Optimal-Transport Flow Matching for Stochastic Model Updating: A Comparison with Transitional MCMC-based Bayesian Updating

Tairan Wang and Sifeng Bi
Department of Aeronautics and Astronautics Engineering, University of Southampton, Southampton, SO16 7QF, United Kingdom, tairan.wang@soton.ac.uk, sifeng.bi@soton.ac.uk

Abstract: Deep generative models have attracted considerable attention as inference engines for stochastic model updating. Among these deep generative models, conditional flow matching offers a particularly efficient formulation, in which a continuous transport map from a latent distribution to the posterior is learned by regression against optimal-transport probability paths, avoiding both the invertibility constraints of normalising flows and the iterative denoising of diffusion models. However, its performance in engineering model updating has not been examined against an established baseline such as the Bayesian updating framework under controlled conditions. In this work, conditional flow matching is therefore assessed against the Bayesian model updating approach with transitional Markov chain Monte Carlo, in which the Bhattacharyya distance provides the approximate likelihood for the Bayesian approach. The two methods are assessed with respect to posterior fidelity, evaluated by statistical distances against the known ground truth and computational expenditure in model evaluations, with the offline training budget reported separately from the inference cost of each subsequent conditioning. A 3-DOF spring–mass system with uncertain stiffness parameters serves as the benchmark. Since the calibrated posterior constitutes the input description upon which subsequent reliability assessment depends, the accuracy and the marginal cost of the updating step both bear directly on reliability computations. In addition, the amortised nature of the learned transport map is discussed in this work, as it permits the posterior to be re-evaluated under new observations at negligible cost, which is a property of practical relevance to reliability assessment under sequentially acquired monitoring data.

Keywords: Stochastic model updating, Flow matching, Bayesian inference, Uncertainty quantification.

1. Introduction

Numerical models of engineering structures invariably contain parameters that cannot be determined directly from design information, and whose values should instead be inferred from measured responses. In practice, these parameters are not merely unknown but genuinely variable. Manufacturing tolerances, material scatter and assembly conditions introduce aleatory variability across nominally identical specimens, while limited and imprecise observations leave the description of that variability itself epistemically uncertain. Stochastic model updating addresses this issue by treating the parameters as random variables and inferring the parameters of their distributions, rather than single deterministic values (Mares et al., 2006). The resulting distributional description is not an end in itself; it constitutes the input specification upon which any subsequent reliability computation is founded, so that both the fidelity and the cost of the updating step propagate directly into reliability practice (Sankararaman and Mahadevan, 2015).

The Bayesian model updating framework is recognised as the standard solution for the stochastic model updating problem. However, within the Bayesian formulation, the difficulty is that an explicit likelihood is rarely available (Beck and Katafygiotis, 1998, Bi et al., 2023). When the observations take the form of a scattered sample of responses rather than a single measurement, the comparison to be made is between two distributions, and the conventional geometric distance between two points is no longer appropriate. Approximate Bayesian computation circumvents this by replacing the likelihood with a kernel of a statistical distance between simulated and observed response sets. Among the candidate distances, the Bhattacharyya distance has been widely adopted in this context, since it measures the overlap of the two distributions across the full joint response space and thereby retains information on correlation that marginal comparisons discard (Bi et al., 2019a, Bi et al., 2019b, Kitahara et al., 2022). Sampling from the resulting posterior is most commonly performed by transitional Markov chain Monte Carlo (Ching and Chen, 2007), which traverses a sequence of tempered intermediate distributions and is robust to the flat and multimodal surfaces that distance-based likelihoods frequently exhibit. However, this cost cannot be amortised across datasets, because the tempering path is constructed for one specific set of observations and retains no reusable structure, each new measurement set requires the procedure to be executed in full, and the total computational expenditure therefore scales linearly with the number of updating instances.

Deep generative models redistribute this cost from inference to training. Rather than sampling a posterior that is specified implicitly, a conditional generative model is trained on simulated parameter–response pairs to represent the mapping from observations to posterior directly, and each subsequent update is obtained by integrating the learned generative model, which requires only inexpensive network evaluations and no further calls to the forward model. This is the perspective of simulation-based inference and neural posterior estimation, whose defining feature is amortisation, where the training cost is incurred only once, and after which any new observation is processed at negligible marginal cost (Papamakarios and Murray, 2016, Cranmer et al., 2020). Conditional normalising flows provide exact density amortised inference but require invertible architectures and tractable Jacobians, which constrain their expressivity (Papamakarios et al., 2021, Wang et al., 2025). Conditional denoising diffusion models remove these constraints and generate high-quality posteriors, but at the cost of integrating a reverse-time stochastic process over discrete steps (Ho et al., 2020, Song et al., 2020, Wang and Bi, 2026). Flow matching has more recently been proposed as a formulation that avoids the principal drawbacks of each: a continuous vector field transporting a reference distribution to the target is learned by regression against a prescribed conditional probability path (Lipman et al., 2022), so that training requires neither the invertible architectures and tractable Jacobians of normalising flows nor the long iterative denoising trajectories of diffusion models. When the probability path is chosen as the displacement interpolation between the coupled endpoints, the induced trajectories are close to straight, and accurate samples may be generated with a small number of integration steps (Liu et al., 2022, Tong et al., 2023).

