Comparison of random walk metropolis and adaptive metropolis MCMC algorithms in deep learning enabled Bayesian inference

Saumik Dana[a]

[a] *University of Southern California,*

Abstract

Different Markov chain Monte Carlo (MCMC) sampling algorithms are compared in this work. The parameter being estimated is injection rate and the forward model is coupled flow and geomechanics in the presence of fault. The forward model is reduced using deep learning in the form of LSTM autoencoder, and this reduced order model is used in the evalation passes in the MCMC algorithm.

Keywords: LSTM autoencoder, Bayesian inference, Markov chain Monte Carlo, coupled flow and geomechanics

1. Introduction

Estimating fault geometry from ground displacement is one of the most critical problems in the realm of geoscience. As a first stab at the problem, we estimate injection rate from ground displacement. Bayesian inference has come to the forefront as a standard bearer for inverse problems in the realm, and the advent of applicable deep learning has sped its acceptance. One of the critical aspects of Bayesian inference is Markov chain Monte Carlo (MCMC) sampling, which is often either misunderstood or misinterpreted. In this work, we compare a few popular MCMC sampling algorithms for a problem in which the injection rate is estimated from displacement data generated by forward simulations of a sophisticated coupled flow and geomechanics [1–15] model. The Bayesian inference framework works on the basic tent of uncovering a distribution and starts off with an initial guess for the distribution also called prior to eventually get to the desired distribution also called posterior through a likelihood. The Bayes theorem in a nutshell is:

$$\pi(\theta|\mathcal{D}) = \frac{\pi(\mathcal{D}|\theta)\pi(\theta)}{\int_\theta \pi(\mathcal{D}|\theta)\pi(\theta)} \tag{1}$$

where $\pi(\theta|\mathcal{D})$ is the posterior, $\pi(\mathcal{D}|\theta)$ is the likelihood, $\pi(\theta)$ is the prior given that θ is the parameter being estimated and $\mathcal{D}$ is the data. The quantity that makes evaluation of the posterior difficult is the integral term in the denominator. Since direct evaluation of the integral using quadrature rules is expensive, MCMC sampling [16–26] is used. The prior ideally carries information about the posterior and the likelihood tells us if the sampling points are far away from the posterior or not. Since we do not know the posterior, we can only make educated guesses about the prior unless some complicated mathematical concepts are invoked to inform us about the prior. Such mathematics is an area of active research, and

it makes sense to take the prior to be a uniform distribution with equal probability assigned to all valid sampling points. To formalize the problem, we map the relationship between the injection rate and time to displacement through a forward model $\mathcal{F}$ as follows

$$\mathcal{F}(q,t) := (q,t) \rightarrow u \tag{2}$$

The goal of the inverse problem is to estimate the injection rate as a probability distribution given data $\mathcal{D}$ as follows

$$\pi(q|\mathcal{D}) = \frac{\pi(\mathcal{D}|q)\pi_0(q)}{\int_q \pi(\mathcal{D}|q)\pi_0(q)} \tag{3}$$

where $\pi_0(q)$ is the prior distribution which we take to be a uniform distribution and $\pi(\mathcal{D}|q)$ is the likelihood given by

$$\pi(\mathcal{D}|q) = \prod_{i=1}^{n} \pi(\mathcal{D}(t_i)|q) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\mathcal{F}(q,t_i)-\mathcal{D}(t_i)}{\sigma}\right)^2} \tag{4}$$

where n is the number of data points.

2. Bayesian/MCMC applied to our problem

For the random walk metropolis algorithm, the likelihood is taken to be a normal distribution centered around the current sampling point with a certain variance.

2.1. Initial variance

At the start of the MCMC simulation, the covariance matrix is computed as

$$\boldsymbol{C} = \sigma_0^2[\boldsymbol{\chi}^T\boldsymbol{\chi}]^{-1}$$

where the initial variance is computed using

$$\sigma_0^2 = \sqrt{\frac{\sum_{i=1}^{n}\left(\mathcal{F}(q^0,t_i) - \mathcal{D}(t_i)\right)^2}{n-p}}$$

and the sensitivity matrix $\boldsymbol{\chi}$ is computed using finite differences as

$$\boldsymbol{\chi}_{ij} = \frac{\partial\mathcal{F}(t_i)}{\partial q_j} \equiv \frac{\mathcal{F}(t_i, q_j + q_j * \epsilon) - \mathcal{F}(t_i, q_j)}{q_j * \epsilon}$$

where ϵ is a small number. This initial covariance matrix will hopefully represent our closest guess for the shape and scale of the covariance matrix of the target density function.

Algorithm 1: Adaptive metropolis algorithm

$\sigma_0 \leftarrow \sqrt{\frac{\sum_{i=1}^{n}(\mathcal{F}(q^0,t_i)-\mathcal{D}(t_i))^2}{n-p}}$; // Initial standard deviation

$\boldsymbol{\chi} \leftarrow \frac{\partial \mathcal{F}}{\partial q}$; // Sensitivity matrix $\boldsymbol{\chi}$ with elements $\boldsymbol{\chi}_{ij} := \frac{\partial \mathcal{F}(t_i)}{\partial q_j}$, $i = 1,..,n,\ j = 1,..,p$ where n and p are the number of data points and model parameters being estimated respectively

