Chapter 2
Bayesian Finite Element Model Updating in the Modal Domain via Integration of Normalizing Flows with Markov Chain Monte Carlo

Lin Sun, Zhen Hu, and Michael D. Todd

Abstract Bayesian finite element (FE) model updating is a powerful framework for calibrating FE models by quantifying uncertainties in model parameters. Markov Chain Monte Carlo (MCMC) methods are commonly used to draw samples from the posterior distribution of these parameters. MCMC operates by accepting or rejecting proposed samples generated from a transition kernel that satisfies the detailed balance condition, ensuring convergence to the target distribution. However, conventional MCMC methods face several limitations: (1) they often require a large number of iterations to reach high-probability regions, (2) they may struggle to explore all modes in a multi-modal posterior distribution, and (3) they incur significant computational cost when evaluating the likelihood function for large-scale and/or nonlinear FE models. To address these challenges, the MCMC framework can be enhanced by integrating a machine learning technique known as normalizing flows (NFs). The resulting algorithm runs multiple Markov chains in parallel, with proposals alternately drawn from a standard Gaussian distribution and a learned NF-based distribution. We demonstrate the proposed method on a multi-degree-of-freedom (DOF) dynamic system, using modal properties (natural frequencies and mode shapes). Results indicate that the NF-enhanced MCMC approach achieves fast convergence and improved sampling accuracy. This study introduces a novel and effective strategy for improving the efficiency and robustness of Bayesian FE model updating through machine learning.

Keywords Bayesian finite element model updating · Normalizing flow · Markov Chain Monte Carlo

List of Notations and Abbreviations

Finite element: FE
Bayesian finite element model updating: BFEMU
Markov Chain Monte Carlo: MCMC
Normalizing flow: NF
Degree of freedom: DOF
Metropolis-Hastings: MH
Metropolis-adjusted Langevin algorithm: MALA
Kullback–Leibler: DL
D: measured response data
θ: unknown parameter vector
λ: eigenvalue
$\boldsymbol{\Phi}$: eigenvector / mode shape

Lin Sun · Michael D. Todd
Department of Structural Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, CA 92093, USA
e-mail: lsun@ucsd.edu; mdtodd@ucsd.edu

Zhen Hu
Department of Industrial and Manufacturing Systems Engineering,
University of Michigan-Dearborn, 4901 Evergreen Road, Dearborn, MI 48128, USA
e-mail: zhennhu@umich.edu

Kristin B. Zimmerman, PhD (eds.), *Model Validation and Uncertainty Quantification, 2026, Vol. 3*, Conference Proceedings of the Society for Experimental Mechanics Series, https://doi.org/10.1007/97887-438-1442-9_2

σ: standard deviation
N_m: number of modes
N_D: number of datasets
N_{chain}: number of Markov chains
N_{iter}: number of iterations
f: natural frequency
∂: scaling factor
$p_{base}(\boldsymbol{\alpha})$: base distribution
$F(\cdot)$: one-to-one mapping
$p(\boldsymbol{\beta})$: target distribution
k_{local}: switching step
$s(\cdot)$: scale function
$t(\cdot)$: translation function
$\boldsymbol{\alpha}(\cdot)$: acceptance probability
$L(\cdot)$: loss function
$\mathbf{w}$: weights
$\mathbf{b}$: bias
ε: learning rate

Introduction

The finite element (FE) method is a cornerstone of structural engineering, widely used to simulate the behavior of complex systems such as buildings, bridges, and mechanical assemblies [1, 2]. However, due to modeling assumptions, parameter uncertainties, and simplifications, discrepancies often arise between FE predictions and experimental or field measurements. Bayesian finite element model updating (BFEMU) has emerged as a powerful framework to calibrate model parameters and quantify uncertainty, providing more reliable and interpretable models [3].

Traditional BFEMU algorithms, such as Markov Chain Monte Carlo (MCMC) [4, 5], are effective but face notable limitations. When applied to large-scale and/or nonlinear models, they incur high computational costs due to repeated FE model evaluations. Additionally, MCMC samplers may require excessive iterations to reach high-probability regions—particularly when the prior is distant from the posterior—and often struggle with multi-modal posteriors. To address these challenges, researchers have begun integrating machine learning techniques into inference frameworks. One promising approach is normalizing flows (NFs), a class of invertible neural networks capable of learning complex probability distributions [6, 7]. While NF has shown success in general-purpose Bayesian inference, its application to BFEMU remains limited.

