Near Real-Time Probabilistic Damage Diagnosis Using Surrogate Modeling and High Performance Computing

James E. Warner*

NASA Langley Research Center, Hampton, VA, 23666, USA.

Mohammad Zubair† and Desh Ranjan†

Old Dominion University, Norfolk, VA, 23665, USA.

This work investigates novel approaches to probabilistic damage diagnosis that utilize surrogate modeling and high performance computing (HPC) to achieve substantial computational speedup. Motivated by Digital Twin, a structural health management (SHM) paradigm that integrates vehicle-specific characteristics with continual *in-situ* damage diagnosis and prognosis, the methods studied herein yield near real-time damage assessments that could enable monitoring of a vehicle's health while it is operating (*i.e. online* SHM). High-fidelity modeling and uncertainty quantification (UQ), both critical to Digital Twin, are incorporated using finite element method simulations and Bayesian inference, respectively. The crux of the proposed Bayesian diagnosis methods, however, is the reformulation of the numerical sampling algorithms (*e.g.* Markov chain Monte Carlo) used to generate the resulting probabilistic damage estimates. To this end, three distinct methods are demonstrated for rapid sampling that utilize surrogate modeling and exploit various degrees of parallelism for leveraging HPC. The accuracy and computational efficiency of the methods are compared on the problem of strain-based crack identification in thin plates. While each approach has inherent problem-specific strengths and weaknesses, all approaches are shown to provide accurate probabilistic damage diagnoses and several orders of magnitude computational speedup relative to a baseline Bayesian diagnosis implementation.

I. Introduction

Digital Twin is a structural health management (SHM) paradigm that integrates vehicle-specific characteristics including as-built components and as-experienced loading with continual *in-situ* damage diagnosis and prognosis. The framework places a strong emphasis on high-fidelity, physics-based simulation for modeling complex components and requires rigorous, end-to-end uncertainty quantification (UQ) to enable probabilistic forecasts of a vehicle's reliability into the future. Since Digital Twin aims to continually monitor a vehicle while in service and operating (*i.e.*, *online* SHM), the foundational software it is built on must also be computationally efficient. While normal operating conditions necessitate feedback on the order of minutes, cases of unexpected, discrete damage events require near real-time damage diagnosis and prognosis in order to adjust the current mission parameters accordingly. Since incorporating UQ in a framework that relies on physics-based simulation can be computationally prohibitive, delivering high-fidelity, probabilistic assessments with such a high degree of efficiency is extremely challenging.

In terms of the damage diagnosis capability of SHM, a common approach to integrate UQ is to first pose the analysis as an inverse problem (model-based diagnosis[1]) and apply Bayesian statistics to form a probabilistic solution.[2] Here, the inverse problem is to estimate parameters characterizing damage in a structural component given measurement data from sensors describing its mechanical response (*e.g.* strain, vibrations, etc.). The Bayesian solution to the inverse problem (*i.e.*, the diagnosis) is then a probability distribution of the damage parameters that is based on a computational model's predicted response in the presence of damage. The resulting probability distribution can rarely be evaluated analytically, so numerical

*Research Computer Scientist; Durability, Damage Tolerance, and Reliability Branch.

†Professor; Computer Science Department.

sampling algorithms (*e.g.*, Markov chain Monte Carlo (MCMC)[3]) must be employed to generate sample-based, probabilistic estimates of the damage parameters.

With respect to a deterministic approach to model-based damage diagnosis, the Bayesian approach has the significant advantage of recovering a fully probabilistic description of likely damage states rather than simply a point estimate with no regard for inherent uncertainties. Furthermore, when the model used is a high-fidelity simulation (*e.g.*, a finite element (FE) model), probabilistic diagnosis of arbitrarily complex components with various damage types can be enabled. The drawback of such an approach, however, is the substantial computation overhead associated with obtaining solutions using MCMC sampling. This is both due to potentially slow convergence behavior (*i.e.*, requiring a huge number of samples to accurately resolve the underlying probability distribution) and the need to run a new FE simulation for each sample drawn. For example, if a FE model takes minutes or hours to evaluate, a Bayesian MCMC approach requiring thousands of samples that utilizes a FE model would take days or weeks to complete, making its use for rapid, online damage diagnosis infeasible.

To alleviate this computational burden, advanced MCMC methods have been developed to improve the convergence rate of the sampling process.[4–7] Another common approach is to replace the original physics-based model with a computationally efficient surrogate model using probabilistic spectral methods,[8] reduced-order modeling,[9,10] or machine learning algorithms.[11] This technique requires the *offline* pre-computation and storage of an input-output pair dataset from the original computational model to train a surrogate model for use during online sampling. The development of accelerated Bayesian approaches that combine both advanced MCMC methods and surrogate modeling, however, remains relatively limited for model-based SHM applications.[12,13] Furthermore, due to the inherent dependence between subsequent samples drawn with MCMC (*i.e.*, the *Markov* property), there have been few successful attempts[14] to parallelize these algorithms and take advantage of high performance computing (HPC) capabilities.

Motivated by online SHM with Digital Twin, this study explores new numerical sampling approaches that utilize surrogate modeling and HPC to enable probabilistic damage diagnoses in near real time. In particular, three distinct methods for rapid sampling that exploit various degrees of parallelism are studied to speed up Bayesian model-based diagnosis, each relying on a precomputed training dataset to avoid the online evaluation of the original FE model. While the proposed approaches are generally applicable to many model-based diagnosis problems, they are demonstrated on the problem of strain-based crack identification in thin plates. Here, simulated strain data from a limited number of locations on the plate are used for both probabilistic damage localization (crack location estimates) as well as full characterization (crack location, size, and orientation estimates). The tradeoffs between each method in terms of accuracy, efficiency, and parallel scalability are discussed and demonstrated. While the strengths and weaknesses of each are illustrated, all approaches are shown to produce accurate probabilistic damage diagnoses in just a fraction of the computation time of a baseline Bayesian implementation using serial MCMC (*i.e.*, on one processor) and FE simulations.

The remainder of the paper is organized as follows. First, a formulation is provided in the following section that begins with a brief background on Bayesian model-based diagnosis and is followed by individual subsections devoted to each of the three sampling approaches studied herein. Next, results from the strain-based crack characterization examples are presented, illustrating the effectiveness of each sampling approach with respect to one another as well as a baseline Bayesian implementation. Finally, the findings of the study are summarized in the conclusion section.

II. Formulation

In this section, the proposed rapid sampling algorithms for near real-time probabilistic damage diagnosis are presented. First, a brief overview of model-based diagnosis is provided, including a probabilistic solution approach that covers Bayesian inference and Markov chain Monte Carlo (MCMC) sampling. Then, individual subsections are devoted to each of the proposed sampling algorithms to accelerate/replace traditional MCMC with surrogate modeling and high performance computing (HPC).

A. Background

1. *Model-Based Diagnosis*

Damage diagnosis methods operate under the assumption that the mechanical response of a structural component is altered in the presence of damage. To this end, the goal of diagnosis methods is to use measured response data $\mathbf{d}^{\text{obs}} \in \mathbb{R}^m$ to detect if damage is present and then ideally estimate some parameters $\mathbf{c} \in \mathbb{R}^d$ that characterize the damage (location, size, etc.). Model-based approaches to diagnosis require a model of the structural component capable of predicting the mechanical response for a given set of damage parameters

$$\mathcal{M}(\mathbf{c}) = \mathbf{y} \in \mathbb{R}^m, \tag{1}$$

where $\mathbf{y} \approx \mathbf{d}^{\text{obs}}$ for a damage estimate $\mathbf{c}$ that effectively characterizes the true damage. Here, it is assumed that $\mathcal{M}$ encompasses properly calibrated boundary conditions, material properties, etc. and postprocessing to extract predicted responses in $\mathbf{y}$ that correspond to the time/location of the measurements $\mathbf{d}^{\text{obs}}$.

