16th World Congress in Computational Mechanics (WCCM)
21-26 July 2024, Vancouver, Canada
A. Korobenko, M. Laforest, S. Prudhomme, R. Vaziri (Eds)

MARKOV CHAIN MONTE CARLO CAPABILITIES IN DAKOTA

ERNESTO E. PRUDENCIO AND JOHN A. STEPHENS

Sandia National Laboratories, Albuquerque, NM
eeprude@sandia.gov (corresponding author) and jasteph@sandia.gov

Key words: Markov chain Monte Carlo, Bayesian calibration, Dakota software package.

Abstract. We give an overview of MCMC capabilities in the Dakota software package from Sandia National Laboratories, and present some Bayesian calibration results.

1 INTRODUCTION

The proper quantification of prediction uncertainties is essential to establishing the credibility of computational models, especially for high-consequence problems. In many cases the model parameters and their uncertainties must be inferred from experimental data. Bayesian analysis [1, 2], in connection with Markov chain Monte Carlo (MCMC) methods [3, 4], allows us to calibrate model parameters under uncertainty.

This paper is organized as follows. In Section 2 we briefly describe Bayesian calibration, followed by a review in Section 3 of some of the MCMC methods included in the Dakota software package, which is briefly described in Section 4. In Section 5 we present the problems used in some numerical exercises, followed by the presentation of numerical results in Section 6. We finalize the paper with conclusions in Section 7.

2 BAYESIAN CALIBRATION

Given experimental data and a computational model aimed to explain the experimental results, Bayesian analysis offers a way to calibrate model parameters under uncertainty. Given $n_y \geq 1$, $n_d \geq 1$, $n_m \geq 1$, and $n_{\text{obs}} \geq 1$, let $\boldsymbol{\theta}_y \in \mathbb{R}^{n_y}$ denote model parameters, $\mathbf{y} : \mathbb{R}^{n_y} \to \mathbb{R}^{n_{\text{obs}}}$ denote the computational model, $\boldsymbol{\theta}_m \in \mathbb{R}^{n_m}$ denote parameters for the model form uncertainty [5], $\boldsymbol{\epsilon}_{\text{model}} : \mathbb{R}^{n_m} \to \mathbb{R}^{n_{\text{obs}}}$ denote the model form uncertainty, $\mathbf{d} \in \mathbb{R}^{n_{\text{obs}}}$ denote the available experimental data, $\boldsymbol{\theta}_d \in \mathbb{R}^{n_d}$ denote parameters for data uncertainty, $\boldsymbol{\epsilon}_{\text{data}} : \mathbb{R}^{n_d} \to \mathbb{R}^{n_{\text{obs}}}$ denote the uncertainty in the data, $n_\theta = n_y + n_d + n_m$ denote the amount of all parameters, and

$$\boldsymbol{\theta} = (\boldsymbol{\theta}_y, \boldsymbol{\theta}_d, \boldsymbol{\theta}_m) \in \mathbb{R}^{n_\theta} \tag{1}$$

denote the full parameter vector. The Bayes' formula then reads

$$\pi(\boldsymbol{\theta}|\mathbf{d}) = \frac{\pi(\mathbf{d}|\boldsymbol{\theta}) \cdot \pi(\boldsymbol{\theta})}{\pi(\mathbf{d})} = \frac{\pi(\mathbf{d}|\boldsymbol{\theta}) \cdot \pi(\boldsymbol{\theta})}{\int \pi(\mathbf{d}|\boldsymbol{\theta}) \cdot \pi(\boldsymbol{\theta})\; d\boldsymbol{\theta}}, \tag{2}$$

where $\pi(\cdot)$ indicates a probability density function (PDF), $\pi(\boldsymbol{\theta})$ is the prior PDF, $\pi(\mathbf{d}|\boldsymbol{\theta})$ is the likelihood PDF, $\pi(\mathbf{d})$ is the model evidence, and $\pi(\boldsymbol{\theta}|\mathbf{d})$ is the posterior PDF. Model evidence allows one to quantitatively rank models according to their ability to explain given data, as well as to quantify Occam's razor [6].

In the following we describe a typical likelihood PDF. An usual assumption is that the errors in the data and in the model output are additive, as in

$$\mathbf{d} + \boldsymbol{\epsilon}_{\text{data}}(\boldsymbol{\theta}_d) = \text{unknown reality} = \mathbf{y}(\boldsymbol{\theta}_y) + \boldsymbol{\epsilon}_{\text{model}}(\boldsymbol{\theta}_m). \tag{3}$$

Other uncertainty assumptions are also possible, e.g. multiplicative errors. In the case of (3), if one further assumes that $\boldsymbol{\epsilon}_{\text{data}}(\boldsymbol{\theta}_d)$ and $\boldsymbol{\epsilon}_{\text{model}}(\boldsymbol{\theta}_m)$ are independent Gaussian random vectors with zero means and respective (positive definite) covariance matrices $\mathbf{C}_{\text{data}}(\boldsymbol{\theta}_d)$ and $\mathbf{C}_{\text{model}}(\boldsymbol{\theta}_m)$, and letting $\mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$ denote a Gaussian distribution of mean $\boldsymbol{\mu}$ and covariance $\mathbf{C}$, then one can write

$$\mathbf{d} = \mathbf{y}(\boldsymbol{\theta}_y) + \mathcal{N}(\mathbf{0}, \mathbf{C}_{\text{data}}(\boldsymbol{\theta}_d) + \mathbf{C}_{\text{model}}(\boldsymbol{\theta}_m)), \tag{4}$$

that is,

$$\pi(\mathbf{d}|\boldsymbol{\theta}) = \frac{1}{(\sqrt{2\pi})^{n_{\text{obs}}}} \cdot \frac{1}{\sqrt{\det(\mathbf{C})}} \cdot e^{-\frac{1}{2}\left\{\mathbf{r}^T \cdot \mathbf{C}^{-1} \cdot \mathbf{r}\right\}}, \tag{5}$$

where $\mathbf{r} = \mathbf{d} - \mathbf{y}(\boldsymbol{\theta}_y) \in \mathbb{R}^{n_{\text{obs}}}$ and $\mathbf{C} = \mathbf{C}_{\text{data}} + \mathbf{C}_{\text{model}} \in \mathbb{R}^{n_{\text{obs}} \times n_{\text{obs}}}$, with $\det(\mathbf{C})$ denoting the determinant of $\mathbf{C}$. It should be noted that not all n_θ parameters need to be open to calibration: some of them might be known or fixed in advance.