This study applies an NF-enhanced MCMC sampler to BFEMU in the modal domain. The method is evaluated on a simulated four-degree-of-freedom shear building subjected to white noise excitation. Modal properties—such as natural frequencies and mode shapes—can be obtained either by perturbing the "true" model parameters or through linear system identification methods [8, 9]. This work contributes a novel integration of machine learning and Bayesian inference for structural model calibration, offering a scalable and accurate alternative to conventional methods.

Bayesian Finite Element Model Updating in Modal Domain

BFEMU is commonly performed using Bayes' theorem:

$$p(\boldsymbol{\theta}|\mathbf{D}) = \frac{p(\mathbf{D}|\boldsymbol{\theta}) \times p(\boldsymbol{\theta})}{p(\mathbf{D})} \sim p(\mathbf{D}|\boldsymbol{\theta}) \times p(\boldsymbol{\theta}) \tag{1}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$ denotes the set of selected FE model parameters, and $\mathbf{D} \in \mathbb{R}^{n_D}$ represents the measured response data. In this expression, $p(\boldsymbol{\theta})$ is the prior distribution of the parameters, $p(\mathbf{D}|\boldsymbol{\theta})$ is the likelihood function, and $p(\mathbf{D})$ is the evidence term. The posterior distribution $p(\boldsymbol{\theta}|\mathbf{D})$ characterizes the updated belief about the model parameters after observing data $\mathbf{D}$. Since the evidence $p(\mathbf{D})$ is a normalization constant, the posterior is proportional to the product of the likelihood and the prior.

In this study, the data $\mathbf{D}$ used for BFEMU consists of modal properties, including natural frequencies and mode shapes. These modal parameters are typically obtained through system identification techniques. Assuming statistical independence

across N_D datasets and N_M modes, the likelihood function can be expressed as [10]:

$$p\left(\mathbf{D}_{1:N_D}|\boldsymbol{\theta}\right)=\prod_{i=1}^{N_D}\prod_{j=1}^{N_M}\frac{1}{\sqrt{2\pi}}\exp\left\{-\frac{e_{i,j,\lambda}^2}{2\sigma_{i,j,\lambda}^2}-\frac{\mathbf{e}_{i,j,\boldsymbol{\Phi}}^T\mathbf{e}_{i,j,\boldsymbol{\Phi}}}{2\sigma_{i,j,\boldsymbol{\Phi}}^2}\right\}, \tag{2}$$

where λ represents the eigenvalue (related to the natural frequency$f$ via $\lambda=(2\pi f)^2$), and $\boldsymbol{\Phi}$ denotes the eigenvector/mode shape. The terms $\sigma_{i,j,\lambda}$ and $\sigma_{i,j,\boldsymbol{\Phi}}$ denote the standard deviations associated with the eigenvalue and eigenvector errors, respectively. The eigenvalue error is defined as:

$$e_{i,j,\lambda}=\lambda_{i,j}^{\exp}-\lambda_j\left(\boldsymbol{\theta}\right), \tag{3}$$

where $\lambda_{i,j}^{\exp}$ is the experimentally estimated eigenvalue for the j^{th} mode from the i^{th} dataset, and $\lambda_j\left(\boldsymbol{\theta}\right)$ is the corresponding FE-predicted eigenvalue. The eigenvector error term defined as:

$$\boldsymbol{e}_{i,j,\boldsymbol{\Phi}}=\frac{\boldsymbol{\Phi}_{i,j}^{\exp}}{\left\|\boldsymbol{\Phi}_{i,j}^{\exp}\right\|}-\partial_j\frac{\boldsymbol{\Phi}_j\left(\boldsymbol{\theta}\right)}{\left\|\boldsymbol{\Phi}_j\left(\boldsymbol{\theta}\right)\right\|}, \tag{4}$$

where $\|\cdot\|$ denotes the L_2 norm. Here, $\boldsymbol{\Phi}_{i,j}^{\exp}$ is the experimental mode shape for the i^{th} dataset and j^{th} mode, and $\boldsymbol{\Phi}_j\left(\boldsymbol{\theta}\right)$ is the corresponding FE-predicted mode shape. To eliminate the effect of arbitrary scaling in the mode shapes, a scaling factor ∂_j is applied and defined as:

$$\partial_j=\frac{\boldsymbol{\Phi}_{i,j}^{\exp}\cdot\boldsymbol{\Phi}_j\left(\boldsymbol{\theta}\right)}{\left\|\boldsymbol{\Phi}_{i,j}^{\exp}\right\|\left\|\boldsymbol{\Phi}_j\left(\boldsymbol{\theta}\right)\right\|}. \tag{5}$$

This formulation ensures proper comparison between experimental and numerical mode shapes during the updating process.

