An efficient and robust method for parameterized nonintrusive reduced-order modeling

Wawrzyniec J. Kostorz | Ann H. Muggeridge | Matthew D. Jackson

Novel Reservoir Modelling and Simulation Group, Department of Earth Science and Engineering, Imperial College London, London, UK

Correspondence
Wawrzyniec J. Kostorz, Novel Reservoir Modelling and Simulation Group, Department of Earth Science and Engineering, Imperial College London, Prince Consort Road, London SW7 2BP, UK.
Email: w.j.kostorz@gmail.com

Summary
A method of constructing parameterized nonintrusive reduced-order models (NIROMs) is given. The approach relies on a geometrical interpretation of NIROM, requires only a single layer of interpolation to be applied for both system state and parametric dependence of the model and is applicable to systems characterized by any number of parameters spanning arbitrary orders of magnitude. The method is applied to three representative test problems and evaluated in terms of accuracy and speed, showing good performance.

KEYWORDS
advection-diffusion equation, explicit, fluids, time integration

1 | INTRODUCTION

Many fluid dynamics applications require the numerical solution of systems of partial differential equations (PDEs) to predict how the spatial distribution of the fluid properties of interest vary in time. These applications include meteorology, oceanography, fluid flow through pipelines, petroleum reservoir engineering, contaminant transport in the subsurface, oceans or atmosphere and defence. In most cases engineers will use commercial computer simulation programs in which the PDEs are discretized and properties determined on a two- (2D) or three-dimensional (3D) mesh which can comprise millions of elements. Each of these simulations can take many hours of CPU time, even on the most powerful computers.

This may mean that it is prohibitively expensive to properly evaluate the impact of uncertainty or risk on the outcomes if a Monte Carlo type approach is used to investigate the impact of varying input parameters.

This has driven the development of reduced-order methods to enable the fast, but approximate estimation of system behavior. Many of these methods are “intrusive” in that they involve modifying the simulation program. This makes them unsuitable for engineers who have to use commercial programs for which the source is not available or for applications based on complex legacy codes with incomplete documentation. In these cases, nonintrusive reduced-order methods (NIROMs), which construct a model based on “snapshots” of system behavior obtained from the “high fidelity” simulation programs for a set of known input parameters, can be used.

Broadly speaking, NIROM is a surrogate for a high-fidelity forward numerical model (simulator), attempting to mimic the behavior of the latter based on a collection of data. In order to construct a nonintrusive reduced model, the high-fidelity model is first probed by using it to generate a number of (typically computationally expensive) realizations. Next, the data is compressed (“reduced”) and the behavior of the simulator is interpolated in an abstract high-dimensional space. The resulting interpolant serves as a rapid proxy for the original model and trades off orders-of-magnitude speedups for mild inaccuracies arising due to the compression and interpolation processes. In contrast with typical (intrusive) ROM, NIROM
NIROM can be understood as a proxy for a high-fidelity forward numerical model (simulator). Generally, the role of any simulator is to take a discretized state of the system modeled and propagate it forward in time by a time step $\Delta t$. A discretized state of the system at time $t$ can be represented as a high-dimensional vector, here denoted $\mathbf{u}(t)$, with the number of its entries being equal to the number of mesh nodes/grid cells. A series of such discretized states (referred to as data snapshots) separate in time by a fixed time interval $\Delta t$ is denoted as $\mathbf{U}$ where

$$
\mathbf{U} = [\mathbf{u}(0), \mathbf{u}(\Delta t), \mathbf{u}(2\Delta t), \ldots].
$$

A forward numerical model can therefore be understood as a function $\mathbf{F}$ satisfying

$$
\mathbf{u}(t + \Delta t) = \mathbf{F}(\mathbf{u}(t); \Delta t; \mathbf{p}),
$$

where $\mathbf{p}$ encodes the time-invariant parameters of the system such as fluid/domain properties (viscosity, density, diffusivity, etc.).
Intuitively, Equation (2) says that given the current system state \( u(t) \), the time step \( \Delta t \) and the system parameters \( p \) the simulator is capable of predicting the future state of the system at time \( t + \Delta t \). In our notation the problem is expressed in an explicit fashion, but in practice this is hardly ever the case. In numerical simulation evaluating \( u(t + \Delta t) \) is often formulated in an implicit way and associated with solving a large sparse linear system. Relevant examples include discrete approximations of solutions to elliptic equations for pressure in Navier-Stokes or Darcy flow simulations. As a consequence, (2) is a high-level symbolic representation that is true for some \( F \) encapsulating all components of a numerical simulator. The main objective of NIROM is constructing a computationally cheap explicit proxy for \( F \) just from a set of simulation data. Low cost and speed of the method are achieved by employing a dimensionality reduction process and a cheap interpolation method, as discussed below.

First, the high-dimensional state of the system \( u(t) \) is represented in an approximate low-dimensional form \( \alpha(t) \). The mapping

\[
\mathbf{u}(t) \rightarrow \mathbf{\alpha}(t),
\]

is generally arbitrary, as long as \( \text{dim}(\mathbf{\alpha}(t)) \ll \text{dim}(\mathbf{u}(t)) \) and the reverse mapping can be provided. In practice, probably the most common choice of such dimensionality reduction is the proper orthogonal decomposition (POD), which is known to be the optimal reduction process with respect to the \( L_2 \) norm. In our notation POD can be briefly summarized as follows.