In the context of model-based diagnosis, $\mathcal{M}$ is referred to as the forward model while the diagnosis problem of using $\mathbf{d}^{\text{obs}}$ to infer $\mathbf{c}$ is the associated inverse problem. A typical deterministic approach to solving this inverse problem is to first pose an error metric between the measured response data and corresponding model response

$$Q(\mathbf{c}, \mathbf{d}^{\text{obs}}) = \sum_{i=1}^{m} \|d_i^{\text{obs}} - \mathcal{M}_i(\mathbf{c})\|^2, \tag{2}$$

where $\mathcal{M}_i(\mathbf{c}) \equiv y_i$. Then, gradient-based or global optimization algorithms are employed to find the damage parameters that minimize Equation (2) to produce the so-called least squares estimator

$$\mathbf{c}^{LS} = \arg\min_{\mathbf{c}} Q(\mathbf{c}, \mathbf{d}^{\text{obs}}). \tag{3}$$

The primary drawback of such deterministic approaches for model-based diagnosis is that only a point estimate of the damage is produced with no regard to uncertainty inherent in the measurement data (noise, sparsity, etc.).

2. *Bayesian Inference*

The Bayesian inference approach to model-based diagnosis reformulates the inverse problem as one of deducing a probability distribution of the unknown damage parameters, $\mathbf{c}$, conditional on the observed measurement data $\mathbf{d}^{\text{obs}}$. This distribution, $p(\mathbf{c}|\mathbf{d}^{\text{obs}})$, known as the *posterior distribution*, is given according to Bayes' Theorem:[15]

$$p(\mathbf{c}|\mathbf{d}^{\text{obs}}) \propto p(\mathbf{d}^{\text{obs}}|\mathbf{c})p(\mathbf{c}), \tag{4}$$

integrating any knowledge about the damage prior to the measurement in the *prior distribution*, $p(\mathbf{c})$, with the information from the data, $\mathbf{d}^{\text{obs}}$, through the *likelihood function*, $p(\mathbf{d}^{\text{obs}}|\mathbf{c})$. While the prior distribution can be a powerful tool for applying an analyst's insight about likely damage states for a component, a non-informative prior (e.g., uniform probability) is chosen in this work as to not bias the results. On the other hand, the likelihood function in this work takes the form

$$\begin{aligned} p(\mathbf{d}^{\text{obs}}|\mathbf{c}) &= \frac{1}{(2\pi\sigma^2)^{m/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{m} \|d_i^{\text{obs}} - \mathcal{M}_i(\mathbf{c})\|^2\right) \\ &\propto \exp\left(-\frac{1}{2\sigma^2} Q(\mathbf{c}, \mathbf{d}^{\text{obs}})\right), \end{aligned} \tag{5}$$

where this expression is the direct result of the following assumed relationship between the measured and computed responses:

$$d_i^{\text{obs}} = \mathcal{M}_i(\mathbf{c}) + \delta_i, \quad \delta_i \sim \text{Normal}(0, \sigma). \tag{6}$$

That is, the measurement data are polluted with errors, δ_i, that are treated as a sequence of independent, identically distributed (i.i.d.) samples drawn from a zero-mean Gaussian (Normal) distribution with variance σ^2 (interpreted as the noise level). Note that this exposition was intentionally kept brief and only used as a means to facilitate the succeeding formulations, the interested reader can consult the references regarding Bayesian inverse problems[2,15] and its application to damage diagnosis[13] for more details.

3. Markov Chain Monte Carlo (MCMC)

The solution of the model-based diagnosis problem as the posterior probability distribution in Equation (4) is not of practical importance on its own since it can rarely be evaluated analytically. MCMC[3] is arguably the most effective and common approach to explore the posterior distribution to form probabilistic damage estimates with a Bayesian inference approach. The goal of MCMC is to generate a collection of N damage parameter samples from the posterior probability distribution

$$\{\mathbf{c}^{(j)}\}_{j=1}^{N} \text{ where } \mathbf{c}^{(j)} \sim p(\mathbf{c}|\mathbf{d}^{\text{obs}}), \tag{7}$$

which can then be used to construct empirical probability distributions, credibility intervals, and moment estimates for $\mathbf{C}$. Algorithm 1 summarizes the most basic form of MCMC, the Metropolis algorithm:[3]

Algorithm 1 Metropolis MCMC

```
Initialize $\mathbf{c}^{(0)}$
for $j = 1 : N$ do
  Sample $u \sim \text{Uniform}(0, 1)$
  Sample $\mathbf{c}^* \sim q(\mathbf{c}^*|\mathbf{c}^{(j-1)})$
  if $u < A(\mathbf{c}^*, \mathbf{c}^{(j-1)}) = \min\{1, \frac{p(\mathbf{c}^*|d^{\text{obs}})}{p(\mathbf{c}^{(j-1)}|d^{\text{obs}})}\}$ then
    $\mathbf{c}^{(j)} = \mathbf{c}^*$
  else
    $\mathbf{c}^{(j)} = \mathbf{c}^{(j-1)}$
  end if
end for
```

Here, the method simply draws a trial sample, $\mathbf{c}^*$, at each iteration from a *proposal distribution*, $q(\mathbf{c}^*|\mathbf{c}^{(j-1)})$, and then decides whether to accept or reject this sample based on the *acceptance probability*, $A(\mathbf{c}^*, \mathbf{c}^{(j-1)})$. The Metropolis algorithm assumes that the proposal distribution is symmetric, where a common choice is a Gaussian distribution centered at the previous sample

$$q(\mathbf{c}^*|\mathbf{c}^{(j-1)}) = \text{Normal}(\mathbf{c}^{(j-1)}, \Sigma_q), \tag{8}$$

where Σ_q is the user-specified covariance matrix. Algorithm 1 with Equation (8) constructs a Markov chain that, by design, is guaranteed to have a stationary distribution that reflects the true posterior distribution in Equation (4).

B. Methods for Rapid Sampling

While the Bayesian model-based diagnosis approach discussed in Section A provides an effective means of generating probabilistic damage estimates, it generally incurs a significant computational overhead making it impractical for online SHM. This computational expense is primarily attributed to the MCMC sampling process (Algorithm 1) detailed in the previous section. Here, the posterior probability distribution must be evaluated to determine the acceptance probability, A, for every sample drawn, so the computational model, $\mathcal{M}$, must also be executed at each iteration. Since generally $N \sim O(10^3 - 10^6)$ samples are needed for convergence with MCMC, utilizing even a modestly complex model yields intractable analysis times. Furthermore, since there is an explicit dependence on the previous sample drawn at each iteration through the acceptance probability, parallelizing MCMC algorithms to leverage high performance computing (HPC) can be challenging.

