Hamiltonian Markov Chain Monte Carlo Using Second Derivatives

Stephen P. Smith, February 2021

Abstract. Hamiltonian Markov Chain Monte Carlo is one of the established methods to conduct a Bayesian simulation. This method uses evaluations of the probability density and its gradient at particular variables. This present paper describes how to incorporate information from second derivatives that relate to a direction set, and describes how to modify the simulation accordingly.

1. Introduction

Hamiltonian Markov Chain Monte Carlo (MCMC) is a Bayesian simulation method that barrows a track mapping technique for dynamical systems that employs the Hamiltonian (see Neal 2011). This method uses the gradient of the log of a posterior distribution that is made to correspond to a potential energy function, and maps a path that is modulated by a momentum vector that corresponds to a kinetic energy function. That is, a Bayesian simulation is made to follow Newtonian dynamics, and at the end of the track is a potential simulant that is either accepted or rejected according to a protocol for the Metropolis algorithm. The connection between Bayesian statistics and dynamical systems looks abstractly stilted, upon first appraisal. However, it's a well justified simulation method to sample from a posterior distribution.

The Hamiltonian simulation only needs the log of the posterior distribution (ignoring the constant term), and its gradient, that are evaluated at particular variables. Second derivatives are not needed, but second derivatives can be evaluated by forward and backward differentiation (Griewank 2000). If n is the number of parameters, then calculating all second derivatives requires an order n, or o(n), times the computing time needed to evaluate the gradient. Given that some of the second derivatives can be evaluated cheaply in a direction set[1], a method that incorporates some second derivatives in a Hamiltonian simulation is justified and is the subject of the present paper.

Section 2 describes some definitions that follow Neal (2011). Section 3 describes the differential equations for the Hamiltonian dynamic that is modified to include second derivatives for a direction set. Section 4 describes how to solve the differential equations, and concluding remarks are made in Section 5.

Understand that known implementations of Hamiltonian simulation already employ analytical solutions like those in Section 4, or use the log-Gaussian distribution to

[1] If r is the dimension of the direction set, then the computing time is o(r) times the work needed to evaluate the gradient.

approximate the potential energy function where by implication second derivatives are used already (see Neal 2011, Section 5.5.1.2). However, what is minimally original in the present paper is the use of second derivatives representing a direction set that changes dynamically within the confines of Hamiltonian MCMC. The present paper establishes the feasibility of a Hamiltonian simulation that uses some second derivatives that are calculated for a general probability density while using the efficient method of backward differentiation.

2. Definitions

The momentum vector, denoted by **p**, is an n×1 vector of momentums. The position vector, denoted by **q**, is also an n×1 vector but contains positions that are matched one-to-one with the elements of **p**. The matching of **p** to **q** is enforced by the Hamiltonian dynamic, which will be described shortly.

The kinetic energy function is denoted by $K(\mathbf{p})=(2m)^{-1}\times\mathbf{p}^{T}\mathbf{I}\,\mathbf{p}$, where m is the mass constant. There are more general versions of K(**p**) that can be used, such as $K(\mathbf{p})=½\mathbf{p}^{T}\mathbf{M}^{-1}\mathbf{p}$ where **M** is a symmetric and positive definite mass matrix, but the present discussion is limited to the simple version.

The potential energy function is denoted by U(**q**) and represents the negative log of a nominated probability density function, where the elements of the position vector are taken as the random variables.

What will be taken as known, and computable, about U(**q**) will be first and some second derivatives that are evaluating at $\mathbf{q}=\mathbf{q}_k$, representing a step k. So the gradient of the potential energy function, denoted by ∇U(**q**), is not known as a free function of **q**, its only known as a constant vector at step k but a constant that can change for different steps. Likewise, the selected second derivatives are only available as calculated quantities that are evaluated at step k, and other steps. It's the function ∇U(**q**) that feeds into the Hamiltonian system that determines the dynamic, but these impacts must be approximated using known quantities rather than the actual gradient function, and approximated as part of the chosen method of discretization.

