A limited-memory acceleration strategy for mcmc sampling in hierarchical bayesian calibration of hydrological models

G. Kuczera, D. Kavetski, Benjamin Renard, M. Thyer

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

A LIMITED-MEMORY ACCELERATION STRATEGY FOR MCMC SAMPLING IN HIERARCHICAL BAYESIAN CALIBRATION OF HYDROLOGICAL MODELS

George Kuczera[1], Dmitri Kavetski[1], Benjamin Renard[2] and Mark Thyer[1]

1 School of Engineering, University of Newcastle, Callaghan, NSW, 2308, Australia

2 Cemagref, UR HHLY, Hydrology-Hydraulics. 3 bis quai Chauveau - CP 220, F-69336 Lyon, France

Abstract

Hydrological calibration and prediction using conceptual models is affected by forcing/response data uncertainty and structural model error. The Bayesian Total Error Analysis methodology (BATEA) uses a hierarchical framework to represent individual sources of uncertainty. However, it is shown that standard multi-block "Metropolis-within-Gibbs" samplers commonly used in traditional Bayesian hierarchical Markov Chain Monte Carlo (MCMC) are exceedingly computationally expensive when applied to hydrologic models based on recursive numerical solution of coupled nonlinear differential equations describing the evolution of catchment states such as soil and groundwater storages. This note develops a "limited-memory" algorithm for accelerating multi-block MCMC sampling from the posterior distributions of such models using low-dimensional jump distributions. The new algorithm exploits the decaying memory of hydrological systems to provide accurate tolerance-based approximations of traditional "full-memory" MCMC methods and is orders of magnitude more efficient than the latter.

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

Introduction

Characterizing uncertainties in streamflow predicted using conceptual rainfall-runoff (CRR) models is a key research and operational challenge [e.g., *Clark et al.*, 2008; *Vrugt et al.*, 2005]. Bayesian Total Error Analysis (BATEA) explicitly characterizes forcing, response and structural errors using a hierarchical formulation [*Kavetski et al.*, 2006; *Kuczera et al.*, 2006], which generally results in high-dimensional posterior distributions with hundreds or more latent variables. More generally, very high-dimensional hierarchical models have been reported in hydrology and elsewhere [*Cressie et al.*, 2009; *Reichert and Mieleitner*, 2009].

Hierarchical inferences are usually implemented using Gibbs sampling, or, more generally, multi-block Markov Chain Monte Carlo (MCMC). However, many statistical applications and software [e.g., BUGS, *Gilks et al.*, 1994] are not well suited to hydrologic modeling, which generally requires inference in coupled nonlinear differential equations [e.g., *Kavetski et al.*, 2003]. For such models, hierarchical (e.g., input-error sensitive) Bayesian inference using standard multi-block MCMC (e.g., "Metropolis-within-Gibbs") is computationally expensive even for moderate calibration data lengths (e.g., a few years of daily data).

This note shows why standard multi-block MCMC samplers are inefficient for full Bayesian CRR inference and presents a general solution strategy. Following an outline of BATEA, we detail multi-block samplers with an emphasis on their computational cost given the recursive time stepping nature of CRR models. A more efficient "limited-memory" MCMC algorithm is then designed and illustrated using the common GR4J model. We conclude with a comment on a hybrid strategy for efficient MCMC-based Bayesian hierarchical inference of CRR models.

1 Outline of the BATEA framework

1.1 Data uncertainty

Consider a time series of length N_t, $\boldsymbol{X} = \left\{ X_{(m)}; m = 1,..,N_t \right\}$, where $X_{(m)}$ is the forcing at the

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

*m*th time step. Next, consider N epochs $\{(m_i, m_{i+1}-1); i=1,..,N\}$, where m_i is the time step index of the start of the i^{th} epoch (e.g., storm or daily [*Thyer et al.*, 2009]). The observed forcing is $\tilde{\boldsymbol{X}}_i = \{\tilde{X}_{(m)}; m = m_i,..,m_{i+1}-1\}$ and the true forcing is $\boldsymbol{X}_i$, while $\tilde{\boldsymbol{Y}}_i$ and $\boldsymbol{Y}_i$ are, respectively, the observed and true responses.

Let a function $\boldsymbol{X}_i = I(\tilde{\boldsymbol{X}}_i \mid \varphi_i)$ relate actual and observed forcings, e.g., $\boldsymbol{X}_i = \varphi_i \tilde{\boldsymbol{X}}_i$ for all steps of the i^{th} storm event [*Kavetski et al.*, 2006], where φ_i is a storm-dependent multiplicative error, treated as a latent variable with "hyper-distribution"

$$\varphi_i \sim p(\varphi \mid \boldsymbol{\Phi}) \tag{1}$$

where the "hyper-parameters" $\boldsymbol{\Phi}$ describe, e.g., the mean and variance of φ.