In this section, three sampling methods are formulated to provide substantial computational speedup for probabilistic diagnosis (by rapidly evaluating Equation (7)), each utilizing surrogate modeling and HPC. Surrogate modeling relies on the *offline* pre-computation and storage of input-output pair datasets from the original model so that the posterior probability distribution can be rapidly evaluated during *online* sampling. Here, a set of T damage parameter arrays $\{\mathbf{c}^{(k)}\}_{k=1}^{T}$ is first selected. Then, the model response corresponding to all m measurements are computed and stored for each damage state,

$$\mathcal{M}_i^{(k)} \equiv \mathcal{M}_i(\mathbf{c}^{(k)}),\ k = 1, ..., T \tag{9}$$

for $i = 1, ..., m$. The result is the following $T \times (d + m)$ input-output dataset

$$\mathcal{S} = \{\mathbf{c}^{(k)}; \mathcal{M}_1^{(k)}, ..., \mathcal{M}_m^{(k)}\}_{k=1}^{T}. \tag{10}$$

Each sampling method to follow differs in how the input-output data, $\mathcal{S}$, is utilized and in the resulting approach to leveraging HPC. The first approach replaces each model predicted response, $\mathcal{M}_i$, with individual, pre-trained surrogate models and employs a parallel MCMC algorithm[14] that allows the models to be partitioned across processors for sampling. The second approach constructs a surrogate model for the error function, $Q(\mathbf{c}, \mathbf{d}^{\text{obs}})$, on-the-fly to facilitate subsequent rapid MCMC sampling. The third approach discretizes the posterior probability distribution function, $p(\mathbf{c}|\mathbf{d}^{\text{obs}})$, to enable completely parallel direct sampling. The following subsections describe each of the three approaches in more detail.

1. *Method 1: MCMC with Model Surrogates*

The first sampling method adopts an existing approach[11, 13] of replacing the original computational model itself with surrogate models and then explores the use of an emerging parallel MCMC algorithm[14] to take advantage of HPC. From a machine learning perspective, $\mathcal{S}$ (Equation (10)) is the training data and a variety of off-the-shelf regression and interpolation algorithms can be utilized to directly infer the input-output mappings. Specifically, a surrogate model that maps a new damage state, $\mathbf{c}^{(*)}$, to the resulting sensor response is generated offline for each individual measurement

$$\widetilde{\mathcal{M}}_i : \mathbf{c}^{(*)} \rightarrow \mathcal{M}_i^{(*)} \text{ for } i = 1, ..., m, \tag{11}$$

resulting in a suite of independent surrogate models, $\{\widetilde{\mathcal{M}}_i\}_{i=1}^{m}$. Then, during online damage diagnosis, the posterior distribution can be efficiently sampled with MCMC using an approximate form of the likelihood (Equation (5)) that relies on these surrogate models

$$p(\mathbf{d}^{\text{obs}}|\mathbf{c}; \{\widetilde{\mathcal{M}}_i\}) \propto \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{m} \|d_i^{\text{obs}} - \widetilde{\mathcal{M}}_i(\mathbf{c})\|^2\right). \tag{12}$$

Since $\widetilde{\mathcal{M}}_i$ can generally be evaluated much faster than the original model $\mathcal{M}$, significant computational speedup can be obtained with this technique alone.[13]

To leverage HPC for additional speedup with this approach, multiple independent chains of MCMC can simply be run on separate computer processors in parallel. While this straightforward parallelization will be used as a baseline method for comparison in this study, a more sophisticated parallel MCMC algorithm[14] will be the primary focus for its potential efficiency and scalability benefits. The algorithm, which enables asymptotically exact and parallel MCMC, was developed for (and has been traditionally applied to) MCMC for big data applications where it is difficult to store and analyze all data observations on one processor. The method allows the data to be randomly partitioned among machines and independent MCMC chains to be run on only subsets of the data in parallel, resulting in improved scalability when utilizing many processors. The crux of the method is the utilization of special combination algorithms to merge samples from each machine in a manner that is provably asymptotically exact from the full-data probability distribution. More details of the approach can be found in the original paper.[14]

To the authors' knowledge, this is the first application of this parallel MCMC method to damage diagnosis or surrogate model-based inverse problems in general. While the diagnosis application tested later in this paper uses a relatively small amount of measurement data (far from necessitating a *big data* method), there is potential benefit due to the independent/individualized surrogate modeling strategy used (Equation (11)). Since there is a one-to-one correspondence between each sensor measurement, d_i^{obs}, and surrogate model $\widetilde{\mathcal{M}}_i$, partitioning the measurement data across processors effectively partitions the surrogate models as well. This strategy can provide significant computational savings for applications with large amounts of sensor data or when the regression/interpolation algorithm used for surrogate modeling is memory and/or computationally intensive.

2. *Method 2: MCMC with Error Function Surrogate*

Rather than construct individual surrogate models for each sensor response offline, the second sampling method builds a single surrogate model for the error function, $Q(\mathbf{c}, \mathbf{d}^{\text{obs}})$, on-the-fly after receiving measurement data but prior to sampling. In this case, the dataset $\mathcal{S}$ is used to first evaluate Equation (2) to generate new training data describing the input-output relationship between damage parameters and errors

$$\mathcal{Q} = \{\mathbf{c}^{(k)}; Q^{(k)}\}_{k=1}^{T}, \tag{13}$$

where $Q^{(k)} \equiv Q(\mathbf{c}^{(k)}, \mathbf{d}^{\text{obs}})$. Using this dataset $\mathcal{Q}$, regression or interpolation can be used to construct a mapping directly from new damage parameters $\mathbf{c}^{(*)}$ to the resulting error

$$\widetilde{Q} : \mathbf{c}^{(*)} \rightarrow Q^{(*)}. \tag{14}$$

Then, the following approximate likelihood function can be formed using the error function surrogate model

$$p(\mathbf{d}^{\text{obs}}|\mathbf{c}; \widetilde{Q}) \propto \exp\left(-\frac{1}{2\sigma^2}\widetilde{Q}(\mathbf{c}, \mathbf{d}^{\text{obs}})\right), \tag{15}$$

allowing for rapid MCMC sampling of the posterior distribution without the evaluation of models or surrogates for sensor values themselves. To leverage HPC with this method, the simple parallel MCMC approach of running several independent chains on separate processors will be used.

The benefit of the error function surrogate method is that there is a single surrogate model to evaluate for each sample drawn using MCMC rather than a suite of models for each sensor prediction, resulting in a speedup proportional to m during sampling. The disadvantage of the approach is that the surrogate model for error, $\widetilde{Q}$, must now be constructed online, after receiving measurement data for diagnosis. Thus, there is an additional initial computational overhead for generating the new training data, $\mathcal{Q}$ (which can be done in parallel), as well as fitting a regression/interpolation algorithm to these data to generate the surrogate. Another more subtle issue is that surrogate verification is more difficult in this case compared with the more standard model surrogate approach in the previous section. Generating surrogates for the model-predicted sensor values is done offline rather than online and therefore time can be taken beforehand to verify the accuracy and effectiveness of the approximate surrogate models versus the original model predictions. This same verification process cannot be carried out with the error surrogate approach, necessitating the future development of *a priori* error estimates/bounds for the approach.

3. Method 3: Direct Probability Discretization & Sampling

Instead of relying on traditional MCMC sampling with an inherently limited degree of parallelism, the third sampling approach uses the stored training data (Equation (10)) to discretize and sample the posterior distribution directly in parallel. To this end, a discretization of the probability distribution is assumed such that the parameter space is divided into grid cells with centroids coinciding with the input training dataset $\{\mathbf{c}^{(k)}\}_{k=1}^{T}$ from $\mathcal{S}$ (*i.e. nearest neighbor cells*). Using the training dataset, a normalized posterior probability value can be computed at each grid point. Then, under the assumption of uniform probability within each nearest neighbor cell, samples can be generated uniformly from each cell in proportion to its normalized probability value.

