A general CFD framework for fault-resilient simulations based on multi-resolution information fusion

Seungjoon Lee\textsuperscript{a}, Ioannis G. Kevrekidis\textsuperscript{b}, George Em Karniadakis\textsuperscript{a,}\textsuperscript{*}

\textsuperscript{a} Division of Applied Mathematics, Brown University, Providence, RI 02912, United States
\textsuperscript{b} Department of Chemical and Biological Engineering, PACM, Princeton University, Princeton, NJ 08544, United States

\begin{abstract}
We develop a general CFD framework for multi-resolution simulations to target multiscale problems but also resilience in exascale simulations, where faulty processors may lead to gappy, in space-time, simulated fields. We combine approximation theory and domain decomposition together with statistical learning techniques, e.g. coKriging, to estimate boundary conditions and minimize communications by performing independent parallel runs. To demonstrate this new simulation approach, we consider two benchmark problems. First, we solve the heat equation (a) on a small number of spatial “patches” distributed across the domain, simulated by finite differences at fine resolution and (b) on the entire domain simulated at very low resolution, thus fusing multi-resolution models to obtain the final answer. Second, we simulate the flow in a lid-driven cavity in an analogous fashion, by fusing finite difference solutions obtained with fine and low resolution assuming gappy data sets. We investigate the influence of various parameters for this framework, including the correlation kernel, the size of a buffer employed in estimating boundary conditions, the coarseness of the resolution of auxiliary data, and the communication frequency across different patches in fusing the information at different resolution levels. In addition to its robustness and resilience, the new framework can be employed to generalize previous multiscale approaches involving heterogeneous discretizations or even fundamentally different flow descriptions, e.g. in continuum-atomistic simulations.

© 2017 Elsevier Inc. All rights reserved.
\end{abstract}

1. Introduction

The present work introduces a new paradigm in Computational Fluid Dynamics (CFD) and is motivated by two facts: (1) The lack of robust and efficient methods to carry out multiscale simulations in practical applications, despite several published papers proposing coupling techniques for heterogeneous flow models \cite{1–5}. (2) The occurrence of random faults from hardware or software that may render the simulation results erroneous on petaflop and on the emerging exaflop computing platforms \cite{6–9}. While one may think that the latter is a computer science issue and can be solved e.g. via fault-tolerant MPI \cite{10–13}, there may be an algorithmic solution to it that can lead to effective recovery of the computation or protecting against the erroneous results in the first place. Here we develop a framework that addresses simultaneously fundamental open issues of the aforementioned two topics. This framework can be readily generalized to accommodate heterogeneous flow models as well as deterministic or stochastic descriptions; in this first paper we focus only on multi-
resolution to layout the basic ideas. One fundamental question is how to reconstruct complete fields from gappy data and some auxiliary information. This issue has been addressed before in different contexts, including the dynamical systems approach with the so-called gap-tooth and patch dynamics algorithm [14–16]. However, this algorithm has not yet been applied to the Navier–Stokes equations, which is the subject we examine in the current work, by also introducing reconstruction techniques employed by the statistical learning community.

There are many types of fault scenarios in exascale simulations. Regardless of types of fault, we focus only on reconstructing and performing the simulation in algorithmic sides after faults have occurred. Considering the faulty processors scenario, we may assume that we encounter permanent spatial gaps in the computational domain at certain time. Hence, we need to reconstruct the gaps with appropriate methods, e.g. based on estimation theories [17–21]. In particular, for spatial estimation for gaps in fluid dynamics, the gappy proper orthogonal decomposition (gappy POD) method was introduced in [22–24], which extrapolates the POD basis from previous data. Another spatial estimation method used often in geophysics is Kriging and cokriging, which are unbiased linear interpolations [25–29]. More recently, a resilient algorithm based on a “re-simulation” method for CFD was introduced in [30], demonstrating the possibility of filling the spatial gaps effectively at a fixed time.

In the present paper, we aim to combine the gap-tooth algorithm with information fusion methods for field reconstruction emphasizing robustness and generality. We assume that we only have gaps in the spatial domain but not gaps in time. We introduce a new CFD algorithm for parallel simulations, namely a gappy simulation with auxiliary data, which is capable of performing uninterrupted simulations despite the occasional presence of spatial gaps. Furthermore, by employing techniques from statistical learning for information fusion, we can potentially increase the overall simulation accuracy by utilizing low-resolution auxiliary data from diverse sources. This new capability allows us to simulate multiscale but also multiresolution discretization. The new framework can be extended to generalize previous multiscale approaches (e.g. continuum-atomistic) [31] in a unified parallel computational framework.