A streamflow error model must also be specified [*Thyer et al.*, 2009],

$$\tilde{\boldsymbol{Y}}_i \sim p(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\Xi}) \tag{2}$$

where $\boldsymbol{\Xi}$ characterize response errors (e.g., variance of rating curve errors).

1.2 CRR models and their recursive structure

The CRR model $\boldsymbol{H}()$ maps the forcings into simulated responses $\hat{\boldsymbol{Y}}_i$. The majority of CRR models are based on numerical solutions of initial-value differential equations (DEs) describing time changes in conceptual stores $\boldsymbol{S}$ such as groundwater, soil and stream, connected via hypothesized fluxes $\boldsymbol{g}()$ [e.g., *Kavetski et al.*, 2003]

$$\begin{aligned} &\hat{\boldsymbol{Y}}_i = \boldsymbol{H}(\boldsymbol{X}_{1:i}, \boldsymbol{\lambda}_{1:i}, \boldsymbol{\theta}) && \text{(a)} \\ &\frac{\mathrm{d}\boldsymbol{S}}{\mathrm{d}t} = \boldsymbol{g}(\boldsymbol{X}, \boldsymbol{\lambda}, \boldsymbol{\theta}, \boldsymbol{S}) \Rightarrow \boldsymbol{S}_{i+1} = \boldsymbol{f}(\boldsymbol{X}_i, \boldsymbol{\lambda}_i, \boldsymbol{\theta}, \boldsymbol{S}_i) && \text{(b)} \\ &\hat{\boldsymbol{Y}}_{i+1} = \boldsymbol{h}(\boldsymbol{X}_i, \boldsymbol{\lambda}_i, \boldsymbol{\theta}, \boldsymbol{S}_i) && \text{(c)} \end{aligned} \tag{3}$$

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

Eqn (3) is formulated deterministically to conserve mass in each store [*Kuczera et al.*, 2006]. Here, $\boldsymbol{\theta}$ are the time-invariant CRR parameters and $\boldsymbol{\lambda}_i$ are epoch-specific CRR parameters,

$$\boldsymbol{\lambda}_i \sim p\left(\boldsymbol{\lambda} \mid \boldsymbol{\Lambda}\right) \tag{4}$$

where $\boldsymbol{\Lambda}$ are the CRR hyper-parameters, e.g., means and variances of stochastic parameters.

A key feature of virtually all CRR models is their recursive structure illustrated in eqn (3). When applied to such models, BATEA is atypical of standard Bayesian hierarchical formulations [e.g., *Gelman et al.*, 2004; *Gilks et al.*, 1994] because the simulated response $\hat{\boldsymbol{Y}}_i$ depends on earlier epochs. For example, effects of a large rainfall error will persist because the induced storage errors affect streamflow over many subsequent steps.

1.3 BATEA posterior distribution

BATEA infers the CRR parameters $\boldsymbol{\theta}$, latent variables $\{\boldsymbol{\varphi}, \boldsymbol{\lambda}\}$ and hyper-parameters $\{\boldsymbol{\Phi}, \boldsymbol{\Lambda}, \boldsymbol{\Xi}\}$ given observed forcing-response data $\{\tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}\}$ and any prior information. To simplify the notation, define the complete set of N epoch-dependent latent variables $\boldsymbol{\omega}_{1:N} = \{\boldsymbol{\varphi}_{1:N}, \boldsymbol{\lambda}_{1:N}\}$, and the corresponding hyper-parameters $\boldsymbol{\Omega} = \{\boldsymbol{\Phi}, \boldsymbol{\Lambda}\}$. The BATEA posterior pdf is then

$$p\left(\boldsymbol{\theta}, \boldsymbol{\omega}_{1:N}, \boldsymbol{\Omega}, \boldsymbol{\Xi} \mid \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}\right) \propto p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\theta}, \boldsymbol{\omega}_{1:N}, \boldsymbol{\Xi}, \tilde{\boldsymbol{X}}\right) p\left(\boldsymbol{\omega}_{1:N} \mid \boldsymbol{\Omega}\right) p\left(\boldsymbol{\Omega}\right) p\left(\boldsymbol{\Xi}\right) p\left(\boldsymbol{\theta}\right) \tag{5}$$

where $p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\theta}, \boldsymbol{\omega}_{1:N}, \boldsymbol{\Xi}, \tilde{\boldsymbol{X}}\right)$ is the likelihood function, $p\left(\boldsymbol{\omega}_{1:N} \mid \boldsymbol{\Omega}\right)$ is the hyper-distribution of $\boldsymbol{\omega}_{1:N}$, and $p(\boldsymbol{\Omega})$, $p(\boldsymbol{\Xi})$ and $p(\boldsymbol{\theta})$ are priors [*Kuczera et al.*, 2006].