Given a snapshot matrix \( U \)

\[
U = [u(0), u(\Delta t), \ldots, u((N_t - 1)\Delta t)],
\]

we first construct the mean data snapshot \( \bar{u} \)

\[
\bar{u} = \frac{\sum_{n=0}^{N_t-1} u(n\Delta t)}{N_t},
\]

and the mean-shifted data array \( \tilde{U} \)

\[
\tilde{U} = [u(0) - \bar{u}, u(\Delta t) - \bar{u}, \ldots, u((N_t - 1)\Delta t) - \bar{u}],
\]

giving rise to the so-called kernel matrix \( K \)

\[
K = \tilde{U}\tilde{U}^T \in \mathbb{R}^{N \times N},
\]

where \( N \) is the dimensionality (length) of each data snapshot.

Next, since \( K \) is square and positive-semidefinite it has an associated eigenvector basis \( \{(v_j, \sigma_j^2)\}_{j=1}^N \) satisfying

\[
Kv_j = \sigma_j^2 v_j, \quad j = 1, \ldots, N,
\]

and without loss of generality it is assumed that \( \sigma_j^2 \) are given in descending order. Subsequently, the reduced dimension \( d \) (typically with \( d \ll N \)) is defined as the smallest positive integer satisfying

\[
\frac{\sum_{j=1}^d \sigma_j^2}{\sum_{j=1}^N \sigma_j^2} < 1 - \epsilon,
\]

where \( \epsilon \ll 1 \) is a small user-specified tolerance. After \( d \) is identified, the transformation matrix \( V \) is defined as

\[
V = [v_1, \ldots, v_d],
\]

and the mapping (3) is given by

\[
\mathbb{R}^N \ni u(t) \rightarrow \alpha(t) = V^T(u(t) - \bar{u}) \in \mathbb{R}^d,
\]
while the inverse mapping is given by
\[ \mathbb{R}^d \ni \alpha(t) \rightarrow \mathbf{u}(t) = \mathbf{V}\alpha(t) + \mathbf{\bar{u}} \in \mathbb{R}^N. \] (12)

We point out that applying the series of transformations \( \mathbf{u}(t) \rightarrow \alpha(t) \rightarrow \mathbf{u}(t) \) (ie, compression followed by decompression) will produce a data snapshot different from the original \( \mathbf{u}(t) \) if it is not spanned by the columns of \( \mathbf{V} \). However, in most practical applications the discrepancy is minimal and can be minimized by decreasing the threshold parameter \( \epsilon \). With the above formulation POD has been applied in a variety of applications in NIROM and other contexts; for more details the reader is kindly referred to Reference 16.

A low-dimensional representation \( \alpha(t) \) gives rise to a new simulator-defining function \( \mathbf{\hat{F}} \) operating in a lower number of dimensions and satisfying
\[ \alpha(t + \Delta t) = \mathbf{\hat{F}}(\alpha(t); \Delta t; \mathbf{p}). \] (13)
Since \( \mathbf{\hat{F}} \) takes inputs that are significantly lower dimensional than those of \( \mathbf{F} \) the former is generally much cheaper and therefore faster to evaluate, but may still rely on a fairly expensive problem such as solution of a dense linear system.

In order to address this, NIROM aims at locally approximating \( \mathbf{\hat{F}} \) in a cheap and fully explicit fashion. This is achieved by first providing a collection of simulation snapshots \( \mathbf{U} \) and reducing them to \( \alpha(0), \alpha(\Delta t), \alpha(2\Delta t), \ldots \); assuming that (13) holds, we write
\[ \alpha((n+1)\Delta t) = \mathbf{\hat{F}}(\alpha(n\Delta t); \Delta t; \mathbf{p}). \] (14)
In the above the time step \( \Delta t \) is assumed to be constant and may be dropped for simplicity of notation. Since the samples of the reduced system state variables are extracted from \( \mathbf{U} \) (known), the only unknown in the above equation is \( \mathbf{\hat{F}} \) itself which is approximated via multidimensional interpolation methods such as the RBF. The interpolant is fully explicit and should be computationally much cheaper than the function it is approximating. It is clear that the NIROM approach is beneficial only if the computational cost of building and running the proxy is much smaller than the cost of running the high-fidelity model. In practical applications this is almost universally guaranteed, especially for high grid resolutions. It should be pointed out that \( \mathbf{\hat{F}} \) approximated in this fashion can in principle be used to propagate system states that are different that the ones used for interpolation (training), assuming that the surrogate is sufficiently accurate.

### 3 | PARAMETERIZED RBF-BASED NIROM

In this section we discuss the previous approach to parameterized RBF-based NIROM and point out its limitations by interpreting the construction procedure geometrically. Subsequently, a new method mitigating the aforementioned limitations is proposed.