$\boldsymbol{C} \leftarrow \sigma_0^2[\boldsymbol{\chi}^T\boldsymbol{\chi}]^{-1}$; // Construct initial covariance matrix

for *m=0,2,...,M-1* **do**

/* M is the number of samples drawn */

  if $(m+1)\%m_0$ **then**

    $\boldsymbol{C} \leftarrow \frac{2.38^2}{p} cov(q^0,..,q^m)$; // Update covariance matrix every m_0 iterations for the adaptive metropolis algorithm

  $q^* \sim \mathcal{N}(q^m, \boldsymbol{C})$; // Sample from a normal distribution with mean q^m and covariance $\boldsymbol{C}$

  if $\min\left\{1, \frac{\pi(\mathcal{D}|q^*)\pi_0(q)}{\pi(\mathcal{D}|q^m)\pi_0(q^m)}\right\} > \phi \sim U(0,1)$ **then**

    $q^{m+1} \leftarrow q^*$; // Accept sample if the computed ratio is greater than a number randomly sampled from a uniform distribution between 0 and 1

  else

    $q^{m+1} \leftarrow q^m$; // Reject sample

  $\sigma^{m+1} \sim Inv-\gamma\left(\frac{1}{2}(n_s+n), \frac{1}{2}(n_s{\sigma^m}^2 + \sum_{i=1}^{n}\left(\mathcal{F}(q^{m+1},t_i)-\mathcal{D}(t_i)\right)^2)\right)$; // Sample new standard deviation from an inverse gamma distribution informed by the current standard deviation

2.2. Acceptance strategy

At every iteration, a new sample q^* are picked from a normal distribution with mean as the current guess q^m and covariance $\boldsymbol{C}$ as follows:

$$q^* \sim \mathcal{N}(q^m, \boldsymbol{C})$$

In case of the adaptive metropolis algorithm, the covariance matrix is adapted as more samples are accepted. In general, the adaptation does not necessarily need to be performed at every MCMC step, but rather for certain time intervals. This form of adaptation improves the mixing properties of the algorithm. As the sampling proceeds, successive samples are either accepted or rejected based on the strategy outlined as follows: At every iteration, a ratio is computed between the numerator of Eq. (3) evaluated at the proposed sample q^* at that iteration and the sample q^m at the previous iteration as follows

$$r = \frac{\pi(\mathcal{D}|q^*)\cancel{\pi_0(q)}}{\pi(\mathcal{D}|q^m)\cancel{\pi_0(q^m)}} = \frac{\prod_{i=1}^{n}\cancel{\frac{1}{\sigma^m\sqrt{2\pi}}}e^{-\frac{1}{2}\left(\frac{\mathcal{F}(q^*,t_i)-\mathcal{D}(t_i)}{\sigma^m}\right)^2}}{\prod_{i=1}^{n}\cancel{\frac{1}{\sigma^m\sqrt{2\pi}}}e^{-\frac{1}{2}\left(\frac{\mathcal{F}(q^m,t_i)-\mathcal{D}(t_i)}{\sigma^m}\right)^2}}$$

Note that the $\pi_0(q)$ terms cancel out as π_0 is a uniform distribution and has the same value regardless of q. As shown in Algorithm 1, we proceed to accept q^* only if r is greater than a

number randomly sampled from a uniform distribution between 0 and 1. In lieu of the fact that evaluating r is evidently computationally expensive, we use the log function to process r as

$$log\, r = \frac{1}{2\sigma^{m^2}} \left(\mathcal{E}(q^m) - \mathcal{E}(q^*)\right)$$

where the error-like term $\mathcal{E}$ is given as

$$\mathcal{E}(q) := \sum_{i=1}^{n} \left(\mathcal{F}(q, t_i) - \mathcal{D}(t_i)\right)^2$$

The acceptance condition is then evaluated as $log\, r > \phi' \sim log(U(0,1))$ instead of $r > \phi \sim U(0,1)$. The standard deviation is updated every iteration by sampling from an inverse gamma distribution informed by the current value of standard deviation as follows

$$\sigma^{m+1} \sim Inv - \gamma \left(\frac{1}{2}\left(n_s + n\right), \frac{1}{2}\left(n_s \sigma^{m^2} + \sum_{i=1}^{n} \left(\mathcal{F}(q^{m+1}, t_i) - \mathcal{D}(t_i)\right)^2\right)\right)$$

where n_s is typically taken to be 0.01.

2.3. Summary

A histogram of the samples is eventually constructed after a lot of samples, and we project confidence that the histogram resembles the posterior as closely as possible. Now the samples should ideally be picked from the likelihood since it is the integrand in the denominator, but the typical likelihood function is too complicated to be sampling points from. For that reason, samples are picked from another distribution from which it is easy to sample, and this distribution is called a proposal distribution. To summarise, samples are picked from a proposal distribution, which are then either accepted or rejected from the likelihood evaluation at those samples, and then the accepted samples are processed as a histogram which we say is the posterior or simply the result of the MCMC algorithm.

3. Forward model and deep learning based reduced order model

As shown in Fig. 1, we consider a two-dimensional plane-strain model with the fault under normal faulting conditions, that is, the vertical principal stress due to gravity is the largest among the three principal stresses. The mathematics and numerics of the forward model is explained in Appendix A. The aquifer is hydraulically compartmentalized with a sealing fault that cuts across it. The storage capacity of the aquifer is limited by overpressurization and slip on the fault.

