Summer 2021

Reduced Order Modeling Based on Hybrid Snapshot Simulation

Feng Bai

Follow this and additional works at: <https://scholarcommons.sc.edu/etd>

Part of the [Mechanical Engineering Commons](#)

Recommended Citation

Bai, F.(2021). *Reduced Order Modeling Based on Hybrid Snapshot Simulation*. (Doctoral dissertation). Retrieved from <https://scholarcommons.sc.edu/etd/6438>

This Open Access Dissertation is brought to you by Scholar Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Scholar Commons. For more information, please contact dillarda@mailbox.sc.edu.

REDUCED ORDER MODELING BASED ON HYBRID SNAPSHOT SIMULATION

by

Feng Bai

Bachelor of Engineering
Tianjin University of Commerce 2009

Master of Science
Shanghai Jiaotong University 2013

Master of Science
Missouri University of Science & Technology 2017

Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy in
Mechanical Engineering
College of Engineering and Computing
University of South Carolina
2021

Accepted by:

Yi Wang, Major Professor

Guiren Wang, Committee Member

Lang Yuan, Committee Member

Zhu Wang, Committee Member

Tracey L. Weldon, Interim Vice Provost and Dean of the Graduate School

© Copyright by Feng Bai, 2021
All Rights Reserved.

ACKNOWLEDGMENTS

This dissertation is really hard work yet is interesting and valuable. I would like to thank all who give me advice, technical suport, finicial support and mental health service:

Advisor of this dissertation: Dr. Yi Wang, shows his kind support and advice to this dissertation work. His advicements let me know how to bridge academics and applications, which is the key of this dissertation. Also the techniques he teaches are very vital and valuable to the students.

My committee members: Dr. Zhu Wang, Dr. Guiren Wang, Dr. Lang Yuan, thank you very much for your kindly support and evaluation in this dissertation and other useful issues including academic knowledge and miscellaneous suggestion. The work experience with you is very nice and I do learn a lot in your valuable help.

My friends, mentors and classmates: Dr. Xiaofeng Yang, Dr. Xiangcheng Zheng, Dr. Zhiyu Wang, Dr. Shuai Yuan, Dr. Peter Binev, Dr. Joshua Cooper, Dr. Wolfgang Dahmen, Dr. Xiaoming He at Missouri S&T, Dr. Daozhi Han at Missouri S&T, Dr. Wan Shou at MIT, Dr. Chao Zeng at PNNL, Dr. Yandong Zhang, et. al. It is very nice experience during my study.

My parents: Qitai Hu and Fengying Hu, thank you very much for your support and love.

ABSTRACT

Reduced order model (ROM) is an emerging technique for rapid and efficient numerical simulation and computation in science and engineering applications. The ROM consists of two stages: (1) the offline snapshot simulation to produce high-quality snapshot data representative of model trajectories, from which the projection subspace can be extracted and used for ROM construction; and (2) the online simulation to utilize the ROM built in stage 1. The snapshot simulation is traditionally performed with full order model (FOM) and is computationally demanding, which is one of the most critical limiting factors for the development and deployment of ROM in real-world applications. In this study, a new hybrid snapshot simulation framework and algorithms are proposed to tackle this challenge by accelerating the offline snapshot simulation and reducing its computational resource usage.

In Chapter.2 of this dissertation, a hybrid snapshot simulation methodology and framework that alternates the simulation automatically between FOM and ROM is proposed, which for the first time adopts local ROMs to accelerate the snapshot data generation and demonstrates up to 50% speedup over the traditional method. In Chapter.3, the proposed hybrid snapshot simulation method is extended to construct ROM with discrete empirical interpolation method (DEIM), which enables fast reconstruction of nonlinear terms in the ROM and significantly speeds up the online simulation. In Chapter.4, a new approach to embed the DEIM into the hybrid snapshot simulation is presented, which gains additional acceleration by 10.5% $\sim$ 27.8% relative to that developed in Chapter.2. In Chapter.5, a hyper-reduction method is introduced into the proposed hybrid simulation framework. It is based on the Gauss-

Newton solver with approximated tensors(GNAT) at the level of discretized algebraic equations to realize additional system approximation of the nonlinear dynamic systems. Its potential to improve computational performance is examined thoroughly with the 2D Burgers equation. It is found that the GNAT-embedded hybrid snapshot simulation achieves 33% computational acceleration relative to the traditional FOM-based snapshot simulation for the example problem: flow of weak gradients ($Re \leq 100$) while preserving ROM accuracy, viz., $RMSE \sim O(10^{-3})$.

All the studies conducted in this dissertation convincingly verify the feasibility of the proposed hybrid snapshot simulation methodology. Additional system identification through DEIM or GNAT could further improve its computational efficiency.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	iii
ABSTRACT	iv
LIST OF TABLES	ix
LIST OF FIGURES	xi
CHAPTER 1 INTRODUCTION	1
1.1 Background	1
1.2 Outline of Dissertation	6
CHAPTER 2 REDUCED ORDER MODEL BASED ON HYBRID SNAPSHOT SIMULATION	8
2.1 Methodology of Hybrid Snapshot Simulation	10
2.2 Numerical Models of Heat Transfer	20
2.3 Results and Discussions	24
2.4 Summary of Chapter.2	42
CHAPTER 3 DEIM REDUCED ORDER MODEL BASED ON HYBRID SNAP- SHOT SIMULATION	45
3.1 Methodology of Hybrid Snapshot Simulation for ROM-DEIM Con- struction	46
3.2 DEIM and Cell-centered FVM Implementation	54

3.3	Numerical Model of Heat Transfer	60
3.4	Numerical Results and Discussions	65
3.5	Summary of Chapter.3	84
CHAPTER 4 DEIM-EMBEDDED HYBRID SNAPSHOT SIMULATION FOR REDUCED ORDER MODEL GENERATION		87
4.1	Methodology of DEIM-Embedded Hybrid Snapshot Simulation	88
4.2	Constructions of FOM, ROM and ROM-DEIM	99
4.3	Numerical Model of Heat Transfer	103
4.4	Numerical Configurations, Results and Discussions	106
4.5	Summary of Chapter.4	122
CHAPTER 5 A REDUCED ORDER MODELING METHOD BASED ON GNAT- EMBEDDED HYBRID SNAPSHOT SIMULATION		124
5.1	Petrov-Galerkin projection and Gauss-Newton approximated tensors .	125
5.2	Methodology of GNAT-embedded hybrid snapshot approach	130
5.3	The FOM, ROM-PG and ROM-GNAT based on finite volume dis- cretization of 2D Burgers equation	138
5.4	Snapshot data generation and model switch criteria in GNAT- embedded hybrid snapshot simulation	148
5.5	Numerical experiments and discussion	153
5.6	Summary of Chapter.5	168
5.7	Appendix to Chapter.5: The coefficient matrix of the finite volume discretization of the 2D Burgers equation	171
CHAPTER 6 CONCLUSIONS AND FUTURE WORK		175

BIBLIOGRAPHY	180
------------------------	-----

LIST OF TABLES

Table 2.1	Material and thermal properties of the constituent materials	27
Table 2.2	The breakdown of the computational time for the hybrid snapshot simulation using a time step size $\delta t = 0.25s$ for the annular cone model(Unit:s)	30
Table 2.3	The breakdown of the computational time for the hybrid snapshot simulation using a time step size of $\delta t = 0.5s$ for the fin array model(Unit:s)	37
Table 3.1	The breakdown of the computational time for the hybrid snapshot simulation for the annular cone model (Unit:s)	70
Table 3.2	The comparison of computational time in FOM , HROM , and HROM-DEIM in the annular cone model. (Unit:s).	75
Table 3.3	The breakdown of the computational time for the hybrid snapshot simulation for the extended fins model (Unit:s)	78
Table 3.4	The comparison of computational time in FOM , HROM , and HROM-DEIM in the extended fin array model. (Unit:s).	83
Table 4.1	Thermal properties of the constituent materials with constant values(SS: Stainless Steel; MW: Mineral Wool) .	106
Table 4.2	Thermal properties of the constituent materials with temperature-dependent properties(SS: Stainless Steel; MW: Mineral Wool)	107
Table 4.3	The breakdown of the computational time for DEIM-embedded hybrid snapshot simulation with the DEIM number 30 in the annular cone model (Unit:s)	112
Table 4.4	The comparison of computational time in FOM , HROM , and HROM-DEIM in the annular cone model. (Unit:s).	114

Table 4.5	The list of the computational time for DEIM-embedded hybrid snapshot simulation with the DEIM number 70 and the time step size $\delta t = 0.5s$ in the extended fin array model(Unit:s)	118
Table 4.6	The comparison of computational time in FOM , HROM , and HROM-DEIM in the extended fin array model. (Unit:s).	121
Table 5.1	Spatial convergence by refining the mesh $n_x \times n_y$. In the L_∞ norm errors the reference solutions are the exact solutions. $\Delta h = \Delta x = \Delta y = 1/n_x = 1/n_y$. All the errors are calculated at $t = 1$ with the time step $\delta t = 0.001$	155
Table 5.2	Computational time usage(Unit:s) of the three models: FOM, ROM-PG and ROM-GNAT with $Re = 10$, $Re = 40$ and $Re = 100$. $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$, $\delta t = 0.001$ are used.	161
Table 5.3	The list of model type and the computational time (Unit:s) in each interval of the GNAT-embedded hybrid snapshot simulation. $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$, $\epsilon = 10^{-2}$, $\epsilon^R = 10^{-2}$, 64×64 mesh configuration in computational domain Ω , and $\delta t = 0.001$ are used.	164

LIST OF FIGURES

Figure 2.1	Schematic of the hybrid snapshot simulation method involving FOM/ROM alternation.	11
Figure 2.2	Finite volume method (FVM) based representation of FOM in a 2D computational domain	21
Figure 2.3	Computational model for the multi-layer annular cone: (a) Geometry and computational grids; and (b) The heat flux profile prescribed at the three quarters of the outer wall of the computational model and used in the snapshot simulation and verification simulation.	25
Figure 2.4	The FOM/ROM alternation in the hybrid snapshot simulation in the annular cone model.	28
Figure 2.5	The spatiotemporal distribution of the snapshots solution produced by (a) SS-FOM and (b) SS-Hybrid in the annular cone model.	29
Figure 2.6	Comparison between SS-FOM and SS-Hybrid for the annular cone model in terms of (a) the absolute error in the entire domain; and (b) the MSRD error in the temporal domain.	29
Figure 2.7	The comparison between (a)FROM and (b)HROM in terms of the temperature solutions during the verification simulation for the annular cone model.	31
Figure 2.8	Relative error of the FROM and HROM relative to the FOM for the annular cone model.	32
Figure 2.9	Contour plot of the temperature solution obtained from the FOM analysis (top row) and the HROM (bottom row) at various instants of time (a) 60 s, (b) 120 s, and (c) 240 s.	33

Figure 2.10	Computational model for the fin array: (a) Geometry and computational grids; and (b) The heat flux profile applied at the bottom of the fin array in the snapshot simulation and verification simulation.	34
Figure 2.11	The FOM/ROM alternation in the hybrid snapshot simulation for the fins array model.	35
Figure 2.12	The spatiotemporal distribution of the snapshots solution produced by (a) SS-FOM and (b) SS-Hybrid in the annular cone model.	35
Figure 2.13	Comparison between SS-FOM and SS-Hybrid for the fins array model in terms of (a) the absolute error in the entire spatiotemporal domain; and (b) the MSRD error in the temporal domain. .	36
Figure 2.14	The comparison between FROM and HROM in the temperature solutions during the verification simulation for the fins array model.	38
Figure 2.15	Comparison between FROM and HROM in terms of the MSRD error in the temporal domain for the fins array model.	39
Figure 2.16	Contour plot of the temperature in the simulation domain obtained from the FOM analysis (top row) and the HROM (bottom row) for the fins array model at various instants of time (a) 240 s, (b) 306 s, and (c) 540 s.	40
Figure 2.17	The FOM/ROM alternation in the hybrid snapshot simulations for the fins array model with different tolerance parameters (a), (b), (c), and (d).	40
Figure 2.18	Comparison of the MSRD errors of SS-Hybrid with different tolerance parameters for the fins array model.	43
Figure 3.1	The proposed methodology to construct ROM-DEIM based on hybrid snapshot simulation: (1) Hybrid snapshot simulation; (2) determination of DEIM points/cells at the end of the hybrid simulation; and (3) verification simulation utilizing ROM-DEIM. .	49
Figure 3.2	The procedure to assemble snapshot data for nonlinear terms using a mix of FOM and reconstructed ROM solutions from the hybrid snapshot simulation.	54

Figure 3.3	The procedure for identifying the (a) DEIM cells; (b) surrounding faces (red lines), and (c) adjacent cells (Ad) of the DEIM cells in a 2D computational domain.	57
Figure 3.4	The data structure used for nonlinear terms assembly at the DEIM cells: (a) Cell-to-Face; and (b) Face-to-Cell.	58
Figure 3.5	The numerical model of the multi-layer annular cone: (a) geometry and computational grids; and (b) the heat flux profile prescribed at the three quarters of the outer wall of the model and used in the snapshot simulation (in pink) and the online simulation (in green).	66
Figure 3.6	The model of extended fin arrays: (a) Geometry and grids; and (b) The heat flux profile prescribed at the bottom of the substrate of the computational model and used both in the snapshot simulation (in pink) and the online simulation (in green).	67
Figure 3.7	: The comparison between SS-FOM and SS-Hybrid in the annular cone model: (a) the snapshot data produced by SS-FOM and (b) the absolute error between SS-FOM and SS-Hybrid.	69
Figure 3.8	The comparison of FROM and HROM with FOM for the annular cone model: the temperature solution of all the computational cells throughout the verification simulation obtained by (a-1) FROM and (b-1) HROM; and the absolute error Err_{abs} of (a-2) FROM and (b-2) HROM relative to FOM.	71
Figure 3.9	The temperature solutions obtained by HROM-DEIM with the different number of DEIM cells in the annular cone model: (a) DEIM 40 ($m_A = 40$), (b) DEIM 60 ($m_A = 60$), (c) DEIM 80 ($m_A = 80$), and (d)non-DEIM.	73
Figure 3.10	The absolute error of the temperature solution obtained by HROM-DEIM with the different number of DEIM cells in the annular cone model relative to the FOM solution: (a) DEIM 40 ($m_A = 40$), (b) DEIM 60 ($m_A = 60$), (c) DEIM 80 ($m_A = 80$), and (d)non-DEIM.	74
Figure 3.11	The MSRD errors vs. time for HROM-DEIM with the different number of DEIM cells and HROM without DEIM in the verification simulation in the annular cone model.	75

Figure 3.12	Effect of the temperature-dependent thermal conductivity on model nonlinearity: (a) the temperature solution obtained by FOM in the annular cone model using the constant thermal conductivity; and (b) the absolute error between the solution in (a) and the solution obtained using temperature-dependent thermal conductivity.	77
Figure 3.13	The comparison between SS-FOM and SS-Hybrid in the extended fin array model: (a) the snapshot data produced by SS-FOM and (b) the absolute error between SS-FOM and SS-Hybrid.	77
Figure 3.14	The comparison of FROM and HROM with FOM for the extended fin array model: the temperature solution of all the computational cells throughout the verification simulation obtained by (a-1) FROM and (b-1) HROM; and the absolute error of (a-2) FROM and (b-2) HROM relative to FOM.	80
Figure 3.15	The temperature solutions obtained by HROM-DEIM with the different number of DEIM cells in the extended fin array model: (a) DEIM 50 ($m_A = 50$), (b) DEIM 75 ($m_A = 75$), (c) DEIM 100 ($m_A = 100$), and (d)non-DEIM.	81
Figure 3.16	The absolute error of the temperature solution obtained by HROM-DEIM with the different number of DEIM cells in the extended fin array model relative to the FOM solution: (a) DEIM 50 ($m_A = 50$), (b) DEIM 75 ($m_A = 75$), (c) DEIM 100 ($m_A = 100$), and (d)non-DEIM.	82
Figure 3.17	The MSRD errors vs. time for HROM-DEIM with the different number of DEIM cells and HROM without DEIM in the verification simulation in extended fin array model.	83
Figure 3.18	Effect of the temperature-dependent thermal conductivity on model nonlinearity: (a) the temperature solution obtained by FOM in the extended fin array model using the constant thermal conductivity; and (b) the absolute error between the solution in (a) and the solution obtained using temperature-dependent thermal conductivity.	84

Figure 4.1	The methodology of DEIM-embedded hybrid snapshot simulation and the construction of ROM-DEIM in online simulation:(1) the whole process of DEIM-embedded hybrid snapshot simulation; (2) the SVD/iSVD updating of $\mathbf{U}_r$ and $\mathbf{U}_r^H$ and DEIM algorithm after each FOM interval(if switch to ROM or ROM-DEIM); (3) the SVD/iSVD updating of $\mathbf{U}_r^H$ and DEIM algorithm after each ROM interval(if switch to ROM-DEIM); (4) no updating in basis after each ROM-DEIM interval; (5) the online simulation in ROM-DEIM constructed by $\mathbf{U}_r$ and $\mathbf{U}_r^H$ in the hybrid snapshot simulation.	90
Figure 4.2	The flow chart of DEIM-embedded hybrid snapshot simulation.	92
Figure 4.3	The configurations of multi-layer annular cone model: (a) Geometry; (b) The heat flux profile prescribed at the three quarters of the outer wall of the computational model and used in both the snapshot simulation (in red) and the online simulation (in green).	107
Figure 4.4	The configurations of extended fins model: (a) Geometry; (b) The heat flux profiles are prescribed at the bottom of the substrate in the computational model and used in both the snapshot simulation (in red) and the online simulation (in green).	108
Figure 4.5	The plots of FOM solutions produced using (a) the constant properties, (b) the temperature dependent properties and (c) the absolute error between (a) and (b) in the cone model.	110
Figure 4.6	The numerical solutions of the snapshot data for the annular cone model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with DEIM number 30, respectively.	110
Figure 4.7	The absolute errors of the snapshot data for the annular cone model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with the DEIM number 30, respectively.	111
Figure 4.8	The numerical solutions of verification simulation for the annular cone model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)30, (c)40, and (d)80, respectively.	113

Figure 4.9	The absolute errors of the numerical solutions in the verification simulation for the annular cone model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)30, (c)40, and (d)80, respectively.	114
Figure 4.10	The MSRD errors of the numerical solutions in the verification simulation for the annular cone model simulated by HROM (in black) and HROM-DEIM with three different DEIM numbers: 30 (in red), 40 (in green), and 80 (in blue), respectively.	115
Figure 4.11	The plots of FOM solutions computed using (a) the constant material properties; (b) the temperature-dependent properties; and (c) the absolute error between (a) and (b) in the extended fin array model.	116
Figure 4.12	The numerical solutions of the snapshot data for the extended fin array model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with the DEIM number 40, respectively.	116
Figure 4.13	The absolute errors of the snapshot data for the extended fin array model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with the DEIM number 40, respectively.	117
Figure 4.14	The numerical solutions of verification simulation for the extended fin array model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)40, (c)70, and (d)150, respectively.	119
Figure 4.15	The absolute errors of the numerical solutions in the verification simulation for the extended fin array model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)40, (c)70, and (d) 150, respectively.	120
Figure 4.16	The MSRD errors of the numerical solutions in the verification simulation for the extended fin array model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: 40 (in red), 70 (in green), and 150 (in blue), respectively.	121

Figure 5.1	The methodology of GNAT-embedded hybrid snapshot simulation and the construction of ROM-GNAT for online simulation:(1) the procedure of GNAT-embedded hybrid snapshot simulation; (2) the SVD/iSVD updating of Φ_Y , Φ_R , Φ_J and the "greedy" algorithm at the end of each FOM interval(if the simulation will switch to ROM-GNAT); (3) the SVD/iSVD updating of Φ_R , Φ_J and the "greedy" algorithm at the end of each ROM-PG interval(if switch to ROM-GNAT); (4) no updating in POD basis after each ROM-GNAT interval; (5) the ROM-GNAT in online simulation is constructed by Φ_Y , Φ_R and Φ_J obtained in the GNAT-embedded hybrid snapshot simulation. . .	131
Figure 5.2	The flow chart of GNAT-embedded hybrid snapshot simulation. .	134
Figure 5.3	The FOM solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$. . .	156
Figure 5.4	The L_∞ norm errors of (a) $L_\infty(u)$ and (b) $L_\infty(v)$ with the three different values of Re : (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$	156
Figure 5.5	The ROM-PG solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 4$ is used in the ROM-PG.	158
Figure 5.6	The absolute errors of ROM-PG solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time of $\delta t = 0.001$ and $r_u = r_v = 4$ is used in the ROM-PG.	159
Figure 5.7	The RMSE of ROM-PG solutions of $\vec{u}$ (left) and $\vec{v}$ (right) with the three different Re numbers: $Re = 10$, $Re = 40$ and $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$. In the upper row: (a) $r_u = r_v = 4$; and in the lower row (b) $r_u = r_v = 12$ are used in the ROM-PG.	160

Figure 5.8	The ROM-GNAT solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$ is used in the ROM-GNAT.	161
Figure 5.9	The absolute errors of the ROM-GNAT solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$ is used in the ROM-GNAT.	162
Figure 5.10	The RMSE of the ROM-GNAT solutions of $\vec{\mathbf{u}}$ (left) and $\vec{\mathbf{v}}$ (right) with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$ is used in the ROM-GNAT.	162
Figure 5.11	The GNAT-embedded hybrid snapshot solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.	165
Figure 5.12	The absolute errors of GNAT-embedded hybrid snapshot solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.	166
Figure 5.13	The RMSE of GNAT-embedded hybrid snapshot solutions of $\vec{\mathbf{u}}$ (left) and $\vec{\mathbf{v}}$ (right) with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.	166

Figure 5.14	The ROM-GNAT (obtained by the hybrid snapshot simulation) solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers in the online simulation: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.	167
Figure 5.15	The absolute errors of the ROM-GNAT (obtained by the hybrid snapshot simulation) solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers in the online simulation: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.	168
Figure 5.16	The RMSE of the ROM-GNAT (obtained by the hybrid snapshot simulation) solutions of $\vec{\mathbf{u}}$ (left) and $\vec{\mathbf{v}}$ (right) with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.	169
Figure 5.17	The flux directions used by the Godunov scheme in 2D mesh cells.	171

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Efficient numerical modeling and computational techniques play continuously increasing roles in science and engineering aspects, and are being developed at unprecedented pace to address urgent needs in almost every aspect of fundamental research and engineering applications, starting from physical phenomena, conceptual design, high-fidelity analysis, real-time simulation, design optimization, etc. Two specific areas of interest for numerical computation are high-performance computing techniques that utilize ever-increasing, highly parallelized computing architecture and hardware to perform demanding simulations (e.g., multi-scale and/or multiphysics); and the reduced order modeling (ROM) algorithms to significantly reduce the model dimension by projecting the full-order model (FOM) onto low-dimensional manifold or subspace, leading to lower usage of computational resources and time. In this context, various innovative ROM techniques have been developed broadly in the past two decades for simulating large, physics-based numerical problems on resource-limited computing platforms [97, 100, 8, 101, 20, 22, 27, 45, 6, 93, 72, 2, 15, 34, 26, 36, 88]. Normally the ROM needs to be constructed by using representative snapshot simulation data, from which Proper Orthogonal Decomposition (POD) basis vectors can be extracted for projection [14]. Traditionally, the snapshot simulation is performed by FOM simulation to achieve high data quality. However, in this dissertation we will focus on investigating, developing, and validating the hybrid snapshot simulation approaches

to accelerate the ROM constructions and improve the scalability and applicability of nonlinear ROMs.

The early research of ROM techniques focused on the Galerkin projection and the POD [76, 57, 97] which has been verified in various applications of the numerical computations, including the thermal macro-models in MEMS [100]; the incompressible flow problem [84]; the scalable macromodels in microfluidics [101]; the ROM using the interpolation method for aerospace problems [4]; the cavity problems solved by the local POD plus Galerkin projection [75, 89]; the thermal analysis in microfluidic devices [93] and spacecraft and aerospace vehicles [74, 56], etc.

In general, the ROM techniques consists of two phases: (1) the offline snapshot simulation for basis vector extraction and ROM construction using representative model parameters; and (2) the online verification simulation that computes the constructed ROM using basis vectors obtained from the offline snapshot simulation stage with the model parameters different from the snapshot simulation. The snapshot simulation is usually performed using the high-fidelity FOM to generate snapshot data, and the projection basis vectors can be determined using the POD of the snapshot data, one of the most widely used low-rank approximation methods for ROMs [85]. In POD, the singular value decomposition (SVD) or the eigenvalue decomposition (e.g., the method of snapshots [85]) is applied onto the entire snapshot data, and significance of the basis vectors also needs to be ranked. The computational cost could be reduced using snapshot data selection methods that have been proposed to accelerate the simulation and basis extraction and to minimize generation of redundant FOM information. An adaptive snapshot simulation method with error estimation was studied and the convergence analysis was also performed [1]. An adaptive snapshot simulation based on the weak form of partial differential equations(PDEs) with Galerkin projection and finite element basis was also demonstrated [91]. A parallelization technique was proposed for snapshot generation by dividing the entire snapshot

data computation into multiple parallel threads, and then the snapshot simulation could be performed in multi-processors for enhanced computational efficiency [96].

Hybrid computational methods by combining FOMs and local ROMs into a streamlined numerical simulation were also reported for computational acceleration. Important criteria for FOM/ROM model switches were introduced in [75, 89] based on the importance of each individual singular values and the trajectory of ROM. Hybrid methods for adaptive dimension reduction and ROM construction to numerically solve PDEs efficiently were discussed in [44]. An approach for online model order reduction based on selective skip of the time steps during FOM simulation was developed for rapidly computing structural deformation [53]. Correspondingly, approaches for subspace basis updating and local ROM construction were also proposed, that is, when new FOM data becomes available, the orthogonal basis of the subspace is updated through rotation and dilation, and the ROM structure is modified by re-projection onto the new basis. There are many methods that can be used for basis updating, including the fast low-rank SVD, such as incremental SVD (iSVD) [17] and “Rank-One” modification [18], and the sequential POD and basis extension [52, 21, 48, 55, 49]. In particular, the rank one modification was also adopted in online adaptive data-driven ROMs for POD basis updating [70] and adaptive interpolation [71]. The PDE models with Galerkin projection need to accommodate the continuous time data with a discrete time incremental POD [41]. The merits of iSVD could also be utilized for adaptive snapshot data selection to cut down the time and memory usage in POD, and the snapshot data could be selected with respect to the approximation error [67]. Despite these seminal works of the hybrid methods, research that can accelerate the snapshot simulation itself and minimize the offline computational cost has not been reported. In contrast to a specific, online simulation, model parameters for the snapshot simulation need to be more representative and accommodating to various scenarios that will be encountered in the verification simulation. More importantly,

it is normal that snapshot data are generated exclusively by FOM simulation for enhanced quality. This is because the results obtained by FOM are regarded as the benchmark, and yield the most accurate POD basis vectors of the solution subspace.

Because the projection subspace of the nonlinear terms is different from that of the solution of PDEs, efficient approximation of the nonlinear terms in low-dimensional subspace during ROM simulation is a formidable challenge, leading to low computational efficiency. Several techniques to combat and overcome this difficulty have been proposed, including the trajectory piecewise linear approximation (TPWL) [77, 78] that approximates the nonlinear terms as a Taylor expansion around linearization points; the hyper-reduction based on the Gauss-Newton method with approximated tensors (GNAT) that minimizes the solution residual based on Petrov-Galerkin projection and reduces the nonlinear term reconstruction with “gappy” POD [23, 25, 51]; and the missing point estimation (MPE) method to compute Galerkin projections over a restricted subset of the spatial domain [8, 104]. In addition, the discrete empirical interpolation method (DEIM) [27, 29, 69] is another very popular approach of nonlinear term approximation, and has been demonstrated to improve computational efficiency of various numerical problems, such as the fluid flow and Navier-Stokes problems [68], viscoplasticity [46], cardiac mechanics [16], and electromagnetic POD-DEIM models [64].

In order to conquer the demanding computational costs of constructing nonlinear ROMs, in this dissertation several new algorithms and methodologies are proposed and developed to accelerate snapshot simulation and ROM generation, and improve utilization and efficiency of ROM. Different extents of approximation are used in the local ROMs to conquer the nonlinearity for the reduction of computational costs in snapshot simulation. FOM, ROM, and ROM with additional nonlinear approximation (e.g., discrete empirical interpolation method/DEIM and the Gauss-Newton with approximated tensors/GNAT) form a hierarchy of model accuracy and computational

cost, from the most accurate and costly to the least. Therefore, they can be utilized selectively to meet different requirements of POD basis generation and ROM updating during the hybrid snapshot simulation. FOM offers the highest-fidelity information, and should be used only when necessary, viz., when all ROMs and their POD modes are inaccurate and need to be updated. Although providing no new information of POD modes, ROM can be used to enrich the snapshot data of nonlinear terms, and enhance ROMs with additional system approximation. On the other hand, while employing more approximations, ROM-DEIM or ROM-GNAT is the most efficient model, yet the lowest accurate, and its use should be maximized at any time when allowed.

The discrete empirical interpolation method(DEIM) is an efficient approach to approximate the nonlinear terms by interpolating their values at a small number of interpolation points within the POD subspace of the nonlinear terms. Thus, the computational cost of assembling the nonlinear terms during simulation could be greatly reduced. The Gauss-Newton with approximated tensors (GNAT), unlike DEIM, decreases the dimension of model systems by using a least-squares Petrov-Galerkin projection and also reduces the computational costs by approximating the residual and action of the Jacobian on the right projection subspace with the Gappy POD approach. In this dissertation, algorithms are designed to embedded DEIM or GNAT into the snapshot simulation and the online simulation are also proposed for efficient computation of the nonlinear terms in ROMs. By combining the hybrid snapshot simulation method with the nonlinear term and/or system approximations, such as DEIM and/or GNAT, representative snapshot data can be produced and ROM-DEIM and/or ROM-GNAT can be constructed with significant efficiency while maintaining salient accuracy.

1.2 OUTLINE OF DISSERTATION

This dissertation is separated into four parts to fulfill the tasks discussed above:

1. A hybrid snapshot simulation framework and methodology for efficient ROM construction is proposed. The whole snapshot simulation is divided into multiple time intervals, and each interval will be simulated by either FOM or ROM. The POD basis vectors will be generated or updated in each FOM interval, while remaining unchanged in each local ROM. A set of model switch criteria are also introduced to determine on-the-fly alternating the snapshot simulation between the FOM and the local ROM. At the end of the hybrid snapshot simulation, the POD basis vectors will be utilized to construct the ROM for the online simulation.

2. Built on the framework in Chapter.2, the focus of Chapter.3 is to construct ROM-DEIM that can be used to accelerate the online simulation. The method to combine the hybrid snapshot simulation, snapshot data generation, and DEIM-based ROM construction is developed. A strategy is proposed to address the challenge associated with a limited number of FOM intervals in the hybrid simulation and poor representation of the subspace underlying nonlinear terms and DEIM. A formulation to construct ROM-DEIMs in the cell-centered finite volume method (FVM) scheme along with properly devised data structures is introduced to consider both the componentwise and non-componentwise transport-related nonlinear terms that are widely encountered in physical phenomena.

3. The objective of the third research component in Chapter.4 is to embed DEIM into the snapshot simulation to further accelerate snapshot data generation. That is, the multiple intervals of the snapshot simulation will be computed by three types of models, i.e., FOM, ROM and ROM-DEIM. Two sets of the snapshot data and POD basis vectors, respectively, for the ROMs and the nonlinear terms in the ROM will be generated and updated during the simulation. The POD modes for ROM solu-

tion will be generated or updated using the snapshot data collected in FOMs, while the POD modes for the nonlinear terms to construct ROM-DEIM will be obtained using the snapshot data of nonlinear terms garnered in each FOM and local ROM. The corresponding model switch criteria are also presented to determine on-the-fly alternating the snapshot simulation among FOM, ROM, and ROM-DEIM.

4. The research topic of the fourth part in Chapter.5 follows the idea in Chapter.4 above except that the GNAT system approximation (instead of DEIM) will be utilized and performed for accelerating hybrid snapshot simulation by performing the FOM, ROM with Petrov-Galerkin projection(ROM-PG) and ROM with GNAT(ROM-GNAT), and the online simulation will be conducted with ROM-GNAT based on the POD modes obtained in the GNAT-embedded hybrid snapshot simulation. The other differences include gappy POD interpolation of the residual and the action of Jacobian on the right projection subspace, new model switch criteria to consider both the residual and the action of Jacobian, system approximation, i.e., hyper-reduction performed at the lower level of discretized algebraic equation, and demonstration of the validation in 2D Burgers' equation.

All the computational models presented are programmed and simulated in Matlab 2017b. All of the numerical tests are performed on a PC equipped with an Intel i7 – 7700 CPU with a clock speed of 3.60 GHz, and the RAM of 32 GB. The operating system is Windows 10 Enterprise 2016 *LTSB*.

CHAPTER 2

REDUCED ORDER MODEL BASED ON HYBRID SNAPSHOT SIMULATION

In this chapter, we propose an adaptive, hybrid snapshot simulation method to accelerate the snapshot data generation, POD basis vector computation, and ROM construction process. The novelties of the method include: (1) in contrast to the traditional method that only adopts FOM for high-fidelity snapshot data production, ours will be performed by a combination of the FOM and local ROM that alternate during the simulation. This dramatically reduces the computational time, and is particularly useful for constructing ROMs for large-scale, three dimensional problems. The methodology is grounded on the understanding that in many cases the data generated from the snapshot simulation are extremely redundant, as manifested by the high data reduction ratio [79] and the notable efficacy of the snapshot selection and adaptive time stepping [67]. Therefore, the present effort aims to decrease the number of snapshots computed by FOM. That is, the information-lean snapshots can be replaced or approximated by the local ROM constructed during simulation to minimize redundant data generation and accelerate the entire ROM development process; (2) the incremental singular value decomposition (iSVD) [17] is exploited to continuously update the POD modes and construct the local ROM during the snapshot simulation, and is well-suited for our hybrid snapshot simulation based on FOM/local ROM alternation. This substantially lowers both the memory usage and the numerical complexity relative to the traditional SVD. The FOM data used in

iSVD can be cleared from the memory immediately after each basis vector updating. Thus only one subset (or block) of the FOM data is kept within the memory usage at most. As demonstrated in our recent study [79] that manipulating data subset for subspace updating in iSVD takes much less computational time than the SVD that factorizes the entire snapshots data matrix; (3) the alternation between the FOM and the local ROM is also determined on-the-fly using the error estimators/criteria that observe the changes and shifts in the singular values and trajectory of the reduced system [75, 89, 44]; and (4) most importantly, the present effort represents a thorough investigation on the feasibility of using the hybrid method for snapshots data generation. Although the hybrid methods for a single, specific simulation has been reported [75, 89], their applicability to the snapshot simulation has not been established. This is because the variations in operating parameters (e.g., the initial, boundary, and volumetric conditions) in the snapshot simulation are usually generic and comprehensive, and different from those in the specific online simulation. It is the common practice in ROM research that snapshot data generation is exclusively performed by FOM simulation in order to maintain adequate snapshot data quality and POD basis representation and accuracy. Moreover, the combination of the adaptive alternation and incremental basis vector updating has not been demonstrated for snapshot simulation yet.

This chapter is organized as follows: in Section 2.1 we will introduce the methodology of hybrid snapshot simulation, including the formulation of the hybrid FOM/ROM method; FOM and ROM construction; incremental proper orthogonal decomposition (iPOD) for basis vector updating; and criteria for FOM/ROM switch. Example models of numerical heat transfer based on the finite volume method (FVM) are then developed in Section 2.2. In Section 2.3 two case studies are performed to demonstrate and verify the proposed hybrid snapshot simulation method by comparing their results against the traditional FOM snapshot simulation, and their computa-

tional performance is also quantitatively characterized. The ROMs constructed by both snapshot simulation methods are also compared with the FOMs. This chapter summaries with a technical summary and future research in Section 2.4.

2.1 METHODOLOGY OF HYBRID SNAPSHOT SIMULATION

In this section we will describe the methodology of the hybrid snapshot simulation, which includes four key components: (1) the hybrid process by alternating the computation between the FOM and ROM; (2) the FOM and local ROM construction during the hybrid simulation; (3) the iPOD to continuously update the POD basis vectors using the latest simulation data generated by the FOM; and (4) the criteria to decide the switch between the FOM and local ROM on-the-fly. Note that the methodology of the hybrid snapshot simulation presented in this section is independent of the physics modeled, and applicable to heat transfer, fluid dynamics, and many others as long as the governing numerical equations are available.

2.1.1 HYBRID FOM/ROM COMPUTATION

During the hybrid simulation, the computation by FOMs and local ROMs for generating the snapshot data will be alternated in an adaptive manner as shown in Fig.2.1. The total snapshot simulation in the whole time span is divided into multiple intervals of the equal length. The span of each interval is denoted as Δt and the i^{th} interval encompasses the time period between $(i - 1) \Delta t$ and $i \Delta t$, where $i \in [1, 2, \dots, N_i]$ and N_i is the total number of intervals in the snapshot simulation. In each interval either the FOM or ROM will be simulated using a time step of δt that is much small than Δt , i.e., $\delta t \ll \Delta t$. Note that Δt is a multiple of δt , i.e., $\Delta t = N \delta t$, where N is the number of time steps within each interval. As shown in Fig. 2.1, the FOM will be used in the first time interval ($0 \rightarrow \Delta t$). At the end of this FOM interval, the POD modes $\mathbf{U}_r$ will be computed simply using the singular value decomposition

Figure 2.1 Schematic of the hybrid snapshot simulation method involving FOM/ROM alternation.

(SVD), and a criterion ($\mathbf{C}_{F2R}$) will be evaluated to decide if the simulation should be switched from the FOM to ROM in the next interval, i.e., $\Delta t \rightarrow 2\Delta t$. If $\mathbf{C}_{F2R}$ is satisfied (true), the FOM will be projected onto the POD subspace $\mathbf{U}_r$ just computed to construct the local ROM that will be used in the following time interval ($\Delta t \rightarrow 2\Delta t$). Otherwise the FOM will remain in use. Correspondingly, there are two possible scenarios for the second interval:

(1) If the FOM remains in use during the second interval, i.e., $\Delta t \rightarrow 2\Delta t$, the simulation will generate more FOM data carrying new basis information not present in the previous POD modes $\mathbf{U}_r$. Therefore, at the end of this interval, viz., $t = 2\Delta t$, the prior projection matrix $\mathbf{U}_r$ needs to be updated by incorporating the new FOM solution data. In order to reduce the computational cost, iPOD will be utilized for updating $\mathbf{U}_r$ as described in Section 2.1.3.

(2) On the other hand, if $\mathbf{C}_{F2R}$ is satisfied, a local ROM will be produced by projecting the FOM onto the POD modes $\mathbf{U}_r$. Note that the term "local ROM"

arises from the fact that the ROM is constructed using the POD modes $\mathbf{U}_r$ extracted from the FOM data preceding the current interval. ROM will be used to simulate the second interval. At the end of this interval, i.e., at $t = 2\Delta t$, iPOD will not be performed because the ROM trajectory always lies within the existing POD subspace, and the ROM will not introduce any new data/information to basis vectors. In other words, the ROM will be performed in the temporal time interval, where information contributing to new basis vectors (relative to the existing modes $\mathbf{U}_r$) is negligible. Therefore, in our method the purpose of using the local ROM is to rapidly bridge FOM simulations for reduced computational cost. Meanwhile the criterion $\mathbf{C}_{R2F}$ will be examined to decide if the simulation should switch to FOM or remain using ROM in the upcoming interval, i.e., $2\Delta t \rightarrow 3\Delta t$. If the criterion is met, the simulation will switch back to FOM; otherwise ROM will continue being used.

The alternating FOM/ROM process above will continue till the end of simulation when all the intervals are simulated. Several points should be noted: (1) in contrast to its traditional counterpart, the volume of the snapshot data in our hybrid method is much lower as the ROM will not contribute to data generation; (2) at the end of the hybrid simulation, the ROM is already available for the online simulation. There is no need to perform another POD or FOM projection; and (3) the hybrid simulation is faster while maintaining excellent snapshot data accuracy and POD subspace representation as shown below. In the following sections, we will discuss, respectively, the FOM/ROM construction, iPOD updating, and criteria for FOM/ROM switch in details.

2.1.2 FULL ORDER MODEL(FOM) AND REDUCED ORDER MODEL(ROM) CONSTRUCTION

As describe above, the ROM will be used to rapidly traverse temporal intervals of less importance and bridge the simulation between FOM intervals. The physics-

based governing equation following spatial discretization [93, 74, 41], such as the finite volume, finite difference, and finite element method can be written in the form of an ordinary differential equation (ODE) set

$$\frac{d\mathbf{Y}}{dt} = \mathbf{H}(\mathbf{Y}), \quad (2.1)$$

where $\mathbf{Y} \in \mathbb{R}^n$ is the vector of the solution variable in the discretized computational domain, and n is the number of variables in the vector; t is the time; and $\mathbf{H} \in \mathbb{R}^n$ is the function arising from the spatial discretization. For the large FOM with thousands or even millions of grids, the computation could be very demanding. Therefore, model order reduction techniques have been developed [100, 98] to project the original full-order system onto a low-dimensional subspace and build a reduced order system for faster computation and more efficient memory usage. One of the most widely used projection subspace $\mathbf{U}_r$ is comprised of the basis vectors computed by POD or iPOD. That is, we approximate the true solution by $\mathbf{Y} \approx \mathbf{U}_r \mathbf{Y}_r$ and then substitute it into Eq.(2.1). Because of the orthonormality of $\mathbf{U}_r$, the reduced order system reads

$$\frac{d\mathbf{Y}_r}{dt} = \mathbf{U}_r^T \mathbf{H}(\mathbf{U}_r \mathbf{Y}_r), \quad (2.2)$$

where $\mathbf{U}_r \in \mathbb{R}^{n \times (r_d + r_e)}$ is the POD projection matrix and r_d is the rank of reduced order and r_e is the rank of extended order of $\mathbf{U}_r$ (see below); $\mathbf{Y}_r \in \mathbb{R}^{r_d + r_e}$ is the vector of the solution variable in the ROM.

2.1.3 INCREMENTAL PROPER ORTHOGONAL DECOMPOSITION (iPOD)

In the present effort, the projection matrix $\mathbf{U}_r$ used to construct the ROM above is obtained by the iPOD method, which is an extension of the traditional batch-POD and well-suited for processing streaming data generated during simulation. Specifically, our iPOD is initiated with the traditional POD at the end of the first interval,

i.e., $i = 1$. It decomposes the snapshots data matrix Ψ collected during the first interval into a POD modes/basis matrix $\mathbf{U}$ and a modal coefficient matrix $\mathbf{W}$, i.e.,

$$\Psi = [Y_1(\mathbf{x}), \dots, Y_j(\mathbf{x}), \dots, Y_N(\mathbf{x})] = \mathbf{U}\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \quad (2.3)$$

where the vector $\mathbf{Y}_j \in \mathbb{R}^n$ is the solution data computed by the FOM at the j^{th} time step in the first interval, i.e., the j^{th} snapshot among N snapshots in one interval; $\mathbf{x}$ is the spatial coordinates of the computational grid, e.g., the cell center in the finite volume discretization. In Eq. 2.3 the POD is obtained through the SVD, that is, the POD modal coefficients $\mathbf{W} = \mathbf{\Sigma}\mathbf{V}^T$. Note that both $\mathbf{U} \in \mathbb{R}^{n \times N}$ and $\mathbf{V} \in \mathbb{R}^{N \times N}$ are orthonormal matrices, and $\mathbf{\Sigma} = \text{diag}\{\sigma_i\} \in \mathbb{R}^{N \times N}$ is the diagonal matrix with the singular values sorted in the descending order on its diagonal, i.e., $\sigma_1 > \sigma_2 > \dots > \sigma_i > \dots > \sigma_N$.

The batch-POD above will only be used in the first interval to extract the initial POD basis vectors $\mathbf{U}$. In the subsequent FOM intervals, snapshot solutions containing the new subspace information will be computed and used to update the existing $\mathbf{U}$. iSVD proposed by [17] is employed to modify the basis vectors when new snapshot solution data is generated, which only requires the low-rank SVD and basis vector rotation, and is proven computationally efficient. Briefly let $\Psi_b \in \mathbb{R}^{n \times N}$ be the new snapshot data generated in the current FOM interval, which is appended to SVD factorization of the FOM data in the previous time interval, yielding

$$\begin{bmatrix} \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T & \Psi_b \end{bmatrix} = \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma} & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T, \quad (2.4)$$

where $\mathbf{J}$ is the orthonormal bases vectors spanning the orthogonal component $\mathbf{H}$ of Ψ_b with respect to $\mathbf{U}$: $\mathbf{H} = (\mathbf{I} - \mathbf{U}\mathbf{U}^T) \Psi_b$, $\mathbf{L} = \mathbf{U}^T \Psi_b$ and $\mathbf{K} = \mathbf{J}^T \mathbf{H}$, where $\mathbf{K}$ is the projection of $\mathbf{H}$ on $\mathbf{J}$, $\mathbf{K}$ and $\mathbf{J}$ can be obtained through the efficient QR

decomposition. The middle matrix in Eq. 2.4 is defined as $\mathbf{Q} = \begin{bmatrix} \mathbf{\Sigma} & \mathbf{L} \\ \mathbf{O} & \mathbf{K} \end{bmatrix}$, and applying SVD to matrix $\mathbf{Q}$ yields:

$$\begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \mathbf{Q} \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^T \approx \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}} \tilde{\mathbf{\Sigma}} \tilde{\mathbf{V}} \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^T, \quad (2.5)$$

where $\tilde{\mathbf{U}} \in \mathbb{R}^{2N \times N}$, $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{N \times N}$, and $\tilde{\mathbf{V}} \in \mathbb{R}^{2N \times N}$ are the truncated SVD matrices of $\mathbf{Q} \in \mathbb{R}^{2N \times 2N}$ at rank N , where again N is the number of time steps within each interval. Thus the updated SVD matrices arising from the new snapshots data are given by: $\mathbf{U} \leftarrow \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}$, $\mathbf{\Sigma} \leftarrow \tilde{\mathbf{\Sigma}}$, and $\mathbf{V} \leftarrow \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tilde{\mathbf{V}}$.

