Polynomial accelerated MCMC, and other sampling algorithms inspired by computational optimization

Colin Fox

Abstract Polynomial acceleration methods from computational optimization can be applied to accelerating MCMC. For example, a geometrically convergent MCMC may be accelerated to be a perfect sampler in special circumstances. An equivalence between Gibbs sampling of Gaussian distributions and classical iterative methods can be established using matrix splittings, allowing direct application of Chebyshev acceleration. The conjugate gradient method can also be adapted to give an accelerated sampler for Gaussian distributions, that is perfect in exact arithmetic.

Key words: (MCMC, polynomial acceleration, Gibbs sampling, Gauss-Seidel, Chebyshev acceleration, conjugate gradient)

1 Introduction

Standard Markov chain Monte Carlo (MCMC) algorithms simulate a *homogeneous* Markov chain by performing a stationary linear iteration on the space of probability distributions. The repeated application of a *fixed* kernel results in geometric convergence of the Markov chain, just as it does for the *stationary* linear iterative solvers used to solve systems of linear equations. Stationary linear solvers were state-of-the-art in the 1950's, but are now considered very slow precisely because they are geometrically convergent.

In this paper, methods for accelerating stationary linear iterations developed in the field of numerical computation are applied to accelerating MCMC, both in the general setting of a Markov chain designed to target an arbitrary distribution π (section 2), and also in the specific setting of Gibbs sampling from the multivariate Gaussian distribution $\mathrm{N}\left(0, \mathbf{A}^{-1}\right)$ with known precision matrix $\mathbf{A}$ (sections 4 and 5).

Colin Fox
Physics Department, University of Otago. e-mail: fox@physics.otago.ac.nz

We will see that polynomial acceleration of a geometrically convergent MCMC can, in certain cases, generate perfect samples in finite time.

The special case of Gibbs sampling applied to Gaussian distributions is precisely equivalent to classical iterative methods for solving linear systems understood in terms of matrix splittings, as shown in section 3. Chebyshev acceleration, which is optimal in a certain sense for matrix-splitting methods, can therefore be used to optimally accelerate the Gibbs sampler, as demonstrated in section 4. The conjugate gradient optimization algorithm may also be viewed as a polynomial acceleration in which the eigenvalues of the iteration operator are estimated within the iteration. A 'conjugate gradient sampler' for Gaussian distributions is presented in section 5.

This work takes place within our ongoing efforts in computational (Bayesian) inference that utilizes sampling methods, specifically MCMC. In these problems one wishes to evaluate expectations with respect to a given (posterior) target distribution π over a typically high-dimensional state space. Since the statistics over π are analytically intractable, the best current technology is Monte Carlo integration with importance sampling using samples drawn from π via a random-walk MCMC. That can be very slow. By identifying sampling with optimization, at mathematical and algorithmic levels, we look to adapt the sophisticated methods developed for accelerating computational optimization to computational sampling.

We were also curious about Gibbs sampling being referred to as "stochastic relaxation" in [11], and whether this was related to the "relaxation" methods of numerical analysis in an intuitive sense or in a more formal mathematical sense.

Throughout this paper it is taken as understood that the tasks of computational optimization and solution of systems of equations are equivalent; the normal equations for the optimization form the system to be solved. The terms *solve* and *optimize* are used interchangeably.

2 Polynomial acceleration of MCMC

This section provides a cartoon of polynomial acceleration of distributional convergence in standard MCMC, to convey the ideas behind polynomial acceleration that can get hidden in a more formal presentation. The weighted-subsampling scheme in section 2.2 does not necessarily lead to a practical technique, but does show the remarkable speedup possible.

2.1 Errors and convergence in standard MCMC

The algorithmic mainstay of MCMC methods is the simulation of a homogeneous Markov chain $\{X_0, X_1, \ldots\}$ that tends to some desired target distribution π. The chain is homogeneous because the Markov chain is constructed by repeatedly simulating a *fixed* transition kernel $\mathscr{P}$ constructed so that π is invariant, i.e.,

$$\pi \mathscr{P} = \pi,$$

typically using Metropolis-Hastings (MH) dynamics that ensures that $\mathscr{P}$ and π are in detailed balance.

When the chain is initialized with $X_0 \sim \pi^{(0)}$, the n-step distribution (over X_n) is

$$\pi^{(n)} = \pi^{(n-1)} \mathscr{P} = \pi^{(0)} \mathscr{P}^n.$$

The difference between this distribution and the target distribution π,

$$\pi^{(n)} - \pi = \left(\pi^{(0)} - \pi\right) \mathscr{P}^n, \tag{1}$$

is called the n-step distribution error. Note how the magnitude of the error goes to zero according to the initial distribution error multiplied by the polynomial $\mathscr{P}^n$ of the transition kernel.

All iteration schemes lead to a n-step distribution error of this form, i.e. the initial error multiplied by an n-th order polynomial P_n of the transition kernel. In numerical analysis it is usual to write this *error polynomial* as a polynomial in $I - \mathscr{P}$. Hence the error polynomial in this case is

$$P_n(I - \mathscr{P}) = \mathscr{P}^n = (I - (I - \mathscr{P}))^n \quad \text{or} \quad P_n(\lambda) = (1 - \lambda)^n. \tag{2}$$

All error polynomials satisfy $P_n(0) = 1$, since $\mathscr{P} = I$ leaves the iterate (and error) unchanged. This error polynomial has only one (repeated) zero at $\lambda = 1$.

The second form in Eq. (2) emphasizes that the error polynomial may be evaluated over the eigenvalues of $I - \mathscr{P}$. Since $\mathscr{P}$ is a stochastic kernel, all eigenvalues of $I - \mathscr{P}$ are contained in $[0,2]$. The error tends to zero when the eigenvalues of $I - \mathscr{P}$ in directions other than π are bounded away from 0 and 2, as is guaranteed by standard results for a *convergent* MCMC.

Thus, a homogeneous MCMC produces a sample correctly distributed as π either after one step (when all eigenvalues of $I - \mathscr{P}$ in directions other than π equal 1), or in the limit $n \to \infty$ (when any eigenvalue in a direction other than π is not 1). In the latter case, the distributional error in Eq. (1) will be dominated by the error in the direction of the eigenvalue of $I - \mathscr{P}$ furthest from 1, λ_*, hence decays as $(1 - \lambda_*)^n$, and the convergence is *geometric*.

