X International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2023
M. Papadrakakis, B. Schrefler and E. Oñate (Eds)

ENSEMBLE KALMAN INVERSION FOR REDUCED MULTI-SCALE MODEL VIA DEEP-LEARNING

Yankun Hong, Harshit Bansal AND Karen Veroy

Department of Mathematics and Computer Science
Eindhoven University of Technology
5600 MB Eindhoven, The Netherlands
e-mail: y.hong@tue.nl

Key words: Inverse Estimation, Kalman Inversion, Physics-informed Deep Learning, Model Order Reduction, Multi-scale Model

Abstract. In the context of nonlinear multi-scale problems, the inverse estimation of macroscopic distribution of some microscopic parameters based on macroscopic measurements poses significant challenges. These challenges arise from *(1)* the high computational cost to solve the complex forward problem, and *(2)* the need for derivatives of the complex multi-scale forward model, which combines macro-scale and micro-scale simulations, both of which are typically nonlinear. To address these challenges, we propose a novel approach that combines ensemble Kalman inversion for derivative-free inverse estimation and a physics-informed deep learning-based model order reduction (DL-MOR) to accelerate the micro-scale simulation. We evaluate the performance of our method using a non-linear hyper-elastic model. The results demonstrate the effectiveness of DL-MOR in significantly speeding up the micro-scale simulation and enabling relatively accurate estimation of the microscopic parameter using only macro-scale boundary measurements.

1 INTRODUCTION

In the domain of nonlinear multi-scale problems, an important objective is to estimate the macroscopic distribution of microscopic parameters based on macroscopic measurements and, ultimately, facilitate the construction of the digital twin of the model. However, the naïve solution of multi-scale models incorporating micro-structures, referred to as the full-scale problem, requires fine discretization, resulting in significant computational challenge - extremely large degrees of freedom. To address this issue, the multi-scale method [6, 7, 9, 18] has been proposed. This method tackles the computational difficulties by solving the problem separately at different scales and coupling the results of micro-scale and macro-scale simulations to obtain the final results. Notably, two multi-scale approaches, namely the asymptotic homogenization method [6, 7] and the computational homogenization method, also called the Finite Element Squared (FE^2) method, [9, 18], have been developed. These approaches stem from different aspects and theories but share similar mathematical formulations. In this paper, the FE^2 method serves as the basic building block of an inverse solver in the multi-scale context.

The inverse problem for multi-scale models has also attracted considerable attention in the research community, as is evident from the studies such as [1, 2, 21], among others. In general,

inverse estimation in a multi-scale context poses significant challenges due to two main factors. Firstly, the forward solver used for the multi-scale model is computationally expensive, even with the implementation of the FE2 method. This computational challenge is further exacerbated since numerous forward multi-scale problems need to be solved for parameter estimation. Secondly, the complexity of the multi-scale forward simulation necessitates the computation of derivatives with respect to the unknown parameter, which entails solving a coupled nonlinear macro-scale problem, multiple nonlinear micro-scale simulations, and corresponding adjoint problems. We now briefly discuss the state of the art with respect to these two challenges.

To accelerate the computation of forward solves, recent advancements in scientific machine learning (SciML) and model order reduction (MOR) techniques have gained significant attention. Machine learning-based MOR methods have emerged as a promising approach due to their ability to effectively handle non-linearities and their non-intrusive nature. They eliminate the need for hyper-reduction methods typically required in conventional MOR approaches [12], and for complex hand-designed coding. These machine learning-based MOR methods can be broadly categorized into two types: linear manifold methods, such as POD-GPR [10], POD-NN variants like PDNN [13], PRNN [5], etc.; and non-linear manifold methods, such as latent variable learning via autoencoders [8] and operator learning [17], etc. These methods have found applications in the context of multi-scale models, as demonstrated in studies like [11, 16]. However, the offline training phase associated with these methods still incurs high computational costs, since it necessitates a significant number of expensive full-order solutions for the training data. Moreover, neural networks trained with limited data may struggle to generalize effectively due to the overfitting, and the training phase often overlooks the underlying physical information. To address these challenges, drawing inspiration from recent developments in physics-informed machine learning, e.g., [5, 19], we proposed a non-intrusive, physics-informed two-tier deep network (TTDN) [14] for accelerating forward simulations.

Multi-scale inverse problems have been previously addressed using deterministic and Bayesian inversion techniques, as seen in studies such as [1, 21]. However, these conventional methods typically rely on first-order derivatives of the forward model with respect to the parameters of interest. Obtaining such derivatives in the context of multi-scale models, which involve coupling between micro and macro scales, is challenging. To overcome this limitation, derivative-free methods, including ensemble Kalman inversion (EnKI) [3, 15, 20], have emerged as alternatives to derivative-based optimization methods. EnKI has demonstrated promise in the inverse estimation of multi-scale models, as exemplified by [2], which presents an application and convergence proof of EnKI in this context. Building upon the foundations laid by [2], we aim to enhance parameter estimation by introducing several improvements. Firstly, instead of a linear asymptotically homogenized model, we consider a non-linear model based on computational homogenization. Secondly, in place of intrusive conventional reduction techniques, we use the non-intrusive TTDN approach. Lastly, we enhance the EnKI method by employing the Tikhonov regularized variant in [3] and incorporating advanced techniques such as the non-constant step accelerator proposed in [4]. These modifications collectively contribute to the improvement of parameter estimation in the multi-scale inverse problem.