Despite this progress, the relative merits of generative and sampling-based inference for stochastic model updating remain difficult to assess, as the two families have rarely been examined under comparable conditions. The present work addresses this gap by examining optimal-transport conditional flow matching against transitional Markov chain Monte Carlo on a common benchmark, in which the Bhattacharyya distance provides the approximate likelihood and the same distance is subsequently used as a common metric to quantify the posterior-predictive performance. The optimal-transport conditional flow matching is formulated for stochastic model updating in second-order form, so that the means and standard deviations of the uncertain parameters are calibrated jointly. The two methods are then assessed on equal terms with respect to posterior fidelity, measured against the known ground-truth uncertainty model and by cross-comparison of the two

posteriors, and with respect to computational expenditure, reported in both model evaluations and time consumption. The calibration performance is further quantified in the term of response prediction.

The remainder of the paper is organised as follows. Section 2 sets out the stochastic model updating problem and its Bayesian solution, comprising the Bhattacharyya-distance approximate likelihood and the transitional Markov chain Monte Carlo sampler. Section 3 develops the optimal-transport conditional flow matching framework, covering the conditional probability path, the training objective, and the resulting amortised updating procedure. Section 4 applies both methods to a three-degree-of-freedom spring–mass benchmark and compares them in terms of posterior fidelity and computational cost. Section 5 concludes the paper and outlines directions for future work.

2. Stochastic model updating and the Bayesian framework

2.1 Bayesian model updating problem

A physical system is represented by a numerical model $\mathbf{y}_{\text{sim}} = \mathcal{M}(\mathbf{x})$, in which $\mathbf{x} \in \mathbb{R}^{n_x}$ collects the physical input parameters and $\boldsymbol{y}_{\text{sim}} \in \mathbb{R}^{n_y}$ the output features. Under the multilevel formulation adopted here, the input parameters are aleatory and are governed entirely by a vector of hyperparameters $\boldsymbol{\theta}$ comprising the statistical moments of their assumed distribution family, i.e., $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$,. The hyperparameters possess distinct but unknown values and thus carry the epistemic uncertainty of the problem, and they constitute the quantity to be calibrated. The observations form a sample set $\mathbf{Y_{obs}} \in \mathbb{R}^{N_{\text{obs}} \times n_y}$ rather than a single measurement, and the updating problem is to infer $\boldsymbol{\theta}$ such that the simulated feature set $\mathbf{Y_{sim}}(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\text{sim}} \times n_y}$ reproduces the $\mathbf{Y_{obs}}$ in distribution. Following Bayes' theorem,

$$p(\boldsymbol{\theta}|\mathbf{Y_{obs}}) \propto P_L(\mathbf{Y_{obs}}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{1}$$

with $p(\boldsymbol{\theta})$ a prior over the admissible intervals, $P_L(\mathbf{Y_{obs}}|\boldsymbol{\theta})$ refers to the likelihood, and $p(\boldsymbol{\theta}|\mathbf{Y_{obs}})$ is the posterior. Two routes to this posterior are contrasted in this work, including the classical Bayesian route, in which an approximate likelihood is constructed and sampled with the transitional Markov chain Monte Carlo (TMCMC), and the flow matching route developed in Section 3, in which the mapping from observations to posterior is learned directly.

2.2 Bhattacharyya distance-based approximate likelihood and transitional Markov chain Monte Carlo

Since the observations are scattered, the exact likelihood requires a density estimate for every candidate $\boldsymbol{\theta}$ and is prohibitive. Approximate Bayesian Computation (ABC) replaces it with a kernel of a statistical distance, for instance, the Bhattacharyya distance as illustrated in Eq. (2),

$$P_L(\mathbf{Y_{obs}}|\boldsymbol{\theta}) \propto \exp\left\{-\frac{d_{\text{BD}}^2\left(\mathbf{Y_{obs}}, \mathbf{Y_{sim}}(\boldsymbol{\theta})\right)}{\varepsilon^2}\right\} \tag{2}$$

in which ε is the width factor controlling the concentration of the posterior. The Bhattacharyya distance is adopted as the metric $d_{\text{BD}}(\cdot)$, evaluated by the binning algorithm in (Bi et al., 2019a, Bi et al., 2021) over a partition of the joint feature space common to both sample sets. It has been demonstrated that this metric measures the overlap of the two distributions and is therefore sensitive to dispersion and dependence as well as to location, whereas the Euclidean distance responds only to the centre of mass. An equally relevant

limitation is that the distance saturates when the two sample sets cease to overlap, so the surface is uninformative far from the solution.