3.1. Reconstruction using LSTM autoencoders

The simulator spits out vtk files for each time stamp in the simulation. Our job is to extract the displacement values corresponding to grid points on the top surface, and construct a time series as the ground truth. A reduced order model would effectively mean reconstructing this time series using LSTM autoencoder, and the optimal deep learning

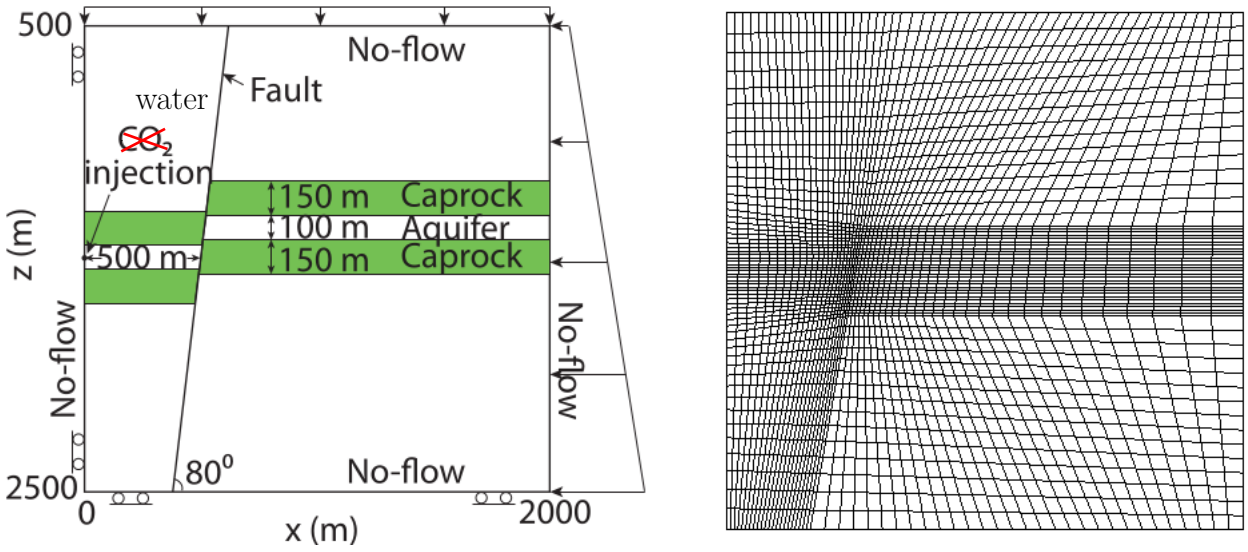

Figure 1: Model of the water injection plane strain case [27]) and the mesh of 3127 nodes and 3016 elements with more refinement around the fault

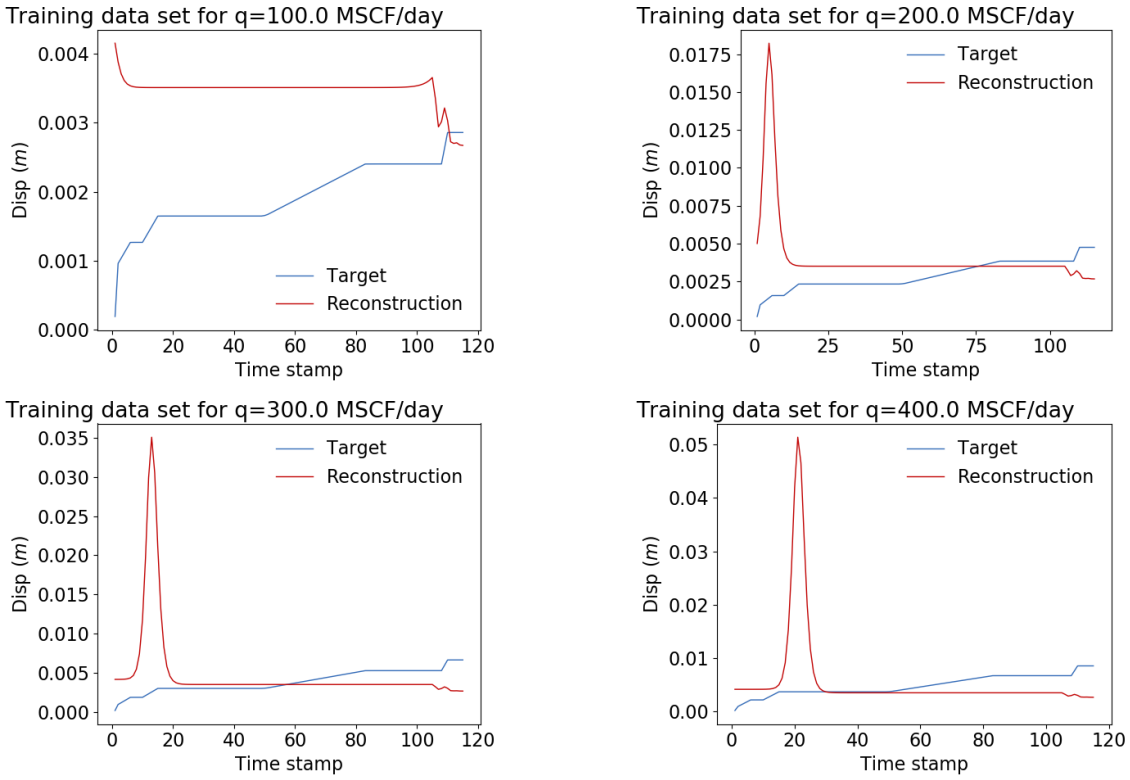

Figure 2: Reconstruction of displacement time series ground truth for u at $(0, 500)$ with sliding window approach

