Statistical Learning Tools for Information Fusion in Computational Fluid Dynamics

by

Seungjoon Lee

B.S., Seoul National University; Seoul, South Korea, 2009

M.S., Carnegie Mellon University; Pittsburgh, PA, 2011

Sc.M., Brown University; Providence, RI, 2013

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in The Division of Applied Mathematics at Brown University

PROVIDENCE, RHODE ISLAND

May 2017

This dissertation by Seungjoon Lee is accepted in its present form
by The Division of Applied Mathematics as satisfying the
dissertation requirement for the degree of Doctor of Philosophy.

Date_____ _____

George E. Karniadakis, Ph.D., Advisor

Date_____ _____

Ioannis G. Kevrekidis, Ph.D., Co-advisor

Recommended to the Graduate Council

Date_____ _____

Martin Maxey, Ph.D., Reader

Date_____ _____

Paris Perdikaris, Ph.D., Reader

Approved by the Graduate Council

Date_____ _____

Peter Weber, Dean of the Graduate School

Vitae

Seungjoon Lee was born on March 16, 1983 in Seoul, Republic of Korea (South). After completed his bachelor degree in mechanical and aerospace engineering at Seoul National University, he continued his graduate study in mechanical engineering at Carnegie Mellon University.

He started to Ph.D. study in applied mathematics at Brown University under guidance of Dr. George Em Karniadakis and Dr. Ioannis G. Kevrekidis. As a candidate of Ph.D., he received the University Fellowship (2012) and the certification of "Extreme Scale Computing Training Program" from Argonne National Laboratory (2014). His works have been supported by AFOSR, ARO, and DARPA. His projects have been focused on combination of the traditional framework (computational fluid dynamics) and the state-the-art framework (the statistical learning techniques). The publications of his projects are listed below.

Publications

1. **Seungjoon Lee**, Ioannis G. Kevrekidis, George E. Karniadakis, "Manifold-driven nonlinear information fusion for bifurcation theory", *in preparation* (2017)

2. **Seungjoon Lee**, Ioannis G. Kevrekidis, George E. Karniadakis, "A resilient and efficient CFD framework: statistical learning tools for multi-fidelity and heterogeneous information fusion", *revision in Journal of Computational Physics*

(2017)

3. **Seungjoon Lee**, Ioannis G. Kevrekidis, George E. Karniadakis, "A general CFD framework for resilient multiscale simulations based on multi-resolution information fusion", *under review in Journal of Computational Physics* (2016)

4. **Seungjoon Lee**, Ioannis G. Kevrekidis, George E. Karniadakis, "Resilient Algorithms for Reconstructing and Simulating Gappy Flow Fields in CFD", Fluid Dynamic Research **47** (2015) 051402, **Invited paper**

Preface and Acknowledgements

It has been a 10-year academic journey. It started from the undergraduate thesis about optimization techniques. During master degree, I combined optimization techniques (including Kriging) with computational fluid dynamics together. Finally, I integrated the statistical learning, optimization, domain decomposition, and computational fluid dynamics for exascale simulations in this dissertation.

In this long but beautiful journey, I would like to express my sincere gratitude to my advisor Dr. George Em Karniadakis. He has always inspired me with not only his great academic advices but also his passionate attitude for the research. Also, he has given to me unconditional supports for continuing my research and taught me how to be a good researcher. I would also like to sincere thank my co-advisor Dr. Yannis G. Kevrekidis. He is always the best motivator and mentor in my research. Without his brilliant intuition and idea, my dissertation could not be filled with anything new. I would also like to thank my dissertation readers, Dr. Martin Maxey and Dr. Paris Perdikaris for their careful reading of the thesis manuscript. Also, their valuable advices during my academic life made me an independent researcher. Also, my special gratitude goes to my previous advisor, Dr. Yoon Young Kim. After met him, I could take my first step of the research journey. Also, I would like to thank Dr. Donghyun You. He taught me all fundamental way for academic research and his passionate advice let me continue advanced study. I would like to thank all members of the *Crunch* group: specially, Dr. Minseok Choi, Dr. Changho Kim, and

Dr. Heyrim Cho. I always remember beautiful discussions about research, study, and life with them. In providence life, another special thanks go to my mentors, Keunhan Park, Youngsub Lee, Ill Ryu, and Myunghoon Roh, who gave and shared the most important advice in my life.

Finally, and the most importantly, I would like to thank my wife Jungmin Lee. Without her endless love and limitless support, I could not start and finish this journey. For me, she is a wise counselor, a strong motivator, a lovely wife, and a perfect mother. I also thank my princesse Jordan Lee and the upcoming princess, who are the greatest gift in my life from God. Also, I would like to the deepest thank my family and family in-law for giving endless support, love, and trust. Under Jehovah-jireh, I could finish my academic journey and I will take the first step for my second journey with God.

Abstract of "Statistical Learning Tools for Information Fusion in Computational Fluid Dynamics" by Seungjoon Lee, Ph.D., Brown University, May 2017

For more than a decade, remarkable scientific progress in computational fluid dynamics (CFD) has been achieved via powerful collection of tools for exascale simulations, data-driven approaches, and machine-learning (or statistical learning) techniques that enable efficient simulations of multiscale and multiphysics problems.

With the prospect of exascale simulations in the next decade, it is clear that new flexible tools and specialized algorithms are required to take advantage of such unprecedented computing environment. For example, in the *DOE ASCAC subcommittee report* [68] it is stated "new algorithms will need to be designed to optimize not only for floating-point performance and accuracy, but also to minimize associated data movement, power, and energy cost". Furthermore, robust and fault-resilient algorithms are increasingly attracting attention in exascale simulations against expected and repeated software or hardware error. Hence, new algorithms for exascale simulations inherently require the seamless integration of robustness, resilience, correctness, and efficiency.

Computational fluid dynamics has been developed following different pathways to address for complex flow physics problems. Many researches have focused on building up their own model to describe and address multi-resolution and multiphysics applications. Based on the recent introduction of data-driven algorithms via machine-learning techniques in computer simulations [26], computational fluid dynamics must rely also on data-driven approaches as much as on the anticipated improvements in computer hardware. In data science, data from various heterogeneous sources can be used effectively to accelerate simulations via multiple fidelity information fusion without additional complex models, equations, and extra state

variables. This could result in establishing a new paradigm of multifidelity simulations in computational fluid dynamics.

In this thesis, we address these new requirements for new algorithms in exascale simulations and introduce a novel framework including fault-resilient, robust, and efficient algorithms via multiple fidelity information fusion realized via statistical learning tools. This achievement addresses the new capability that statistical learning techniques can bring to traditional scientific computing algorithms.

This thesis proposes two possible directions of a next generation of computational frameworks for exascale simulations. The first direction addresses *resilience* via information fusion with auxiliary data. In exascale simulations, 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 based on multi-level Gaussian process regression, on-the-fly. The second direction addresses *efficiency* via adaptive projective time integration. The aforementioned auxiliary data provide additional information about dynamics-informed timestepping via Diffusion maps, which leads to significant acceleration of CFD simulations.

This thesis also presents and demonstrates methods of a robust nonlinear information fusion with multiple fidelity, multi-scale, parameterized data via *manifold-driven* regression algorithms. Finally, all multiple fidelity data can be integrated without complex models or equations.

Put simply, ensuring resilience, efficiency will be required for all new algorithm for simulations at the exascale and beyond. Finally, this thesis sets the foundations of a new class of algorithms that will combine traditional CFD with cutting-edge machine

learning techniques. Moreover, it integrates physics, mathematics and computer science for exascale simulations.

Contents

List of Tables

List of Figures

Chapter One

Introduction

This thesis proposes two directions for a next generation of exascale simulations. The first direction is to setup a fault-resilient and efficient algorithmic framework against spatio-temporal gaps in computational fluid dynamics. It starts from reconstructing missing fields with an auxiliary simulation, which is relatively costless with a coarse resolution. It can be extended to an efficient framework, which can reduce computational redundancy by dynamics-informed projective time integration with Diffusion maps.

The second direction is focused on establishing a robust information fusion technique between multiple resolution, fidelity, scale, and even heterogeneous data. Specifically, we introduce a multi-level Gaussian process regression for multiple resolution, fidelity, and heterogenous data. Also, robust information fusion techniques, different manifold-driven Gaussian process regressions are introduced for nonlinearly correlated data. Finally, all multiple fidelity data can be integrated without complex models or equations and this results in enhancing both accuracy and efficiency.

Through several benchmark problems in computational fluid dynamics, we demonstrate all aforementioned capabilities for exascale simulations. Next, we provide a brief summary of each chapter and organization of this thesis.

In chapter 2, we introduce three empirical approaches to reconstruct missing flow fields for three different fault scenarios – (1) only field variables at the current time step are missing, (2) current and previous field variables are missing, and (3) current and previous field variables are missing but in addition neighboring data are contaminated by *silent* errors. The first approach to reconstruct gappy fields is "projective time integration" with previous temporal snapshots and it is found to be the best method for the scenario (1). The second approach is "CoKriging" with neighboring data at same time step and it is found to be the best method for the

large time gaps in the scenario (2). The third approach is "Resimulation", which resimulate missing flow fields with *estimated* boundary conditions. This is the best method for the worst scenario (3), which the other two approaches fail.

In chapter 3, we introduce a novel algorithmic framework against gaps in space, called "gappy simulation". Through two benchmark problems, we first demonstrate a capability of information fusion with multiple resolution auxiliary data. Also, we investigate important (trade–off) parameters to affect accuracy and efficiency of this framework. The first parameter is a type of covariance kernel in the multi-level Gaussian process regression and it is found that "Matérn" kernel is the best to construct a covariance kernel correctly in all benchmark problems. The second parameter is a buffer size, which can protect our original subdomains from uncertainties of local boundary condition. A large buffer size guarantees a smaller RMS error but we need to solve bigger subdomains and this results in losing efficiency. The third parameter is quality (accuracy) of auxiliary data. The higher accurate auxiliary data provides the smaller RMS error due to impose more accurate boundary conditions in each subdomain. The forth parameter is a non-interaction timestep number (τ). The small τ guarantees the small RMS error because of frequent updating of local boundary conditions. However, all computer node should communicate each other more frequently and this leads to additional computational cost.

In chapter 4, we extend the previous framework to efficient algorithm, called "patch simulation" by dynamics-informed projective time integration via nonlinear dimensionality reduction techniques, *Diffusion maps*. We simulate two benchmark problems as same previous chapter and demonstrate the new capability of information fusion with heterogeneous and multi-fidelity data, which come from a stochastic, particle-based, more "microscopic" simulation, e.g., a Monte Carlo random walk for the heat equation and a dissipative particle dynamics (DPD) model for the Navier-

Stokes equations. We investigate that the auxiliary simulation can support the main simulation in two ways. The first way is to reconstruct (missing) local boundary condition of each subdomain. The second way is to provide the appropriate projection time (jump size) and accelerate the main simulation. Finally, we show that the patch simulation guarantees both efficiency and accuracy against spatio-temporal gaps.

In chapter 5, we introduce an application of the efficient algorithm via dynamics-informed projective time integration, which described previous chapter. We demonstrate a multi-rate problem in a complex biological system. In order to capture all dynamics of each species correctly, we need to employ the smallest timestep size and this leads to computational redundancy. We demonstrate that the dynamics-informed projective time integration with each coarse simulator guarantees the smaller computational cost compared to the original simulation. Moreover, we found that a physics-related tuned parameter, α, can estimate appropriate projection time for each species from the given field variable. In this application, a dynamics-informed projection time from velocity fields (given) with $\alpha = 1/\mathrm{Pr}$ are able to estimate appropriate projection time for each species and this results in reducing huge computational cost. Hence, we demonstrate the capability of (dynamics and physics informed) projective time integration without additional coarse simulations.

In chapter 6, we introduce two methods to extend a robust nonlinear information fusion techniques [79] for nonlinearly correlated data. In order to show capability of introduced approaches, we simulate Van der Pol oscillators, which have a nonlinear damping with a intensity parameter μ. Two introduced methods are focus on making a smooth and one-to-one mappable manifold, where Gaussian process is available. The first method is "iterative Gaussian process", which modify the low fidelity data to make the original manifold smooth (or flatted). It is available in only the similar

limit cycle case, which frequency and shape of responses of two oscillators are similar. The second method, more robust, is the Gaussian process via Diffusion maps. Diffusion maps provide the smooth and one-to-one mappable manifold constructed by the appropriate pair of diffusion coordinates. It guarantees robustness for any two oscillators. Finally, we show the capability of robust nonlinear information fusion for nonlinearly correlated data.

In chapter 7, we conclude and summarize this thesis. Also, we introduce three directions of future works. The first direction is to investigate numerical and sensitivity analysis for supporting our framework with a concrete mathematical background. The second direction is an extension of robust nonlinear information fusion techniques for more practical and realistic problems. The last direction is to setup a new type of efficient framework by auto-adaptive and asynchronous algorithm for demonstrating a complex biological and chemical system effectively.

In appendices, we introduce details of introduced statistical learning tools. In the appendix A, detailed formula for a classical Gaussian process regression (Kriging) and a multi-level Gaussian process regression (CoKriging) are included. In the appendix B, detailed formula and mathematical descriptions for Diffusion maps are included.

Chapter Two

Resilient Algorithms for Reconstructing and Simulating Gappy Flow Fields in CFD

2.1 Introduction

In large-scale simulations of computational fluid dynamics (CFD) on modern massively parallel computers involving hundreds of thousands of processors, it is very possible, and in fact it has already been observed that random hardware or software faults may render the computation useless [40, 107, 72, 94]. While research is ongoing to robustify both hardware and systems software [39, 41], on the algorithmic side it is also important to formulate a new class of resilient algorithms that perform the simulation accurately irrespective of such faults. Hence, in order to make progress on developing new methods or reformulating existing methods but in this new context, we can hypothesize a few different scenarios and apply them to CFD benchmark problems in order to evaluate the effectiveness of such approaches. First, we will assume that being able to reproduce accurately but also efficiently the data from a failed processor is, in principle, equivalent to omitting the processor from the computation. Hence, fault tolerance implies that we have to be able to compute with reasonable accuracy on "partial" spatio-temporal domains [29, 1, 2, 77]. Based on this general assumption, we then proceed to examine three progressively more complex cases: (1) only grid values at the current time step are missing, (2) current and previous grid values are missing, and (3) current and previous values are missing but in addition the neighboring data are contaminated by *silent* errors, detected after some delays such as L1 cache, or double bit flips error [4]. In order to reconstruct missing flow field data on-the-fly, we can employ temporal or spatial estimation methods and other known data recovery techniques.

For *spatial estimation* of gappy flow fields in fluid dynamics, the gappy proper orthogonal decomposition (POD) method was introduced in [106, 110], 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 [103, 43, 65]. While the Kriging method uses one sample set, the coKriging method uses two or more sample sets, e.g., a coarse and a refined set corresponding to different fidelity [46]. For *temporal estimation* of missing data, the projective integration was introduced in [38, 89] to use previous snapshots and POD-assisted bases. Based on the projective integration the equation-free/Galerkin-free method was introduced in [99] for solving the incompressible Navier-Stokes equations.

In the current work, we employ two estimation methods; see 2.1: the projective integration and the coKriging method for the three aforementioned scenarios. We also introduce a new method, the "resimulation" method, which resolves via simulation a missing region only with the proper initial and boundary conditions. Finally, we compare these three methods for two classical problems of fluid dynamics, namely the lid-driven cavity flow and flow past a circular cylinder.

This chapter is organized as follows: In section 2, we introduce the projective integration, the coKriging, and the resimulation method and present the mathematical algorithms for each. In section 3, we present the computational domains and simulation set up. In section 4, we present results of the flow problems for the three aforementioned scenarios and compare them in terms of accuracy. In section 5, we summarize our results and discuss open issues in estimation theory for further developments of fault-resilient methods.

2.2 Methodology

Incompressible flow is described by the divergence-free Navier-Stokes equations:

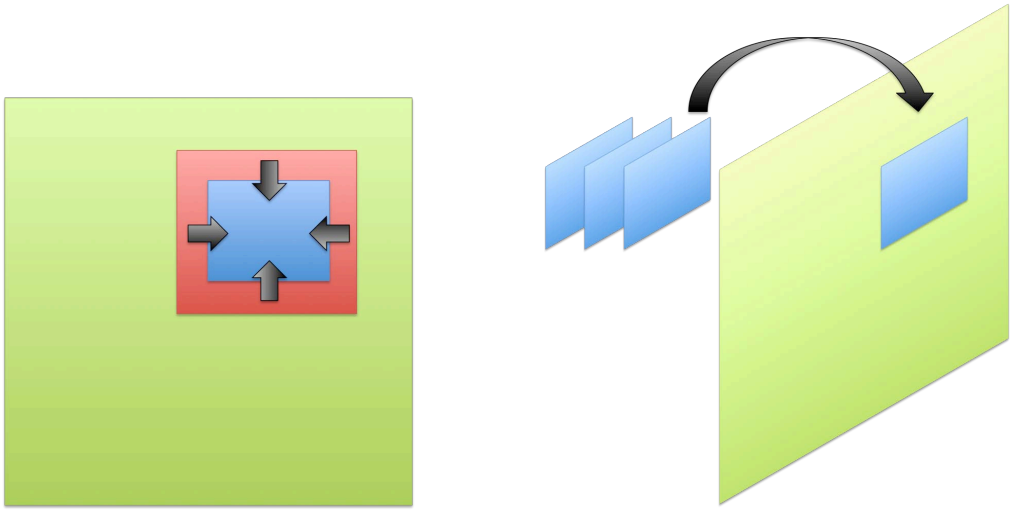
$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} + f \tag{2.1}$$

$$\nabla \cdot \mathbf{v} = 0 \tag{2.2}$$

where $\mathbf{v}$ is the velocity vector, p is pressure, and ν is the kinematic viscosity of the fluid. Here we employ the spectral element method to solve these equations [50]. In order to produce a useful estimate for the missing part, we have to use both temporal and spatial data and corresponding estimators. In this section, we introduce the projective integration for *temporal estimation* and the coKriging for *spatial estimation*. We will also formulate a new method, the *resimulation* method, with different boundary conditions.

2.2.1 Temporal estimation: Projective integration

If numerical solutions are sufficiently smooth in time, the temporal estimation based on previous saved data can give a highly accurate result on a missing part of the solution. To accomplish this, the best and simplest way is to save flow field data every time step and employ good extrapolation schemes to estimate the missing data. Also, if the time step (Δt) of a simulation is small enough, an alternative way is to use just previous flow field data as current flow field data, in analogy zero- and first-order continuation schemes. However, these ways are inefficient due to big memory issues and additional computational cost in large-scale simulations. In order



(a) Spatial estimation (b) Temporal estimation

Figure 2.1: Temporal and Spatial estimations: In (a), the green box denotes the entire domain, the blue box is the missing part, and the red box represents the region of sampling. In (b), the three blue "slices" represent previously saved data to be used for estimation of the current fields in the missing part.

to avoid this drawback, an equation-free/Galerkin-free projective integration can be employed instead [99]. The projective integration is based on the proper orthogonal decomposition (POD) for a dimension reduction [100, 3]. The saved flow field data can be represented by the corresponding temporal modes $a(t)$ and POD-bases $\phi_k(x)$ as follows:

$$u\left(t,x\right)=\sum_{k}a_k(t)\phi_k(x). \tag{2.3}$$

The basic algorithm of the projective integration consists of three stages: the restriction, estimation, and lifting. In the restriction part we calculate the temporal modes $\mathbf{a}(t_i)$ of the POD-assisted bases in equation (4). The number of snapshots (n) we employ determines the number of terms of the POD expansion.

$$\mathbf{a}(t_i)=\mathcal{P}u(t_i,x)=\left\{\left(u(t_i,x),\phi_k(x)\right),\forall k\right\}\qquad i=1,\ldots,n. \tag{2.4}$$

After the restriction, using the computed POD-assisted temporal modes, the required missing coefficients $\mathbf{a}(t^*)$ of the corresponding POD-assisted basis are estimated by an extrapolation scheme ("projected" forward in time) based on the Piecewise Cubic Hermite Interpolating Polynomial in equation (2.5), see the reference [35]. This extrapolation method is found to agree well with current boundary conditions in scenario 3 in section 4. Hence, we have:

$$\mathbf{a}(t^*) = \left[\frac{d_i + d_{i+1} - 2\Delta_i}{(t_{i+1} - t_i)^2}\right](t^* - t_i)^3 + \left[\frac{-2d_i - d_{i+1} + 3\Delta_i}{t_{i+1} - t_i}\right](t^* - t_i)^2 + d_i(t^* - t_i) + a(t_i), \tag{2.5}$$

where $\Delta_i = (\mathbf{a}(t_{i+1}) - \mathbf{a}(t_i))/(t_{i+1} - t_i)$ and $d_i = d\mathbf{a}(t_i)/dt$.

Finally, the lifting step is to find the current flow field data by the estimated temporal modes with the low-dimensional POD-assisted bases by the following equation:

$$u\,(t^*, x) = \mathcal{L}\mathbf{a}(t^*) = \sum_k a_k(t^*)\phi_k(x). \tag{2.6}$$

If we use only temporal data to estimate the current flow fields, the accuracy of this method depends on how many snapshots (number of terms of POD expansion and Taylor expansion for calculating the derivative, d_i), which extrapolation scheme (local truncated error) we use, and how big the time gaps are. This chapter used three snapshots (three POD modes) for each simulation. For the efficiency, the POD modes are calculated in only missing regions.

2.2.2 Spatial estimation: coKriging

While for the temporal estimation we use the previous flow field data and smoothness in time, in the spatial estimation we need to use geometrically neighboring data points at the current time to exploit smoothness in space. There are many ways to estimate gappy flow field data by the neighboring data. In this chapter, a "multi-fidelity coKriging interpolation method", the unbiased linear interpolation, is introduced for estimating the missing part because of higher accuracy compared with the simple Kriging for the same size of neighboring data as shown in 2.3 [46]. A general multi-fidelity coKriging method uses two sample sets: a coarse sample set (small number of points) corresponding to high fidelity and a refined sample set (large number of points) corresponding to low fidelity. In this chapter we choose the same fidelity model for both the coarse and refined sample sets; the coarse sample set can be any subset of the refined sample set, see 2.2. The basic idea of this method is that the target value $\hat{y}(\mathbf{x})$ at the target point $\mathbf{x}$ can be found using a linear interpolation by the two sample sets. The first equation for the coKriging interpolation method is as follows:

$$\hat{y}(\mathbf{x}) = \lambda_{\mathbf{1}}^{\mathbf{T}}\mathbf{y_1} + \lambda_{\mathbf{2}}^{\mathbf{T}}\mathbf{y_2}, \tag{2.7}$$

where $\mathbf{y_1} \in \mathbb{R}^n$ is a neighboring field data vector in the refined sample set and $\mathbf{y_2} \in \mathbb{R}^m$ is in the coarse sample set. In this chapter, the coarse set is chosen by a subset of the refined set. Here we solve a simple optimization problem in minimizing the mean squared error (MSE) of this linear combination.

$$\underset{\lambda\mathbf{1},\lambda\mathbf{2}}{\arg\min} E[(\hat{y}(\mathbf{x}) - y(\mathbf{x}))^2], \tag{2.8}$$

subject to the unbiased constraint

$$E[\hat{y}(\mathbf{x})] = E[y(\mathbf{x})] \quad \text{or} \quad \sum_{i=1}^{n} \lambda_{1_i} = 1 \text{ and } \sum_{i=1}^{m} \lambda_{2_i} = 0. \tag{2.9}$$

Then, we can solve the linear system with Lagrange multipliers μ_1 and μ_2 as follows:

$$\begin{bmatrix} \mathbf{C_{11}} & \mathbf{C_{12}} & \mathbf{1} & \mathbf{0} \\ \mathbf{C_{21}} & \mathbf{C_{22}} & \mathbf{0} & \mathbf{1} \\ \mathbf{1^T} & \mathbf{0^T} & 0 & 0 \\ \mathbf{0^T} & \mathbf{1^T} & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_{\mathbf{1}} \\ \lambda_{\mathbf{2}} \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} \mathbf{c_1}(\mathbf{x}) \\ \mathbf{c_2}(\mathbf{x}) \\ 1 \\ 0 \end{bmatrix}, \tag{2.10}$$

where $\mathbf{C}$ is a covariance matrix whose submatrix $\mathbf{C_{11}}$ is the covariance matrix between sample points in the refined set, $\mathbf{C_{22}}$ between sample points in the coarse set, and $\mathbf{C_{12}}$ between the refined and the coarse set. $\mathbf{c_1}(\mathbf{x})$ is the covariance vector between the target point ($\mathbf{x}$) and sample points in the refined set, and $\mathbf{c_2}(\mathbf{x})$ between the target point ($\mathbf{x}$) and sample points in the coarse set. For the correlation kernel $\mathbf{R}$ for modeling covariance in the sample set S, we employ a spherical correlation model since the flow fields are smooth near the missing part as follows:

$$\mathbf{R}(\theta, x_i, x_j) = A\{1.5\theta(|x_i - x_j|) - 0.5(\theta(|x_i - x_j|))^3\}, \tag{2.11}$$

where x_i and x_j are points in S, θ is the correlation length and A is a scalar parameter calculated by the least-squared method.

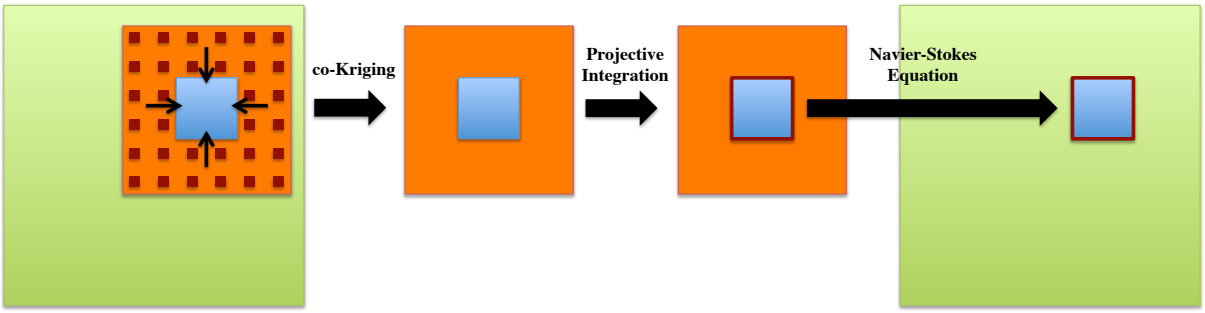


Figure 2.2: Resimulation with estimated boundary condition: First, we estimate the initial condition for the missing part (blue) with two sample sets: refined (orange) and coarse (red). Subsequently, we use the projective integration to update the boundary using the refined sample set. Finally, we solve the Navier-Stokes equations in the missing part only.

2.2.3 Resimulation

The main idea of the resimulation method is to solve the Navier-Stokes equations again on the missing part only. Generally, we prescribe initial and boundary condition by function of space and time at the beginning of the space-time interval – that is, the boundary conditions are not altered by communication in an unit timestep between data dumps. If we have exact initial and boundary conditions for the missing part, we can obtain the highest accurate solution compared to any method we can choose. For example, in steady-state, since a boundary condition of the missing part is independent of time, the resimulation method can give the exact solution [50]. However, if we do not have the correct information about initial and boundary conditions, then it is not so clear how to estimate these data in order to assign appropriate initial and boundary conditions. For initial conditions, if we do not have any previous data (scenarios 2 and 3 in section 4), the coKriging method can be employed for estimating initial conditions of the missing part. For the boundary conditions, there are two ways: the first is to use the latest saved data as a Dirichlet boundary condition. If the solution of the flow problem is smooth enough and changes of flow fields are relatively small during time gaps, then this boundary condition works for resimulation. However, if the solution is not smooth or the difference between the

latest saved and current flow field data at the boundary is not sufficiently small, then the error at the boundary propagates through the entire missing domain, resulting in bigger errors than other estimation methods.

In order to avoid error propagation from assigning an incorrect boundary condition, a boundary estimation technique is introduced in this chapter as shown in 2.2. The basic idea of this method is that we can extrapolate a boundary condition from previous snapshots by the projective integration, which is already explained in section 2.2. In order to reduce the computational cost, the estimated boundary is extracted not from the entire domain but only from the domain of sample sets. The process of calculating the estimated boundary is as follows:

- Assign the size of a sample set by choosing a "*spacing*" parameter.
- Employ the projective integration method and estimate the current flow field data in the sample set.
- Extract the flow field data at the boundary of missing region (boundary condition is constant in time).

In section 4, the "estimated" boundary is found to approximate the boundary condition reasonably well. Hence, we can impose this as a Dirichlet boundary condition for the missing region. This boundary condition is useful during sufficiently short time intervals.

An important objective in any resilient method is to synchronize the simulation in the gappy domain with the rest of the simulation as soon as possible. Hence, we can resimulate the missing part with a coarse grid by interpolation. If the solutions are smooth, a coarse time-step can also be used for the resimulation. In this case,

the error comes from the interpolation scheme, spatial and temporal discretization, and consistency error with respect to initial and boundary conditions.

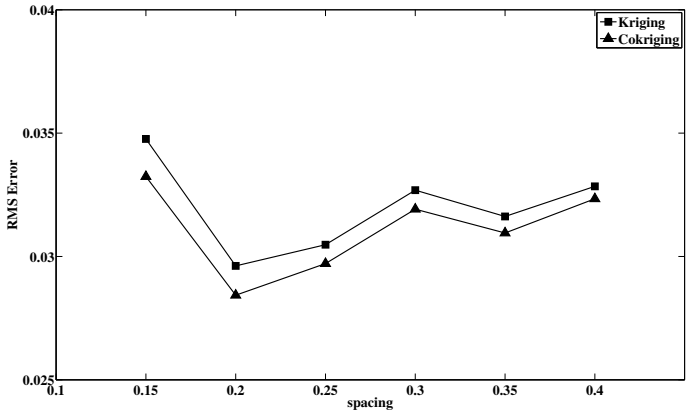
2.2.4 Key parameters

This resilient method has some key parameters that affect accuracy and efficiency. The first key parameter is the size of the missing part; as it grows, the accuracy decreases due to the increased estimation error. Since the spatial estimation uses an interpolation method with neighboring data, points on the middle of the missing region have no accurate neighboring data. Hence, the accuracy of the spatial estimation is dramatically decreased as the missing part is growing.

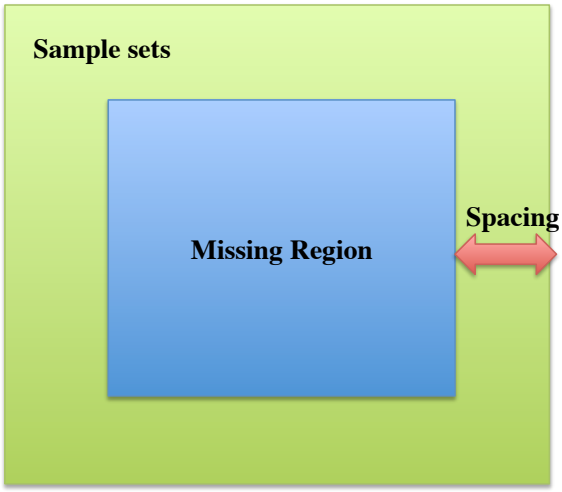
The second key parameter is the size of time gaps (ΔT_g) between current and previous saved data. If the time gaps are small and we have available previous data (scenario 1), then the projective integration or resimulation method can give good estimation results compared to the coKriging method. However, when the time gaps are big, the projective integration cannot give us a good accuracy and also the consistency error of the resimulation method is growing by the propagation of the incorrect boundary conditions. On the other hand, the coKriging method is less dependent on time gaps.

The last key parameter is the "*spacing*" (Δs), which is a physical parameter for choosing the number of sample points, see 2.3. The formulation of choosing the sample set (S) for the missing region (M) is defined as follows:

$$S = \{\mathbf{x} \in \Omega : \min d(\mathbf{x}, \mathbf{y}) < \Delta s,\ \mathbf{y} \in M\}, \tag{2.12}$$



(a) RMS Error



(b) Spacing

Figure 2.3: In (a), the graph shows RMS error for values of different spacing with coKriging and Kriging in flow past a circular cylinder. In (b), the blue box is the missing region, the green box represents the sample set, where the size of sample set is changed by the "spacing" parameter.

where Ω represents the entire domain and Δs is the *spacing*.

If the spacing is too small, the spatial estimation is performed by only a few neighboring sample points. Consequently, the result cannot be very accurate as shown in 2.3. On the other hand, if the spacing is too big, then the computational cost is dramatically increased by calculating a big covariance matrix inversion in the coKriging method in equation (10). Furthermore, the accuracy is also slightly dropped by using uncorrelated data that are too far from the missing region. However, if we do not know the true solution we should use the local residuals in similar fashion as in adaptivity methods.

2.3 Simulation Configuration

In this chapter, two benchmark problems are chosen for comparisons of three different estimation methods in three different scenarios. The first simulation is a two-dimensional lid-driven cavity flow at Reynolds number $Re = 100$, which represents quasi-steady flow. The second simulation is a two-dimensional flow past a circular cylinder at Reynolds number $Re = 100$, which represents quasi-periodic flow. The numerical method is based on a high-order spectral/hp element method using the solver *NekTar* [50]. In order to readily impose the Dirichlet boundary condition, the missing part is chosen always as a rectangle box with a rectangular grid. In order to satisfy this condition, the smoothed profile method (SPM) is employed, see [70], to describe a rectangular structured grid by the use of an indicator function, which is a constraint of a force distribution associated with the flow boundary condition. Details for the simulation setup are presented below.

2.3.1 Lid-driven cavity flow at $Re = 100$

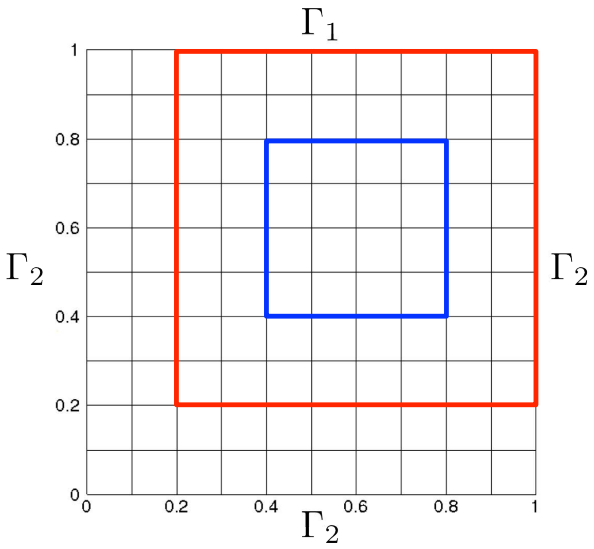
The computational domain is a structured rectangular mesh shown in 2.4(a). Normalized unidirectional flow is imposed on the top boundary (Γ_1) while on the other boundaries (Γ_2) we impose no-slip condition on the velocity. The domain has the same size of discretization in the x and the y directions defined by $dx = dy = 0.1$ with 3rd order Jacobi polynomial basis, for a total 961 nodes to solve for. The temporal discretization corresponds to $dt = 0.005$. Our simulation is scheduled to stop at two different times: t=2.0 and 2.5. At those times, we assume that we have three saved snapshots at t=1.40, 1.45 and 1.50. Hence, the time gaps (ΔT_g) are 0.5 and 1.0, respectively.

In order to investigate error propagation of three methods, the missing region is chosen as a rectangle box in the middle of cavity. The missing region (M_l) is defined as follows:

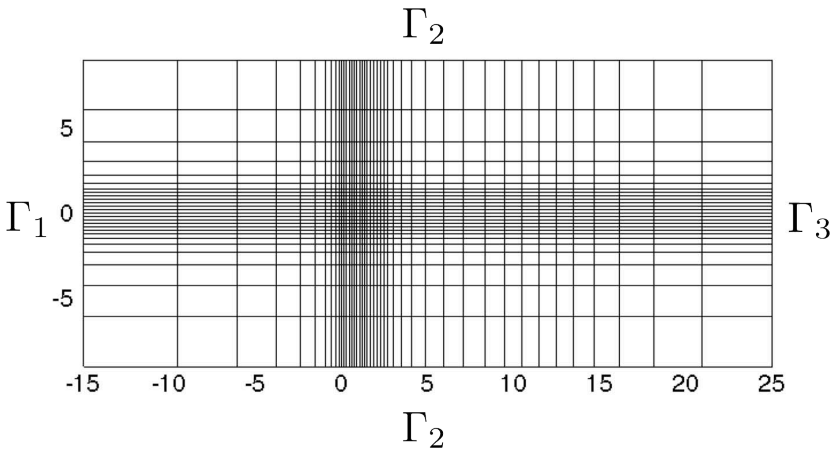
$$M_l = \{(x_1, x_2) \in \Omega : 0.4 \leq x_1 \leq 0.8 \text{ and } 0.4 \leq x_2 \leq 0.8 \}$$

where $\Omega = [0, 1] \times [0, 1]$ and the origin $(0, 0)$ is located in the Southwest corner of the cavity.

The *spacing* for the sample set corresponds to 0.2. The sample set has 433 points for the refined case and 144 points for the coarse case, while the missing region has 144 missing points.



(a) Lid-driven cavity flow



(b) Flow past a circular cylinder

Figure 2.4: Computational domains and spectral element meshes for the two flow problems considered. In (a), the blue line represents the missing region (M_l) and the red line represents the sample set (S). The coarse sample set is a subset of the refined set.

2.3.2 Flow past a cylinder at $Re = 100$

The computational domain consists of 1118 rectangular elements with Jacobi polynomial order $P = 3$; the domain is shown in 2.4(b). The inflow boundary conditions (Γ_1 and Γ_2) correspond to a normalized unidirectional flow ($U = 1$). On the outlet boundary (Γ_3), the zero Neumann boundary condition is applied. The surface of the circular cylinder with diameter $D = 1.0$ has a no-slip boundary condition imposed via the indicator function with a 0.03 interface thickness according to the smoothed profile method. Since the grid resolution near the circular cylinder is 0.15, the interface thickness can capture artificial boundary of the circular cylinder effectively. In order to increase the accuracy of the smoothed profile method, regions near the circular cylinder have a refined grid to capture the geometric boundary better.

We choose the missing region as a rectangular box in the near wake, i.e., the region of absolute instability that is responsible for sustaining the von Karman street, see [51]. The temporal discretization corresponds to $dt = 0.001$. Our simulation is scheduled to stop at two different times, 331.89 and $t = 332.09$, and we have three snapshots available at $t = 331.60$, 331.61, and 331.62. Hence, the time gaps (ΔT_g) correspond to 0.27 and 0.47, respectively. The shedding frequency corresponds to 0.167, i.e., a shedding period is 5.98 unit time. Hence, chosen time gaps are small enough compared to a shedding period. The missing region (M_c) is defined as follows:

$$M_c = \{(x_1, x_2) \in \Omega : 1.5 \leq x_1 \leq 2.5 \text{ and } -0.5 \leq x_2 \leq 0.5 \}$$

where $\Omega = [-15, 25] \times [-9, 9]$ and the origin $(0, 0)$ is located on the center of the circular cylinder.

The *spacing* for sampling corresponds to 0.25. The sample set has 289 points for

Table 2.1: Comparison of RMS error for three different methods in lid-driven cavity flow. ()* indicates Projection Integration

Velocity	Time gaps (ΔT_g)	Scenario	P.I.*	coKriging	Resimulation
streamwise	0.5	1	0.0044	0.0136	0.0075
		2	—	0.0136	0.0074
		3	—	—	0.0078
	1.0	1	0.0156	0.0150	0.0124
		2	—	0.0150	0.0122
		3	—	—	0.0158
crossflow	0.5	1	0.0007	0.0177	0.0060
		2	—	0.0177	0.0059
		3	—	—	0.0088
	1.0	1	0.0116	0.0192	0.0108
		2	—	0.0192	0.0106
		3	—	—	0.0105

the refined case and 97 points for the coarse case, while the missing region has 240 missing points.