It should be noticed that $\mathbf{Q}$ has the rank of $2N$ at most for each update, and SVD on $\mathbf{Q}$ can be performed efficiently. The truncated SVD matrices $\tilde{\mathbf{U}}$, $\tilde{\mathbf{\Sigma}}$, and $\tilde{\mathbf{V}}$ ensure the rank of POD basis vectors $\mathbf{U}$ in iSVD will remain at N and does not increase when new FOM data is added during the simulation. the POD updating utilizes repetitive rotation of basis vectors $\begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix}$, which may cause the loss of their orthogonality, and accumulate tangible numerical errors. Therefore, an efficient QR decomposition can be included within the updating process in order to reorthogonalize the basis when the orthogonality in the newly updated basis is lost. The detailed algorithm of iPOD is summarized in Algorithm. 1

Algorithm 1: The algorithm for iPOD.

```

Input: Current SVD matrices  $\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}, N$ , force_orth (flag), and new snapshot data  $\mathbf{\Psi}_b$ 

 $\mathbf{L} = \mathbf{U}^T \mathbf{\Psi}_b$ ;                                %  $\mathbf{L}$  is the projection of new snapshot data  $\mathbf{\Psi}_b$  onto  $\mathbf{U}$ 
 $\mathbf{H} = \mathbf{\Psi}_b - \mathbf{U}\mathbf{L}$ ;                            %  $\mathbf{H}$  is the orthogonal component of  $\mathbf{\Psi}_b$ 
 $\mathbf{J} = \text{orth}(\mathbf{H})$ ;                            %  $\mathbf{J}$  is the orthogonal basis of  $\mathbf{H}$ 

% Below is to extend  $\mathbf{J}$  if  $\mathbf{H}$  is not full rank.  $\mathbf{P}$  is the zero matrix whose row number is the same as
% that of  $\mathbf{J}$  and column number is the difference in the column number of  $\mathbf{H}$  and  $\mathbf{J}$ 
 $\mathbf{J} = [\mathbf{J} \ \mathbf{P}]$ ;

 $\mathbf{K} = \mathbf{J}^T \mathbf{H}$ ;
 $\mathbf{Q} = \begin{bmatrix} \mathbf{\Sigma} & \mathbf{L} \\ \mathbf{\Omega}' & \mathbf{K} \end{bmatrix}$ ;                %  $\mathbf{\Omega}$  is a zero matrix with the same size of  $\mathbf{L}$ 

 $[\mathbf{U}', \mathbf{\Sigma}', \mathbf{V}'] = \text{svd}(\mathbf{Q})$ ;                % single value decomposition of new matrix  $\mathbf{Q}$ 
 $\mathbf{U}' = \mathbf{U}'(:, 1:N)$ ;
 $\mathbf{\Sigma}' = \mathbf{\Sigma}'(1:N, 1:N)$ ;
 $\mathbf{V}' = \mathbf{V}'(:, 1:N)$ ;

 $\mathbf{U} = [\mathbf{U} \ \mathbf{J}]\mathbf{U}'$ ;                            % update the POD matrices
 $\mathbf{\Sigma} = \mathbf{\Sigma}'$ ;
 $\mathbf{V} = [\mathbf{V} \ \mathbf{0}; \mathbf{0} \ \mathbf{I}]\mathbf{V}'$ ;

if (force_orth == true)                            % Preserving orthogonality
     $[\mathbf{U}_1, \mathbf{U}_2] = \text{qr}(\mathbf{U}, 0)$ ;
     $[\mathbf{V}_1, \mathbf{V}_2] = \text{qr}(\mathbf{V}, 0)$ ;
     $[\mathbf{U}_n, \mathbf{\Sigma}_n, \mathbf{V}_n] \leftarrow \text{svd}(\mathbf{U}_2 \mathbf{\Sigma}' \mathbf{V}_2^T)$ ;
     $\mathbf{U} \leftarrow \mathbf{U}_1 \mathbf{U}_n$ ;
     $\mathbf{V} \leftarrow \mathbf{V}_1 \mathbf{V}_n$ ;
     $\mathbf{\Sigma} \leftarrow \mathbf{\Sigma}_n$ ;
end

Output:  $\mathbf{U}, \mathbf{\Sigma}$ , and  $\mathbf{V}$ 

```

Once the new POD modes become available, they will be used for local ROM construction and computation. POD matrices can be truncated to retain the first $(r_d + r_e)$ leading modes

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \approx \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^T, \quad (2.6)$$

where $\mathbf{U}_r \in \mathbb{R}^{n \times (r_d + r_e)}$, $\mathbf{\Sigma}_r \in \mathbb{R}^{(r_d + r_e) \times (r_d + r_e)}$ and $\mathbf{V}_r \in \mathbb{R}^{N \times (r_d + r_e)}$ are the truncated SVD matrices and will be used for local ROM construction and simulation. r_d and r_e are, respectively, termed the basic and the extended modes. The reason to separately consider the basic and the extended modes for ROM construction and hybrid simu-

lation is elucidated in the next section. Since POD modes will be updated whenever the FOM data containing new information is generated, they will contain the most significant orthogonal basis vectors spanning critical FOM intervals across the entire snapshot simulation at the end. Therefore, the corresponding ROM could be immediately utilized for the online simulation with different operating conditions. Note that in the present effort, r_d and r_e are manually selected from a conservative perspective. In general, they can be determined by evaluating the singular value σ_i in Σ that indicates the dominance of the i^{th} POD mode $\mathbf{u}_j \in \mathbf{U}$ in the snapshot from an energy perspective.

2.1.4 THE CRITERIA FOR FOM/ROM SWITCHES

The switch between FOM and ROM will be determined by evaluating several criteria at the end of each time interval, which are elucidated in detail below:

(1) Criterion $\mathbf{C}_{F2R}$ to switch from FOM to ROM: it is based on the distribution of singular values σ_i since each σ_i determines the contribution of the corresponding POD basis to the variance of the snapshot data. Quantitatively, the relative significance of the j^{th} POD mode can be described by the energy ratio [44]:

$$E_j = \frac{\sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \in [0, 1]. \quad (2.7)$$

$\mathbf{C}_{F2R}$ includes two sub-criteria: $\mathbf{C}_{F2R,1}$ and $\mathbf{C}_{F2R,2}$. Since the number of the retained POD modes is normally much less than the total POD modes, i.e., $(r_d + r_e) \ll N$, the first sub-criterion $\mathbf{C}_{F2R,1}$ is defined as the cumulative energy ratio of the POD modes $r_d + 1 \rightarrow N$ relative to the total $1 \rightarrow N$. This criterion is computed as the relative root mean square (RRMS) [89] and needs to be sufficiently small with respect to a prescribed tolerance ϵ_1 to ensure the first r_d basic POD modes retained provides excellent representation of the solution data subspace:

$$\mathbf{C}_{F2R,1} = \left(\frac{\sum_{j=r_d+1}^N \sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \right)^{\frac{1}{2}} \leq \epsilon_1. \quad (2.8)$$

Recall that r_d is the number of basic modes retained. Although $\mathbf{C}_{F2R,1}$ quantifies the cumulative energy in the first r_d POD modes, it is not able to monitor the shift in energy among them, i.e., the variation of the data representation with respect to the modes number. To address this limitation, another criterion $\mathbf{C}_{F2R,2}$ below is introduced to quantitatively evaluate the cumulative energy contained within the POD modes from $r_d + r_e + 1 \rightarrow N$, which are the removed POD modes in Eq. 2.8 [75]:

$$\mathbf{C}_{F2R,2} = \left(\frac{\sum_{j=r_d+r_e+1}^N \sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \right)^{\frac{1}{2}} \leq \epsilon_2, \quad (2.9)$$

where r_e again is the number of extended POD modes retained. The use of the extended modes allows to reveal the changes in data energy from the mode $r_d + 1$ to $r_d + r_e$. At the end of each FOM simulation interval, both criteria $\mathbf{C}_{F2R,1}$ and $\mathbf{C}_{F2R,2}$ will be compared against their corresponding tolerances ϵ_1 and ϵ_2 , respectively. The value of ϵ_2 is less than ϵ_1 since the truncated POD modes ($r_d + r_e + 1 \rightarrow N$) in Eq. 2.9 are only a subset of the POD modes ($r_d + 1 \rightarrow N$) in Eq. 2.8. In the present effort ϵ_2 is selected to be one magnitude lower than ϵ_1 . If both criteria are satisfied, the current POD modes are able to accurately capture the FOM characteristics and its local solution subspace. Then the local ROM could be constructed and utilized in the following time interval. It should be noted that FOM is always used in the first interval since no prior snapshot data is available for POD.

(2) Criterion $\mathbf{C}_{R2F}$ to switch from ROM back to FOM: Another monitoring criterion is proposed, and will be evaluated at the end of the local ROM interval to determine whether the local ROM should remain in use for the next interval or the simulation should switch back from ROM to FOM.

$$\mathbf{C}_{R2F} = \left(\frac{\sum_{j=r_d+1}^{r_d+r_e} \mathbf{T}_{r,j}^2(i\Delta t)}{\sum_{j=1}^N \mathbf{T}_{r,j}^2(i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon, \quad (2.10)$$

where $\mathbf{T}_r(i\Delta t)$ represents the ROM solution at the end of the i^{th} time interval. Eq. 2.10 represents the ratio of the two L2 norms and it utilizes the modal coefficient of the r_e extended modes as a probe to examine if it is still appropriate to use the local ROM solver in the next time interval. If the modal coefficients (equivalent to the intensity) of the r_e extended modes grow significantly and exceeds the tolerance ϵ , it indicates that the dominance of the leading modes decays and the higher-order modes take over, and the local ROM starts to loss the ability to capture the key dynamics of the full system. Thus the snapshot simulation should be switched from ROM back to FOM.

It is noticed that due to their different dimensions, several measures will be taken to streamline the simulation when the models are switched. When the simulation changes from FOM to ROM, the full solution $\mathbf{Y}$ at the end of the FOM interval needs to be projected onto the updated POD modes $\mathbf{U}_r$ to initialize ROM computation in the next time interval. On the other hand, at the moment the solver switches from ROM to FOM, the reduced order solution $\mathbf{Y}_r$ at the end of the ROM time interval should be translated back to the full domain through $\mathbf{Y} \approx \mathbf{U}_r \mathbf{Y}_r$. In this manner, both solvers can be stitched seamlessly during the whole snapshot simulation. The algorithm for the hybrid snapshot simulation involving alternating FOM and ROM is summarized in Algorithm. 2

Algorithm 2: The hybrid algorithm of the snapshots computation in the training process.

```

Input:  $r_d, r_e, 1, \varepsilon_2$ , and  $\varepsilon_3, N, N_i, \Delta t$  and the initial temperature  $\mathbf{T}_0$ .

% The first interval in the time span
 $\Psi = \text{RunFOM}(\mathbf{T}_0);$  % Compute snapshot data  $\Psi$  using FOM
 $\Psi = \mathbf{U}\Sigma\mathbf{V}^T;$  % Compute POD modes  $\mathbf{U}$  using SVD
% Below is to evaluate criterion  $C_{F2R}$  to decide which model to use in the next interval: FOM or ROM
if ( $C_{F2R,1} \leq \varepsilon_1$  and  $C_{F2R,2} \leq \varepsilon_2$ )
    SolverFlag = ROM with an order of ( $r_d+r_e$ ) % ROM will be used in the next interval
     $\mathbf{U}_r = \mathbf{U}(:, 1:r_d+r_e);$  % Truncate the POD modes at ( $r_d+r_e$ ) for ROM solution
     $\mathbf{T}_{r0} = \mathbf{U}_r^T \Psi(:, N);$  % Extract the solution at the last time step and project it onto  $\mathbf{U}_r$ 
else
    SolverFlag = FOM % FOM will be used in the next interval
     $\mathbf{T}_0 = \Psi(:, N)$  % Set initial temperature for FOM solution in the next interval
end

for  $i = 2: N_i$  % Compute the time intervals from 2 to  $N_i$ 
    if SolverFlag == FOM
         $\Psi_b = \text{RunFOM}(\mathbf{T}_0);$  % Compute snapshot data  $\Psi_b$  using FOM
         $[\mathbf{U}, \Sigma, \mathbf{V}] = \text{iPOD}[\mathbf{U}, \Sigma, \mathbf{V}, \Psi_b]$  % iPOD updating of  $\mathbf{U}, \Sigma$  and  $\mathbf{V}$ 
        % Evaluate criterion  $C_{F2R}$  to determine which model to use in the next interval: FOM or ROM
        if ( $C_{F2R,1} \leq \varepsilon_1$  and  $C_{F2R,2} \leq \varepsilon_2$ )
            SolverFlag = ROM with an order of ( $r_d+r_e$ ) % ROM will be used in the next interval
             $\mathbf{U}_r = \mathbf{U}(:, 1:r_d+r_e);$ 
             $\mathbf{T}_{r0} = \mathbf{U}_r^T \Psi_b(:, N);$  % Extract the solution at the last time step and project it onto  $\mathbf{U}_r$ 
        else
            SolverFlag = FOM;
             $\mathbf{T}_0 = \Psi_b(:, N)$  % Set initial temperature for FOM solution in the next interval
        end
    end
    if SolverFlag == ROM
         $\mathbf{T}_r = \text{RunROM}(\mathbf{T}_{r0})$  % Compute ROM solution  $\mathbf{T}_r$  with the POD mode  $\mathbf{U}_r$ 
        % Evaluate criterion  $C_{R2F}$  to determine which model to use in the next interval: FOM or ROM
        if ( $C_{R2F} \geq \varepsilon_3$ )
            SolverFlag = FOM; % FOM will be used in the next interval
             $\mathbf{T}_0 = \mathbf{U}_r^T \mathbf{T}_r(:, N);$  % Set initial temperature for FOM solution in the next interval
        else
            SolverFlag = ROM; % ROM will continue being used in the next interval
             $\mathbf{T}_{r0} = \mathbf{T}_r(:, N);$  % Set initial condition for ROM solution in the next interval
        end
    end
end
end

Output: POD modes  $\mathbf{U}_r$  and the corresponding ROM

```

2.2 NUMERICAL MODELS OF HEAT TRANSFER

In this section, a numerical model of heat transfer based on the finite volume method (FVM) will be developed and used to demonstrate and verify the hybrid FOM/ROM

Figure 2.2 Finite volume method (FVM) based representation of FOM in a 2D computational domain

snapshot simulation method, although the method is generally applicable to other physics and engineering problems. The heat transfer model includes a computational domain comprised of different materials, and their thermal conductivities are temperature-dependent. Only the thermal conduction is considered within the interior of the computational domain, and the prescribed heat flux, and convective and radiative heat transfer are used as the boundary conditions. The governing equation of the heat transfer can be written as:

$$\rho C_p \frac{dT}{dt} = \nabla \cdot (k(T) \nabla T) + Q, \quad (2.11)$$

where T is the scalar temperature; ρ is the density; k and C_p are, respectively, the temperature-dependent thermal conductivity and specific heat of the constitute materials; and Q is the heat source term. The spatial differential of the thermal governing equation Eq. 2.11 is discretized by FVM following the schematic in Fig. 2.2, yielding [93]:

$$\begin{aligned}
& \rho C_p V_p \frac{dT_P}{dt} \\
& = -(\alpha_e + \alpha_s + \alpha_w + \alpha_n) T_P + \alpha_e T_E + \alpha_s T_S + \alpha_w T_W + \alpha_n T_N \\
& \quad + Q_P V_P + A q_b + R_s (T_P^4 - T_{\infty,R}^4) \\
& = -\alpha_p T_P + \alpha_e T_E + \alpha_s T_S + \alpha_w T_W + \alpha_n T_N + Q_P V_P + A q_b + R_s (T_P^4 - T_{\infty,R}^4)
\end{aligned} \tag{2.12}$$

where V_p is the volume of the cell P ; α_j is the conductive transport link at the cell face j (see section 3.2); $a_P = a_e + a_s + a_w + a_n$; q_b is the heat flux applied at boundaries; A is the area of the face of the boundary cell where the heat flux is applied, and $R_s = A\sigma_r$ is the radiative links between the present cell and the farfield temperature, and σ_r denotes the Stefan-Boltzmann Constant. Note that for the interior cells, the last two terms in Eq. 2.12 for boundary conditions are not available. The state space form is obtained by semi-discretization of the spatial terms, which is given the finite volume discretization yields [93]:

$$\tilde{\mathbf{M}} \frac{d\mathbf{T}}{dt} = \tilde{\mathbf{A}}_n \mathbf{T} + \tilde{\mathbf{A}}_s (\mathbf{T}_{\infty,h} - \mathbf{T}) + \tilde{\mathbf{R}}_s (\mathbf{T}_{\infty,R}^{*4} - \mathbf{T}^{*4}) + \tilde{\mathbf{B}}_c + \tilde{\mathbf{B}}_u, \tag{2.13}$$

where the superscript asterisk denotes the element-wise operation; $\tilde{\mathbf{M}} \in \mathbb{R}^{n \times n}$ is a diagonal matrix with the thermal capacitances of all the computational cells on its diagonal; $\tilde{\mathbf{A}}_n \in \mathbb{R}^{n \times n}$ is the internal heat transfer coefficient matrix contributed by conduction through both internal faces and the isothermal boundaries, and its element at (i, j) location represents the thermal conductance between the i^{th} and the j^{th} cell; $\tilde{\mathbf{A}}_s \in \mathbb{R}^{n \times n}$ represents the external heat transfer coefficient matrix contributed by external convection; $\tilde{\mathbf{R}}_s \in \mathbb{R}^{n \times n}$ is the external radiative heat transfer coefficient matrix contributed by radiation at external faces. Both $\tilde{\mathbf{A}}_s$ and $\tilde{\mathbf{R}}_s$ are diagonal, and only the entries corresponding to the cells adjacent to the convection and radiation boundaries are given non-zero values. $\mathbf{T}_{\infty,h} \in \mathbb{R}^n$ and $\mathbf{T}_{\infty,R} \in \mathbb{R}^n$ are the values of the temperature at the far field for the convection and radiation heat transfer at

boundaries, respectively. Note that the radiative term in Eq. 2.13 has the fourth power of the temperature, representing a strong nonlinear term in the equation. $\tilde{\mathbf{B}}_c \in \mathbb{R}^n$ is the heat source term that is independent of time. $\tilde{\mathbf{B}} \in \mathbb{R}^{n \times d}$ is the scatter matrix representing the contribution from the time dependent input vector $\mathbf{u} \in \mathbb{R}^d$ to each cell with the input dimension d . The input vector $\mathbf{u}$ denotes the time-varying heat flux prescribed at the external boundaries. Eq. 2.13 is the state space equation of the entire computational domain, and can be written in a more concise form:

$$\frac{d\mathbf{T}}{dt} = \mathbf{A}\mathbf{T} + \mathbf{R}\mathbf{T}^{*4} + \mathbf{D} + \mathbf{B}\mathbf{u}, \quad (2.14)$$

where $\mathbf{A} = \tilde{\mathbf{M}}^{-1} (\tilde{\mathbf{A}}_n - \tilde{\mathbf{A}}_s)$, $\mathbf{R} = \tilde{\mathbf{M}}^{-1} \tilde{\mathbf{R}}_s$, $\mathbf{D} = \tilde{\mathbf{M}}^{-1} (\tilde{\mathbf{A}}_s \mathbf{T}_{\infty,h} + \tilde{\mathbf{R}}_s \mathbf{T}_{\infty,R}^{*4} + \tilde{\mathbf{B}}_c)$ and $\mathbf{B} = \tilde{\mathbf{M}}^{-1} \tilde{\mathbf{B}}$. It is noted that since $\tilde{\mathbf{M}}$ is a diagonal matrix, its inverse can be readily computed by taking the reciprocal of the diagonal elements. The RHS of Eq. 2.14 represents the specific realization $\mathbf{F}(\mathbf{T})$ of the nonlinear heat transfer FOM in Eq. 2.1. The corresponding ROM could be obtained by the truncated approximation: $\mathbf{T} \approx \mathbf{U}_r \mathbf{T}_r$:

$$\frac{d\mathbf{T}_r}{dt} = \mathbf{A}_r \mathbf{T}_r + \mathbf{R}_r (\mathbf{U}_r \mathbf{T}_r)^{*4} + \mathbf{D}_r + \mathbf{B}_r \mathbf{u}, \quad (2.15)$$

where $\mathbf{A}_r = \mathbf{U}_r^T \mathbf{A} \mathbf{U}_r$, $\mathbf{R}_r = \mathbf{U}_r^T \mathbf{R}$, $\mathbf{D}_r = \mathbf{U}_r^T \mathbf{D}$, and $\mathbf{B}_r = \mathbf{U}_r^T \mathbf{B}$. It is clear that since the thermal conductance matrix $\mathbf{A}$ is temperature dependent, its reduced order counterpart $\mathbf{A}_r$ cannot be pre-computed before the simulation. Instead it has to be updated for each iteration during the entire simulation. Specifically, $\mathbf{T}_r$ is first used to reconstruct its solution in the full domain through $\mathbf{T} \approx \mathbf{U}_r \mathbf{T}_r$. Then the full matrix $\mathbf{A}$ can be assembled, and its reduced order counterpart is then constructed as $\mathbf{A}_r = \mathbf{U}_r^T \mathbf{A} \mathbf{U}_r$ for the next solution.

The FOM in Eq. 2.14 and the ROM in Eq. 2.15 can be solved by discretizing the temporal term using various differencing schemes. The Crank-Nicolson scheme,

a second-order, numerically semi-implicit method is adopted, leading to a set of algebraic equations for solution [50]. Specifically, two computing loops, including the time loop and the iteration loop are developed to drive the numerical computation. The time loop allows the computation to march in time, and the iteration loop aims to make the solution converge within one time step to accommodate the temperature dependence of the material properties and the nonlinear terms. Since the Crank-Nicolson scheme has been reported extensively in the literature for CFD simulation, it will not be repeated in this chapter.

2.3 RESULTS AND DISCUSSIONS

Two case studies are performed to verify and demonstrate the hybrid snapshot simulation method, including nonlinear heat transfer within an annular cone and a fin array. The analysis involves interior conduction with temperature dependent thermal conductivity, and convection and radiation at the boundaries. Each case study includes the snapshot simulation and the verification simulation used to, respectively, construct and examine the ROM. In the next section, the computational models of both case studies will first be described, and the computational performance, such as the accuracy and computational time/speedup, will be compared between the hybrid FOM/ROM method and the traditional FOM method. In order to quantitatively characterize the discrepancy between the FOM and the ROM results, two metrics indicative of numerical errors are defined: (a) the absolute error $\mathbf{Err}_{abs}$ and (b) the mean square relative difference (MSRD) $\mathbf{Err}_{msrd}$ [100].

$$\mathbf{Err}_{abs} = | \mathbf{U}_r \mathbf{T}_r - \mathbf{T} |, \quad (2.16)$$

$$\mathbf{Err}_{msrd} = \sqrt{\frac{\sum_{j=1}^n \left(\frac{\mathbf{U}_{r,j} \mathbf{T}_{r,j} - \mathbf{T}_j}{\mathbf{T}_j} \right)^2}{n}} \times 100\%, \quad (2.17)$$

Figure 2.3 Computational model for the multi-layer annular cone: (a) Geometry and computational grids; and (b) The heat flux profile prescribed at the three quarters of the outer wall of the computational model and used in the snapshot simulation and verification simulation.

where again $\mathbf{T}_r$ and $\mathbf{T}$ are the values of the temperature computed by the ROM and the FOM, respectively. It should be noted that for each time step $\mathbf{Err}_{abs}$ is a vector representing the temperature difference at all the computational cells, while $\mathbf{Err}_{msrd}$ is a scalar obtained by applying mean square relative difference to all the cells and represents the average relative error in the entire computational domain. These two metrics will be evaluated in all the case studies.

2.3.1 CASE STUDY 1: CONDUCTION AND RADIATION HEAT TRANSFER OF AN ANNULAR CONE

MODEL DESCRIPTION

The first case study is the thermal model of an annular cone, which is comprised of an outer layer made of stainless steel and an inner layer of mineral wool with equal layer thickness. Fig. 2.3(a) shows the geometry, size, and grid of the model, which

has 9,408 computational cells. The time step size of the computation is $\delta t = 0.25s$. An adiabatic boundary condition (zero heat flux) is specified at the top, the bottom, the inner wall, and a quarter of the outer lateral wall. A boundary condition with time-varying heat flux is set at the rest (three quarters) of the outer lateral wall. All the outer lateral walls can dissipate heat to the environment through radiation. Such configurations will induce thermal gradient and transport in all directions throughout the cone by interior conduction, and also involve thermal radiation at the boundary. The nonlinearity arises from the radiative heat transfer at the boundary and the temperature-dependent thermal conductivity of both constituent materials as listed in Table .2.1, and the emissivity at the outer walls is 1. The model configuration parameters are: $\epsilon_1 = 1e^{-5}$, $\epsilon_2 = 1e^{-6}$, $\epsilon = 1e^{-4}$, $r_d = 20$ and $r_e = 4$.

The case study includes two simulations: (1) the snapshot simulation to generate representative solution data and construct the POD modes and ROMs; and (2) the verification simulation to examine the ROM generated at the end of snapshot simulation using the operating parameters different from that in the snapshot simulation. Fig. 2.3(b) illustrates the heat flux profiles specified at the three quarters of the outer wall for the snapshot simulation and for the verification simulation, which are, respectively, annotated by the curves in red and in blue. It can be seen that both input profiles are quite different, and the peak value of the verification simulation almost doubles that of the snapshot simulation, which hence can be used to assess the robustness of the constructed ROM. In the snapshot simulation, results obtained from two methods are compared: **SS-FOM**—the snapshot simulation data is obtained exclusively by computing the FOM and the POD modes are constructed by the SVD of the data, followed by ROM construction; and **SS-Hybrid**—the proposed hybrid FOM/ROM simulation is performed to generate the snapshot data adaptively, in which POD modes and local ROM are built and updated by iPOD at the end of each FOM interval. The **SS-FOM** will be used as the benchmark for comparison.

Correspondingly, two ROMs, **FROM** and **HROM**, generated from the **SS-FOM** and **SS-Hybrid**, respectively, will be inspected in the verification simulation. The numerical solutions obtained from **FROM** and **HROM** will be compared against the verification simulation data produced by the FOM only.

Table 2.1 **Material and thermal properties of the constituent materials**

Materials	Thermal Conductivity (W/m K)	Density (kg/m ³)	Specific Heat (J/kg K)
Stainless Steel	$11.13 + 1.27 \times 10^{-2} \times T$	8030	510
Mineral Wool	$0.03 + 0.2438 \times \left(\frac{T}{1000}\right)^3$	165	840

COMPARISON BETWEEN SS-FOM AND SS-HYBRID

In this section, the snapshot simulation data generated by the SS-FOM and the SS-Hybrid are compared in terms of accuracy and computational time. Fig. 2.4 illustrates the temperatures at four selected computational cells and the FOM/ROM alternation during the hybrid snapshot simulation for various time intervals. The total time span is 240s and divided into 10 time intervals of the equal duration $\Delta t = 24s$. In the figure the intervals simulated by the ROM are presented by solid lines in color, and those simulated by FOM are denoted by black dash curves. It could be seen that there are 3 FOM intervals and 7 ROM intervals used in the snapshot simulation. The first interval utilizes the FOM to initiate the simulation and obtain the initial POD modes. The criterion C_{F2R} is met at the end of each FOM interval, which switches the simulation to the ROM. ROM simulation rapidly traverses the majority of the time span and successively in multiple intervals until the criterion C_{R2F} is satisfied, and then the simulation returns to FOM.

The spatiotemporal distributions of the snapshots solution produced by SS-FOM and SS-Hybrid are, respectively, portrayed in Fig. 2.5(a) and Fig. 2.5(b), which include the data for all the 9408 cells in the computational model and the physical duration of 240s. The results of both simulations are almost the same within the

Figure 2.4 The FOM/ROM alternation in the hybrid snapshot simulation in the annular cone model.

entire domain, confirming the accuracy of the hybrid FOM/ROM method. It also clearly shows that the temperatures in the cells numbered in the range of $1 - 4,000$ corresponding to the outer stainless steel layer rise rapidly and reach the maximum value around $900K$. When the heat flux starts to drop down at $t = 120s$, the temperatures at the heat flux boundaries are the highest, and also respond and decrease quickly. The temperature decrease is caused by the net heat dissipation at these boundaries due to radiation. The cells numbered $5,000 - 9,000$ are in the inner layer (mineral wool), and respond slowly but with a steady growth of the temperature.

The error of the SS-Hybrid relative to the SS-FOM, including the absolute error and the MSRD error are shown in Fig. 2.6. Both of them reach the peak values at $t = 144s$, corresponding to $15K$ and 0.45% , respectively. Then the snapshot simulation returns to the FOM in the 7^{th} interval (i.e., time span $144s - 168s$). Therefore, the error drops quickly with a clear tendency for continued decrease. Thus the ROM is selected for snapshot simulation for the 8^{th} and the 9^{th} interval (time span $168s - 216s$). However, in the last interval ($216s - 240s$) the FOM is selected again since the errors tend to grow in the 9^{th} interval.

Figure 2.5 The spatiotemporal distribution of the snapshots solution produced by (a) SS-FOM and (b) SS-Hybrid in the annular cone model.

Figure 2.6 Comparison between SS-FOM and SS-Hybrid for the annular cone model in terms of (a) the absolute error in the entire domain; and (b) the MSRD error in the temporal domain.

The detailed information of computational time for the hybrid snapshot simulation, including FOM interval computation, iPOD updating, and ROM interval simulation are listed in Table. 2.2. The computational time for the three FOM intervals are, respectively, 21.188s, 16s, and 10.582s, and the corresponding iPOD updating after FOM simulation are 0.171s, 0.157s, and 0.085s. The computational time for ROM intervals falls in the range of 2.2s to 2.6s. Thus, the total time of the hybrid snapshot simulation is $t_{SS-Hybrid} = t_{FOM} + t_{iPOD} + t_{ROM} = 47.77s + 0.413s + 16.774s = 64.96s$. For comparison, the traditional snapshot simulation method that only uses the FOM to produce the data is also conducted with the total time of 168.84s, in which FOM computation for all 10 intervals and the batch POD operation to create projection subspace cost 168.09s and 0.748s, respectively. Therefore, the hybrid method could accelerate the snapshot simulation greatly and the speedup ratio is $\gamma = \frac{t_{FOM}}{t_{Hybrid}} = \frac{168.84s}{64.96s} \approx 2.6$.

Table 2.2 The breakdown of the computational time for the hybrid snapshot simulation using a time step size $\delta t = 0.25s$ for the annual cone model(Unit:s)

Time Interval	FOM	iPOD	ROM
<i>1st</i>	21.188	0.171	...
<i>2nd</i>	...	...	2.64
<i>3th</i>	...	...	2.305
<i>4th</i>	...	...	2.324
<i>5th</i>	...	...	2.396
<i>6th</i>	...	...	2.521
<i>7th</i>	16	0.157	...
<i>8th</i>	...	...	2.296
<i>9th</i>	...	...	2.292
<i>10th</i>	10.582	0.085	...
Total	47.77	0.413	16.774

COMPARISON BETWEEN FROM AND HROM IN VERIFICATION SIMULATION

In this section we compare the two sets of ROMs, i.e., the FROM and HROM, obtained, respectively, through the SS-FOM and the SS-Hybrid in the verification simulation. Recall that the verification simulation uses a different profile of the input

Figure 2.7 The comparison between (a)FROM and (b)HROM in terms of the temperature solutions during the verification simulation for the annular cone model.

heat flux at the external lateral walls. Fig. 2.7 shows the temperature solution of FROM and HROM at each computational cell across the temporal domain. It could be seen that both solutions are nearly the same, which indicates the POD modes and the ROM constructed using the hybrid snapshot data are equivalent to those by the traditional method. Next the MSRD error of the FROM and HROM relative to the FOM are calculated and illustrated in Fig. 2.8, where FOM results refer to the verification simulation only by FOM. It can be seen that both MSRD errors grow slowly at the beginning and increase faster when the input heat flux starts to decay. The MSRD error of the FROM is lower than that of HROM in the entire time period, and their peak values reach up to 0.41% and 0.83%, respectively, at the end of the verification simulation. Fig. 2.9 shows the 3D visualization of the temperature solution obtained from FOM and HROM at different instants of time. Again the agreement between them is excellent. The thermal fronts propagate in all the directions (circumferential, radial, and longitudinal) along with notable temperature gradients. The computational time in both FROM and HROM are nearly the same, respectively, 27.19s and 26.48s, which indicates that the POD modes and the ROM constructed through the hybrid snapshot simulation maintain the same

Figure 2.8 Relative error of the FROM and HROM relative to the FOM for the annular cone model.

computational efficiency.

2.3.2 CASE STUDY 2: CONDUCTION AND CONVECTION HEAT TRANSFER OF A FIN ARRAY

The second case study considers the heat transfer of a fin array made of stainless steel. Fig. 2.10(a) shows the geometry, size, and grid of the computational model, which consists of 38,400 computational cells. The time step size of the computation is $\delta t = 0.5s$. The adiabatic boundary condition is specified at the lateral faces of the substrate. The boundary condition with a prescribed time-varying heat flux is set at the bottom of the substrate, and the profiles used in the snapshot and the verification simulation are shown in Fig. 2.10(b). Different from Case study 1, in this case the profile used in the snapshot simulation has a higher peak value and a longer time span than the verification simulation. The convection boundary condition is assigned to the rest boundaries with the effective heat transfer coefficient of $h = 20W/(M^2 \cdot K)$ and the environmental temperature of $300K$. Again such a heat transfer scenario

Figure 2.9 Contour plot of the temperature solution obtained from the FOM analysis (top row) and the HROM (bottom row) at various instants of time (a) 60 s, (b) 120 s, and (c) 240 s.

could involve noticeable thermal gradients and propagation in all the directions, and the nonlinearity arises from the temperature-dependent thermal conductivity of the stainless steel as listed in Table 2.1 above. The tolerance parameters for model switch in the snapshot simulation are $\epsilon_1 = 1e^{-5}$, $\epsilon_2 = 1e^{-6}$, $\epsilon = 1e^{-4}$, $r_d = 20$ and $r_e = 4$, which are the same as Case study 1.

COMPARISON BETWEEN SS-FOM AND SS-HYBRID

In this section, the SS-FOM and SS-Hybrid are compared in terms of accuracy and computational time. The utilization of the FOM and the ROM and their alternation during the hybrid snapshot simulation is portrayed in Fig. 2.11 using the time-dependent temperature solution selected at four computational cells. Likewise, the solution obtained by simulating FOM is presented by black dash curves and that by the ROM are denoted by solid lines in color. The total time span of the snapshot simulation $t = 1200s$ is partitioned into 12 time intervals equally with $\Delta t = 100s$

Figure 2.10 Computational model for the fin array: (a) Geometry and computational grids; and (b) The heat flux profile applied at the bottom of the fin array in the snapshot simulation and verification simulation.

for each. It shows that there are 2 FOM intervals and 10 ROM intervals in the simulation. The FOM is used in the first interval as always to construct the initial POD modes. The criterion $\mathbf{C}_{F2R}$ is then evaluated using the singular values of the initial POD and found to satisfy the switch requirement, leading to the use of the ROM in the second interval. The ROM simulation continues till the end of the 7th interval ($t = 700s$) when the criterion $\mathbf{C}_{R2F}$ is met, which switches the simulation back to FOM in the 8th time interval. Similarly, one FOM simulation significantly improves the subspace representation of the leading POD modes, and the simulation goes back to ROM for the following four intervals (the 9th to 12th).

Fig. 2.12 illustrates the comparison between SS-FOM and SS-Hybrid in terms of the snapshot temperature solution in each computational cell (38,400 in total) across the entire time span ($t = 1200s$). The excellent match between them is clearly observed, which confirms that the hybrid method can serve as an exceptional substitute for the traditional snapshot simulation method

Figure 2.11 The FOM/ROM alternation in the hybrid snapshot simulation for the fins array model.

Figure 2.12 The spatiotemporal distribution of the snapshots solution produced by (a) SS-FOM and (b) SS-Hybrid in the annular cone model.

The absolute error and the MSRD error of the SS-Hybrid solutions relative to SS-FOM are illustrated in Fig. 2.13. The absolute error of SS-Hybrid at all computational cells increases slowly during the initial period ($t = 0 \sim 600s$), jumps abruptly at $t = 600s$, and reaches the peak value of $0.3K$ at $t = 700s$ for most of the computational cells. This is because at the beginning of the 6th interval of the snapshot simulation, the heat flux immediately drops to zero (see Fig. 2.10), and the temperature responds with a rapid decay (as stainless steel is an excellent thermal conductor).

Figure 2.13 Comparison between SS-FOM and SS-Hybrid for the fins array model in terms of (a) the absolute error in the entire spatiotemporal domain; and (b) the MSRD error in the temporal domain.

Consequently, the solution subspace of the temperature decay is poorly represented by the POD modes obtained in the first FOM interval that is characterized by a rapid rise in temperature due to the instantaneously applied heat flux. Fig. 2.13(b) depicts the MSRD error of SS-Hybrid relative to SS-FOM in the time domain. The same trend as the absolute error is clearly observed, and the peak value reaches 0.03% at $t = 700s$. Both quantitative results indicate that the hybrid snapshot simulation method is able to generate the data of equivalent accuracy/fidelity as the traditional FOM snapshot simulation.

The breakdown of the computational time for the hybrid snapshot simulation in the fin array model, including FOM computation, ROM computation, and iPOD updating is summarized in Table. 2.3. The two FOM intervals cost 100.484s and 65.934s, respectively, and the corresponding iPOD only takes 1.145s and 1.128s and is almost negligible. The computational time for the ROM intervals ranges from 23.6s to 29.7s. The total computational time of the hybrid method is then $t_{SS-Hybrid} = t_{FOM} + t_{iPOD} + t_{ROM} = 166.42s + 2.273s + 251.723s = 420.42s$.

Meanwhile, the traditional FOM snapshot simulation is also performed to benchmark the performance of the hybrid method. FOM simulation for all 12 intervals

and the one-shot POD cost 863.19s and 15.46s, respectively, leading to the total time of 878.65s. Essentially, the speedup ratio of SS-Hybrid relative to SS-FOM is $\gamma \approx 2.09$. Note that the theoretical limit of the speedup ratio would be the total number of intervals divided by the number of FOM intervals, which in this case study is $\gamma = 12/2 = 6$ if the computational time of the ROM is negligible. The low value of the actual speedup ratio can be attributed to the fact that although ROM can be solved at the faster speed, the ROM assembly process is time consuming because it involves reconstruction of the full domain solution, FOM assembly, and the FOM projection onto the low-dimensional subspace. In the future, we will investigate several techniques, such as the discrete empirical interpolation algorithm (DEIM) [27, 29] and the missing point estimation (MPE) [104] to further accelerate the ROM assembly process for enhanced speedup ratio.

Table 2.3 The breakdown of the computational time for the hybrid snapshot simulation using a time step size of $\delta t = 0.5s$ for the fin array model(Unit:s)

Time Interval	FOM	iPOD	ROM
<i>1st</i>	100.484	1.145	...
<i>2nd</i>	...	...	24.274
<i>3th</i>	...	...	24.841
<i>4th</i>	...	...	25.289
<i>5th</i>	...	...	25.338
<i>6th</i>	...	...	24.456
<i>7th</i>	...	...	29.719
<i>8th</i>	65.934	1.128	...
<i>9th</i>	...	...	23.612
<i>10th</i>	...	...	24.289
<i>11th</i>	...	...	25.018
<i>12th</i>	...	...	24.887
Total	166.418	2.273	251.723

COMPARISON BETWEEN FROM AND HROM IN VERIFICATION SIMULATION

We also compare the FROM and HROM in the verification simulation, which are obtained from SS-FOM and SS-Hybrid snapshot simulation, respectively. The temperature solutions in all the computational cells within the entire time domain ($t = 960s$)

Figure 2.14 The comparison between FROM and HROM in the temperature solutions during the verification simulation for the fins array model.

are shown in Fig. 2.14. Similar to the first case study, both ROM results are nearly the same, which again substantiates that the proposed hybrid snapshot simulation method can produce high-fidelity data and representative POD modes for ROM development.

We then examine the accuracy of both ROMs by calculating their MSRD errors relative to the FOM solution in the verification simulation. The time-dependent MSRD errors for FROM and HROM are depicted, respectively, as the blue solid curve and the red dashed curve in Fig. 2.15. Both MSRD errors increase gradually at the beginning, and then rapidly rise up to the peak values (0.034% and 0.038%, respectively, for FROM and HROM) around 200s as a result of the sudden jump in the applied heat flux. Subsequently both errors drop to the lower level because of the drop in the heat flux and temperature solution, which continues till end the simulation. In addition, the MSRD error of FROM is slightly lower than that of HROM during the majority of the verification simulation, indicating that the FROM outperforms the HROM to a certain extent. However, the hybrid simulation method is still able to generate high-quality, representative snapshot data and POD modes and achieve adequate accuracy for ROM applications. The temperature solutions

Figure 2.15 Comparison between FROM and HROM in terms of the MSRD error in the temporal domain for the fins array model.

obtained from the FOM and the HROM at different instants of time are visualized in the 3D computational model, and the excellent match between them is observed as shown in Fig. 2.16. The thermal profile at the bottom around the substrate is the highest and exhibits significant non-uniformity due to notable geometric variations, and the temperature decreases in the longitudinal direction of the fins along with the more uniform distribution. Similar to the above the computational time of both FROM and HROM for the verification simulation are almost the same, which are $236.92s$ and $238.12s$, respectively.

EFFECTS OF THE TOLERANCE PARAMETERS FOR MODEL SWITCH CRITERIA

It is explicit that the course of the hybrid snapshot simulation will be dictated by the tolerance parameters ϵ_1 , ϵ_2 and ϵ in criteria $\mathbf{C}_{F2R}$ and $\mathbf{C}_{R2F}$. In this section, the effects of these parameters on the model switch during the hybrid simulation will be thoroughly investigated. Fig. 2.17 illustrates the FOM/ROM alternation in the hybrid snapshot simulation with four different combinations of the tolerance

Figure 2.16 Contour plot of the temperature in the simulation domain obtained from the FOM analysis (top row) and the HROM (bottom row) for the fins array model at various instants of time (a) 240 s, (b) 306 s, and (c) 540 s.

Figure 2.17 The FOM/ROM alternation in the hybrid snapshot simulations for the fins array model with different tolerance parameters (a), (b), (c), and (d).

parameters. In Fig. 2.17(a), the tolerance parameters are increased by 10 times from the original values ($\epsilon_1 = 10^{-5}$ and $\epsilon_2 = 10^{-6}$ in Fig. 2.11) to $\epsilon_1 = 10^{-4}$ and $\epsilon_2 = 10^{-5}$ while keeping the value of ϵ the same, i.e. $\epsilon = 10^{-4}$. In general, the increase in both ϵ_1 and ϵ_2 make it easier to meet the criterion $\mathbf{C}_{F2R}$ and for the solver to switch from FOM to ROM. However, the tolerance parameters ($\epsilon_1 = 10^{-5}$ and $\epsilon_2 = 10^{-6}$) adopted in Fig. 2.11 are already large enough and $\mathbf{C}_{F2R}$ is quite relaxed, and therefore, the FOM computation only spans one interval although it is invoked twice (the 1st and the 8th interval). Under this circumstance, further increase them to $\epsilon_1 = 10^{-4}$ and $\epsilon_2 = 10^{-5}$ will not further reduce the number of intervals simulated by each FOM call (as one interval is already the minimum), or change the FOM/ROM alternation. This observation explains why the temperature solution in Fig. 2.17(a) is exactly the same as that in Fig. 2.11. In Fig. 2.17(b), the tolerance parameter ϵ for ROM switch to FOM is increased by 10 times, i.e., $\epsilon = 10^{-3}$ while keeping the values of ϵ_1 and ϵ_2 same as the original ones. The increase in ϵ tends to make it difficult to meet the criterion $\mathbf{C}_{R2F}$ (by switching from ROM to FOM), leading to more preferred use of ROM at the cost of the larger numerical error, which is manifested evidently through Fig. 2.17(b) and Fig. 2.18. It clearly shows that except for the first FOM interval, the rest intervals are simulated by the ROM, and the error of SS-Hybrid (dashed cyan curve in Fig. 2.18) grows mildly at the beginning and then dramatically following the sudden drop in applied heat flux at 600s, and eventually reaches 0.057% at the end of simulation. In Fig. 2.17(c) $\epsilon_1 = 10^{-6}$ and $\epsilon_2 = 10^{-7}$ are used, which are 10 times less than the original value and make it harder to switch from FOM to ROM. Three successive intervals between 700s and 1000s are simulated by FOM, which is in distinct contrast to that in Fig. 2.11 and Fig. 2.17(a) where only one FOM interval is simulated between 700s and 800s. The simulation of two additional FOM interval in Fig. 2.17(c) improves the snapshot simulation accuracy slightly, as indicated by the lower MSRD error (in blue) at the second half of the simulation than that in Fig. 2.11

and the green curve in Fig. 2.18. In Fig. 2.17(d) $\epsilon = 10^{-5}$ is 10 times less than its original value and the other two tolerance parameters, i.e., $\epsilon_1 = 10^{-5}$ and $\epsilon_2 = 10^{-6}$ remain the same. This change imposes less stringent requirement on $\mathbf{C}_{R2F}$ criterion and renders the ROM to FOM switch easier and the ROM use in successive intervals more difficult. As we can see, there are three FOM intervals throughout the entire simulation as opposed two in Fig. 2.11. The second and the third FOM intervals are located at $600s - 700s$ and $800s - 900s$ during the second half of the simulation in order to update the POD modes and correct the deviation of the ROM trajectory caused by the rapid dynamics arising from the sudden drop in applied heat flux. It is also astonishing to find in Fig. 2.18 that the decrease in ϵ seems more effective in improving the ROM accuracy in contrast to that in Fig. 2.17(c) that also includes four FOM intervals by constraining ϵ_1 and ϵ_2 . Quantitatively, the MSRD error of the snapshot simulation in Fig. 2.17(d) always stays below 0.01% (red curve in Fig. 2.18) and notably lower than all the others. Therefore, adopting a lower tolerance parameter for criterion $\mathbf{C}_{R2F}$ to improve the sensitivity of detecting deviation of ROM trajectory and monitoring the dominance of leading modes for more judicious utilization of FOM simulation in the hybrid method seems more effective for boosting snapshot accuracy.