The paper is organized as follows: In section 2, we introduce the gappy simulation algorithms with a definition and a flow chart. In section 3, we present the computational domains and simulation set up for two benchmark problems. In section 4, we present results of a parametric study and analyze them in terms of accuracy. In section 5, we summarize our results and discuss open issues for further developments of gappy simulations.

2. Gappy simulation framework

2.1. Problem set up

In general frameworks for exascale simulations, each sub-domain needs local boundary conditions which come from adjacent sub-domains. If the values of field variables at the local boundaries are missing, we are not able to assign correct local boundary conditions, which makes our simulation unsatisfactory. Hence, resilient algorithms require to (1) simulate with spatial gaps and (2) assign appropriate local boundary conditions within reasonable error margin.

In the gappy simulation framework we compute explicitly the solution to a PDE not on the entire domain but only partially on some sub-domains (colored by green) with some auxiliary data that are distributed across the entire domain (colored by blue) and obtained independently, see Fig. 1. The main idea is to combine the global coarse information with some finely resolved sub-domains and appropriately fuse the two solutions to obtain a more accurate solution on the entire domain. This set up admits two different interpretations. From the multiscale perspective, the global coarse solution represents the large scales, whereas the solution on the fine-resolution sub-domains represents dynamics of finer scales. From the parallel computing perspective, the gappy sub-domains may be regions corrupted by random software or hardware faults whereas the global coarse solution is obtained on an independent small set of processors, which is assumed to be immune to such faults that the big computer system may suffer from. This framework has some similarities with the
“gap-tooth” algorithm for micro-simulators [14–16]. In the micro-systems, the computational cost increases greatly as the number of particles or the size of the computational domain increases. Hence, the “gap-tooth” algorithm was introduced to solve only partial sub-domains (in a “micro” sense) and smoothly interpolate/extrapolate state variables (in a “macro” sense) at the expense of some penalty in accuracy but with a large computational gain and thus enhanced computational efficiency.

2.2. Auxiliary data

The auxiliary data is a key component of the gappy simulation. First, to facilitate resilience, the auxiliary data should be independent of the main simulation by computing them on a separate computer node or even by obtaining them experimentally, e.g. using a Particle–Image–Velocimetry (PIV) technique [32–34], see Fig. 2. Moreover, the auxiliary data should be small in size so that it can be saved on a fast disk, e.g. a solid state device. In order to be (approximately) in synch with the parallel main simulation, the auxiliary data should be much faster to compute compared to the main parallel simulation or be given in advance. Alternatively, a lower-dimensional model can be constructed based on a coarse-grained simulation and be run in parallel with the main simulation. Since the auxiliary data can be obtained by a relatively small number of computer nodes (or only one node) or given in advance, it is rare to detect faults in the auxiliary data compared to the main parallel simulation. Irrespective of the scenario employed to obtain the auxiliary data, we will assume here that they provide information of lower resolution and of course at lower fidelity but can be acquired much faster compared to the main simulation without failures. Second, because the auxiliary data span the entire domain, they contain information of global connectivity, which the main simulation does not have. Based on methods of information fusion, we can then endow the estimated field variables at the local boundary with both global (from auxiliary data) as well as local information (from the gappy domains).

2.3. Algorithm flow chart

A flow chart of the gappy simulation is shown in Fig. 3. First, upon notification of a fault detection (not discussed here), we check which domains are affected by errors, and define computational sub-domains and gaps. Next, we choose a proper buffer size and corresponding boundary conditions for each sub-domain. After imposing the boundary conditions, the gappy simulation estimates the field variables at the local boundaries of each sub-domain by the information fusion method using also the independent auxiliary data (coKriging). After setting-up all the parameters and variables, the gappy simulation solves each sub-domain on independent nodes during non-interaction time \( \tau \cdot \Delta t \). After time \( \tau \cdot \Delta t \), all sub-domains are re-joined together and the buffer region of each sub-domain is cut-off. Finally, using the auxiliary data, the new field variables at the boundaries can be updated via coKriging. The gappy simulation repeats again this procedure until the main simulation ends or all faults are fixed.

