University of Leipzig

Doctoral Thesis

Probabilistic models of natural language semantics

Author:
Ingmar Schuster

Supervisor:
Prof. Dr. Gerhard Heyer

A thesis submitted in fulfilment of the requirements
for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

June 2015

Declaration of Authorship

I, Ingmar SCHUSTER, declare that this thesis titled, 'Probabilistic models of natural language semantics' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY OF LEIPZIG (IN BLOCK CAPITALS)

Abstract

Faculty Name
Department or School Name

Doctor of Philosophy

Probabilistic models of natural language semantics

by Ingmar Schuster

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgements and the people to thank go here, don't forget to include your project advisor...

Contents

List of Figures

List of Tables

Abbreviations

LAH **L**ist **A**bbreviations **H**ere

Physical Constants

Speed of Light $c = 2.997\ 924\ 58 \times 10^8\ \mathrm{ms}^{-\mathrm{S}}$ (exact)

Symbols

a	distance	m
P	power	W (Js^{-1})
ω	angular frequency	rads^{-1}

For/Dedicated to/To my...

Chapter 1

Adaptive Monte Carlo

1.1 Introduction

In this exposition to Adaptive MCMC I largely follow the review by Rosenthal (2011). I will discuss adaptivity for special cases of Metropolis-Hastings, as this is the case which has received the most attention in the literature. Tuning parameters of the sampling algorithm in MCMC, before the rise of adaptive MCMC, had to be done manually. This involves running the sampling algorithm several times and checking its performance. For an applied problem this performance check could be average loss with respect to a pre-specified loss function. Another possibility would be to try to get low Monte Carlo variance, an estimate of which is available in black box problems, i.e. problems in which the ground truth is unknown (the only case where one would apply MCMC anyway). Simply lowering Monte Carlo variance of course holds the danger of increasing the algorithms bias without noticing. We will discuss this in section 1.1.1. As we will see in section 1.2 though, in the MH-case one can also resort to tuning the algorithm for a certain acceptance rate. The theoretical results proving the optimality of an acceptance rate for a particular algorithm typically make strong assumptions regarding the target distribution f. However if these assumptions are violated, these acceptance rates still often times give good results experimentally.

A natural research direction then is to develop algorithms that adapt automatically to the target by using the information from collected samples. This way, one could hope to get rid of the time-consuming manual tuning. A natural first step in this direction is to adapt algorithm parameters in order to hit optimal acceptance rates. Another would be to adapt the proposal distribution to the posterior in some way.

1.1.1 Criteria for evaluating MCMC algorithms

In this section, I will focus on evaluation criteria that only make use of the output of the Markov Chain (as opposed to using criteria such as average loss on a dataset). First, we can compare the *speed of convergence* of the Markov Chain. A Markov Kernel κ_1 is said to converge faster than κ_2 if

$$sup_{A\subset\mathcal{X}}|\kappa_1^n(x,A) - f(A)| \leq sup_{A\subset\mathcal{X}}|\kappa_2^n(x,A) - f(A)|$$

Check: can I really use "markov kernel" here instead of "markov chain" as in Rosenthal? Is the total variation distance correct? shouldn't it be $A \in F$ where F is the Sigma-Algebra on the statspace $\mathcal{X}$? for any number of applications of the kernels n and any x. Here $sup_{A\subset\mathcal{X}}|\cdot - \cdot|$ is the total variation distance between two distributions, f is the target density as usual and $\mathcal{X}$ is the state space of the chain. The total variation distance is closely related to the Kullback-Leibler distance D_{KL} in that D_{KL} provides a tight upper bound via Pinsker's inequality (see Lemma 2.5 in Tsybakov, 2008).
Given some integrand h, κ_1 has *lower variance* than κ_2 if

$$\mathbb{V}(\sum_{i=1}^{n} h(X_i^{\kappa_1})/n) \leq \mathbb{V}(\sum_{i=1}^{n} h(X_i^{\kappa_2})/n)$$

where $X_i^{\kappa_j}$ is a sample from κ_j. This of course depends on what integrand h is to be computed, on the number of samples n and on the starting state of the chains.
If κ leaves f invariant (i.e. the Markov Kernel is ergodic or stationary wrt f), then for large n we have $\mathbb{V}(\sum_{i=1}^{n} h(X_i^{\kappa})/n) \approx n^{-1}\mathbb{V}_f(h)\tau_h^{\kappa}$ where

$$\tau_h^{\kappa} = \sum_{k=-\infty}^{\infty} \mathrm{Corr}(h(X_0^{\kappa}), h(X_k^{\kappa})) = 1 + 2\sum_{i=1}^{\infty} \mathrm{Corr}(h(X_0^{\kappa}), h(X_i^{\kappa}))$$

is the integral of the autocorrelation time under κ if this is an integral, why is there no division by something? Check Rosenthal in Handbook of MCMC again. In other words, if there is no autocorrelation, we are left with the variance of h under f, which is very intuitive - if our kernel really generates independent samples, it does not add any further variance to the estimate of $\mathbb{V}_f(h)$. This results in another evaluation criterion: κ_1 has *smaller asymptotic variance* than κ_2 if $\tau_h^{\kappa_1} < \tau_h^{\kappa_2}$.
Finally, a Kernel is better if the resulting chain takes larger steps in the state space on average. We say κ_1 *mixes faster* than κ_2 if

$$\mathbb{E}\{(X_k^{\kappa_1} - X_{k-1}^{\kappa_1})^2\} > \mathbb{E}\{(X_k^{\kappa_2} - X_{k-1}^{\kappa_2})^2\}$$

Here we can use the estimate $\mathbb{E}\{(X_k^\kappa - X_{k-1}^\kappa)^2\} \approx n^{-1}\sum_{i=1}^n (X_i^\kappa - X_{i-1}^\kappa)^2$ where the samples are assumed to come from a chain that has converged. This expectation provides another insight into the performance of a Markov kernel. If in a Metropolis-Hastings setting a move is rejected, then $(X_i^\kappa - X_{i-1}^\kappa)^2 = 0$. But a proposal perturbing the Chains state just a tiny bit and resulting in almost every move being accepted does not help much either because $(X_i^\kappa - X_{i-1}^\kappa)^2$ will still be very small.
These criteria for evaluation Markov Kernels can be shown to be equivalent under certain conditions, resulting in the possibility to choose algorithm parameters in an optimal way if the conditions are met. If they are not met of course, different criteria might result in different decisions on how to set parameters.

1.2 Scaling

In MH, the optimal proposal distribution $q(\cdot|X')$ would be the target $f(\cdot)$, as in this case every move is accepted and we directly sample from the distribution of interest (the algorithm reduces to ordinary Monte Carlo). The case of MH used very often in practice however is that of a Random Walk using increments, i.e. a proposal is found by adding some increment $Z \sim q_{\text{inc}}(\sigma)$ to the current chain state, where σ is a scaling parameter of the density q_{inc}. The question now is which σ optimizes MCMC for generating samples that are as close to independent samples from the target f as possible. A hint at the difference that different values of σ make might be drawn from a trace of a chains state space across samples. Figure 1.1 shows the location of a chain with scalar state across samples, where σ is the variance of normally distributed increments. When picking σ too low (upper right), almost every proposed move is accepted but the increments are tiny and the chain does not explore the state space well. On the other hand, when picking σ too high (upper right), proposals are only rarely accepted. Again, the exploration properties of the chain are poor.
When picking σ approximately optimal however (lower left), random walk sampling, informally, produces a trace closest to i.i.d. samples from the true target (lower right). In this case, proposals are neither mostly rejected nor mostly accepted. This insight initiated research into finding optimal acceptance rates for random walk and other MCMC algorithms.
While the theoretical results make rather strong assumptions as we will see, the optimal rates derived often give good results in practice even if assumptions are not met Rosenthal (2011).