2.4 Results for Three Fault Scenarios

We introduce three possible fault scenarios that may occur in massively parallel simulations. The Root-Mean-Squared (RMS) errors are presented in Tables 1-3 while contour plots of absolute errors are shown in Figures 5-14. Details about each scenario are presented in the following subsections.

2.4.1 Scenario 1

The first scenario assumes that at some point in time a gappy region arises, but that we have complete information up to that point. In this case, saved data is available to estimate the current missing data. Thus, we can employ all three methods for

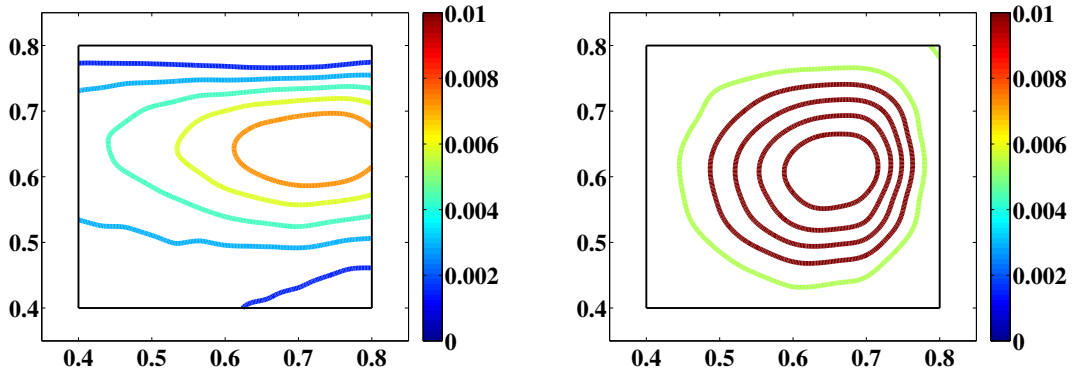
Table 2.2: Comparison of RMS error for three different methods in flow past a circular cylinder. $()^*$ indicates Projection Integration

Velocity	Time gaps (ΔT_g)	Scenario	P.I.*	coKriging	Resimulation
streamwise	0.27	1	0.0039	0.0219	0.0060
		2	—	0.0219	0.0172
		3	—	—	0.0175
	0.47	1	0.0193	0.0251	0.0144
		2	—	0.0251	0.0235
		3	—	—	0.0291
crossflow	0.27	1	0.0046	0.0178	0.0065
		2	—	0.0178	0.0168
		3	—	—	0.0189
	0.47	1	0.0231	0.0159	0.0149
		2	—	0.0159	0.0241
		3	—	—	0.0374

reconstructing the flow fields, namely projective integration, coKriging, and the resimulation method with correct initial conditions. In assigning a boundary condition for the resimulation method, we employ a Dirichlet boundary condition, which is extracted from the "current" neighboring flow subdomains.

Lid-driven cavity flow:

For $\Delta T_g = 0.5$, 2.1 shows that the projective integration method appears to be the best way to reconstruct gappy flow fields because there are no big changes in the flow field within that time interval; this is true for both the streamwise and crossflow velocity. In particular for the crossflow velocity, we observe that the projective integration method reduces the error greatly compared to other methods. Thus, the projective integration seems to be the most effective method for short time gaps in this simulation. By increasing the time gap to $\Delta T_g = 1.0$, however, we observe that the magnitude of error of the projective integration is increased substantially compared to $\Delta T_g = 0.5$. On the other hand, the changes in error for the coKriging



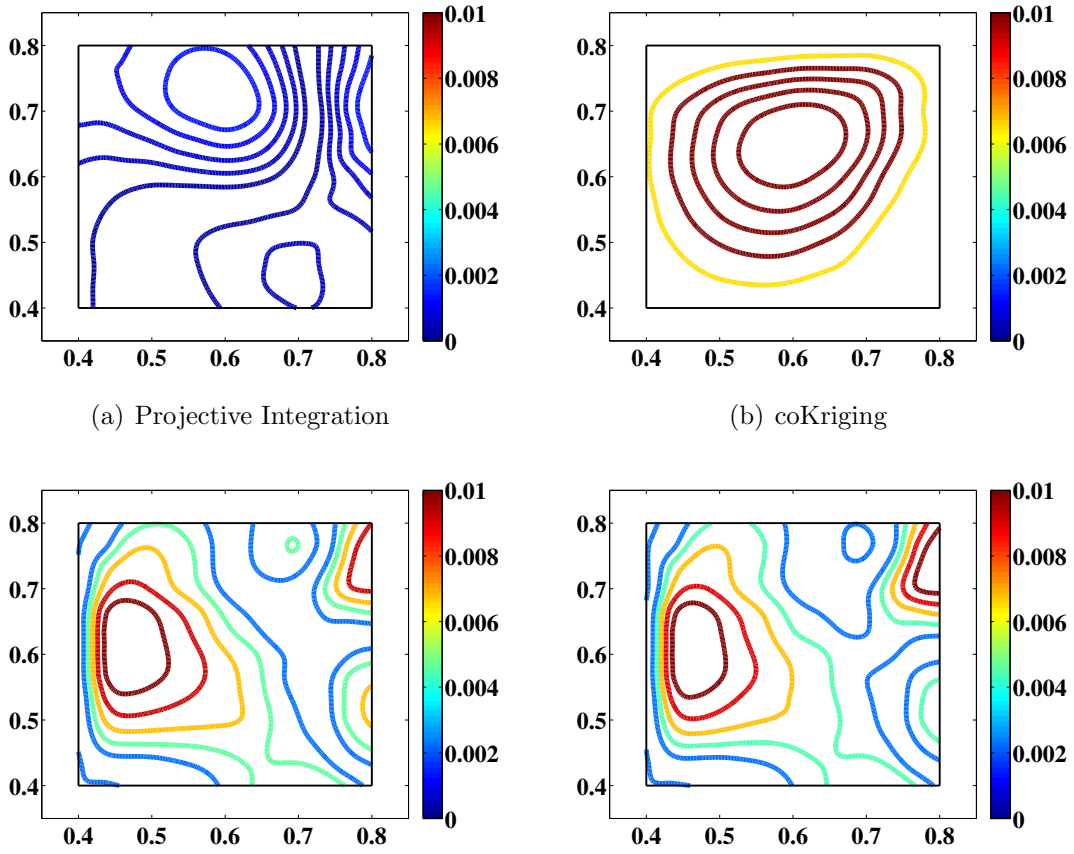
(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.5: Lid-driven cavity flow in scenario 1 and 2: Absolute error of streamwise velocity at $\Delta T_g = 0.5$.



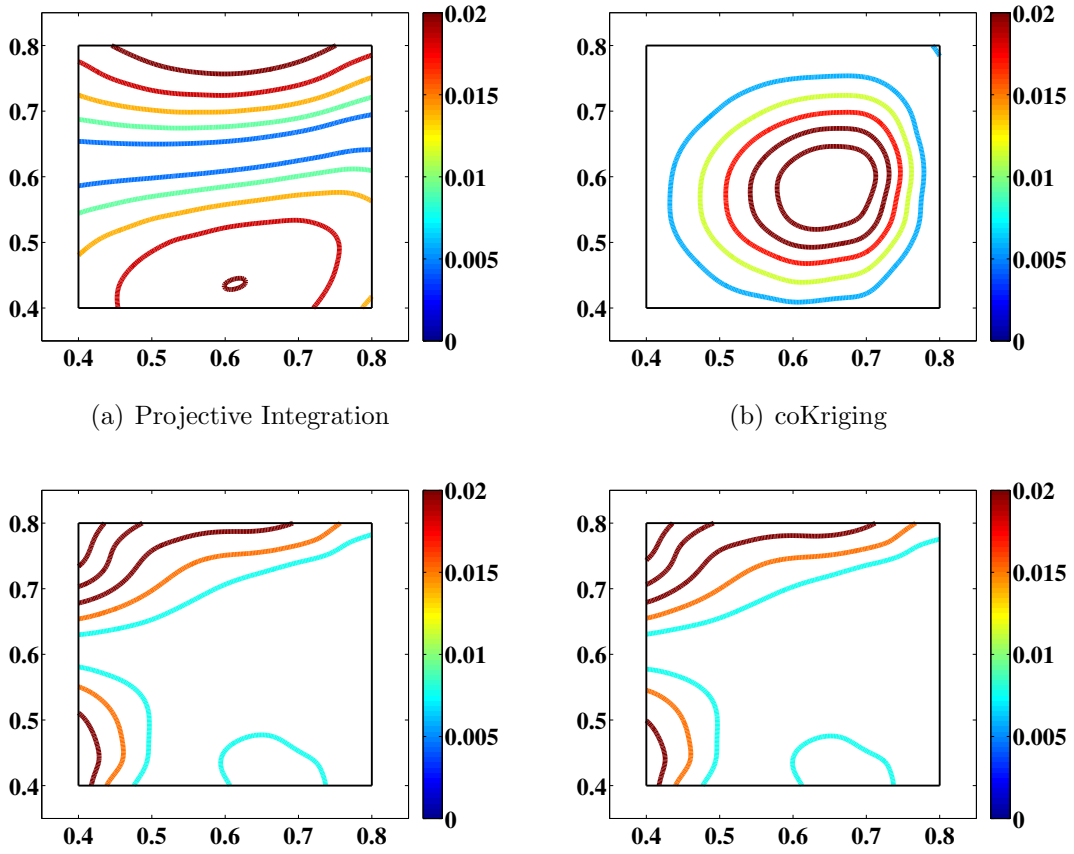
(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.6: Lid-driven cavity flow in scenario 1 and 2: Absolute error of crossflow velocity at $\Delta T_g = 0.5$.



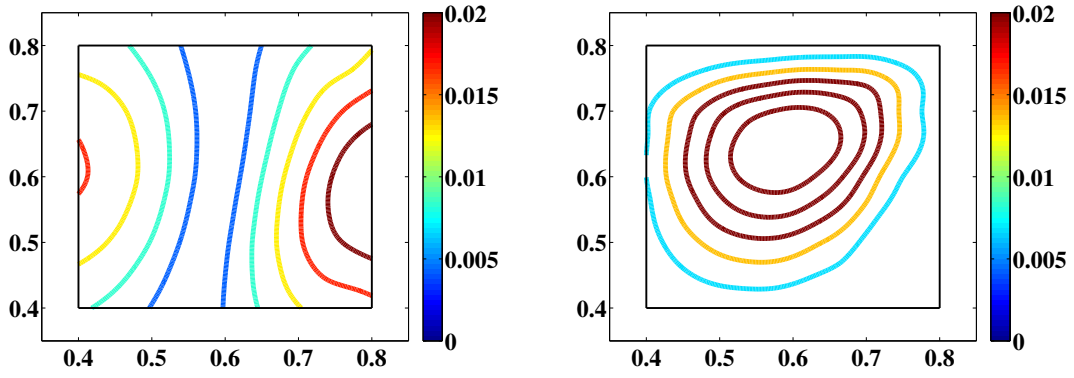
(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.7: Lid-driven cavity flow in scenario 1 and 2: Absolute error of streamwise velocity at $\Delta T_g = 1.0$.



(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

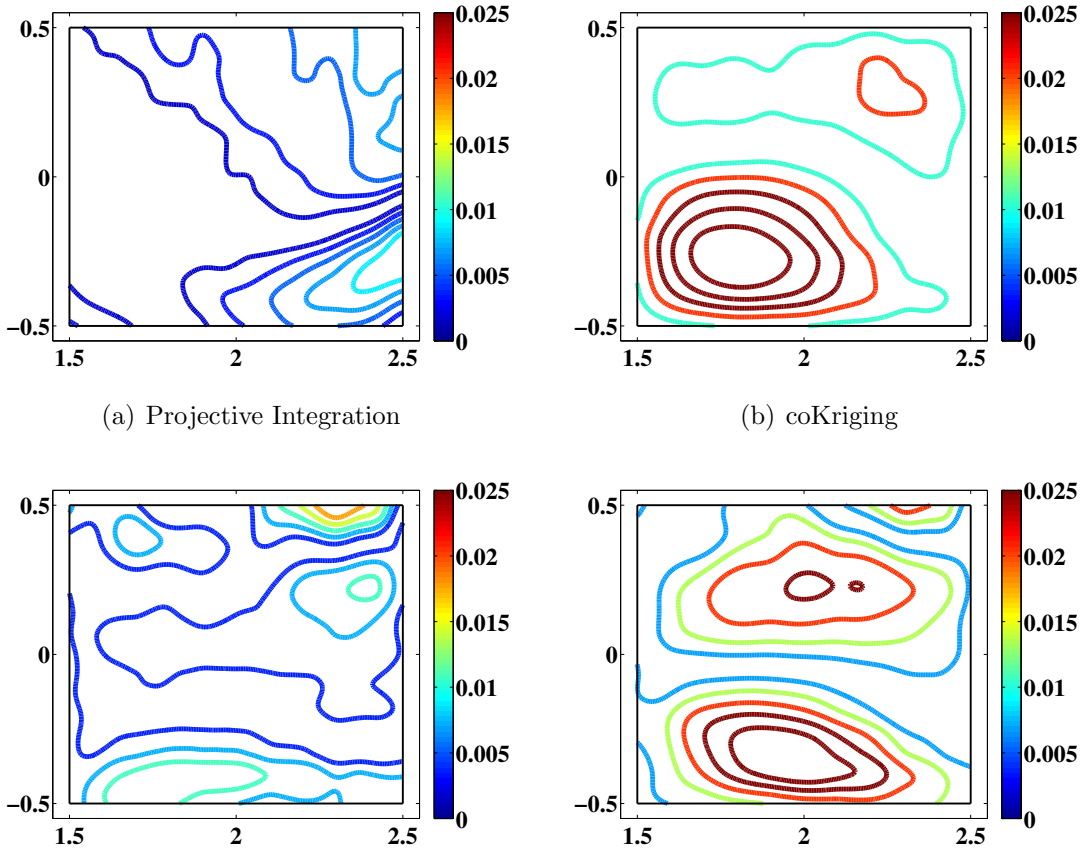
Figure 2.8: Lid-driven cavity flow in scenario 1 and 2: Absolute error of crossflow velocity at $\Delta T_g = 1.0$.

and resimulation method are small compared to the projective integration method. For this time gap, the resimulation method seems to be the most effective approach to recover the flow field in the gappy region. In both cases, since the accuracy of the coKriging method depends only on surrounding data, the magnitude of the error is not changed significantly for either time gaps. In general, the resimulation method is competitive to the projective integration and seems to be robust for different time gap sizes. With regard to error distribution, the maximum error in the coKriging method is located at the middle of the spatial domain whereas the maximum error of the resimulation method is located near the boundary due to the incorrect type of boundary condition, see 2.5 and 2.6.

Flow past a circular cylinder:

Since the gappy region (M_c) is located in the near-wake, relying on the neighboring data only cannot guarantee a strong correlation with the missing flow field data.

First, for $\Delta T_g = 0.27$, the results shown in 2.2 lead to the same conclusion as in the lid-driven cavity flow. Even though the flow is unsteady, the projective integration is the most effective approach to reconstruct the gappy flow fields because the "current" flow field is still similar to the saved one. In particular, we see in 2.9 that the velocity at the boundary does not deviate significantly from the saved values. Hence, the resimulation method can also estimate the current flow field well while the coKriging method seems to be the worst method due to the rapid spatial variation of velocity in this region. However, increasing the time gap to $\Delta T_g = 0.47$ seems to result in a big change in velocity and hence the projective integration method cannot be used effectively. Since saved or neighboring data are not available to estimate the current flow field accurately, all three methods show similar results but the best



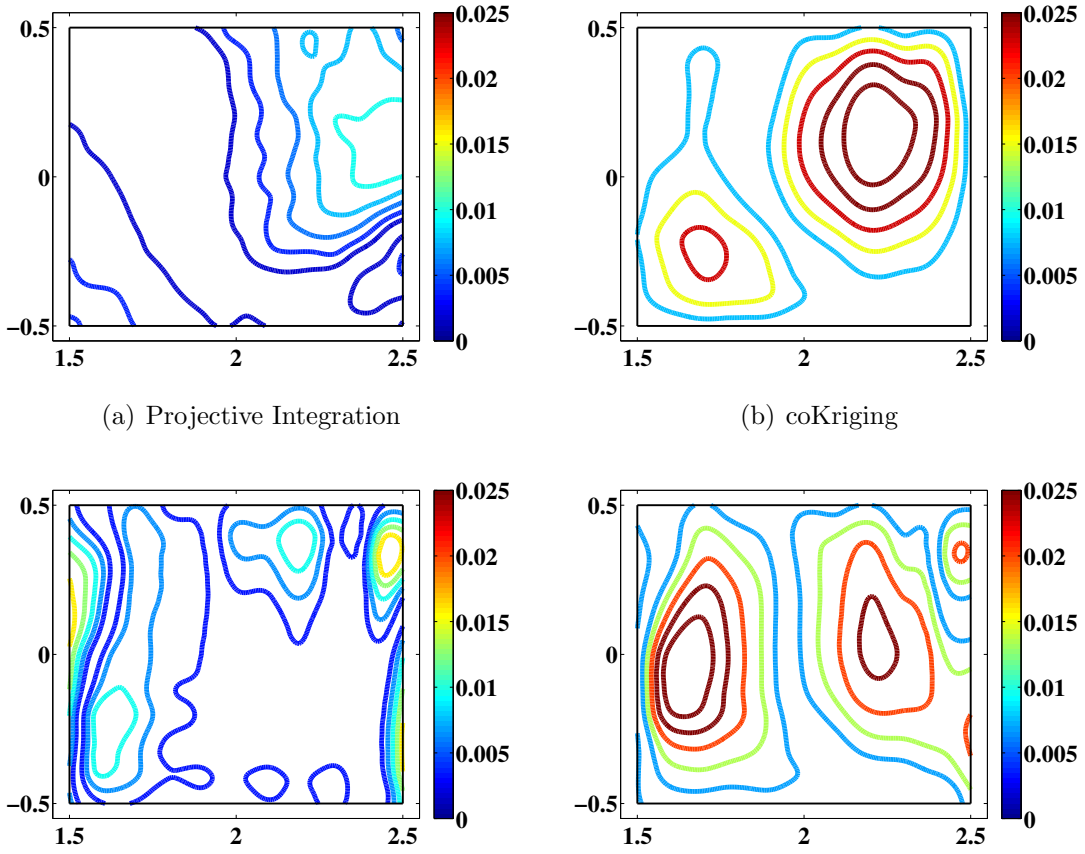
(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.9: Flow past a circular cylinder in scenario 1 and 2: Absolute error of streamwise velocity at $\Delta T_g = 0.27$.



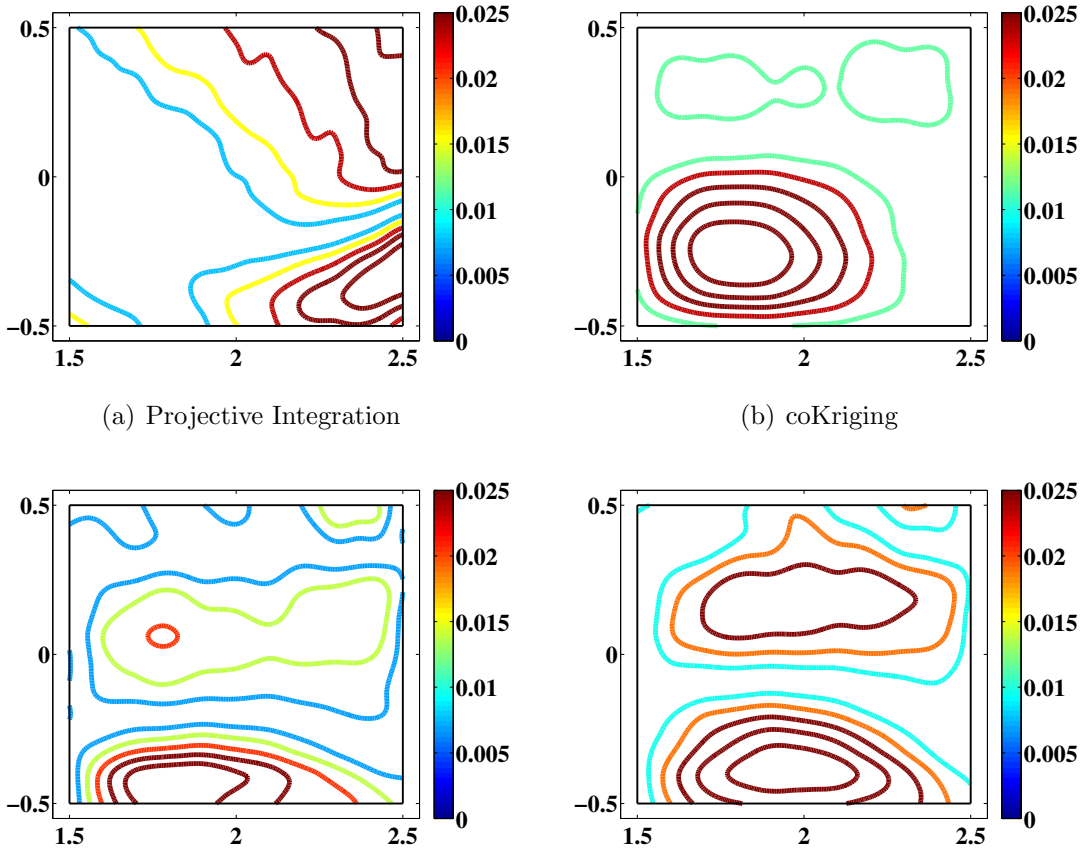
(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.10: Flow past a circular cylinder in scenario 1 and 2: Absolute error of crossflow velocity at $\Delta T_g = 0.27$.



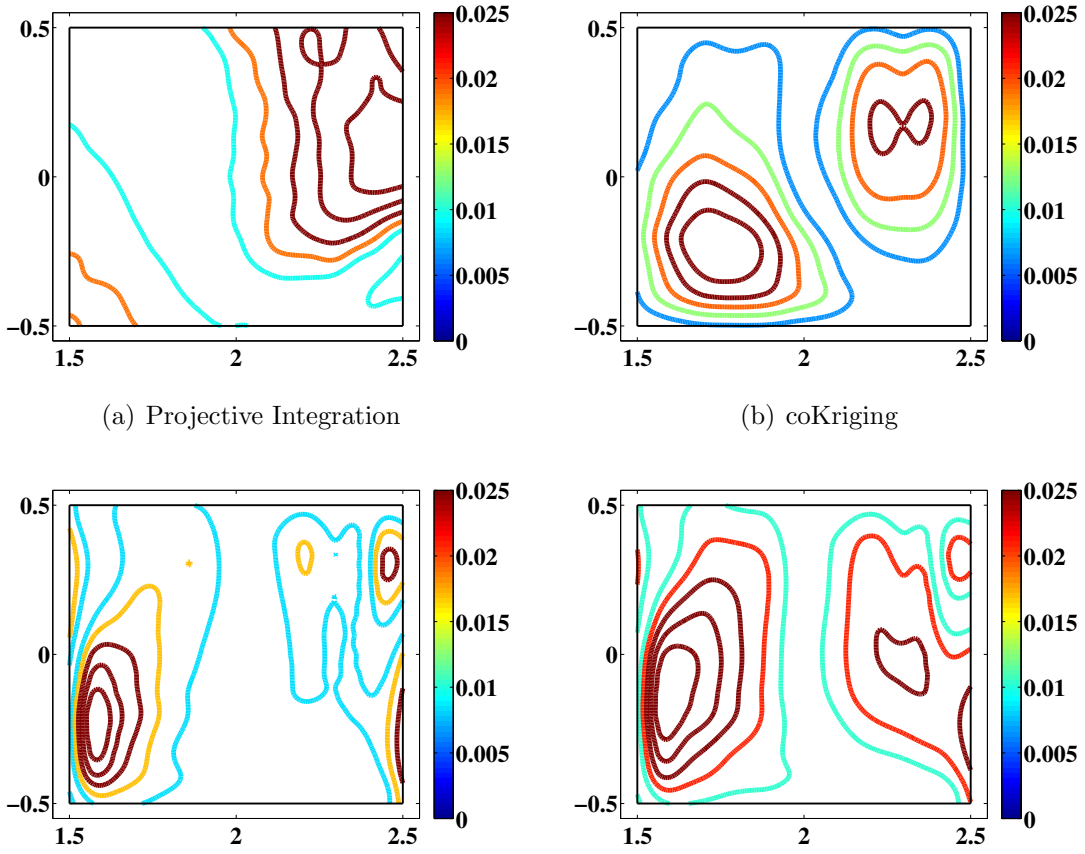
(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.11: Flow past a circular cylinder in scenario 1 and 2: Absolute error of streamwise velocity at $\Delta T_g = 0.47$.



(a) Projective Integration

(b) coKriging

(c) Resimulation in Scenario 1

(d) Resimulation in Scenario 2

Figure 2.12: Flow past a circular cylinder in scenario 1 and 2: Absolute error of crossflow velocity at $\Delta T_g = 0.47$.

method seems to be the resimulation method.

2.4.2 Scenario 2

The second scenario assumes that there exist previous gaps but there is no contamination in the neighboring data, which can be used in the reconstruction. In this scenario, the projective integration cannot be employed because of the absence of previous data in the missing region. Hence, we employ the coKriging and resimulation methods only for reconstruction of the gappy flow field. Also, due to lack of correct initial conditions to be used in the resimulation method, we employ the coKriging interpolation for estimating the initial condition, see 2.2. Hence, the resimulation method will inherit an error due to approximate initial conditions but it will employ the correct (Dirichlet) boundary conditions since we assumed that the current neighboring data are not contaminated.

Lid-driven cavity flow:

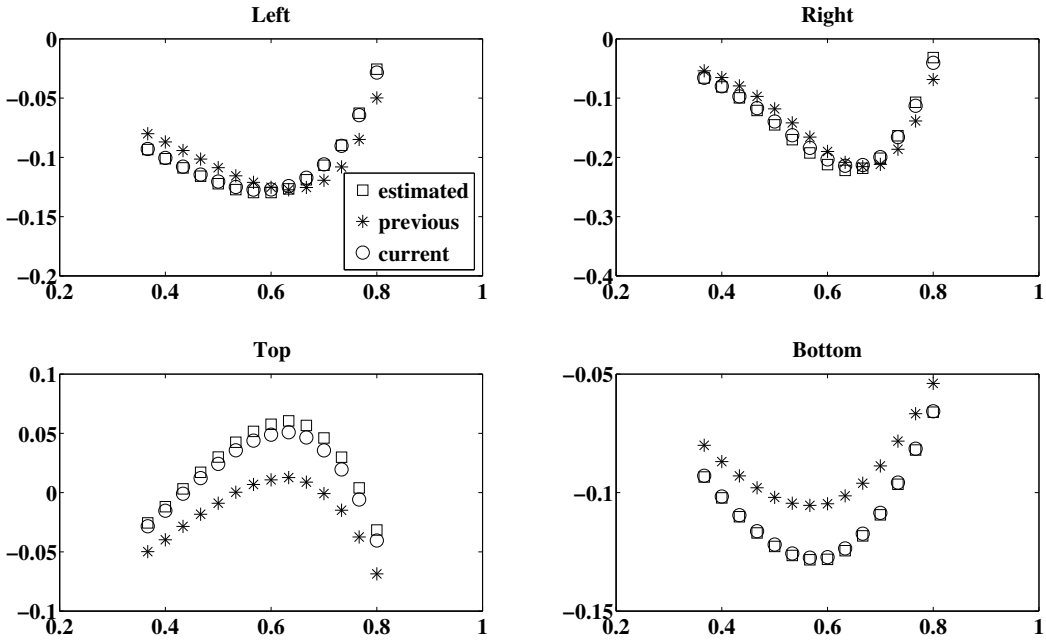
Even though we employ the coKriging interpolation for the initial condition of the resimulation method, the resimulation method seems to be more accurate than the coKriging method, see 2.1. Moreover, the error distribution is similar to scenario 1, which uses correct initial conditions. Based on this result we observe that the initial perturbation by the coKriging method is "forgotten" during resimulation with a divergence-free constraint. Furthermore, the maximum RMS error is located near the boundary as shown in Figures 2.5-2.8.

Flow past a circular cylinder:

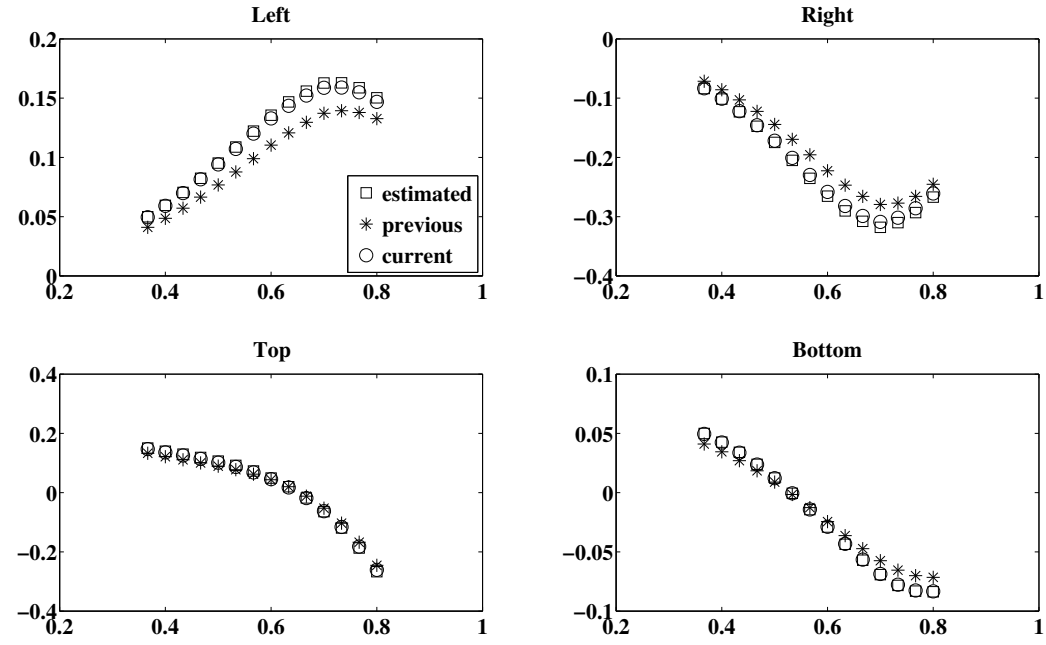
While the resimulation method is better than the coKriging method for $\Delta T_g = 0.27$, the coKriging method is competitive with the resimulation method for $\Delta T_g = 0.47$. The reason is that the resimulation method is affected by the initial condition approximation by the coKriging method and the error due to the Dirichlet boundary condition for big time gaps. Specifically, the coKriging method is better in predicting the crossflow velocity. As shown in 2.2, the RMS errors of the coKriging and resimulation methods are bigger than the same methods in scenario 1, which means that the initial perturbation in the gappy region affects the entire missing domain. Figures 2.9-2.12 show that the error distribution is affected by both initial and boundary conditions while the error in the lid-driven cavity flow appears to be affected by the boundary condition only. Hence, the initial perturbation is a key factor in error propagation for unsteady flows.

2.4.3 Scenario 3

The third scenario represents the worst case. There exist previous time gaps but, in addition, the current neighboring data are somehow contaminated by silent error. In this scenario, we cannot employ the projective integration or the coKriging method because of the absence of accurate previous data and the contamination of neighboring data. Hence, the only way to reconstruct gappy flow fields is the resimulation method, where we need to consider carefully how to set appropriate initial and boundary conditions. A rather simple way is to employ coKriging to estimate the initial condition and employ previously saved flow field data as a boundary condition, which we will refer to as "previous boundary". However, this may propagate

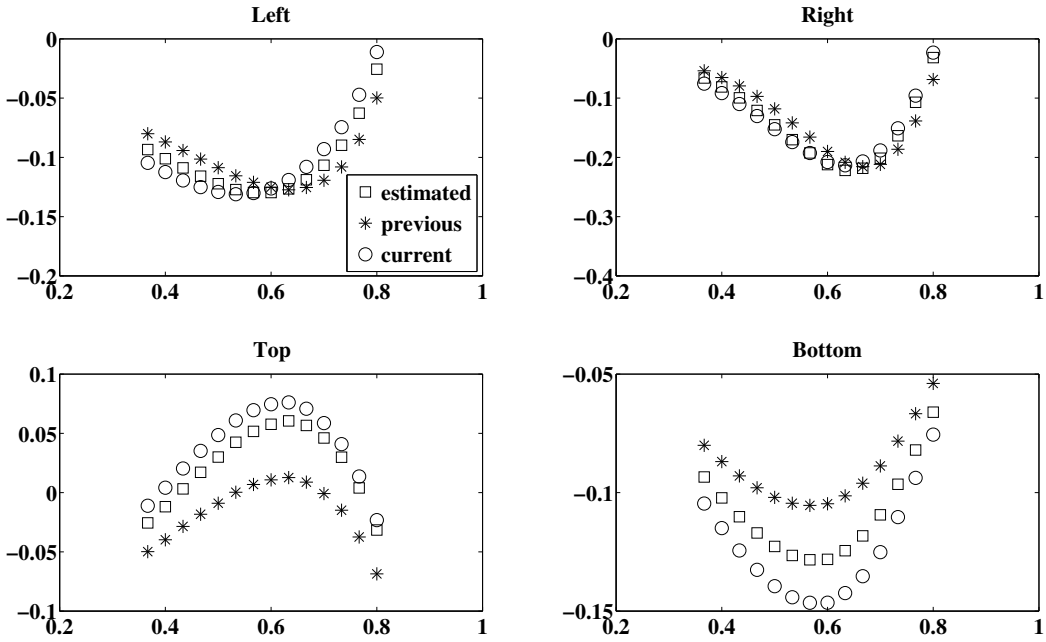


(a) Streamwise velocity

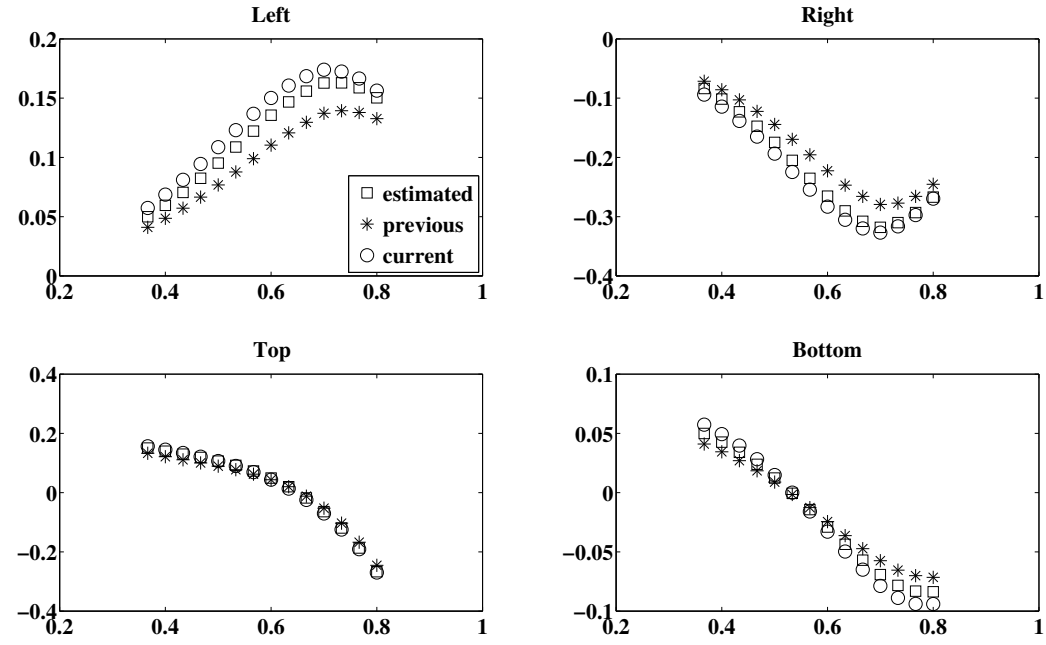


(b) Crossflow velocity

Figure 2.13: Lid-driven cavity flow in scenario 3: Comparison of velocity at the boundary of the gappy region at $\Delta T_g = 0.5$. "Current" represents the correct value, "previous" represents last saved data, and "estimated" is calculated by Projective Integration.

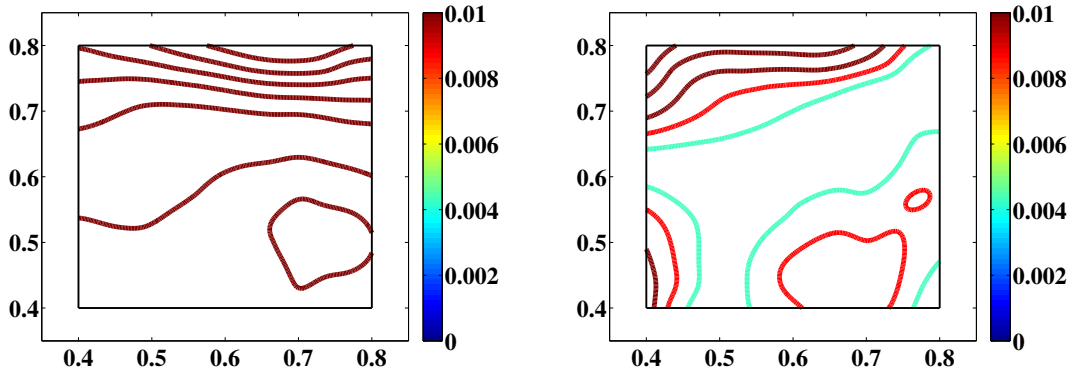


(a) Streamwise velocity



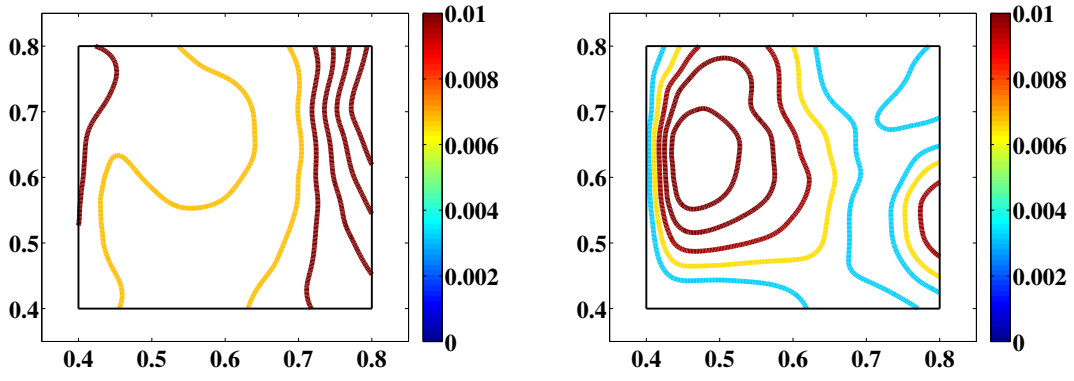
(b) Crossflow velocity

Figure 2.14: Lid-driven cavity flow in scenario 3: Comparison of velocity at the boundary of the gappy region at $\Delta T_g = 1.0$. "Current" represents the correct value, "previous" represents last saved data, and "estimated" is calculated by Projective Integration.



