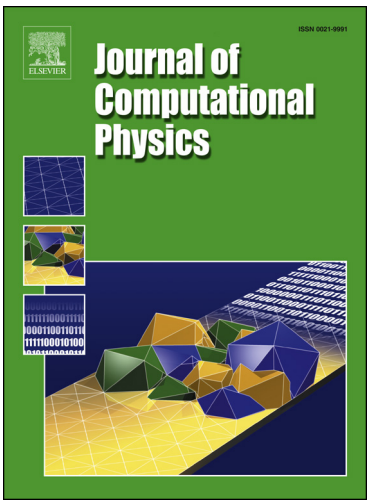
Accepted Manuscript



A resilient and efficient CFD framework: Statistical learning tools for multi-fidelity and heterogeneous information fusion

Seungjoon Lee, Ioannis G. Kevrekidis, George Em Karniadakis

PII: S0021-9991(17)30396-0
DOI: http://dx.doi.org/10.1016/j.jcp.2017.05.021
Reference: YJCPH 7368

To appear in: *Journal of Computational Physics*

Received date: 25 November 2016
Revised date: 7 May 2017
Accepted date: 10 May 2017

Please cite this article in press as: S. Lee et al., A resilient and efficient CFD framework: Statistical learning tools for multi-fidelity and heterogeneous information fusion, *J. Comput. Phys.* (2017), http://dx.doi.org/10.1016/j.jcp.2017.05.021

A resilient and efficient CFD framework: statistical learning tools for multi-fidelity and heterogeneous information fusion

Seungjoon Lee[a], Ioannis G. Kevrekidis[b], George Em Karniadakis[a,*]

[a]*Division of Applied Mathematics, Brown University, Providence, RI 02912*
[b]*Department of Chemical and Biological Engineering; also PACM, Princeton University, Princeton, NJ 08544*

Abstract

Exascale-level simulations require fault-resilient algorithms that are robust against repeated and expected software and/or hardware failures during computations, which may render the simulation results unsatisfactory. If each processor can share some global information about the simulation from a coarse, limited accuracy but relatively costless auxiliary simulator we can effectively fill-in the missing spatial data at the required times by a statistical learning technique – multi-level Gaussian process regression, on the fly; this has been demonstrated in previous work [1]. Based on the previous work, we also employ another (nonlinear) statistical learning technique, *Diffusion Maps*, that detect computational redundancy in time and hence accelerate the simulation by *projective* time integration, giving the overall computation a "patch dynamics" flavor. Furthermore, we are now able to perform information fusion with multi-fidelity and heterogeneous data (including stochastic data). Finally, we set the foundations of a new framework in CFD, called *patch simulation*, that combines information fusion techniques from, in principle, multiple fidelity and resolution simulations (and even experiments) with a new adaptive timestep refinement technique. We present two benchmark problems (the heat equation and the Navier-Stokes equations) to demonstrate the new capability that statistical learning tools can bring to traditional scientific computing algorithms. For each problem, we rely on heterogeneous and multi-fidelity data, either from a coarse simulation of the same equation or from a stochastic, particle-based, more "microscopic" simulation. We consider, as such "auxiliary" models, a Monte Carlo random walk for the heat equation and a dissipative particle dynamics (DPD) model for the Navier-Stokes equations. More broadly, in this paper we demonstrate the symbiotic and synergistic combination of statistical learning, domain decomposition, and scientific computing in exascale simulations.

Keywords: gappy data, patch dynamics, machine learning, domain decomposition, exascale simulation, Gaussian process regression, diffusion maps

1. Introduction

As demands for solving complex physics problems are growing in computational fluid dynamics, exascale simulations will be performed in the near future, e.g. [2, 3]. However, exascale

*Corresponding author
Email address: george_karniadakis@brown.edu (George Em Karniadakis)

simulations, which use massive numbers of processors, require not only an efficient algorithm but also a fault-resilient algorithm to address traditional open issues [4]. The key issue is "fault resilience" against repeated and expected (but random) software or hardware failures during computations, which may render the simulation results unsatisfactory or simply unusable. Many research approaches have been developed to recover the missing data on both sides, computer systems [5, 6, 7, 8] and mathematical algorithms [9, 10, 11, 12]. The other important issue is "computational efficiency" against computational redundancy from inadequate spatio-temporal discretization or dynamics of a problem itself. Over the last few decades, several numerical methods are introduced to overcome the computational redundancy [13, 14].

This paper is motivated by the approach introduced by Lee *et al.* [1], which demonstrably achieved fault-resilience when solving PDEs and processor failures results in spatial gaps in the simulation output. This was achieved by an information fusion method with multi-resolution auxiliary data from a repeatedly reinitialized short-duration coarse resolution auxiliary simulation of the problem. Based on this algorithm, we now introduce a new framework, namely "patch simulation", for accelerating simulations via a statistical learning technique, *Diffusion Maps* (DMaps), that helps detect computational redundancy in time and hence accelerating the simulation by *projective time integration*. The projective time integration was introduced by Gear and Kevrekidis [15] and used in Sirisup *et al.* [16] in a CFD context, demonstrating computational acceleration by an equation-free POD-assisted approach. However, choosing appropriate projection steps is still an open issue. In this paper, learning from the auxiliary data via diffusion maps provides a dynamics-informed temporal discretization, which leads to an acceleration of the computation in time.

The original concept of the patch simulation comes from "patch dynamics" [15, 17], which predicts large scale spatio-temporal dynamics (macro level) from a series of short spatio-temporal dynamics (micro level). In order to extend this concept to a large scale simulation of a CFD problem, we employ two statistical learning techniques to provide additional and important spatio-temporal information from the auxiliary data, produced by a coarse, auxiliary, repeatedly restarted, relatively costless simulations (even experiments). The first method is a multi-level Gaussian process regression, which supports to "fill-in" the missing data in space [18, 19, 20, 21]. The other method is diffusion maps, which provides lower-dimensional information to estimate the appropriate projection time [22, 23, 24, 25]. In this paper, we show that these two statistical learning techniques can use the auxiliary data to improve the accuracy and the efficiency simultaneously.

Furthermore, in order to investigate multiscale phenomena in complex problems, many research efforts have sought to couple numerical simulations and even experiments, with multi-fidelity and heterogeneous models via information fusion techniques [26, 27, 28]. In this paper, the auxiliary data is generalized to stochastic descriptions such as a random walk model for the heat equation and a dissipative particle dynamics (DPD) model for the Navier-Stokes equations. This results in a guarantee of the generality of the auxiliary data, which can come, in principle, from any scale, any fidelity and any heterogeneous model. The heat equation (linear) and the lid-driven cavity flow (nonlinear) are simple problems but they have been used as general benchmark problems for demonstrating effectiveness of new algorithms. These problems serve also as useful benchmarks in stochastic modeling and uncertainty quantification [29].

The paper is organized as follows: In section 2, we introduce a general patch simulation framework with a flow chart. In section 3, we introduce the two statistical learning techniques used here, the multi-level Gaussian process regression and Diffusion Maps (DMaps). In section 4, we introduce heterogeneous and multi-fidelity models as the source of auxiliary data for each

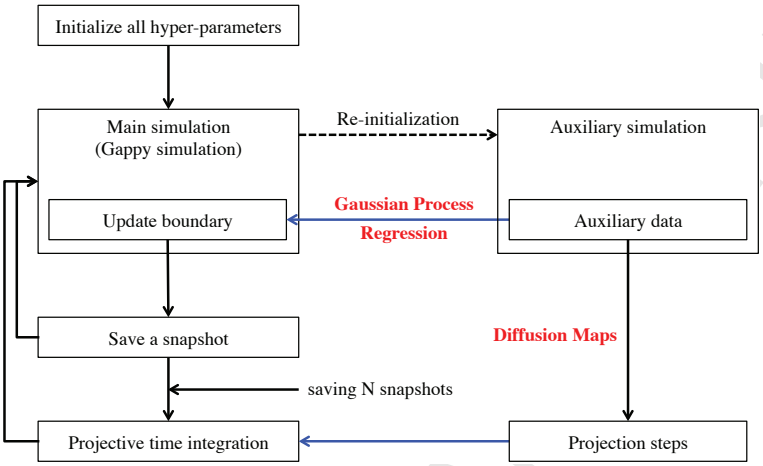
benchmark problem. In section 5, we present the computational domains and simulation set up for the benchmark problems. In section 6, we present results of parametric studies and analyze them in terms of the overall accuracy. In section 7, we summarize our results and discuss open issues for further development of a robust and efficient CFD framework.

2. The Patch Simulation Framework

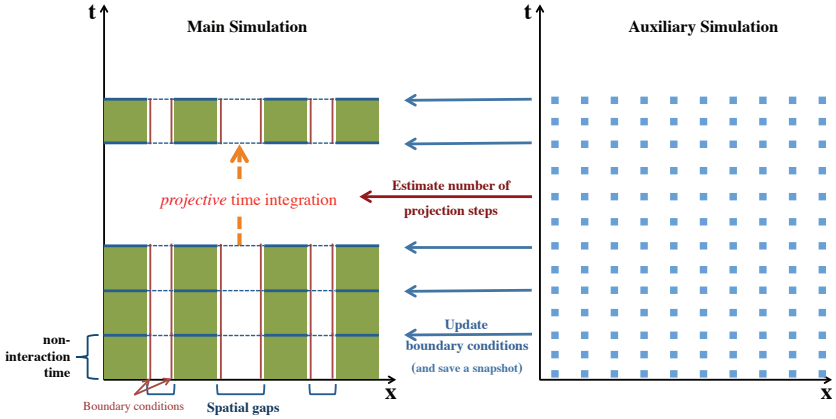
In the new CFD framework we propose, we split up the simulation into a main simulation and an auxiliary simulation. The main simulation computes a solution to a PDE on some (non-overlapping) fine-resolution sub-domains with spatial gaps. The auxiliary simulation, on the other hand, computes a solution on the *entire* domain but with less accuracy. This auxiliary data can come from any fidelity, scale, or model. In previous work (see reference [1]), data from the auxiliary simulation provide information about the global continuity of the solution fields to the gappy, main simulation via multi-level Gaussian process regression, namely coKriging. Then We estimate field variables at each local boundary using both global (with low accuracy) and local (with high accuracy) information. This leads to a fault-resilience with respect to spatial gaps.