2.4. CoKriging

In order to estimate field variables at the local boundary, a multi-fidelity coKriging interpolation method is introduced [30,35,36]. In this paper, we use two data sets: data from the gappy simulation, locally computed with high accuracy, and data from auxiliary data, globally computed with low accuracy, see Fig. 4. The basic idea of this method is that the estimated field variable \( \hat{y}(x) \) at the local boundary \( x \) can be represented by a linear combination of two data sets as follows:

\[
\hat{y}(x_b) = \lambda_1^T x + \lambda_2^T y,
\]

where \( x \in \mathbb{R}^n \) is a field data vector in the auxiliary data and \( y \in \mathbb{R}^m \) is in the gappy simulation data. Also, \( \lambda_1 \) and \( \lambda_2 \) are vectors of weights which will be determine below. Here we solve a simple optimization problem in minimizing the mean squared error (MSE) of this linear combination, i.e.,

\[
\arg\min_{\lambda_1, \lambda_2} E[ (\hat{y}(x_b) - y(x_b))^2 ].
\]
Fig. 3. A flow chart for a gappy simulation (start from left-top): We first check where the gappy domains are located. Next, we choose a buffer size, impose appropriate boundary conditions, and estimate field variables at local boundaries. Each sub-domain is solved in parallel and independently during non-interaction time $\tau \cdot \Delta t$. Subsequently, all gappy domains are re-joined together after cutting-off the buffer region. Finally, fusing information from the fine resolution patches with auxiliary data, all field variables are updated at the local boundaries (buffers) of the sub-domains. This is one complete cycle of the gappy simulation algorithm.

subject to the unbiased constraints

$$E[\hat{y}(\mathbf{x}_b)] = E[y(\mathbf{x}_b)] \quad \text{or} \quad \sum_{i=1}^{n} \lambda_{1i} = 1 \quad \text{and} \quad \sum_{i=1}^{m} \lambda_{2i} = 0. \quad (3)$$

Then, we can solve the linear system with Lagrange multipliers $\mu_1$ and $\mu_2$ as follows:

$$\begin{bmatrix} C_{11} & C_{12} & 1 & 0 \\ C_{21} & C_{22} & 0 & 1 \\ 1^T & 0^T & 0 & 0 \\ 0^T & 1^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} c_1(\mathbf{x}_b) \\ c_2(\mathbf{x}_b) \\ 1 \\ 0 \end{bmatrix} \quad (4)$$

where $C$ is a covariance matrix whose submatrix $C_{11}$ is the covariance matrix between points in auxiliary data, $C_{22}$ between points in the fine-resolution sub-domains, and $C_{12}$ between the auxiliary data and the sub-domains. Here, $c_1(\mathbf{x})$ is the
covariance vector between the target point \((\mathbf{x})\) and points in the auxiliary data, and \(c_{2}(\mathbf{x})\) between the target point \((\mathbf{x})\) and points in the sub-domains. In order to set a covariance matrix, we need to choose the proper correlation kernel, see section 4.1.

2.5. Buffer region

In this framework we choose a Dirichlet boundary condition as the default type. If we have no auxiliary data (via Kriging) and no buffer, the field variables at the local boundary of each sub-domain cannot be changed by the basic property of Kriging estimation - the estimated value at the training data point should be the same as the training data. Even though we employ the coKriging estimation with auxiliary data, there is still “default” error coming from the estimation, which may render our simulation useless after long time integration. In order to prevent this negative effect from polluting the solution in the interior of our sub-domains during the non-interaction time \(\tau \cdot \Delta t\), we introduce the concept of a “buffer”, which plays a similar role as that of a “dashpot” in the classical vibration system of a mechanical design. We choose a proper size of the buffer as shown in Fig. 5 by balancing computational cost and desired accuracy; this size should be correlated to the dominant time scales of the problem, e.g. diffusion or convection time scale.

3. Simulation setup

3.1. Heat equation

We consider a square domain for simplicity, and we solve the two-dimensional heat equation with diffusivity, \(\kappa\), given by:

\[
\frac{\partial T}{\partial t} = \kappa \nabla^2 T, \tag{5}
\]

with proper boundary conditions at the boundaries of the domain. This is a simpler problem than the Navier–Stokes equations but it serves as a pedagogical example to introduce all the steps of the algorithmic framework we propose.