(a) Streamwise velocity

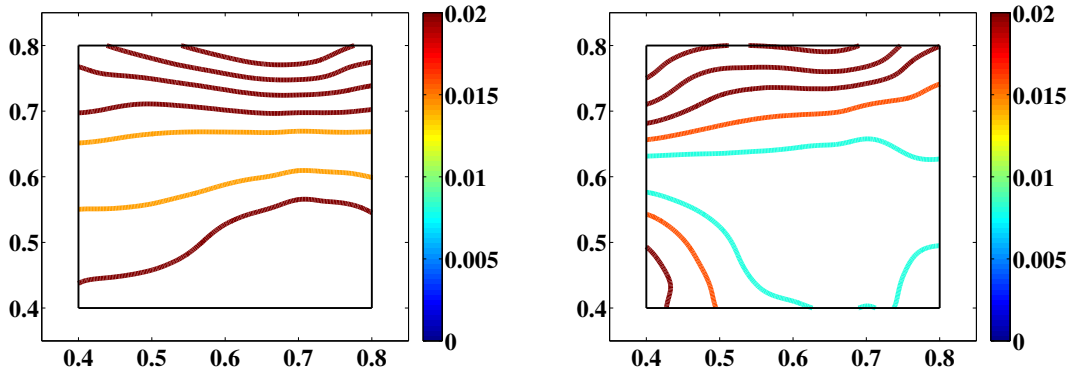
(b) Streamwise velocity



(c) Crossflow velocity

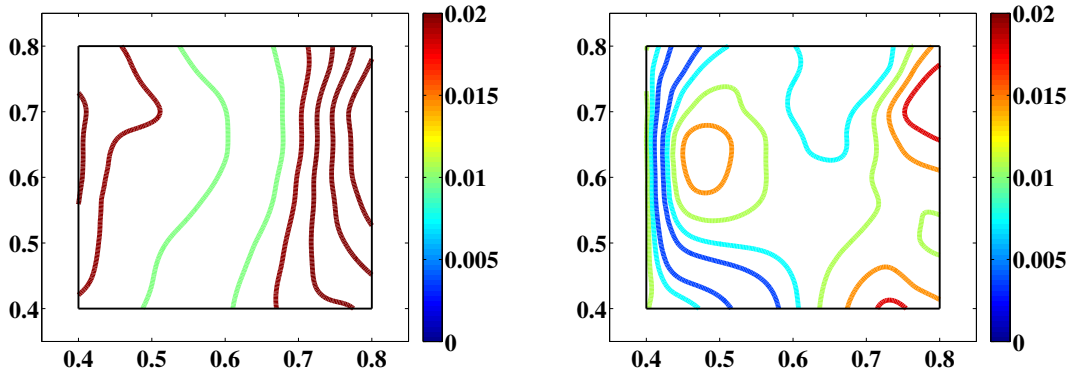
(d) Crossflow velocity

Figure 2.15: Lid-driven cavity in scenario 3: Absolute error of different boundary at $\Delta T_g = 0.5$. Left: previous boundary, Right: estimated boundary.



(a) Streamwise velocity

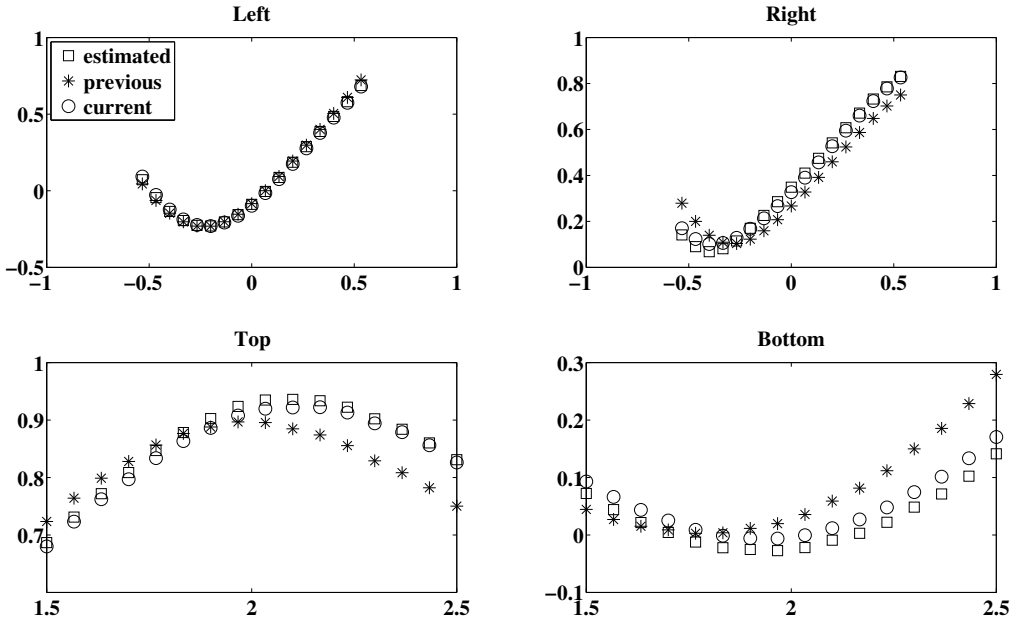
(b) Streamwise velocity



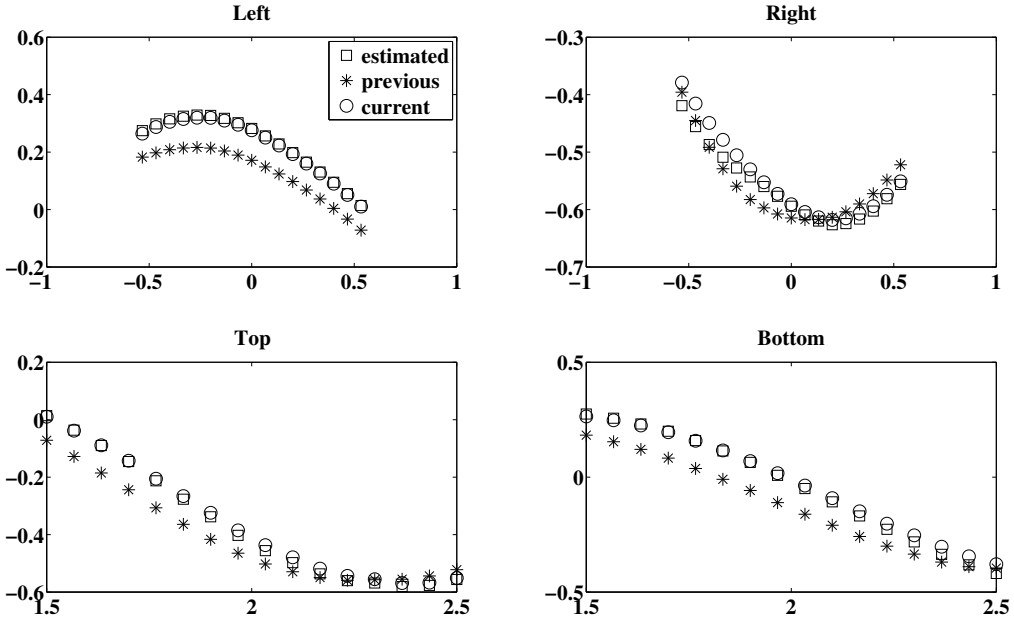
(c) Crossflow velocity

(d) Crossflow velocity

Figure 2.16: Lid-driven cavity in scenario 3: Absolute error of different boundary at $\Delta T_g = 1.0$. Left: previous boundary, Right: estimated boundary.

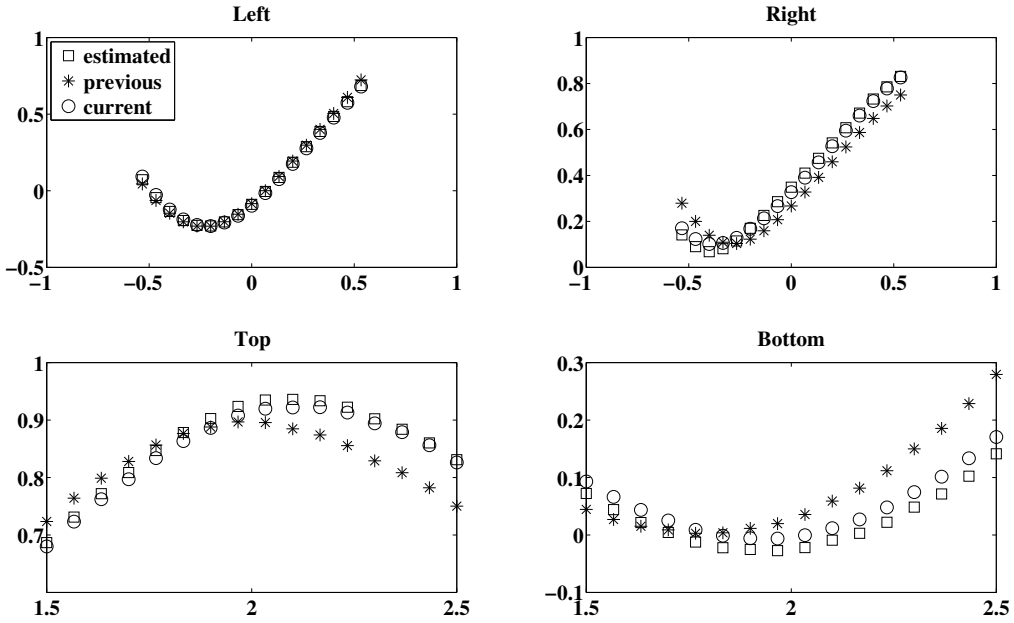


(a) Streamwise velocity

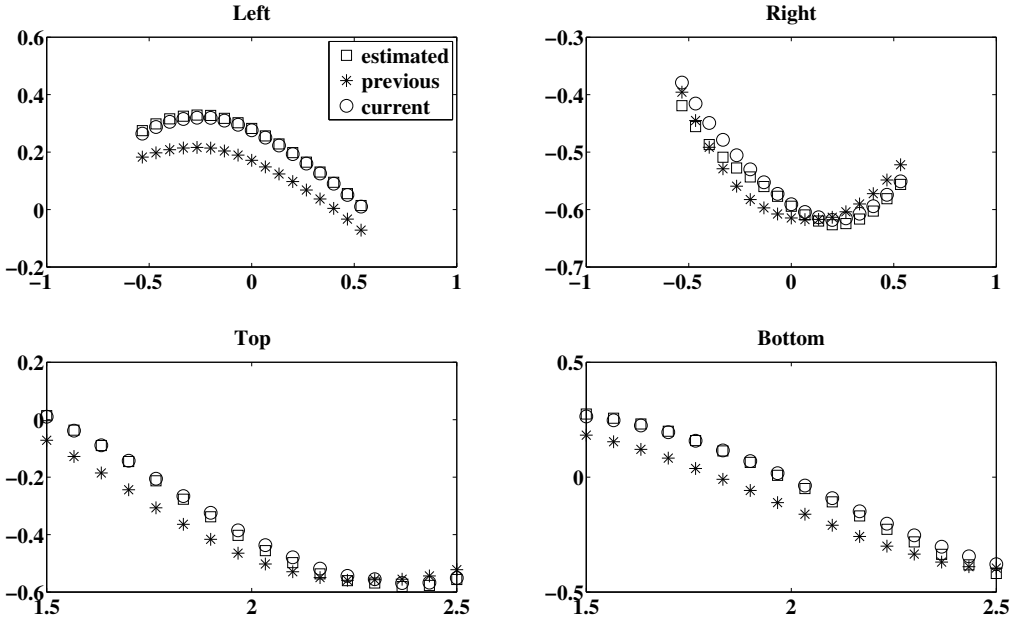


(b) Crossflow velocity

Figure 2.17: Flow past a circular cylinder in scenario 3: Comparison of velocity at the boundary of the gappy region at $\Delta T_g = 0.27$. "Current" represents the correct value, "previous" represents last saved data, and "estimated" is calculated by Projective Integration.

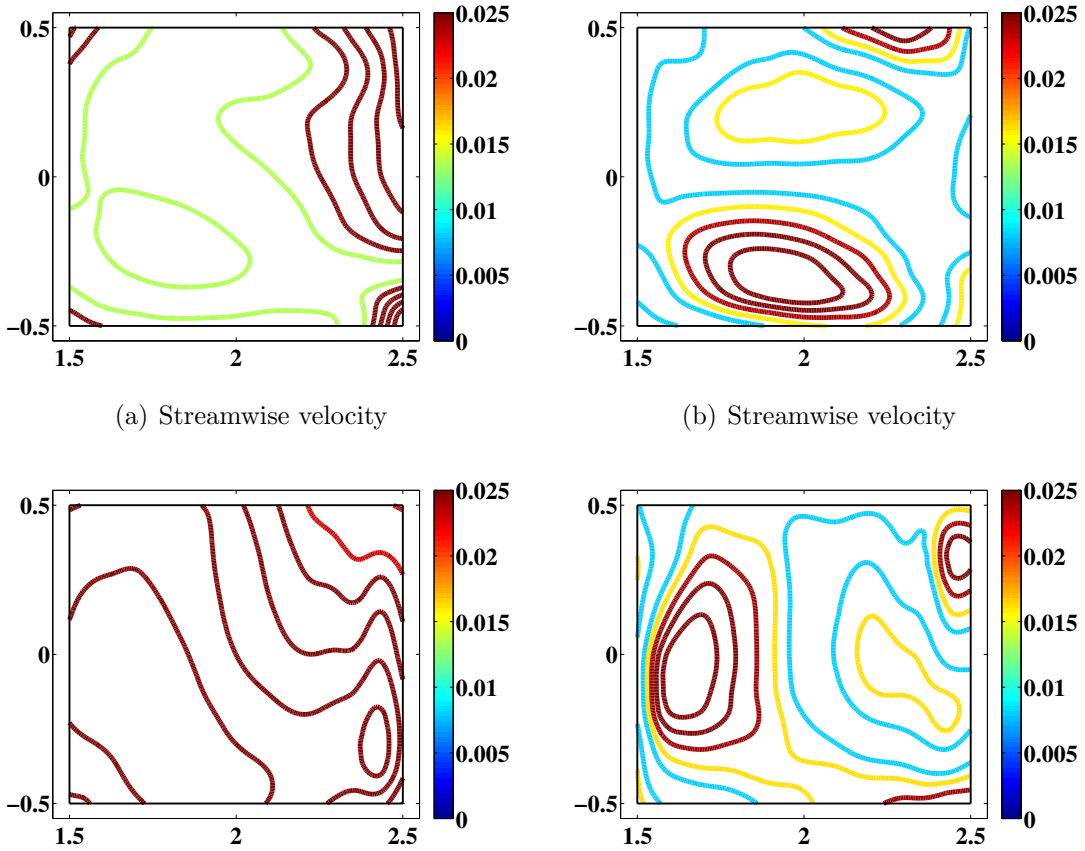


(a) Streamwise velocity



(b) Crossflow velocity

Figure 2.18: Flow past a circular cylinder in scenario 3: Comparison of velocity at the boundary of the gappy region at $\Delta T_g = 0.47$. "Current" represents the correct value, "previous" represents last saved data, and "estimated" is calculated by Projective Integration.



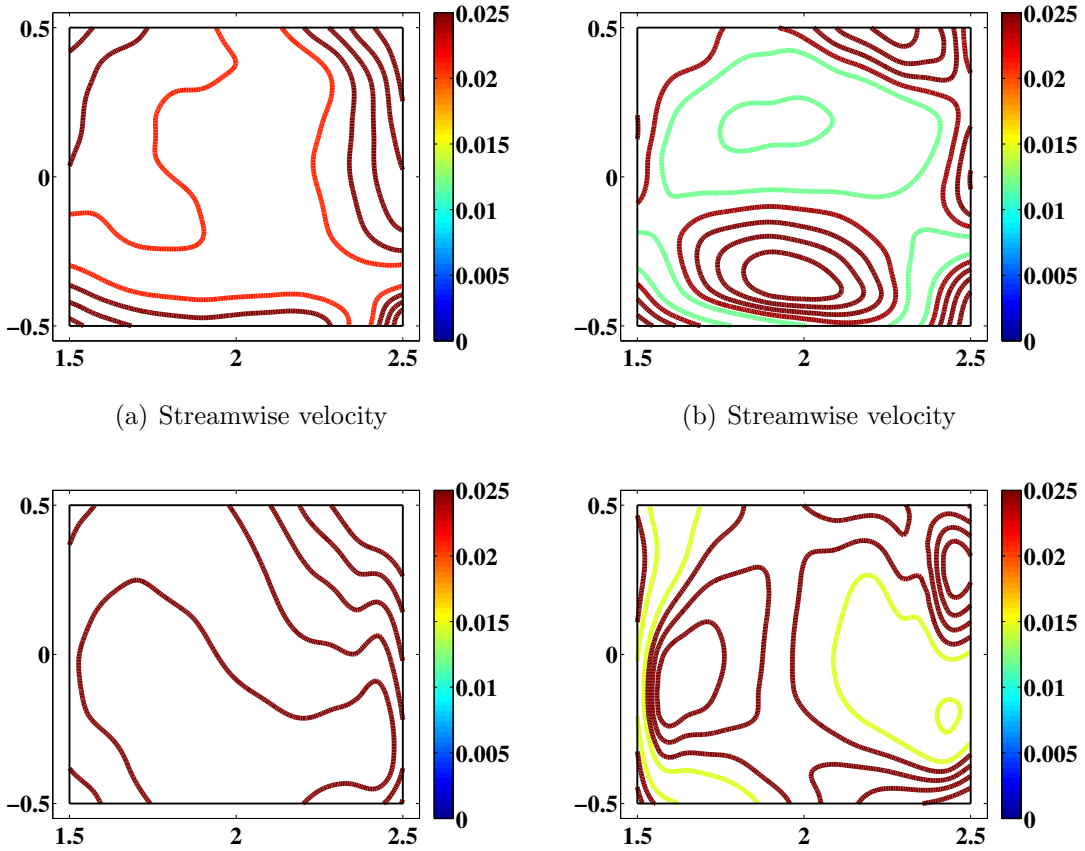
(a) Streamwise velocity

(b) Streamwise velocity

(c) Crossflow velocity

(d) Crossflow velocity

Figure 2.19: Flow past a circular cylinder in scenario 3: Absolute error of different boundary at $\Delta T_g = 0.27$. Left: previous boundary, Right: estimated boundary.



(a) Streamwise velocity

(b) Streamwise velocity

(c) Crossflow velocity

(d) Crossflow velocity

Figure 2.20: Flow past a circular cylinder in scenario 3: Absolute error of different boundary at $\Delta T_g = 0.47$. Left: previous boundary, Right: estimated boundary.

Table 2.3: RMS Errors for different boundary conditions in scenario 3.

Type	Velocity	Time gaps (ΔT_g)	Previous	Estimated
Lid-driven Cavity[1]	streamwise	0.5	0.0170	0.0078
		1.0	0.0310	0.0158
	crossflow	0.5	0.0116	0.0088
		1.0	0.0214	0.0105
Flow Past a Cylinder[2]	streamwise	0.27	0.0248	0.0175
		0.47	0.0386	0.0291
	crossflow	0.27	0.0980	0.0189
		0.47	0.1797	0.0374

errors into the entire gappy region and more globally due to the erroneous boundary condition. In order to avoid this issue and increase accuracy, we introduce the concept of "estimated boundary" by the projective integration method. This concept uses the same initial condition as before (by the coKriging method) but employs the projective integration method for extracting proper boundary condition from points in a sample set. In this section, we examine the quality of the simulation results for two different boundary conditions, the "previous" and "estimated" boundary conditions.

Lid-driven cavity flow:

We see from 2.3 that the "estimated" boundary condition reduces the error significantly. However, the crossflow velocity is captured reasonably well by the "previous" boundary condition as well. The reason is that boundary values are not changing rapidly as shown in Figures 2.13 and 2.14. In other cases the "estimated" boundary gives better result because it can estimate the current velocity at the boundary better. When imposing "estimated" boundary, the absolute error at the boundary is reduced dramatically compared to "previous" boundary, see 2.15 and 2.16. This results in total error reduction in the entire missing region.

Flow past a circular cylinder:

In this case, the resimulation method with the "estimated" boundary seems to provide the better approach. In this unsteady flow, the current boundary is totally different from the "previous" saved boundary as shown in Figures 2.17-2.18. The "estimated" boundary by the projective integration can capture the current boundary condition well for both $\Delta T_g = 0.27$ and 0.47. This results in a noticeable error reduction at the boundary in this unsteady flow even though the inner part of the missing region still has large error, see 2.3, 2.19 and 2.20. Furthermore, in the cross-flow velocity, the "estimated" boundary leads to the significant error reduction in 2.20. So in general, the "estimated" boundary approach seems to be a more robust method for predicting the proper boundary values, see results in 2.3.

2.5 Summary

Data assimilation in *Computational* Fluid Dynamics is a relatively unexplored area. The techniques that we have developed in this chapter by combining numerical approximation with statistical learning methods (e.g., Kriging/coKriging) for reconstruction of gappy fields can also be used for data assimilation. Here, our interest has been the "restarting" of a CFD simulation in a massively parallel environment, which may be interrupted due to hardware or software problems in only one relatively small part of the space-time computational domain. To this end, we developed and tested three empirical approaches to reconstruct gappy flow fields. Specifically, we performed simulations of two "textbook" CFD problems, namely flow in a lid-driven cavity and flow past a circular cylinder in two dimensions, and under three different fault scenarios with progressive complexity. We summarize here the main findings

of our study:

- For sufficiently small time gaps the projective integration method is the best while for longer time gaps the coKriging method is better.
- Overall, the resimulation method seems to be the most robust method, performing well in all three fault scenarios.
- Estimating the boundary condition using projective integration leads to accurate results for the resimulation method in scenario 3 where the other two methods fail.

Chapter Three

Fault-resilient simulations based on multi-resolution information fusion

3.1 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 [76, 113, 84, 6, 108]. (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 [94, 4, 72, 14]. While one may think that the latter is a computer science issue and can be solved e.g. via fault-tolerant MPI [41, 39, 47, 7], there may be an algorithmic solution to it that can lead to affective 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 chapter 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 [37, 91, 54]. 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.

Considering the faulty processors scenario, we may assume that we encounter 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

[117, 1, 77, 2, 55]. In particular, for spatial estimation for gaps in fluid dynamics, the gappy proper orthogonal decomposition (gappy POD) method was introduced in [29, 106, 110], 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 [103, 65, 43, 46, 80]. More recently, a resilient algorithm based on a "re-simulation" method for CFD was introduced in [62], demonstrating the possibility of filling the spatial gaps effectively at a fixed time.

In this chapter, 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 discretizations. The new framework can be extended to generalize previous multiscale approaches (e.g., continuum-atomistic) [32] in a unified parallel computational framework.

This chapter 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.

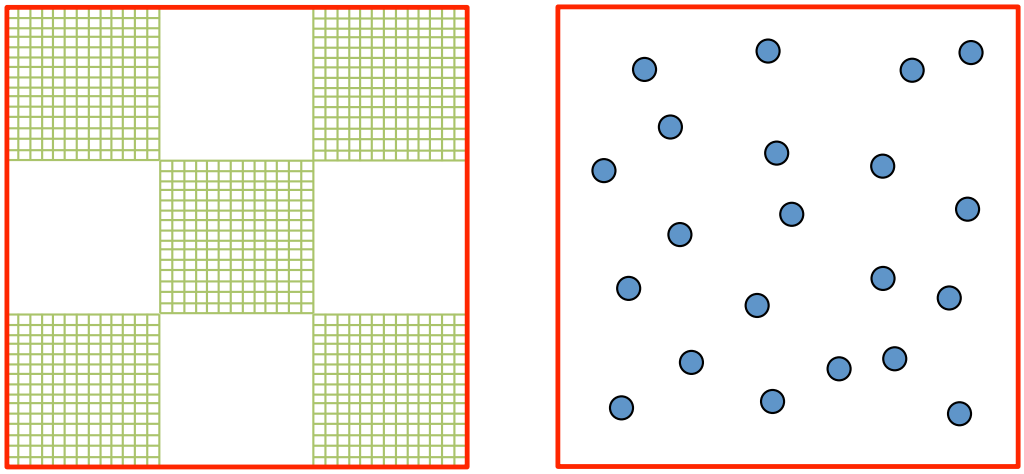


Figure 3.1: A schematic illustration of a gappy simulation: Computations are performed only on the green subdomains (left) with independent auxiliary data on a small number of points (right).

3.2 Gappy Simulation Framework

3.2.1 Problem set up

In the gappy simulation framework we compute explicitly the solution to a PDE not on the entire domain but only partially on some subdomains (colored by green) with some auxiliary data that are distributed across the entire domain (colored by blue) and obtained independently, see Figure 3.1. The main idea is to combine the global coarse information with some finely resolved subdomains 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 subdomains represents dynamics of finer scales. From the parallel computing perspective, the gappy subdomains 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"

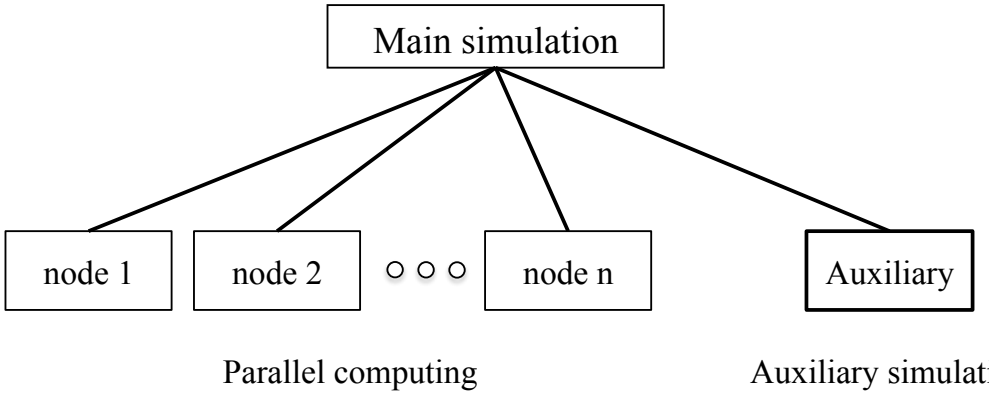


Figure 3.2: Breakup of the computation into fine resolution on a few patches (parallel execution) and into an auxiliary computation in the entire domain. Auxiliary data should be obtained from an independent computer node and should be tuned to run (approximately) synchronously with the main parallel simulation.

algorithm for micro-simulators [37, 91, 54]. 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 subdomains (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.

3.2.2 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 [71, 42, 98], see Figure 3.2. 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. Alternatively, a lower-dimensional model

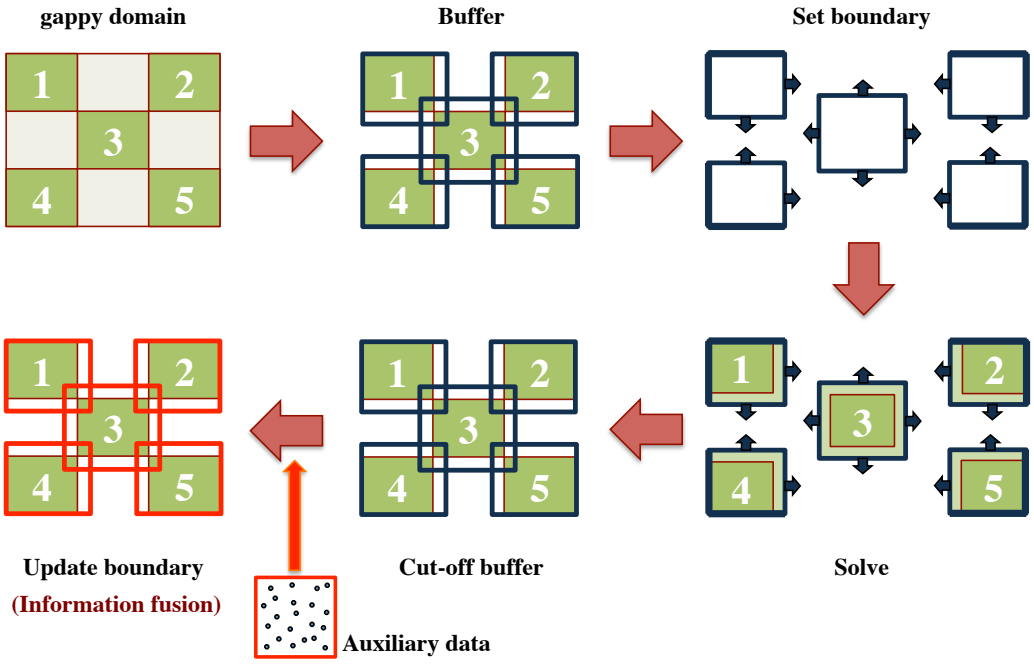


Figure 3.3: A flow chart for a gappy simulation (start from left-top): We first check where the gappy domains are located. Next, we choose a buffer size, impose appropriate boundary conditions, and estimate field variables at local boundaries. Each subdomain is solved in parallel and independently during 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 subdomains. This is one complete cycle of the gappy simulation algorithm.

can be constructed based on a coarse-grained simulation and be run in parallel with the main simulation but on a smaller fault-free computer. 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. 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).

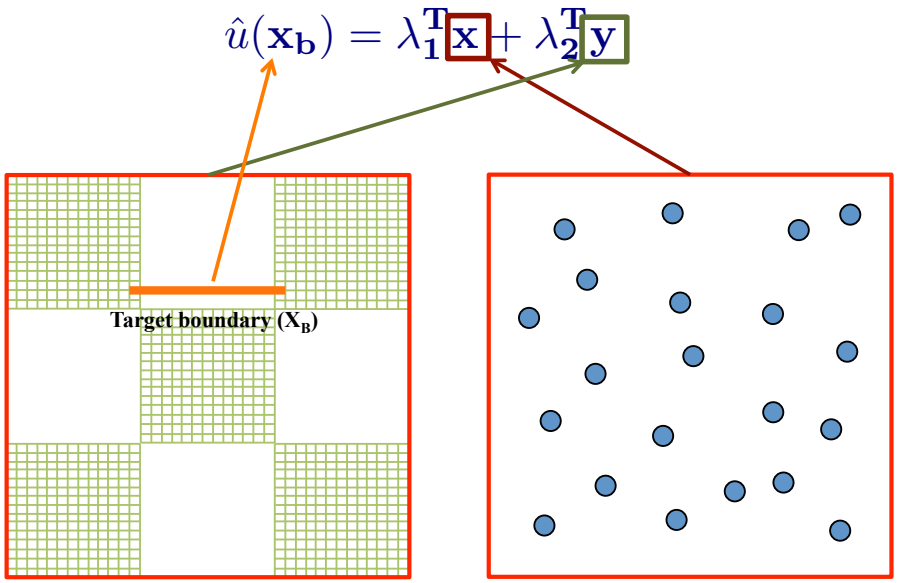


Figure 3.4: A schematic representation of information fusion via coKriging: coKriging uses two data sets, red and green: a red box represents a set of auxiliary data and a green box represents a set of data from the gappy simulation.

3.2.3 Algorithm flow chart

A flow chart of the gappy simulation is shown in Figure 3.3. First, upon notification of a fault detection (not discussed here), we check which domains are affected by errors, and define computational subdomains and gaps. Next, we choose a proper buffer size and corresponding boundary conditions for each subdomain. After imposing the boundary conditions, the gappy simulation estimates the field variables at the local boundaries of each subdomain 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 subdomain on independent nodes during non-interaction time $\tau \cdot \Delta t$. After time $\tau \cdot \Delta t$, all subdomains are re-joined together and the buffer region of each subdomain 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.

3.2.4 CoKriging

In order to estimate field variables at the local boundary, a multi-fidelity coKriging interpolation method is introduced [62, 23, 52]. In this chapter, 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 Figure 3.4. The basic idea of this method is that the estimated field variable $\hat{y}(\mathbf{x})$ at the local boundary $\mathbf{x}$ can be represented by a linear combination of two data sets as follows:

$$\hat{y}(\mathbf{x_b}) = \lambda_\mathbf{1}^\mathbf{T}\mathbf{x} + \lambda_\mathbf{2}^\mathbf{T}\mathbf{y}, \tag{3.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is a field data vector in the auxiliary data and $\mathbf{y} \in \mathbb{R}^m$ is in the gappy simulation data. Also, λ_1 and λ_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.,

$$\underset{\boldsymbol{\lambda 1},\boldsymbol{\lambda 2}}{\arg\min}\ E[(\hat{y}(\mathbf{x_b}) - y(\mathbf{x_b}))^2], \tag{3.2}$$

subject to the unbiased constraints

$$E[\hat{y}(\mathbf{x_b})] = E[y(\mathbf{x_b})] \quad \text{or} \quad \sum_{i=1}^{n} \lambda_{1_i} = 1 \text{ and } \sum_{i=1}^{m} \lambda_{2_i} = 0. \tag{3.3}$$

Then, we can solve the linear system with Lagrange multipliers μ_1 and μ_2 as follows:

$$\begin{bmatrix} \mathbf{C_{11}} & \mathbf{C_{12}} & \mathbf{1} & \mathbf{0} \\ \mathbf{C_{21}} & \mathbf{C_{22}} & \mathbf{0} & \mathbf{1} \\ \mathbf{1^T} & \mathbf{0^T} & 0 & 0 \\ \mathbf{0^T} & \mathbf{1^T} & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_\mathbf{1} \\ \lambda_\mathbf{2} \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} \mathbf{c_1}(\mathbf{x_b}) \\ \mathbf{c_2}(\mathbf{x_b}) \\ 1 \\ 0 \end{bmatrix}, \tag{3.4}$$

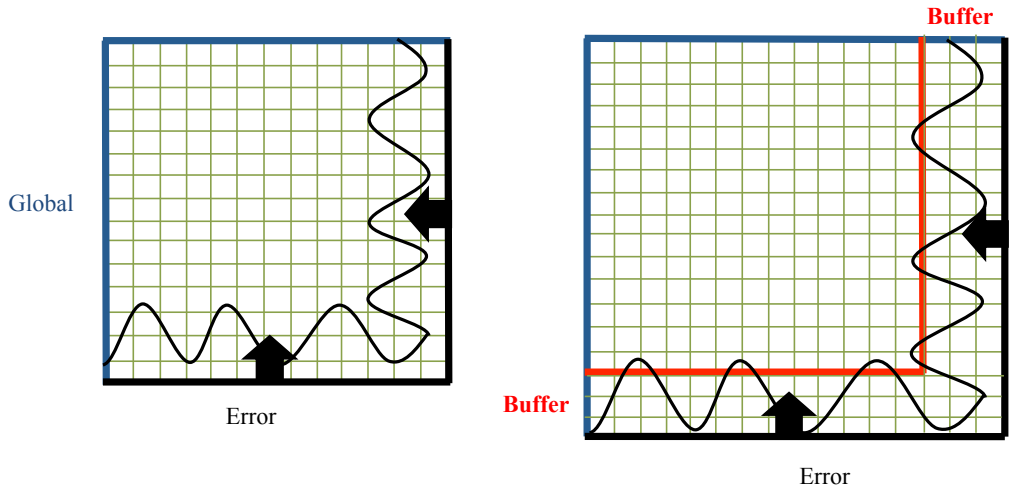


Figure 3.5: A schematic illustration of a buffer: The left plot shows numerical errors propagating from the boundaries into the subdomains. The right plot shows how the buffer can prevent the error at the local boundary from entering the fine-resolution subdomains.

where $\mathbf{C}$ is a covariance matrix whose submatrix $\mathbf{C_{11}}$ is the covariance matrix between points in auxiliary data, $\mathbf{C_{22}}$ between points in the fine-resolution subdomains, and $\mathbf{C_{12}}$ between the auxiliary data and the subdomains. Here, $\mathbf{c_1}(\mathbf{x})$ is the covariance vector between the target point ($\mathbf{x}$) and points in the auxiliary data, and $\mathbf{c_2}(\mathbf{x})$ between the target point ($\mathbf{x}$) and points in the subdomains. In order to set a covariance matrix, we need to choose the proper correlation kernel, see section 4.1.

3.2.5 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 subdomain 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 subdomains 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 Figure 3.5 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.

3.3 Simulation setup

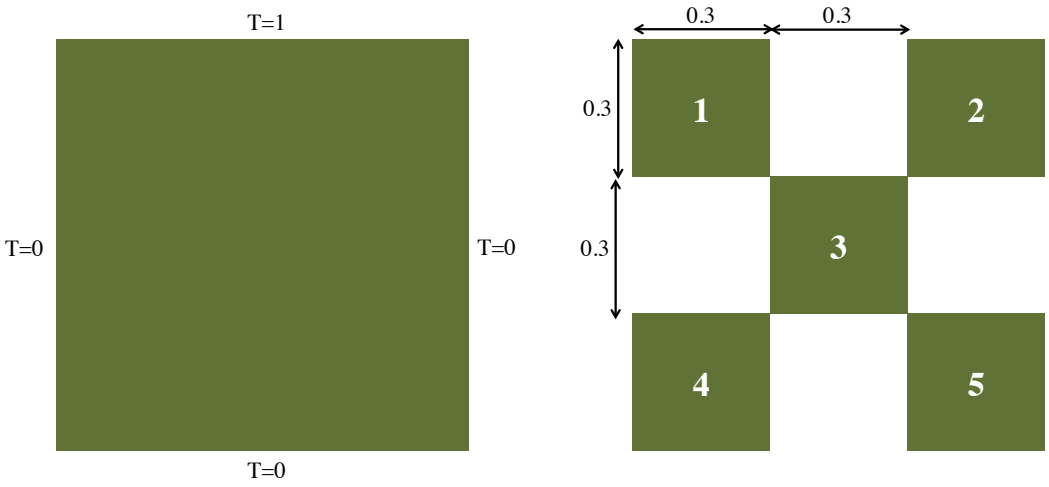
3.3.1 Heat equation

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

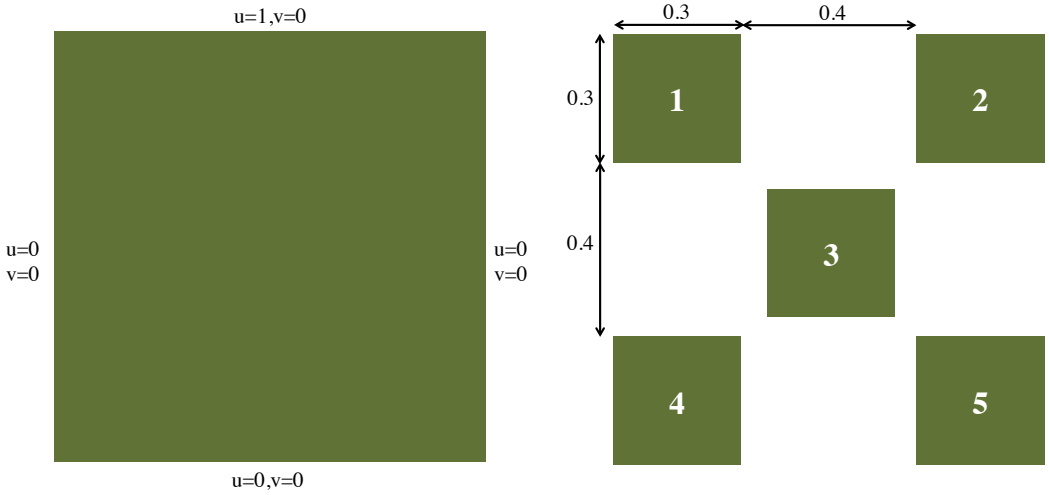
$$\frac{\partial T}{\partial t} = \kappa \nabla^2 T, \tag{3.5}$$

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 11×11 per each subdomain (total of



(a) The heat equation.



(b) The Navier-Stokes Equations.

Figure 3.6: 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 subdomains colored by green. In the heat equation (a), the fine-resolution subdomains are connected by only the corner point of cell "3" while the fine-resolution subdomains are totally disconnected in the Navier-Stokes equations (b).

5 subdomains), with a total of 605 grid points in the gappy simulation while the grid resolution of the reference simulation is 31×31. In order to compare the influence of auxiliary data on the overall accuracy, we employ a coarse grid with resolution 4×4, 6×6, and 10×10. Four subdomains have two global boundaries but one subdomain in the middle, cell 3, has no global boundary, see Figure 4.6(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 Figure 3.7.

3.3.2 Navier-Stokes equations

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

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} + f, \tag{3.6}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{3.7}$$

where $\mathbf{v}$ is the velocity vector, p is pressure, and ν 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 Figure 4.6(b). The resolution of a rectangular cell for each subdomain is 8×8, 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 4×4, 8×8, and 16×16 for the entire domain. The gappy regions are similar as in the

previous example but the only difference is that all fine-resolution subdomains are totally disconnected, see Figure 4.6(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 Figure 3.8.

