BAYESIAN STATISTICS 7, pp. 000–000
J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid,
D. Heckerman, A. F. M. Smith and M. West (Eds.)

Gaussian Processes to Speed up Hybrid Monte Carlo for Expensive Bayesian Integrals

CARL EDWARD RASMUSSEN
Gatsby Computational Neuroscience Unit, UCL, UK
edward@gatsby.ucl.ac.uk

SUMMARY

Hybrid Monte Carlo (HMC) is often the method of choice for computing Bayesian integrals that are not analytically tractable. However the success of this method may require a very large number of evaluations of the (un-normalized) posterior and its partial derivatives. In situations where the posterior is computationally costly to evaluate, this may lead to unacceptable computational load for HMC. I propose to use a Gaussian Process model of the (log of the) posterior for most of the computations required by HMC. Within this scheme only occasional evaluation of the actual posterior is required to guarantee that the samples generated have exactly the desired distribution, even if the GP model is somewhat inaccurate. The method is demonstrated on a 10 dimensional problem, where 200 evaluations suffice for the generation of 100 roughly independent points from the posterior. Thus, the proposed scheme allows Bayesian treatment of models with posteriors that are computationally demanding, such as models involving computer simulation.

Keywords: GAUSSIAN PROCESS; MARKOV CHAIN MONTE CARLO; HYBRID MONTE CARLO; NON-PARAMETRIC MODELS; DERIVATIVE PROCESS; QUADRATURE; DESIGN.

1. INTRODUCTION

Evaluation of integrals is at the heart of Bayesian inference: integrating wrt. the posterior to make predictions, and integrating the likelihood wrt. the prior to obtain the marginal likelihood for model comparison. The later is often more difficult than the former, since the prior is typically fairly broad and the likelihood very peaked. In this paper we consider the former, and will refer to the posterior wrt. which we wish to integrate as the *target* density.

For many interesting statistical models, direct analytical integration is not possible and Markov Chain Monte Carlo (MCMC) methods are often used. Simple forms of MCMC may require a very large number of evaluations of the target density in order to produce reliable results. One of the main causes is that the target density often exhibits strong dependencies between variables, such that the regions of high target density form a narrow extended manifold; situations where this manifold is broken up into several distinct regions (multi-modality) is even more difficult and will not be specifically dealt with here.

Strong dependencies between variables in the target density are probably typical to most Bayesian inference problems, and extended directions within the manifold corresponds to possible families of explanations for the observed data. Integration wrt. these uncertainties is one of the main virtues of the Bayesian treatment, contrasting with paradigms based on point estimates.

The strong dependencies between variables make it difficult to explore the target density while remaining within regions of high probability. In situations where the posterior itself is computationally demanding to evaluate, this makes simple MCMC schemes impractical. Computationally demanding posteriors arise in several contexts: 1) hierarchical models where the lower levels of the hierarchy have been integrated out to reduce the number of parameters typically results in complex posteriors over the remaining parameters, and 2) models that use computer simulation as an integral part of their specification, such as is typical for many geophysical or meteorological applications. In these situations evaluation of the target density at a single point can in itself be computationally demanding. In such circumstances it is obvious that we need to use the information gained from each evaluation as effectively as possible.

The new method proposed in this paper is an elaboration of the Hybrid Monte Carlo (HMC) method of Duane *et al.* (1987), using a Gaussian Process (GP) model to guide the search and avoid too many references to the target density function. Gaussian process models have previously been suggested for evaluating integrals by O'Hagan (1991) under the name of Bayes-Hermite quadrature, see also the excellent O'Hagan (1992). In this methodology an "importance" density is introduced, and the Gaussian process is used to model the integrand times the ratio of the posterior and importance density. For analytical tractability the importance density must be Normal, although this requirement has been relaxed by Kennedy (1998). As we will see, the current approach attempts to tackle exactly the same problems, but in a very different way.

The Hybrid Monte Carlo (HMC) method of Duane *et al.* (1987) is a fairly generally applicable approach to sampling more efficiently from highly correlated densities by suppressing (inefficient) random walk behaviour. Even though HMC is often much faster than sampling schemes relying on random walks, it may still require a large number of evaluations of the target density. In this paper I propose an extension to HMC which allows a further reduction in the number of target evaluations. The key idea is to use an approximate model of the target density for most of the computations and only occasionally require the evaluation of the true target; this is done in such a way, that the generated samples will have exactly the desired distribution.