Sampling from the resulting posterior is performed by transitional Markov chain Monte Carlo (TMCMC), which traverses a sequence of tempered intermediate distributions. The exponent β_j is increased adaptively so that the coefficient of variation of the plausibility weights remains bounded, with resampling and Metropolis–Hastings propagation at each stage. The tempering is what renders the flat distance surface tractable.

$$p_j(\boldsymbol{\theta}) \propto P_L(\mathbf{Y_{obs}}|\boldsymbol{\theta})^{\beta_j} p(\boldsymbol{\theta}), \qquad 0 = \beta_0 < \cdots < \beta_m = 1 \tag{3}$$

It is noted that each likelihood call requires N_{sim} evaluations of $\mathcal{M}(\cdot)$, so the cost of a single update is the product of the population size, the number of proposals per chain, and the number of tempering stages. And the tempering sequence is constructed for one dataset alone, which means a new set of observations requires the whole path to be traversed again, so the total cost grows in proportion to the number of updates performed.

3. Optimal-transport flow matching for stochastic model updating

3.1 Flow matching

The flow matching route bypasses the likelihood evaluation entirely. Instead of evaluating how well a candidate $\boldsymbol{\theta}$ reproduces the observations, a transformation is sought that carries a simple latent distribution onto the posterior itself, conditioned on the observed data. This is the principle underlying the flow-based approaches previously applied to model updating, where a conditional invertible network learns a bijection between the parameter space and a latent Gaussian (Wang et al., 2025).

Flow matching realises the same principle without requiring bijectivity. Rather than composing invertible layers with tractable Jacobians, a continuous-time velocity field is learned, and the transformation is obtained by integrating it. Let $\boldsymbol{\theta_0} \sim \mathcal{N}(0, \mathbf{I})$ denote a reference sample and $\boldsymbol{\theta_1}$ a sample of the target. A field $v_\phi(\boldsymbol{\theta_t}, t, \boldsymbol{s})$, conditioned on a summary $\boldsymbol{s}$ of the observations, defines the flow as in Eq. (4). Integrating this equation from $t = 0$ to $t = 1$ carries a sample drawn from the base distribution $\boldsymbol{\theta_0}$ to a sample of the posterior $p(\boldsymbol{\theta}|\boldsymbol{s})$. Neither an architectural constraint nor a Jacobian determinant is involved, and the change-of-variable rule is not evaluated at any point.

$$\frac{\mathrm{d}\boldsymbol{\theta_t}}{\mathrm{d}t} = v_\phi(\boldsymbol{\theta_t}, t, \boldsymbol{s}), \qquad t \in [0,1] \tag{4}$$

3.2 Conditional probability paths and training objective

Flow matching constructs the field by regression against a prescribed conditional probability path, which is a family of distributions interpolating, for a single target sample, between the reference and that sample. The conditional probability path adopted here is expressed as,

$$p_t(\boldsymbol{\theta}|\boldsymbol{\theta_1}) = \mathcal{N}(\boldsymbol{\theta}|t\boldsymbol{\theta_1}, (1-t)^2\mathbf{I}), \qquad t \in [0,1] \tag{5}$$

whose mean and standard deviation both vary linearly in t, so that p_0 is the standard Gaussian reference ($p_0 \sim \mathcal{N}(0, \mathbf{I})$) and p_1 concentrates on $\boldsymbol{\theta_1}$. This particular choice is the optimal-transport displacement interpolation between the two endpoints. Among all paths connecting them, it is the one along which the samples move in

straight lines at constant speed, which is what makes it the optimal-transport path. Samples of the path and the velocity generating it take the elementary form

$$\boldsymbol{\theta_t} = (1-t)\boldsymbol{\theta_0} + t\boldsymbol{\theta_1}, \qquad \mathbf{u_t}(\boldsymbol{\theta_t}|\boldsymbol{\theta_1}) = \boldsymbol{\theta_1} - \boldsymbol{\theta_0} \tag{6}$$

with $\boldsymbol{\theta_0} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The conditional velocity is constant in t, in contrast with the diffusion paths of denoising models, whose trajectories curve and whose velocity varies sharply near the endpoints.

The marginal field of interest is recovered by regressing against these conditional velocities. The parameters follow from the expectation being estimated by minibatch sampling over the training set and over t and $\boldsymbol{\theta_0}$, denoted in Eq. (7).

$$\mathcal{L}(\phi) = \mathbb{E}_{t\sim\mathcal{U}(0,1),(\boldsymbol{\theta_1},\boldsymbol{s}),\boldsymbol{\theta_0}\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \left\| v_\phi(\boldsymbol{\theta_t}, t, \boldsymbol{s}) - (\boldsymbol{\theta_1} - \boldsymbol{\theta_0}) \right\|^2 \tag{7}$$

Although the regression targets are defined conditionally on individual samples, the minimiser of this objective is the marginal field transporting the reference distribution onto the posterior, and the conditional construction serves only to render the target computable.

The objective is simulation-free that no trajectory is integrated at training time, and the loss is an ordinary least-squares regression, in contrast with the maximum-likelihood objective of invertible architectures, which requires the Jacobian log-determinant at every layer. And because the conditional paths are straight and traversed at constant speed, the learned marginal field is close to straight wherever the paths do not intersect, so that accurate samples are obtained by explicit integration with a modest number of steps.