In summary, our proposed numerical framework leverages three key components: computational homogenization for multi-scale problems, SciML-MOR to accelerate micro-scale computations, and parallelizable, derivative-free EnKI for inverse estimation. We evaluate the performance of this framework on a nonlinear hyper-elastic model. The results demonstrate

significant speed-up and satisfactory generalization accuracy for micro-scale simulations, even with limited full-order training data. These advancements ensure efficient computation of the forward simulations. Moreover, using only boundary measurements from the macro-scale that may be affected by noise, our framework enables effective inverse estimation of micro-scale parameters in the multi-scale context.

2 REDUCED FORWARD MODEL

The present study focuses on the examination of a multi-scale model, which effectively characterizes a material exhibiting a micro-structure at the micro-scale. In order to tackle this model, a numerical method called computational homogenization is employed, while leveraging the application of a Two-Tier Deep Network (TTDN) to significantly reduce the computational burden associated with the forward simulations.

2.1 Multi-scale model

The computational homogenization theory enables the separation of scales through the utilization of the Hill-Mandel condition [9, 18]. This approach divides the simulation into two distinct components: the macro-scale simulation (MSS) and the micro-scale simulation (mSS). In the macro-scale simulation, the boundary conditions of the full-scale problem are imposed, while the constitutive law defined by the micro-scale simulation is applied. Conversely, in the micro-scale simulation, the constitutive law of the full-scale problem is employed, but periodic boundary conditions are enforced in accordance with the Hill-Mandel condition. The macro-scale simulation is implemented in the full-scale domain Ω_{M}, while the micro-scale simulation is implemented in Ω_{m}, which is the representative volume element (RVE) that represents the micro-structure at the macro-scale point.

The multi-scale problem at hand is founded upon a hyper-elastic model. For macro- and micro-scale simulations, the governing partial differential equations are, respectively, of the form:

$$\begin{cases} \nabla \cdot \mathbf{P}_{\mathrm{M}}\left(\mathbf{F}_{\mathrm{M}}(\boldsymbol{x}_{\mathrm{M}}), s(\boldsymbol{x}_{\mathrm{M}})\right) = 0 & \text{on } \Omega_{\mathrm{M}}; \\ \mathbf{P}_{\mathrm{M}} \cdot \boldsymbol{n} = \boldsymbol{t}_n & \text{on } \partial\Omega_n; \\ \boldsymbol{u}_{\mathrm{M}} = \boldsymbol{u}_d & \text{on } \partial\Omega_d; \\ \mathbf{P}_{\mathrm{M}} = \int_{\Omega_{\mathrm{m}}} \mathbf{P}_{\mathrm{m}} \, \mathrm{d}V. \end{cases} \tag{1}$$

and

$$\begin{cases} \nabla \cdot \mathbf{P}_{\mathrm{m}}(\boldsymbol{x}_{\mathrm{m}}) = 0; \quad \mathbf{P}_{\mathrm{m}} = \frac{\partial \mathcal{E}(\mathbf{F}_{\mathrm{m}})}{\partial \mathbf{F}_{\mathrm{m}}}; \quad \mathbf{F}_{\mathrm{m}} = \mathbf{I} + \nabla \boldsymbol{u}_{\mathrm{m}}; \\ \mathcal{E}(\mathbf{F}) = 20\left(\mathrm{Tr}(\mathbf{F}^T \cdot \mathbf{F}) - 3 - 2\ln\left(\det \mathbf{F}\right) + \left(\det \mathbf{F} - 1\right)^2\right); \\ \boldsymbol{u}_{\mathrm{m}}^{+} - \boldsymbol{u}_{\mathrm{m}}^{-} = \varepsilon(\mathbf{F}_{\mathrm{M}} - \mathbf{I}) \cdot (\boldsymbol{x}_{\mathrm{m}}^{+} - \boldsymbol{x}_{\mathrm{m}}^{-}); \\ \mathbf{P}_{\mathrm{m}}^{+} \cdot \boldsymbol{n}^{+} = -\mathbf{P}_{\mathrm{m}}^{-} \cdot \boldsymbol{n}^{-}. \end{cases} \tag{2}$$

Here $\mathbf{P}$, $\mathbf{F}$, and $\boldsymbol{u}$, respectively, represent the stress, strain, and displacement, the subscripts "$_{\mathrm{M}}$" and "$_{\mathrm{m}}$" represent the macro- and micro-scales, respectively, $\boldsymbol{t}_n$ is the imposed traction at the Neumann boundary $\partial\Omega_n$, $\boldsymbol{u}_d$ is the enforced displacement at the Dirichlet boundary $\partial\Omega_d$, ε represents the size of the RVE, $^+$ and $^-$ indicate opposite boundaries of the RVE, and s is a

parameter of the micro-structure and is a function of the macro-scale point $\boldsymbol{x}_{\mathrm{M}}$. In this setting, the parameter for the micro-scale simulation is $\boldsymbol{\mu} = (\mathbf{F}_{\mathrm{M}}, s(\boldsymbol{x}_{\mathrm{M}}))$.