2 MCMC methods for hierarchical Bayesian inference

2.1 General Metropolis-Hastings sampler

The Metropolis-Hastings (MH) algorithm is a general MCMC method for sampling from multivariate distributions [*Gelman et al.*, 2004]. If $p(\boldsymbol{\psi})$ is the target distribution, e.g.,

posterior (5), the MH method samples a proposal $\boldsymbol{\psi}^{*(k+1)}$ from a jump distribution $J\left(\boldsymbol{\psi} \mid \boldsymbol{\psi}^{(k)}\right)$ at the *k*th iteration. It accepts $\boldsymbol{\psi}^{(k+1)} = \boldsymbol{\psi}^{*(k+1)}$ with probability given by the jump ratio $r\left(\boldsymbol{\psi}^{*(k+1)} \mid \boldsymbol{\psi}^{(k)}\right)$ below, otherwise $\boldsymbol{\psi}^{(k+1)} = \boldsymbol{\psi}^{(k)}$,

$$r\left(\boldsymbol{\psi}^{*(k+1)} \mid \boldsymbol{\psi}^{(k)}\right) = \frac{p\left(\boldsymbol{\psi}^{*(k+1)}\right) J\left(\boldsymbol{\psi}^{(k)} \mid \boldsymbol{\psi}^{*(k+1)}\right)}{p\left(\boldsymbol{\psi}^{(k)}\right) J\left(\boldsymbol{\psi}^{*(k+1)} \mid \boldsymbol{\psi}^{(k)}\right)} \tag{6}$$

When the jump distribution *J*() is symmetric (e.g., Gaussian centered on the current sample), the MCMC algorithm is referred to as a "Metropolis" scheme [*Gelman et al.*, 2004].

2.2 Multi-block MCMC sampler

The blocking of sampled variables considerably affects the efficiency of MCMC sampling [e.g., *Fu and Gomez-Hernandez*, 2009]. Sampling inferred quantities "all-at-once" leads to "single-block" schemes. However, since deriving and adapting efficient jump distributions for high-dimensional posteriors such as (5) is challenging, Bayesian literature and software tend to favor multi-block schemes using a sequence of low-dimensional jump distributions [*Gelman et al.*, 2004; *Gilks et al.*, 1994]. For BATEA, the following three-block sampler is natural:

Block 1: Sample the hyper-parameters $\boldsymbol{\Omega}^{(k+1)}$ from their conditional posterior

$$p\left(\boldsymbol{\Omega} \mid \boldsymbol{\omega}_{1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}\right) \propto p\left(\boldsymbol{\omega}_{1:N}^{(k)} \mid \boldsymbol{\Omega}\right) p\left(\boldsymbol{\Omega}\right) \tag{7}$$

where the simplification occurs due to the hierarchical structure of (5).

When conjugate hyper-distributions and priors are used, conditional posterior (7) can be sampled analytically ("Gibbs sampling") [*Gelman et al.*, 2004]. More generally, however, a Metropolis acceptance-rejection step (Section 2.1) is used to sample from pdf (7).

Block 2: Sample the latent variables $\boldsymbol{\omega}_{1:N}^{(k+1)}$ from their conditional posterior

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

$$p\left(\boldsymbol{\omega}_{1:N} \mid \boldsymbol{\theta}^{(k)}, \boldsymbol{\Omega}^{(k+1)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}\right) \propto p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\theta}^{(k)}, \boldsymbol{\omega}_{1:N}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}\right) p\left(\boldsymbol{\omega}_{1:N} \mid \boldsymbol{\Omega}^{(k+1)}\right) \tag{8}$$

The implementation of this step lies at the focus of this note and is detailed in the next section.

Block 3: Sample the time-invariant CRR parameters and the output error parameters $\left(\boldsymbol{\theta}^{(k+1)}, \boldsymbol{\Xi}^{(k+1)}\right)$ from their conditional posterior

$$p\left(\boldsymbol{\theta}, \boldsymbol{\Xi} \mid \boldsymbol{\omega}_{1:N}^{(k+1)}, \boldsymbol{\Omega}^{(k+1)}, \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}\right) \propto p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\theta}, \boldsymbol{\omega}_{1:N}^{(k+1)}, \boldsymbol{\Xi}, \tilde{\boldsymbol{X}}\right) p\left(\boldsymbol{\theta}\right) p\left(\boldsymbol{\Xi}\right) \tag{9}$$

Again, while conjugate probability models permit direct Gibbs sampling, in general sampling from (9) is implemented using a Metropolis iteration.

