MODEL REDUCTION FOR ALLEN-CAHN EQUATION

BY

MR. CHUTIPONG DECHANUBEKSA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN MATHEMATICS
FACULTY OF SCIENCE AND TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2015

MODEL REDUCTION FOR ALLEN-CAHN EQUATION

BY

MR. CHUTIPONG DECHANUBEKSA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN MATHEMATICS
FACULTY OF SCIENCE AND TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2015

THAMMASAT UNIVERSITY

FACULTY OF SCIENCE AND TECHNOLOGY

THESIS

BY

MR. CHUTIPONG DECHANUBEKSA

ENTITED

MODEL REDUCTION FOR ALLEN-CAHN EQUATION

was approved as partial fulfillment of the requirements for

the degree of Master of Science in Mathematics

on January 5, 2016

Chairman

(Assistant Professor Archara Pacheenburawana, Ph.D.)

Member and Advisor

(Saifon Chaturantabut, Ph.D.)

Member

(Jutarat Kongson, Ph.D.)

Member

(Adoon Pansuwan, Ph.D.)

Dean

(Associate Professor Pakorn Sermsuk)

Thesis Title	MODEL REDUCTION FOR ALLEN-CAHN EQUATION
Author	Mr. Chutipong Dechanubeksa
Degree	Master of Science in Mathematics
Department/Faculty/University	Department of Mathematics and Statistics Faculty of Science and Technology Thammasat University
Thesis Advisor	Saifon Chaturantabut, Ph.D.
Academic Year	2015

ABSTRACT

Many mathematical models were constructed to describe the physical phenomena in nature. However, solving for the numerical result of these models generally has high computational complexity, which may be impossible to simulate in suitable time. In this thesis, we apply a model order reduction methods called Proper Orthogonal Decomposition (POD), Discrete Empirical Interpolation Method (DEIM), and Gappy Proper Orthogonal Decomposition (GPOD) on Allen-Cahn equation, which has been widely used in material science and fluid dynamics. These methods can be used to efficiently decrease the dimension of the original system and hence they can reduce the computational cost in the simulation. The numerical tests demonstrate that the resulting reduced systems can accurately give approximate solutions to the original system with various parameter values.

Keywords: Model order reduction, Allen-Cahn equation, Proper Orthogonal Decomposition, Discrete Empirical Interpolation Method, Gappy Proper Orthogonal Decomposition

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor Dr. Saifon Chaturantabut, for the constructive guidance, good suggestions, understanding, support, and encouragement during the time of research and writing of this thesis. She helped me throughout the dissertation procedure. This thesis would never be complete without her.

I would like to express my thanks to the members of my thesis committee: Asst. Prof. Dr. Archara Pacheenburawana, Dr. Jutarat Kongson and Dr. Adoon Pansuwan, for their constructive comments and helpful advice.

I would also like to thank all teachers in the Department of Mathematics and Statistics, Faculty of Science and Technology Thammasat University, for useful knowledge and good advice during my study time here.

Finally, I would like to thank my parents, my sister, and my brother, for their love, understanding, encouragement and financial support. I want to thank my friends, for a good relationship throughout my academic life.

Mr. Chutipong Dechanubeksa

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the mathematical and computational field, model order reduction (MOR) is a technique for constructing efficient lower dimensional models for reducing the computational cost of large scale dynamical systems from partial differential equations (PDEs) in numerical simulation. Many modern mathematical models are useful for describing physical phenomena in real life. However, the process for solving the numerical solution of these mathematical models often has high computational complexity. To overcome this problem, A reduced order model can be used to approximate the original system while preserving the main properties and containing the important features of the original model.

1.1 Motivation and Objective

This thesis considers nonlinear partial differential equation called Allen-Cahn equation. This equation is used in mathematical physics, to describe the process of phase separation. The Allen-Cahn equation has been widely used in many applications, such as image analysis [10, 24], crystal growth [59], and motion by mean curvature flow [26]. In particular, it has become a basic model equation for the diffuse interface approach developed to study phase transitions and interfacial dynamics in material science [21]. The focus of this thesis is to study and develop efficient reduced systems for Allen-Cahn equation, which can decrease computational complexity of full discretized system in numerical simulation. The advantage of a reduced model often lies in the decrease of simulation time and the provision accurate results when compared to the high dimensional system. There are many MOR techniques such as Balance Truncation [42],

Krylov subspace MOR methods [27], Projection based MOR [4], and the Moment-matching method [23]. The Allen-Cahn equation is the nonlinear PDEs system. The computational cost depends on the dimension of a system. Here we use MOR techniques called Proper Orthogonal Decomposition (POD), Discrete Empirical Interpolation Method (DEIM), and Gappy POD (GPOD) to construct the reduced model. POD approach generally gives high accurate reduced models with much smaller dimensions by generating a set of optimal basis that represents a given data set with minimum error. DEIM approach can efficiently reduce dimensions of nonlinear term in the reduced model. We apply GPOD approach for reducing order nonlinear term. This method can provide the approximation that minimizes error in the Euclidean norm. POD and DEIM have been used to construct the reduced models in many applications, such as photovoltaic modules [46], the shallow water equation model [56], Navier-Stokes equations [62] and Drift-Diffusion Equations [32]. GPOD has been applied as a data repair approach in many applications, such as [12, 43, 39]. The following are the objectives of this thesis.

- Construct the reduced model of the Allen-Cahn equation by using POD-DEIM.
- Test parameter variation of the reduced models for the Allen-Cahn equation.
- Use new method called POD-GPOD for constructing reduced model and compare with reduced model by using POD-DEIM.

1.2 Literature review

1.2.1 Allen-Cahn equation

The Allen-Cahn equation was first introduced by Cahn and Allen to describe the motion of anti-phase boundaries in metallic alloys [3]. Feng and Prohl [26] used the equation, as a simple model for phase separation of metallic components within a binary alloy at a fixed temperature. Beně, Chalupeckýý and Mikula [10] proposed an algorithm of pattern recovery (image segmentation), based on the solution of this equation. Yang [63] proposed a Semi-Implicit method of this equation. A nonlocal version of this equation which may model a variety of physical and biological phenomena, was derived

by Bates, Brown and Han [9]. Cabrĕ [13] established the uniqueness of a saddle-shaped solution to this equation. Several works have investigated numerical analysis of Allen-Cahn equation [64, 22, 10, 24, 59].

1.2.2 Model Order Reduction (MOR)

Model order reduction (MOR) was developed in the area of systems and control theory to reduce their complexity, while preserving their input-output behavior. There are several works on MOR such as [37, 53, 5, 6, 14, 19, 61]. Numerical solutions of large scale dynamical systems can be computationally expensive, because both space and time discretizations are required to meet the desired accuracy. However, in many cases much of the solution information lies within a subspace whose dimension is significantly lower than the original full dimension used in the discretization. The purpose of reduced order modeling (ROM) is to construct a low dimensional model that preserves the necessary information from the full discretized system, while saving computational time and memory storage .

Noor et al.[45] presented a reduced-basis technique and a computational algorithm within the context for analyzing nonlinear structure. Peterson [49] demonstrated that the reduced basis method can efficiently approximate the solution of incompressible viscous flow. Ravindran [36] demonstrated a reduced-order modeling approach for simulation and control of incompressible flows. Gunzburger, Peterson, and Shadid [31] focused on reduced order modeling of time-dependent PDEs with multiple parameters on the proper orthogonal decomposition (POD) approach to the reduced order model. Carlberg and Ferhat [16] used a compact POD to compute a reduced basis for an optimization-oriented reduced order model and applied this to compute a reduced basis for model reduction of static systems [17]. There are several MOR techniques used to model reduction, such as Balance Truncation [42], Krylov subspace MOR methods [27], projection based MOR [4] and Moment-matching methods [23]. In this work, we focus on POD combined with the Discrete Empirical Interpolation Method (DEIM) and Gappy POD for constructing efficiently reduced order system.

1.2.3 Proper Orthogonal Decomposition (POD) and Discrete Empirical Interpolation Method (DEIM)

POD is a popular method for model reduction, first proposed by Lumley [41], which is also known as Karhunen-Loève decomposition [40] or principal component analysis [34]. It provides a technique for analyzing multidimensional data. This method constructs an orthonormal basis for representing the given data in a certain least squares optimal sense. The basic properties of the POD method as it is applied to data compression and model reduction of finite dimensional linear systems have been studied [51]. Prajna [50] purposed sufficient conditions for preserving stability in POD model reduction. POD has been widely applied in many applications such as data analysis, data compression and model reduction in various fields of engineering and science. Applications of POD include image processing [52], data compression, signal analysis [1], turbulence modeling [33], control of fluids [28], electrical power grids [47], and modeling and control of chemical reaction systems [54, 55].

Discrete Empirical Interpolation Method (DEIM) is a technique for approximating nonlinear terms of a dynamical system to reduce computational complexity. The DEIM approach was introduced by Chaturantabut and Sorensen [19], and approximates a nonlinear function by combining projection with interpolation. This approach is a discrete variant of the empirical interpolation method (EIM) introduced by Barrault et al. [8], which was originally described for an empirically derived infinite-dimensional function space. The DEIM approximation was extended to localization (LDEIM) [48]. There are other nonlinear techniques for approximating nonlinear terms, such as the Gauss-Newton with approximated tensors (GNAT) method [15], the best points interpolation method [44] and Missing Point Estimation [7]. There are many works combining a reduced model by POD with DEIM such as the photovoltaic module [46], the shallow water equations model [56], the Navier-Stokes equations [62] and Drift-Diffusion Equations [32].

1.2.4 Gappy Proper Orthogonal Decomposition (GPOD)

Gappy POD (GPOD) is a data repair approach, which was first purposed by Everson and Sirovich [25]. Thanh, Damodaran, and Willcox [12] presented this in context of fluid dynamic application. Bos et al., [11] presented this in context of solving the numerical simulation of nonlinear system. Willcox [60] extented GPOD to handle unsteady flow reconstruction problem. Lee and Mavris [39] developed this method for various problem in aerospace engineering. Murray and Ukeiley [43] applied GPOD in context of particle image velocimetry (PIV) data for subsonic cavity flow.

1.3 Overview of Thesis

In this thesis, we focus on the Allen-Cahn equation, which has been widely used in material science and fluid dynamics. The main goal of this thesis is to construct the reduced system of this equation.

In Chapter 2, we introduce three model reduction methods i.e. POD, DEIM, and GPOD for constructing a reduced model. This chapter reviews the process for each of these model reduction methods.

In Chapter 3, we apply the reduced model approaches in Chapter 2 for the Allen-Cahn equation. This chapter uses three model reduction methods to reduce order of the full order discretized Allen-Cahn system, which is obtained by the finite difference method.

In Chapter 4, we present numerical examples for model reduction of the Allen-Cahn equation, and test parameter variation. This chapter consists of numerical examples for homogeneous boundary conditions, and numerical examples for non-homogeneous boundary conditions with different initial conditions.

In Chapter 5, we conclude all of work on this thesis and future work.

CHAPTER 2

METHODS

In this chapter, we discuss the model order reduction (MOR) techniques used in this thesis. There are several excellent works on MOR such as [37, 53, 5, 6, 14, 19, 61]. MOR techniques used in this thesis are Proper Orthogonal Decomposition (POD), the Discrete Empirical Interpolation Method (DEIM) ,and Gappy Proper Orthogonal Decomposition (GPOD). These methods are used to construct the reduced models in many applications, such as the photovoltaic module [46], the shallow water equations model [56], the Navier-Stokes equations [62], the particle image velocimetry (PIV) data for subsonic cavity flow [43]. and the Drift-Diffusion Equations [32].

The model reduction process use a basic concept of singular value decomposition (SVD). Note that every matrix $\mathbf{A} \in \mathbb{C}^{m\times n}$ has a singular value decomposition, as given in the following theorem.

Theorem[29] Let $\mathbf{A} \in \mathbb{C}^{m\times n}$. Then there exist unitary matrices $\mathbf{V} \in \mathbb{C}^{m\times m}$, $\mathbf{W} \in \mathbb{C}^{n\times n}$, and a diagonal matrix $\mathbf{\Sigma} \in \mathbb{C}^{m\times n}$ with nonnegative diagonal entries $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_{min\{m,n\}} \geq 0$, such that

$$\mathbf{A} = \mathbf{V}\mathbf{\Sigma}\mathbf{W}^*.$$

where $\mathbf{W}^*$ is conjugate transpose of $\mathbf{W}$.

If the rank of $\mathbf{A}$ is r, then r is the number of nonzero singular values [58]. Moreover, the nonzero singular values of $\mathbf{A}$ are the square roots of the nonzero eigenvalues of $\mathbf{A}^*\mathbf{A}$ or $\mathbf{A}\mathbf{A}^*$ when the matrices have the same nonzero eigenvalues.

In particular,

$$\mathbf{A}^*\mathbf{A} = (\mathbf{V}\mathbf{\Sigma}\mathbf{W}^*)^*(\mathbf{V}\mathbf{\Sigma}\mathbf{W}^*) = \mathbf{W}\mathbf{\Sigma}^\mathbf{2}\mathbf{W}^*$$

is eigendecomposition of $\mathbf{A}^*\mathbf{A}$ and

$$\mathbf{A}\mathbf{A}^* = (\mathbf{V}\mathbf{\Sigma}\mathbf{W}^*)(\mathbf{V}\mathbf{\Sigma}\mathbf{W}^*)^* = \mathbf{V}\mathbf{\Sigma}^\mathbf{2}\mathbf{V}^*$$

is eigendecomposition of $\mathbf{A}\mathbf{A}^*$. This means that the columns of $\mathbf{W}$ are the eigenvectors of $\mathbf{A}^*\mathbf{A}$, and the eigenvalues of $\mathbf{A}^*\mathbf{A}$ are the squares of the singular values of $\mathbf{A}$. The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{A}\mathbf{A}^*$, and the eigenvalues of $\mathbf{A}\mathbf{A}^*$ are the squares of the singular values of $\mathbf{A}$.

2.1 Proper Orthogonal Decomposition (POD)

Proper Orthogonal Decomposition (POD) is a model reduction technique for constructing a low dimensional model of a dynamic system. Its main idea is similar to the singular value decomposition (SVD). This method can be used to find a low-dimensional basis efficiently. The POD basis is used in the projection for reducing the dimension of the original full-order system.

Consider the following simplified nonlinear dynamical system in matrix form, which can arise from the discretization of PDEs.

$$\mathbf{A}\mathbf{y}(t_{j+1}) = \mathbf{B}\mathbf{y}(t_j) + \mathbf{F}(\mathbf{y}(t_j)). \tag{2.1}$$

Therefore

$$\mathbf{y}(t_{j+1}) = \mathbf{A}^{-1}\mathbf{B}\mathbf{y}(t_j) + \mathbf{A}^{-1}\mathbf{F}(\mathbf{y}(t_j)), \tag{2.2}$$

where $\mathbf{A}$ and $\mathbf{B}$ are finite difference matrices, which are constructed from discretization by the finite difference method (see Chapter 3), $\mathbf{y}$ is state variable, t is independent variable, and $\mathbf{F}$ is a nonlinear vector-valued function.

In general, to obtain an accurate numerical solution, the dimension n of the discretized system has to be *large*. This can result in a high computational cost.

2.1.1 POD Basis

In this subsection, we construct the POD basis $\mathbf{V}_k = [\mathbf{v}_1, ..., \mathbf{v}_k]$ with $k \ll n$ from the solution of full-order system (2.2) called snapshots in section 2.1. This was used to construct the reduced system by POD introduced in the next subsection.

Suppose $\mathbf{V}_r = [\mathbf{v}_1, ..., \mathbf{v}_r]$ is an orthonormal basis for $\mathbf{Y}$ =span$\{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{n_s}\}$. Then, each snapshot $\mathbf{y}_j$ for $j = 1, ..., n_s$ can be written as

$$\mathbf{y}_j = \mathbf{V}_r \mathbf{V^T}_r \mathbf{y}_j$$

where r =dim$(\mathbf{Y})$, and n_s is the number of snapshots of (2.2).

We will construct an orthonormal basis of dimension k by using the singular value decomposition. Suppose $\{\mathbf{y}_i\}_{i=1}^k$ be the best approximate in the form

$$\tilde{\mathbf{y}}_j = \mathbf{V}_k \mathbf{V^T}_k \mathbf{y}_j$$

to the snapshot $\mathbf{y}_j$ for all $j = 1, ..., n_s$. Then it can be shown [18] that $\mathbf{V}_k = [\mathbf{v}_1, ..., \mathbf{v}_k]$ consists of the first k columns of the left singular vectors corresponding to the k largest singular values, and $\mathbf{V}_k$ is called POD basis of dimension k.