The description above can be extended to a case with multiple experimental data vectors $\mathbf{d}^{(j)} \in \mathbb{R}^{n_{\text{obs}}^{(j)}}$, $1 \leq j \leq n_e$. For instance, the n_e experiments could correspond to different experiment scenarios (e.g. different temperatures), and have different sizes $n_{\text{obs}}^{(j)}$ (e.g. different amounts of probes used, and/or probes used at different locations), while the same computational model and the same parametrized model form uncertainty could be used with different deterministic values for the scenario parameters, as well as different output vectors in accordance to the corresponding experiments. In such case, equation (4) is replaced by (note that $\boldsymbol{\theta}_y$ and $\boldsymbol{\theta}_m$ are not written with superscript "(j)")

$$\mathbf{d}^{(j)} = \mathbf{y}^{(j)}(\boldsymbol{\theta}_y) + \mathcal{N}(\mathbf{0}, \mathbf{C}_{\text{data}}^{(j)}(\boldsymbol{\theta}_d^{(j)}) + \mathbf{C}_{\text{model}}^{(j)}(\boldsymbol{\theta}_m)). \tag{6}$$

Denoting $N_{\text{obs}} = \sum_{j=1}^{n_e} n_{\text{obs}}^{(j)}$, $N_d = \sum_{j=1}^{n_e} n_d^{(j)}$, $N_\theta = n_y + N_d + n_m$, $\mathbf{d} = (\mathbf{d}^{(1)}, \ldots, \mathbf{d}^{(n_e)}) \in \mathbb{R}^{N_{\text{obs}}}$, $\boldsymbol{\theta} = (\boldsymbol{\theta}_y, \boldsymbol{\theta}_d^{(1)}, \ldots, \boldsymbol{\theta}_d^{(n_e)}, \boldsymbol{\theta}_m) \in \mathbb{R}^{N_\theta}$, and further assuming that the experiments are independent among themselves, then equation (5) is replaced by

$$\pi(\mathbf{d}|\boldsymbol{\theta}) = \prod_{j=1}^{n_e} \frac{e^{-\frac{1}{2}\left\{\left(\mathbf{r}^{(j)}\right)^T \cdot \left[\mathbf{C}^{(j)}\right]^{-1} \cdot \mathbf{r}^{(j)}\right\}}}{(\sqrt{2\pi})^{n_{\text{obs}}^{(j)}} \cdot \sqrt{\det(\mathbf{C}^{(j)})}} = \frac{e^{-\frac{1}{2}\left\{\mathbf{r}^T \cdot \mathbf{C}^{-1} \cdot \mathbf{r}\right\}}}{(\sqrt{2\pi})^{N_{\text{obs}}} \cdot \sqrt{\det(\mathbf{C})}}, \tag{7}$$

where $\mathbf{r}^{(j)} = \mathbf{d}^{(j)} - \mathbf{y}^{(j)}(\boldsymbol{\theta}_y) \in \mathbb{R}^{n_{\text{obs}}^{(j)}}$, $\mathbf{C}^{(j)} = \mathbf{C}_{\text{data}}^{(j)} + \mathbf{C}_{\text{model}}^{(j)} \in \mathbb{R}^{n_{\text{obs}}^{(j)} \times n_{\text{obs}}^{(j)}}$, $\mathbf{r} \in \mathbb{R}^{N_{\text{obs}}}$ is the vector formed by concatenating all $\mathbf{r}^{(j)}$'s, and $\mathbf{C} \in \mathbb{R}^{N_{\text{obs}} \times N_{\text{obs}}}$ is the block diagonal matrix formed with all $\mathbf{C}^{(j)}$'s.

If we consider Cholesky factorization, (7) can be rewritten as

$$\pi(\mathbf{d}|\boldsymbol{\theta}) = \prod_{j=1}^{n_e} \frac{1}{(\sqrt{2\pi})^{n_{\text{obs}}^{(j)}}} \cdot \frac{1}{\sqrt{\det(\mathbf{C}^{(j)})}} \cdot e^{-\frac{1}{2}\|\hat{\mathbf{r}}^{(j)}\|_2^2} = \frac{1}{(\sqrt{2\pi})^{N_{\text{obs}}}} \cdot \frac{1}{\sqrt{\det(\mathbf{C})}} \cdot e^{-\frac{1}{2}\|\hat{\mathbf{r}}\|_2^2}, \quad (8)$$

where $\hat{\mathbf{r}}^{(j)} = \mathbf{L}^{(j)} \cdot \mathbf{r}^{(j)} \in \mathbb{R}^{n_{\text{obs}}^{(j)}}$, $\mathbf{L}^{(j)} \in \mathbb{R}^{n_{\text{obs}}^{(j)} \times n_{\text{obs}}^{(j)}}$ is the Cholesky decomposition of $\mathbf{C}^{(j)} = \mathbf{L}^{(j)} \cdot \mathbf{L}^{(j)^T}$, $\hat{\mathbf{r}} = \mathbf{L} \cdot \mathbf{r} \in \mathbb{R}^{N_{\text{obs}}}$, and $\mathbf{L} \in \mathbb{R}^{N_{\text{obs}} \times N_{\text{obs}}}$ is the Cholesky decomposition of $\mathbf{C} = \mathbf{L} \cdot \mathbf{L}^T$. The vector $\hat{\mathbf{r}}$ can also be seen as the vector formed by concatenating all $\hat{\mathbf{r}}^{(j)}$'s, and the matrix $\mathbf{L}$ can also be seen as the block diagonal matrix formed with all $\mathbf{L}^{(j)}$'s.