2.3 Epoch-by-epoch MCMC sampler

Since direct sampling of $\boldsymbol{\omega}_{1:N}^{(k+1)}$ from its conditional posterior in Block 2 is usually impossible, MH sampling is used within this block. The temporal structure of latent variables $\boldsymbol{\omega}$ suggests epoch-by-epoch sampling. For the j^{th} epoch within the k^{th} iteration, we sample $\boldsymbol{\omega}_j^{(k+1)}$ from the conditional posterior $p\left(\boldsymbol{\omega}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Omega}^{(k+1)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}\right)$, with the MH jump ratio

$$\begin{aligned} r\left(\boldsymbol{\omega}_j^{*(k+1)} \middle| \boldsymbol{\omega}_j^{(k)}\right) &= \frac{p\left(\boldsymbol{\omega}_j^{*(k+1)} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Omega}^{(k+1)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{Y}}, \tilde{\boldsymbol{X}}\right)}{p\left(\boldsymbol{\omega}_j^{(k)} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Omega}^{(k+1)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{Y}}, \tilde{\boldsymbol{X}}\right)} \times \frac{J\left(\boldsymbol{\omega}_j^{(k)} \middle| \boldsymbol{\omega}_j^{*(k+1)}\right)}{J\left(\boldsymbol{\omega}_j^{*(k+1)} \middle| \boldsymbol{\omega}_j^{(k)}\right)} \\ &= \frac{p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}\right)}{p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}\right)} \times \frac{p\left(\boldsymbol{\omega}_j^{*(k+1)} \mid \boldsymbol{\Omega}^{(k+1)}\right)}{p\left(\boldsymbol{\omega}_j^{(k)} \mid \boldsymbol{\Omega}^{(k+1)}\right)} \times \frac{J\left(\boldsymbol{\omega}_j^{(k)} \middle| \boldsymbol{\omega}_j^{*(k+1)}\right)}{J\left(\boldsymbol{\omega}_j^{*(k+1)} \middle| \boldsymbol{\omega}_j^{(k)}\right)} \end{aligned} \tag{10}$$

Such algorithms are often referred to as "Metropolis-within-Gibbs" [e.g., *Reichert and Mieleitner*, 2009; *Roberts and Rosenthal*, 2009].

2.4 The likelihood ratio: A computational bottleneck

The evaluation of the likelihood ratio in (10),

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

$$\rho_j^{(k+1)} = \frac{p(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}})}{p(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+1:N}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}})} \tag{11}$$

dominates the CPU cost of MCMC methods for physically-motivated models $H()$ commonly used in environmental engineering contexts, because it requires the numerical solution of the (usually coupled nonlinear) differential equations underlying the model $\boldsymbol{H}()$, which is far costlier than evaluating the hyper-distributions [e.g., *Fu and Gomez-Hernandez,* 2009].

Given (2) and (3), ratio (11) can be expanded with respect to the epoch index j as

$$\begin{aligned}
\rho_j^{(k+1)} &= \prod_{i=1}^{N} \frac{q(\tilde{\boldsymbol{Y}}_i \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+1:N}^{(k)})}{q(\tilde{\boldsymbol{Y}}_i \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+1:N}^{(k)})} \\
&= \underbrace{\prod_{i=1}^{j-1} \frac{q(\tilde{\boldsymbol{Y}}_i \mid \boldsymbol{\omega}_{1:i}^{(k+1)})}{q(\tilde{\boldsymbol{Y}}_i \mid \boldsymbol{\omega}_{1:i}^{(k+1)})}}_{\textit{past relative to } j} \times \underbrace{\frac{q(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)})}{q(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)})}}_{\textit{present } j} \times \underbrace{\prod_{i=1}^{N-j} \frac{q(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+i:N}^{(k)})}{q(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+i:N}^{(k)})}}_{\textit{future relative to } j} \\
&= \frac{q(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)})}{q(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)})} \times \prod_{i=1}^{N-j} \frac{q(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+i:N}^{(k)})}{q(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+i:N}^{(k)})}
\end{aligned} \tag{12}$$

where for convenience $q\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\omega}_{1:j-1}, \boldsymbol{\omega}_j, \boldsymbol{\omega}_{j+1:N}\right) = p\left(\tilde{\boldsymbol{Y}} \mid \boldsymbol{\omega}_{1:j-1}, \boldsymbol{\omega}_j, \boldsymbol{\omega}_{j+1:N}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\Xi}^{(k)}, \tilde{\boldsymbol{X}}\right)$.