2.4 SUMMARY OF CHAPTER.2

In this chapter, an adaptive hybrid simulation methodology is proposed for rapid generation of high-quality snapshot data that can be used to construct POD basis vectors and ROMs. In contrast to the traditional method, several new elements are introduced: first, the entire snapshot simulation is partitioned into multiple intervals, and each is simulated by either the FOM or the local ROM. Second, the snapshot simulation alternates between FOM and local ROM, and the decision for model switch is made on-the-fly by evaluating several criteria that monitor the dominance of the

Figure 2.18 Comparison of the MSRD errors of SS-Hybrid with different tolerance parameters for the fins array model.

leading POD basis vectors and the trajectories of the reduced system. The aim of adaptively embedding local ROM into the snapshot simulation is to accelerate the data generation process without appreciably compromising the fidelity and representation of the computed POD modes. Third, iPOD is used to continuously update the local POD modes by the low-rank SVD and subspace rotation that is computationally efficient and well-suited for our hybrid simulation. The hybrid simulation method is then applied to case studies of nonlinear numerical heat transfer. Key technical findings include:

1. The accuracy and representation of the local ROM is improved progressively through iPOD during the snapshot simulation, yielding a global ROM at the end that is more broadly applicable to online simulation uses (with various operating parameters).
2. The hybrid snapshot simulation demonstrates excellent accuracy (with the relative error $< 0.45\%$) and acceleration with the speedup ratio $2.09X - 2.6X$ relative to the traditional method in the present case studies. The ROM developed by the

hybrid method offers the same simulation speed and level of accuracy as that by the traditional method. Therefore, the feasibility of using the computationally efficient hybrid snapshot simulation for constructing robust and accurate ROMs is established.

3. The selection of the tolerance parameters influences the model accuracy and the computational time, but not in a very sensitive manner. In our case studies, scaling the tolerance parameters by 10 times only increase the number of FOM intervals by two. Applying a lower tolerance parameter to $\mathbf{C}_{R2F}$ criterion seems more effective to improve the snapshot simulation accuracy without imposing additional computational load as its decision on FOM utilization is made by directly observing the dominance of the leading POD modes in the ROM and the trajectory of its higher-order modes.

The speedup ratio of $2.09X - 2.6X$ achieved by the hybrid snapshot simulation in this chapter is still appreciably lower than the theoretical limit, viz., the ratio of the FOM intervals to the total intervals in the simulation if the computational time of ROM is negligible. This is caused by the significant amount of time spent on the ROM assembly that involves full solution reconstruction, FOM assembly, and FOM projection in each solver iteration. The following chapters will include the development of selective ROM assembly using the DEIM [27, 29] and GNAT [23, 25] techniques to further accelerate the computation. Another limitation of this chapter is that the number of POD modes retained in the ROM is preset and in general is more conservative. We will also develop the approaches to enable automated selection of the mode numbers, e.g., by comparing the difference in the number of the basic and extended POD modes.

CHAPTER 3

DEIM REDUCED ORDER MODEL BASED ON HYBRID SNAPSHOT SIMULATION

The goal of this chapter is to investigate the feasibility of establishing the DEIM approach in the hybrid snapshot simulation framework [10] in the previous chapter to significantly reduce the computational cost associated with ROM-DEIM construction. Its novel aspects include: (1) the method to combine the hybrid snapshot simulation (alternating between FOM and ROM), snapshot data generation, and DEIM-based ROM construction; (2) the strategy to address the challenge associated with a limited number of FOM intervals in the hybrid simulation and poor representation of the subspace underlying nonlinear terms and DEIM. In DEIM the POD modes of the nonlinear terms rather than those of solutions are used to determine the interpolation points. The FOM/ROM switch is based on the singular values and coefficients of the POD modes of FOM solution, and hence, the data are inadequate or unrepresentative for extracting accurate POD modes of nonlinear terms. Therefore, a new procedure is proposed to reconstruct and enrich snapshot data of nonlinear terms from the ROM solution in the hybrid snapshot simulation to improve the accuracy of their POD modes and DEIM; and (3) the formulation to construct ROM-DEIMs in the cell-centered finite volume method (FVM) scheme along with properly devised data structures to consider both the componentwise and non-componentwise nonlinear terms that are widely encountered in physical transport phenomena.

This chapter is organized as following: the methodology of the hybrid snapshot simulation (including criteria for FOM/ROM switch, SVD and iSVD for POD mode generations, and reconstruction of snapshot data for nonlinear terms) is described in Section 3.1. The DEIM algorithm in the cell-centered FVM implementation is introduced in Section 3.2. A numerical heat transfer model including both the componentwise and non-componentwise nonlinearities is used as an example to elucidate the proposed method (FOM, ROM, and ROM-DEIM) in Section 3.3. In Section 3.4, the numerical results obtained from two case studies are discussed, and the computational performance of the present method is also evaluated. This chapter will be summarized in Section 3.5.

3.1 METHODOLOGY OF HYBRID SNAPSHOT SIMULATION FOR ROM-DEIM CONSTRUCTION

In this section, we will describe the methodology to combine the hybrid computational approach in snapshot simulation and the ROM-DEIM construction using the snapshot data in snapshot simulation. The developed ROM-DEIM could be utilized in the online simulation for the acquisition of model solutions as illustrated in Fig. 3.1. The hybrid snapshot simulation is similar to the approach described in Chapter.2: the entire time span of the snapshot simulation (labeled “1” in Fig. 3.1) is divided into multiple intervals. Each interval will be simulated by either FOM or ROM, and the latter will be constructed and updated during the snapshot simulation, i.e., local ROM. The FOM can be written in the form of an ordinary differential equation (ODE) set:

$$\frac{d\mathbf{Y}(t)}{dt} = \mathbf{H}(\mathbf{Y}(t)), \quad (3.1)$$

where $\mathbf{Y}(t) \in \mathfrak{R}^n$ is the vector of the solution variable in the discretized computational domain and n is the dimension of the FOM solutions. $\mathbf{H} \in \mathfrak{R}^n$ represents the function

arising from the spatial discretization of the governing equation. The corresponding form of ROM then can be written as:

$$\frac{d\mathbf{Y}_r(t)}{dt} = \mathbf{U}_r^T \mathbf{H}(\mathbf{U}_r \mathbf{Y}_r(t)), \quad (3.2)$$

where $\mathbf{U}_r$ is the truncated POD basis vector of a reduced dimension r ($r \ll n$). The solution of ROM is denoted as $\mathbf{Y}_r(t) \in \mathfrak{R}^r$. In each FOM interval, full solution will be computed and used as snapshot data to create and update POD modes/basis vectors $\mathbf{U}_r$ of the solution variable $\mathbf{Y}(t) \in \mathfrak{R}^n$ using the incremental singular value decomposition (iSVD). In addition, the snapshot data of the nonlinear terms will also be assembled and stored. These steps are annotated in blue in Fig. 3.1. In each ROM interval, the local ROM will be first built using $\mathbf{U}_r$ extracted in the preceding FOM interval and then computed to obtain the solution $\mathbf{Y}_r(t) \in \mathfrak{R}^r$ in the subspace, where r is the dimension of the ROM solution. It is followed by reconstructing solution in the full domain $\hat{\mathbf{Y}}(t)$, and assembling the reconstructed snapshot data of the nonlinear terms (see the annotation in red in Fig. 3.1). The reconstructed snapshot data for the nonlinear terms will be combined and accumulated with that from the FOM interval. The rationale underlying the hybrid snapshot simulation approach is to reduce the computational complexity and cost by using the local ROMs to quickly traverse the intervals that offer negligible new information for POD mode $\mathbf{U}_r$ construction.

The DEIM procedure is performed at the end of the snapshot simulation, which is labeled “2” in Fig. 3.1. SVD will first be used to extract POD modes out of the accumulated snapshot data of the nonlinear terms, and DEIM will be applied onto these POD modes to determine critical interpolation points/cells. During the online simulation (labeled “3” in Fig. 3.1), the POD modes of the solution variables, the POD modes of the nonlinear terms, and the DEIM interpolation points/cells will be utilized to assemble ROM-DEIM. The ROM-DEIM will be exclusively computed throughout the entire verification simulation. Because only a small number of the

nonlinear terms at the grid points/cells will be assembled, while the rest will be interpolated during each iteration, the simulation can be run at a very fast speed.

The key elements of the methodology will be discussed below. Specifically, the hybrid FOM/ROM simulation along with FOM/ROM switch criteria are similar to the issues in Chapter.2 and will be briefly introduced in Section 3.1.1, SVD and iSVD in Section 3.1.2, the reconstruction of the snapshot data of the nonlinear terms in Section 3.1.3. The DEIM algorithm to select the DEIM points/cells for interpolation and the procedure to assemble the nonlinear terms for ROM construction in the cell-centered FVM formulation are elucidated in detail in Section 3.2. A heat transfer model used to demonstrate the process of obtaining ROM-DEIM following the procedure above will be described in Section 3.3.

3.1.1 HYBRID(FOM/ROM) SNAPSHOT SIMULATION

Following Chapter. 2, the hybrid snapshot simulation method will be briefly described in this section, and its details are reported in our previous work [10]. As discussed above, the entire snapshot simulation is divided into N_i equal intervals Δt with the size of the time step $\delta t (\Delta t = N\delta t)$. The first interval is FOM in order to initialize the POD modes $\mathbf{U}_r$ using the snapshot data collected, while the simulation in the subsequent intervals will be performed using either FOM or ROM. The switch criteria are evaluated at the end of each interval [10]:

(1) $\mathbf{C}_{F2R}$ to switch from FOM to ROM: It is evaluated at the end of the FOM interval, and consists of two sub-criteria $\mathbf{C}_{F2R,1}$ and $\mathbf{C}_{F2R,2}$ given by:

$$\mathbf{C}_{F2R,1} = \left(\frac{\sum_{j=r_d+1}^N \sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \right)^{\frac{1}{2}} \leq \epsilon_1, \quad \mathbf{C}_{F2R,2} = \left(\frac{\sum_{j=r_d+r_e+1}^N \sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \right)^{\frac{1}{2}} \leq \epsilon_2. \quad (3.3)$$

The switch is based on the importance of the POD basis vectors in terms of their contribution to snapshot data, which is determined by the singular values

Figure 3.1 The proposed methodology to construct ROM-DEIM based on hybrid snapshot simulation: (1) Hybrid snapshot simulation; (2) determination of DEIM points/cells at the end of the hybrid simulation; and (3) verification simulation utilizing ROM-DEIM.

$\sigma_1, \dots, \sigma_j, \dots, \sigma_N$ with j^{th} singular value corresponding to the j^{th} POD basis vector [85]. The detailed information about the criteria is introduced in Chapter.2 . If both sub-criteria are satisfied, the FOM solution subspace can be captured very well by existing POD basis vectors. Then the local ROM will be built and used in the next time interval.

(2) C_{R2F} to switch from ROM to FOM: It is used to determine whether the simulation should keep using ROM or switch back to FOM in the next interval. This criterion will be evaluated at the end of the local ROM interval, and is given by:

$$\mathbf{C}_{R2F} = \left(\frac{\sum_{j=r_d+1}^{r_d+r_e} \mathbf{Y}_{r,j}^2(i\Delta t)}{\sum_{j=1}^{r_d+r_e} \mathbf{Y}_{r,j}^2(i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon, \quad (3.4)$$

where $\mathbf{Y}_{r,j}^2$ represents the square of the j^{th} element of vector $\mathbf{Y}_r$, and the criterion represents the ratio of the two L2 norms. If $\mathbf{C}_{R2F}$ exceeds the tolerance ϵ , it indicates that the trajectory of the ROM starts to deviate from existing POD subspace, and the simulation needs to switch back to FOM. Otherwise, POD modes are still suitable for ROM simulation in the next time interval [75]. Such FOM/ROM switch will be inspected at the end of each interval and continue until the end of the snapshot simulation.

3.1.2 SVD AND INCREMENTAL SVD

POD modes $\mathbf{U}_r$ of the solution variable $\mathbf{Y}(t)$ are generated from SVD of the snapshot data from the initial FOM interval, then are updated using the new snapshot data collected in each FOM interval by iSVD(not in the ROM interval). At the end of the entire snapshot simulation, the ultimate POD modes $\mathbf{U}_r$ are obtained and could be utilized for online simulation in Fig. 3.1. From Chapter.2, it is noted that the snapshot data $\mathbf{\Psi} \in \mathbb{R}^{n \times N}$ is collected in the initial time interval, and applying SVD:

$$\mathbf{\Psi} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \quad (3.5)$$

where $\mathbf{U} \in \mathbb{R}^{n \times N}$, $\mathbf{\Sigma} \in \mathbb{R}^{N \times N}$ and $\mathbf{V} \in \mathbb{R}^{N \times N}$. In the subsequent FOM interval that produces new snapshot data, iSVD [17, 10] is performed to update the basis vectors in $\mathbf{U}$ through a computationally efficient procedure of low-rank SVD and basis vectors rotation. Similar to Chapter.2, the existing POD matrices are update by iSVD at the end of each FOM interval (except for the initial one):

$$\begin{bmatrix} \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T & \mathbf{\Psi}_b \end{bmatrix} = \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma} & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T \approx \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^T \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T, \quad (3.6)$$

then the new POD modes are updated by: $\mathbf{U} \leftarrow \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}, \mathbf{\Sigma} \leftarrow \tilde{\mathbf{\Sigma}}, \mathbf{V} \leftarrow \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \tilde{\mathbf{V}}.$

During the ROM interval, the newest POD modes $\mathbf{U}$ obtained above are truncated to yield ROM projection matrix $\mathbf{U}_r \in \mathbb{R}^{n \times r}$ with a lower rank of $r = r_d + r_e \ll N$, which in Matlab syntax is $\mathbf{U}_r = \mathbf{U}(:, 1:r)$. Then $\mathbf{U}_r$ will be used for ROM construction and computation. Since the truncated dimension r is much smaller than the FOM dimension n , the computational time and resource usage for the ROM interval is notably less than that of the FOM interval. Detailed information can be found in Chapter.2 [10].

3.1.3 RECONSTRUCTION OF SNAPSHOT DATA FOR NONLINEAR TERMS

When nonlinear terms are present in the ROM, its computational efficiency can be appreciably compromised. As described above, various techniques have been proposed to address the issue, including the discrete empirical interpolation method-DEIM [27] [29]. In order to perform DEIM, the snapshot data for the nonlinear terms needs to be computed first, which traditionally is obtained by substituting FOM solution data into the nonlinear operators. In this section we will describe the process to obtain the snapshot data for the nonlinear terms using the hybrid FOM and ROM solution.

The numerical model under present consideration is a typical transport equation spatially discretized by cell-centered FVM, and includes two kinds of nonlinearities: (1) the inter-cell nonlinear term $\mathbf{F}(\mathbf{Y})$ that describes the transport flux among mesh cells, such as thermal conduction with temperature-dependent thermal conductivity; and (2) the cell-wise nonlinear term $\mathbf{G}(\mathbf{Y})$ that only involves contribution from a single cell, such as the nonlinear boundary conditions and heat sources. Thus the governing transport equation in the ordinary differentiation equation (ODE) form is given by:

$$\frac{d\mathbf{Y}(t)}{dt} = \mathbf{F}(\mathbf{Y}(t)) + \mathbf{G}(\mathbf{Y}(t)). \quad (3.7)$$

Recall that $\mathbf{Y} \in \mathbb{R}^n$ is the FOM solution vector in the whole computational domain with a dimension n , and t is the time. Note that Eq. 3.7 is obtained by discretizing the spatial differential terms of the governing partial differentiation equation (PDE). If there is any constant or linear term in Eq. 3.7, they can be incorporated into either $\mathbf{F}(\mathbf{Y})$ or $\mathbf{G}(\mathbf{Y})$ without losing the generality of analysis below. To reduce computational cost, the FOM solution can be projected onto a low-dimensional POD modal subspace $\mathbf{U}_r$ determined by the snapshot data $\mathbf{\Psi}$ above, and is approximated by $\mathbf{Y} \approx \mathbf{U}_r \mathbf{Y}_r$, which when substituted into Eq. 3.7 yields:

$$\frac{d\mathbf{Y}_r(t)}{dt} = \mathbf{U}_r^T \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t)) + \mathbf{U}_r^T \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t)), \quad (3.8)$$

where $\mathbf{Y}_r$ is the ROM solution vector with the dimension of $r = r_d + r_e \ll n$. Eq. 3.8 clearly reveals the root for low efficiency of ROM caused by the nonlinear terms. That is, the ROM solution $\mathbf{Y}_r$ needs to be first reconstructed in the full domain by $\hat{\mathbf{Y}} = \mathbf{U}_r \mathbf{Y}_r$, and then the nonlinear terms are assembled in the full domain using $\hat{\mathbf{Y}}$. To tackle the limitation, DEIM accelerates the assembly of the vectorized nonlinear terms by identifying their POD modal subspace and then interpolating to obtain all their entries using a smaller number of selected entries at important mesh cells. Accordingly, the snapshot data of the nonlinear terms $\mathbf{F}$ and $\mathbf{G}$ (rather than the solution variable $\mathbf{Y}$) needs to be collected to extract their POD modes and undertake DEIM.

Traditionally, the snapshot data for the nonlinear terms is computed using FOMs only [27] [21] [91] [47] because of the demanding requirement for data quality and basis vector representation. Specifically, the FOM solution $\mathbf{Y}(t)$ is obtained at the end of the snapshot simulation, and then substituted into the nonlinear operators $\mathbf{F}(\mathbf{Y}(t))$ and $\mathbf{G}(\mathbf{Y}(t))$ on the RHS of Eq. 3.7. However, in our hybrid method, only a few intervals are computed by FOM during the snapshot simulation, leading to a limited number of snapshot data and inadequate presentation of POD subspace of the nonlinear terms. The criteria to determine FOM/ROM switch above are based

on the singular values and ROM trajectory of the solution variable $\mathbf{Y}$. Therefore, the solution data generated in the FOM interval in general does not represent well the subspace of nonlinear terms. In addition, DEIM selects interpolation points in the spatial domain of computational meshes. For large models, a large number of DEIM interpolation points may be required, and the snapshot data collected in the FOM interval may be insufficient for DEIM. To address the issue, a new approach proposed in this chapter, which includes two steps: (1) at the end of each time step in FOM interval, e.g., the i^{th} interval, following the traditional method, the FOM solution is substituted into the nonlinear operators to yield the snapshot data of the nonlinear terms, i.e.,

$$\mathbf{N}_F^i = \mathbf{F}(\mathbf{Y}(\tau)) \in \mathbb{R}^{n \times N}, \quad \mathbf{N}_G^i = \mathbf{G}(\mathbf{Y}(\tau)) \in \mathbb{R}^{n \times N}, \quad (3.9)$$

where $\tau = (i-1)\Delta t + [\delta, 2\delta, \dots, j\delta, N\delta]$ is all the time instances in the i^{th} interval and signifies concatenating their solution data. Hereafter $\mathbf{N}$ is used to denote the snapshot data of the nonlinear terms to be distinguished from that of the FOM solutions $\mathbf{\Psi}$; and (2) at the end of each time step in ROM interval, the approximate solution in the full domain is first reconstructed by $\hat{\mathbf{Y}} = \mathbf{U}_r \mathbf{Y}_r$, which is then used to assemble the nonlinear terms and create their snapshot data,

$$\hat{\mathbf{N}}_F^i = \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(\tau)), \quad \hat{\mathbf{N}}_G^i = \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(\tau)), \quad (3.10)$$

where $\hat{\cdot}$ denotes the quantities from or reconstructed from the ROM solution. In other words, $\hat{\mathbf{N}}_F^i$ and $\hat{\mathbf{N}}_G^i$ are the snapshot data for the nonlinear terms obtained by applying the nonlinear operators onto the approximate solutions $\hat{\mathbf{Y}}$ in the full domain.

The procedure above is then repeated for each FOM or ROM interval to obtain the snapshot data of the nonlinear terms in the entire time span of the hybrid snapshot simulation, which is assembled into a single matrix $\mathbf{N}_F(\mathbf{Y}(t))$ and $\mathbf{N}_G(\mathbf{Y}(t))$. Note that $\mathbf{N}_F$ and $\mathbf{N}_G$ without superscript i denote the snapshot data of the entire snapshot

Figure 3.2 The procedure to assemble snapshot data for nonlinear terms using a mix of FOM and reconstructed ROM solutions from the hybrid snapshot simulation.

simulation for the two nonlinear parts $\mathbf{F}$ and $\mathbf{G}$, respectively. Fig. 3.2 illustrates the procedure to the assemble snapshot data for nonlinear terms, and recall that N_i is the total number of the time intervals used in the simulation. The parts in orange and in blue represent the snapshot data from the FOM solutions and the reconstructed ROM solutions, respectively.

3.2 DEIM AND CELL-CENTERED FVM IMPLEMENTATION

In this section the DEIM algorithm and its implementation in cell-centered FVM formulation will be described.

3.2.1 DEIM ALGORITHM

DEIM was developed to reduce the cost of assembling nonlinear terms in ROMs [27] [29], and includes three key steps: (1) taking POD of the entire snapshot data of the nonlinear terms, which is performed by SVD at the end of the hybrid snapshot simulation in order to preserve high accuracy; (2) selecting important points/cells (i.e., their indices) that correspond to the entry with the largest residual of reconstructing the next basis vector in POD modes [27] [29]; and (3) assembling the nonlinear terms only at the selected points/cells, which are then used to interpolate the rest nonlinear terms in the whole computational domain. Note that steps 1 and 2 are undertaken at the end of the hybrid snapshot simulation, and step 3 is during the online simulation for computational acceleration.

Specifically, SVD is first applied to mixed snapshot data of the nonlinear terms $\mathbf{N}_F$ and $\mathbf{N}_G$ above separately to generate POD modes $\mathbf{U}_F$ and $\mathbf{U}_G$ with the low dimensions:

$$\begin{aligned}\mathbf{N}_F &\approx \mathbf{U}_F \mathbf{\Sigma}_F \mathbf{V}_F^T, \\ \mathbf{N}_G &\approx \mathbf{U}_G \mathbf{\Sigma}_G \mathbf{V}_G^T.\end{aligned}\tag{3.11}$$

Note that subscript F and G are employed to be distinguished from the POD modes $\mathbf{U}_r$ of the FOM solution above. POD matrices for both are truncated at the dimension m_F and m_G and span the dominant low-dimensional subspace of the nonlinear terms, $\mathbf{F}$ and $\mathbf{G}$, respectively. Correspondingly, their truncated POD modes have the size of $\mathbf{U}_F \in \mathbb{R}^{n \times m_F}$ and $\mathbf{U}_G \in \mathbb{R}^{n \times m_G}$.

In the DEIM algorithm [27] a matrix $\mathbf{P} \in \mathbb{R}^{n \times m}$ is defined to select the interpolation indices (i.e., grid points/cells) in a n dimensional system. Denote m the number of the selected indices, the $\mathbf{P}$ matrix can be expressed by:

$$\mathbf{P} = [\mathbf{e}_{\varphi_1}, \mathbf{e}_{\varphi_2}, \dots, \mathbf{e}_{\varphi_m}], \tag{3.12}$$

where $\mathbf{e}_{\varphi_i} = [0, \dots, 0, 1, 0, \dots, 0]^T$ is the φ_i th column of the identity matrix $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ with n dimension for $i = 1, \dots, m$. In our model two different selection matrices $\mathbf{P}_F$ and $\mathbf{P}_G$ with the rank of m_F and m_G , respectively, are used for DEIM approximation of the inter-cell term $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ and the cell-wise nonlinear term $\mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t))$ in Eq. 3.8 [27] [29] for the online simulation:

$$\begin{aligned}\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t)) &\approx \mathbf{U}_F \left(\mathbf{P}_F^T \mathbf{U}_F \right)^{-1} \mathbf{P}_F^T \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t)), \\ \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t)) &\approx \mathbf{U}_G \left(\mathbf{P}_G^T \mathbf{U}_G \right)^{-1} \mathbf{P}_G^T \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t)).\end{aligned}\tag{3.13}$$

The interpolation indices $\varphi_1, \dots, \varphi_{m_F}$ in $\mathbf{P}_F$ and $\varphi_1, \dots, \varphi_{m_G}$ in $\mathbf{P}_G$ are selected by the DEIM algorithm in [27].

Eq. 3.13 implies two means to accelerate computation during the online simulation: first, $\mathbf{P}_F^T$ and $\mathbf{P}_G^T$ left multiplied to $\mathbf{F}$ and $\mathbf{G}$ essentially indicate that only m_F and m_G rows of $\mathbf{F}$ and $\mathbf{G}$ need to be assembled; and second, the following matrices remain the same during online simulation and can be pre-computed to further cut down the computational cost:

$$\mathbf{C}_F = \mathbf{U}_F \left(\mathbf{P}_F^T \mathbf{U}_F \right)^{-1}, \quad \mathbf{C}_G = \mathbf{U}_G \left(\mathbf{P}_G^T \mathbf{U}_G \right)^{-1}.\tag{3.14}$$

This is because during online simulation the quantities associated with DEIM, such as $\mathbf{P}_F^T$, $\mathbf{P}_G^T$, $\mathbf{U}_F$, and $\mathbf{U}_G$ will remain the same (and will be obtained at the end of the snapshot simulation). In the next section we will describe the procedure to assemble both the inter-cell and the cell-wise nonlinear terms in the cell-centered FVM formulation, both of which need to be updated during the online simulation that poses a challenge for efficient ROM construction and will be addressed by DEIM.

3.2.2 NONLINEAR TERMS UPDATING

As shown in Eq. 3.8, the RHS of the ROM consists of two kinds of nonlinear terms, the inter-cell $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ and the cell-wise $\mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t))$. The latter is caused by the nonlinear phenomena that do not involve transport between two cells, e.g., the

Figure 3.3 The procedure for identifying the (a) DEIM cells; (b) surrounding faces (red lines), and (c) adjacent cells (Ad) of the DEIM cells in a 2D computational domain.

nonlinear heat source within the cell or from boundaries. However, $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ is contributed by the transport flux between two connected cells in the computational domain (normalized by the mass or capacitance term associated with the temporal derivative, see Section 3.3, such as the heat flux through thermal conduction. The flux between a cell P and its adjacent cell J is written as:

$$\Psi = -\alpha_j (Y_J - Y_P), \quad (3.15)$$

where P is the index of the present cell, on which the thermal balance needs to be established; Y is the value of the solution variable at the center of cell J and P , and j is the index of the face between cell P and cell J . As shown in Fig. 2.2 for a FVM model in 2D, J can be E , S , W , and N depending on the position of the adjacent cell relative to the present cell; and j can take e , s , w , and n representing the face linking cell P and its adjacent cells. $\alpha_j = (kA/\delta)_j$ is the transport link at the cell face j , where A and δ are, respectively, the area of the face and the distance between cell P and its adjacent cell J , and k is the transport coefficient (e.g., thermal conductivity in heat transfer), and gives rise to the model nonlinearity if depending on Y_J and Y_P .

The computation of the inter-cell nonlinear term $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ is demanding and involves multiple steps, including (1) reconstructing the solution in the full domain

Cell Index	Face Index			
	Face 1	Face 2	Face 3	Face 4
l	...	...	...	...
...	...	...	...	...
l	l_1	l_2	l_3	l_4
...	...	...	...	...
m	m_1	m_2	m_3	m_4
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
n	n_1	n_2	n_3	n_4

(a)

Face Index	Adjacent Cell Index	
	Cell 1	Cell 2
l	...	...
...	...	...
l	l_1	l_2
...	...	...
m	m_1	m_2
...	...	...
nf	nf_1	nf_2

(b)

Figure 3.4 The data structure used for nonlinear terms assembly at the DEIM cells: (a) Cell-to-Face; and (b) Face-to-Cell.

$\mathbf{U}_r \mathbf{Y}_r(t)$; (2) updating variable-dependent transport coefficient k at the face between cells; and (3) assembling $\mathbf{F}$ terms by establishing the flux balance for each computational cell using Eq. 3.15. Step (1)-(3) need to be performed for each iteration in each time step, which slows down the ROM simulation. To reduce the computational loads, the DEIM method discussed above will be utilized. As indicated by Eq. 3.13, once the POD modes $\mathbf{U}_F$ and the interpolation indices $\mathbf{P}_F$ of the nonlinear term are available, only entries at DEIM interpolation points, i.e., $\mathbf{P}_F^T \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ rather than the entire vector $\mathbf{F}$ need to be assembled, and the rest entries can be interpolated. Although DEIM can determine those cells for interpolation (which hereafter is termed DEIM cell or interpolation cell as shown in Fig. 3.3(a)), $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ arises from the inter-cell transport flux, and therefore, the adjacent cells of the DEIM cells also need to be extracted in order to calculate the transport fluxes across their interfaces (red lines in Fig. 3.3b and c) and assemble the entries in $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ at the DEIM cells. The retrieval of the adjacent cells can be accomplished by constructing two data structures during numerical model initialization: the cell-to-face data structure and the face-to-cell data structure as shown in Fig. 3.4.

Fig. 3.4(a) illustrates the cell-to-face data structure for a 2D computational domain with n quadrilateral cells. It takes the index of the cell as the entry (the 1st

column) and retrieves the indices of the four faces surrounding the cell (the $2^{nd} \sim 5^{th}$ column). As shown for the l^{th} and the m^{th} cells, the face indices $l_1 \sim l_4$, and $m_1 \sim m_4$ are, respectively, extracted. Fig. 3.4(b) illustrates the corresponding face-to-cell data structure with nf faces in total, which takes the index of the face as the entry (the 1^{st} column) and extracts the indices of the two cells on both sides of the face since each face connects two cells. For example, the cell indices $l_1 \sim l_2$ and $m_1 \sim m_2$ are, respectively, extracted for the l^{th} and the m^{th} faces. With both data structures, we then can follow the procedure in Fig. 3.3 to determine all the adjacent cells, which starts with the DEIM algorithm presented in Section 3.1 to determine the DEIM cells (Fig. 3.3(a)). Then cell-to-face data structure will be queried using the indices of the DEIM cells as the entry to determine their surround surfaces, i.e., red lines in Fig. 3.3(b). Note that the face indices retrieved may be repeated because two DEIM cells could be neighboring to each other, such as the two DEIM cells at the top of Fig. 3.3(b). This can be resolved by eliminating the redundant indices, e.g., using the function 'unique' in Matlab or other equivalents. Then the indices of the surround faces are filed into the face-to-cell data structure to extract indices of all cells on both sides of those faces including the DEIM cell and their adjacent cells. Again the retrieved cell indices may be redundant for several reasons: two DEIM cells are next to each other (e.g., the two DEIM cells at the top of Fig. 3.3(c)) and one cell is adjacent to two DEIM cells (e.g., the cell at the center of Fig. 3.3(c) adjacent to the two DEIM cells above and below). Likewise, those repeated cell indices can be eliminated easily. In summary, at the end of the DEIM, three sets of indices will be kept for the online simulation, which are the DEIM cells $\mathbf{P}_F$, the surrounding face $\mathbf{I}_f$, and the combined DEIM and adjacent cells $\mathbf{I}_c$, and the size of $\mathbf{P}_F$ is smaller than $\mathbf{I}_c$. Again $\mathbf{I}_c$ is used to obtain the values of variable $\mathbf{Y}$ at all DEIM and adjacent cells for computing the transport fluxes across the surrounding faces with indices $\mathbf{I}_f$. The transport fluxes are then used to assemble the entries with indices $\mathbf{P}_F$ in $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$. Note that for

the cell-wise nonlinear term $\mathbf{G}(\mathbf{U}_r, \mathbf{Y}_r(t))$, only indices $\mathbf{P}_G$ are needed. More detailed information about approximation of the transport phenomenon could be found in [25](for finite volume) and in [27](for finite difference). When the second- or higher-order spatial discretization is employed, neighbors of neighbors of the DEIM cells need to be extracted to construct the governing equation of the DEIM cell. Thus, the process to identify neighboring cells will be performed twice or multiple times with the first one to search for the neighbors of the DEIM cells, and the subsequent ones to identify the neighbors of neighbors. The duplicated cell indices will be eliminated at the end of the search for enhanced DEIM approximation accuracy. In the next section, the entire process will be described using the heat transfer as an example problem.

3.3 NUMERICAL MODEL OF HEAT TRANSFER

The numerical model of heat transfer based on FVM will be presented in this section and used for verifying ROM-DEIM obtained by the hybrid snapshot simulation method. We will first describe FOM and ROM without DEIM used in the offline hybrid snapshot simulation, and then introduce ROM-DEIM that is constructed at the end of the snapshot simulation and used for the online simulation.

3.3.1 FOM AND ROM

The heat transfer model in this chapter consists of different materials and involves the thermal conduction at the interior of the computational domain, and the prescribed heat flux and radiation (and convection as needed) at the boundaries. The thermal conductivity is temperature dependent, and therefore, introducing nonlinearity to the inter-cell transport term $\mathbf{F}$. The cell-wise nonlinear term $\mathbf{G}$ arises from thermal contributions at boundaries, such as applied heat flux and radiation. Following FVM-based discretization of the spatial differential terms in the governing equation, FOM

in the form of ODE is [93]:

$$\mathbf{M} \frac{d\mathbf{T}}{dt} = \tilde{\mathbf{A}}_n(\mathbf{T}) \mathbf{T} + \tilde{\mathbf{R}}_s (\mathbf{T}_{\infty,R}^{*4} - \mathbf{T}^{*4}) + \tilde{\mathbf{B}}_c + \tilde{\mathbf{B}} \mathbf{u}, \quad (3.16)$$

where $\mathbf{T} \in \mathbb{R}^n$ is the vector of the temperature solution in the full computational domain with the dimension of n ; $*$ on the superscript denotes the elements-wise operation; $\mathbf{M} \in \mathbb{R}^{n \times n}$ is the diagonal matrix, and its diagonal is populated with the thermal capacitance/mass of all computational cells; $\tilde{\mathbf{A}}_n(\mathbf{T}) \in \mathbb{R}^{n \times n}$ stores the thermal conductance among cells and at the isothermal boundaries. Specifically, its element at (i, j) location is contributed by the thermal conductance between the i^{th} and the j^{th} cell, and the thermal conductance between the i^{th} cell and its adjacent isothermal boundary will be added onto the i^{th} diagonal element. Since the thermal conductivities are temperature-dependent, the term $\tilde{\mathbf{A}}_n(\mathbf{T}) \mathbf{T}$ is a nonlinear term. $\tilde{\mathbf{R}}_s \in \mathbb{R}^{n \times n}$ is another diagonal matrix to describe the external radiation, and its diagonal will be populated with the external radiative heat transfer coefficient at the boundaries. Note that along the diagonal of $\tilde{\mathbf{R}}_s$, only the locations corresponding to the cells adjacent to the radiation boundaries will have non-zero values. $\mathbf{T}_{\infty,R} \in \mathbb{R}^n$ is the environmental temperature values at the far field for the external radiation heat transfer. $\tilde{\mathbf{B}}_c \in \mathbb{R}^n$ is the constant source term. $\tilde{\mathbf{B}} \in \mathbb{R}^{n \times d}$ is the matrix converting the contribution from the time-dependent input vector $\mathbf{u} \in \mathbb{R}^d$ to each cell, and d is the number of different inputs. $\mathbf{u}$ is the time-dependent heat flux applied at the external boundaries of the model. Although the present method is developed using finite volume discretization, it could also be formulated and solved with the FEM approach, which possesses optimality properties and may be more favorable for solving parabolic equations.

The FOM in Eq. 3.16 can be written in a concise form as:

$$\frac{d\mathbf{T}}{dt} = \mathbf{A}\mathbf{T}(t) + \mathbf{R}\mathbf{T}(t)^{*4} + \mathbf{D} + \mathbf{B}\mathbf{u}, \quad (3.17)$$

where $\mathbf{A}$, $\mathbf{R}$, $\mathbf{D}$, and $\mathbf{B}$ are obtained by combining several terms in Eq. 3.16, viz.,

$$\begin{aligned}\mathbf{A} &= \mathbf{M}^{-1} \tilde{\mathbf{A}}_n, \\ \mathbf{R} &= \mathbf{M}^{-1} \tilde{\mathbf{R}}_s, \\ \mathbf{D} &= \mathbf{M}^{-1} \left(\tilde{\mathbf{R}}_s \mathbf{T}_{\infty, R}^{*4} + \tilde{\mathbf{B}}_c \right), \\ \mathbf{B} &= \mathbf{M}^{-1} \tilde{\mathbf{B}}.\end{aligned}\tag{3.18}$$

$\mathbf{AT}(t)$ captures the dependence of internal thermal conduction and the external convection on the temperature, and forms the inter-cell nonlinear term $\mathbf{F}(\mathbf{T}(t))$ in Eq. 3.7. The cell-wise nonlinear term $\mathbf{G}(\mathbf{T}(t))$ consists of three parts: the radiation term $\mathbf{RT}(t)^{*4}$, the term $\mathbf{D}$ incorporating far field conditions for external radiation and constant source terms, and the term $\mathbf{Bu}$ associated with target inputs. The thermal capacitance/mass matrix is diagonal, and $\mathbf{M}^{-1}$ in Eq. 3.18 represents the normalization of each individual equation in Eq. 3.16 by the corresponding thermal mass/capacitance of that equation. Therefore, $\mathbf{M}^{-1}$ is simply obtained by replacing diagonal elements of $\mathbf{M}$ with their reciprocals without need for matrix factorization. Note that in this chapter, each individual nonlinear term is treated separately rather than being combined together following [28] [19].

Local ROM utilized in the hybrid snapshot simulation is obtained by projecting FOM onto the subspace spanned by the POD basis $\mathbf{U}_r$ with the dimension r , i.e., $\mathbf{T}(t) \approx \mathbf{U}_r \mathbf{T}_r(t)$. Note that no DEIM is applied to ROM during the snapshot simulation in order to maintain high quality of the snapshot data:

$$\frac{d\mathbf{T}_r(t)}{dt} = \mathbf{A}_r \mathbf{T}_r(t) + \mathbf{R}_r (\mathbf{U}_r \mathbf{T}_r(t))^{*4} + \mathbf{D}_r + \mathbf{B}_r \mathbf{u},\tag{3.19}$$

where $\mathbf{T}_r \in \mathfrak{R}^r$ is the solution in the reduced domain.

$$\begin{aligned}\mathbf{A}_r &= \mathbf{U}_r^T \mathbf{A} \mathbf{U}_r \in \mathfrak{R}^{r \times r}, \\ \mathbf{R}_r &= \mathbf{U}_r^T \mathbf{R} \in \mathfrak{R}^{r \times n}, \\ \mathbf{D}_r &= \mathbf{U}_r^T \mathbf{D} \in \mathfrak{R}^r, \\ \mathbf{B}_r &= \mathbf{U}_r^T \mathbf{B} \in \mathfrak{R}^{r \times d}.\end{aligned}\tag{3.20}$$

For our heat transfer model, the thermal conductance matrix $\mathbf{A}$ depends on the thermal conductivity that varies with temperature values at the present and the adjacent cells. Therefore, $\mathbf{A}_r$ in Eq. 3.19 is also temperature dependent, and cannot be precomputed prior to the online simulation. As discussed above in Section 3.1.1, its computation requires multiple steps, which for the numerical heat transfer example includes reconstructing the solution in the full domain $\mathbf{T}(t) \approx \mathbf{U}_r \mathbf{T}_r(t)$; computing the thermal conductivity and the conductance at the face, assembling the thermal conductance matrix $\mathbf{A}$, and projecting $\mathbf{A}$ onto $\mathbf{U}_r$ to obtain its reduced counterpart $\mathbf{A}_r$.

3.3.2 ROM-DEIM FOR ONLINE SIMULATION

Updating and assembling the inter-cell transport matrix $\mathbf{A}_r$ in ROM above needs to be performed for all computational cells in each iteration loop and each time step, which is computationally demanding and time consuming. In fact, even for the cell-wise nonlinear terms, the solution reconstruction and assembly in the full domain need to be carried out. Therefore, ROM-DEIM will be applied in the online simulation to achieve further acceleration. Following the procedure presented in Section 3.2.1, the data in the FOM interval and the data reconstructed in the ROM interval are combined to form the snapshot data $\mathbf{N}_A$, $\mathbf{N}_R$ and $\mathbf{N}_B$ for the nonlinear terms $\mathbf{A}\mathbf{T}(t)$, $\mathbf{R}\mathbf{T}(t)^{*4}$ and $\mathbf{B}\mathbf{u}$, respectively. Note that $\mathbf{D}$ is a constant term in the present study, and DEIM will not be applied onto it. Then, their individual POD modes can be computed using SVD according to Eq. 3.11, yielding $\mathbf{U}_A \in \mathbb{R}^{n \times m_A}$, $\mathbf{U}_R \in \mathbb{R}^{n \times m_R}$ and $\mathbf{U}_B \in \mathbb{R}^{n \times m_B}$, respectively, where m_A , m_R and m_B , are the number of POD modes kept and also the number of the DEIM cells for interpolation. Thus, the DEIM interpolation matrices for each nonlinear term can be precomputed prior to the online ROM verification simulation following Eq. 3.13:

$$\begin{aligned} \frac{d\mathbf{T}_r(t)}{dt} = & \left(\mathbf{U}_r^T \mathbf{C}_A\right) \mathbf{P}_A^T \mathbf{A} \mathbf{U}_r \mathbf{T}_r(t) + \left(\mathbf{U}_r^T \mathbf{C}_R\right) \mathbf{P}_R^T \mathbf{R} \left(\mathbf{U}_r \mathbf{T}_r(t)\right)^{*4} + \\ & \mathbf{D}_r + \left(\mathbf{U}_r^T \mathbf{C}_B\right) \mathbf{P}_B^T \mathbf{B} \mathbf{u}, \end{aligned} \quad (3.21)$$

where $\mathbf{C}$ in each term on the RHS of Eq. 3.21 can be written as:

$$\mathbf{C}_i = \mathbf{U}_i \left(\mathbf{P}_i^T \mathbf{U}_i\right)^{-1}, \quad (3.22)$$

where subscript i can be A , R and B to represent the quantities for the nonlinear term, $\mathbf{A}\mathbf{T}(t)$, $\mathbf{R}\mathbf{T}(t)^{*4}$ and $\mathbf{B}\mathbf{u}$, respectively. In Eq. 3.21, $\mathbf{P}_A^T \mathbf{A}$, $\mathbf{P}_R^T \mathbf{R}$ and $\mathbf{P}_B^T \mathbf{B}$ clearly show that only m_A , m_R and m_B rows of $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{B}$ matrices corresponding to the DEIM cells need to be assembled, and hence, reducing the computing cost. In fact, since $\mathbf{R}$ matrix is diagonal and subscript $*$ denotes element-wise operation, $\mathbf{P}_R^T \mathbf{R} \left(\mathbf{U}_r \mathbf{T}_r(t)\right)^{*4}$ in the 2^{nd} nonlinear term associated external radiation can be further simplified by $\mathbf{P}_R^T \mathbf{R} \mathbf{P}_R \left(\mathbf{P}_R^T \mathbf{U}_r \mathbf{T}_r(t)\right)^{*4}$, which signifies that only m_R columns of $\mathbf{R}$ and m_R rows of $\mathbf{U}_r$ at the DEIM cells will be used in matrix and vector multiplication to further improve efficiency. Eq. 3.21 eventually can be written in a more concise form used for the online simulation:

$$\frac{d\mathbf{T}_r(t)}{dt} = \mathbf{Q}_A \mathbf{A}_P \mathbf{U}_r \mathbf{T}_r(t) + \mathbf{Q}_R \mathbf{R}_P \left(\hat{\mathbf{U}}_r \mathbf{T}_r(t)\right)^{*4} + \mathbf{D}_r + \mathbf{Q}_B \mathbf{B}_P \mathbf{u}, \quad (3.23)$$

where the matrices:

$$\begin{aligned} \mathbf{Q}_A &= \mathbf{U}_r^T \mathbf{C}_A, \\ \mathbf{Q}_R &= \mathbf{U}_r^T \mathbf{C}_R, \\ \mathbf{Q}_B &= \mathbf{U}_r^T \mathbf{C}_B \end{aligned} \quad (3.24)$$

combine the subspace projection onto $\mathbf{U}_r$ with DEIM interpolation terms in Eq. 3.21 and can be pre-computed.

$$\begin{aligned} \mathbf{A}_P &= \mathbf{P}_A^T \mathbf{A}, \\ \mathbf{B}_P &= \mathbf{P}_B^T \mathbf{B} \end{aligned} \quad (3.25)$$

indicate that only rows at DEIM cells will be assembled for $\mathbf{A}$ and $\mathbf{B}$ matrix.

$$\mathbf{R}_P = \mathbf{P}_R^T \mathbf{R} \mathbf{P}_R \quad (3.26)$$

means only rows and columns of $\mathbf{R}$ matrix at DEIM cells will be assembled because it is a diagonal matrix, and accordingly $\hat{\mathbf{U}}_r$ denotes the rows of $\mathbf{U}_r$ selected at DEIM cells.

The temporal differential term of the ODE set of the FOM and ROM is discretized by the second-order, implicit Crank-Nicolson method. The resultant linear equation set is solved using Matlab's built-in function "\ " (backslash), which examines on the matrix properties and determines an appropriate solution technique to minimize computation time.

3.4 NUMERICAL RESULTS AND DISCUSSIONS

Two numerical case studies based on the aforementioned nonlinear heat transfer model are performed to validate the feasibility of hybrid snapshot simulation for ROM-DEIM construction. Similar to Chapter.2, the first case study is the annular cone model shown in Fig. 3.5(a). The adiabatic boundary condition is applied at the bottom, the top, the inner wall, and a quarter of the outer lateral wall. A time-varying heat flux is prescribed at the rest of the outer lateral walls (the rest three quarters) with the profiles in both the snapshot simulation and the online simulation are depicted in Fig. 3.5(b).

One part of the model nonlinearity is introduced through the temperature-dependent thermal conductivity of both materials as $k = 11.13 + 1.27 \times 10^{-2} \times T$ for stainless steel and $k = 0.03 + 0.2438 \times \left(\frac{T}{1000}\right)^3$ for mineral wool. In order to confirm its importance, another analysis, which takes constant thermal conductivity at room temperature(300K), i.e., $k = 15W/mK$ and $0.0366W/mK$, is also performed and used as a benchmark for comparison.

The second case study considers the fin array made of stainless steel with the geometry and computational grids presented in Fig. 3.6(a). The lateral faces of the substrate are given the adiabatic boundary condition. The bottom of the sub-

Figure 3.5 The numerical model of the multi-layer annular cone: (a) geometry and computational grids; and (b) the heat flux profile prescribed at the three quarters of the outer wall of the model and used in the snapshot simulation (in pink) and the online simulation (in green).

strate is assigned a boundary condition of time-varying heat flux, whose profiles for the snapshot simulation and the online simulation are shown in Fig. 3.7(b). For the rest surfaces, the radiation boundary condition is prescribed with the far-field environmental temperature set to 300K.