2.2 Acceleration by weighted subsampling

The key idea in polynomial acceleration is to modify the iteration so that the error polynomial is 'better' than the stationary case in Eqs. (1) and (2), in the sense of smaller error. A simple way to modify the iteration in the setting of MCMC is to subsample with weights. This does not allow complete freedom in choosing the error polynomial, hence there is room for improvement. (Finding an optimal modification is an open problem.) The recipe I will use is: run n steps of a standard MCMC

starting at $x^{(0)} \sim \pi^{(0)}$ to produce the realization $\{x^{(1)}, x^{(2)}, \ldots x^{(n)}\}$ and then choose $x = x^{(i)}$ w.p. (with probabilities) $\{\alpha_i\}_{i=1}^n$ (where $\alpha_i \geq 0$ and $\sum_i \alpha_i = 1$). The resulting sample is distributed as the mixture model

$$x \sim \pi^{(0)} \sum_{i=1}^{n} \alpha_i \mathscr{P}^i$$

with the individual distributions related by increasing powers of $\mathscr{P}$. Weighted subsampling is also considered by Łatuszyński & Roberts [16]. The associated error polynomial is then

$$Q_n = \sum_{i=1}^{n} \alpha_i (1-\lambda)^i$$

which is an n-th order Lorentz polynomial. Since we choose the coefficients $\{\alpha_i\}_{i=1}^n$ we have some freedom in choosing the error polynomial. In special circumstances, it is possible to choose an error polynomial that is zero at the eigenvalues of $I - \mathscr{P}$ other than $\lambda = 0$, in which case subsampling with weights generates a *perfect* sample from π. That is possible, for example, when the sample space is finite, with s states. Then $I - \mathscr{P}$ has at most s distinct eigenvalues and when the $s-1$ eigenvalues other than 0 can be the zeros of a Lorentz polynomial it is possible to choose Q_n to give zero distribution error.

Consider the simple example in which we want to sample from a state-space with $s = 3$ states with target pmf $\pi = (1/3, 1/3, 1/3)$. A Markov chain that targets π can be generated by repeatedly simulating the transition matrix

$$\mathscr{P} = \begin{pmatrix} \frac{1}{48} & \frac{11}{24} & \frac{25}{48} \\ \frac{11}{24} & \frac{1}{12} & \frac{11}{24} \\ \frac{25}{48} & \frac{11}{24} & \frac{1}{48} \end{pmatrix}$$

which can easily be seen to be in detailed balance with π and gives a chain that is irreducible and aperiodic. Note that convergence is geometric, and that

$$\mathscr{P}^2 = \begin{pmatrix} \frac{185}{384} & \frac{55}{192} & \frac{89}{384} \\ \frac{55}{192} & \frac{41}{96} & \frac{55}{192} \\ \frac{89}{384} & \frac{55}{192} & \frac{185}{384} \end{pmatrix}, \mathscr{P}^3 = \begin{pmatrix} \frac{805}{3072} & \frac{539}{1536} & \frac{1189}{3072} \\ \frac{539}{1536} & \frac{229}{768} & \frac{539}{1536} \\ \frac{1189}{3072} & \frac{539}{1536} & \frac{805}{3072} \end{pmatrix}, \ldots, \mathscr{P}^\infty = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{pmatrix}$$

so that as $n \to \infty$ the chain converges to a sample from π that is independent of the starting state. This chain can be accelerated by weighted subsampling, as follows:

1. Start with (any) $x^{(0)}$, simulate 3 steps with $\mathscr{P}$ to get $x^{(1)}$, $x^{(2)}$, $x^{(3)}$.
2. Sample x from $(x^{(1)}, x^{(2)}, x^{(3)})$ w.p. $\left(\frac{1}{11}\ \frac{14}{33}\ \frac{16}{33}\right)$.

The resulting x is an exact draw from π, and independent of the starting state, because $\frac{1}{11}\mathscr{P} + \frac{14}{33}\mathscr{P}^2 + \frac{16}{33}\mathscr{P}^3 = \mathscr{P}^\infty$. It is left as an exercise to explicitly construct the error polynomial to see how the example was constructed.

As mentioned above, there are a few practical difficulties with this simple subsampling scheme. An obvious limitation is that the zeros of a Lorentz polynomial

only occur for eigenvalues that decorrelate the chain, in which case the polynomial 'acceleration' that draws exact (and i.i.d.) samples actually *increases* the variance in a CLT (see e.g. [16]). However, one might argue that distributional convergence is improved, which may be important in some settings. A further difficulty occurs when n needs to be large since we really want to specify the zeros of the error polynomial yet these are not a stable numerical function of the $\{\alpha_i\}$. Furthermore, n must be chosen in advance which is typically not convenient. All these difficulties may be circumvented in the case of a Gaussian target by using a second-order iteration.

3 Gibbs sampling of Gaussians is Gauss-Seidel iteration

The Gibbs sampling algorithm [11] repeatedly samples from (block) conditional distributions of π. We consider the simplest, and original, version of Gibbs sampling in which one iteration consists of conditional sampling along each coordinate direction in sequence, see e.g. Turčin 1971 [21], also known as Glauber dynamics [12], the local heat-bath algorithm [5], and the sequential updating method.

3.1 Normal distributions

We now focus on the case of Gibbs sampling from the multivariate Normal (or Gaussian) distribution $\mathrm{N}\left(0, \mathbf{A}^{-1}\right)$ with known precision matrix $\mathbf{A}$. This situation commonly occurs in (hierarchical) Bayesian analyses when spatial dependencies are modelled via neighbourhood relationships, leading to a Gaussian Markov random field (GMRF) with sparse precision matrix [15]. Both $\mathbf{A}$ and the covariance matrix $\Sigma = \mathbf{A}^{-1}$ are symmetric positive definite. In d dimensions the density function is (written in the *natural parametrization*)

$$\pi(\mathbf{x}) = \sqrt{\frac{\det(\mathbf{A})}{2\pi^d}} \exp\left\{-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x}\right\}. \tag{3}$$

The mean vector $\bar{\mathbf{x}}$ satisfies

$$\mathbf{A}\bar{\mathbf{x}} = \mathbf{b} \tag{4}$$

which gives the first indication that solution of linear equations is relevant to Gaussian distributions.