This procedure for direct discretization and sampling of the posterior distribution is described in Algorithm 2 below. Here, the evaluation of the posterior probability values for each grid point utilizes the precomputed and stored model values $\mathcal{M}_i^{(k)}$ from Equation (10). Note also that since a uniform training grid is used in this study, the nearest neighbor cells are simply hypercubes centered on the training data, but the approach could be extended to the case of nonuniform grids as well. It can be seen from Algorithm

Algorithm 2 Direct Sampling

```
for each grid point $\mathbf{c}^{(k)}$ do
    $p^{(k)} \equiv p(\mathbf{c}^{(k)}|\mathbf{d}^{\text{obs}}) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{m}\|d_i^{\text{obs}} - \mathcal{M}_i^{(k)}\|^2\right)$
end for
$p^{\text{sum}} = \Sigma_k \, p^{(k)}$
for each grid point $\mathbf{c}^{(k)}$ do
    $\bar{p}^{(k)} = p^{(k)}/p^{\text{sum}}$
    $N^{(k)} = \bar{p}^{(k)} \times N$
    Generate $N^{(k)}$ random points uniformly from the nearest neighbor cell of $\mathbf{c}^{(k)}$
end for
```

2 that the approach is highly amenable to parallel computing with all normalized probabilities being computed independent of one another. The normalized probabilities are then used to compute p^{sum} using an efficient parallel sum reduction routine before communicating this value to all the processors. Thereafter, if N samples are needed from the posterior distribution and P processors are used, each processor simply generates N/P samples in parallel. The procedure can be implemented on a CPU with mulitple cores and also on a graphics processing unit (GPU) device.

The direct sampling approach has the advantage of being highly parallel with the potential of enabling tremendous computational speedup over a MCMC-based sampling approach. If the training grid used is fine enough, the method is also expected to produce a distribution that closely approximates the actual posterior distribution. Similar to the error surrogate MCMC approach, verifying the accuracy of direct sampling is not as straightforward as the model surrogates approach where models can be trained/verified before diagnosis takes place. While the approach will be shown empirically to yield highly accurate damage estimates in the results to follow, development of an *a priori* approach for verification of the method is worthy of future study.

III. Results

A. Application - Strain-Based Crack Characterization

While the damage diagnosis algorithms presented in Section II are general with respect to the measurement data, $\mathbf{d}^{\text{obs}}$, model, $\mathcal{M}$, and damage description, $\mathbf{c}$, they are demonstrated here on the specific problem of strain-based crack characterization in thin plates. A diagram illustrating this application can be seen in Figure 1. Here, the damage is characterized by four parameters defining the location, size, and orientation of a straight crack in the plate: $\mathbf{c} = [x, y, a, \theta]$. The measurement data, $\mathbf{d}^{\text{obs}}$, used to infer these crack parameters are m strain observations throughout the domain. The model, $\mathcal{M}$, is a linear elastic FE simulation using the Scalable Implementation of Finite Elements by NASA (ScIFEN)[16] parallel FE code. The implementation of surrogate models and MCMC sampling for the three damage diagnosis algorithms was carried out in Python.[17]

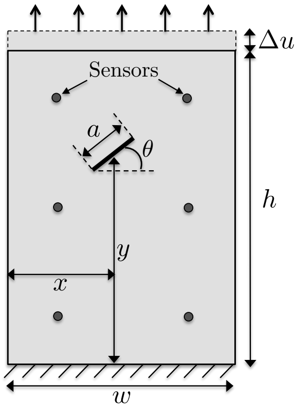


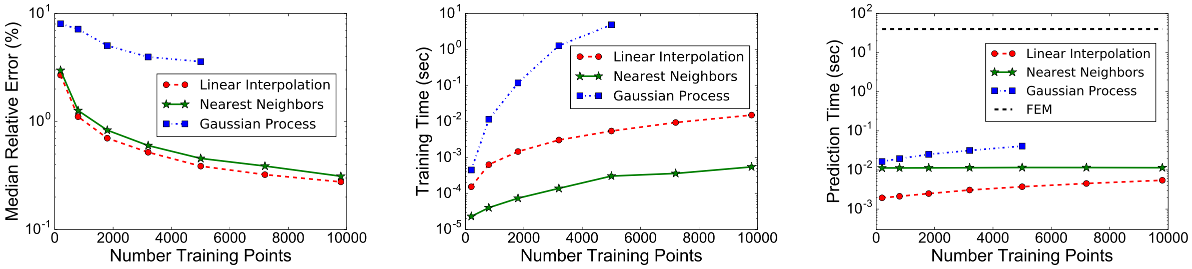
Figure 1. Diagram of the problem domain and damage parameterization considered for the strain-based crack characterization application.

The measurement data, $\mathbf{d}^{\text{obs}}$, for the succeeding examples were generated by inserting a target damage, $\mathbf{c}^{\text{ref}}$, into the finite element model, storing strains at predetermined sensor locations, and adding random noise according to Equation (6) to simulate sensor errors. Here, the noise level σ was selected such that there was approximately 15% sensor error with respect to the measurement in each case. The plate geometry had a width (w) of $96mm$, height (h) of $215mm$, and thickness of $5mm$. The Poisson ratio was 0.3 and the applied displacement (Δu) was $1.0mm$ (the Young's Modulus was arbitrary since the quantity of interest was strain). Two separate diagnosis examples are considered: 1) damage localization - estimating the location of the crack only ($\mathbf{c} = [x, y]$) and 2) damage characterization - estimating the location, size, and orientation of the crack ($\mathbf{c} = [x, y, a, \theta]$). The accuracy and computational efficiency of the three sampling approaches formulated in Section II are compared on these two examples for varying numbers of measurements (m), training data sizes (T - Equation (10)), and number of samples drawn (N). The results for damage localization and characterization are provided in the remaining subsections after a brief comparison of different surrogate modeling algorithms.

B. Surrogate Model Comparison

In this section, a brief comparison of the surrogate modeling algorithms considered in this work is provided. Recall that the MCMC with model surrogates approach (Section II.B.1) requires pre-trained/stored surrogate models for each sensor value (Equation (11)) using interpolation or regression methods. Therefore, several algorithms can be tested offline before damage diagnosis is performed to identify the most accurate and fastest technique for surrogate modeling. Three algorithms were tested here, Gaussian process and K-nearest neighbors regression from the `scikit-learn` Python module[18] and multilinear interpolation from the `SciPy` module.[19] For brevity, only the results for the damage localization surrogate models (mapping damage location to predicted sensor values) are shown while a similar analysis was conducted for damage characterization as well.

Training datasets (Equation (10)) of different sizes ($T = \{200, 800, 1800, 3200, 5000, 7200, 9800\}$) were first generated using FE simulations. A testing dataset of 1000 randomly drawn crack locations and the resulting predicted strains at the sensor locations was also generated for evaluating surrogate model accuracy. The three algorithms were then used to train surrogate models for each training dataset and generate predictions on the testing dataset, calculating the error versus the original FE solution. The results of this comparison are shown in Figure 2. Figure 2(a) shows the median relative error while Figures 2(b) and 2(c) compare the training and prediction times for the different models tested, respectively.