3. The Hamiltonian Dynamic

The time driven Hamiltonian dynamic is determined by the following system of differential equations (see Neal, 2011, for more details), given initial conditions.

$$\frac{\partial \mathbf{q}}{\partial t} = \nabla K(\mathbf{p}) = \frac{1}{m}\mathbf{p}$$

$$\frac{\partial \mathbf{p}}{\partial t} = -\nabla U(\mathbf{q})$$

Given values of **p** and **q** at step k, denoting these by $\mathbf{p}_k$ and $\mathbf{q}_k$, the challenge remains to use the above dynamic to approximate $\mathbf{p}_{k+1}$ and $\mathbf{q}_{k+1}$ for some time corresponding to $t_{k+1}=t_k+\epsilon$. With ∇U(**q**) treated as a constant at step k, the approximation might take the standard path and move to discretization by using Euler's method with leapfrog adjustments. However, the method considered in the present discussion will investigate approximating ∇U(**q**) by a linear function representing first and second derivatives, leading to a possible analytical solution to the Hamiltonian dynamic. With analytical solution in hand, $\mathbf{p}_{k+1}$ and $\mathbf{q}_{k+1}$ are computed for step k+1 thus finishing step k and moving on to repeat the process.

The linear approximation of ∇U(**q**) might involve Taylor's expansion, while using all first and second derivatives, and this option is presented below.

$$\nabla U(\mathbf{q}) \approx \nabla U(\mathbf{q_k}) + \mathbf{H}_k\left[\mathbf{q}-\mathbf{q_k}\right]$$

$$where$$

$$\mathbf{H}_k = \left\{ \left. \frac{\partial^2 U(\mathbf{q})}{\partial q_i \partial q_j} \right|_{\mathbf{q}=\mathbf{q}_k} \right\}$$

This approximation should work well in a small neighborhood around $\mathbf{q}_k$. However, evaluating all n(n+1)/2 second derivative might prove expensive, and so a less demanding linear approximation can be sough that involves a direction set **W**, representing an n×r matrix with r<n. The columns of **W** define a basis that span over a vector space that represent the permitted directions where information on second derivatives is to be employed. Restricting an approximate Hessian matrix[2] to be rank r forces the linear approximation to become the following.

$$\nabla U(\mathbf{q}) \approx \nabla U(\mathbf{q}_k) + \mathbf{H}_k\mathbf{W}\left[\mathbf{W}^T\mathbf{H}_k\mathbf{W}\right]^{-1}\mathbf{W}^T\mathbf{H}_k\left[\mathbf{q}-\mathbf{q}_k\right]$$

Both the gradient ∇U($\mathbf{q}_k$) and $\mathbf{H}_k\mathbf{W}$ can be evaluated efficiently by backward differentiation. The evaluation of $\mathbf{H}_k\mathbf{W}$ only involves r passes through the recursion list for backward differentiation; simply initialize the q-array (at step-1)[3] with the columns of **W** in turn, and out (in the s-array at step 4) will come the particular column of $\mathbf{H}_k\mathbf{W}$. A recommended assignment of **W** is presented in Section 5, where r=2 and makes the calculations frugal.

The Hamiltonian dynamic can now all be represented by the following system of non-homogeneous linear differential equations, of which a solution must be sought.

[2] If $\mathbf{HW} = \mathbf{H}_k\mathbf{W}$ and **H** is symmetric with rank r, then $\mathbf{H} = \mathbf{H}_k\mathbf{W}[\mathbf{W}^T\mathbf{H}_k\mathbf{W}]^{-1}\mathbf{W}^T\mathbf{H}_k$.

[3] In Section 3.4 of Smith (2000).

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{a} + \mathbf{B}\mathbf{x} \qquad (1)$$

$$where$$

$$\frac{\partial \mathbf{x}}{\partial t} = \begin{bmatrix} \frac{\partial \mathbf{q}}{\partial t} \\ \frac{\partial \mathbf{p}}{\partial t} \end{bmatrix}, \mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \end{bmatrix}, \mathbf{a} = \begin{bmatrix} \mathbf{0} \\ \mathbf{H}_k \mathbf{W}\left[\mathbf{W}^T \mathbf{H}_k \mathbf{W}\right]^{-1} \mathbf{W}^T \mathbf{H}_k \mathbf{q}_k - \nabla U(\mathbf{q}_k) \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & \frac{1}{m}\mathbf{I} \\ -\mathbf{H}_k \mathbf{W}\left[\mathbf{W}^T \mathbf{H}_k \mathbf{W}\right]^{-1} \mathbf{W}^T \mathbf{H}_k & \mathbf{0} \end{bmatrix}$$

Solutions follow in the wake of a singular value decomposition that is described in Section 4.