Cholesky factorization is the preferred method for solving moderately sized linear systems with symmetric and positive definite coefficient matrix, and also for sampling from moderate dimension Gaussian distributions [19] (also called global heat bath [5]). We are interested in the case where the state-space dimension d is large and $\mathbf{A}$ is sparse. Then, iterative methods such as the Gibbs sampler are attractive as the main cost per iteration is operation by the precision matrix $\mathbf{A}$, which is cheap, and memory requirements are low.

The Gibbs sampler updates components via the conditional distributions, which are also Gaussian. Hence choosing $\pi^{(0)}$ to be Gaussian results in a sequence of Gaussian n-step distributions. Since these n-step distributions converge to π, the sequence of n-step covariance matrices converge to Σ, i.e., $\Sigma^{(n)} \to \Sigma$. One of the motivations for this work was to understand what decomposition of the matrix Σ this sequence is effectively performing. Many matrix decompositions are known in numerical analysis and we were curious to see if Gibbs sampling was effectively performing one of them.

3.2 Matrix formulation of Gibbs sampling from $\mathrm{N}(0,\mathbf{A}^{-1})$

Let $\mathbf{y} = (y_1, y_2, ..., y_n)^T$ denote the state of the Gibbs sampler. Component-wise Gibbs updates each component in sequence from the (normal) conditional distributions. One 'sweep' over all n components can be written [14]

$$\mathbf{y}^{(k+1)} = -\mathbf{D}^{-1}\mathbf{L}\mathbf{y}^{(k+1)} - \mathbf{D}^{-1}\mathbf{L}^T\mathbf{y}^{(k)} + \mathbf{D}^{-1/2}\mathbf{z}^{(k)} \tag{5}$$

where $\mathbf{D} = \mathrm{diag}(\mathbf{A})$, $\mathbf{L}$ is the strictly lower triangular part of $\mathbf{A}$, and $\mathbf{z}^{(k-1)} \sim \mathrm{N}(\mathbf{0},\mathbf{I})$. Since $\mathbf{D}$ is invertible, the iteration can be written as the stochastic AR(1) process

$$\mathbf{y}^{(k+1)} = \mathbf{G}\mathbf{y}^{(k)} + \mathbf{c}^{(k)}$$

where $\mathbf{c}^{(k)}$ are i.i.d. draws from a 'noise' distribution with zero mean and finite covariance.

3.3 Matrix splitting form of stationary iterative methods

Since about 1965, the *matrix splitting* formalism has been the standard for formulating and understanding the classical iteration schemes used to solve linear systems of equations, as in Eq. (4). The *splitting* $\mathbf{A} = \mathbf{M} - \mathbf{N}$ converts the linear system to $\mathbf{M}\mathbf{x} = \mathbf{N}\mathbf{x} + \mathbf{b}$. When $\mathbf{M}$ is invertible, this may be written

$$\mathbf{x} = \mathbf{M}^{-1}\mathbf{N}\mathbf{x} + \mathbf{M}^{-1}\mathbf{b}.$$

Classical iterative methods compute successive approximations to the solution by repeatedly applying the iteration

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \mathbf{M}^{-1}\mathbf{N}\mathbf{x}^{(k)} + \mathbf{M}^{-1}\mathbf{b} \\ &= \mathbf{G}\mathbf{x}^{(k)} + \mathbf{g}.\end{aligned}$$

The iteration is *convergent* if the sequence of iterates converge for any $\mathbf{x}^{(0)}$.

Many splittings use terms in $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ where $\mathbf{L}$ is the strictly lower triangular part of $\mathbf{A}$, $\mathbf{D}$ is the diagonal of $\mathbf{A}$, and $\mathbf{U}$ is the strictly upper triangular part of $\mathbf{A}$. For example, Gauss-Seidel iteration, that sequentially solves for each component using the most recent values, corresponds to the splitting $\mathbf{M} = \mathbf{L} + \mathbf{D}$. The resulting iteration for a sweep over all components in sequence is

$$\mathbf{x}^{(k+1)} = -\mathbf{D}^{-1}\mathbf{L}\mathbf{x}^{(k+1)} - \mathbf{D}^{-1}\mathbf{L}^{\mathrm{T}}\mathbf{x}^{(k)} + \mathbf{D}^{-1}\mathbf{b}. \tag{6}$$

The similarity between Gauss-Seidel iteration in Eq. (6) and the matrix formulation of Gibbs sampling in Eq. (5) is obvious. The only difference is that whereas in each iteration of Gauss-Seidel the constant vector $\mathbf{D}^{-1}\mathbf{b}$ is added, in Gibbs sampling the i.i.d. random vector $\mathbf{D}^{-1/2}\mathbf{z}^{(k)}$ is added. This equivalence has been known for some time; it was explicitly stated in Amit and Grenander 1991 [2] and is implicit in Adler 1981 [1].

3.4 Matrix splittings give generalized Gibbs samplers

The standard Gibbs sampler in Eq. (6) and Gauss-Seidel iteration in Eq. (5) are equivalent in the sense that they correspond to the same splitting of the precision matrix. In fact any splitting of the precision matrix leads to a (generalized) Gibbs sampler for $\mathrm{N}(0, \mathbf{A}^{-1})$. What makes this equivalence interesting and useful is that the generalized Gibbs sampler converges (in distribution) if and only if the stationary linear iteration converges (in value); hence convergent Gibbs samplers are equivalent to convergent matrix splittings. The following theorem formalizes this statement.

Theorem 1. *Let* $\mathbf{A} = \mathbf{M} - \mathbf{N}$ *be a splitting with* $\mathbf{M}$ *invertible. The stationary linear solver*

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \mathbf{M}^{-1}\mathbf{N}\mathbf{x}^{(k)} + \mathbf{M}^{-1}\mathbf{b} \\ &= \mathbf{G}\mathbf{x}^{(k)} + \mathbf{M}^{-1}\mathbf{b} \end{aligned} \tag{7}$$

converges, if and only if the random iteration

$$\begin{aligned} \mathbf{y}^{(k+1)} &= \mathbf{M}^{-1}\mathbf{N}\mathbf{y}^{(k)} + \mathbf{M}^{-1}\mathbf{c}^{(k)} \\ &= \mathbf{G}\mathbf{y}^{(k)} + \mathbf{M}^{-1}\mathbf{c}^{(k)} \end{aligned} \tag{8}$$

converges in distribution. Here $\mathbf{c}^{(k)} \overset{\text{iid}}{\sim} \pi_{\mathrm{n}}$ *is any 'noise' distribution that has zero mean and finite variance.*

Proof. (outline) Each converges iff the spectral radius $\rho(\mathbf{G}) < 1$. □

A complete proof is given in Fox & Parker 2013 [9]. (A more general theory allowing $\mathbf{G}$ to be random can be found in [6].) We first saw this result in one direction in Goodman & Sokal 1989 [14] and Galli & Gao 2001 [10]. Further, it can

be shown [9] that the mean converges with asymptotic convergence factor $\rho(\mathbf{G})$, and covariance with $\rho(\mathbf{G})^2$ (see also [18]). Thus, the *rate* of convergence is also the same for both the Gibbs sampler and the linear solver derived from a splitting. Hence the optimal solver leads to the optimal Gibbs sampler, and vice versa.