(a)

(b)

(c)

Figure 2. Performance comparison of three different surrogate modeling algorithms in terms of a) relative error, b) training time, and c) prediction time.

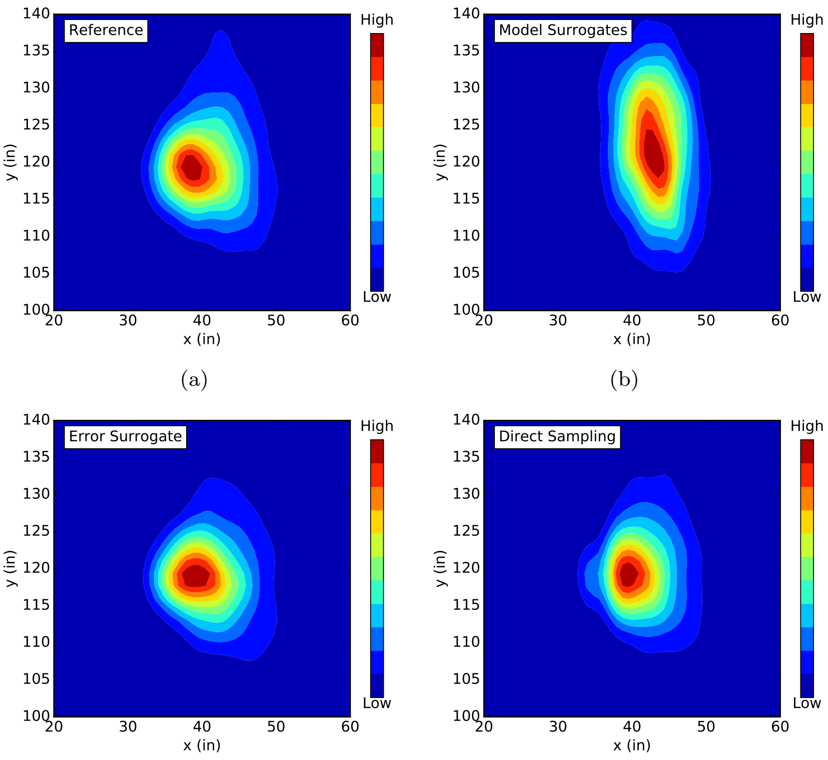
Accuracy and prediction time are most important for the effectiveness of the model surrogates sampling approach, and it is clear from Figure 2 that the linear interpolation algorithm is most accurate and efficient in this particular application. Note that the average FE solution time was approximately 40 seconds and is depicted in Figure 2(c) for comparison. Here, the substantial computational speedup that surrogate modeling provides is evident as all three methods deliver several orders of magnitude improvement in prediction time. While offline training time has less performance impact for online analysis, it can be seen that training the Gaussian process algorithm can be prohibitively expensive for larger datasets and so it was only considered for training data sizes up to $T = 5000$. Training time is critical, however, for the error surrogate sampling approach (Section II.B.2) where the surrogate model must be constructed on-the-fly during damage diagnosis. In this case, the nearest neighbor algorithm was chosen based on its significantly faster training times and comparative accuracy. Note that the nearest neighbor algorithm was also used for the damage characterization example, as it shows the best scaling with problem dimension and training dataset size.

C. Damage Localization

The sampling algorithms presented in Section II were first compared on a damage localization example. Here, the location of a crack ($\mathbf{c} = [x, y]$) was estimated using simulated strain data assuming a known crack length and orientation. The performance of the methods was studied for different numbers of measurements ($m = \{12, 24, 48, 100, 200\}$) and training grid sizes ($T = \{200, 1800, 3200, 5000, 9800\}$) as well as in serial (on 1 computer core) and in parallel (between 2-4 cores) on a quad-core 2.4GHz AMD Opteron processor.

In order to evaluate the accuracy of the methods, a reference solution was obtained for each number of measurements considered by generating 10,000 samples from the posterior $p(\mathbf{c}|\mathbf{d}^{\text{obs}})$ using serial MCMC with the original FE simulation. Treating this collection of samples as the ground truth for each case, the accuracy of the samples drawn with the three proposed methods was evaluated using the multidimensional Kolmogorov-Smirnov (KS) test.[20]

First, an example of the probabilistic damage localization solutions generated with each method is shown in Figure 3 for the case of $m = 100$ measurements and $T = 5000$ training data executed on $P = 4$ processors. Here, the probability density of the crack location is shown for a) the reference solution, b) MCMC with model surrogates, c) MCMC with an error surrogate, and d) direct probability sampling. The computation time to generate each solution was approximately a) 4.5 days, b) 8.4 seconds, c) 2.7 seconds, and d) 0.3 seconds. While the approximate probability distributions show generally good agreement with the reference distribution, the model surrogates approach is noticeably less accurate in comparison to the error surrogate and direct sampling approaches.



(a)

(b)

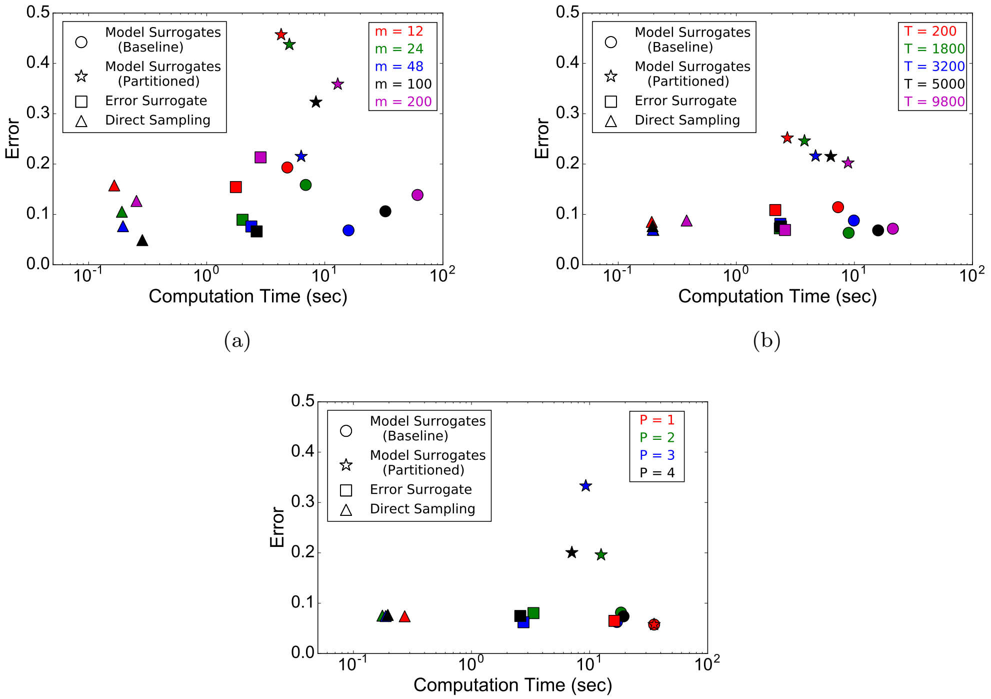
(c)

(d)

Figure 3. Crack location probability contours for $m = 100$ measurements: a) reference solution using serial MCMC and FE simulation and approximate solutions using b) MCMC with model surrogates, c) MCMC with an error surrogate, and d) direct probability sampling.