Integration of Normalizing Flow with Markov Chain Monte Carlo

NFs are a class of generative models that transform a simple base distribution $p_{base}\left(\boldsymbol{\alpha}\right)$ into a complex target distribution $p\left(\boldsymbol{\beta}\right)$ using a sequence of invertible and differentiable mappings. This transformation is modeled as a one-to-one mapping function $\boldsymbol{\beta}=F\left(\boldsymbol{\alpha}\right)$. This formulation enables the use of the change-of-variables formula to compute the target distribution:

$$\hat{p}\left(\boldsymbol{\beta}\right)=p_{base}\left(\hat{F}^{-1}\left(\boldsymbol{\alpha}\right)\right)\left|\det\left(\nabla_{\boldsymbol{\alpha}}\hat{F}^{-1}\left(\boldsymbol{\alpha}\right)\right)\right|, \tag{6}$$

where $\hat{F}\left(\cdot\right)$ is an approximation of the optimal mapping $F^*\left(\cdot\right)$; $\hat{F}^{-1}\left(\cdot\right)$ is its inverse; $\nabla_{\boldsymbol{\alpha}}\hat{F}^{-1}\left(\boldsymbol{\alpha}\right)$ is denotes the Jacobian matrix of the inverse transformation, and $\det\left(\cdot\right)$ represents the determinant operator.

In practice, the design of the mapping $\hat{F}\left(\cdot\right)$ must enable efficient density evaluation, which requires computing Jacobian determinants and inverses in a tractable manner. To achieve this, coupling layers—known for their computational efficiency and structural simplicity—are often employed in NF architectures [6]. A key design objective in NF models is to ensure these computations remain easily tractable. Addressing these challenges has been a central focus in the literature [6].

Coupling layers typically adopt alternating patterns to construct the one-to-one transformation $\hat{F}\left(\cdot\right)$ [7]. Let $\boldsymbol{\alpha}^{(i)}\in\mathbb{R}^D$ and $\boldsymbol{\beta}^{(i)}\in\mathbb{R}^D$ denote the input and output vectors in the i^{th} coupling layer, respectively. The transformation between $\boldsymbol{\alpha}^{(i)}$ and $\boldsymbol{\beta}^{(i)}$ is defined as:

$$\begin{cases}\boldsymbol{\beta}_{1:d}^{(i)}=\boldsymbol{\alpha}_{1:d}^{(i)}\\ \boldsymbol{\beta}_{d+1:D}^{(i)}=\boldsymbol{\alpha}_{d+1:D}^{(i)}\odot\exp\left(s\left(\boldsymbol{\alpha}_{1:d}^{(i)}\right)\right)+t\left(\boldsymbol{\alpha}_{1:d}^{(i)}\right)\end{cases}, \tag{7}$$

where d partitions the input vector into two subsets, $s\left(\cdot\right)$ and $t\left(\cdot\right)$ are scale and translation functions, respectively, and $\odot$ denotes element-wise multiplication. Since the Jacobian matrix of the transformation in Eq. (7) is triangular, its determinant simplifies to:

$$\det\left(\frac{\partial\boldsymbol{\beta}^{(i)}}{\partial\boldsymbol{\alpha}^{(i),T}}\right)=\operatorname{diag}\left(\exp\left(s\left(\boldsymbol{\beta}_{1:d}^{(i)}\right)\right)\right). \tag{8}$$

The inverse transformation is also tractable:

$$\begin{cases} \boldsymbol{\alpha}_{1:d}^{(i)} = \boldsymbol{\beta}_{1:d}^{(i)} \\ \boldsymbol{\alpha}_{d+1:D}^{(i)} = \left(\boldsymbol{\beta}_{d+1:D}^{(i)} - t\left(\boldsymbol{\beta}_{1:d}^{(i)}\right)\right) \odot \exp\left(-s\left(\boldsymbol{\beta}_{1:d}^{(i)}\right)\right) \end{cases} . \tag{9}$$

