Supplementary Material: Accelerated Gibbs sampling of normal distributions using matrix splittings and polynomials

COLIN FOX[1] and ALBERT PARKER[2]

[1] *Department of Physics, University of Otago, Dunedin, New Zealand*
E-mail: `fox@physics.otago.ac.nz`

[2] *Center for Biofilm Engineering, Department of Mathematical Sciences, Montana State University, Bozeman, MT, USA E-mail:* `parker@math.montana.edu`

Implementation details of the Chebyshev accelerated sampler

The Chebyshev accelerated sampler requires estimates of the extreme eigenvalues of the operator $\mathbf{M}^{-1}\mathbf{A}$ that acts on the error in mean and variance of the samples. We obtain these estimates via a conjugate-gradient (CG) algorithm at no significant increase in computational cost (Meurant, 2006; Scales, 1989; Parker and Fox, 2012). The CG algorithm itself may be adapted to sample from normal distributions (Fox, 2008; Parker and Fox, 2012). Thus, in addition to generating eigenvalue estimates, the Chebyshev SSOR may also be further accelerated (significantly) by initializing with a CG sample that is approximately distributed as $\mathrm{N}(\mathbf{0}, \mathbf{A}^{-1})$, at no increase in computational cost. These ideas are discussed in this supplementary material, as well as explicit details for implementing the Chebyshev SSOR sampler.

Acceleration by CG

The CG algorithm is an algorithm that is direct ('perfect' in the language of statistics) in exact arithmetic (Nocedal and Wright, 2000). The CG solver and corresponding sampler are depicted in panels E and F of Figure 1. In the figure, the direct nature of CG is shown by convergence of the solver in a finite number of steps, and the sampler aligning with directions that are independent under the target distribution. Unlike Chebyshev acceleration, the CG algorithm does not correspond to acceleration by a fixed polynomial; instead, the polynomial depends on residuals, and is outside the scope of this paper. To further illustrate these ideas, for the small example ($n = 100$) over a 10×10 lattice in section 6.1, two implementations of a CG solver were applied to solve $\mathbf{Ax} = \mathbf{b}$: CG (un-preconditioned), and CG-SSOR (i.e., CG preconditioned by SSOR) (Table 3).

Acceleration by initialization and relaxation parameter

In addition to generating estimates of the eigenvalues of $\mathbf{M}^{-1}\mathbf{A}$ required by the Chebyshev accelerated SSOR sampler, the sampler may also be further accelerated (significantly) by initializing with a CG sample $\mathbf{y}^{(0)}$, at no increase in computational cost. Acceleration by initialization follows from the specification of the error reduction given in Corollary 6,

$$\mathbf{Var}(\mathbf{y}^{(k)}) = \mathbf{A}^{-1} - P_k(\mathbf{M}^{-1}\mathbf{A})(\mathbf{A}^{-1} - \mathbf{Var}(\mathbf{y}^{(0)}))P_k(\mathbf{M}^{-1}\mathbf{A})^T.$$

Thus, the sampler is accelerated, by a constant factor, when $\mathbf{Var}(\mathbf{y}^{(0)}) \approx \mathbf{A}^{-1}$, in particular when $(\mathbf{A}^{-1} - \mathbf{Var}(\mathbf{y}^{(0)}))$ lies in the eigenspaces of $P_k(\mathbf{M}^{-1}\mathbf{A})$ that correspond to small eigenvalues.

The SOR and SSOR samplers use a splitting $\mathbf{A} = \mathbf{M}_\omega - \mathbf{N}_\omega$, where we have included the subscript to stress the dependence on the relaxation parameter ω. These are accelerated Gibbs samplers when $\varrho(\mathbf{M}_\omega^{-1}\mathbf{N}_\omega) < \varrho(\mathbf{M}_{\text{GS}}^{-1}\mathbf{N}_{\text{GS}})$. By Corollary 3, the same value of ω that optimizes the convergence of the stationary linear solver (Table 3), $\omega^* = \text{argmin}_{0<\omega<2}\varrho(\mathbf{M}_\omega^{-1}\mathbf{N}_\omega)$, also optimizes the convergence of the stationary sampler (Figure 2B). Thus, choice of an optimal relaxation parameter achieves acceleration by reducing the convergence factor.