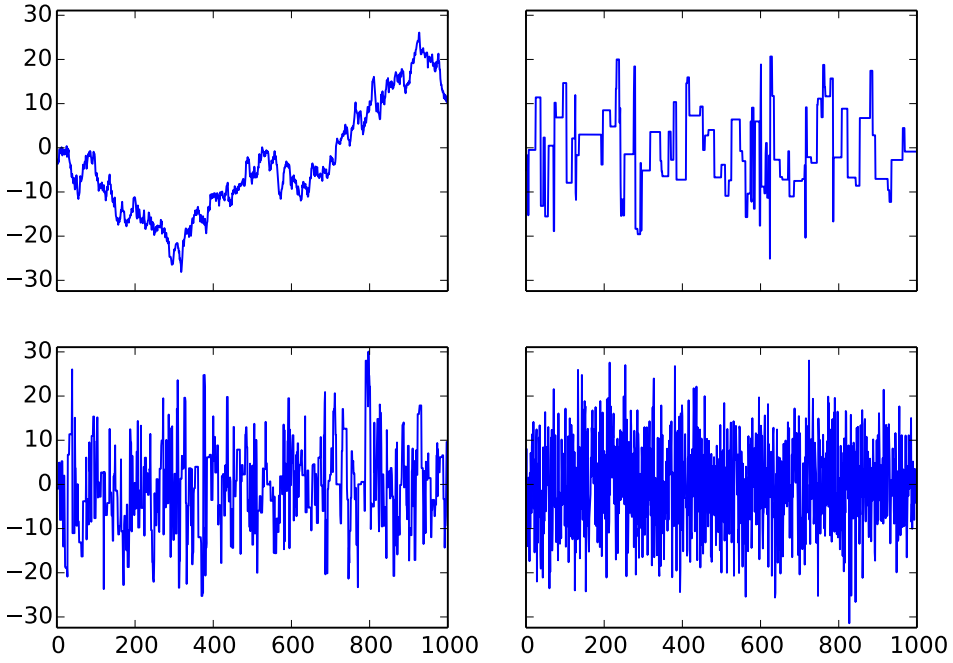

FIGURE 1.1: *Top*: Too high high (left, 0.96) and too low (right, 0.13) acceptance rates. *Bottom*: approx. optimal rate for 1D target (left, 0.47), i.i.d. target samples (right).

1.2.1 Optimal scaling in Random Walk MH

We consider d-dimensional target densities which are completely determined by the marginals, i.e. every component follows some density f_c and is i.i.d.:

$$f(x_1, x_2, \ldots, x_d) = f_c(x_1) f_c(x_2) \cdots f_c(x_d) \tag{1.1}$$

A further assumption is that f_c is smooth. Under these strong assumptions and for an increment distribution with isotropic covariance matrix $\mathcal{N}(0, \sigma^2 I_d)$, Roberts et al. (1997) proved that the optimal acceptance rate is exactly 0.234 as $d \to \infty$. The derivation of the result proceeds along the following lines. Let $\sigma = \mathcal{L}/\sqrt{d}$ where $\mathcal{L} > 0$. FOLLOWING COPIED VERBATIM FROM ROSENTHAL - Consult Roberts et al. (1997) for understanding! Then as $d \to \infty$, if time is speeded up by a factor of d, and space is shrunk by a factor of $\sqrt{d}$, then each component of the Markov chain converges to a diffusion having stationary distribution f_c, and speed function given by $\mathcal{S}(\mathcal{L}) = 2\mathcal{L}^2\Phi(-0.5\sqrt{I}\mathcal{L})$. Here Φ is the CDF of the standard normal distribution, and I is a constant given by $I = \mathbb{E}_{f_c}\left[\left(\frac{f_c'(X)}{f_c(X)}\right)^2\right]$. Then by maximizing speed, the diffusion is optimal by all of the criteria given in 1.1.1:

$$\mathcal{L}_{\text{opt}} = \arg\max_{\mathcal{L}} \mathcal{S}(\mathcal{L})$$

Nummerically, Roberts et al. (1997) found $\mathcal{L}_{\text{opt}} = 2.38/\sqrt{I}$ did Roberts et al. (1997) really state that? check!, and as they also derived the asymptotic acceptance rate as $2\Phi(-0.5\sqrt{I}\mathcal{L})$, we can just plug in $\mathcal{L}_{\text{opt}}$ to get the optimal acceptance rate A_{opt} (again, under the strong assumptions above and asymptotically for $d \to \infty$) as

$$A_{\text{opt}} = 2\Phi(-0.5\sqrt{I}\ \mathcal{L}_{\text{opt}}) = 0.234$$

. More results for more general cases, all leading to about $A_{opt} = 0.234$ with some exceptions

1.2.2 Optimal scaling in MALA

The Metropolis Adjusted Langevin Algorithm (MALA) Roberts and Tweedie (1996) is similar to random walk MH with the difference that the random increments to the state of the Markov Chain are not centered on 0. Rather, the increments at a given current state X_n are distributed as $\mathcal{N}(\frac{\sigma^2}{2}\nabla \log f(X_n), \sigma^2 I_d)$ (see A.1.1 for an overview of MALA and its truncated variant MALTA). In this case, Roberts and Rosenthal (1998) proved that under the assumption (1.1) and $\sigma = \mathcal{L}/\sqrt[6]{d}$, we get

$$A_{\text{opt}}^{\text{MALA}} = 0.574$$

The optimal scaling σ is bigger than in the random walk case check in original Roberts and Tweedie (1996).

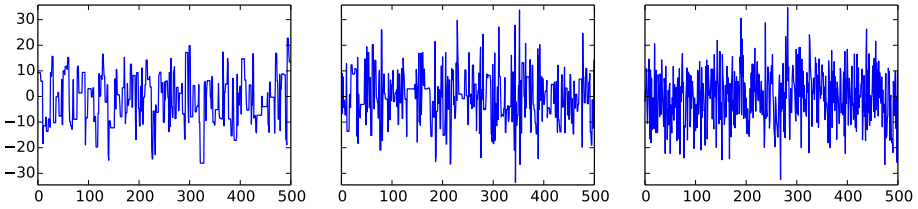

FIGURE 1.2: Traces of state space of RW (0.43 left) and MALA (0.58 middle) with chains tuned for optimal acceptance rate, i.i.d. samples (right) for comparison

1.2.3 Adaptivity

When optimal acceptance rates are known, the experimenter might just run the MCMC algorithm and tune σ to get close to A_{opt}. This of course is tedious as it requires human intervention. While this can be automated in a straight forward way, tuning a single

scale parameter might not be the only type of adaptivity that makes sense. The following section will thus look at the general case of algorithms that aim at automatically improving the sampling procedure while still ensuring ergodicity with respect to the target density.

1.3 Automatic adaptation and ergodicity of sampling algorithms

In the following, we will take a closer look at *adaptive MCMC* algorithms and under which conditions they are ergodic with respect to some target density. Traditional Markov Chain Monte Carlo algorithms assume that the transition from state X_n to state X_{n+1} is markovian (i.e. can be described by a Markov Kernel $\kappa(\cdot, X_n)$, hence *Markov Chain*). Confusingly, adaptive MCMC algorithms do not necessarily construct a Markov Chain with respect to X. For many algorithms, the process $\{X_n, \kappa_n\}_{n=0}^{\infty}$ (where κ_n is the kernel used at iteration n) is Markovian, but not even this is a necessary condition. A more appropriate name would thus be something along the lines of *adaptive ergodic Monte Carlo*.