Following the procedure presented above, the hybrid snapshot simulation is undertaken, and alternates between FOM and ROM according to the criteria. iSVD is used to update the POD modes of the temperature solution at the end of each FOM interval. At the end of snapshot simulation, the snapshot data of all the nonlinear terms are combined, from which the POD modes, and indices of DEIM interpolation points (i.e., DEIM cells) and their adjacent cells are also obtained. Then ROM-DEIM is built, and the online simulation is carried out. Three ROMs are compared, including ROM obtained from FOM snapshot simulation, ROM from hybrid snapshot simulation without DEIM, and ROM with DEIM (i.e., ROM-DEIM). Several abbreviations are defined first to facilitate result discussion below. For snapshot simulation, **SS-Hybrid** is defined to represent the hybrid simulation that alternates between

Figure 3.6 The model of extended fin arrays: (a) Geometry and grids; and (b) The heat flux profile prescribed at the bottom of the substrate of the computational model and used both in the snapshot simulation (in pink) and the online simulation (in green).

FOM and ROM simulation to generate snapshot data. The ROM obtained in this way without DEIM is termed **HROM**. Alternatively, at the end of **SS-Hybrid**, the solution during the ROM interval can be used to reconstruct the data in the full domain, which is then combined with FOM data to form the mixed snapshot data of the nonlinear terms. The POD modes and the DEIM interpolation indices of the nonlinear terms then can be computed. The ROM constructed by this process is named **HROM-DEIM**, which is the main contribution of the present effort. The traditional snapshot simulation exclusively using FOM is named **SS-FOM**, which is also used as the benchmark for comparison with other ROMs. In **SS-FOM**, the POD modes will be extracted only at the end of the snapshot simulation, and the ROM constructed is termed **FROM**.

In the online simulation, the three ROMs built by different methods above, including **HROM**, **HROM-DEIM**, and **FROM** will be compared in terms of accuracy and computational speed. Our investigation indicates that the assembly of the inter-cell transport-related term, i.e., $\mathbf{AT}(t)$ is most time consuming as it involves all the

interior cells rather than only the boundary cells, and the number of the DEIM cells (interpolation points) notably affects the accuracy of **HROM-DEIM**. Therefore, m_A is varied to examine its influence on accuracy and computational time, while keeping the DEIM numbers in the cell-wise terms (i.e., m_R and m_B) fixed.

In order to examine the errors quantitatively, two error metrics widely used for FOM and ROM comparison, including the absolute error $\mathbf{Err}_{abs}$ and the mean square relative difference(MSRD) $\mathbf{Err}_{msrd}$ [100] [10] are defined:

$$\mathbf{Err}_{abs}(t) = | \mathbf{U}_r \mathbf{T}_r(t) - \mathbf{T}(t) |, \quad (3.27)$$

$$\mathbf{Err}_{msrd}(t) = \sqrt{\frac{\sum_{j=1}^n \left(\frac{(\mathbf{U}_r \mathbf{T}_r(t))_j - \mathbf{T}_j(t)}{\mathbf{T}_j(t)} \right)^2}{n}} \times 100\%, \quad (3.28)$$

where subscript j denotes the j^{th} row of the reconstructed temperature solution $\mathbf{U}_r \mathbf{T}_r(t)$ corresponding to the j^{th} cell; $\mathbf{T}_r$ is the ROM/ROM-DEIM solutions of the temperature in the j^{th} cell; and $\mathbf{T}_j$ is the FOM solutions at the j^{th} cell. It should be noted that both of these errors $\mathbf{Err}_{abs}(t)$ and $\mathbf{Err}_{msrd}(t)$ vary with time. The common parameters in the numerical model and computation in both case studies are listed as follows: the number of basic and extended POD modes is $r_d = 20$ and $r_e = 4$, respectively; the FOM/ROM switch tolerance parameters are $\epsilon_1 = 1e^{-5}$, $\epsilon_2 = 1e^{-6}$ and $\epsilon = 1e^{-4}$.

3.4.1 CASE STUDY 1

For the annular cone model, the total time span of the snapshot simulation is 480s, which is divided into 12 time intervals, corresponding $\Delta t = 40s$ for each interval. The time span for the online simulation is also 480s, in which various ROMs will be simulated and their results will be compared with FOMs. In both the snapshot and online simulation, the time step is $\delta t = 0.25s$.

Figure 3.7 : The comparison between SS-FOM and SS-Hybrid in the annular cone model: (a) the snapshot data produced by SS-FOM and (b) the absolute error between SS-FOM and SS-Hybrid.

Fig. 3.7 shows the numerical results obtained in the snapshot simulation using FOM only (i.e., SS-FOM), and the absolute error of the snapshot data computed by the hybrid FOM/ROM (i.e., SS-Hybrid) relative to SS-FOM. Note that SS-hybrid data is not shown here, because SS-FOM and SS-Hybrid snapshot data are nearly the same with very small errors that are almost visually indistinguishable (maximum $Err_{abs} \leq 10K$ with respect to the peak temperature $\sim 800K$). This also confirms the accuracy of SS-Hybrid to generate snapshot data. As show in Table 3.1, among 12 intervals, there are 4 intervals and 8 intervals, respectively, simulated by FOM and ROM. That is, the 1st, 5th, 8th, and 10th intervals are the FOM interval, and the rest are the ROM interval. Table. 3.1 also lists the computational time breakdown of each interval during the entire snapshot simulation, including the FOM simulation (4 intervals), the iSVD after each FOM interval (3 iSVD and 1 SVD), and the ROM simulation (8 intervals). Note that following the initial FOM interval the SVD is applied rather than iSVD to construct the initial POD modes. It also indicates that with the hybrid method, 8 out of 12 intervals are simulated by ROM in place of FOM to generate the snapshot data without appreciably compromising the data accuracy

and quality. The computational time of ROM is greatly less than FOM, and hence, accelerating the snapshot simulation. Quantitatively, the total simulation time of SS-Hybrid and SS-FOM is, respectively, 132.759s and 264.564s, yielding 50% speedup.

Table 3.1 The breakdown of the computational time for the hybrid snapshot simulation for the annular cone model (Unit:s)

Time Section	FOM	iSVD	ROM
1 st	36.767	0.058	...
2 nd	...	...	4.476
3 th	...	...	4.641
4 th	...	...	4.553
5 th	19.385	0.229	...
6 th	...	...	4.431
7 th	...	...	4.770
8 th	18.981	0.168	...
9 th	...	...	4.522
10 th	20.061	0.173	...
11 th	...	...	4.863
12 th	...	...	4.681
Total	95.194	0.628	36.937

Next, several ROMs constructed by different methods are compared in the online simulation, including FROM by SS-FOM, HROM by SS-Hybrid, and HROM-DEIM by SS-Hybrid. The solution of the FROM and HROM are shown in Fig. 3.8(a-1) and Fig. 3.8(b-1), and their comparisons with the FOM solution, i.e., Err_{abs} are presented in Fig. 3.8(a-2) and Fig. 3.8(b-2). It is clear that solutions of FROM and HROM are nearly the same with negligible difference, which confirms the accuracy of ROM and POD modes constructed using the hybrid snapshot simulation. Fig. 3.8(a-2) and Fig. 3.8(b-2) show that absolute errors Err_{abs} of both FROM and HROM increase with time and reach the highest values of $12K$ at the end of the online simulation, when the temperature of the whole model also approaches the peak. The value of Err_{abs} of HROM is slightly higher than that of FROM, because less high-fidelity FOM data is generated in SS-Hybrid.

(a-1)

(b-1)

(a-2)

(b-2)

Figure 3.8 The comparison of FROM and HROM with FOM for the annular cone model: the temperature solution of all the computational cells throughout the verification simulation obtained by (a-1) FROM and (b-1) HROM; and the absolute error Err_{abs} of (a-2) FROM and (b-2) HROM relative to FOM.

Next HROM-DEIM is constructed for the online simulation in order to reduce the computational cost associated with the nonlinear term calculation in the reduced domain. By keeping constant the number of the DEIM cells (interpolation points) for all the cell-wise terms ($m_R = 100$ and $m_B = 100$), we investigate the effect of the DEIM cell number of the inter-cell transport term on accuracy and speed. The rationale for us to choose a large number of DEIM cells for the cell-wise nonlinear terms is to ensure they can be approximated with higher accuracy and the effect of DEIM interpolation on them can be minimized. Thus the analysis can be concentrated on the inter-cell transport nonlinear term. Fig. 3.9 illustrates the temperature solution of all computational cells throughout the verification simulation produced by the three HROM-DEIMs with the different number of DEIM cells for the inter-cell nonlinear term ($m_A = 40, 60$ and 80) and HROM without DEIM. It is observed that the numerical results for all of them are almost the same in the entire spatiotemporal domain. The initial temperature values are equal to the environmental one ($300K$). The temperature rises quickly between $100s$ and $300s$, and eventually it reaches the highest value about $850K$ at the end of the simulation. Fig. 3.10 illustrates the absolute errors Err_{abs} of the numerical results of the HROM-DEIM in Fig. 3.9 relative to the FOM solution. It clearly shows that Err_{abs} increases with time for all the computational cells, and there seems a trend that Err_{abs} is larger when a smaller number of DEIM cells are used. The maximum value of Err_{abs} produced by HROM-DEIM with $m_A = 40$ is about $15K$, while it is less than $10K$ with $m_A = 80$. In addition, it is also observed that the values of Err_{abs} vs. time for HROM (Fig. 3.10d) are slightly lower than that of HROM-DEIM (Fig. 3.10a-c), which verifies that HROM without DEIM interpolation (or approximation) is more accurate than HROM-DEIM. However, with DEIM the computational complexity and cost could be reduced significantly, leading to salient simulation acceleration as shown below.

Figure 3.9 The temperature solutions obtained by HROM-DEIM with the different number of DEIM cells in the annular cone model: (a) DEIM 40 ($m_A = 40$), (b) DEIM 60 ($m_A = 60$), (c) DEIM 80 ($m_A = 80$), and (d) non-DEIM.

The values of the mean square root difference error (MSRD) Err_{msrd} of all the ROM solutions in Fig. 3.10 relative to FOM are shown in Fig. 3.11. A trend similar to Fig. 3.10 is observed. That is, Err_{msrd} seems to decrease as a result of the increasing number of DEIM cells, and Err_{msrd} of HROM without DEIM is the lowest in the comparison. Even with only 40 DEIM cells (i.e., $m_A = 40$), Err_{msrd} is still less than 0.8%. This indicates that DEIM interpolation of the nonlinear inter-cell transport (such as temperature-dependent thermal conduction) is an effective means to improve nonlinear ROM simulation speed while preserving the accuracy. It should

Figure 3.10 The absolute error of the temperature solution obtained by HROM-DEIM with the different number of DEIM cells in the annular cone model relative to the FOM solution: (a) DEIM 40 ($m_A = 40$), (b) DEIM 60 ($m_A = 60$), (c) DEIM 80 ($m_A = 80$), and (d) non-DEIM.

Figure 3.11 The MSRD errors vs. time for HROM-DEIM with the different number of DEIM cells and HROM without DEIM in the verification simulation in the annular cone model.

be noted that DEIM selects the cells or interpolation points based on the largest magnitude of the residual of reconstructing the next basis vector in POD modes [27] [29]; therefore, which is not necessarily equivalent to ROM accuracy.

Table 3.2 The comparison of computational time in **FOM**, **HROM**, and **HROM-DEIM** in the annular cone model. (Unit:s).

Model Type	FOM	HROM	HROM-DEIM		
			$m_A = 40$	$m_A = 60$	$m_A = 80$
Time	293.517	54.702	15.065	15.117	15.459

Table. 3.2. lists the computational time of FOM, HROM, and HROM-DEIM in the online simulation. It is clearly seen that ROMs could accelerate the simulation greatly, and the computational time of HROM is only about 18.6% of FOM. However, HROM-DEIM is able to achieve an even higher acceleration than HROM due to the reduced cost of assembling the nonlinear terms. The ratio of the computational time of HROM-DEIM to HROM is 27.5% and to FOM is only 5.13% $\sim$ 5.27%, latter of which indicates that the time usage by HROM-DEIM is almost negligible compared to the traditional method. Most importantly, with the hybrid simulation method, the

snapshot generation used to construct the HROM-DEIM can be reduced by almost half (see Table. 3.1). In addition, despite various m_A used (40, 60, and 80) in HROM-DEIMs, they can be simulated almost with the same time $\sim 15s$, because all m_{AS} are quite small compared to the dimension of FOM.

For the annular cone model, the difference caused by the temperature-dependent thermal conductivity relative to the constant thermal conductivity is also briefly examined in order to demonstrate that the inter-cell transport-induced nonlinearity in this chapter plays an important role for accurate modeling. The values of temperature-dependent conductivity and the constant conductivity in constituent materials are given above. Fig. 3.12 shows the temperature solution of FOM computed using the constant conductivity, and its difference (Err_{abs}) relative to the solution using temperature-dependent conductivity. For the sake of conciseness, the FOM solutions are not shown as they are almost the same as those in Figs. 3.8 and 3.9. It clearly shows that the difference(Err_{abs}) is larger as the temperature increases with time, and the maximum could reach nearly $200K$ in the sub-domain containing the material of low thermal conductivity that is more sensitive to the temperature. Therefore, it is confirmed that the temperature-dependent, inter-cell transport phenomenon gives rise to notable model nonlinearity, and the nonlinear term updating and assembly during each iteration and each time step is necessary.

3.4.2 CASE STUDY 2

In the second case study of the extended fin array model, the total time span of the snapshot simulation is $1200s$ and divided into 12 time intervals for the hybrid FOM/ROM simulation, yielding $\Delta t = 100s$ for each time interval. The time span for the entire online simulation is $960s$. The time step for both the snapshot and online simulation is $\delta t = 0.5s$.

Figure 3.12 Effect of the temperature-dependent thermal conductivity on model nonlinearity: (a) the temperature solution obtained by FOM in the annular cone model using the constant thermal conductivity; and (b) the absolute error between the solution in (a) and the solution obtained using temperature-dependent thermal conductivity.

Figure 3.13 The comparison between SS-FOM and SS-Hybrid in the extended fin array model: (a) the snapshot data produced by SS-FOM and (b) the absolute error between SS-FOM and SS-Hybrid.

Fig. 3.13(a) shows the numerical results of the temperature solution generated using SS-FOM in the snapshot simulation, and Fig. 3.13(b) portrays the absolute error Err_{abs} of the snapshot data between SS-FOM and SS-Hybrid for all the computational cells throughout the entire time span. Again in order to avoid presenting redundant data, the temperature solution of SS-Hybrid is not shown. We can see that the maximum value of Err_{abs} is less than $8K$ and is negligible considering the largest magnitude of the snapshot data (up to $1,100K$), which again substantiates the robustness and accuracy of the hybrid snapshot simulation.

The breakdown of the computational time using SS-Hybrid in the snapshot simulation is listed in Table. 3.3. It can be seen that among the 12 time intervals, there are only 2 intervals simulated by FOM (the 1st and the 8th), and the rest 10 intervals all by ROM. Except for the first FOM interval that uses SVD to construct the initial POD modes, iSVD is only applied once following the 8th interval in the whole snapshot simulation. The table also clearly shows that the snapshot simulation switches to 10 ROM intervals based on the FOM/ROM criteria, and the total computational time ($489.686s$) is greatly lowered compared to the traditional FOM-based snapshot simulation ($929.748s$), i.e., 47% reduction.

Table 3.3 The breakdown of the computational time for the hybrid snapshot simulation for the extended fins model (Unit:s)

Time Section	FOM	iSVD	ROM
1 st	123.085	0.362	...
2 nd	...	...	29.270
3 th	...	...	28.072
4 th	...	...	28.146
5 th	...	...	28.051
6 th	...	...	28.003
7 th	...	...	41.118
8 th	72.070	1.031	...
9 th	...	...	27.902
10 th	...	...	27.534
11 th	...	...	27.226
12 th	...	...	27.816
Total	195.155	1.393	293.138

Next, we will compare performance of all the ROMs in the online simulation. Fig. 3.14(a-1) and (b-1) illustrates numerical results of the temperature solution obtained by FROM and HROM, respectively. Again, both exhibit excellent agreement with very minor difference. Their absolute errors Err_{abs} relative to the FOM solution are shown in Fig. 3.14(a-2) and (b-2), and the values of both Err_{abs} are extremely low ($\leq 0.5K$) and indeed negligible compared with the absolute values of the temperature solution, although Err_{abs} of HROM is somewhat higher than FROM. The observations also indicate that the POD modes and ROM constructed by the hybrid snapshot method are accurate to be utilized in the online simulation.

The temperature solutions in the online simulation obtained by the three HROM-DEIMs with the different number of DEIM cells ($m_A = 50, 75$, and 100) and HROM without DEIM are illustrated in Figure 3.15. Similar to the previous case study, all the numerical results are almost identical with tiny difference. The temperature rises up rapidly in the first $300s$ until arriving the peak value ($\sim 1,000K$) and then gradually declines to about $450K$ at the end of the simulation as a result of the decreasing heat flux applied. Their absolute errors Err_{abs} relative to the FOM solution are shown in Fig. 3.16, and the profiles of Err_{abs} in all sub-figures exhibit the similar trend: Err_{abs} grows quickly from $0s$ to $200s$, decreases a little from $200s$ to $300s$, and then mostly maintains at approximately the same level from $300s$ till the end. Fig. 3.16 also seems to reveal the correlation between the DEIM cell number and the magnitude of Err_{abs} , that is, Err_{abs} decreases as the number of DEIM cells for the inter-cell transport term increases from $m_A = 50$ to $m_A = 100$. It is also noticed that Err_{abs} of HROM-DEIM (Fig. 3.16(c)) is slightly higher than that of HROM (Fig. 3.16(d)) because no DEIM interpolation or approximation is used in the latter.

Fig. 3.17 shows the MSRD error Err_{msrd} of all the HROM-DEIM solutions in Fig. 3.15 relative to the FOM solution. The trend similar to Err_{abs} in Fig. 3.16 is also found in Err_{msrd} . As the number of DEIM cells increases, Err_{msrd} seems to

Figure 3.14 The comparison of FROM and HROM with FOM for the extended fin array model: the temperature solution of all the computational cells throughout the verification simulation obtained by (a-1) FROM and (b-1) HROM; and the absolute error of (a-2) FROM and (b-2) HROM relative to FOM.

Figure 3.15 The temperature solutions obtained by HROM-DEIM with the different number of DEIM cells in the extended fin array model: (a) DEIM 50 ($m_A = 50$), (b) DEIM 75 ($m_A = 75$), (c) DEIM 100 ($m_A = 100$), and (d) non-DEIM.

decrease. It is also observed that Err_{msrd} without DEIM exhibits the lowest error compared with those with DEIM. Given 38400 computational cells in total, even a very small number of DEIM cells ($m_A = 50$) for inter-cell transport matrix updating and assembly can yield excellent accuracy (the highest $Err_{msrd} \leq 0.12\%$).

Table. 3.4 lists the computational time of FOM, HROM, and HROM-DEIM in the online simulation. The time used by HROM is about 33.3% of FOM, i.e., 66.6% improvement in simulation speed. Furthermore, with HROM-DEIM, more significant computational acceleration can be achieved. Specifically, the computational time

Figure 3.16 The absolute error of the temperature solution obtained by HROM-DEIM with the different number of DEIM cells in the extended fin array model relative to the FOM solution: (a) DEIM 50 ($m_A = 50$), (b) DEIM 75 ($m_A = 75$), (c) DEIM 100 ($m_A = 100$), and (d) non-DEIM.

Figure 3.17 The MSRD errors vs. time for HROM-DEIM with the different number of DEIM cells and HROM without DEIM in the verification simulation in extended fin array model.

ratio of HROM-DEIM to FOM is in the range of $5.48\% \sim 6.04\%$ and to HROM is $16.38\% \sim 18.03\%$. The studies above indicate that the DEIM interpolation is a viable approach to notably improve ROM efficiency while preserving the simulation accuracy, and the hybrid snapshot simulation could be utilized to construct high-quality HROM-DEIMs with much lower computational efforts/costs.

Table 3.4 The comparison of computational time in **FOM**, **HROM**, and **HROM-DEIM** in the extended fin array model. (Unit:s).

Model Type	FOM	HROM	HROM-DEIM		
			$m_A = 50$	$m_A = 75$	$m_A = 100$
Time	817.24	273.548	44.818	44.932	49.332

Similar to the above, the importance of the nonlinearity of the extended fin array model arising from the temperature-dependent thermal conductivity is also inspected. The temperature-dependent thermal conductivity and the constant thermal conductivity (evaluated at the environmental temperature) are given above. Fig. 3.18(a) shows the temperature solution predicted by FOM using the constant conductivity, and Fig. 3.18(b) portrays its difference, i.e., absolute errors Err_{abs} , from the FOM

Figure 3.18 Effect of the temperature-dependent thermal conductivity on model nonlinearity: (a) the temperature solution obtained by FOM in the extended fin array model using the constant thermal conductivity; and (b) the absolute error between the solution in (a) and the solution obtained using temperature-dependent thermal conductivity.

results using the temperature-dependent conductivity (not shown to avoid redundant results) for all the computational cells throughout the entire online simulation. We can see that the maximum value of Err_{abs} goes beyond $60K$, confirming that the aforementioned transport-related nonlinearity of the thermal conduction is important in the model.

3.5 SUMMARY OF CHAPTER.3

In this chapter, a methodology by combining the hybrid snapshot simulation and on-line DEIM is proposed to reduce the computational cost associated with ROM-DEIM construction and improve computational efficiency of online verification simulation for nonlinear ROMs. The new aspects of this study include: First, an approach is proposed to streamline the hybrid snapshot simulation, snapshot data generation, SVD/iSVD for POD mode construction (both solution variables and nonlinear terms), and ROM-DEIM construction. Second, the method of reconstructing snapshot data

of nonlinear terms using ROM solutions is presented to address the issue associated with limited FOM data in the hybrid simulation for enhanced data representation. Last, formulation and data structures are also developed to cast the ROM-DEIM in the hybrid snapshot simulation using the cell-centered FVM scheme to account for the transport-related (non-componentwise) nonlinearity. Key technical findings include:

1). The feasibility of using the snapshot data generated by hybrid simulation to construct ROM-DEIM is established. That is, the POD modes of the nonlinear terms distilled from the mix of the FOM and reconstructed ROM solution data accurately represent their underlying subspace, and can be used to determine DEIM cells/interpolation points.

2). In the cell-centered FVM formulation, the transport flux at the interface depends on the values of solution variables in the computational cells on both sides. Therefore, to construct ROM-DEIM, the cells adjacent to the DEIM cells/interpolation points also need to be extracted in order to update the quantities at the interface and assemble the entries of the nonlinear terms at the DEIM cells.

3). Following the method proposed in this chapter, an accurate global ROM-DEIM is immediately obtained at the end of the hybrid snapshot simulation that can be utilized for fast online simulation.

4). The combination of the hybrid snapshot simulation and ROM-DEIM construction demonstrate excellent computational performance in case studies of numerical heat transfer. Only a small number of intervals, 4 and 2 out of 12, are simulated by FOM in the two case studies, and the computational cost of the snapshot simulation is reduced by $\sim 50\%$ without compromising accuracy of ROM-DEIM. The performance of the developed ROM-DEIM with the different number of DEIM cells is also evaluated by comparing with the FOM solution for the online verification simulation. MSRD error less than 0.8% and 0.12% could be achieved for both case studies. The

precomputed POD modes of the nonlinear terms and nonlinear term updating and assembly at the DEIM cells significantly accelerate ROM simulation during the on-line verification simulation. ROM-DEIM only takes $\sim 5.13\%$ and $\sim 5.48\%$ relative to FOM simulation, and $\sim 27.5\%$ and $\sim 16.38\%$ relative to HROM without DEIM in both cases.

It should be noted that the ROM-DEIM and hybrid snapshot simulation may be ill-suited for convection-dominated problems that involve moving fronts of the field variables, such as Burgers equation with a shock propagating in a 1D domain. This is because the POD modal information and ROM constructed in earlier time intervals may not accurately represent that in the later time intervals, leading to poor approximation, which will be thoroughly investigated in Chapter.5. It will also include embedding GNAT into the snapshot simulation by developing new model switch criteria to further improve the computational efficiency of the hybrid snapshot simulation.

CHAPTER 4

DEIM-EMBEDDED HYBRID SNAPSHOT SIMULATION FOR REDUCED ORDER MODEL GENERATION

In this chapter, a DEIM-embedded hybrid approach is proposed in the snapshot simulation for further computational acceleration of snapshot data generation. Several novelties are featured: (1) a new methodology is proposed to embed DEIM into the snapshot simulation, which according to the best of our knowledge has not been reported before. Its feasibility and applicability is systematically investigated; (2) accordingly, multiple model switch criteria are proposed to make the model switch among FOM, ROM, and ROM-DEIM on-the-fly during the snapshot simulation; (3) two groups of POD basis vectors, including one for the solution variable and the other for the nonlinear terms generated in the snapshot simulation are used to monitor the trajectory of ROM and ROM-DEIM separately for rational model switch. Both of them are updated efficiently during the simulation using incremental SVD [17, 18] to minimize the computational cost of POD; and (4) a computationally efficient ROM-DEIM will form at the end of the proposed hybrid snapshot simulation without the need for another DEIM procedure, and the ROM-DEIM can be used to accelerate the online simulation.

The organization of this chapter is as follows: the methodology of the DEIM-embedded hybrid snapshot simulation is presented in Section 4.1. In Section 4.2 the general process of FOM, ROM and ROM-DEIM will be described, respectively. The reconstruction of snapshot data for nonlinear terms in ROM will also be discussed. In

Section 4.3 FOM, ROM, and ROM-DEIM for a numerical heat transfer model will be elucidated. In Section 4.4 we will introduce two case studies based on the numerical heat transfer model, and the performance of the proposed DEIM-embedded hybrid snapshot simulation will be characterized quantitatively. This chapter ends with a technical summary in Section 4.5.

4.1 METHODOLOGY OF DEIM-EMBEDDED HYBRID SNAPSHOT SIMULATION

In this section the proposed methodology of DEIM-embedded hybrid snapshot simulation to accelerate snapshot data generation and ROM construction is presented. As shown in Fig. 4.1, the whole time span of the snapshot simulation (labeled "1" in Fig. 4.1) is divided into multiple intervals. Each interval will be simulated by FOM, ROM, or ROM-DEIM. Note that the difference between ROM and ROM-DEIM is that in the former, the approximation only arises from projection of the solution variable, while in the latter both projection of the solution variable and DEIM approximation of the nonlinear terms are used to further accelerate the simulation [27, 29]. The computations undertaken within the FOM, ROM, and ROM-DEIM interval are given as follows:

- 1.) In the FOM interval, the FOM will be simulated to produce the snapshot data Ψ of the solution variable $\mathbf{Y}(t) \in \mathbb{R}^n$ in the full domain and the snapshot data Ψ^H for the nonlinear term will be obtained by applying the nonlinear operator $\mathbf{H}$ onto Ψ and stored in memory. where n is the dimension of FOM. At the end of the FOM interval (labelled "2" in Fig. 4.1), several steps of computation will be carried out, including: (i) singular value decomposition (SVD) or incremental singular value decomposition (iSVD) is applied onto Ψ and Ψ^H to create or update the local POD basis vectors $\mathbf{U}_r$ and $\mathbf{U}_r^H$; and (ii) two switch criteria, $\mathbf{C}_{F2R}$ and $\mathbf{C}_{R2D}$ will be evaluated to determine if the simulation in the next interval should remain using FOM

or switch to ROM or ROM-DEIM. If switched to ROM-DEIM, the DEIM procedure will be performed.

2.) In the ROM interval, the local ROM should be constructed first by projecting FOM onto the latest $\mathbf{U}_r$ generated at the end of the previous FOM interval, and the solutions in reduced domain $\mathbf{Y}_r(t) \in \mathbb{R}^r$ will be computed, where r is the dimension of ROM; the reduced solutions $\mathbf{Y}_r$ will be used to reconstruct the solutions in the full domain $\hat{\mathbf{Y}}(t)$, and the nonlinear operator $\mathbf{H}$ will run through $\hat{\mathbf{Y}}_r$ to yield reconstructed snapshot data of the nonlinear term $\hat{\Psi}^H$. The computation conducted at the end of the ROM interval (labeled “3” in Fig. 4.1) includes: (i) iSVD is then applied onto $\hat{\Psi}^H$ to update the local POD basis vectors $\mathbf{U}_r^H$; and (ii) two switch criteria, $\mathbf{C}_{R2F}$ and $\mathbf{C}_{R2D}$ will be evaluated to determine if ROM should continue being used in the next interval or the simulation should switch to FOM or ROM-DEIM. If switched to ROM-DEIM, the DEIM procedure will be performed.

3.) In the ROM-DEIM interval, ROM-DEIM will be built using the two POD basis vectors $\mathbf{U}_r$ and $\mathbf{U}_r^H$ generated from the previous FOM and ROM intervals. $\mathbf{U}_r$ is used to project the FOM onto the POD subspace, and $\mathbf{U}_r^H$ to determine the DEIM index and interpolate the nonlinear term $\mathbf{H}$ to accelerate the snapshot simulation. At the end of the ROM-DEIM interval (i.e., labeled “4” in Fig. 4.1), two criteria, $\mathbf{C}_{R2F}$ and $\mathbf{C}_{D2R}$ will be evaluated to determine which model, ROM-DEIM, ROM, or FOM should be utilized in the next interval.

The key idea of the proposed DEIM-embedded hybrid snapshot simulation approach is that the local ROM and ROM-DEIM can be used to quickly traverse intervals containing less or no new information of subspace updating to reduce the computational cost of snapshot simulation. As discussed above, FOM, ROM, and ROM-DEIM represents a hierarchy of model approximation from the most accurate and costly to the least. Therefore, they can be utilized selectively to meet different requirements of POD modes and accuracy during the hybrid snapshot simulation.

Figure 4.1 The methodology of DEIM-embedded hybrid snapshot simulation and the construction of ROM-DEIM in online simulation: (1) the whole process of DEIM-embedded hybrid snapshot simulation; (2) the SVD/iSVD updating of $\mathbf{U}_r$ and $\mathbf{U}_r^H$ and DEIM algorithm after each FOM interval (if switch to ROM or ROM-DEIM); (3) the SVD/iSVD updating of $\mathbf{U}_r^H$ and DEIM algorithm after each ROM interval (if switch to ROM-DEIM); (4) no updating in basis after each ROM-DEIM interval; (5) the online simulation in ROM-DEIM constructed by $\mathbf{U}_r$ and $\mathbf{U}_r^H$ in the hybrid snapshot simulation.

Specifically, FOM offers the highest-fidelity computation, and should be used only when necessary, viz., when both ROM and ROM-DEIM and their POD modes $\mathbf{U}_r$ and $\mathbf{U}_r^H$ are inaccurate and need to be updated. Although providing no new information of $\mathbf{U}_r$, ROM can be used to enrich the snapshot data of nonlinear terms, refine $\mathbf{U}_r^H$, and enhance the ROM-DEIM. On the other hand, while employing more approximations, ROM-DEIM is the most computationally efficient model, and its use should be maximized when allowed.

During the online simulation (label “5” in Fig. 4.1), the ROM-DEIM created at the end of the snapshot simulation along with the POD basis vectors $\mathbf{U}_r$ for the solution variable and $\mathbf{U}_r^H$ for the nonlinear terms and the DEIM interpolation points is used for computation. Since the numbers of DEIM interpolation points and the dimension of the projection subspace $\mathbf{U}_r$ are quite small compared to the FOM dimension, ROM-DEIM in the online simulation will run at an extremely fast speed. Note that time span partitioning is not necessary in the online simulation.

The key components of the method will be discussed below. Specifically, the logic threads of DEIM-embedded hybrid snapshot simulation are introduced in Section 4.1.1. The switch criteria for model selection are presented in Section 4.1.2. The SVD and iSVD procedure to generate and update POD modes of the solution variable and the nonlinear term is described in Section 4.1.3.

4.1.1 DEIM-EMBEDDED HYBRID SNAPSHOT SIMULATION

The entire time span of the snapshot simulation is divided into N_i intervals of the equal length, and the length of each interval is Δt . A constant time step of $\delta t \ll \Delta t$ is used for computing all the models with $\Delta t = N\delta t$, where N is the number of the time steps in each interval. Thus, the physical time that is simulated at the end of the i^{th} interval is $iN\delta t = i\Delta t$. As discussed above, the DEIM-embedded hybrid snapshot simulation is organized by switching the model among FOM, ROM and ROM-DEIM

1. As shown by logic thread “a”, two switch criteria, $\mathbf{C}_{F2R}$ and $\mathbf{C}_{R2D}$, respectively, labeled as “a-1” and “a-2” in Fig. 4.2, will be evaluated at the end of the FOM interval to determine which model to use in the next interval. $\mathbf{C}_{F2R}$ will first be evaluated. If it is false, FOM will remain in use in the next interval, which implies that both ROM and ROM-DEIM are not sufficiently accurate, and their POD modes $\mathbf{U}_r$ and $\mathbf{U}_r^H$ need to be updated. Otherwise, either ROM or ROM-DEIM should be used in the next interval, and the second criterion $\mathbf{C}_{R2D}$ will be examined. If it is false, ROM-DEIM is still inaccurate, and only the POD modes of the nonlinear term $\mathbf{U}_r^H$

needs to be improved, and ROM should be employed in the next interval. If true, ROM-DEIM is considered accurate, and will be accepted in the next interval.

2. At the end of the ROM interval, two switch criteria, $\mathbf{C}_{R2F}$ and $\mathbf{C}_{R2D}$, respectively, labeled as “b-1” and “b-2” in logic thread “b” in Fig. 4.2, will be evaluated. $\mathbf{C}_{R2F}$ will be evaluated first. If it is true, ROM starts to deviate from its subspace, leading to large errors, and the simulation will return to FOM in the next interval. Otherwise, the second criterion $\mathbf{C}_{R2D}$ will be inspected. Similar to the above, depending on its outcome (false or true), ROM-DEIM or ROM should be utilized.

3. At the end of the ROM-DEIM interval, two switch criteria, $\mathbf{C}_{R2F}$ and $\mathbf{C}_{D2R}$, respectively, labeled as “c-1” and “c-2” in logic thread “c” in Fig. 4.2, will be evaluated. $\mathbf{C}_{R2F}$ will be evaluated first. If true, the simulation will return to FOM in the next interval because ROM-DEIM starts to become inaccurate. Otherwise, the second criterion $\mathbf{C}_{D2R}$ will be evaluated to examine the trajectory of the nonlinear terms, and a “true” response indicates that ROM-DEIM and associated POD modes of nonlinear terms are not accurate, giving rise to large errors. Therefore, the simulation is switched to ROM. Otherwise, ROM-DEIM will remain in use.

The POD basis vectors $\mathbf{U}_r$ need updates after each FOM interval while $\mathbf{U}_r^H$ need updates after each FOM or each ROM. The interpolating points of DEIM [27, 29] should be updated based on the updates of nonlinear terms $\mathbf{U}_r^H$ in each FOM or ROM (if switch to ROM-DEIM in the next interval). Algorithm. 3 show the entire algorithm of DEIM-embedded hybrid snapshot simulation.

Algorithm 3: The DEIM-embedded hybrid algorithm in snapshot simulation.

```

Input:  $r_d, r_e, \varepsilon_1, \varepsilon_2$ , and  $\varepsilon, \varepsilon^{H_1}, \varepsilon^{H_2}$  and  $\varepsilon^{H_3}, N, N_i, \Delta t$  and the initial  $\mathbf{Y}_0$ .

% The first interval in the time span
 $[\Psi, \Psi^H, \mathbf{Y}] = \text{RunFOM}();$  % Compute snapshot data  $\Psi, \Psi^H$  using FOM
%  $\Psi^H = \mathbf{H}(\mathbf{Y})$  % Represent snapshot data  $\Psi^H$  for nonlinear terms in FOM
 $\Psi = \mathbf{U}\Sigma\mathbf{V}^T; \mathbf{U}_r = \mathbf{U}(:, 1:r_d+r_e);$  % Compute POD modes  $\mathbf{U}_r$  using SVD
 $\Psi^H = \mathbf{U}^H\Sigma^H\mathbf{V}^{H^T}; \mathbf{U}_r^H = \mathbf{U}^H(:, 1:m_H);$  % Compute POD modes for nonlinear parts  $\mathbf{U}_r^H$  using SVD
% Evaluate switching criteria to decide which model to use in the next interval: FOM/ROM/ROM-DEIM
if  $(C_{F2R,1} \leq \varepsilon_1$  and  $C_{F2R,2} \leq \varepsilon_2)$ 
    SolverFlag = ROM/ROM-DEIM with an order of  $(r_d + r_e)$  % ROM/ROM-DEIM will be used in the next interval
else
    SolverFlag = FOM % FOM will be used in the next interval
end
if SolverFlag == ROM/ROM-DEIM
    if  $(C_{R2D,1} \leq \varepsilon_1^H$  and  $C_{R2D,2} \leq \varepsilon_2^H)$ 
        SolverFlag = ROM-DEIM with an order of  $(r_d + r_e)$  % ROM-DEIM will be used in the next interval
        DEIM in  $\mathbf{H}(\mathbf{Y})$  using  $\mathbf{U}_r^H$ ; % DEIM procedure for nonlinear parts
    else
        SolverFlag = ROM % ROM will be used in the next interval
    end
end

for  $i = 2: N_i$  % Compute the time intervals from 2 to  $N_i$ 
    if SolverFlag == FOM
         $[\Psi, \Psi^H, \mathbf{Y}] = \text{RunFOM}();$  % Compute snapshot data  $\Psi, \Psi^H$  using FOM
        %  $\Psi^H = \mathbf{H}(\mathbf{Y})$  % Represent snapshot data  $\Psi^H$  for nonlinear terms in FOM
         $[\mathbf{U}, \Sigma, \mathbf{V}] = \text{iSVD}[\mathbf{U}, \Sigma, \mathbf{V}, \Psi]; \mathbf{U}_r = \mathbf{U}(:, 1:r_d+r_e);$  % iSVD updating of  $\mathbf{U}_r$ 
         $[\mathbf{U}^H, \Sigma^H, \mathbf{V}^H] = \text{iSVD}[\mathbf{U}^H, \Sigma^H, \mathbf{V}^H, \Psi^H]; \mathbf{U}_r^H = \mathbf{U}^H(:, 1:m_H);$  % iSVD updating of  $\mathbf{U}_r^H$ 
        if  $(C_{F2R,1} \leq \varepsilon_1$  and  $C_{F2R,2} \leq \varepsilon_2)$ 
            SolverFlag = ROM/ROM-DEIM with an order of  $(r_d + r_e)$  % ROM/ROM-DEIM will be used in the next interval
        else
            SolverFlag = FOM % FOM will be used in the next interval
        end
        if SolverFlag == ROM/ROM-DEIM
            if  $(C_{R2D,1} \leq \varepsilon_1^H$  and  $C_{R2D,2} \leq \varepsilon_2^H)$ 
                SolverFlag = ROM-DEIM with an order of  $(r_d + r_e)$  % ROM-DEIM will be used in the next interval
                DEIM in  $\mathbf{H}(\mathbf{Y})$  using  $\mathbf{U}_r^H$ ; % DEIM procedure for nonlinear parts
            else
                SolverFlag = ROM % ROM will be used in the next interval
            end
        end
    elseif SolverFlag == ROM-DEIM
         $[\hat{\mathbf{H}}, \Psi, \mathbf{Y}_r] = \text{RunROM-DEIM}();$  % Compute snapshot data  $\Psi$  using ROM-DEIM
        if  $(C_{R2F} \geq \varepsilon)$ 
            SolverFlag = FOM % FOM will be used in the next interval
        else
            SolverFlag = ROM/ROM-DEIM % ROM/ROM-DEIM will be used in the next interval
        end
        if SolverFlag == ROM/ROM-DEIM
            if  $(C_{D2R} \geq \varepsilon^H)$ 
                SolverFlag = ROM % ROM will be used in the next interval
            else
                SolverFlag = ROM-DEIM % ROM-DEIM will be used in the next interval
            end
        end
    elseif SolverFlag == ROM
         $[\Psi, \Psi^H, \mathbf{Y}_r] = \text{RunROM}();$  % Compute snapshot data  $\Psi, \Psi^H$  using ROM
        %  $\Psi^H = \hat{\mathbf{H}}(\mathbf{U}, \mathbf{Y}_r)$  % Represent snapshot data  $\Psi^H$  using reconstructed full solution for nonlinear terms
         $[\mathbf{U}^H, \Sigma^H, \mathbf{V}^H] = \text{iSVD}[\mathbf{U}^H, \Sigma^H, \mathbf{V}^H, \Psi^H]; \mathbf{U}_r^H = \mathbf{U}^H(:, 1:m_H);$  % iSVD updating of  $\mathbf{U}_r^H$ 
        if  $(C_{R2F} \geq \varepsilon)$ 
            SolverFlag = FOM % FOM will be used in the next interval
        else
            SolverFlag = ROM/ROM-DEIM % ROM/ROM-DEIM will be used in the next interval
        end
        if SolverFlag == ROM/ROM-DEIM
            if  $(C_{R2D,1} \leq \varepsilon_1^H$  and  $C_{R2D,2} \leq \varepsilon_2^H)$ 
                SolverFlag = ROM-DEIM with an order of  $(r_d + r_e)$  % ROM-DEIM will be used in the next interval
                DEIM in  $\mathbf{H}(\mathbf{U}, \mathbf{Y}_r)$  using  $\mathbf{U}_r^H$ ; % DEIM procedure for nonlinear parts
            else
                SolverFlag = ROM % ROM will be used in the next interval
            end
        end
    end
end

Output: POD modes  $\mathbf{U}_r$  and  $\mathbf{U}_r^H$  and ROM-DEIM Model

```

4.1.2 THE CRITERIA FOR MODEL SWITCH

In this section, the quantitative criteria used in Section 4.1.1 to decide the model switch are presented, including $\mathbf{C}_{F2R}$ (from FOM to ROM), $\mathbf{C}_{R2F}$ (from ROM to FOM), $\mathbf{C}_{R2D}$ (from ROM to ROM-DEIM) and $\mathbf{C}_{D2R}$ (from ROM-DEIM to ROM). Note that the model switch between ROM-DEIM and FOM can be determined by evaluating these criteria in combination. The criteria are designed based on the singular values corresponding to the POD modes $\mathbf{U}_r$ or $\mathbf{U}_r^H$ [85, 10], and the trajectory of the ROM solution and the nonlinear term in the subspace $\mathbf{U}_r$. The details of the criteria are given below:

1) Switch of FOM→ROM is decided by evaluating the criterion $\mathbf{C}_{F2R}$ which is based on the singular values $\sigma_1, \dots, \sigma_j, \dots, \sigma_N$ that indicates importance of the POD basis vectors of the snapshot data in the underlying subspace [85]. It consists of two sub-criteria $\mathbf{C}_{F2R,1}$ and $\mathbf{C}_{F2R,2}$

$$\mathbf{C}_{F2R,1} = \left(\frac{\sum_{j=r_d+1}^N \sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \right)^{\frac{1}{2}} \leq \epsilon_1, \quad \mathbf{C}_{F2R,2} = \left(\frac{\sum_{j=r_d+r_e+1}^N \sigma_j^2}{\sum_{i=1}^N \sigma_i^2} \right)^{\frac{1}{2}} \leq \epsilon_2, \quad (4.1)$$

where r_d is the number of basic POD modes retained and r_e is the number of extended modes, and that $r = r_d + r_e$. ϵ_1 and ϵ_2 are the tolerance parameters for the two sub-criteria. These criteria are discussed in Chapter.2. If both sub-criteria are satisfied, the current POD basis vectors accurately represent the subspace of the FOM solution, the local ROM will be built and simulation will switch to ROM in the next interval. Due to the fact that POD modes from $r_d + r_e + 1$ to N is a subset of those from $r_d + 1$ to N , $\epsilon_2 < \epsilon_1$ is always valid. In addition, $r = r_d + r_e$ is the dimension of the ROM in simulation.

2) Switch from ROM→FOM is determined by evaluating the criterion $\mathbf{C}_{R2F}$ at the end of each ROM interval or ROM-DEIM interval:

$$\mathbf{C}_{R2F} = \left(\frac{\sum_{j=r_d+1}^{r_d+r_e} \mathbf{Y}_{r,j}^2(i\Delta t)}{\sum_{j=1}^{r_d+r_e} \mathbf{Y}_{r,j}^2(i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon, \quad (4.2)$$

where $\mathbf{Y}_{r,j}(i\Delta t)$ represents the coordinates of the j^{th} mode of the ROM solution vector at the end of the i^{th} ROM interval. This criterion is also discussed in Chapter.2. A very low value of $\mathbf{C}_{R2F}$ indicates excellent representation of the POD subspace, and the ROM trajectories align well with the subspace. However, when $\mathbf{C}_{R2F}$ exceeds the tolerance parameter ϵ , it means that the existing POD subspace cannot capture the trajectory of ROM very well, and the simulation needs to switch back to FOM [75, 10].

3) Switch of ROM→ROM-DEIM is determined by the criterion $\mathbf{C}_{R2D}$, which examines the singular values of the POD modes of nonlinear terms $\mathbf{U}_r^H$. Similar to $\mathbf{C}_{F2R}$, two sub-criteria $\mathbf{C}_{R2D,1}$ and $\mathbf{C}_{R2D,2}$ are defined:

$$\mathbf{C}_{R2D,1} = \left(\frac{\sum_{j=m_H+1}^N (\sigma_j^H)^2}{\sum_{i=1}^N (\sigma_i^H)^2} \right)^{\frac{1}{2}} \leq \epsilon_1^H, \quad \mathbf{C}_{R2D,2} = \left(\frac{\sum_{j=m_H+r_e+1}^N (\sigma_j^H)^2}{\sum_{i=1}^N (\sigma_i^H)^2} \right)^{\frac{1}{2}} \leq \epsilon_2^H. \quad (4.3)$$

where ϵ_1^H and ϵ_2^H are the two tolerance parameters, and H denotes the quantities associated with nonlinear terms. The switch is based on importance of the POD basis vectors in the snapshot data of nonlinear terms, which is determined by the singular values $\sigma_1^H, \dots, \sigma_j^H, \dots, \sigma_N^H$ corresponding to the basis vectors. Similar to the above, r_e extended modes are introduced as the probe to inspect the change rate of the cumulative energy ratio, however, m_H denotes the DEIM number of the nonlinear terms. If sub-criteria are satisfied, the POD basis vectors $\mathbf{U}_r^H$ represent well the subspace of the nonlinear terms and are suitable for DEIM and nonlinear term interpolation. Thus, simulation will switch from ROM to ROM-DEIM in the next interval.