3.3 Flow matching-based model updating framework

The proposed flow matching-based framework casts stochastic model updating as an amortised, simulation-based inference task. Rather than repeatedly evaluating a likelihood to explore the parameter space for every new set of measurements, a single conditional generative network is trained once on simulated data and can thereafter infer the updated parameters of any new dataset almost instantaneously. The framework comprises three stages, including the generation of training data, the training of the network, and the posterior inference.

In the first stage, the training data is generated entirely by simulation. Because the updating is second order, the parameters to be calibrated are the means and standard deviations of the uncertain parameters. Each training instance is formed by drawing one hyperparameter vector $\boldsymbol{\theta}$ from the prior distribution and $\boldsymbol{\theta}$ defines a distribution of the physical parameters $p(\mathbf{x}|\boldsymbol{\theta})$, from which a population of samples is drawn and propagated through the numerical model to yield the corresponding responses. The resulting responses are then condensed into a compact feature vector $\boldsymbol{s}$, their means, standard deviations and correlations, which forms the input paired with the hyperparameters $\boldsymbol{\theta}$ to form one training sample $(\boldsymbol{\theta}, \boldsymbol{s})$. Repeating this for many draws yields the training database, which is standardised before training.

In the second stage, the network is trained to reproduce the relationship between a population of responses and the hyperparameter vector $\boldsymbol{\theta}$ that generated it. The network gradually transforms samples drawn from a simple standard distribution into descriptor samples consistent with a given response feature, with that feature supplied as an additional input throughout the transformation. Since the response feature enters as a conditioning input rather than being fixed, one trained network applies to every possible dataset, and the cost of running the numerical model is incurred only once, during the construction of the training database.

In the third stage, the trained network performs the actual updating. For a given set of measured responses, the corresponding feature vector is formed and supplied to the network, which transforms a large number of samples from the base distribution into samples of the updated hyperparameter vector $\boldsymbol{\theta}$. Collecting these samples reconstructs the full posterior distribution, from which the updated means, the dispersions and the

joint dependence between parameters are obtained. As the numerical model is not called again at this stage, the updating of any new measurement is completed in a single, near-instantaneous pass of the trained network.

4. Case study: 3 DOF spring-mass system benchmark

4.1 Problem description

The benchmark is the classical three-degree-of-freedom spring–mass system shown in Figure 1. Three masses are connected to one another and to ground by six springs. The mass and stiffness matrices are illustrated as in Eq. (8) and Eq. (9).

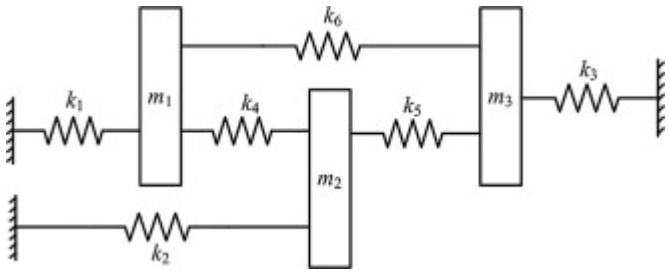

Figure 1. The 3 DOF spring-mass system.

$$\mathbf{M} = \mathrm{diag}(m_1, m_2, m_3) \tag{8}$$

and

$$\mathbf{K} = \begin{bmatrix} k_1 + k_4 + k_6 & -k_4 & -k_6 \\ -k_4 & k_2 + k_4 + k_5 & -k_5 \\ -k_6 & -k_5 & k_3 + k_5 + k_6 \end{bmatrix} \tag{9}$$

In the system, the masses and the three coupling stiffnesses are treated as known constants, $m_1 = 0.7$kg, $m_2 = 0.5$kg, $m_3 = 0.3$kg and $k_4 = k_5 = k_6 = 5.0$N/m, and require no updating. The remaining stiffnesses k_1, k_2, and k_3 are the uncertain input parameters, $\mathbf{x} = [k_1, k_2, k_3]^T$.

The output features are the three natural frequencies, obtained from the eigenproblem and collected in ascending order as $\mathbf{y} = [\omega_1, \omega_2, \omega_3]^T$.

$$|(\mathbf{K} - \omega_r^2 \mathbf{M})| = 0, \qquad r = 1,2,3 \tag{10}$$

The simulator $\mathcal{M}(\cdot)$ is thus a single eigensolution, and one model evaluation is defined throughout as one such solution for one realisation of $\mathbf{x}$. Its low cost is what allows both methods to be run at large simulation budgets without training a surrogate. Computational expenditure is accordingly reported primarily in the number of model evaluations and the time consumption.

4.2 Uncertainty characterisation and synthetic observations

The three uncertain stiffnesses are modelled as independent Gaussian random variables whose means and standard deviations are both unknown, so that the system involves aleatory and epistemic uncertainty. The

hyperparameter vector to be calibrated is accordingly six-dimensional, as shown in Eq. (11), with uniform priors $\mu_i \in [3.0,7.0]$ and $\sigma_i \in [0.0, 0.5], i = 1,2,3$.

$$\boldsymbol{\theta} = [\mu_1, \mu_2, \mu_3, \sigma_1, \sigma_2, \sigma_3]^T \tag{11}$$

The target values are $\mu_1 = 4.0$, $\mu_2 = 5.0$, $\mu_3 = 6.0$ and $\sigma_1 = 0.3$, $\sigma_2 = 0.1$, $\sigma_3 = 0.2$, as summarised in Table I.