The Hamiltonian MCMC starts with momentums selected from a multivariate normal distribution with mean vector null and variance matrix m×**I**. The position vector, **q**, is first initialized making it and **p** the "current value" of **x**. These starting points are used to move down the path defined by the Hamiltonian dynamic, several iterative steps. The final value for **x** at the end of the path is either selected or rejected, according to an acceptance/rejection rule based on the Metropolis method. If the value is rejected, the current value (that initialized the path) is added to the simulation results. Otherwise the new value at the end of the path is taken as the current value and added to the simulation results. Lastly, the algorithm returns to the start by selecting new momentums (keeping the current value for spatial coordinates **q**), and the Hamiltonian dynamic is repeated making a new path. It's the collected values for **q**, ignoring **p**, that represent samples from the posterior distribution.

4. Efficient Calculation of the Exact Solution

An exact solution can be calculated for the differential equations (1). Understand that "exactness" is not intended to imply that the Hamiltonian dynamic is a perfect calculation because the gradient vector, ∇U(**q**), is still approximated by a linear equation. Its only that the differential equations have an analytical solution that generates $\mathbf{x}_{k+1}$ from the starting values given by $\mathbf{x}_k$, which is a small step represented by the time differential $\epsilon = t_{k+1} - t_k$. The analytic solutions for different k are used as part of a discretization that maps out the Hamiltonian dynamic as an approximation.

Equation (1) represents two partitions, of which the top partition is the following.

$$\frac{\partial \mathbf{q}}{\partial t} = \frac{1}{m}\mathbf{p} \qquad (2)$$

Both sides of (2) can be differentiated with respect to t to give these equations,

$$\frac{\partial^2 \mathbf{q}}{\partial t^2} = \frac{1}{m}\frac{\partial \mathbf{p}}{\partial t},$$

and these can now be plugged back into the bottom partition of (1) to find the reduced differential equations where **p** has been removed:

$$\begin{aligned}
&m\frac{\partial^2 \mathbf{q}}{\partial t^2} = \mathbf{a}_2 + \mathbf{B}_{21}\mathbf{q} \\
&where \\
&\mathbf{a}_2 = \mathbf{H}_k\mathbf{W}\left[\mathbf{W}^T\mathbf{H}_k\mathbf{W}\right]^{-1}\mathbf{W}^T\mathbf{H}_k\mathbf{q}_k - \nabla U(\mathbf{q}_k) \\
&\mathbf{B}_{21} = -\mathbf{H}_k\mathbf{W}\left[\mathbf{W}^T\mathbf{H}_k\mathbf{W}\right]^{-1}\mathbf{W}^T\mathbf{H}_k
\end{aligned} \tag{3}$$

The arrays $\mathbf{a}_2$ and $\mathbf{B}_{21}$ represent partitions of **a** and **B** in (1), and going forward the subscripts will be dropped to simplify notation; i.e., $\mathbf{a} \Leftarrow \mathbf{a}_2$, $\mathbf{B} \Leftarrow \mathbf{B}_{21}$.

The equations (3) represent a system of non-homogeneous second-order differential equations, coming with standard solutions. What is needed to extrapolate these equations to step k+1, to determine $\mathbf{q}_{k+1}$, are the initial values at k, namely $\mathbf{q}_k$ and $\mathbf{p}_k/m$ (or $\mathbf{q}_k{}' = \partial\mathbf{q}/\partial t$ evaluated at $\mathbf{q}_k$).

The calculation proceeds by diagonalizing the symmetric matrix $\mathbf{B}=\mathbf{P}\mathbf{D}\mathbf{P}^T$ where $\mathbf{P}^T\mathbf{P}=\mathbf{P}\mathbf{P}^T=\mathbf{I}$ and **D** is a diagonal matrix, which in principle can be very difficult. Fortunately, because **B** is greatly reduced rank (r<<n) and has special structure, its singular value decomposition can be computed very efficiently.