4) Switch of ROM-DEIM→ROM will be decided by examining two criteria $\mathbf{C}_{R2F}$ and $\mathbf{C}_{D2R}$ (i.e., logic thread “c” in Fig. 4.2) at the end of each local ROM-DEIM

interval. Since ROM-DEIM is a ROM with nonlinear term interpolation and approximation, the first evaluation of $\mathbf{C}_{R2F}$ will inspect if its trajectory in the reduced domain is still well contained within the subspace $\mathbf{U}_r$ (similar to above). If it is not, the simulation will immediately switch to FOM. Otherwise, the second criterion $\mathbf{C}_{D2R}$ will be evaluated, which is given by:

$$\mathbf{C}_{D2R} = \left(\frac{\sum_{j=r_d+1}^{r_d+r_e} \hat{\mathbf{H}}_{r,j}^2(i\Delta t)}{\sum_{j=1}^{r_d+r_e} \hat{\mathbf{H}}_{r,j}^2(i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon^H, \quad \hat{\mathbf{H}}_r = \mathbf{U}_r^T \hat{\mathbf{H}} \quad (4.4)$$

where $\hat{\mathbf{H}}$ is the nonlinear term interpolated through DEIM, $\hat{\mathbf{H}}_r$ represents its projection onto the subspace $\mathbf{U}_r$, and the specific form of $\hat{\mathbf{H}}$ will be given below. Subscript j is the j^{th} element of $\hat{\mathbf{H}}_r$, i is the i^{th} interval, and ϵ^H is the tolerance parameter. Eq. 4.4 essentially evaluates the relative ratio (and significance) of the modal coordinate of the extended modes of $\hat{\mathbf{H}}$ (from $r_d + 1$ to $r_d + r_e$) relative to all $r = r_d + r_e$ modes. A high value of $\mathbf{C}_{R2F}$ beyond the tolerance parameter ϵ^H signifies that the interpolated nonlinear term starts to deviate from the POD subspace for ROM computation, leading to large error. Therefore, DEIM should not remain in use in the next interval, instead, ROM should be considered.

5) The switch of FOM→ROM-DEIM will be determined by sequentially evaluating both criteria in (1) and (3), i.e., logic thread “a” in Fig. 4.2. If both are valid, the simulation will be directly switched from FOM to ROM-DEIM.

6) The switch of ROM-DEIM→FOM will be determined by evaluating $\mathbf{C}_{R2F}$ (“c-1” in Fig. 4.2). The argument is the same as that in 4). That is, ROM-DEIM is a ROM with DEIM approximation, and its trajectory in the reduced domain $\mathbf{U}_r$ needs to be first examined to decide if the simulation should return to FOM immediately.

It should be noted that the criteria 3)–6) proposed above are the most important novelty of the present research, which allows to combine the models with hierarchical approximation and fidelity into the hybrid snapshot simulation.

4.1.3 POD MODE CREATION(SVD) AND UPDATING(iSVD)

The POD modes $\mathbf{U}_r$ and $\mathbf{U}_r^H$ are first generated by applying SVD, respectively, to the snapshot data of the solution variable and the nonlinear terms at the end of the first FOM interval, and then updated by incremental SVD (iSVD) in the subsequent intervals when new snapshot data are produced. It should be emphasized that $\mathbf{U}_r$ will be updated only at the end of the FOM interval, but not at the end of any ROM or ROM-DEIM interval. This is because the ROM trajectory is within the given subspace $\mathbf{U}_r$, and ROM simulation does not contribute to any new information to the subspace $\mathbf{U}_r$. On the other hand, $\mathbf{U}_r^H$ will be updated with new snapshot data of nonlinear terms obtained in both the FOM and ROM intervals, but not in any ROM-DEIM interval for the same reason. The ultimate POD modes $\mathbf{U}_r$ and $\mathbf{U}_r^H$ generated at the end of the snapshot simulation are then utilized for constructing ROM-DEIM for the online simulation.

Similar to previous chapters $\Psi \approx \mathbf{U}\Sigma\mathbf{V}^T$. Once the new snapshot data Ψ_b becomes available, iSVD [17] based on a computationally efficient low-rank SVD procedure will be used to update the POD basis vector $\mathbf{U}$:

$$\begin{bmatrix} \Psi & \Psi_b \end{bmatrix} \approx \begin{bmatrix} \mathbf{U}\Sigma\mathbf{V}^T & \Psi_b \end{bmatrix} \approx \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^T \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T, \quad (4.5)$$

then the updated the POD modes are $\mathbf{U} \leftarrow \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}$, $\Sigma \leftarrow \tilde{\Sigma}$, and $\mathbf{V} \leftarrow \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \tilde{\mathbf{V}}$. Apart from the updates of POD modes, the iSVD procedure above is also applied to update the POD modes of the nonlinear terms $\mathbf{U}_r^H$ with the following difference. The snapshot data of nonlinear terms Ψ^H will be used, which can be generated from two sources: the solutions in the full domain directly computed by FOM or reconstructed from the ROM solution. Therefore, iSVD to update $\mathbf{U}_r^H$ will be performed at the end of both the FOM and the ROM interval. In addition, the order of truncation for $\mathbf{U}_r^H$,

which is the number of the DEIM interpolation points/indices, will also be different from that of $\mathbf{U}_r$.

4.2 CONSTRUCTIONS OF FOM, ROM AND ROM-DEIM

In this chapter a transport-related nonlinear numerical model is obtained based on the cell-centered finite volume method (FVM) according to the previous chapters. It includes two kinds of nonlinearities: the inter-cell transport nonlinear term arising from the transport flux between computational cells, when the material property of the transport depends on the solution variable (e.g., temperature-dependent thermal conductivity and concentration-dependent diffusivity); and the cell-wise nonlinear term that involves nonlinear contributions within a computational cell (e.g., nonlinear boundary conditions and source terms).

4.2.1 FOM CONSTRUCTION

The FOM of the governing transport equations can be written in the form of an ordinary differential equation(ODE) set [10, 11, 12]:

$$\frac{d\mathbf{Y}}{dt} = \mathbf{H}(\mathbf{Y}), \quad (4.6)$$

where $\mathbf{Y}(t) \in \mathfrak{R}^n$ is the vector of the FOM solution variable with the dimension n , which depends on time t . n denotes the number of mesh cells in the discretized computational domain. $\mathbf{H} \in \mathfrak{R}^n$ is also a vector arising from the discretization of the spatial differential terms and any others that can be separate from the temporal differential in the original PDE. As discussed above, since two types of nonlinearities are taken into account, $\mathbf{H}(\mathbf{Y})$ in Eq 4.6 can be separated into the transport-related nonlinear term and the cell-wise nonlinear term, both of which are also functions of $\mathbf{Y}$ and denoted, respectively, by $\mathbf{F}(\mathbf{Y}) \in \mathfrak{R}^n$ (transport-related) and $\mathbf{G}(\mathbf{Y}) \in \mathfrak{R}^n$ (cell-wise). Thus, Eq. 4.6 can be rewritten as:

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(\mathbf{Y}) + \mathbf{G}(\mathbf{Y}). \quad (4.7)$$

Note that Eq. 4.7 is also applicable to the scenarios when the linear and/or the constant terms are present. Since their counterparts in the reduced domain can be pre-computed in the ROM, they can be added as an additional term in Eq. 4.7 or included into $\mathbf{F}$ or $\mathbf{G}$ without affecting the generality of the method.

4.2.2 ROM CONSTRUCTION

In this section, we will describe the process of ROM construction that reduces the computational dimension of FOM using the subspace projection. In other words, FOM is projected onto the POD basis vectors $\mathbf{U}_r \in \mathbb{R}^{n \times r}$ obtained above, yielding a ROM with a low dimension r , that is, $\mathbf{Y} \approx \mathbf{U}_r \mathbf{Y}_r$, where $\mathbf{Y}_r \in \mathbb{R}^r$ represents the solution variables in the reduced domain. By substituting $\mathbf{Y} \approx \mathbf{U}_r \mathbf{Y}_r$ into Eq. 4.7, left-multiplying $\mathbf{U}_r$ on both sides, and applying orthogonality of $\mathbf{U}_r$, ROM can be obtained:

$$\frac{d\mathbf{Y}_r}{dt} = \mathbf{U}_r^T \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r) + \mathbf{U}_r^T \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r), \quad (4.8)$$

4.2.3 ROM-DEIM CONSTRUCTION

Discrete empirical interpolation method (DEIM) is an efficient approach to approximate nonlinear terms in ROMs through interpolation of the entries at selected spatial locations to reduce the cost of assembling nonlinear terms [27, 29]. There are three steps to implement DEIM in our hybrid snapshot simulation. First the POD basis vectors for the snapshot data of the nonlinear terms are computed by SVD or iSVD at the end of each FOM or ROM interval. Second, grid points/cells (i.e., their indices) are selected at the spatial locations, where the residual of reconstructing the next POD basis vector is largest. Last, the entries of the nonlinear terms only at the selected grid points/cells are assembled, which are then used to interpolate the rest

in the full domain within the POD subspace identified in the first step. Accordingly, the ROM-DEIM generation requires the POD modes of both the solution variables $\mathbf{U}_r$ and the nonlinear terms $\mathbf{U}_r^H$, which are acquired and updated in the preceding FOM and ROM intervals. Since two types of nonlinear terms, the transport-related $\mathbf{F}(\mathbf{Y})$ and the cell-wise $\mathbf{G}(\mathbf{Y})$ are considered separately, their POD modes are also different, which are denoted by $\mathbf{U}_r^F \in \mathbb{R}^{n \times m_F}$ and $\mathbf{U}_r^G \in \mathbb{R}^{n \times m_G}$, respectively, where $m_F \ll n$ and $m_G \ll n$ are the number of POD modes retained for the two nonlinear terms, and also the number of the DEIM interpolation points/cells for them.

In DEIM, an interpolation matrix $\mathbf{P} \in \mathbb{R}^{n \times m}$ is defined to store m interpolation indices (i.e., grid points/cells) selected out of from the n spatial locations in the FOM, i.e.,

$$\mathbf{P} = [\mathbf{e}_{\varphi_1}, \mathbf{e}_{\varphi_2}, \dots, \mathbf{e}_{\varphi_m}], \quad (4.9)$$

where $\mathbf{e}_{\varphi_i} = [0, \dots, 0, 1, 0, \dots, 0]^T$ represents the φ_i^{th} column of the n dimensional identity matrix $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ for $i = 1, \dots, m$. Again because of the presence of two types of nonlinear terms $\mathbf{F}(\mathbf{Y})$ and $\mathbf{G}(\mathbf{Y})$, there are two corresponding interpolation matrices $\mathbf{P}_F \in \mathbb{R}^{n \times m_F}$ and $\mathbf{P}_G \in \mathbb{R}^{n \times m_G}$. $\mathbf{P}_F$ and $\mathbf{P}_G$ are determined by the DEIM algorithm [27, 28].

It should be emphasized that $\mathbf{F}(\mathbf{Y})$ originates from the transport fluxes at the cell interfaces that depend on the solution values at the present cell and its neighboring cells. Therefore, in addition to the DEIM cells $\mathbf{P}_F$, the indices of their surrounding cells also need to be extracted in order to assemble $\mathbf{F}(\mathbf{Y})$ that conserves transport quantities within the DEIM cell. That is, for each iteration of simulation, solution values at both the DEIM cells and their surrounding cells will be used to update the transport fluxes at their interfaces to assemble $\mathbf{F}(\mathbf{Y})$ at DEIM points/cells. The detailed procedure to develop DEIM in the FVM formulation with nonlinear transport is presented elsewhere by the authors [11]. Thus, the nonlinear terms $\mathbf{F}(\mathbf{Y}(t))$ and

$\mathbf{G}(\mathbf{Y}(t))$ can be approximated in the ROM-DEIM, i.e.,

$$\begin{aligned}\mathbf{F}(\mathbf{Y}(t)) &\approx \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t)) \approx \mathbf{U}_r^F \left(\mathbf{P}_F^T \mathbf{U}_r^F \right)^{-1} \mathbf{P}_F^T \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t)), \\ \mathbf{G}(\mathbf{Y}(t)) &\approx \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t)) \approx \mathbf{U}_r^G \left(\mathbf{P}_G^T \mathbf{U}_r^G \right)^{-1} \mathbf{P}_G^T \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t)).\end{aligned}\tag{4.10}$$

Eq. 4.10 clearly reveals the two approximations used in the ROM-DEIM. The first " $\approx$ " in each of Eq. 4.10 is a result of reconstructing the full domain solution $\mathbf{Y}$ using the POD subspace $\mathbf{U}_r$ and the ROM solution $\mathbf{Y}_r$, and the second " $\approx$ " is attributed to DEIM approximation. Multiplying $\mathbf{P}_F$ and $\mathbf{P}_G$ on the left side of $\mathbf{F}(\mathbf{U}_r \mathbf{Y}_r(t))$ and $\mathbf{G}(\mathbf{U}_r \mathbf{Y}_r(t))$, means that only m_F rows of $\mathbf{F}$ and m_G rows of $\mathbf{G}$ need to be assembled, respectively, during ROM-DEIM simulation, leading to computational acceleration. Additional computational efficiency can be gained through pre-computation of the matrices that remain unchanged during ROM-DEIM simulation, such as

$$\mathbf{C}_F = \mathbf{U}_r^F \left(\mathbf{P}_F^T \mathbf{U}_r^F \right)^{-1}, \quad \mathbf{C}_G = \mathbf{U}_r^G \left(\mathbf{P}_G^T \mathbf{U}_r^G \right)^{-1}.\tag{4.11}$$

$\mathbf{C}_F$ and $\mathbf{C}_G$ in Eq. 4.11 represents combined matrix operations of the DEIM interpolation and POD reconstruction from the selected rows of $\mathbf{F}$ and $\mathbf{G}$. Substituting Eqs. 4.10 and 4.11 into Eq. 4.8, the whole ROM-DEIM with separate inter-cell transport and cell-wise nonlinear terms is given by:

$$\frac{d\mathbf{Y}_r}{dt} = \mathbf{U}_r^T \mathbf{C}_F \mathbf{P}_F^T \mathbf{F}(\mathbf{U}_r \mathbf{Y}_r) + \mathbf{U}_r^T \mathbf{C}_G \mathbf{P}_G^T \mathbf{G}(\mathbf{U}_r \mathbf{Y}_r).\tag{4.12}$$

It should be noted that the POD mode $\mathbf{U}_r$ used to project the entire FOM and POD modes $\mathbf{U}_r^F$ and $\mathbf{U}_r^G$ for DEIM will be created and updated at the end of the FOM interval. However, updating of POD modes $\mathbf{U}_r^F$ and $\mathbf{U}_r^G$ could also be performed at the end of the ROM intervals using the snapshot data of the nonlinear terms reconstructed in the full domain which will be further elucidated in the next section.

4.2.4 ENRICHMENT OF SNAPSHOT DATA FOR NONLINEAR TERMS

Traditionally, the snapshot data for the nonlinear terms is obtained by simulating FOMs only, which however may not be well-suited for our hybrid snapshot simulation

for several reasons: first, as show in Chapter.2 and Chapter.3 [10, 11], FOM is only used in a small fraction of intervals, which may not generate adequate snapshot data to represent POD subspace of the nonlinear terms; second, DEIM interpolates the entries at selected grid points/cells in the spatial domain. The number of the entire computational mesh is normally much larger than the dimension of the ROM, and the solution data only collected in the FOM interval may be insufficient for generating DEIM interpolation points/cells. Therefore, in this chapter we enrich the snapshot data of the nonlinear terms by utilizing the solution in the ROM interval as well. However, all the solution data obtained in the ROM interval are in the reduced domain with the low dimension r , i.e., $\mathbf{Y}_r \in \mathfrak{R}^r$, which needs to be reconstructed into the full dimension first using $\hat{\mathbf{Y}} = \mathbf{U}_r \mathbf{Y}_r$, where $\hat{\mathbf{Y}} \in \mathfrak{R}^n$ is the reconstructed solution in the full domain. Then the snapshot data of nonlinear terms, i.e., $\hat{\Psi}_F$ and $\hat{\Psi}_G$ could be acquired by running the nonlinear operators $\mathbf{F}$ and $\mathbf{G}$ through $\hat{\mathbf{Y}}$. The idea is based on the fact that the spanning basis vectors of the nonlinear terms are different from the solution variables of ROM. Then the POD modes, $\mathbf{U}_r^F$ and $\mathbf{U}_r^G$ of the nonlinear terms will be updated using iSVD and truncation following the procedure outlined in Section 2.3 at the end of both the FOM and ROM intervals. The details for enriching snapshot data of the nonlinear terms are given in Chapter.3 [11].

4.3 NUMERICAL MODEL OF HEAT TRANSFER

A numerical heat transfer model discretized by FVM to study, evaluate, and demonstrate the DEIM-embedded hybrid snapshot simulation according to the previous chapters. For the sake of completeness, the corresponding FOM, ROM, and ROM-DEIM will be briefly presented below following the generalized formulation in Eqs. 4.7, 4.8, and 4.12, and more detailed description can be found in Chapter.3 [11].

With the FVM discretization, the ODE form of the numerical heat transfer model [93, 10, 11] is given by:

$$\mathbf{M}(\mathbf{T}(t)) \frac{d\mathbf{T}(t)}{dt} = \tilde{\mathbf{A}}_n(\mathbf{T}(t)) \mathbf{T}(t) + \tilde{\mathbf{R}}_s \left(\mathbf{T}_{\text{inf},R}^{*4} - \mathbf{T}^{*4}(t) \right) + \tilde{\mathbf{B}}_c + \tilde{\mathbf{B}}\mathbf{u}, \quad (4.13)$$

where $\mathbf{T}(t) \in \mathbb{R}^n$ denotes the vector of the temperature solution in the full spatial domain; the superscript $*$ denotes the element-wise operation; $\mathbf{M}(\mathbf{T}(t)) \in \mathbb{R}^{n \times n}$ is a diagonal matrix with nonzero diagonal elements representing the thermal capacitance/mass of all computational cells, which is temperature dependent in the present work; $\tilde{\mathbf{A}}_n(\mathbf{T}(t)) \in \mathbb{R}^{n \times n}$ is a sparse matrix representing the inter-cell thermal conductance among cells and conductance between the cell center and the boundary face at the isothermal boundaries; $\tilde{\mathbf{R}}_s \in \mathbb{R}^{n \times n}$ is also a diagonal matrix describing the external radiation at the boundaries, and the diagonal elements are the external radiative heat transfer links. Only the position in the matrix corresponding to the cells adjacent to the radiative boundaries will have non-zero values; $\mathbf{T}_{\text{inf},R} \in \mathbb{R}^n$ denotes the environmental temperature values at the far field for the external radiation; $\tilde{\mathbf{B}}_c \in \mathbb{R}^n$ is the constant source term; $\tilde{\mathbf{B}} \in \mathbb{R}^{n \times d}$ is the input matrix that distributes the effects of the input $\mathbf{u}$ into each computational cells, and d is the dimension of inputs. In the present work, the heat flux applied at the external boundaries of the numerical model is treated as the input $\mathbf{u}$.

After being normalized by the thermal capacity matrix, it is given by:

$$\frac{d\mathbf{T}(t)}{dt} = \mathbf{A}\mathbf{T}(t) + \mathbf{R}\mathbf{T}(t)^{*4} + \mathbf{D} + \mathbf{B}\mathbf{u}. \quad (4.14)$$

When the simulation switches to ROM, the POD basis vectors $\mathbf{U}_r$ generated/updated in the previous FOM interval are utilized for subspace projection, i.e., $\mathbf{T}(t) \approx \mathbf{U}_r \mathbf{T}_r(t)$, yielding

$$\frac{d\mathbf{T}_r(t)}{dt} = \mathbf{A}_r \mathbf{T}_r(t) + \mathbf{R}_r (\mathbf{U}_r \mathbf{T}_r(t))^{*4} + \mathbf{D}_r + \mathbf{B}_r \mathbf{u}. \quad (4.15)$$

It should be pointed out that the introduction of the temperature-dependent thermal capacitance $\mathbf{M}$ gives rise to the nonlinearity of all terms above, which is different

from the model used in Chapter.2 and Chapter.3 [10, 11]. In other words, during ROM simulation all the terms on the RHS of Eq. 4.14 need to be assembled and updated first and then projected onto the POD subspace to obtain ROM in Eq. 4.15. This will be carried out on all computational grid points/cells in each time step and iteration, resulting in significant computational costs.

Therefore, the ROM-DEIM is needed to further reduce the computational cost by interpolating the entries of the nonlinear terms at selected DEIM grid points/cells. In this chapter, DEIM will be performed on the four terms on RHS of Eq. 4.14 separately rather than their combinations for enhanced interpolation accuracy [28, 19]. Correspondingly, four sets of the POD modes of the nonlinear terms will be maintained and updated throughout the snapshot simulation, i.e., $\mathbf{U}_r^A$ for the inter-cell transport term, $\mathbf{U}_r^R$ for the external radiation term; $\mathbf{U}_r^D$ for the source term, and $\mathbf{U}_r^B$ for the input contribution. All of them are generated or updated by SVD or iSVD and then truncated at their individual DEIM numbers m_A , m_R , m_D , and m_B as discussed above. Following Section 3.3, the pre-computed matrices for DEIM interpolation are written as:

$$\mathbf{C}_i = \mathbf{U}_r^i \left(\mathbf{P}_i^T \mathbf{U}_r^i \right)^{-1} \quad (4.16)$$

where subscript i can be A , R , D , and B to represent the quantities for the corresponding nonlinear terms, and $\mathbf{P}_i$ is the interpolation index matrix. Thus, following Eq. 4.12 the ROM-DEIM of this numerical heat transfer model is given by:

$$\frac{d\mathbf{T}_r(t)}{dt} = \mathbf{Q}_A \mathbf{A}_P \mathbf{U}_r \mathbf{T}_r(t) + \mathbf{Q}_R \mathbf{R}_P \left(\tilde{\mathbf{U}}_r \mathbf{T}_r(t) \right)^{*4} + \mathbf{Q}_D \mathbf{D}_p + \mathbf{Q}_B \mathbf{B}_P \mathbf{u}, \quad (4.17)$$

where

$$\begin{aligned} \mathbf{Q}_A &= \mathbf{U}_r^T \mathbf{C}_A, \\ \mathbf{Q}_R &= \mathbf{U}_r^T \mathbf{C}_R, \\ \mathbf{Q}_D &= \mathbf{U}_r^T \mathbf{C}_D, \\ \mathbf{Q}_B &= \mathbf{U}_r^T \mathbf{C}_B \end{aligned} \quad (4.18)$$

are the combined operation of ROM projection and DEIM interpolation and can be pre-computed. The matrices in assembly and updating could be in row-selection:

$$\begin{aligned}
\mathbf{A}_P &= \mathbf{P}_A^T \mathbf{A}, \\
\mathbf{R}_P &= \mathbf{P}_R^T \mathbf{R} \mathbf{P}_R, \\
\mathbf{D}_P &= \mathbf{P}_D^T \mathbf{D}, \\
\mathbf{B}_P &= \mathbf{P}_B^T \mathbf{B}
\end{aligned} \tag{4.19}$$

are the operations that only assemble selected rows (corresponding to the DEIM indices) of the $\mathbf{A}$, $\mathbf{R}$, $\mathbf{D}$ and $\mathbf{B}$ matrices. It is noticed that in the radiation term $*$ is the element-wise operation and $\mathbf{R}$ is a diagonal matrix, which means that the term $\mathbf{P}_R^T \mathbf{R} (\mathbf{U}_r \mathbf{T}_r(t))^*4$ is equivalent to $\mathbf{P}_R^T \mathbf{R} \mathbf{P}_R (\mathbf{P}_R^T \mathbf{U}_r \mathbf{T}_r(t))^*4$. The latter is more computationally efficient, as only m_R columns of $\mathbf{R}$ and m_R rows of $\mathbf{U}_r$ will be used in their multiplication. Therefore,

$$\tilde{\mathbf{U}}_r = \mathbf{P}_R^T \mathbf{U}_r \tag{4.20}$$

is pre-computed before the ROM-DEIM interval. Again it should be noted that in Eq. 4.17, DEIM is also applied to the source term $\mathbf{D}$ because of the temperature-dependent thermal capacitance $\mathbf{M}$, which is different from our previous model [11].

4.4 NUMERICAL CONFIGURATIONS, RESULTS AND DISCUSSIONS

Case studies based on two numerical models of nonlinear heat transfer are performed to verify the DEIM-embedded hybrid snapshot simulation approach.

4.4.1 MODEL CONFIGURATIONS

Table 4.1 **Thermal properties of the constituent materials with constant values(SS: Stainless Steel; MW: Mineral Wool)**

Materials	Thermal Conductivity (W/m K)	Density (kg/m3)	Specific Heat (J/kg K)
SS	14.89	8020	490
MW	0.043	150	840

Figure 4.3 The configurations of multi-layer annular cone model: (a) Geometry; (b) The heat flux profile prescribed at the three quarters of the outer wall of the computational model and used in both the snapshot simulation (in red) and the online simulation (in green).

Table 4.2 Thermal properties of the constituent materials with temperature-dependent properties(SS: Stainless Steel; MW: Mineral Wool)

Materials	Thermal Conductivity (W/m K)	Density (kg/m ³)	Specific Heat (J/kg K)
SS	$10.33 + 0.0154 \times T$ $-0.7 \times 10^{-6} \times T^2$	8020	$443 + 0.2 \times T$ $-8 \times 10^{-7} \times T^2$
MW	$0.05524 - 1.328 \times 10^{-4} \times T$ $+3.143 \times 10^{-7} \times T^2$	150	840

Similar to Chapter. 3, the first benchmark model is an annular cone (Fig. 4.3a) in which heat is transferred from the heated boundaries to all directions by conduction. The time-dependent profiles of the heat flux in both of the snapshot simulation and the online simulation are different as shown in Fig. 4.3b. In addition, the model is simulated with both the constant material properties and the temperature-dependent material properties to manifest the effect of the transport-related nonlinearity on the temperature solution [11]. The material properties for both scenarios are listed in Table. 4.1 and Table. 5.1, respectively.

The second benchmark model is an extended fin array (Fig. 4.4a). The prescribed heat flux profiles of the snapshot simulation and the online simulation are illustrated

Figure 4.4 The configurations of extended fins model: (a) Geometry; (b) The heat flux profiles are prescribed at the bottom of the substrate in the computational model and used in both the snapshot simulation (in red) and the online simulation (in green).

in Fig. 4.4b. Likewise, to evaluate their nonlinear effects on the temperature solution, the constant material properties and the temperature-dependent properties of the model are given in Table. 4.1 and Table. 5.1.

4.4.2 RESULTS AND DISCUSSIONS

In this section, numerical results for both the snapshot simulation and the online simulation are presented and discussed. Again, in the former, hybrids snapshot data is generated by alternating simulation among FOM, ROM, and ROM-DEIM, and POD modes are computed for solution variables and the nonlinear terms for ROM-DEIM. In the online simulation, only the generated ROM-DEIM is used to accelerate the computational speed. To facilitate discussion below, several abbreviations are defined: **SS-Hybrid** denotes the hybrid snapshot simulation that only involves the model switch between FOM and ROM in Chapter.2 [10]; **SS-Hybrid-DEIM** denotes the hybrid snapshot simulation proposed in this chapter, including FOM, ROM, and ROM-DEIM. The traditional snapshot simulation based on FOM only is termed as **SS-FOM**. **FROM, HROM**, and **HROM-DEIM** denote the ROMs used in the online simulation that are generated by performing **SS-FOM**, **SS-Hybrid**, and **SS-**

Hybrid-DEIM, respectively. The numerical solutions directly produced by the full order model is defined as **FOM**. To quantitatively examine the errors in various ROMs, two metrics are defined, the absolute error $\mathbf{Err}_{abs}$ and the mean square relative difference(MSRD) error $\mathbf{Err}_{msrd}$:

$$\mathbf{Err}_{abs}(t) = | \mathbf{U}_r \mathbf{T}_r(t) - \mathbf{T}(t) |, \quad (4.21)$$

$$\mathbf{Err}_{msrd}(t) = \sqrt{\frac{\sum_{j=1}^n \left(\frac{(\mathbf{U}_r \mathbf{T}_r(t))_j - \mathbf{T}_j(t)}{\mathbf{T}_j(t)} \right)^2}{n}} \times 100\%, \quad (4.22)$$

where j is the cell index; and both errors are time dependent.

CASE STUDY 1: THE ANNULAR CONE MODEL

In the numerical study of the cone model, the DEIM-embedded hybrid snapshot simulation is first conducted to generate the POD modes and construct HROM-DEIM rapidly, and HROM-DEIM is then used for the online simulation. The time span of the snapshot simulation is 480s, and divided in to 12 intervals, yielding the time span of $\Delta t = 40s$ for each interval. In the online simulation, the total time span is also 480s as shown in Fig. 4.3. The time step for both simulations is fixed at $\delta t = 0.25s$.

First the numerical results of SS-FOM produced using the constant properties and the temperature-dependent properties are compared to reveal the importance of the model nonlinearity caused by materials properties in our model. Fig. 4.5 illustrates both solutions and the absolute errors Err_{abs} between them. It is evident that the difference between them is significant.

Next the snapshot data generated by the hybrid approach with or without embedded DEIM is shown in Fig. 4.6. Specifically, Fig. 4.6(a) depicts the numerical solutions by SS-Hybrid and Fig. 4.6(b) the numerical solutions by SS-Hybrid-DEIM

Figure 4.5 The plots of FOM solutions produced using (a) the constant properties, (b) the temperature dependent properties and (c) the absolute error between (a) and (b) in the cone model.

Figure 4.6 The numerical solutions of the snapshot data for the annular cone model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with DEIM number 30, respectively.

with the DEIM number 30. Comparison of both plots indicates that snapshot solutions produced by both approaches are almost the same and the difference is almost negligible. The absolute errors Err_{abs} of both solutions in Fig. 4.6 are compared with SS-FOM, which is shown in Fig. 4.7. It is clear that the absolute errors of both SS-Hybrid and SS-Hybrid-DEIM are less than $10K$, which is very low considering the magnitude of the temperature solution that is close to $1,000K$ as shown in Fig. 4.6. The numerical results confirm the accuracy achieved by SS-Hybrid-DEIM for snapshot data generation.

Figure 4.7 The absolute errors of the snapshot data for the annular cone model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with the DEIM number 30, respectively.

Table. 5.2 lists the breakdown of the computational time for each individual interval of the proposed DEIM-embedded hybrid snapshot simulation with the DEIM number of 30. The parameters of tolerance for model type switches are $\epsilon_1 = 1e^{-5}$, $\epsilon_2 = 1e^{-6}$, $\epsilon = 1e^{-4}$, $\epsilon_1^F = 1e^{-3}$, $\epsilon_2^F = 1e^{-4}$, $\epsilon^F = 1e^{-2}$. For this annular cone model, 4 FOM intervals, 2 ROM intervals, and 6 ROM-DEIM intervals are used, which is determined on-the-fly by evaluating the switch criterion during snapshot simulation. It also shows that at the end of each FOM interval (i.e., the 1st, the 5th, the 8th, and the 10th interval), SVD/iSVD is applied to the snapshot data of the solution variable and the nonlinear terms to update both POD basis vectors $\mathbf{U}$ and $\mathbf{U}^H$. Again SVD is only adopted at the end of the first FOM interval to construct initial modes. However, at the end of each ROM interval (i.e, the 7th and the 12th interval), only POD modes $\mathbf{U}^H$ of the nonlinear terms are updated. In addition, the DEIM process is performed at the end of each FOM or ROM interval only when the model criteria (3) and (5) (in Section 2.2) are met and the snapshot simulation switches from FOM or ROM to ROM-DEIM. Table. 5.2 also indicates that the computational time of ROM-DEIM is greatly less than that of FOM and ROM, and it can be utilized to accelerate snapshot

data generation, although it also incurs additional costs of SVD/iSVD and DEIM computation. In summary, the total snapshot simulation time using the proposed DEIM-embedded approach is 112.432s. For comparison, two benchmark simulations, i.e., SS-Hybrid and SS-FOM are also conducted (results not shown in Table. 5.2). Their computational time is 125.56s and 246.574s, and the acceleration achieved by the proposed approach is 10.5% and 54.4%, respectively.

Table 4.3 The breakdown of the computational time for DEIM-embedded hybrid snapshot simulation with the DEIM number 30 in the annular cone model (Unit:s)

Interval	FOM	ROM	SVD/iSVD (U^H only)	ROM DEIM	DEIM	SVD/iSVD (U and U^H)
1 st	35.526	...	...	...	0.324	0.281
2 nd	...	...	...	1.252	...	...
3 th	...	...	...	1.246	...	...
4 th	...	...	...	1.123	...	...
5 th	18.482	...	...	...	0.269	0.969
6 th	...	...	...	1.268	...	...
7 th	...	4.474	0.669	...	...	...
8 th	18.478	...	...	...	0.259	0.824
9 th	...	...	...	1.175	...	...
10 th	18.492	...	...	...	0.251	0.830
11 th	...	...	...	1.156	...	...
12 th	...	4.413	0.671	...	...	...
Total	90.978	8.887	1.340	7.220	1.103	2.904

Next we examine the accuracy of HROM-DEIM in the online simulation that is generated by the proposed DEIM-embedded hybrid approach, viz., SS-Hybrid-DEIM. Fig. 4.8 illustrates the comparison of the results in the online simulation obtained by (a) HROM generated by SS-Hybrid but without DEIM, and (b-d) HROM-DEIM with three different DEIM numbers (30, 40, and 80). It shows that the numerical solutions produced by HROM-DEIM are nearly the same as those in HROM, and it seems that the DEIM number does not cause significant difference. Fig. 4.9 depicts the absolute errors Err_{abs} of HROM and HROM-DEIM related to the FOM solutions. Although the accuracy of HROM-DEIM is slightly worse than that of HROM, their absolute errors Err_{abs} are at the same level, both less than $20K$, which is very low considering

Figure 4.8 The numerical solutions of verification simulation for the annular cone model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)30, (c)40, and (d)80, respectively.

the large magnitude (up to $1,000K$) of the temperature solutions. Fig. 4.9(b-d) also reveals that by increasing the DEIM number, the values of Err_{abs} decreases, leading to more accurate solutions that are closer to HROM. However, it should be noticed that the increase of DEIM number gives rise to longer computational time of both the snapshot and online simulation.

The values of the mean square root difference error(MSRD) Err_{msrd} of all ROM solutions in Fig. 4.8 are shown in Fig. 4.10. A similar trend is observed that the magnitudes of Err_{msrd} seems to decrease as a result of the increasing DEIM

Figure 4.9 The absolute errors of the numerical solutions in the verification simulation for the annular cone model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)30, (c)40, and (d)80, respectively.

number, and the magnitude of Err_{msrd} in HROM without DEIM is the lowest in the comparison. The magnitude of Err_{msrd} does not exceed 1% even the DEIM number decrease to 30, which confirms that the DEIM approach for nonlinear ROM could improve the computational speed of ROM while preserving the accuracy.

Table 4.4 The comparison of computational time in **FOM**, **HROM**, and **HROM-DEIM** in the annular cone model. (Unit:s).

Model Type	FOM	HROM	HROM-DEIM		
			30	40	80
Time	270.625	53.312	13.014	13.763	14.137

Figure 4.10 The MSRD errors of the numerical solutions in the verification simulation for the annular cone model simulated by HROM (in black) and HROM-DEIM with three different DEIM numbers: 30 (in red), 40 (in green), and 80 (in blue), respectively.

Table. 5.3. lists the computational time of FOM, HROM, and HROM-DEIM in online simulation. It can be observed that ROMs could accelerate the online simulation greatly compared with FOMs. The computational time of HROM only is about 19.7% of the computational time in FOM. However, HROM-DEIM is able to achieve even higher computational acceleration than HROM due to the reduced cost of assembling the nonlinear terms: the ratio of the computational time of HROM-DEIM to HROM is 24.4% $\sim$ 26.5% and to FOM is only 4.81% $\sim$ 5.22%.

CASE STUDY 2: THE EXTENDED FIN MODEL

In the second case study of the extended fin model, the total time span of snapshot simulation is 1200s and is divided into 12 intervals, corresponding to $\Delta t = 100s$ for each time interval. The whole time span of online simulation is 960s. The time step for all the numerical computations is fixed as $\delta t = 0.5s$.

First, the effect of the nonlinearity caused by the temperature-dependent material properties is studied by comparing its numerical results of snapshot simulation against those using the constant material properties as illustrated in Fig. 4.11 (a) and (b).

Figure 4.11 The plots of FOM solutions computed using (a) the constant material properties; (b) the temperature-dependent properties; and (c) the absolute error between (a) and (b) in the extended fin array model.

Figure 4.12 The numerical solutions of the snapshot data for the extended fin array model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with the DEIM number 40, respectively.

In both simulations, FOMs are performed. The absolute errors Err_{abs} between them are shown in Fig. 4.11 (c). It is clearly observed that their difference is magnificent, and the nonlinearity of the material properties plays an important role, and needs to be effectively handled for the acquisition of accurate ROMs.

Fig. 4.12(a) illustrates the numerical solutions generated by SS-Hybrid without DEIM, while those by SS-Hybrid-DEIM are depicted in Fig. 4.12(b) with the DEIM number 40. It clearly shows that the snapshot data generated by both approaches are

Figure 4.13 The absolute errors of the snapshot data for the extended fin array model generated by (a) SS-Hybrid (involving FOM and ROM) and (b) SS-Hybrid-DEIM (involving FOM, ROM, ROM-DEIM) with the DEIM number 40, respectively.

almost the same. The absolute errors Err_{abs} corresponding to Fig. 4.12 are shown in Fig. 4.13. It could be seen that in both hybrid approaches the absolute errors Err_{abs} are lower than $15K$, which is very small relative to the magnitude of the solution in Fig. 4.12. This also confirms salient accuracy of SS-Hybrid-DEIM in snapshot simulation.

The breakdown of the computational time for each interval during the entire snapshot simulation is listed in Table. 4.5. The tolerance parameters for model switches are $\epsilon_1 = 1e^{-5}$, $\epsilon_2 = 1e^{-6}$, $\epsilon = 1e^{-4}$, $\epsilon_1^F = 7.5e^{-3}$, $\epsilon_2^F = 7.5e^{-4}$, $\epsilon^F = 7.5e^{-2}$. The snapshot simulation includes 12 time intervals, in which 2 are simulated by FOM (the 1st and the 8th), 2 by ROM (the 10th and the 12th), and 8 by ROM-DEIM (the 2nd – 7th, 9th, and 11th). As discussed above, the first interval is always simulated by FOM, followed by SVD to generate the initial POD modes $\mathbf{U}$ and $\mathbf{U}^H$ for the solution variable and the nonlinear terms, respectively. Then iSVD is applied at the end of the 8th FOM interval for updating both sets of POD modes. At the end of the two ROM intervals, iSVD is also utilized to only update the POD modes of the nonlinear terms $\mathbf{U}^H$. In addition, the DEIM process is performed at the end of each FOM or

ROM interval (i.e., the 1st, 8th, 10th, and 12th) when ROM-DEIM is adopted for the simulation in the next interval.

Table 4.5 The list of the computational time for DEIM-embedded hybrid snapshot simulation with the DEIM number 70 and the time step size $\delta t = 0.5s$ in the extended fin array model(Unit:s)

Interval	FOM	ROM	SVD/iSVD (U^H only)	ROM DEIM	DEIM	SVD/iSVD (U and U^H)
1 st	117.128	...	...	...	6.439	1.710
2 nd	...	...	...	6.152	...	...
3 th	...	...	...	5.813	...	...
4 th	...	...	...	5.953	...	...
5 th	...	...	...	5.779	...	...
6 th	...	...	...	5.795	...	...
7 th	...	...	...	8.421	...	...
8 th	69.759	...	...	...	6.582	4.993
9 th	...	...	...	5.785	...	...
10 th	...	25.613	3.678	...	6.597	...
11 th	...	...	...	5.796	...	...
12 th	...	25.306	3.703	...	...	...
Total	186.887	50.919	7.381	49.494	19.618	6.703

An interesting observation is that 8 ROM-DEIM intervals and 2 ROM intervals out of the 12 time intervals are accepted in this simulation to accelerate the snapshot data generation, which however, preserves the accuracy of solutions very well as indicated by the comparison in Fig. 4.13. It also reflects that the snapshot data may include a significant amount of redundant information in both the solution variables and the nonlinear terms, and the traditional FOM-based snapshot simulation is indeed not necessary for the type of problems under the present study. Since simulation of ROM-DEIM interval is much faster than FOM and even ROM, a salient reduction in computational time can be achieved by the proposed approach (i.e., SS-ROM-DEIM). Specifically, in this case study, its total computational time is only 321.002s, which is less than the hybrid approach without DEIM (i.e., SS-Hybrid) 444.885s and the traditional FOM approach SS-FOM, 889.367s, corresponding to 27.85% and 63.91% acceleration. This case study outperforms the first case study in the computational

Figure 4.14 The numerical solutions of verification simulation for the extended fin array model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)40, (c)70, and (d)150, respectively.

efficiency, which can be mainly attributed to more adoption of ROM and ROM-DEIM in the snapshot simulation.

Next the numerical solutions of HROM-DEIM generated by the proposed approach, viz., SS-Hybrid-DEIM for the online simulation are interrogated. Fig. 4.14 compares the results of (a) HROM generated by SS-Hybrid without DEIM with the results of (b-d) HROM-DEIM with three different DEIM numbers. It can be seen that the results of both are nearly the same, and the difference in the solutions caused by varying DEIM numbers is negligible. Correspondingly, the absolute errors Err_{abs} are

Figure 4.15 The absolute errors of the numerical solutions in the verification simulation for the extended fin array model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: (b)40, (c)70, and (d) 150, respectively.

presented in Fig. 4.15, which are computed as the difference between the solutions in Fig. 4.14 and the FOM solutions. Several interesting observations can be made: first, HROM-DEIM produces accurate results with $Err_{abs} < 10K$. All the errors initially increase and reach the highest value corresponding to the peak temperature in Fig. 4.14, and then start to drop and level off at a relatively low level; second, using HROM-DEIM will not appreciably raise the solution errors compared with HROM; third, Fig. 4.15(b-d) shows a general trend that the values of Err_{abs} decreases when a larger DEIM number is employed, leading to more accurate solutions. However, it should be noticed that the selection of the DEIM indices is based on their indi-

Figure 4.16 The MSRD errors of the numerical solutions in the verification simulation for the extended fin array model simulated by (a) HROM and (b-d) HROM-DEIM with three different DEIM numbers: 40 (in red), 70 (in green), and 150 (in blue), respectively.

vidual contributions to reconstructing the POD basis vectors [27, 29], and hence, the DEIM number is not directly indicative of ROM accuracy. Likewise, the DEIM number needs to be chosen with respect to a tradeoff between the accuracy and the computational efficiency.

In Fig. 4.16, the mean square root difference errors (MSRD) Err_{msrd} of the numerical solutions in HROM and HROM-DEIM are calculated. Err_{msrd} of HROM and HROM-DEIM follow the same trend as Err_{abs} , in which the values increase at the beginning, decrease after reaching the peak, and then level off. All Err_{msrd} are less than 1% during the whole online simulation. Similarly, a larger DEIM number could result in a lower value of Err_{msrd} , and makes the numerical solutions in HROM-DEIM closer to that in HROM.

Table 4.6 The comparison of computational time in **FOM**, **HROM**, and **HROM-DEIM** in the extended fin array model. (Unit:s).

Model Type	FOM	HROM	HROM-DEIM		
			40	70	150
Time	734.684	255.351	52.164	58.590	62.858

Table. 4.6. lists the computational time of FOM, HROM, and HROM-DEIM in the online simulation of the extend fin array model. It can be observed that compared with FOMs, HROM could accelerate the simulation greatly: the computational time of HROM is only about 34.76% of the computational time in FOM. However, HROM-DEIM is able to achieve even higher acceleration than HROM due to the reduced costs of assembling the nonlinear terms: the ratio of the computational time of HROM-DEIM to HROM is 20.43% $\sim$ 24.62% and to FOM is only 7.1% $\sim$ 8.56%.

4.5 SUMMARY OF CHAPTER.4

In this chapter, a DEIM-embedded hybrid snapshot approach is proposed for rapid and selective generation of high-quality snapshot data and ROM-DEIM. In contrast to the traditional FOM and the hybrid approach without DEIM [10], the new approach represents several novelties. First, the entire time span of the snapshot simulation is divided into multiple intervals, and each of them is simulated by the models in a hierarchy of approximations, fidelities, and computational costs, viz., FOM, ROM, or ROM-DEIM. Second, quantitative criteria that monitor the POD characteristics and trajectories are proposed to determine the model switch among FOM, ROM, and ROM-DEIM on-the-fly. Third, two sets of POD modes, respectively, for the solution variable and the nonlinear terms are generated and continuously updated using iSVD [17, 18] to minimize the computational cost during snapshot simulation. The updated POD modes are then used for rational decision of model switch and DEIM approximation. Thus, FOM only needs to be simulated at selected temporal intervals, where new subspace information most likely occurs. On the other hand, ROM and ROM-DEIM will be utilized to rapidly traverse intervals that contain least or no new information. Case studies of nonlinear numerical heat transfer model are then conducted using the proposed DEIM-embedded snapshot simulation method. Important technical findings include:

(1) It is confirmed that the proposed method allows not only rapid and selective snapshot data generation, but also salient data representation. The iSVD updates two sets of POD modes to improve the accuracy of ROMs and ROM-DEIMs.

(2) Compared with FOM and the hybrid approach without DEIM [10], the proposed method demonstrates notable improvement in computational efficiency: 54.4%–63.91% and 10.5% – 27.85% acceleration, respectively.

(3) In the online simulation, the computational time ratio of ROM-DEIM to FOM is 4.81%–5.22% and 7.1%–8.56%, respectively, in both case studies. Compared with HROM without DEIM, the ratio is, respectively, 24.4%–26.5% and 20.43%–24.62%. The MSRD error of ROM-DEIM relative to FOM is less than 1%, which can be further reduced to 0.2% by increasing the DEIM number.

(4) During the snapshot simulation, more than half of the intervals are simulated by ROM and ROM-DEIM, e.g., 2 and 6 out of 12 intervals in case study 1, and 2 and 8 out of 12 intervals in case study 2. This verifies the foundation of the proposed method, viz., the snapshot data produced by the traditional FOM is highly redundant. Embedding the local ROM and ROM-DEIM is proven effective to reduce redundancy and accelerate snapshot data generation while maintaining ROM accuracy.