The remainder of the paper is structured in the following way: In section 2 the motivation for seeking to suppress random walks is discussed, and the Hybrid Monte Carlo approach is outlined. In section 3 some results for Gaussian process models are reviewed including conditioning on observed values of function derivatives. The new algorithm is given in section 4 and its performance is demonstrated on a couple of simple problems. Conclusion and further discussion are given in section 5.

2. RANDOM WALK METHODS AND HYBRID MONTE CARLO

In highly correlated distributions, sampling methods relying on random walks are inefficient in exploring the target density. Using steps of size s, it will take an expected L^2/s^2 steps to move some distance L using random walk, since the trajectory will often fold back on itself. This is illustrated in a simple bivariate Normal case in figure 1, for the very commonly used methods of Gibbs sampling and Metropolis sampling.

Since the typical step sizes are confined to be of the order of the narrowest direction

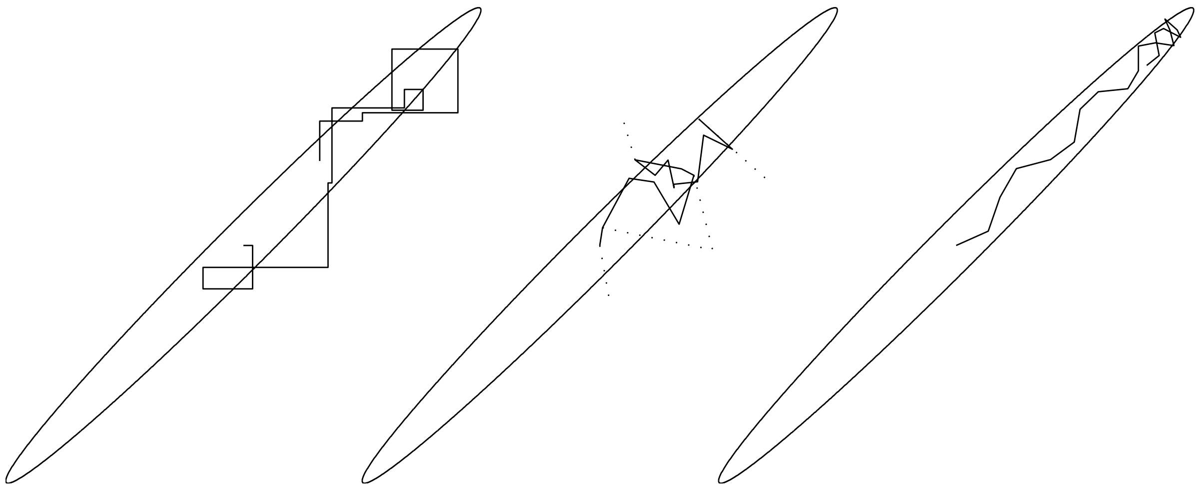

Figure 1. *Examples of sampling from a correlated bivariate Normal density; the one standard deviation contour of the density is drawn. Sample trajectories for three different sampling algorithms are shown: left: 20 (half) iterations of Gibbs sampling, middle: 20 Metropolis proposals (rejected proposals are indicated with dotted lines), right: 20 sets of leapfrog iterations for Hybrid Monte Carlo. The three trajectories represent comparable computational burdens, although the HMC requires evaluation of derivatives in addition to density values.*

in order to get good acceptance probabilities[1], the quadratic exploration time of the random walk will lead to poor performance as the strength of the correlation increases. The situation is similar for the Gibbs sampling algorithm. The most straight forward solution to this problem would be to make a variable transformation to eliminate the strong dependencies, but for complex distributions encountered in Bayesian inference this can be very difficult. Other approaches include the ordered overrelaxation scheme of Neal (1998) and the Hybrid Monte Carlo algorithm, which can be seen as a more elaborate way of generating Metropolis proposals.

The HMC algorithm is reviewed in a statistical setting by Neal (1993). In HMC a fictitious dynamical system is introduced in which the variables θ are interpreted as positions, and are augmented by a set of "momentum" variables, ϕ. Metropolis proposals are made by simulating the dynamical system through fictitious time. Because of momentum in the physical system, the random walks are suppressed. The energies of the physical system is related to the probability via the usual duality:

$$E_{\text{pot}} = -\log p(\theta), \quad E_{\text{kin}} = \frac{1}{2}\sum_i \phi_i^2, \quad H = E_{\text{pot}} + E_{\text{kin}}, \quad p(\theta,\phi) = \exp(-H). \quad (1)$$

Thus the potential energy E_{pot} is related to the target distribution $p(\theta)$, which we wish to sample from. The kinetic energy is related to the square of the momentum. Note that the joint probability of θ and ϕ factors into independent parts; thus we can obtain samples from $p(\theta)$ by sampling from the joint and ignoring the momenta.