In order to perform a fine-resolution gappy simulation and a fine-resolution reference simulation on the complete domain, we employ the second-order finite difference method. The physical boundary conditions are: \(T = 1\) at the top and \(T = 0\) on all other sides. The computational domain consists of a structured rectangular mesh. The grid resolution of the gappy simulation is \(11 \times 11\) per each sub-domain (total of 5 sub-domains), with a total of 605 grid points in the gappy simulation while the grid resolution of the reference simulation is \(31 \times 31\). In order to compare the influence of auxiliary data on the overall accuracy, we employ a coarse grid with resolution \(4 \times 4, 6 \times 6,\) and \(10 \times 10\). Four sub-domains have two global boundaries but one sub-domain in the middle, cell 3, has no global boundary, see Fig. 6(a). The temporal discretization corresponds to time step \(\Delta t = 0.0125\) with heat diffusivity, \(\kappa = 0.01\). The simulation is integrated from \(t = 0\) to \(t = 30\) (the steady-state is near \(t = 25\)). Temperature contours at the steady-state are shown in Fig. 7.

3.2. Navier–Stokes equation

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

\[
\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} + \mathbf{f}, \tag{6}
\]

\[
\nabla \cdot \mathbf{v} = 0, \tag{7}
\]
Fig. 6. A schematic illustration of gappy domains for two benchmark problems: (left) the global (physical) boundary condition of the reference simulation, (right) the location and index of the fine-resolution sub-domains colored by green. In the heat equation (a), the fine-resolution sub-domains are connected by only the corner point of cell “3” while the fine-resolution sub-domains are totally disconnected in the Navier-Stokes equation (b). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 7. Temperature contours for the heat equation: the reference simulation solves the entire domain (31 × 31 grid) by a second-order finite difference method.

where $\mathbf{v}$ is the velocity vector, $p$ is pressure, and $\nu$ is the kinematic viscosity of the fluid. For spatial discretization we employ a two-dimensional finite difference method. For the physical boundary condition, we prescribe the streamwise velocity, $u = 1$ at the top and $u = 0$ on all other sides; the crossflow velocity $v = 0$ at all boundaries, see Fig. 6(b). The resolution of a rectangular cell for each sub-domain is $8 \times 8$, with total 320 cells for the gappy simulation. In order to
compare errors from different auxiliary data quantitatively, we employ a coarse simulation with resolution $4 \times 4$, $8 \times 8$, and $16 \times 16$ for the entire domain. The gappy regions are similar as in the previous example but the only difference is that all fine-resolution sub-domains are totally disconnected, see Fig. 6(b). The time step is $\Delta t = 0.005$ with Reynolds number, $Re = 100$. The discretized equations are integrated from $t = 0$ to $t = 15$ (the steady-state is near $t = 10$). Streamwise velocity contours at the steady-state are shown in Fig. 8.

4. Results

In this section, we show results of the parametric study in terms of the correlation kernel, the size of buffer, the auxiliary data, and the non-interaction time $\tau \cdot \Delta t$. In order to compare effectiveness quantitatively, we employ two measures of RMS error, namely the “total” RMS error for temporal accuracy and the “time-averaged” RMS error for overall accuracy. The “total” RMS error at time $t$ is calculated by the following formula

$$\text{RMS}_T(t) = \sqrt{\frac{1}{nN} \sum_{j=1}^{n} \sum_{i=1}^{N} (u_{r,j}(i, t) - u_{g,j}(i, t))^2},$$

(8)

where $N$ is the number of grid point of each sub-domain and $n$ is the number of fine-resolution sub-domains while $u_{r,j}(i, t)$ and $u_{g,j}(i, t)$ represent the reference solution and the gappy solution in $j$th fine-resolution sub-domain, respectively. We also introduce another accuracy metric, the time-averaged RMS error from $t = 0$ to $t = T$, calculated by

$$\text{RMS}_T(T) = \frac{1}{T} \int_{0}^{T} \text{RMS}_T(t) \, dt.$$  

(9)

4.1. Correlation kernel

In order to set a covariance matrix by the information fusion via Kriging or coKriging, see section 2.4, we need to choose a proper correlation kernel (also called correlation function). We employ four different kernels widely used in general problems as follows:

- Gaussian
  $$\kappa_g(x_i, x_j) = \exp\left(-\theta d(x_i, x_j)^2\right).$$

(10)

- Exponential
  $$\kappa_e(x_i, x_j) = \exp\left(-\theta d(x_i, x_j)\right).$$

(11)