parameters to best reconstruct the time series. The deep learning piece is built on the PyTorch framework, and all simulations are run on a basic AMD Ryzen 3 3200U with Radeon Vega Mobile Gfx $\times$ 4 processor. We use a window size of 10 for 115 time steps,

hidden state size of 5, number of LSTM layers per encoder and decoder is 1, and we deploy the Adam optimizer to train the model using only 10 epochs to avoid overfitting. The 115 data points are divided into windows of 10 data points, and each window slides forward by 1, and the LSTM autoencoder is trained for these windows. During reconstruction, these data chunks in the form of windows are fed into the trained LSTM autoencoder. As we observe in Fig. 2, the sliding window approach allows the reconstructions over these overlaps to be averaged out, which smooths out the spikes that are observed in the nonoverlapping window approach. The reality is that the time series across the injection rates in the spectrum of the generated data spans an order of magnitude, and to fit all that into a LSTM autoencoder in a one size fits all manner is not a trivial task.

4. Results

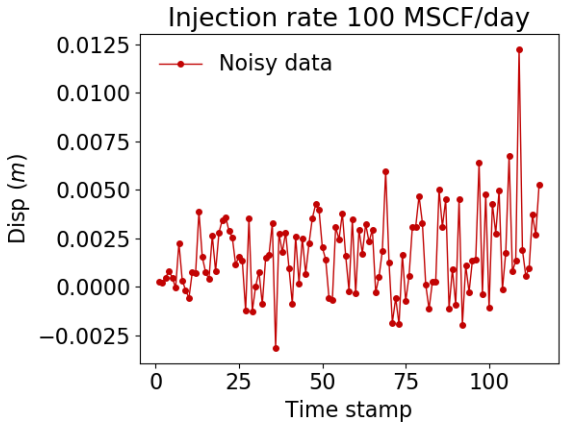

Figure 3: Displacement time series noisy data for u at $(0, 500)$. The noisy data is generated from the ground truth using a normal distribution. This noisy data operates as data to the Bayesian framework although the reduced order model is constructed using the non-noisy data. The reason for using noisy data for the Bayesian framework is that the real scenarios always work with data carrying some noise.

Assuming the $\epsilon \sim N(0, \sigma^2)$ as unbiased, independent and identical normal distribution with standard deviation σ allows us to conveniently generate the synthetic data as shown in Fig. 3.

We start with an initial guess of 250 MSCF/day for all Bayesian/MCMC simulations. Since the initial guess is more often than not way off the desired value, the initial half of the number of samples are burnt-in, which is common practice in MCMC simulations.

In the random walk metropolis algorithm, the covariance matrix remains the same as computed initially before the sampling iterations proceed. Fig. 4 are results of the Bayesian/MCMC inference framework using random walk metropolis algorithm. For the adaptive metropolis algorithm, the interval of adapting the covariance matrix is 50, which means the matrix is modified every 50 samples. Fig. 5 are results of the Bayesian/MCMC inference framework using adaptive metropolis algorithm.