The evolution of the system through fictitious time at constant total energy H is governed by Hamilton's equations:

[1]strictly speaking, this holds only for $D > 2$ dimensions; in $D = 2$ dimensions the acceptance rate falls only linearly with the proposal width, so the optimal width is the longer of the two directions. So figure 1 is slightly misleading in this respect.

$$\frac{d\theta_i}{dt} = \frac{\partial H}{\partial \phi_i} = p_i, \qquad \frac{d\phi_i}{dt} = -\frac{\partial H}{\partial \theta_i} = -\frac{\partial E_{\text{pot}}}{\partial \theta_i}. \tag{2}$$

Unfortunately, for most target densities equation (2) are coupled nonlinear differential equations (involving the partial derivative of the log posterior wrt. the parameters) which we cannot solve in closed form. Instead we approximate the solution by iterating first order steps:

$$\theta_i(t+\varepsilon) := \theta_i(t) - \varepsilon \frac{\partial E_{\text{pot}}(\theta(t))}{\partial \theta_i}, \qquad \phi_i(t+\varepsilon) := \phi_i(t) + \varepsilon \theta_i(t+\varepsilon) \tag{3}$$

In practice we use the slightly modified "leapfrog" scheme, and take first a half step for the position, then iterate full steps for momentum and position, and finally another half step for the position; this is done to make the approximation reversible (which is needed for detailed balance for the Metropolis step).

The full algorithm for Hybrid Monte Carlo becomes iteration of the following steps: 1) Gibbs sample for the momentum variables, whose density is the standard normal (equation. (1)), 2) simulate the physical system for L leapfrog steps to generate a proposal, and 3) accept or reject the proposed state in a Metropolis step using the joint densities $p(\theta, \phi)$. This procedure generates samples from the joint density $p(\theta, \phi)$, and we get the desired samples from the marginal $p(\theta)$ by ignoring the momenta. In figure 1 right the success of HMC in suppressing random walks is shown. Note that the so-called Langevin Monte Carlo method is equivalent to HMC with $L = 1$, but this of course eliminates the desirable suppression of random walks.

If we had solved Hamilton's equations exactly, then the Metropolis step would always accept, since the total energy is conserved by the dynamics. However, since we only followed the dynamics approximately, the energy may change and this may lead to rejections. Notice however, that the samples obtained have exactly the desired distribution, although the dynamics were approximate.

2.1. *Using Hybrid Monte Carlo*

When using the HMC procedure in practice, one needs to address two main issues to ensure that the algorithm is performing well. Firstly, we need to chose a step size ε for the leapfrog iterations. If using a single step size as in equation (3) this is fairly easily done by monitoring the resulting acceptance rate. However, it is quite possible that the appropriate step sizes vary widely between different coordinates of θ, which may be difficult to assess, either by estimating curvature of the posterior or by experimentation. Secondly, once good step sizes have been established, we need to determine the appropriate length L of the simulation. This can usually be done with some experimentation, *e.g.*, by monitoring the auto covariance function for parameters and increasing L until roughly independent samples are obtained. Too short trajectories will cause a failure to suppress random walks and too long trajectories will be wasteful of computation.

Once reasonable values for ε and L have been determined, the HMC algorithm can be run to produce the desired samples. In figure 1 the bivariate Normal has one direction which is 10 times a long as the perpendicular direction. Using a simple Metropolis scheme we would expect the step sizes to be confined by the narrowest direction, and thus expect roughly 10^2 steps to be necessary to generate roughly independent samples. In this example, we see that HMC is able to traverse the distribution using 20