CHAPTER 5

A REDUCED ORDER MODELING METHOD BASED ON GNAT-EMBEDDED HYBRID SNAPSHOT SIMULATION

In this chapter, we propose a computational method that embeds the GNAT hyper-reduction into the hybrid snapshot simulation developed in our prior chapters [10, 12]. In this method, three ROMs are used in the snapshot simulation, including the FOM, the ROM based on Petrov-Galerkin projection (ROM-PG), and the ROM based on hyper-reduction by Gauss-Newton with approximated tensors (ROM-GNAT) to form a hierarchy of model approximation. Then the snapshot simulation is partitioned into multiple intervals, and the simulation in each interval is switched among the FOM, ROM-PG, and ROM-GNAT to produce necessary, high-quality snapshot data in a more economic manner. In other words, the local ROM-PG and the local ROM-GNAT will be used to rapidly traverse regions that contribute to negligible new information to update the POD basis vectors of solution variables and nonlinear terms, leading to accelerated snapshot simulation. As shown later, the proposed method is compatible to fully discretized nonlinear PDE models [6, 7, 68, 105, 80].

The novelties in this chapter include: (1) to the best of our knowledge, combining the GNAT-ROM with the hybrid snapshot simulation to allow efficient generation of snapshot data using hierarchical model approximation (FOM, ROM-PG, and ROM-GNAT) is new and has not been reported before; (2) several new criteria that evaluate the POD basis vectors and the trajectories of the ROM-PG and ROM-GNAT are proposed to determine model switch on-the-fly during the snapshot simulation; (3) two

sets of POD modes used, respectively, for Petrov-Galerkin projection and GNAT hyper-reduction are updated by the iSVD during the snapshot simulation. The local POD modes are adopted to guide model switch; (4) the GNAT-embedded hybrid snapshot simulation is developed and demonstrated with the finite volume discretization of the 2D Burgers equation in the regime of weak gradient flow recently proposed by [82]; and (5) a ROM-GNAT will be generated at the end of the proposed GNAT-embedded snapshot simulation pipeline, and is immediately applicable for fast and accurate online simulations.

The organization of this chapter is organized as follows: in Section.5.1, we will provide an overview of Petrov-Galerkin projection and GNAT in nonlinear dynamic systems. The methodology of the GNAT-embedded hybrid snapshot simulation is described in Section.5.2. In Section.5.3 the FOM of the 2D Burgers equation is obtained by the finite volume discretization, and the nonlinear implicit Gauss-Newton solver is described. The ROM with Petrov-Galerkin projection (ROM-PG) and the ROM with Gauss-Newton approximated tensor(ROM-GNAT) are presented. In Section.5.4 the detailed information of the GNAT-embedded hybrid snapshot simulation for the 2D Burgers equation will be discussed, including the POD modes of the Petrov-Galerkin projection and the residuals, snapshot data acquisition, and the model switch criteria. In Section.5.5, the numerical experiments will be presented and discussed in terms of accuracy and computational efficiency. The chapter summaries with technical findings of the proposed method in Section.5.6.

5.1 PETROV-GALERKIN PROJECTION AND GAUSS-NEWTON APPROXIMATED TENSORS

In this section we introduce the approaches of Petrov-Galerkin projection and Gauss-Newton approximated tensor for generating ROMs of dynamic systems at the discrete time level. The approaches introduce hierarchical approximations when the PDE is

discretized in both time and space. First, the residual of the nonlinear system in a fully implicit framework is given by:

$$\mathbf{R}_Y^{n_t}(\mathbf{Y}^{n_t+1}) = 0, \quad (5.1)$$

where $n_t = 0, \dots, n_T - 1$, and n_T represents the number of time steps, and $\mathbf{Y} \in \mathbb{R}^n$ represents the variable of the full dimensional dynamic system with a dimension n . Then the dimension of the operator is $\mathbf{R}_Y \in \mathbb{R}^n$, which depends on the index of the time step n_t . Eq. 5.1 can be cast in a shorthand notation:

$$\mathbf{R}_Y(\mathbf{Y}) = 0. \quad (5.2)$$

5.1.1 PETROV-GALERKIN PROJECTION AND THE GAUSS-NEWTON SOLVER

The Petrov-Galerkin projection is employed for reducing the dimension of the residual equation of the nonlinear dynamic model, and the over-determined least squares system in the reduced domain is solved by the Gauss-Newton method. Using the POD basis vectors, this approach constructs the projection with iteration-dependent Jacobian by minimizing the L_2 norm of the residuals per Gauss-Newton iteration.

In order to reduce the dimension of Eq. 5.2, we first define the variable in the reduced domain through subspace projection:

$$\tilde{\mathbf{Y}} = \Phi_Y \mathbf{Y}_r, \quad (5.3)$$

where $\Phi_Y \in \mathbb{R}^{n \times r}$ is the projection matrix consisting of the basis vectors of the subspace; $\mathbf{Y}_r \in \mathbb{R}^r$ denotes the generalized coordinates (i.e., solution in the reduced domain); r denotes the dimension of the solution in the reduced domain with $r \ll n$.

The least squares problem in the over-determined system is:

$$\min_{\tilde{\mathbf{Y}} \in \Phi_Y \mathbf{Y}_r} \|\mathbf{R}_Y(\tilde{\mathbf{Y}})\|_2. \quad (5.4)$$

By using the test basis representing the actions of Jacobian on Φ_Y , i.e., $\mathbf{J}_Y^k \Phi_Y$

[23, 6, 7], the reduced Gauss-Newton iteration solver with Petrov-Galerkin projection is given by:

$$\Phi_Y^T (\mathbf{J}_Y^k)^T \mathbf{J}_Y^k \Phi_Y \delta \mathbf{Y}_r^k = -\Phi_Y^T (\mathbf{J}_Y^k)^T \mathbf{R}_Y^k, \quad (5.5a)$$

$$\mathbf{Y}_r^{k+1} = \mathbf{Y}_r^k + \delta \mathbf{Y}_r^k, \quad (5.5b)$$

where $\mathbf{J}_Y^k$ denotes the Jacobian of $\mathbf{R}_Y$ at the k^{th} iteration. It should be noted that the Jacobian changes during iterations, and thus, the actions of Jacobian on the projection subspace Φ_Y ($\mathcal{L}^k = \mathbf{J}_Y^k \Phi_Y$) is iteration-dependent, where $\mathcal{L}^k$ is also named column-reduced Jacobian in this chapter. The optimality of using Petrov-Galerkin projection approximation to compute the search direction to minimize the residual in Eq. 5.4 and Eq. 5.5 have been thoroughly discussed in [23, 25]. The iteration continues until the convergence condition is satisfied.

Although the nonlinear dynamic system is projected onto the subspace of low dimension by Petrov-Galerkin projection, the computational operations cannot be greatly reduced since $\mathbf{J}_Y^k \in \mathbb{R}^{n \times n}$ is iteration-dependent and must be re-computed in full dimension during iterations. In order to further reduce computational cost associated with the actions of the Jacobian on the right reduced-order basis vectors, i.e., $\mathbf{J}_Y^k \Phi_Y$, the system approximation based on the ‘‘gappy POD’’ method can be performed for hyper reduction.

5.1.2 THE GAPPY POD METHOD

To circumvent the large computational costs of updating $\mathbf{J}_Y^k \Phi_Y$ during iterations, the hyper reduction approach has been proposed to identify the subspace of $\mathbf{J}_Y^k \Phi_Y$ and approximate it using the gappy POD method [86, 45]. Essentially, this introduces the system approximation to the least squares problem above during the iterations of the reduced solutions [23, 105]. $\mathbf{R}_Y^k$ and the column-reduced Jacobian $\mathbf{J}_Y^k \Phi_Y$

could be approximated by $\widetilde{\mathbf{R}_Y^k}$ and $\widetilde{\mathbf{J}_Y^k \Phi_Y}$ with the corresponding POD basis vectors $\Phi_R \in \Re^{n \times n_R}$ and $\Phi_J \in \Re^{n \times n_J}$:

$$\widetilde{\mathbf{R}_Y^k} = \Phi_R \mathbf{R}_{Y_R}^k, \quad (5.6a)$$

$$\widetilde{\mathbf{J}_Y^k \Phi_Y} = \Phi_J \mathbf{J}_{Y_J}^k, \quad (5.6b)$$

where n_R and n_J are the numbers of the POD basis vectors kept to approximate $\widetilde{\mathbf{R}_Y^k}$ and $\widetilde{\mathbf{J}_Y^k \Phi_Y}$, respectively; $\mathbf{R}_{Y_R}^k \in \Re^{n_R}$ and $\mathbf{J}_{Y_J}^k \in \Re^{n_J \times r}$ are the POD modal coordinates of the residuals and column-reduced Jacobian in the subspace.

The gappy POD method can be treated as least squares regression or interpolation in discrete variables by empirical basis functions [23, 25]. Only a small number of selected rows in $\mathbf{R}_Y^k$ and $\mathbf{J}_Y^k \Phi_Y$ are assembled, followed by the least squares to determine the POD modal coordinates to reconstruct all the entries, i.e., $\widetilde{\mathbf{R}_Y^k}$ and $\widetilde{\mathbf{J}_Y^k \Phi_Y}$. The selected rows are described by a set of "sample indices", which form the sampling matrix [23, 25]:

$$\mathbf{P} = [e_{\varphi_1}, e_{\varphi_2}, \dots, e_{\varphi_{n_m}}] \in \Re^{n \times n_m}, \quad (5.7)$$

where $\mathbf{e}_{\varphi_i}$ represents the φ_i^{th} column of the sample matrix. The GNAT procedure approximates $\mathbf{R}_Y^k$ and $\mathbf{J}_Y^k \Phi_Y$ via gappy POD as:

$$\widetilde{\mathbf{R}_Y^k} \approx \Phi_R (\mathbf{P}^T \Phi_R)^+ \mathbf{P}^T \mathbf{R}_Y^k, \quad (5.8a)$$

$$\widetilde{\mathbf{J}_Y^k \Phi_Y} \approx \Phi_J (\mathbf{P}^T \Phi_J)^+ \mathbf{P}^T \mathbf{J}_Y^k \Phi_Y, \quad (5.8b)$$

where $+$ denotes the Moore-Penrose pseudo-inverse. $\mathbf{P}^T \mathbf{R}_Y^k$ and $\mathbf{P}^T \mathbf{J}_Y^k \Phi_Y$ in Eq. 5.8 clearly show that only n_m rows need to be assembled and computed per iteration. Then the search direction could be computed by solving the least squares problem:

$$\delta \mathbf{Y}_r^k = \arg \min_{\alpha \in \Re^r} \|\widetilde{\mathbf{J}_Y^k \Phi_Y} \alpha + \widetilde{\mathbf{R}_Y^k}\|_2, \quad (5.9a)$$

$$\mathbf{Y}_r^{k+1} = \mathbf{Y}_r^k + \delta \mathbf{Y}_r^k. \quad (5.9b)$$

Note that the least squares problem in Eq. 5.9 can be computed in a computationally efficient manner. This is because only n_R and n_J rows, respectively, in $\widetilde{\mathbf{R}}_Y^k$ and $\widetilde{\mathbf{J}}_Y^k \Phi_Y$ need to be assembled, and the Gauss-Newton iteration is performed in a reduced domain with r dimensions only.

It should be noticed that there are different options to select the snapshot data to extract the POD basis vectors Φ_R and Φ_J above. The sufficient conditions of the snapshot selection to meet different consistency requirements for the system approximation are discussed in details in [23]. The snapshot data collection in the present work is presented in Section.5.4.

5.1.3 MESH SAMPLING FOR GNAT IN FVM

As discussed above, without the hyper-reduction and system approximation, the least squares problem in Eq. 5.4 need to directly assemble the entire residual $\mathbf{R}_Y^k$ and the $\mathbf{J}_Y^k$ at all the mesh points during each iteration, leading to significant computational time and resource usage, which is the one of the foremost reasons why it is difficult for the nonlinear ROM to achieve the comparable computational efficiency as the linear ROM. The efficiency of Gauss-Newton iterations of the ROM could be improved by the gappy POD method, in which the model for the least squares solution in Eq. 5.9 only requires assembling a small number of selected rows in the residual $\mathbf{R}_Y^k$ and the column-reduced Jacobian $\mathbf{J}_Y^k \Phi_Y$. These rows correspond to sampled mesh points/indices in the numerical model, such as those discretized by the finite element method or finite volume method [24, 25]. Therefore, the sampled mesh points/indices should be those with strong representation of $\mathbf{R}_Y^k$ and $\mathbf{J}_Y^k \Phi_Y$ in gappy POD reconstruction.

For FVM-discretized PDEs (including the Godunov scheme used in this chapter), the calculation of transport flux at the cell faces also involves the values at the neighboring cells. Therefore, in order to assemble $\mathbf{R}_Y^k$ and $\mathbf{J}_Y^k \Phi_Y$ at the sample

indices, i.e., Eq.5.7, the solutions at their neighboring cells also need to be retrieved during each iteration. The detailed procedure to extract the solutions at both the sample indices and their neighboring cells is presented elsewhere [11].

5.2 METHODOLOGY OF GNAT-EMBEDDED HYBRID SNAPSHOT APPROACH

Following the brief introduction to the PG projection and the GNAT, in this section the GNAT-embedded hybrid snapshot approach is presented, which is proposed to accelerate snapshot data generation for constructing the efficient ROM-GNAT for online simulation. As shown in Fig. 5.1, in this method, the total time span of the snapshot simulation (labeled "1" in Fig. 5.1) is divided into multiple intervals, and each will be simulated by the FOM, ROM-PG, or ROM-GNAT. Recall that the ROM-PG refers to the ROM generated by the Petrov-Galerkin projection of the nonlinear FOM, while the ROM-GNAT is the ROM arising from both Petrov-Galerkin projection and hyper reduction of the nonlinear FOM. Since ROM-GNAT includes additional system approximation through the gappy POD process, it is usually less accurate, but much more computationally efficient relative to the ROM-PG [23, 6]. The computations performed in the FOM, ROM-PG and ROM-GNAT intervals are described below:

(I). In each FOM interval, the FOM will be performed to generate snapshot data Ψ of the incremental solution variable $\Delta \mathbf{Y} \in \mathbb{R}^n$, snapshot data Ψ_R of the residuals $\mathbf{R}_Y \in \mathbb{R}^n$ and snapshot data Ψ_J of the actions of Jacobian $\mathbf{J}_Y \in \mathbb{R}^{n \times n}$ on Φ_Y , where n denotes the full dimension and $\tilde{\mathbf{Y}} = \mathbf{Y}^0 + \Delta \mathbf{Y}$, and $\mathbf{Y}^0$ denotes the initial solution of the present time interval and also the solution at the end of the previous time interval, and hence, $\Delta \mathbf{Y}$ denotes the incremental solutions of $\mathbf{Y}$ relative to $\mathbf{Y}^0$. At the end of each FOM interval(labelled "2" in Fig. 5.1), the following computational procedures will be performed: (1) SVD or iSVD will be applied onto Ψ , Ψ_R and Ψ_J to generate or update the local POD basis vectors Φ_Y , Φ_R and Φ_J ; (2) both of

Figure 5.1 The methodology of GNAT-embedded hybrid snapshot simulation and the construction of ROM-GNAT for online simulation:(1) the procedure of GNAT-embedded hybrid snapshot simulation; (2) the SVD/iSVD updating of Φ_Y , Φ_R , Φ_J and the “greedy” algorithm at the end of each FOM interval(if the simulation will switch to ROM-GNAT); (3) the SVD/iSVD updating of Φ_R , Φ_J and the “greedy” algorithm at the end of each ROM-PG interval(if switch to ROM-GNAT); (4) no updating in POD basis after each ROM-GNAT interval; (5) the ROM-GNAT in online simulation is constructed by Φ_Y , Φ_R and Φ_J obtained in the GNAT-embedded hybrid snapshot simulation.

switch criteria, $\mathbf{C}_{F2R}$ and $\mathbf{C}_{R2G}$ (see Section 5.4.2) will be evaluated to determine if the simulation should remain in FOM or switch to ROM-PG or ROM-GNAT in the next interval. If switched to ROM-GNAT, the greedy algorithm will be performed for hyper reduction.

(II). In each ROM-PG interval, the ROM-PG will be performed to compute the incremental solutions of the reduced variable $\Delta \mathbf{Y}_r \in \mathbb{R}^r$ ($r \ll n$) ($\Delta \tilde{\mathbf{Y}} = \Phi_Y \Delta \mathbf{Y}_r$, in which $\Delta \mathbf{Y}_r$ denotes the incremental solutions in the reduced domain) and reconstructed snapshot data $\hat{\Psi}_R$ and $\hat{\Psi}_J$ per iteration. To attain the snapshot data $\hat{\Psi}_R$ and $\hat{\Psi}_J$ in the full dimension, the incremental solutions of the reduced variable $\Delta \mathbf{Y}_r$ should be used to reconstruct the incremental solution $\Delta \tilde{\mathbf{Y}}$ of the FOM first. At the end of each ROM-PG interval (labelled “3” in Fig. 5.1), the following computational procedure will be conducted: (1) iSVD is applied onto snapshot data $\hat{\Psi}_R$ and $\hat{\Psi}_J$ to update the local POD basis vectors Φ_R and Φ_J ; (2) the switch criteria, $\mathbf{C}_{R2F}$ and $\mathbf{C}_{R2G}$ will be evaluated to determine if ROM-PG should stay in use or the simulation will switch to FOM or ROM-GNAT in the next interval. If switched to ROM-GNAT, the greedy algorithm will be performed for hyper reduction.

(III). In each ROM-GNAT interval, ROM-GNAT will be simulated by using the POD basis vectors Φ_Y , Φ_R and Φ_J acquired from the previous FOM or ROM-PG intervals. Φ_Y is used for the Petrov-Galerkin projection of FOM, while Φ_R and Φ_J are used for the system approximation through the gappy POD and hyper-reduction procedure in order to greatly reduce the computational cost and accelerate the snapshot simulation. At the end of each ROM-GNAT interval (labelled “4” in Fig. 5.1), both of switch criteria, $\mathbf{C}_{R2F}$ and $\mathbf{C}_{G2R}$ will be evaluated to determine which model, FOM, ROM-PG or ROM-GNAT will be used in the next interval.

In the online simulation (labeled “5” in Fig. 5.1), the ROM-GNAT is constructed using the system approximation with the final POD basis vectors Φ_Y , Φ_R and Φ_J obtained in snapshot simulation. The system approximation loads into memory only

a small number of sample mesh indices determined by the greedy algorithm and their neighboring cells. The residual and the column-reduced Jacobian at the sample mesh points are assembled, i.e., hyper-reduction. The Gauss-Newton iteration based on the Petrov-Galerkin projection then solves the ROM as the least-squares problem.

The key novelty of the GNAT-embedded hybrid snapshot approach is to use the locally generated ROM-PG and ROM-GNAT to rapidly traverse intervals with negligible new information to update the subspace of the solutions $\mathbf{Y}$ and the residuals $\mathbf{R}_Y^k$ and the column-reduced Jacobian $\mathbf{J}_Y^k \Phi_Y$, respectively, and hence, reducing the computational cost and accelerating snapshot simulation. It should be noted that the model hierarchy of FOM→ROM-PG→ROM-GNAT represents the increasing levels of model approximation, that is, from the most accurate and costly to the least. Therefore, the model selection could be determined by the accuracy requirements of POD basis vectors and snapshot data generation. In Section 5.2.1 below, the logic threads of the GNAT-embedded hybrid snapshot simulation to determine which model to use are described in details, and the SVD/iSVD procedure for the generation and update of the POD basis vectors is introduced in Section 5.2.2.

5.2.1 GNAT-EMBEDDED HYBRID SNAPSHOT SIMULATION

The whole snapshot simulation is divided into N_i equal time intervals and the length of each interval is Δt . The time step in simulation is fixed at a constant value $\delta t, \delta t \ll \Delta t$. Let N represents the number of time steps in each interval, corresponding to $\Delta t = N\delta t$. The schematic in Fig. 5.2 illustrates how the GNAT-embedded snapshot simulation is organized by evaluating the model switches criteria at the end of each interval in order to determine which model, FOM, ROM-PG and ROM-GNAT will be used in the next interval on the fly:

1). In Fig. 5.2 the logic thread “a” includes two switch criteria $\mathbf{C}_{F2R}$ and $\mathbf{C}_{R2G}$ which are labeled as “a-1” and “a-2”, respectively. $\mathbf{C}_{F2R}$ and $\mathbf{C}_{R2G}$ will be evaluated

Figure 5.2 The flow chart of GNAT-embedded hybrid snapshot simulation.

at the end of each FOM interval to determine which model will be used in the next interval. C_{F2R} will be evaluated first: if it is false, which means that the Petrov-Galerkin projection cannot keep accuracy, then the FOM will still be employed in the next interval, and the POD basis vectors Φ_Y , Φ_R and Φ_J need to be updated. Otherwise, the FOM will not be used in the next interval and the determination of ROM-PG or ROM-GNAT will depend on the evaluation of the criterion C_{R2G} : if it is false, which implies that the GNAT approach is not accurate, and then the POD basis vectors Φ_R and Φ_J need to be updated and ROM-PG will be employed in the next interval. On the other hand, if it is true, which means that the GNAT approach is accurate enough, the simulation will switch to ROM-GNAT in the next interval, then GNAT procedure will be performed at the end of the current FOM interval.

2). The logic thread “b” includes two switch criteria C_{R2F} and C_{R2G} which are labeled as “b-1” and “b-2” in Fig. 5.2, respectively. The criterion C_{R2F} will be evaluated first at the end of each ROM-PG interval: if it is true, which means that the trajectory of the reduced solutions obtained by Petrov-Galerkin projection starts

to deviate from the subspace Φ_Y , then FOM will be employed in the next interval, and the POD basis vectors Φ_Y , Φ_R and Φ_J need to be updated. Otherwise, the evaluation of the criterion C_{R2G} will be examined: if it is true, then the GNAT approach can be accurate for system approximation of the numerical model, then the GNAT procedure is performed at the end of the current ROM-PG interval and ROM-GNAT will be employed in the next interval. Conversely, if it is false, which implies that the GNAT approach is not accurate, then the POD basis vectors Φ_R and Φ_J need to be updated and ROM-PG will still be employed in the next interval.

3). The logic thread "c" includes two switch criteria C_{R2F} and C_{G2R} which are labeled as "c-1" and "c-2", respectively. C_{R2F} and C_{G2R} will be evaluated at the end of each ROM-GNAT interval. C_{R2F} will be evaluated first: if it is true, which means that the GNAT approach will lose the accuracy of system approximation, then the simulation will switch to FOM in the next interval. Otherwise, the choice of switching to FOM is eliminated and the evaluation of the criterion C_{G2R} will be performed to determine whether the simulation will switch to ROM-PG: if it is false, which implies that the GNAT approach is still accurate enough and will be employed in the next interval; if it is true, which implies that the GNAT approach begins to deviate from the subspace of approximating the residual and the column-reduced Jacobian, leading to large errors, therefore the model will switch to ROM-PG.

The full algorithm of GNAT-embedded hybrid snapshot simulation is summarized in Algorithm. 4. It is noted that the POD basis vectors Φ_Y from Ψ should be updated after each FOM interval, while the POD basis vectors Φ_R and Φ_J (in this chapter $\Phi_R = \Phi_J$, see Section 5.4) from Ψ_R and Ψ_J should be updated after each FOM or ROM-PG interval.

Algorithm 4: The GNAT-embedded hybrid algorithm in snapshot simulation.

```

Input:  $r_d, r_e, \epsilon_1, \epsilon_2$ , and  $\epsilon, \epsilon_I^R, \epsilon_2^R$  and  $\epsilon^R, n_m, N, N_i, \Delta t$  and the initial  $\mathbf{Y}_0$ .

% The first interval in the time span
 $[\Psi, \Psi_R, \mathbf{Y}] = \text{RunFOM}();$  % Compute  $\mathbf{Y}, \Psi, \Psi_R$  using FOM
 $\Phi_Y = \mathbf{U}\Sigma\mathbf{V}^T; \Phi_Y = \mathbf{U}(:, 1:r_d+r_e);$  % Compute POD modes  $\Phi_Y$  using SVD
 $\Psi_R = \mathbf{U}_R\Sigma_R\mathbf{V}_R^T; \Phi_R = \mathbf{U}_R(:, 1:n_R);$  % Compute POD modes for  $\Phi_R$  using SVD of the residual snapshots
% Evaluate switching criteria to decide which model to use in the next interval: FOM or ROM-PG or ROM-GNAT
if  $(C_{F2R,1} \leq \epsilon_1$  and  $C_{F2R,2} \leq \epsilon_2)$ 
    SolverFlag = ROM-PG or ROM-GNAT with an order of  $(r_d + r_e)$  % ROM-PG or ROM-GNAT will be used in the next interval
else
    SolverFlag = FOM % FOM will be used in the next interval
end
if SolverFlag == ROM-PG or ROM-GNAT
    if  $(C_{R2G,1} \leq \epsilon_1^R$  and  $C_{R2G,2} \leq \epsilon_2^R)$ 
        SolverFlag = ROM-GNAT with an order of  $(r_d + r_e)$  % ROM-GNAT will be used in the next interval
        GNAT procedure using  $\Phi_R, n_m;$  % GNAT procedure for hyper reduction
    else
        SolverFlag = ROM-PG % ROM-PG will be used in the next interval
    end
end

for  $i = 2: N_i$  % Compute the time intervals from 2 to  $N_i$ 
    if SolverFlag == ROM-GNAT
         $[\delta\mathbf{Y}_r, \Delta\mathbf{Y}_r, \mathbf{Y}] = \text{RunROM-GNAT}();$  % Compute  $\mathbf{Y}$  using ROM-GNAT
        if  $(C_{R2F} \geq \epsilon)$ 
            SolverFlag = FOM % FOM will be used in the next interval
        else
            SolverFlag = ROM-PG or ROM-GNAT % ROM-PG or ROM-GNAT will be used in the next interval
        end
        if SolverFlag == ROM-PG or ROM-GNAT
            if  $(C_{G2R} \geq \epsilon^R)$ 
                SolverFlag = ROM-PG % ROM-PG will be used in the next interval
            else
                SolverFlag = ROM-GNAT % ROM-GNAT will be used in the next interval
            end
        end
    end

elseif SolverFlag == ROM-PG
     $[\Psi_R, \Delta\mathbf{Y}_r, \mathbf{Y}] = \text{RunROM-PG}();$  % Compute  $\mathbf{Y}, \Psi_R$  using ROM-PG
     $[\mathbf{U}_R, \Sigma_R, \mathbf{V}_R] = \text{iSVD}[\mathbf{U}_R, \Sigma_R, \mathbf{V}_R, \Psi_R]; \Phi_R = \mathbf{U}_R(:, 1:n_R);$  % iSVD updating of  $\Phi_R$ 
    if  $(C_{R2F} \geq \epsilon)$ 
        SolverFlag = FOM % FOM will be used in the next interval
    else
        SolverFlag = ROM-PG or ROM-GNAT % ROM-PG or ROM-GNAT will be used in the next interval
    end
    if SolverFlag == ROM-PG or ROM-GNAT
        if  $(C_{R2G,1} \leq \epsilon_1^R$  and  $C_{R2G,2} \leq \epsilon_2^R)$ 
            SolverFlag = ROM-GNAT with an order of  $(r_d + r_e)$  % ROM-GNAT will be used in the next interval
            GNAT procedure using  $\Phi_R, n_m;$  % GNAT procedure for hyper reduction
        else
            SolverFlag = ROM-PG % ROM-PG will be used in the next interval
        end
    end

elseif SolverFlag == FOM
     $[\Psi, \Psi_R, \mathbf{Y}] = \text{RunFOM}();$  % Compute  $\mathbf{Y}, \Psi, \Psi_R$  using FOM
     $[\mathbf{U}, \Sigma, \mathbf{V}] = \text{iSVD}[\mathbf{U}, \Sigma, \mathbf{V}, \Psi]; \Phi_Y = \mathbf{U}(:, 1:r_d+r_e);$  % iSVD updating of  $\Phi_Y$ 
     $[\mathbf{U}_R, \Sigma_R, \mathbf{V}_R] = \text{iSVD}[\mathbf{U}_R, \Sigma_R, \mathbf{V}_R, \Psi_R]; \Phi_R = \mathbf{U}_R(:, 1:n_R);$  % iSVD updating of  $\Phi_R$ 
    % Evaluate criteria to determine which model to use in the next interval: FOM or ROM-PG or ROM-GNAT
    if  $(C_{F2R,1} \leq \epsilon_1$  and  $C_{F2R,2} \leq \epsilon_2)$ 
        SolverFlag = ROM-PG or ROM-GNAT with an order of  $(r_d + r_e)$  % ROM-PG or ROM-GNAT will be used in the next interval
    else
        SolverFlag = FOM % FOM will be used in the next interval
    end
    if SolverFlag == ROM-PG or ROM-GNAT
        if  $(C_{R2G,1} \leq \epsilon_1^R$  and  $C_{R2G,2} \leq \epsilon_2^R)$ 
            SolverFlag = ROM-GNAT with an order of  $(r_d + r_e)$  % ROM-GNAT will be used in the next interval
            GNAT procedure using  $\Phi_R, n_m;$  % GNAT procedure for hyper reduction
        else
            SolverFlag = ROM-PG % ROM-PG will be used in the next interval
        end
    end
end

Output: POD modes  $\Phi_Y$  and  $\Phi_R$  and ROM-GNAT Model

```

5.2.2 INCREMENTAL SINGULAR VALUE DECOMPOSITION(iSVD)

The POD basis vectors Φ_Y , Φ_R and Φ_J are all initially generated by applying the SVD, respectively, on Ψ , Ψ_R and Ψ_J at the end of the first FOM interval, and by incremental SVD(also called the iSVD hereafter) in all the subsequent intervals. Again, Φ_Y will only be updated at the end of each FOM interval since no new information in Φ_Y comes from ROM-PG or ROM-GNAT, while Φ_R and Φ_J will be updated in both of FOM and ROM-PG intervals since new information of residuals in FOM and ROM-PG intervals can contribute to Φ_R and Φ_J . The latest POD basis vectors Φ_Y , Φ_R and Φ_J generated at the end of the whole snapshot simulation will eventually be used for ROM-GNAT in the online simulation.

We first present the process to generate and update the POD basis vectors Φ_Y using the iSVD from the snapshot data Ψ . The initial Φ_Y is obtained at the end of the FOM interval by applying the SVD to the snapshot data $\Psi \in \Re^{n \times N}$:

$$\Psi \approx \mathbf{U} \Sigma \mathbf{V}^T, \quad (5.10)$$

where $\mathbf{U} \in \Re^{n \times N}$, $\Sigma \in \Re^{N \times N}$, and $\mathbf{V} \in \Re^{N \times N}$ are the POD basis vectors (also called POD modes), the singular value matrix, and the right singular vectors, respectively. It should be noted that N represents the number of the time steps in each interval. Once the new snapshot data Ψ_{new} comes to be available, the iSVD [17, 18, 10, 12], which is based on a computationally efficient low-rank SVD procedure, will be performed to update the POD modes $\mathbf{U}$

$$\begin{bmatrix} \Psi & \Psi_{new} \end{bmatrix} \approx \begin{bmatrix} \mathbf{U} \Sigma \mathbf{V}^T & \Psi_{new} \end{bmatrix} = \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \mathbf{Q} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T = \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \Sigma & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T \quad (5.11)$$

where matrix $\mathbf{J}$ is the orthonormal basis vectors spanning the orthogonal component Ω of Ψ_{new} , i.e., $\Omega = (\mathbf{I} - \mathbf{U} \mathbf{U}^T) \Psi_{new}$, $\mathbf{L} = \mathbf{U}^T \Psi_{new}$ and $\mathbf{K} = \mathbf{J}^T \Omega$. Since $\mathbf{Q} = \begin{bmatrix} \Sigma & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} \in \Re^{2N \times 2N}$ has a low rank $2N \ll n$, with truncation of all these singular

matrices at rank N , the iSVD procedure can be written as:

$$\begin{bmatrix} \boldsymbol{\Psi} & \boldsymbol{\Psi}_{new} \end{bmatrix} \approx \begin{bmatrix} \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T & \boldsymbol{\Psi}_{new} \end{bmatrix} \approx \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\mathbf{V}}^T \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T, \quad (5.12)$$

where $\tilde{\mathbf{U}} \in \mathbb{R}^{2N \times N}$, $\tilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{N \times N}$ and $\tilde{\mathbf{V}} \in \mathbb{R}^{2N \times N}$ are the truncated SVD matrices of $\mathbf{Q}$.

Thus the POD modes are updated by $\mathbf{U} \leftarrow \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \tilde{\mathbf{U}}$, $\boldsymbol{\Sigma} \leftarrow \tilde{\boldsymbol{\Sigma}}$, and $\mathbf{V} \leftarrow \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \tilde{\mathbf{V}}$.

This process will repeat at the end of each subsequent interval when $\boldsymbol{\Psi}_{new}$ is generated.

The POD basis vectors $\boldsymbol{\Phi}$ will be obtained by the truncation of $\mathbf{U}$ at rank r with the expression $\boldsymbol{\Phi} = \mathbf{U}(:, 1:r)$ in Matlab and the detailed discussion is presented in [10].

This iSVD procedure can be extended to update the POD basis vectors $\boldsymbol{\Phi}_R$ and $\boldsymbol{\Phi}_J$ with the following difference: (1) the dimensions of the snapshot data $\boldsymbol{\Psi}_R$ and $\boldsymbol{\Psi}_J$ depend on the numbers of total iterations in each interval, therefore we have $\boldsymbol{\Psi}_R \in \mathbb{R}^{n \times N_s}$ and $\boldsymbol{\Psi}_J \in \mathbb{R}^{n \times N_s}$ ($\boldsymbol{\Psi}_R = \boldsymbol{\Psi}_J$ see Section 5.4) with N_s as the number of total iterations in each time interval; (2) the snapshot data $\boldsymbol{\Psi}_R$ and $\boldsymbol{\Psi}_J$ will be generated in both FOM and ROM-PG intervals, and the iSVD will be utilized to update $\boldsymbol{\Phi}_R$ and $\boldsymbol{\Phi}_J$ with the truncation rank n_R and n_J , respectively, at the end of each FOM or ROM-PG interval (in this chapter, $n_R = n_J$, see Section 5.4); and (3) the dimension of the iSVD in Eq. 5.11 and Eq. 5.12 will be N_s instead of N .

5.3 THE FOM, ROM-PG AND ROM-GNAT BASED ON FINITE VOLUME DISCRETIZATION OF 2D BURGERS EQUATION

In this chapter, the 2D Burgers equation based on the finite volume method (FVM) will be used to evaluate the proposed GNAT-embedded snapshot simulation method. Therefore, in this section, the numerical approaches used to generate the FOM, ROM-PG and ROM-GNAT are described. Although the numerical solutions of the system can be solved using a variety of temporal integration techniques, the implicit Gauss-Newton method is employed due to the use of the GNAT in the proposed approach.

5.3.1 2D BURGERS EQUATION

The full-order governing partial differential equation(PDE) of the 2D Burgers equation [42, 9, 102, 94] is given by:

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \frac{1}{Re} \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \quad (x, y, t) \in \Omega \times (0, \tau], \quad (5.13a)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = \frac{1}{Re} \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right), \quad (x, y, t) \in \Omega \times (0, \tau], \quad (5.13b)$$

subject to the initial conditions:

$$u(x, y, 0) = u_0(x, y), \quad (x, y) \in \Omega, \quad (5.14a)$$

$$v(x, y, 0) = v_0(x, y), \quad (x, y) \in \Omega, \quad (5.14b)$$

and the boundary conditions:

$$u(x, y, t) = g_u(x, y, t), \quad (x, y, t) \in \partial\Omega \times (0, \tau], \quad (5.15a)$$

$$v(x, y, t) = g_v(x, y, t), \quad (x, y, t) \in \partial\Omega \times (0, \tau], \quad (5.15b)$$

where x and y are the spatial coordinates; t is the time; $u_0(x, y)$ and $v_0(x, y)$ are the initial values; $g_u(x, y, t)$ and $g_v(x, y, t)$ are the Dirichlet boundary conditions for u and v , respectively; Ω is the arbitrary computational domain and $\partial\Omega$ represents its boundaries. Re is the Reynolds number representing the ratio of the inertia term to the viscous term.

5.3.2 FOM IN FINITE VOLUME DISCRETIZATION AND GAUSS-NEWTON SOLVER

Recently the FVM in the 2D Burgers equation is studied in a triangular mesh and the optimal error bounds in H^1 norm are found [82], in which the weak form is presented:

$$(\psi_t, \gamma w) + ((\psi \cdot \nabla) \psi, \gamma w) - \frac{1}{Re} (\Delta \psi, \gamma w) = 0, \quad (5.16)$$

by testing $\gamma w \in W_h$. The approximation space for variable ψ : $X_h = \{w_h \in X : w_h|_K \in (P_1(K))^2, K \in T_h\}$ where $P_1(K)$ is the linear polynomial set on element K in

the partition domain T_h . The approximation space for the test function: $W_h = \{w_h \in (L_2(\Omega))^2 : w_h|_{K_P^*} \in P_0(K_P^*) = C, \forall P \in N_h, w_h|_{K_P^*} = 0, \forall P \in \partial\Omega\}$ with the mapping $\gamma : X \rightarrow W_h, \gamma\psi = \sum_{P \in N_h} \psi(P)\chi_P, \forall \psi \in X$ where χ_P is the characteristic function of the dual element K_P^* . Then the semi-discrete scheme of the FVM is also given to seek $\psi_h \in X_h$ by testing $\gamma w \in W_h$:

$$(\psi_{t,h}, \gamma w_h) + A(\psi_h, \gamma w_h) + a(\psi_h, \psi_h, \gamma w_h) = 0, \quad \forall w_h \in X_h, \quad (5.17a)$$

$$\psi_h(x, 0) = P_h \psi_0(x), \quad (5.17b)$$

where $P_h \psi_0$ is the projection function of initial ψ_0 in W_h defined by:

$$(P_h \psi_0 - \psi_0, \gamma w_h) = 0, \quad (5.18)$$

and also the defined terms:

$$A(\psi_h, \gamma w_h) = -\frac{1}{Re} \langle \nabla \psi \cdot n, \gamma w \rangle, \quad (5.19a)$$

$$a(\psi_h, \psi_h, \gamma w_h) = \langle \psi \cdot n, \psi \cdot \gamma w \rangle - \langle \psi, \text{div} \psi \cdot \gamma w \rangle. \quad (5.19b)$$

In the studies [65, 81] the similar numerical treatment is also applied to solve the Navier-Stokes equations. However, the optimal error in [82] is derived with the condition that $C\|\nabla \psi\| \|\psi - \psi_h\| \leq C\|\psi - \psi_h\| \leq \varepsilon$, which means the norm of gradients is bounded by a small positive value. A good numerical treatment of the divergence term $|\langle \psi, \text{div} \psi \cdot \gamma w \rangle|$ in Eq. 5.19 cannot be easily found in the variational formulation of the FVM with Dirichlet boundary conditions. However, we can numerically simulate the weak gradient flow with the assumption $|\langle \psi, \text{div} \psi \cdot \gamma w \rangle| \rightarrow 0$ by measuring the L_∞ norm $\|\psi - \psi_h\|$ or the L_2 norm $\|\nabla \psi\|$ [54] according to [43, 59, 63] or minimizing the L_∞ errors by using the divergence free field [99]. Leveraging these previous efforts, we present the matrix and vector form of the 2D Burgers equation to generate the FOM in a rectangular mesh by considering the effects of Dirichlet boundary conditions and reconstructing the data structures. The 2D spatial computational domain Ω is discretized by $n_x \times n_y$ cell-centered finite volume mesh grids

with an equal size in x and y directions. The geometry of the spatial computational domain is $\Omega = \{(x, y) : [a, b] \times [c, d]\}$. Thus, the dimension of FOM is $n = n_x \times n_y$.

Given $\Delta x = (b - a) / n_x, \Delta y = (d - c) / n_y$:

$$\begin{aligned} & \frac{\vec{\mathbf{u}}^{n_t+1} - \vec{\mathbf{u}}^{n_t}}{\Delta \tau} + \frac{\mathbf{C}_{uu} \vec{\mathbf{u}}^{n_t+1} \cdot * \vec{\mathbf{u}}^{n_t+1}}{\Delta x} + \frac{\mathbf{C}_{uv} \vec{\mathbf{u}}^{n_t+1} \cdot * \vec{\mathbf{v}}^{n_t+1}}{\Delta y} - \frac{\mathbf{B}_{c,uu} \vec{\mathbf{u}}_L \cdot * \vec{\mathbf{u}}_L + \mathbf{B}_{c,uu} \vec{\mathbf{u}}_R \cdot * \vec{\mathbf{u}}_R}{\Delta x} \\ & - \frac{\mathbf{B}_{c,uv} \vec{\mathbf{u}}_B \cdot * \vec{\mathbf{v}}_B + \mathbf{B}_{c,uv} \vec{\mathbf{u}}_T \cdot * \vec{\mathbf{v}}_T}{\Delta y} = \frac{\mathbf{D}_{uu} \vec{\mathbf{u}}^{n_t+1}}{Re(\Delta x)^2} + \frac{\mathbf{D}_{uv} \vec{\mathbf{u}}^{n_t+1}}{Re(\Delta y)^2} + \frac{\mathbf{B}_{d,uu} (\vec{\mathbf{u}}_L + \vec{\mathbf{u}}_R)}{Re(\Delta x)^2} + \frac{\mathbf{B}_{d,uv} (\vec{\mathbf{u}}_B + \vec{\mathbf{u}}_T)}{Re(\Delta y)^2}, \end{aligned} \quad (5.20a)$$

$$\begin{aligned} & \frac{\vec{\mathbf{v}}^{n_t+1} - \vec{\mathbf{v}}^{n_t}}{\Delta \tau} + \frac{\mathbf{C}_{vu} \vec{\mathbf{v}}^{n_t+1} \cdot * \vec{\mathbf{u}}^{n_t+1}}{\Delta x} + \frac{\mathbf{C}_{vv} \vec{\mathbf{v}}^{n_t+1} \cdot * \vec{\mathbf{v}}^{n_t+1}}{\Delta y} - \frac{\mathbf{B}_{c,vu} \vec{\mathbf{v}}_L \cdot * \vec{\mathbf{u}}_L + \mathbf{B}_{c,vu} \vec{\mathbf{v}}_R \cdot * \vec{\mathbf{u}}_R}{\Delta x} \\ & - \frac{\mathbf{B}_{c,vv} \vec{\mathbf{v}}_B \cdot * \vec{\mathbf{v}}_B + \mathbf{B}_{c,vv} \vec{\mathbf{v}}_T \cdot * \vec{\mathbf{v}}_T}{\Delta y} = \frac{\mathbf{D}_{vu} \vec{\mathbf{v}}^{n_t+1}}{Re(\Delta x)^2} + \frac{\mathbf{D}_{vv} \vec{\mathbf{v}}^{n_t+1}}{Re(\Delta y)^2} + \frac{\mathbf{B}_{d,vu} (\vec{\mathbf{v}}_L + \vec{\mathbf{v}}_R)}{Re(\Delta x)^2} + \frac{\mathbf{B}_{d,vv} (\vec{\mathbf{v}}_B + \vec{\mathbf{v}}_T)}{Re(\Delta y)^2}, \end{aligned} \quad (5.20b)$$

where $*$ denotes the element-wise multiplication and the vectors of the two variables u and v in the FOM are:

$$\vec{\mathbf{u}} = \left(u_{1,1}, u_{2,1}, \dots, u_{n_x,1}, u_{1,2}, u_{2,2}, \dots, u_{n_x,2}, \dots, u_{1,n_y}, u_{2,n_y}, \dots, u_{n_x,n_y} \right)^T \in \mathbb{R}^n, \quad (5.21a)$$

$$\vec{\mathbf{v}} = \left(v_{1,1}, v_{2,1}, \dots, v_{n_x,1}, v_{1,2}, v_{2,2}, \dots, v_{n_x,2}, \dots, v_{1,n_y}, v_{2,n_y}, \dots, v_{n_x,n_y} \right)^T \in \mathbb{R}^n, \quad (5.21b)$$

where the superscript T represents the transpose. $\vec{\mathbf{u}}_L \in \mathbb{R}^n, \vec{\mathbf{u}}_R \in \mathbb{R}^n, \vec{\mathbf{u}}_B \in \mathbb{R}^n$ and $\vec{\mathbf{u}}_T \in \mathbb{R}^n$ are the boundary values of the vector $\vec{\mathbf{u}}$ at the left, right, bottom and top sides, respectively; $\vec{\mathbf{v}}_L \in \mathbb{R}^n, \vec{\mathbf{v}}_R \in \mathbb{R}^n, \vec{\mathbf{v}}_B \in \mathbb{R}^n$ and $\vec{\mathbf{v}}_T \in \mathbb{R}^n$ are the boundary values of the vector $\vec{\mathbf{v}}$ on the left, right, bottom and top sides, respectively. $\mathbf{C}_{uu} \in \mathbb{R}^{n \times n}, \mathbf{C}_{uv} \in \mathbb{R}^{n \times n}, \mathbf{C}_{vu} \in \mathbb{R}^{n \times n}$ and $\mathbf{C}_{vv} \in \mathbb{R}^{n \times n}$ are the sparse matrices of the convection term, which are obtained by the Godunov scheme [90]; $\mathbf{B}_{c,uu} \in \mathbb{R}^{n \times n}, \mathbf{B}_{c,uv} \in \mathbb{R}^{n \times n}, \mathbf{B}_{c,vu} \in \mathbb{R}^{n \times n}$ and $\mathbf{B}_{c,vv} \in \mathbb{R}^{n \times n}$ are the sparse matrices representing the fluxes at boundaries $\partial\Omega$ arising from the Godunov scheme; $\mathbf{D}_{uu} \in \mathbb{R}^{n \times n}, \mathbf{D}_{uv} \in \mathbb{R}^{n \times n}, \mathbf{D}_{vu} \in \mathbb{R}^{n \times n}$ and $\mathbf{D}_{vv} \in \mathbb{R}^{n \times n}$ are the sparse matrices of the viscous term obtained by the central differencing scheme; $\mathbf{B}_{d,uu} \in \mathbb{R}^{n \times n}, \mathbf{B}_{d,uv} \in \mathbb{R}^{n \times n}, \mathbf{B}_{d,vu} \in \mathbb{R}^{n \times n}$ and $\mathbf{B}_{d,vv} \in \mathbb{R}^{n \times n}$ are the sparse matrices representing the viscous fluxes at boundaries

$\partial\Omega$. We will measure the L_∞ norm of $\|u - u_h\|$ and $\|v - v_h\|$ to examine the intensity of the flow gradient during numerical experiments.