In the following, we use a set of Markov Kernels $\{\kappa_i\}_{i\in\mathcal{I}}$, all of which have f as their unique stationary distribution. At the nth iteration, we denote by κ_n the Kernel that is used to generate a new sample. One might expect that, as every κ_i has f as stationary distribution, any mixture of the kernels in $\{\kappa_i\}_{i\in\mathcal{I}}$ has the same property. However, Rosenthal (2011) provides a simple counter example. Let f be a probability mass function that is nonzero for each element of $\mathcal{X} = \{1, 2, 3, 4\}$

x	1	2	3	4
$f(x)$	0.333	0.001	0.333	0.333

and $f(x) = 0$ for $x \notin \mathcal{X}$. Now let κ' be a Metropolis-Hastings kernel with proposal $q(\cdot|X_n) = \text{Uniform}\{X_n - 1, X_n + 1\}$ and κ'' one with proposal $q(\cdot|X_n) = \text{Uniform}\{X_n - 2, X_n - 1, X_n + 1, X_n + 2\}$. Then because of the MH correction, both κ' and κ'' are reversible and thus ergodic with respect to f. Now consider the following algorithm for adaptivity: set $\kappa_{n+1} = \kappa''$ if the nth move was accepted, $\kappa_{n+1} = \kappa'$ else. Now the adaptive algorithm can get stuck with using κ' in the state $X = 1$ for many consecutive proposals and only has a probability of $0.001/0.333$ to switch to the usage of κ''. Thus, informally, the adaptive mixture of ergodic (wrt f) kernels results in a sampling procedure that will yield state 1 way more often than the equally probable states $3, 4$.

Thus, we need to establish sufficient conditions for ensuring that adaptive MCMC algorithms preserve ergodicity. To this end, Roberts and Rosenthal (2007) provided results

for a broad class of algorithms under rather weak assumptions. If the assumptions hold, asymptotic convergence (in the number of samples) of the sample approximation to the target f holds, i.e.

$$\lim_{n\to\infty} \sup_{A\subseteq\mathcal{X}} \|P(X_n \in A) - f(A)\| = 0 \tag{1.2}$$

and furthermore a weak law of large numbers, i.e.

$$\lim_{n\to\infty} \sum_{i=1}^{n} h(X_i) = f(h) \tag{1.3}$$

for all bounded integrands $h : \mathcal{X} \to \mathbb{R}$. The assumptions are *Diminishing Adaptation*

$$\lim_{n\to\infty} \sup_{x\in\mathcal{X}} \|\kappa_{n+1}(x,\cdot) - \kappa_n(x,\cdot)\| = 0 \text{ in probability} \tag{1.4}$$

and *Bounded Convergence* (or *Containment*)

$$\{M_\epsilon(X_n, n)\}_{n=0}^{\infty} \text{ is bounded in probability} \tag{1.5}$$

where $M_\epsilon(x,n) = \inf\{\|\kappa_n^i(x,\cdot) - f(\cdot)\| \leq \epsilon : i \geq 1\}$ for some $\epsilon > 0$ is called the ϵ convergence time for kernel κ_n when starting from state x. Assumption (1.4) is the stronger of the two. In English, it says that the amount of adaptation goes to 0 for growing n. An important note here is that there can still be an infinite total amount of adaptation (as the assumption is not $\sum_{n=1}^{\infty} \sup_{x\in\mathcal{X}} \|\kappa_{n+1}(x,\cdot) - \kappa_n(x,\cdot)\| < \infty$). Also, the limit $\lim_{n\to\infty} \kappa_n$ does not need to exist, i.e. the sequence of used Markov Kernels does not have to converge to some fixed kernel. One way of ensuring (1.4) is to adapt at iteration n with some probability $p(n)$ and have $\lim_{n\to\infty} p(n) = 0$. Also, in the case where κ_n uses an average of the previous n samples, then as the new sample only contributes $\frac{1}{n}$th to the average, the amount of adaptation goes to 0 for growing n.

Assumption (1.5) on the other hand holds when $\mathcal{X} \times S_\kappa$ is finite (where S_κ is the set of used Markov Kernels), or compact in a topology where the kernels in S_κ or the Metropolis-Hastings proposal densities $\{q_n\}_{n=1}^{\infty}$ have a jointly continuous density (Roberts and Rosenthal, 2007, Rosenthal, 2011). It basically says that for any combination of current sample X and chosen kernel κ, iteratively applying κ will ensure that the sampling approximation is less than ϵ away from target f in a finite number of iterations. Thus, this condition is often easily satisfied (though counterexamples can be constructed, see Rosenthal, 2011).

In the following I will introduce some of the adaptive MCMC algorithms from the literature. The first important modern algorithm is the Adaptive Metropolis algorithm of Haario et al. (2001).

1.3.1 Adaptive Metropolis

The first representative algorithm in the class of adaptive MCMC, the Adaptive Metropolis (AM) Algorithm Haario et al. (2001), fits a Gaussian approximation with mean $\bar{X}_n$ and covariance matrix C_n to the samples acquired up until iteration n. This is achieved by the recursion formula

$$C_n = \begin{cases} C_0 & n \leq n_0 \\ \sigma_d(cov(X_0, \ldots, X_{n-1}) + \epsilon I) & n > n_0 \end{cases} \tag{1.6}$$

For an initial C_0 and some n_0 and where $\sigma_d > 0$ and $\epsilon > 0$ are tunable parameters. Adding ϵ to the diagonal ensures that C_n does not degenerate to zero. To update C_n with a new sample X_n, we can use a recursion formula

$$C_{n+1} = \frac{n-1}{n} C_n + \frac{\sigma_d}{n} \left(n\bar{X}_{n-1}\bar{X}_{n-1}^T - (n+1)\bar{X}_n\bar{X}_n^T + X_n X_n^T + \epsilon I \right)$$

The proposal distribution for a new point given the last sample is X_n then is

$$q_{\mathrm{AM}}(\cdot|X_n) = \mathcal{N}(X_n, C_n) \tag{1.7}$$

and a value X^* is accepted with probability $min(1, f(X^*)/f(X_n))$. Haario et al. (2001) suggested that σ_d will be set depending only on the dimensionality of the domain of f. In the case where f can be approximated by a gaussian, Rosenthal (2011) suggest to use $\sigma_d = 2.38^2/d$. A minor variant of the AM algorithm is given in Roberts and Rosenthal (2009). Instead of adding ϵ to the diagonal, they propose using a mixture proposal

$$q'_{\mathrm{AM}}(\cdot|X_n) = (1-\beta)\mathcal{N}(X_n, \sigma_d\ cov(X_0, \ldots, X_{n-1})) + \beta\mathcal{N}(X_n, \Sigma) \tag{1.8}$$

for a $\beta \in (0, 1)$ and some fixed covariance matrix Σ.

Since the adaptation at the nth step decreases as $\mathcal{O}(\frac{1}{n})$, the diminishing adaptation assumption (1.4) is satisfied. The bounded convergence conditon (1.5) holds if the target density f decays at least polynomially in each coordinate, thus ensuring that the entries of the fitted covariance matrix will stay finite and actually converge to some fixed value. In conclusion, AM is ergodic with respect to f. The literature (Haario et al., 2001, Roberts and Rosenthal, 2009) and my own experiments in section 1.7 suggest that AM is a very robust algorithm even for very high dimensions and outperforms non-adaptive sampling algorithms using gradient information such as Hamiltonian Monte Carlo (Neal, 2011) tuned for optimal acceptance rate. The main problem AM poses is a computational one. In high dimensions, computing a matrix inverse or Cholesky factorization of the empirical covariance matrix after one new sample is acquired is prohibitively slow. One

way to ameliorate this problem is to only adapt every k iterations (and take all k new samples into account). Another is to directly update the empirical covariance in its Cholesky form (see e.g. Rasmussen and Williams, 2006). In this case, the proposal in (1.8) is the natural choice, as it is not clear whether rank one updates are possible if some ϵ is added to the diagonal of the empirical covariance. However, updating the eigendecomposition of the empirical covariance one might still be able to use of the proposal distribution (1.7)[1].