The "past relative to j" term drops out because epochs 1:j-1 are causally independent from $\boldsymbol{\omega}_j$. However, the last term in (12) does not cancel out because changes in storages propagate into future epochs due to the recursive model structure (3),

$$\boldsymbol{s}_{j+1}^{*(k+1)} := f\left(\tilde{\boldsymbol{X}}_j, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{s}_j\right) \neq \boldsymbol{s}_{j+1}^{(k)} := f\left(\tilde{\boldsymbol{X}}_j, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{s}_j\right) \tag{13}$$

These memory effects critically impact on the computational efficiency of multi-block MCMC.

2.5 Implications for hydrological modeling

Evaluation of (12) requires N–j+1 epoch evaluations. Since the number of epochs is roughly proportional to the number of time steps, $N \propto N_t$, the computational cost for a fixed number of multi-block MCMC samples is $O(N_t^2)$, i.e., is approximately quadratic (Figure 1). Standard

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

multi-block MCMC hence quickly becomes very expensive even for moderate-length calibrations. Even the GR4J model [*Perrin et al.*, 2003], used operationally in French forecasting systems, with typical individual runtimes below 1 sec for a few years of data, would require days or weeks to calibrate using input-error sensitive approaches. For distributed models, the cost is even more staggering. E.g., a single SWAT model run simulating soil moisture in irrigated landscapes may require several minutes in operational settings [*Tolson and Shoemaker*, 2007]. Expected runtimes of standard multi-block MCMC analysis could then exceed months! Given the growing interest in using CRR models to gain insights into catchment dynamics and structural model errors [*Clark et al.*, 2008], which requires calibration of multiple model configurations in multiple catchments while accounting for input errors, such computational burden is a serious practical impediment.

2.6 A limited-memory MCMC sampler

Reducing the computational cost of the inference requires addressing the storage memory issue. Simply ignoring it cuts the computational cost up to by a factor of N:

$$\rho_j^{(k+1)} = \frac{q\left(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}\right)}{q\left(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}\right)} \tag{14}$$

However, applying (14) to a recursive model (3) can seriously alter its posterior distribution, degrading the quality of the inference. Fortunately, a much better alternative is possible.

Note that the memory effect (13) decays over time because the CRR model "forgets" differences in the initial conditions at the j^{th} epoch expressed by (13). The likelihood ratio can then be approximated by marching forward from epoch j and terminating after $M_j^{(k+1)}(\tau) << N$ epochs, when the likelihood ratio converges to within a pre-specified numerical tolerance τ.

$$\rho_j^{(k+1)}(\tau) = \frac{q\left(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}\right)}{q\left(\tilde{\boldsymbol{Y}}_j \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}\right)} \times \prod_{i=1}^{M_j^{(k+1)}(\tau)} \frac{q\left(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+1:j+i}^{(k)}\right)}{q\left(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+1:j+i}^{(k)}\right)} \tag{15}$$

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

$$M_j^{(k+1)}(\tau) := \min[i] \text{ such that } \left| \log\left(\frac{q\left(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{*(k+1)}, \boldsymbol{\omega}_{j+1:j+i}^{(k)}\right)}{q\left(\tilde{\boldsymbol{Y}}_{j+i} \mid \boldsymbol{\omega}_{1:j-1}^{(k+1)}, \boldsymbol{\omega}_j^{(k)}, \boldsymbol{\omega}_{j+1:j+i}^{(k)}\right)} \right) \right| \leq \tau \quad (16)$$

This approach exploits the decaying memory of CRR models and any other model based on (stable) initial-value DEs such as (3): the influence of initial conditions vanishes over time.

We refer to algorithm (15) as the "limited-memory" MCMC sampler. The naming is inspired by "limited-memory" quasi-Newton methods for large-scale optimization [*Nocedal and Wright*, 1999], which also exploit the decaying memory of convergent recursive relations.

3 Empirical assessment

3.1 Experimental setup

We now compare four MCMC samplers for CRR model inference, including three multi-block samplers differing in model memory treatment: (i) "full-memory" (12); (ii) A "no-memory" (14); and (iii) limited-memory (15) with $\tau = 10^{-3}$. For consistency, all multi-block schemes sample one variable at a time using univariate Gaussian jump pdfs. A single-block Metropolis with a multivariate Gaussian jump pdf is used to independently confirm the accuracy of the samplers and to motivate an efficient hybrid MCMC strategy. Since the single-block Metropolis was pre-tuned over a series of trial runs, its practical computational cost is notably higher than may appear solely from the reported CPU time for generating the output samples.