Definition[18] A POD basis of dimension $k < r$ is a set of orthogonal basis $\mathbf{V}_k = [\mathbf{v}_1, ..., \mathbf{v}_k]$ which is the solution of

$$\min \sum_{j=1}^{n_s} \left\| \mathbf{y}_j - \mathbf{V}_k \mathbf{V^T}_k \mathbf{y}_j \right\|_2^2.$$

We can construct the POD basis $\mathbf{V}_k$ from the following algorithm [61], derived from the solution of the full system for constructing snapshots matrix $\mathbf{S} = [\mathbf{y}(t_1), ..., \mathbf{y}(t_{n_s})]$.

Algorithm 1 Proper Orthogonal Decomposition (POD)

INPUT : $\mathbf{S} = [\mathbf{y}(t_1), ..., \mathbf{y}(t_{n_s})] \in \mathbb{R}^{n \times n_s}$

OUTPUT : $\mathbf{V}_k = [\mathbf{v}_1, ..., \mathbf{v}_k] \in \mathbb{R}^{n \times k}$

1. Perform the singular value decomposition (SVD) of $\mathbf{S} = \mathbf{V}\mathbf{\Sigma}\mathbf{W}^T$ to produce orthogonal matrices $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_r] \in \mathbb{R}^{n \times r}$ and $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_r] \in \mathbb{R}^{n_s \times r}$ and diagonal matrix $\mathbf{\Sigma}$ =diag$(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{r \times r}$, where r is the rank of $\mathbf{S}$.

2. Set a threshold to select the k largest modes from the diagonal matrix $\mathbf{\Sigma}$.

3. Select the columns in matrix $\mathbf{V}$ which correspond to modes selected in 2 to generate the POD basis $\{\mathbf{v}_1, ..., \mathbf{v}_k\} \in \mathbb{R}^{n \times k}$ and construct projection matrix $\mathbf{V}_k$.

2.1.2 Reduced Order Model (ROM)

To use the POD basis $\mathbf{V}_k$ with $k \ll n$ for constructing the reduced system by POD, we approximate the full order solution by $\mathbf{y} \approx \mathbf{V}_k\tilde{\mathbf{y}}$, so we have

$$\mathbf{V}_k\tilde{\mathbf{y}}(t_{j+1}) = \mathbf{A}^{-1}(\mathbf{B}\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) + \mathbf{A}^{-1}(\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j))). \tag{2.3}$$

Next, by applying the projection along the subspace generated by columns of $\mathbf{V}_k$ which form an orthonormal basis $\left(\mathbf{V}_k^T\mathbf{V}_k = \mathbf{I} \in \mathbb{R}^{k \times k}\right)$, we have

$$\tilde{\mathbf{y}}(t_{j+1}) = (\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{B}\mathbf{V}_k)\tilde{\mathbf{y}}(t_j) + \mathbf{V}_k^T\mathbf{A}^{-1}(\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j))) \tag{2.4}$$

with $\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{B}\mathbf{V}_k = \mathbf{C}$ and where $\mathbf{C} \in \mathbb{R}^{k \times k}$ can be precomputed and used in each iteration.

Finally, we obtain the reduced system by POD (POD reduced system)

$$\tilde{\mathbf{y}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\mathbf{precomputed}:k \times k}\ \underbrace{\tilde{\mathbf{y}}(t_j)}_{k \times 1} + \underbrace{\mathbf{V}_k^T\mathbf{A}^{-1}}_{\mathbf{precomputed}:k \times n}\ \underbrace{\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j))}_{n \times 1}. \tag{2.5}$$

Since computing the nonlinear term of (2.5) still depends on the *large* dimension n, which can result in high computational complexity. In the next section, we use DEIM to approximate this nonlinear term.

2.2 Discrete Empirical Interpolation Method (DEIM)

The Discrete Empirical Interpolation Method (DEIM) is a technique for constructing a reduced order approximation in nonlinear terms of a dynamical system. The main concept of DEIM is based on a greedy algorithm. This method uses the nonlinear function of the full system to generate the snapshots used in the POD algorithm. This basis of nonlinear snapshots is used to find the interpolation indices in the DEIM algorithm for approximating the nonlinear term.

2.2.1 DEIM: Algorithm for selecting Interpolation Indices

The goal of this subsection is to reduce the computational cost of the nonlinear term of the reduced system (2.5) constructed by POD. We started by collecting snapshots from a nonlinear function of the full-order system for constructing the snapshot matrix $\mathbf{F} = \{\mathbf{F}(\mathbf{y}(t_1)), ..., \mathbf{F}(\mathbf{y}(t_{n_s}))\}$. Next, we use POD algorithm to find the POD basis $\{\mathbf{u}_1, ..., \mathbf{u}_m\}$ with $m \ll n$, called DEIM basis. Finally, we used this basis to find the interpolation indices $[\varrho_1, ..., \varrho_m]$ by the following DEIM algorithm [19].

Algorithm 2 Discrete Empirical Interpolation Method (DEIM)

INPUT : $\{\mathbf{u}_i\}_{i=1}^m \subset \mathbb{R}^n$ linearly independent

OUTPUT : $\vec{\varrho} = [\varrho_1, ..., \varrho_m] \in \mathbb{R}^m$

1. $\varrho_1 = \text{argmax}_{j=1,...,n}\{|u_{j_1}|\}$
2. $\mathbf{U} = |\mathbf{u}_1|, \mathbf{P} = [\mathbf{e}_{\varrho_1}], \vec{\varrho} = [\varrho_1]$
3. for $i = 2$ to m do
4. Solve $(\mathbf{P}^T\mathbf{U})\mathbf{c} = \mathbf{P}^T\mathbf{u}_i$ for $\mathbf{c}$
5. $\mathbf{r} = \mathbf{u}_i - \mathbf{U}\mathbf{c}$
6. $\varrho_i = \text{argmax}_{j=1,...,n}\{|r_{j_i}|\}$
7. $\mathbf{U} \leftarrow \begin{bmatrix}\mathbf{U} & \mathbf{u}_i\end{bmatrix}, \mathbf{P} \leftarrow \begin{bmatrix}\mathbf{P} & \mathbf{e}_{\varrho_i}\end{bmatrix}, \vec{\varrho} \leftarrow \begin{bmatrix}\vec{\varrho} \\ \varrho_i\end{bmatrix}$
8. end for

The input to the DEIM algorithm is the linearly independent basis vector $\{\mathbf{u}_i\}_{i=1}^m$. The output is the vector $\vec{\varrho}$ that contains the indices for unity components in

matrix $\mathbf{P}$ with $\mathbf{P} = [\mathbf{e}_{\varrho_1}, ..., \mathbf{e}_{\varrho_m}] \in \mathbb{R}^{n\times m}$, where $e_{\varrho_i} = [0 \ ... \ 0 \ 1 \ 0 \ ... \ 0]^T \in \mathbb{R}^n$ is the ϱ_ith column of the identity matrix $\mathbf{I} \in \mathbb{R}^{n\times n}$ for $i = 1, ..., m$.

We start with lines 1-2 of the algorithm. In line 1, the index ϱ_1 are determined by the largest absolute value of element in $\mathbf{u}_1$. In line 2, we define $\mathbf{U} = \mathbf{u}_1 \in \mathbb{R}^{n\times 1}$ and $\mathbf{P} = \mathbf{e}_{\varrho_1} \in \mathbb{R}^{n\times 1}$. The selected index is stored in $\varrho_1 \in \mathbb{R}$.

In line 3, we start the iterative process for $i = 2, ..., m$ and solve $\left(\mathbf{P}^T\mathbf{U}\right)\mathbf{c} = \mathbf{P}^T\mathbf{u}_i$ to find the constant c in line 4. In line 5, we consider the residual $\mathbf{r} = \mathbf{u}_i - \mathbf{U}\mathbf{c}$ and determine the next index by finding the largest component in $|\mathbf{r}|$. In line 6, the index ϱ_i are determined by the largest absolute value of element in $\mathbf{r}$. In line 7, this index is included in the matrix $\mathbf{P}$ by updating it with a new vector $\mathbf{e}_{\varrho_i}$. From the iterative process, we obtain $\mathbf{U}, \mathbf{P}$ and $\vec{\varrho}$. Note that the matrix $\mathbf{P}^T\mathbf{U}$ is a nonsingular matrix as proved in [18].

2.2.2 Nonlinear Approximation

We approximate the nonlinear term of (2.5) by the matrix $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_m] \in \mathbb{R}^{n\times m}$ with $m \ll n$ and $\mathbf{P} = [\mathbf{e}_{\varrho_1}, ..., \mathbf{e}_{\varrho_m}] \in \mathbb{R}^{n\times m}$, where $e_{\varrho_i} = [0 \ ... \ 0 \ 1 \ 0 \ ... \ 0]^T \in \mathbb{R}^n$ is the ϱ_ith column of the identity matrix $\mathbf{I} \in \mathbb{R}^{n\times n}$ for $i = 1, ..., m$ in the previous subsection. This is used for constructing the reduced system by POD-DEIM. Consider nonlinear term from POD,

$$\mathbf{N}(\mathbf{y}(t_j)) = \mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{F}\left(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)\right). \tag{2.6}$$

Suppose that $\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) = \mathbf{f}(t_j)$ and define the approximation

$$\mathbf{f}(t_j) \approx \mathbf{U}\mathbf{c}(t_j) \tag{2.7}$$

where $\mathbf{c}(t_j)$ can be determined by selecting m rows from the system (2.7), i.e. $\mathbf{c}(t_j)$ can be computed by solving the system

$$\mathbf{P}^T\mathbf{f}(t_j) = (\mathbf{P}^T\mathbf{U})\mathbf{c}(t_j), \tag{2.8}$$

where multiplying by $\mathbf{P}^T$ is equivalent to selecting m rows in this system (2.8).

Since $\mathbf{P}^T\mathbf{U}$ is nonsingular [18], the approximation of (2.7) is

$$\mathbf{f}(t_j) \approx \mathbf{U}\mathbf{c}(t_j) = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{f}(t_j). \tag{2.9}$$

From $\mathbf{f}(t_j) = \mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j))$, (2.9) can be written as

$$\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)). \tag{2.10}$$

Since $\mathbf{F}$ has to be evaluated for the original dimension before being interpolated by the matrix $\mathbf{P}$, we use the fact that $\mathbf{F}$ is a componentwise function, which implies $\mathbf{P}^T\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) = \mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j))$ and

$$\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) \approx \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j)). \tag{2.11}$$

Therefore (2.6) can now be written as

$$\mathbf{N}(\mathbf{y}(t_j)) = \mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j)). \tag{2.12}$$

From the POD reduced system (2.5), we obtain

$$\tilde{\mathbf{y}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\mathbf{precomputed}:k\times k}\underbrace{\tilde{\mathbf{y}}(t_j)}_{k\times 1} + \underbrace{\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}}_{k\times m}\underbrace{\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j))}_{m\times 1}, \tag{2.13}$$

where $\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1} = \mathbf{D}$ and $\mathbf{D} \in \mathbb{R}^{k\times m}$ can be precomputed and used in each iteration. We obtain the reduced system by the POD-DEIM approach (POD-DEIM reduced system):

$$\tilde{\mathbf{y}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\mathbf{precomputed}:k\times k}\underbrace{\tilde{\mathbf{y}}(t_j)}_{k\times 1} + \underbrace{\mathbf{D}}_{\mathbf{precomputed}:k\times m}\underbrace{\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j))}_{m\times 1}. \tag{2.14}$$

The dimension n of the nonlinear term in (2.5) decreases to m with $m \ll n$ in (2.14), which can reduce computational cost in the simulation.

2.3 Gappy POD (GPOD)

This section considers a similar approach to DEIM called Gappy POD for reducing the computational cost of nonlinear term in a POD reduced system. This method uses a dimension of the nonlinear basis less than the number of the indices used for selecting

row in the approximation.

Suppose that q is the number of selected rows and m is the dimension of the nonlinear basis. We obtain a rectangular matrix (skinny matrix) $\mathbf{P}^T\mathbf{U} \in \mathbb{R}^{q\times m}$, $m < q$ for using in the nonlinear approximation. Consider the nonlinear function $\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) \approx \mathbf{U}\mathbf{c}(t_j)$; $\mathbf{c}(t_j) \in \mathbb{R}^m$ and solve $\mathbf{P}^T\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) \approx (\mathbf{P}^T\mathbf{U})\mathbf{c}(t_j)$ for $\mathbf{c}(t_j)$ such that $\min_{\mathbf{c}(t_j)} \left\|\mathbf{P}^T\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) - (\mathbf{P}^T\mathbf{U})\mathbf{c}(t_j)\right\|_2^2$, then

$$\begin{aligned}\mathbf{c}(t_j) &= [(\mathbf{P}^T\mathbf{U})^T(\mathbf{P}^T\mathbf{U})]^{-1}(\mathbf{P}^T\mathbf{U})^T\mathbf{P}^T\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j)) \\ &= (\mathbf{P}^T\mathbf{U})^+\mathbf{P}^T\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{y}}(t_j),\end{aligned}$$

where $\mathbf{P}^T\mathbf{U} = [(\mathbf{P}^T\mathbf{U})^T(\mathbf{P}^T\mathbf{U})]^{-1}(\mathbf{P}^T\mathbf{U})^T$ is the pseudoinverse of $\mathbf{P}^T\mathbf{U}$.

Since $\mathbf{F}$ is a componentwise function

$$\mathbf{c}(t_j) = (\mathbf{P}^T\mathbf{U})^+\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j), \tag{2.15}$$

and we obtain the reduced system by the POD-GPOD approach (POD-GPOD reduced system):

$$\tilde{\mathbf{y}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\mathbf{precomputed}:k\times k}\underbrace{\tilde{\mathbf{y}}(t_j)}_{k\times 1} + \underbrace{\mathbf{E}}_{\mathbf{precomputed}:k\times q}\underbrace{\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t_j))}_{q\times 1}, \tag{2.16}$$

where $\mathbf{E} = \mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^+$ and $\mathbf{E} \in \mathbb{R}^{k\times q}$, which can be precomputed and used in each iteration for $m, q \ll n$.

In the next Chapter, we use these methods to construct reduced system for Allen-cahn equation.

CHAPTER 3

MODEL PROBLEM: ALLEN-CAHN EQUATION

In this chapter, we present the model problem of the Allen-Cahn equation and construct the corresponding discretized system by the finite difference method. Next, we use POD, DEIM, and GPOD to construct reduced models.

3.1 Model Problem: Finite Difference Discretized System

This section consists of two subsections: the model problem of the Allen-Cahn equation, and its finite difference discretization.

3.1.1 Model Problem

Consider the Allen-Cahn equation of the form

$$\frac{\partial u}{\partial t} = \epsilon \frac{\partial^2 u}{\partial x^2} + F(u), \quad x \in \Omega, \quad t \geq 0 \tag{3.1}$$

with $F(u) = u - u^3$, and $\Omega = [a, b]$, for some real number a, b with $a < b$ where

the initial condition : $u(x, 0) = f(x)$, and
the boundary conditions : $u(a, t) = g_1(t), \quad u(b, t) = g_2(t), \quad t > 0.$

Here, we have variables t for time and x for position, reaction term $F(u)$ is nonlinear, and with boundary conditions dependent on the problem for a domain Ω. In this context, $u(x, t)$ denotes the phase function of two kinds of liquid at each position in space, and time and ϵ is a small positive parameter value, which is related to the width of the

interface of two kinds of liquid or the interaction length, which describe the thickness of phase boundary in the laboratory scale [3]. The term $\epsilon\frac{\partial^2 u}{\partial x^2}$ represents the diffusion of the liquid, and the term $F(u)$ is the kinetic potential of the liquid. For $u(x_1, t_1) = 1$, it means that at the time t_1, the position x_1 is filled only one kind of liquid; for $u(x_2, t_2) = -1$, it means that at the time t_2, the position x_2 is filled only the other kind of liquid. For $t = 0$, we will provide the initial data of $u(x, 0)$. This equation has three constant steady states, $u = -1$, $u = 0$, and $u = 1$. The middle state is unstable, but the states $u = 1$, and $u = -1$ are attracting, and solutions tend to exhibit at areas close to these values separated by interfaces that may coalesce or vanish on a long time scale, a phenomenon known as metastability [57]. Thus it is called a bistable reaction-diffusion equation or a reaction-diffusion equation with bistable nonlinearity.