1.3.2 Adaptive MALTA

An algorithm similar to AM making use of gradient information is the adaptive Metropolis Adjusted Langevin Algorithm with truncated drift (adapt. MALTA, Atchadé, 2006). It uses parameters $0 < \epsilon_1 < A_1, \epsilon_2 > 0$ and γ_n which depends on the iteration index. To the process (X_n) on the sample space $\mathcal{X}$ adaptive MALTA adds an adaptation process $(\mu_n, \Gamma_n, \sigma_n)$ on a space denoted Θ. To give an intuition, μ_n will be (close to) a fit of the empirical mean of (X_n), Γ_n will be close to the empirical covariance of (X_n) and the parameter σ_n is a scale parameter, similar to the parameter of the same name in AM. Let $C_n = \sigma_n^2(\Gamma_n + \epsilon_2 I)$ for some parameter ϵ_2. When the last sample is X_n, adaptive MALTA uses a proposal distribution

$$q(\cdot|X_n) = \mathcal{N}(X_n + \frac{C_n}{2} D(n, X_n), C_n)$$

and a proposed move to X^* is accepted with the standard MH probability

$$p_n^{\text{acc}}(X^*|X_n) = \min\left(1, \frac{f(X^*)q(X_n|X^*)}{f(X_n)q(X^*|X_n)}\right)$$

The subscript n hints at the fact that the acceptance probability depends on the parameter values at time n through q. $D(\cdot,\cdot)$ is a drift function and Atchadé (2006) suggests

$$D(k, x) = \frac{\delta}{\max(\delta, |\nabla \log f(x)|)} \nabla \log f(x)|$$

The scale parameter is constrained to the interval $[\epsilon_1, A_1]$ by the projection

$$p_1(\sigma) = \begin{cases} \epsilon_1 & \sigma < \epsilon_1 \\ A_1 & \sigma > A_1 \\ \sigma & \text{else} \end{cases}$$

[1] Personal communication with Yee Whye Teh

The covariance Γ_n is constrained to lie in the convex compact cone defined by the projection

$$p_2(\Gamma) = \begin{cases} \Gamma & |\Gamma| \leq A_1 \\ \frac{A_1}{|\Gamma|}\Gamma & \text{else} \end{cases}$$

where $|\cdot|$ is the Frobenius norm. Finally, μ_n is constrained to the ball $B(0, A_1)$ in $\mathbb{R}^d$ (where d is the dimensionality of the problem) by the projection

$$p_3(\mu) = \begin{cases} A_1 & |\mu| \leq A_1 \\ \mu & \text{else} \end{cases}$$

Suppose we have generated a new sample X_{n+1}. Now the parameters are adapted using the following equations.

$$\begin{aligned} \mu_{n+1} &= p_3(\mu_n + \gamma_n(X_{n+1} - \mu_n)) \\ \Gamma_{n+1} &= p_2(\Gamma_n + \gamma_n((X_{n+1} - \mu_n)(X_{n+1} - \mu_n)^T - \Gamma_n) \\ \sigma_{n+1} &= p_1(\sigma_n + \gamma_n(p_n^{\text{acc}}(X^*|X_n) - \tau)) \end{aligned}$$

Where τ is the optimal acceptance rate that the algorithm should try to attain and γ_n is a factor decreasing the amount of adaptation for each new sample. Concretely, $\lim_{n\to\infty} \gamma_n = 0$ and Atchadé (2006) used $\gamma_n = 10/n$. The values for the other parameters where set as $\epsilon_1 = 10^{-7}, \epsilon_2 = 10^{-6}, A_1 = 10^7$.

As is evident even from this concise exposition to adaptive MALTA, it is a complex algorithm with many parameters (even though according to Atchadé (2006) the algorithm is not very sensitive to ϵ_1, ϵ_2 and A_1, so they might be considered fixed). Of course it has the advantage of working correctly even if the expectation or variance of f does not exist. However, even when tuned to optimal acceptance rates, I found the performance of adaptive MALTA to often be worse than that of AM.

1.4 Introduction: Gradient Importance Sampling

Monte Carlo methods have been developed into one of the mainstream inference methods of Bayesian Statistics and Machine Learning over the last thirty years Robert and Casella (2004). They can be used to approximate expextations with respect to posterior distributions of a Bayesian Model given data. The most widely used Monte Carlo Method for this purpose today is Markov Chain Monte Carlo (MCMC). In this approach, a Markov Chain is constructed which is ergodic with respect to the given posterior distribution. In parallel, a scheme for sampling from a sequence of target distributions,

called Sequential Monte Carlo (SMC), has been developed Doucet et al. (2001). SMC has traditionally been used for inference in time series models and for online tracking applications. However, there has been a considerable amount of research on using SMC for inference in static models as well Chopin (2002), Del Moral et al. (2006), Schäfer and Chopin (2013). In this paper, I will develop a powerful variant of SMC for static models making use of gradient information, dubbed Gradient Importance Sampling.

The paper will proceed as follows. In subsection A.1.1 I will give a short overview of the a simple well-known MCMC algorithm making use of gradient information, the Metropolis Adjusted Langevin Truncated Algorithm. In subsection A.2.1 an exposition to Importance Sampling and SMC is given. Gradient IS and its variants are introduced in 1.5. Related work, especially in adaptive MCMC and Importance Sampling, is discussed in Section 1.6. Gradient IS is evaluated and compared against previous algorithms in Section 1.7. The last section concludes.

1.5 Gradient IS

Gradient IS (GRIS) is a variant of Sequential Monte Carlo Doucet et al. (2001), or, when targeting the same density at each iteration, of its special case Population Monte Carlo Cappé et al. (2004). GRIS accomplishes adaptivity by fitting a covariance matrix C_t to samples from the target density that have been gathered at time t. The proposal distribution for a new sample given an old sample X' is then given by

$$q_t(\cdot|X') = N(\cdot|X' + D(t, \nabla \log f(X')), C_t)$$

where D is a drift function. We used

$$D(t, \nabla \log f(X')) = (\delta/t^{1.5}) \nabla \log f(X')$$

thus introducing a parameter $\delta \geq 0$ (and usually $\delta \leq 1$) which ought to be tuned for each target density. To fit C_t we use the parametrization How strongly does this overlap with the section on Adaptive Metropolis in the adaptive MCMC part?

$$C_t = \begin{cases} C_0 & t \leq t_0 \\ s_d(cov(X_0, \ldots, X_{t-1}) + \epsilon I) & t > t_0 \end{cases}$$

For an initial C_0 and some t_0 and where $s_d > 0$ and $\epsilon > 0$ are tunable parameters. To update C_t with a new sample X_t, we can a recursion formula

$$C_{t+1} = \frac{t-1}{t} C_t + \frac{s_d}{t} \left(t \bar{X}_{t-1} \bar{X}_{t-1}^T - (t+1) \bar{X}_t \bar{X}_t^T + X_t X_t^T + \epsilon I \right)$$

Algorithm 1 Gradient IS algorithm

Input: unnormalized density f, gradient $\nabla \log f$, population size p, p intial samples S, sample size m
Output: list S of m samples
while $len(S) < m + p$ **do**
  Initialize $P = List()$
  Initialize $W = List()$
  for $i = 1$ **to** p **do**
    (a) sample X' uniformly from last p samples in S
    (b) generate $X \sim q_t(\cdot|X')$, append it to P
      append weight $f(X)/q_t(X)$ to W
  end for
  resample p values from P with repl. according
    to weights and append samples to S
  compute C_{t+1}
end while
remove first p samples from S

Where $\bar{X}_t$ is the running average at iteration t (with an obvious recursion formula) Haario et al. (2001). Directly updating either the Cholesky or Eigendecomposition of C_t results in significant computational savings, especially when the updates are done frequently. Both decompositions can be used to draw samples from and evaluate the density of the respective multivariate normal, both of which are prerequisites of the GRIS algorithm. A complementary method to tradeoff speed and adaptivity is to collect more than one new sample before updating. GRIS then proceeds as given in Algorithm 1.