In [16], a data-driven "equation-free/Galerkin-free POD-assisted computation" was presented that exploited brief bursts of the full Navier-Stokes simulation over the entire computational domain and *the low-dimensionality of the behavior* to accelerate the overall simulation. In this paper we show how to extend this approach so as to avoid doing the detailed simulation over the entire spatial domain. The main new enabling ingredient is the availability of a "cheap and coarse" auxiliary simulation, which is reliable only over short time integration due to error accumulation in time, and which is therefore repeatedly (and carefully) reinitialized. Keeping the same assumption (low-dimensionality of the long-term dynamics) this new ingredient will, as we will show in section 6, enable significant computational savings over the spatial extent of the full detailed simulation. While this ingredient allows the full simulation to be performed systematically in only parts of the domain, the same idea can be also used in enabling the "filling in" of computational information loss through hardware/software failure locally in space-time. We repeat that the main assumption is that the long-term dynamics lie on a slow manifold, which is parameterized by a few POD basis functions [30]. Then, we perform simulations on this manifold through short time computations via "projective time integration", and this results in accelerating the main simulation. However, it is nontrivial to systematically and accurately estimate the length of the appropriate projection steps due to rapid changes of dynamics in a transient period. The auxiliary data help estimate this *projection time*, or "a jump size", for the main simulation via a statistical learning technique, diffusion maps. Finally, this framework bears some similarities with "patch dynamics" for a micro-macro simulator [15, 17]. The projection time represents a macro time scale for dynamics on the slow manifold while detailed computation on the fine-resolution sub-domains represents the micro (fine)-level of the description of the dynamics.

A flow chart of the patch simulation is shown in Figure 1. A cycle of the main simulation with fine-resolution sub-domains is described in Figure 1 (a) (see reference [1] for details). We first check spatial gaps due to computational faults and choose a buffer size for each fine-resolution sub-domain. Next, we estimate the local boundary condition of each sub-domain with auxiliary data and obtain the solution (in embarrassingly parallel) during a macro timestep, called the non-interaction timestep. After each macro timestep, we save a snapshot of the field variables on fine-resolution sub-domains. We repeat this cycle until we have "N" snapshots. After that, we employ the projective time integration scheme on each fine-resolution sub-domain independently with an appropriate projection time. This constitutes one complete cycle of the patch simulation



(a) A flow chart of the patch simulation.



(b) A schematic illustration of information fusion between the two simulations.

Figure 1: In (a), we first initialize all hyper-parameters of the main simulation. Next, we advance the main simulation (the gappy simulation) and save a snapshot of field variables. After "N" snapshots are saved, we estimate time derivatives and use them to approximate long term variables by projective time integration. The auxiliary simulation provides two types of information to the main simulation (colored blue): global but inaccurate estimates of the field variables via Gaussian process regression and projection time for the projective time integration via diffusion maps. In (b), the auxiliary simulation helps to update local boundary conditions every non-interaction time (colored by blue) and to estimate projection time (colored by red).

algorithm. The patch simulation repeats this cycle until the main simulation ends or all faults are fixed. The number of saved snapshots, N, is a hyper-parameter of the patch simulation and will be discussed in section 6.3.

As shown in Figure 1 (a), the auxiliary data provides two different types of information (colored by blue) through two different statistical learning techniques (colored by red) to the main simulation, which has missing data at local boundaries of each fine-resolution sub-domain. As shown in Figure 1 (b), first is the global and spatial information with low accuracy (blue arrows), via the multi-level Gaussian process regression, employed to estimate local boundary conditions of each fine-resolution sub-domains. Second is the temporal information of a projection size (a red arrow), via diffusion maps, for the projective time integration. Details of these statistical learning techniques are described below.

3. Statistical Learning Algorithms

In order to connect the auxiliary simulation with the main simulation, we take advantage of two statistical learning algorithms, namely multi-level Gaussian process regression and diffusion maps. Multi-level Gaussian process regression has to do with fine-resolution sub-domains, and can be done independent of projection. The other, Diffusion Maps, have to do with projection, and can be done independent of gaps in space. Here we will put them together but we first describe each "without the other".

3.1. Gaussian Process Regression

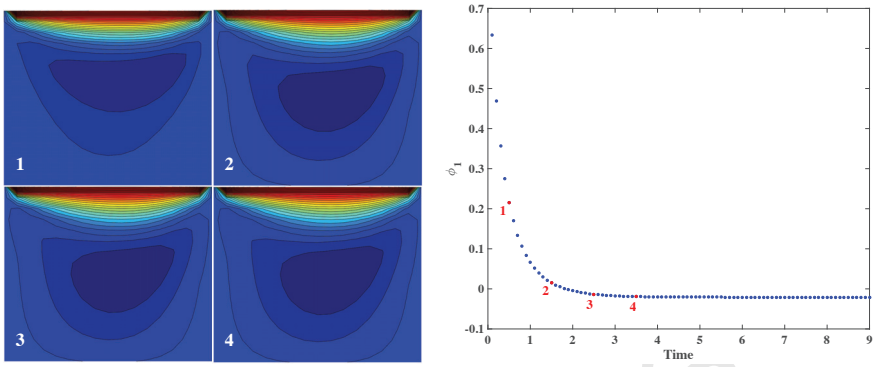
The auxiliary simulation, which is relatively costless, provides a data set to "fill in" the missing data spatially, such as the boundary conditions of each sub-domain. In this framework there are 2-levels of data sets: (1) data from the main simulation, the local information with high accuracy and (2) data from the auxiliary simulation, the global information with low accuracy. Two data sets, from the main simulation and the auxiliary simulation, are modeled by two different Gaussian Processes, y and y_a, respectively. There is a scaling parameter (ρ), which represents a correlation between two Gaussian processes. Then, the auto-regressive scheme of Kennedy and O'Hagan [31] is defined by:

$$y(x) = \rho(x) y_a(x) + \delta(x), \tag{1}$$

where $\delta(x)$ is a bias Gaussian field, which is independent of y and y_a. $\hat{y}(x^*)$ and $\hat{y}_a(x^*)$ are the mean value of predicted field variables, i.e., temperature or velocities, at the local boundary x^* of each sub-domain. The mean ($\hat{y}(x^*)$) and variance ($\hat{s}^2(x^*)$) of predicted field variables are obtained by the following equation with $\{\mu, \sigma^2\}$ and $\{\mu_a, \sigma_a^2\}$, which represent means and variances of Gaussian processes of y and y_a, respectively:

$$\hat{y}(x^*) = \mu + \rho \hat{y}_a(x^*) + r^T \left(R + \sigma_\epsilon^2 I\right)^{-1} \left[y(x) - \mathbf{1}\mu - \rho \hat{y}_a(x)\right], \tag{2}$$

$$\hat{s}^2(x^*) = \rho^2 \hat{s}_a^2(x^*) + \sigma^2 \left[1 - r^T \left(R + \sigma_\epsilon^2 I\right)^{-1} r + \frac{\left[1 - r^T \left(R + \sigma_\epsilon^2 I\right)^{-1} r\right]^2}{\mathbf{1}^T \left(R + \sigma_\epsilon^2 I\right)^{-1} \mathbf{1}}\right], \tag{3}$$



(a) Snapshots of the streamwise velocity.

(b) The component of the data on the leading Diffusion Map eigenfunction.

Figure 2: A schematic illustration of our use of diffusion maps: the left contours show a series of snapshots of the streamwise velocity from a 16×16 grid of auxiliary data in the Navier-Stokes equations. The right plot represents the component of these snapshots in the first nontrivial diffusion map eigenvector obtained from the data ensemble. The first diffusion map coordinate of the sample snapshots in (a) are colored by red in (b).

where $R = \kappa(x, x', \theta)$ and $r = \kappa(x, x^*, \theta)$ represent a correlation matrix and a correlation vector, respectively, and σ_ϵ^2 is the variance of the normally distributed random field. The correlation kernel, $\kappa(x, x', \theta)$, is constructed by the following Matérn kernel [32]:

$$\kappa(x, x', \theta) = \left(1 + \sqrt{3}\theta \mathrm{d}(x, x')\right) \exp\left(-\sqrt{3}\theta \mathrm{d}(x, x')\right), \qquad x, x' \in y, \tag{4}$$

where $\mathrm{d}(\cdot, \cdot)$ represents the Euclidean distance between two data.

An optimal hyperparameter set, $\{\mu, \sigma, \sigma_\epsilon, \rho, \theta\}$, are obtained by maximum likelihood estimation (MLE) from the aforementioned data sets. For details of the formulation and hyperparameters, see reference [21].

3.2. *Diffusion Maps*

Diffusion maps, based on a discrete Laplace-Beltrami operator, provides a parametrization of the low-dimensional nonlinear manifold as well as a (diffusion) distance on it [23, 24, 25]. As shown in Figure 2, diffusion maps obtained by temporal snapshots of field variables provide a "dynamics-related" temporal discretization on the diffusion coordinate. This observation can be used as a criterion of the computational redundancy of the original uniform discretization in time. Based on the diffusion distance, we can assign an appropriate projection time for the projective time integration. The diffusion distance, $D(\cdot, \cdot)$, between adjacent snapshots are calculated on a one-dimensional slow manifold.

$$D^2(t_{i+1}, t_i) = \lambda_1^2 \left[\phi_1(t_{i+1}) - \phi_1(t_i)\right]^2, \tag{5}$$

where (λ_1, ϕ_1) represents the first nontrivial eigenvalue and eigenvector of the discrete Laplace-Beltrami operator via a diffusion kernel, W.

$$W_{i,j} = \exp\left(-\frac{\|y_i - y_j\|^2}{\epsilon^2}\right), \tag{6}$$

where y represents each data vector (temporal snapshot) and ϵ represents a characteristic distance between data vectors. In this framework, we choose ϵ to be the median distance between all data from the coarse but full simulation.