3.1.2 Discretization

In this subsection, we present the discretization of the Allen-Cahn equation (3.1) using the Crank-Nicolson finite difference method for the linear term and the forward Euler method for the nonlinear term. The resulting discretized system is given by

$$\begin{aligned}\frac{u(x_i, t_{j+1}) - u(x_i, t_j)}{\Delta t} &= \frac{\epsilon}{2}\left(\frac{u(x_{i+1}, t_j) - 2u(x_i, t_j) + u(x_{i-1}, t_j)}{(\Delta x)^2}\right) \\ &\quad + \frac{\epsilon}{2}\left(\frac{u(x_{i+1}, t_{j+1}) - 2u(x_i, t_{j+1}) + u(x_{i-1}, t_{j+1})}{(\Delta x)^2}\right) \\ &\quad + u(x_i, t_j) - u^3(x_i, t_j), \end{aligned} \tag{3.2}$$

where Δx and Δt are the step-sizes of space and time discretizations, respectively. The notations i and j are the indices of the discretization in space and time. Equation (3.2) can be written as

$$\begin{aligned}&\left[-\frac{\epsilon}{2(\Delta x)^2}u(x_{i+1}, t_{j+1}) + \left(\frac{1}{\Delta t} + \frac{\epsilon}{(\Delta x)^2}\right)u(x_i, t_{j+1}) - \frac{\epsilon}{2(\Delta x)^2}u(x_{i-1}, t_{j+1})\right] = \\ &\left[\frac{\epsilon}{2(\Delta x)^2}u(x_{i+1}, t_j) + \left(\frac{1}{\Delta t} - \frac{\epsilon}{(\Delta x)^2}\right)u(x_i, t_j) + \frac{\epsilon}{2(\Delta x)^2}u(x_{i-1}, t_j)\right] + u(x_i, t_j) \\ &- u^3(x_i, t_j). \end{aligned} \tag{3.3}$$

In matrix notation, (3.3) is given by

$$\mathbf{A}\mathbf{u}(t_{j+1}) = \mathbf{B}\mathbf{u}(t_j) + \mathbf{F}(\mathbf{u}(t_j)), \tag{3.4}$$

where $\mathbf{A}$ and $\mathbf{B}$ are a finite difference matrices.

The matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{F}(\mathbf{u}(t_j))$, and $\mathbf{u}(t_j)$ a defined as follows:

$$\mathbf{A} = \begin{bmatrix} \left(\frac{1}{\Delta t} + \frac{\epsilon}{(\Delta x)^2}\right) & -\frac{\epsilon}{2(\Delta x)^2} & 0 & \cdots & 0 \\ -\frac{\epsilon}{2(\Delta x)^2} & \left(\frac{1}{\Delta t} + \frac{\epsilon}{(\Delta x)^2}\right) & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \left(\frac{1}{\Delta t} + \frac{\epsilon}{(\Delta x)^2}\right) & -\frac{\epsilon}{2(\Delta x)^2} \\ 0 & \cdots & 0 & -\frac{\epsilon}{2(\Delta x)^2} & \left(\frac{1}{\Delta t} + \frac{\epsilon}{(\Delta x)^2}\right) \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} \left(\frac{1}{\Delta t} - \frac{\epsilon}{(\Delta x)^2}\right) & \frac{\epsilon}{2(\Delta x)^2} & 0 & \cdots & 0 \\ \frac{\epsilon}{2(\Delta x)^2} & \left(\frac{1}{\Delta t} - \frac{\epsilon}{(\Delta x)^2}\right) & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \left(\frac{1}{\Delta t} - \frac{\epsilon}{(\Delta x)^2}\right) & \frac{\epsilon}{2(\Delta x)^2} \\ 0 & \cdots & 0 & \frac{\epsilon}{2(\Delta x)^2} & \left(\frac{1}{\Delta t} - \frac{\epsilon}{(\Delta x)^2}\right) \end{bmatrix},$$

$$\mathbf{F}(\mathbf{u}(t_j)) = \begin{bmatrix} u(x_1, t_j) - u^3(x_1, t_j) \\ u(x_2, t_j) - u^3(x_2, t_j) \\ \vdots \\ u(x_{n-1}, t_j) - u^3(x_{n-1}, t_j) \\ u(x_n, t_j) - u^3(x_n, t_j) \end{bmatrix},$$

$$\mathbf{u}(t_{j+1}) = \begin{bmatrix} u(x_1, t_{j+1}) \\ u(x_2, t_{j+1}) \\ \vdots \\ u(x_{n-1}, t_{j+1}) \\ u(x_n, t_{j+1}) \end{bmatrix}, \quad \mathbf{u}(t_j) = \begin{bmatrix} u(x_1, t_j) \\ u(x_2, t_j) \\ \vdots \\ u(x_{n-1}, t_j) \\ u(x_n, t_j) \end{bmatrix}.$$

We can also write (3.4) in the form

$$\mathbf{u}(t_{j+1}) = \underbrace{\mathbf{A}^{-1}\mathbf{B}}_{\textbf{precomputed}:n\times n}\underbrace{\mathbf{u}(t_j)}_{n\times 1} + \underbrace{\mathbf{A}^{-1}}_{n\times n}\underbrace{\mathbf{F}(\mathbf{u}(t_j))}_{n\times 1}.$$

In general, when the solution of the full discretized system is required to be highly accurate, solving the system may require high computational complexity, or has large discretized dimension.

3.2 Reduced system

This section presents three methods for efficiently reducing original model: Proper Orthogonal Decomposition (POD), the Discrete Empirical Interpolation Method (DEIM), and Gappy POD (GPOD).

3.2.1 POD reduced system

In this subsection, we will reduce the dimension of the system for the Allen-Cahn equation using POD, described in Chapter 2.

For constructing the POD reduced system, we start by collecting n_s snapshots $\mathbf{u}(t)$ from the solution of the full system of (3.4) to generate the snapshot matrix.

$$\mathbf{S} = [\mathbf{u}(t_1), ..., \mathbf{u}(t_{n_s})] .$$

Next, we will construct the POD basis $\mathbf{V}_k$ from the Algorithm 1 in Section 2.1.

We can use POD basis $\mathbf{V}_k$ for constructing the reduced system for the full system, by the following two steps.

Step 1 We consider the representation $\mathbf{u} \approx \mathbf{V}_k\tilde{\mathbf{u}}$, where $\mathbf{V}_k \in \mathbb{R}^{n\times k}$, and $\tilde{\mathbf{u}} \in \mathbb{R}^{\mathbf{k}}$ $(k \ll n)$

$$\mathbf{V}_k\tilde{\mathbf{u}}(t_{j+1}) = \mathbf{A}^{-1}(\mathbf{B}\mathbf{V}_k\tilde{\mathbf{u}}(t_j)) + \mathbf{A}^{-1}(\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{u}}(t_j))). \tag{3.5}$$

Step 2 We apply the projection along the subspace generated by the columns of $\mathbf{V}_k$, which forms an orthonormal basis ($\mathbf{V}_k^T\mathbf{V}_k = \mathbf{I} \in \mathbb{R}^{k\times k}$). We have

$$\tilde{\mathbf{u}}(t_{j+1}) = (\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{B}\mathbf{V}_k)\tilde{\mathbf{u}}(t_j) + \mathbf{V}_k^T\mathbf{A}^{-1}(\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{u}}(t_j))). \tag{3.6}$$

Finally, we obtain the POD reduced system

$$\tilde{\mathbf{u}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\textbf{precomputed}:k\times k}\underbrace{\tilde{\mathbf{u}}(t_j)}_{k\times 1} + \underbrace{\mathbf{V}_k^T\mathbf{A}^{-1}}_{\textbf{precomputed}:k\times n}\underbrace{\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{u}}(t_j))}_{n\times 1}. \tag{3.7}$$

with $\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{B}\mathbf{V}_k = \mathbf{C}$ where $\mathbf{C} \in \mathbb{R}^{k\times k}$ can be precomputed and reused in each iteration.

3.2.2 POD-DEIM reduced system

Since the nonlinear term of (3.7) in the previous subsection still has a *large* computational cost which depends on the dimension n, we will use DEIM to approximate this nonlinear term by collecting snapshots from the nonlinear function of the full-order system to construct the nonlinear snapshot matrix $\mathbf{F} = \{F(\mathbf{u}(t_1)), ..., F(\mathbf{u}(t_{n_s}))\}$. This matrix is used to find the projection basis $[\mathbf{u}_1, ..., \mathbf{u}_m]$ with $m \ll n$ by the POD algorithm. Then this basis is used to find the interpolation indices $[\varrho_1, ..., \varrho_m]$ from the DEIM algorithm.

We obtain $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_m] \in \mathbb{R}^{n\times m}$ and $\mathbf{P} = [\mathbf{e}_{\varrho_1}, ..., \mathbf{e}_{\varrho_m}] \in \mathbb{R}^{n\times m}$ where $e_{\varrho_i} = [0 \ ... \ 0 \ 1 \ 0 \ ... \ 0]^T \in \mathbb{R}^n$ is the ϱ_ith column of the identity matrix $\mathbf{I} \in \mathbb{R}^{n\times n}$ for $i = 1, ..., m$. We define the nonlinear term from (3.7) by

$$\mathbf{N}(\tilde{\mathbf{u}}(t_j)) = \underbrace{\mathbf{V}_k^T\mathbf{A}^{-1}}_{k\times n}\underbrace{\mathbf{F}(\mathbf{V}_k\tilde{\mathbf{u}}(t_j))}_{n\times 1}. \tag{3.8}$$

From subsection 2.2.2, by replacing $\tilde{\mathbf{y}}$ by $\tilde{\mathbf{u}}$ in (3.8), we have the following DEIM approximation for $\mathbf{N}(\tilde{\mathbf{u}}(t_j))$:

$$\mathbf{N}(\tilde{\mathbf{u}}(t_j)) \approx \underbrace{\mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}}_{k\times m}\underbrace{\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{u}}(t_j))}_{m\times 1}, \tag{3.9}$$

From (3.7) and (3.9), we obtain the POD-DEIM reduced system:

$$\tilde{\mathbf{u}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\textbf{precomputed}:k\times k}\underbrace{\tilde{\mathbf{u}}(t_j)}_{k\times 1} + \underbrace{\mathbf{M}}_{\textbf{precomputed}:k\times m}\underbrace{\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{u}}(t_j))}_{m\times 1}. \tag{3.10}$$

where $\mathbf{M} = \mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}$ and $\mathbf{M} \in \mathbb{R}^{k\times m}$ can be precomputed and used in each iteration.

3.2.3 POD-GPOD reduced system

Gappy POD (GPOD) can be use to reduce the computational cost in solving the Allen-Cahn equation as shown in (2.16). From Section 2.3, by replacing $\tilde{\mathbf{y}}$ by $\tilde{\mathbf{u}}$ in (3.7), we use a dimension of the nonlinear basis, which is less than the number of selected indices. We then obtain the POD-GPOD reduced system

$$\tilde{\mathbf{u}}(t_{j+1}) = \underbrace{\mathbf{C}}_{\mathbf{precomputed}:k\times k}\underbrace{\tilde{\mathbf{u}}(t_j)}_{k\times 1} + \underbrace{\mathbf{N}}_{\mathbf{precomputed}:k\times q}\underbrace{\mathbf{F}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{u}}(t_j))}_{q\times 1}, \tag{3.11}$$

where $\mathbf{N} = \mathbf{V}_k^T\mathbf{A}^{-1}\mathbf{U}(\mathbf{P}^T\mathbf{U})^+$ and $\mathbf{N} \in \mathbb{R}^{k\times q}$ and $\mathbf{P}^T\mathbf{U} = [(\mathbf{P}^T\mathbf{U})^T(\mathbf{P}^T\mathbf{U})]^{-1}(\mathbf{P}^T\mathbf{U})^T$ which can be precomputed and used in each iteration and $q \ll n$ and $q > m$.

Table 3.1: Computational complexity for the Allen-Cahn model.

System	Complexity (one iteration step)
The full discretized system	$\mathcal{O}(n^3)$
The POD reduced system	$\mathcal{O}(k^3 + nk^2)$
The POD-DEIM reduced system	$\mathcal{O}(k^3 + mk^2)$
The POD-GPOD reduced system	$\mathcal{O}(k^3 + qk^2)$

From Table 3.1, computational cost of the FD full system is $\mathcal{O}(n^3)$ with dimension n is *large*. The cost of the POD reduced system is $\mathcal{O}(k^3 + nk^2)$ with $k \ll n$. We reduced nonlinear term of the POD reduced system by DEIM, and GPOD. We obtain that the complexity of the POD-DEIM reduced system is $\mathcal{O}(k^3+mk^2)$ with $m \ll n$. The complexity of the POD-GPOD reduced system is $\mathcal{O}(k^3 + qk^2)$ with $q \ll n$, and $q > m$.

Next Chapter presents the numerical results to demonstrate that the resulting reduced systems can accurately give approximate solutions to the original system.

CHAPTER 4

NUMERICAL RESULTS

This Chapter applies the model reduction techniques: POD, POD-DEIM, and POD-GPOD approaches discuss in the previous chapters on Allen-Cahn equation. It mainly considers two type of numerical tests. In Section 4.1, we test these model reduction with the same parameter value for three different initial conditions. In Section 4.2, we test these model reduction approaches with various parameter values for two different initial conditions. In these tests, we use the MATLAB program for solving the numerical solutions.

4.1 Numerical examples with same parameter value

This section consists of three examples of model reduction for the Allen-Cahn equation. Example 1 is the initial boundary value problem of this equation, with homogeneous boundary conditions. Examples 2, and 3 are the initial boundary value problem with non-homogeneous boundary conditions.

4.1.1 Example 1 (Homogeneous boundary conditions)

Consider the initial boundary value problem [20]

$$\frac{\partial u}{\partial t} = \epsilon \frac{\partial^2 u}{\partial x^2} + u - u^3, \quad x \in \Omega, \quad t \geq 0,$$

with $\Omega = \left[\frac{2\pi}{n}, 2\pi\right]$, $n \in \mathbb{Z}^+$, $n > 1$,

the initial condition : $u(x,0) = 0.25\sin(x)$, and
the homogeneous boundary conditions : $u(0,t) = 0, \quad u(2\pi,t) = 0, \quad t > 0.$

In our numerical tests: the number of internal points is $n = 600$ in $\left[2\pi, \frac{2\pi}{n}\right]$, the number of time steps is $n_t = 700$ on $[0,5]$ and $\epsilon = 0.01$. We used a space step-size $\Delta x = \frac{2\pi}{n-1}$, time step-size $\Delta t = \frac{5}{n_t - 1}$ and the basis sets used in POD, DEIM, and GPOD approximations were constructed from $n_s = 700$ snapshots.

Numerical Result of Finite Difference full-order System

First, we solved the numerical solution of the full discretized system. We obtained the evolution of phase function of Allen-Cahn equation from time $t = 0$ to $t = 5$ as shown in Figure 4.1.

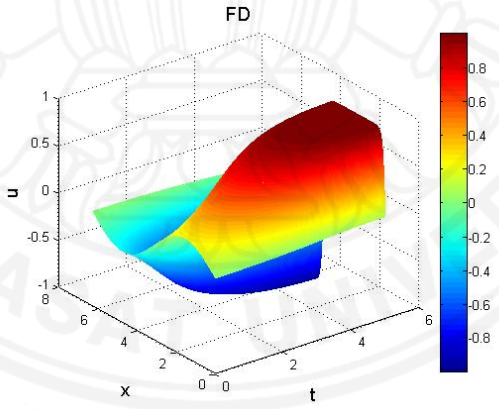

Figure 4.1: Solution of the full discretized system for Example 1 with $\epsilon = 0.01$.

Numerical Results of POD, POD-DEIM, and POD-GPOD Reduced Systems

Next, we solved the numerical solution of the reduced system and used dimensions of POD basis $k = 30$ and DEIM basis $m = 50$. We obtained the approximate evolution of phase function of Allen-Cahn equation from POD, and POD-DEIM approaches for time $t = 0$ to $t = 5$ as shown in Figures 4.2 and 4.3, respectively. Finally, GPOD is used for reducing the complexity of the nonlinear term from the POD reduced

system using a dimension of nonlinear basis m that is less than the number of selected rows q. The dimension of POD basis, nonlinear basis, and the number of selected rows in this numerical test are $k = 30, m = 50$, and $q = 80$, respectively. We obtained the evolution of the phase function of Allen-Cahn equation by using POD-GPOD method for time $t = 0$ to $t = 5$ as shown in Figure 4.4.