#### 3.1 | Previous approach

The RBF-based parameterized NIROM construction procedure was first introduced by Xiao et al\(^7\) in order to study a parameterized porous media flow problem. Generally, NIROM can be understood as a process of constructing a surrogate function \( f(\cdot;\cdot) \), which we refer to as the propagator, satisfying
\[ \alpha^{n+1} = f(\alpha^n; \mathbf{p}), \] (15)
where \( \alpha^n \) is a representation of the state of the system at time \( n\Delta t \), \( \Delta t \) is a fixed time step for the model specified a priori and \( \mathbf{p} \) is a vector of parameters characterizing the system. Typically, a low-dimensional representation of the system state \( \alpha \) is constructed via POD. Clearly for the problem to be truly parameterized we require \( \mathbf{p} \) to be allowed to vary, otherwise the dependence on \( \mathbf{p} \) can be dropped.
FIGURE 1  Visualization of the two-stage interpolation process. A collection of function samples (dots) representing the training data snapshots are given at a number of fixed parameter levels $p$ (a). In the first step interpolation is performed separately at each level of $p$ (b), so that $f$ is fully defined at each constant parameterization. Practically, this step constructs a separate nonparametric nonintrusive reduced-order model, characterized by $f_s$, for each individual dynamical system with a fixed sampled parameterization $p_s$ and satisfying $f_s(\cdot) = f(\cdot, p_s)$. In the second step (c) the interpolation of $f_s(\alpha)$ with respect to $p$ can be performed at any fixed value of $\alpha$. The latter has to be supplied a priori and is treated as a constant at this stage [Colour figure can be viewed at wileyonlinelibrary.com]

It is clear that for a parameterized $f(\cdot, \cdot)$ multiple training simulations characterized by different parameterizations must be provided. We denote the number of such simulations to be $N_s$ with the corresponding compatibility condition

$$\alpha_s^{n+1} = f(\alpha_s^n, p_s), \ s = 1, \ldots, N_s,$$

where $\alpha_s^n$ is the system state in training run $s$ at time instance $n\Delta t$ and $p_s$ is the parameterization associated with simulation $s$. In the process of constructing the propagator is split into two steps

1. For each $s = 1, \ldots, N_s$ construct a surrogate $f_s(\cdot)$ satisfying

$$\alpha_s^{n+1} = f_s(\alpha_s^n),$$

via the RBF interpolation method

2. For a generic system state $\alpha$ to be propagated and its parameterization $p$ first evaluate

$$\beta_s = f_s(\alpha),$$

for all $s$. Afterward, construct (via the RBF method) the function $g(\cdot)$ satisfying

$$g(p_s) = \beta_s,$$

where $p_s$ is the parameterization corresponding to $f_s$. Finally evaluate

$$f(\alpha; p) \approx g(p),$$

which is the estimate of the system state at the next time instance.

The collection of surrogates $\{f_s\}_{s=1}^{N_s}$ can be constructed a priori and stored as a collection of RBF coefficients. For a generic system state $\alpha$, $N_s$ values of $f_s(\alpha)$ are cheaply evaluated first (18) and their values are stored. Next, a new surrogate dependent on parameterization only is constructed locally (for this specific $\alpha$) from $\{(p_s, f_s(\alpha))\}_{s=1}^{N_s}$ and evaluated at $p$. Intuitively, the method just described separates the interpolation process into two processes applicable to two disjoint subsets of the total number of dimensions and requires a parameter-dependent interpolant to be constructed and evaluated for each system state modeled. A schematic representation of this procedure is given in Figure 1.
**FIGURE 2** Illustration of the miscommunication problem. Two data samples in the blue and red planes (A) are in the vicinity of the black point in the green plane. However, the value of the propagator \( f \) at that location can only be influenced by the samples in the green plane. Since the black point is relatively far away from the green sample, the error of approximation is expected to be high in that area. The resulting second-stage interpolant may be therefore characterized by severe inaccuracies (B). In (B), the sample taken at 0 corresponds to the black point in (A) and the horizontal axis corresponds to the vertical parametric axis. Note that the local inaccuracy influences the overall accuracy in a nonlocal way [Colour figure can be viewed at wileyonlinelibrary.com]

**FIGURE 3** Severely suboptimal (A) and more appropriate (B) structure of the parametric samples. For different magnitudes of \( p^{(1,2)} \) and sensitivities of the interpolant to \( p^{(1,2)} \) the radial basis function method may be severely ill-conditioned. A more uniform sample structure (B) is desirable, with the ideal case being a regular square grid of samples

It can be observed that the method discussed suffers from a severe problem of miscommunication in the parametric space, precisely due to two split interpolation processes. In order to illustrate this, consider a toy problem where \( \alpha \in \mathbb{R}^2 \), \( p \in \mathbb{R}^2 \) as in Figure 1. Let three uniformly spaced values of \( p_{1,2,3} \) be given in an ascending order, and let \( f_{1,2,3}(\cdot) \) be sampled at positions \( \alpha_{1,2,3} \) with \( \alpha_1 = \alpha_3 \), and \( \|\alpha_2 - \alpha_1\| \gg \|p_2 - p_1\| \) (Figure 2A). It is expected that \( f_2(\alpha_1) = f(\alpha_1; p_2) \) will be severely inaccurate if no samples of it are provided in the vicinity of \( \alpha_2 \) at the parameterization level \( p_2 \). However, this inaccuracy can in principle be lowered significantly if the information from the adjacent layers 1 and 3 could influence the interpolant at layer 2. The method discussed does not allow for the information to propagate across layers in this fashion. Moreover, if the error at position \( \alpha_1 \) in layer 2 is large, the second-stage interpolation across layers may be close to non-smooth, which can further lower the accuracy at other adjacent layers (Figure 2B).