By stacking n such coupling layers in alternating order, the full transformation can be constructed as (see Figure 1):

$$F = F_1 \circ \cdots \circ F_n. \tag{10}$$

The scale and translation functions in each layer are typically parameterized by neural networks. The model is trained by minimizing the Kullback–Leibler (KL) divergence between the transformed distribution and the target distribution. The objective function is given by:

$$D_{KL}\left(\hat{p}\left(\boldsymbol{\beta}\right)||p_*\left(\boldsymbol{\beta}\right)\right) = \int_{\Omega_{\boldsymbol{\beta}}} \left[-\log\left(p_*\left(\boldsymbol{\beta}\right)\right) + \log\left(\hat{p}\left(\boldsymbol{\beta}\right)\right)\right] \hat{p}\left(\boldsymbol{\beta}\right) d\boldsymbol{\beta} \ . \tag{11}$$

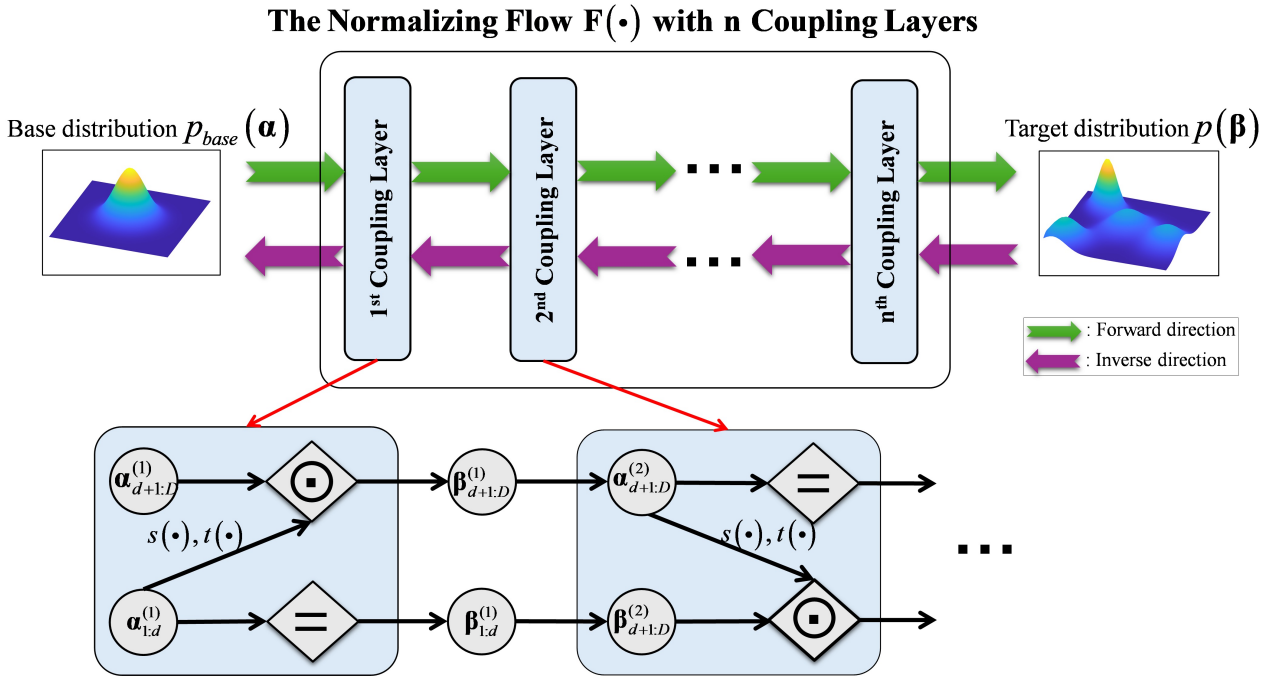

Fig. 1 Normalizing flow model with n alternating coupling layers

MCMC algorithms are widely used to generate samples from a target posterior distribution. In the context of BFEMU, the goal is to sample from $p\left(\boldsymbol{\theta}|\mathbf{D}\right)$. A central challenge in MCMC is the design of effective transition kernels $\pi\left(\boldsymbol{\theta},\ \boldsymbol{\theta}'\right)$ that enable efficient exploration of complex, high-dimensional distributions while maintaining detailed balance. Traditional MCMC methods such as Metropolis–Hastings (MH) [11] and Metropolis-adjusted Langevin algorithm (MALA) [12] are commonly employed. However, these methods can exhibit slow convergence when the posterior has multiple modes or nonlinear correlations. To address these limitations, recent approaches integrate NFs into MCMC frameworks, leveraging their capacity to learn expressive, invertible transformations that map simple base distributions to complex targets.