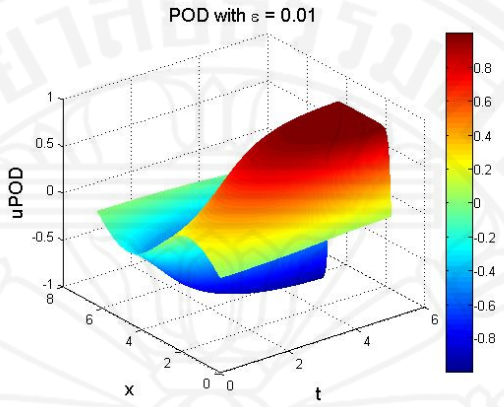

Figure 4.2: Solution of the POD reduced system for Example 1 with $\epsilon = 0.01$

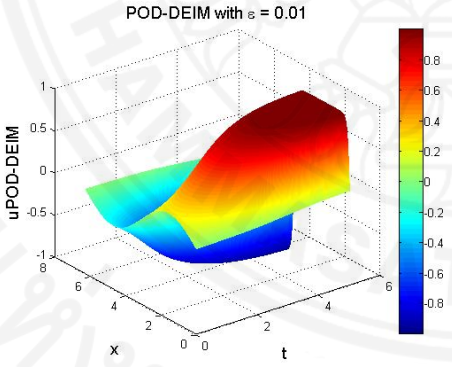

Figure 4.3: Solution of the POD-DEIM reduced system for Example 1 with $\epsilon = 0.01$

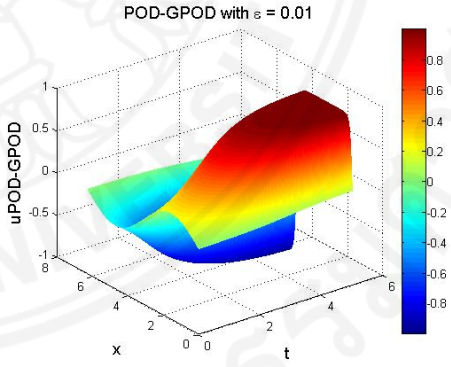

Figure 4.4: Solution of the POD-GPOD reduced system for Example 1 with $\epsilon = 0.01$

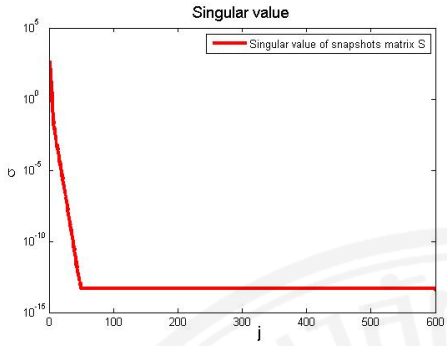

Figure 4.5: Plot of the singular values of the snapshot matrix $\mathbf{S}$ for Example 1 from SVD using POD basis.

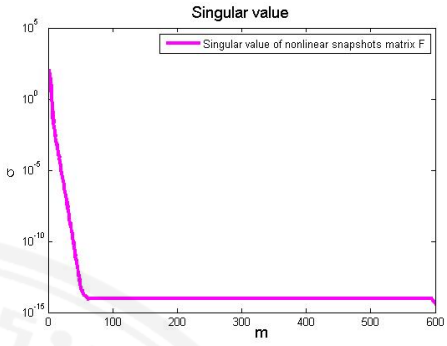

Figure 4.6: Plot of the singular values of the nonlinear snapshot matrix $\mathbf{F}$ for Example 1 from SVD using DEIM basis.

Runtime and error

The error approximations from the reduced systems constructed by POD, POD-DEIM, and POD-GPOD approaches are defined as

$$e = \frac{\|\mathbf{u} - \tilde{\mathbf{u}}\|_2}{\|\mathbf{u}\|_2} \tag{4.1}$$

where $\mathbf{u}$ and $\tilde{\mathbf{u}}$ are snapshot matrices from solving the full discretized system and reduced system, respectively. For the POD, POD-DEIM, and POD-GPOD reduced systems, we computed the runtime and error using the MATLAB program, as shown in Table 4.1, Table 4.2, and Table 4.3, respectively.

Table 4.1: Runtime and error of the POD reduced systems for Example 1.

POD basis (k)	Error	Runtime
10	2.4601×10^{-6}	3.546203 s
20	3.7046×10^{-7}	3.655103 s
30	1.7271×10^{-7}	3.824681 s
40	1.0819×10^{-7}	3.916436 s
50	3.3184×10^{-8}	3.944950 s
60	3.7444×10^{-8}	4.047922 s
70	3.3093×10^{-8}	4.050324 s

Table 4.2: Runtime and error of the POD-DEIM reduced systems using POD basis with dimention $k = 30$ for Example 1.

DEIM basis (m)	Error	Runtime
10	3.2007×10^{-3}	0.017012 s
20	3.4134×10^{-4}	0.017546 s
30	1.1348×10^{-4}	0.019273 s
40	1.4441×10^{-5}	0.023158 s
50	2.4285×10^{-6}	0.024547 s
60	2.7139×10^{-6}	0.026493 s
70	7.6205×10^{-7}	0.026964 s

For the POD-GPOD reduced system, we computed the runtime and error from (4.1), as shown in Table 4.3. Error of the POD-GPOD reduced systems with the dimension of POD basis $k = 30$, for Example 1, where m is the dimension of nonlinear basis, and q is the number of selected rows.

Table 4.3: Runtime and error of the POD-GPOD reduced systems for Example 1.

m	q	Error	Runtime
10	40	3.1224×10^{-3}	0.028104 s
20	50	4.8364×10^{-4}	0.028222 s
30	60	3.9332×10^{-5}	0.029131 s
40	70	3.7254×10^{-6}	0.029560 s
50	80	9.5442×10^{-7}	0.031125 s
60	90	6.7214×10^{-7}	0.033241 s
70	100	6.5710×10^{-7}	0.034160 s

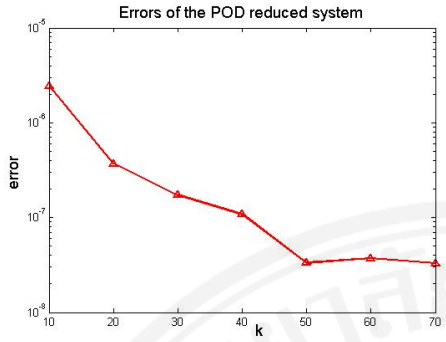

Figure 4.7: Error plot of the approximation from reduced system with dimension of POD basis k for Example 1.

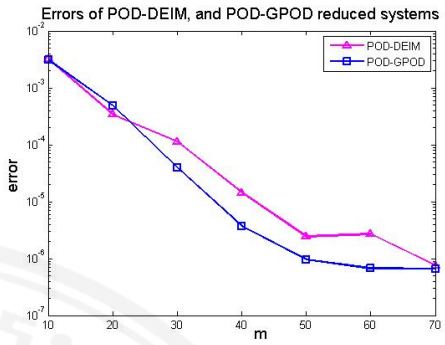

Figure 4.8: Error plot of the approximation the POD-DEIM, and POD-GPOD reduced systems with dimension of DEIM or nonlinear basis m for Example 1.

From Figures 4.5 and 4.6, the decreasing of singular values of the snapshot matrices $\mathbf{S}, \mathbf{F}$ and then beginning to stabilize implies that solution information lies within a subspace whose dimension is significantly lower than the full dimension used in the discretization. That is, we can reduce the dimensions of the full discretized system. It can be observed that the decrease of the error in Figure 4.7 corresponds to the increase of dimension of POD basis in Table 4.1. At a fixed POD dimension of $k = 30$, the decrease of the error corresponds to the increase of dimension of DEIM basis in Table 4.2. At a fixed POD dimension of $k = 30$, the decrease of the error corresponds to the increase of dimension of nonlinear basis, and the number of selected rows in Table 4.3. In Figure 4.8, the error for the POD-GPOD reduced system is decreased more than the error from the POD-DEIM reduced system. The runtime of the full discretized system with $\epsilon = 0.01$ is 8.789236 seconds. The average runtime of the reduced systems by POD approach was almost 2.3 times less than the full discretized system (Table 4.1). The average runtime using POD-DEIM approach was almost 414 times less than the runtime of the full discretized system (Table 4.2). The average runtime using POD-GPOD approach was almost 334 times less than the runtime of the full discretized system (Table 4.3). Tables 4.4 summarizes the average error and the scaled runtime of the reduced systems when compared with the full-order system (the runtime of full-order system is normalized to be one).

Table 4.4: The ratio of the runtime of these systems to the runtime of the full discretized system, and error of these systems for Example 1 with $k = 30$, $m = 40$, and $q = 70$.

System	Error	Ratio of runtime
The full discretized system	-	1
The POD reduced system	$\mathcal{O}(10^{-7})$	$1/2.3$
The POD-DEIM reduced system	$\mathcal{O}(10^{-5})$	$1/382$
The POD-GPOD reduced system	$\mathcal{O}(10^{-6})$	$1/303$

4.1.2 Example 2 (Non-homogeneous boundary conditions)

Consider the initial boundary value problem [57]

$$\frac{\partial u}{\partial t} = \epsilon \frac{\partial^2 u}{\partial x^2} + u - u^3, \quad x \in \Omega, \quad t \geq 0,$$

for $\Omega = [-1, 1]$, with

the initial condition : $u(x, 0) = 0.53x + 0.47 \sin(1.5\pi x)$, and
the non homogeneous boundary conditions : $u(-1, t) = -1, \quad u(1, t) = 1, \quad t > 0$.

In our numerical tests: the number of internal points is $n = 600$ in $[-1, 1]$, the number of time steps is $n_t = 700$ on $[0, 60]$ and $\epsilon = 0.01$. We used a space step-size $\Delta x = \frac{2}{n-1}$, time step-size $\Delta t = \frac{60}{n_t - 1}$ and the basis sets used in POD, DEIM, and GPOD approximations were constructed from $n_s = 700$ snapshots.

Numerical Result of Finite Difference full-order System

First, we solved the numerical solution of the full discretized system. We obtained the evolution of phase function of Allen-Cahn equation from time $t = 0$ to $t = 60$ as shown in Figure 4.9.

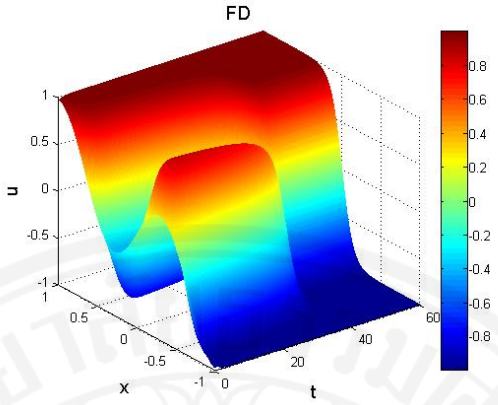

Figure 4.9: Solution of the full discretized system for Example 2 with $\epsilon = 0.01$.

Numerical Results of POD, POD-DEIM, and POD-GPOD Reduced Systems

Next, we solved the numerical solution of reduced system and used dimensions of POD basis $k = 50$ and DEIM basis $m = 60$. We obtained the approximate evolution of phase function of Allen-Cahn equation from POD, and POD-DEIM approaches for time $t = 0$ to $t = 60$ as shown in Figure 4.10 and 4.11, respectively. Finally, GPOD is used for reducing the complexity of the nonlinear term from the POD reduced system by using a dimension of nonlinear basis m that is less than the number of selected rows q. The dimension of POD basis, nonlinear basis, and the number of selected rows in this numerical test are $k = 50, m = 50$, and $q = 80$, respectively. We obtained the evolution of the phase function of Allen-Cahn equation by using POD-GPOD method for time $t = 0$ to $t = 60$ as shown in Figure 4.12.

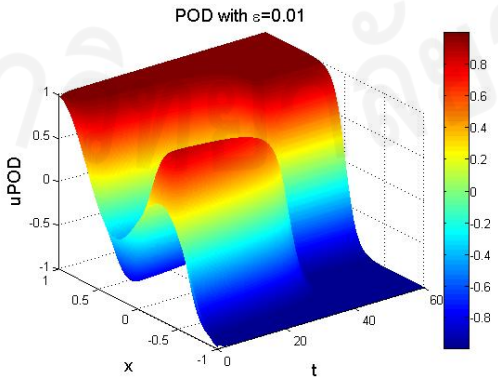

Figure 4.10: Solution of the POD reduced system with $\epsilon = 0.01$ for Example 2.

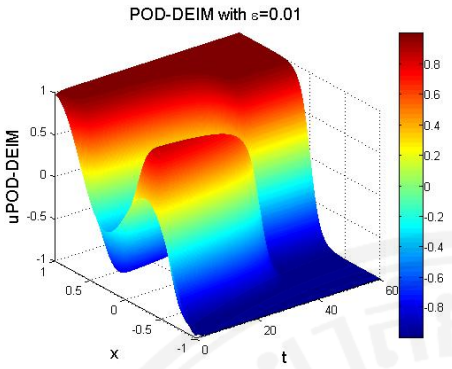

Figure 4.11: Solution of the POD-DEIM reduced system for Example 2 with $\epsilon = 0.01$.

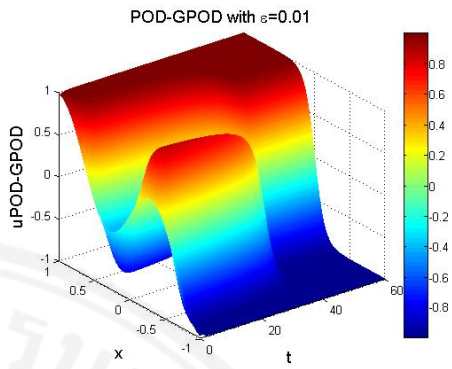

Figure 4.12: Solution of the POD-GPOD reduced system for Example 2 with $\epsilon = 0.01$.

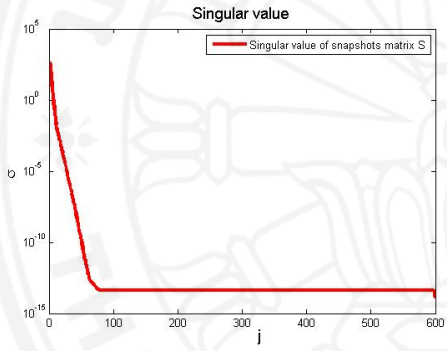

Figure 4.13: Plot of the singular values of the snapshot matrix $\mathbf{S}$ from SVD using POD basis for Example 2.

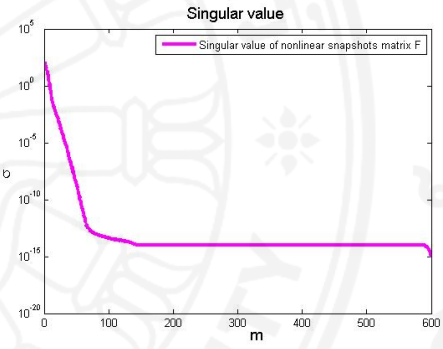

Figure 4.14: Plot of the singular values of the nonlinear snapshot matrix $\mathbf{F}$ from SVD using DEIM basis for Example 2.

For the POD, POD-DEIM, and POD-GPOD reduced systems of Example 2, we computed the runtime and error from (4.1), as shown in Table 4.5, Table 4.6, and Table 4.7 respectively.

Table 4.5: Runtime and error of the POD reduced systems for Example 2.

POD basis (k)	Error	Runtime
10	6.2880×10^{-5}	3.414315 s
20	1.1173×10^{-5}	3.472555 s
30	6.4042×10^{-6}	3.462643 s
40	2.0096×10^{-6}	3.487229 s
50	3.0130×10^{-8}	3.573060 s
60	5.3100×10^{-9}	3.596739 s
70	4.1197×10^{-9}	3.542982 s

Table 4.6: Runtime and error of the POD-DEIM reduced systems using POD basis with dimension $k = 50$ for Example 2.

DEIM basis (m)	Error	Runtime
10	8.7695×10^{-3}	0.014838 s
20	7.8254×10^{-4}	0.015784 s
30	1.1173×10^{-4}	0.016634 s
40	4.2691×10^{-5}	0.017479 s
50	3.2536×10^{-6}	0.025762 s
60	1.0416×10^{-7}	0.027731 s
70	1.1034×10^{-7}	0.027573 s

For the POD-GPOD reduced system, we computed the runtime and error from (4.1), as shown in Table 4.7. Error of the POD-GPOD reduced systems with use POD basis of dimension $k = 50$, for Example 2, where m is the dimension of nonlinear basis, and q is the number of selected rows.

Table 4.7: Runtime and error of the POD-GPOD reduced systems for Example 2.

m	q	Error	Runtime
10	40	6.8678×10^{-3}	0.038945 s
20	50	5.3332×10^{-4}	0.039222 s
30	60	4.6130×10^{-5}	0.039542 s
40	70	5.1778×10^{-6}	0.040523 s
50	80	1.8131×10^{-6}	0.041596 s
60	90	2.9794×10^{-8}	0.043719 s
70	100	2.9774×10^{-8}	0.044837 s

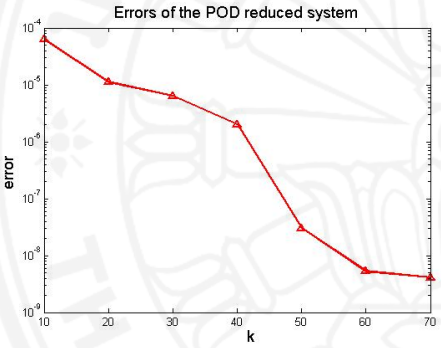

Figure 4.15: Error plot of the approximation from POD reduced system with different dimensions of POD basis k for Example 2.