We point out that that the unified notation \( \|\cdot\| \) is used throughout this work for the Euclidean norm in arbitrary number of dimensions. Intuitively, the inequality \( \|\alpha_2 - \alpha_1\| \gg \|p_2 - p_1\| \) indicates that the Euclidean distance between the two reduced representations of system states is much greater than the separation between the (hyper)planes in which \( \alpha \) evolves corresponding to different parameterizations \( p \).

We also observe that a naive RBF approach applied to the parameterization vector \( p \) is another source of inconsistencies and potential errors. We demonstrate this by considering a two-dimensional parameter vector \( p = [p^{(1)}, p^{(2)}]^T \) with parameter ranges

\[
\begin{align*}
p^{(1)} &\in [1, 10] \\
p^{(2)} &\in [10^{-6}, 10^{-4}],
\end{align*}
\]  

(21)

where \( p^{(1,2)} \) may represent two physical parameters spanning different orders of magnitude. It follows that \( \|p\| \approx |p^{(1)}| \), and since the RBF method relies on pairwise distances between datapoints this implies that the influence of \( p^{(2)} \) on the interpolant is negligible. However, if the system is sensitive to the values of \( p^{(2)} \) then the resulting interpolant will necessarily fail to capture this dependency and the interpolation problem becomes severely ill-conditioned (Figure 3).
3.2 | New approach

Our approach mitigates the two issues discussed in the previous section, namely the problems of miscommunication and non-uniform data scaling. This is achieved by renormalizing the parameterization $\mathbf{p}$ and the samples $\alpha^n$ in an appropriate fashion and allowing any data sample $\alpha^n$ to influence the value of $f(\cdot;\cdot)$ at any position in the $(\alpha;\mathbf{p})$ space.

First, instead of employing the ansatz (15), we modify it to be

$$\alpha^{n+1} = \alpha^n + f(\alpha^n; \mathbf{p}).$$

which is accurate up to zeroth order in the time interval $\Delta t$.

Second, the normalization is applied to the system states $\alpha$. We write $\mathbf{A}_s$ to be the array of discretized system states

$$\mathbf{A}_s = [\alpha^0_s, \ldots, \alpha^{N_s-1}_s],$$

where $s = 1, \ldots, N_s$, $N_s$ is the number of training runs provided and $N_s^t$ is the number of system state snapshots in training run $s$. For each $s$ we compute

$$\Delta_s = \sum_{n=1}^{N_s^t-1} ||\alpha^n_s - \alpha^{n-1}_s||,$$

and then evaluate

$$\Delta_\alpha = \frac{\sum_{s=1}^{N_s^t} \Delta_s}{\sum_{s=1}^{N_s^t} (N_s^t - 1)}.$$  

It is clear that $\Delta_\alpha$ is the mean magnitude of $f(\cdot;\cdot)$ defined as in (22). All system state data is then renormalized via

$$\alpha \leftarrow \frac{\alpha}{\Delta_\alpha},$$

and modeled in such nondimensional form with the typical value of $f(\cdot;\cdot)$ being normalized to unity.

Next, the parameterization $\mathbf{p}$ is normalized. The parameter vector $\mathbf{p}$ is split into a number $P$ of separate, and potentially multidimensional, parameters

$$\mathbf{p} = [\mathbf{p}^{(1)}, \ldots, \mathbf{p}^{(P)}].$$

An appropriate splitting has either $P = \dim(\mathbf{p})$ with each $\{\mathbf{p}^{(i)}\}_{i=1}^P$ being one-dimensional or separates quantities with different physical meaning (e.g., density, viscosity). In order to illustrate the two alternatives consider the parameterization

$$\mathbf{p} = [g_x, g_y, g_z, \mu, \rho],$$

where $g_x, g_y, g_z$ are three gravity components, $\mu$ is the fluid viscosity and $\rho$ is the fluid density. The first (trivial) type of splitting separates $\mathbf{p}$ into five independent parameters. The second type of splitting is

$$\mathbf{p}^{(1)} = [g_x, g_y, g_z]$$

$$\mathbf{p}^{(2)} = \mu$$

$$\mathbf{p}^{(3)} = \rho,$$

and takes into account the fact that $\mathbf{g} = [g_x, g_y, g_z]$ can be treated as a single vector parameter with all entries representing the same physical quantity.
For \( i = 1, \ldots, P \) each set \( \{ \mathbf{p}_s^{(i)} \}_{s=1}^{N_s} \) is rescaled separately. This is achieved by first computing the mean separation \( \Delta_p^{(i)} \) between the nearest neighbors in each set as follows

\[
\Delta_p^{(i)} = \frac{\sum_{s=1}^{N_s} \min_{s' \neq s} \| \mathbf{p}_s^{(i)} - \mathbf{p}_{s'}^{(i)} \|}{N_s},
\]

and then renormalizing

\[
\mathbf{p}_s^{(i)} \leftarrow \frac{\mathbf{p}_s^{(i)}}{\Delta_p^{(i)}}.
\]