3 SOME MARKOV CHAIN MONTE CARLO (MCMC) METHODS

With the computational power increase since the 1990s, it became feasible to use MCMC to sample the posterior PDF (2) and thus perform Bayesian calibration in situations other than those involving linear models and Gaussian distributions. Also, if one needs to "only" calibrate parameters (as in this present paper), caring only for the ratio between the posterior PDF values at two different samples, then the model evidence term in (2) can be neglected. We will denote the target PDF by $\pi_{\text{target}}(\cdot)$ (which is the posterior PDF in the case of Bayesian analysis), and the chain positions by $\mathbf{x}^{(k)},\ k = 0, 1, 2, \ldots$.

In practice the values of posterior PDFs might get very small, in which case it is convenient to work with ln values. When using (8), one gets:

$$\begin{aligned} \ln[\pi(\boldsymbol{\theta}|\mathbf{d})] &= \ln[\pi(\boldsymbol{\theta})] - \frac{1}{2}\left\{N_{\text{obs}} \cdot \ln(2\pi) + \sum_{j=1}^{N_e} \ln[\det(\mathbf{C}^{(j)})] + \sum_{j=1}^{N_e} \|\hat{\mathbf{r}}^{(j)}\}\|_2^2\right\} \\ &= \ln[\pi(\boldsymbol{\theta})] - \frac{1}{2}\left\{N_{\text{obs}} \cdot \ln(2\pi) + \ln[\det(\mathbf{C})] + \|\hat{\mathbf{r}}\|_2^2\right\}. \end{aligned} \quad (9)$$

Table 1 lists the input parameters for all methods described in this section.

3.1 Vanilla Metropolis Hastings Algorithm (MH)

The vanilla MCMC algorithm is based on [7, 8]. Given the current position $\mathbf{x}^{(k)}$, and a proposal distribution (PD) $q(\mathbf{x}^{(k)}, \mathbf{c})$ (typically a Gaussian PDF centered at $\mathbf{x}^{(k)}$, assuming values at positions $\mathbf{c}$), the algorithm generates a candidate $\mathbf{c}^{(k)}$ from $q(\mathbf{x}^{(k)}, \cdot)$. The algorithm then sets $\mathbf{x}^{(k+1)} = \mathbf{c}^{(k)}$ with probability

$$p = \min\left\{1, \frac{\pi_{\text{target}}(\mathbf{c}^{(k)})}{\pi_{\text{target}}(\mathbf{x}^{(k)})} \cdot \frac{q(\mathbf{c}^{(k)}, \mathbf{x}^{(k)})}{q(\mathbf{x}^{(k)}, \mathbf{c}^{(k)})}\right\}, \quad (10)$$

otherwise it sets $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}$.

MCMC Method (Subsection)	Input Parameter	Description
MH (3.1)	$\mathbf{M}$	Covariance matrix for the PD $q(\mathbf{x}^{(k)}, \cdot)$
DR (3.2)	$\mathbf{M}$	Covariance matrix for the PD
	n_{stages}	Number of stages
	scaling	Method: “linear” or ”power”
	b	Scale parameter related to PD scaling
AM (3.3)	$\mathbf{M}^{(0)}$	Initial covariance matrix for the PD
	n_{start}	Number of MCMC steps to start AM
	n_{period}	Number of MCMC steps between AM instances
	α	Scale parameter for the adapted covariance matrix
MALA (3.5)	$\mathbf{M}$	Covariance matrix for the PD
	τ	Step size

Table 1: Input parameters for the methods (with Gaussian proposal distributions) used in this paper.

3.2 Delayed Rejection (DR)

DR is a local adaptive strategy [9, 10] for MH. The basic idea is that, within the k-th MCMC step, one generates a candidate position $\mathbf{c}^{(k,i=1)}$ in the first ($i = 1$) stage using PD $q_1(\mathbf{c}^{(k,1)}, \cdot)$. If this candidate is rejected, the algorithm generates another candidate $\mathbf{c}^{(k,2)}$ in a second ($i = 2$) stage using a PD $q_2(\mathbf{x}^{(k)}, \mathbf{c}^{(k,1)}, \cdot)$, and so on. In the case of Dakota, the user provides a covariance matrix for a single PD $q(\cdot, \cdot)$, a parameter $b > 0$, and the maximum number of stages $n_{\text{stages}} \geq 1$. Then, for $i = 1, \ldots, n_{\text{stages}}$, DR sets

$$q_i(\cdot, \cdot) = \frac{b}{i+1} \cdot q(\cdot, \cdot) \quad \text{or} \quad q_i(\cdot, \cdot) = \frac{b}{2^i} \cdot q(\cdot, \cdot), \tag{11}$$

depending if the scaling type (another input option) is “linear” or “power”, respectively.