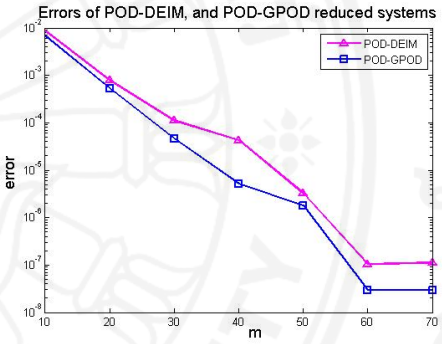

Figure 4.16: Error plot of the approximation the POD-DEIM, and POD-GPOD reduced systems with dimension of DEIM or nonlinear basis m for Example 2.

From Figures 4.13 and 4.14, the decreasing of singular values of the snapshot matrices $\mathbf{S}, \mathbf{F}$ and then beginning to stabilize implies that solution information lies within a subspace whose dimension is significantly lower than the full dimension used in the discretization. That is, we can reduce the dimensions of the full discretized system. It can be observed that the decrease of the error in Figure 4.15 corresponds to the increase of dimension of POD basis in Table 4.5. At a fixed POD dimension of $k = 50$, the decrease of the error corresponds to the increase of dimension of DEIM basis in Table 4.6. At a fixed POD dimension of $k = 50$, the decrease of the error corresponds to the increase of dimension of nonlinear basis, and the number of selected

rows in Table 4.7. In Figure 4.16, the error for the POD-GPOD reduced system is decreased more than the error from the POD-DEIM reduced system. The runtime of the full discretized system with $\epsilon = 0.01$ is 11.361802 seconds. The average runtime of the reduced systems by POD approach was almost 3.14 times less than the runtime of the full discretized system (Table 4.5). The average runtime using POD-DEIM approach was almost 550 times less than the runtime of the full discretized system (Table 4.6). The average runtime using POD-GPOD approach was almost 283 times less than the runtime of the full discretized system (Table 4.7). Tables 4.8 summarizes the average error and the scaled computational time of the reduced systems when compared to the full-order system (the runtime of full-order system is normalized to be one).

Table 4.8: The ratio of the runtime of these systems to the runtime of the full discretized system, and error of these systems for Example 2 with $k = 50$, $m = 40$, and $q = 70$.

System	Error	Ratio of runtime
The full discretized system	-	1
The POD reduced system	$\mathcal{O}(10^{-8})$	$1/3.2$
The POD-DEIM reduced system	$\mathcal{O}(10^{-5})$	$1/668$
The POD-GPOD reduced system	$\mathcal{O}(10^{-6})$	$1/284$

4.1.3 Example 3 (Non-homogenious boundary conditions) with Square Block Initial Data

In this subsection, we used a square block initial condition to test model reduction approaches. The initial condition is defined by a piecewise constant function with value a on wave crest and value b on wave trough. We used the following inputs: internal points $(n = 100)$ in $[-1, 1]$, time steps $(n_t = 600)$ on $[0, 12]$ and $\epsilon = 0.01$. We used a space step-size $\Delta x = \frac{2}{n-1}$, time step-size $\Delta t = \frac{12}{n_t-1}$ and the basis sets used in POD, DEIM, and GPOD approximations were constructed from $n_s = 600$ snapshots.

Numerical Result of Finite Difference full-order System

First, we selected wave crest $a = 1$ and wave trough $b = -1$. We solved the numerical solution of the full discretized system. When the time increases, the evolution of the Allen-Cahn equation still keeps the phase separation after long time calculation in Figure 4.17.

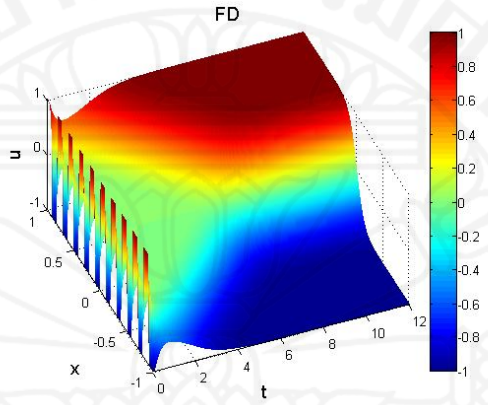

Figure 4.17: Solution of the full discretized system for Example 3 with $\epsilon = 0.01$

Numerical Results of POD, POD-DEIM, and POD-GPOD Reduced Systems

Next, we solved the numerical solution of the reduced system and used dimensions of POD basis $k = 50$ and DEIM basis $m = 60$. We obtained the approximate evolution of phase function of Allen-Cahn equation from POD, and POD-DEIM approaches for time $t = 0$ to $t = 12$ as shown in Figures 4.18 and 4.19, respectively. Finally, GPOD is used for reducing the complexity of the nonlinear term from the POD reduced system using a dimension of nonlinear basis m that is less than the number of selected rows q. The dimension of POD basis, nonlinear basis, and the number of selected rows in this numerical test are $k = 50, m = 60$, and $q = 90$, respectively. We obtained the evolution of the phase function of Allen-Cahn equation by using POD-GPOD method for time $t = 0$ to $t = 12$ as shown in Figure 4.20.

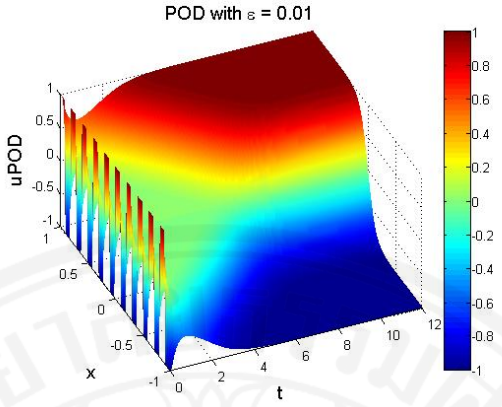

Figure 4.18: Solution of the POD reduced system with $\epsilon = 0.01$ for Example 3.

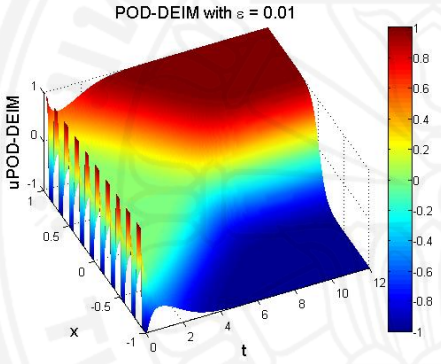

Figure 4.19: Solution of the POD-DEIM reduced system for Example 3 with $\epsilon = 0.01$.

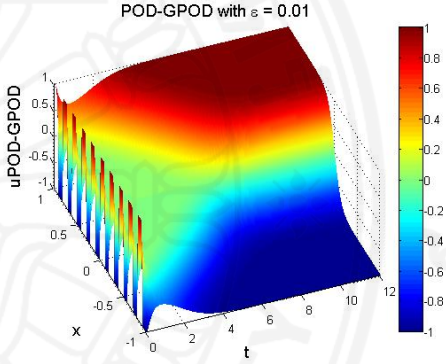

Figure 4.20: Solution of the POD-GPOD reduced system for Example 3 with $\epsilon = 0.01$.

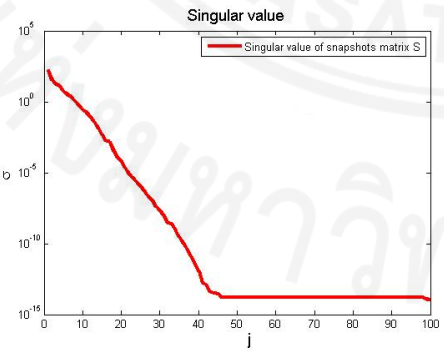

Figure 4.21: Plot of the singular values of the snapshot matrix $\mathbf{S}$ from SVD using POD basis for Example 3.

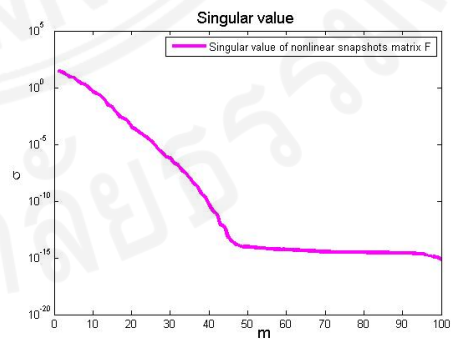

Figure 4.22: Plot of the singular values of the nonlinear snapshot matrix $\mathbf{F}$ from SVD using DEIM basis for Example 3.

For the POD, POD-DEIM, and POD-GPOD reduced systems of Example 3, we computed the runtime and error from (4.1), as shown in Table 4.9, Table 4.10, and Table 4.11 respectively.

Table 4.9: Runtime and error of the POD reduced systems for Example 3.

POD basis (k)	Error	Runtime
10	1.0178×10^{-3}	0.106746 s
20	1.2202×10^{-7}	0.107846 s
30	4.9045×10^{-11}	0.110317 s
40	7.2924×10^{-14}	0.113508 s
50	7.3258×10^{-14}	0.115325 s
60	7.3410×10^{-14}	0.124748 s
70	7.3256×10^{-14}	0.161217 s

Table 4.10: Runtime and error of the POD-DEIM reduced systems with POD basis with dimension $k = 50$ for Example 3.

DEIM basis (m)	Error	Runtime
10	4.3498×10^{-3}	0.012784 s
20	7.4722×10^{-4}	0.015815 s
30	4.3474×10^{-6}	0.019955 s
40	7.8929×10^{-10}	0.021829 s
50	1.7124×10^{-10}	0.023489 s
60	2.4349×10^{-11}	0.023178 s
70	2.0649×10^{-11}	0.024868 s

For the POD-GPOD reduced system, we computed the runtime and error from (4.1), as shown in Table 4.11. Error of the POD-GPOD reduced systems with POD basis of dimension $k = 50$ for Square Block Initial Data of Example 3, where m is the dimension of nonlinear basis, and q is the number of selected rows.

Table 4.11: Runtime and error of the POD-GPOD reduced systems for Example 3.

m	q	Error	Runtime
10	40	2.5195×10^{-3}	0.028002 s
20	50	1.9981×10^{-4}	0.028197 s
30	60	1.0603×10^{-7}	0.030217 s
40	70	1.7405×10^{-11}	0.030498 s
50	80	1.2332×10^{-11}	0.031554 s
60	90	3.8857×10^{-12}	0.032192 s
70	100	1.2603×10^{-13}	0.033486 s

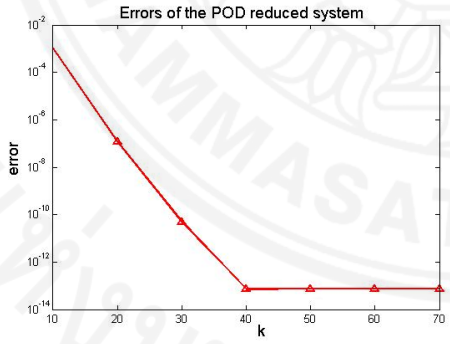

Figure 4.23: Error plot of the approximation from POD reduced system with different dimensions of POD basis k for Example 3.

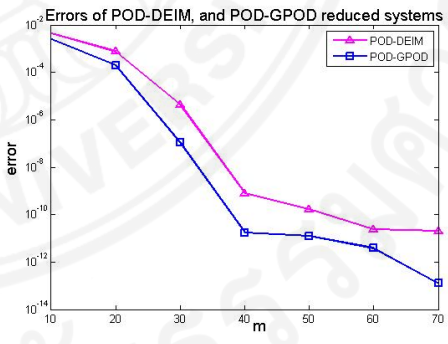

Figure 4.24: Error plot of the approximation the POD-DEIM, and POD-GPOD reduced systems with dimension of DEIM or nonlinear basis m for Example 3.

From Figures 4.21 and 4.22, the decreasing of singular values of the snapshot matrices $\mathbf{S}, \mathbf{F}$ and then beginning to stabilize implies that solution information lies within a subspace whose dimension is significantly lower than the full dimension used in the discretization. That is, we can reduce the dimensions of the full discretized system. It can be observed that the decrease of the error in Figure 4.23 corresponds to the increase of dimension of POD basis in Table 4.9. At a fixed POD dimension of $k = 50$, the decrease of the error corresponds to the increase of dimension of DEIM basis in Table 4.10. At a fixed POD dimension of $k = 50$, the decrease of the error corresponds to the increase of dimension of nonlinear basis, and the number of selected rows in Table 4.11. In Figure 4.24, the error for the POD-GPOD reduced system is decreased more than the error from the POD-DEIM reduced system. The runtime of the full discretized system with $\epsilon = 0.01$ is 0.510052 seconds. The average runtime of the reduced systems by POD approach was almost 4.2 times less than the runtime of the full discretized system (Table 4.9). The average runtime using POD-DEIM approach was almost 33.3 times less than the runtime of the full discretized system (Table 4.10). The average runtime using POD-GPOD approach was almost 19 times less than the runtime of the full discretized system (Table 4.11).

In numerical Example 3, we concluded that the initial condition in example 2 can be changed to discontinuous initial condition, and reduced computational cost with the same boundary conditions.

Table 4.12: The ratio of the runtime of these systems to the runtime of the full discretized system, and error of these systems for Example 3 with $k = 50$, $m = 60$, and $q = 90$.

System	Error	Ratio of runtime
The full discretized system	-	1
The POD reduced system	$\mathcal{O}(10^{-14})$	$1/4.4$
The POD-DEIM reduced system	$\mathcal{O}(10^{-11})$	$1/22$
The POD-GPOD reduced system	$\mathcal{O}(10^{-12})$	$1/16$

From Tables 4.4, 4.8, and 4.12, the ratio of the runtime of the full discretized system is scaled to be 1. The POD reduced system is more accurate than the POD-DEIM, and POD-GPOD reduced systems. However, the POD reduced system may not efficiently decrease the computational time. When DEIM and GPOD are used, respectively, to obtain the POD-DEIM and POD-GPOD reduced systems, the computational time can substantially decrease further. The POD-GPOD reduced system is relatively more accurate than the POD-DEIM reduced system. However, the POD-DEIM reduced system uses less computational time than the POD-GPOD reduced system.

4.2 Numerical examples with various parameter values

This section consists of parameter variation for two different initial conditions as used for Example 1 and Example 2 in Section 4.1. Example 1 is the initial boundary value problem of this equation, with homogeneous boundary conditions. Example 2 is the initial boundary value problem with non-homogeneous boundary conditions.

4.2.1 Parameter Variation of Example 1 (Homogeneous boundary conditions)

In this subsection, we consider the reduced systems using various parameters. We used snapshots from the full systems with $\epsilon = 0.01, 0.99$, as shown in Figure 4.25 and 4.26, respectively, to construct snapshot matrices $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{F}_1$, and $\mathbf{F}_2$. These snapshot matrices were used to construct basis sets for constructing reduced models of full systems with different parameter values ϵ, i.e. $\epsilon = 0.2, 0.5, 0.8$ as shown in Figures 4.27, 4.28, 4.29, respectively. Notice from Figures 4.25-4.29 that are the behaviors of the solutions are clearly different as the parameter changes.

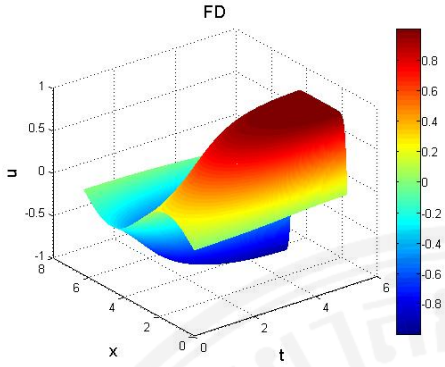

Figure 4.25: Solution of the full discretized system for Example 1 with $\epsilon = 0.01$.

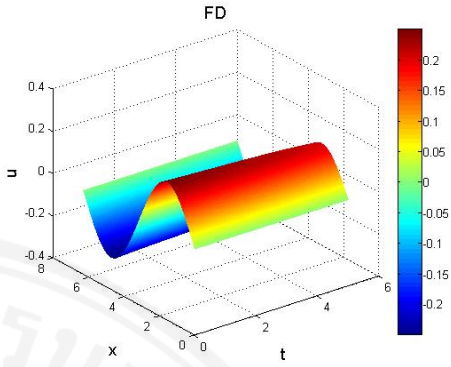

Figure 4.26: Solution of the full discretized system for Example 1 with $\epsilon = 0.99$.

