propto \exp\left(- \frac{||d - m(x)||^2}{2\sigma^2} \right) \tag{B.5}$$

Calculation of likelihood-informed Active Subspace in the Bayesian inverse problem with additive noise

We choose as function f of (4.8) the negative log-likelihood as seen in equation (4.16), which will be, as it is clear from its formulation in equation (B.6), a measure of the *data misfit*. By applying the negative-log to (B.5), we have

$$f(x) = \frac{||d - m(x)||^2}{2\sigma^2} \tag{B.6}$$

We know that the first necessary pre-processing step is to use the gradient of f to find the directions along which f varies the most (see equation (4.2) and subsequent). The gradient of f in (B.6) can be easily calculated

$$\nabla f(x) = \frac{1}{\sigma^2} \nabla m(x)^T (d - m(x)) \tag{B.7}$$

Now that we have the gradient from equation (B.7), in order to build the matrix C of (4.2) we have to perform an integration of $\nabla f(x) \nabla f(x)^T$ against the posterior density of our problem. Since we have to assume that integrating on the posterior, and even drawing directly *iid* samples from the posterior and use $\hat{C}$ of (4.12), is not convenient or tractable (otherwise we would not use the MCMC to approximate it, in the first place), we choose the approximation that we called $\hat{C}_{pri}$ in (4.28), obtained by (4.12) when we sample from the prior distribution.

Appendix C

AS-MH algorithm in cases of perfect Active Subspaces

This section relates to the simplified version of Algorithm 8 of Section 4.8.2. In case of perfectly inactive variables, equation (4.25) becomes an exact marginal, not a pseudo-marginal, and in this case Algorithm 8 simplifies in Algorithm 21 which becomes a MCMC on the targeting the marginal posterior $\pi_a = p_a l_a$

Alg. 21 Exact Active Subspaces Based MCMC

1: Compute the AS and using the procedure outlined in Section 4.2.3 estimate matrices B_a and B_i.
2: Initialize the algorithm by choosing an initial value a^1 and calculate $l_a(a^1)$.
3: **for** $k = 2$ to T **do**
4: $\quad a^* \sim q_a(\cdot|a^{k-1})$.
5: $\quad$ Calculate $l_a(a^*)$
6: $\quad$ Set $a^k = a^*$ with probability $1 \wedge \frac{p_a(a^*)l_a(a^*)q_a(a^*|a^{t-1})}{p_a(a^{k-1})l_a(a^{k-1})q_a(a^{t-1}|a^*)}$
7: $\quad$ Else let $a^k = a^{k-1}$.
8: **end for**

see the differences between line 5 of Algorithm 21 where an exact marginal is used and line 5 of Algorithm 8 where an estimate was used through the calculation of a pseudo-marginal.

Appendix D

AS-MCMC algorithms: number of output samples per likelihood evaluation

When comparing the performances of the Active Subspaces algorithms presented in Chapter 5, we have tried to keep the number of likelihood evaluations constant in each run, to ensure a fair comparison. Due to the structure of the algorithms, the same number of likelihood evaluations may result in a different number of output samples. Taking a reference figure of 100000 likelihood evaluations, it will result in:

- Standard MCMC: 100000 iterations, and therefore as many samples;
- AS-MH: if we use 10 inactive variables in the pseudo-marginal calculation, we will have 10000 outer MCMC iterations ($10000 \times 10 = 100000$);
- AS-PMMH: if we use 10 inner inactive variables and 6 tempering steps of the inner SMC sampler,then we have 1666 outer MCMC steps and therefore as many output samples ($1666 \times 10 \times 6 = 99960$ which is the closest integer to 100000);
- AS-PMMH-i: same of AS-PMMH, only roles of active and inactive are inverted;
- AS-Gibbs: considering that we first operate on the inactive and then active, 100000 likelihood evaluations are done in 50000 iterations and therefore produce 50000 output samples.

We summarise in Table D.1

Method	Nr of output samples
MCMC	100000
AS-MH*	10000
AS-PMMH	1666
AS-PMMH-i	1666
AS-Gibbs	50000

Table D.1: Comparison of number of output samples when performing 100000 likelihood evaluations in different MCMC methods. **For AS-MH the figure in the table indicates number of samples when one inactive sample is used per active variable, like in formula (4.23). If instead all inactive particles are used, like in formula (4.24), the relative figure must be multiplied by the number of inactive variables used,* 10 *in this case.*

In Table 5.1, for AS-MH the figure in the table indicates number of samples when one inactive sample is used per active variable, like in formula (4.23). If instead all inactive particles are used, like in formula (4.24), the relative figure must be multiplied by 10 (number of inactive variables used) to consider all the samples.