3.3 Adaptive Metropolis (AM)

AM is a global adaptive strategy [11] for MH. The basic idea is to adapt the Gaussian PD covariance matrix as the chain progresses. Given an initial covariance matrix $\mathbf{M}^{(0)}$, integers $n_{\text{start}} \geq 1$ and $n_{\text{period}} \geq 1$, as well as a real scalar $\alpha > 0$, the algorithm uses $\mathbf{M}^{(0)}$ in the first n_{start} steps. At the n_{start}-th step, the algorithm computes a new PD covariance $\mathbf{M}$ using all the previous n_{start} chain positions, and uses $\alpha\mathbf{M}$ for PD in the next n_{period} steps, when a new PD covariance is computed using the previous PD covariance and the last n_{period} chain positions. This procedure then repeats itself every n_{period} steps.

3.4 Delayed Rejection Adaptive Metropolis (DRAM)

DRAM combines DR and AM into one method [12].

3.5 Metropolis-Adjusted Langevin Algorithm (MALA)

The MALA algorithm [13, 14, 15, 16] generates candidate positions through a Langevin dynamics using $\nabla_{\boldsymbol{\theta}}\{\ln[\pi_{\text{target}}(\boldsymbol{\theta})]\}$, and rejects or accepts them as in MH. Given the current MCMC position $\mathbf{x}^{(k)}$, and a covariance matrix $\mathbf{M}$, the candidate position $\mathbf{c}^{(k)}$ is sampled form the Gaussian PD with mean $\mathbf{x}^{(k)}+\tau\mathbf{M}\nabla_{\boldsymbol{\theta}}\{\ln[\pi_{\text{target}}(\boldsymbol{\theta})]\}$ and covariance $2\tau\mathbf{M}$. This candidate is then accepted or rejected using rule (10). When using MALA for Bayesian calibration, the term $\nabla_{\boldsymbol{\theta}}\{\ln[\pi_{\text{target}}(\boldsymbol{\theta})]\}$ becomes

$$\nabla_{\boldsymbol{\theta}}\{\ln[\pi(\boldsymbol{\theta}|\mathbf{d})]\} = \nabla_{\boldsymbol{\theta}}\{\ln[\pi(\boldsymbol{\theta})]\} - \frac{1}{2}\left\{\nabla_{\boldsymbol{\theta}}\{\ln[\det(\mathbf{C})]\} + \sum_{k=1}^{N_{\text{obs}}} \hat{\mathbf{r}}_k \cdot \nabla_{\boldsymbol{\theta}}\hat{\mathbf{r}}_k\right\}. \tag{12}$$

3.6 Dimension-Independent Likelihood-Informed (DILI)

The DILI algorithm [17] is currently being implemented, and it will be available in Dakota 6.21 in November of 2024.

4 THE DAKOTA PACKAGE

The Dakota (https://dakota.sandia.gov) software package is an open-source (LGPL) software for black-box, ensemble analysis of computational models [18]. Developed primarily at Sandia National Laboratories, Dakota contains methods for forward and inverse uncertainty quantification, global sensitivity analysis, optimization, and model calibration. Source code and pre-built binaries for Linux, macOS, and Windows are available at https://github.com/snl-dakota/.

4.1 Bayesian Calibration in Dakota

Dakota follows (9) and provides interfaces to different packages, e.g. QUESO [19] and MUQ [20]. Methods MH, AM, DR, DRAM, and MALA are all available in Dakota 6.20.

5 PROBLEMS USED FOR THE NUMERICAL EXERCISES

In this section we present the problems used in our numerical exercises.

5.1 Cantilever

Figure 1 shows the cantilever problem. It involves only one parameter ($n_y = 1$) to be calibrated: the Young's modulus E ($N/m^2 = Pa$). Given the total beam length L (m), the horizontal cross section length w_x (m), the vertical cross section length w_y (m), the horizontal force F_x (N), the vertical force F_y (N), and the resulting total displacement δ (m), the problem is to infer E. Assuming the beam behaves elastically, we have:

$$\delta_x = \frac{4L^3}{Ew_xw_y}\cdot\frac{F_x}{w_x^2}, \quad \delta_y = \frac{4L^3}{Ew_xw_y}\cdot\frac{F_y}{w_y^2}, \quad \delta = \frac{4L^3}{Ew_xw_y}\cdot\sqrt{\left(\frac{F_x}{w_x^2}\right)^2+\left(\frac{F_y}{w_y^2}\right)^2}. \tag{13}$$

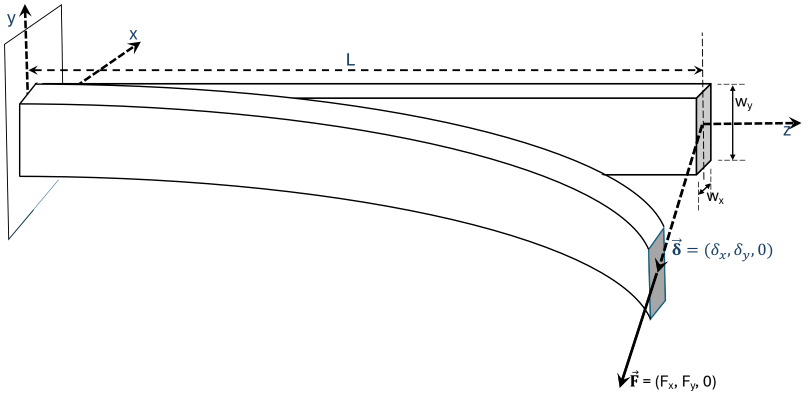
y
x
L
w$_y$
z
w$_x$
$\vec{\boldsymbol{\delta}} = (\delta_x, \delta_y, 0)$
$\vec{\mathbf{F}}$ = (F$_x$, F$_y$, 0)

Figure 1: The cantilever problem, with a beam of rectangular cross section.

5.2 Horizontal Sequence of Springs

Figure 2 shows the springs problem. It involves $n \geq 1$ spring constants $k_1, \ldots, k_n$ (all in N/m). Given the force F (N), the values of $0 \leq n_{\text{known}} < n$ known spring constants, and the resulting total displacement x (m), the problem is to infer the values of the $n - n_{\text{known}}$ unknown spring constants. That is, $n_y = n - n_{\text{known}}$.