Finally, for the interpolation process we define the notion of distance \( d(\cdot; \cdot) \) between two normalized state-parameter pairs as

\[
d^2((\alpha_1, \mathbf{p}_1); (\alpha_2, \mathbf{p}_2)) = \| \alpha_1 - \alpha_2 \|^2 + \| \mathbf{p}_1 - \mathbf{p}_2 \|^2,
\]

the above definition is used in conjunction with the RBF interpolation method and applied to the state-parameter pairs extracted from the collection of training data. The resulting structure of the data samples is schematically visualized in Figure 3B. We point out that the rescaling factors \( \Delta_{\alpha}, \Delta_p^{(i)} \) can be premultiplied by an additional factor of order \( O(1) \) before the normalization process. Such treatment will emphasize the influence (importance) of the corresponding variable/parameter if the factor is \( \gg 1 \) and make the problem less sensitive to the corresponding variable for factors \( \ll 1 \).

The new approach can be therefore summarized as follows

1. Given the reduced data arrays and the corresponding parameterizations \( (\mathbf{A}, \mathbf{p}) \) \( s=1 \cdots N \), compute the normalization factors \( \Delta_{\alpha}, \Delta_p^{(i)} \) and renormalize all training data accordingly
2. Construct a single RBF/PS interpolant satisfying

\[
\alpha_s^{n+1} = \alpha_s^n + f(\alpha_s^n; \mathbf{p}_s),
\]

from all available combinations \((s,n)\) by employing the joint state-parameter notion of distance (32) in the RBF method
3. Given a generic state-parameterization pair \((\alpha, \mathbf{p})\) appropriately renormalize the inputs by \( \Delta_{\alpha}, \Delta_p^{(i)} \) and propagate \( \alpha \) one step forward in time by employing the RBF/PS surrogate with the notion of distance (32)

We point out that our method is conceptually much simpler than its predecessor since it relies on just a single (universal) surrogate being evaluated once per simulation time step. The model is fully constructed \emph{a priori} and no local surrogates need to be constructed or evaluated, which leads to a much lower computational cost and a significant speedup, as will be demonstrated later.

4 | RESULTS

In this section the method proposed is tested and evaluated on three test problems. A comparison with the previous method is also provided. The generic workflow for each test problem can be summarized as follows

1. Set the simulation time \( T \) and the time step \( \Delta t \) (output time step for the high-fidelity model and the forward time step for NIROM). Set the ranges of all model parameters and define the box parametric domain.
2. Specify a number of points in the parametric domain corresponding to an ensemble of parameterizations for the system modeled. In this work the regular (uniform) grid sampling is used.
3. For each parameter combination specified in the previous step generate a number of high-fidelity snapshot data arrays (training runs) with potentially different initial conditions.
4. Generate a NIROM from the training runs via the two different methods discussed. In this work POD with \( \epsilon = 10^{-3} \) (9) was employed for the reduction process.
5. Generate an ensemble of unseen parameterizations and/or initial conditions within the range modeled and propagate them in time via the high-fidelity model and NIROM.
6. Evaluate NIROM’s accuracy and speedup, with the latter being the ratio of CPU times required to propagate the realizations in time via NIROM and the high-fidelity tool.

As described in step 4, the tolerance $\epsilon = 10^{-3}$ for POD was used in this work. This choice of $\epsilon$ may often result in the required value of $d$ being such that $d \ll N$ no longer holds, resulting in decreased efficiency of POD compression. However, in our examples the efficiency of POD is high enough to impose a low threshold (corresponding to low compression noise) while comfortably maintaining $d \ll N$.

### 4.1 One-dimensional advection-diffusion

First, for validation purposes, we model the one-dimensional advection-diffusion equation on $[0, 1]$ with two Dirichlet boundary conditions

$$u_t + U u_x = \nu u_{xx}$$

$x \in [0, 1], \quad t \in [0, 1]$

$$u(0; t) = 1, \quad u(1, t) = 0$$

$$U, \nu \in [0.05, 0.25],$$

where $U, \nu$ are the (constant) advection velocity and diffusivity, respectively. The time interval $\Delta t$ between two consecutive snapshots was set to $1/60$ and the RBF used for interpolation was $\phi(r) = r^3$.

Three regularly spaced points per dimension were used to sample the parametric domain, resulting in $(U, \nu) = [0.05, 0.25]^2$ being sampled at nine locations, namely $[0.05, 0.15, 0.25] \otimes [0.05, 0.15, 0.25]$. At each parameter combination 15 random initial conditions (satisfying the appropriate Dirichlet boundary conditions) with $u \in [-1, 1]$ were generated and propagated via a high-fidelity finite difference scheme, resulting in total of 135 training runs used to build the NIROM.