3.5 Some (not so common) Gibbs samplers for $\mathrm{N}(0,\mathbf{A}^{-1})$

There are many matrix splittings known in the numerical analysis community, with conditions for convergence being well established. Most introductory texts on numerical analysis cover the topic of stationary iterative methods and give several classical splittings. Some of these are tabulated in Table 1 with increasing sophistication and (roughly) speed listed from top to bottom. Conditions that guarantee convergence, taken from the numerical analysis literature, are also listed for the case where $\mathbf{A}$ is symmetric positive-definite.

Table 1 Some classical matrix splittings and the derived Gibbs samplers. Conditions for convergence are given in the right-most column, for $\mathbf{A}$ symmetric positive definite. Jacobi iteration converges when $\mathbf{A}$ is strictly diagonally dominant (SDD).

splitting/sampler	$\mathbf{M}$	$\mathbf{Var}(\mathbf{c}^{(k)}) = \mathbf{M}^T + \mathbf{N}$	converge if
Richardson	$\frac{1}{\omega}\mathbf{I}$	$\frac{2}{\omega}\mathbf{I} - \mathbf{A}$	$0 < \omega < \frac{2}{\rho(\mathbf{A})}$
Jacobi	$\mathbf{D}$	$2\mathbf{D} - \mathbf{A}$	$\mathbf{A}$ SDD
GS/Gibbs	$\mathbf{D} + \mathbf{L}$	$\mathbf{D}$	always
SOR/B&F	$\frac{1}{\omega}\mathbf{D} + \mathbf{L}$	$\frac{2-\omega}{\omega}\mathbf{D}$	$0 < \omega < 2$
SSOR/REGS	$\frac{\omega}{2-\omega}\mathbf{M}_{\mathrm{SOR}}\mathbf{D}^{-1}\mathbf{M}_{\mathrm{SOR}}^{\mathrm{T}}$	$\frac{\omega}{2-\omega}\left(\mathbf{M}_{\mathrm{SOR}}\mathbf{D}^{-1}\mathbf{M}_{\mathrm{SOR}}^{T} + \mathbf{N}_{\mathrm{SOR}}^{T}\mathbf{D}^{-1}\mathbf{N}_{\mathrm{SOR}}\right)$	$0 < \omega < 2$

The convenience of a splitting depends on being able to cheaply solve systems of the form $\mathbf{Mu} = \mathbf{r}$ given any vector $\mathbf{r}$. When the splitting is used to generate a Gibbs sampler, as in Eq. (8), it is also necessary to draw realizations of the noise $\mathbf{c}^{(k)} \sim \mathrm{N}(0, \mathbf{M}^T + \mathbf{N})$, so the covariance matrix $\mathbf{M}^T + \mathbf{N}$ needs to have some convenient form.

It is interesting to note that the simplest splittings – Richardson and Jacobi – give simple stationary iterative solvers because it is cheap to operate by $\mathbf{M}^{-1}$ in these cases. However, the required noise covariance matrix is not necessarily simple and so these splittings don't give particularly useful Gibbs samplers.

The Gauss-Seidel (GS) splitting, that gives the standard component-wise Gibbs sampler, hits a 'sweet-spot' in terms of simplicity of the required matrix solution and noise sampling problems. The matrix $\mathbf{M}$ is lower-triangular, so operation by $\mathbf{M}^{-1}$ is straightforward by *forward substitution*, while the noise covariance is diagonal which presents a simple sampling problem. It is no surprise, therefore, that the standard Gibbs sampler was the first of these methods to be discovered. We see from the right column in Table 1 that the Gauss-Seidel iteration is unconditionally

convergent, hence Thm. 1 guarantees that so is the component-wise Gibbs sampler – but we already knew this from standard convergence results for the Gibbs sampler.

An early method for accelerating the Gauss-Seidel iteration, due to Young and Frankel in 1950, introduces a *relaxation parameter* ω and modifies the iteration to $\mathbf{x}^{(k+1)} = (1-\omega)\mathbf{x}^{(k)} + \omega\left(\mathbf{G}\mathbf{x}^{(k)} + \mathbf{M}^{-1}\mathbf{b}\right)$. This *successive over-relaxation* (SOR) method effectively uses the splitting shown on the row labeled SOR in Table 1. It can be shown that the method converges for $0 < \omega < 2$, though finding values of ω that actually increase convergence speed is problem-specific and can be difficult. The equivalent accelerated Gibbs sampler has been discovered a few times: initially by Adler in 1981 [1] in the physics literature, later in the statistics literature by Barone & Frigessi in 1990 [4] who subsequently referred to it (immodestly) as the 'method of Barone and Frigessi', and in Amit & Grenander 1991 [2].

A *symmetric* splitting, for which $\mathbf{M}$ and hence $\mathbf{N}$ is symmetric, has the desirable property that the iteration operator $\mathbf{G}$ has real eigenvalues. A simple way to achieve this is to perform a forwards then backwards sweep of SOR giving the *symmetric successive over-relaxation* (SSOR) method introduced by Young [22]. The effective splitting is listed in Table 1. The equivalent Gibbs sampler was introduced by Roberts & Sahu 1997 [18] as a reversible kernel produced by a forward then backward sweep of the standard Gibbs sampler, under the title of the REGS sampler. Polynomial acceleration of this sampler is developed in the next section.