We note here that, to solve the macro-scale PDE, the stiffness $\mathbf{K}_{\mathrm{M}} = \frac{\partial \mathbf{P}_{\mathrm{M}}}{\partial \mathbf{F}_{\mathrm{M}}}$, i.e., the derivative of the micro-scale problem w.r.t. its parameter $\mathbf{F}_{\mathrm{M}}$, is required.

2.2 Two-Tier Deep Network

To accelerate the forward simulation, we use TTDN for micro-scale simulation. The two-tier deep network, which we refer to as NN_{t} (where "t" denotes "total") is a neural network based regression from the parameter $\boldsymbol{\mu}$ to the quantities of interest. It is a combination of two tiers of network. The first tier, denoted by NN_u, achieves the regression $g_u : \boldsymbol{\mu} \mapsto \mathbf{u}^r_{\mathrm{m}}$, the reduced vector of $\boldsymbol{u}_{\mathrm{m}}$. The second tier, denoted by NN_p, realizes the regression $g_p : (\mathbf{u}^r_{\mathrm{m}}, \boldsymbol{\mu}) \mapsto \mathbf{p}^r_{\mathrm{m}}$, the reduced vector of $\mathbf{P}_{\mathrm{m}}$. It can be understood as an approximation of the non-linearity, i.e., the constitutive law, in this problem, by using neural networks. Finally, the entire NN_{t} has the structure $g_{\mathrm{TTDN}} : \boldsymbol{\mu} \overset{\mathrm{NN}_u}{\longmapsto} \mathbf{u}^r_{\mathrm{m}} \to (\mathbf{u}^r_{\mathrm{m}}, \boldsymbol{\mu}) \overset{\mathrm{NN}_p}{\longmapsto} \mathbf{p}^r_{\mathrm{m}}$, where the second arrow represents the concatenation of $\mathbf{u}^r_{\mathrm{m}}$ and $\boldsymbol{\mu}$.

NN_t is trained via a pretraining strategy and semi-supervised learning. The associated loss function is expressed below:

$$\begin{aligned}\mathrm{loss}_{SS}(g_{\mathrm{TTDN}}) =& \frac{1}{M_U} \sum_{(\boldsymbol{\mu}^m, \mathbf{u}^m) \in \Xi_U} \|g_u(\boldsymbol{\mu}^m) - \mathbf{u}^m\|^2 + \frac{1}{M_P} \sum_{(\mathbf{u}^m, \boldsymbol{\mu}^m, \mathbf{p}^m) \in \Xi_P} \|g_p(\mathbf{u}^m, \boldsymbol{\mu}^m) - \mathbf{p}^m\|^2 \\ &+ \frac{1}{M_T} \sum_{\boldsymbol{\mu}^m \in \Xi_T} \|\mathbf{R}(g_{\mathrm{TTDN}}(\boldsymbol{\mu}^m))\|^2.\end{aligned} \tag{3}$$

The two tiers, NN_u and NN_p, are pretrained individually in the beginning; see [14]. For the network NN_u, the training data $\Xi_U = \{(\boldsymbol{\mu}^m, \mathbf{u}^{r,m}_{\mathrm{m}})\}_{m=1}^{M_U}$ includes a sample of parameter $\boldsymbol{\mu}$ and reduced solutions, $\boldsymbol{u}^r_{\mathrm{m}}$, via the full-order solver, which is expensive. The network NN_p is pretrained in an analogous manner using the training data $\Xi_P = \{(\mathbf{u}^{r,m}_{\mathrm{m}}, \boldsymbol{\mu}^m, \mathbf{p}^{r,m}_{\mathrm{m}})\}_{m=1}^{M_P}$. Here, $\mathbf{u}^{r,m}_{\mathrm{m}}$ in Ξ_P is obtained by the pretrained NN_u, i.e., $\mathbf{u}^{r,m}_{\mathrm{m}} = g_u(\boldsymbol{\mu}^m)$, and the reduced stress vector, $\mathbf{p}^{r,m}_{\mathrm{m}}$, is obtained by the evaluation of the constitutive law. The computation of the data Ξ_P is relatively cheap since it does not involve the full-order PDE solver. These training data are also used during supervised learning, as shown in the loss described by (3). The last part of the loss function corresponds to the unsupervised learning, since $\Xi_T = \{\boldsymbol{\mu}^m\}_{m=1}^{M_T}$ contains only the input to the network. Hence, this dataset is generated almost for free. Furthermore, $\mathbf{R}$ is the reduced residual operator of the PDE in $\mathbf{P}_{\mathrm{m}}$. It is given by