For ease of reference in some of the sections, we also report the table for 200000 likelihood evaluations (it is the above Table D.1 with numbers $\times 2$)

Method	Nr of output samples
MCMC	200000
AS-MH*	20000
AS-PMMH	3332
AS-PMMH-i	3332
AS-Gibbs	100000

Table D.2: Comparison of number of output samples when performing 200000 likelihood evaluations in different MCMC methods, this is the equivalent of Table 5.1, adapted for 200000. **For AS-MH the figure in the table indicates number of samples when one inactive sample is used per active variable, like in formula (4.23). If instead all inactive particles are used, like in formula (4.24), the relative figure must be multiplied by the number of inactive variables used,* 10 *in this case.*

Bibliography

S. Agapiou, O. Papaspiliopoulos, D. Sanz-Alonso, and A. M. Stuart. Importance sampling: Intrinsic dimension and computational cost. *Statist. Sci.*, 32(3):405–431, 2017. doi: 10.1214/17-STS611. URL https://projecteuclid.org/euclid.ss/1504253124.

Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell*. Garland Science, 4 edition, 2002.

M Amaya, N Linde, and E Laloy. Adaptive sequential Monte Carlo for posterior inference and model selection among complex geological priors. *Geophysical Journal International*, 226(2):1220–1238, 04 2021. ISSN 0956-540X. doi: 10.1093/gji/ggab170. URL https://doi.org/10.1093/gji/ggab170.

Macarena Amaya, Niklas Linde, and Eric Laloy. Hydrogeological multiple-point statistics inversion by adaptive sequential monte carlo. *Advances in Water Resources*, 166: 104252, 2022. ISSN 0309-1708. doi: https://doi.org/10.1016/j.advwatres.2022.104252. URL https://www.sciencedirect.com/science/article/pii/S0309170822001221.

C. Andrieu and G. O. Roberts. The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2), 2009. doi: 10.1214/07-AOS574.

Christophe Andrieu and Matti Vihola. Convergence properties of pseudo-marginal markov chain monte carlo algorithms. *The Annals of Applied Probability*, 25(2), April 2015. ISSN 1050-5164. doi: 10.1214/14-aap1022. URL http://dx.doi.org/10.1214/14-AAP1022.

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010. doi: 10.1111/j.1467-9868.2009.00736.x.

M. A. Beaumont. Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):11391160, 2003.

A. Beskos, D. Crisan, and A. Jasra. On the stability of sequential monte carlo methods in high dimensions. *Ann. Appl. Probab.*, 24(4):1396–1445, 2014. doi: 10.1214/13-AAP951.

Alexandros Beskos, Dan Crisan, Ajay Jasra, and Nick Whiteley. Error bounds and normalizing constants for sequential monte carlo in high dimensions. 2011.

R. Bouckaert and P. Lockhart. Capturing heterotachy through multi-gamma site models. *bioRxiv*, 2015. doi: 10.1101/018101.

Remco Bouckaert, Tim G. Vaughan, Joëlle Barido-Sottani, Sebastian Duchêne, Mathieu Fourment, Anastasia Gavryushkina, Joseph Heled, Graham Jones, Denise Khnert, Nicola De Maio, Michael Matschiner, Filipe K. Mendes, Nicola F. Mller, Hlne A. Ogilvie, Louis du Plessis, Alex Popinga, Andrew Rambaut, David Rasmussen, Sagi Snir, Chi Zhang, Tanja Stadler, Philippe Lemey, Marc A. Suchard, David A. Westfall, Guy Baele, and Alexei J. Drummond. Beast 2.5: An advanced software platform for bayesian evolutionary analysis. *PLOS Computational Biology*, 15(4):e1006650, 2019. doi: 10.1371/journal.pcbi.1006650.

R. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta Numerica*, 7:1–49, 1998. doi: 10.1017/S0962492900002804.

N. Chopin. A sequential particle filter method for static models. *Biometrika*, 89(3):539–551, 2002.

Nicolas Chopin, Pierre E. Jacob, and Omiros Papaspiliopoulos. Smcˆ2: an efficient algorithm for sequential analysis of state-space models, 2012.

P. G. Constantine. *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*. Society for Industrial and Applied Mathematics, 2015. ISBN 1611973856.