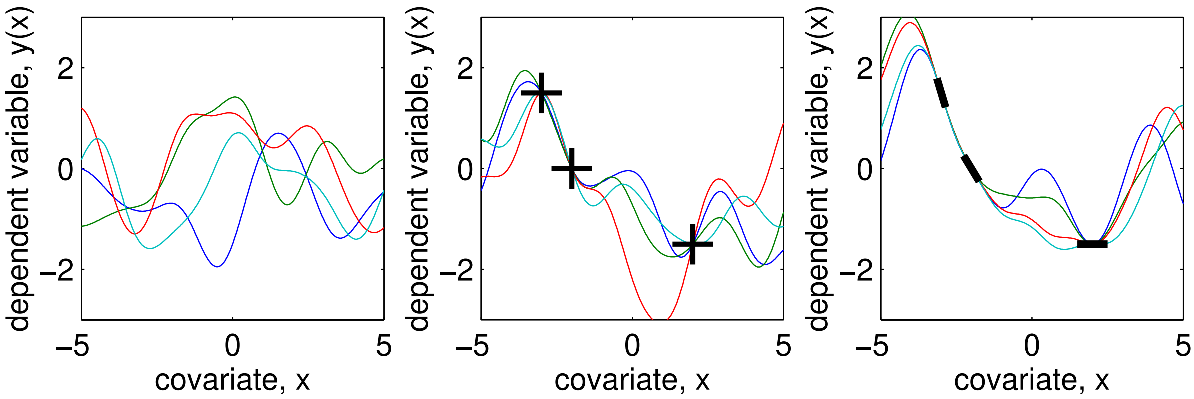

Figure 2. *Examples of random functions drawn from a Gaussian process with Gaussian covariance function (with $w_0 = 1$ and $w_1 = 1$): left: four functions drawn at random, middle: four functions drawn at random conditional on 3 observations indicated with + signs, right: same observations as the middle plot, but now the four random functions are also conditional on the derivative at those points, indicated by small tangent segments. It is seen how more information about the function confines the distribution over functions, and how the uncertainty about the functions varies with x.*

evaluations (of the posterior and its partial derivatives), thus providing improved performance. up. However, we may still need a large number of evaluations to achieve adequate samples. The key idea in this paper is that we can use a model of the posterior instead of the posterior itself for the dynamical simulation, and thereby save greatly on the number of evaluations necessary. It turns out that Gaussian processes are a very convenient tool to model the posterior.

3. GAUSSIAN PROCESS REGRESSION

We can use a noise free[2] Gaussian process to specify a distribution over smooth functions. Consider a set of n pairs of covariate x and dependent variables y: $\{x^{(c)}, y^{(c)} \,|\, c = 1, \ldots, n\}$. We specify that the dependent variables have a joint Normal distribution, with a covariance matrix that depends on the covariates. For example:

$$p(\boldsymbol{y} \,|\, \boldsymbol{x}) \sim \mathrm{N}(\boldsymbol{y} \,|\, 0, \Sigma), \quad \Sigma_{pq} = C(x^{(p)}, x^{(q)}) = w_0 \exp\big(-\frac{1}{2}\sum_{d=1}^{D}(x_d^{(p)} - x_d^{(q)})^2/w_d^2\big), \tag{4}$$

where x_d is the d'th covariate. This defines a zero mean Gaussian process with a stationary (Gaussian) covariance function, see O'Hagan (1978) and Neal (1998) for further background on GP models. The covariance is parameterised by hyperparameters: length scales w_d and overall scale w_0. Obviously there are many other possible choices for covariance functions, but here we will stick with equation (4).

Having observed the values of some y variables, we're interested in predicting the y^* corresponding to a new value of the covariate, x^* which is obtained by conditioning the joint on the observed values, yielding a Normal:

$$p(y^* \,|\, x^*, \boldsymbol{y}, \boldsymbol{x}) = \mathrm{N}(y^* \,|\, \mu, \sigma^2), \qquad \begin{aligned} \mu &= C(x^*, \boldsymbol{x})^T C(\boldsymbol{x}, \boldsymbol{x})^{-1}\boldsymbol{y} \\ \sigma^2 &= C(x^*, x^*) - C(x^*, \boldsymbol{x})^T C(\boldsymbol{x}, \boldsymbol{x})^{-1} C(x^*, \boldsymbol{x}) \end{aligned} \tag{5}$$

[2]In practice I add a tiny bit of noise (by adding a small multiple of the identity to the covariance matrix) to improve numerical conditioning.

where bold face signifies a vector of values, such that *e.g.*, $C(x^*, \boldsymbol{x})$ is the vector of covariances between y^* and each of the n values of the (observed) dependent variable.