Gabrie et al. [13] proposed an alternating MCMC scheme that switches between a local Gaussian kernel and a global NF-based kernel. This hybrid design enables the sampler to maintain robustness during local exploration while benefiting from the expressivity of the NF in global moves. Crucially, proposals from the global kernel are independent of previous samples in the chain, leading to improved mixing and reduced autocorrelation.

Let N_{iter} denote the number of MCMC iterations and N_{chain} the number of parallel chains. The algorithm generates samples $\{\boldsymbol{\theta}_{i,j};\ i = 0,\ 1,\ ...,\ N_{chain};\ j = 0, 1, ..., N_{iter}\}$ from the posterior distribution $p\left(\boldsymbol{\theta}|\mathbf{D}\right)$. Each chain is initialized with $\{\boldsymbol{\theta}_{i,0};\ i = 0,\ 1,\ ...,\ N_{chain}\} \sim p\left(\boldsymbol{\theta}\right)$. At each iteration j, a decision is made to use either the global or local kernel based on a predefined schedule (e.g., alternating every k_{local} steps).

If the global kernel is selected, a base sample $\theta'_{base,i} \sim p_{base}(\theta)$ is drawn from a standard normal distribution and transformed via the NF mapping $\hat{F}(\cdot)$ to obtained the proposal θ'_i (see step 2 in Table 1). The proposed sample θ'_i is accepted with the acceptance probability:

$$\alpha_{global}\left(\theta_{i,j},\ \theta'_i\right) = \min\left(1,\ \frac{\hat{p}(\theta_{i,j})\, p\left(\theta'_i|\mathbf{D}\right)}{\hat{p}\left(\theta'_i\right) p(\theta_{i,j}|\mathbf{D})}\right), \tag{12}$$

where $\hat{p}(\theta)$ denotes the probability density estimated by the NF model. Notably, $\hat{p}(\theta)$ does not necessarily match the target distribution $p(\theta|\mathbf{D})$ as the NF model may not be optimal. If the local kernel is selected, the proposed sample θ'_i is drawn from the conditional Gaussian distribution (see step 5 in Table 1), and accepted with the standard MH probability $\alpha_{local}\left(\theta_{i,j},\ \theta'_i\right)$.

If the j^{th} iteration is completed, the NF model is updated by minimizing the negative log-likelihood loss $L(\mathbf{w},\ \mathbf{b})$ over samples, where $(\mathbf{w},\ \mathbf{b})$ are the trainable parameters of the scale and translation networks. The model is updated via gradient descent with learning rate ε. The training process ensures that future proposals from the NF model better approximate the target posterior distribution.

Table 1 General procedure for integrating NF with MCMC

input: $\left\{\boldsymbol{\theta}_{i,0}; i = 0,1,...,N_{chain}\right\}$, σ, ε

while $j < N_{iter}$:

  for $i = 1,...,N_{chain}$

    if j mod $k_{local} = 0$:

      1 : $\boldsymbol{\theta}'_{base,i} \sim p_{base}\left(\boldsymbol{\theta}\right)$

      2 : $\boldsymbol{\theta}'_i = \hat{F}\left(\boldsymbol{\theta}'_{base,i}\right)$

      3 : accept: $\boldsymbol{\theta}_{i,j+1} = \boldsymbol{\theta}'_i$ with probability $\alpha_{global}\left(\boldsymbol{\theta}_{i,j}, \boldsymbol{\theta}'_i\right)$

      4: otherwise: $\boldsymbol{\theta}_{i,j+1} = \boldsymbol{\theta}_{i,j}$

    else:

      5 : $\boldsymbol{\theta}'_i \sim N\left(\boldsymbol{\theta}_{i,j}, \sigma^2\mathbf{I}\right)$

      6 : accept: $\boldsymbol{\theta}_{i,j+1} = \boldsymbol{\theta}'_i$ with probability $\alpha_{local}\left(\boldsymbol{\theta}_{i,j}, \boldsymbol{\theta}'_i\right)$

      7: otherwise: $\boldsymbol{\theta}_{i,j+1} = \boldsymbol{\theta}_{i,j}$

  8 : evaluate loss function: $L\left(\mathbf{w},\mathbf{b}\right) = -\frac{1}{N_{chain}}\sum_{i=1}^{N_{chain}} \log\left(\hat{p}\left(\boldsymbol{\theta}_{i,j+1}\right)\right)$

  9 : update the mapping: $\left(\mathbf{w},\mathbf{b}\right) = \left(\mathbf{w},\mathbf{b}\right) - \varepsilon\nabla L\left(\mathbf{w},\mathbf{b}\right)$