Chebyshev accelerated SSOR sampler

By application of Theorem 5 to the case of Chebyshev polynomials, we derived a Chebyshev accelerated SSOR sampler by iteratively updating parameters via equation (13) and then evaluating equation (17) (Fox and Parker, 2014). The implementation in Algorithm 1 follows the exposition due to (Axelsson, 1996).

References

Axelsson, O. (1996). *Iterative Solution Methods.* Cambridge University Press.

Fox, C. (2008). A conjugate direction sampler for normal distributions, with a few computed examples. Technical Report 2008-1, Electronics Group, University of Otago.

Fox, C. and A. Parker (2014). Convergence in variance of Chebyshev accelerated Gibbs samplers. *SIAM Journal on Scientific Computing 36*(1), A124–A147.

Meurant, G. (2006). *The Lanczos and Conjugate Gradient Algorithms.* Philadelphia: SIAM.

Nocedal, J. and S. J. Wright (2000). *Numerical Optimization.* Springer, New York.

Parker, A. and C. Fox (2012). Sampling Gaussian distributions in Krylov spaces with conjugate gradients. *SIAM Journal on Scientific Computing 34*(3), B312–B334.

Scales, J. A. (1989). On the use of conjugate gradient to calculate the eigenvalues and singular values of large, sparse matrices. *Geophysical Journal International 97*, 179–183.

Algorithm 1: Chebyshev accelerated SSOR sampler from $\mathrm{N}(\boldsymbol{\mu} = \mathbf{A}^{-1}\boldsymbol{\nu}, \mathbf{A}^{-1})$

```
input  : SSOR parameter $\omega : 0 < \omega < 2$; the vector $\nu$ so that $\boldsymbol{\mu} = \mathbf{A}^{-1}\boldsymbol{\nu}$; SOR splitting
         $\mathbf{A} = \mathbf{M}_\omega - \mathbf{N}_\omega$; extreme eigenvalues $0 < \lambda_{\min} < \lambda_{\max}$ of $\mathbf{M}_{\mathrm{SSOR}}^{-1} A$; initial state $\mathbf{y}^{(0)}$;
         maximum iteration $k_{\max}$
output: $\mathbf{y}^{(k_{\max}+1)}$ approximately distributed as $\mathrm{N}(0, \mathbf{A}^{-1})$
Set $\mathbf{D}_\omega^{1/2} = ((2/\omega - 1)\,\mathrm{diag}(\mathbf{A}))^{1/2}$, $\delta = ((\lambda_{\max} - \lambda_{\min})/4)^2$, $\tau = 2/(\lambda_{\max} + \lambda_{\min})$;
$\beta = 2\tau$;
$\alpha = 1$;
$b = 2/\alpha - 1$;
$a = (2/\tau - 1)\,b$;
$\kappa = \tau$;
for $k = 0, \dots, k_{\max}$ do
    sample $\mathbf{z} \sim \mathrm{N}(\boldsymbol{\nu}, \mathbf{I})$;
    $\mathbf{c} = b^{1/2}\mathbf{D}_\omega^{1/2}\mathbf{z}$;
    $\mathbf{x} = \mathbf{y}^{(k)} + \mathbf{M}_\omega^{-1}(\mathbf{c} - \mathbf{A}\mathbf{y}^{(k)})$;
    sample $\mathbf{z} \sim \mathrm{N}(\nu, \mathbf{I})$;
    $\mathbf{c} = a^{1/2}\mathbf{D}_\omega^{1/2}\mathbf{z}$;
    $\mathbf{w} = \mathbf{x} - \mathbf{y}^{(k)} + \mathbf{M}_\omega^{-T}(\mathbf{c} - \mathbf{A}\mathbf{x})$;
    if $k = 0$ then
        $\mathbf{y}^{(k+1)} = \alpha(\mathbf{y}^{(k)} + \tau\mathbf{w})$;
    else
        $\mathbf{y}^{(k+1)} = \alpha(\mathbf{y}^{(k)} - \mathbf{y}^{(k-1)} + \tau\mathbf{w}) + \mathbf{y}^{(k-1)}$;
    end
    $\beta = (1/\tau - \beta\delta)^{-1}$;
    $\alpha = \beta/\tau$;
    $b = 2\kappa(1 - \alpha)/\beta + 1$;
    $a = (2/\tau - 1) + (b - 1)\,(1/\tau + 1/\kappa - 1)$;
    $\kappa = \beta + (1 - \alpha)\kappa$;
end
```