$$\mathbf{R}(\mathbf{p}^r_{\mathrm{m}}) := \langle \boldsymbol{t}_n, v^r \rangle_{\partial\Omega_n} - \langle \mathbf{P}^r_{\mathrm{m}}, \nabla v^r \rangle_{\Omega_{\mathrm{m}}}, \tag{4}$$

where v^r is the reduced basis function and $\mathbf{P}^r_{\mathrm{m}}$ is the stress corresponding to $\mathbf{p}^r_{\mathrm{m}}$. It essentially imposes physical information while training the network. It also allows us to generate less expensive full-order data in Ξ_U, while still having good generalization ability by obtaining large, but inexpensive data in Ξ_P and Ξ_T.

Via the TTDN method, the required $\frac{\partial \mathbf{P}_{\mathrm{M}}}{\partial \mathbf{F}_{\mathrm{M}}}$ is accessible by an efficient back-propagation algorithm. For further details about the TTDN method, please refer to [14].

3 INVERSE ESTIMATION

In this section, we will introduce the (inverse) problem setting and briefly explain the mathematical details of the EnKI technique.

3.1 Problem Setting

In this work, the objective is to estimate the unknown parameter, denoted as s in (1), within the macro-scale domain using a set of measured data obtained from the multi-scale forward model $\mathcal{G}(s)$. To this end, it becomes crucial to determine what (specific) data points to measure and where in order to ensure the most accurate inverse estimation. In the context of the solid mechanics problem at hand, we assume that only boundary measurements at $\partial\Omega_{\mathrm{M}}$ are feasible. The boundary measurements encompass $\boldsymbol{u}_{\mathrm{M},1}$, $\boldsymbol{u}_{\mathrm{M},2}$, $\nabla \boldsymbol{u}_{\mathrm{M},1} \cdot \boldsymbol{n}_{\perp}$, and $\nabla \boldsymbol{u}_{\mathrm{M},2} \cdot \boldsymbol{n}_{\perp}$. Here, the subscripts $_1$ and $_2$ denote the components in the first and second directions, respectively, while $\boldsymbol{n}_{\perp}$ represents the orthonormal vector to the boundary direction vector $\boldsymbol{n}$. To identify the measurements, we conduct forward simulations on a sample of s and compute the variances of the candidate measurements from those simulations. Subsequently, we select the measurements with the largest variances, employing a variance-based sensor selection method. We denote the resulting measurement operator as $\mathcal{H}$. Finally, the given data can be represented as $y = \mathcal{H} \circ \mathcal{G}(s^t) + \epsilon$, where s^t stands for the true solution and $\epsilon \sim \mathcal{N}(0, \boldsymbol{\Gamma})$ represents Gaussian noise, with $\boldsymbol{\Gamma}$ denoting the covariance.

With a prior of the form $\mathcal{N}(s_0, \mathbf{C}_0)$, the inverse problem can be written as the Tikhonov regularized optimization problem:

$$\arg\min_{s} \|\mathcal{H} \circ \mathcal{G}(s) - y\|_{\boldsymbol{\Gamma}^{-1}} + \|s - s_0\|_{\mathbf{C}_0^{-1}}. \tag{5}$$

3.2 Ensemble Kalman Inversion

EnKI is a derivative-free method employed for inverse estimation, rendering it adaptable and well-suited for complex forward models. The classical formulation of EnKI tends to disregard the influence of prior information as the iteration progresses, potentially resulting in overfitting. To emphasize the significance of prior knowledge, we utilize Tikhonov regularized EnKI, incorporating improved techniques to enhance convergence speed and robustness [3, 4, 15, 20]. These improvements aim to strike a balance between the observed data and the prior information, thereby improving the overall performance of the inverse estimation.

Tikhonov regularized EnKI extends the given data with the prior by constructing the new observation $\begin{bmatrix} s_0 \\ y \end{bmatrix}$ and the new observation uncertainty $\tilde{\boldsymbol{\Gamma}} = \begin{bmatrix} \mathbf{C}_0 & 0 \\ 0 & \boldsymbol{\Gamma} \end{bmatrix}$. In this way, at each iteration step, we update the estimated result by applying not only the observation information, but also the prior, thus putting the emphasis on the prior. Finally, the Kalman update for n-th iteration step is given by

$$s_n^i = s_{n-1}^i + \mathbf{K}_n \cdot \left(\begin{bmatrix} s_0 \\ y \end{bmatrix} + \tilde{\epsilon}_n^i - \begin{bmatrix} s_{n-1}^i \\ \mathcal{H} \circ \mathcal{G}(s_{n-1}^i) \end{bmatrix} \right) + \alpha_n \hat{\epsilon}_n^i, \tag{6}$$

$$\mathbf{K}_n = \begin{bmatrix} \mathbf{I} & 0 \end{bmatrix} \cdot \mathbf{Q}_n \cdot (\mathbf{P}_n + \beta_n \tilde{\boldsymbol{\Gamma}})^{-1}, \tag{7}$$