At each point in the $(U, \nu)$ parametric space the accuracy is estimated as the time-averaged relative $L_2$ error averaged over 20 random initial conditions. More precisely, given a parameter combination $[U_0, v_0]$ we first generate 20 appropriate random initial conditions and project them onto the reduced space to generate their 20 reduced representations $\mathbf{a}(0)$. Each full-resolution initial state is then propagated via the high-fidelity model while each reduced initial condition is propagated via NIROM and backprojected onto the full resolution grid. For each simulation the time-averaged relative $L_2$ error is then computed, and averaged over 20 realizations. A representative example of the high-fidelity and NIROM solutions is given in Figure 4, and the relative error map in the $(U, \nu)$ space is shown in Figure 5.
4.2 Two-dimensional porous media flow

Next, the method was applied to two porous media flow problems. The governing equations for the underlying dynamics are

\[
\begin{align*}
\mathbf{q}_i &= -\frac{k_i}{\mu_i}(\nabla p_i - \rho_i \mathbf{g}) \\
\phi \frac{\partial S_i}{\partial t} + \nabla \cdot \mathbf{q}_i &= Q_i \\
\nabla \cdot \sum_i \mathbf{q}_i &= 0 \\
\sum_i S_i &= 1,
\end{align*}
\]

(35)

where \(i = 1, \ldots, M\) is the phase index, \(\mathbf{q}\) is the fluid flux, \(\mathbf{g}\) is the gravity vector, \(\rho, \mu, k\) are fluid density, viscosity, and permeability, respectively, \(Q\) is the source term and \(\phi\) is the rock porosity. In this work \(M = 2\) immiscible phases (e.g., water and oil) were modeled. It is assumed that both phases are incompressible (hence no equation of state is required) and that capillary effects are negligible leading to \(p\) being the same in all phases \((p_1 = p_2)\). Both of these assumptions make the model simpler and therefore computationally cheaper without affecting the objective of this study. The fluid permeability \(k\) is a function of phase saturation according to Corey’s model with all critical and residual saturation values set to 0.0 and all Corey exponents set to 2.0.

Each porous media flow problem is modeled on a rectangular domain with no flow domain boundaries. Additionally, the nondimensional notion of time (pore volume injected, \(P/V\)) was used, so that the volumetric flux \(Q = 1\) corresponds to the amount of fluid injected over unit time is equal to the pore volume of the computational domain. Equations (35) were solved by an in-house finite-difference implicit pressure explicit saturation porous media flow simulator with first- and second-order accuracies in time and space, respectively. The \(P/V\) time step \(\Delta t = 0.02\) is enforced to be constant between each pair of consecutive data snapshots and is employed by both high-fidelity model output and NIROM.

4.2.1 Asymmetric four-spot injection

The first case modeled was a two-dimensional system with a single injector and three producers (Figure 6). The size of the domain is \(1.0 \times 1.0\) and the sources are located at \([0.5, 0.5]\) (injector) and \([0.75, 0.5], [0.33, 0], [0.25, 0.75]\) (producers). We point out that arbitrary (in particular nondimensional) units of length can be used due to the nondimensional notion of time. The mesh resolution was set to \(50 \times 50\). Initially the system is fully saturated with the first phase (oil), hence the initial condition is common across all simulations of interest. The second phase (water) is injected at a constant volumetric rate \(Q = 1\) and the pressure boundary condition \(P = 0\) is specified at each producer.
**FIGURE 6** Schematic representation of the computational domain. The domain has dimensions $1.0 \times 1.0$, the positions of the injector and three producers are marked in blue and red, respectively.

**FIGURE 7** Example water saturation distributions at time $T = 1P/V$. Water phase saturation is visualized in the top row with the positions of injection/production points marked in red/green. Saturation cross sections along the lines connecting the points $[0, 0],[1, 1]$ (red) and $[0, 1],[1, 0]$ (blue) are plotted in the bottom row. Low viscosity ratios result in the shock front being less steep and lower (A) and gravity variations influence the overall shape of the region swept by water (B,C).

Three parameters were varied in this model, namely two gravity components $g_x, g_y \in [-1.0, 1.0]$ and the water to oil viscosity ratio $\mu \in [0.5, 1.1]$. The water to oil density ratio was set to 1.2. The ranges just specified ensure that a nontrivial range of physical behaviors are modeled, as shown in Figure 7.

The resulting 3D parametric space was sampled in a regular fashion with equal resolution 3,4,5 along each dimension. Consequently, the corresponding NIROMs were constructed based on 27, 64, and 125 training runs. The RBF used in conjunction with the PS method was $\phi(r) = r$. Each model was tested against the high-fidelity simulator on a set of 1000 random, uniformly distributed and unseen parametric combinations. The notion of error was again the relative time-averaged $L^2$ discrepancy between the NIROM and high-fidelity runs. The histograms of the relative error are given in Figure 8 and the histograms of the NIROM vs high-fidelity model speedup are given in Figure 9.

### 4.2.2 Four barriers case

The second multiphase flow problem investigated is a reconstruction of the Edward’s model studied in References 7,18. In this example the 2D square domain exhibits a degree of heterogeneity by including four barriers (Figure 10), with each barrier characterized by the isotropic permeability value between $10^{-4}$ and $10^2$ relatively to the red background. We point out that the permeability range used in this study is much wider than in Reference 7 and leads to a nontrivial range of flow patterns (Figure 11), not present if the original range is used.