Let **L** be the Cholesky decomposition of the r×r matrix, $\mathbf{W}^T\mathbf{H}_k\mathbf{W}$, i.e., $\mathbf{L}\mathbf{L}^T=\mathbf{W}^T\mathbf{H}_k\mathbf{W}$. Compute **Y** where $\mathbf{Y}^T=\mathbf{L}^{-1}\mathbf{W}^T\mathbf{H}_k$, and noting that the inversion of **L** is avoided by using forward-substitution. Note that $\mathbf{Y}\mathbf{Y}^T=\mathbf{B}$. The rectangular n×r matrix, **Y**, is now subjected to the QR-algorithm, and because r<<n the method based on Householder transformations is likely preferred. This produces the n×n orthogonal matrix **Q**, where $\mathbf{Q}^T\mathbf{Q}=\mathbf{Q}\mathbf{Q}^T=\mathbf{I}$, and the n×r rectangular matrix **R**, such that **Y=QR**, and where **R** has the special structure indicated by (4).

$$\mathbf{R}_{n\times r} = \begin{bmatrix} \mathbf{V}_{r\times r} \\ \mathbf{0}_{(n-r)\times r} \end{bmatrix} \tag{4}$$

In (4), **V** is define to be an upper triangular matrix, and everything below it in (4) is null. The matrix **Q** is rarely formed explicitly, because its more efficient to form it implicitly as a computation involving a series of rank-1 Householder transformations.

Given that $\mathbf{B} = \mathbf{Q}\mathbf{R}\mathbf{R}^T\mathbf{Q}^T$, the QR algorithm does most of the work of diagonalizing **B** because $\mathbf{RR}^T$ would be diagonal[4] except for the small r×r matrix, $\mathbf{VV}^T$, located at the upper left corner. The last step is to find the orthogonal matrix $\mathbf{Q}_s$, and the diagonal matrix $\mathbf{D}_s$, where $\mathbf{Q}_s\mathbf{D}_s\mathbf{Q}_s^T=\mathbf{VV}^T$. Because r is small, any proven method will be applicable with only minor computing cost. However, $\mathbf{VV}^T$ does not need to be explicitly formed, because **V** can be diagonalized directly by the pre-multiplications and post-multiplications that represent Givens rotations that lead to a bi-diagonal form[5] and ultimately diagonalization, and this is one tactic offered by the singular value decomposition that's applied to the rectangular matrix **Y** (e.g., Golub and Van Loan, 1996, Sections 5.4.4 and 5.4.5). Regardless of which methods are used, because r is small the calculations are feasible. The sought diagonalization of **B**, where $\mathbf{B}=\mathbf{PDP}^T$, is now provided by the following specifications[6].

$$\mathbf{P}_{n\times n} = \mathbf{Q}\begin{bmatrix} \mathbf{Q}_s & \\ & \mathbf{I} \end{bmatrix}$$

$$\mathbf{D}_{n\times n} = \begin{bmatrix} \mathbf{D}_s & \\ & \end{bmatrix}$$

By making the transformation of variables, $\mathbf{q}^*=\mathbf{P}^T\mathbf{q}$, equation (3) becomes (5).

$$m\frac{\partial\, \mathbf{q}^*}{\partial t^2} = \mathbf{Pa} + \mathbf{Dq}^* \qquad (5)$$

The equation (5) can now be solved for the i-th entry of the vector $\mathbf{q}^*$, one item at a time. Given $\mathbf{q}_k^*$ and $\mathbf{q}_k^{*\prime}$ (or $\mathbf{P}^T\mathbf{q}_k{}'$), at time t_k, updates $\mathbf{q}_{k+1}^*$ and $\mathbf{q}_{k+1}^{*\prime}$ are derived at time $t_{k+1} = t_k + \epsilon$. These solutions are described in Figure 1, where $\{\mathbf{v}\}_i$ is a function that points to the i-th entry of a vector **v**, or where $\{\mathbf{D}\}_i$ is a function that points to the i-th diagonal of the diagonal matrix **D**. Once $\mathbf{q}_{k+1}^*$ and $\mathbf{q}_{k+1}^{*\prime}$ are fully estimated, they are transformed into $\mathbf{q}_{k+1} = \mathbf{Pq}_{k+1}^*$ and $\mathbf{q}_{k+1}{}'=\mathbf{Pq}_{k+1}^{*\prime}$, completing step k and preparing the way for step k+1.

[4] Actually, almost completely zeroed out.