where, denoting m_n and H_n as the mean of $\{s_{n-1}^i\}_{i=1}^S$ and $\{\mathcal{H} \circ \mathcal{G}(s_{n-1}^i)\}_{i=1}^S$, respectively, we

define

$$\mathbf{Q}_n := \frac{1}{S}\sum_{i=1}^{S}\left(s_{n-1}^i - m_{n-1}\right)\cdot\left(\begin{bmatrix} s_{n-1}^i \\ \mathcal{H}\circ\mathcal{G}(s_{n-1}^i)\end{bmatrix} - \begin{bmatrix} m_{n-1} \\ H_{n-1}\end{bmatrix}\right)^T, \tag{8}$$

$$\mathbf{P}_n := \frac{1}{S}\sum_{i=1}^{S}\left(\begin{bmatrix} s_{n-1}^i \\ \mathcal{H}\circ\mathcal{G}(s_{n-1}^i)\end{bmatrix} - \begin{bmatrix} m_{n-1} \\ H_{n-1}\end{bmatrix}\right)\cdot\left(\begin{bmatrix} s_{n-1}^i \\ \mathcal{H}\circ\mathcal{G}(s_{n-1}^i)\end{bmatrix} - \begin{bmatrix} m_{n-1} \\ H_{n-1}\end{bmatrix}\right)^T, \tag{9}$$

and start the iteration from the initial ensemble $\{s_0^i\}_{i=1}^S$ drawn independently from $\mathcal{N}(s_0, \mathbf{C}_0)$.

The terms $\tilde{\epsilon}_n^i$, $\hat{\epsilon}_n^i$ and β_n in (6)-(7) correspond to the following modifications:

(i) **Observation perturbation** [15, 20, 22]. One can introduce a stochastic perturbation $\tilde{\epsilon}_n^i \sim \mathcal{N}(0, \tilde{\mathbf{\Gamma}})$ to the observation at each iteration step. Intuitively, this means that, as the observation is corrupted by noise, the perturbation enables the algorithm to explore a larger observation region for the true result.

(ii) **Covariance inflation** [4, 20]. Covariance inflation, $\hat{\epsilon}_n^i$, can improve numerical stability by decreasing the condition number of the covariance during the iteration. It is equivalent to introducing inflation $\alpha_n^2\mathbf{C}_0$ to the covariance at each iteration if $\hat{\epsilon}_n^i \sim \mathcal{N}(0, \mathbf{C}_0)$.

(iii) **Non-constant step size** [4]. As an iterative method, the Kalman gain $\mathbf{K}_n$ in (6) can be understood as an update of the step size. One can thus use a non-constant step size to improve the convergence rate and stability by introducing a factor β_n to $\tilde{\mathbf{\Gamma}}$ in (7).

In [4], the authors provide the guidance for setting the value of the hyperparameters associated to the above modifications. In this work, we set $\beta_n = \beta_0 n^{-\gamma_1}, \alpha_n = \alpha_0\beta_n^{\frac{1}{2}} n^{\gamma_2 - 1}, \beta_0 = 2, \alpha_0 = 0.2, \gamma_1 = 0.8, \gamma_2 = 0.9$.

4 NUMERICAL RESULTS

We examine a forward model as illustrated in Figure 1. We aim to estimate the radius r of the hole in the micro-structure as a function of the macro-scale location $\boldsymbol{x}_{\mathrm{M}}$. Since r has a range which is set to be $0.1 \leq r \leq 0.3$, we introduce a reparametrization

$$r = \frac{0.2}{1+\exp\left(-0.32s\right)} + 0.1, \qquad s = \frac{1}{0.32}\ln\left(\frac{r-0.1}{0.3-r}\right),$$

so that $s \in \mathbb{R}$ is unconstrained and the optimization is simplified. The macro-scale computational domain is divided into 3 blocks, with each block corresponding to a specific parameter value. Consequently, the parameter for the inverse estimation is represented as $s = (s_1, s_2, s_3)$.

To establish the boundary conditions for the macro-scale, a fixed boundary is imposed on the left edge of Ω_{M}, while a traction boundary is applied on the right edge of Ω_{M}. Additionally, the RVE employed in the analysis has a size $\varepsilon = 0.001$.

4.1 Results of TTDN

The parameter range is set to $(\mathbf{F}_{\mathrm{M}} - \mathbf{I}, r) \in (-0.08, 0.08)^3 \times [0.1, 0.3]$. In this setting, we evaluate the generalization ability of the TTDN method by training it with different sizes of the training data and employing an early-stop strategy to avoid overfitting. The obtained results are compared to the PDNN method, an existing machine-learning based MOR approach [13].