A priori we may not have any clear preferences for particular values of the hyperparameters. In principle, we should therefore set a prior over these and integrate out the hyperparameters as in Neal (1998). For simplicity I will simply optimize the likelihood wrt. the hyperparameters. The log likelihood L and its partial derivatives, Williams and Rasmussen (1996), are given by:

$$\begin{aligned} L &= \log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) = -\tfrac{1}{2}\log|\Sigma| - \tfrac{1}{2}\boldsymbol{y}^T\Sigma^{-1}\boldsymbol{y} - \tfrac{n}{2}\log(2\pi), \\ \tfrac{\partial L}{\partial w_i} &= -\tfrac{1}{2}\mathrm{tr}(\Sigma^{-1}\tfrac{\partial \Sigma}{\partial w_i}) + \tfrac{1}{2}\boldsymbol{y}^T\Sigma^{-1}\tfrac{\partial \Sigma}{\partial w_i}\Sigma^{-1}\boldsymbol{y}. \end{aligned} \tag{6}$$

The likelihood can conveniently be optimized with a standard numerical tool such as a conjugate gradient method.

3.1. *Gaussian Process Derivatives*

Since differentiation is a linear operator, the derivative of a Gaussian process is again a Gaussian process. For example, the mean of the derivative is equal to the derivative of the mean for predictions:

$$E[\frac{\partial y^*}{\partial x^*_d}] = \frac{\partial E[y^*]}{\partial x^*_d} = \frac{\partial \mu}{\partial x^*_d} = \frac{\partial C(x^*, \boldsymbol{x})}{\partial x^*_d} C(\boldsymbol{x}, \boldsymbol{x})^{-1}\boldsymbol{y}, \tag{7}$$

and we can similarly find the variance of the distribution of the derivative. Equally importantly for the present purposes, we wish to observe values for the partial derivatives and condition on these to make inference about the target function. The covariance between function values and derivatives, and between derivatives are given by:

$$\mathrm{Cov}\Big(y^{(p)}, \frac{\partial y^{(q)}}{\partial x^{(q)}_d}\Big) = \frac{\partial C(x^{(p)}, x^{(q)})}{\partial x^{(q)}_d}, \qquad \mathrm{Cov}\Big(\frac{\partial y^{(p)}}{\partial x^{(p)}_d}, \frac{\partial y^{(q)}}{\partial x^{(q)}_f}\Big) = \frac{\partial^2 C(x^{(p)}, x^{(q)})}{\partial x^{(p)}_d \partial x^{(q)}_f}, \tag{8}$$

so we can simply write out an extended covariance matrix (between all values and partial derivatives) and extend the observation vector in equation (5) to include observed derivatives, to make predictions as illustrated in figure 2 right.

4. THE GPHMC ALGORITHM

The key idea in the proposed algorithm is to combine the HMC algorithm with a GP model of the potential energy, or equivalently the negative log target density. The zero mean GP is used to model the observations with the empirical mean subtracted. The GPHMC algorithm is initialized by evaluating the target function at a number, say D of random locations (perhaps sampled from the prior). The algorithm now proceeds in two phases:

Exploratory phase. The goal in this phase is to gather information about the target density. We use HMC with long trajectories (I use $L = 1000$ leapfrog iterations) using the energy approximated by the Gaussian process model, by plugging equation (7) into equation (3). In the exploratory phase we set the energy for HMC to be the mean GP prediction minus the one standard error uncertainty (one could allow other trade offs between log density and uncertainty, the one chosen here is simple and seems to have the desired effect), *i.e.*, $E_{\text{pot}} = \mu - \sigma$ from equation (5) and in addition it will require the partial derivative of the uncertainties, which are also easily computed. This will cause the dynamical simulation to seek out regions of high target density and regions in which the GP model is uncertain about the shape of the target density. The value of the uncertainty is monitored along the trajectory, and if ever the uncertainty reaches a certain threshold (I use a value of $\sigma = 3$), further simulation seems unlikely to yield meaningful information, so the simulation is stopped, and the target density (and its partial derivatives) are evaluated. In this way the algorithm seeks out good placements of design points, a feature which is crucial if a low number of evaluations are to suffice. The next trajectory is then simulated, either starting from the new point, or from the beginning of the previous trajectory (as to avoid restarting at points of very low probability), based on a Metropolis criterion.

Between each trajectory the hyperparameters of the covariance function are updated by maximizing the likelihood (using equation (6)), including the newly gathered information. I allow a small number (20) of function evaluations by a conjugate gradient maximization algorithm (since we don't expect the hyperparameters to change drastically between each data acquisition). After each trajectory, we also update the momenta in accordance with the HMC procedure. Once a reasonable number of data points have been collected, the initial (random) data points are gradually removed, since they will tend to have very low target density values, which renders them unimportant for inference, and they may lead to numerical instabilities for the GP model.