[5] With (r-1)(r-2) Givens rotations an upper triangular matrix may transform into a bi-diagonal matrix.

[6] White space taken as zeroed out.

Figure 1. Solutions to Differential Scalar Equation; for i=1, 2, ... n.

Equation

$$\frac{\partial^2 g}{\partial t^2} = A + Dg$$

$$A = \frac{1}{m}\{\mathbf{Pa}\}_i$$

$$D = \frac{1}{m}\{\mathbf{D}\}_i$$

Case D=0

$$g(t_k + \boldsymbol{\varepsilon}) = a + b\boldsymbol{\varepsilon} + c\boldsymbol{\varepsilon}^2$$

$$g'(t_k + \boldsymbol{\varepsilon}) = b + \mathbf{2}c\boldsymbol{\varepsilon}$$

$$where$$

$$a = \{\mathbf{g}_k *\}_i$$

$$b = \{\mathbf{g}_k *'\}_i$$

$$c = \frac{A}{2}$$

Case D>0

$$g(t_k + \boldsymbol{\varepsilon}) = -\frac{A}{D} + a \times \exp\left(\boldsymbol{\varepsilon}\sqrt{D}\right) + b \times \exp\left(-\boldsymbol{\varepsilon}\sqrt{D}\right)$$

$$g'(t_k + \boldsymbol{\varepsilon}) = a\sqrt{D}\exp\left(\boldsymbol{\varepsilon}\sqrt{D}\right) - b\sqrt{D}\exp\left(-\boldsymbol{\varepsilon}\sqrt{D}\right)$$

$$where$$

$$a = \frac{1}{2}\left(\{\mathbf{g}_k *\}_i + \frac{\{\mathbf{g}_k *'\}_i}{\sqrt{D}} + \frac{A}{D}\right)$$

$$b = \frac{1}{2}\left(\{\mathbf{g}_k *\}_i - \frac{\{\mathbf{g}_k *'\}_i}{\sqrt{D}} + \frac{A}{D}\right)$$

Figure 1 continued.

Case D<0

$$g(t_k+\boldsymbol{\varepsilon}) = -\frac{A}{D} + a\times\cos\left(\boldsymbol{\varepsilon}\sqrt{-D}\right) + b\times\sin\left(\boldsymbol{\varepsilon}\sqrt{-D}\right)$$

$$g'(t_k+\boldsymbol{\varepsilon}) = -a\sqrt{-D}\sin\left(\boldsymbol{\varepsilon}\sqrt{-D}\right) + b\sqrt{-D}\cos\left(\boldsymbol{\varepsilon}\sqrt{-D}\right)$$

where

$$a = \left\{\mathbf{g}_k{}^*\right\}_i + \frac{A}{D}$$

$$b = \frac{\left\{\mathbf{g}_k{}^{*\prime}\right\}_i}{\sqrt{-D}}$$

Conclusion

$$\left\{\mathbf{g}_{k+1}{}^*\right\}_i = g(t_k+\boldsymbol{\varepsilon})$$

$$\left\{\mathbf{g}_{k+1}{}^{*\prime}\right\}_i = g'(t_k+\boldsymbol{\varepsilon})$$

5. Conclusions

Up to now, little has been offered on how to select directions to be placed in the matrix **W** that lead to second directional derivatives. At step k, the most prominent directions that impact the Hamiltonian dynamic are the gradient direction, ∇U($\mathbf{q}_k$), and the momentum direction, or $\mathbf{g}_k{}'$. Both of these directions will be available prior to the calculation of second derivatives, and hence its recommended to place these directions in **W**.

In the case that K(**p**)=$\frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}$, rather than $(2m)^{-1}\times\mathbf{p}^T\mathbf{I}\ \mathbf{p}$, the system (1) that leads to (3) can be transformed by introducing the Cholesky decomposition of **M**, i.e., where $\mathbf{ŁŁ}^T$=**M** for $\mathbf{Ł}_{n\times n}$ lower triangular. With the change of variables, $\mathbf{q}^{**} = \mathbf{Ł}^T\mathbf{q}$, equation (3) becomes:

$$\frac{\partial^2\mathbf{g}^{**}}{\partial t^2} = \mathbf{Ł}^{-1}\mathbf{a} + \mathbf{Ł}^{-1}\mathbf{BŁ}^{T-1}\mathbf{g}^{**} \qquad (6)$$

To solve (6), the effort turns into diagonalizing the matrix $\mathbf{Ł}^{-1}\mathbf{B}\ \mathbf{Ł}^{T\ -1}$, and may involve the singular value decomposition of the rectangular matrix $\mathbf{Ł}^{-1}\mathbf{H}_k\mathbf{WL}^{T\ -1}$, and otherwise following the methods in Section 4. Note than $\mathbf{Ł}_{n\times n}$ and $\mathbf{L}_{r\times r}$ are different matrices.