A more comprehensive assessment of each sampling algorithm's performance for damage localization can be seen in Figure 4. Here, the error (as evaluated using the KS test) and computation time are illustrated for each proposed approach and compared with a naive parallelization method (*Model Surrogates (Baseline)*) that entails independent MCMC chains of duplicated model surrogates across each computer core. Figure 4(a) compares the performance of each method for different numbers of measurements (m) for fixed training data size ($T = 5000$) and number of processors ($P = 4$). Figure 4(b) provides a comparison for varying training data sizes (T) for a fixed number of measurements ($m = 48$) and number of processors ($P = 4$). Finally, Figure 4(c) demonstrates the scaling of each method for different numbers of processors (P) while holding the number of measurements ($m = 48$) and training data sizes ($T = 5000$) constant.

From Figure 4(a), it is clear that the direct sampling approach is significantly more efficient than the other two proposed sampling methods and the baseline implementation for each different number of measurements considered. Furthermore, the accuracy of direct sampling is comparable or better than each of the other



(a)

(b)

(c)

Figure 4. Performance comparison of the proposed sampling methods for the damage localization example for (a) different numbers of measurements (with $T = 5000$ and $P = 4$), (b) different training dataset sizes (with $m = 48$ and $P = 4$) and c) different number of processors (with $m = 48$ and $T = 5000$).

methods. Another observation is that the computation time for direct sampling and error surrogate sampling are less sensitive to the number of measurements used in comparison with the model surrogates approach. This is due to the fact that a surrogate is trained and evaluated for each individual measurement in the latter approach and therefore the complexity of the method scales linearly with m. While it can be seen that the proposed partitioned parallel approach for model surrogates sampling provides significant computational speedup over the baseline approach, it also incurs a noticeable penalty in terms of accuracy. This source of error will be discussed further in the next section.

Figure 4(b) demonstrates the effect of training data size on the performance of each sampling method. The plot illustrates that direct sampling results in the most efficient solutions with accuracy that is comparable or better than the other methods for varying training dataset sizes. As T increases, there is generally an increase in computation time and decrease in error for each method, which is expected as the resolution of the training grid is increased. The computation time for the model surrogates (both partitioned and baseline) approach shows the most significant dependence on training data size. Again, partitioning the model surrogates across processors results in a significant efficiency increase at the expense of accuracy.

Finally, the scalability of each approach for different numbers of processors is illustrated in Figure 4(c). The partitioned model surrogates and error surrogate approaches appear to enable the best scalability in the sense that the decrease in computation time is most significant as processors are added in comparison to the other approaches. The partitioned model surrogates approach achieves better scalability with respect to the baseline approach since the number of surrogates that must be evaluated on each processor during sampling is decreased by a factor of P. However, it can be seen that the accuracy is negatively impacted for increasing P since less data will be present on each processor, resulting in less information from which to infer the damage during sampling. Note that the performance for the baseline and partitioned model surrogates approach coincide for $P = 1$ since the methods only differ in how they are parallelized ($P > 1$), so these markers coincide in Figure 4(c). With respect to the reference implementation with serial MCMC and FE simulation that took over four days to perform damage localization here, each of the proposed sampling approaches provide tremendous computational speedup with a relatively high degree of accuracy.

In particular, the model surrogates, error surrogate, and direct sampling approaches provide $O(10^4)$, $O(10^5)$, and $O(10^6)$ computational speedup, respectively.

D. Damage Characterization

The performance of the proposed sampling methods is now illustrated for general crack characterization where simulated strain data were used to infer the location, size, and orientation of a crack in the domain ($\mathbf{c} = [x, y, a, \theta]$). The performance of each sampling method is compared for different numbers of measurements ($m = \{12, 24, 48, 100\}$) and different training data sizes ($T = \{10143, 19712, 39780\}$), where a substantial increase in grid size is necessary to accommodate the increased dimension of the unknown parameters ($d = 4$). Each method was again executed on between $P = 1$ and $P = 4$ processors and reference solutions were obtained using serial MCMC with FE simulation, drawing 10,000 samples for each approach. For this example, the evaluation of the KS score for the approximate samples was not feasible since the computational cost increases exponentially with dimension so qualitative comparisons against the reference solution were used instead. An efficient and scalable implementation of the KS score computation is an area of continuing research to rigorously study the sampling method accuracy for higher dimensions.

Figure 5 provides a qualitative assessment of the accuracy of each sampling method versus the reference solution for damage characterization. Here, the cumulative distribution functions (CDFs) for each unknown crack parameter are shown for each sampling method and the reference implementation for the particular case of $m = 48$ measurements and $T = 19712$ training data points executed on $P = 2$ processors. The computation time required to produce these results was 4.5 days for the reference implementation with serial MCMC and FE simulation, 44.2 seconds for the model surrogates MCMC approach, 4.5 seconds for the error surrogate MCMC approach, and 0.4 seconds for direct probability sampling. While each method provides substantial computational speedup, Figure 5 shows that they also provide accurate and comparable approximations to the probability distribution of the unknown parameters as well. Note that while only one set of solutions are provided here for brevity, similar qualitative trends in accuracy were seen for different parameter combinations as compared with the damage localization results in Figure 4. Here, decreasing the number of measurements and training dataset size generally increased errors slightly for all the methods, while varying the processors only affected the partitioned model surrogates approach, where the accuracy decreased with increasing number of processors. Future research will seek to provide more quantitative assessments of errors for the higher dimensional damage characterization case.

Figure 6 represents a more in depth look at the performance of the sampling methods, comparing their computation times against the naive parallelization approach with model surrogates as a baseline. Figure 6(a) plots computation time as a function of the number of measurements (m) for a fixed training data size ($T = 19712$) and number of processors ($P = 4$). Figure 6(b) shows computation time as a function of training dataset size (T) for a fixed number of measurements ($m = 48$) and number of processors ($P = 4$). In Figure 6(c), the normalized computation time is shown (normalized by the time for $P = 1$) as a function of the number of processors used to demonstrate the scalability of the methods for a fixed number of measurements ($m = 48$) and training data size ($T = 19712$). Overall, it can be seen that direct probability sampling is the most efficient by a substantial margin, followed by the error surrogate MCMC approach, and then the model surrogates MCMC approach. The approach consistently yields damage estimates in under one second irrespective of the parameter $\{m, T, P\}$ being varied. It is also observed that utilizing the parallel MCMC approach,[14] that partitions model surrogates across processors, provides significant computational speedup over the baseline approach.

Figure 6(a) demonstrates the dependency of computation times on the number of measurements used for each approach. It can be seen that the model surrogates approach displays a solution time roughly proportional to the number of measurements since a surrogate model is trained/evaluated for each individual sensor value. On the other hand, the number of measurements in the range tested here display a negligible impact on the error surrogate MCMC approach and direct probability sampling. Similarly, the size of the training dataset used shows little effect on the computation time for each sampling method, as seen in Figure 6(b). This lack of sensitivity is primarily due to the efficient scaling of the prediction times with training dataset size (Figure 2(c)) for the nearest neighbor regression algorithm, which was used for the model surrogates and error surrogate approaches in this example. It is also likely that the relatively small range in training dataset sizes considered in this example contributes to the apparent lack of sensitivity in computation times.