Table I. Uncertainty characterisation with prior distributions and true target values.

Parameter	**Uncertainty characteristic**	**Target values**
k_1	Gaussian, $\mu_1 \in [3.0,7.0], \sigma_1 \in [0.0,0.5]$	$\mu_1 = 4.0, \sigma_1 = 0.3$
k_2	Gaussian, $\mu_2 \in [3.0,7.0], \sigma_2 \in [0.0,0.5]$	$\mu_2 = 5.0, \sigma_2 = 0.1$
k_3	Gaussian, $\mu_3 \in [3.0,7.0], \sigma_3 \in [0.0,0.5]$	$\mu_3 = 6.0, \sigma_3 = 0.2$
$k_4 - k_6$ $m_1 - m_3$	Constants with no updating needed.	

Updating the standard deviations alongside the means is what distinguishes stochastic from deterministic calibration, since the objective is no longer a single set of parameter values reproducing the observed value, but a set of distributions reproducing the observed scatter. It also makes the choice of likelihood metric consequential in Bayesian updating that a distance responding only to the centre of mass of the sample sets leaves the standard deviations essentially at their priors, whereas the Bhattacharyya distance is sensitive to the dispersion and the dependence of the features and can therefore identify all six hyperparameters within a single updating stage.

The observations are synthetic with a set of $N_{\text{obs}} = 100$ realisations of $\mathbf{x}$ drawn from the target distributions and propagated through $\mathcal{M}(\cdot)$, and each simulated frequency is then perturbed to represent experimental error, independently for every realisation and every mode, with $\delta = 0.1\%$.

$$\omega_r^{\text{obs}} = \omega_r\,(1 + \eta_r), \qquad \eta_r \sim \mathcal{N}(0, \delta^2) \tag{12}$$

The magnitude is chosen relative to the physical scatter of the responses that under the target input distributions the three frequencies exhibit coefficients of variation of approximately 1.5%, 0.3% and 0.5%, so that at 0.1% the measurement error is a visible but minority contribution to the total dispersion. This margin matters here since the standard deviations are inferred from precisely that dispersion and would be biased upwards were the perturbation comparable to it. The resulting sample set $\mathbf{Y_{obs}} \in \mathbb{R}^{100\times3}$ is generated once and supplied unchanged to both methods, so that any difference in the inferred posteriors is attributable to the inference engine alone. The perturbation is applied to every synthetic dataset generated for training the velocity field, so that the vectors $\boldsymbol{s}$ seen in training and at inference arise from the same process. It is not applied within the sampling route, whose simulated sets are produced by the model alone, and the Bhattacharyya likelihood compares candidates against one another rather than in absolute terms, and a small unmodelled error component raises the attainable minimum of the distance without displacing its location.

4.3 Model updating implementations and results comparisons

The two frameworks are applied to the same second-order updating problem of the three-degree-of-freedom system, in which the stiffnesses are treated as independent Gaussian random variables and the six hyperparameters, the means and standard deviations $[\mu_1, \mu_2, \mu_3, \sigma_1, \sigma_2, \sigma_3]$, are to be calibrated. The synthetic

experimental data consists of $N_{\text{obs}} = 100$ frequency observations generated from the target distribution $\mathcal{N}\left([4,5,6], \text{diag}([0.3,0.1,0.2]^2)\right)$ and they are shared across the two methods.

For the Bayesian model updating framework, the six distribution hyperparameters are updated in a single stage by comparing the simulated and experimental frequency sample sets directly with the Bhattacharyya distance. The Bhattacharyya distance is evaluated by the binning algorithm with $n_{\text{bin}} = 3$ per dimension over the joint (three-mode) frequency space, and $N_{\text{sim}} = 1000$ realisations are propagated per likelihood evalutaion. The width factor is set to $\varepsilon = 0.01$. Transitional MCMC is run with a population of $N = 1000$, a proposal scaling of $\beta = 0.2$, and $n_{MH} = 13$ proposals (burn-in of 10 and thinning of 3) per chain per stage, with the tempering schedule advanced adaptively until the tempering parameter reaches unity.

For the flow matching method, the feature set of each dataset is reduced to the vector $\boldsymbol{s} \in R^9$ of the per-mode sample means, standard deviations and cross-correlations of the frequencies, standardised using the training-set statistics. The training set comprises $N_{\text{train}} = 10^4$ pairs, corresponding to $N_{\text{train}} \times N_{obs} = 10^6$ offline model evaluations, with $N_{\text{obs}} = 100$ realisations propagated per pair. The velocity field is represented by a fully connected network of three hidden layers of 256 units with SiLU activations, taking $\boldsymbol{\theta_t}$, a sinusoidal embedding of t, and $\boldsymbol{s}$ as its input. Training uses the Adam optimiser with an initial learning rate of 2×10^{-4} under cosine decay and a batch size of 512, run for 600 epochs with the pairs split 80/10/10 into training, validation and test sets. At inference the flow is integrated by the explicit fourth-order Runge–Kutta scheme with 100 uniform steps, and 1000 posterior samples are drawn per conditioning.