1.5.1 Gradient IS with a sequence of target distributions

Instead of using the correct target distribution f at every IS iteration like in PMC, it is also possible to construct a sequence of intermediate target distributions $(g_t)_{t=1}^T$ where $g_T = f$. In the existing SMC literature addressing static targets, only the samples acquired when targeting the actual distribution of interest are then used for estimating the integral H Del Moral et al. (2006), Schäfer and Chopin (2013). This approach is very robust when compared to MCMC algorithms, as it explores the probability space better Chopin (2002), Schäfer and Chopin (2013).
One possibility for constructing a sequence of distributions is the geometric bridge defined by $g_t \propto g_0^{1-\rho_t} f^{\rho_t}$ for some initial distribution g_0 and where $(\rho_t)_{t=1}^T$ is an increasing sequence satisfying $\rho_T = 1$. Another is to use a mixture $g_t \propto (1 - \rho_t)g_0 + \rho_t f$. When f is a Bayesian posterior, one can also add more data with increasing t, e.g. by defining the intermediate distributions as $g_t(X) = f(X|d_1, \ldots, d_{\lfloor \rho_t D \rfloor})$ where D is the number of data points, resulting in an online inference algorithm Chopin (2002).

However, when using a distribution sequence that computes the posterior density f using the full dataset (such as the geometric bridge or the mixture sequence), one can reuse the intermediate samples when targeting g_t for posterior estimation using a simple trick. As the value of f is computed anyway for the harmonic bridge and the mixture sequence, we can just use the weight $f(X)/q_t(X)$ for posterior estimation while employing $g_t(X)/q_t(X)$ to inform future proposal distributions. This way the evaluation of f (which is typically costly) is put to good use for improving the posterior estimate. To the best of my knowledge, this recycling of samples in SMC for static targets has not been reported in the literature.

1.6 Related work

1.6.1 Adaptive Monte Carlo using ergodic stochastic processes

rewrite in light of the earlier section on adaptive MCMC Adaptive MCMC algorithms use samples acquired so far in order to tune parameters of the sampling algorithm to adapt it to the target. which is another difference from GRIS, which uses importance weighting instead. The AM algorithm is the first adaptive MCMC algorithm and is provably ergodic wrt the target distribution f.

Informally, adapting the Markov Kernels used in MCMC will result in an algorithm that is ergodic with respect to target f in the general case as long as

1. all Markov kernels used are ergodic wrt target f (have f as their stationary distribution)
2. adaptation diminishes, e.g. adaptation probability p_t for iteration t satisfies $p_t \to 0$ as $t \to \infty$.

(Theorem 1 in Roberts and Rosenthal (2007)). We can still have $\sum_{t=1}^{N} p_t \to \infty$ as $N \to \infty$, i.e. infinite overall adaptation (as in Sejdinovic et al. (2014)). It is important to note that the diminishing adaptation is sufficient but not necessary. Other schemes might still be ergodic, one example being the AM algorithm, where adaptation is not diminished.

1.6.2 Adaptive Importance Sampling Schemes

Adaptive Importance Sampling schemes in related to SMC have mostly been trying to fit a closed-form approximation to the posterior for usage as a proposal distribution in the next iteration. In particular, previous work used optimality criteria such as minimization

of KL-divergence between proposal distribution and posterior to asses the quality of the proposal distribution. After collecting new samples, D-Kernel approximations Douc et al. (2007) or Mixture distributions (such as Mixtures of Gaussians or Student-T mixtures) Cappé et al. (2008) are updated to fit the posterior better. In another line of research, better ways of using the produced samples have been sought. A particularly interesting approach is Adaptive Multiple Importance Sampling Cornuet et al. (2012), Marin et al. (2012). Here, samples from previous proposal distributions are reweighted after the sampling is finished in order to reflect the overall proposal distribution used across iterations. This is achieved by simply assuming that each weighted sample is produced by a mixture over the proposal distributions used, where the mixture weight is proportional to the number of samples actually drawn from a particular proposal. This reweighting has been shown to be consistent and considerably improve consecutive estimates Marin et al. (2012).

1.7 Evaluation

For an experimental evaluation of the adaptive Gradient IS algorithm, I targeted three synthetic distributions (i.e. not posteriors based on some dataset) and one posterior of a Bayesian Logistic regression model. The synthetic distributions have the advantage that the expected value H of the target is known in closed form, for the Logistic Regression dataset ground truth was estimated in a separate Monte Carlo experiment. I started the simulation at H to avoid having to handle burn-in for MCMC and ran 20 Monte Carlo simulations with 3000 target or gradient evaluations each. If the target and its gradient are evaluated at the same point, this is counted as a single evaluation. This is justified by the fact that most of the time either likelihood or gradient calculation dominate complexity and computing the other value uses only little additional time (since factors common to both likelihood and gradient can be cached). I compared to the AM algorithm, adaptive T-MALA, and Hamiltonian Monte Carlo Neal (2011) MCMC algorithms, where I tuned the parameters to give optimal acceptance rates as reported in the literature. GRIS was tuned for low estimates of Monte Carlo variance, an information that is available in typical black-box situations (i.e. where nothing is known about the target f except the information produced by the sampling algorithm). Is used the variant of GRIS that did not use a sequence of target distributions as it gave good results and was easier to implement.

1.7.1 Maximum squared errors and Effective Sample Size

Estimating Effective Sample Size (ESS) is easy in traditional Importance Sampling and does not require establishing ground truth. Given importance weights of N collected samples $(w_i)_{i=1}^N$ and their normalized version $\widehat{w}_i = w_i / \sum_{i=1}^N w_i$ we can compute ESS as

$$\mathrm{ESS}_N^{\mathrm{IS}} = \frac{1}{\sum_{i=1}^N (\widehat{w}_i)^2}$$

If all the weights are the same (i.e. we have a sample coming from our target distribution), we have $\mathrm{ESS}_N^{\mathrm{IS}} = N$, whereas if only one of the weights is non-zero, $\mathrm{ESS}_N^{\mathrm{IS}} = 1$. A necessary condition for this estimate to be valid however is that drawn importance samples are iid. For MCMC a similar estimate (under the same name) is available after establishing ground truth for an integral of interest. An exposition to this is given in Hoffman and Gelman (2014) whom we will follow. Given a function h which we want to integrate wrt the target density f, and samples $X_1, \ldots, X_N$ from a Markov Chain targeting f, that follow the distribution q_{MC} we can use the estimate

$$\mathrm{ESS}_N^{\mathrm{MC}} = \frac{N}{1 + 2\sum_{i=1}^{N-1}(1 - \frac{i}{N})\rho_i^h}$$

where ρ_i^h is the autocorrelation of h under q_{MC} at lag i. If the autocorrelation is exactly 0 at each lag (i.e. all of the samples are independent), then again $\mathrm{ESS}_N^{\mathrm{MC}} = N$, as desired. An estimate of autocorrelation is given by

$$\widehat{\rho}_i^h = \frac{1}{\mathbb{V}[h(X)](N-i)} \sum_{j=i+1}^{N} (h(X_j) - \mathbb{E}[h(X)])(h(X_{j-i}) - \mathbb{E}[h(X)])$$