Note that the important topic of adaption of jump distributions [e.g., *Roberts and Rosenthal*, 2009] lies largely outside the scope of this technical note, which focuses strictly on accelerating the evaluation of the jump ratio. The multi-block samplers were tuned based on jump rates [*Gelman et al.*, 2004], while the single-block Metropolis was pre-optimized using the results of the multi-block sampler. The same statistical models and assumptions were employed in all MCMC methods, ensuring that differences between the multi-block samplers were dominated by the treatment of model memory. Note that while all methods converge to the target posterior

(5), for a finite number of samples they inevitably exhibit minor discrepancies due to: (i) different autocorrelation structure of "epoch-by-epoch" versus "all-at-once" sampling, and (ii) histogram smoothing to estimate the underlying probability densities.

The accuracy and efficiency of the samplers were stringently verified using a synthetic case study. The "true" inputs comprised 6 years of observed daily rainfall and potential evapotranspiration for the 144 km^2 Yzeron catchment (France). The GR4J model [*Perrin et al.*, 2003] simulated the "true" daily streamflow using known "true" parameters. The "observed" streamflow was corrupted with 10% heteroscedastic Gaussian errors, while the "observed" rainfall was corrupted with log-normal multiplicative errors, $\log_e \varphi \sim N(0.0, 0.25^2)$.

3.2 Results and discussion

Figure 1 reports the CPU time to generate 10,000 MCMC samples and its dependence on the calibration data length, while Figure 2 shows the posterior distributions estimated from 1 year of data (86 epochs). Figure 1 lucidly illustrates the rapid CPU cost growth of the full-memory sampler with increasing calibration periods. As expected from algorithmic considerations (section 2.5), CPU time increases approximately quadratically with N_t, prohibiting the use of long calibration datasets. However, while the no-memory algorithm drastically cuts the CPU time, it provides a very poor approximation of the actual posterior. Even allowing a 1-epoch memory (M = 1 in eqn (15)) results in a significantly mis-specified mode and a markedly overestimated posterior uncertainty, while the no-memory approximation was off-the-chart. This confirms that uncontrollably modifying the model to discard its history is unacceptable.

In contrast, the limited-memory algorithm provides a very close approximation to the distributions obtained using the full-memory and single-block Metropolis schemes. This confirms the robustness of the convergence test (15), which ensures that the jump ratio of the limited-memory algorithm is within a tolerance of the jump ratio of the full-memory method.

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

Since in practice MCMC methods are seldom run to perfect convergence (nor is this even feasible in most cases), the discrepancies in Figure 2 are within MCMC sampling variability and other approximation errors. Importantly, tightening the tolerance τ forces a progressively closer agreement between the limited-memory method and its full-memory counterpart.

The CPU cost of the limited-memory sampler is near-linear with respect to calibration length. In particular, it was only 2-4 times slower than the no-memory sampler. In general, the computational acceleration of the limited-memory approximation depends on the calibration data and its epochs, the catchment response time, the CRR model, and the limited-memory tolerance τ. In this study, Figure 1 suggests an acceleration by a factor of 20 for the GR4J model applied to 6 years of daily data (463 epochs) with memory tolerance $\tau = 10^{-3}$.

Finally, the single-block ("all-at-once") sampler with *pre-tuned* jump distributions is generally more efficient than multi-block schemes because it requires only a single CRR model run per sample. However, in the absence of tuning it can be very inefficient and slowly convergent because a poorly selected high-dimensional jump distribution can lead to particularly poor mixing of the MCMC chains [e.g., see *Fu and Gomez-Hernandez*, 2009 for an analysis of the effect of block-size on MCMC convergence]. Moreover, adapting a high-dimensional jump distribution creates a considerable overhead not reported in this technical note because it is case-specific and depends on the MCMC initialization and adaption strategies.

Given the difficulty in tuning high-dimensional jump distributions, a hybrid MCMC strategy that exploits the limited-memory multi-block sampler to estimate a good jump distribution for a single-block Metropolis sampler can be advantageous. Since the multi-block sampler uses simple univariate Gaussian distributions in all blocks, their variances can be readily estimated and tuned. Once sufficient samples have been obtained, the entire covariance matrix can be estimated and kept fixed in a single-block Metropolis sampler. The design and evaluation of the hybrid MCMC strategy will be detailed in a separate study.