Conditioned on the shared observation data, the two methods return closely consistent posteriors, as shown in Figure 2 and summarised in Table II. Transitional MCMC recovers posterior means of [3.98,5.01,5.98] for the stiffness means and [0.321,0.094,0.200] for the standard deviations, while flow matching recovers [3.96,5.03,5.99] and [0.312,0.092,0.203], respectively. The two solutions agree to within their posterior scatter and both enclose the ground truth. It is worth noting that both posteriors are displaced from the truth in the same direction, reflecting the empirical spread of the particular 100-sample record rather than a deficiency of either estimator, and the agreement of two independent methods on the same data is itself a consistency check. The flow-matching marginals are marginally broader, consistent with its reduction of each dataset to a nine-dimensional summary, whereas the Bhattacharyya likelihood exploits the full joint histogram of the frequency set. Additionally, Figure 3 shows that the per-stiffness joint posteriors from flow matching and TMCMC overlap closely and enclose the true values, confirming the two methods are in close agreement.

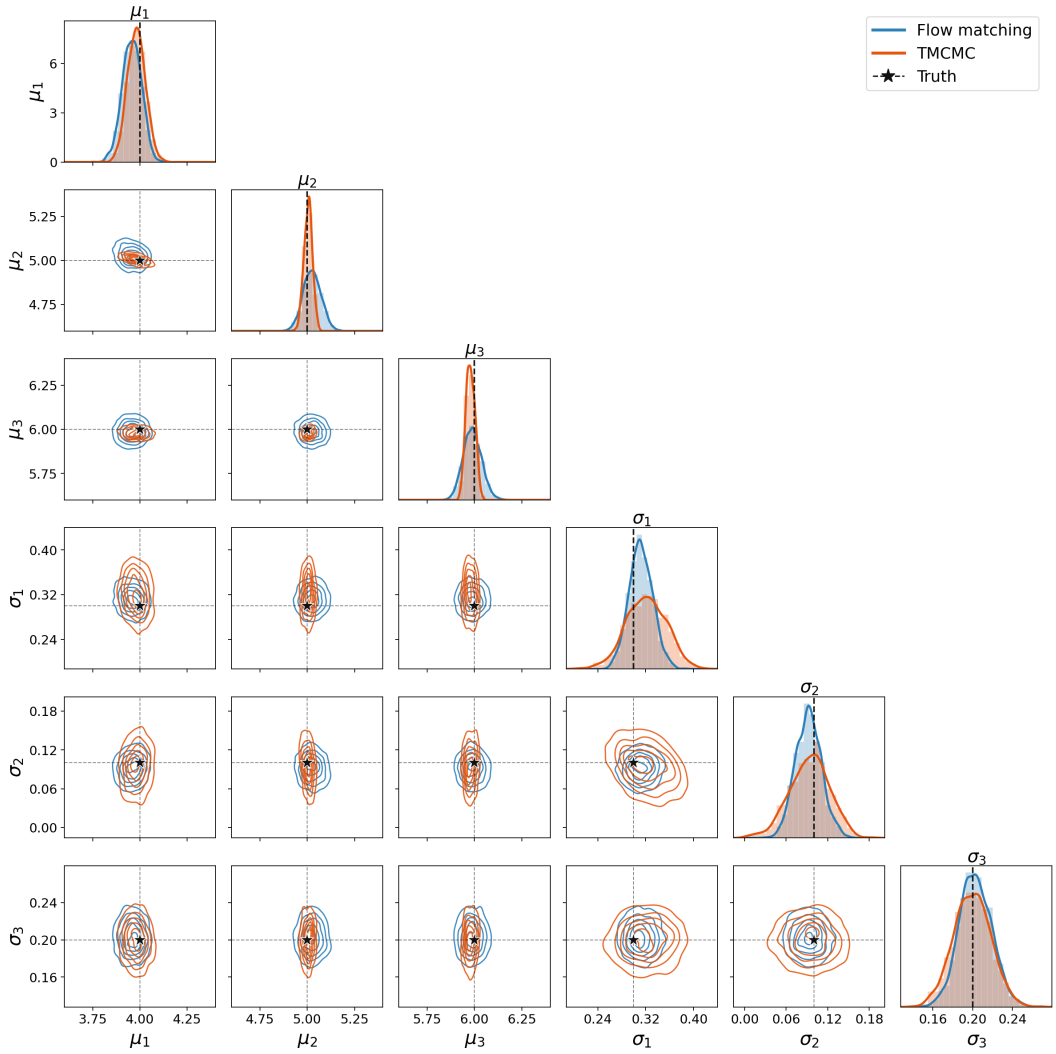

Figure 2. Posterior distributions over the six hyperparameters in comparison to the truth values.