Finally, Figure 6(c) demonstrates the scalability of each approach by showing the impact of the number of

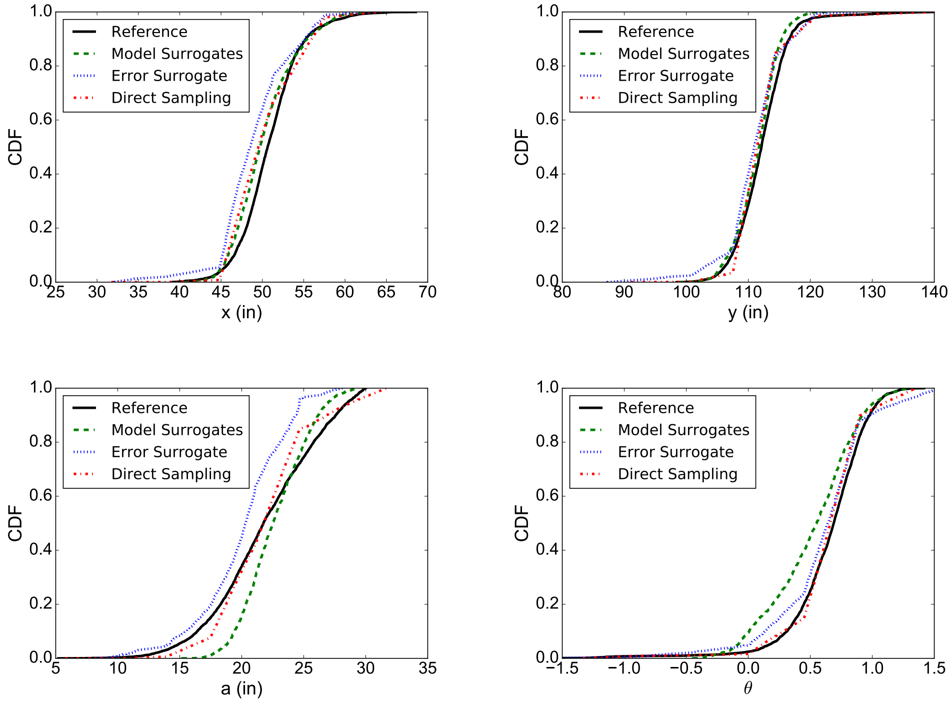


Figure 5. Comparison of the estimated cumulative distribution functions using each sampling method with the reference solution for each crack parameter a) x, b) y, c) a, and d) θ. The results shown are for $m = 48$ measurements and a training grid of size $T = 19712$ executed on $P = 2$ processors.

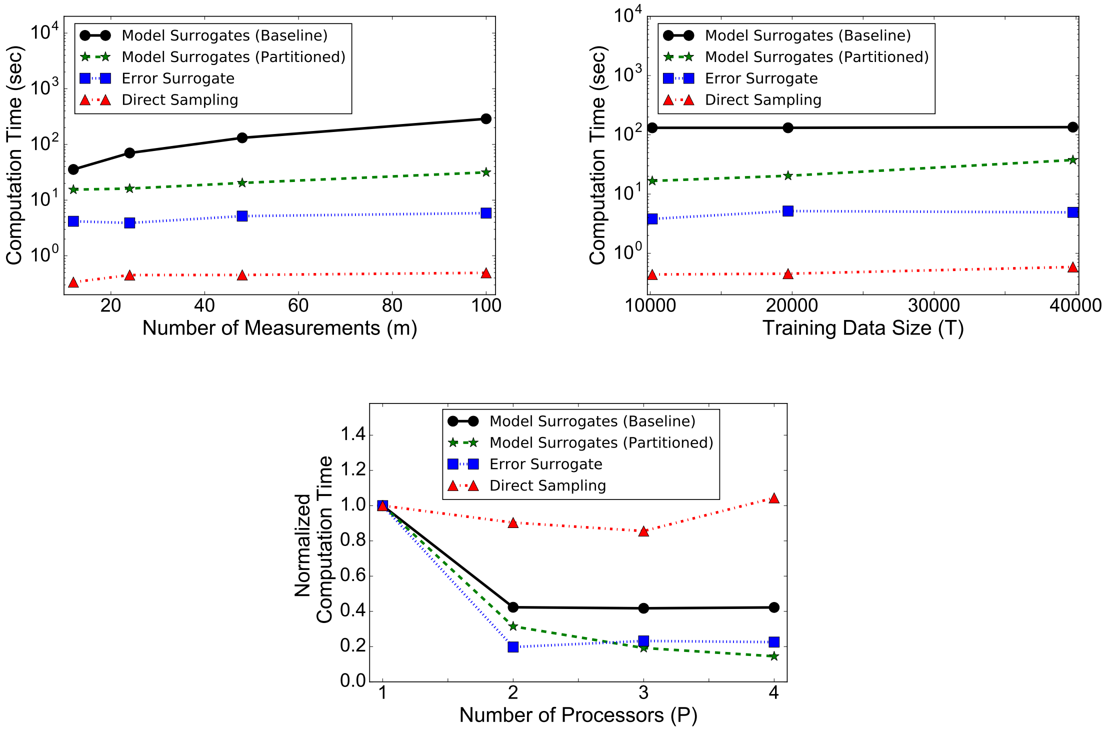


Figure 6. Performance (computation time) comparison of the proposed sampling methods for the damage characterization example for (a) different numbers of measurements (with $T = 19712$ and $P = 4$), (b) different training dataset sizes (with $m = 48$ and $P = 4$) and c) different number of processors (with $m = 48$ and $T = 19712$).

processors used on the resulting normalized computation time. While the direct sampling approach yields the fastest computation times by far, it also shows the smallest dependence on increasing processor count. Here, given the efficiency observed on just one processor, it is likely that the training dataset size and number of measurements needs to be increased significantly before parallel performance gains can outweigh the inherent communication overheads with the approach. While the remaining approaches generally show a performance increase when using more processors, the partitioned model surrogates MCMC approach demonstrates the best scalability as the computation time consistently improves with each added processor. This indicates that this method has the potential to benefit applications where there is a large amount of measurement data and access to adequate HPC resources.

E. Discussion

Some general observations and comparisons of the three proposed sampling methods will now be made in reference to the formulations and performance results from the preceding sections. First, there is an assumption made with each method that the pre-computed/stored training dataset (Equation (10)) is large/extensive enough to facilitate accurate approximations during diagnosis without additional evaluations of the original model, $\mathcal{M}$. This may not be feasible in cases where the unknown damage, $\mathbf{c}$, has a high dimension, when these parameters take a wide range of values or are unbounded, or when $\mathcal{M}$ is too computationally intensive. It can be argued, however, that damage diagnosis is particularly well-suited for this type of approach since typically the dimension of $\mathbf{c}$ will necessarily be low since sensor measurements will not be sensitive to higher order descriptions of damage (*e.g.* inferring crack curvature versus crack length). Furthermore, the parameters describing damage will generally have well-defined bounds (*e.g.* the damage must lie within the component geometry) that can be readily discretized. Finally, the generation of the training dataset is done offline and can be done completely in parallel, fully utilizing available HPC resources to alleviate the computational expense.

While it may be feasible to generate an adequate training dataset for the proposed diagnosis methods, knowing beforehand how large it must be to be considered *adequate* can be more challenging. For the case of the MCMC approach with model surrogates (Section II.B.1), it is straightforward to perform an offline verification study between the surrogate models and the original model. More input/output pairs from the original model can continually be added to the training dataset until a prescribed accuracy has been obtained, along the lines of Figure 2(a). However, since both the MCMC with an error surrogate approach (Section II.B.2) and direct probability sampling (Section II.B.3) utilize the training dataset in an online manner after measurement data have been obtained, this same type of offline verification is not possible. While the methods demonstrated a high degree of accuracy in the diagnosis examples presented here, a systematic method to estimate or bound the error in the recovered probability distribution *a priori* will be a worthwhile topic for future work.