Test parameter variation

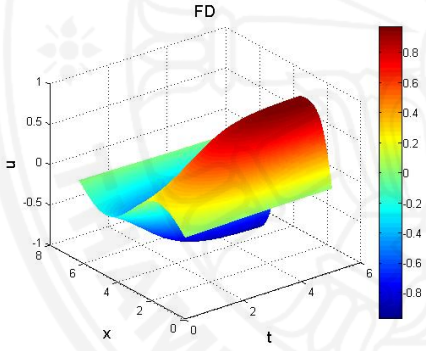

Figure 4.27: Solution of the full discretized system for Example 1 with $\epsilon = 0.2$.

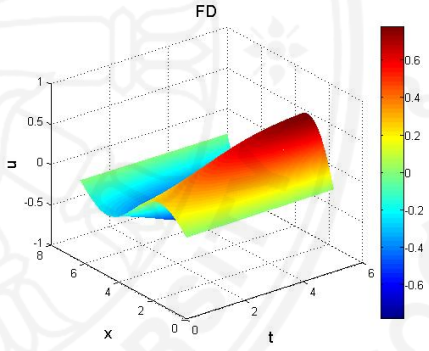

Figure 4.28: Solution of the full discretized system for Example 1 with $\epsilon = 0.5$.

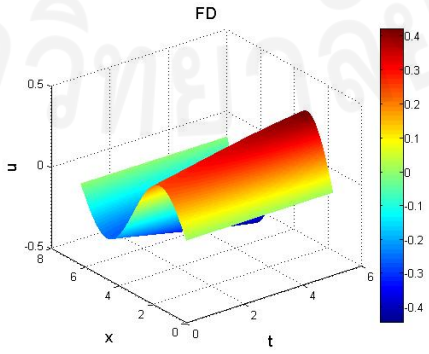

Figure 4.29: Solution of the full discretized system for Example 1 with $\epsilon = 0.8$.

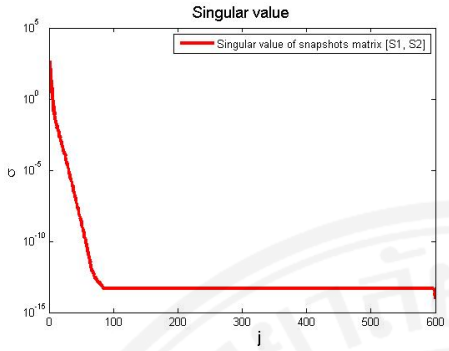

Figure 4.30: Plot of the singular values of the snapshot matrix $[\mathbf{S}_1, \mathbf{S}_2]$ from SVD using POD basis for Example 1.

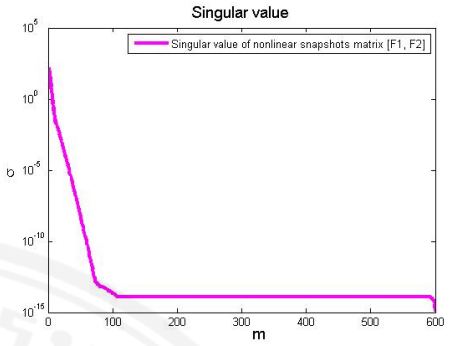

Figure 4.31: Plot of the singular values of the nonlinear snapshot matrix $[\mathbf{F}_1, \mathbf{F}_2]$ from SVD using DEIM basis for Example 1.

In Table 4.13, Table 4.14, and Table 4.15, we present average runtimes and average errors from the parameter variation using $\epsilon = 0.2, 0.5$, and 0.8, for POD, POD-DEIM, and POD-GPOD approaches, respectively. We computed the runtime and error from (4.1).

Table 4.13: Average runtime and average error of the POD reduced systems with various parameter values for Example 1.

POD basis (k)	Average error	Average runtime
10	2.9347×10^{-3}	3.455903 s
20	2.1171×10^{-4}	3.597402 s
30	2.1341×10^{-5}	3.777328 s
40	3.8210×10^{-7}	3.987502 s
50	5.0148×10^{-8}	4.044850 s
60	4.9060×10^{-8}	4.052223 s
70	3.7179×10^{-9}	4.055364 s

Table 4.14: Average runtime and average error of the POD-DEIM reduced systems with various parameter values and POD basis with dimension $k = 50$ for Example 1.

DEIM basis (m)	Average error	Average runtime
10	2.4654×10^{-3}	0.016540 s
20	2.8762×10^{-4}	0.017132 s
30	2.3596×10^{-4}	0.019433 s
40	2.3799×10^{-5}	0.021238 s
50	5.6333×10^{-6}	0.023897 s
60	1.8494×10^{-7}	0.025333 s
70	4.0323×10^{-8}	0.027674 s

Table 4.15 shows the average error of the POD-GPOD reduced systems with POD basis of dimension $k = 50$ for Example 1, where m is the dimension of nonlinear basis, and q is the number of selected rows.

Table 4.15: Average runtime and average error of the POD-GPOD reduced systems with various parameter values for Example 1.

m	q	Average error	Average runtime
10	40	7.6060×10^{-4}	0.033527 s
20	50	1.6507×10^{-4}	0.034459 s
30	60	4.7335×10^{-5}	0.034520 s
40	70	3.3937×10^{-6}	0.034978 s
50	80	2.7384×10^{-7}	0.035124 s
60	90	1.3214×10^{-7}	0.035971 s
70	100	1.2307×10^{-8}	0.036525 s

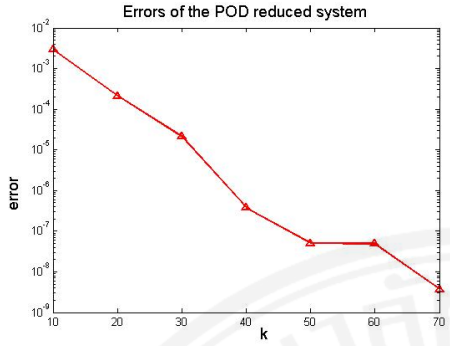

Figure 4.32: Average error plot of the approximation from POD reduced system with different dimensions of POD basis k for Example 1.

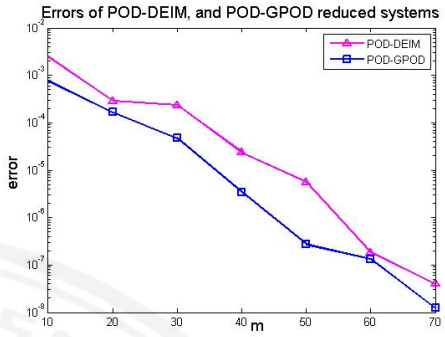

Figure 4.33: Average error plot of the approximation the POD-DEIM, and POD-GPOD reduced systems with dimension of DEIM or nonlinear basis m for Example 1.

Similar to the previous examples, From Figures 4.30 and 4.31, the decreasing of singular values of the snapshot matrices $\mathbf{S}, \mathbf{F}$ and then beginning to stabilize implies that solution information lies within a subspace whose dimension is significantly lower than the full dimension used in the discretization. That is, we can reduce the dimensions of the full discretized system. From Table 4.13, Table 4.14, and Table 4.15, we conclude that POD, POD-DEIM, and POD-GPOD can be used to reduce computational cost in the simulation at various parameter values. It can be observed that the decrease of the average error in Figure 4.32 corresponds to the increase of dimension of POD basis in Table 4.13. At a fixed POD dimension of $k = 50$, the decrease of the average error corresponds to the increase of dimension of DEIM basis in Table 4.14. At a fixed POD dimension of $k = 50$, the decrease of the average error corresponds to the increase of dimension of nonlinear basis, and the number of selected rows in Table 4.15. In Figure 4.33, the average error for the POD-GPOD reduced system is decreased more than the average error from the POD-DEIM reduced system. The average runtime of the full discretized systems with various parameter values $\epsilon = 0.2, 0.5$ and 0.8 is 8.645238 seconds. The average runtime of the reduced systems by POD approach was almost 2.32 times less than the average runtime of the full discretized system with various parameter values (Table 4.13). The average runtime using POD-DEIM approach was almost 410 times less than the average runtime of the full discretized system with var-

ious parameter values (Table 4.14). The average runtime using POD-GPOD approach was almost 253 times less than the average runtime of the full discretized system with various parameter values (Table 4.15). Table 4.16 summarizes the accuracy and the scaled computational time of the reduced systems when compared with the full-order system (the runtime of full-order system is normalized to be one).

Table 4.16: The ratio of the average runtime of these systems to the runtime of the full discretized system, and average error of these systems for Example 1 with various parameter values using $k = 50$, $m = 40$, and $q = 70$.

System	Average error	Ratio of average runtime
The full discretized system	-	1
The POD reduced system	$\mathcal{O}(10^{-8})$	$1/2.1$
The POD-DEIM reduced system	$\mathcal{O}(10^{-5})$	$1/411$
The POD-GPOD reduced system	$\mathcal{O}(10^{-6})$	$1/254$

4.2.2 Parameter Variation of Example 2 (Non-homogeneous boundary conditions)

In this subsection, we consider the reduced systems using various parameter values. We used snapshots from the full systems with $\epsilon = 0.011, 0.009$, as shown in Figure 4.34 and Figure 4.35, respectively, to construct snapshot matrices $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{F}_1$, and $\mathbf{F}_2$. These snapshot matrices were used to construct basis sets for constructing reduced models of full systems with different parameter values ϵ, i.e. $\epsilon = 0.0095, 0.01, 0.0105$, as shown in Figures 4.36, 4.37, 4.38, respectively. Notice from Figures 4.34-4.38 that are the behaviors of the solutions are clearly different as the parameter changes.

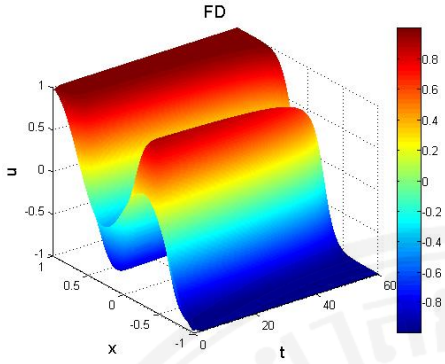

Figure 4.34: Solution of the full discretized system for Example 2 with $\epsilon = 0.009$.

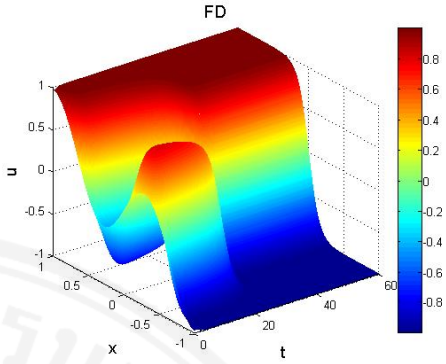

Figure 4.35: Solution of the full discretized system for Example 2 with $\epsilon = 0.011$.

Test parameter variation

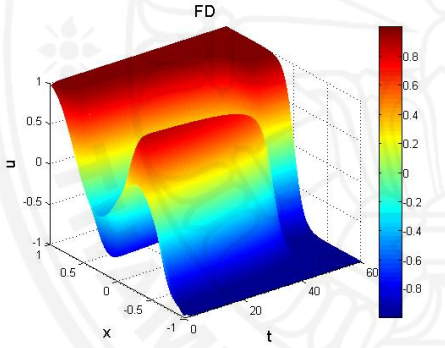

Figure 4.36: Solution of the full discretized system for Example 2 with $\epsilon = 0.0095$.

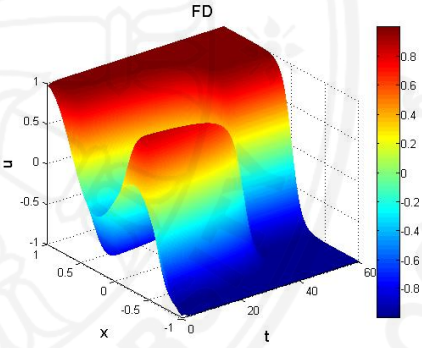

Figure 4.37: Solution of the full discretized system for Example 2 with $\epsilon = 0.01$.

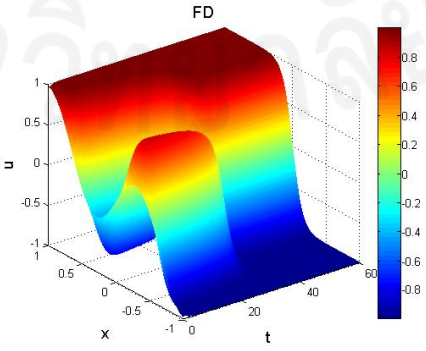

Figure 4.38: Solution of the full discretized system for Example 2 with $\epsilon = 0.0105$.

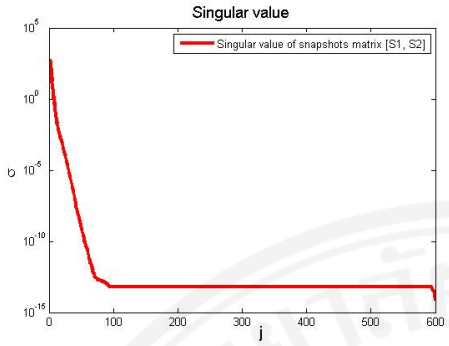

Figure 4.39: Plot of the singular values of the snapshot matrix $[\mathbf{S}_1, \mathbf{S}_2]$ from SVD using POD basis for Example 2.

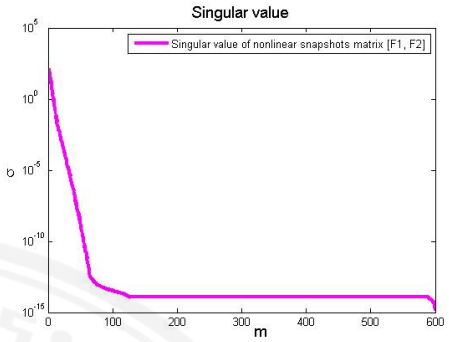

Figure 4.40: Plot of the singular values of the nonlinear snapshot matrix $[\mathbf{F}_1, \mathbf{F}_2]$ from SVD using DEIM basis for Example 2.

In Table 4.17, Table 4.18, and Table 4.19, we present average runtimes and average errors from the parameter variation using $\epsilon = 0.0105, 0.01$, and 0.0095, for POD, POD-DEIM, and POD-GPOD approaches, respectively. We computed the runtime and error from (4.1).

Table 4.17: Average runtime and average error of the POD reduced systems with various parameter values for Example 2.

POD basis (k)	Average error	Average runtime
10	6.9201×10^{-5}	3.332072 s
20	1.3413×10^{-6}	3.543711 s
30	1.2395×10^{-7}	3.633455 s
40	4.2427×10^{-8}	3.633789 s
50	4.7703×10^{-9}	3.699877 s
60	3.0531×10^{-9}	3.700031 s
70	3.0827×10^{-10}	3.834290 s

Table 4.18: Average runtime and average error of the POD-DEIM reduced systems with various parameter values and POD basis with dimension $k = 50$ for Example 2.

DEIM basis (m)	Average error	Average runtime
10	2.8847×10^{-3}	0.016453 s
20	1.7506×10^{-3}	0.015786 s
30	3.5277×10^{-4}	0.016378 s
40	4.8921×10^{-5}	0.017002 s
50	2.8691×10^{-6}	0.021445 s
60	1.9001×10^{-7}	0.022776 s
70	4.7991×10^{-8}	0.024003 s

Table 4.19 shows the average error of the POD-GPOD reduced systems with POD basis of dimension $k = 50$ for Example 2, where m is the dimension of nonlinear basis ,and q is the number of selected rows.

Table 4.19: Average runtime and average error of the POD-GPOD reduced systems with various parameter values for Example 2.

m	q	Average error	Average runtime
10	40	1.8960×10^{-3}	0.038985 s
20	50	5.4464×10^{-4}	0.039494 s
30	60	7.5895×10^{-5}	0.040552 s
40	70	1.1070×10^{-6}	0.041449 s
50	80	2.4638×10^{-7}	0.042578 s
60	90	1.3442×10^{-8}	0.043955 s
70	100	3.8939×10^{-9}	0.045869 s

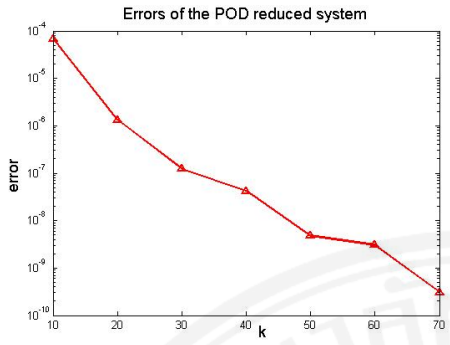

Figure 4.41: Average error plot of the approximation from POD reduced system with different dimensions of POD basis k for Example 2.