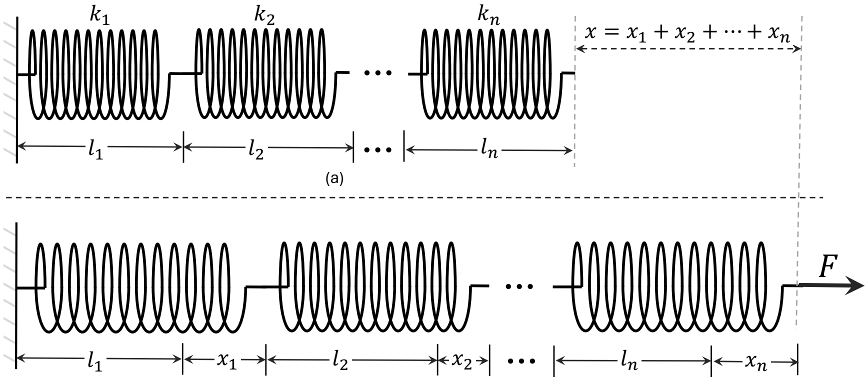

Figure 2: The springs problem, with $n \geq 1$.

6 RESULTS FROM THE NUMERICAL EXERCISES

In all our numerical exercises we used one experiment ($n_e = 1$), and only model parameters were to be calibrated. That is, $n_d = n_m = 0$, and $\nabla_{\boldsymbol{\theta}}\{\ln[\det(\mathbf{C})]\} = \mathbf{0} \in \mathbf{R}^{N_\theta}$ in (12). All exercises were conducted with the Dakota-MUQ interface in a Linux single compute node: no parallelism was used. Table 2 lists the input parameters and some output information for all tests.

Input Parameters and Output	Problem and MCMC Method						
	Cantilever		Springs 1D		Springs 2D		Springs 3D
	DRAM	MALA	DRAM	MALA	DRAM	MALA	DRAM
# samples	5,000	10,000	10,000	10,000	100,000	100,000	1,000,000
$\mathbf{x}^{(0)}$	2.e11	2.e10	3.e3	3.e3	3.e3	3.e3	3.e3
$\mathbf{M}^{(0)}$ or $\mathbf{M}$	1.e16	1.6e19	4.e6	9.e9	4.e6·$\mathbf{I}_{2\times2}$	9.e9·$\mathbf{I}_{2\times2}$	4.e6·$\mathbf{I}_{3\times3}$
n_{stages}	2	–	2	–	2	–	2
scaling	Power	–	Power	–	Power	–	Power
b	1.2	–	1.2	–	1.2	–	1.2
n_{start}	20	–	20	–	20	–	20
n_{period}	20	–	20	–	20	–	20
α	0.1	–	0.1	–	0.1	–	0.1
τ	–	0.01	–	5.e-7	–	5.e-7	–
run time (s)	4.7	13.8	9.9	21.8	128.2	211.2	1372.6
# calls	6,263	40,000	12,909	40,000	166,299	400,000	1,833,581
# new	6,263	20,000	12,909	20,000	166,299	200,000	1,833,581
# duplicated	0	20,000	0	20,000	0	200,000	0

Table 2: Input parametes and some output information related to the numerical results in Section 6. $\mathbf{I}_{2\times2}$ and $\mathbf{I}_{3\times3}$ indicate identity matrices.

6.1 Cantilever

We used $L = 1\ m$, $w_x = 0.01\ m$, $w_y = 0.02\ m$, $F_x = 3\ N$, and $F_y = 3\ N$. For mimicking the experimental results we used $E = 69\ GPa$ (Aluminum) and formulas in (13), set the measured total displacement equal to 8.9 mm, and set the additive measurement noise as $\mathcal{N}(0, 1.e - 6\ m^2)$. For prior PDF we used uniform $\mathcal{U}(1.e10,\ 5.1e11)$. Results and comments are presented in Figure 3.

6.2 Horizontal Sequence of Springs (One-Dimensional Parameter Space)

We used $n = 10$, $n_{\text{known}} = 9$, all n_{known} known spring constants equal to $2.e4\ N/m$, and $F = 1\ N$. For mimicking the experimental results we used the unknown spring constants equal to $2.e4\ N/M$, set the measured total displacement equal to 5.1 mm, and set the additive measurement noise as $\mathcal{N}(0, 2.25e-8\ m^2)$. For prior PDF we used uniform $\mathcal{U}(1.e3, 1.e5)$. Results and comments are presented in Figure 4.

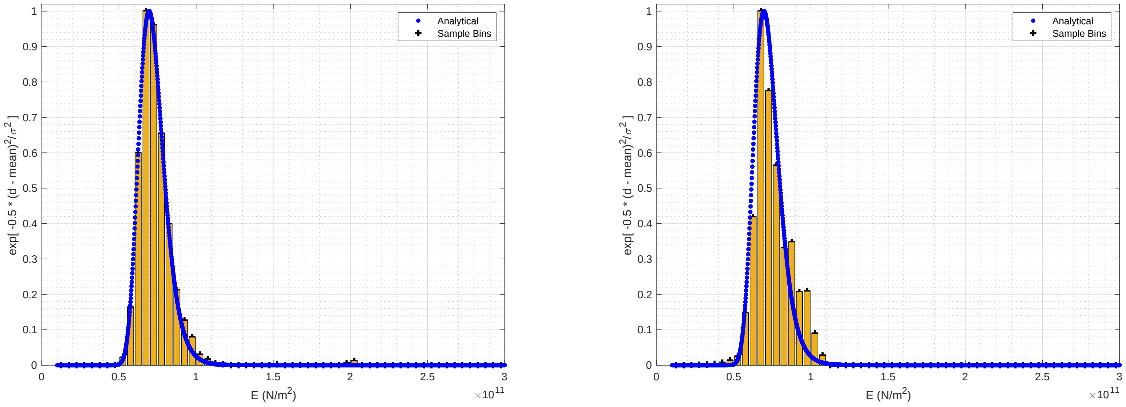

Figure 3: Results for the cantilever problem, using DRAM (left) and MALA (right). The histograms are normalized so that that maximum value is 1. We see that both histrograms follow well the true PDF, although the histogram obtained with DRAM follows the true PDF better, even though the amount of requested samples with DRAM was half the amount requested with MALA (see Table 2).