P. G. Constantine, C. Kent, and T. Bui-Thanh. Accelerating markov chain monte carlo with active subspaces. *SIAM Journal on Scientific Computing*, 2016. doi: 10.1137/15m1042127.

C. Cui, K. Zhang, T. Daulbaev, J. Gusak, I. V. Oseledets, and Z. Zhang. Active subspace of neural networks: Structural analysis and universal attacks. *CoRR*, abs/1910.13025, 2019. URL http://arxiv.org/abs/1910.13025.

P. Del Moral and A. Doucet. Sequential monte carlo samplers. 2003. Condensed Matter (cond-mat) CUED-F-INFENG-443, arXiv:cond-mat/0212648.

A. J. Drummond, G. K. Nicholls, A. G. Rodrigo, and W. Solomon. Estimating mutation parameters, population history and genealogy simultaneously from temporally spaced sequence data. *GENETICS*, 161(3):1307–1320, 2002.

Alexei J. Drummond and Remco R. Bouckaert. *Bayesian Evolutionary Analysis with BEAST*. Cambridge University Press, Cambridge, 2015.

Alexei J Drummond and Andrew Rambaut. Beast: Bayesian evolutionary analysis by sampling trees. *BMC evolutionary biology*, 7(1):214, 2007.

Alexei J Drummond, Marc A Suchard, Dong Xie, and Andrew Rambaut. Bayesian phylogenetics with beauti and the beast 1.7. *Molecular biology and evolution*, 29(8):1969–1973, 2012.

V. Elvira, L. Martino, and C. P. Robert. Rethinking the effective sample size. 2018. arXiv: 1809.04129.

J. Felsenstein. Evolutionary trees from dna sequences: A maximum likelihood approach. *J Mol Evol*, 17:368–376, 1981.

R. A. Fisher. The genetical theory of natural selection. 1930.

Andrew Gelman, Walter R. Gilks, and Gareth O. Roberts. Weak convergence and optimal scaling of random walk metropolis algorithms. *Annals of Applied Probability*, 7(1):110–120, 1997.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 3 edition, 2013. doi: 10.1201/b16018.

Donald Geman and Stefanie Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

Edward I. George and Robert E. McCulloch. Approaches for bayesian variable selection. *Statistica Sinica*, 7(2):339–373, 1997.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97109, 1970.

J. Hein, M. Schierup, and C. Wiuf. Gene genealogies, variation and evolution: a primer in coalescent theory. 2004.

Joseph Heled and Alexei J. Drummond. Bayesian inference of population size history from multiple loci. *BMC Evolutionary Biology*, 8:289, Oct 2008. ISSN 1471-2148. doi: 10.1186/1471-2148-8-289. URL https://pubmed.ncbi.nlm.nih.gov/18947398.

Zheng Hou, Xiaoya Ma, Xuan Shi, Xi Li, Lingxiao Yang, Shuhai Xiao, Olivier De Clerck, Frederik Leliaert, and Bojian Zhong. Phylotranscriptomic insights into a mesoproterozoicneoproterozoic origin and early radiation of green seaweeds (ulvophyceae). *Nature Communications*, 13(1):1610, 2022. ISSN 2041-1723. doi: 10.1038/s41467-022-29282-9. URL https://doi.org/10.1038/s41467-022-29282-9. Open Access. This article is licensed under a Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution, and reproduction in any medium or format, as long as appropriate

Hemant Ishwaran and J Sunil Rao. Spike and slab variable selection: Frequentist and bayesian strategies. *The Annals of Statistics*, 33(2):730–773, 2005.

X. Jiao. Using surrogate distributions to improve the convergence properties of gibbs-type samplers. 2017. Doctor of Philosophy (PhD) thesis, Imperial College London, Mathematics Department. Supervisor: van Dyk, D. DOI: https://doi.org/10.25560/67577.

Adam M. Johansen and L. Evers. Monte carlo methods lecture notes. 2007. University Of Bristol.

Thomas H Jukes and Charles R Cantor. Evolution of protein molecules. *Mammalian Protein Metabolism*, 1969.

J. Kingman. The coalescent. *Stochastic processes and their applications*, 13(3):235248, 1982.

A. Kong. A note on importance sampling using standardized weights. 1992. University of Chicago, Dept. of Statistics, Tech. Rep 348.

A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and bayesian missing data problems. *Journal of the American Statistical Association*, 9:278–288, 1994.

N. Metropolis and S. Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247), 1949.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:10871091, 1953.