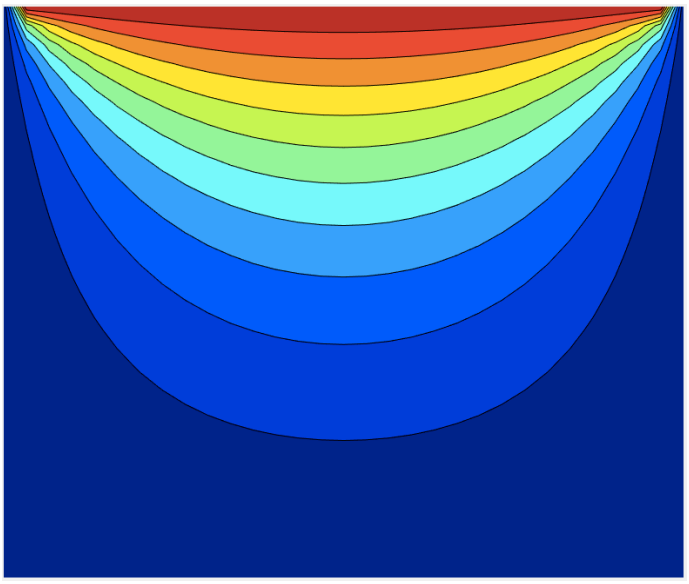
3.4 Results

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

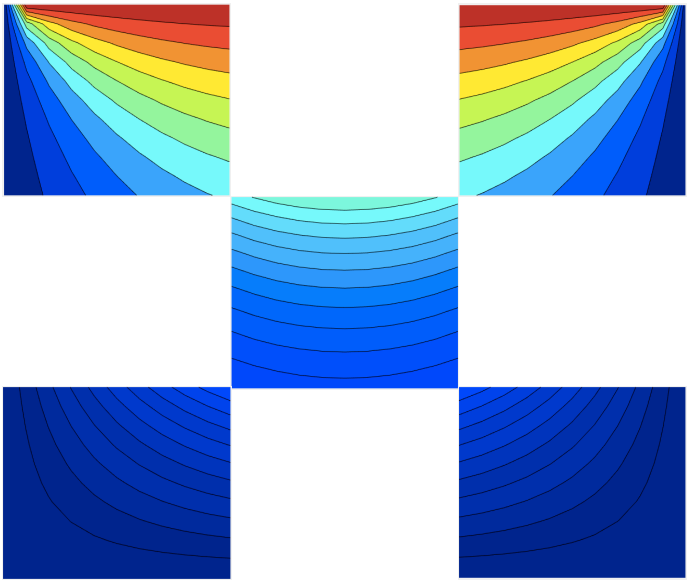
$$\mathrm{RMS_T}(t) = \sqrt{\frac{1}{nN}\sum_{j=1}^{n}\sum_{i=1}^{N}(u_{r,j}(i,t) - u_{g,j}(i,t))^2}, \tag{3.8}$$

where N is the number of grid point of each subdomain and n is the number of fine-resolution subdomains 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 subdomain, respectively. We also introduce another accuracy metric, the time-averaged RMS error from t=0 to t=T, calculated by

$$\overline{\mathrm{RMS_T(T)}} = \frac{1}{T}\int_0^T \mathrm{RMS_T}(t)\mathrm{dt}. \tag{3.9}$$

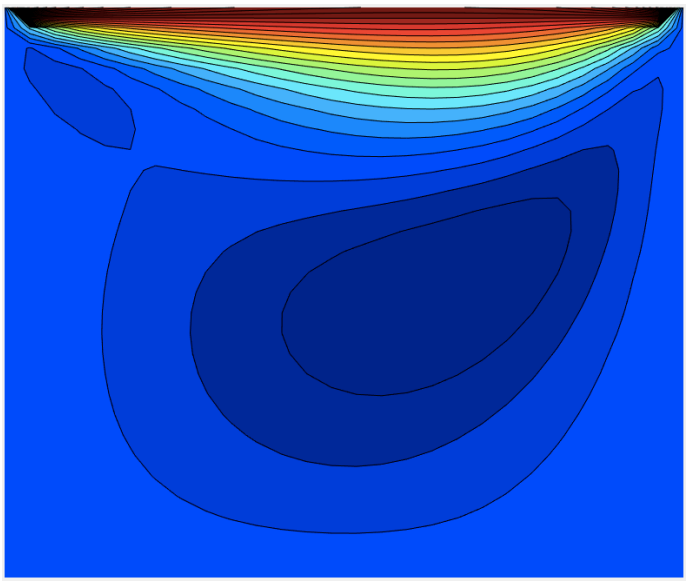


(a) The reference simulation.

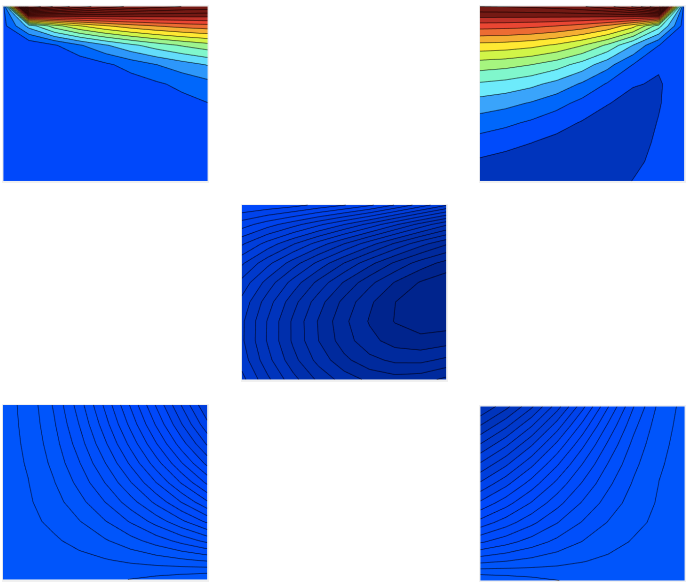


(b) The gappy simulation.

Figure 3.7: Temperature contours for the heat equation: the reference simulation solves the entire domain (31×31 grid) by a second-order finite difference method.



(a) The reference simulation.



(b) The gappy simulation.

Figure 3.8: Streamwise velocity contours for the Navier-Stokes equations: the reference simulation solves the entire domain (27×27 cell), by a second-order finite difference method.

3.4.1 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

$$\kappa_{\mathbf{g}}(x_i, x_j) = \exp\left(-\theta \mathrm{d}(x_i, x_j)^2\right). \tag{3.10}$$

- Exponential

$$\kappa_{\mathbf{e}}(x_i, x_j) = \exp\left(-\theta \mathrm{d}(x_i, x_j)\right). \tag{3.11}$$

- Spherical

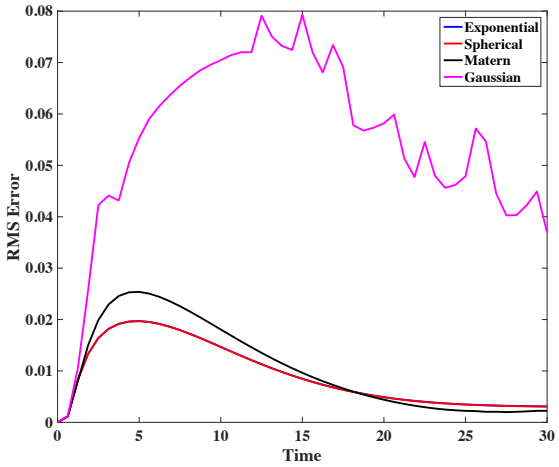
$$\kappa_{\mathbf{s}}(x_i, x_j) = 1 - \frac{3}{2}\min(\theta \mathrm{d}(x_i, x_j), 1) + \frac{1}{2}\left(\min(\theta \mathrm{d}(x_i, x_j), 1)\right)^3. \tag{3.12}$$

- Matérn

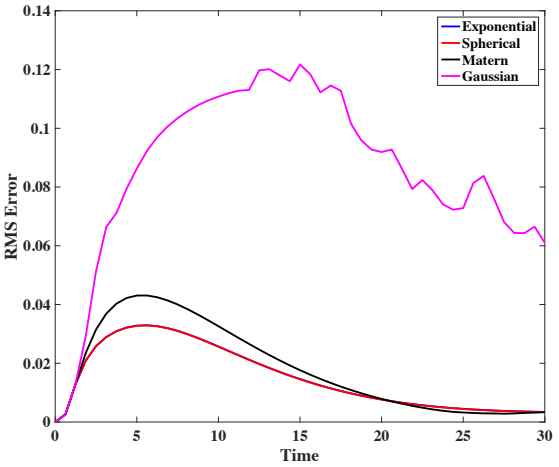
$$\kappa_{\mathbf{m}}(x_i, x_j) = \left(1 + \sqrt{3}\theta \mathrm{d}(x_i, x_j)\right) \exp\left(-\sqrt{3}\theta \mathrm{d}(x_i, x_j)\right). \tag{3.13}$$

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

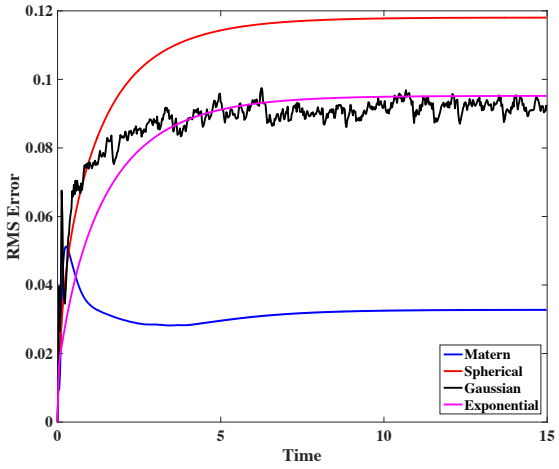
The results of constructing the covariance matrix at the steady-state by different correlation kernels are shown in Figures 3.9 and 3.10. In order to compare the



(a) The heat equation: $\mathrm{RMS_T}$.

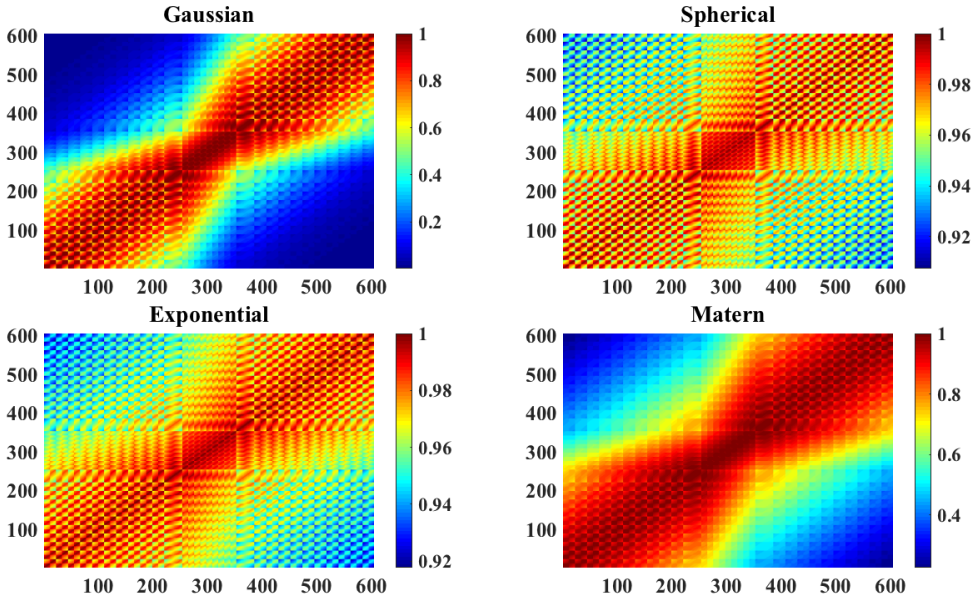


(b) The heat equation: RMSE in cell 3.

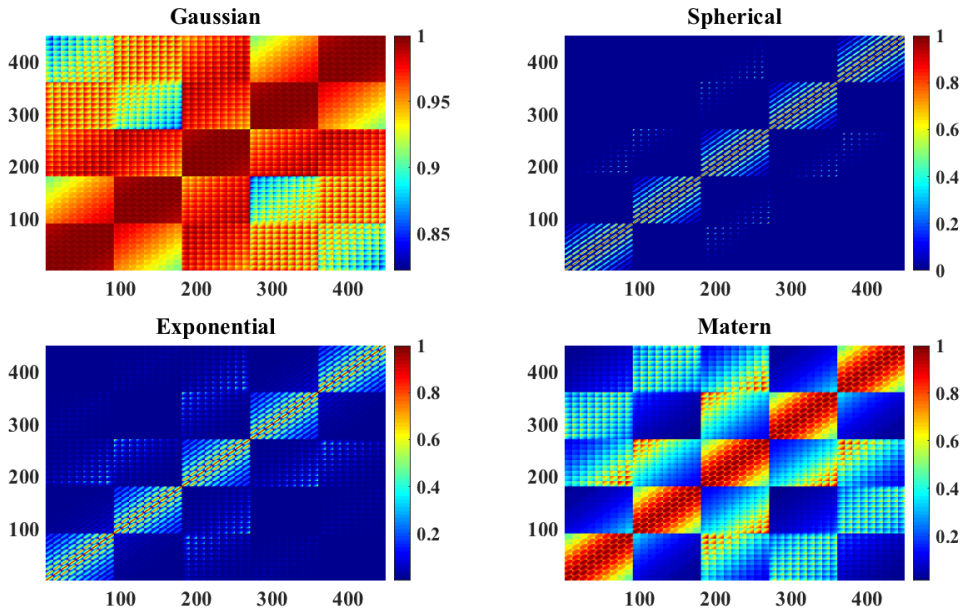


(c) The Navier-Stokes equations.

Figure 3.9: The time history of the total RMS error for different correlation kernels. In (a) and (b), fixed parameters are the size of buffer at 30%, no auxiliary data (by Kriging), and a non-interaction timestep number (τ) of 50. In (c), the fixed parameters are the size of buffer at 25%, no auxiliary data (by Kriging), and a non-interaction timestep number (τ) of 5. (In plots (a) and (b), the blue and red curves coincide).

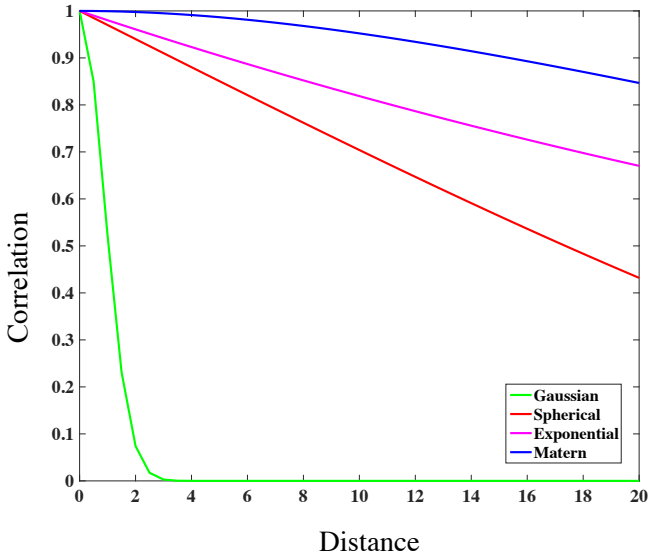


(a) The heat equation.

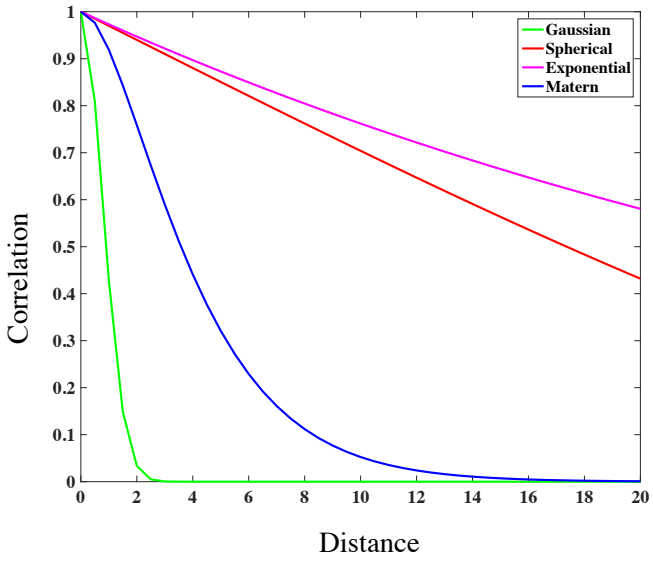


(b) The Navier-Stokes equations.

Figure 3.10: Covariance matrix of different correlation kernels at the steady-state. Fixed parameters are same as in Figure 3.9. In (a), each fine-resolution subdomain has 121 points, totally 605 points. In (b), each fine-resolution subdomain has 90 points, totally 450 points.



(a) t=5 (at transient).



(b) t=30 (at steady-state).

Figure 3.11: Correlation functions (kernels) at two different times. Fixed parameters are same as in Figure 3.9.

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 (τ) is 50, and no auxiliary data are provided by Kriging. In the Navier-Stokes equations, the size of buffer is fixed at 25%, the non-interaction timestep number (τ) is 5, and no auxiliary data are available by Kriging.

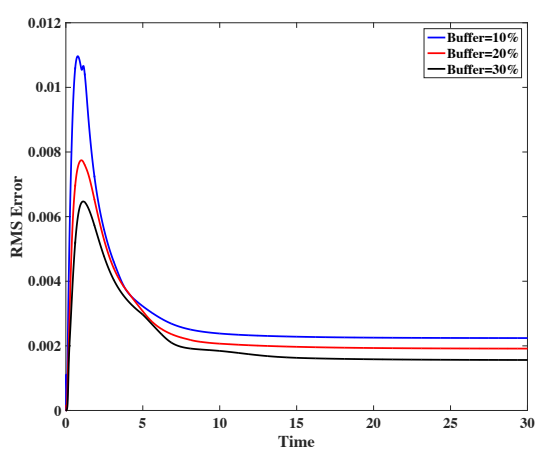
In the heat equation, we found that the best kernel at steady-state is the Matérn kernel. Theoretically, solutions of a diffusion problems yield strong spatial correlations between subdomains. Thus, all kernels show a strong correlation between subdomains except the Gaussian kernel. As shown in Figures 3.10 (a) and 3.11, the Gaussian kernel has weak correlation between other subdomains and this leads to the highest RMS error compared to other kernels. Specifically, as shown in Figure 3.9 (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 subdomains 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 Figure 3.11.

In the Navier-Stokes equations, the spatial correlation between different subdomains 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 points between other subdomains. The exponential kernel has similar but relatively stronger correlation compared to the spherical kernel, see Figure

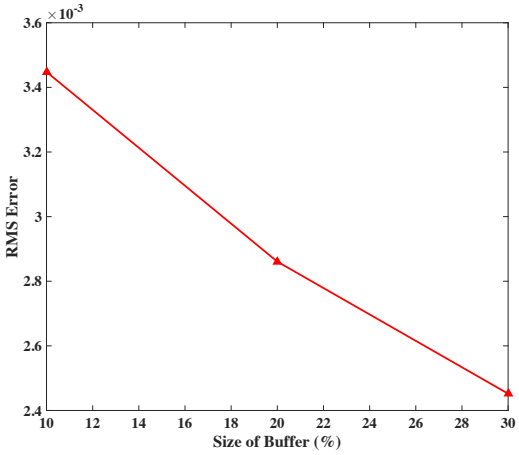
3.10, and this leads to reduction of the RMS error. In the Gaussian kernel, however, the correlations between other subdomains are too strong, that is, too many data can affect an estimated boundary condition. This over-fitting leads to wiggles in the total RMS error. Finally, we found that the Matérn kernel is the best because it has very strong correlation for the inner subdomains 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.

3.4.2 Size of a buffer

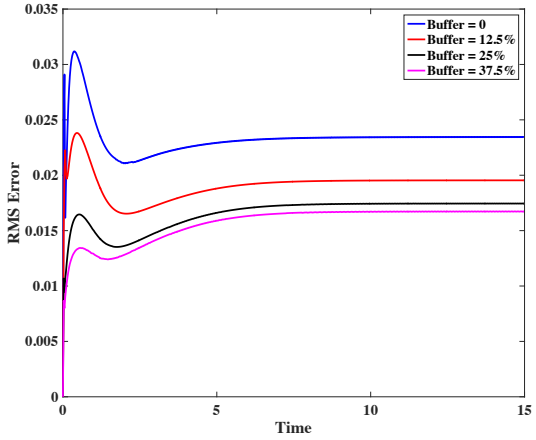
The next parameter we focus on is the size of the buffers we employ for estimating the boundary conditions on each fine-resolution subdomains. The results of the total RMS error and time-averaged RMS error are shown in Figure 3.12. The main observation is that the total and time-averaged RMS errors are 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 Figure 3.12 (a). On the other hand, because the uncertainties at the local boundary are 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 equations, 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



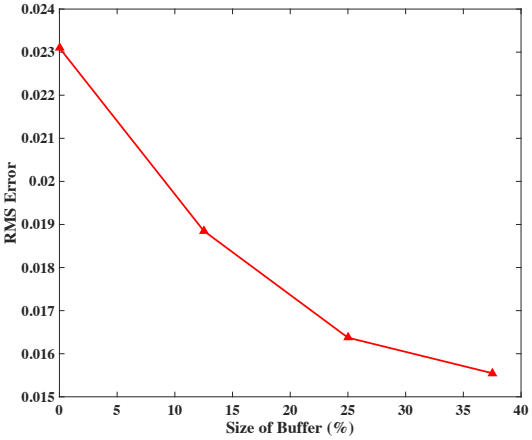
(a) Time history of total RMS error in the heat equation.



(b) Time-averaged RMS error in the heat equation.



(c) Time history of total RMS error in the Navier-Navier-Stokes equations.



(d) Time-averaged RMS error in the Navier-Stokes equations.

Figure 3.12: 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 6×6, and a non-interaction timestep number (τ) of 1. In the Navier-Stokes equations, the fixed parameters are as follows: resolution of auxiliary data is 8×8, and a non-interaction timestep number (τ) of 5.

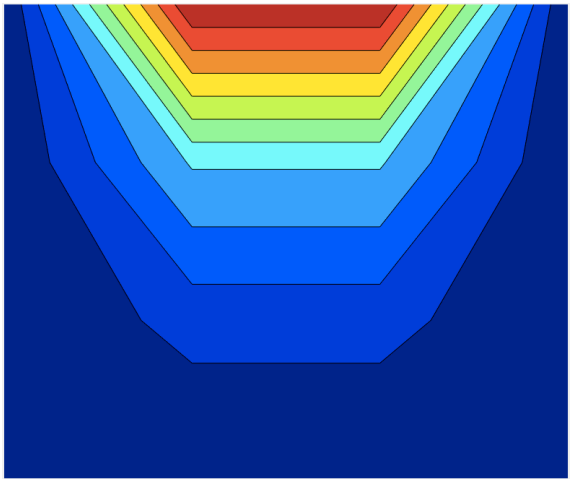
during the transient period and at steady-state.

3.4.3 Auxiliary data

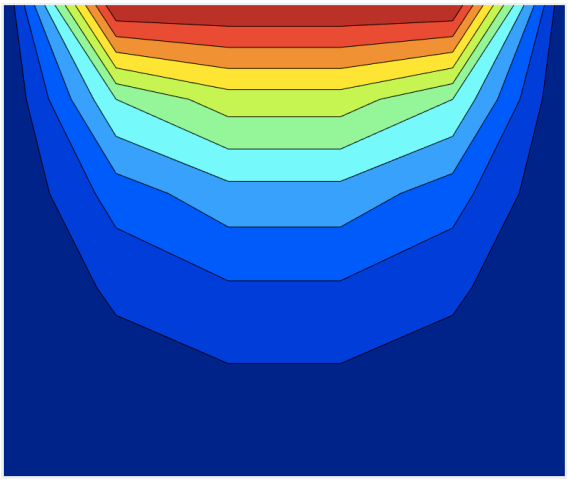
The auxiliary data affects the accuracy of the estimation scheme via coKriging in estimating the boundary conditions for all fine-resolution subdomains. The results of total and time-averaged RMS error for different auxiliary data are shown in Figure 3.15. As a reference case, the results of the gappy 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 Figure 3.13. The auxiliary data with finer resolution gives the lower RMS error at the steady state. However, at the transient region ($t \leq 10$), the case with no auxiliary data with Kriging leads also to a good reduction of RMS error. This is because all fine-resolution subdomains 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.

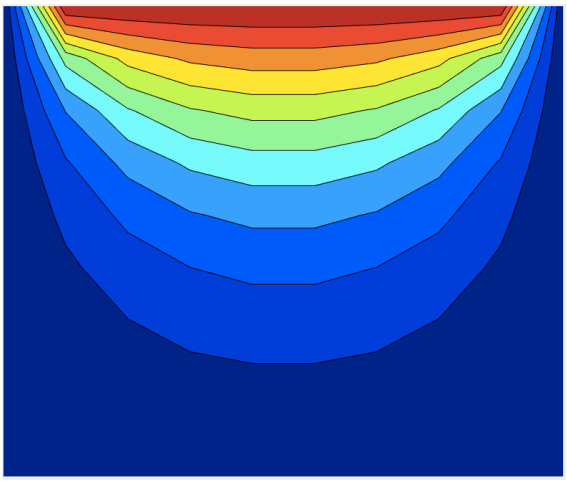
In the simulation of the Navier-Stokes equations, the different resolutions of auxiliary data, see Figure 3.14, show the relatively bigger effect of the auxiliary data



(a) 4×4 grid.

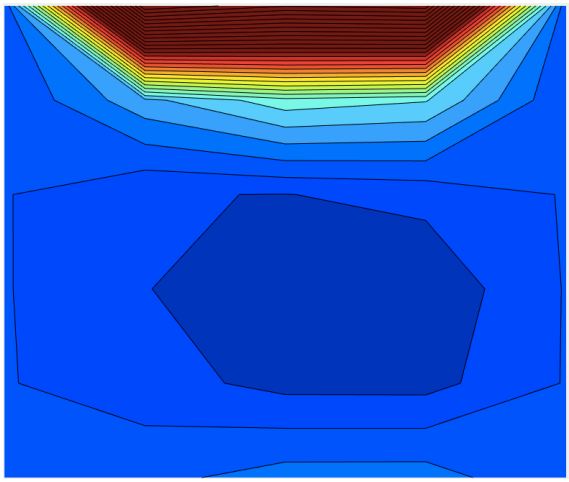


(b) 6×6 grid.

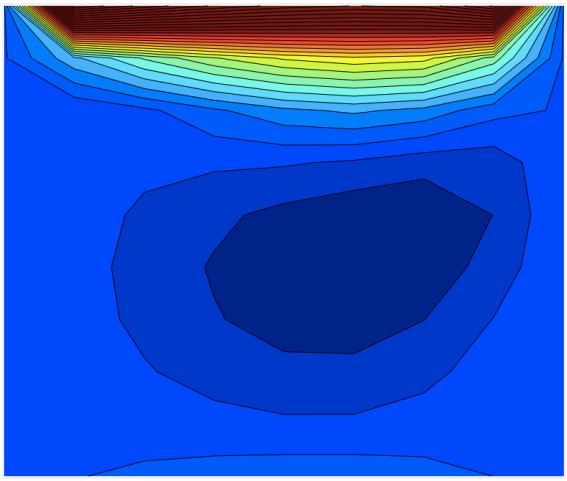


(c) 10×10 grid.

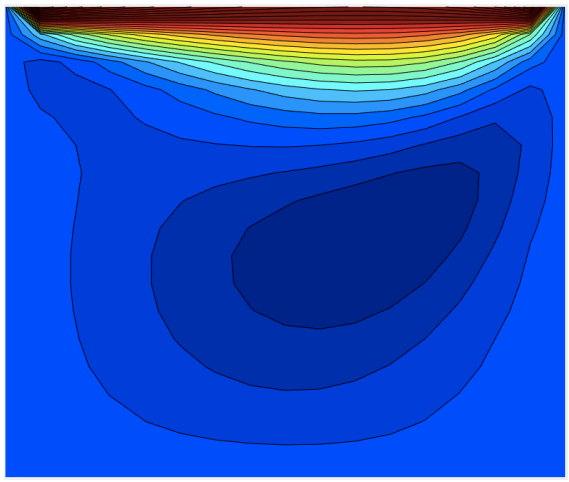
Figure 3.13: Temperature contours for the heat equation by different auxiliary data from coarse grids.



(a) 4×4 cell.



(b) 8×8 cell.



(c) 16×16 cell.

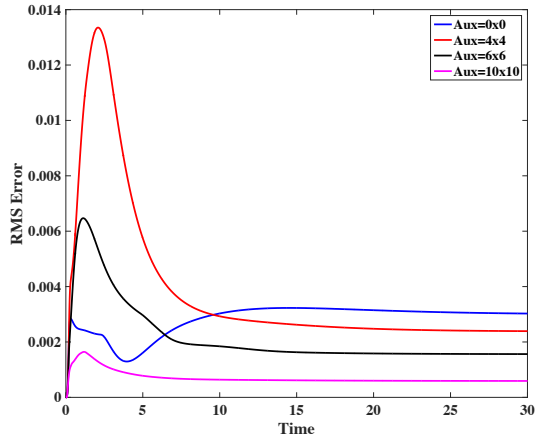
Figure 3.14: Streamwise velocity contours for the Navier-Stokes equations by different auxiliary data from coarse grids.

compared to any other tested parameter. Because of the weak spatial correlations of the solution between the fine-resolution subdomains, 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 Figure 3.14(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.

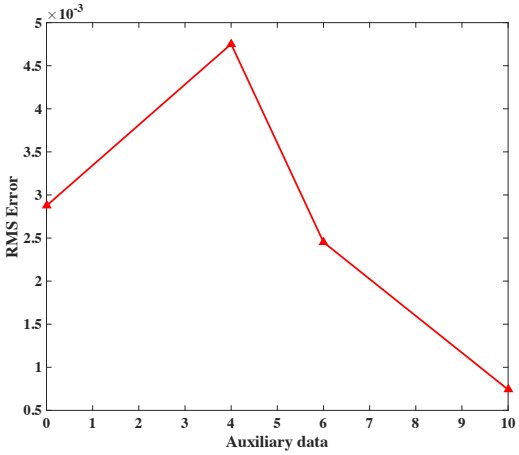
3.4.4 Non-interaction timestep number (τ)

Employing a non-interaction timestep number, τ, as an independent parameter allows us to minimize communications by performing independent parallel runs during time $\tau \cdot \Delta t$. From the multiscale modeling perspective, $\tau \cdot \Delta t$ can represent a "macro" time step as compared to a micro time step, Δt. The results of total and time-averaged RMS error with different τ are shown in Figure 3.15. 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.

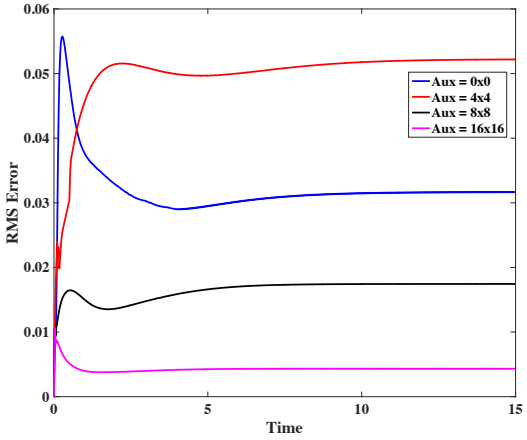
In the heat equation, we have observed that the optimal τ depends on the penetration length for diffusion – the measure of how long each field variable can penetrate into the neighborhood numerically during a unit time step. Generally, the



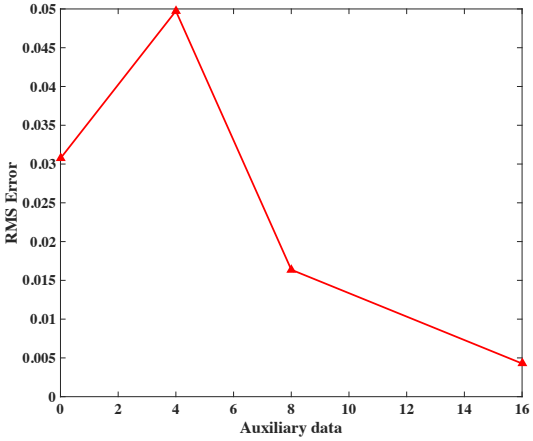
(a) Time history of total RMS error in the heat equation.



(b) Time-averaged RMS error in the heat equation.

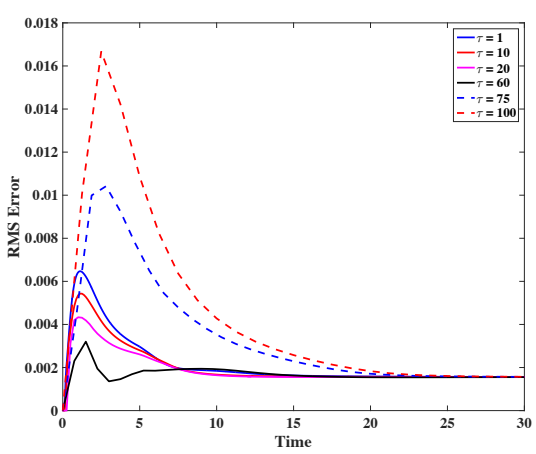


(c) Time history of total RMS error in the Navier-Stokes equations.

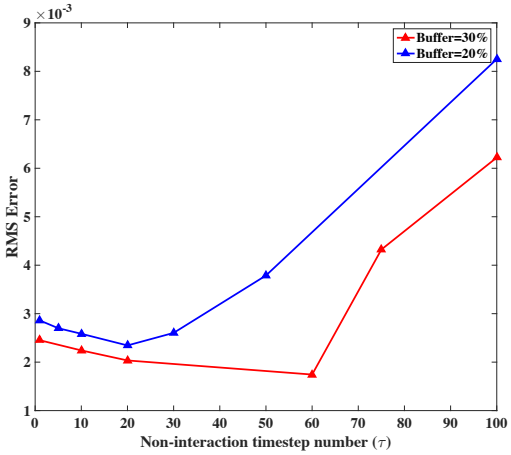


(d) Time-averaged RMS error in the Navier-Stokes equations.

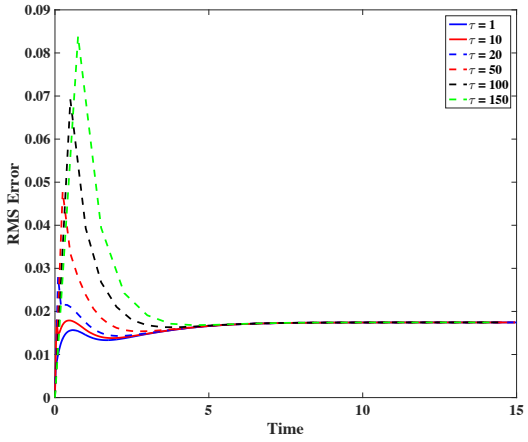
Figure 3.15: The time history of time-averaged and total RMS errors for different auxiliary data. In the heat equation, fixed parameters are as follows: size of buffer at 30% and non-interaction timestep number (τ) of 1. In the Navier-Stokes equations, the fixed parameters are as follows: size of buffer at 25% and with non-interaction timestep number (τ) of 5.



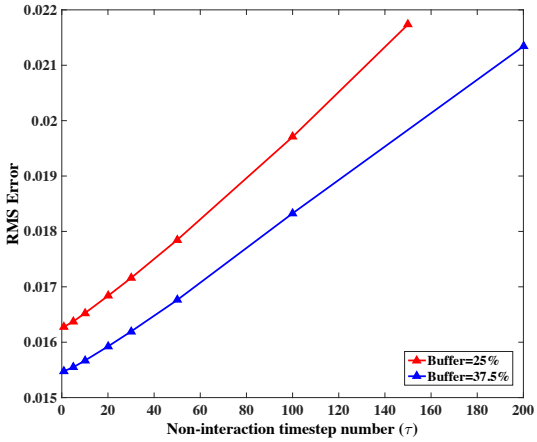
(a) Time history of total RMS error in the heat equation with a 20% buffer length.



(b) Time-averaged RMS error in the heat equation.



(c) Time history of total RMS error in the Navier-Navier-Stokes equations with a 25% buffer length.



(d) Time-averaged RMS error in the Navier-Stokes equations.

Figure 3.16: Time history of time-averaged and total RMS errors for different non-interaction timestep number (τ). 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 6×6, and buffer of 30%. In Navier-Stokes equations, (c) and (d), the fixed parameters are as follows: auxiliary data 8×8, and buffer of 25%.

penetration length for the diffusion can be estimated by

$$l_{p,d} \sim \sqrt{\kappa \cdot \tau \Delta \mathrm{t}}. \tag{3.14}$$

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

$$\tau_d = \frac{l_b^2}{\kappa \Delta \mathrm{t}}. \tag{3.15}$$

First, we obtain τ_d for a 30% and a 20% buffer, which is about 64.8 and 28.8, respectively. As shown in Figure 3.16, if the selected value of τ is smaller than τ_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 τ becomes larger. On the other hand, if we choose τ bigger than τ_d, the error from the local boundary can penetrate the fine-resolution subdomains and this would lead to the increase of RMS error. In the case of Navier-Stokes equations, we have an extra length scale associated with convection, namely

$$l_{p,a} \sim V \cdot \tau \Delta \mathrm{t}. \tag{3.16}$$

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

$$\tau_v = \frac{l_b}{V \Delta \mathrm{t}}, \tag{3.17}$$

where V is the maximum streamwise velocity. Finally, the allowable non-interaction

timestep number, τ_a is the minimum between these two different non-interaction timestep numbers as follows:

$$\tau_a = \min\{\tau_d, \tau_v\}. \tag{3.18}$$

In this simulation, τ_a for a 25% buffer is around 15. However, as shown in Figure 3.14, the RMS error increases linearly as τ increases. This result shows that the RMS error in the Navier-Stokes equations appears to be independent of the allowable non-interaction timestep number τ_a. The reason is that it is hard to find optimal τ_a 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 τ is the best with respect to RMS error.

3.5 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, τ. 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 (τ)***: In the heat equation (only diffusion), near the *allowable* τ_d, 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 τ (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 next chapter, 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. We will also address the *spatio-temporal* gappy simulation with gaps both in time and space.

Chapter Four

Resilient and efficient simulations based on multi-fidelity and heterogeneous information fusion

4.1 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., [27, 101]. However, exascale simulations, which use massive numbers of processors, require not only an efficient algorithm but also a fault-resilient algorithm to address traditional open issues [14]. 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 [83, 30, 119, 60] and mathematical algorithms [110, 114, 67, 62]. 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 [66, 97].

This chapter is motivated by the approach introduced by Lee *et. al.*, which demonstrably achieved fault-resilience when solving PDEs and processor failures resulting 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 [63]. 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 and used in Sirisup *et. al.* in a CFD context, demonstrating computational acceleration by an equation-free POD-assisted

approach [99]. However, choosing appropriate projection steps is still an open issue. In this chapter, learning from the auxiliary data via diffusion maps provides a dynamics-informed temporal discretization, which can lead to an acceleration of the computation in time.

The original concept of the patch simulation comes from "patch dynamics" [53, 90], which can predict 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 can help to "fill-in" the missing data in space [85, 86, 80, 82]. The other method is diffusion maps, which can provide lower-dimensional information to estimate the appropriate projection time [22, 20, 74, 19]. In this chapter, 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 [76, 113, 108]. In this chapter, 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.

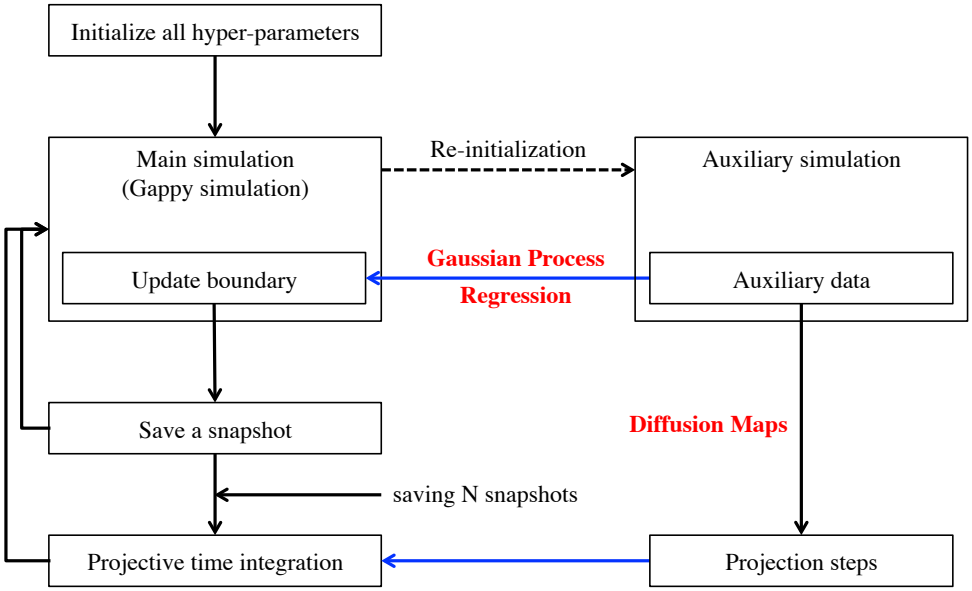
This chapter is organized as follows: In section 2, we introduce a general patch

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.