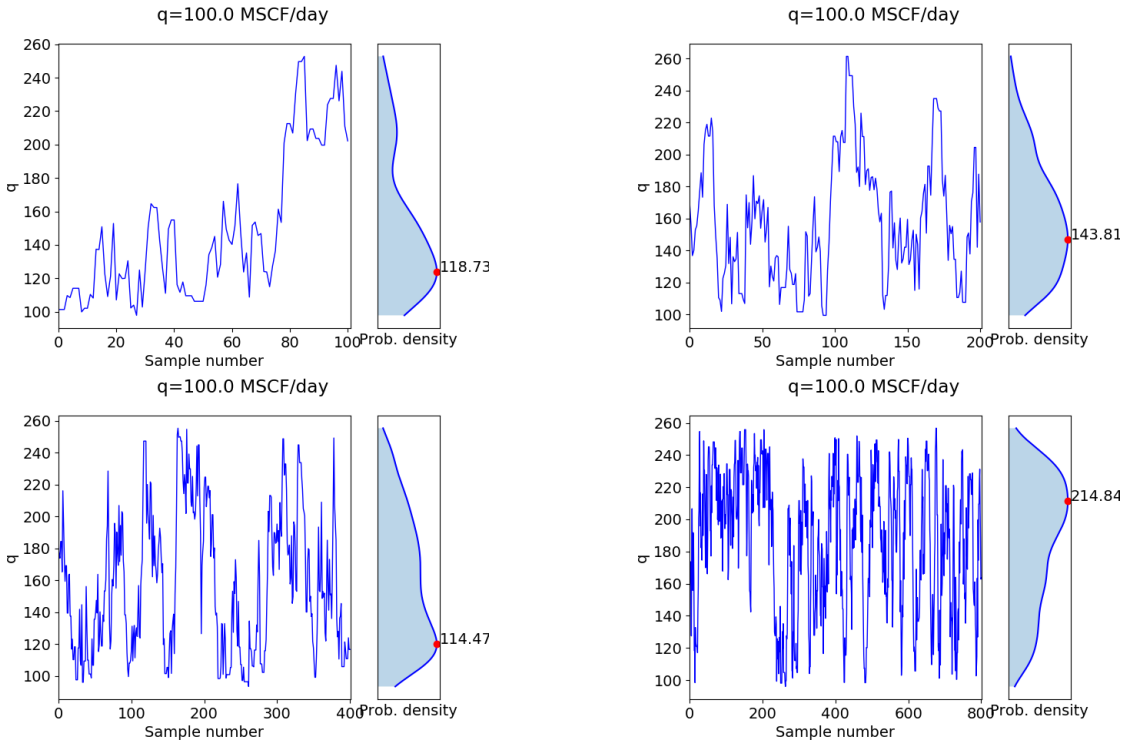

Figure 4: Inversion results using random walk metropolis algorithm

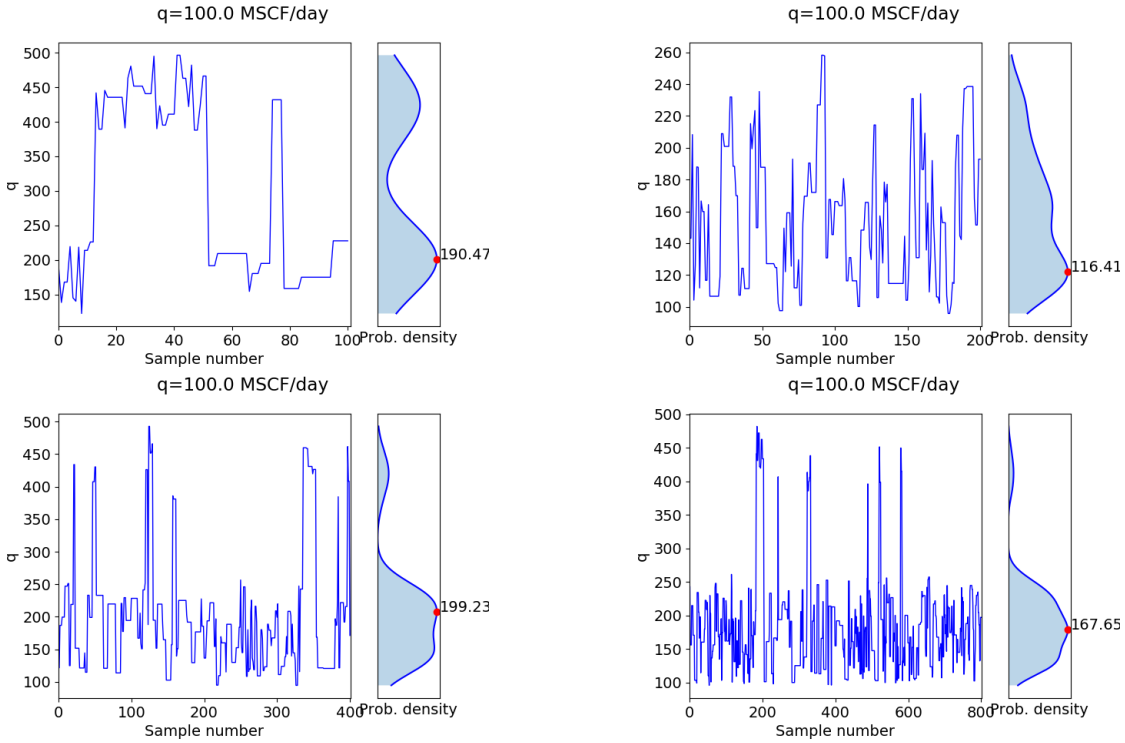

Figure 5: Inversion results using adaptive metropolis algorithm

4.1. Observations

We see from Figs. 4 and 5 that the probability distributions for the injection rate estimates are more spread out in case of random walk metropolis compared to adaptive metropolis algorithm. This indicates that the estimates using random walk metropolis carry more uncertainty compared to those using adaptive metropolis. The peaks of the distributions as marked in Figs. 4 and 5 do not necessarily point to the superiority of one algorithm over the other, but it is evident that the adaptive metropolis algorithm provides more robust estimates in terms of the certainty associated with the estimate.

5. Conclusions and outlook

The MCMC sampling algorithm is as important as the fineness of the reduced order model in deep learning enabled Bayesian inference. The mathematical rigor associated with Bayesian/MCMC allows one to use the most sophisticated MCMC algorithm with confidence attached to the fact that more sophistication would automatically mean less uncertain parameter estimates.

Appendix A. The high fidelity forward model

The governing PDE for displacement $\mathbf{u}$ is the linear momentum balance given by

$$\nabla \cdot \boldsymbol{\sigma} + \rho_b \mathbf{g} = \mathbf{0} \tag{A.1}$$

with the constitutive laws relating poroelastic stress tensor $\boldsymbol{\sigma}$, effective stress tensor $\boldsymbol{\sigma'}$, strain tensor $\boldsymbol{\epsilon}$, volumetric strain $\epsilon = tr(\boldsymbol{\epsilon})$ and pore pressure p given by

$$\begin{aligned} \boldsymbol{\sigma} &= \boldsymbol{\sigma}' - bp\mathbf{I} \\ \boldsymbol{\sigma}' &= \lambda\epsilon\mathbf{I} + 2G\boldsymbol{\epsilon} = \boldsymbol{D}\boldsymbol{\epsilon} \end{aligned} \tag{A.2}$$

with boundary and initial conditions given by

$$\begin{aligned} \mathbf{u} = \overline{\mathbf{u}} \text{ on } \Gamma_D,\ \dot{\mathbf{u}} = \dot{\overline{\mathbf{u}}} \text{ on } \Gamma_D,\ \boldsymbol{\sigma}^T\mathbf{n} = \overline{\mathbf{t}} \text{ on } \Gamma_N \\ \mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}),\ \dot{\mathbf{u}}(\mathbf{x}, 0) = \dot{\mathbf{u}}_0(\mathbf{x}) \end{aligned}$$