Figure 2 displays the algorithmic calculation time on the x-axis, which varies with the size of the training data. The specific data sizes are detailed in Table 1. Table 2 provides

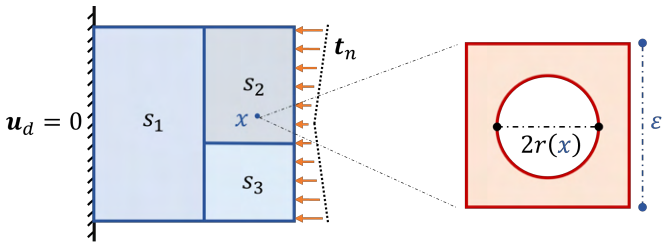

Figure 1: Multi-scale model.

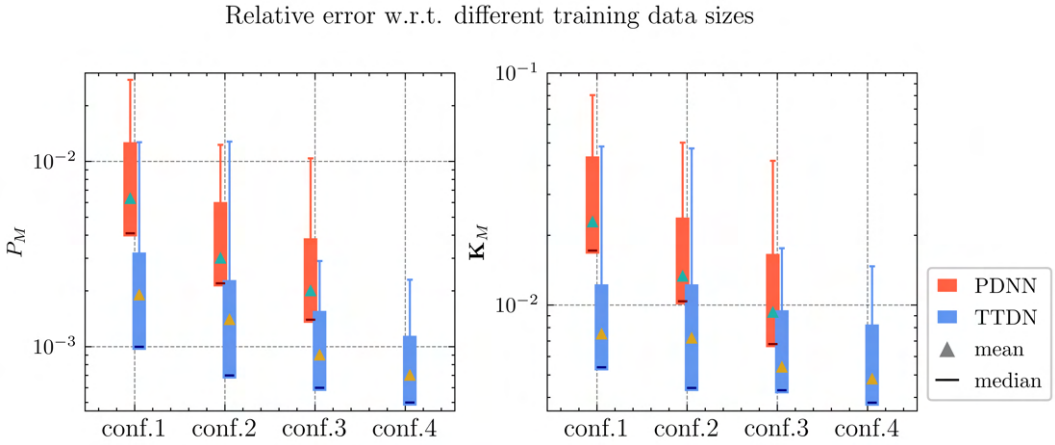

Figure 2: Relative error of $\mathbf{P}_{\mathrm{M}}$ and its derivative $\mathbf{K}_{\mathrm{M}}$ by PDNN method and TTDN methods versus different size of the training data shown in Table 1. The upper edge of the bar is the 95%-quantile and the uppermost line represents the maximum.

the percentage of computational time for each algorithmic process. From these figures and tables, we make the following observations: TTDN effectively reduces computational time, while yielding better generalization accuracy than what PDNN achieves. Notably, with a time budget of 2.5 h (conf. 1), the accuracy achieved by TTDN surpasses that of PDNN, and is even better than that of PDNN in conf. 3, with an offline running time of 10 h. This improvement is attributed to generating significantly fewer full-order data points in Ξ_U, for example, 300 compared to 1500, despite incurring the cost of generating additional training data for the constitutive law Ξ_P, which is significantly less expensive than Ξ_U. Additionally, the involvement of the PDE in training enhances the physical consistency in the prediction, in the sense that the algorithm predicts both $\boldsymbol{u}_{\mathrm{m}}$ and $\mathbf{P}_{\mathrm{m}}$ and employs them to mutually calibrate one another.

4.2 Results of EnKI

As a first test, we impose 3% Gaussian noise to the observation, which means that the standard deviation of the noise is around 3% of the observation value. The prior for this test, and also for all the tests in the following, is $s_0 \sim \mathcal{N}(0, 3\mathbf{I})$, while the true solution is $s^{\mathrm{t}} = [3, 0, -2]^T$. Furthermore, the number of particles in the ensemble is 18, while we have 3 parameters to be estimated. Figure 3 demonstrates the convergence behaviour of EnKI under the aforementioned setting in the context of the problem at hand. Within 10 iteration steps, we see that we reach the relative error of less than 0.5%, even though the observation is corrupted by a 3% noise.

Table 1: Data points in training data sets.

Configuration	Time budget	TTDN		PDNN
		Ξ_U	Ξ_P	Ξ_U
1	$2.5h$	300	1800	375
2	$5h$	600	3600	750
3	$10h$	1200	7200	1500
4	$12.5h$	1500	9000	N/A

Table 2: Time components; 'rest' includes time for NN training, reduced basis generation, etc.

	TTDN			PDNN	
	Ξ_U generation	Ξ_P generation	rest	Ξ_U generation	rest
percentage	70%	15%	15%	90%	10%

In fact, the Tikhonov regularized EnKI for such problem shows robustness under different noise levels. The corresponding results are shown in Figure 4. It can be observed that, although the resulting relative error increases as the noise level increases, we still achieve an $L2$ relative error of less than 1% even when the observation is polluted by 15% noise. In addition, with the size of the ensemble set to be 18, all the tests converge within 10 iteration steps.