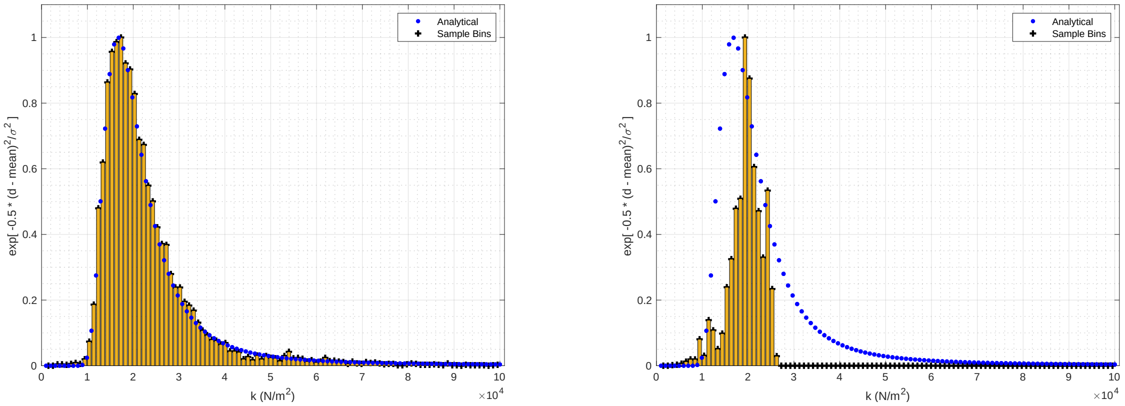

Figure 4: Results for the one-dimensional springs problem, using DRAM (left) and MALA (right). The histograms are normalized so that that maximum value is 1. We see that the histogram obtained with DRAM follows the true PDF much better. The MALA algorithm seems to get attracted to the "peak" region, having difficulty to explore the "tail" region.

6.3 Horizontal Sequence of Springs (Two-Dimensional Parameter Space)

We used $n = 10$, $n_{\text{known}} = 8$, all n_{known} known spring constants equal to $2.e4$, and $F = 1\ N$. The experimental data was mimicked as in Subsection 5.1. For prior PDF we used uniform $\mathcal{U}((1.e3, 1.e5)^2)$. Results and comments are presented in Figures 5 and 6.

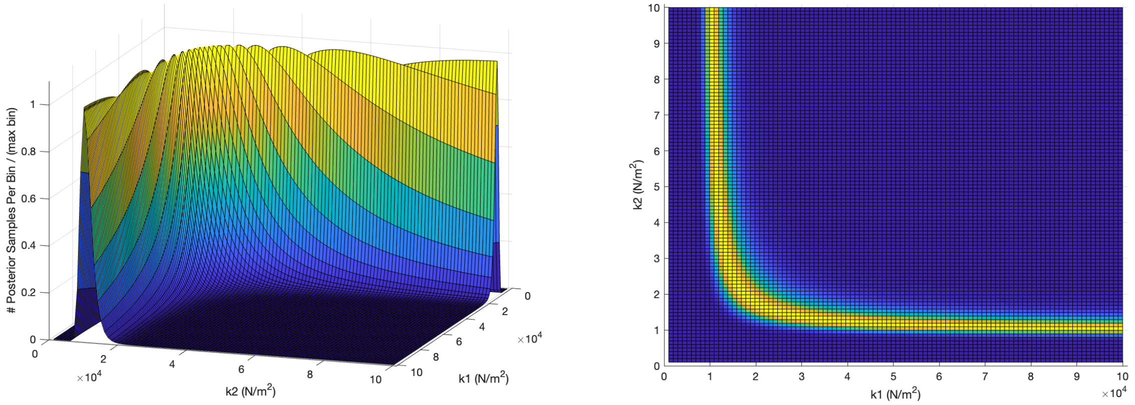

Figure 5: The two-dimensional surface of the analytical likelihood PDF in the two-dimensional springs problem. The two figures refer to different points of view of the same surface. We will refer the region of highest PDF values as the "peak" or "wing" region. As in the one-dimensional case (Figure 4), we see that the two-dimensional PDF is practically zero away from the peak region.

6.4 Horizontal Sequence of Springs (Three-Dimensional Parameter Space)

We used $n = 10$, $n_{\text{known}} = 7$, all n_{known} known spring constants equal to $2.e4$, and $F = 1\ N$. The experimental data was mimicked as in Subsection 5.1. For prior PDF we used uniform $\mathcal{U}((1.e3, 1.e5)^3)$. Results and comments are presented in Figure 7.

7 CONCLUSIONS

In this paper we presented Bayesian calibration results obtained with the Dakota 6.20 package and its interface to MUQ. Working with low-dimensional problems, before advancing to high-dimensional ones, helps researchers to better understand MCMC algorithms and gain an intuition about pros and cons of different methods. In our experiments, DRAM explored the posterior PDF surfaces better than MALA, which requires a careful study for the proper setting of the input parameters $\mathbf{M}$ and τ. As a final remark, we envision future versions of Dakota consistently adding python interfaces to its algorithms.