Sampling phase. The goal of this phase is to generate the desired samples from the target distribution using the GP model generated in the previous phase. Now, we set the energy of the dynamical system equal to the mean prediction from the GP, and always continue for the full trajectory length ($L = 1000$). At the end of the trajectory a Metropolis step is done, requiring a single evaluation of the target density. This latter Metropolis step will ensure that the samples generated will have exactly the desired distribution, although the GP model of the target density may still be imperfect. In fact, the rejection rate will give an indication of how well the GP model has captured the target density. If we get too many rejections, we could try to extend the exploratory phase.

It should be noted that in the new algorithm, the setting of the step sizes ε and trajectory lengths L is much less critical than for the original HMC algorithm. The step size is set low enough that long trajectories get accepted fairly early on in the exploratory phase. The more difficult L parameter, which must be adjusted carefully for optimal performance in HMC, can now simply be set to some quite large value (such as $L = 1000$), since the simulation no longer refers to the actual target density, but used the GP model, which is assumed much cheaper to evaluate. This property may make the new scheme easier to use in practice.

4.1. *A Demonstration*

In this section I will demonstrate the viability of the proposed approach, by applying it to two problems with known density. The first toy example is a $D = 2$ dimensional target density proportional to $\exp(-8(x_1^2 + x_2^2 - 1)^2)$, see figure 3. The GPHMC method is run for 100 iterations in each of the two phases. In the sampling phase there were no rejections, confirming that the model of the target density is quite accurate, in fact 100 exploratory iterations seems to be overkill in this example. The generated samples seem

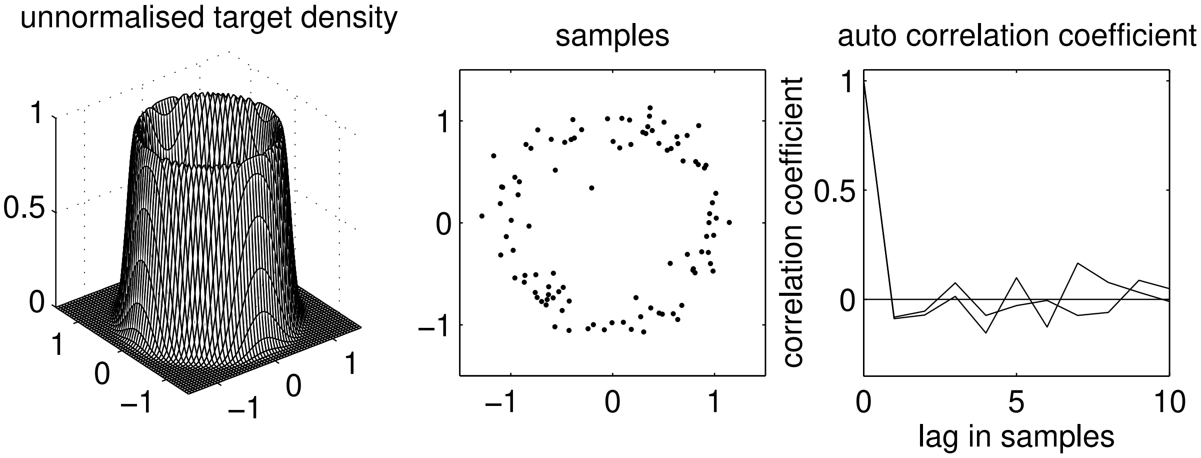

Figure 3. *The $D = 2$ dimensional example. Left: the (un-normalized) target density, middle: the 100 generated samples, right: the auto covariance for each of the two dimensions as a function of the iteration lag number.*

to be essentially independent judging by the auto covariance function. In conclusion we have generated 100 independent samples from the desired distribution, evaluating the target function 200 times and its partial derivatives 100 times.

The second example is more challenging in $D = 10$ dimensions. For simplicity of exposition I have chosen a Normal target density, whose covariance matrix has an eigenvector $(1, 1, \ldots, 1)^T$ with a corresponding eigenvalue of 1.0, and all other eigenvalues are 0.01. Thus, the target density has 9 confined directions and a single direction which is 10 times wider, and all variables are (positively) correlated. Although there is a simple parametric form for the density, this does not make it any easier for the procedure, since no use is made of this fact. However, using a standard density will make it easier for us to determine whether the samples adequately span the desired space. Notice also, that the parameterisation of the covariance function does not allow the GP to discover directly that there is one long direction and 9 confined ones (by setting the w hyperparameters) since I have chosen the directions to be diagonal rather than axis aligned. Further, the GP is using a stationary covariance function to model the log of the target density (which is of course a quadratic), which is not tailored to the situation. Of course in any practical situation we should try to use a covariance function which reflects our beliefs about the actual target function at hand; however, we typically won't have strong knowledge of this, and also the covariance functions that are convenient to handle in practice is limited. All in all, I have attempted to create an example which has a realistic amount of difficulty, while being easy to evaluate.