A formulation for finding an appropriate projection time starts from a fundamental observation of the diffusion distance between temporal snapshots – the diffusion distance between adjacent snapshots in time corresponds to *the data variability of the slow dynamics* during that time interval. A large diffusion distance, for example, represents a large change in the dynamics, i.e., we have to employ the projective time integration with small timesteps, and vice versa. Hence, the projection time at time t_i is generalized to an "adaptive timestep refinement" by the following equation:

$$J(t_i) = \alpha \log \frac{D_m}{D(t_{i+1}, t_i)}, \tag{7}$$

where α is a tuned parameter and D_m represents the maximum diffusion distance along the time series. Since the projection time should be an integer, $J(t_i)$ is rounded to the nearest integer. Then, we project and estimate field variables at time $t^* = t + J(t) \cdot \tau \cdot \Delta t$ by a proper orthogonal decomposition (POD) [33] assisted projection [16], where τ is a non-interaction timestep number, which decides for how many timesteps ($\tau \cdot \Delta t$) to solve each fine-resolution subdomain independently (without communications between each other), see reference [1]. Finally, we can save computation during $J(t) \cdot \tau \cdot \Delta t$ and this leads to efficiency gain.

In this formulation, there is no projection when the diffusion distance is the maximum, while there is the biggest projection time when the diffusion distance is the minimum. There is a trade-off between the computational efficiency and accuracy via the tuned parameter α. The tuned parameter α is obtained by prior knowledge about flow physics (proportional to α). In this framework, α is set to 1 as a default, which assumes there is no prior knowledge about flow physics.

4. Multi-fidelity and Heterogeneous Auxiliary Data

In order to demonstrate the capability of the information fusion with multi-fidelity and heterogenous auxiliary data, we introduce two stochastic and particle-based auxiliary simulations, namely a two-dimensional random walk model for the heat equation and a dissipative particle dynamics (DPD) model for the Navier-Stokes equations in this section. In order to understand and characterize the capability for multi-fidelity and heterogeneous simulations, we used accurate and expensive models (but small number of particles) as the auxiliary simulation for analysis purpose. In the future, we will use "cheap and dirty but useful" auxiliary simulations. Details for simulation and problem setup are presented below.

4.1. Two-dimensional random walk model for the heat equation

The two-dimensional heat equation in a uniform rectangular grid is solved by a simple random walk model via the Monte Carlo method. Specifically, by Fick's law, a particle with temperature $T(x, \tau)$ at a point x can move north, south, east, or west with equal probability, $p(N) = p(S) = p(E) = p(W) = 0.25$, after a unit time $t = \tau + \Delta t$. Hence, after $t = \tau + n\Delta t$, all particles originally located at the point x are distributed by the random walk model. Using those distributed data, we calculate the updated temperature $T(x, t + n\Delta t)$ at every grid point. The equation for calculating a heat quantity in the arbitrary domain D after time $n\Delta t$ is as follows:

$$T(x^*, \tau + n\Delta t) = \oint T(x, \tau) d\mu(x) \quad \forall x \in D, \tag{8}$$

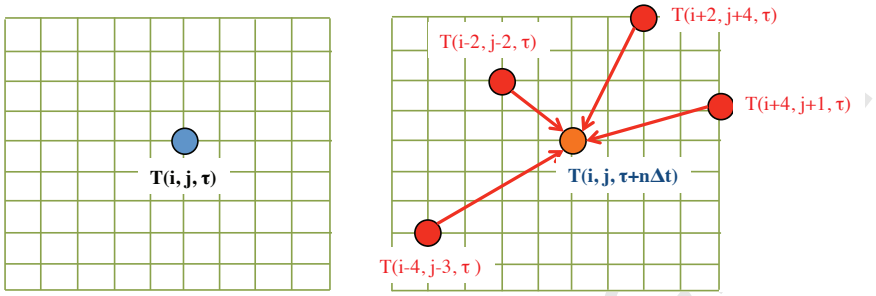


Figure 3: A schematic illustration of a random walk model for the two-dimensional heat equation: the temperature T at the node (i, j) at the next time $t + n\Delta t$ (colored orange) is obtained by averaging the field variables over all sample paths that visited (i, j) at time $t + n\Delta t$ (colored red) by the Monte Carlo method.

where $\mu(x)$ is a probability measure representing the probability that a particle moves from x to x^* after time $n\Delta t$ (n steps).

As shown in Figure 3, the continuous model is rewritten as a two-dimensional 10×10 grid discrete model with N sample paths (particles) of the Monte Carlo method as:

$$T(x^*, \tau + n\Delta t) = \sum_{x \in D} T(x, \tau) P(x), \tag{9}$$

where x is the position of each node (i, j) and $P(x)$ is a discrete probability measure calculated by the Monte Carlo method:

$$P(x) = \frac{n_p}{N_p}, \tag{10}$$

where N_p is the number of total sample paths originating from x, and n_p is the number of those paths that land at x^* at time $t + n\Delta t$.

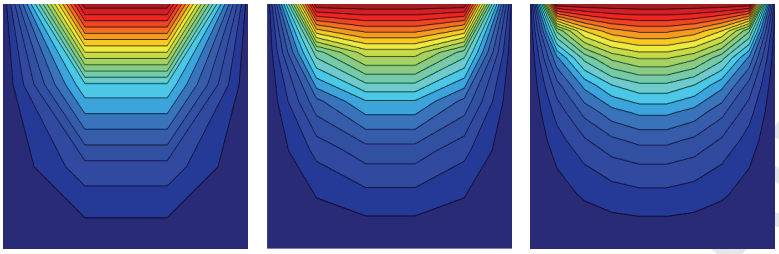
In this work, we choose 20, 200 and 2000 sample paths at each grid point (totaling 2000, 20000 and 200000 computations, respectively) for calculating the updated temperature at each point through equations (9) and (10). By the equivalent numerical discretization corresponding to the random walk model, the diffusivity $\kappa = h^2/4\Delta t$ is determined by temporal and spatial discretization (Δt and h) of the random walk model. Temperature contours of different auxiliary data are shown in Figure 4.

4.2. Dissipative Particle Dynamics (DPD) for the Navier-Stokes equations

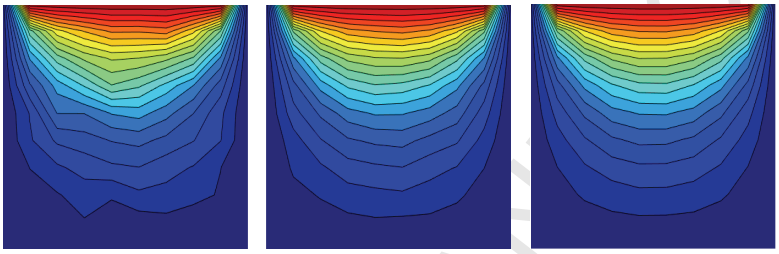
Next, we introduce a dissipative particle dynamics (DPD) model, a particle-based mesoscale simulation, as auxiliary data-producing simulator for the Navier-Stokes equations. The DPD method uses virtual particles, which represent "molecular clusters" moving together in Lagrangian fashion. In DPD systems, we assume that particles, whose mass m_i, position $\mathbf{r_i}$, and velocity $\mathbf{v_i}$, interact with each other by pairwise-additive forces, which consist of three terms: (1) conservative force ($\mathbf{F}_{ij}^C$), (2) dissipative force ($\mathbf{F}_{ij}^D$), and (3) random force ($\mathbf{F}_{ij}^R$):

$$\mathbf{F}_{ij}^C = F_{ij}^C(r_{ij})\mathbf{r}_{ij}, \tag{11}$$

$$\mathbf{F}_{ij}^D = -\gamma\omega^D(r_{ij})(\mathbf{v}_{ij} \cdot \mathbf{r}_{ij})\mathbf{r}_{ij}, \tag{12}$$



(a) 4×4 grid. (b) 6×6 grid. (c) 10×10 grid.



(d) 20 sample paths. (e) 200 sample paths. (f) 2000 sample paths.

Figure 4: Temperature contours of different auxiliary data for the heat equation at time $t = 15$ (a transient period). (a)-(c): the finite difference method. (d)-(f): the random walk model by the Monte Carlo method.