$j = j+1$

return $F(\cdot)$, $\left\{\boldsymbol{\theta}_{i,j}; i = 0,1,...,N_{chain}; j = 0,1,...,N_{iter}\right\}$

Case Study: Numerically Simulated 3-DOF Dynamic System

Figure 2(a) illustrates a numerically simulated three-degree-of-freedom (3-DOF) dynamic system. The nominal story mass is 625 tons, and the nominal story stiffness is 1×10^9 N/m. In the context of structural health monitoring, story stiffness is of particular interest, as it reflects the condition and potential damage within a building. To account for possible structural changes, the story stiffness is treated as an unknown parameter vector θ:

$$\theta = \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix}^T. \tag{13}$$

Eigenvalues and mode shapes are extracted via eigen-analysis from the system's stiffness and mass matrices. To generate modal data for BMU, the "true" eigenvalues and mode shapes are computed using the nominal mass and stiffness matrices and a set of "true" parameter values θ^{true}. For numerical stability, each eigenvalue is normalized by the largest eigenvalue (corresponding to the third mode), and each mode shape is normalized by its maximum nodal displacement. To simulate measurement noise, 100 datasets are generated by perturbing the "true" eigenvalues and mode shapes with white noise. The eigenvalues are perturbed with Gaussian noise having a standard deviation equal to 5% of their corresponding "true" values. Similarly, each nodal value in the mode shapes is perturbed with Gaussian noise with a standard deviation of 5% of its respective normalized value (see Figure 2(b) and (c)). The prior distribution for the parameters is assumed to be Gaussian, with a mean equal to 1.5 times the corresponding "true" value, a coefficient of variation of 20%, and no correlation among parameters.