4 Polynomial acceleration of Gibbs sampling

Sampling from $\mathrm{N}(\bar{\mathbf{x}}, \mathbf{A}^{-1})$, where $\mathbf{A}\bar{\mathbf{x}} = \mathbf{b}$, using the matrix splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$, with $\mathbf{M}$ invertible, determines the iteration operator $\mathbf{G} = \mathbf{M}^{-1}\mathbf{N}$ and noise distribution $\mathbf{c}^{(k)} \overset{\text{iid}}{\sim} \mathrm{N}(0, \mathbf{M}^T + \mathbf{N})$. One sweep of the resulting Gibbs sampler is the matrix iteration

$$\mathbf{y}^{(k+1)} = \mathbf{G}\mathbf{y}^{(k)} + \mathbf{M}^{-1}(\mathbf{c}^{(k)} + \mathbf{b}) \tag{9}$$

that combines Eqs. (7) and (8) to converge in both mean and covariance.

4.1 A closer look at convergence

Since both the mean and covariance are invariant under the iteration in Eq. (9), the n-step error in the mean is

$$\mathrm{E}\left(\mathbf{y}^{(n)}\right) - \bar{\mathbf{x}} = \mathbf{G}^n\left[\mathrm{E}\left(\mathbf{y}^{(0)}\right) - \bar{\mathbf{x}}\right],$$

and the error in variance is

$$\mathrm{Var}\left(\mathbf{y}^{(n)}\right) - \mathbf{A}^{-1} = \mathbf{G}^n\left[\mathrm{Var}\left(\mathbf{y}^{(0)}\right) - \mathbf{A}^{-1}\right]\mathbf{G}^n.$$

Both these error terms show that the n-step error is the initial error operated on by the n-th order (matrix) polynomial $\mathbf{G}^n$. Hence, the asymptotic average convergence factor is $\rho(\mathbf{G})$ for the mean, and $\rho(\mathbf{G})^2$ for the covariance. These results also appear in Roberts & Sahu 1997 [18].

Thus, the error polynomial for the iteration is

$$P_n(\mathbf{I}-\mathbf{G}) = (\mathbf{I}-(\mathbf{I}-\mathbf{G}))^n = \left(\mathbf{I}-\mathbf{M}^{-1}\mathbf{A}\right)^n \quad \text{or} \quad P_n(\lambda) = (1-\lambda)^n$$

which has the same form as in Eq. (2) because this iteration is also stationary, though now the eigenvalues are of the matrix $\mathbf{M}^{-1}\mathbf{A}$.

In particular, the solver and sampler have exactly the same error polynomial. This is a very important observation, since it means that methods for improving the error polynomial of the solver will also improve convergence of the generalized Gibbs sampler. Further, since the solver and sampler have exactly the same asymptotic average convergence factor, the optimal solver will also be the optimal sampler. Thus, the task of finding a fast Gibbs sampler (for Gaussian distributions) is reduced to the task of consulting the numerical linear algebra literature to find a fast linear iterative solver.

4.2 Chebyshev acceleration

Golub and Varga 1961 [13] introduced the splitting

$$\mathbf{A} = \frac{1}{\tau}\mathbf{M} + \left(1-\frac{1}{\tau}\right)\mathbf{M} - \mathbf{N},$$

with parameter τ, that the gives the iteration operator

$$\mathbf{G}_\tau = \left(\mathbf{I}-\tau\mathbf{M}^{-1}\mathbf{A}\right). \tag{10}$$

Repeated iteration using this splitting results in the error polynomial $P_n(\lambda) = (1-\tau\lambda)^n$, while n iterations using the *sequence* of parameters $\tau_1, \tau_2, \ldots, \tau_n$ results in the error polynomial

$$P_n(\lambda) = \prod_{l=1}^{n}(1-\tau_l\lambda).$$

Note that the zeros of P_n can be chosen; they are just $1/\tau_1, 1/\tau_2, \ldots, 1/\tau_n$. The resulting iteration is non-stationary (because the iteration operator changes each iteration), hence the derived Gibbs sampler simulates a non-homogeneous Markov chain.

When estimates of the extreme eigenvalues $\lambda_{\min}$ and $\lambda_{\max}$ of $\mathbf{M}^{-1}\mathbf{A}$ are available ($\lambda_{\min}$ and $\lambda_{\max}$ are real when $\mathbf{M}$ is symmetric), the error polynomial may be chosen to be optimal in the sense that it has minimum maximum modulus over the interval $[\lambda_{\min}, \lambda_{\max}]$. The solution is the well-known scaled Chebyshev polynomial with

zeros

$$\frac{1}{\tau_l} = \frac{\lambda_{\max} + \lambda_{\min}}{2} + \frac{\lambda_{\max} - \lambda_{\min}}{2} \cos\left(\pi \frac{2l+1}{2n}\right) \quad l = 0, 1, 2, \ldots, n-1. \qquad (11)$$

The potential improvement in rate of convergence achievable by the Chebyshev error polynomial is shown in Fig. 1 that shows the Chebyshev (solid) and default (dashed) error polynomials for a random covariance over $d = 10$ variables, after $n = 10$ iterations.

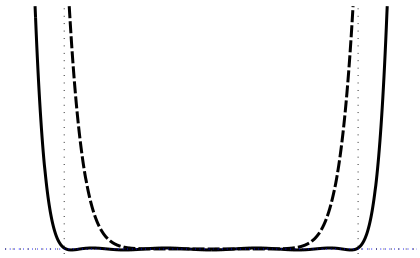


Fig. 1 The default error polynomial (dashed) and Chebyshev error polynomial (solid) after 10 iterations. Vertical dotted lines show the minimum and maximum eigenvalues of $\mathbf{M}^{-1}\mathbf{A}$.

The largest value of the default error polynomial occurs at the extreme eigenvalues of $\mathbf{M}^{-1}\mathbf{A}$, as we expect from standard MCMC convergence theory. The Chebyshev polynomial achieves a much lower maximum value over the interval, at the expense of some 'ripple' in the interval that is of no consequence for convergence. In this case the Chebyshev acceleration gives a factor of 300 improvement in convergence, i.e. the distribution error is 300 times smaller, after just 10 iterations.

An explicit calculation of the maximum of the scaled Chebyshev polynomial over the interval $[\lambda_{\min}, \lambda_{\max}]$ shows that the asymptotic average reduction factor (see e.g. Axelsson 1996 [3]) is

$$\sigma = \frac{1 - \sqrt{\lambda_{\min}/\lambda_{\max}}}{1 + \sqrt{\lambda_{\min}/\lambda_{\max}}},$$

and that this is necessarily better (smaller) than the per-iteration error reduction factor of the un-accelerated iteration.