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

4 Concluding remarks

Hierarchical methods such as BATEA hold considerable promise for environmental modeling [see *Cressie et al.*, 2009, for a state-of-the-art discussion]. However, standard multi-block MCMC samplers (e.g., Metropolis-within-Gibbs) commonly used in the Bayesian hierarchical literature are computationally infeasible for recursive hydrological models simulating time-evolving storages, e.g., soil and groundwater. A careful "limited-memory" implementation of the jump ratio in the multi-block MCMC algorithm, exploiting the decaying memory of hydrological systems, overcomes the computational inefficiency, while controlling the accuracy using a numerical tolerance. We stress the broad applicability of the limited-memory acceleration strategy detailed in this note: it can be exploited by other hierarchical Bayesian MCMC formulations [*Cressie et al.*, 2009], and, more generally, it can be used for other computationally expensive recursive models with decaying memory.

References

Clark, M. P., A. G. Slater, D. E. Rupp, R. A. Woods, J. A. Vrugt, H. V. Gupta, T. Wagener, and L. E. Hay (2008), Framework for Understanding Structural Errors (FUSE): A modular framework to diagnose differences between hydrological models, *Water Resources Res*, *44*.

Cressie, N., C. A. Calder, J. S. Clark, J. M. ver Hoef, and C. K. Wikle (2009), Accounting for uncertainty in ecological analysis: the strengths and limitations of hierarchical statistical modeling, *Ecological Applications*, *19*(3), 553-570.

Fu, J., and J. J. Gomez-Hernandez (2009), A blocking Markov Chain Monte Carlo method for inverse stochastic hydrogeological modeling, *Math Geosci*, *41*, 105-128.

Gelman, A., J. B. Carlin, H. S. Stern and D. B. Rubin (2004) *Bayesian data analysis*, Chapman.

Gilks, W. R., A. Thomas, and D. J. Spiegelhalter (1994), A Language and Program for Complex Bayesian Modeling, *Statistician*, *43*(1), 169-177.

262 Kavetski, D., G. Kuczera, and S. W. Franks (2003), Semidistributed hydrological modeling: A
263 "saturation path" perspective on TOPMODEL and VIC, *Water Resources Research*, *39*(9).

264 Kavetski, D., G. Kuczera, and S. W. Franks (2006), Bayesian analysis of input uncertainty in
265 hydrological modeling: 1. Theory, *Water Resources Research*, *42*(3).

266 Kuczera, G., D. Kavetski, S. Franks, and M. Thyer (2006), Towards a Bayesian total error
267 analysis of conceptual rainfall-runoff models: Characterising model error using storm-
268 dependent parameters, *Journal of Hydrology*, *331*(1-2), 161-177.

269 Perrin, C., C. Michel, and V. Andreassian (2003), Improvement of a parsimonious model for
270 streamflow simulation, *Journal of Hydrology*, *279*(1-4), 275-289.

271 Reichert, P., and J. Mieleitner (2009), Analyzing input and structural uncertainty of nonlinear
272 dynamic models with stochastic, time-dependent parameters, *Water Resources Research*, *45*.

273 Roberts, G. O., and J. S. Rosenthal (2009), Examples of adaptive MCMC, *Journal of*
274 *Computational and Graphical Statistics*, *18*(2), 349-367.

275 Thyer, M., B. Renard, D. Kavetski, G. Kuczera, S. Franks, and S. Srikanthan (2009), Critical
276 evaluation of parameter consistency and predictive uncertainty in hydrological modelling: a
277 case study using bayesian total error analysis, *Water Resources Research*, *45*.

278 Tolson, B. A., and C. A. Shoemaker (2007), Dynamically dimensioned search algorithm for
279 computationally efficient watershed model calibration, *Water Resources Research*, *43*(1).

280 Vrugt, J. A., C. G. H. Diks, H. V. Gupta, W. Bouten, and J. M. Verstraten (2005), Improved
281 treatment of uncertainty in hydrologic modeling: Combining the strengths of global
282 optimization and data assimilation, *Water Resources Research*, *41*(1).