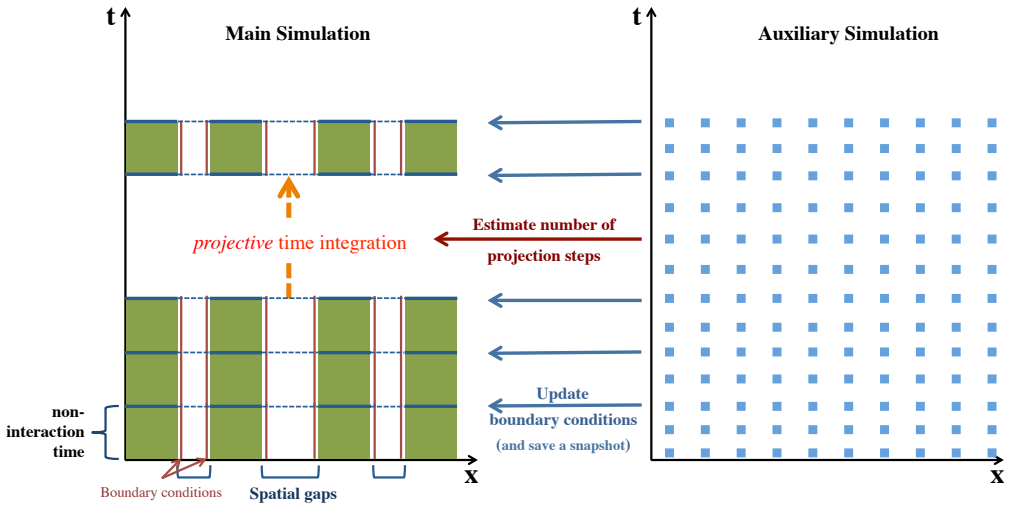
4.2 The Patch Simulation Framework

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 subdomains 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 [63]), 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. We can then 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 [99], 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 ac-



(a) A flow chart of the patch simulation.



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

Figure 4.1: 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 a jump size (projection steps) 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 a jump size (colored by red).

celerate the overall simulation. In this chapter 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 times, 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, 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 can be parameterized by a few POD basis functions [29]. Then, we can 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 can 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 [53, 90]. The projection time represents a macro time scale for dynamics on the slow manifold while detailed computation on the fine-resolution subdomains represents the micro (fine)-level of the description of the dynamics.

A flow chart of the patch simulation is shown in Figure 4.1. A cycle of the main simulation with fine-resolution subdomains is described below (see reference [63] for details). We first check spatial gaps due to computational faults and choose a buffer size for each fine-resolution subdomain. Next, we estimate the local bound-

ary condition of each subdomain with auxiliary data and obtain the solution (in embarrassingly parallel or EP) during a macro timestep, called the non-interaction timestep. After each macro timestep, we save a snapshot of the field variables on fine-resolution subdomains. We repeat this cycle until we have "N" snapshots. After that, we employ the projective time integration scheme on each fine-resolution subdomain independently with an appropriate projection time. This constitutes one complete cycle of the patch simulation 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 4.6.3.

As shown in Figure 4.1 (a), the auxiliary data provides two different types of information (colored by blue) through two different statistical learning techniques (colored by red). As shown in Figure 4.1 (b), first is the global and spatial information of field variables with low accuracy, via the multi-level Gaussian process regression, for estimating local boundary conditions for the fine-resolution subdomains. Second is the temporal information of a projection size, via diffusion maps, for the projective time integration. Details of these statistical learning techniques are described below.

4.3 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 subdomains, 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".

4.3.1 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 subdomain. In this framework we have 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, can be modeled by two different Gaussian Processes, Z and Z_a, respectively. Then, the auto-regressive scheme of Kennedy and O'Hagan [52] can be written as follows:

$$Z(x) = \rho(x)Z_a(x) + \delta(x), \tag{4.1}$$

where ρ is a scaling parameter, which represents a correlation between two Gaussian processes, and $\delta(x)$ is a bias Gaussian field, which is independent of Z and Z_a. $\hat{y}(x^*)$ is a predicted field variable, i.e., temperature or velocities, at the local boundary x^* of each subdomain. Predicted field variables can be obtained by the following equation:

$$\hat{y}(x^*) = \mu + \rho\hat{y}_a(x^*) + r^T \left(R + \sigma_\epsilon^2 I\right)^{-1} \left[y(x) - \mathbf{1}\mu - \rho\hat{y}_a(x)\right], \tag{4.2}$$

$$\hat{s}^2(x^*) = \rho^2 \hat{s}_a^2(x^*) + \sigma^2 \left[1 - r^T \left(R + \sigma_\epsilon^2 I \right)^{-1} r + \frac{\left[1 - r^T \left(R + \sigma_\epsilon^2 I \right)^{-1} r \right]^2}{\mathbf{1}^T \left(R + \sigma_\epsilon^2 I \right)^{-1} \mathbf{1}} \right], \quad (4.3)$$

where $\{\mu, \sigma^2\}$ and $\{\mu_a, \sigma_a^2\}$ represent means and variances of Gaussian processes of Z and Z_a, respectively. $R = \kappa(x, x', \theta)$ 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 for Z. The correlation kernel, $\kappa(x, x', \theta)$, is constructed by the Matérn kernel [103] as follows:

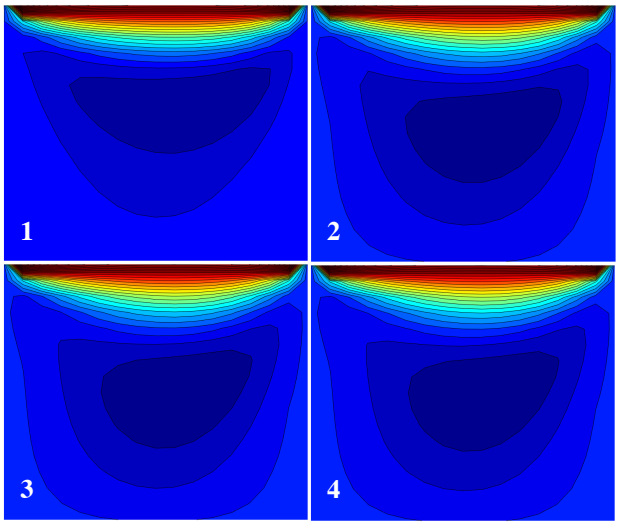
$$\kappa(x, x', \theta) = \left(1 + \sqrt{3}\theta \mathrm{d}(x, x') \right) \exp\left(-\sqrt{3}\theta \mathrm{d}(x, x') \right), \quad (4.4)$$

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

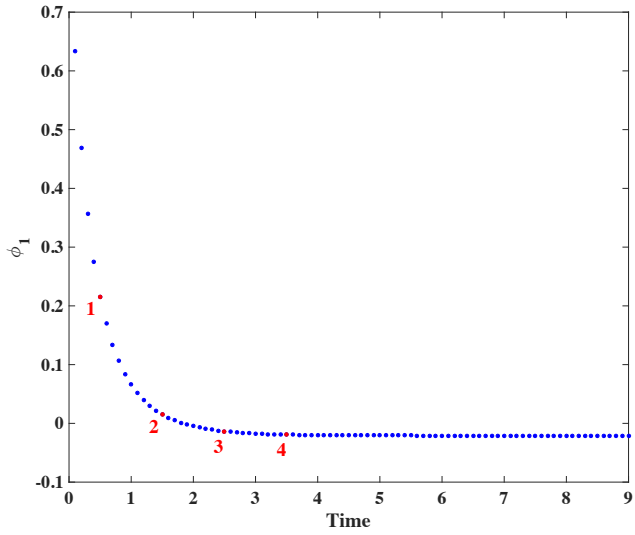
An optimal hyperparameter set, $\{\mu, \sigma, \sigma_\epsilon, \rho, \theta\}$, can be obtained by maximum likelihood estimation (MLE) from the aforementioned data sets. For details of the formulation and hyper-parameters, see reference [82].

4.3.2 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 [20, 74, 19]. As shown in Figure 4.2, diffusion maps obtained by temporal snapshots of field variables can 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 jump size for the projective time



(a) Snapshots of the streamwise velocity.



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

Figure 4.2: 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).

integration. The diffusion distance, $D(\cdot,\cdot)$, between adjacent snapshots on a one dimensional slow manifold can be calculated by the following equation:

$$D^2(t_{i+1},t_i) = \lambda_1^2\left[\phi_1(t_{i+1}) - \phi_1(t_i)\right]^2, \tag{4.5}$$

where (λ_1,ϕ_1) represents the first nontrivial eigenvalue and eigenvector of the discrete Laplace-Beltrami operator, which is a generalized Laplace operator on the Riemannian manifold, via a diffusion kernel, W, like the following:

$$W_{i,j} = \exp\left(-\frac{\|y_i - y_j\|^2}{\epsilon^2}\right), \tag{4.6}$$

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 jump size 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 jump size at time t_i for the projective time integration can be generalized to an "adaptive timestep refinement" by the following equation:

$$J(t_i) = \alpha \log \frac{D_m}{D(t_{i+1},t_i)}, \tag{4.7}$$

where α is a tuned parameter and D_m represents the maximum diffusion distance along the time series. Since the jump size should be an integer, $J(t_i)$ is rounded

to the nearest integer. Then, we can project and estimate field variables at time $t^* = t + J(t) \cdot \tau \cdot \Delta t$ by a POD assisted projection, where τ is a non-interaction timestep number (see reference [63]).

In this formulation, there is no jump (no projection) when the diffusion distance is the maximum, while there is the biggest jump when the diffusion distance is the minimum. There is a trade-off between the computational efficiency and accuracy via the tuned parameter α. For example, if α is bigger than 1, there is an efficiency gain (proportional to α) but a loss of accuracy due to the long time projection. In this framework, α is set to 1 as a default.

4.4 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 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.

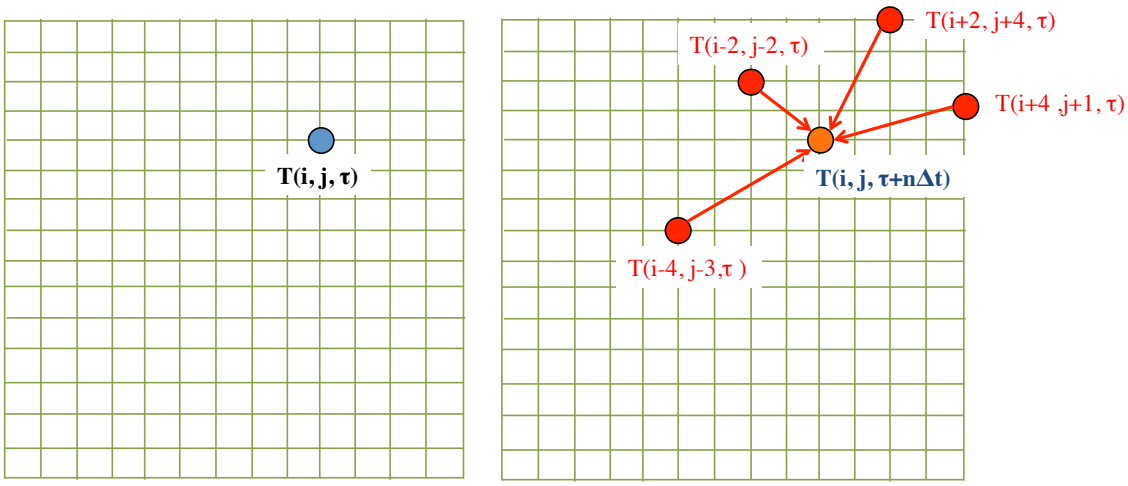


Figure 4.3: A schematic illustration of a random walk model for the two-dimensional heat equation: the temperature T at the node (i,j) at the next time $t + n\Delta t$ (colored orange) is obtained by averaging the field variables over all sample paths that visited (i,j) at time $t + n\Delta t$ (colored red) by the Monte Carlo method.

4.4.1 Two-dimensional random walk model for the heat equation

The two-dimensional heat equation in a uniform rectangular grid can be 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:

$$T(x^*, \tau + n\Delta t) = \oint T(x, \tau) d\mu(x) \quad \forall x \in D, \tag{4.8}$$

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 4.3, the continuous model can be rewritten as a two-dimensional 10×10 grid discrete model with N sample paths (particles) of the Monte Carlo method as:

$$T(x^*, \tau + n\Delta t) = \sum_{x \in D} T(x, \tau) P(x), \tag{4.9}$$

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

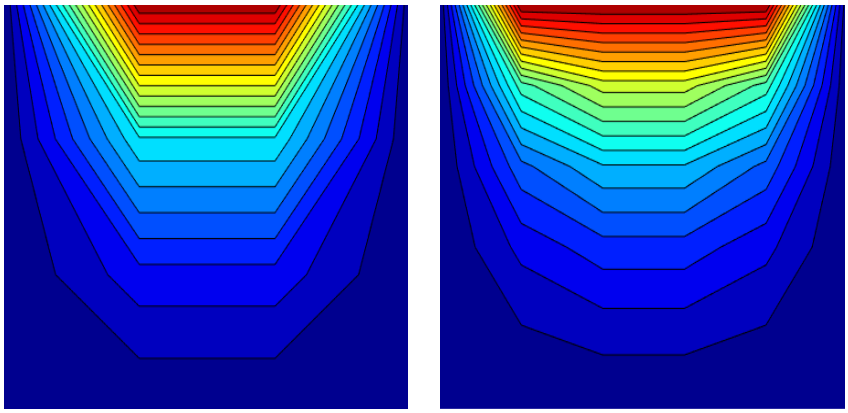
$$P(x) = \frac{n}{N}, \tag{4.10}$$

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

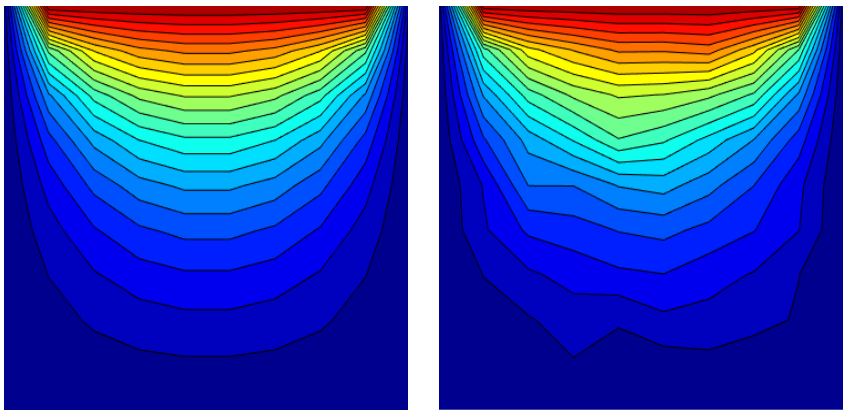
In this work, we choose 20, 200 and 2000 sample paths at each grid point (totaling 2000, 20000 and 200000 computations, respectively) for calculating the updated temperature at each point through equations (4.9) and (4.10). By the equivalent numerical discretization corresponding to the random walk model, 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.

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

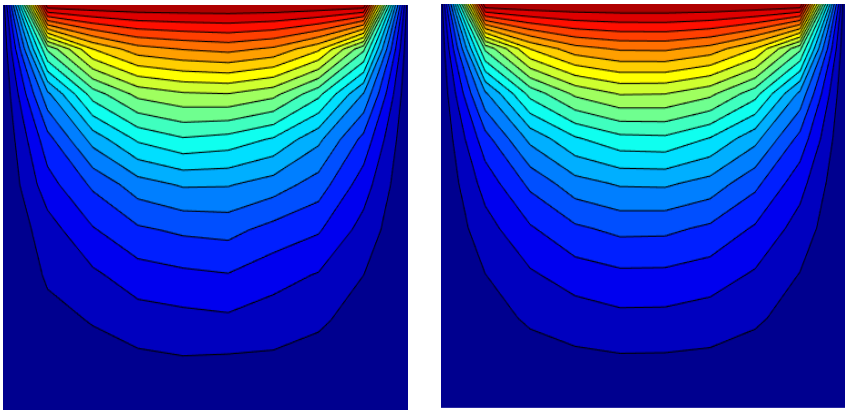
Next, we introduce a dissipative particle dynamics (DPD) model, a particle-based mesoscale simulation, as auxiliary data-producing simulator for the Navier-Stokes equations. The DPD method uses virtual particles, which represent "molecular clus-



(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 4.4: 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.

ters" moving together in Lagrangian fashion. In DPD systems, we assume that particles 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 ($\mathbf{F}_{ij}^{R}$) as follows:

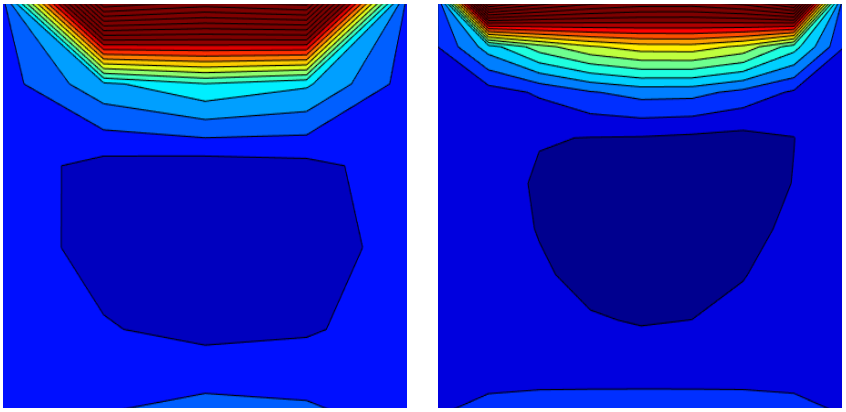
$$\mathbf{F}_{ij}^{C} = F_{ij}^{C}(r_{ij})\mathbf{r}_{ij}, \tag{4.11}$$

$$\mathbf{F}_{ij}^{D} = -\gamma\omega^{D}(r_{ij})(\mathbf{v}_{ij}\cdot\mathbf{r}_{ij})\mathbf{r}_{ij}, \tag{4.12}$$

$$\mathbf{F}_{ij}^{R} = \sigma\omega^{R}(r_{ij})\xi_{ij}\mathbf{r}_{ij}, \tag{4.13}$$

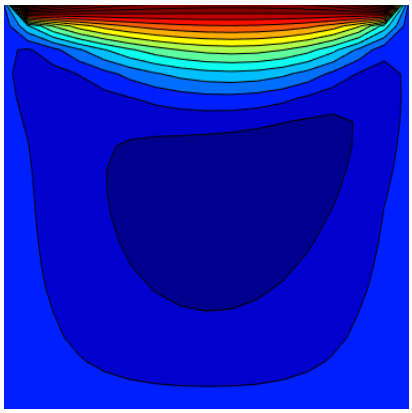
where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$, r_{ij} is a distance between i and j particles; γ and σ are hyper-parameters for dissipative and random force, respectively; ω^D and ω^R are weight functions, and ξ_{ij} is a normally distributed random variable. For details of DPD systems, see reference [34].

In this chapter, 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 [5]. 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 time, the DPD results are averaged by every 1000 time steps with $t = 0.1$. 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 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 4.5.

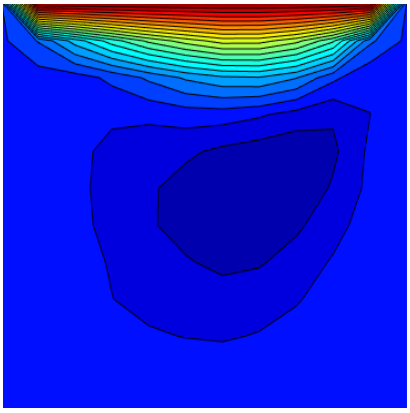


(a) 4×4 grid.

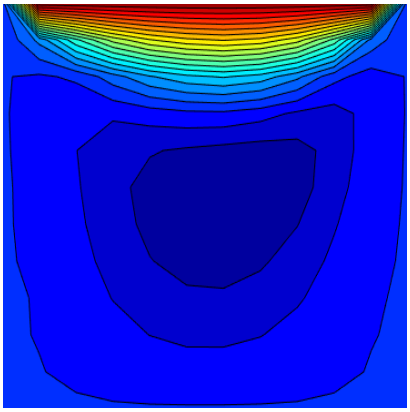
(b) 8×8 grid.



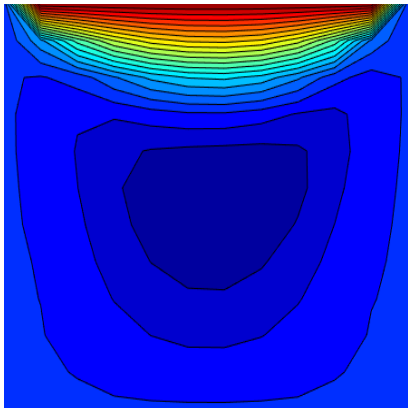
(c) 16×16 grid.



(d) 1000 DPD particles.



(e) 4000 DPD particles.



(f) 8000 DPD particles.

Figure 4.5: 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.

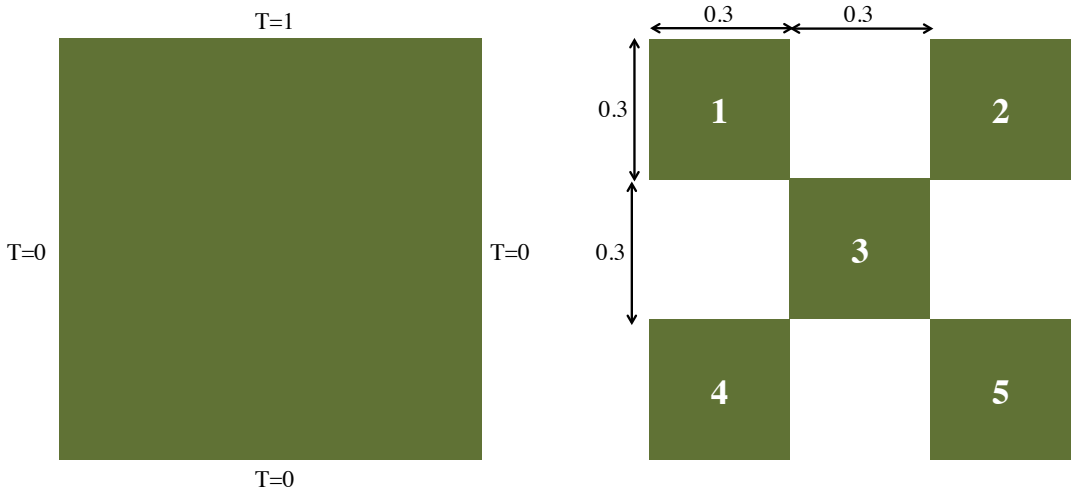
4.5 Simulation setup

4.5.1 Heat equation

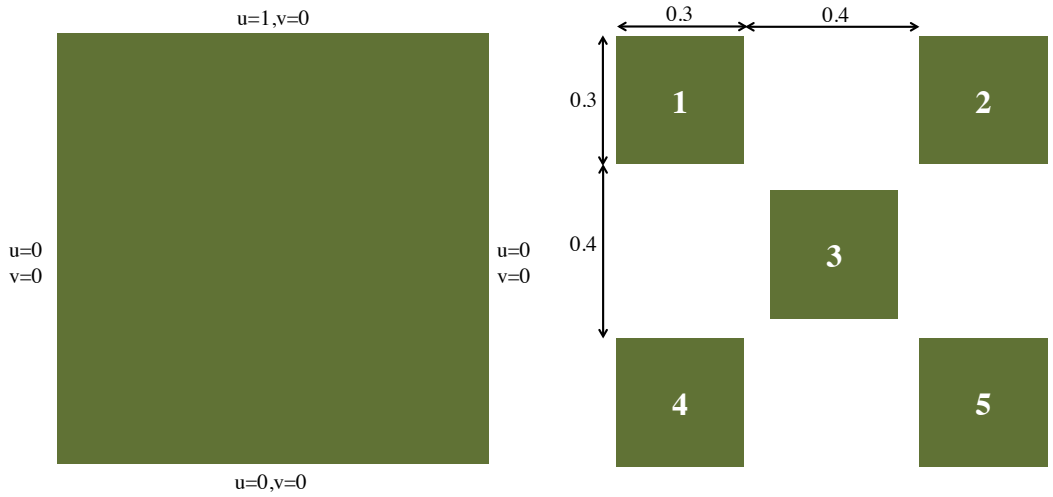
First, we solve the two-dimensional heat equation with diffusivity κ, given by:

$$\frac{\partial T}{\partial t} = \kappa \nabla^2 T. \tag{4.14}$$

The physical boundary conditions are described in Figure 4.6(a). The grid resolution of the patch simulation is 11×11 per each subdomain (total of 5 subdomains), 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. The temporal discretization corresponds to time step $\Delta t = 0.01$ 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).



(a) Heat equation.



(b) Navier-Stokes Equations.

Figure 4.6: A schematic illustration of gappy domains for two benchmark problems [63]: (left) the global (physical) boundary condition of the reference simulation. (right) the location and index of the fine-resolution subdomains colored by green.

4.5.2 Navier-Stokes equations

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

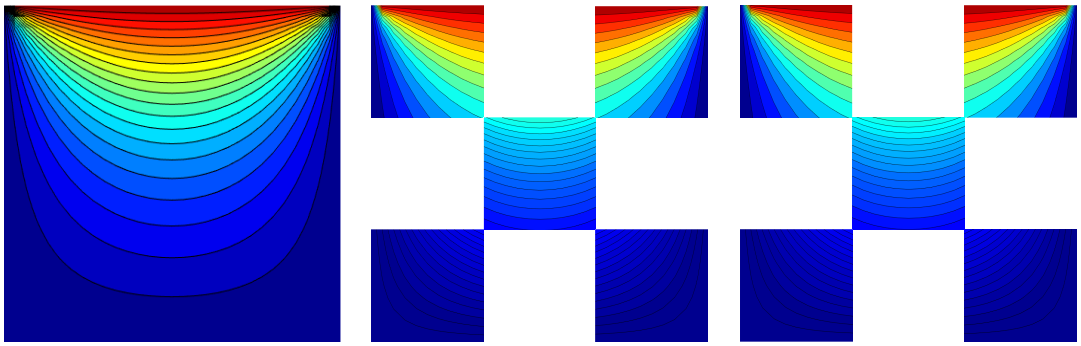
$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} + f, \tag{4.15}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{4.16}$$

where $\mathbf{v}$ is the velocity vector, p is the pressure, and ν is the kinematic viscosity of the fluid. For spatial discretization we employ a two-dimensional finite difference method. The physical boundary conditions are shown in Figure 4.6(b). The resolution of a rectangular cell for each subdomain 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. The time step is $\Delta t = 0.005$ with Reynolds number 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).

4.6 Results

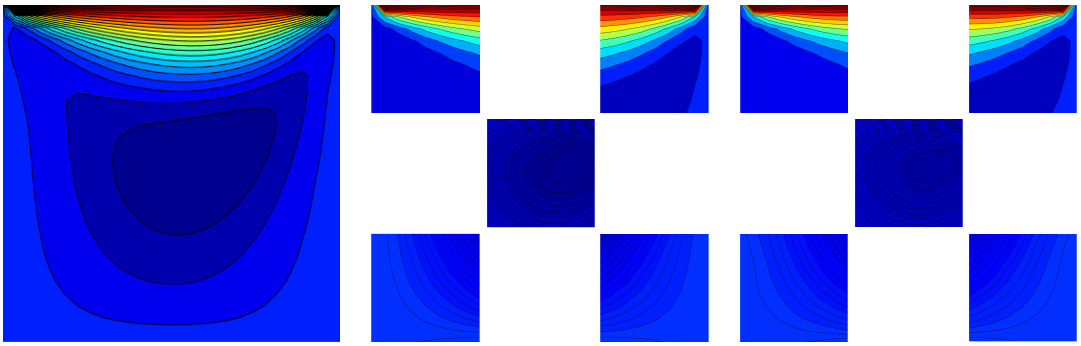
The contour plots of field variables compared to reference simulations (no spatial gaps or projective time integration) are shown in Figure 4.7. Also, the results of



(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.

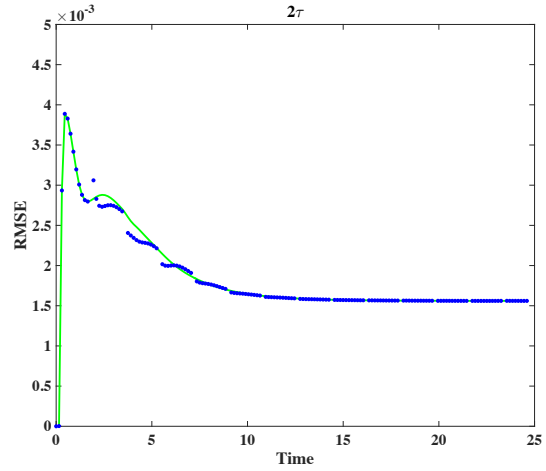


(d) Reference solution.

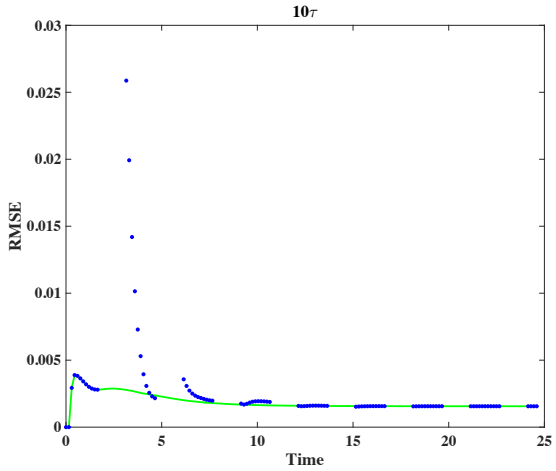
(e) Auxiliary data from a finite difference model with 8×8 grid resolution.

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

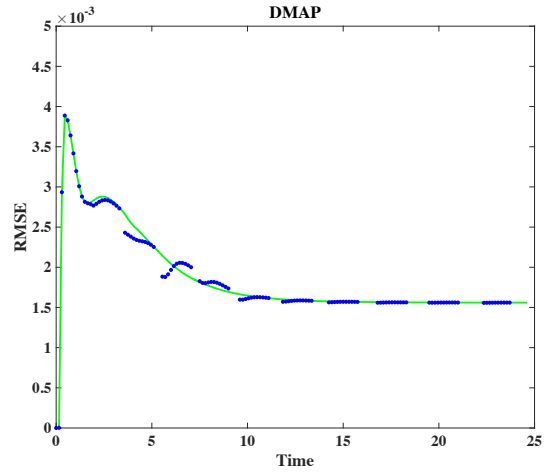
Figure 4.7: Contours of field variables at the steady-state. (a)-(c): Temperature contours for the heat equation. The reference simulation is obtained on the entire domain (31×31 grid). (d)-(f): Streamwise velocity contours for the Navier-Stokes equations. The reference simulation is obtained on the entire domain (27×27 cell). Results of patch simulations are based on fine-resolution subdomains with different auxiliary data.



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

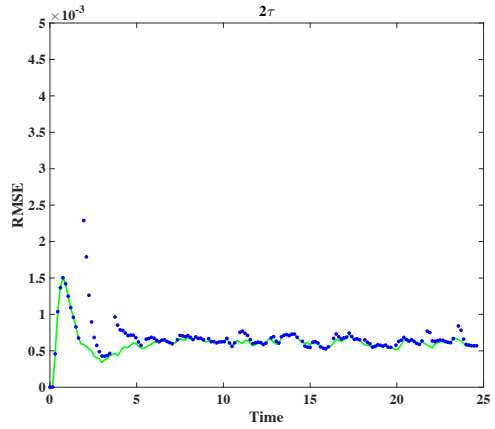


(b) Projection to $10 \cdot \tau \cdot \Delta t$.

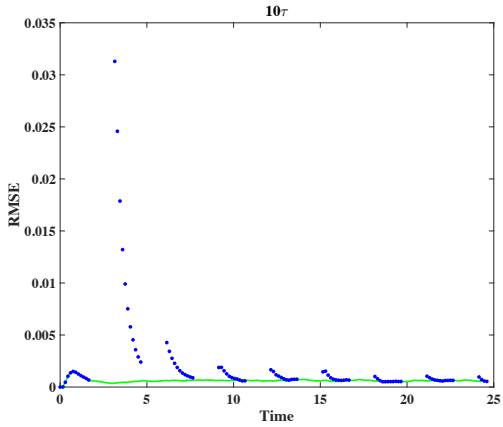


(c) Projection to $J(t) \cdot \tau \cdot \Delta t$.

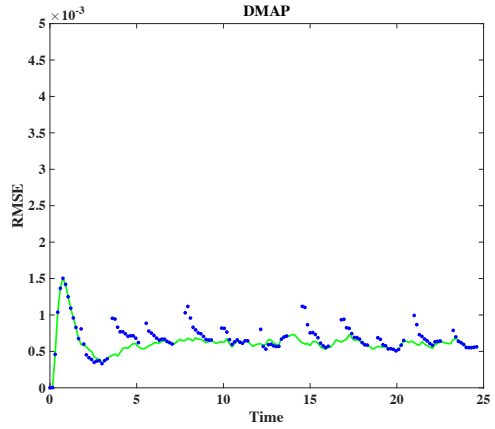
Figure 4.8: The time history of time-averaged RMS error for different jump sizes in the heat equation ((b) has a different RMSE scale). The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with different jump sizes. The auxiliary data comes from a finite difference model with resolution 6×6. We employ 10 snapshots in the projective time integration.



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

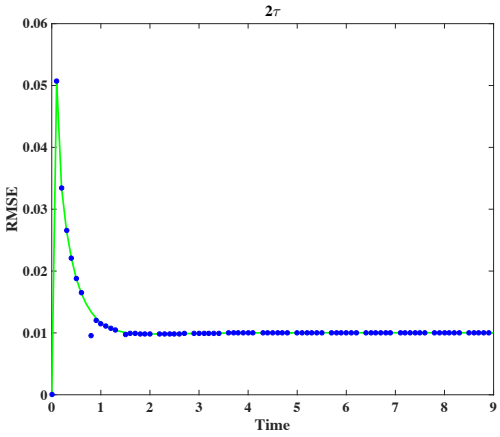


(b) Projection to $10 \cdot \tau \cdot \Delta t$.

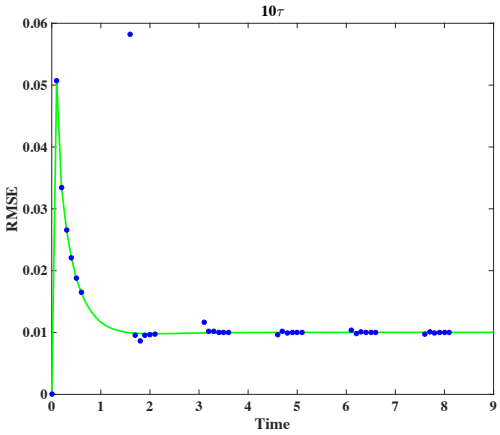


(c) Projection to $J(t) \cdot \tau \cdot \Delta t$.

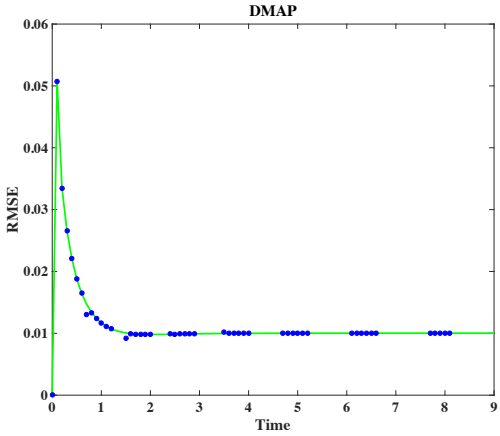
Figure 4.9: The time history of time-averaged RMS error for different jump sizes in the heat equation ((b) has a different RMSE scale). The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with different jump sizes. 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 $2 \cdot \tau \cdot \Delta t$.

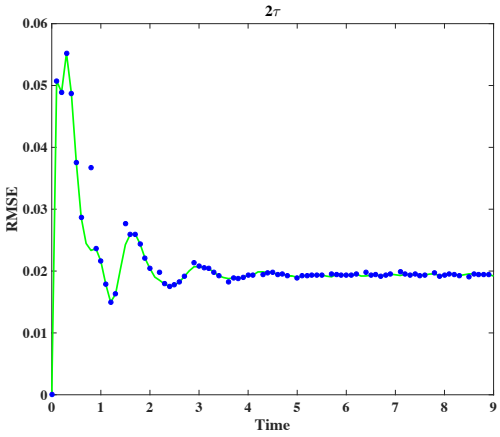


(b) Projection to $10 \cdot \tau \cdot \Delta t$.

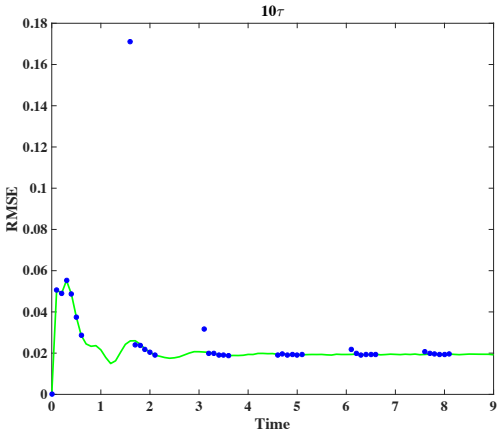


(c) Projection to $J(t) \cdot \tau \cdot \Delta t$

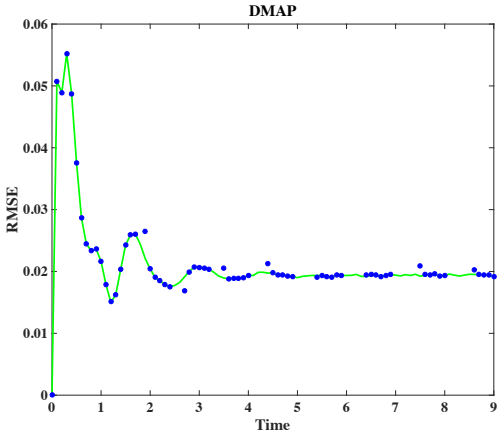
Figure 4.10: The time history of time-averaged RMS error for different jump sizes in the Navier-Stokes equations. The green line represents a solution of the gappy simulation, which has no projective time integration. The blue dots represent a solution of the patch simulation with different jump sizes. The auxiliary data comes from a finite difference model with resolution 8×8. We employ 5 snapshots in the projective time integration.



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



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



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

Figure 4.11: The time history of time-averaged RMS error for different jump sizes in the Navier-Stokes equations ((c) 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 different jump sizes. The auxiliary data comes from the DPD result with 4000 DPD particles. We employ 5 snapshots in the projective time integration.