8 ACKNOWLEDGMENTS

The authors thank Mike Eldred, Russel Hooper, Moe Khalil, and Mathew Parno for technical discussions. This work was conducted at Sandia National Laboratories.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

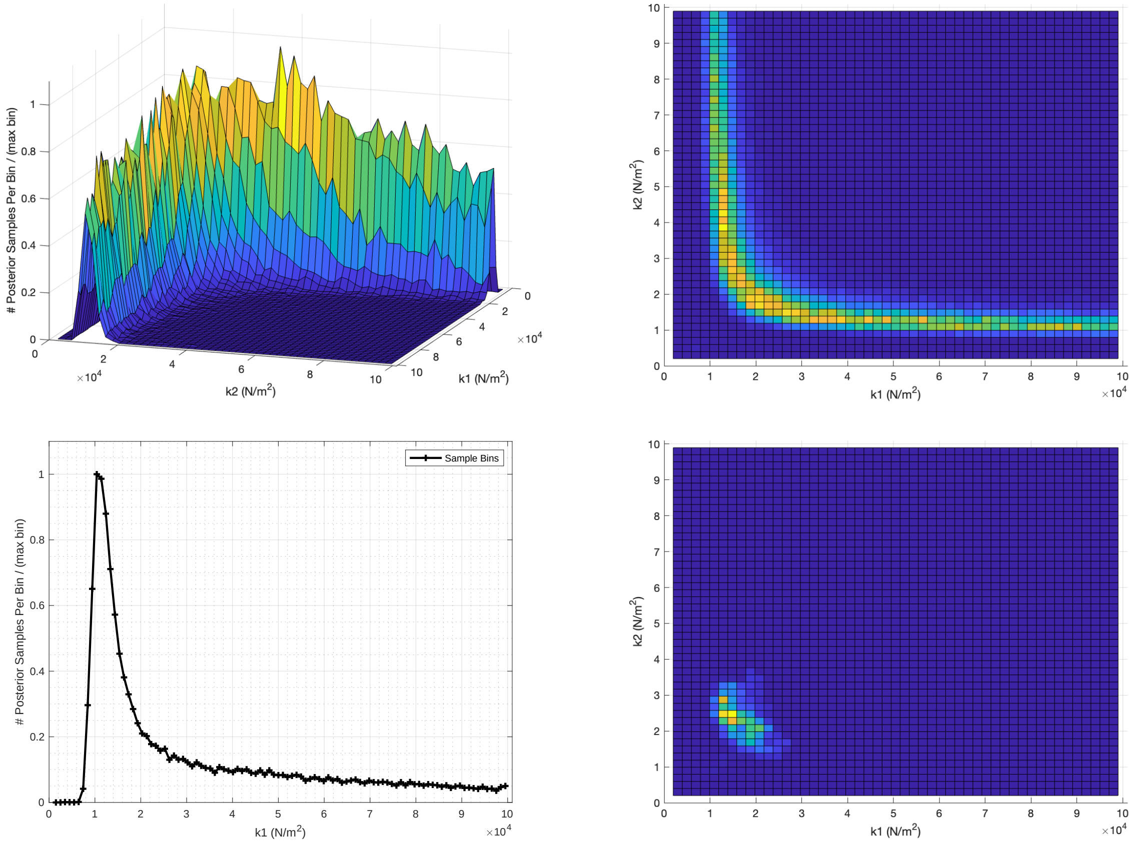

Figure 6: Results for the two-dimensional springs problem. The two figures on the top refer to different points of view of the two-dimensional histogram obtained with DRAM. Comparing them to Figure 5, we see that the two-dimensional histogram captures well the true PDF surface. The bottom-left figure is the marginal histogram for parameter k_1. Comparing to Figure 4, we see that the marginal PDF has a clear non-zero value away from the peak region, and that the peak region gets thinner and moves a bit closer to the origin. The bottom-right figure shows the two-dimensional histogram obtained with MALA. The same issue we observed in Figure 4 is more pronounced here: the algorithm seems to get attracted to the central portion of the "wing" region and has difficulty to explore its two sides.

REFERENCES

[1] Daniella Calvetti and Erkki Somersalo. *Introduction to Bayesian Scientific Computing: Ten Lectures on Subjective Computing.* Springer, 2007.

[2] Jari Kaipio and Erkki Somersalo. *Statistical and Computational Inverse Problems.* Springer, 2005.

[3] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods.* Springer, 2004.

[4] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng (Editors). *Handbook of Markov Chain Monte Carlo.* Chapman & Hall/CRC, 1st edition, 2011.