Toby J. Mitchell and John J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

H. N. Najm. Statistical inverse problems and bayesian inference. 2018. Sandia National Laboratories, Livermore, CA, USA. UQ Lecture Series at the American University of Beirut, Beirut, Lebanon, April 23-27, 2018. URL: https://www.osti.gov/servlets/purl/1508912.

Radford M. Neal. Annealed importance sampling. 1998.

A. Owen. Monte carlo theory, methods and examples. 2013. Available at: http://statweb.stanford.edu/owen/mc/.

M. M. T. Parente. Active subspaces in bayesian inverse problems. 2020. Doctoral Dissertation at Technische Universitt Mnchen, Munich, Germany. URL: https://mediatum.ub.tum.de/doc/1546065/1546065.pdf.

Technologies Inc. Plotly. Collaborative data science, 2015. URL https://plot.ly.

Andrew Rambaut. Figtree, 2023. URL http://tree.bio.ed.ac.uk/software/figtree/.

Andrew Rambaut and Nicholas C Grassly. Seq-gen: an application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Computer applications in the biosciences: CABIOS*, 13(3):235–238, 1997.

Douglas Reynolds. *Gaussian Mixture Models*, pages 659–663. Springer US, Boston, MA, 2009. ISBN 978-0-387-73003-5. doi: 10.1007/978-0-387-73003-5_196. URL https://doi.org/10.1007/978-0-387-73003-5_196.

C. P. Robert and G. Casella. Monte carlo statistical methods. 2004.

G. Roberts and J. Rosenthal. Optimal scaling for various metropolis-hastings algorithms. *Statistical Science*, 16, 2001. doi: 10.1214/ss/1015346320.

D. F. Robinson and Les R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53:131–147, 1981.

Håvard Rue, Sara Martino, and Nicolas Chopin. Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(2):319–392, 2009.

I. Schuster, P. G. Constantine, and T. J. Sullivan. Exact active subspace metropolis-hastings, with applications to the lorenz-96 system. 2017. arXiv:stat.CO/1712.02749.

P. Tataru, M. Simonsen, T. Bataillon, and A. Hobolth. Statistical inference in the wright-fisher model using allele frequency data. *Syst Biol.*, 66(1):e30–e46, 2017. doi: 10.1093/sysbio/syw056.

Simon Tavaré. Some probabilistic and statistical problems in the analysis of dna sequences. *Lectures on Mathematics in the Life Sciences*, 17:57–86, 1986.

Luke Tierney. Markov chains for exploring posterior distributions. *Annals of Statistics*, 22 (4):1701–1728, 1994.

Luke Tierney and Joseph B Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393), 1986.

Dootika Vats, James M Flegal, and Galin L Jones. Multivariate output analysis for Markov chain Monte Carlo. *Biometrika*, 106(2):321–337, 04 2019. ISSN 0006-3444. doi: 10.1093/biomet/asz002. URL https://doi.org/10.1093/biomet/asz002.

Liangliang Wang, Shijia Wang, and Alexandre Bouchard-Ct. An Annealed Sequential Monte Carlo Method for Bayesian Phylogenetics. *Systematic Biology*, 69(1):155–183, 06 2019. ISSN 1063-5157. doi: 10.1093/sysbio/syz028. URL https://doi.org/10.1093/sysbio/syz028.

S. Wright. Evolution in mendelian populations. *Genetics*, 16:97159, 1931.

F. Wu, S. Zhao, B. Yu, Y. M. Chen, W. Wang, Z. G. Song, Y. Hu, Z. W. Tao, J. H. Tian, Y. Y. Pei, M. L. Yuan, Y. L. Zhang, F. H. Dai, Y. Liu, Q. M. Wang, J. J. Zheng, L. Xu, E. C. Holmes, and Y. Z. Zhang. A new coronavirus associated with human respiratory disease in china. *Nature*, 579(7798):265–269, 2020. doi: 10.1038/s41586-020-2008-3.

Z. Yang. Maximum likelihood phylogenetic estimation from dna sequences with variable rates over sites: Approximate methods. *J. Mol Evol*, 39:306–314, 1994.

Guangchuang Yu. *Data Integration, Manipulation and Visualization of Phylogenetic Trees*, chapter 1. Chapman and Hall/CRC, 1 edition, 2022. ISBN 9781003279242. doi: 10.1201/9781003279242. URL https://doi.org/10.1201/9781003279242.

Yan Zhou, Adam M Johansen, and John A D Aston. Towards automatic model comparison: An adaptive sequential monte carlo approach. 2013. doi: 10.48550/ARXIV.1303.3123.