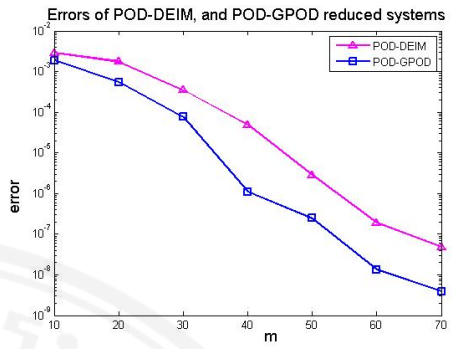

Figure 4.42: Average error plot of the approximation the POD-DEIM, and POD-GPOD reduced systems with dimension of DEIM or nonlinear basis m for Example 2.

Similar to the previous examples, From Figures 4.39 and 4.40, the decreasing of singular values of the snapshot matrices $\mathbf{S}, \mathbf{F}$ and then beginning to stabilize implies that solution information lies within a subspace whose dimension is significantly lower than the full dimension used in the discretization. Then we can decrease dimensions of the full discretized system. From Table 4.17, Table 4.18, and Table 4.19, we conclude that POD, POD-DEIM, and POD-GPOD can be used to reduce computational cost in the simulation at various parameter values. It can be observed that the decrease of the average error in Figure 4.41 corresponds to the increase of dimension of POD basis in Table 4.17. At a fixed POD dimension of $k = 50$, the decrease of the average error corresponds to the increase of dimension of DEIM basis in Table 4.18. At a fixed POD dimension of $k = 50$, the decrease of the average error corresponds to the increase of dimension of nonlinear basis, and the number of selected rows in Table 4.19. In Figure 4.42, the average error for the POD-GPOD reduced system is decreased more than the average error from the POD-DEIM reduced system. The average runtime of the full discretized systems with various parameter values $\epsilon = 0.0095, 0.01$ and 0.0105 is 11.226242 seconds. The average runtime of the reduced systems by POD approach was almost 3.11 times less than the average runtime of the full discretized system with various parameter values (Table 4.17). The average runtime using POD-DEIM approach was almost 622.2 times less than the average runtime of the full discretized system with

various parameter values (Table 4.18).

Table 4.20: The ratio of the average runtime of these systems to the runtime of the full discretized system, and average error of these systems for Example 2 with various parameter values using $k = 50$, $m = 60$, and $q = 90$.

System	Average error	Ratio of average runtime
The full discretized system	-	1
The POD reduced system	$\mathcal{O}(10^{-9})$	$1/3.0$
The POD-DEIM reduced system	$\mathcal{O}(10^{-7})$	$1/534$
The POD-GPOD reduced system	$\mathcal{O}(10^{-8})$	$1/261$

From Tables 4.16, and 4.20, the ratio of the average runtime of the full discretized system is normalized to be 1. The POD reduced system is more accurate than the POD-DEIM, and POD-GPOD reduced systems. However, the POD reduced system may not efficiently decrease the computational time. The DEIM and GPOD can be incorporated to drastically reduced the computation time further. The POD-GPOD reduced system is relatively more accurate than the POD-DEIM reduced system. However, the POD-DEIM reduced system uses less computational time than the POD-GPOD reduced system.

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this thesis, we construct reduced order models for the Allen-Cahn equation by POD, POD-DEIM and POD-GPOD approaches. We conclude that three methods can be used to reduce computational cost in the simulation. They can accurately give approximate solutions to the original system with various parameter values. Since the number of unknowns in the original system can be substantially reduced by these methods, then this can reduce the memory space used in the simulation. The POD reduced system is more accurate than the POD-DEIM reduced system. However, the POD-DEIM reduced system can be used to reduced the computational complexity in the nonlinear terms, and therefore the runtime for solving is less than the POD reduced system. We extend the POD-DEIM reduced system to the POD-GPOD reduced system by increasing the selected rows. In most cases, the POD-GPOD reduced system can further reduce the approximation error when compare with the POD-DEIM reduced system.

Future Work

The following are possible extensions of this work.

- Extend this work to construct a reduced model of Allen-Cahn equation in two, and three spatial dimensions by these three methods.
- Extend this work to many applications of the Allen-Cahn equation, such as image analysis, crystal growth, and the motion by mean curvature flow in the context of reduced model.
- Extend this work to construct a reduced model of a more complicated Cahn-Hilliard (C-H) equation by these three methods.

APPENDIX A

CONVERGENCE OF THE θ METHOD

To solve the discretized system in the previous subsection, we considered convergence using the θ method, as follows.

The forward Euler, backward Euler and Crank-Nicolson methods are special cases of the θ method. In this subsection, we consider a convergence proof of the θ method to confirm the convergence for numerical results of the discretized system in the previous subsection. In particular, we will show that the solution of the following discretized system will converge to the exact solution in (3.1) by using the proof from [20], as follows.

$$u_{j+1} = u_j + \Delta t[\theta f(u_j, t_j) + (1-\theta) f(u_{j+1}, t_{j+1})] \tag{A.1}$$

where $j = 0, 1, \ldots$ and $\theta \in [0, 1]$. Note that the forward Euler method ($\theta = 1$) and backward Euler method ($\theta = 0$), both converge to order 1. When $\theta = 1/2$ we use the Crank-Nicolson method, which converges to order 2.

We divide the proof into two steps.

Step 1 We substitute the exact solution $u(t)$

$$u(t_{j+1}) - u(t_j) - \Delta t[\theta f(u(t_j), t_j) + (1-\theta) f(u(t_{j+1}), t_{j+1})]$$
$$= u(t_{j+1}) - u(t_j) - \Delta t[\theta u'(t_j) + (1-\theta) u'(t_{j+1})]$$

Next, using the Taylor expansion, we have

$$\begin{aligned} &u(t_{j+1}) - u(t_j) - \Delta t[\theta f(u(t_j), t_j) + (1-\theta) f(u(t_{j+1}), t_{j+1})] \\ &= [u(t_j) + (\Delta t) u'(t_j) + \frac{1}{2}(\Delta t)^2 u''(t_j) + \frac{1}{6}(\Delta t)^3 u'''(t_j)] \\ &\quad - u(t_j) - \Delta t\{\theta u'(t_j) + (1-\theta)[u'(t_j) + (\Delta t) u''(t_j) + \frac{1}{2}(\Delta t)^2 u'''(t_j)]\} + \mathcal{O}((\Delta t)^4) \\ &= \left(\theta - \frac{1}{2}\right)(\Delta t)^2 u''(t^j) + \left(\frac{1}{2}\theta - \frac{1}{3}\right)(\Delta t)^3 u'''(t_j) + \mathcal{O}((\triangle t)^4). \end{aligned} \tag{A.2}$$

Step 2 From (A.1), we have

$$u_{j+1} - u_j - \Delta t[\theta f(u_j, t_j) + (1-\theta) f(u_{j+1}, t_{j+1})] = 0, \tag{A.3}$$

Consider (A.3)-(A.2) and let $e_i = u_i - u(t_i)$, we obtain that for $(\Delta t) > 0$ is small sufficiently

$$\begin{aligned} &e_{j+1,\Delta t} - e_{j,\Delta t} - \theta(\Delta t)[f(u(t_j) + e_{j,\Delta t}, t_j) - f(u(t_j), t_j)] \\ &\quad - (1-\theta)(\Delta t)[u(u(t_{j+1}) + e_{j+1,\Delta t}, t_{j+1}) - f(u(t_{j+1}), t_{j+1})] \\ &= \left(\theta - \frac{1}{2}\right)(\Delta t)^2 u''(t^j) + \left(\frac{1}{2}\theta - \frac{1}{3}\right)(\Delta t)^3 u'''(t_j) + \mathcal{O}((\Delta t)^4). \end{aligned} \tag{A.4}$$

Therefore,

$$\begin{aligned} &e_{j+1,\Delta t} \\ &= e_{j,\Delta t} + \theta(\Delta t)[f(u(t_j) + e_{j,\Delta t}, t_j) - f(u(t_j), t_j)] \\ &\quad + (1-\theta)(\Delta t)[u(u(t_{j+1}) + e_{j+1,\Delta t}, t_{j+1}) - f(u(t_{j+1}), t_{j+1})] \\ &= \left(\theta - \frac{1}{2}\right)(\Delta t)^2 u''(t^j) + \left(\frac{1}{2}\theta - \frac{1}{3}\right)(\Delta t)^3 u'''(t_j) + \mathcal{O}((\Delta t)^4) \\ &\begin{cases} -\frac{1}{12}(\Delta t)^3 u'''(t_j) + \mathcal{O}((\Delta t)^4), & \theta = \frac{1}{2} \\ +(\theta - \frac{1}{2})(\Delta t)^2 u''(t_j) + \mathcal{O}((\Delta t)^3), & \theta \neq \frac{1}{2} \end{cases} \end{aligned} \tag{A.5}$$

Using the triangle inequality and by the Lipschitz continuity of f, there exist constants

c and λ such that

$$\|e_{j+1,\Delta t}\| \tag{A.6}$$
$$\leq \|e_{j,\Delta t}\| + \theta(\Delta t)\lambda \|e_{j,\Delta t}\| + (1-\theta)(\Delta t)\lambda \|e_{j+1,\Delta t}\| + \begin{cases} c(\Delta t)^3 & \theta = \frac{1}{2} \\ c(\Delta t)^2 & \theta \neq \frac{1}{2} \end{cases}.$$

Case 1 For $\theta = \frac{1}{2}$, the theta method reduces to the trapezoidal rule. It is possible to show that the Crank-Nicolson method has second order convergence, we have studied proof of convergence from Iserles [35].

$$\|e_{j+1,\Delta t}\| \leq \|e_{j,\Delta t}\| + \frac{1}{2}(\Delta t)\lambda \|e_{j,\Delta t}\| + \frac{1}{2}(\Delta t)\lambda \|e_{j+1,\Delta t}\| + c(\Delta t)^3$$

$$\|e_{j+1,\Delta t}\| - \frac{1}{2}(\Delta t)\lambda \|e_{j+1,\Delta t}\| \leq \|e_{j,\Delta t}\| + \frac{1}{2}(\Delta t)\lambda \|e_{j,\Delta t}\| + c(\triangle t)^3$$

$$\left(1 - \frac{1}{2}(\Delta t)\lambda\right) \|e_{j+1,\Delta t}\| \leq \left(1 + \frac{1}{2}(\Delta t)\lambda\right) \|e_{j,\Delta t}\| + c(\Delta t)^3$$

$$\|e_{j+1,\Delta t}\| \leq \left[\frac{1 + \frac{1}{2}(\Delta t)\lambda}{1 - \frac{1}{2}(\Delta t)\lambda}\right] \|e_{j,\Delta t}\| + \left[\frac{c}{1 - \frac{1}{2}(\Delta t)\lambda}\right] (\Delta t)^3. \tag{A.7}$$

We claim that

$$\|e_{j,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1 + \frac{1}{2}(\Delta t)\lambda}{1 - \frac{1}{2}(\Delta t)\lambda}\right)^j - 1\right](\Delta t)^2 \tag{A.8}$$

by using math induction

Basic step For $j = 0$ we will prove that $\|e_{0,\Delta t}\| = 0$ since at t_0 the numerical solution correspond to the initial condition and error is 0.

$$\|e_{0,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1 + \frac{1}{2}(\Delta t)\lambda}{1 - \frac{1}{2}(\Delta t)\lambda}\right)^0 - 1\right](\Delta t)^2 = 0$$

Induction step We will prove that if (A.8) is true for $j = k$ with $k \in \mathbb{Z}^+ \cup \{0\}$ then (A.8) is true for $j = k + 1$.

Assume that (A.8) is true for $j = k$ with $k \in \mathbb{Z}^+ \cup \{0\}$, prove that (A.8) is true for $j = k + 1$.

$$\|e_{k+1,\Delta t}\| \leq \left[\frac{1 + \frac{1}{2}(\Delta t)\lambda}{1 - \frac{1}{2}(\Delta t)\lambda}\right] \|e_{k,\Delta t}\| + \left[\frac{c}{1 - \frac{1}{2}(\Delta t)\lambda}\right] (\Delta t)^3. \tag{A.9}$$

From assumption, we have

$$\|e_{k,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)^{k}-1\right](\Delta t)^2$$

Substitution $\|e_{k,\Delta t}\|$ into (A.9), we obtain

$$\|e_{k+1,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)^{k+1}-\left(\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)\right](\Delta t)^2+\left[\frac{c}{1-\frac{1}{2}(\Delta t)\lambda}\right](\Delta t)^3.$$

Therefore,

$$\|e_{k+1,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)^{k+1}-1\right](\Delta t)^2. \tag{A.10}$$

By math induction, we conclude that (A.8) for all $j = 0, 1, ...$ is true.

Note that

$$\begin{aligned}\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda} &= 1+\frac{(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\\ &\leq \exp\left(\frac{(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)\end{aligned}$$

By a Taylor expansion of the exponential function, we have

$$\begin{aligned}\|e_{j,\Delta t}\| &\leq \frac{c}{\lambda}\left[\left(\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)^{j}-1\right](\Delta t)^2\\ &\leq \frac{c}{\lambda}\left(\frac{1+\frac{1}{2}(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right)^{j}(\Delta t)^2\\ &\leq \frac{c(\Delta t)^2}{\lambda}\exp\left(\frac{n(\Delta t)\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right).\end{aligned}$$

This bound is true for all $j = 0, 1, ...$ such that $j(\triangle t) \leq t^*$. Therefore

$$\|e_{j,\Delta t}\| \leq \frac{c(\Delta t)^2}{\lambda}\exp\left(\frac{t^*\lambda}{1-\frac{1}{2}(\Delta t)\lambda}\right) \tag{A.11}$$

and we obtain $\lim_{(\Delta t)\to 0}\|e_{j,\Delta t}\| = 0$ and $0 \leq j(\Delta t) \leq t^*$.

We conclude that the θ method is convergent order 2 for $\theta = \frac{1}{2}$ which mean that Crank-Nicolson method has second order convergence.

Case 2 For $\theta \neq \frac{1}{2}$, we have

$$\|e_{j+1,\Delta t}\| \leq \|e_{j,\Delta t}\| + \theta(\Delta t)\lambda\|e_{j,\Delta t}\| + (1-\theta)(\Delta t)\lambda\|e_{j+1,\Delta t}\| + c(\Delta t)^2$$

$$\|e_{j+1,\Delta t}\| - (1-\theta)(\Delta t)\lambda \|e_{j+1,\Delta t}\| \leq \|e_{j,\Delta t}\| + \theta(\Delta t)\lambda \|e_{j,\Delta t}\| + c(\Delta t)^2$$

$$(1-(1-\theta)(\Delta t)\lambda) \|e_{j+1,\Delta t}\| \leq (1+\theta(\Delta t)\lambda) \|e_{j,\Delta t}\| + c(\Delta t)^2$$

$$\|e_{j+1,\Delta t}\| \leq \left[\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right] \|e_{j,\Delta t}\| + \left[\frac{c}{1-(1-\theta)(\Delta t)\lambda}\right] (\Delta t)^2. \quad \text{(A.12)}$$

We claim that

$$\|e_{j,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^j - 1\right](\Delta t) \quad \text{(A.13)}$$

by using math induction

Basic step For $j = 0$ we will prove that $\|e_{0,\Delta t}\| = 0$ since at t_0 the numerical solution correspond to the initial condition and error is 0.

$$\|e_{0,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^0 - 1\right](\Delta t) = 0$$

Induction step We will prove that if (A.13) is true for $j = k$ with $k \in \mathbb{Z}^+ \cup \{0\}$ then (A.13) is true for $j = k+1$.

Assume that (A.13) is true for $j = k$ with $k \in \mathbb{Z}^+ \cup \{0\}$, prove that (A.13) is true for $j = k+1$.

$$\|e_{k+1,\Delta t}\| \leq \left[\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right] \|e_{k,\Delta t}\| + \left[\frac{c}{1-(1-\theta)(\Delta t)\lambda}\right] (\Delta t)^2. \quad \text{(A.14)}$$

From assumption, we have

$$\|e_{k,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^k - 1\right](\Delta t)$$

Substitution $\|e_{k,\Delta t}\|$ into (A.14), we obtain

$$\begin{aligned}\|e_{k+1,\Delta t}\| \leq & \frac{c}{\lambda}\left[\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^{k+1} - \left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)\right](\Delta t)^2 \\ & + \left[\frac{c}{1-(1-\theta)(\Delta t)\lambda}\right](\Delta t)^3.\end{aligned}$$

Therefore,

$$\|e_{k+1,\Delta t}\| \leq \frac{c}{\lambda}\left[\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^{k+1} - 1\right](\Delta t). \quad \text{(A.15)}$$

By math induction, we conclude that (A.13) for all $j = 0, 1, ...$ is true.