283

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

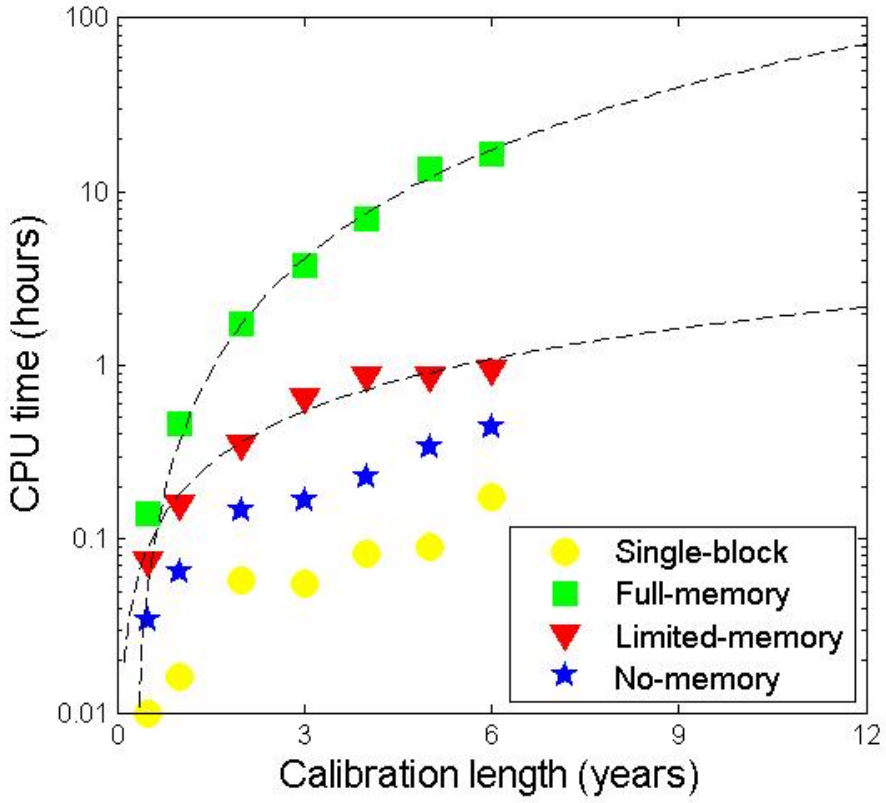

Figure 2. CPU time to generate 10,000 MCMC samples from the BATEA posterior of GR4J, as a function of the calibration data length. A 2.0 GHz laptop CPU with 1 GB of RAM was used.

Author-produced version of the article published in Water Resources Research, 2010, vol.46.
The original publication is available at http://www.agu.org, doi : 10.1029/2009WR008985

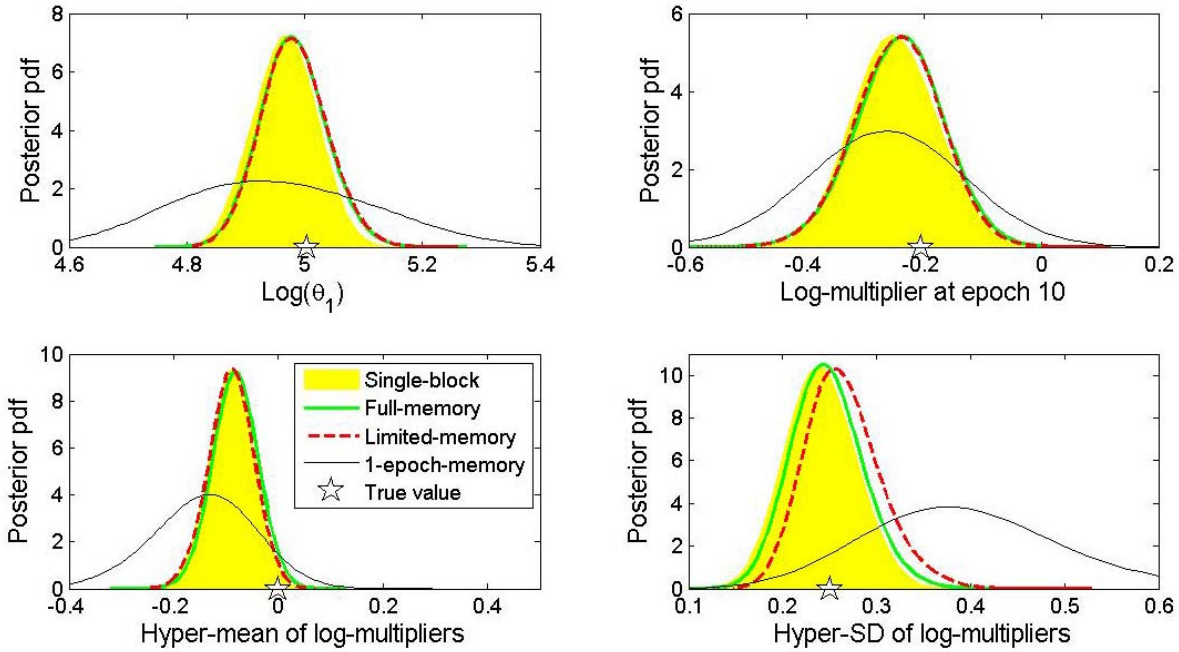

Figure 3. Posterior distributions of selected quantities estimated using different MCMC samplers (100,000 samples). “Hyper-SD” is the standard deviation of hyper-distribution (1).