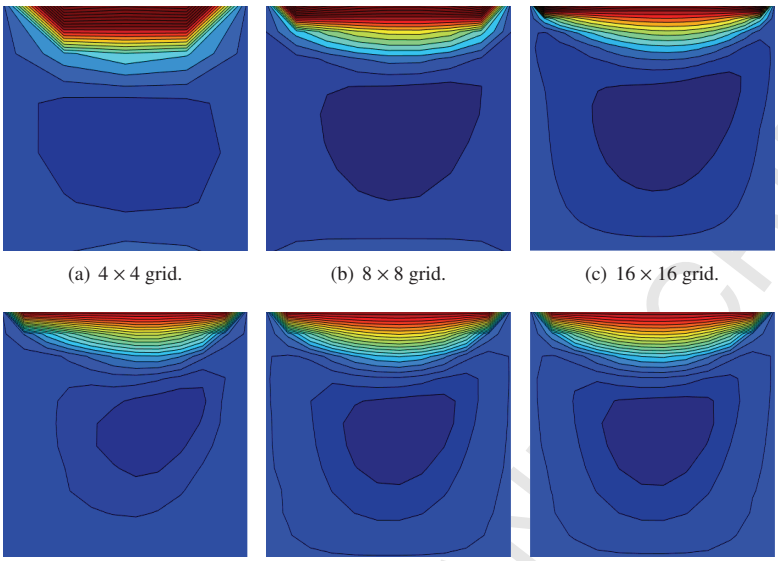
$$\mathbf{F}_{ij}^{R} = \sigma\omega^{R}(r_{ij})\xi_{ij}\mathbf{r}_{ij}, \tag{13}$$

where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$, $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$, r_{ij} is a distance between i and j particles; γ and σ are hyper-parameters for dissipative and random force, respectively; ω^D and ω^R are distance dependent weight functions, and ξ_{ij} is a normally distributed random variable. The conservative force $F_{ij}^{C}(r_{ij})$ with a cutoff radius r_c is obtained by

$$F_{ij}^{C}(r_{ij}) = \begin{cases} a_{ij}(1 - r_{ij}/r_c) & \text{if } r_{ij} \le r_c; \\ 0 & \text{if } r_{ij} > r_c. \end{cases} \tag{14}$$

where $a_{ij} = \sqrt{a_i a_j}$ and a_i, a_j represent conservative force coefficient for particle i and j, respectively. For details of DPD systems, see reference [34].

In this paper, the kinematic viscosity, ν, of the DPD fluid is equal to 2.86 (in DPD units), corresponding to Reynolds number Re = 35. This value was obtained by fitting a double parabola to the DPD results using the reverse Poiseuille flow method described in reference [35]. For boundary condition, we employ boundary particles with particle reflection rules, for details, see reference [34]. The field variables of the DPD results are obtained by a spatiotemporal averaging. In space, the DPD results are averaged on 10×10 rectangular bins, total 100 bins. In particular, we take average of all field variable in each bin and set this mean value as a representative field variable of each bin. In time, the DPD results are averaged over every 1000 time steps with $t = 0.1$ with a same process in space. The accuracy of averaged values are dependent on the number of particles, which is controlled by the length in the z-direction. Due to our definition of



(a) 4×4 grid. (b) 8×8 grid. (c) 16×16 grid.

(d) 1000 DPD particles. (e) 4000 DPD particles. (f) 8000 DPD particles.

Figure 5: Streamwise velocity contours of different auxiliary data for the Navier-Stokes equations at time $t = 5$ (a transient period). (a)-(c): the finite difference method. (d)-(f): the DPD model.

the auxiliary data as relatively costless, we employ small numbers of particles like 1000, 4000, and 8000 DPD particles to obtain field variables. Streamwise velocity contours of different auxiliary data are shown in Figure 5.

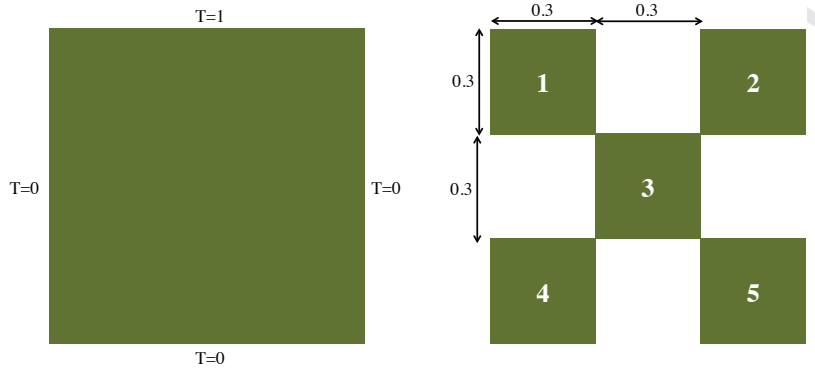
5. Simulation setup

5.1. Heat equation

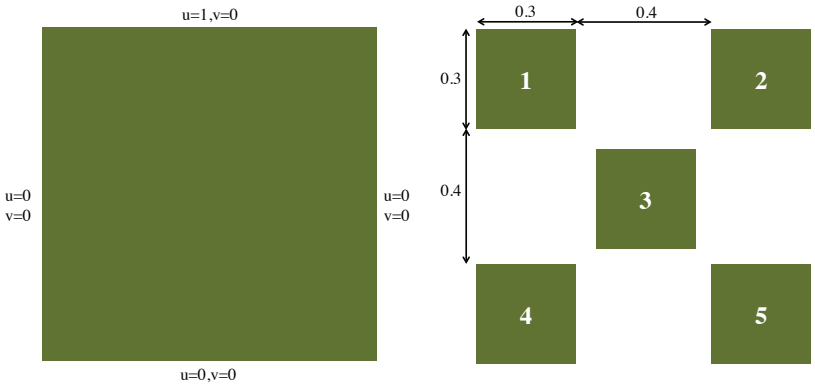
First, we solve the two-dimensional heat equation with diffusivity κ,

$$\frac{\partial T}{\partial t} = \kappa \nabla^2 T. \tag{15}$$

In order to perform a patch simulation and a reference simulation on the complete domain, we employ the second-order finite difference method. We employ Dirichlet boundary condition for physical boundary conditions, which are described in Figure 6(a). The grid resolution of the patch simulation is 11×11 per each sub-domain (total of 5 sub-domains), with a total of 605 grid points in the patch simulation, while the grid resolution of the reference simulation is 31×31. In order to investigate the influence of multi-fidelity auxiliary data on the overall accuracy, we employ not only a coarse grid with resolution 4×4, 6×6, and 10×10, but also the random walk model with different numbers of sample paths as 20, 200, and 2000 on a 10×10 grid. We use an ensemble average of the data by 30 trials of the random walk model. Boundary conditions for each fine-resolution sub-domain are of the Dirichlet type estimated by multi-level Gaussian process regression with auxiliary data. The temporal discretization corresponds to time step



(a) Heat equation.



(b) Navier-Stokes Equations.

Figure 6: A schematic illustration of gappy domains for two benchmark problems [1]: (left) the global (physical) boundary condition of the reference simulation. (right) the location and index of the fine-resolution sub-domains colored by green.

$\Delta t = 0.01$ with heat diffusivity $\kappa = 1/60$ given by the random walk model. The non-interaction timestep number τ is 15. The discretized equations are integrated from $t = 0$ to $t = 30$ (the steady-state is established around $t = 25$).

5.2. Navier-Stokes equations

Next, we consider incompressible flow for the lid-driven cavity described by the divergence-free Navier-Stokes equations:

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} + f, \tag{16}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{17}$$

where $\mathbf{v}$ is the velocity vector, p is the pressure, and ν is the kinematic viscosity of the fluid. We employ a second-order finite difference method. The physical boundary conditions are of Dirichlet type, as shown in Figure 6(b). The resolution of a rectangular cell for each sub-domain is 8×8, with a total of 320 cells for the patch simulation. In order to compare errors from different auxiliary data quantitatively, we employ not only coarse simulations with resolution 4×4, 8×8, and 16×16 for the entire domain, but also DPD simulations with different numbers of DPD particles as 1000, 4000, and 8000. For the DPD result, we use an ensemble average data of 30 trials. We employ Dirichlet boundary conditions for each fine-resolution sub-domain estimated by multi-level Gaussian process regression with auxiliary data. The time step is $\Delta t = 0.005$ with Reynolds number $\mathrm{Re} = 35$ given by the DPD model. The non-interaction timestep number τ is 20. The discretized equations are integrated from $t = 0$ to $t = 9$ (the steady-state is established around $t = 5$).

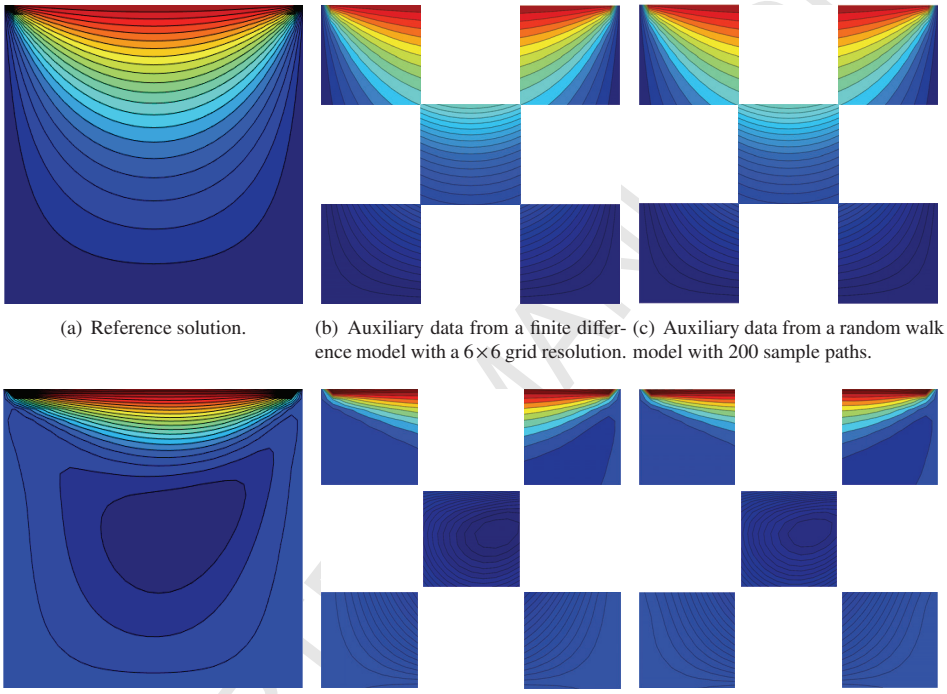
6. Results

The contour plots of field variables compared to reference simulations (no spatial gaps or projective time integration) qualitatively are shown in Figure 7. Also, the results of the patch simulation with different auxiliary data and projection time, compared to the results of the reference gappy simulation (no projective time integration), are shown in Figures 8 - 11. The green line represents the result of the gappy simulation, which has no projective time integration as in previous work [1]. The blue dots represent the result of the patch simulation, which employ projective time integration with different projection time. The quantitative comparisons among different auxiliary data are shown in Figure 12.