- Spherical
  $$\kappa_s(x_i, x_j) = 1 - \frac{3}{2} \min(\theta d(x_i, x_j), 1) + \frac{1}{2} \left(\min(\theta d(x_i, x_j), 1)\right)^3.$$

(12)
Matérn

\[ \kappa_m(x_i, x_j) = \left( 1 + \sqrt{\frac{3}{4}} d(x_i, x_j) \right) \exp \left( -\sqrt{3} d(x_i, x_j) \right). \]  

(13)

The hyperparameter \( \theta \) is obtained by maximum likelihood estimation (MLE); it decides how fast the correlation kernel converges to zero, i.e. the larger \( \theta \) represents that the points are affected by only nearer points; on the other hand, for small \( \theta \), all points are correlated to each other.

The results of constructing the covariance matrix at the steady-state by different correlation kernels are shown in Figs. 9 and 10. In order to compare the effectiveness of the kernel only, other test parameters should be fixed. In the heat equation, the size of the buffer is fixed at 30%, the non-interaction timestep number \( (\tau) \) is 50, and no auxiliary data are provided by Kriging. In the Navier–Stokes equation, the size of buffer is fixed at 25%, the non-interaction timestep number \( (\tau) \) is 5, and no auxiliary data are available by Kriging.

In the heat equation, we found that the best kernel at steady-state is the Matérn kernel. Theoretically, solutions of a diffusion problems yield strong spatial correlations between sub-domains. Thus, all kernels show a strong correlation between sub-domains except the Gaussian kernel. As shown in Figs. 10(a) and 11, the Gaussian kernel has weak correlation between other sub-domains and this leads to the highest RMS error compared to other kernels. Specifically, as shown in Fig. 9(b), the dominant RMS error comes from cell 3, where there is no global boundary condition. Hence, the accuracy of the boundary conditions in cell 3 is much more critical than the boundary conditions in other cells. Since the Gaussian kernel exhibits less communication between sub-domains due to its weak correlation, the updated boundary condition of cell 3 becomes inaccurate compared to the other kernels. Moreover, similar to the weak correlation, the strong correlation leads to higher RMS error as well. For example, the strong correlation of the Matérn kernel during the transient period makes the RMS error increase. However, as we approach the steady-state, the correlation is decreased smoothly and this results in a reduction of the RMS error, see Fig. 11.

In the Navier–Stokes equation, the spatial correlation between different sub-domains is relatively weaker than the heat equation. After the maximum likelihood estimation, we observed that the worst kernel is the spherical one which has weak correlations with other sub-domains. The exponential kernel has similar but relatively stronger correlation compared to the spherical kernel, see Fig. 10, and this leads to reduction of the RMS error. In the Gaussian kernel, however, the correlations between other sub-domains are too strong, that is, too many data can affect an estimated boundary condition. This over-fitting leads to wiggles in the total RMS error. Finally, we found that the Matérn kernel is the best because it has very strong correlation for the inner sub-domains but also an appropriate correlation with respect to directions. Specifically, in cell 1, the distance from cell 2 and cell 4 is the same but the correlation with the cell 4 is relatively strong, which means that the spatial correlation along the y-axis is much stronger than along the x-axis.

4.2. Size of a buffer

The next parameter we focus on is the size of the buffers we employ for estimating the boundary conditions on each fine-resolution sub-domains. The results of the total RMS error and time-averaged RMS error are shown in Fig. 12. The main observation is that the total and time-averaged RMS errors are reduced as the buffer becomes larger during the transient period but also at steady-state. Specifically, considering the heat equation first, the uncertainties from the estimation during the transient period due to a rapid change of field variables are large and this leads to high effectiveness of a large buffer, e.g., see the peak of the RMS error in Fig. 12(a). On the other hand, because the uncertainties at the local boundary are
Fig. 10. Covariance matrix of different correlation kernels at the steady-state. Fixed parameters are same as in Fig. 9. In (a), each fine-resolution sub-domain has 121 points, totally 605 points. In (b), each fine-resolution sub-domain has 90 points, totally 450 points.

relatively small near the steady-state, the associated errors can be diffused in the buffer region. Hence, the difference of the RMS error at the steady-state is smaller than the error during the transient states. Considering now the Navier–Stokes equation, the convection mechanism introduces a different time scale and the vector field presents another complication so the results are somewhat different compared with the diffusion equation. The size of the buffer becomes even more important both during the transient period and at steady-state.