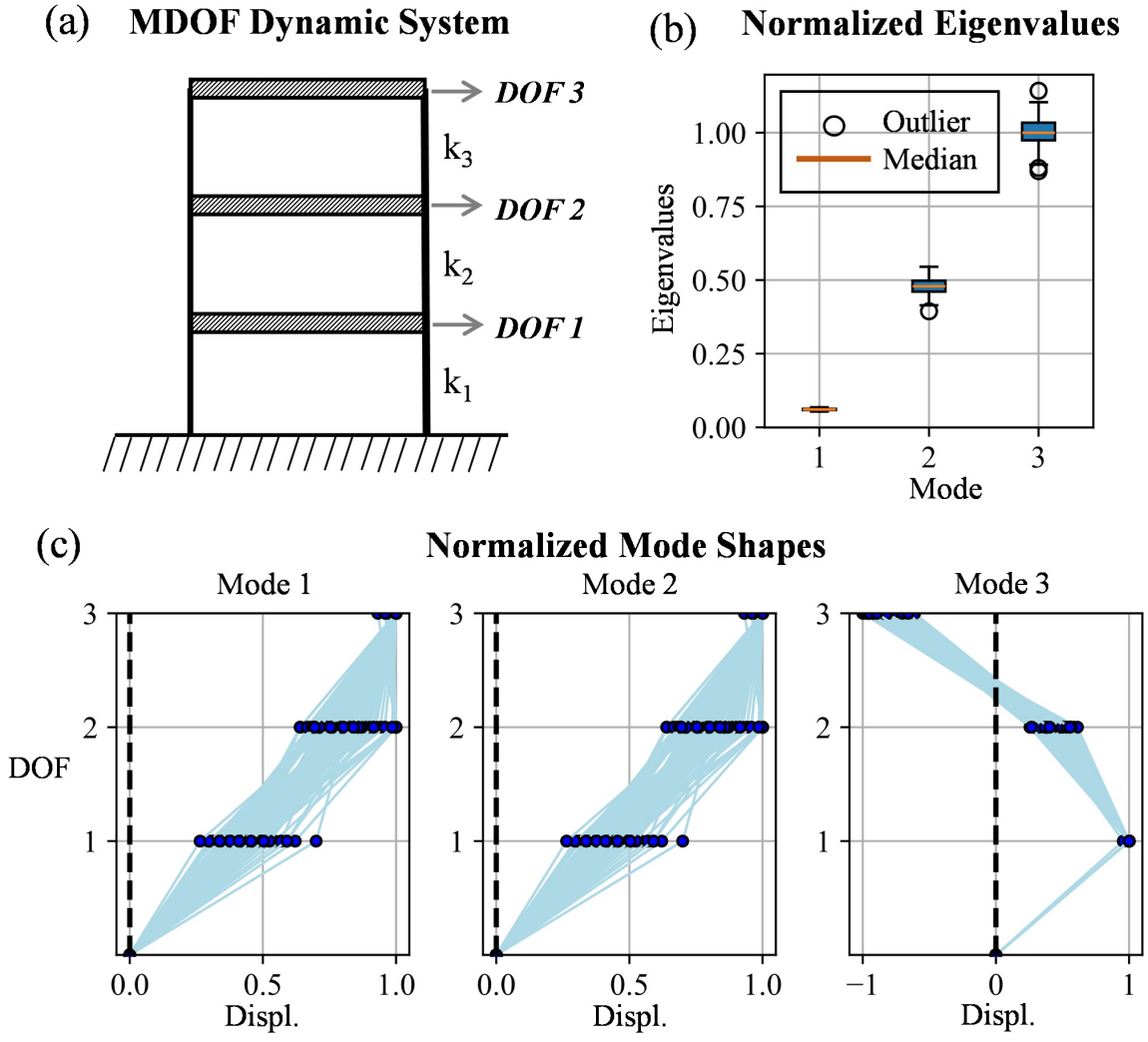

Fig. 2 (a) Schematic of the 3-DOF dynamic system; (b) normalized eigenvalues; (c) normalized mode shapes

We run three Markov chains using the MCMC algorithm integrated with the NF. The initial positions of each Markov chain are drawn from the prior distribution of the parameters. The switching step k_{switch} is set to 2, and the total number of iterations for each chain is set to 4000. This means the transition kernel switches between the local and global kernels at each iteration. Specifically, 2000 iterations use the local kernel, and the other 2000 iterations use the global kernel for each chain.

Figure 3(a) displays the 3D trace plot of the three Markov chains in the parameter space (k_1, k_2, k_3). The red stars represent the initial positions of the chains, while the red dot indicates the "true" parameter values. The plot shows that after a sufficient number of iterations, all three chains converge toward the true parameter values, i.e., the high-probability region.

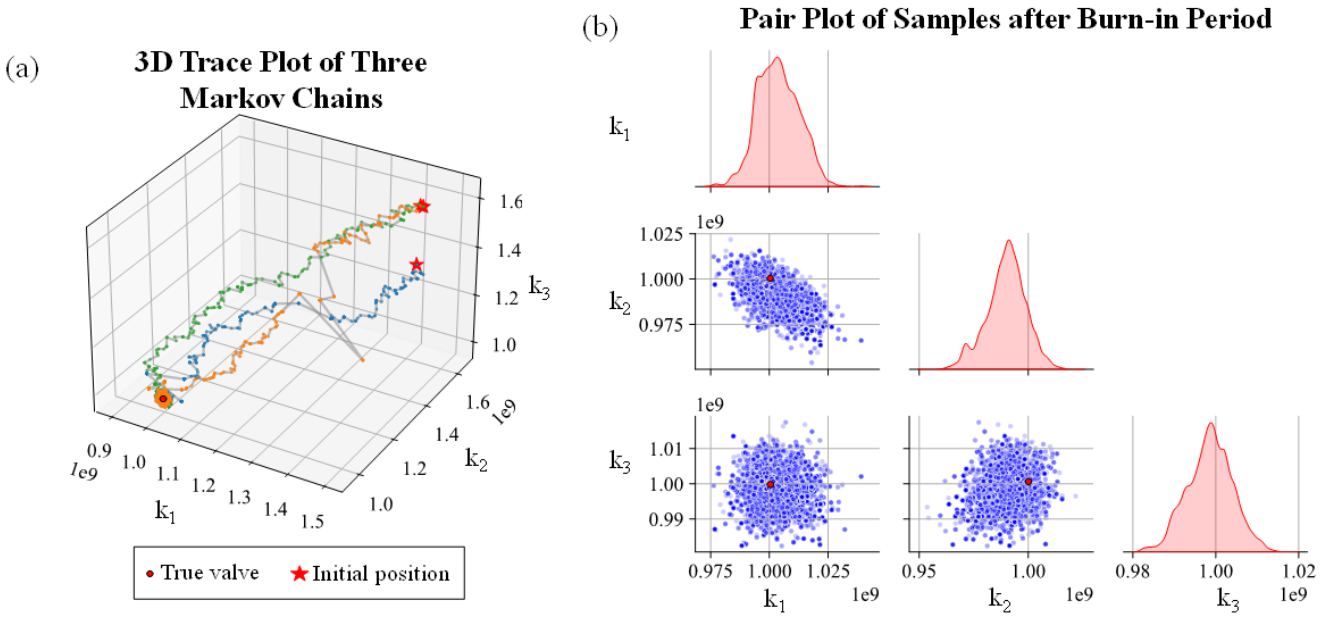

Fig. 3 3D trace plot of the three Markov chains in the parameter space (k_1, k_2, k_3); (b) pair plot of the samples after the burn-in period