As shown in Fig. A.6, slip on the fault is the displacement of the positive side relative to the negative side:

$$(\boldsymbol{u}_+ - \boldsymbol{u}_-) - \boldsymbol{d} = \mathbf{0} \text{ on } \Gamma_f, \tag{A.3}$$

Recognizing that fault tractions are analogous to the boundary tractions, we add in the contributions from integrating the Lagrange multipliers $\boldsymbol{l} \equiv \boldsymbol{\sigma'}\boldsymbol{n}$ over the fault surface in the conventional finite element formulation to get

$$\begin{aligned} &\int_\Omega \nabla\boldsymbol{\eta} : (\boldsymbol{\sigma}' - bp\mathbf{I})\, d\Omega - \int_\Omega \boldsymbol{\eta} \cdot \rho_b\boldsymbol{g}\, d\Omega - \int_{\Gamma_N} \boldsymbol{\eta} \cdot \overline{\boldsymbol{t}}\, d\Gamma \\ &+ \int_{\Gamma_{f_+}} \boldsymbol{\eta} \cdot (\boldsymbol{l} - bp_+\boldsymbol{n})\, d\Gamma - \int_{\Gamma_{f_-}} \boldsymbol{\eta} \cdot (\boldsymbol{l} - bp_-\boldsymbol{n})\, d\Gamma = 0 \end{aligned} \tag{A.4}$$

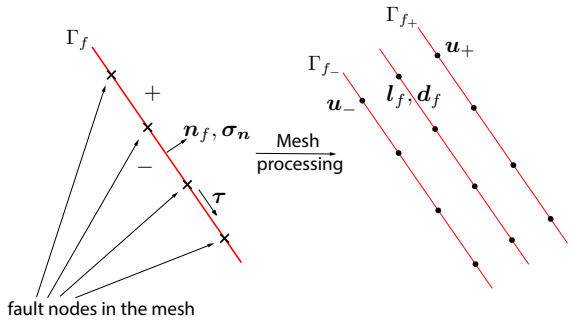

Figure A.6: A fault surface Γ_f is processed to create three surfaces: the positive side surface Γ_{f+} containing $\boldsymbol{u}_+$, the negative side surface Γ_{f-} containing $\boldsymbol{u}_-$, and the slip surface containing the fault effective traction vector $\boldsymbol{l}$ and the slip vector $\boldsymbol{d}$

We use the Mohr-Coulomb theory [28] to define the stability criterion for the fault and define a fault pressure $p_f = \frac{p_- + p_+}{2}$. The shear stress and frictional stresses on the fault are

$$\tau = |\boldsymbol{l} - \sigma_n' \boldsymbol{n}| \equiv |\boldsymbol{l} - (\boldsymbol{l} \cdot \boldsymbol{n})\boldsymbol{n}|$$

$$\tau_f = \begin{cases} \tau_c - \mu_f \boldsymbol{l} \cdot \mathbf{n}, & \boldsymbol{l} \cdot \mathbf{n} < 0, \\ \tau_c, & \boldsymbol{l} \cdot \mathbf{n} \geq 0 \end{cases}$$

where τ_c is the cohesive strength of the fault, μ_f is the coefficient of friction which evolves as

$$\mu_f = \begin{cases} \mu_s - (\mu_s - \mu_d)\frac{|\boldsymbol{d}|}{d_c}, & |\boldsymbol{d}| \leq d_c, \\ \mu_d, & |\boldsymbol{d}| > d_c \end{cases} \tag{A.5}$$

where d_c is a critical slip distance. The fields are approximated as follows:

$$\boldsymbol{u} \approx \boldsymbol{u}_h = \sum_{b=1}^{n_{\text{node}}} \eta_b \boldsymbol{U}_b, \quad \boldsymbol{l} \approx \boldsymbol{l}_h = \sum_{b=1}^{n_{f,\text{node}}} \eta_b \boldsymbol{L}_b, \quad \boldsymbol{d} \approx \boldsymbol{d}_h = \sum_{b=1}^{n_{f,\text{node}}} \eta_b \boldsymbol{D}_b,$$

where n_{node} is the total number of nodes and $n_{f,\text{node}}$ is the number of Lagrange nodes. After substitution of the finite element approximations into the weak form of the problem, we obtain the fully discrete aligns in residual form for all nodes a and lagrange nodes $\bar{a}$:

$$\begin{aligned} \boldsymbol{R}_{u,a}^{Stat} = &\int_\Omega \boldsymbol{B}_a^T (\boldsymbol{\sigma}_h'^{n+1} - b p_h^{n+1} \mathbf{1}) d\Omega - \int_\Omega \boldsymbol{\eta}_a^T \rho_{b,h}^{n+1} \boldsymbol{g} d\Omega - \int_{\Gamma_N} \boldsymbol{\eta}_a^T \bar{\boldsymbol{t}} d\Gamma \\ &+ \int_{\Gamma_{f_+}} \boldsymbol{\eta}_a^T (\boldsymbol{l}_h^{n+1} - b p_{f,h}^{n+1} \boldsymbol{n}) d\Gamma - \int_{\Gamma_{f_-}} \boldsymbol{\eta}_a^T (\boldsymbol{l}_h^{n+1} - b p_{f,h}^{n+1} \boldsymbol{n}) d\Gamma = \mathbf{0} \end{aligned} \tag{A.6}$$