In order to compare results of patch simulations with different auxiliary data, we employ a measure of RMS error, namely the "total" RMS error ($\mathrm{RMSE_T}$) for temporal accuracy. The "total" RMS error at time t is calculated by the following formula:

$$\mathrm{RMSE_T}(t) = \sqrt{\frac{1}{n_s N_g} \sum_{j=1}^{n_s} \sum_{i=1}^{N_g} (u_{r,j}(i,t) - u_{p,j}(i,t))^2}, \tag{18}$$

where N_g is the number of grid points of each fine-resolution sub-domain and n_s is the number of fine-resolution sub-domains, while $u_{r,j}(i,t)$ and $u_{p,j}(i,t)$ represent the fine-resolution reference solution and the patch solution on j^{th} fine-resolution sub-domain, respectively.



(a) Reference solution.

(b) Auxiliary data from a finite difference model with a 6×6 grid resolution.

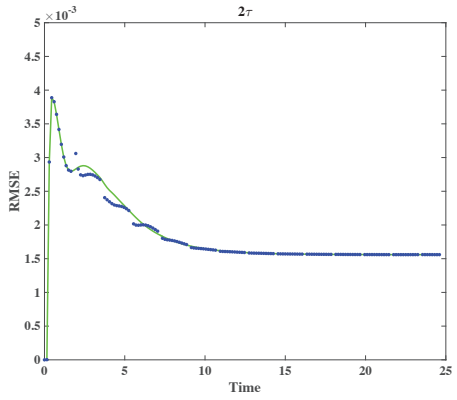
(c) Auxiliary data from a random walk model with 200 sample paths.

(d) Reference solution.

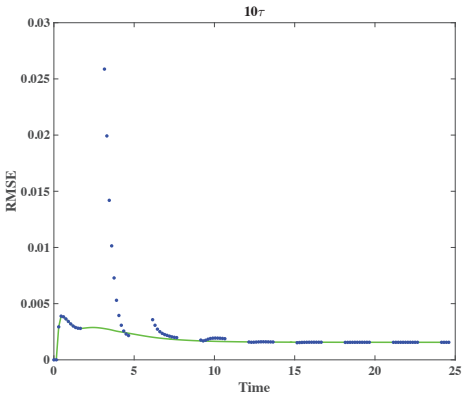
(e) Auxiliary data from a finite difference model with 8×8 grid resolution.

(f) Auxiliary data from a DPD model with 4000 paritcles.

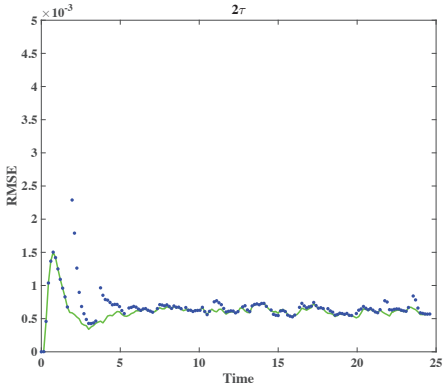
Figure 7: Contours of field variables at the steady-state. (a)-(c): Temperature contours for the heat equation. The reference simulation is obtained on the entire domain (31×31 grid). (d)-(f): Streamwise velocity contours for the Navier-Stokes equations. The reference simulation is obtained on the entire domain (27×27 cell). Results of patch simulations are based on fine-resolution sub-domains with different auxiliary data.



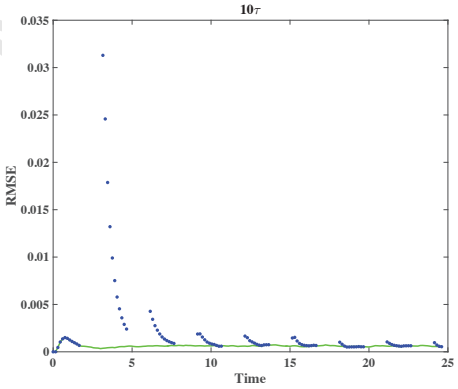
(a) Projection to $2 \cdot \tau \cdot \Delta t$.



(b) Projection to $10 \cdot \tau \cdot \Delta t$.

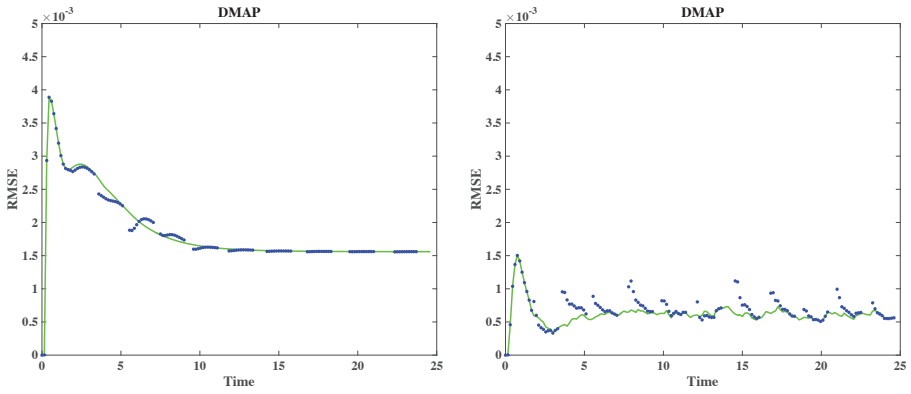


(c) Projection to $2 \cdot \tau \cdot \Delta t$.



(d) Projection to $10 \cdot \tau \cdot \Delta t$.

Figure 8: The time history of time-averaged RMS error for different projection time in the heat equation ((b) and (d) have different RMSE scales). The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with fixed projection time. (a)-(b): The auxiliary data comes from a finite difference model with resolution 6×6. (c)-(d): The auxiliary data comes from the random walk model with 2000 sample paths. We employ 10 snapshots in the projective time integration.



(a) Projection to $J(t) \cdot \tau \cdot \Delta t$. (b) Projection to $J(t) \cdot \tau \cdot \Delta t$.

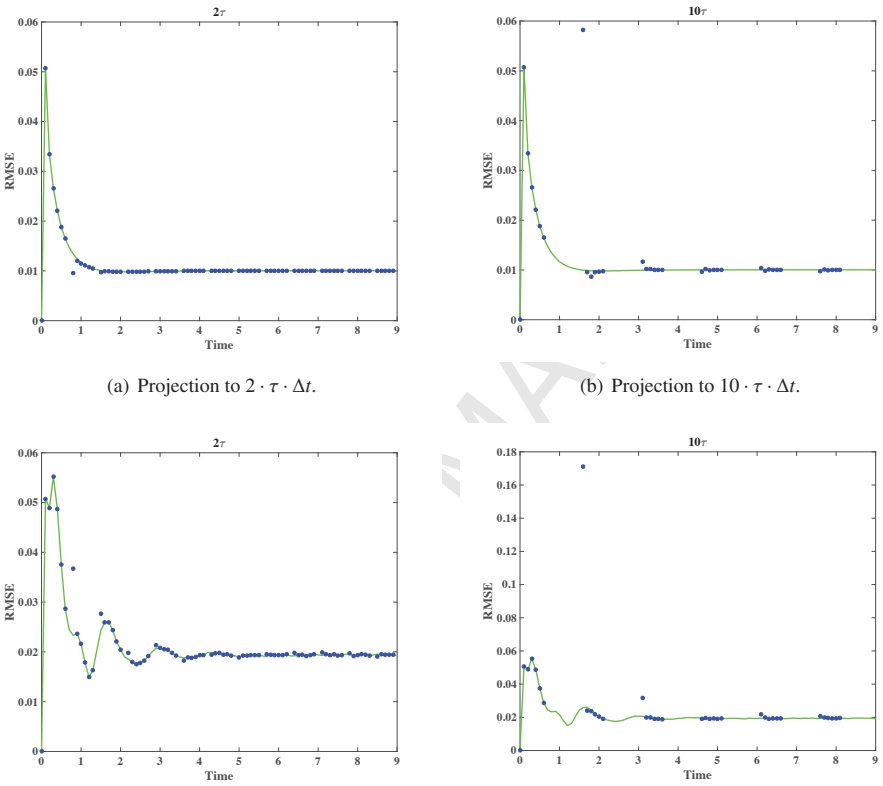
Figure 9: The time history of time-averaged RMS error for different projection time in the heat equation. The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with varying projection time by Diffusion maps. (a): The auxiliary data comes from a finite difference model with resolution 6×6. (b): The auxiliary data comes from the random walk model with 2000 sample paths. We employ 10 snapshots in the projective time integration.

As shown in Figures 8 and 10, for a small fixed projection time, the difference between $\mathrm{RMSE_T}$ of the gappy simulation and the patch simulation is relatively small. However, too many computations are needed near the steady-state when the change of dynamics is negligible. For a large fixed projection time, on the other hand, a large difference in $\mathrm{RMSE_T}$ is observed during the transient period. For projection time by diffusion maps in Figures 9 and 11, the gap between results of the gappy simulation and the patch simulation is small enough during the transient period (due to a small projection time), and the computation redundancy is also small enough near the steady-state (due to a large projection time).