Another important hyperparameter for EnKI is the number of particles used in the EnKI technique. As is well known, it has a significant impact on the convergence rate and the computational cost. Figures 5-6 show the corresponding results. Here, the noise level is set to be 3%. Figure 5 shows the behavior of the relative error with respect to the iteration step. The convergence of the one with 6 particles is the slowest. However, it also corresponds to the least expensive iteration step, since the major computational cost for each iteration step is due to the forward simulation for each particle. Figure 6 shows the same curves but with respect to the required number of forward solutions, thereby representing the associated computational costs. We observe that ensembles with excessively large sizes, such as 144 or 72, do not lead to faster convergence. Conversely, using a small ensemble may risk high variance. In this example, with only 3 quantities to be estimated, an intermediate ensemble size, e.g., of 18 particles, is preferred to achieve a faster convergence speed. The reason why fewer particles seem to work well may be due to the fact that the prior is close enough to the underlying *true* solution and, hence, is informative. Consequently, fewer particles drawn from the prior can represent the behavior of the forward solution sufficiently well. Further studies are required for an exhaustive and clearer understanding of the observed behaviors.

Finally, we discuss the computational costs of the whole method. The offline cost, already presented in Section 4.1, is around 12.5 h. The online cost is attributed to the implementation time of the (reduced) EnKI technique. Considering the setting of Figure 6 as an example, which includes 180 forward solutions for the multi-scale problem, each forward solve consumes 0.5 min per CPU core. Upon running on a 6-core CPU, the online time is reduced to 15 min. Hence, the total computational cost is around 13 h. However, if MOR would not have been employed in conjunction with the EnKI technique, recalling that micro-scale PDE and its

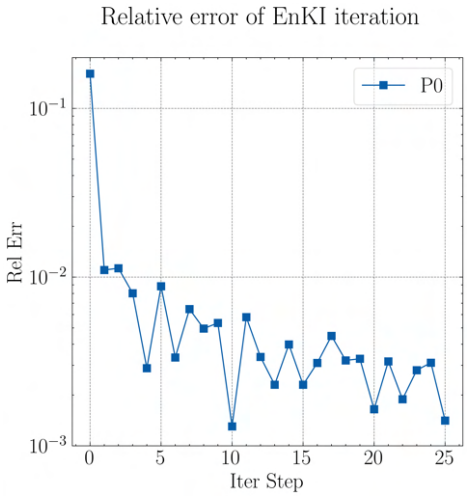

Figure 3: Convergence of the EnKI of L_2 error.

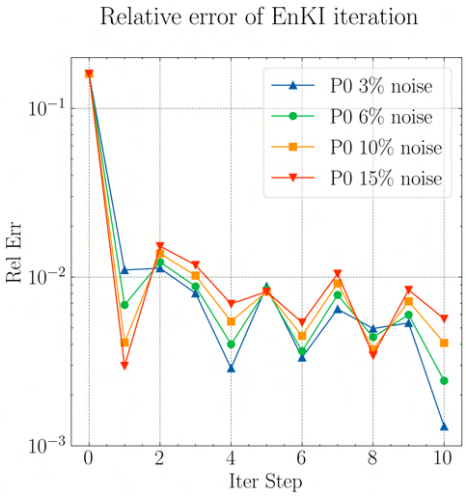

Figure 4: Sensitivity to the noise.

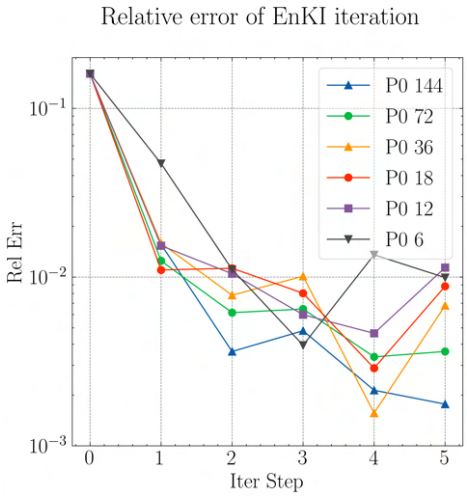

Figure 5: Comparison of different sizes of ensemble w.r.t. iteration steps.

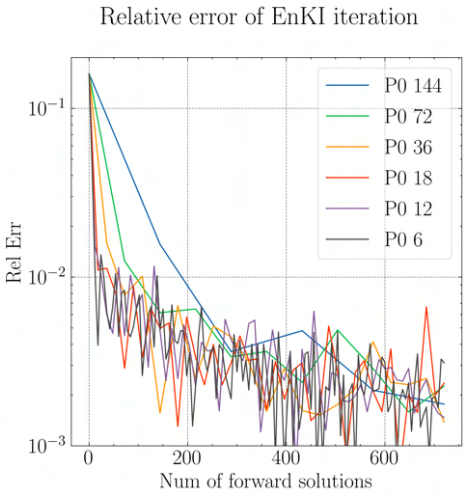

Figure 6: Comparison of different sizes of ensemble w.r.t. computational cost.

adjoint problems should be solved at all the quadrature points of the macro-scale domain at every Newton iteration, a forward simulation would need more than 100 000 PDE solutions, which is unaffordable on a 6-core CPU. It is also worth mentioning that the MOR techniques will still play a crucial role in speeding up the simulations and, thereby, accelerating the inverse estimation, even if one uses the best possible parallelized implementation (of the full-order solver) applied on the most novel hardware architecture.