As previously, parameter resolutions 3, 4, 5 (81-625 training runs) were used along each dimension in the parametric space. A random ensemble of 1000 unseen cases was used for NIROM evaluation purposes and the time-averaged $L^2$
**FIGURE 8** Relative error histograms for a batch of 1000 uniformly distributed test cases for nonintrusive reduced-order models generated with 27 (A), 64 (B), and 125 (C) training runs. The thick red bar indicates that all values of the relative error produced by the original method were 1.0 (100%) or above.

**FIGURE 9** Speedup histograms for a batch of 1000 uniformly distributed test cases for nonintrusive reduced-order models generated with 27 (A), 64 (B), and 125 (C) training runs.

**FIGURE 10** Schematic representation of the second multiphase parameterized nonintrusive reduced-order model problem. The porous domain is characterized by the homogeneous background area (red) and four homogeneous low permeability barriers (blue). Four independent parameters of the model are the permeability values within the blue barriers. Injection and production points are located in the lower left and upper right corners of the domain, respectively.

5 | DISCUSSION

NIROM captured the dynamics of the advection-diffusion validation problem with high accuracy. The relative error is typically below $10^{-3}$, except for very low diffusivities $v$. Such behavior can be expected, since in practice diffusion has a significant impact on the shape of the long-time solution profile. In particular, low $v$ allows for maintaining higher gradients in the solution profile which are more problematic to capture with a reduced POD representation. A significant

$$k_1 = [-0.5, 1, -4]; \quad k_2 = [-0.1, -4, -0.1]; \quad k_3 = [-4, -4, 0.2]; \quad k_4 = [-4, 2, 1];$$
Example water saturation distributions for barriers’ permeability values within the range investigated at time $T = 0.4P/V$. Log-permeability values $\log_{10}(k)$ within the barriers clockwise from the bottom are $-0.5$, $-0.1$, $-4.0$, $-4.0$ (A), $1.0$, $-4.0$, $-4.0$ (B), and $-4.0$, $-0.1$, $0.2$, $1.0$ (C). Observe highly nontrivial flow pattern variations posing a significant challenge for nonintrusive reduced-order model problem.

Relative error histograms for the barrier test case and models characterized by 81 (A), 256 (B), and 625 (C) training runs.

Nonintrusive reduced-order model problem simulation speedup histograms for the barrier test cases in comparison with the high-fidelity model (corresponding numbers of training runs as in Figure 12).

Source of error in the advection-diffusion problem is the random and non-smooth nature of the initial condition $u(x;t)$. The low-rank POD representation of the system is unable to capture the fine details of the initial condition, in particular high-frequency variations. However, those are dissipated relatively quickly due to nonzero diffusion and do not affect the long-time solution profile, which is captured correctly by NIROM. One of the main advantages of NIROM, namely the speedup in comparison with the high-fidelity model, is practically nonexistent in the advection-diffusion study. The reason is the fact that the forward model is very cheap to start with; in practice, NIROM would be applied to much more computationally expensive models.
Two subsequent test cases were more representative of real-life applications of NIROM, in particular since the parameter ranges were chosen so that physically significant variations in the dynamics were present in the training data. In such scenarios our method significantly outperformed that of Xiao et al both in terms of accuracy and speedup, with the latter expected due to a simpler model evaluation process. Very low accuracy of the latter method can be attributed to the miscommunication issue discussed previously and explicitly addressed in our formulation. The typical value of the NIROM approximation error was observed to be around 10% while enjoying the speedup of up to three orders of magnitude in comparison to the high-fidelity model. Naturally, as the number of training runs increases the corresponding speedup is decreased since the underlying data-driven model becomes larger in size. It should be pointed out that the latter two test problems were characterized by fixed initial conditions across all training and test simulations. The effect of varying the initial condition can be included with the model in the same fashion as for the advection-diffusion problem, but generally requires a larger number of training runs to be provided and is not a crucial aspect of this study.

The normalization procedure is another reason for increased stability and accuracy of our approach. Despite the parameters \( p \) being of the same size and having the same span, their rescaling had to be performed to match the scaling of the system state data \( \alpha \) and construct a set of more suitable samples for the RBF method. Data normalization still had a non-trivial effect, especially in the two porous media flow examples, even when all parameters in our examples spanned the same orders of magnitude (near-equal \( \Delta_p^{(0)} \) all of order \( O(1) \)). This is because the system state normalization factor \( \Delta_{\alpha} \) was of order \( O(10) \), meaning that without the normalization the sample structure would be too squeezed along the parameter dimensions resulting in a sub-optimal and highly non-uniform sample structure.

It can also be observed that the average approximation error does not always behave intuitively. More precisely, in Figure 8 the error histogram gets skewed toward right (larger error) as the amount of training data increases. This is in contrast with the intuitive expectation where a larger amount of data points used for interpolation should increase the overall accuracy and consequently decrease the average error.

Finally, our choice of the reduction method (POD) is not mandatory. Alternative reduction processes such as dynamic mode decomposition\(^6\) are potentially viable but it should be noted that they will necessarily be less efficient at capturing the dynamics. In practical terms, a higher-dimensional reduced representation will be required for the same level of accuracy.