In order to quantify these observations, we introduce three different measures: namely "a scaled maximum difference of $\mathrm{RMSE_T}$ (D_M)" for quantifying the accuracy, "a normalized computation time (T_C)" for the efficiency, and "an overall accuracy (ζ) compared to a reference gappy simulation. As shown in Figure 12, the RMS errors converge as the accuracy of auxiliary data increases for both the finite difference method and the particle based method. In the patch simulation, we add the projective time integration to the gappy simulation for acceleration of simulation. Since spatial gaps are preassigned by software or hardware error, the RMS error and computational time of the gappy simulation, which has same spatial gaps of the patch simulation, are used to measure "accuracy loss" and "efficiency gain" of the patch simulation. Hence, we employ the gappy simulation as the reference simulation to measure accuracy and efficiency in the same circumstance. In this paper, we choose as auxiliary data simulations from the finite difference method with a 24×24 grid resolution for both the heat equation and the Navier-Stokes equations in the reference simulations. Then, the D_M is calculated by the difference between RMSE of two simulations:

$$D_M = \max\left(\left|\mathrm{RMSE}_{\mathrm{T}}^{\mathrm{p}}(t) - \mathrm{RMSE}_{\mathrm{T}}^{\mathrm{g}}(\mathrm{t})\right|\right) \cdot R, \quad t \in (0, T), \tag{19}$$

where superscripts "p" and "g" represent the patch simulation and the gappy simulation, respec-



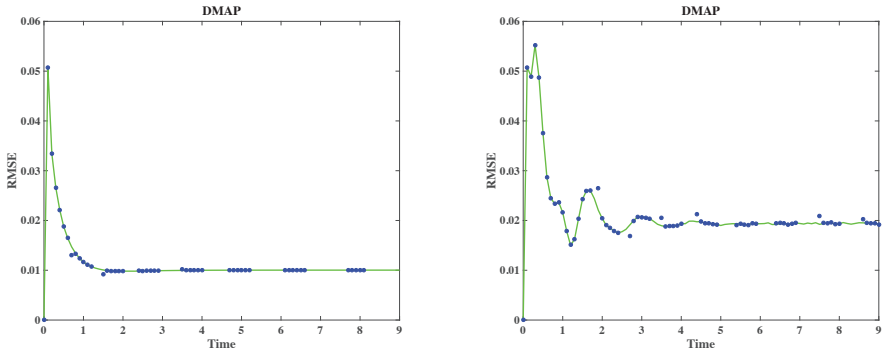
(a) Projection to $2 \cdot \tau \cdot \Delta t$.

(b) Projection to $10 \cdot \tau \cdot \Delta t$.

(c) Projection to $2 \cdot \tau \cdot \Delta t$.

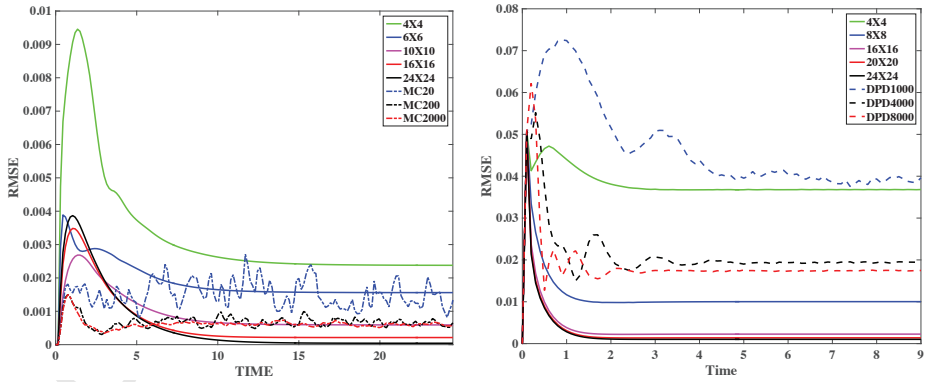
(d) Projection to $10 \cdot \tau \cdot \Delta t$.

Figure 10: The time history of time-averaged RMS error for different projection time in the Navier-Stokes equations ((d) has a different RMS scale). The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with fixed projection time. (a)-(b): The auxiliary data comes from a finite difference model with resolution 8×8. (c)-(d): The auxiliary data comes from the DPD result with 4000 DPD particles. We employ 5 snapshots in the projective time integration.



(a) Finite difference method. (b) Dissipative particle dynamics (DPD).

Figure 11: The time history of time-averaged RMS error for different projection time in the Navier-Stokes equations. The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with varying projection time by Diffusion maps. (a): The auxiliary data comes from a finite difference model with resolution 8×8. (b): The auxiliary data comes from the DPD result with 4000 DPD particles. We employ 5 snapshots in the projective time integration.



(a) Heat equation. (b) Navier-Stokes Equations.

Figure 12: $\mathrm{RMSE}_{\mathrm{T}}$ for different auxiliary data. The RMS errors are converged as the accuracy of the auxiliary data increases in both cases.

tively, while R is the ratio of the mean $\mathrm{RMSE}_{\mathrm{T}}^{\mathrm{g}}$ between each auxiliary data and the reference auxiliary data. Then, the RMS error of all auxiliary data can be scaled with respect to the reference gappy simulation. Another measure T_C is calculated by the ratio of total computation time of the patch simulation to the reference gappy simulation. There are inverse correlations between the two aforementioned measures, i.e., if D_M becomes larger, then T_C becomes smaller. Hence, we introduce the "overall accuracy (ζ)":

$$\zeta = 1 - (D_M \cdot T_C). \tag{20}$$

6.1. Accuracy versus efficiency

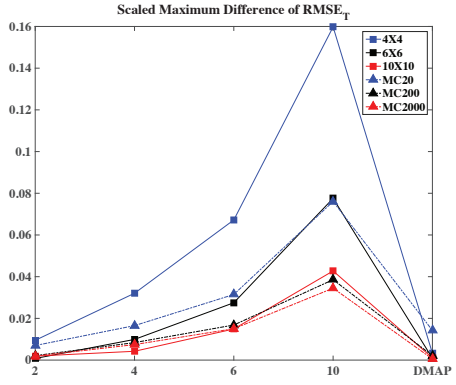
In the heat equation, for cases of fixed projection time only, T_C decreases linearly while D_M grows exponentially as projection time increases (see Figure 13). Hence, in the projective time integration, a small projection time requires a large computation time linearly while a large projection time loses accuracy exponentially. That is, a small projection time has higher overall accuracy compared to a large projection time. However, varying the projection time by the diffusion distance gives a value for D_M that is similar to the value for a fixed projection time of 2τ, and a value for T_C that is similar to the value for a fixed projection time of 6τ. This leads to a better overall accuracy (ζ). The patch simulation with the random walk model (stochastic and particle based auxiliary data) has the same trend as the finite difference method (deterministic and grid based auxiliary data), i.e., varying projection time by the diffusion distance always provides the highest overall accuracy for any auxiliary data. The random walk model with 2000 sample paths is found to be the best auxiliary data with respect to the overall accuracy, although the finite difference model with a 10×10 grid is comparable. Since the random walk model solves the heat equation on a 10×10 grid resolution, D_M appears to be similar to the finite difference method with a 10×10 grid.

In the Navier-Stokes equations, for cases of fixed projection time with a finite difference method only, results are same as the heat equation, i.e., varying projection time by the diffusion distance provides the highest overall accuracy; D_M is at a similar level as for a fixed projection time of 2τ, and T_C is at a similar level as for fixed projection time of 4τ or 6τ. However, the DPD model with 1000 particles (too coarse) cannot provide an adequate projection time due to its incorrect slow manifold, which results in a decrease in overall accuracy even though we employ the varying projection time. As the number of DPD particles increases from 1000 to 8000, the diffusion map provides the appropriate projection time, which leads to large improvement of both the accuracy and the efficiency (see Figure 14). Furthermore, since differences in T_C for all cases are negligible (except the finite difference with 4×4 grid), D_M determines the overall accuracy. Finally, the finite difference model with a 16×16 grid (the lowest D_M) is found to be the best auxiliary data with respect to the overall accuracy.

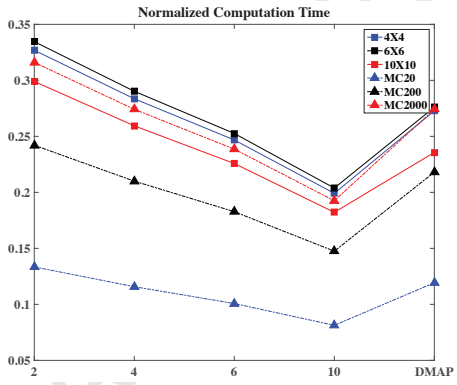
From the results of all simulations, we conclude that varying projection time at every projection step, based on the rate of temporal variation of the local leading diffusion maps coordinate, is the best method to achieve the highest overall accuracy for any auxiliary data. Moreover, the heterogenous models (based on particles) are comparable as the number of particles increase. This result shows that the patch simulation provides high quality even when using heterogeneous information fusion.

6.2. Projection time and auxiliary data

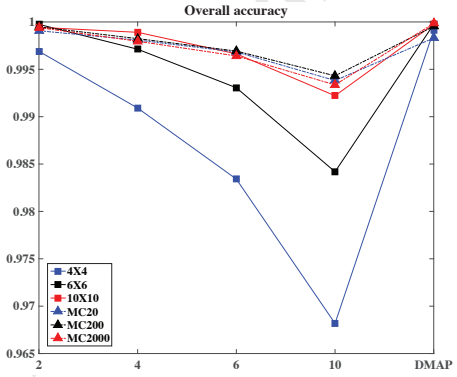
Diffusion maps provide a non-linear low-dimensional manifold associated with the slow dynamics of the original problem. The diffusion maps from accurate auxiliary data draw a smooth



(a) Scaled maximum difference of $\mathrm{RMSE_T}$ (D_M).