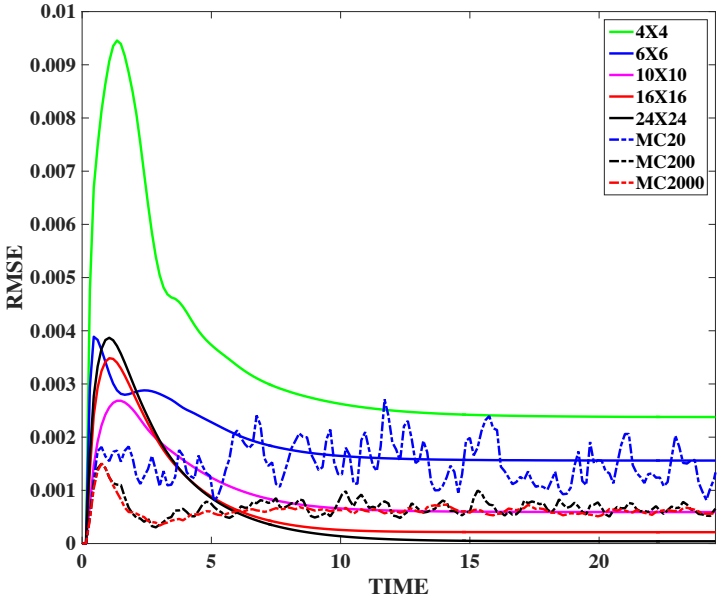
the patch simulation with different auxiliary data and jump sizes, compared to the results of the gappy simulation (no projective time integration), are shown in Figures 4.8 - 4.11. The green line represents the result of the gappy simulation, which has no projective time integration as in previous work [63]. The blue dots represent the result of the patch simulation, which employ projective time integration with different jump sizes.

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:

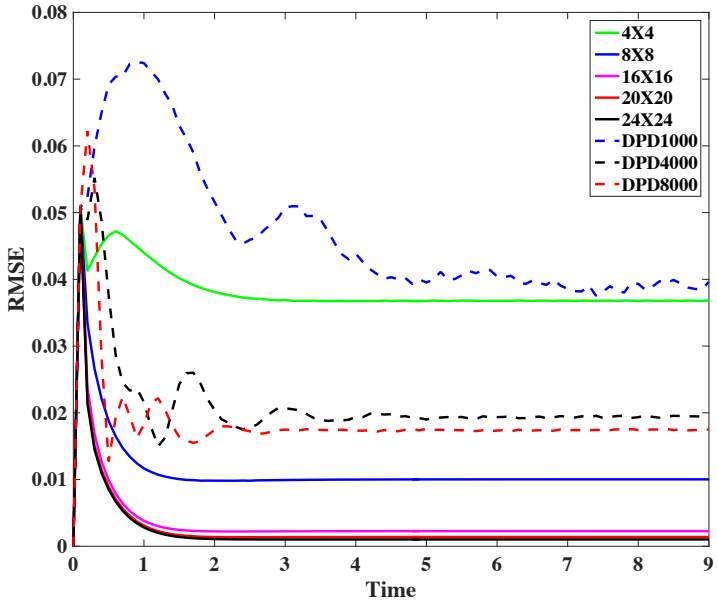
$$\mathrm{RMSE_T}(t) = \sqrt{\frac{1}{nN_g}\sum_{j=1}^{n}\sum_{i=1}^{N_g}(u_{r,j}(i,t) - u_{p,j}(i,t))^2}, \tag{4.17}$$

where N_g is the number of grid points of each fine-resolution subdomain and n is the number of fine-resolution subdomains, 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 subdomain, respectively.

As shown in Figures 4.8 - 4.11, for a small fixed jump size, 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 jump size, on the other hand, a large difference in $\mathrm{RMSE_T}$ is observed during the transient period. For a jump size given by diffusion maps, the gap between results of the gappy simulation and the patch simulation is small enough during the transient period (due to a small jump size), and the computation redundancy is also small enough near the steady-state (due to a large jump size).



(a) Heat equation.



(b) Navier-Stokes Equations.

Figure 4.12: RMSE_{T} for different auxiliary data. The RMS errors are converged as the accuracy of the auxiliary data increases in both cases.

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 4.12, the RMS errors converge as the accuracy of auxiliary data increases for both the finite difference method and the particle based method. Hence, we can choose a fine-resolution simulation as the reference gappy simulation for each problem – the finite difference method with a 24×24 grid resolution for both the heat equation and the Navier-Stokes equations. Then, the D_M is calculated by the following equation:

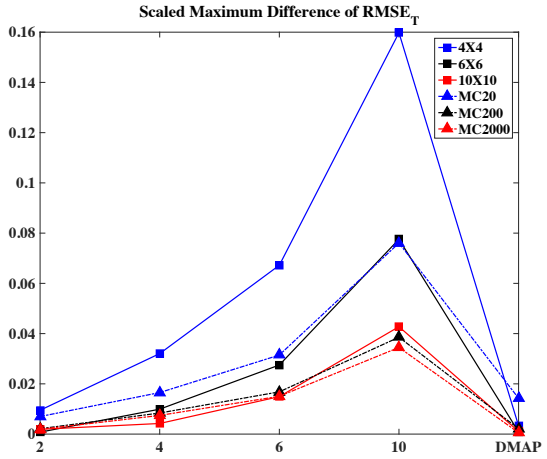
$$D_M = \max\left(|\mathrm{RMSE}_{\mathrm{T}}^{\mathrm{p}}(t) - \mathrm{RMSE}_{\mathrm{T}}^{\mathrm{g}}(\mathrm{t})|\right) \cdot R, \quad t \in (0, T), \tag{4.18}$$

where superscripts "p" and "g" represent the patch simulation and the gappy simulation, respectively, 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 can be 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)" as follows:

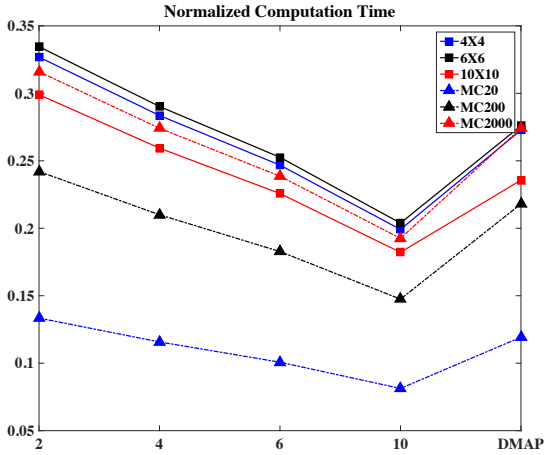
$$\zeta = 1 - (D_M \cdot T_C). \tag{4.19}$$

4.6.1 Accuracy versus efficiency

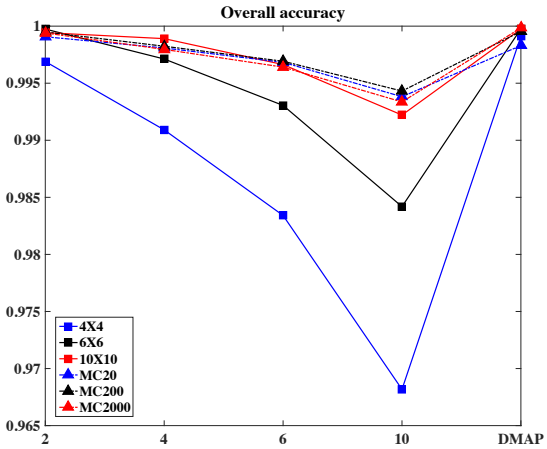
In the heat equation, for cases of fixed jump sizes only, T_C decreases linearly while D_M grows exponentially as the jump size increases (see Figure 4.13). Hence, a



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

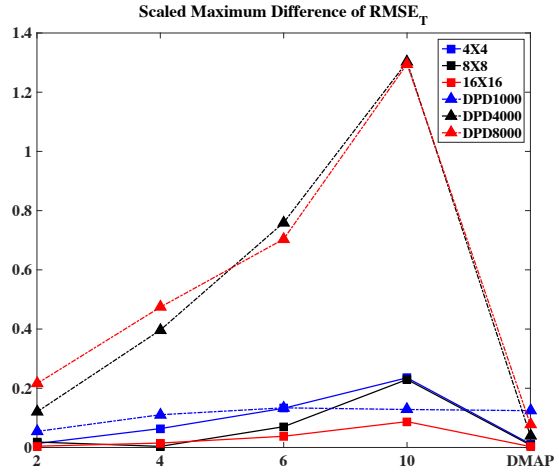


(b) Normalized computation time (T_C).

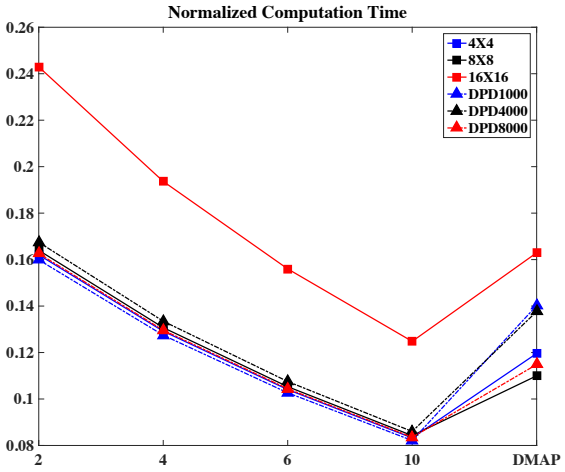


(c) Overall accuracy (ζ).

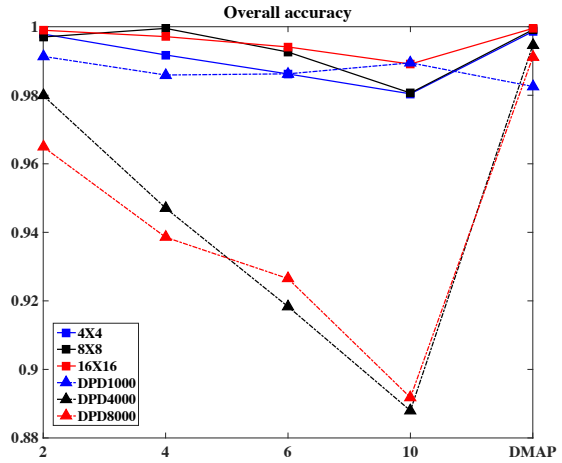
Figure 4.13: Three computational quality measures in the heat equation: the x-axis represents a fixed jump size for the projective time integration and "DMAP" represents a varying jump size by the diffusion maps. Square and triangle markers represent auxiliary data from a finite difference model with 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 4.14: Three computational quality measures in the Navier-Stokes equations: the x-axis represents a fixed jump size for the projective time integration and "DMAP" represents a varying jump size 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.

small jump size requires a large computation time linearly while a large jump size loses accuracy exponentially. That is, a small jump size has higher overall accuracy compared to a large jump size. However, varying the jump size by the diffusion distance gives a value for D_M that is similar to the value for a fixed jump size of 2, and a value for T_C that is similar to the value for a fixed jump size of 6. 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 the jump size 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 jump sizes with a finite difference method only, results are same as the heat equation, i.e., varying the jump size by the diffusion distance provides the highest overall accuracy; D_M is at a similar level as for a fixed jump size of 2, and T_C is at a similar level as for a fixed jump size of 4 or 6. However, the DPD model with 1000 particles (too coarse) cannot provide an adequate jump size due to its incorrect slow manifold, which results in a decrease in overall accuracy even though we employ the varying jump size. As the number of DPD particles increases from 1000 to 8000, the diffusion map provides the appropriate jump size, which leads to large improvement of both the accuracy and the efficiency (see Figure 4.14). 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

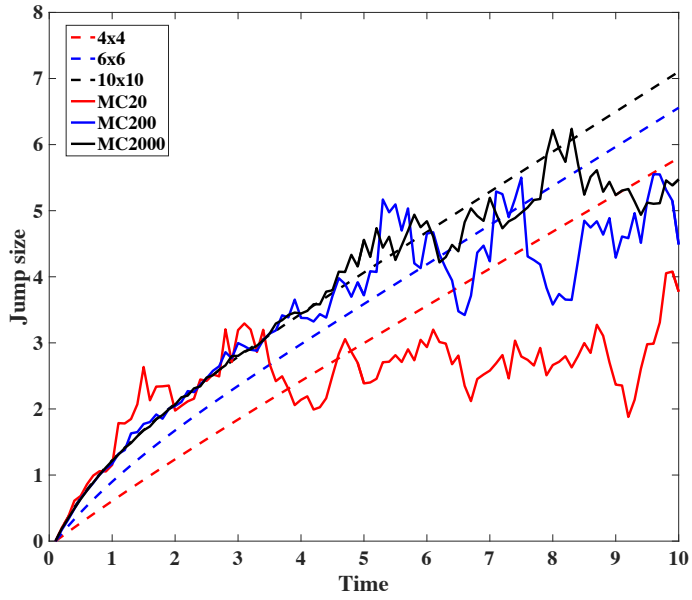
be the best auxiliary data with respect to the overall accuracy.

From the results of all simulations, we conclude that varying the jump size 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 can provide high quality even when using heterogeneous information fusion.

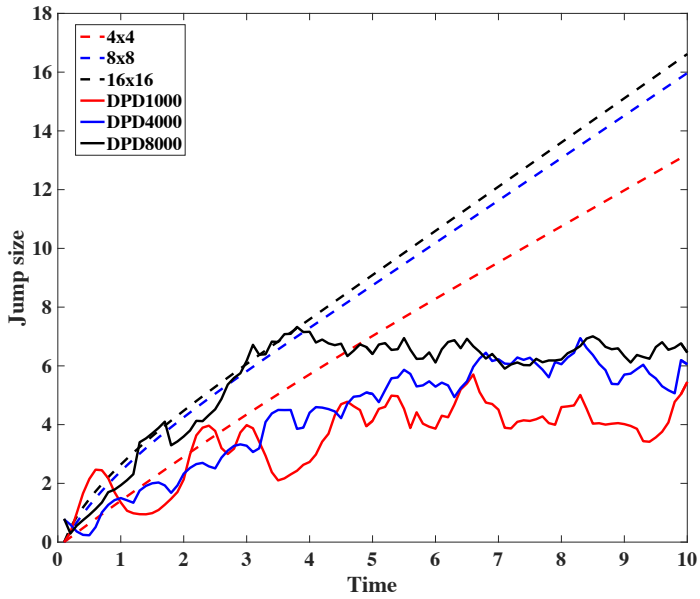
4.6.2 Jump size and auxiliary data

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 can draw a smooth coordinate which contains slow dynamics precisely and this results in increasing the jump size compared to inaccurate auxiliary data (see Figure 4.15). 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 the multi-level Gaussian process regression. For example, finite difference methods with high accuracy have more data points, which results in consuming extra computation time to obtain optimal hyper-parameters 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 the jump size of heterogeneous models (stochastic and particle-



(a) Heat equation.



(b) Navier-Stokes equations.

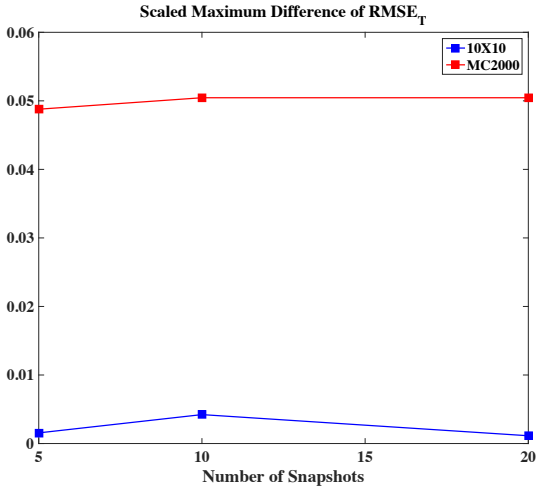
Figure 4.15: The jump sizes with respect to time given by the diffusion maps . The dashed lines represent the jump sizes for the finite difference model in both cases. The solid lines represent the jump sizes for the random walk model in the heat equation and the DPD model in the Navier-Stokes equations, respectively.

based models). As shown in Figure 4.15, the jump size 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, the jump size (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 the jump size near the steady-state in both benchmark problems.

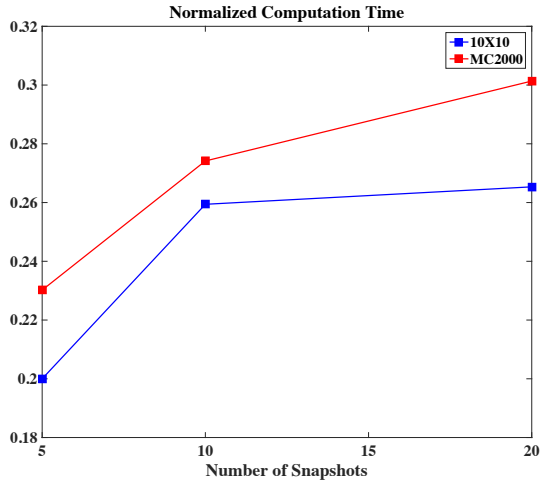
4.6.3 Number of snapshots

The number of saved snapshots, N, determines the number of terms of a POD expansion for reconstructing field variables in the projective time integration. If we use more snapshots for the projection with the same auxiliary data and same jump size, the accuracy of the patch simulation can be increased while the extent of simulation "saved" becomes less, leading to less efficiency.

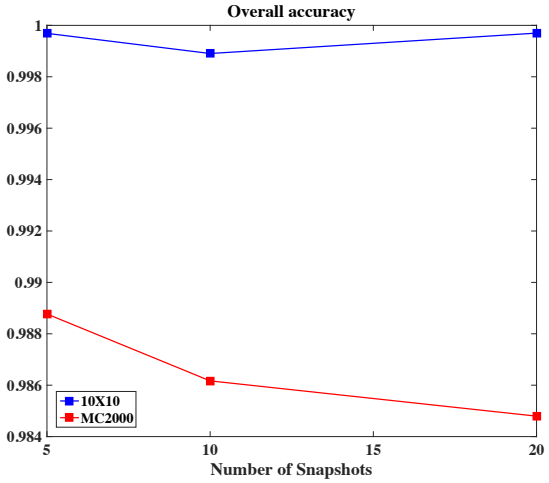
In the heat equation, as shown in Figures 4.16, accuracy gains with respect to D_M are very small even though T_C increases. This result shows that 5 snapshots, i.e., 5 terms of POD expansion, are enough to reconstruct and estimate field variables of the heat equation we setup. On the other hand, for the Navier-Stokes equations (which are much more complex than the heat equation), D_M decreases as the number of snapshots increases (see Figure 4.17(a)). 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 4.17(b)). From our observations, D_M has a



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

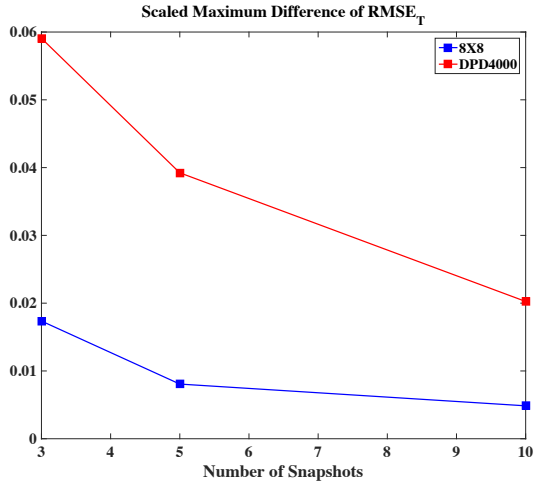


(b) Normalized computation time (T_C).

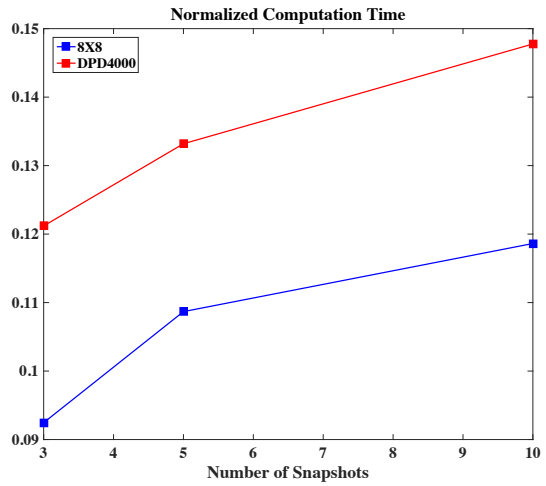


(c) Overall accuracy (ς).

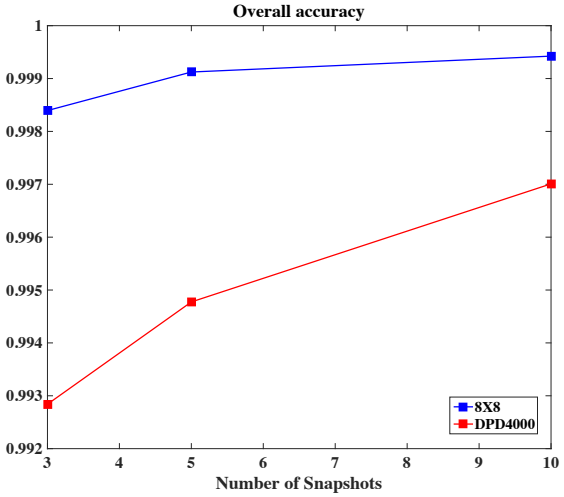
Figure 4.16: Results of three computational quality measures with different number of snapshots in the heat equation. x-axis represents the number of snapshots for the projective time integration. The blue and the red represent the coarse grid (10×10) and the random walk model with 2000 sample paths, respectively.



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



(b) Normalized computation time (T_C).



(c) Overall accuracy (ζ).

Figure 4.17: Results of three computational quality measures with different number of snapshots in the Navier-Stokes equations. x-axis represents the number of snapshots for the projective time integration. The blue and the red represent the coarse grid (8×8) and the DPD model with 4000 particles, respectively.

more pronounced effect than T_c on the overall accuracy. Hence, as shown in Figure 4.17(c), more snapshots are suitable for this problem setup.

4.7 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 can 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.

Chapter Five

Efficient simulations based on dynamics-informed projective time integration

5.1 Introduction

In exascale simulations, the acceleration of simulations is as important as robustness and resilience in previous chapters. Many researches have been developed in not only computer frameworks [10, 24] but also mathematical algorithmic side [8, 105, 75] to accelerate simulations. Among many applications of the exascale simulation, some physics problems can be accelerated by an adaptive timestep, if we know its dynamics "briefly". For example, a biological system where many species are mixed together [111, 118, 59, 13]. Each species has its own dynamics of concentration and thus corresponding evolution time are totally different depending on some parameters of physics. If we simulate this system with the smallest equidistant timestep for capturing all different evolution of dynamics effectively, the computational cost increases dramatically as fast as a number of species and it is impossible to solve the dynamical system within reasonable computation time.

In order to reduce the computation cost in this *multi-rate* dynamics problem, we extend the previous framework, "adaptive" projective time integration. First, we obtain a concentration of each species in time by a coarse simulator and find a dynamics-informed timestep for each species by Diffusion maps. Then, we perform the fine simulation with the aforementioned dynamics-informed timestep. In this case, we should perform coarse simulators as many as a number of species and this leads to additional computation coset. However, if concentrations of each species are highly related to other field variables, e.g., velocity fields (given from auxiliary data), we avoid performing coarse simulators to obtain adaptive timestep. For example, we put the velocity field data as the auxiliary data. Then, Diffusion maps for the velocity field provide a suggested timestep with a tuning parameter. If we find a relation between dynamics-informed timestep for velocity fields and concentration,

we can solve the multi-species problem without coarse simulators and this guarantees efficiency gain.

In this chapter, we introduce "one-way" coupled the transport equation with the Navier-Stokes equations. We demonstrate that the dynamics of different species can be parameterized by a physical-informed parameter and finally, we show this framework enhances the computational efficiency by the adaptive timestep for each species.

5.2 Simulation setup

A general *multi-rate* dynamics system has two governing equations. The first is a (scalar)"transport" equation for evolution of the concentration of each species (C), which consists of a convection and diffusion term as

$$\frac{dC}{dt} + \mathbf{u} \cdot \nabla C = \frac{1}{\mathrm{Pe}} \Delta C, \tag{5.1}$$

where Pe is a dimensionless number which is ratio of the convective transport rate to the diffusion transport rate as

$$\mathrm{Pe} = \frac{Lu}{\alpha} \tag{5.2}$$

where L represents a characteristic length, u and α represent a maximum velocity and a diffusivity of species, respectively.

In the transport equation, the convective term has a velocity vector field $\mathbf{u}$, which

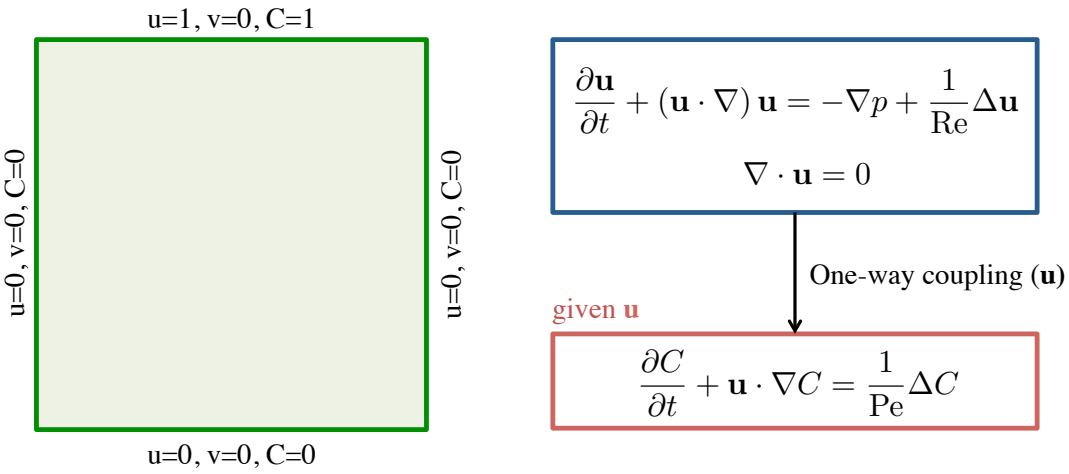


Figure 5.1: (left) The physical boundary condition for velocity fields and concentrations. (right) The Navier-Stokes equations provide the velocity fields ($\mathbf{u}$) to the transport equation.

is given by the Navier-Stokes equations as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u} + f, \tag{5.3}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{5.4}$$

where $\mathbf{u}$ is the velocity vector, p is the pressure, and ν is the kinematic viscosity of the fluid. In this framework, the Navier-Stokes equations are independent of the transport equation. Thus, this system is "one-way" coupled from the Navier-Stokes equations to the transport equation.

In order to perform a simulation, we employ a second-order finite difference method for both equations. The physical boundary conditions are imposed Dirichlet condition and shown in Figure 5.1. For the concentration, $C = 1$ at the top and $C = 0$ on all other sides and for the velocity fields, $u = 1$ and $v = 0$ at the top and $u = v = 0$ at all other sides. The computational domain consists of a structured rectangular mesh and their grid resolutions of the transport equation and the Navier-

Stokes equations are 20×20 and 8×8, respectively. The temporal discretization corresponds to timestep $\Delta t = 0.001$ with Peclet number, Pe = 1, 2, 5, and 10·Re – Peclet numbers are proportional to Reynolds number (Re) of driving flow. The simulation is integrated from t=0 to t=10 and the number of snapshot for projective time integration corresponds to 30 snapshots. For details about projective time integration, see the reference [99]. Due to one-way coupling system, we solve the Navier-Stokes equations first and obtain velocity fields at every time step. After that, we interpolate velocity fields on 8×8 grid to 20×20 grid for the transport equation.

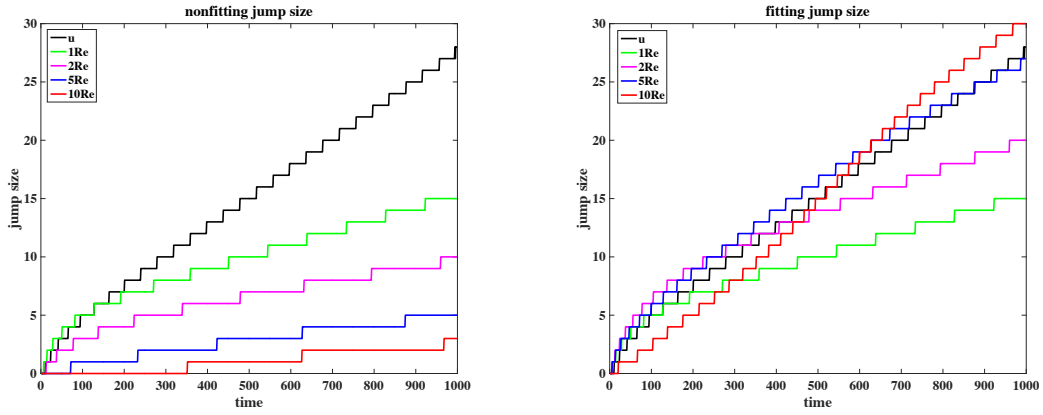
5.3 Results

First of all, in order to find "reference" projection time for different Peclet number, we solve all the transport equations for different Peclet number Pe= 1, 2, 5,and 10·Re with a same grid resolution. After that, we obtain "adaptive" projection time via the first nontrivial diffusion coordinate with 1000 temporal snapshots from $t = 0$ to $t = 10$ for each species. A projection time (jump size) $J(t_i)$ is obtained by

$$J(t_i) = \alpha \log \frac{D_m}{D(t_{i+1}, t_i)}, \tag{5.5}$$

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 can project and estimate field variables at time $t^* = t + J(t) \cdot \Delta t$ by a POD assisted projection, see [99].

First, Figure 5.2 (a) shows reference jump sizes for different Pe obtained by the



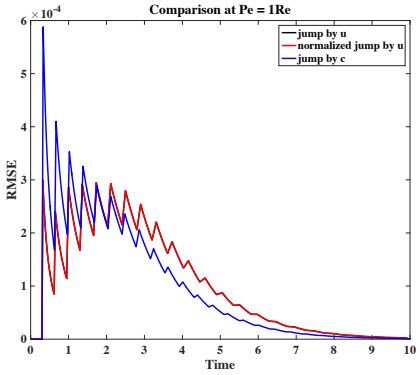
(a) Untuned jump size of each species.

(b) Tuned jump size of each species via tuned parameters $\alpha = 1/\mathrm{Pr}$.

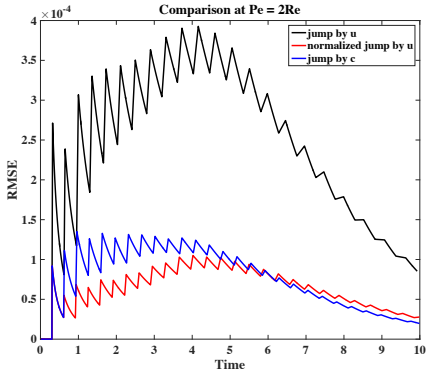
Figure 5.2: Jump sizes of each species with different Pe. The black line represents a jump size calculated by streamwise velocity field in the Navier-Stokes equations. (a): the original (untuned) jump size of each species. (b): all jump sizes are tuned by $\alpha = 1/\mathrm{Pr}$.

equation (5.5). The black line represents the adaptive jump size by snapshots of the streamwise velocity fields (auxiliary data). Each species has a different jump size at time t, which means that each species has different dynamics evolution. Hence, we should employ the smallest jump size at every jumps in order to capture all dynamics effectively. However, if there is correlation between jump sizes of each species and velocity fields (auxiliary data), we estimate the jump size of each species from the jump size of velocity fields, which is given.

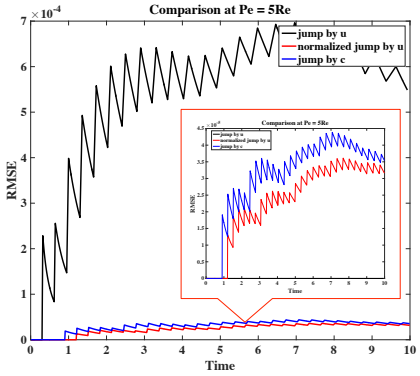
As shown in Figure 5.2 (b), if we employ a linear multiplier $\beta = \mathrm{Pe}/\mathrm{Re}$ in jump sizes of each species, then jump sizes of all species are well matched to the jump size of velocity fields. This β has a physical meaning, called a Prandtl number (Pr), which is ratio of viscous momentum of fluid to diffusivity of each species. Hence, we are able to choose a tuning parameter as $\alpha = 1/\mathrm{Pr}$ in the jump size of velocity fields and this results in huge computations saving because we eliminate the computational redundancy to solve coarse simulations for each species to obtain the adaptive jump size.



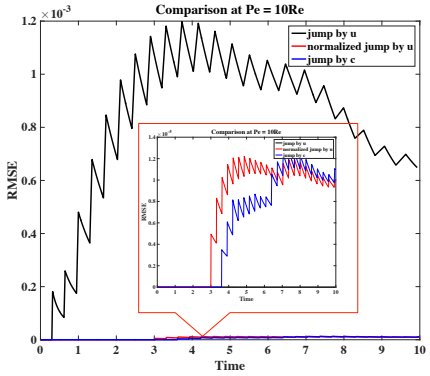
(a) RMSE at $\mathrm{Pe} = 1 \cdot \mathrm{Re}$



(b) RMSE at $\mathrm{Pe} = 2 \cdot \mathrm{Re}$



(c) RMSE at $\mathrm{Pe} = 5 \cdot \mathrm{Re}$



(d) RMSE at $\mathrm{Pe} = 10 \cdot \mathrm{Re}$

Figure 5.3: Temporal RMSE of each concentration in different Peclet number (Pe). The black, blue, and red lines represent jump size by velocity fields, velocity fields with tuned parameters, and each concentration, respectively. (a): the black line coincides with the red line because the tuned parameter $\alpha = 1$.

In order to show the capability of this framework with physical-informed tuned parameters $\alpha = 1/\mathrm{Pr}$, we compare RMS error of three different approaches in time: the jump size by snapshot of each concentration (the reference), velocity fields (untuned), and velocity fields tuned by physical-informed parameter α, see Figure 5.3. From results, RMS error from three different approaches are bounded in acceptable magnitude of error. Moreover, the RMS error from velocity fields tuned by physical-informed parameter is much smaller than the RMS error from untuned velocity fields and also comparable to the RMS error from the reference. Thus, in this framework, we are able to save the computational cost without performing a coarse simulator for each species.

5.4 Conclusions

We demonstrate the efficient algorithm for the multi-rate problem in the complex biological system. First, we demonstrate the projective time integration adaptively with appropriate projection time (jump size). Also, different but physically related field variables are able to give the appropriate jump sizes via physical-informed tuned parameter α. If multi-species are solved by different projective time integration asynchronously, then there is a huge computational cost saving in exascale simulations because of less communication between computer nodes, see chapter 7.

Furthermore, if we have time series data of concentrations but do not know the Peclet number, a tuned parameter α given by Diffusion maps provides the Prandtl number and finally provide the (unknown) physical parameter by data mining approaches. Hence, it shows the data are able to parameterize the dynamics intrinsically, without explicit reference to fundamental physical quantities [116].

Chapter Six

Robust nonlinear information fusion via manifold-driven Gaussian process regression

6.1 Introduction

Thanks to advance in the new paradigm of computer science, the burst of data-driven approaches has been widespread through all scientific computing area, for example, exascale simulations for multiscale and multiphysics problems [112, 31, 33, 64]. In specific, many researches have been developed by their own models for simulations or experiments to demonstrate a same physics problem much effectively. Hence, multiple fidelity data for the same physics problem are omnipresent and it is possible to combine data to gain accuracy and efficiency. In computational fluid dynamics, many results from different algorithms or experiments have been published for describing same flow physics. Hence, we can compare accuracy and efficiency of multiple models and distinguish their own strength and weakness. Finally, we are able to integrate multiple models to take their advantages only.

In data-driven approaches, it has shown that the information fusion between multiple fidelity data gain the computational efficiency if it is very expensive to obtain the highest fidelity data. In statistical learning theory, the Gaussian process has been widely used to fuse multiple fidelity data [80, 82, 79, 78, 63, 96]. The classical technique is to find a linear correlation between low and high fidelity via the approach of Kennedy and O'Hagan [52]. However, if multiple fidelity data not lie on the same linear manifold, i.e., there is no linear correlation, this linear approach loses its capability. In order to resolve this nonlinearity problem, nonlinear auto-regressive Gaussian process (NARGP) was introduced [79] and shows the capability of fusion for multiple fidelity data which are nonlinearly correlated. This approach introduces a hidden (latent) dimension to find a strong correlation between low and high fidelity data on $(d+1)$ dimensional manifold. In specific, low fidelity data provides an additional dimension and high fidelity data is projected onto $(d+1)$

manifold, where Gaussian process has a simple and correct kernel.

Sometimes, results from same physics problem are different depending on the problem setup (the initial or the boundary condition and physical parameters) or uncertainty. For example, in computational fluid dynamics, there is a textbook problem, "flow past a circular cylinder". In this simulation or experiment, depending on mesh or experiment size, order of accuracy of scheme, type of scheme, and other parameter uncertainties, the flow dynamics are changed [44, 115, 69, 15]. In this case, we cannot guarantee the high fidelity data lie on the introduced $(d+1)$ dimensional manifold, where Gaussian process is available.

In order to resolve this problem, we may need to find another appropriate manifold where the Gaussian process is available. In this chapter, we introduce two different approaches to find the appropriate manifold, where low and high fidelity functions are invertible each other. This invertibility provides a (one-to-one) map between two functions, which makes Gaussian process available. The first method is *iterative Gaussian process*, which modifies the low fidelity data iteratively until we get the smooth manifold. Another approach is to construct the appropriate manifold with (intrinsic) diffusion coordinates via Diffusion maps. In this approach, we construct a $(d+x)$ dimensional manifold by choosing pairs of diffusion coordinates. Specifically, Diffusion maps check a number of additional dimensions (x) for one-to-one mapping between two fidelity data and also provide the corresponding $(d+x)$ dimensional manifold where the Gaussian process works.

In order to show the capability of our approaches, we demonstrate a nonlinear dynamic system, Van der Pol oscillators [109], which have a bifurcation by a (scalar) nonlinear damping parameter μ. Depending on μ, responses of oscillators are totally different with respect to its frequency and shape. Hence, without physical intuition,

it is hard to find a reasonable relation between data from different Van der Pol oscillators.

In this chapter, we introduce general Van der Pol oscillators. After that, we present two different approaches of the manifold-driven Gaussian process. In order to demonstrate effectiveness of two approaches, we present two examples: intensity parameters of two data are 1) similar or 2) totally different. Finally, we show two approaches are good methods to find the GP-available (one-to-one invertible) manifold for the nonlinear information fusion.

6.2 Problem setup: Van der Pol oscillator

Van der Pol oscillators, which have a nonlinear damping, draw different response curves depending on an intensity parameter of nonlinear damping. The Van der Pol oscillator evolves in time according to following ODE:

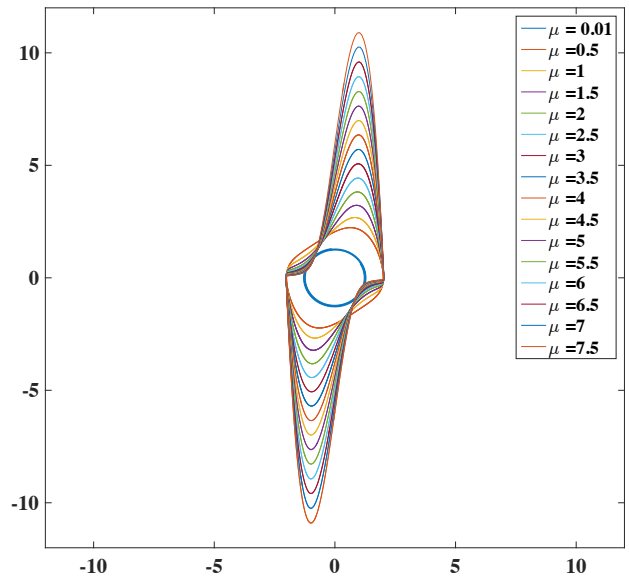
$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x = 0, \tag{6.1}$$

where μ is a scalar parameter, which controls the intensity of nonlinear damping.