In this autocorrellation estimate we make use of the ground truth values $\mathbb{E}[h(X)]$ and $\mathbb{V}[h(X)]$ which can only be estimated by a long run of some Monte Carlo method when f is a black block (for example when it is given only proportionally as a Bayesian posterior). Now as the estimate of autocorrelation is noisy for large lags, Hoffman and Gelman (2014) suggests to truncate the autocorrelations at the smallest cutoff c with $\rho_c^h < 0.05$, yielding the estimator

$$\widehat{\mathrm{ESS}}_N^{\mathrm{MC}} = \frac{N}{1 + 2\sum_{i=1}^{c}(1 - \frac{i}{N})\widehat{\rho}_i^h}$$

Now as $\widehat{\mathrm{ESS}}^{\mathrm{MC}}$ is a univariate measure, Girolami and Calderhead (2011), Hoffman and Gelman (2014) suggest the following approach: calculate ESS for each dimension separately and take the minimum across dimensions. Because usually in a bayesian setting

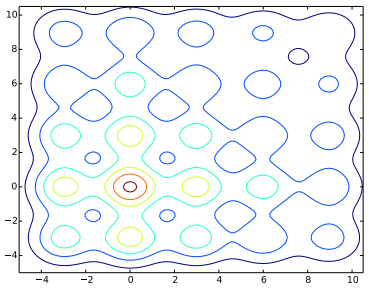

FIGURE 1.3: Gaussian mixture target: Contour plot

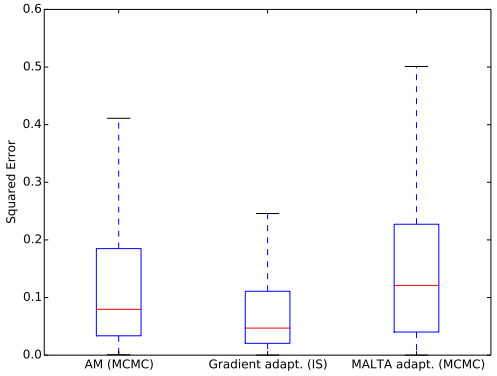

FIGURE 1.4: Gaussian Mixture target: SE performance. Algorithms not shown are widely off scale.

one is interested in estimation of variance, the suggested procedure is to do this for an estimate of both expected value and variance of f and take the minimum of those as the final ESS estimate.

Ironically, an ESS estimate does not exist for Sequential Monte Carlo when targeting the same density across iterations (the situation is different in a dynamic model with multiple targets). For one, it is not possible to use the estimate ESS^{IS} because SMC induces dependency between samples. Also, ESS^{MC} is not usable because the dependency structure in SMC, and thus the computation of autocorrelation, is much more complicated than in MCMC, as a sample at one iteration of the algorithm can spawn several samples in the next iteration.

However, using ground truth estimates $\mathbb{V}[h(X)]$ and $\mathbb{E}[h(X)]$ as ESS^{MC} does, it is possible to reproduce the worst case properties suggested by Girolami and Calderhead (2011), Hoffman and Gelman (2014). To this end, one measures maximum squared error (MaxSE) by computing, for each Monte Carlo run, squared errors with respect to $\mathbb{V}[h(X)]$ and $\mathbb{E}[h(X)]$ and taking the maximum across those and across dimensions.

1.7.2 Gaussian Grid

The first target was a mixture of 25 2D Gaussians with equidistant means and mixture weights depending on distance from origin, a contour plot for which is given in Figure 1.3. A box plot for the squared error across different numbers of target evaluations is given in Figure 1.4, where the box ranges from 1st to 3rd quartile and whiskers extend to $1.5 * \text{IQR}$ (interquartile range). Hamiltonian Monte Carlo (HMC) was not plotted here, because the algorithm performed much worse than the other three candidates and proper scaling was impossible.

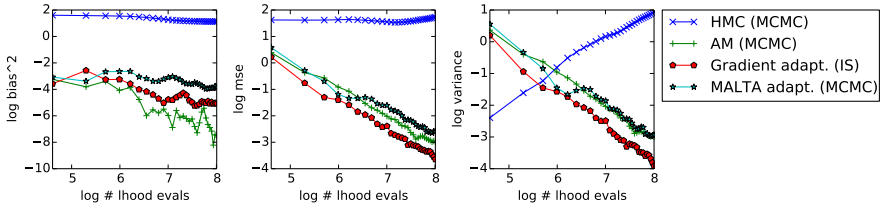

FIGURE 1.5: Gaussian mixture target: Squared bias, MSE and variance as a function of number of target evaluations

For this target, the performance was very close for the adaptive algorithms. Adaptive Gradient IS exhibits smallest MSE and variance (see Figure 1.5), but the AM algorithm is a close second. As multimodal targets are traditionally problematic for gradient informed sampling algorithms, it is interesting to see that adapting to the posterior covariance structure of the target can help mitigate problems in this case. The particularly weak performance of HMC possibly stems from the algorithm getting stuck in different modes for different runs, explaining in particular the high variance (Figure 1.5).

1.7.3 Banana

The 2D Banana shaped unnormalized measure was given by $f(x) = \exp(-\frac{1}{2s}x_1^2 - \frac{1}{2}(x_2 - b(x_1^2 - s))^2$. The measure is determined by parameters b and v which where set to 100 and 0.03, respectively (see Figure 1.6 for a contour plot). For unimodal targets, gradient based samplers are traditionally strong, though that advantage might be irrelevant when we adapt a scale matrix for a simple random walk algorithm. As is evident from Figures 1.7 and 1.8, the simple AM algorithm actually is competitive with adaptive T-MALA for this target. However, adaptive Gradient IS again shows gains here when compared to the other samplers. This is particularly encouraging since T-MALA and Gradient IS are using the exact same drift function and T-MALA adapts more parameters than Gradient IS does. If the remaining parameters of Gradient IS can be adapted automatically, further improvements might be possible.