$$\boldsymbol{R}_{l,\bar{a}}^{Stat} = \int_{\Gamma_{f_+}} \boldsymbol{\eta}_{\bar{a}}^T \boldsymbol{u}_{h_+}^{n+1} d\Gamma - \int_{\Gamma_{f_-}} \boldsymbol{\eta}_{\bar{a}}^T \boldsymbol{u}_{h_-}^{n+1} \, d\Gamma - \int_{\Gamma_f} \boldsymbol{\eta}_{\bar{a}}^T \boldsymbol{d}_h^{n+1} d\Gamma = \mathbf{0} \tag{A.7}$$

We find the system Jacobian matrix by isolating the term for the increments in displacements and Lagrange multipliers at time step $n+1$. The system of linear aligns is:

$$\left[\begin{array}{ccc:c} \boldsymbol{K}_{rr} & \boldsymbol{K}_{r+} & \boldsymbol{K}_{r-} & \mathbf{0} \\ \boldsymbol{K}_{+r} & \boldsymbol{K}_{++} & \mathbf{0} & \boldsymbol{C}_+^T \\ \boldsymbol{K}_{-r} & \mathbf{0} & \boldsymbol{K}_{--} & -\boldsymbol{C}_-^T \\ \hdashline \mathbf{0} & \boldsymbol{C}_+ & -\boldsymbol{C}_- & \mathbf{0} \end{array}\right] \left[\begin{array}{c} d\boldsymbol{U}_r \\ d\boldsymbol{U}_+ \\ d\boldsymbol{U}_- \\ \hline d\boldsymbol{L} \end{array}\right] = - \left[\begin{array}{c} \boldsymbol{R}_{u,r}^{Stat} \\ \boldsymbol{R}_{u,+}^{Stat} \\ \boldsymbol{R}_{u,-}^{Stat} \\ \hline \boldsymbol{R}_l^{Stat} \end{array}\right] \tag{A.8}$$

Algorithm 2: Time-marching in poroelastostatics

$n \leftarrow 0$; $t \leftarrow 0$;
$\boldsymbol{d}_h^0 \leftarrow \boldsymbol{0}$; // Initialize fault slip at virgin state
$\boldsymbol{u}_h^0 \leftarrow \boldsymbol{u}_h^{Prestep}$; // Initial condition based on a elastic prestep solve
while $t < T$ **do**
  /* time marching till final time */
  while *Not converged* **do**
    /* Staggered solution algorithm loop */
    Solve flow problem for pressures;
    $\boldsymbol{d}_h^{n+1} \leftarrow \boldsymbol{d}_h^n$; // Initialize fault slip for next time step
    Solve system of aligns using GMRES; // Krylov subspace solver
    $\boldsymbol{L} \leftarrow \boldsymbol{L} + d\boldsymbol{L}$; // Update lagrange multipliers
    $\boldsymbol{U} \leftarrow \boldsymbol{U} + d\boldsymbol{U}$; // Update displacements
    for *Loop over lagrange nodes* **do**
      $\tau \leftarrow |\boldsymbol{l}_h^{n+1} - (\boldsymbol{l}_h^{n+1} \cdot \boldsymbol{n})\boldsymbol{n}|$; // Obtain shear stress on fault
      $\tau_f \leftarrow \tau_f|_{\boldsymbol{d}_h^n}$; // Compute fault friction based on the slip value at previous time step
      while $\tau > \tau_f$ **do**
        /* Satisfy fault constitutive law */
        $\boldsymbol{U}_+ \leftarrow \boldsymbol{U}_+ - \boldsymbol{K}_{++}^{-1} \frac{\tau - \tau_f}{\tau} \boldsymbol{L}$;
        $\boldsymbol{U}_- \leftarrow \boldsymbol{U}_- + \boldsymbol{K}_{--}^{-1} \frac{\tau - \tau_f}{\tau} \boldsymbol{L}$;
        $\boldsymbol{d}_h^{n+1} \leftarrow \boldsymbol{u}_{h_+}^{n+1} - \boldsymbol{u}_{h_-}^{n+1}$; // Update fault slip
        $\tau_f \leftarrow \tau_f|_{\boldsymbol{d}_h^{n+1}}$; // Update fault friction
      $\boldsymbol{d}_h^{n+1} \leftarrow \boldsymbol{u}_{h_+}^{n+1} - \boldsymbol{u}_{h_-}^{n+1}$; // Update fault slip
  $n \leftarrow n + 1$;
  $t \leftarrow t + \Delta t$;

where the top row corresponds to displacement nodes excluding the fault positive and negative side nodes. In many quasi-static simulations it is convenient to compute a static problem with elastic deformation prior to computing a transient response. The heavy lifting for the time marching is done at the Krylov solver [29] stage to solve the system of Eqs. (A.8) as shown in Algorithm 2.

References

[1] S. Dana, M. F. Wheeler, Convergence analysis of fixed stress split iterative scheme for small strain anisotropic poroelastoplasticity: a primer, arXiv preprint arXiv:1810.04163 (2018).