The results of the simulation is shown in figure 4. The exploration phase was run for 100 iterations, the initial part of the run locating regions of high target density. During the subsequent sampling phase 5 of the 100 proposed samples were rejected, indicating that the GP approximation to the log target density is not perfect, but still adequate for a high acceptance rate. Also in this example, we generated 100 roughly independent samples from the target density using only 200 evaluations of the density, and 100 evaluations of its partial derivatives. The empirical means are within the tolerance expected based on 100 independent samples, and the eigenvalues of the empirical covariance of the 100 samples also seem typical (when informally compared to draws from the corresponding Wishart distribution). So there are reasons to believe

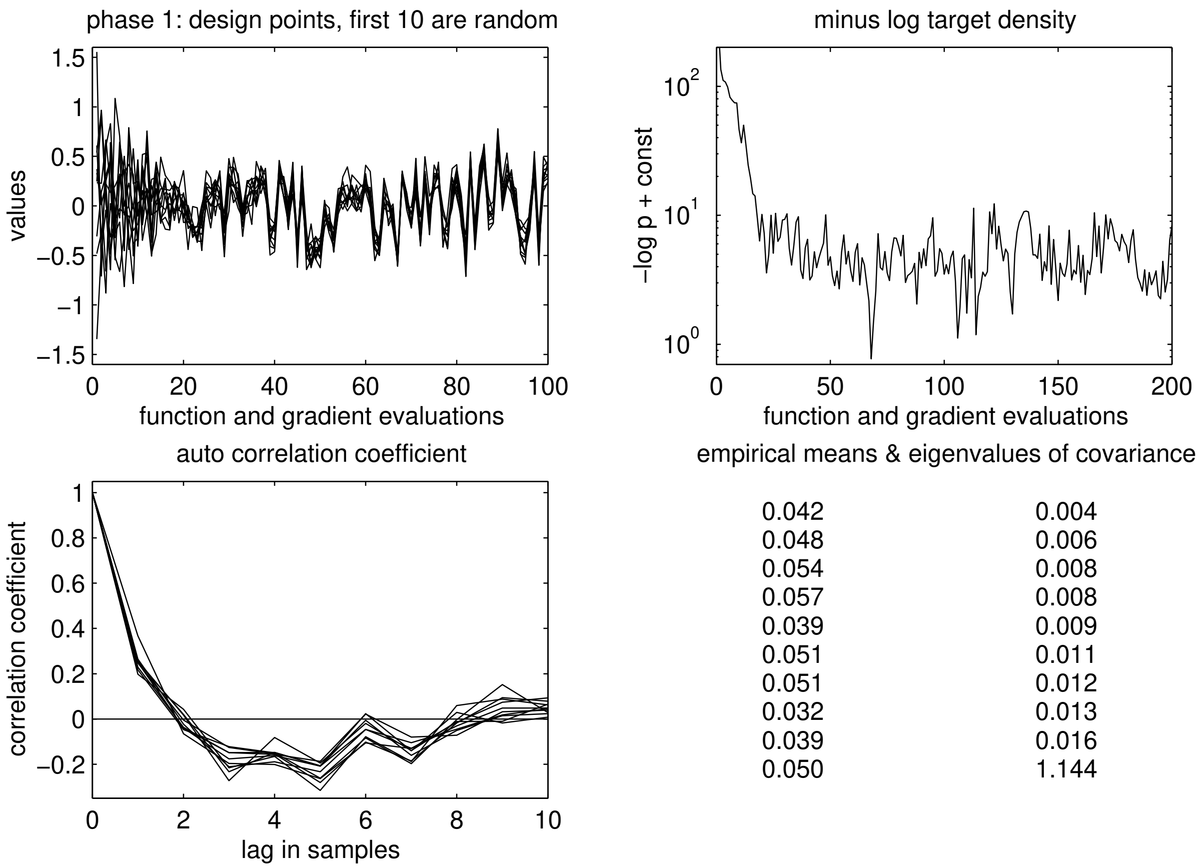

Figure 4. *The $D = 10$ dimensional Normal example. Top left: the design points generated by the procedure (the first 10 are random* N*(0,1)); all 10 dimensions are plotted on top of each other; it is seen how the correlation in the target density is gradually recovered, top right: negative log density gradually falls as regions of high probability are found, bottom left: the 10 auto correlation function for each dimension suggests that the samples generated are close to independent, bottom right: empirical mean and eigenvalues of the empirical covariance matrix.*

that the procedure produced the correct results in this example.