5 CONCLUSIONS

In this study, we employed Two-tier Deep Network (TTDN), a deep-learning-based Model Order Reduction (MOR) method, to address the computational burden associated with a complex multi-scale model. The application of TTDN yields substantial speed-ups in forward simulations without compromising accuracy, all while minimizing data generation costs. These speed-ups facilitate the use of Ensemble Kalman Inversion (EnKI) for inverse estimation in the multi-scale model. Additionally, we demonstrate an effective and efficient inverse estimation

for the micro-scale parameter, utilizing only macro-scale boundary measurements. It is worth noting that the proposed algorithm is implementation-friendly, as EnKI is a derivative-free method, and TTDN is a non-intrusive MOR method.

ACKNOWLEDGEMENT

This research has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Programme (Grant Agreement No. 818473).

REFERENCES

[1] Abdulle, A. and Blasio, A.D. Numerical homogenization and model order reduction for multiscale inverse problems. *Multiscale Modeling & Simulation* (2019) **17(1)**:399–433.

[2] Abdulle, A. and Garegnani, G. and Zanoni, A. Ensemble Kalman filter for multiscale inverse problems. *Multiscale Modeling & Simulation* (2020) **18(4)**: 1565-1594.

[3] Chada, N.K. and Stuart, A.M. and Tong, X.T. Tikhonov regularization within ensemble Kalman inversion. *SIAM Journal on Numerical Analysis* (2020) **58(2)**:1263-1294.

[4] Chada, N. and Tong, X. Convergence acceleration of ensemble Kalman inversion in nonlinear settings. *Mathematics of Computation* (2022) **91(335)**:1247–1280.

[5] Chen, W. and Wang, Q. and Hesthaven, J.S. and Zhang, C. Physics-informed machine learning for reduced-order modeling of nonlinear problems. *Journal of Computational Physics* (2021) 446:110666.

[6] Efendiev, Y. and Pankov, A. Numerical homogenization of monotone elliptic operators. *Multiscale Modeling & Simulation* (2003) **2(1)**:62–79.

[7] Flodén, *L. G-convergence and Homogenization of Some Sequences of Monotone Differential Operators*. PhD thesis, Mid Sweden University, Department of Engineering and Sustainable Development, (2009).

[8] Fresca, S. and Dedé, L. and Manzoni, A. A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs. *Journal of Scientific Computing* (2021) 87:61.

[9] Geers, M.G.D. and Kouznetsova, V.G. and Matouš, K. and Yvonnet, J. Homogenization methods and multiscale modeling: Nonlinear problems. *Encyclopedia of Computational Mechanics Second Edition* (2017) 1–34.

[10] Guo, M. and Hesthaven, J.S. Reduced order modeling for nonlinear structural analysis using Gaussian process regression. *Computer Methods in Applied Mechanics and Engineering* (2018) **341**:807–826.

[11] Guo, T. and Rokoš, O. and Veroy, K. Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method. *Computer Methods in Applied Mechanics and Engineering* (2021) 384:113924.

[12] Hesthaven, J.S. and Rozza, G. and Stamm, B. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer Briefs in Mathematics. Springer International Publishing, Cham, 2015

[13] Hesthaven, J.S. and Ubbiali, S. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics* (2018) **363**:55–78.

[14] Hong, Y. and Bansal, H. and Veroy, K. Physics-informed two-tier neural network for nonlinear model order reduction. *in preparation.*

[15] Iglesias, M.A. and Law, K.J.H. and Stuart, A.M. Ensemble Kalman methods for inverse problems. *Inverse Problems* (2013) **29(4)**:045001.

[16] Lißner, J., and Fritzen, F. Data-driven microstructure property relations. *Mathematical and Computational Applications* (2019) 24(2), 57.

[17] Lu, L. and Jin, P. and Pang, G. and Zhang, Z. and Karniadakis, G.E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence* (2021) **3(3)**:218–229.

[18] Miehe, C, and Koch, A. Computational micro-to-macro transitions of discretized microstructures undergoing small strains. *Archive of Applied Mechanics* (2002) **72(4)**:300–317.

[19] Raissi, M. and Perdikaris, P. and Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* (2019) **378**:686–707.

[20] Schillings, C. and Stuart, A.M. Analysis of the ensemble Kalman filter for inverse problems. *SIAM Journal on Numerical Analysis* (2017) **55(3)**:1264–1290.

[21] Schmidt, U. and Steinmann, P. and Mergheim, J. Two-scale elastic parameter identification from noisy macroscopic data. *Archive of Applied Mechanics* (2016) **86**:303–320.

[22] Silva, F.A.B. and Pagliantini, C. Grepl, M. and Veroy, K. A reduced basis ensemble Kalman method. *arXiv* (2022).