[2] S. Dana, M. F. Wheeler, The correspondence between voigt and reuss bounds and the decoupling constraint in a two-grid staggered solution algorithm to coupled flow

and deformation in heterogeneous poroelastic media, arXiv preprint arXiv:1810.09443 (2018).

[3] S. Dana, M. F. Wheeler, Convergence analysis of fixed stress split iterative scheme for anisotropic poroelasticity with tensor biot parameter, Computational Geosciences 22 (2018) 1219–1230.

[4] S. Dana, M. F. Wheeler, Convergence analysis of two-grid fixed stress split iterative scheme for coupled flow and deformation in heterogeneous poroelastic media, Computer Methods in Applied Mechanics and Engineering 341 (2018) 788–806.

[5] S. Dana, M. F. Wheeler, Design of convergence criterion for fixed stress split iterative scheme for small strain anisotropic poroelastoplasticity coupled with single phase flow, arXiv preprint arXiv:1912.06476 (2019).

[6] S. Dana, B. Ganis, M. F. Wheeler, A multiscale fixed stress split iterative scheme for coupled flow and poromechanics in deep subsurface reservoirs, Journal of Computational Physics 352 (2018) 1–22.

[7] S. Dana, S. Srinivasan, S. Karra, N. Makedonska, J. D. Hyman, D. O'Malley, H. Viswanathan, G. Srinivasan, Towards real-time forecasting of natural gas production by harnessing graph theory for stochastic discrete fracture networks, Journal of Petroleum Science and Engineering 195 (2020) 107791.

[8] S. Dana, M. Jammoul, M. F. Wheeler, Performance studies of the fixed stress split algorithm for immiscible two-phase flow coupled with linear poromechanics, Computational Geosciences (2021) 1–15.

[9] S. Dana, J. Ita, M. F. Wheeler, The correspondence between voigt and reuss bounds and the decoupling constraint in a two-grid staggered algorithm for consolidation in heterogeneous porous media, Multiscale Modeling & Simulation 18 (2020) 221–239.

[10] S. Dana, System of equations and staggered solution algorithm for immiscible two-phase flow coupled with linear poromechanics, arXiv preprint arXiv:1912.04703 (2019).

[11] S. Dana, Addressing challenges in modeling of coupled flow and poromechanics in deep subsurface reservoirs, Ph.D. thesis, The University of Texas at Austin, 2018.

[12] S. Dana, M. F. Wheeler, Augmented lagrangian for treatment of hanging nodes in hexahedral meshes, arXiv preprint arXiv:1809.04031 (2018).

[13] S. Dana, M. F. Wheeler, An efficient algorithm for numerical homogenization of fluid filled porous solids: part-i, arXiv preprint arXiv:2002.03770 (2020).

[14] S. Dana, M. F. Wheeler, A machine learning accelerated fe^2 homogenization algorithm for elastic solids, arXiv preprint arXiv:2003.11372 (2020).

[15] S. Dana, A simple framework for arriving at bounds on effective moduli in heterogeneous anisotropic poroelastic solids, arXiv preprint arXiv:1912.10835 (2019).

[16] A. Shapiro, Monte carlo sampling methods, Handbooks in operations research and management science 10 (2003) 353–425.

[17] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, Oxford University Press, 1970.

[18] H. Haario, E. Saksman, J. Tamminen, An adaptive metropolis algorithm, Bernoulli (2001) 223–242.

[19] C. L. Mueller, Exploring the common concepts of adaptive MCMC and Covariance Matrix Adaptation schemes, 2010.

[20] J. A. Vrugt, Markov chain monte carlo simulation using the dream software package: Theory, concepts, and matlab implementation, Environmental Modelling & Software 75 (2016) 273–316.

[21] S. Kumbhare, A. Shahmoradi, Matdram: A pure-matlab delayed-rejection adaptive metropolis-hastings markov chain monte carlo sampler, arXiv preprint arXiv:2010.04190 (2020).

[22] B. Ballnus, S. Hug, K. Hatz, L. Görlitz, J. Hasenauer, F. J. Theis, Comprehensive benchmarking of markov chain monte carlo methods for dynamical systems, BMC systems biology 11 (2017) 1–18.

[23] H. Haario, M. Laine, A. Mira, E. Saksman, Dram: efficient adaptive mcmc, Statistics and computing 16 (2006) 339–354.

[24] S. Dana, Comparison of lstm autoencoder based deep learning enabled bayesian inference using two time series reconstruction approaches, arXiv preprint arXiv:2203.01936 (2022).

[25] S. Dana, K. Reddy, Bayesian inference and markov chain monte carlo based estimation of a geoscience model parameter, arXiv preprint arXiv:2201.01868 (2021).

[26] S. Dana, K. R. Lyathakula, Uncertainty quantification in friction model for earthquakes using bayesian inference, arXiv preprint arXiv:2104.11156 (2021).

[27] F. Cappa, J. Rutqvist, Impact of CO_2 geological sequestration on the nucleation of earthquakes, Geophys. Res. Lett. 38 (2011) L17313.

[28] J. C. Jaeger, N. G. W. Cook, Fundamentals of Rock Mechanics, Chapman and Hall, London, 1979.

[29] H. A. Van Der Vorst, Krylov subspace iteration, Computing in science & engineering 2 (2000) 32–37.