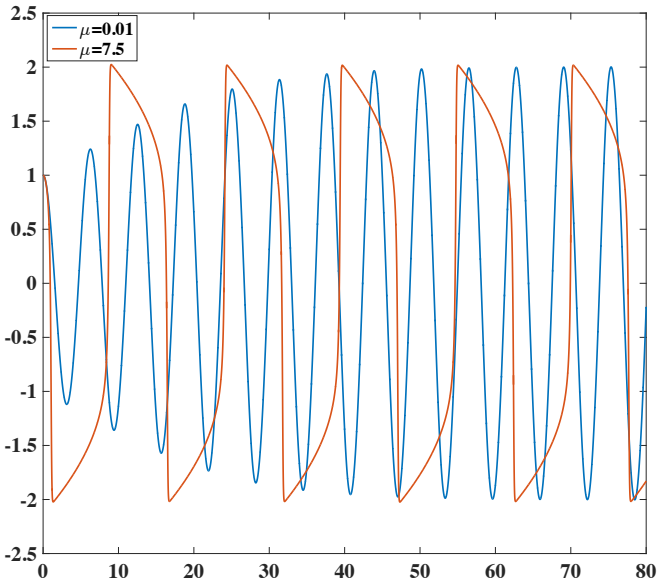
Generally, time evolution of dynamics of the Van der Pol oscillator can be rewritten to two-dimensional ODE system by Lienard transformation as

$$\dot{x} = y, \tag{6.2}$$

$$\dot{y} = \mu(1 - x^2)y - x. \tag{6.3}$$



(a) The evolution of a limit cycle.



(b) A example of response curves of two different oscillators.

Figure 6.1: Dynamics of Van der Pol oscillators. (a): the evolution of a limit cycle depending on a parameter μ. Peak points increase as μ increases. (b): Responses of two oscillator with different parameters $\mu = 0.1$ and $\mu = 7.5$. The x-axis and y-axis represent time t and response x, respectively.

The limit cycle of oscillators are parameterized by the scalar parameter μ, see Figure 6.1 (a). We employ Matlab solver ODE45 for solving the ordinary differential equation (ODE) system 6.2 and 6.3. For simplicity, in this thesis, we use only two different oscillators for the nonlinear information fusion.

In a view of data mining, our merged and expanded data set $Y = \{\mathbf{x_h}, \mathbf{y_h}, \mathbf{x_l}, \mathbf{y_l}\}$ (introduced in next section), supports to find appropriate pair of coordinates via nonlinear statistical learning algorithm, Diffusion maps. Hence, we can find latent dimensions on parametric space and this results in finding intrinsic function effectively.

Data from two oscillators cannot be distinguished with respect to fidelity. However, in order to match the general concept of multiple fidelity information fusion framework, we assume one oscillator as a high fidelity model and another as a low fidelity model.

6.3 Methods

6.3.1 Nonlinear auto-regressive Gaussian Process

A generalized auto-regressive scheme for multiple fidelity is

$$f_h(\mathbf{x}) = z_l\left(f_l(\mathbf{x})\right) + \delta_h(\mathbf{x}), \tag{6.4}$$

where f_l and f_h are low and high fidelity data, respectively. $z_l(\cdot)$ is an unknown mapping from a low fidelity model to a high fidelity model. If there is a linear

map $z_l(\cdot)$ between f_l and f_h, then we employ a classical linear approach of Kennedy and O'Hagan, i.e., z_l is independent of x. For a nonlinear map between low and high fidelity model, the generalized scheme still enable to find appropriate mapping via nonlinearity of $z_l(\cdot)$. However, the computational cost to find the appropriate nonlinear map is dramatically increasing and this results in losing total efficiency of information fusion.

In order to gain efficiency, Paris *et al.* [79] employ the additional dimension via previous inference level f_l. Hence, the auto-regressive scheme on a $(d+1)$ dimensional manifold can be rewritten as

$$f_h(\mathbf{x}) = g\left(\mathbf{x}, f_l(\mathbf{x})\right). \tag{6.5}$$

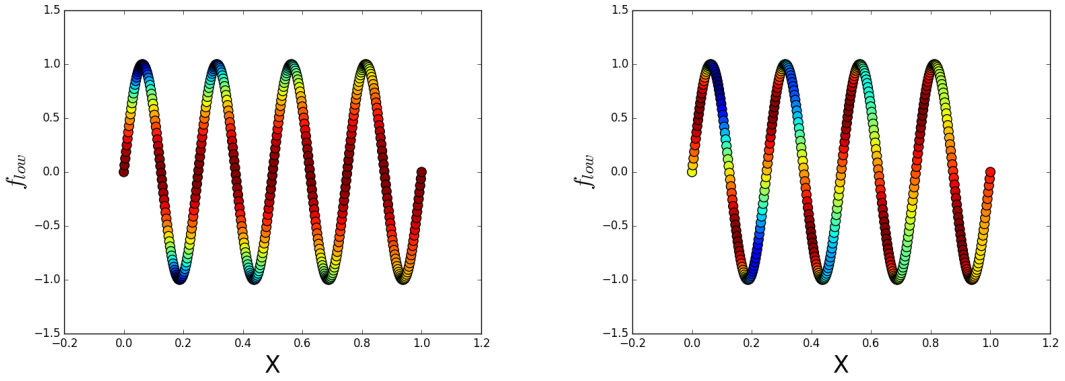
and g is a Gaussian process as follows:

$$g \sim \mathrm{GP}\left(f_h | k_h((\mathbf{x}, f_l(\mathbf{x})), (\mathbf{x}^*, f_l(\mathbf{x}^*)); \theta_{\mathbf{h}}\right), \tag{6.6}$$

where $\mathbf{x}, \mathbf{x}^*$ are given data points. For more details, see the reference [79].

If high fidelity data are embedded on the low fidelity data and invertible (or have directionality) as Figure 6.2 (a), i.e., the high fidelity data lie on the $(d+1)$ dimensional manifold constructed by x and $f_l(x)$, the kernel on this manifold for Gaussian process has a main direction and this leads to gain higher accuracy of Gaussian process compared to the original d dimensional manifold. However, if distribution of high fidelity data has no directionality on the $(d+1)$ dimensional manifold as Figure 6.2 (b), the introduced NARGP fails and finally there is no accuracy gain even though we use multiple fidelity data.

From this simple counter-example, we show that the introduced NARGP is not



(a) $f_h = (x - \sqrt{2}) f_l(x)^2$. (b) $f_h = (x - \sqrt{2}) f_l(x + 1/10)^2$.

Figure 6.2: A example of $(d+1)$ dimensional manifold. Colormaps of different high fidelity models embedded on low fidelity data, $f_l = \sin(8\pi x)$. A color distribution represents high fidelity data at location x (x-axis). (a): the high fidelity data are embedded on the low fidelity model with directionality in y direction. (b): the high fidelity model are not embedded on the low fidelity model because of non-directionality.

capable for information fusion with nonlinearly correlated data. In this case, we need to find other approaches to fuse multiple information or find the appropriate manifold where the high and low fidelity data are invertible each other. In this chapter, first, we introduce "iterative" Gaussian process" to find a smooth $(d+1)$ manifold by modifying low fidelity data.

6.3.2 Iterative Gaussian Process

Generally, it is relatively costless to obtain data from low fidelity models (either simulations or experiments) compared to high fidelity models. Since a number of low fidelity data is enough (but less accurate), we can modify our low fidelity data until fitting targeted high fidelity data. Then, we are able to employ nonlinear auto-regressive Gaussian process with modified data sets. In geometrical interpretation, the original manifold constructed by x, $f_l(x)$, and $f_h(x)$ is not smooth and has wiggles, which makes the accuracy of NARGP down. Hence, we stretch this uneven

manifold until we get a smooth (or flat) manifold by modifying low fidelity data iteratively.

The iterative Gaussian process (IGP) has an iteration and regression. First, two multiple fidelity data sets are merged into one big data sets as

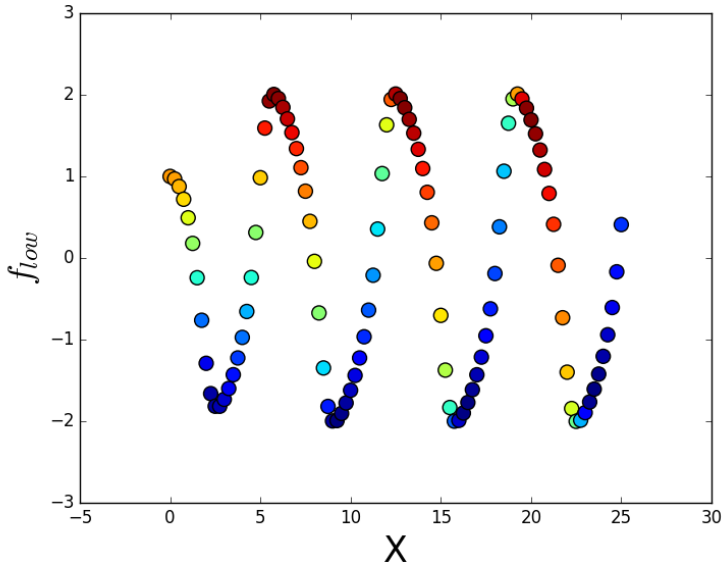
$$y_T = \{(y_l(x_1), \ldots, y_l(x_{nl}), y_h(x_1), \ldots y_h(x_{nh})\}. \tag{6.7}$$

where subscript l and h represent the low and high fidelity data, respectively. After that, we perform the Gaussian process regression with y_T for estimating new low fidelity data at location x_l. Then, we have updated low fidelity data y_{l1} by regression and perform this process iteratively until we get a smooth manifold, see the algorithm 1.

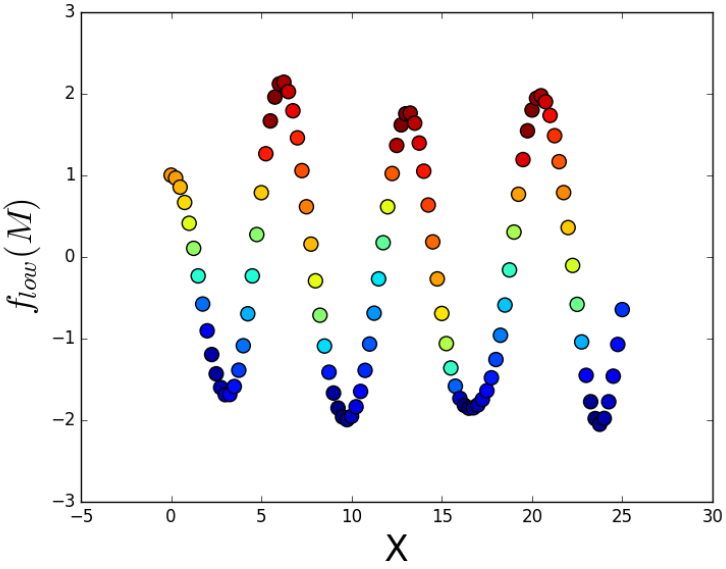
Algorithm 1: Iterative Gaussian Process (IGP) for modifying the low fidelity data

Gaussian Process;
Data : $(\{x_l, x_h\}, \{y_l, y_h\})$
Result: (x_l, y_l^*)
for $i \leftarrow 1$ **to** l **do**
  $(x_l, y_l) \leftarrow (x_l, y_l^*)$;
  $D = (\{x_l, x_h\}, \{y_l, y_h\})$;
  $h \sim GP(f_l | k_h(x, x^*); \theta_l)$ for $x, x^* \in D$;
  $y^* = h(x_l)$;
  (x_l, y_l^*)
end

In the iterative Gaussian process framework, the high fidelity data pick up low fidelity data to match its own frequency iteratively. Finally, two fidelity data are matched their frequency, which makes our manifold smooth. In this thesis, our stop criteria of iterations is given by the physical intuition but generally, (d) dimensional total variation is one of the best candidates to measure the smoothness (or flatness) of the manifold.



(a) The original low fidelity model.



(b) The modified low fidelity model.

Figure 6.3: A example of iterative Gaussian process. The distribution of high fidelity data on (a): the original low fidelity data and (b): the modified low fidelity data. The color represents the high fidelity data at location x (x-axis). The modified low fidelity data is obtained after 10 iterations. The high and low fidelity data have parameters $\mu = 1.5$ and $\mu = 1.1$, respectively.

In Van der Pol oscillators, changing a intensity parameter (μ) changes the period of the limit cycle and this results in changing the shape and phase of responses. As difference between two parameters getting bigger, corresponding responses of oscillators have large shifted phase. It may render low fidelity data useless since different frequency and phase provides spurious frequency to the high fidelity oscillator. Hence, we need another approach which is capable to find a smooth manifold correctly and effectively. In next section, we introduce another *manifold-driven* Gaussian process via a nonlinear statistical learning tool, Diffusion maps.

6.3.3 Gaussian Process via Diffusion maps

Diffusion maps have been widely used in dimensionality reduction. It extracts a *nonlinear* low dimensional manifold without losing of slow dynamics of the system. Hence, it guarantees the minimum dimensional manifold compared to classical linear techniques like principal component analysis (PCA) [48].

In this chapter, the other way around, we employ Diffusion maps for finding hidden (latent) dimensions, i.e., dimensionality expansion. Two or multiple fidelity data, which have space and time dependent correlations, can be embedded on a parametric manifold where Gaussian process is available. Diffusion maps extract several principal nonlinear coordinates from our expanded data set and some pairs of these coordinates construct a GP-available manifold.

Given n multiple fidelity data $\{\mathbf{y_1}(\mathbf{x}), \ldots, \mathbf{y_n}(\mathbf{x})\}$ of N observations, we set an $N \times N$ affinity matrix via diffusion kernel by

$$\mathbf{W_{ij}} = \exp\left(-\frac{\|\mathbf{Y_i} - \mathbf{Y_j}\|^2}{\epsilon}\right), \tag{6.8}$$

where $\mathbf{Y_i} = (y_1(x_i), \ldots, y_n(x_i))$ is an observation vector of multiple fidelity models at same location x_i. Then,

$$\tilde{\mathbf{W}} = \mathbf{D}^{-1}\mathbf{W}, \tag{6.9}$$

where D is a diagonal matrix given by

$$\mathbf{D_{ii}} = \sum_j \mathbf{W_{ij}}. \tag{6.10}$$

Then, the matrix $\tilde{\mathbf{W}}$ can be a transition probability matrix of a Markov chain and this matrix is equivalent to discrete Laplace-Beltrami operator [20]. Eigenfunctions of the matrix $\tilde{\mathbf{W}}$, called diffusion coordinates, describe the embedded (nonlinear) coordinates effectively. Among those eigenfunctions, we choose a pair of diffusion coordinates which high and low fidelity data embedded with directionality. Then, the number of elements in the chosen pair informs the number of hidden (latent) dimension for mapping each other. On the chosen manifold, the high and low fidelity data are constructed by corresponding Gaussian processes g_h and g_l on the m dimensional manifold as:

$$f_h(x) = g_h\left(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_m(\mathbf{x})\right), \tag{6.11}$$

$$f_l(x) = g_l\left(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_m(\mathbf{x})\right), \tag{6.12}$$

where g_h and g_l are Gaussian processes constructed by

$$g_h \sim \mathrm{GP}\left(f_h | k_h((\phi_{\mathbf{1}}(\mathbf{x}), \ldots, \phi_{\mathbf{m}}(\mathbf{x})), (\phi_{\mathbf{1}}(\mathbf{x}^*), \ldots, \phi_{\mathbf{m}}(\mathbf{x}^*))); \theta_{\mathbf{h}}\right), \tag{6.13}$$

$$g_l \sim \mathrm{GP}\left(f_l | k_l((\phi_{\mathbf{1}}(\mathbf{x}), \ldots, \phi_{\mathbf{m}}(\mathbf{x})), (\phi_{\mathbf{1}}(\mathbf{x}^*), \ldots, \phi_{\mathbf{m}}(\mathbf{x}^*))); \theta_{\mathbf{l}}\right). \tag{6.14}$$

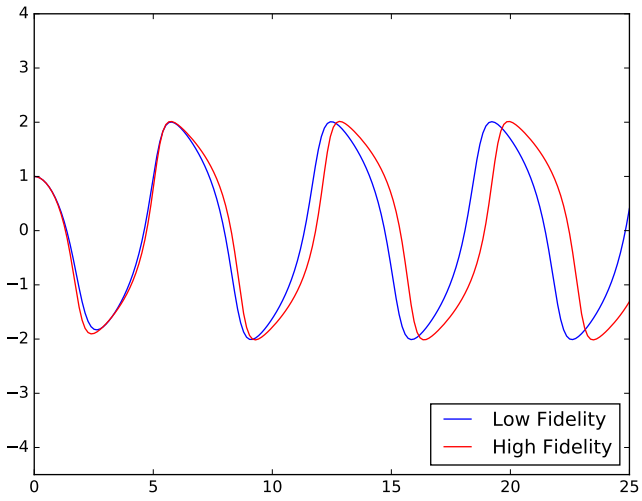
Hence, we employ Gaussian process g_h on the $(d + x)$ parametric space for the estimation of high fidelity function. For details about Diffusion maps and Gaussian process, see the Appendices.

6.4 Result

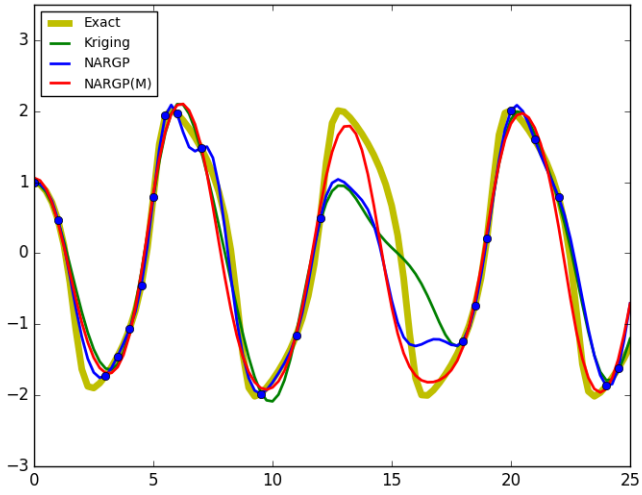
6.4.1 On the similar limit cycle ($\mu_h \sim \mu_l$)

As shown in Figure 6.1, if parameters μ of two oscillators are similar, i.e., two data have similar limit cycles, we are able to employ iterative Gaussian process (IGP). In this chapter, we choose parameters $\mu = 1.5$ for the high fidelity model and $\mu = 1.1$ for the low fidelity model. Modified low fidelity data by IGP give a stretched (and smoothed) manifold, where high and low fidelity data have one-to-one mapping, see Figure 6.3. As shown in Figure 6.4 (b), modified low fidelity data (labeled by NARGP(M)) have the lower RMSE compared to Gaussian process with high fidelity data only (Kriging) and NARGP with original low fidelity data. Moreover, it is found that IGP shows the better result if we have a few high fidelity data. Hence, this framework is robust to number of high fidelity data compared to other approaches.

Moreover, we employ another manifold-driven Gaussian process. Each oscillator, solved by equations (6.2) and (6.3), has two data sets, x and $\dot{x}$ (totaling four data sets). We construct an expanded data set as $Y = \{x_h, \dot{x}_h, x_l, \dot{x}_l\}$. And Diffusion maps provide several diffusion coordinates from Y and we choose the best pair of

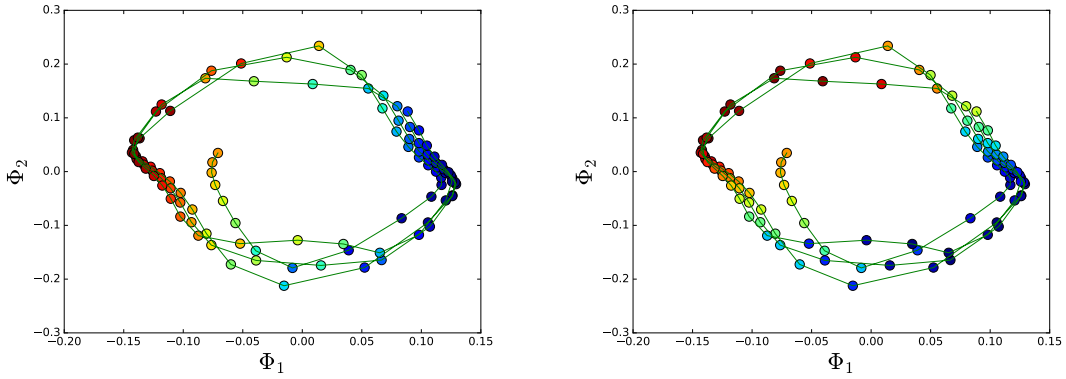


(a) Responses of two oscillators on the similar limit cycle.



(b) Regression results of different approaches.

Figure 6.4: A result of iterative Gaussian process. (a): exact responses of two different oscillators – $\mu = 1.1$ for low fidelity and $\mu = 1.5$ for high fidelity. (b): regression results with different approaches. (M) represents the modified low fidelity data by IGP.



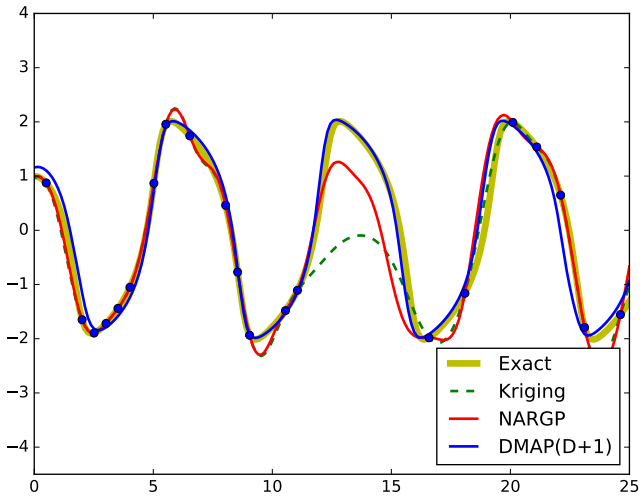
(a) High fidelity data. (b) Low fidelity data.

Figure 6.5: Distribution of two fidelity data on the GP-available manifold. The pair of (ϕ_1 and ϕ_2) is the best to embed high and low fidelity data with directionality. The green line represents the path along with x. (a): The color represents high fidelity data. (b): The color represents low fidelity data.

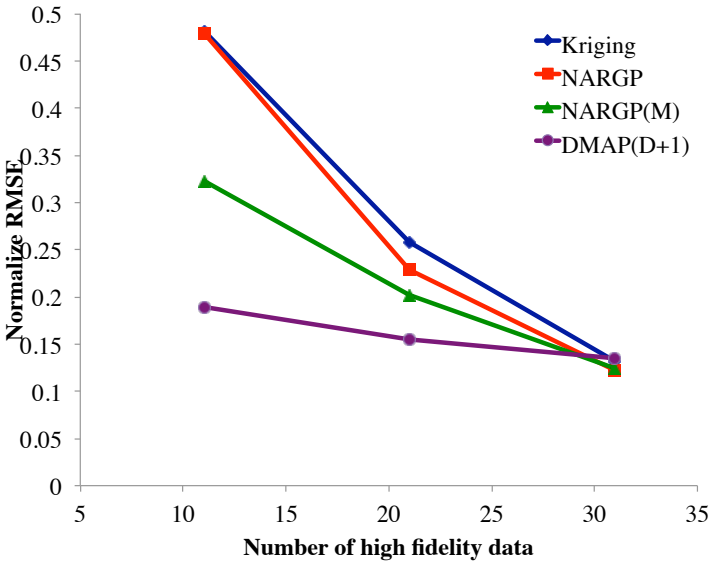
them, see Figure 6.5. On this chosen manifold constructed by diffusion coordinates ϕ_1 and ϕ_2, the high and low fidelity data are invertible and one-to-one mappable. Color distributions of two fidelity data on this manifold are tilted each other. It informs the phase shift between two oscillators, see the Figure 6.4 (a). The Gaussian process on the GP-available manifold is found to be the best approach compared to simple Gaussian process and original NARGP see the Figure 6.6. However, if the number of high fidelity data increase, other approaches provide better results because the chosen manifold does not have a clear directionality everywhere, which makes small wiggles on the manifold.

6.4.2 On the different limit cycle ($\mu_h \neq \mu_l$)

We choose parameters of two different oscillators as $\mu = 1.5$ for high fidelity model and $\mu = 5.0$ for low fidelity model and the exact responses of two oscillators are shown in Figure 6.7 (a). The shape and frequency are totally different and hence it is hard to find a nonlinear relation between data without physical intuition or prior

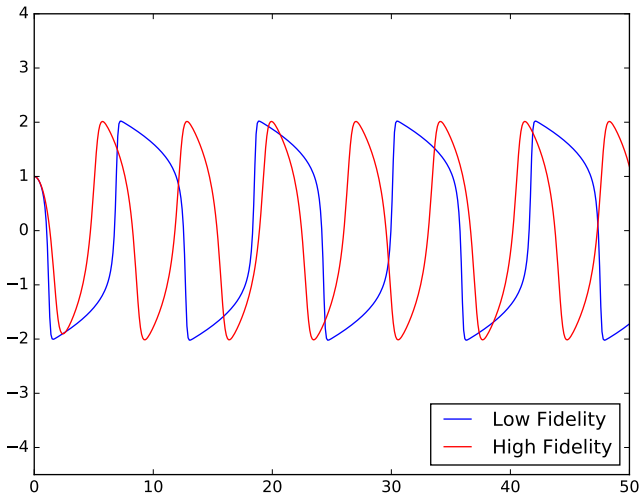


(a) Regression results of different approaches.

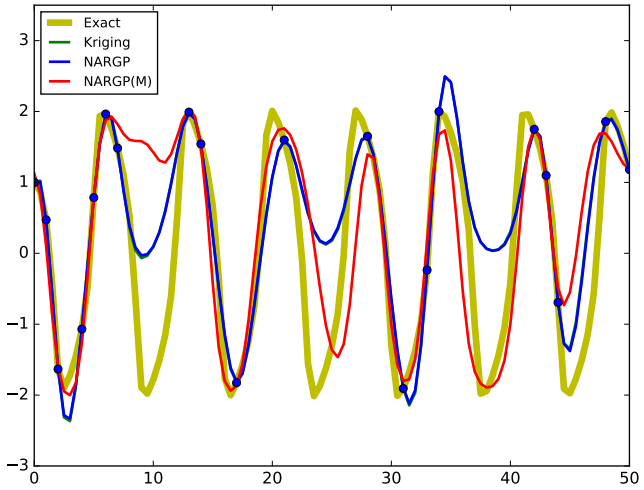


(b) Normalized RMSE with different approaches.

Figure 6.6: The result of manifold-driven Gaussian process in the similar limit cycle. (a): Regression results of different approaches. (D+1) represents that we employ one additional dimension. Blue dots represent high fidelity data points. (b): the x-axis represents number of high fidelity data. Each RMSE is normalized by max f_h and averaged by 10 simulations.

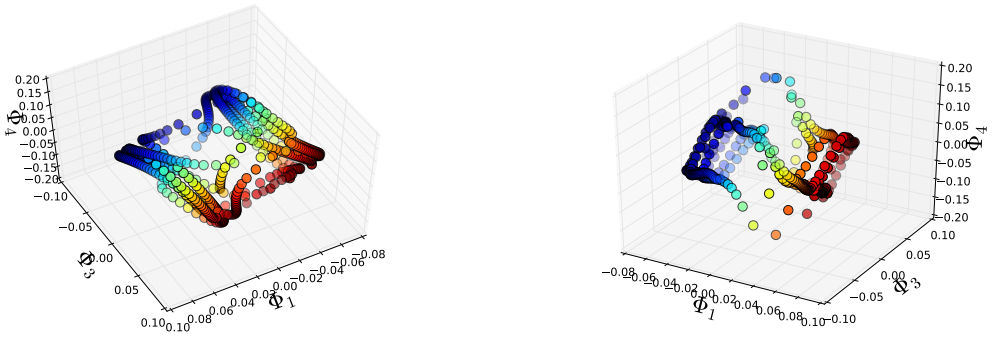


(a) Responses of different approaches on the different limit cycle.



(b) Regression results of different approaches.

Figure 6.7: The result of iterative Gaussian process. (a): exact responses of two different oscillators: $\mu = 5.0$ for low fidelity and $\mu = 1.5$ for high fidelity. (b): regression results of different approaches. (M) represents the modified low fidelity data by IGP.



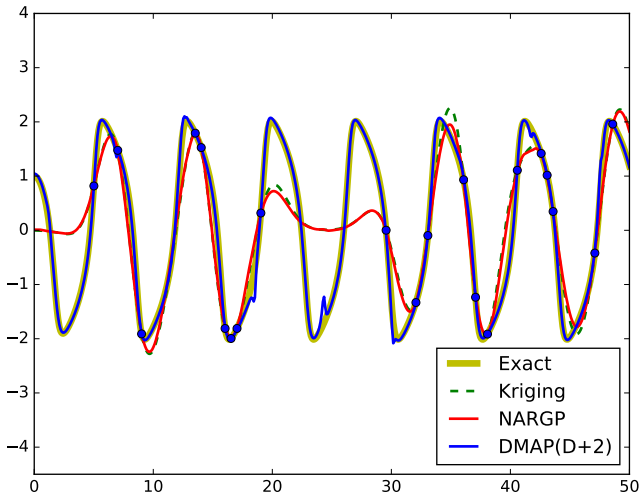
(a) High fidelity data.

(b) Low fidelity data.

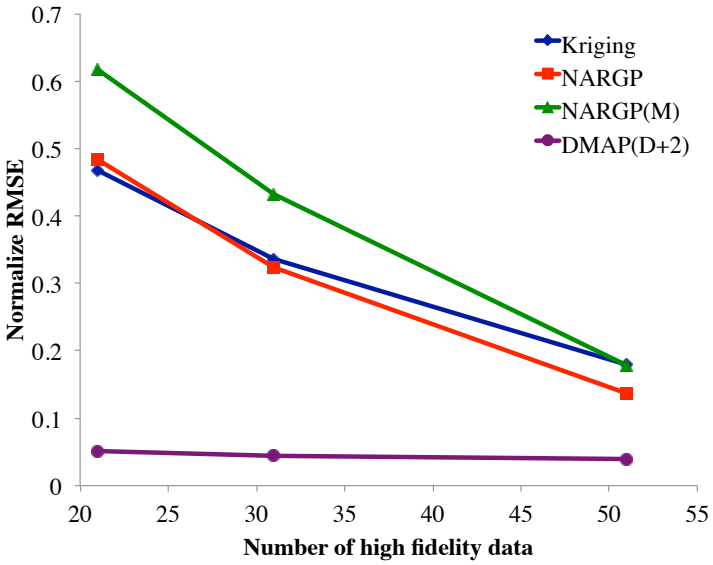
Figure 6.8: Distribution of two fidelity data on the GP-available manifold. The pair of (ϕ_1, ϕ_3, and ϕ_4) is found the best manifold to embed high and low fidelity data effectively. (a): The color represents high fidelity data. (b): The color represents low fidelity data.

knowledge of the dynamical system.

First, we employ IGP and its results are shown in Figure 6.7 (b). IGP is no longer good approach with two data sets because two oscillators have totally different frequencies and thus, IGP produces a spurious oscillation between two frequencies. This result makes IGP worse than Kriging (GP with only high fidelity data) even though we employ two data sets. However, the manifold-driven GP still provides the better result if we add two additional dimension, see Figures 6.8 and 6.9. While similar oscillators need only one additional dimension to map each other, different oscillators need two additional dimensions to construct GP-available manifold because there is a bifurcation of the limit cycle between two oscillators. In this case, we choose a pair of triple diffusion coordinates of ϕ_1, ϕ_3, and ϕ_4, see the Figure 6.8. The manifold-driven GP are able to reduce the RMSE dramatically compared to other approaches, see Figure 6.9 (b) and this reduction is independent of the number of high fidelity data. Because we construct a smooth-enough manifold for GP, the RMSE is always small independent of the number of high fidelity data.



(a) Regression results of different approaches.



(b) Normalized RMSE with different approaches.

Figure 6.9: The result of manifold-driven Gaussian process on the different limit cycle. (a): regression results of different approaches. (D+2) represents that we employ two additional dimensions. Blue dots represent high fidelity data points. (b): the x-axis represents number of high fidelity data. Each RMSE is normalized by max f_h and averaged by 10 simulations.

From this result, we are able to find a (hidden) nonlinear correlation between two data sets via the pair of Diffusion coordinates. Also, we demonstrate the capability of the manifold-driven for bifurcated dynamical systems. Finally, we setup the algorithm to find the appropriate manifold to (one-to-one) map each other, whose manifold becomes the important clue to describe physics between oscillators.

6.5 Conclusion

In this chapter, we introduce two novel approaches for the nonlinear auto-regressive information fusion and demonstrate their capabilities in the Van der Pol oscillators. Specifically, if parameters μ are similar, i.e., limit cycles are similar, the IGP framework can stretch and smooth the original manifold. Also, it is a clever way to find the GP-available manifold via Diffusion maps. If parameters μ are totally different, on the other hand, only Gaussian process with Diffusion maps presents the reasonable result and it shows the robustness for nonlinear correlated data. The ultimate goal of two approaches are same; to find a smooth (one-to-one) manifold with high and low fidelity data, where Gaussian process is available. Finally, we present the new type of the information fusion between nonlinearly correlated data.

Moreover, we show that the manifold given by Diffusion maps is able to map two oscillators each other. In this mapping, the number of additional dimensions may provide the important physical meaning, which informs the location of the bifurcation in dynamical systems. From data, we are able to find a system identification without physical models, equations, variables, and parameters. Hence, this approach will provide a breakthrough in data mining of complex dynamical systems.

Chapter Seven

Conclusions and Future work

7.1 Conclusions

In this thesis, we have demonstrated a fault-resilient, robust, and efficient framework for exascale computational fluid dynamics. The novel framework contains cutting-edge techniques in data science. For example, we employed a supervised statistical learning technique, multi-level Gaussian process regression, for information fusion with multiple fidelity or scale simulations to equip the resilience against expected hardware or software error. Moreover, we employed a nonlinear dimensionality reduction technique, Diffusion maps, to detect the computational redundancy and accelerate simulations via projective time integration adaptively. Furthermore, we combine these statistical learning techniques, a nonlinear manifold-driven autoregressive Gaussian process to fuse nonlinearly correlated 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. More broadly, in this thesis we demonstrate the symbiotic and synergistic combination of statistical learning, domain decomposition, and scientific computing in exascale simulations.

In chapter 2, we introduced three different approaches to fill-in the spatial missing gaps. In chapter 3, we set a novel framework, the *gappy simulation*, which are able to keep simulating against spatial missing data with limited accuracy but costless auxiliary simulation vis information fusion techniques. In chapter 4 and 5, we extended the aforementioned framework for spatio-temporal gaps, called the *patch simulation*, which are able to not only accelerate simulation by projective time integration with dynamics-informed time steps but also combine multiple fidelity, rate, or heterogeneous data. In chapter 6, we introduced a new information fusion tech-

nique, *manifold-driven auto-regressive Gaussian process*, which combine Diffusion maps and nonlinear auto-regressive Gaussian process for nonlinearly correlated data in complex dynamical system. Here we draw up a list for future works to extend our framework.

7.2 Future work

7.2.1 Numerical analysis of the patch (or gappy) simulations

In previous chapters, we formulated the novel framework, the patch (or gappy) simulation and demonstrated various benchmark problems to show its effectiveness, robustness, and efficiency. In mathematical view, 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. The reference frameworks (the gap-tooth algorithm and patch dynamics) have been shown their numerical analysis and sensitivity in [87, 88, 90, 12].

The first idea for the numerical analysis of the proposed framework is the comparison of two operators. We formulate an equivalent operator of the patch (or gappy) simulation, $\hat{\mathbf{A}}$ compared to the original (reference) operator, $\mathbf{A}$ in simple dynamics given by

$$\mathbf{x^{n+1}} = \mathbf{A}\mathbf{x^{n}}, \tag{7.1}$$

where $\mathbf{x}$ is a state variable vector, e.g., velocity fields and pressure field in computational fluid dynamics. Depending on gap sizes, auxiliary data, projection time, equivalent operators $\hat{\mathbf{A}}$ are changed. Then, we compare eigenfunctions of two oper-

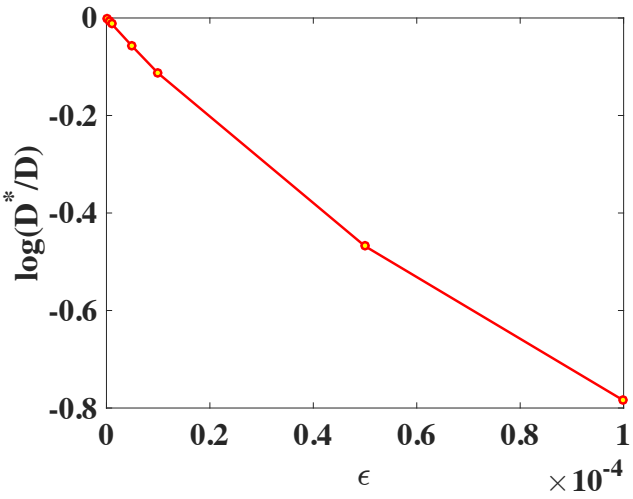


Figure 7.1: The error of $\log \hat{D}/D$ for fixed error ϵ.

ators and obtain sensitivity or error analysis from difference of eigenfunctions with respect to targeted parameters. Equivalent operator $\hat{\mathbf{A}}$ can be obtained by "dynamic mode decomposition", which is capable to find approximated eigenfunctions of a nonlinear dynamic operator [92, 17].

The second idea for the numerical analysis is to use Diffusion maps with time series data and compare principal diffusion coordinates. A rough sketch for the sensitivity analysis by Diffusion maps is following. If we assume that the equivalent operator $\hat{\mathbf{A}} = \mathbf{A} + \epsilon$ has (fixed) constant error, without loss of generality, then

$$\|\hat{\mathbf{A}}\mathbf{x} - \mathbf{x}\| = \|(\mathbf{A} + \epsilon)\mathbf{x} - \mathbf{x}\| \leq \|\mathbf{A}\mathbf{x} - \mathbf{x}\| + \|\epsilon\mathbf{x}\|. \tag{7.2}$$

And then,

$$\|\hat{\mathbf{A}}\mathbf{x} - \mathbf{x}\|^2 \leq \|\mathbf{A}\mathbf{x} - \mathbf{x}\|^2 + 2\epsilon\|\mathbf{A}\mathbf{x} - \mathbf{x}\|\|\mathbf{x}\|. \tag{7.3}$$

We can rewrite above equation to a normal form of Diffusion maps as

$$\exp\left(-\frac{\|\hat{\mathbf{A}}\mathbf{x}-\mathbf{x}\|^2}{\sigma^2}\right) \geq \exp\left(-\frac{\|\mathbf{A}\mathbf{x}-\mathbf{x}\|^2}{\sigma^2} - \frac{2\epsilon\|\mathbf{A}\mathbf{x}-\mathbf{x}\|\|\mathbf{x}\|}{\sigma^2}\right). \tag{7.4}$$

Finally, we finish the rough sketch of the sensitivity analysis and get the relation between a constant error ϵ and approximated Laplace operator as

$$\hat{\mathbf{A}} = \mathbf{A} + \epsilon \quad \leftrightarrow \quad \hat{D} = D \cdot \exp(\alpha\|\epsilon\|), \tag{7.5}$$

where D represents a diffusion kernel. The preliminary result for fixed ϵ shows that we estimate the error in the right way, see Figure 7.1.

From this rough sketch, we will develop the numerical analysis with respect to targeted parameters. Finally, this fundamental numerical analysis are able to support and provide mathematical foundation to the proposed framework, and this gives the generality of this framework.

7.2.2 The extension of *manifold-driven* Gaussian process

We demonstrated that manifold-driven Gaussian process are able to fuse two data which are nonlinearly correlated each other. Now, we can extend this scheme to find a structure of nonlinear dynamical system from data [25, 9, 93, 104, 11]. If we construct the $(d+x)$ manifold which is able to map each data, we find the knowledge about a bifurcation by data mining in complex dynamical system. However, $(d+x)$ manifold cannot be visualized if $d+x>3$. Hence, we need new algorithms to find the appropriate manifold without visualization.