Despite the potential difficulty with offline verification of its accuracy, the direct probability sampling method is the one approach studied here that may enable real time damage diagnosis in its current state. The approach consistently yielded solution times that were well below one second for all test cases in both the damage localization and damage characterization examples considered here, showing potential for time-critical discrete damage event applications. In comparison to the other methods, direct sampling provided six orders of magnitude computational speedup over the reference serial MCMC implementation with FE simulations and was over two orders of magnitude faster than the baseline model surrogates parallel approach. The straightforward, brute force-nature of the approach also circumvents many of the traditional complications associated with MCMC algorithms, including generating an appropriate initial guess, tuning the proposal distribution parameters, fully resolving multimodal distributions, and assessing convergence. As this style of algorithm is amenable to parallel computing, the migration of the method to use GPU computing will be explored in future work as well.

Finally, the performance of the parallel MCMC algorithm[14] for the model surrogates approach is worthy of discussion. It was demonstrated that the method yields more efficient and scalable damage diagnosis estimates relative to the simple baseline parallel approach, but generally at the expense of accuracy. The errors observed here can be mainly attributed to two causes. First, the algorithm was originally designed for big data problems involving so many measurements/observations that they may not all fit on a single machine. Damage diagnosis applications will generally have relatively fewer measurements, and the results presented herein showed that the accuracy generally degraded as the number of measurements decreased and number of processors used increased. In these cases, it is likely that some or all processors may have too few measurement data points to generate accurate estimates of the damage parameters. This inaccuracy could potentially be improved by developing heuristics for partitioning the sensor data appropriately among processors in cases of sparse measurements or by estimating lower limits on the number of measurement data points needed on each processor. The second source of errors observed with the parallel model surrogates MCMC approach was the choice of combination algorithm used in this study. The original work proposed three combination algorithms for creating a final collection of samples from the full-data distribution based off each processor's samples and the crudest, but most efficient, of these algorithms was used in this work. Future work will examine the efficiency/accuracy tradeoffs of each of the combination algorithms for damage diagnosis applications.

IV. Conclusion

Motivated by the Digital Twin structural health management concept, this study presented new approaches to enable high fidelity, probabilistic damage diagnosis in near real time. While the foundation of these methods is based on finite element modeling, Bayesian inference, and surrogate modeling, the focus was on reformulating traditional numerical sampling algorithms to leverage high performance computing to gain substantial computational speedup. To this end, three distinct methods for accelerating sampling were proposed and compared on the application of strain-based crack characterization. The accuracy, computational efficiency, and scalability of the methods were illustrated for two examples of damage localization and damage characterization.

While each parallel approach demonstrated several orders of magnitude improvement in computational efficiency over a sequential Bayesian approach with finite element simulation, the particular strengths and weaknesses of each approach were illustrated and discussed. In particular, the direct probability discretization and sampling approach was the fastest method on the examples tested, consistently yielding probabilistic diagnosis estimates in well under one second, and retained a high degree of accuracy with respect to reference solutions. To this end, the approach shows potential for enabling time-critical diagnosis for discrete damage events for online SHM frameworks like Digital Twin. The difficulty of the direct sampling method with respect to a more conventional model surrogates-based MCMC approach, however, is the absence of a straightforward means of verifying/estimating the accuracy of the approximation *a priori*. Thus, formulating error bounds and studying the convergence properties of the approach with respect to the size of the available training dataset is a worthwhile area of future research.

References

[1] Barthorpe, R. J., *On Model- and Data-Based Approaches to Structural Health Monitoring*, Ph.D. thesis, University of Sheffield, 2010.

[2] Idier, J., *Bayesian Approach to Inverse Problems*, Wiley, New York, 1st ed., 2008.

[3] Gamerman,, D. and Lopes, H. F., *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*, Chapman and Hall/CRC, Boca Raton, Florida, Second ed., 2006.

[4] Haario, H., Saksman, E., and Tamminen, J., "Adaptive proposal distribution for random walk Metropolis algorithm," *Computational Statistics*, Vol. 14, No. 3, 1999, pp. 375–395.

[5] Haario, H., Saksman, E., and Tamminen, J., "An adaptive Metropolis algorithm," *Bernoulli*, Vol. 7, No. 2, 04 2001, pp. 223–242.

[6] Haario, H., Laine, M., and Mira, A., "DRAM: Efficient adaptive MCMC," *Statistics and Computing*, Vol. 16, No. 4, 2006, pp. 339–354.

[7] Vrugt, J., ter Braak, C. J. F., Diks, C. G. H., Robinson, B. A., Hyman, J. M., and Higdon, D., "Accelerating Markov chain Monte Carlo simulation by differential evolution with self-adaptive randomized subspace sampling," *Internation Journal of Nonlinear Sciences and Numerical Simulation*, Vol. 10, No. 3, 2009, pp. 271–288.

[8] Marzouk, Y. M., Najm, H. N., and Rahn, L. A., "Stochastic spectral methods for efficient Bayesian solution of inverse problems," *Journal of Computational Physics*, Vol. 224, 2006, pp. 339–354.

[9] Galbally, D., Fidkowski, K., Willcox, K., and Ghattas, O., "Non-linear model reduction for uncertainty quantification in large-scale inverse problems," *Internation Journal for Numerical Methods in Engineering*, Vol. 81, 2009, pp. 1581–1608.

[10] Wang, J., and Zabaras, N., "Using Bayesian statistics in the estimation of heat source in radiation," *International Journal of Heat and Mass Transfer*, Vol. 48, 2005, pp. 15–29.

[11] Meeds, E., and Welling, M., "GPS-ABC: Gaussian process surrogate approximate Bayesian computation," *CoRR*, Vol. abs/1401.2838, 2014.

[12] Warner, J. E., and Hochhalter, J. D., "Probabilistic damage characterization using a computationally-efficient Bayesian approach," NASA/TP-2016-219169, 2016.

[13] Warner, J. E., Hochhalter, J. D., Leser, W. P., Leser, P. E., and Newman, J. A., "A computationally-efficient inverse

approach to probabilistic strain-based damage diagnosis," *Annual Conference of the Prognostics and Health Management Society*, Denver, CO, October 2016.

[14] Neiswanger, W., Wang, C., and Xing, E., "Asymptotically exact, embarrassingly parallel MCMC," *arXiv preprint arXiv:1311.4780*, 2013.

[15] Kaipio, J. and Somersalo, E., *Statistical and Computational Inverse Problems*, Springer, 2004.

[16] Warner, J. E., Bomarito, G. B., Heber, G., and Hochhalter, J. D., "Scalable Implementation of Finite Elements by NASA - Implicit (ScIFEi)," NASA/TM-2016-219180, 2016.

[17] Python Software Foundation, "Python Language Reference, version 2.7," 2016.

[18] Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G., "API design for machine learning software: experiences from the scikit-learn project," *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[19] Jones, E., Oliphant, T., Peterson, P., et al., "SciPy: Open source scientific tools for Python," 2001–.

[20] Fasano, G., and Franceschini, A., "A multidimensional version of the Kolmogorov-Smirnov test," *Monthly Notices of the Royal Astronomical Society*, Vol. 225, March 1987, pp. 155–170.