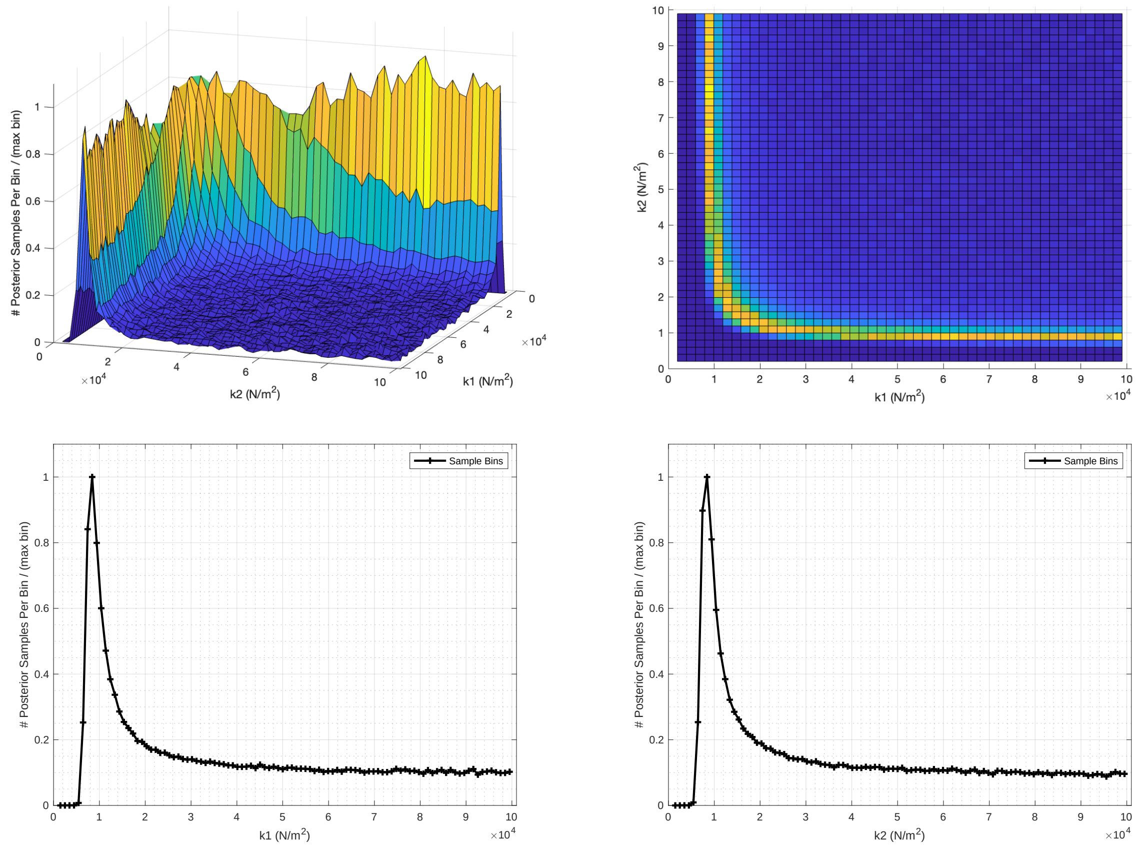

Figure 7: Results for the three-dimensional springs problem, all obtained with DRAM. The two figures on the top refer to different points of view of the two-dimensional joint marginal histogram for k_1 and k_2. Similarly to the differences between the histogram in Figure 4 and the 1D marginal histogram in Figure 6, we see that the 2D marginal histogram in this figure has a clear non-zero value away from the "wing" region, when compared to the 2D histogram in Figure 6. We also see that the "wing" region gets thinner and moves a bit closer to the origin. The two figures on the bottom are the 1D marginal histograms for parameter k_1 and k_2. The PDF values away from the peak region get larger than the marginal histogram in Figure 6, and the peak region gets even thinner and even closer to the origin.

[5] Marc C. Kennedy and Anthony O'Hagan. Bayesian Calibration of Computer Models. *Journal of the Royal Stat. Soc., Series B*, 63(3):425–464, 2001.

[6] Ernesto E. Prudencio and Sai Hung Cheung. Parallel Adaptive Multilevel Sampling Algorithms for the Bayesian Analysis of Mathematical Models. *International Journal for Uncertainty Quantification*, 2(3):215–237, 2012.

[7] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 06 1953.

[8] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[9] L. Tierney and A. Mira. Some adaptive Monte Carlo methods for Bayesian inference. *Statistics in Medicine*, 18:2507–2515, 1999.

[10] A. Mira. On Metropolis-Hastings algorithms with delayed rejection. *Metron*, LIX(3-4):231–241, 2001.

[11] H. Haario, E. Saksman, and J. Tamminen. An adaptive Metropolis algorithm. *Bernoulli*, 7:223–242, 2001.

[12] Heikki Haario, Marko Laine, Antonietta Mira, and Eero Saksman. DRAM: Efficient adaptive MCMC. *Stat Comput*, 16:339–354, 2006.

[13] P. J. Rossky, J. D. Doll, and H. L. Friedman. Brownian Dynamics as smart Monte Carlo simulation. *J. Chem. Physics*, 69(10):4628, 1978.

[14] Ulf Grenander and Michael I. Miller. Representations of Knowledge in Complex Systems. *J. R. Statist. Soc. B*, 56(4):549–581, 1994.

[15] G. O. Roberts and R. L. Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

[16] G. O. Roberts and J. S. Rosenthal. Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Stat. Soc., Series B*, 60(1):255–268, 1998.

[17] Tiangang Cui, Kody J.H. Law, and Youssef M. Marzouk. Dimension-independent likelihood-informed MCMC. *Journal of Computational Physics*, 304:109–137, 2016.

[18] B. M. Adams, W. J. Bohnhoff, K. R. Dalbey, M. S. Ebeida, J. P. Eddy, M. S. Eldred, R. W. Hooper, P. D. Hough, K. T. Hu, J. D. Jakeman, M. Khalil, K. A. Maupin, J. A. Monschke, E. E. Prudencio, E. M. Ridgway, P. Robbe, A. A. Rushdi, D. T. Seidl, J. A. Stephens, L. P. Swiler, and J. G. Winokur. Dakota 6.20.0 documentation. Technical Report SAND2024-05812O, Sandia National Laboratories, Albuquerque, NM, May 2024. Available online from `http://snl-dakota.github.io`.

[19] Ernesto E. Prudencio and Karl W. Schulz. The Parallel C++ Statistical Library 'QUESO': Quantification of Uncertainty for Estimation, Simulation and Optimization. In M. Alexander et. al., editor, *Euro-Par 2011: Parallel Processing Workshops*, pages 398–407, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[20] Matthew Parno, Andrew Davis, and Linus Seelinger. MUQ: The MIT Uncertainty Quantification Library. *Journal of Open Source Software*, 6(68):3076, 2021.