6 | CONCLUSIONS

In this work a new and simple method of constructing parameterized NIROMs via radial basis interpolation methods was presented. A previous method by Xiao et al was first discussed and the miscommunication problem and its consequences were pointed out. Subsequently, two improvements were proposed, namely the rescaling procedure and a single-layer interpolation approach.

The resulting method can handle an arbitrary number of parameters spanning a range of orders of magnitude. Additionally, the sensitivity of the model to state variables or parameters can be controlled easily by rescaling the normalization factors \( \Delta_{\alpha}, \Delta_p^{(0)} \). Any radial basis method can be used for the interpolation process, including the RBF, PS, or Kriging.

The approach was tested on three parameterized problems, the one-dimensional advection-diffusion equation and two porous media flow examples with different kinds of model parameterizations. NIROM provided speed ups up to three orders of magnitude in comparison to the high-fidelity model at the cost of relative errors of order 10%. The NIROM implementation used in this work used MATLAB programming language for model construction and propagation. Due to its interpreted nature even higher speed ups are expected if a compiled language (eg, Fortran, C) is to be used. Our method outperformed the previous approach both in terms of speedup and accuracy.

It was observed that in some cases the approximation error increased with the amount of training data provided. It was concluded that the reason behind such behavior is the nonsmooth nature of the field \( f \) propagating the system state forward in time. A more detailed analysis of this phenomenon, a method for predicting it just from training data and a detailed analysis of suitability of NIROM in the context of reservoir engineering will be addressed in our future work.

ORCID

Wawrzyniec J. Kostorz \( \text{https://orcid.org/0000-0003-4690-2475} \)
REFERENCES
1. Xiao D, Fang F, Pain C, Hu G. Non-intrusive reduced-order modelling of the Navier-Stokes equations based on RBF interpolation. *Int J Numer Methods Fluids*. 2015;79(11):580-595.
2. Yu J, Yan C, Guo M. Non-intrusive reduced-order modeling for fluid problems: a brief review. *Proc Inst Mech Eng Pt G J Aerospace Eng*. 2019;233(16):5896-5912.
3. Chen W, Hesthaven J, Junqiang B, Yang Z, Tihao Y. Greedy non-intrusive reduced order model for fluid dynamics. *AIAA Journal*. 2018;56(12):4927-4943.
4. Xiao D, Yang P, Fang F, et al. A non-intrusive reduced-order model for compressible fluid and fractured solid coupling and its application to blasting. *J Comput Phys*. 2017;330:221-244.
5. Xiao D, Yang P, Fang F, Xiang J, Pain C, Navon I. Non-intrusive reduced order modelling of fluid structure interactions. *Comput Methods Appl Mech Eng*. 2016;303:35-54.
6. Bistrian D, Navon I. Randomized dynamic mode decomposition for nonintrusive reduced order modelling. *Int J Numer Methods Eng*. 2017;112(1):3-25.
7. Xiao D, Lin Z, Fang F, et al. Non-intrusive reduced-order modeling for multiphase porous media flows using Smolyak sparse grids. *Int J Numer Methods Fluids*. 2017;83(2):205-219.
8. Klie H. Unlocking fast reservoir predictions via nonintrusive reduced-order models. Paper presented at: 2013 SPE Reservoir Simulation Symposium. Society of Petroleum Engineers; The Woodlands, Texas, USA; 2013.
9. Xiao C, Leeuwenburgh O, Lin H, Heemink A. Non-intrusive subdomain POD-TPWL for reservoir history matching. *Comput Geosci*. 2019;23(3):537-565.
10. Jin Z, Durlofsky LJ. Reduced-order modeling of co2 storage operations. *Int J Greenhouse Gas Control*. 2018;68:49-67.
11. Wang Q, Hesthaven J, Ray D. Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem. *J Comput Phys*. 2019;384:289-307.
12. Hesthaven J, Ubbiali S. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *J Comput Phys*. 2018;363:55-78.
13. Pawar S, Rahman M, Vaddireddy H, San O, Rasheed A, Vedula P. A deep learning enabler for nonintrusive reduced order modeling of fluid flows. *Phys Fluids*. 2019;31:085101.
14. Xiao D. Error estimation of the parametric non-intrusive reduced order model using machine learning. *Comput Methods Appl Mech Eng*. 2019;355:513-534.
15. Jones D, Schonlau M, Welch W. Efficient global optimization of expensive black-box functions. *J Global Optimiz*. 1998;13(4):455-492.
16. Awrejcewicz J, Krys’ko VA, Vakakis AF. *Order Reduction by Proper Orthogonal Decomposition (POD) Analysis*. Berlin, Germany: Springer; 2004, pp. 177–238.
17. Christie M.A., Bond D.J. Detailed Simulation of Unstable Processes in Miscible Flooding. *SPE Reserv Eng*. 1987;2(04):514-522.
18. Salinas P, Pavlidis D, Xie Z, Osman H, Pain CC, Jackson MD. A discontinuous control volume finite element method for multi-phase flow in heterogeneous porous media. *J Comput Phys*. 2018;352:602-614.

SUPPORTING INFORMATION
Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Kostorz WJ, Muggeridge AH, Jackson MD. An efficient and robust method for parameterized nonintrusive reduced-order modeling. *Int J Numer Methods Eng*. 2020;121:4674–4688. https://doi.org/10.1002/nme.6461