Without experience, its an open question if using information on some second derivatives compares well with the more established leapfrog method (see Neal 2011, Section 5.2.3.3) that uses only first derivatives. Nevertheless, it may be possible to adapt the new approach to make leapfrog-like steps by some unexplored method, e.g., by introducing half-steps where $\mathbf{q}_{k+1}$ and $\mathbf{q}_{k+1}'$ are updated in turn, or by alternating the updates[7] for $\{\mathbf{q}_{k+1}{}^{*}\}_i$ and $\{\mathbf{q}_{k+1}{}^{*\prime}\}_i$ but only for all i where D=0 (refer to Figure 1). However, it is worth noting that all three transformations in Figure 1 are volume preserving, and represent using an approximate Hamiltonian as permitted by Neal (2011, Section 5.5.5), and therefore its unclear if any leapfrog-like modification is warranted. Indeed, all the proposed calculations in Section 4 are volume preserving and reversible[8]. In any regard, the new method is just a tool that is available, and can be adapted for use with all the other established tools.

Lastly, there are alternative ways to approximate a Hessian matrix, **H**, that lead to linear equations that may substitute for Taylor's expansion, and also possibly find utility in Hamiltonian MCMC. If as before, $\mathbf{HW}=\mathbf{H}_k\mathbf{W}$, but additionally $\mathbf{H}=\mathbf{A}+\mathbf{ZZ}^T$, for a symmetric matrix **A** that is nominated, and for some n×r matrix **Z** that may have columns of entirely imaginary numbers, then provided that $\mathbf{W}^T\mathbf{H}_k\mathbf{W} - \mathbf{W}^T\mathbf{AW}$ is non-singular the following result holds.

$$\mathbf{H} = \mathbf{A} + \left[\mathbf{H}_k\mathbf{W} - \mathbf{AW}\right]\left[\mathbf{W}^T\mathbf{H}_k\mathbf{W} - \mathbf{W}^T\mathbf{AW}\right]^{-1}\left[\mathbf{H}_k\mathbf{W} - \mathbf{AW}\right]^T$$

When **A** is a null matrix, the result generates what was already described in footnote 2. If $\mathbf{A}=\alpha\mathbf{I}$, the special structure of **H** permits it rapid diagonalization, though the details are more involved. Perhaps with an appropriate selection of the parameter α, this may find application with Hamiltonian MCMC where the matrix **H** is now non-singular, or close to it. It is worth noting that both the Hessian, and its inverse, can be approximated this way given side conditions that relate second derivatives to a direction set. Such approximations for the inverse Hessian matrix find use in quasi-Newton iteration that's part of optimization.

[7] Keeping one set unchanged and allowing the other set to change, depending on half-step.

[8] With the step-dependent vector **a** and matrices **P** and **D** available in reverse order (with checkpointing), where step k returns from its high value back to 1, the functions of Figure 1 become available where $\mathbf{q}_{k+1}$ and $-\mathbf{q}_{k+1}'$ map in reverse back to $\mathbf{q}_k$ and $-\mathbf{q}_k'$. With reversibility, **q** and **q'** at the end of the Hamiltonian path return to their initial values when **q'** is negated (before and after).

References

Golub, G.H., and C.F. Van Loan, 1996, *Matrix Computation*, Third Edition. The John Hopkins University Press.

Griewank, A., 2000, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM.

Neal, R.M., 2011, *Handbook of Markov Chain Monte Carlo*, Chapter 5. Brooks, S., A. Gelman, G.L. Jones and X-L Meng, editors. CRC Press.

Smith, S.P., 2000, A Tutorial on Simplicity and Computation Differentiation for Statisticians. Memo. viXra: 1702.0243.