In order to solve this discretized nonlinear system, a backward Euler solver is employed [9, 94, 35]. This implicit numerical solver can converge fastly by iterations when a good initial guess is provided. A good option for the initial guesses is to take the solution values at the end of the previous time step. Then the nonlinear system discretized in Eq. 5.20 can be written in a residual form as:

$$\mathbf{R}_u(\vec{\mathbf{u}}^{n_t+1}, \vec{\mathbf{v}}^{n_t+1}) = 0, \quad (5.22a)$$

$$\mathbf{R}_v(\vec{\mathbf{u}}^{n_t+1}, \vec{\mathbf{v}}^{n_t+1}) = 0, \quad (5.22b)$$

where the residuals $\mathbf{R}_u$ and $\mathbf{R}_v$ are expressed as:

$$\begin{aligned} \mathbf{R}_u = & \frac{\vec{\mathbf{u}}^{n_t+1} - \vec{\mathbf{u}}^{n_t}}{\Delta\tau} + \frac{\mathbf{C}_{uu}\vec{\mathbf{u}}^{n_t+1} \cdot * \vec{\mathbf{u}}^{n_t+1}}{\Delta x} + \frac{\mathbf{C}_{uv}\vec{\mathbf{u}}^{n_t+1} \cdot * \vec{\mathbf{v}}^{n_t+1}}{\Delta y} - \frac{\mathbf{B}_{c,uu}\vec{\mathbf{u}}_L \cdot * \vec{\mathbf{u}}_L + \mathbf{B}_{c,uu}\vec{\mathbf{u}}_R \cdot * \vec{\mathbf{u}}_R}{\Delta x} \\ & - \frac{\mathbf{B}_{c,uv}\vec{\mathbf{u}}_B \cdot * \vec{\mathbf{v}}_B + \mathbf{B}_{c,uv}\vec{\mathbf{u}}_T \cdot * \vec{\mathbf{v}}_T}{\Delta y} - \left(\frac{\mathbf{D}_{uu}\vec{\mathbf{u}}^{n_t+1}}{Re(\Delta x)^2} + \frac{\mathbf{D}_{uv}\vec{\mathbf{u}}^{n_t+1}}{Re(\Delta y)^2} + \frac{\mathbf{B}_{d,uu}(\vec{\mathbf{u}}_L + \vec{\mathbf{u}}_R)}{Re(\Delta x)^2} + \frac{\mathbf{B}_{d,uv}(\vec{\mathbf{u}}_B + \vec{\mathbf{u}}_T)}{Re(\Delta y)^2} \right), \end{aligned} \quad (5.23a)$$

$$\begin{aligned} \mathbf{R}_v = & \frac{\vec{\mathbf{v}}^{n_t+1} - \vec{\mathbf{v}}^{n_t}}{\Delta\tau} + \frac{\mathbf{C}_{vu}\vec{\mathbf{v}}^{n_t+1} \cdot * \vec{\mathbf{u}}^{n_t+1}}{\Delta x} + \frac{\mathbf{C}_{vv}\vec{\mathbf{v}}^{n_t+1} \cdot * \vec{\mathbf{v}}^{n_t+1}}{\Delta y} - \frac{\mathbf{B}_{c,vu}\vec{\mathbf{v}}_L \cdot * \vec{\mathbf{u}}_L + \mathbf{B}_{c,vu}\vec{\mathbf{v}}_R \cdot * \vec{\mathbf{u}}_R}{\Delta x} \\ & - \frac{\mathbf{B}_{c,vv}\vec{\mathbf{v}}_B \cdot * \vec{\mathbf{v}}_B + \mathbf{B}_{c,vv}\vec{\mathbf{v}}_T \cdot * \vec{\mathbf{v}}_T}{\Delta y} - \left(\frac{\mathbf{D}_{vu}\vec{\mathbf{v}}^{n_t+1}}{Re(\Delta x)^2} + \frac{\mathbf{D}_{vv}\vec{\mathbf{v}}^{n_t+1}}{Re(\Delta y)^2} + \frac{\mathbf{B}_{d,vu}(\vec{\mathbf{v}}_L + \vec{\mathbf{v}}_R)}{Re(\Delta x)^2} + \frac{\mathbf{B}_{d,vv}(\vec{\mathbf{v}}_B + \vec{\mathbf{v}}_T)}{Re(\Delta y)^2} \right). \end{aligned} \quad (5.23b)$$

Accordingly, the set of algebraic equations could be solved by the Gauss-Newton iterations:

$$\begin{bmatrix} \mathbf{J}_{uu}^k & \mathbf{J}_{uv}^k \\ \mathbf{J}_{vu}^k & \mathbf{J}_{vv}^k \end{bmatrix} \begin{bmatrix} \delta\vec{\mathbf{u}}^{n_t+1,k} \\ \delta\vec{\mathbf{v}}^{n_t+1,k} \end{bmatrix} = - \begin{bmatrix} \mathbf{R}_u^k \\ \mathbf{R}_v^k \end{bmatrix}, \quad (5.24a)$$

$$\vec{\mathbf{u}}^{n_t+1,k+1} = \vec{\mathbf{u}}^{n_t+1,k} + \delta\vec{\mathbf{u}}^{n_t+1,k}, \quad (5.24b)$$

$$\vec{\mathbf{v}}^{n_t+1,k+1} = \vec{\mathbf{v}}^{n_t+1,k} + \delta\vec{\mathbf{v}}^{n_t+1,k}, \quad (5.24c)$$

where k is the iteration number. $\mathbf{J}_{uu} \in \mathbb{R}^{n \times n}$, $\mathbf{J}_{uv} \in \mathbb{R}^{n \times n}$, $\mathbf{J}_{vu} \in \mathbb{R}^{n \times n}$ and $\mathbf{J}_{vv} \in \mathbb{R}^{n \times n}$ are the corresponding Jacobian computed by:

$$\begin{aligned}\mathbf{J}_{uu} &= \frac{\partial \mathbf{R}_u}{\partial \vec{\mathbf{u}}^{n_t+1}}, \mathbf{J}_{uv} = \frac{\partial \mathbf{R}_u}{\partial \vec{\mathbf{v}}^{n_t+1}}, \\ \mathbf{J}_{vu} &= \frac{\partial \mathbf{R}_v}{\partial \vec{\mathbf{u}}^{n_t+1}}, \mathbf{J}_{vv} = \frac{\partial \mathbf{R}_v}{\partial \vec{\mathbf{v}}^{n_t+1}}.\end{aligned}\tag{5.25}$$

The iteration continues until the following convergence condition can be satisfied:

$$\|\delta \vec{\mathbf{u}}^{n_t+1,k} / \vec{\mathbf{u}}^{n_t+1,k}\|_\infty < \xi, \tag{5.26a}$$

$$\|\delta \vec{\mathbf{v}}^{n_t+1,k} / \vec{\mathbf{v}}^{n_t+1,k}\|_\infty < \xi, \tag{5.26b}$$

where $\xi > 0$ denotes the tolerance of Gauss-Newton iterations.

5.3.3 ROM BASED ON PETROV-GALERKIN PROJECTION

Eq. 5.22 contains the FOM with the full dimension and can be reduced by the subspace projection, and the approximated numerical solutions $\tilde{\vec{\mathbf{u}}}$ and $\tilde{\vec{\mathbf{v}}}$ are given by:

$$\tilde{\vec{\mathbf{u}}} = \mathbf{\Phi}_u \vec{\mathbf{u}}_r, \tag{5.27a}$$

$$\tilde{\vec{\mathbf{v}}} = \mathbf{\Phi}_v \vec{\mathbf{v}}_r, \tag{5.27b}$$

where $\mathbf{\Phi}_u \in \mathbb{R}^{n \times r_u}$ and $\mathbf{\Phi}_v \in \mathbb{R}^{n \times r_v}$ are the POD basis vectors spanning the projection subspace. Then $\vec{\mathbf{u}}_r \in \mathbb{R}^{r_u}$ and $\vec{\mathbf{v}}_r \in \mathbb{R}^{r_v}$ are solutions in the reduced domain, i.e., modal coordinates in the subspace, where $r_u \ll n, r_v \ll n$.

As shown by Eqs. 5.24 and 5.27, with the Petrov-Galerkin projection the Jacobian used in the Gauss-Newton iteration will be multiplied by the POD basis vectors:

$$\begin{aligned}\mathbf{Jr}_{uu} &= \mathbf{J}_{uu} \mathbf{\Phi}_u, & \mathbf{Jr}_{uv} &= \mathbf{J}_{uv} \mathbf{\Phi}_v, \\ \mathbf{Jr}_{vu} &= \mathbf{J}_{vu} \mathbf{\Phi}_u, & \mathbf{Jr}_{vv} &= \mathbf{J}_{vv} \mathbf{\Phi}_v,\end{aligned}\tag{5.28}$$

where $\mathbf{Jr}_{uu} \in \mathbb{R}^{n \times r_u}$, $\mathbf{Jr}_{uv} \in \mathbb{R}^{n \times r_v}$, $\mathbf{Jr}_{vu} \in \mathbb{R}^{n \times r_u}$ and $\mathbf{Jr}_{vv} \in \mathbb{R}^{n \times r_v}$ are also called the column-reduced Jacobian or the action of Jacobian on the right basis vectors in

this article. Then the reduced Gauss-Newton iteration with Petrov-Galerkin projection (Eq. 5.5) reads:

$$\begin{bmatrix} \mathbf{J}\mathbf{r}_{uu}^k & \mathbf{J}\mathbf{r}_{uv}^k \\ \mathbf{J}\mathbf{r}_{vu}^k & \mathbf{J}\mathbf{r}_{vv}^k \end{bmatrix}^T \begin{bmatrix} \mathbf{J}\mathbf{r}_{uu}^k & \mathbf{J}\mathbf{r}_{uv}^k \\ \mathbf{J}\mathbf{r}_{vu}^k & \mathbf{J}\mathbf{r}_{vv}^k \end{bmatrix} \begin{bmatrix} \delta\vec{\mathbf{u}}_r^{n_t+1,k} \\ \delta\vec{\mathbf{v}}_r^{n_t+1,k} \end{bmatrix} = - \begin{bmatrix} \mathbf{J}\mathbf{r}_{uu}^k & \mathbf{J}\mathbf{r}_{uv}^k \\ \mathbf{J}\mathbf{r}_{vu}^k & \mathbf{J}\mathbf{r}_{vv}^k \end{bmatrix}^T \begin{bmatrix} \mathbf{R}_u^k \\ \mathbf{R}_v^k \end{bmatrix}, \quad (5.29a)$$

$$\vec{\mathbf{u}}_r^{n_t+1,k+1} = \vec{\mathbf{u}}_r^{n_t+1,k} + \delta\vec{\mathbf{u}}_r^{n_t+1,k}, \quad (5.29b)$$

$$\vec{\mathbf{v}}_r^{n_t+1,k+1} = \vec{\mathbf{v}}_r^{n_t+1,k} + \delta\vec{\mathbf{v}}_r^{n_t+1,k}, \quad (5.29c)$$

which corresponds to the least squares problem that needs to be solved to obtain the solution in the reduced domain $\vec{\mathbf{u}}_r^{n_t+1,k+1} \in \mathfrak{R}^u$ and $\vec{\mathbf{v}}_r^{n_t+1,k+1} \in \mathfrak{R}^v$. They can be used to reconstruct the solutions of the FOM:

$$\tilde{\mathbf{u}}^{n_t+1,k+1} = \Phi_u \vec{\mathbf{u}}_r^{n_t+1,k+1}, \quad (5.30a)$$

$$\tilde{\mathbf{v}}^{n_t+1,k+1} = \Phi_v \vec{\mathbf{v}}_r^{n_t+1,k+1}. \quad (5.30b)$$

Then the reduced Gauss-Newton iteration continues until the convergence condition in Eq. 5.26 is satisfied.

5.3.4 HYPER-REDUCTION BASED ON GNAT

Essentially, the GNAT method employs the gappy POD method to enable the system approximation of both residuals and the column-reduced Jacobians. In other words, their POD modes during the Gauss-Newton iterations are first obtained using the snapshot data, and then used to reconstruct the residuals and the column-reduced Jacobians by only assembling a few rows of them at the sample mesh points. Following [23, 25], the POD basis vectors and the number of basis vectors of the residual (Φ_R and n_R) and the column-reduced Jacobian (Φ_J and n_J) employed in the system approximation can be the same, and then a common notation Φ_P , n_u and n_v is used:

$$\Phi_{Ru} = \Phi_{Ju} = \Phi_{Pu}, \quad n_u = n_{Ru} = n_{Ju}, \quad (5.31a)$$

$$\Phi_{Rv} = \Phi_{Jv} = \Phi_{Pv}, \quad n_v = n_{Rv} = n_{Jv}, \quad (5.31b)$$

where the subscript u and v , respectively, denote the quantities in the two directions $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$. The sampling matrix $\mathbf{P}$ will be updated at the end of each FOM or ROM-PG interval with new information added into the snapshot data. The detailed process to obtain the sampling matrix $\mathbf{P}$ by the greedy algorithm is given in Section 5.4. Then following Eq 5.8, the GNAT-based system approximation of the residual and the column-reduced Jacobian for the 2D Burgers equation is given by:

$$\widetilde{\mathbf{R}}_u^k \approx \Phi_{Pu} \left(\mathbf{P}^T \Phi_{Pu} \right)^+ \mathbf{P}^T \mathbf{R}_u^k, \quad (5.32a)$$

$$\widetilde{\mathbf{R}}_v^k \approx \Phi_{Pv} \left(\mathbf{P}^T \Phi_{Pv} \right)^+ \mathbf{P}^T \mathbf{R}_v^k \quad (5.32b)$$

and

$$\begin{bmatrix} \widetilde{\mathbf{Jr}}_{uu}^k & \widetilde{\mathbf{Jr}}_{uv}^k \\ \widetilde{\mathbf{Jr}}_{vu}^k & \widetilde{\mathbf{Jr}}_{vv}^k \end{bmatrix} \approx \begin{bmatrix} \Phi_{Pu} \left(\mathbf{P}^T \Phi_{Pu} \right)^+ \mathbf{P}^T \mathbf{Jr}_{uu}^k & \Phi_{Pu} \left(\mathbf{P}^T \Phi_{Pu} \right)^+ \mathbf{P}^T \mathbf{Jr}_{uv}^k \\ \Phi_{Pv} \left(\mathbf{P}^T \Phi_{Pv} \right)^+ \mathbf{P}^T \mathbf{Jr}_{vu}^k & \Phi_{Pv} \left(\mathbf{P}^T \Phi_{Pv} \right)^+ \mathbf{P}^T \mathbf{Jr}_{vv}^k \end{bmatrix}, \quad (5.33)$$

where all the residuals and the column-reduced Jacobians are left-multiplied by $\mathbf{P}^T$, indicating that only the entries at the sample points/indices are needed for the approximation. $\Phi_{Pu} \left(\mathbf{P}^T \Phi_{Pu} \right)^+$ and $\Phi_{Pv} \left(\mathbf{P}^T \Phi_{Pv} \right)^+$ can be pre-computed, and in other words, they can be computed during the FOM and the ROM-PG interval for use in the ROM-GNAT interval. Then the Gauss-Newton iteration for the ROM is given by:

$$\begin{bmatrix} \widetilde{\mathbf{Jr}}_{uu}^k & \widetilde{\mathbf{Jr}}_{uv}^k \\ \widetilde{\mathbf{Jr}}_{vu}^k & \widetilde{\mathbf{Jr}}_{vv}^k \end{bmatrix}^T \begin{bmatrix} \widetilde{\mathbf{Jr}}_{uu}^k & \widetilde{\mathbf{Jr}}_{uv}^k \\ \widetilde{\mathbf{Jr}}_{vu}^k & \widetilde{\mathbf{Jr}}_{vv}^k \end{bmatrix} \begin{bmatrix} \delta \vec{\mathbf{u}}_r^{n_t+1,k} \\ \delta \vec{\mathbf{v}}_r^{n_t+1,k} \end{bmatrix} = - \begin{bmatrix} \widetilde{\mathbf{Jr}}_{uu}^k & \widetilde{\mathbf{Jr}}_{uv}^k \\ \widetilde{\mathbf{Jr}}_{vu}^k & \widetilde{\mathbf{Jr}}_{vv}^k \end{bmatrix}^T \begin{bmatrix} \widetilde{\mathbf{R}}_u^k \\ \widetilde{\mathbf{R}}_v^k \end{bmatrix}, \quad (5.34a)$$

$$\vec{\mathbf{u}}_r^{n_t+1,k+1} = \vec{\mathbf{u}}_r^{n_t+1,k} + \delta \vec{\mathbf{u}}_r^{n_t+1,k}, \quad (5.34b)$$

$$\vec{\mathbf{v}}_r^{n_t+1,k+1} = \vec{\mathbf{v}}_r^{n_t+1,k} + \delta \vec{\mathbf{v}}_r^{n_t+1,k}. \quad (5.34c)$$

Therefore, the GNAT-based implicit Gauss-Newton solver in Eq. 5.34 is then used to compute the solution, i.e., model coordinates in the reduced domain subject to the convergence condition in Eq. 5.26. Note that since the residuals and the Jacobians in Eq. 5.34 can be assembled efficiently using the gappy POD approach only at the sample mesh points, the ROM-GNAT in both the offline snapshot simulation and the online simulation can be computed at the fast speedup. Finally the numerical solutions in the reduced domain $\vec{\mathbf{u}}_r^{n_t+1,k+1}$ and $\vec{\mathbf{v}}_r^{n_t+1,k+1}$ can be used to reconstruct the solutions in the full domain by Eq. 5.30.

5.3.5 THE GREEDY ALGORITHM AND GAPPY POD METHOD IN GNAT

The gappy POD method enables to approximate the residual and column-reduced Jacobian by only calculating a small number of selected row entries. Therefore, sample mesh points/indices need to be determined judiciously. It should also be reiterated that the full-domain solutions at the cells neighboring the sample mesh cells are also needed, because they affect the transport fluxes at the cell interfaces, and the residuals and column-reduced Jacobians at those selected rows. As introduced in Section 5.1.3, the full domain solution at both the sample mesh cells and their neighboring cells need to be retrieved during each iteration. Fortunately, the indices of the neighboring cells can be extracted before the ROM-GNAT interval simulation, and the computational cost is almost negligible. The detailed procedure of obtaining the neighboring cell indices is presented elsewhere [11] by the authors.

In this study, a greedy algorithm is used to identify the sample cell indices $\mathfrak{S} = [\wp_1, \wp_2, \dots, \wp_{n_m}]$ and sampling matrix $\mathbf{P}$ for the residuals and column-reduced Jacobians based on the greedy method [27]. Algorithm. 5 below shows its pseudo-codes. Since the same POD modes are used for both the residual and the column-reduced Jacobian approximation in this chapter (see Eq. 5.31 above), there is no need to find another set for the sample index or the sample matrix for the latter.

Algorithm 5: The greedy algorithm for sampling index $\mathfrak{S}$ and sampling matrix $\mathbf{P}$.

Input: Current POD matrices $\mathbf{U}_{Ru}$ and $\mathbf{U}_{Rv}$ of residuals, number of sample indices n_m , and the number of POD basis vectors n_{Ru} , and n_{Rv} (after truncation and used in ROM)
% $\mathbf{U}_{Ru}$ and $\mathbf{U}_{Rv}$ are the POD basis vectors of residuals in $\mathbf{u}$ and $\mathbf{v}$ directions without truncation

1. $\Phi_{Ru} = \mathbf{U}_{Ru}(:, 1:n_{Ru});$ *% POD basis vectors along u direction in ROM*
2. $\Phi_{Rv} = \mathbf{U}_{Rv}(:, 1:n_{Rv});$ *% POD basis vectors along v direction in ROM*
3. $\mathfrak{S} \leftarrow \emptyset;$ *% initialize index set*
4. $\mathbf{P} \leftarrow \emptyset;$ *% initialize identity matrix*
5. $[\sim, \varphi_1] = \max\{|\mathbf{U}_{Ru}(:, 1)|^2 + |\mathbf{U}_{Rv}(:, 1)|^2\};$ *% find the first index*
6. $\mathfrak{S} = [\varphi_1];$
7. $\mathbf{P} = [\mathbf{e}_{\varphi_1}];$
8. for $i = 2$ to n_m *% for loop to find other index*
9. if $i \leq n_{Ru}$ *% compute reconstruction error $\mathbf{r}_u$ in $\mathbf{u}$ direction*
using the greedy algorithm and the least squares method
 10. $\mathbf{A}_u = \mathbf{P}^T \Phi_{Ru}(:, 1:i-1);$
 11. $\mathbf{b}_u = \mathbf{P}^T \mathbf{U}_{Ru}(:, i);$
 12. solve $\mathbf{A}_u \mathbf{c}_u = \mathbf{b}_u;$
 13. $\mathbf{r}_u = \mathbf{U}_{Ru}(:, i) - \Phi_{Ru}(:, 1:i-1) * \mathbf{c}_u;$
14. else
15. $\mathbf{A}_u = \mathbf{P}^T \Phi_{Ru};$
16. $\mathbf{b}_u = \mathbf{P}^T \mathbf{U}_{Ru}(:, i);$
17. solve $\mathbf{A}_u \mathbf{c}_u = \mathbf{b}_u;$
18. $\mathbf{r}_u = \mathbf{U}_{Ru}(:, i) - \Phi_{Ru} * \mathbf{c}_u;$
19. end
20. if $i \leq n_{Rv}$ *% compute reconstruction error $\mathbf{r}_v$ in $\mathbf{v}$ direction*
using the greedy algorithm and the least squares method
 21. $\mathbf{A}_v = \mathbf{P}^T \Phi_{Rv}(:, 1:i-1);$
 22. $\mathbf{b}_v = \mathbf{P}^T \mathbf{U}_{Rv}(:, i);$
 23. solve $\mathbf{A}_v \mathbf{c}_v = \mathbf{b}_v;$
 24. $\mathbf{r}_v = \mathbf{U}_{Rv}(:, i) - \Phi_{Rv}(:, 1:i-1) * \mathbf{c}_v;$
25. else
26. $\mathbf{A}_v = \mathbf{P}^T \Phi_{Rv};$
27. $\mathbf{b}_v = \mathbf{P}^T \mathbf{U}_{Rv}(:, i);$
28. solve $\mathbf{A}_v \mathbf{c}_v = \mathbf{b}_v;$
29. $\mathbf{r}_v = \mathbf{U}_{Rv}(:, i) - \Phi_{Rv} * \mathbf{c}_v;$
30. end
31. $[\sim, \varphi_i] = \max\{|\mathbf{r}_u|^2 + |\mathbf{r}_v|^2\};$ *% find new index*
32. $\mathbf{P} \leftarrow [\mathbf{P} \ \mathbf{e}_{\varphi_i}];$ *% update identity matrix*
33. $\mathfrak{S} \leftarrow [\mathfrak{S}^T \ \varphi_i]^T;$ *% update index set*
34. end

Output: $\mathfrak{S}$ and $\mathbf{P}$

As we can see, the greedy search will be conducted at the end of each FOM or ROM-PG interval and take the updated POD basis vectors $\mathbf{U}_{Ru}$ and $\mathbf{U}_{Rv}$ without truncation, the number of sample indices n_m , and the number of POD basis vectors (n_{Ru} and n_{Rv}) in the ROM-GNAT as the input. Line 1-2 extract the POD basis vectors by truncation. Line 3-7 initialize the pool of sample indices and find the first index. Line 9-19 and Line 20-30, respectively, compute the sample indices in u and v directions, respectively. Specifically, Line 10-13 find the first n_{Ru} indices, and are equivalent to the DEIM algorithm [27] and Line 15-18 are used to determine additional indices based on the reconstruction errors of higher-order POD modes. Once all sample indices $\mathfrak{S}$ are found, the residual and the column-reduced Jacobian at these mesh cell locations can be assembled. As discussed above, in order to do this, the solution values at their neighboring cells are also needed to evaluate the transport fluxes at the cell interfaces that are dependent on the specific differencing schemes used [11].

5.4 SNAPSHOT DATA GENERATION AND MODEL SWITCH CRITERIA IN GNAT-EMBEDDED HYBRID SNAPSHOT SIMULATION

The methodology of GNAT-embedded hybrid snapshot simulation is introduced in Section 5.2. In this section, the details how the snapshot data are collected and what are the criteria to enable on-the-fly model switches are provided in the specific context of the 2D Burgers equation.

5.4.1 SNAPSHOT DATA ACQUISITION AND POD MODES

In the 2D Burgers equation, the POD modes Φ_u and Φ_v are generated by using the snapshot data of the incremental solutions $\Delta \vec{\mathbf{u}}$ and $\Delta \vec{\mathbf{v}}$, which are collected at the end of each time step within the FOM interval. Therefore the dimensions of the snapshot data are $\Psi_u \in \mathbb{R}^{n \times N}$ and $\Psi_v \in \mathbb{R}^{n \times N}$, where N denotes the number of time steps

in each time interval. We define that $\Delta\vec{\mathbf{u}}$ and $\Delta\vec{\mathbf{v}}$ are the incremental solutions of $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ (similar to $\Delta\mathbf{Y}$ in Section 5.2: $\vec{\mathbf{u}} = \vec{\mathbf{u}}^0 + \Delta\vec{\mathbf{u}}$ and $\vec{\mathbf{v}} = \vec{\mathbf{v}}^0 + \Delta\vec{\mathbf{v}}$), i.e., the difference between the solution $\vec{\mathbf{u}}^{n_t}$ or $\vec{\mathbf{v}}^{n_t}$ at time step n_t and the solution at the end of the previous interval $\vec{\mathbf{u}}^0$ or $\vec{\mathbf{v}}^0$, which are also the initial solutions of the current time interval. Note that $\Delta\vec{\mathbf{u}}$ and $\Delta\vec{\mathbf{v}}$ are different from $\delta\vec{\mathbf{u}}$ and $\delta\vec{\mathbf{v}}$, and the latter are the search directions computed in each Gauss-Newton iteration.

There are sufficient conditions on snapshot data selection to obtain Φ_R and Φ_J that meet different levels of consistency requirements in the system approximation of GNAT [23, 25]. However, a strong guarantee of consistency will need large computational cost. By considering the balance among the computational load, consistency, and convergence difficulty, Proposition. 1 in [23] is selected with a modification: the snapshot data for the residuals $\mathbf{R}^k$ and the actions of Jacobian on the right basis vectors $\mathbf{J}^k\Phi_Y$ are also collected in each FOM interval, where subscript Y can be u or v to represent the two spatial directions of the 2D Burgers equation. The difference of our GNAT-embedded hybrid approach from the Proposition.1 in [23] is that the snapshot data are collected during both the FOM and ROM-PG intervals for $\mathbf{R}_Y^k$ and $\mathbf{J}^k\Phi_Y$. It means that in FOM intervals, the snapshot data are collected for $\mathbf{R}_Y^k$ and $\mathbf{J}^k\Phi_Y$; in ROM-PG intervals, the snapshot data are also collected by reconstructing the data from the reduced domain to the full domain $\widetilde{\mathbf{R}}_Y^k$ and $\widetilde{\mathbf{J}^k\Phi_Y}$.

It should be noted that the snapshot data of residuals are collected in each Gauss-Newton iteration. The number of the columns of the snapshot data Ψ_{Ru} and Ψ_{Rv} are the total number of Newton iteration in each time interval N_s , which is not the same for each time interval and is larger than N .

5.4.2 CRITERIA OF MODEL SWITCH

In Section 5.2 the model switch criteria have been briefly introduced to illustrated the flow chart of the GNAT-embedded hybrid snapshot approach. The detailed infor-

mation about these criteria in the context of the 2D Burgers equation are described as follows.

1.) $\mathbf{C}_{F2R}$ consists of two sub-criteria $\mathbf{C}_{F2R,1}$ and $\mathbf{C}_{F2R,2}$, both of which are based on the singular values that indicate the importance of POD basis vectors [85]. $\mathbf{C}_{F2R,1}$ represents the cumulative energy ratio of the POD modes $r_d + 1 \rightarrow N$ relative to the total $1 \rightarrow N$, and $\mathbf{C}_{F2R,2}$ is the cumulative energy ratio of the POD modes $r_d + r_e + 1 \rightarrow N$ relative to the total $1 \rightarrow N$ [89, 10, 12]. In the 2D Burgers equation, the criteria in both u and v directions have the same form but with different POD modes:

$$\mathbf{C}_{F2R,u1} = \left(\frac{\sum_{j=r_{ud}+1}^N \sigma_{u,j}^2}{\sum_{i=1}^N \sigma_{u,i}^2} \right)^{\frac{1}{2}} \leq \epsilon_1, \quad \mathbf{C}_{F2R,u2} = \left(\frac{\sum_{j=r_{ud}+r_{ue}+1}^N \sigma_{u,j}^2}{\sum_{i=1}^N \sigma_{u,i}^2} \right)^{\frac{1}{2}} \leq \epsilon_2, \quad (5.35a)$$

$$\mathbf{C}_{F2R,v1} = \left(\frac{\sum_{j=r_{vd}+1}^N \sigma_{v,j}^2}{\sum_{i=1}^N \sigma_{v,i}^2} \right)^{\frac{1}{2}} \leq \epsilon_1, \quad \mathbf{C}_{F2R,v2} = \left(\frac{\sum_{j=r_{vd}+r_{ve}+1}^N \sigma_{v,j}^2}{\sum_{i=1}^N \sigma_{v,i}^2} \right)^{\frac{1}{2}} \leq \epsilon_2, \quad (5.35b)$$

where $\sigma_{u,1}, \dots, \sigma_{u,j}, \dots$, and $\sigma_{u,N}$ are singular values of POD subspace $\mathbf{U}_u \in \mathbb{R}^{n \times N}$ and $\sigma_{v,1}, \dots, \sigma_{v,j}, \dots$, and $\sigma_{v,N}$ are singular values of POD subspace $\mathbf{U}_v \in \mathbb{R}^{n \times N}$; r_{ud} and r_{vd} are the numbers of basic POD modes retained for Φ_u and Φ_v ; and r_{ue} and r_{ve} are the numbers of extended modes for Φ_u and Φ_v . Thus we have $r_u = r_{ud} + r_{ue}$ and $r_v = r_{vd} + r_{ve}$. ϵ_1 and ϵ_2 are the tolerance parameters for the sub-criteria to ensure the POD modes retained provides excellent representation of the solution data subspace.

2.) $\mathbf{C}_{R2F}$ examines how accurate the existing POD modes capture the trajectory of the local ROM-PG and whether the Petrov-Galerkin projection should maintain in use for the next interval. In the 2D Burgers equation, $\mathbf{C}_{R2F}$ given by:

$$\mathbf{C}_{R2F,u} = \left(\frac{\sum_{j=r_{ud}+1}^{r_{ud}+r_{ue}} (\Delta \vec{\mathbf{u}}_{r,j})^2 (i\Delta t)}{\sum_{j=1}^{r_{ud}+r_{ue}} (\Delta \vec{\mathbf{u}}_{r,j})^2 (i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon, \quad (5.36a)$$

$$\mathbf{C}_{R2F,v} = \left(\frac{\sum_{j=r_{vd}+1}^{r_{vd}+r_{ve}} (\Delta \vec{\mathbf{v}}_{r,j})^2 (i\Delta t)}{\sum_{j=1}^{r_{vd}+r_{ve}} (\Delta \vec{\mathbf{v}}_{r,j})^2 (i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon, \quad (5.36b)$$

where $\Delta \vec{\mathbf{u}}_r(i\Delta t) = \Phi_u^T \Delta \vec{\mathbf{u}}(i\Delta t)$ and $\Delta \vec{\mathbf{v}}_r(i\Delta t) = \Phi_v^T \Delta \vec{\mathbf{v}}(i\Delta t)$ are the incremental solutions in the reduced domain at the end of the i^{th} interval, respectively; $\Delta \vec{\mathbf{u}}_{r,j}(i\Delta t)$ and $\Delta \vec{\mathbf{v}}_{r,j}(i\Delta t)$ represent the coordinates of the j^{th} mode of these solutions. $\mathbf{C}_{R2F,u}$ and $\mathbf{C}_{R2F,v}$ monitor the relative ratio of the modal coordinates of the extended modes (r_{ue} and r_{ve}) to those of all the retained modes ($r_{ud} + r_{ue}$ and $r_{vd} + r_{ve}$), and examine the dominance of the current POD basis vectors Φ_u and Φ_v [89, 10, 12]. ϵ is the tolerance parameter for the criteria to ensure the dominance of the leading modes to capture the key dynamic information in the projection subspace.

3). $\mathbf{C}_{R2G}$ also consists of two sub-criteria $\mathbf{C}_{R2G,1}$ and $\mathbf{C}_{R2G,2}$ both of which examine the singular values of POD modes of residuals Φ_R . Similar to $\mathbf{C}_{F2R}$, $\mathbf{C}_{R2G,1}$ and $\mathbf{C}_{R2G,2}$, respectively, denote the cumulative energy ratio of the POD modes $n_R + 1 \rightarrow N_s$ and $n_R + n_e + 1 \rightarrow N_s$ relative to the total $1 \rightarrow N_s$, where superscript R denotes the quantities associated with the residuals, and can be either u or v to denote the two spatial directions for the 2D Burgers equation. The criteria are then given as:

$$\mathbf{C}_{R2G,u1} = \left(\frac{\sum_{j=n_u+1}^{N_s} (\sigma_{u,j}^R)^2}{\sum_{i=1}^{N_s} (\sigma_{u,i}^R)^2} \right)^{\frac{1}{2}} \leq \epsilon_1^R, \quad \mathbf{C}_{R2G,u2} = \left(\frac{\sum_{j=n_u+n_{ue}+1}^{N_s} (\sigma_{u,j}^R)^2}{\sum_{i=1}^{N_s} (\sigma_{u,i}^R)^2} \right)^{\frac{1}{2}} \leq \epsilon_2^R, \quad (5.37a)$$

$$\mathbf{C}_{R2G,v1} = \left(\frac{\sum_{j=n_v+1}^{N_s} (\sigma_{v,j}^R)^2}{\sum_{i=1}^{N_s} (\sigma_{v,i}^R)^2} \right)^{\frac{1}{2}} \leq \epsilon_1^R, \quad \mathbf{C}_{R2G,v2} = \left(\frac{\sum_{j=n_v+n_{ve}+1}^{N_s} (\sigma_{v,j}^R)^2}{\sum_{i=1}^{N_s} (\sigma_{v,i}^R)^2} \right)^{\frac{1}{2}} \leq \epsilon_2^R, \quad (5.37b)$$

where $n_u \ll n$ and $n_v \ll n$ denote the dimensions of POD modes for the residuals Φ_{Ru} and Φ_{Rv} , and n_{ue} and n_{ve} are the numbers of extended modes in the corresponding directions. ϵ_1^R and ϵ_2^R are the two tolerance parameters.

4). $\mathbf{C}_{G2R}$ inspects the accuracy of the system approximation, which determines whether the ROM-GNAT could still be used for the next interval. In the 2D Burgers equation, the criteria are specifically given by:

$$\mathbf{C}_{G2R,u} = \left(\frac{\sum_{j=r_{ud}+1}^{r_{ud}+r_{ue}} (\delta \vec{\mathbf{u}}_{r,j})^2 (i\Delta t)}{\sum_{j=1}^{r_{ud}+r_{ue}} (\delta \vec{\mathbf{u}}_{r,j})^2 (i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon^R, \quad (5.38a)$$

$$\mathbf{C}_{G2R,v} = \left(\frac{\sum_{j=r_{vd}+1}^{r_{vd}+r_{ve}} (\delta \vec{\mathbf{v}}_{r,j})^2 (i\Delta t)}{\sum_{j=1}^{r_{vd}+r_{ve}} (\delta \vec{\mathbf{v}}_{r,j})^2 (i\Delta t)} \right)^{\frac{1}{2}} \geq \epsilon^R, \quad (5.38b)$$

where $(\delta \vec{\mathbf{u}}_{r,j})(i\Delta t)$ and $(\delta \vec{\mathbf{v}}_{r,j})(i\Delta t)$ denote the coordinates of the j^{th} mode of $\delta \vec{\mathbf{u}}_r$ and $\delta \vec{\mathbf{v}}_r$ at the end of the i^{th} interval in both of $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ directions, respectively (see Eq. 5.34). Here, $\delta \vec{\mathbf{u}}_r$ and $\delta \vec{\mathbf{v}}_r$ are the solutions in the reduced domain computed by ROM-GNAT for system approximation: $\delta \vec{\mathbf{u}}_r \leftarrow \arg \min_{\alpha_u \in \mathbb{R}^r} \|\mathbf{J}^k \widetilde{\Phi} \alpha_u + \widetilde{\mathbf{R}}^k\|_2$; $\delta \vec{\mathbf{v}}_r \leftarrow \arg \min_{\alpha_v \in \mathbb{R}^r} \|\mathbf{J}^k \widetilde{\Phi} \alpha_v + \widetilde{\mathbf{R}}^k\|_2$, where α_u and α_v are search directions. $\mathbf{C}_{G2R,u}$ and $\mathbf{C}_{G2R,v}$ evaluate the relative ratios of the modal coordinates of the extended modes (r_{ue} and

r_{ve}) to those of all the retained modes ($r_{ud} + r_{ue}$ and $r_{vd} + r_{ve}$), assess the modal trajectory of the least squares solution, and examine the accuracy of the system approximation. ϵ^R is the tolerance parameter.

5.5 NUMERICAL EXPERIMENTS AND DISCUSSION

Numerical experiments with the 2D Burgers equation are conducted and the results are presented in this section to evaluate the performance (including efficiency and accuracy) of the proposed GNAT-embedded snapshot simulation methodology.

The computational domain is a 2D square $\Omega = \{(x, y) : [0, 1] \times [0, 1]\}$ and the time set is $\tau = \{t : [0, 1]\}$. The exact solution of the 2D Burgers equation [42, 9] is used to examine the accuracy.

$$u(x, y, t) = \frac{3}{4} - \frac{1}{4[1 + e^{(-4x+4y-t)Re/32}]}, \quad (5.39a)$$

$$v(x, y, t) = \frac{3}{4} + \frac{1}{4[1 + e^{(-4x+4y-t)Re/32}]}, \quad (5.39b)$$

in which the initial conditions could be derived by setting $t = 0$ and the boundary conditions could be derived directly by setting x and y to the values on $\partial\Omega$.

We define the numerical errors:

$$L_\infty(u) = \|u - u_h\|_\infty, \quad (5.40a)$$

$$L_\infty(v) = \|v - v_h\|_\infty, \quad (5.40b)$$

to examine the accuracy of numerical solutions relative to the exact solution and the intensity of convection subject to the assumption of the weak gradient flow. The root mean square error (**RMSE**) is also employed to measure the difference between the ROM and the FOM in both u and v directions [94, 35]:

$$\mathbf{RMSE}(u) = \sqrt{\frac{\sum_{i=1}^n (u_i^{FOM} - u_i^{ROM})^2}{n}}, \quad (5.41a)$$

$$\mathbf{RMSE}(v) = \sqrt{\frac{\sum_{i=1}^n (v_i^{FOM} - v_i^{ROM})^2}{n}}. \quad (5.41b)$$

where superscript FOM and ROM represents the FOM solution and the ROM solution, respectively, and subscript i denotes the solution of the i^{th} computational mesh cell.

The numerical experiments and discussion in this section will be presented as follows: in Section 5.5.1 the convergence test of the FOM of the 2D Burgers equation will be examined by comparing the FOM solution with the exact solution above and the errors will be discussed quantitatively; in Section 5.5.2 the numerical results and errors of the ROM-PG and the ROM-GNAT of the 2D Burgers equation will be presented; in Section 5.5.3 the numerical experiments of the GNAT-embedded hybrid snapshot approach will be implemented and the results of both of the snapshot simulation and the online simulation will be described. The tolerance of the convergence condition in Eq. 5.26 is set as $\xi = 10^{-6}$.

5.5.1 HIGH-FIDELITY FULL ORDER MODEL

We first present the convergence test and numerical results of the FOM.

CONVERGENCE TESTS OF FOM

The convergence test of the FOM is performed by refining the mesh density $n_x \times n_y$ from 8×8 to 64×64 , and the numerical error of the FOM is quantified by comparing its solution with the exact solution. Table. 5.1 lists the errors at time $t = 1$ for three different Re numbers: $Re = 10$, $Re = 40$, and $Re = 100$. We can see that as the mesh density increases, the discrepancy between the FOM and the exact solution decreases and the FOM accuracy keeps improving, indicating convergence of mesh refinement of the 2D Burgers equation. However, when Re is higher, the convergence order is lower.

Table 5.1 Spatial convergence by refining the mesh $n_x \times n_y$. In the L_∞ norm errors the reference solutions are the exact solutions. $\Delta h = \Delta x = \Delta y = 1/n_x = 1/n_y$. All the errors are calculated at $t = 1$ with the time step $\delta t = 0.001$.

Δh	$Re = 10$				$Re = 40$				$Re = 100$			
	$L_\infty(u)$	order	$L_\infty(v)$	order	$L_\infty(u)$	order	$L_\infty(v)$	order	$L_\infty(u)$	order	$L_\infty(v)$	order
$\frac{1}{8}$	$5.48e^{-3}$	-	$7.29e^{-3}$	-	$25.51e^{-3}$	-	$30.06e^{-3}$	-	$55.99e^{-3}$	-	$64.60e^{-3}$	-
$\frac{1}{16}$	$2.71e^{-3}$	1.013	$3.47e^{-3}$	1.070	$15.48e^{-3}$	0.721	$16.69e^{-3}$	0.850	$42.17e^{-3}$	0.409	$42.50e^{-3}$	0.605
$\frac{1}{32}$	$1.31e^{-3}$	1.050	$1.56e^{-3}$	1.153	$8.70e^{-3}$	0.831	$8.42e^{-3}$	0.987	$28.08e^{-3}$	0.588	$27.69e^{-3}$	0.618
$\frac{1}{64}$	$0.64e^{-3}$	1.033	$0.70e^{-3}$	1.156	$4.67e^{-3}$	0.897	$4.24e^{-3}$	0.990	$17.04e^{-3}$	0.721	$16.51e^{-3}$	0.748

FOM SOLUTIONS AND ACCURACY

The FOM solutions at $t = 1$ with three different values of the Re number ($Re = 10$, $Re = 40$, and $Re = 100$) are shown in Fig. 5.3. Since Re number indicates the ratio of inertia force to viscous force, the higher value of Re corresponds to stronger convection and flow gradient. Fig. 5.3 clearly shows that the values of $\vec{u}$ and $\vec{v}$ in the small Re value ($Re = 10$) exhibit the smooth and almost linear transition across the computational domain, while those in $Re = 100$ display the sharper gradient in the transition region. A very high Re number can lead to a very sharp front [94].

The L_∞ norm errors of the FOM results for the three different Re numbers are also calculated by comparing with the exact solutions in Eq. 5.39, and the results are presented in Fig. 5.4 as a function of time from 0 to 1. It is seen that the high Re number could increase L_∞ norm error from 6×10^{-6} at $Re = 10$ to 1.7×10^{-2} at $Re = 100$ with the same mesh configuration ($n_x \times n_y = 64 \times 64$) due to stronger convection and gradients. Therefore, in this chapter only the Re number up to 100 is considered in order to satisfy the weak gradient flow requirement of finite volume discretization of the 2D Burgers equation.

5.5.2 ROM-PG AND ROM-GNAT

Since our hybrid snapshot method involves both ROM-PG and ROM-GNAT and the model switch, in this section, the ROM-PG and ROM-GNAT will be first veri-

Figure 5.3 The FOM solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$.

Figure 5.4 The L_∞ norm errors of (a) $L_\infty(u)$ and (b) $L_\infty(v)$ with the three different values of Re : (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$.

fied independently by comparing their results with the FOM results. The ROM with Petrov-Galerkin projection, i.e., ROM-PG is constructed following the procedure presented by [23], which includes: (1) the snapshot simulation is performed by the FOM and the snapshot data Ψ_u and Ψ_v of the incremental solutions $\Delta\vec{u}$ and $\Delta\vec{v}$ are collected. Then the POD basis vectors $\Phi_u \in \mathbb{R}^{n \times r_u}$ and $\Phi_v \in \mathbb{R}^{n \times r_v}$ are computed by the SVD with truncation; and (2) the Petrov-Galerkin projection is performed by using Φ_u and Φ_v , and the ROM solutions $\vec{u}_r$ and $\vec{v}_r$ are then computed. The ROM solutions are then used to reconstruct the solutions in the full domain. Only ROM-PG is used in the online simulation.

Likewise, the ROM with hyper reduction (ROM-GNAT) also follows that in [23], which consists of three steps: (1) the snapshot simulation is performed by the FOM and the snapshot data Ψ_u and Ψ_v of incremental solutions $\Delta\vec{u}$ and $\Delta\vec{v}$ are acquired. The POD modes $\Phi_u \in \mathbb{R}^{n \times r_u}$ and $\Phi_v \in \mathbb{R}^{n \times r_v}$ are computed by the SVD; (2) the ROM-PG is computed using the POD modes Φ_u and Φ_v . The snapshot data Ψ_{Pu} and Ψ_{Pv} ($\Psi_{Pu} = \Psi_{Ru} = \Psi_{Ju}$ and $\Psi_{Pv} = \Psi_{Rv} = \Psi_{Jv}$) of the residuals $\mathbf{R}_u$ and $\mathbf{R}_v$ are acquired during Gauss-Newton iterations. The system approximation procedure is then performed using Φ_{Pu} and Φ_{Pv} to generate the "sample indices" $\mathfrak{S}$ and the precomputed matrices $\Phi_{Pu} (\mathbf{P}^T \Phi_{Pu})^+$ and $\Phi_{Pv} (\mathbf{P}^T \Phi_{Pv})^+$; (3) the ROM-GNAT is constructed by the system approximation using Φ_{Pu} and Φ_{Pv} and the pre-computed matrices. Likewise, the ROM-GNAT is used in the online simulation.

ROM-PG SOLUTIONS AND ACCURACY

Fig. 5.5 illustrates the ROM-PG solutions with three different Re numbers at $t = 1$. Similar to the FOM solutions in Fig. 5.3, when Re increases from 10 to 100, indicative of stronger convection, the gradients of $\vec{u}$ and $\vec{v}$ become sharper. The ROM-PG solutions exhibit great agreement with the FOM results in Fig. 5.3 with only minor differences. Visually, all important flow characteristics observed in the FOM are also

Figure 5.5 The ROM-PG solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 4$ is used in the ROM-PG.

presented in the ROM-PG results. Fig. 5.6 displays the absolute errors of the ROM-PG solutions in Fig. 5.5 compared to the FOM solutions in Fig. 5.3. The maximum values of the absolute errors in both of $\vec{u}$ and $\vec{v}$ increase with the Re number and the regions of large absolute errors are mostly on both sides of the diagonal line from $x = 0$ and $y = 0$ to $x = 1$ and $y = 1$. As the Re number increases, the gradient becomes stronger and the maximum numerical errors increase.