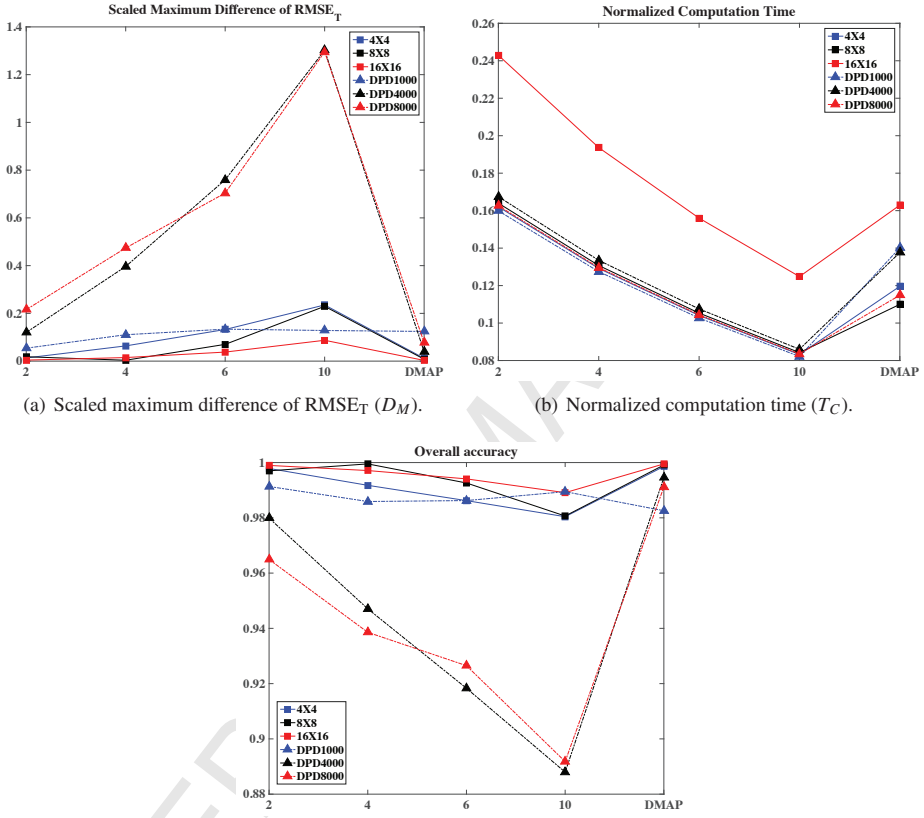


(b) Normalized computation time (T_C).



(c) Overall accuracy (ζ).

Figure 13: Three computational quality measures in the heat equation: the x-axis represents a fixed projection time for the projective time integration and "DMAP" represents a varying projection time by the diffusion maps. Square and triangle markers represent auxiliary data from a finite difference model with a coarse grid and a random walk model by Monte Carlo method, respectively.

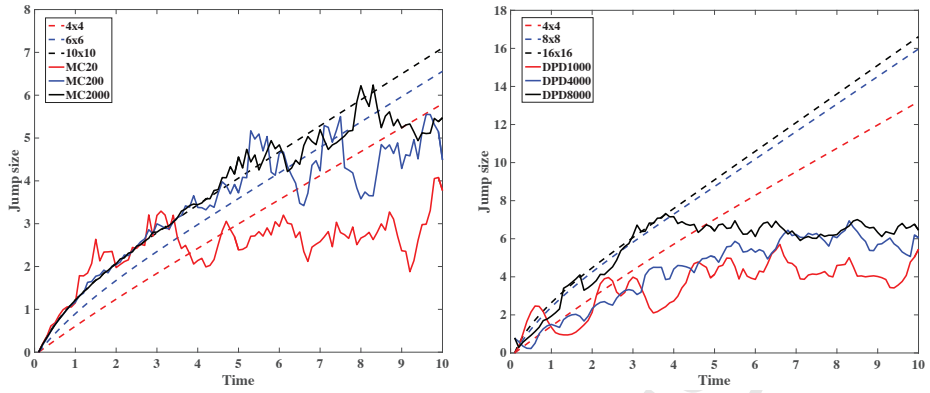


(a) Scaled maximum difference of $\mathrm{RMSE_T}$ (D_M).

(b) Normalized computation time (T_C).

(c) Overall accuracy (ζ).

Figure 14: Three computational quality measures in the Navier-Stokes equations: the x-axis represents a fixed projection time for the projective time integration and "DMAP" represents a varying projection time by the diffusion maps. Square and triangle markers represent auxiliary data from a finite difference model with a coarse grid and a DPD model, respectively.



(a) Heat equation. (b) Navier-Stokes equations.

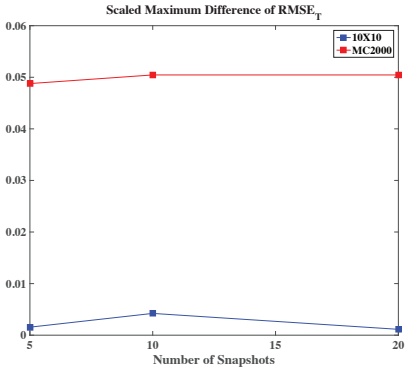
Figure 15: Projection time with respect to time by the diffusion maps. The dashed lines represent projection time for the finite difference model in both cases. The solid lines represent projection time for the random walk model in the heat equation and the DPD model in the Navier-Stokes equations, respectively.

coordinate which contains slow dynamics precisely and this results in increasing projection time compared to inaccurate auxiliary data (see Figure 15). Hence, T_C decreases as the auxiliary data becomes accurate due to a large projection time. The accuracy of the auxiliary data also affects computation time. For example, finite difference methods with high accuracy require a higher order formula or a more refined grid resolution, which results in consuming extra computation time to perform the auxiliary simulation. If we employ more refined grid resolution, we need additional computational time to obtain optimal hyper-parameters for the multi-level Gaussian process regression by the maximum likelihood estimation. Hence, more accurate auxiliary data cannot guarantee smaller computation time. However, more accurate auxiliary data provides more accurate snapshots, which results in a smaller D_M with the same projection time. Finally, we show that more accurate auxiliary data gives higher overall accuracy (ζ).

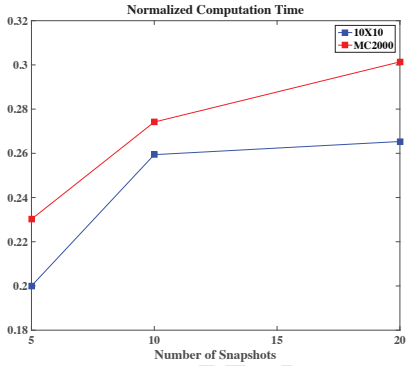
Next, we focus on projection time of heterogeneous models (stochastic and particle-based models). As shown in Figure 15, projection time is bounded above and has wiggles near the steady-state. As the system approaches the steady-state, the change of dynamics is very small (or negligible). This, in turn, means that the diffusion distance between adjacent snapshots at that time is very small. Thus, projection time (inverse logarithm scale of the diffusion distance) becomes very large and it can be affected strongly by a small perturbation of each snapshot. Thus, intrinsic perturbations of both stochastic models, the random walk model and the DPD model, cause the oscillation of projection time near the steady-state in both benchmark problems.

6.3. Number of snapshots

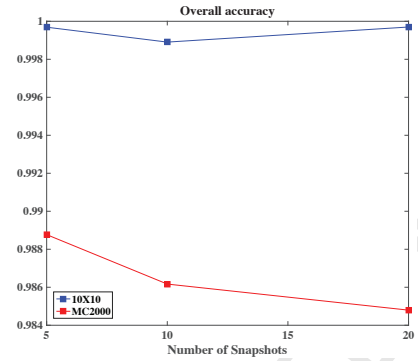
The number of saved snapshots, N, determines the number of terms of a POD expansion for reconstructing field variables in the projective time integration. Generally, more snapshots for the projective time integration with same auxiliary data and projection time reduce the truncated error of POD expansion and this leads to gaining accuracy [16] while the extent of simulation "saved" becomes less, leading to less efficiency.



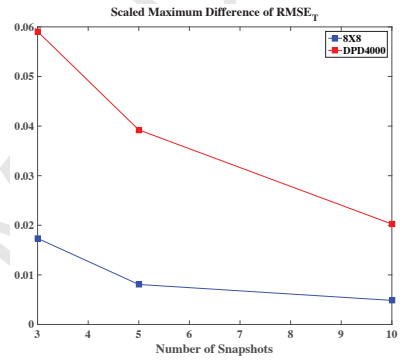
(a) Scaled maximum difference of $\mathrm{RMSE_T}$ (D_M).



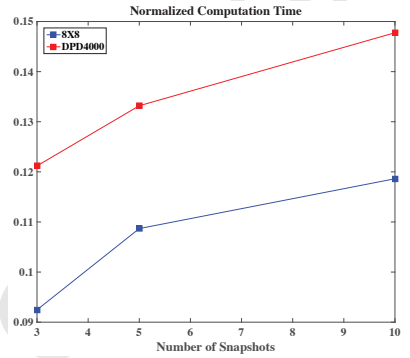
(b) Normalized computation time (T_C).



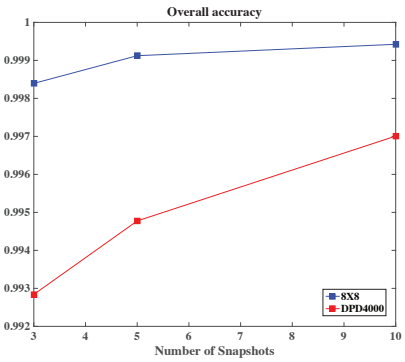
(c) Overall accuracy (ζ).



(d) Scaled maximum difference of $\mathrm{RMSE_T}$ (D_M).



(e) Normalized computation time (T_C).



(f) Overall accuracy (ζ).