Another topic we are making effort is to construct same manifold with a few high fidelity data. In the previous chapter, we demonstrated and addressed the capability of the GP-available manifold with a large set of high and low fidelity data. This approach provided not only the number of hidden (latent) dimension for Gaussian process but also the pair of Diffusion coordinates of the GP-available manifold. More practical issue is that we have a few high fidelity data in general cases. In this case, we find diffusion coordinates by a few data and extend these coordinates by interpolation.

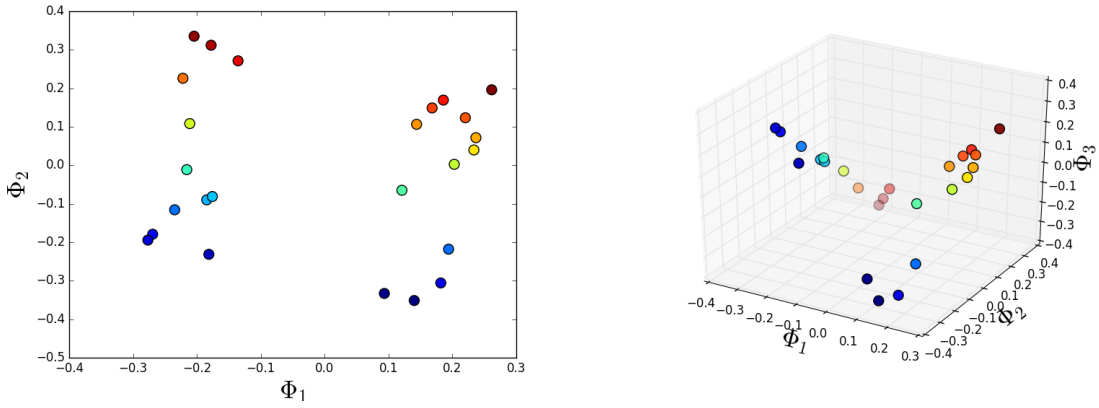
As shown in preliminary result, we employ a spline interpolation with *equidistant* data, which works better than randomly distributed data. As shown in Figure 7.2, a few data construct not a same but similar manifold compared to the original manifold constructed by a large set. A simple interpolation (like a spline) provides similar Diffusion coordinates but it works for only equidistant data.

In order to overcome this limitation, we suggest to employ geometric harmonics [58, 21]. Geometric harmonics, based on Nyström method, is able to extend targeted function to large dimensional manifold. Geometric harmonics introduce the extension ($\mathbf{K}$) and restriction ($\mathbf{K}^*$) operators as

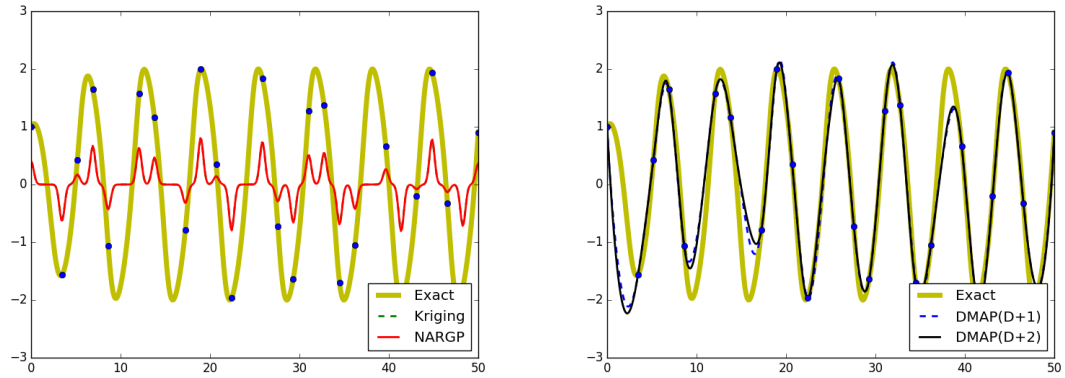
$$\mathbf{K}\psi_j = \lambda_j \Psi_j, \tag{7.6}$$

$$\mathbf{K}^*\Psi_j = \psi_j, \tag{7.7}$$

where ψ and Ψ represent eigenfunctions on restricted and extended manifolds, respectively. Then, the extension algorithm starts to project a targeted function onto



(a) The GP-available manifold constructed by 25 high fidelity data.



(b) Regression results of different approaches.

Figure 7.2: The result of manifold-driven Gaussian process with a few data. (a): the $(d+1)$ and $(d+2)$ dimensional GP-available manifold by 25 high fidelity data. (b): regression results by Kriging (simple GP), NARGP, $(d+1)$ dimensional manifold, and $(d+2)$ dimensional manifold.

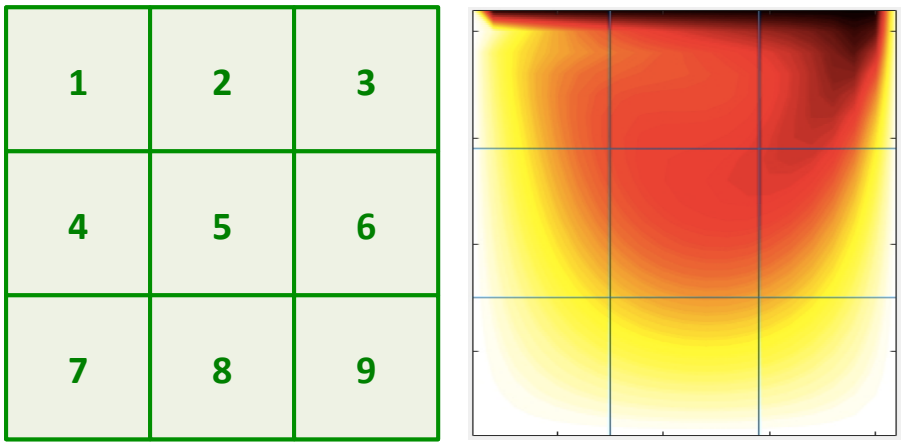


Figure 7.3: A schematic illustration of auto-adaptive scheme in space. (a): a computational domain is divided by nine subdomains and we choose four subdomains among them. (b): the concentration of species in transport equation.

the orthonormal manifold as

$$f \mapsto \mathbf{P}_\delta f = \sum_{j \in S_\delta} \langle f, \psi_j \rangle \, \psi_j. \tag{7.8}$$

After that, we can extend this function to the larger manifold constructed by an extend operator $\mathbf{E}$ as

$$\mathbf{E}f(x) = \sum_{j \in S_\delta} \langle f, \psi_j \rangle \, \Psi_j. \tag{7.9}$$

Then, we construct extended (mapped) diffusion coordinates with a few high and low fidelity data. Finally, we will obtain much accurate manifold compared to the reference manifold constructed by all high and low fidelity data. We will set the best approach to construct GP-available manifold with a few data and this results in computational cost saving in multiple fidelity information fusion.

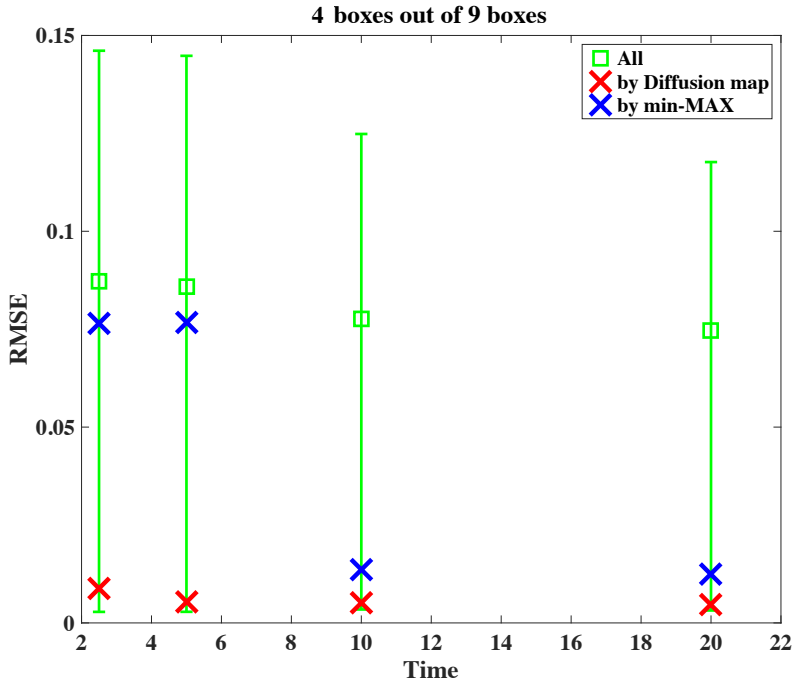


Figure 7.4: The RMS error of the interpolation by four subdomains in time. The green represents mean (square box), minimum, and maximum RMSE for all choosing possibility. The blue and red marker represent the "min-max" approach and the "Diffusion map", respectively.

7.2.3 Auto-adaptive and *asynchronous* exascale simulations

In exascale simulations, the multiscale simulation has been developed to demonstrate real physics problem effectively [112, 31, 33, 64]. In algorithmic sides, the efficient algorithm is highly required to solve the problem within reasonable computational time. We demonstrated that the patch simulation works against spatio-temporal gaps. Then, we can extend this framework to an auto-adaptive framework for the multiscale simulation. We choose gappy subdomains and corresponding timestep adaptively, i.e., we make spatio-temporal gaps intentionally, with relatively cost-less auxiliary data (given). Then, the auxiliary data provides not only appropriate locations of gappy domains but also dynamics-informed timestep size. Thus, this framework will be equivalent to classical adaptive mesh refinement (AMR) and this results in reducing computational cost dramatically. Moreover, if each gappy sub-

domain has its own projective time asynchronously, then the computational cost is reduced by less communication between computer nodes. For example, as shown in Figure 7.3, we choose four subdomains among nine subdomains to reduce computational time, which is governed by the equation 5.1. As shown in Figure 7.4, the chosen subdomains by Diffusion maps show the less RMS error compared to "min-max" algorithm, which find subdomains which have the largest difference of field variable.

Now we extend this framework to targeted problems. For example, in simulation for flow past a circular cylinder, velocity fields and pressure gradient in far field are not affected strongly by vortex shedding of the cylinder. Thus, we assign the spatial-temporal gaps in far field and furthermore, we can employ a different computational model, e.g., a potential flow model. This results in huge computational cost saving compared to classical approaches, which should employ same equations in whole fields.

Another example is the multiscale simulation in biological system. In complex biological system, there are multiple models for describing multiple scales, from atomic scale to continuum scale. If we solve all domains by (atomic scaled) molercular dynamics (MD), it is impossible to get results in reasonable computational time. However, this framework allows multiple information fusion between multiple scales with adaptive spatio-temporal gaps. Hence, we obtain results efficiently within reasonable simulation error, which is smaller than the modeling error.

Appendix A

Statistical learning tool: Gaussian Process regression

A.1 Classical Gaussian process – Kriging

A general Gaussian process, a supervised learning technique, has been widely used in the regression [86, 85, 102, 95, 16, 36]. The definition of Gaussian process is "a collection of random variables, any finite of which have a joint Gaussian distribution" [86]. Hence, we assume that our observations come from a realization of a Gaussian random field, f, with additive independent identically distributed Gaussian noise ϵ with variance σ_n^2, i.e., $\epsilon(\mathbf{x}) \sim N(0, \sigma_n^2)$. Then, our observation can be modeled by

$$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon(\mathbf{x}). \tag{A.1}$$

This framework is specified by a mean function $\mu(\mathbf{x})$ and its covariance kernel $k(\mathbf{x}, \mathbf{x}^*; \theta)$ with hyperparameters θ as

$$\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \tag{A.2}$$

$$k(\mathbf{x}, \mathbf{x}^*) = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}^*) - \mu(\mathbf{x}^*))], \tag{A.3}$$

where $\mathbf{x}$ and $\mathbf{x}^*$ are elements of a training set.

We find a hyperparameter via maximum likelihood estimation in training part. Using reconstructed $y(\mathbf{x})$, we can predict $y(\mathbf{x_{new}})$ at test location $\mathbf{x_{new}}$ by trained mean $\hat{\mu}$, variance $\hat{\sigma}^2$, and hyperparameter θ. The predicted mean, $\hat{y}(\mathbf{x_{new}})$ and the predicted variance $s^2(\mathbf{x_{new}})$ are calculated [49] as

$$\hat{y}(\mathbf{x_{new}}) = \hat{\mu} + k^{\mathrm{T}} \left(K + \hat{\sigma}_n^2 \mathrm{I}\right)^{-1} (\mathbf{y} - \mathbf{1}\hat{\mu}), \tag{A.4}$$

$$s^2(\mathbf{x_{new}}) = \hat{\sigma}^2 \left[1 - k^{\mathrm{T}} \left(K + \hat{\sigma_n}^2 \mathrm{I} \right)^{-1} k + \frac{\left[1 - k^{\mathrm{T}} \left(K + \hat{\sigma_n}^2 \mathrm{I} \right)^{-1} k \right]^2}{\mathbf{1}^{\mathrm{T}} \left(K + \hat{\sigma_n}^2 \mathrm{I} \right)^{-1} \mathbf{1}} \right], \tag{A.5}$$

where $k = k(\mathbf{x}, \mathbf{x_{new}}; \theta)$ and $K = k(\mathbf{x}, \mathbf{x}^*; \theta)$ represent the covariance vector and matrix. For detail, see the references [86, 81].

A.2 multi-level Gaussian Process - CoKriging

Depending on a chosen scheme, model, and parameter, we have multiple fidelity data for same physical problem. If these multiple fidelity data have a linear correlation, we are able to employ a multi-level Gaussian process regression by a well-known auto-regressive scheme introduced by Kennedy and O'Hagan [52]. Without loss of generality, in this thesis, we assume that there are two fidelity data: high and low fidelity data. High and low fidelity data can be constructed by different Gaussian process $f(\mathbf{x})$ and $f_l(\mathbf{x})$, respectively. Then an auto-regressive scheme sets a linear correlation as

$$f(\mathbf{x}) = \rho f_l(\mathbf{x}) + \delta(\mathbf{x}), \tag{A.6}$$

where $\delta \sim N(\mu_\delta, \sigma^2 K)$ is a Gaussian field and ρ is a (scalar) linear correlation parameter, which is also trained by given data set via maximum likelihood estimation. Then, the predicted mean and variance at test location $\mathbf{x_{new}}$ is calculated [61] as

$$\hat{y}(\mathbf{x_{new}}) = \hat{\mu} + \hat{\rho}_l \hat{y}_l(\mathbf{x_{new}}) + k^{\mathrm{T}} \left(K + \hat{\sigma}_n^2 \mathrm{I} \right)^{-1} \left(\mathbf{y_h} - \mathbf{1}\hat{\mu} - \hat{\rho}\hat{\mathbf{y}}_\mathbf{l} \right), \tag{A.7}$$

$$s^2(\mathbf{x_{new}}) = \hat{\rho}^2 s_l^2(\mathbf{x_{new}}) + \hat{\sigma}^2 \left[1 - k^{\mathrm{T}} \left(K + \hat{\sigma_n}^2 \mathrm{I} \right)^{-1} k + \frac{\left[1 - k^{\mathrm{T}} \left(K + \hat{\sigma_n}^2 \mathrm{I} \right)^{-1} k \right]^2}{\mathbf{1}^{\mathrm{T}} \left(K + \hat{\sigma_n}^2 \mathrm{I} \right)^{-1} \mathbf{1}} \right], \tag{A.8}$$

where subscript l indicates low fidelity. Furthermore, we are able to employ the auto-regressive scheme for multiple fidelity data through iterated process. For detail, see the reference [81].

Appendix B

Statistical learning tool: Diffusion maps

B.1 Diffusion maps

Diffusion maps have been used in model and dimensionality reduction fields and it is chosen one of the best candidate for finding nonlinear embedding manifold [22, 20, 19, 18, 45, 58, 57, 56, 73, 74]. Nonlinear dimensionality reduction schemes are able to project high dimensional data onto a lower dimensional manifold and it guarantees the lower dimension compared to linear dimensionality reduction schemes like principal component analysis (PCA).

In recent year, they have been studied that eigenfunctions of Laplace-Beltrami operator, which is a generalized Laplace operator and can be extended to form curvilinear manifold to Riemannian manifold, on the nonlinear manifold provide principal coordinates of lower dimensional manifold. For example in [28], on the two-dimensional rectangle ($L_1 \times L_2$), the first two eigenfunctions are two cosine functions, $\phi_1 = \cos(\pi x/L_1)$ and $\phi_2 = \cos(\pi y/L_2)$ except a constant eigenvector. These two eigenfunctions are one-to-one with x and y coordinates and hence we are able to parameterize given observations on the manifold which is constructed by two eigenfunctions. This results in the good parametric manifold via eigenfunctions of Laplace-Beltrami operator.

However, in general, observed data is not a continuous but scattered. Hence, it is needed to introduce "discrete" and "approximated" Laplace-Beltrami operator for scattered data. In order to provide approximation of continuous Laplace-Beltrami operator with scattered data, we employ a normalized Diffusion kernel matrix between observations as

$$\mathbf{W_{ij}} = \exp\left(-\frac{\|\mathbf{y_i} - \mathbf{y_j}\|^2}{\epsilon}\right), \tag{B.1}$$

where $\mathbf{y_i}$ is an observation at location $\mathbf{x_i}$. After that, we obtain $\tilde{\mathbf{W}}$ by

$$\tilde{\mathbf{W}} = \mathbf{D}^{-1}\mathbf{W}, \tag{B.2}$$

where D is a diagonal matrix by

$$\mathbf{D_{ii}} = \sum_j \mathbf{W_{ij}}. \tag{B.3}$$

Then $\tilde{\mathbf{W}}$ is a Markovian matrix whose elements represent the probability of jump from $\mathbf{y_i}$ to $\mathbf{y_j}$ on the true manifold and this matrix is equivalent to discrete Laplace-Beltrami operator.

Finally, high dimensional observations lie on the low dimensional manifold constructed by first d eigenfunctions as

$$\mathbf{y_i} \mapsto \left(\lambda_1\phi_{1,i}, \lambda_2\phi_{2,i}, \ldots, \lambda_d\phi_{d,i}\right)^{\mathrm{T}}. \tag{B.4}$$

where descending ordered eigenvalues λ_i represent scalar parameters which decide relative diffusion distance of corresponding coordinates. And its diffusion distance D on the embedded d-dimensional manifold is calculated by a normal form of Euclidean distance as

$$\mathrm{D}^2(\mathbf{y_i}, \mathbf{y_j}) = \sum_{k=1}^{d} \lambda_{k,i}^2(\phi_{k,i} - \phi_{k,j})^2. \tag{B.5}$$

And we can use this diffusion distance to the distance between data on the lower dimensional manifold. Also, it can be used for constructing a kernel in Gaussian process. For all details, see the reference [28].

Bibliography

[1] J L Arbuckle. *Full information estimation in the presence of incomplete data*. New Jersey: Lawrence Erlbaum Associates, 1996.

[2] P Astrid, S Weiland, K Willcox, and T Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53:2237–2251, 2008.

[3] Nadine Aubry. On the hidden beauty of the proper orthogonal decomposition. *Theoretical and Computational Fluid Dynamics*, 2(5):339–352, 1991.

[4] G Aupy, A Benoit, T Hérault, Y Robert, F Vivien, and Zaidoun D. On the combination of silent error detection and checkpointing. In *Proceedings of 2013 IEEE 19th Pacific Rim International Symposium on Dependable Computing*, 2013.

[5] JA Backer, CP Lowe, HCJ Hoefsloot, and PD Iedema. Poiseuille flow to measure the viscosity of particle model fluids. *The Journal of Chemical Physics*, 122(15):154503, 2005.

[6] Xin Bian, Zhen Li, Mingge Deng, and George Em Karniadakis. Fluctuating hydrodynamics in periodic domains and heterogeneous adjacent multidomains: Thermal equilibrium. *Physical Review E*, 92(5):053302, 2015.

[7] Wesley Bland, Aurelien Bouteiller, Thomas Herault, George Bosilca, and Jack J. Dongarra. Post-failure recovery of MPI communication capability: Design and rationale. *International Journal of High Performance Computing Applications*, 27(3):244–254, 2013.

[8] John A Board, Jeffrey W Causey, James F Leathrum, Andreas Windemuth, and Klaus Schulten. Accelerated molecular dynamics simulation with the parallel fast multipole algorithm. *Chemical Physics Letters*, 198(1-2):89–94, 1992.

[9] Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

[10] Tobias Brandvik and Graham Pullan. Acceleration of a 3d euler solver using commodity graphics hardware. In *46th AIAA aerospace sciences meeting and exhibit*, page 607, 2008.

[11] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[12] Judith Bunder and AJ Roberts. Patch dynamics for macroscale modelling in one dimension. *ANZIAM Journal*, 53:280–295, 2012.

[13] Mette Burmølle, Dawei Ren, Thomas Bjarnsholt, and Søren J Sørensen. Interactions in multispecies biofilms: do they actually matter? *Trends in microbiology*, 22(2):84–91, 2014.

[14] Franck Cappello, Al Geist, William Gropp, Sanjay Kale, Bill Kramer, and Marc Snir. Toward exascale resilience: 2014 update. *Supercomputing frontiers and innovations*, 1(1):5–28, 2014.

[15] Jyoti Chakraborty, Nishith Verma, and RP Chhabra. Wall effects in flow past a circular cylinder in a plane channel: a numerical study. *Chemical Engineering and Processing: Process Intensification*, 43(12):1529–1537, 2004.

[16] Krzysztof Chalupka, Christopher KI Williams, and Iain Murray. A framework for evaluating approximation methods for gaussian process regression. *Journal of Machine Learning Research*, 14(Feb):333–350, 2013.

[17] Kevin K Chen, Jonathan H Tu, and Clarence W Rowley. Variants of dynamic mode decomposition: boundary condition, koopman, and fourier analyses. *Journal of nonlinear science*, 22(6):887–915, 2012.

[18] Ronald R Coifman and Matthew J Hirn. Diffusion maps for changing data. *Applied and Computational Harmonic Analysis*, 36(1):79–107, 2014.

[19] Ronald R Coifman, Ioannis G Kevrekidis, Stéphane Lafon, Mauro Maggioni, and Boaz Nadler. Diffusion maps, reduction coordinates, and low dimensional representation of stochastic systems. *Multiscale Modeling & Simulation*, 7(2):842–864, 2008.

[20] Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[21] Ronald R Coifman and Stéphane Lafon. Geometric harmonics: a novel tool for multiscale out-of-sample extension of empirical functions. *Applied and Computational Harmonic Analysis*, 21(1):31–52, 2006.

[22] Ronald R Coifman, Stephane Lafon, Ann B Lee, Mauro Maggioni, Boaz Nadler, Frederick Warner, and Steven 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):7426–7431, 2005.

[23] I. Couckuyt, T. Dhaene, and P Demeester. ooDACE toolbox: A flexible object-oriented Kriging implementation. *Journal of Machine Learning Research*, 15:3183–3186, 2014.

[24] Alejandro C Crespo, Jose M Dominguez, Anxo Barreiro, Moncho Gómez-Gesteira, and Benedict D Rogers. Gpus, a new tool of acceleration in cfd: efficiency and reliability on smoothed particle hydrodynamics methods. *PloS one*, 6(6):e20685, 2011.

[25] James P Crutchfield and Bruce S McNamara. Equations of motion from a data series. *Complex systems*, 1(417-452):121, 1987.

[26] Frederica Darema. Dynamic data driven applications systems: A new paradigm for application simulations and measurements. In *International Conference on Computational Science*, pages 662–669. Springer, 2004.

[27] Jack Dongarra et al. The international exascale software project roadmap. *International Journal of High Performance Computing Applications*, 25(1):3–60, 2011.

[28] Carmeline Joan Dsilva. *Manifold Learning for Dynamical Systems*. PhD thesis, Princeton University, 2015.

[29] R Everson and L Sirovich. Karhunen-Loeve procedure for gappy data. *Journal of the Optical Society of America A*, 12:1657–1664, 1995.

[30] Graham E Fagg and Jack 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*, pages 346–353. Springer, 2000.

[31] Dmitry A Fedosov, Bruce Caswell, and George Em Karniadakis. A multiscale red blood cell model with accurate mechanics, rheology, and dynamics. *Biophysical journal*, 98(10):2215–2225, 2010.

[32] Dmitry A. Fedosov and George Em Karniadakis. Triple-decker: Interfacing atomistic–mesoscopic–continuum flow regimes. *Journal of Computational Physics*, 228(4):1157–1171, 2009.

[33] Dmitry A Fedosov, Wenxiao Pan, Bruce Caswell, Gerhard Gompper, and George E Karniadakis. Predicting human blood viscosity in silico. *Proceedings of the National Academy of Sciences*, 108(29):11772–11777, 2011.

[34] Dmitry A Fedosov, Igor V Pivkin, and George Em Karniadakis. Velocity limit in DPD simulations of wall-bounded flows. *Journal of Computational Physics*, 227(4):2540–2559, 2008.

[35] Frederick N Fritsch and Ralph E Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246, 1980.

[36] Yarin Gal, Mark van der Wilk, and Carl Edward Rasmussen. Distributed variational inference in sparse gaussian process regression and latent variable models. In *Advances in Neural Information Processing Systems*, pages 3257–3265, 2014.

[37] C. W. Gear, J. Li, and I. G. Kevrekidis. The gap-tooth method in particle simulations. *Physics Letters A*, 316:190–195, SEP 22 2003 2003.

[38] C William Gear and Ioannis G Kevrekidis. Projective methods for stiff differential equations: problems with gaps in their eigenvalue spectrum. *SIAM Journal on Scientific Computing*, 24(4):1091–1106, 2003.

[39] R Gioiosa, J C Sancho, S Jiang, F Petrini, and Davis K. Transparent, incremental checkpointing at kernel level: a foundation for fault tolerance for parallel computers. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing (SC '05)*, 2005.

[40] Jim Gray. Why do computers stop and what can be done about it? In *Symposium on reliability in distributed software and database systems*, pages 3–12. Los Angeles, CA, USA, 1986.

[41] W Gropp and E Lusk. Fault tolerance in message passing interface programs. *International Journal of High Performance Computing Applications*, 18:363–372, 2004.

[42] Hasan Gunes and Ulrich Rist. Spatial resolution enhancement/smoothing of stereo–particle-image-velocimetry data using proper-orthogonal-decomposition–based and Kriging interpolation methods. *Physics of Fluids*, 6:064101, 2007.

[43] Hasan Gunes, Sirod Sirisup, and George Em Karniadakis. Gappy data: To Krig or not to Krig? *Journal of Computational Physics*, 212(1):358–382, 2006.

[44] Oktay Güven, Cesar Farell, and Virendra C Patel. Surface-roughness effects on the mean flow past circular cylinders. *Journal of Fluid Mechanics*, 98(04):673–701, 1980.

[45] Laleh Haghverdi, Florian Buettner, and Fabian J Theis. Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics*, 31(18):2989–2998, 2015.

[46] Z Han, R Zimmermann, and S Gortz. A new co-Kriging method for variable-fidelity surrogate modeling of aerodynamic data. *AIAA*, pages AIAA2010–1225, 2010.

[47] Yulu Jia, George Bosilca, Piotr Luszczek, and Jack J. Dongarra. Parallel reduction to Hessenberg form with algorithm-based fault tolerance. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 88:1–88:11, New York, NY, USA, 2013. ACM.

[48] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[49] Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383, 2001.

[50] George Karniadakis and Spencer Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2013.

[51] George Em Karniadakis and George S Triantafyllou. Frequency selection and asymptotic states in laminar wakes. *Journal of Fluid Mechanics*, 199:441–469, 1989.

[52] M. C. Kennedy and A. O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

[53] Ioannis G. Kevrekidis, C. William Gear, James M. Hyman, Panagiotis G Kevrekidis, Olof Runborg, and Constantinos Theodoropoulos. Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis. *Communications in Mathematical Sciences*, 1(4):715–762, 12 2003.

[54] Ioannis G. Kevrekidis and Giovanni Samaey. Equation-free multiscale computation: Algorithms and applications. *Annual Review of Physical Chemistry*, 60(1):321–344, 2009.

[55] Milan Kilibarda, Tomislav Hengl, Gerard B. M. Heuvelink, Benedikt Gräler, Edzer Pebesma, Melita Perčec Tadić, and Branislav Bajat. Spatio-temporal interpolation of daily temperatures for global land areas at 1km resolution. *Journal of Geophysical Research: Atmospheres*, 119(5):2294–2313, 2014.

[56] Stephane Lafon, Yosi Keller, and Ronald R Coifman. Data fusion and multicue data matching by diffusion maps. *IEEE Transactions on pattern analysis and machine intelligence*, 28(11):1784–1797, 2006.

[57] Stephane Lafon and Ann B Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE transactions on pattern analysis and machine intelligence*, 28(9):1393–1403, 2006.

[58] Stéphane S Lafon. *Diffusion maps and geometric harmonics*. PhD thesis, Yale University, 2004.

[59] Christian D Langevin, Daniel T Thorne Jr, Alyssa M Dausman, Michael C Sukop, and Weixing Guo. Seawat version 4: A computer program for simulation of multi-species solute and heat transport. Technical report, Geological Survey (US), 2008.

[60] J.W. Larson, M. Hegland, B. Harding, S. Roberts, L. Stals, A.P. Rendell, P. Strazdins, M.M. Ali, C. Kowitz, R. Nobes, J. Southern, N. Wilson, M. Li, and Y. Oishi. Fault-tolerant grid-based solvers: Combining concepts from sparse grids and mapreduce. *Procedia Computer Science*, 18:130–139, 2013.

[61] Loic Le Gratiet and Josselin Garnier. Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. *International Journal for Uncertainty Quantification*, 4(5), 2014.

[62] Seungjoon Lee, Ioannis G Kevrekidis, and George Em Karniadakis. Resilient algorithms for reconstructing and simulating gappy flow fields in CFD. *Fluid Dynamics Research*, 47(5):051402, 2015.

[63] Seungjoon Lee, Ioannis G. Kevrekidis, and George Em Karniadakis. A general CFD framework for fault-resilient simulations based on multi-resolution information fusion. *Journal of Computational Physics,* under review, 2016.

[64] Xuejin Li, Zhangli Peng, Huan Lei, Ming Dao, and George Em Karniadakis. Probing red blood cell mechanics, rheology and dynamics with a two-component multi-scale model. *Phil. Trans. R. Soc. A*, 372(2021):20130389, 2014.

[65] R Little and D Rubin. *Statistical analysis with missing data*. New York: Wiley-Interscience, 2002.

[66] Rainald Löhner. An adaptive finite element scheme for transient problems in CFD. *Computer Methods in Applied Mechanics and Engineering*, 61(3):323–338, 1987.

[67] Hatem Ltaief, Edgar Gabriel, and Marc Garbey. Fault tolerant algorithms for heat transfer problems. *Journal of Parallel and Distributed Computing*, 68(5):663–677, 2008.

[68] Robert Lucas, J Ang, K Bergman, S Borkar, W Carlson, L Carrington, G Chiu, R Colwell, W Dally, J Dongarra, et al. Top ten exascale research challenges. *DOE ASCAC subcommittee report*, pages 1–86, 2014.

[69] Didier Lucor and George Em Karniadakis. Noisy inflows cause a shedding-mode switching in flow past an oscillating cylinder. *Physical review letters*, 92(15):154501, 2004.

[70] Xian Luo, Martin R Maxey, and George Em Karniadakis. Smoothed profile method for particulate flows: Error analysis and simulations. *Journal of Computational Physics*, 228(5):1750–1769, 2009.

[71] X. Ma, G. E. Karniadakis, H. Park, and M. Gharib. DPIV-driven flow simulation: a new computational paradigm. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 459(2031):547–565, 2003.

[72] B Murphy and T Gent. Measuring system and software reliability using an automated data collection process. *Quality and Reliability Engineering International*, 11:no. 5, 1995.

[73] Boaz Nadler, Stephane Lafon, Ronald Coifman, and Ioannis Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In *NIPS*, pages 955–962, 2005.

[74] Boaz Nadler, Stéphane Lafon, Ronald R Coifman, and Ioannis G Kevrekidis. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21(1):113–127, 2006.

[75] Yen Ting Ng, Chohong Min, and Frédéric Gibou. An efficient fluid–solid coupling algorithm for single-phase flows. *Journal of Computational Physics*, 228(23):8807–8829, 2009.

[76] XB Nie, SY Chen, and MO Robbins. A continuum and molecular dynamics hybrid method for micro-and nano-fluid flow. *Journal of Fluid Mechanics*, 500:55–64, 2004.

[77] J R Partington. *Interpolation, Identification, and Sampling*. Clarendon Press, 1997.

[78] L Parussini, D Venturi, P Perdikaris, and GE Karniadakis. Multi-fidelity gaussian process regression for prediction of random fields. *Journal of Computational Physics*, 336:36–50, 2017.

[79] P Perdikaris, M Raissi, A Damianou, ND Lawrence, and GE Karniadakis. Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling. In *Proc. R. Soc. A*, volume 473, page 20160751. The Royal Society, 2017.

[80] P Perdikaris, D Venturi, JO Royset, and GE Karniadakis. Multi-fidelity modelling via recursive co-kriging and Gaussian–Markov random fields. In *Proceedings of the Royal Society A*, volume 471, page 20150018. The Royal Society, 2015.

[81] Paris Perdikaris. *Data-Driven Parallel Scientific Computing: Multi-Fidelity Information Fusion Algorithms and Applications to Physical and Biological Systems*. PhD thesis, Brown University, 2015.

[82] Paris Perdikaris, Daniele Venturi, and George Em Karniadakis. Multifidelity information fusion algorithms for high-dimensional systems and massive data sets. *SIAM Journal on Scientific Computing*, 38(4):B521–B538, 2016.

[83] Dhiraj K Pradhan. *Fault-tolerant computer system design*. Prentice-Hall, Inc., 1996.

[84] Matej Praprotnik, Silvina Matysiak, Luigi Delle Site, Kurt Kremer, and Cecilia Clementi. Adaptive resolution simulation of liquid water. *Journal of Physics: Condensed Matter*, 19(29):292201, 2007.

[85] Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

[86] Carl Edward Rasmussen and C K I Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[87] AJ Roberts and IG Kevrekidis. Higher order accuracy in the gap-tooth scheme for large-scale dynamics using microscopic simulators. *ANZIAM Journal*, 46:637–657, 2005.

[88] Anthony J Roberts and Ioannis G Kevrekidis. General tooth boundary conditions for equation free modeling. *SIAM Journal on Scientific Computing*, 29(4):1495–1510, 2007.

[89] G Samaey, AJ Roberts, and IG Kevrekidis. Equation-free computation: An overview of patch dynamics. *Multiscale methods: bridging the scales in science and engineering*, page 216, 2010.

[90] Giovanni Samaey, Ioannis G Kevrekidis, and Dirk Roose. Patch dynamics with buffers for homogenization problems. *Journal of Computational Physics*, 213(1):264–287, 2006.

[91] Giovanni Samaey, Dirk Roose, and Ioannis G. Kevrekidis. The gap-tooth scheme for homogenization problems. *Multiscale Modeling & Simulation*, 4(1):278–306, 2005.

[92] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[93] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[94] B Schroeder and G A Gibson. Understanding failures in petascale computers. *Journal of Physics: Conference Series*, 78:12–22, 2007.

[95] Matthias Seeger, Christopher Williams, and Neil Lawrence. Fast forward selection to speed up sparse gaussian process regression. In *Artificial Intelligence and Statistics 9*, number EPFL-CONF-161318, 2003.

[96] LEE Seungjoon, I. G. Kevrekidis, and G. E. Karniadakis. A resilient and efficient cfd framework: statistical learning tools for multi-fidelity and heterogeneous information fusion. *Journal of Computational Physics,* under review, 2017.

[97] Fengyan Shi, James T Kirby, Jeffrey C Harris, Joseph D Geiman, and Stephan T Grilli. A high-order adaptive time-stepping TVD solver for Boussinesq modeling of breaking waves and coastal inundation. *Ocean Modelling*, 43:36–51, 2012.

[98] S. Sirisup, G. E. Karniadakis, Y. Yang, and D. Rockwell. Wave-structure interaction: Simulation driven by quantitative imaging. *Proceedings: Mathematical, Physical and Engineering Sciences*, 460(2043):729–755, 2004.

[99] Sirod Sirisup, George Em Karniadakis, Dongbin Xiu, and Ioannis G. Kevrekidis. Equation-free/Galerkin-free POD-assisted computation of incompressible flows. *Journal of Computational Physics*, 207(2):568 – 587, 2005.

[100] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of applied mathematics*, 45(3):561–571, 1987.

[101] Jeffrey Slotnick, Abdollah Khodadoust, Juan Alonso, David Darmofal, William Gropp, Elizabeth Lurie, and Dimitri Mavriplis. Cfd vision 2030 study: a path to revolutionary computational aerosciences. 2014.

[102] Alexander J Smola, Peter Bartlett, et al. Sparse greedy gaussian process regression. *Advances in neural information processing systems*, 13:619–625, 2001.

[103] M Stein. *Interpolation of spatial data: some theory for Kriging.* Berlin: Springer, 1999.

[104] George Sugihara, Robert May, Hao Ye, Chih-hao Hsieh, Ethan Deyle, Michael Fogarty, and Stephan Munch. Detecting causality in complex ecosystems. *science*, 338(6106):496–500, 2012.

[105] DL Sun, ZG Qu, YL He, and WQ Tao. An efficient segregated algorithm for incompressible fluid flow and heat transfer problems—ideal (inner doubly iterative efficient algorithm for linked equations) part i: Mathematical formulation and solution procedure. *Numerical Heat Transfer, Part B: Fundamentals*, 53(1):1–17, 2008.

[106] B-T Tan, K Willcox, and M Damodaran. Applications of proper orthogonal decomposition for inviscid transonic aerodynamics. *AIAA*, pages AIAA2003–4213, 2003.

[107] Dong Tang, Ravishankar K Iyer, and Sujatha S Subramani. Failure analysis and modeling of a vaxcluster system. In *Fault-Tolerant Computing, 1990. FTCS-20. Digest of Papers., 20th International Symposium*, pages 244–251. IEEE, 1990.

[108] Yu-Hang Tang, Shuhei Kudo, Xin Bian, Zhen Li, and George Em Karniadakis. Multiscale universal interface: A concurrent framework for coupling heterogeneous solvers. *Journal of Computational Physics*, 297:13–31, 2015.

[109] Balth Van der Pol. The nonlinear theory of electric oscillations. *Proceedings of the Institute of Radio Engineers*, 22(9):1051–1086, 1934.

[110] Daniele Venturi and George Em Karniadakis. Gappy data and reconstruction procedures for flow past a cylinder. *Journal of Fluid Mechanics*, 519:315–336, 2004.

[111] Oskar Wanner and Willi Gujer. A multispecies biofilm model. *Biotechnology and bioengineering*, 28(3):314–328, 1986.

[112] E Weinan and Bjorn Engquist. Multiscale modeling and computation. *Notices of the AMS*, 50(9):1062–1070, 2003.

[113] Thomas Werder, Jens H Walther, and Petros Koumoutsakos. Hybrid atomistic–continuum method for the simulation of dense fluid flows. *Journal of Computational Physics*, 205(1):373–390, 2005.

[114] Karen Willcox. Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition. *Computers & Fluids*, 35(2):208–226, 2006.

[115] Dongbin Xiu and George Em Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of computational physics*, 187(1):137–167, 2003.

[116] Or Yair, Talmon Ronen, Ronald R. Coifman, and Ioannis G. Kevrekidis. No equations, no parameters, no variables: data, and the reconstruction of normal forms by learning informed observation geometries. *arXive preprint*, 2016.

[117] F Yates. The analysis of replicated experiments when the field results are incomplete. *Empire Journal of Experimental Agriculture*, 1:129–142, 1933.

[118] Gour-Tsyh Yeh and Vijay S Tripathi. A model for simulating transport of reactive multispecies components: model development and demonstration. *Water Resources Research*, 27(12):3075–3094, 1991.

[119] Ben Yanbin Zhao, John Kubiatowicz, Anthony D Joseph, et al. Tapestry: An infrastructure for fault-tolerant wide-area location and routing, UCB/CSD-01-1141. Technical report, 2001.