Fig. 5.7 illustrates the RMSE of ROM-PG solutions at the three different Re numbers using different ROM orders: $r_u = r_v = 4$ in the upper row and $r_u = r_v = 12$ in the lower row. It is obvious that when more POD modes are used, the Petrov-Galerkin projection yields more accurate numerical results and lower RMSE compared with the FOM. Similar to Fig. 5.6 the higher Re number leads to larger ROM errors. When the number of POD modes increases from $r_u = r_v = 4$ to $r_u = r_v = 12$, the

Figure 5.6 The absolute errors of ROM-PG solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time of $\delta t = 0.001$ and $r_u = r_v = 4$ is used in the ROM-PG.

RMSE can decrease from 2×10^{-4} to 2×10^{-6} .

ROM-GNAT SOLUTIONS AND ACCURACY

Fig. 5.8 shows the ROM-GNAT results with the three different Re numbers at $t = 1$. It can be found that the ROM-GNAT solutions match very well with the FOM results in Fig. 5.3 and the ROM-PG results in Fig. 5.5, and the difference is visually negligible. The absolute errors of the ROM-GNAT in the computational domain at $t = 1$ is illustrated in Fig. 5.9. The observation is similar to the ROM-PG results that the magnitude of the absolute errors increases with the Re number, and at higher Re number numerical errors of larger magnitudes are mostly found around the diagonal “wave line”, where the dramatic gradients of both $\vec{u}$ and $\vec{v}$ are present.

Figure 5.7 The RMSE of ROM-PG solutions of $\vec{\mathbf{u}}$ (left) and $\vec{\mathbf{v}}$ (right) with the three different Re numbers: $Re = 10$, $Re = 40$ and $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$. In the upper row: (a) $r_u = r_v = 4$; and in the lower row (b) $r_u = r_v = 12$ are used in the ROM-PG.

Fig. 5.10 depicts the RMSE of ROM-GNAT solutions relative to the FOM solutions with the three different Re numbers. It can be seen that the higher Re number leads to larger RMSE, and the RMSE grows from around 8×10^{-7} at $Re = 10$ to around $4 \times 10^{-6} \sim 2 \times 10^{-5}$ at $Re = 100$. Compared with Fig. 5.7 in which the largest RMSE is between 4×10^{-8} and 1×10^{-6} approximately, the hyper-reduction used in the ROM-GNAT causes relatively larger errors. However, the computation of ROM-GNAT can be greatly accelerated as shown in Table. 5.2, which lists the CPU time of simulating the three types of models. It is seen that the reduction of the computational time by the ROM-PG is limited relative to the FOM because the

Figure 5.8 The ROM-GNAT solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$ is used in the ROM-GNAT.

ROM-PG need to reconstruct the solution and assemble the residual and Jacobian matrices in the full domain in each iteration. In contrast, ROM-GNAT can obtain magnificent acceleration by more than 70% compared with the FOM due to the use of hyper-reduction and selective update of the residual and column-reduced Jacobian.

Table 5.2 Computational time usage(Unit:s) of the three models: FOM, ROM-PG and ROM-GNAT with $Re = 10$, $Re = 40$ and $Re = 100$. $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$, $\delta t = 0.001$ are used.

Model Type	$Re = 10$	$Re = 40$	$Re = 100$
FOM	2059.73s	1972.28s	2116.23s
ROM-PG	1878.55s	1778.68s	1777.48s
ROM-GNAT	560.68s	536.47s	536.25s

Figure 5.9 The absolute errors of the ROM-GNAT solutions of $\vec{u}(\text{top})$ and $\vec{v}(\text{bottom})$ at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$ is used in the ROM-GNAT.

Figure 5.10 The RMSE of the ROM-GNAT solutions of $\vec{u}(\text{left})$ and $\vec{v}(\text{right})$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, and $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$ is used in the ROM-GNAT.

5.5.3 GNAT-EMBEDDED HYBRID SNAPSHOT SIMULATION AND ONLINE SIMULATION

The results of the proposed GNAT-embedded hybrid snapshot simulation will be presented in this section. Recall that different from the ROM-PG and the ROM-GNAT above that use the FOM to generate snapshot, the hybrid simulation method uses a combination of the FOM, local ROM-PG, and local ROM-GNAT as presented in Section 5.2. The ROM-GNAT used in the online simulation is constructed based on the POD basis vectors and sample indices obtained from the GNAT-embedded snapshot simulation.

GNAT-EMBEDDED HYBRID SNAPSHOT SIMULATION

In the snapshot simulation, the total time span is divided into 10 equal intervals with $\Delta t = 0.1$ in each interval, and the time step is $\delta t = 0.001$. The FOM is always used in the initial interval (See Algorithm. 4), and the computational parameters are given as follows: $r_u = r_{ud} + r_{ue} = 8 + 4 = 12$, $r_v = r_{vd} + r_{ve} = 8 + 4 = 12$, $n_u = 36$, $n_v = 36$, $n_{ue} = 8$, $n_{ve} = 8$, $n_m = 100$, $\epsilon_1 = 10^{-2}$, $\epsilon_2 = 10^{-3}$, $\epsilon = 10^{-2}$, $\epsilon_1^R = 10^{-2}$, $\epsilon_2^R = 10^{-3}$ and $\epsilon^R = 10^{-2}$.

The computational time and accuracy of the GNAT-embedded hybrid snapshot simulation are presented in Table. 5.3, which shows the breakdown of the computational time usage with the three different Re numbers. It can be seen that the snapshot simulation takes slightly more computational time at higher Re numbers due to the more frequent use of the FOMs, e.g., 3, 5, and 6 FOM intervals used at $Re = 10$, 40, and 100, respectively. The computational time of $Re = 40$ and $Re = 100$ are almost the same, although in the latter one ROM-PG interval is replaced by one FOM interval. This is because the computational acceleration by the ROM-PG is limited relative to the FOM as discussed above. Compared to the FOM-based snapshot simulation, i.e., the entry of the FOM in Table. 5.2, the GNAT-embedded snapshot simulation is accelerated by about 33%.

Table 5.3 The list of model type and the computational time (Unit:s) in each interval of the GNAT-embedded hybrid snapshot simulation. $r_u = r_v = 12$, $n_u = n_v = 36$, $n_m = 100$, $\epsilon = 10^{-2}$, $\epsilon^R = 10^{-2}$, 64×64 mesh configuration in computational domain Ω , and $\delta t = 0.001$ are used.

Interval	$Re = 10$		$Re = 40$		$Re = 100$	
No.	Model Type	Time(s)	Model Type	Time(s)	Model Type	Time(s)
1 st	FOM	203.20	FOM	201.34	FOM	201.54
2 nd	ROM-GNAT	55.65	ROM-GNAT	54.21	ROM-GNAT	54.31
3 rd	ROM-PG	183.82	ROM-PG	182.07	FOM	202.47
4 th	FOM	203.96	FOM	201.43	ROM-GNAT	51.01
5 th	ROM-GNAT	55.19	ROM-GNAT	54.31	FOM	203.65
6 th	ROM-PG	183.66	FOM	201.37	ROM-GNAT	40.80
7 th	ROM-GNAT	54.42	ROM-GNAT	54.32	FOM	204.44
8 th	ROM-PG	181.62	FOM	200.61	ROM-GNAT	46.99
9 th	ROM-GNAT	54.62	ROM-GNAT	54.68	FOM	206.26
10 th	FOM	203.20	FOM	202.98	FOM	207.61
<i>Total</i>	3 FOM s, 3 ROM-PG s 4 ROM-GNAT s	1379.34	5 FOM s, 1 ROM-PG 4 ROM-GNAT s	1407.32	6 FOM s, 0 ROM-PG 4 ROM-GNAT s	1419.08

Fig. 5.11 shows the snapshot solutions of $\vec{u}$ and $\vec{v}$ in the GNAT-embedded hybrid simulation at $t = 1$. Although the ROM-PG and ROM-GNAT are both utilized in the hybrid approach, the numerical results at the last time step also agree with the FOM in Fig. 5.3. The absolute errors of the GNAT-embedded hybrid simulation relative to the FOM are presented in Fig. 5.12. Likewise, the absolute errors increase with the Re number and the intensity of convection. The regions containing larger errors at $Re = 100$ are mostly distributed on both sides of the diagonal of the computational domain with the value around 1.06×10^{-3} .

The RMSE of GNAT-embedded hybrid snapshot simulation is shown in Fig. 5.13. It is observed that the RMSE tends to increase in each ROM-PG or ROM-GNAT interval while dropping in each FOM interval in both of the $\vec{u}$ and $\vec{v}$ according to the models used in each interval listed in Table. 5.3. However, the magnitude of the RMSE remains below 5×10^{-4} in the entire hybrid snapshot simulation, which confirms the feasibility and accuracy of the GNAT-embedded hybrid methodology.

Figure 5.11 The GNAT-embedded hybrid snapshot solutions of $\vec{\mathbf{u}}$ (top) and $\vec{\mathbf{v}}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.

ROM-GNAT IN ONLINE SIMULATION

Fig. 5.14 depicts the ROM-GNAT solutions of $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ with $Re = 10$, $Re = 40$ and $Re = 100$ in the online simulation at $t = 1$. Recall that in the online simulation only the ROM-GNAT constructed using the POD modes and sample indices at the end of the hybrid snapshot simulation, is computed. We can find that at $Re = 10$ and $Re = 40$, the ROM-GNAT solutions are smooth and accurate compared with the FOM, while at $Re = 100$ the wrinkles near the transition region of the rapidly-varying gradient are observed, which may be attributed to the difficulty of the hybrid snapshot simulation approach to capture the fast propagation of wave fronts for strong convective flow as pointed out in our prior research [11]. The computational time is similar to the ROM-GNAT in Table. 5.2.

Figure 5.12 The absolute errors of GNAT-embedded hybrid snapshot solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.

Figure 5.13 The RMSE of GNAT-embedded hybrid snapshot solutions of $\vec{u}$ (left) and $\vec{v}$ (right) with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.

Figure 5.14 The ROM-GNAT (obtained by the hybrid snapshot simulation) solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers in the online simulation: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.

The absolute errors of the ROM-GNAT solutions in the online simulation are shown in Fig. 5.15. It can be seen that the magnitude of numerical errors increase with the Re number, and the distribution of the large errors is similar to the above, and mostly on both sides of the diagonal line across the computational domain. At $Re = 100$ the largest absolute error is nearly 9.73×10^{-3} in the region of large velocity gradients. We also examined even higher values of the Re number (i.e. $Re = 200$) and found that it is challenging to obtain accurate results equivalent to the FOM. Thus, the proposed methodology may not be applicable to high-gradient flow with strong convection, in particular, the one involving moving wave fronts[62, 61].

Fig. 5.16 illustrates the RMSE of the ROM-GNAT in the online simulation with $Re = 10$, $Re = 40$ and $Re = 100$. It can be found that from $t = 0 \rightarrow t = 0.1$ the values of the RMSE are very low, while after this period the RMSE quickly rises up

Figure 5.15 The absolute errors of the ROM-GNAT (obtained by the hybrid snapshot simulation) solutions of $\vec{u}$ (top) and $\vec{v}$ (bottom) at $t = 1$ with the three different Re numbers in the online simulation: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.

to around 5×10^{-5} in $Re = 10$ and around 3×10^{-3} in $Re = 100$, which indicates the higher Re can lead to larger numerical errors of the ROM-GNAT in the online simulation.

5.6 SUMMARY OF CHAPTER.5

This chapter proposes a GNAT-embedded hybrid snapshot simulation method to accelerate generation of high-quality snapshot data and accurate GNAT-ROMs. By using the obtained snapshot data, a ROM-GNAT model can be constructed more efficiently while preserving the numerical accuracy. In distinct contrast to our previous research [10], the novelties of the present work include: each of the time interval is

Figure 5.16 The RMSE of the ROM-GNAT (obtained by the hybrid snapshot simulation) solutions of $\vec{u}$ (left) and $\vec{v}$ (right) with the three different Re numbers: (a) $Re = 10$, (b) $Re = 40$ and (c) $Re = 100$. $n_x \times n_y = 64 \times 64$ mesh configuration is used in the computational domain Ω with the time step of $\delta t = 0.001$, $r_u = r_v = 12$, $n_u = n_v = 36$, and $n_m = 100$ are used in the ROM-GNAT.

simulated by the models with varying levels of approximation, accuracy, and computational cost, viz., FOM, local ROM-PG, or local ROM-GNAT. Quantitative criteria are proposed to examine the characteristics of POD modes and trajectories in the reduced subspaces in order to determine the model switch among FOM, local ROM-PG and local ROM-GNAT on-the-fly. Two sets of POD modes, respectively, for the solutions and the residuals are maintained and continuously updated by the iSVD [17, 18] to minimize the computational cost during the hybrid snapshot simulation. The former is used to construct the local ROM-PG, while both need to be used for constructing the local ROM-GNAT. The key idea underlying the hybrid snapshot simulation is that only intervals containing new subspace information will be simulated by the FOM or the ROM-PG, while the rest with negligible contribution can be quickly traversed by the ROM-GNAT. Therefore, the two sets of POD modes and ROM trajectories in these two POD subspace can be used to guide the model switch among FOM, ROM-PG, and ROM-GNAT. In the online simulation, the ROM-GNAT based on the POD basis vectors generated in hybrid snapshot simulation could

achieve accurate numerical results and salient computational performance. The proposed GNAT-embedded hybrid snapshot approach is examined and demonstrated using the 2D Burgers equation in the FVM form recently presented by [82]. The corresponding FOM, ROM-PG, and ROM-GNAT are obtained, and their numerical results in the snapshot and the online simulation are compared and discussed. Important technical findings include:

(1) The ROMs of the 2D Burgers equation based on the FVM discretization is constructed and verified in weak gradient flow. Petrov-Galerkin projection and hyper reduction produce accurate numerical results compared to the FOM solution.

(2) The feasibility, accuracy and efficiency of the proposed GNAT-embedded snapshot approach is verified. It allows to generate less snapshot data while keeping subspace representation. The POD basis vectors generated by the hybrid snapshot simulation are accurate for ROM-PG and ROM-GNAT generation, and can be used to decide the switch of models with varying levels of approximation.

(3) The proposed GNAT-embedded hybrid approach could reduce the computational time of the snapshot simulation by about 33% while maintaining good numerical accuracy with the RMSE around 10^{-3} .

(4) In the online simulation, the ROM-GNAT constructed using the POD basis vectors generated from the GNAT-embedded hybrid snapshot simulation, could also achieve good numerical accuracy with the RMSE less than 10^{-2} and accelerate the simulation by around 70%.

It should be noticed that the assumption of weak gradient flow ($Re \leq 100$) has been taken in order to analyze the 2D Burgers equation with the proposed GNAT-embedded hybrid snapshot simulation method. Approaches to relax (or at least partially) this assumption for stronger flows will be investigated. In addition, the extension of this method to other physics problems, such as heat transfer and Navier-Stokes flow will also be considered in the future work.

Figure 5.17 The flux directions used by the Godunov scheme in 2D mesh cells.

5.7 APPENDIX TO CHAPTER.5: THE COEFFICIENT MATRIX OF THE FINITE VOLUME DISCRETIZATION OF THE 2D BURGERS EQUATION

In Section 5.3, the Godunov scheme is used to derive the matrices $\mathbf{C}_{uu}$, $\mathbf{C}_{uv}$, $\mathbf{C}_{vu}$ and $\mathbf{C}_{vv}$ in Eq. 5.20 which store the coefficients of the flux across faces between cells and the matrices $\mathbf{B}_{c,uu}$, $\mathbf{B}_{c,uv}$, $\mathbf{B}_{c,vu}$ and $\mathbf{B}_{c,vv}$ which store the coefficients of the flux across domain boundaries. The matrices produced by the Godunov scheme for 2D Burgers equation with a 2D rectangular mesh configuration according to [90] are given below.

Fig. 5.17 illustrates the 2×2 mesh configuration. The flux flow in the horizontal direction in (A-1), (A-2), (A-3) and (A-4) determine $\mathbf{C}_{uu}$, $\mathbf{C}_{vu}$, $\mathbf{B}_{c,uu}$ and $\mathbf{B}_{c,vu}$. In (A-1):

$$\mathbf{C}_{uu} = \mathbf{C}_{vu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}, \mathbf{B}_{c,uu} = \mathbf{B}_{c,vu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (5.42)$$

. In (A-2):

$$\mathbf{C}_{uu} = \mathbf{C}_{vu} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uu} = \mathbf{B}_{c,vu} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.43)$$

. In (A-3):

$$\mathbf{C}_{uu} = \mathbf{C}_{vu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uu} = \mathbf{B}_{c,vu} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (5.44)$$

. And in (A-4): the left flux F_L is along the horizontal flow direction across the cell interface (from Cell 1 to Cell 2, and also from Cell 3 to Cell 4), the right flux F_R , is along the horizontal flow direction across the cell interface (from Cell 2 to Cell 1, and also from Cell 4 to Cell 3),

$$(1) F_L > F_R, \mathbf{C}_{uu} = \mathbf{C}_{vu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}, \mathbf{B}_{c,uu} = \mathbf{B}_{c,vu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.45a)$$

$$(2) F_L < F_R, \mathbf{C}_{uu} = \mathbf{C}_{vu} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uu} = \mathbf{B}_{c,vu} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.45b)$$

.

The flux flow in the vertical direction in (A-5), (A-6), (A-7) and (A-8) determine $\mathbf{C}_{uv}$, $\mathbf{C}_{vv}$, $\mathbf{B}_{c,uv}$ and $\mathbf{B}_{c,vv}$. In (A-5):

$$\mathbf{C}_{uv} = \mathbf{C}_{vv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uv} = \mathbf{B}_{c,vv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (5.46)$$

. In (A-6):

$$\mathbf{C}_{uv} = \mathbf{C}_{vv} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uv} = \mathbf{B}_{c,vv} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.47)$$

. In (A-7):

$$\mathbf{C}_{uv} = \mathbf{C}_{vv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uv} = \mathbf{B}_{c,vv} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (5.48)$$

And in (A-8): the bottom flux F_B is along the vertical flow direction across the cell interface (from Cell 1 to Cell 3, and also from Cell 2 to Cell 4), and the top flux F_T , is along the vertical flow direction across the cell interface (from Cell 3 to Cell 1, and also from Cell 4 to Cell 2),

$$(1) F_B > F_T, \mathbf{C}_{uv} = \mathbf{C}_{vv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}, \mathbf{B}_{c,uv} = \mathbf{B}_{c,vv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.49a)$$

$$(2) F_B < F_T, \mathbf{C}_{uv} = \mathbf{C}_{vv} = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B}_{c,uv} = \mathbf{B}_{c,vv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5.49b)$$

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this dissertation, a new hybrid snapshot simulation methodology is proposed to accelerate snapshot data generation and reduced order model (ROM) construction. The key idea underlying this method is to judiciously utilize local ROMs to rapidly traverse regions in the time domain where snapshot data has negligible contribution to new information for projection subspace while the full order model (FOM) is still employed in the regions of importance. Several candidates approaches for local ROMs based on proper orthogonal decomposition (POD) are investigated and compared in computational accuracy and efficiency, including the traditional ROM, ROM-DEIM, Petrov-Galerkin projection ROM (ROM-PG), and the Gauss-Newton with approximated tensors (GNAT).

In Chapter 2, the main methodology of the hybrid snapshot simulation is presented. The whole snapshot simulation is divided into multiple time intervals, and each interval will be simulated by either FOM or ROM. The POD basis vectors will be generated or updated in each FOM interval using the singular value decomposition (SVD) or incremental singular value decomposition (iSVD), while remaining unchanged in each local ROM. A set of model switch criteria are also introduced to determine on-the-fly the alternation between the FOM and the local ROM. At the end of the hybrid snapshot simulation, the POD basis vectors will be utilized to construct the ROM for the online simulation. The numerical results on the nonlinear heat transfer model indicate that the computational time of the hybrid snapshot simula-

tion could be reduced by $\sim 50\%$ compared with the traditional snapshot simulation based on the FOM only, while keeping excellent accuracy (relative error $\leq 1\%$) in both the snapshot and the online simulation. The feasibility of the proposed method to accelerate snapshot data and ROM generation is convincingly verified.

In Chapter 3, a methodology of combining the hybrid snapshot simulation approach and DEIM construction for online simulation is proposed. An approach is developed to streamline the hybrid snapshot simulation, snapshot data generation, SVD/iSVD for POD construction/updates, and ROM-DEIM construction for the online simulation. To accurately capture the dominant subspace information of nonlinear terms, the data for nonlinear terms collected in the local ROMs needs to be reconstructed into the full domain first, and reconstructed data is used to update the POD basis vectors for nonlinear terms. In the case studies of the numerical thermal models obtained by the cell-centered finite volume method, the feasibility of the proposed methodology is proven effective to construct the online ROM-DEIM, and the online ROM-DEIM demonstrates excellent computational performance: the error is below 0.8% and computational acceleration is about 16.4% to 27.5% compared the ROM without DEIM.

In Chapter.4, a DEIM-embedded hybrid approach is proposed to perform the hybrid snapshot simulation by alternating among FOM, ROM and ROM-DEIM intervals. Different from the methods in Chapter 2 and 3, the novelties in this approach include: (1) multiple time intervals in the snapshot simulation are simulated by a model hierarchy, viz., FOM, ROM, and ROM-DEIM; and (2) two groups of POD modes are generated and updated, the POD basis vectors for solution variables and POD basis vectors for nonlinear terms; and (3) the POD modes and trajectories in the reduced domain are examined by quantitative criteria to determine the model

switch among FOM, ROM and ROM-DEIM on the fly. The numerical results confirm that the proposed method can achieve acceleration in both snapshot simulation and online simulation while maintaining the accuracy (relative error $\leq 1\%$). In the snapshot simulation the computational acceleration is improved by $10.5\% \sim 27.85\%$, and in the online simulation the computational time of the ROM-DEIM is only about 25% of the ROM without DEIM.

In Chapter.5, we propose a GNAT-embedded hybrid approach to accelerate high-quality snapshot data generation and ROM construction at the low discretization level (i.e., algebraic equation), in which the simulation alternates among FOM, ROM-PG, and ROM-GNAT intervals, and ROM-GNAT is used for efficient online simulation. This method is verified with the 2D Burgers equation. The novelties in this chapter include: (1) multiple time intervals in the snapshot simulation are simulated with a hierarchical model approximations for Gauss-Newton iterations, viz., FOM, ROM-PG, ROM-GNAT; (2) two groups of POD modes are constructed and updated throughout the snapshot simulation, which are the POD basis vectors for Petrov-Galerkin projection to construct the local ROM-PG and ROM-GNAT, and the POD basis vectors for residuals (same as column-reduced Jacobian) to construct the local ROM-GNAT, respectively; and (3) new quantitative criteria are presented to determine the model switch among FOM, ROM-PG and ROM-GNAT. In the online simulation, the system approximation is adopted to construct the ROM-GNAT using the two groups of POD basis vectors to accelerate computation. Numerical results indicate that the GNAT-embedded hybrid approach is feasible to accelerate the snapshot simulation by 33% in the runtime while attaining desirable numerical accuracy. The ROM-GNAT in the online simulation also demonstrates excellent performance, viz., computational acceleration by 70% and relative error $\leq 10^{-2}$ compared to the FOM.

In summary, this dissertation investigates and develops the hybrid snapshot simulation framework to accelerate snapshot data generation and ROM construction. The future work will extend the proposed methodology along several directions, including:

(1) The local ROM-based acceleration strategies will also be extended to other numerical approaches, such as missing point estimation (MPE), trajectory piecewise linear approximation (TPWL), and others. The hybrid method can embed ROM-MPE or ROM-TPWL into the snapshot simulation for computational acceleration. A detailed comparison among these hybrid snapshot simulation methods will be conducted to further investigate their merits, limitations, and computational efficiency and accuracy.

(2) The different hybrid methods and local ROM approaches above should demonstrate distinct snapshot simulation performance for various physical problems. For example, the TPWL-embedded method may be more appropriate for weakly nonlinear (e.g., electric circuit) and parabolic PDEs (convective or conductive heat equations), while GNAT and DEIM ROMs are better suited for strongly nonlinear and Navier-Stokes type problems. Therefore, one of the future efforts will be applying these hybrid methods to various problems, and identify more robust and efficient one for problems under specific categories, paving the way towards real-world uses of the hybrid snapshot simulation methods.

(3) Additional improvements on POD basis vectors update, model switch criteria, and adaptive intervals and time steps, could also be studied to boost computational performance in both accuracy and efficiency. For example, the iSVD in the constant time intervals and constant number of time steps is used as the current POD basis update method, but it can be improved to accommodate adaptive time intervals and

the number of time steps depending on the simulation characteristics. The model switch criteria could be parameterized to monitor the POD basis vectors in a robust way. The time steps could also be adapted during simulation to automatically reflect the precision of POD basis vectors in the low-dimensional approximation.

BIBLIOGRAPHY

- [1] Ali, M., Steih, K., Urban, K. Reduced basis methods with adaptive snapshot computations, *Advances in Computational Mathematics*, **43**(2) 2017, 257–294.
- [2] Alla, A., Kutz, JN. Nonlinear model order reduction via dynamic mode decomposition, *SIAM Journal on Scientific Computing*, **39**(5) 2017, B778–B796.
- [3] Allery, C., Hamdouni, A., Ryckelynck, D., Verdon, N. A priori reduction method for solving the two-dimensional Burgers’ equations, *Applied Mathematics and Computation*, **217**(15) 2011, 6671–6679.
- [4] Amsallem, D., Farhat, C. Interpolation method for adapting reduced-order models and application to aeroelasticity, *AIAA Journal*, **46**(7) 2008, 1803–1813.
- [5] Amsallem, D., Zahr, M. J., Farhat, C. Nonlinear model order reduction based on local reduced-order bases, *International Journal for Numerical Methods in Engineering*, **92**(10) 2012, 891–916.
- [6] Amsallem, D., Farhat, C. Stabilization of projection-based reduced-order models, *International Journal for Numerical Methods in Engineering*, **91**(4) 2012, 358–377.
- [7] Amsallem, D., Zahr, M.J., Washabaugh, K. Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction, *Advances in Computational Mathematics*, **41**(5) 2015, 1187–1230.
- [8] Astrid, P., Weiland, S., Willcox, K., Backx, T. Missing point estimation in models described by proper orthogonal decomposition, . *IEEE Transactions on Automatic Control*, **53**(10) 2008, 2237–2251.
- [9] Bahadır, A. A fully implicit finite-difference scheme for two-dimensional Burgers’ equations, *Applied Mathematics and Computation*, **137**(1) 2003, 131–137.
- [10] Bai, F., Wang, Y. Reduced order modeling based on hybrid snapshot simulation, *International Journal of Computational Methods*, **2050029** 2020.

- [11] Bai, F., Wang, Y. DEIM reduced order model constructed by hybrid snapshot simulation, *SN Applied Sciences*, **2** 2165 2020.
- [12] Bai, F., Wang, Y. DEIM-embedded hybrid snapshot simulation for reduced order model generation, *Sustainable Computing: Informatics and Systems*, in review 2020.
- [13] Bai, F., Wang, Y. A reduced order modeling method based on GNAT-embedded hybrid snapshot simulation, *Computers & Mathematics with Applications*, submitted 2021.
- [14] Benner, P., Gugercin, S., Willcox, K. A survey of projection-based model reduction methods for parametric dynamical systems, *SIAM Review*, **57**(4) 2015, 2015, 483–531.
- [15] Billaud-Friess, M., Nouy, A. Dynamical model reduction method for solving parameter-dependent dynamical systems, *SIAM Journal on Scientific Computing*, **39**(4) 2017, A1766–1792.
- [16] Bonomi, D., Manzoni, A., Quarteroni, A. A matrix DEIM technique for model reduction of nonlinear parametrized problems in cardiac mechanics, *Computer Methods in Applied Mechanics and Engineering*, **300** 2017, 300–326.
- [17] Brand, M. Incremental singular value decomposition of uncertain data with missing values, in *Computer Vision*, Heyden, A., Sparr, G., Nielsen, M. and Johansen, P. (eds.), Berlin, Heidelberg, (Springer Berlin Heidelberg) 2002, 707–720.
- [18] Brand, M. Fast low-rank modifications of the thin singular value decomposition, *Linear Algebra and Its Applications*, **415**(1) 2006, 20–30.
- [19] Bremer, J., Goyal, P., Feng, L., Benner, P., Sundmacher, K. POD-DEIM for efficient reduction of a dynamic 2D catalytic reactor model, *Computers & Chemical Engineering*, **106**, 2017, 777–784.
- [20] Bui-Thanh, T., Willcox, K., Ghattas, O. . Model reduction for large-scale systems with high-dimensional parametric input space, *SIAM Journal on Scientific Computing*, **30**(6) 2008, 3270–3288.
- [21] Camphouse, R.C., Myatt, J., Schmit, R., Glauser, M., Ausseur, J., Andino, M., Wallace, R. A snapshot decomposition method for reduced order modeling and boundary feedback control, *AIAA Conference* 2008.

- [22] Cardoso, M.A., Durlinsky, L.J., Sarma, P. Development and application of reduced-order modeling procedures for subsurface flow simulation, *International Journal for Numerical Methods in Engineering*, **77**(9) 2009, 1322–1350.
- [23] Carlberg, K., Bou-Mosleh, C., Farhat, C. Efficient non-linear model reduction via a least-squares Petrov-Galerkin projection and compressive tensor approximations, *International Journal for Numerical Methods in Engineering*, **86**(2) 2011, 155–181.
- [24] Carlberg, K., Amsallem, D., Avery, P., Zahr, M., Farhat, C. The GNAT nonlinear model reduction method and its application to fluid dynamics problems, *6th AIAA Theoretical Fluid Mechanics Conference, Honolulu, Hawaii, June 2011*.
- [25] Carlberg, K., Farhat, C., Cortial, J., Amsallem, D. The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows, *Journal of Computational Physics*, **242** 2013, 623–647.
- [26] Carlberg, K., Choi, Y., Sargsyan, S. Conservative model reduction for finite-volume models, *Journal of Computational Physics*, **371** 2018, 280–314.
- [27] Chaturantabut, S. and Sorensen, D. C. Nonlinear model reduction via discrete empirical interpolation, *SIAM Journal on Scientific Computing*, **32**(5) 2010, 2737–2764.
- [28] Chaturantabut, S., Sorensen, D.C. Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media, *Mathematical and Computer Modelling of Dynamical Systems*, **17**(4) 2011, 337–353.
- [29] Chaturantabut, S. and Sorensen, D. C. A state space error estimate for pod-deim nonlinear model reduction, *SIAM Journal on Numerical Analysis*, **50**(1) 2012, 46–63.
- [30] Chaturantabut, S. Temporal localized nonlinear model reduction with a priori error estimate, *Applied Numerical Mathematics*, **119** 2017, 225–238.
- [31] Chen, G., Li, Y. M. and Yan, G. R. Limit cycle oscillation prediction and control design method for aeroelastic system based on new nonlinear reduced order model, *International Journal of Computational Methods*, **8**(1) 2011, 77–90.

- [32] Chen, G., Yan, G. R. and Li, Y. M. Reduced order model based placement optimization of multiple control surfaces for active aeroelastic control, *International Journal of Computational Methods*, **11**(6) 2014, 1350081.
- [33] Dehghan, M., Abbaszadeh, M. Proper orthogonal decomposition variational multiscale element free Galerkin (POD-VMEFG) meshless method for solving incompressible Navier–Stokes equation, *Computer Methods in Applied Mechanics and Engineering*, **311** 2016, 856–888.
- [34] Dehghan, M., Abbaszadeh, M. The use of proper orthogonal decomposition (POD) meshless RBF-FD technique to simulate the shallow water equations, *Journal of Computational Physics*, **351** 2017, 478–510.
- [35] Dehghan, M., Abbaszadeh, M. A combination of proper orthogonal decomposition-discrete empirical interpolation method (POD-DEIM) and meshless local RBF-DQ approach for prevention of groundwater contamination, *Computer & Mathematics with Applications*, **75** 2018, 1390–1412.
- [36] Dehghan, M., Abbaszadeh, M. An upwind local radial basis functions-differential quadrature (RBF-DQ) method with proper orthogonal decomposition (POD) approach for solving compressible Euler equation, *Engineering Analysis with Boundary Elements*, **92** 2018, 244–256.
- [37] Dehghan, M., Abbaszadeh, M. A reduced proper orthogonal decomposition (POD) element free Galerkin (PODEFEG) method to simulate two-dimensional solute transport problems and error estimate, *Applied Numerical Mathematics*, **126**, 92–112.
- [38] Dehghan, M., Abbaszadeh, M. Error analysis and numerical simulation of magnetohydrodynamics (MHD) equation based on the interpolating element free Galerkin (IEFG) method, *Applied Numerical Mathematics*, **137** 2019, 252–273.
- [39] Everson, R., Sirovich, L. Karhunen–loève procedure for gappy data, *Journal of the Optical Society of America A*, **12**(8) 1995, 1657–1664.
- [40] Elman, HC., Forstall, V. Numerical solution of the parameterized steady-state Navier-Stokes equations using empirical interpolation methods, *Computer Methods in Applied Mechanics and Engineering*, **317** 2017, 380–399.
- [41] Fareed, H., Singler, J. R., Zhang, Y., Shen, J. Incremental proper orthogonal decomposition for PDE simulation data, *Computer & Mathematics with Applications*, **75**(6) 2018, 1942–1960.

- [42] Fletcher, CAJ. Generating exact solutions of the two-dimensional Burgers' equations, *International Journal for Numerical Methods in Fluids*, **3**(3) 1983, 213–216.
- [43] Fonseca, G., Linares, F., Ponce, G. The IVP for the dispersion generalized Benjamin–Ono equation in weighted sobolev spaces, *Annales de l'Institut Henri Poincaré (C) Nonlinear Analysis*, **30**(5) 2013, 763–790.
- [44] Fu, X., Nathan Kutz, J. Adaptive dimensionality-reduction for time-stepping in differential and partial differential equations, *Numerical Mathematics: Theory, Methods and Applications*, **10**(4) 2017, 872–894.
- [45] Galbally, D., Fidkowski, K., Willcox, K., Ghattas, O. Nonlinear model reduction for uncertainty quantification in large-scale inverse problems, *International Journal for Numerical Methods in Engineering*, **81**(12) 2010, 1581–1608.
- [46] Ghavamian, F., Tiso, P., Simone, A. POD-DEIM model order reduction for strain-softening viscoplasticity, *Computer Methods in Applied Mechanics and Engineering*, **317** 2017, 458–479.
- [47] Grle, C., Hinze, M. POD reduced-order modeling for evolution equations utilizing arbitrary finite element discretizations, *Advances in Computational Mathematics*, **44** 2018, 1941–1978.
- [48] Haasdonk, B., Dihlmann, M., Ohlberger, M. A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space, *Mathematical and Computer Modelling of Dynamical Systems*, **17**(4) 2011, 423–442.
- [49] Hoang, K., Khoo, B., Liu, G., Nguyen, N., Patera, A. Rapid identification of material properties of the interface tissue in dental implant systems using reduced basis method, *Inverse Problems in Science and Engineering*, **21**(8) 2013, 1310–1334.
- [50] Hughes, T. J. Unconditionally stable algorithms for nonlinear heat conduction, *Computer Methods in Applied Mechanics and Engineering*, **10**(2) 1977, 135–139.
- [51] Jiang, R., Durlofsky, L.J. Implementation and detailed assessment of a GNAT reduced-order model for subsurface flow simulation, *Journal of Computational Physics*, **379** 2019, 192–213.

- [52] Jørgensen, B., Sørensen, J., Brøns, M. Low-dimensional modeling of a driven cavity flow with two free parameters, *Theoretical and Computational Fluid Dynamics*, **16**(4) 2003, 299–317.
- [53] Kim, T., James, D. L. Skipping steps in deformable simulation with online model reduction, *ACM Transactions on Graphics*, **28**(5) 2009, 1-9.
- [54] Klein, C., Saut, JC. A numerical approach to blow-up issues for dispersive perturbations of Burgers’ equation, . *Physica D: Nonlinear Phenomena*, **295-296** 2015, 46–65.
- [55] Knezevic, D.J., Nguyen, N.C., Patera, A.T. Reduced basis approximation and a posteriori error estimation for the parametrized unsteady Boussinesq equations, *Mathematical Models and Methods in Applied Sciences*, **21**(7) 2011, 1415–1442.
- [56] Klock, R. J., Cesnik, C. E. Nonlinear thermal reduced-order modeling for hypersonic vehicles, *AIAA Journal*, **55**(7) 2017, 1-11.
- [57] Kunisch, K., Volkwein, S. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical Analysis* **40**(2) 2002, 492–515.
- [58] Kutz, JN., Brunton, SL., Brunton, BW., Proctor, JL. Dynamic mode decomposition: data-driven modeling of complex systems, Philadelphia, PA: Society for Industrial and Applied Mathematics, 2016.
- [59] Linares, F., Pilod, D., Saut, JC. Dispersive perturbations of Burgers and hyperbolic equations i: Local theory, *SIAM Journal on Mathematical Analysis*, **46**(2) 2014, 1505–1537.
- [60] Liu, G. R. A generalized gradient smoothing technique and the smoothed bilinear form for Galerkin formulation of a wide class of computational methods, *International Journal of Computational Methods*, **5**(2) 2008, 199-236.
- [61] Lucia, D.J., Beran, P.S., Silva, W.A. Reduced-order modeling: new approaches for computational physics. *Progress in Aerospace Sciences*, **40**(1) 2004, 51–117.
- [62] Lucia, D.J., King, P.I., Beran, P.S. Reduced order modeling of a two-dimensional flow with moving shocks. *Computers & Fluids*, **32**(7) 2003, 917–938.

- [63] Molinet, L., Pilod, D., Vento, S. On well-posedness for some dispersive perturbations of Burgers' equation, *Annales de l'Institut Henri Poincaré (C) Nonlinear Analysis*, **35**(7) 2018, 1719–1756.
- [64] Mordhorst, M., Strecker, T., Wirtz, D., Heidlauf, T., Rhrle, O. POD-DEIM reduction of computational EMG models, *Journal of Computational Science*, **19** 2017, 86–96.
- [65] Mu, L., Ye, X. A finite volume method for solving Navier-Stokes problems, *Nonlinear Analysis: Theory, Methods & Applications*, **74**(17) 2011, 6686–6695.
- [66] Okumura, S., Suzuki, Y., Takeuchi, I. Quick sensitivity analysis for incremental data modification and its application to leave-one-out CV in linear classification problems, *Proceedings 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, **New York, NY, USA** 2015, 885–894.
- [67] Oxberry, G. M., Kostova-Vassilevska, T., Arrighi, W., Chand, K. Limited memory adaptive snapshot selection for proper orthogonal decomposition, *International Journal for Numerical Methods in Engineering*, **109**(2) 2017, 198–217.
- [68] Pando, M.F.D., Schmid, P.J., Sipp, D. Nonlinear model-order reduction for compressible flow solvers using the discrete empirical interpolation method, *Journal of Computational Physics*, **324** 2016, 194–209.
- [69] Peherstorfer, B., Butnaru, D., Willcox, K., Bungartz, HJ. Localized discrete empirical interpolation method, *SIAM Journal on Scientific Computing*, **36**(1) 2014, A168–A192.
- [70] Peherstorfer, B., Willcox, K. Dynamic data-driven reduced-order models, *Computer Methods in Applied Mechanics and Engineering*, **291** 2015, 21–41.
- [71] Peherstorfer, B., Willcox, K. Online adaptive model reduction for nonlinear systems via low-rank updates, *SIAM Journal on Scientific Computing*, **37**(4) 2015, A2123–A2150.
- [72] Peherstorfer, B., Willcox, K. Data-driven operator inference for nonintrusive projection-based model reduction, *Computer Methods in Applied Mechanics and Engineering*, **306** 2016, 196–215.
- [73] Peng, L., Mohseni, K. Nonlinear model reduction via a locally weighted POD method, *International Journal for Numerical Methods in Engineering*, **106**(5) 2016, 372–396.

- [74] Qian, J., Wang, Y., Song, H., Pant, K., Peabody, H., Ku, J., Butler, C. D. Projection-based reduced-order modeling for spacecraft thermal analysis, *Journal of Spacecraft and Rockets*, **52**(3) 2015, 978–989.
- [75] Rapun, M. L., Vega, J. M. Reduced order models based on local POD plus Galerkin projection, *Journal of Computational Physics*, **229**(8) 2010, 3046–3063.
- [76] Rathinam, M., Petzold, L. R. A new look at proper orthogonal decomposition, *SIAM Journal of Numerical Analysis*, **41**(5) 2003, 1893–1925.
- [77] Rewienski, M., White, J. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **22**(2) 2003, 155–170.
- [78] Rewienski, M., White, J. Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations, *Linear Algebra and Its Applications*, **415**(2) 2006, 426–454.
- [79] Robertson, E. D., Wang, Y., Pant, K., Grismer, M. J., Camberos, J. A. A flow feature detection framework for large-scale computational data based on incremental proper orthogonal decomposition and data mining, *International Journal of Computational Fluid Dynamics*, **32**(6-7), 261–277.
- [80] San, O., Maulik, R., Ahmed, M. An artificial neural network framework for reduced order modeling of transient flows, *Communications in Nonlinear Science and Numerical Simulation*, **77** 2019, 271–287.
- [81] Sheng, Y., Zhang, T., Jiang, ZZ. A stabilized finite volume method for the stationary Navier–Stokes equations, *Chaos, Solitons & Fractals*, **89** 2016, 363–372.
- [82] Sheng, Y., Zhang, T. The finite volume method for two-dimensional Burgers’ equation, *Personal Ubiquitous Computing*, **22**(5-6) 2018, 1133–1139.
- [83] Shinde, V., Longatte, E., Baj, F., Hoarau, Y., Braza, M. A Galerkin-free model reduction approach for the Navier-Stokes equations, *Journal of Computational Physics*, **309** 2016, 148–163.
- [84] Sirisup, S., Karniadakis, G. E., Xiu, D., Kevrekidis, I. G. Equation free/Galerkin-free POD-assisted computation of incompressible flows, *Journal of Computational Physics*, **207**(2) 2005, 568–587.

- [85] Sirovich, L. Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations. III - Dynamics and scaling, *Quarterly of Applied Mathematics*, **45**(3) 1987, 561–571, 573–590.
- [86] Everson, R., Sirovich, L. Karhunen–loève procedure for gappy data, *Journal of the optical Society of America A*, **12**(8) 1995, 1657–1664.
- [87] Stefanescu, R., Navon, I. POD/DEIM nonlinear model order reduction of an ADI implicit shallow water equations model, *Journal of Computational Physics*, **237** 2013, 95–114.
- [88] Swischuk, R., Mainini, L., Peherstorfer, B., Willcox, K. Projection-based model reduction: Formulations for physics-based machine learning, *Computes & Fluids*, **179** 2019, 704–717.
- [89] Terragni, F., Valero, E. and Vega, J. M. Local POD plus galerkin projection in the unsteady lid-driven cavity problem, *SIAM Journal on Scientific Computing*, **33**(6) 2011, 3538–3561.
- [90] Toro, EF. Riemann solvers and numerical methods for fluid dynamics: a practical introduction, *Springer Science & Business Media* 2013.
- [91] Ullmann, S., Rotkvic, M. and Lang, J. POD-Galerkin reduced-order modeling with adaptive finite element snapshots, *Journal of Computational Physics*, **325**, 244–258.
- [92] Vendl, A., Faßbender, H. Projection-based model order reduction for steady aerodynamics, Berlin, Heidelberg: Springer Berlin Heidelberg 2013, Page 151–166.
- [93] Wang, Y., Song, H. and Pant, K. A reduced-order model for whole-chip thermal analysis of microfluidic lab-on-a-chip systems, *Microfluidics and Nanofluidics*, **16**(1) 2014, 369–380.
- [94] Wang, Y., Navon, IM., Wang, X., Cheng, Y. 2D Burgers equation with large reynolds number using POD/DEIM and calibration. *International Journal for Numerical Methods in Fluids*, **82**(12) 2016, 909–931.
- [95] Wang, Z., Akhtar, I., Borggaard, J., Iliescu, T. Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison, *Computer Methods in Applied Mechanics and Engineering*, **237-240** 2012, 10–26.

- [96] Wang, Z., McBee, B. and Iliescu, T. Approximate partitioned method of snapshots for POD, *Journal of Computational and Applied Mathematics*, **307** 2016, 374–384.
- [97] Willcox, K. and Peraire, J. Balanced model reduction via the proper orthogonal decomposition, *AIAA Journal*, **40**(11) 2002, 2323–2330.
- [98] Williams, M., Schmid, P. and Kutz, J. Hybrid reduced-order integration with proper orthogonal decomposition and dynamic mode decomposition, *Multiscale Modeling and Simulation*, **11**(2) 2013, 522–544.
- [99] Yan, B. Minimizing L_∞ -norm functional on divergence-free fields, *Annales de l’Institut Henri Poincaré (C) Nonlinear Analysis*, **28**(3) 2011, 325–355.
- [100] Yang, Y. J. and Shen, K. Y. Nonlinear heat-transfer macromodeling for MEMS thermal devices, *Journal of Micromechanics and Microengineering*, **15**(2) 2005, 408–418.
- [101] Yang, Y. J. J. and Kuo, C. W. Generating scalable and modular macromodels for microchannels using the Galerkin-based technique, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **27**(9) 2008, 1545–1554.
- [102] Zhao, G., Yu, X., Zhang, R. The new numerical method for solving the system of two-dimensional Burgers’ equations, *Computers & Mathematics with Applications*, **62**(8) 2011, 3279–3291.
- [103] Zimmermann, R., Gortz, S. Nonlinear reduced order models for steady aerodynamics, *Procedia Computer Science*, **1**(1) 2010, 165–174.
- [104] Zimmermann, R. and Willcox, K. An accelerated greedy missing point estimation procedure, *SIAM Journal on Scientific Computing*, **38**(5) 2016, A2827–A2850.
- [105] Zimmermann, R., Peherstorfer, B., Willcox, K. Geometric subspace updates with applications to online adaptive nonlinear model reduction, *SIAM Journal on Matrix Analysis and Applications*, **39**(1) 2018, 234–261.