Figure 16: Results of three computational quality measures with different number of snapshots. x-axis represents the number of snapshots for the projective time integration. (a)-(c): in the heat equation, the blue and the red represent the coarse grid (10×10) and the random walk model with 2000 sample paths, respectively. (d)-(f): in the Navier-Stokes equations, the blue and the red represent the coarse grid (8×8) and the DPD model with 4000 particles, respectively.

In the heat equation, as shown in Figures 16(a)-(c), accuracy gains with respect to D_M are very small even though T_C increases. This result shows that 5 snapshots, i.e., 5 terms of POD expansion, are enough to reconstruct and estimate field variables since the governing equation is linear. On the other hand, for the Navier-Stokes equations (which are much more complex than the heat equation and nonlinear equations), D_M decreases as the number of snapshots increases (see Figure 16(d)). The additional snapshots can reconstruct the field variables precisely, which leads to high accuracy for the patch simulation by using more terms of POD expansion. However, additional snapshots require more time steps to perform fine simulations for each projective time integration, which make T_C increase (see Figure 16(e)). From our observations, D_M has a more pronounced effect than T_c on the overall accuracy. Hence, as shown in Figure 16(f), more snapshots are suitable for this problem setup. Generally, the nonlinear governing equation requires more snapshots to gain overall accuracy.

7. Summary

We formulated and implemented a new framework to extend gappy simulation with a "patch dynamics" flavor and tested simulations of two benchmark problems, namely the heat equation and flow in a lid-driven cavity in two dimensions. Furthermore, we demonstrate the new capability that statistical learning tools help achieve with heterogeneous and multi-fidelity data. By learning from the auxiliary data, the new algorithmic framework can achieve not only high accuracy via the Gaussian process regression, but also high temporal integration efficiency via the diffusion maps. From results of simulations, we observed important trends about the patch simulation. Accurate auxiliary data provides accurate data sets in the Gaussian process regression and the appropriate projection time from the diffusion map. Moreover, the number of snapshots we use for the projective time integration totally depends on the dynamics of the problem. Generally, more snapshots increase the accuracy but decrease the efficiency.

We demonstrated that this framework can be extended to enable *multi-fidelity* and *multiscale* parallel simulations in a resilient way. In future work, we will further extend this framework to address *multi-rate* dynamics. For example, we can employ velocity field data as the auxiliary data for simulating concentrations of species driven by a convection-diffusion equation. Then, multi-species concentrations in the same velocity can be solved much more accurately and efficiently by considering only one velocity problem (as the auxiliary data) in this framework. It is also highly desirable to investigate the numerical stability of this framework with respect to auxiliary data, gap size and projection time, and we are currently pursuing such an investigation.

8. Acknowledgment

S. Lee and G. E. Karniadakis gratefully acknowledge the financial support of the Army Research Office (ARO W911NF-14-1-0425), and also the Air Force Office of Scientific Research (AFOSR FA-9550-12-1-0463). I. G. Kevrekidis gratefully acknowledges the partial support of the National Science Foundation. All DPD simulations were performed at the facilities of Center for Computation and Visualization in Brown University.

References

[1] S. Lee, I. G. Kevrekidis, G. E. Karniadakis, A general CFD framework for fault-resilient simulations based on multi-resolution information fusion, Journal of Computational Physics, *under review*.

[2] J. Dongarra, et al., The international exascale software project roadmap, International Journal of High Performance Computing Applications 25 (1) (2011) 3–60.
[3] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D. Mavriplis, CFD vision 2030 study: a path to revolutionary computational aerosciences, NASA/CR–2014-218178, Tech. rep., NASA (2013).
[4] F. Cappello, A. Geist, W. Gropp, S. Kale, B. Kramer, M. Snir, Toward exascale resilience: 2014 update, Supercomputing frontiers and innovations 1 (1) (2014) 5–28.
[5] D. K. Pradhan, Fault-tolerant computer system design, Prentice-Hall, Inc., 1996.
[6] G. E. Fagg, J. J. Dongarra, FT-MPI: Fault tolerant MPI, supporting dynamic applications in a dynamic world, in: European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting, Springer, 2000, pp. 346–353.
[7] B. Y. Zhao, J. Kubiatowicz, A. D. Joseph, et al., Tapestry: An infrastructure for fault-tolerant wide-area location and routing, UCB/CSD-01-1141, Tech. rep. (2001).
[8] J. Larson, M. Hegland, B. Harding, S. Roberts, L. Stals, A. Rendell, P. Strazdins, M. Ali, C. Kowitz, R. Nobes, J. Southern, N. Wilson, M. Li, Y. Oishi, Fault-tolerant grid-based solvers: Combining concepts from sparse grids and mapreduce, Procedia Computer Science 18 (2013) 130–139.
[9] D. Venturi, G. E. Karniadakis, Gappy data and reconstruction procedures for flow past a cylinder, Journal of Fluid Mechanics 519 (2004) 315–336.
[10] K. Willcox, Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition, Computers & Fluids 35 (2) (2006) 208–226.
[11] H. Ltaief, E. Gabriel, M. Garbey, Fault tolerant algorithms for heat transfer problems, Journal of Parallel and Distributed Computing 68 (5) (2008) 663–677.
[12] S. Lee, I. G. Kevrekidis, G. E. Karniadakis, Resilient algorithms for reconstructing and simulating gappy flow fields in CFD, Fluid Dynamics Research 47 (5) (2015) 051402.
[13] R. Löhner, An adaptive finite element scheme for transient problems in CFD, Computer Methods in Applied Mechanics and Engineering 61 (3) (1987) 323–338.
[14] F. Shi, J. T. Kirby, J. C. Harris, J. D. Geiman, S. T. Grilli, A high-order adaptive time-stepping TVD solver for Boussinesq modeling of breaking waves and coastal inundation, Ocean Modelling 43 (2012) 36–51.
[15] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, C. Theodoropoulos, Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis, Communications in Mathematical Sciences 1 (4) (2003) 715–762.
[16] S. Sirisup, G. E. Karniadakis, D. Xiu, I. G. Kevrekidis, Equation-free/Galerkin-free POD-assisted computation of incompressible flows, Journal of Computational Physics 207 (2) (2005) 568 – 587.
[17] G. Samaey, I. G. Kevrekidis, D. Roose, Patch dynamics with buffers for homogenization problems, Journal of Computational Physics 213 (1) (2006) 264–287.
[18] J. Quiñonero-Candela, C. E. Rasmussen, A unifying view of sparse approximate Gaussian process regression, Journal of Machine Learning Research 6 (Dec) (2005) 1939–1959.
[19] C. E. Rasmussen, C. K. I. Williams, Gaussian processes for machine learning, MIT Press, 2006.
[20] P. Perdikaris, D. Venturi, J. Royset, G. Karniadakis, Multi-fidelity modelling via recursive co-kriging and Gaussian–Markov random fields, in: Proceedings of the Royal Society A, Vol. 471, The Royal Society, 2015, p. 20150018.
[21] P. Perdikaris, D. Venturi, G. E. Karniadakis, Multifidelity information fusion algorithms for high-dimensional systems and massive data sets, SIAM Journal on Scientific Computing 38 (4) (2016) B521–B538.
[22] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, S. W. Zucker, Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps, Proceedings of the National Academy of Sciences of the United States of America 102 (21) (2005) 7426–7431.
[23] R. R. Coifman, S. Lafon, Diffusion maps, Applied and Computational Harmonic Analysis 21 (1) (2006) 5–30.
[24] B. Nadler, S. Lafon, R. R. Coifman, I. G. Kevrekidis, Diffusion maps, spectral clustering and reaction coordinates of dynamical systems, Applied and Computational Harmonic Analysis 21 (1) (2006) 113–127.
[25] R. R. Coifman, I. G. Kevrekidis, S. Lafon, M. Maggioni, B. Nadler, Diffusion maps, reduction coordinates, and low dimensional representation of stochastic systems, Multiscale Modeling & Simulation 7 (2) (2008) 842–864.
[26] X. Nie, S. Chen, M. Robbins, A continuum and molecular dynamics hybrid method for micro-and nano-fluid flow, Journal of Fluid Mechanics 500 (2004) 55–64.
[27] T. Werder, J. H. Walther, P. Koumoutsakos, Hybrid atomistic–continuum method for the simulation of dense fluid flows, Journal of Computational Physics 205 (1) (2005) 373–390.
[28] Y.-H. Tang, S. Kudo, X. Bian, Z. Li, G. E. Karniadakis, Multiscale universal interface: A concurrent framework for coupling heterogeneous solvers, Journal of Computational Physics 297 (2015) 13–31.
[29] H. Cho, X. Yang, D. Venturi, G. E. Karniadakis, Algorithms for propagating uncertainty across heterogeneous domains, SIAM Journal on Scientific Computing 37 (6) (2015) A3030–A3054.
[30] R. Everson, L. Sirovich, Karhunen-Loeve procedure for gappy data, Journal of the Optical Society of America A 12 (1995) 1657–1664.
[31] M. C. Kennedy, A. O'Hagan, Predicting the output from a complex computer code when fast approximations are

available, Biometrika 87 (1) (2000) 1–13.
[32] M. Stein, Interpolation of spatial data: some theory for Kriging, Berlin: Springer, 1999.
[33] L. Sirovich, Turbulence and the dynamics of coherent structures. i. coherent structures, Quarterly of applied mathematics 45 (3) (1987) 561–571.
[34] D. A. Fedosov, I. V. Pivkin, G. E. Karniadakis, Velocity limit in DPD simulations of wall-bounded flows, Journal of Computational Physics 227 (4) (2008) 2540–2559.
[35] J. Backer, C. Lowe, H. Hoefsloot, P. Iedema, Poiseuille flow to measure the viscosity of particle model fluids, The Journal of Chemical Physics 122 (15) (2005) 154503.