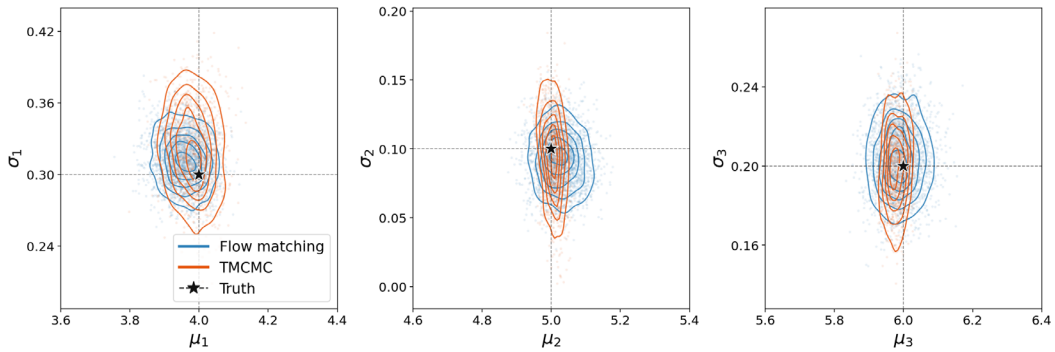

Figure 3. Joint posterior of the mean and standard deviation of each stiffness.

Table II. Calibrated uncertain parameters and computational cost comparison.		
Quantity	**Flow matching**	**Bayesian (with TMCMC)**
$[\mu_1, \mu_2, \mu_3]$	$[3.96, 5.03, 5.99]$	$[3.98, 5.01, 5.98]$
$[\sigma_1, \sigma_2, \sigma_3]$	[0.312,0.092,0.203]	$[0.321, 0.094, 0.200]$
Forward-model evaluations	10^6 (offline, one-off)	5.51×10^8 (per dataset)
Evaluations per new dataset	0	5.51×10^8
Time consumption	22.6s (Data generation: 0.77s; Training: 21.7s; Posterior inference: 0.17s)	140.5s

The essential difference between the two approaches is computational. The Bayesian scheme requires 5.51×10^8 forward-model evaluations and 140.5s of wall-clock time for this single dataset with a size of 100, and the entire procedure must be repeated for every new set of measurements. Flow matching instead spends its whole simulation budget of 10^6 evaluations once, during data generation (0.77 s). The network is then trained in a further 21.7s. For any observation dataset with the same size, it is updated in a single amortised pass of 0.17s without any further model evaluation and retraining, indicating a reduction of more than two orders of magnitude in both model calls and inference time.

To further validate the calibrated parameters at the observation level, each method's second-order posterior is propagated into a posterior-predictive response distribution. For every posterior sample of the hyperparameters, stiffness vectors are drawn from $\mathcal{N}\left(\mu, \mathrm{diag}(\sigma^2)\right)$ and propagated through the model, and the resulting frequencies are collected. This construction marginalises over the entire second-order posterior, thereby folding in the hyper-parameter uncertainty, rather than fixing a point estimate such as the posterior mean or MAP. The predictive distributions are compared with the 100 groups of synthetic observations (as shown in Figure 4). The Bhattacharyya distance is used for quantification of the discrepancies between the predictive distributions from the two methods and the target distribution with the results summarised in Table III. Both predictive distributions closely reproduce the observations, and the posterior-predictive Bhattacharyya distances to the target are 0.004 for flow matching and 0.003 for TMCMC, confirming that the recovered stiffness distributions are statistically indistinguishable from the target.

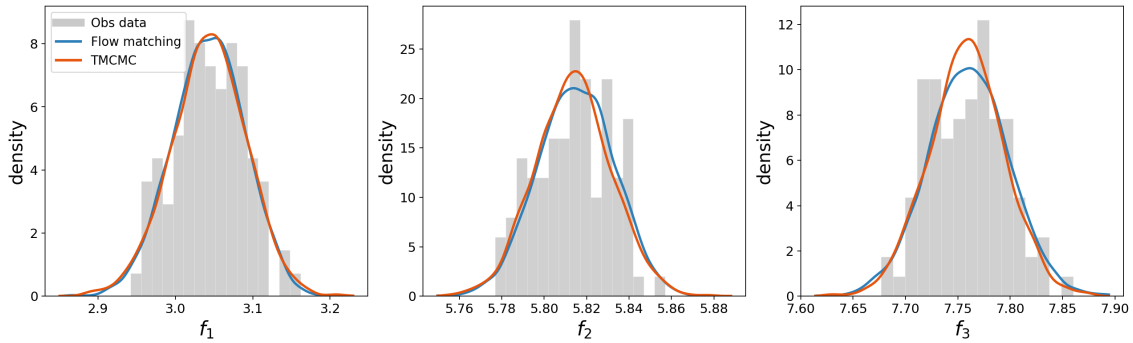

Figure 4. Posterior-predictive validation at the response level.

Table III. Posterior-predictive Bhattacharyya distance to the target observations.		
Metric	**Flow matching**	**Bayesian (with TMCMC)**
Bhattacharyya distance	0.004	0.003

5. Conclusion

This work has presented a controlled comparison between optimal-transport conditional flow matching and transitional Markov chain Monte Carlo for stochastic model updating, benchmarked on a three-degree-of-freedom spring–mass case study. The updating was posed in second-order form, in which the three stiffness means and standard deviations are calibrated jointly from a single shared set of frequency observations, so that any difference in the inferred posteriors is attributable to the inference engine alone.