1.7.4 Mixture of T distributions

The third target considered was a mixture of three 10D multivariate t-distributions with different means, 10 degrees of freedom and varying mixture weights. The scale matrices where drawn from an Inverse Wishart distribution with identity scale matrix and 10 degrees of freedom, yielding mixture components with strong correlations among dimensions. A contour of the marginal density of the last two coordinates is given in Figure 1.9. This case was considered because t-distributions have heavier tails than

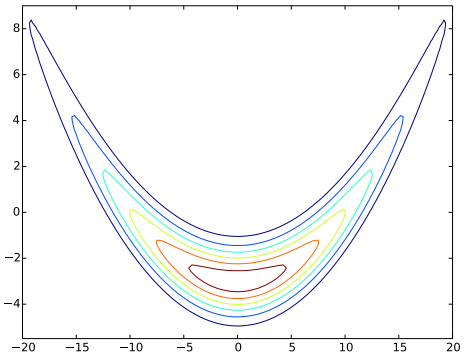

FIGURE 1.6: Banana target: Contour plot

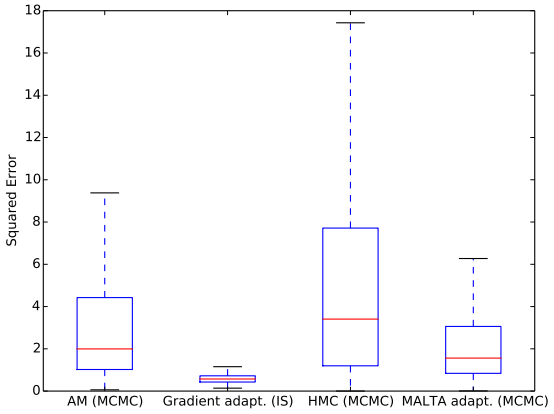

FIGURE 1.7: Banana target: SE performance

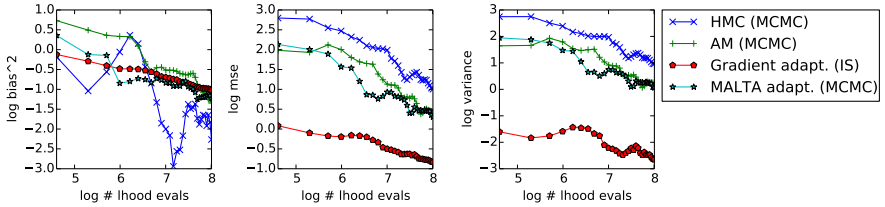

FIGURE 1.8: Banana target: Squared bias, MSE and variance as a function of number of target evaluations

Gaussians. As a standard rule of thumb, IS proposal distributions should have heavier tails than the target to ensure finite variance of the estimate Robert and Casella (2004). With a mixture of t-distributions, I wanted to experimentally check for any problems stemming from the Gaussian proposal used in all algorithms. GRIS is better when averaging squared error across dimensions (Figure 1.10). Also, when when comparing maximum log squared errors GRIS is clearly an improvement over previous algorithms (Figure 1.11).

1.7.5 German Credit Dataset: Logistic regression

The German Credit dataset was used with the Logistic Regression model developed in Hoffman and Gelman (2014). This model exhibited a 25D posterior distribution which allowed for a Laplace approximation. To find ground truth, I collected ordinary Importance Samples from a mixture between the Laplace Approximation and the sample approximation with slightly increased scale, a method known as defensive Importance Sampling (Owen and Zhou, 2000). The Effective Sample Size of this approximation was over $66,000$.

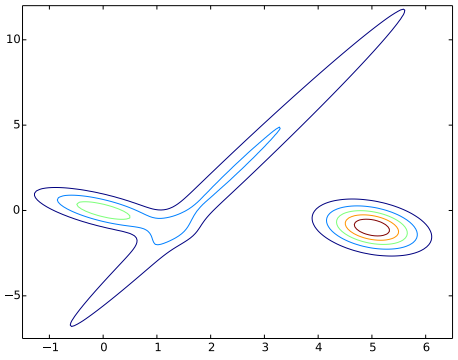

FIGURE 1.9: 10 D t-Mixture target: Contour plot of the marginal density of the last two dimensions

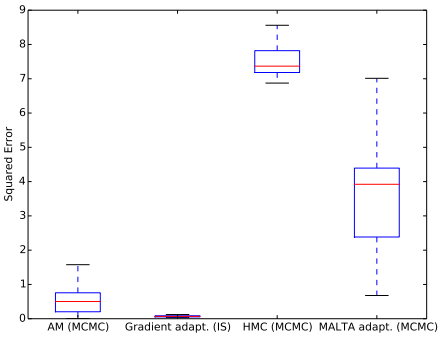

FIGURE 1.10: t-Mixture target: SE of mean estimates, averaged across dimensions

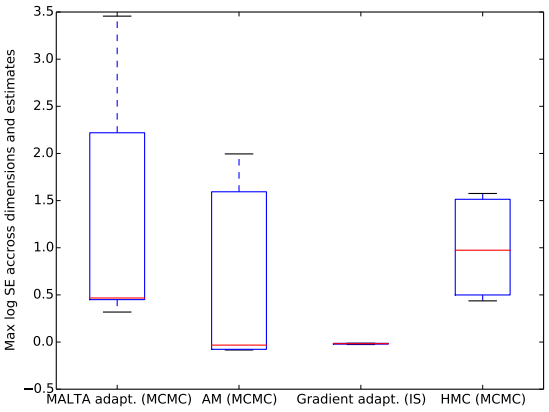

FIGURE 1.11: t-Mixture target: Squared bias, MSE and variance as a function of number of target evaluations

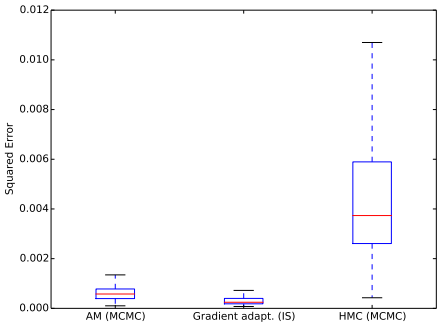

FIGURE 1.12: Logistic regression: Squared errors of mean estimate averaged across dimensions. Algorithms not shown are widely off scale.

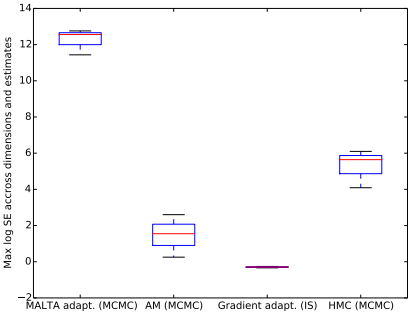

FIGURE 1.13: Logistic regression: Maximum log SE across estimates of posterior variance and mean and across dimensions.

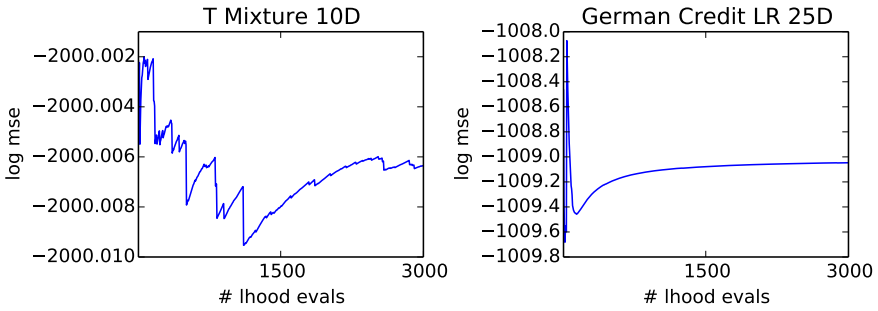

FIGURE 1.14: Evidence estimates

1.7.6 Evidence Estimation

Evidence estimates using GRIS quickly stabilized. I assesed MSE of evidence estimates for the t-Mixture target and the Logistic Regression model (Figure 1.14). The log evidences where -1000 and -504, respectively.

1.8 Conclusions

In this paper, I presented Gradient IS, a variant of Sequential Monte Carlo. The algorithm uses gradient information and a covariance matrix adapted to collected posterior samples to construct a multivariate normal proposal distribution for SMC with static target distributions. GRIS was shown to give very good performance in posterior sampling and provide stable estimates for the normalizing constant of the target distribution (also known as model evidence).

Appendix A

Fundamental Monte Carlo Algorithms

A.1 Fundamental MCMC algorithms

A.1.1 Langevin Monte Carlo

Metropolis Adjusted Langevin Truncated Algorithm (MALTA) Roberts and Tweedie (1996) and Hamiltonian Monte Carlo (HMC) Neal (2011) are two well known sampling algorithms that make use of gradient information. In the following, we will denote the target density as f. Often times, this will we the posterior of a Bayesian model which can be evaluated only proportionally by multiplying the prior and likelihood at a given a point.