We can compare this performance to simpler MCMC schemes. The generation of 100 roughly independent samples using simple Metropolis would take on the order of 10000 evaluations (100 samples times the squared ratio of widest to most confined direction). The HMC procedure with a step size resulting in around 20% rejections, needs roughly $L = 20$ (found through experimentation) resulting in 100 roughly independent samples requiring about 2000 evaluations. In conclusion, this example shows that GPHMC is about an order of magnitude faster than HMC, which itself is about an order of magnitude faster than simple Metropolis. The speed up gained for any task is of course dependent on the magnitude of the dependencies under the target density.

5. CONCLUSIONS, DISCUSSION AND FUTURE DIRECTIONS

A new version of the HMC algorithm has been proposed which is suitable for doing integrals over distributions that are computationally difficult to evaluate. The way the algorithm has been presented here we also require evaluation of the partial derivatives, which may in some cases be comparatively easy to compute; for example, this is the case for GP models themselves equation (6), where evaluation of the likelihood is cubic in the number of constraints, but once this has been done each of the partial derivatives take only quadratic time. If derivatives are not easy to compute, it is of course possible to run the algorithm using only function values.

A simple example in $D = 10$ dimensions has shown the viability of the new method. It is natural to wonder how large problems the method can be used for. This depends on how many constraints are necessary to pin down the target density. Even if you know the distribution to be Normal, the number of required constraints still scales quadratically in the dimension. Since we are doing a non-parametric fit, and because of the sequential nature of the selection of design points, we probably cannot get close to the globally optimal design, and would expect to need at least some small factor times D^2 constraints. In the $D = 10$ example, at the end of the exploration phase we have retained 75 (out of the 100, since we gradually remove the initial points which have low density), giving $75 \times 11 = 825$ constraints. This means that the covariance matrix which needs inverting in the GP model has size 825×825, which takes on the order of one minute on a modern laptop computer. Since the computation times scales cubically in the number of constraints, the overall computation times is expected to be something like D^6 for direct implementations, which unfortunately precludes sizes much larger than $D = 15$ or so.

Interesting future work should compare this new proposed method to the related Bayes-Hermite quadrature of O'Hagan (1991). In particular it seem worth investigating whether more appropriate covariance functions for log densities or ratios of densities could be handled. The current use of a stationary covariance with a fixed mean seems inappropriate, but perhaps it is not a bad model in regions of high density, which is the regions that are important to model well. Further understanding and development of these ideas should enable Bayesian treatment of computationally complex models, especially those involving computer simulations.

REFERENCES

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo, *Physics letters B* **55**, 2774–2777.

Kennedy, M. (1998) Bayesian quadrature with non-normal approximating functions, *Statist. Computing* **8**, 365-375.

Neal, R. M. (1993). Probabilistic Inference Using Markov Chain Monte Carlo Methods. *Tech. Rep.*, CRG-TR-93-1, Department of Computer Science, University of Toronto.

Neal, R. M. (1998). Regression and classification using Gaussian process priors. *Bayesian Statistics 6* (J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith, eds.). Oxford: University Press, 475–501, (with discussion).

Neal, R. M. (1998) Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation. *Learning in Graphical Models* (M. I. Jordan, eds), Dordrecht: Kluwer Academic Publishers, 205–225.

O'Hagan, A. (1978). Curve Fitting and Optimal Design for Prediction. *J. Roy. Statist. Soc. B* **40**, 1–42 (with discussion).

O'Hagan, A. (1991). Bayes-Hermite Quadrature. *J. Statist. Planning and Inference* **29**, 245–260.

O'Hagan, A. (1992). Some Bayesian Numerical Analysis. *Bayesian Statistics 4* (J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith, eds.). Oxford: University Press, 345–365, (with discussion).

Williams, C. K. I., and Rasmussen, C. E. (1996). Gaussian Processes for Regression, *Advances in Neural Information Processing Systems 8* (D. S. Touretzky, M. C. Mozer, M. E. Hasselmo, eds), MIT Press, 514–520.