4.3 Second-order accelerated sampler

The first-order polynomial-accelerated iteration turns out to be numerically unstable, because the iteration operators in Eq. (10) may have spectral radius greater than 1, and also suffers from having to choose the number of iterations n in advance. Numerical stability, and optimality at each step, is given by the second-order iteration [3]

$$\mathbf{y}^{(k+1)} = (1-\alpha_k)\mathbf{y}^{(k-1)} + \alpha_k \mathbf{y}^{(k)} + \alpha_k \tau_k \mathbf{M}^{-1}(\mathbf{c}^{(k)} - \mathbf{A}\mathbf{y}^{(k)}) \tag{12}$$

with α_k and τ_k chosen so the error polynomial satisfies the Chebyshev recursion.

Theorem 2. *If $\{\alpha_k\}$ and $\{\tau_k\}$ are such that the 2nd-order solver converges, then the 2nd-order sampler in Eq. (12) converges. Further, the error polynomial is optimal, at each step, for both mean and covariance.*

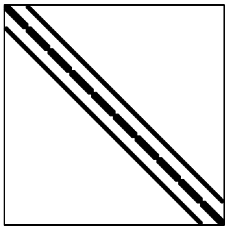
A proof of this theorem and details of a practical second-order Chebyshev accelerated Gibbs sampling algorithm are given in Fox and Parker 2013 [8].

4.3.1 An example with $d = 10 \times 10$

Consider the locally-linear Gaussian distribution defined by the precision matrix [15]

$$[\mathbf{A}]_{ij} = 10^{-4}\delta_{ij} + \begin{cases} n_i & \text{if } i = j, \\ -1 & \text{if } i \neq j \text{ and } ||s_i - s_j||_2 \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

We compute an example on the square 10×10 lattice, so the problem dimension is $d = 100$. The precision matrix inherits the neighbourhood structure of the lattice, so is sparse, with non-zero pattern:



The convergence in n-step covariance of various Gibbs samplers applied to this distribution is shown in Fig. 2. The dashed line shows the SSOR (or REGS) sampler using the optimal SOR parameter of $\omega = 1.6641$. The solid curve shows the standard REGS (forward and backward sweep of Gibbs) sampler ($\omega = 1$). Dash-dot lines show the Chebyshev accelerated SSOR sampler. It is clear that the Chebyshev accelerated sampler is considerably faster than standard Gibbs sampling, in this case $\approx 10^4$ times faster. The dotted lines in Fig. 2 show the work and error for a sample drawn using the Cholesky factorization of $\mathbf{A}$, and confirm that Cholesky factoring is the method of choice for moderately-sized problems.

4.3.2 An example with $d = 10^6$

Fig. 3 shows a sample from a locally linear Gaussian random field, with the same definition of the precision matrix as the previous example, on the 3-dimensional lattice with $d = 100 \times 100 \times 100$, computed using the Chebyshev accelerated SSOR sampler. This problem has $d = 10^6$ which is much larger than could be calculated

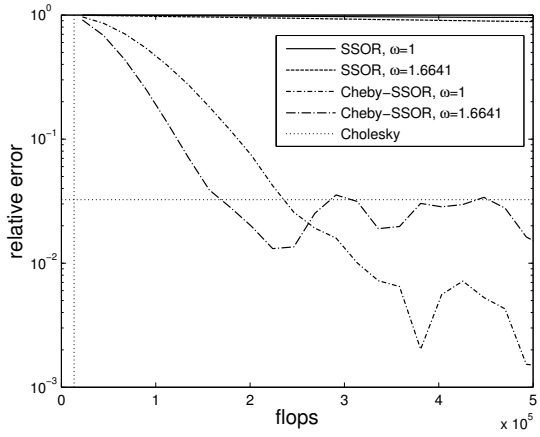


Fig. 2 Convergence of n-step covariance as a function of computational work, for plain and accelerated Gibbs samplers applied to $d = 100$ dimensional problem. The work and error for the Cholesky factorization is shown as dotted lines, for reference.

using a Cholesky factorization. However, the iterative structure of the Gibbs sampler is able to take advantage of the sparse precision matrix, which is the only special structure exploited here. (The Fourier transform is also applicable in this case because the GMRF is stationary.)

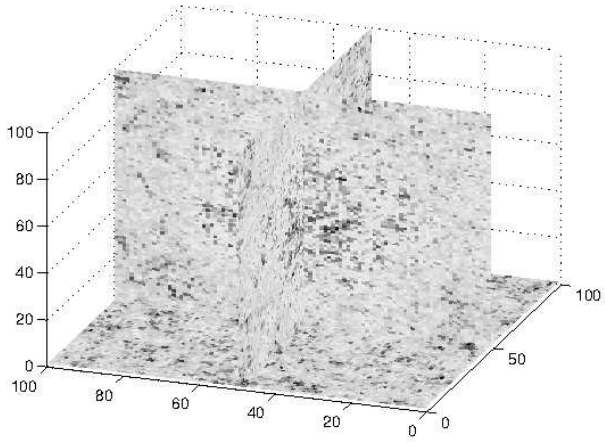


Fig. 3 Slices through a sample on the 3-dimensional lattice with $d = 100 \times 100 \times 100$.

5 A conjugate gradient sampling algorithm

The conjugate gradient (CG) optimization method may be viewed as a polynomial acceleration in which the optimal error polynomial is chosen by also calculating the eigenvalues of the iteration operator within the procedure. However, we present the method here by focusing on the mutually **A**-conjugate directions that are generated at each iteration.

Fig. 4 shows a schematic of the iterative structure implemented by Gauss-Seidel (left) and conjugate gradient optimization (right) of a quadratic function in $d = 2$ dimensions. The sequence of *search directions* is depicted by dashed lines. The Gauss-Seidel iteration performs optimization along each coordinate direction, in sequence. As we have seen, this implements exactly the same iteration structure as the Gibbs sampler, depicted by solid lines with the sequence of conditional samples denoted $\mathbf{x}^{(0)}$, $\mathbf{x}^{(1)}$, etc. In contrast the CG algorithm uses a sequence of search directions that are mutually **A**-conjugate, seeded by the gradient at each iterate, as depicted in the right panel of Fig. 4. By performing conditional sampling along this sequence of directions, as opposed to 1-dimensional optimization, we get the conjugate gradient sampler (solid lines).