4.3. Auxiliary data

The auxiliary data affects the accuracy of the estimation scheme via coKriging in estimating the boundary conditions for all fine-resolution sub-domains. The results of total and time-averaged RMS error for different auxiliary data are shown in
As a reference case, the results of the gappy simulation with no auxiliary data are added in the same graph; by that we refer to using the Kriging method instead of coKriging, which estimates the field variables with only one data set from the gappy simulation due to the absence of auxiliary data. From these results we can appreciate that the accuracy of the auxiliary data greatly affects the RMS errors in both cases.

In the simulation of the heat equation, the temperature contours of different resolutions of auxiliary data are shown in Fig. 13. The auxiliary data with finer resolution gives the lower RMS error at the steady state. However, at the transient region (t < 10), the case with no auxiliary data with Kriging leads also to a good reduction of RMS error. This is because all fine-resolution sub-domains have near zero value except the top boundary, i.e., cells 1 and 2. Thus, even though the Kriging method estimates near zero value at the local boundaries at cell 3, 4, and 5, these estimated values are not that different from the exact value. However, after t = 5, cell 3 exhibits nonzero values in the field variables at the local boundary. Hence, the RMS errors of the no-auxiliary data cases increase up to the steady-state.

In the simulation of the Navier–Stokes equation, the different resolutions of auxiliary data, see Fig. 14, show the relatively bigger effect of the auxiliary data compared to any other tested parameter. Because of the weak spatial correlations of the solution between the fine-resolution sub-domains, the auxiliary data can support the information associated with global interactions, and this results in significant reduction of the RMS error. However, the lowest resolution (4 × 4) is too coarse to be useful to the information fusion technique, see Fig. 14(a). Specifically, the center of vortex in the 4 × 4 auxiliary data is located near the middle of the domain whereas in truth it should lie toward near the top-right corner in the exact solution. This mismatch leads to big RMS error at cells 2 and 3, which affect the total RMS error.

4.4. Non-interaction timestep number (τ)

Employing a non-interaction timestep number, τ, as an independent parameter allows us to minimize communications by performing independent parallel runs during time \( t \cdot \Delta t \). From the multiscale modeling perspective, \( t \cdot \Delta t \) can represent a “macro” time step as compared to a micro time step, \( \Delta t \). The results of total and time-averaged RMS error with different τ are shown in Fig. 15. The difference of RMS errors between various τ is large during the transient period while the difference is relatively small (or negligible) at steady-state. Hence, we need to choose a proper τ by balancing computational cost and desired accuracy at target simulation time. For example, for long time integration until the steady-state, a large value of τ is a good choice with respect to computational cost.

In the heat equation, we have observed that the optimal τ depends on the penetration length for diffusion – the measure of how long each field variable can penetrate into the neighborhood numerically during a unit time step. Generally, the penetration length for the diffusion can be estimated by

\[
I_{p,d} \sim \sqrt{\kappa \cdot \tau \Delta t}.
\]

Then, the allowable non-interaction timestep number \( \tau_d \) for the diffusion, corresponding to the length of the buffer \( l_b \), is obtained as

\[
\tau_d = \frac{l_b^2}{\kappa \Delta t}.
\]
Fig. 12. The time history of total RMS error and time-averaged RMS error for different buffer sizes. In (b) and (d), the x-axis represents the percentage of the buffer with respect to the size of a fine-resolution domain. In the heat equation, the fixed parameters are as follows: resolution of auxiliary data is $6 \times 6$, and a non-interaction timestep number ($\tau$) of 1. In the Navier–Stokes equation, the fixed parameters are as follows: resolution of auxiliary data is $8 \times 8$, and a non-interaction timestep number ($\tau$) of 5.

Fig. 13. Temperature contours for the heat equation by different auxiliary data from coarse grids.
Fig. 14. Streamwise velocity contours for the Navier–Stokes equation by different auxiliary data from coarse grids.

Fig. 15. The time history of total RMS error in the heat equation.

(a) Time history of total RMS error in the heat equation.
(b) Time-averaged RMS error in the heat equation.
(c) Time history of total RMS error in the Navier-Stokes equation.
(d) Time-averaged RMS error in the Navier-Stokes equation.