HMC is probably better known in the Machine Learning community, but it is notoriously complex and its description is beyond the scope of this pape. For a thorough introduction see e.g. Bishop (2007), Neal (2011). The special case of HMC however, MALTA, is closely related to the algorithm proposed in this paper and a concise introduction will be given. MALTA is a variant of the Metropolis-Hastings MCMC algorithm where, given the current state of the Markov Chain X', a proposal for a new state X is sampled from the multivariate normal density

$$q(\cdot|X') = N(\cdot|X' + D(\nabla \log f(X')), C)$$

where D is a drift function. For consistency reasons, the MALTA variant used in the evaluation section will use $D(\nabla \log f(X')) = \delta \nabla \log f(X')$ for $0 \leq \delta \leq 1$. The covariance matrix C is fixed by the user prior to running the algorithm. The proposed new state

X is then accepted with the usual Metropolis-Hasting acceptance probability

$$min\left(1, \frac{f(X')q(X|X')}{f(X)q(X'|X)}\right)$$

and recorded as a sample. If the proposed state is rejected, the chain remains at state X' (which is recorded as a sample again). The Markov Chain is ergodic with respect to f, i.e. the samples produced are approximately from f, which is guaranteed by using the Metropolis-Hastings correction. The samples can be used to estimate the expectation H of some function of interest h with respect to the target density f using the law of large numbers as

$$H = \int h(x)f(x)\mathrm{d}x \approx 1/N \sum_{i=1}^{N} h(X_i)$$

where X_i ranges over the samples and N is the number of samples.

A.2 Fundamental SMC algorithms

A.2.1 Importance Sampling and SMC

Importance Sampling takes a different approach. Instead of trying to sample approximately from f, it samples from some proposal density q instead. Rather than correcting for the change of distribution using Metropolis-Hastings, the Importance Sampling estimator simply weighs each sample X by the so called importance weight $w(X) = f(X)/q(X)$. In the case where f is not normalized, which is the usual case when estimating a Bayesian posterior, the self-normalized Importance Sampling estimator for H given by

$$H = \int h(x)w(x)q(x)\mathrm{d}x \approx \frac{1}{\sum_{i=1}^{N} w(X_i)} \sum_{i=1}^{N} w(X_i)h(X_i)$$

Sequential Monte Carlo (SMC) Doucet et al. (2001) builds on Importance Sampling and was originally devised to sample from a sequence of target distributions. For ease of exposition, I will first consider the case where the same target distribution is used at each iteration, a special case known as Population Monte Carlo (PMC). From this, an extension to sequence of targets is straight forward and given in Section 1.5.1 for the case of static models (i.e. not time series). In Population Monte Carlo, we first gather a set of p samples (also called *particles* in SMC) $X_1, \ldots, X_p$ from proposal densities $q_1, \ldots, q_p$ which are assigned weights $w(X_i) = f(X_i)/q_i(X_i)$. Instead of using these weighted samples directly with the Importance Sampling estimator to evaluate the integral of interest,

we resample $X_1, \ldots, X_p$ with replacement according to their respective weights, adding the resulting set to a set of unweighted samples S. This is called Importance Resampling and produces a sample set that is approximately coming from the posterior (Rubin, 1987). Several methods exist for this step, the easiest being multinomial resampling. See Douc and Cappé (2005) for a review including some theoretical results. Previous samples can now be used to construct proposal distributions for the next iteration. In the simplest case this could be centering a proposal distribution on a previous sample. The procedure is iterated until S is deemed large enough. The integral of interest can now simply be computed by

$$H = \int h(x)f(x)\mathrm{d}x \approx 1/|S| \sum_{X \in S}^{N} h(X)$$

Moreover, the marginal likelihood Z of the data (also called evidence of the model or normalizing constant of f) can be approximated by the formula

$$Z \approx 1/N_w \sum_{i=1}^{N_w} w_i$$

where w_i are the weights that have been gathered from the stage before resampling and N_w is the total number of weights.

A major argument for Gradient IS is the ability to approximate the marginal likelihood *and* the target distribution as good as or better than previous gradient-informed and/or adaptive sampling algorithms while being extremely simple to implement. For example, this opens the possibility to routinely compute Bayes factors (and thus do Bayesian Model selection) as a by-product of very efficient posterior sampling instead of using special inference techniques geared towards only computing Z.

Bibliography

Atchadé, Y. F. (2006). An adaptive version for the metropolis adjusted Langevin algorithm with a truncated drift. *Methodology and Computing in Applied Probability*, 8(March):235–254.

Bishop, C. M. (2007). *Pattern recognition and machine learning*. Springer.

Cappé, O., Douc, R., Guillin, A., Marin, J.-M., and Robert, C. P. (2008). Adaptive importance sampling in general mixture classes. *Statistics and Computing*, 18(4):447–459.

Cappé, O., Guillin, a., Marin, J. M., and Robert, C. P. (2004). Population Monte Carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–552.

Cornuet, J. M., Marin, J. M., Mira, A., and Robert, C. P. (2012). Adaptive Multiple Importance Sampling. *Scandinavian Journal of Statistics*, 39(4):798–812.

Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436.

Douc, R. and Cappé, O. (2005). Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pages 64–69.

Douc, R., Guillin, a., Marin, J. M., and Robert, C. P. (2007). Convergence of adaptive mixtures of importance sampling schemes. *Annals of Statistics*, 35(1):420–448.

Doucet, A., Freitas, N. D., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Springer.

Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 73(2):123–214.

Haario, H., Saksman, E., and Tamminen, J. (2001). An Adaptive Metropolis Algorithm. *Bernoulli*, 7(2):223–242.

Hoffman, M. and Gelman, A. (2014). The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15:1351–1381.

Marin, J.-m., Pudlo, P., and Sedki, M. (2012). Consistency of the Adaptive Multiple Importance Sampling.

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC.

Owen, A. B. and Zhou, Y. (2000). Safe and effective importance sampling. *Journal of the American Statistical Association*, 95(449):135–143.

Rasmussen, C. E. and Williams, C. K. I. (2006). Gaussian processes for machine learning.

Robert, C. P. and Casella, G. (2004). *Monte Carlo statistical methods*. Springer, second edi edition.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *Annals of Applied Probability*, 7(1):110–120.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 60(1):255–268.

Roberts, G. O. and Rosenthal, J. S. (2007). Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *Journal of Applied Probability*, 44(2):458–475.

Roberts, G. O. and Rosenthal, J. S. (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2).

Roberts, G. O. and Tweedie, R. L. (1996). Exponential Convergence of Langevin Distributions and Their Discrete Approximations. *Bernoulli*, 2(4):pp. 341–363.

Rosenthal, J. S. (2011). Optimal Proposal Distributions and Adaptive MCMC. In *Handbook of Markov Chain Monte Carlo*, chapter 4, pages 93–112. Chapman & Hall.

Rubin, D. B. (1987). *Multiple Imputation for Nonresponse in Surveys*. Wiley Publishing.

Schäfer, C. and Chopin, N. (2013). Sequential Monte Carlo on large binary sampling spaces. *Statistics and Computing*, 23(2):163–184.

Sejdinovic, D., Strathmann, H., Garcia, M. L., Andrieu, C., and Gretton, A. (2014). Kernel Adaptive Metropolis-Hastings. *arXiv*, 32.

Tsybakov, A. B. (2008). *Introduction to Nonparametric Estimation*. Springer.