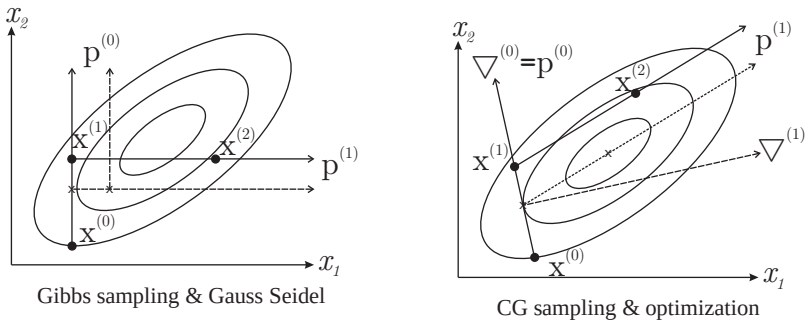


Fig. 4 Schematic in $d = 2$ dimensions depicting the path taken by the Gauss-Seidel iteration and Gibbs sampler (left) and the CG optimizer and sampler (right). Contours of the quadratic objective function and log-target density are also shown. Path of the optimizer shown in dotted lines, sampler shown in solid lines. Search directions are $\mathbf{p}^{(0)}, \mathbf{p}^{(1)}, \ldots$, iterates are $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots$, while $\nabla^{(0)}, \nabla^{(1)}, \ldots$ show the direction of gradients at iterates.

Mutually conjugate vectors (with respect to **A**) are independent directions for $\mathrm{N}(0, \mathbf{A}^{-1})$, since

$$\mathbf{V}^{\mathrm{T}}\mathbf{A}\mathbf{V} = \mathbf{D} \Rightarrow \mathbf{A}^{-1} = \mathbf{V}\mathbf{D}^{-1}\mathbf{V}^{\mathrm{T}}$$

where $\mathbf{V}$ has mutually conjugate columns and $\mathbf{D}$ is a diagonal matrix. Hence, if $\mathbf{z} \sim \mathrm{N}(0, I)$ then $\mathbf{x} = \mathbf{V}\sqrt{\mathbf{D}^{-1}}\mathbf{z} \sim \mathrm{N}(0, \mathbf{A}^{-1})$. Thus the problem of sampling from $\mathrm{N}(0, \mathbf{A}^{-1})$ is reduced to sampling from standard normal distributions. Both the Cholesky factorization and eigen-decomposition are examples of sets of mutually conjugate vectors [7].

Algorithm 1 (CD sampler producing $\mathbf{x} \sim \mathrm{N}(0, \mathbf{A}^{-1})$) *initialize* $\mathbf{x}$ *and* $\mathbf{b}$ *(*$\mathbf{Ax} \neq \mathbf{b}$*)*

1. $\mathbf{r} \leftarrow \mathbf{b} - \mathbf{Ax}$
2. $\mathbf{p} \leftarrow \mathbf{r}$
3. *for* $k = 1$ *to n do:*
4. $\quad \mathbf{q} \leftarrow \mathbf{Ap}$
5. $\quad$ *set* $d \leftarrow \mathbf{q}^{\mathrm{T}}\mathbf{p}$, $e \leftarrow \mathbf{q}^{\mathrm{T}}\mathbf{x}/d$, $f \leftarrow \mathbf{p}^{\mathrm{T}}\mathbf{b}/d$
6. $\quad$ *draw* $z \sim \mathrm{N}(0,1)$ *and set* $\alpha \leftarrow z/\sqrt{d}$
7. $\quad \mathbf{x} \leftarrow \mathbf{x} + (\alpha - e)\,\mathbf{p}$
8. $\quad \mathbf{b} \leftarrow \mathbf{b} + (\alpha - f)\,\mathbf{q}$
9. $\quad \mathbf{r} \leftarrow \mathbf{r} - (f - e)\mathbf{q}$
10. $\quad \mathbf{p} \leftarrow \mathbf{r} - \dfrac{\mathbf{r}^{\mathrm{T}}\mathbf{q}}{d}\mathbf{p}$

The sequential conjugate-direction algorithm given in Fox 2008 [7] is shown in Alg. 1. This algorithm operates locally, so can potentially be generalized to non-Gaussian targets. An earlier Krylov-space method was presented in Schneider & Willsky 2003 [20]. Ceriotti *et al.* [5] gave an algorithm that solves $\mathbf{Ax} = \mathbf{b}$ by standard linear CG and separately accumulates the sample $\mathbf{y}$. They mitigated problems associated with loss of conjugacy and degenerate eigenspaces by a combination of random restarts and orthogonalization over a small set of vectors. Parker and Fox 2012 [17] presented a convergence criterion based on the residual, also for an algorithm that solves $\mathbf{Ax} = \mathbf{b}$ by standard linear CG and separately accumulates the sample $\mathbf{y}$. They also established that, after k steps, $\mathrm{Var}(\mathbf{y}^k)$ is the CG polynomial, and gave following best-approximation property:

Theorem 3. *(Parker 2009)*
The covariance matrix

$$\mathrm{Var}(\mathbf{y}^k|\mathbf{x}^0, \mathbf{b}^0) = V_k T_k^{-1} V_k^T$$

has k non-zero eigenvalues which are the Lanczos estimates of the eigenvalues of $\mathbf{A}^{-1}$*. The eigenvectors of* $\mathrm{Var}(\mathbf{y}^k|\mathbf{x}^0, \mathbf{b}^0)$ *are the Ritz vectors* $V_k v^i$ *which estimate the eigenvectors of* $\mathbf{A}$*.*

That is, the k-step variance $\mathrm{Var}(\mathbf{y}^k|\mathbf{x}^0, \mathbf{b}^0)$ approximates $\mathbf{A}^{-1}$ in the eigenspaces corresponding to the extreme and well separated eigenvalues of $\mathbf{A}$.

Fig. 5 shows two samples drawn from a 100×100-dimensional Gaussian field with a second-order locally linear precision matrix. The left panel was drawn using the CG sampler of Parker and Fox 2012 [17], while the right sample was evaluated using the Cholesky factorization of $\mathbf{A}$. Loss of conjugacy in the CG algorithm means that the algorithm terminates before sampling all d-dimensions of the problem. For typical covariance functions, this results in over smooth samples as can be seen in the left panel of Fig. 5. However, the connection with iterative solvers immediately suggests the efficient solution which is to initialize the (accelerated) Gibbs sampler with the CG sample. This plays to the strengths of each method; the CG sampler efficiently calculates smooth structures in the Gaussian field, while relaxation techniques such as Gauss-Seidel (hence Gibbs sampling) are efficient in removing high-frequency errors.