After discarding the burn-in period for each chain, the collected samples are shown in the pair plot in Figure 3(b). It is observed that the sample pair (k_1, k_2) exhibits a weak negative correlation, while the pairs (k_1, k_3) and (k_2, k_3) show minimal correlation. Furthermore, the marginal distributions of k_1, k_2 and k_3 indicate that the samples are concentrated near their respective true values.

Conclusions

In this study, we applied a Markov Chain Monte Carlo (MCMC) algorithm integrated with a normalizing flow (NF) to perform Bayesian inference on a 3-degree-of-freedom (3-DOF) dynamic system in modal domain. The three story stiffnesses are elected as parameters to update. By running three independent Markov chains with a total of 4000 iterations each—switching between a local kernel and a global kernel—we observed robust convergence behavior in the parameter space.

The 3D trace plot demonstrated that all chains effectively converged to the high-probability region corresponding to the "true" parameter values. After discarding the burn-in period, the pair plot revealed a slight negative correlation between k_1 and k_2, while (k_1, k_3) and (k_2, k_3) showed minimal correlation. Moreover, the marginal distributions of each parameter were sharply peaked around the true values, confirming the accuracy and efficiency of the inference. These results highlight the effectiveness of the NF-enhanced MCMC framework in capturing complex posterior distributions and ensuring rapid convergence in multi-parameter settings.

Acknowledgments Funding for this work by the U.S. Army Corps of Engineers through the U.S. Army Engineer Research and Development Center Research Cooperative Agreement W9132T-22-20014 is gratefully acknowledged. The opinions and findings in this study are those of the authors and do not necessarily reflect the views of the sponsor.

References

1. K.J. Bathe, Finite Element Procedures, Prentice Hall, 2007.
2. R.D. Cook, D.S. Malkus, M.E. Plesha, R.J. Witt, Concepts and Applications of Finite Element Analysis, 4th ed., Wiley, 2001.
3. T. Marwala, L. Mdlazi, S. Sibisi, "Finite element model updating using Bayesian approach," 2007.
4. W.K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," Biometrika 57 (1970) 97–109. https://doi.org/10.1093/biomet/57.1.97.

5. J. Ching, Y.C. Chen, "Transitional Markov Chain Monte Carlo method for Bayesian model updating, model class selection, and model averaging," J. Eng. Mech. 133 (2007) 816–832. https://doi.org/10.1061/(ASCE)0733-9399(2007)133:7(816).
6. G. Papamakarios, E. Nalisnick, D.J. Rezende, S. Mohamed, B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," 2019. arXiv:1912.02762.
7. L. Dinh, J. Sohl-Dickstein, S. Bengio, "Density estimation using Real NVP," 2016. arXiv:1605.08803.
8. L. Sun, Linear System Identification and Bayesian Finite Element Model Updating of Civil Structural and Substructural Systems, Ph.D. Dissertation, University of California, San Diego, 2024. https://escholarship.org/uc/item/2090j6fz
9. L. Sun, J.P. Conte, M.D. Todd, R. Astroza, Y. Bock, G. Offield, F. Vernon, "Linear system identification of the UC San Diego Geisel Library building under ambient vibration," J. Civ. Struct. Health Monit. (2024). https://doi.org/10.1007/s13349-024-00889-4.
10. M. Song, L. Renson, J. Noël, B. Moaveni, G. Kerschen, "Bayesian model updating of nonlinear systems using nonlinear normal modes," Struct. Control Health Monit. 25 (2018) e2258. https://doi.org/10.1002/stc.2258.
11. N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, "Equation of state calculations by fast computing machines," J. Chem. Phys. 21 (1953) 1087–1092. https://doi.org/10.1063/1.1699114.
12. G. Roberts, R. Tweedie, "Exponential convergence of Langevin distributions and their discrete approximations," Bernoulli 2 (1996) 341–363.
13. M. Gabrié, G.M. Rotskoff, E. Vanden-Eijnden, "Adaptive Monte Carlo augmented with normalizing flows," Proc. Natl. Acad. Sci. U.S.A. 119 (2022) e2109420119. https://doi.org/10.1073/pnas.2109420119.