The time history of time-averaged and total RMS errors for different auxiliary data. In the heat equation, fixed parameters are as follows: size of buffer at 30% and non-interaction timestep number (τ) of 1. In the Navier–Stokes equation, the fixed parameters are as follows: size of buffer at 25% and with non-interaction timestep number (τ) of 5.
Fig. 16. Time history of time-averaged and total RMS errors for different non-interaction timestep number ($\tau$). In (a), the dashed lines correspond to $\tau \geq \tau_d = 64.8$. In (c), the dashed lines correspond to $\tau \geq \tau_a = 15$. In the heat equation, (a) and (b), the fixed parameters are as follows: auxiliary data $6 \times 6$, and buffer of 30%. In Navier-Stokes equation, (c) and (d), the fixed parameters are as follows: auxiliary data $8 \times 8$, and buffer of 25%.

First, we obtain $\tau_d$ for a 30% and a 20% buffer, which is about 64.8 and 28.8, respectively. As shown in Fig. 16, if the selected value of $\tau$ is smaller than $\tau_d$, i.e. the penetration length is smaller than the length of the buffer, we observe that the total and time-averaged RMS errors become smaller as $\tau$ becomes larger. On the other hand, if we choose $\tau$ bigger than $\tau_d$, the error from the local boundary can penetrate the fine-resolution sub-domains and this would lead to the increase of RMS error. In the case of Navier–Stokes equation, we have an extra length scale associated with convection, namely

$$l_{p,a} \sim V \cdot \tau \Delta t.$$  \hspace{1cm} (16)

Thus, the allowable non-interaction timestep number $\tau_v$ for the convection with the length of the buffer $l_b$ is

$$\tau_v = \frac{l_b}{V \Delta t},$$  \hspace{1cm} (17)

where $V$ is the maximum streamwise velocity. Finally, the allowable non-interaction timestep number, $\tau_d$ is the minimum between these two different non-interaction timestep numbers as follows:
\[ \tau_0 = \min\{\tau_d, \tau_v\}. \]  

In this simulation, \( \tau_0 \) for a 25% buffer is around 15. However, as shown in Fig. 14, the RMS error increases linearly as \( \tau \) increases. This result shows that the RMS error in the Navier–Stokes equation appears to be independent of the allowable non-interaction timestep number \( \tau_0 \). The reason is that it is hard to find optimal \( \tau_0 \) in a nonlinear convection–diffusion equation. In order to investigate further, we perform a similar simulation of the same problem but with different size of the buffer, 37.5%. The results for this case follow the same trend as before, i.e., the smallest \( \tau \) is the best with respect to RMS error.

5. Summary

We developed a new algorithmic framework and tested simulations of two benchmark problems, namely the heat equation and flow in a lid-driven cavity in two dimensions, and under different resolutions of auxiliary data. We obtained important first insights via a parametric study by varying: 1) type of correlation kernel, 2) size of buffer, 3) accuracy of auxiliary data, and 4) non-interaction timestep number, \( \tau \). We summarize here the main findings of our study:

- **Kernel**: The Matérn kernel is found to be the best kernel with respect to RMS error and stability in both problems.
- **Buffer**: A bigger buffer can guarantee a smaller RMS error in both problems because the error at the local boundary can be diffused in a buffer region. Moreover, as the auxiliary data is inaccurate or auxiliary data may not be available, the size of the buffer enhances the robustness of the method.
- **Auxiliary data**: Higher resolution auxiliary data lead to a smaller RMS error in both problems because of increasing accuracy of results by information fusion. The accuracy of auxiliary data is found to be the most important parameter to reduce the RMS error effectively.
- **Non-interaction timestep number (\( \tau \))**: In the heat equation (only diffusion), near the allowable \( \tau_0 \), calculated by the estimation of a penetration length for a diffusion, we can guarantee the smallest RMS error. However, in the Navier–Stokes equations (combined diffusion and convection), the smaller \( \tau \) (update boundary values more frequently) gives the smallest RMS error.

From the fact that the auxiliary data can be of any fidelity, scale, or model, this framework can be extended to enable multifidelity and multiscale parallel simulations in a resilient way. In ongoing work, we have employed the Monte Carlo method for the heat equation, and dissipative particle dynamics (DPD) for the Navier–Stokes equations to obtain auxiliary data. In future work, it is also necessary to investigate the numerical analysis of a gappy simulation and show theoretically stability and convergence rates. we will also address the spatio-temporal gappy simulation with gaps both in time and space.

Acknowledgements

S. Lee and G.E. Karniadakis gratefully acknowledge the financial support of the Army Research Office (ARO W911NF14-10425), 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.

References