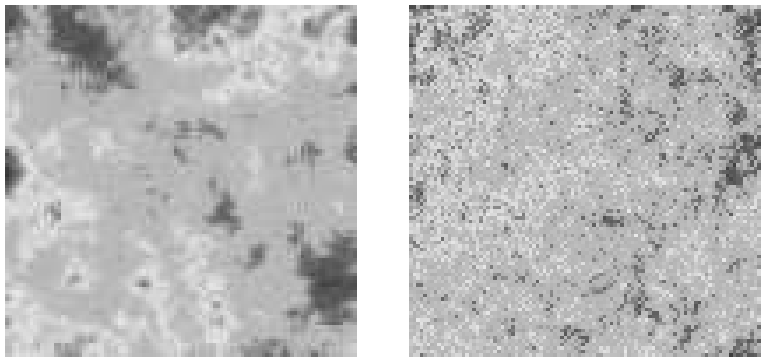


Fig. 5 A CG sample (left panel) $\mathbf{x} \sim \mathrm{N}(0, \mathbf{A}^{-1})$ from a 100×100-dimensional Gaussian field with a second-order locally linear precision matrix. The realized variance $\mathrm{Var}(\mathbf{x})$ accounts for 80% of the variability in $\mathbf{A}^{-1}$. A Cholesky sample is shown on the right panel.

6 Discussion

The motivating query of whether "stochastic relaxation" is formally equivalent to "relaxation" has been answered in the affirmative, in the Gaussian setting; Gibbs sampling is precisely equivalent to Gauss-Seidel iteration. This result generalizes to any splitting of the precision matrix, to give both a "stochastic relaxation" and a "relaxation" with identical conditions for convergence, rates of convergence, and error polynomial.

Hence, existing efficient solvers (multigrid, fast multipole, parallel tools) can all be used to perform sampling from Gaussian distributions; indeed, these 'best' solvers are necessarily the 'best' samplers for Gaussian distributions.

As was shown in section 2, polynomial acceleration may also be applied to the Markov chain that targets a non-Gaussian distribution. The example presented, while rather special and not of practical use, did demonstrate that polynomial acceleration of a geometrically convergent chain can lead to an algorithm that draws 'perfect' samples in finite compute time.

For general target distributions, Chebyshev acceleration of convergence in mean and covariance is also not limited to Gaussian targets. The requirement of explicitly knowing the precision matrix $\mathbf{A}$ may be circumvented by *adapting* to it [8]. Applications in the setting of diffusion tomography show good results, though no proof of convergence exists for the accelerated adaptive algorithm.

Acknowledgements Polynomial acceleration of Gibbs sampling is the brainchild of Al Parker, to whom I am indebted. This research was supported by Marsden grant 10-UOO-221.

References

1. Adler, S.L.: Over-relaxation method for the Monte Carlo evaluation of the partition function for multiquadratic actions. Physical Review D **23**(12), 2901–2904 (1981)
2. Amit, Y., Grenander, U.: Comparing sweep strategies for stochastic relaxation. Journal of Multivariate Analysis **37**, 197–222 (1991)
3. Axelsson, O.: Iterative Solution Methods. Cambridge University Press (1996)
4. Barone, P., Frigessi, A.: Improving stochastic relaxation for Gaussian random fields. Probability in the Engineering and Informational Sciences **4**, 369–384 (1990)
5. Ceriotti, M., Bussi, G., Parrinello, M.: Conjugate gradient heat bath for ill-conditioned actions. Physical Review E **76**, 026707-1–8 (2007)
6. Diaconis, P., Freedman, D.: Iterated random functions. SIAM Review **41**(1) 45–76 (1999)
7. Fox, C.: A conjugate direction sampler for normal distributions, with a few computed examples. Technical Reports from the Electronics Group, University of Otago (2008)
8. Fox, C., Parker, A.: Convergence in variance of first-order and second-order Chebyshev accelerated Gibbs samplers. SIAM Journal on Scientific Computing (to be published 2013)
9. Fox, C., Parker, A.: Gibbs sampling of normal distributions using matrix splittings and polynomial acceleration. In preparation (2013)
10. Galli, A., Gao, H.: Rate of convergence of the Gibbs sampler in the Gaussian case. Mathematical Geology **33**(6), 653–677 (2001)
11. Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Transactions Pattern Analysis and Machine Intelligence **6**, 721–741 (1984)
12. Glauber, R.: Time dependent statistics of the Ising model. Journal of Mathematical Physics **4**, 294–307 (1963)
13. Golub, G.H., Varga, R.S.: Chebyshev semi-iterative methods, successive over-relaxation iterative methods, and second-order Richardson iterative methods, parts I and II. Numerische Mathematik **3**, 147–156, 157–168 (1961)
14. Goodman, J., Sokal, A.D.: Multigrid Monte Carlo method. Conceptual foundations. Physical Review D **40**(6), 2035–2071 (1989)
15. Higdon, D.: A primer on space-time modelling from a Bayesian perspective. In: B. Finkenstadt, L. Held, V. Isham (eds.) Statistics of Spatio-Temporal Systems, pp. 217–279. Chapman & Hall/CRC, New York (2006)
16. Łatuszyński, K., Roberts, G.O.: CLTs and asymptotic variance of time-sampled Markov chains. Methodology and Computing in Applied Probability **15**(1), 237–247 (2013)
17. Parker, A., Fox, C.: Sampling Gaussian distributions in Krylov spaces with conjugate gradients. SIAM Journal on Scientific Computing **34**(3), B312–B334 (2012)
18. Roberts, G.O., Sahu, S.K.: Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler. Journal of the Royal Statistical Society: Series B **59**(2), 291–317 (1997)
19. Rue, H.: Fast sampling of Gaussian Markov random fields. Journal of the Royal Statistical Society: Series B **63**, 325–338 (2001)
20. Schneider, M.K., Willsky, A.S.: A Krylov subspace method for covariance approximation and simulation of random processes and fields. Multidimensional Systems and Signal Processing **14**, 295–318 (2003)
21. Turchin, V.F.: On the computation of multidimensional integrals by the Monte Carlo method. Theory of Probability and Its Applications **16**, 720–724 (1971)
22. Young, D.M.: Iterative Solution of Large Linear Systems. Academic Press (1971)