Conditioned on identical data, the two methods returned closely consistent posteriors, where both recovered all six hyperparameters accurately and enclosed the ground truth, and the posterior-predictive response distributions reproduced the observed frequencies with Bhattacharyya distances to the target of 0.004 and 0.003 respectively. The flow-matching marginals were slightly broader, reflecting its reduction of each dataset to a fixed nine-dimensional summary, whereas the Bhattacharyya likelihood exploits the full joint response histogram.

The critical distinction lies in computational cost. The Bayesian framework required 5.5×10^{8} forward-model evaluations and over 140s of wall-clock time for a single dataset, and this cost recurs in full for every new set of measurements. Flow matching instead spent a one-off offline budget of 10^{6} evaluations and about 22 s of training, after which each new dataset was updated in a single amortised pass of 0.17 s with no further model calls, which is a reduction of more than two orders of magnitude in both model evaluations and inference time. The negligible cost of each subsequent update makes the framework especially suitable to time-sensitive or online tasks such as structural health monitoring, in which inference must be repeated as new data arrive and propagated into reliability assessment over the life of the structure.

Several extensions follow naturally from this study. The fixed summary statistic used here could be replaced by a conditioning network trained jointly with the velocity field, so that the informative features of the response are learned rather than prescribed and higher-order dependence is retained. The controlled comparison could then be broadened to higher-dimensional and experimentally measured structures, and to compare against conditional normalising-flow and diffusion posterior estimators evaluated under the same cost accounting so that the standing of optimal-transport flow matching among generative inference engines can be established more generally.

References

Beck, J. L. & Katafygiotis, L. S. 1998. Updating Models and Their Uncertainties. I: Bayesian Statistical Framework. *Journal of Engineering Mechanics,* 124**,** 455–461.

Bi, S., Beer, M., Cogan, S. & Mottershead, J. 2023. Stochastic Model Updating with Uncertainty Quantification: An Overview and Tutorial. *Mechanical Systems and Signal Processing,* 204.

Bi, S., Beer, M., Zhang, J., Yang, L. & He, K. 2021. Optimization or Bayesian Strategy? Performance of the Bhattacharyya Distance in Different Algorithms of Stochastic Model Updating. *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg,* 7.

Bi, S., Broggi, M. & Beer, M. 2019a. The role of the Bhattacharyya distance in stochastic model updating. *Mechanical Systems and Signal Processing,* 117**,** 437–452.

Bi, S. F., Broggi, M., Wei, P. F. & Beer, M. 2019b. The Bhattacharyya distance: Enriching the P-box in stochastic sensitivity analysis. *Mechanical Systems and Signal Processing,* 129**,** 265–281.

Ching, J. Y. & Chen, Y. C. 2007. Transitional markov chain monte carlo method for Bayesian model updating, model class selection, and model averaging. *Journal of Engineering Mechanics,* 133**,** 816–832.

Cranmer, K., Brehmer, J., Louppe, G., Cranmer, K., Brehmer, J. & Louppe, G. 2020. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences,* 117.

Ho, J., Jain, A. & Abbeel, P. 2020. Denoising Diffusion Probabilistic Models. *Advances in neural information processing systems,* 33**,** pp.6840–6851.

Kitahara, M., Bi, S. F., Broggi, M. & Beer, M. 2022. Distribution-free stochastic model updating of dynamic systems with parameter dependencies. *Structural Safety,* 97.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M. & Le, M. 2022. Flow Matching for Generative Modeling. *arXiv preprint arXiv:2210.02747*.

Liu, X., Gong, C. & Liu, Q. 2022. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. *arXiv preprint arXiv:2209.03003*.

Mares, C., Mottershead, J. E. & Friswell, M. I. 2006. Stochastic model updating: part 1—theory and simulated example. *Mechanical systems and signal processing,* 20**,** 1674–1695.

Papamakarios, G. & Murray, I. 2016. Fast ε-free Inference of Simulation Models with Bayesian Conditional Density Estimation. *Advances in Neural Information Processing Systems 29 (Nips 2016),* 29.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S. & Lakshminarayanan, B. 2021. Normalizing Flows for Probabilistic Modeling and Inference. *Journal of Machine Learning Research,* 22.

Sankararaman, S. & Mahadevan, S. 2015. Integration of model verification, validation, and calibration for uncertainty quantification in engineering systems. *Reliability Engineering & System Safety,* 138.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S. & Poole, B. 2020. Score-Based Generative Modeling through Stochastic Differential Equations. *arXiv preprint arXiv:2011.13456*.

Tong, A., Fatras, K., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Wolf, G. & Bengio, Y. 2023. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*.

Wang, T., Bi, S., Zhao, Y., Dinh, L. & Mottershead, J. 2025. Data-driven stochastic model updating and damage detection with deep generative model. *Mechanical Systems and Signal Processing,* 232.

Wang, T. R. & Bi, S. F. 2026. Stochastic model updating using conditional diffusion-based probabilistic generative models. *Mechanical Systems and Signal Processing,* 246.