Note that

$$\begin{aligned}\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda} &= 1+\frac{(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda} \\ &\leq \exp\left(\frac{(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)\end{aligned}$$

By a Taylor expansion of the exponential function, we have

$$\begin{aligned}\|e_{j,\Delta t}\| &\leq \frac{c}{\lambda}\left[\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^j - 1\right](\Delta t) \\ &\leq \frac{c}{\lambda}\left(\frac{1+\theta(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right)^j (\Delta t) \\ &\leq \frac{c(\Delta t)^2}{\lambda}\exp\left(\frac{n(\Delta t)\lambda}{1-(1-\theta)(\Delta t)\lambda}\right).\end{aligned}$$

This bound is true for all $j = 0, 1, ...$ such that $j(\Delta t) \leq t^*$. Therefore

$$\|e_{j,\Delta t}\| \leq \frac{c(\Delta t)^2}{\lambda}\exp\left(\frac{t^*\lambda}{1-(1-\theta)(\Delta t)\lambda}\right) \tag{A.16}$$

and we obtain $\lim_{(\Delta t)\to 0}\|e_{j,\Delta t}\| = 0$ and $0 \leq j(\Delta t) \leq t^*$.

We conclude that the θ method is convergent order 1 for $\theta \neq \frac{1}{2}$.

Q.E.D.

APPENDIX B

ADDITIONAL NUMERICAL STUDY FOR THE POD-GPOD REDUCED SYSTEM.

We present numerical error (4.1) from using POD-GPOD approach with different dimension of m, and q. Table B.1, B.2, and B.3 are used to describe more of the error from Section 4.1. Table B.4, and B.5 are used to describe more of the error from Section 4.2. These will be shown in the following Tables.

Table B.1: Error of the POD-GPOD reduced system for Example 1 from Section 4.1.

Error				
q	$m = 10$	$m = 20$	$m = 30$	$m = 40$
10	3.2007×10^{-3}	-	-	-
20	2.7355×10^{-3}	3.1434×10^{-4}	-	-
30	3.1952×10^{-3}	4.2512×10^{-4}	1.1348×10^{-4}	-
40	3.1224×10^{-3}	4.9270×10^{-4}	6.8405×10^{-5}	1.4441×10^{-5}
50	3.2370×10^{-3}	4.8364×10^{-4}	3.2637×10^{-5}	5.6958×10^{-6}
60	3.0286×10^{-3}	4.8772×10^{-4}	3.9352×10^{-5}	5.4505×10^{-6}
70	2.8869×10^{-3}	4.9516×10^{-4}	3.4253×10^{-5}	3.7254×10^{-6}
80	2.8860×10^{-3}	4.3732×10^{-4}	2.9064×10^{-5}	2.8278×10^{-6}
90	2.8932×10^{-3}	4.1485×10^{-4}	2.9058×10^{-5}	2.5623×10^{-6}
100	2.7776×10^{-3}	3.7542×10^{-4}	1.6710×10^{-5}	1.3426×10^{-6}

Error			
q	$m = 50$	$m = 60$	$m = 70$
10	-	-	-
20	-	-	-
30	-	-	-
40	-	-	-
50	2.4285×10^{-6}	-	-
60	2.6269×10^{-6}	2.7139×10^{-6}	-
70	8.4438×10^{-7}	7.7370×10^{-7}	7.6205×10^{-7}
80	9.5442×10^{-7}	7.5597×10^{-7}	7.2166×10^{-7}
90	7.7595×10^{-7}	6.7214×10^{-7}	6.7197×10^{-7}
100	7.5511×10^{-7}	6.5284×10^{-7}	6.5710×10^{-7}

Table B.2: Error of the POD-GPOD reduced system for Example 2 from Section 4.1.

Error				
q	$m = 10$	$m = 20$	$m = 30$	$m = 40$
10	8.7695×10^{-3}	-	-	-
20	7.8165×10^{-3}	7.8254×10^{-4}	-	-
30	6.9890×10^{-3}	6.1443×10^{-4}	1.1173×10^{-4}	-
40	6.8678×10^{-3}	6.1032×10^{-4}	1.0765×10^{-4}	4.2691×10^{-5}
50	6.0353×10^{-3}	5.3332×10^{-4}	3.2425×10^{-5}	1.3775×10^{-5}
60	5.3560×10^{-3}	5.6321×10^{-4}	4.6130×10^{-5}	9.6770×10^{-6}
70	5.0024×10^{-3}	5.9600×10^{-4}	7.2155×10^{-6}	5.1778×10^{-6}
80	1.1876×10^{-3}	2.3362×10^{-4}	3.0358×10^{-6}	4.9910×10^{-6}
90	4.7997×10^{-3}	1.2514×10^{-4}	2.8365×10^{-6}	2.1076×10^{-6}
100	9.4457×10^{-4}	1.4875×10^{-4}	2.8209×10^{-6}	1.1117×10^{-6}

Error			
q	$m = 50$	$m = 60$	$m = 70$
10	-	-	-
20	-	-	-
30	-	-	-
40	-	-	-
50	3.2536×10^{-6}	-	-
60	2.0292×10^{-6}	1.0416×10^{-6}	-
70	1.1564×10^{-6}	3.2632×10^{-7}	1.1034×10^{-7}
80	1.8131×10^{-6}	1.4639×10^{-7}	6.3690×10^{-8}
90	9.5989×10^{-7}	2.9794×10^{-8}	2.9781×10^{-8}
100	1.9445×10^{-7}	2.0511×10^{-8}	2.9774×10^{-8}

Table B.3: Error of the POD-GPOD reduced system for Example 3 from Section 4.1.

Error				
q	$m = 10$	$m = 20$	$m = 30$	$m = 40$
10	4.3498×10^{-3}	-	-	-
20	1.2829×10^{-2}	7.4722×10^{-4}	-	-
30	1.0246×10^{-2}	7.5332×10^{-4}	4.3474×10^{-6}	-
40	2.5195×10^{-3}	7.0818×10^{-4}	2.5416×10^{-7}	7.8929×10^{-10}
50	2.4891×10^{-3}	1.9981×10^{-4}	3.2579×10^{-7}	2.6499×10^{-10}
60	2.4247×10^{-3}	1.8897×10^{-4}	1.0603×10^{-7}	1.3038×10^{-10}
70	2.4752×10^{-3}	7.4719×10^{-4}	1.0130×10^{-7}	1.7405×10^{-11}
80	2.5075×10^{-3}	4.2085×10^{-4}	5.7263×10^{-7}	1.9551×10^{-11}
90	2.4228×10^{-3}	6.5241×10^{-5}	2.5399×10^{-7}	1.8703×10^{-11}
100	2.3278×10^{-3}	1.6994×10^{-5}	3.2568×10^{-8}	9.7813×10^{-12}

Error			
q	$m = 50$	$m = 60$	$m = 70$
10	-	-	-
20	-	-	-
30	-	-	-
40	-	-	-
50	1.7124×10^{-10}	-	-
60	1.1160×10^{-10}	2.4349×10^{-11}	-
70	3.5554×10^{-11}	2.3471×10^{-11}	2.0649×10^{-11}
80	1.2332×10^{-11}	6.8642×10^{-12}	1.2500×10^{-12}
90	3.6989×10^{-11}	3.8857×10^{-12}	1.6321×10^{-12}
100	2.7301×10^{-13}	2.1938×10^{-13}	1.2603×10^{-13}

Table B.4: Average error of the POD-GPOD reduced system for Example 1 from Section 4.2.

Error				
q	$m = 10$	$m = 20$	$m = 30$	$m = 40$
10	2.4654×10^{-3}	-	-	-
20	8.0732×10^{-4}	2.8762×10^{-4}	-	-
30	9.3199×10^{-4}	3.1133×10^{-4}	2.3596×10^{-4}	-
40	7.6060×10^{-4}	2.1971×10^{-4}	6.3986×10^{-5}	2.3799×10^{-5}
50	7.6459×10^{-4}	1.6507×10^{-4}	6.1776×10^{-5}	6.7683×10^{-6}
60	7.4493×10^{-4}	1.6831×10^{-4}	4.7335×10^{-5}	5.8129×10^{-6}
70	7.6857×10^{-4}	1.3604×10^{-4}	4.1197×10^{-5}	3.3937×10^{-6}
80	7.4903×10^{-4}	2.1732×10^{-4}	3.6629×10^{-5}	6.7683×10^{-6}
90	6.2336×10^{-4}	2.0104×10^{-5}	5.4091×10^{-6}	5.8129×10^{-6}
100	3.1668×10^{-4}	1.4326×10^{-5}	3.6654×10^{-6}	3.3937×10^{-6}

Error			
q	$m = 50$	$m = 60$	$m = 70$
10	-	-	-
20	-	-	-
30	-	-	-
40	-	-	-
50	5.6333×10^{-6}	-	-
60	1.0182×10^{-6}	1.8494×10^{-7}	-
70	7.0711×10^{-7}	1.5138×10^{-7}	4.0323×10^{-8}
80	2.7384×10^{-7}	1.5138×10^{-7}	3.7547×10^{-8}
90	2.0654×10^{-7}	1.3214×10^{-7}	2.5873×10^{-8}
100	1.5713×10^{-7}	3.9785×10^{-8}	1.2307×10^{-8}

Table B.5: Average error of the POD-GPOD reduced system for Example 2 from Section 4.2

Error				
q	$m = 10$	$m = 20$	$m = 30$	$m = 40$
10	2.8847×10^{-3}	-	-	-
20	2.4403×10^{-3}	1.7506×10^{-3}	-	-
30	2.5654×10^{-3}	7.6900×10^{-4}	3.5277×10^{-4}	-
40	1.8960×10^{-3}	6.0528×10^{-4}	3.4150×10^{-4}	4.8921×10^{-5}
50	1.9530×10^{-3}	5.4464×10^{-4}	3.3678×10^{-4}	2.1105×10^{-6}
60	1.8004×10^{-3}	2.8692×10^{-4}	7.5895×10^{-5}	1.6493×10^{-6}
70	1.8405×10^{-3}	2.5473×10^{-4}	6.7276×10^{-6}	1.1070×10^{-6}
80	1.8521×10^{-3}	2.4367×10^{-4}	5.8944×10^{-6}	1.3531×10^{-7}
90	1.9057×10^{-3}	5.4483×10^{-5}	3.4793×10^{-6}	1.2507×10^{-7}
100	1.7054×10^{-3}	3.9752×10^{-5}	5.6457×10^{-6}	1.1221×10^{-7}

Error			
q	$m = 50$	$m = 60$	$m = 70$
10	-	-	-
20	-	-	-
30	-	-	-
40	-	-	-
50	2.8691×10^{-6}	-	-
60	2.9083×10^{-7}	1.9001×10^{-7}	-
70	1.7283×10^{-7}	1.5467×10^{-8}	4.7991×10^{-8}
80	2.4638×10^{-7}	1.4790×10^{-8}	1.0473×10^{-8}
90	1.0346×10^{-7}	1.3442×10^{-8}	5.9645×10^{-9}
100	3.3975×10^{-8}	1.0395×10^{-9}	3.8939×10^{-9}

BIBLIOGRAPHY

[1] Vidal R. Algazi and David J. Sakrison. On the optimality of the karhunen-loève expansion. *Journal of Process Control*, pages 319–320, 1969.

[2] Nicholas D Alikakos, Peter W Bates, and Xinfu Chen. Convergence of the cahn-hilliard equation to the hele-shaw model. *Archive for rational mechanics and analysis*, 128(2):165–205, 1994.

[3] Samuel M Allen and John W Cahn. A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening. *Acta Metallurgica*, 27(6):1085–1095, 1979.

[4] David Amsallem and Charbel Farhat. Stabilization of projection-based reduced-order models. *International Journal for Numerical Methods in Engineering*, 91(4):358–377, 2012.

[5] David Amsallem, Matthew J Zahr, and Charbel Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916, 2012.

[6] Athanasios C Antoulas, Danny C Sorensen, and Serkan Gugercin. A survey of model reduction methods for large-scale systems. *Contemporary mathematics*, 280:193–220, 2001.

[7] Patricia Astrid, Siep Weiland, Karen Willcox, and Ton Backx. Missing point estimation in models described by proper orthogonal decomposition. *Automatic Control, IEEE Transactions on*, 53(10):2237–2251, 2008.

[8] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T Patera. An empirical interpolationmethod: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[9] Peter W Bates, Sarah Brown, and Jianlong Han. Numerical analysis for a nonlocal allen-cahn equation. *Int. J. Numer. Anal. Model*, 6(1):33–49, 2009.

[10] Michal Beneš, Vladimır Chalupeckỳ, and Karol Mikula. Geometrical image segmentation by the allen–cahn equation. *Applied Numerical Mathematics*, 51(2):187–205, 2004.

[11] Robert Bos, Xavier Bombois, and Paul Van den Hof. Accelerating large-scale non-linear models for monitoring and control using spatial and temporal correlations. In *American Control Conference, 2004. Proceedings of the 2004*, volume 4, pages 3705–3710. IEEE, 2004.

[12] Tan Bui-Thanh, Murali Damodaran, and Karen E Willcox. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA journal*, 42(8):1505–1516, 2004.

[13] Xavier Cabr. Uniqueness and stability of saddle-shaped solutions to the allencahn equation. *Journal de Mathmatiques Pures et Appliques*, 98(3):239 – 256, 2012.

[14] Long Cai and Ralph E White. Reduction of model order based on proper orthogonal decomposition for lithium-ion battery simulations. *Journal of The Electrochemical Society*, 156(3):A154–A161, 2009.

[15] Kevin Carlberg, Charbel Bou-Mosleh, and Charbel Farhat. 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):155–181, 2011.

[16] Kevin Carlberg and Charbel Farhat. A compact proper orthogonal decomposition basis for optimization-oriented reduced-order models. *AIAA Paper*, 5964:10–12, 2008.

[17] Kevin Carlberg and Charbel Farhat. A low-cost, goal-oriented compact proper orthogonal decompositionbasis for model reduction of static systems. *International Journal for Numerical Methods in Engineering*, 86(3):381–402, 2011.

[18] Saifon Chaturantabut. *Dimension reduction for unsteady nonlinear partial differential equations via empirical interpolation methods*. ProQuest, 2009.

[19] Saifon Chaturantabut and Danny C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[20] Gong Chen, Brandon Cloutier, Ning Li, Benson K Muite, Paul Rigge, Sudarshan Balakrishnan, Andre Souza, and Jeremy West. Parallel spectral numerical methods. *Blue Waters Undergraduate Petascale Module*, 2012.

[21] Long-Qing Chen. Phase-field models for microstructure evolution. *Annual review of materials research*, 32(1):113–140, 2002.

[22] Jeong-Whan Choi, Hyun Geun Lee, Darae Jeong, and Junseok Kim. An unconditionally gradient stable numerical method for solving the allen–cahn equation. *Physica A: Statistical Mechanics and its Applications*, 388(9):1791–1803, 2009.

[23] Luca Daniel, Ong Chin Siong, Low Sok Chay, Kwok Hong Lee, and Jacob White. A multiparameter moment-matching model-reduction approach for generating geometrically parameterized interconnect performance models. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 23(5):678–693, 2004.

[24] Julia A Dobrosotskaya, Andrea L Bertozzi, et al. A wavelet-laplace variational technique for image deconvolution and inpainting. *IEEE Transactions on Image Processing*, 17(5):657–663, 2008.

[25] Richard Everson and Lawrence Sirovich. Karhunen–loeve procedure for gappy data. *JOSA A*, 12(8):1657–1664, 1995.

[26] Xiaobing Feng and Andreas Prohl. Numerical analysis of the allen-cahn equation and approximation for mean curvature flows. *Numerische Mathematik*, 94(1):33–65, 2003.

[27] Roland W. Freund. Krylov-subspace methods for reduced-order modeling in circuit simulation. *Journal of Computational and Applied Mathematics*, 123(12):395 – 421, 2000. Numerical Analysis 2000. Vol. III: Linear Algebra.

[28] Sonja Glavaški, Jerrold E Marsden, and Richard M Murray. Model reduction, centering, and the karhunen-loeve expansion. 1998.

[29] Mark S. Gockenbach. *Finite-dimensional linear algebra*, volume Discrete mathematics and its applications. CRC Press, 2010.

[30] Christopher P Grant. Spinodal decomposition for the cahn-hilliard equation. *Communications in Partial Differential Equations*, 18(3-4):453–490, 1993.

[31] Max D Gunzburger, Janet S Peterson, and John N Shadid. Reduced-order modeling of time-dependent pdes with multiple parameters in the boundary data. *Computer Methods in Applied Mechanics and Engineering*, 196(4):1030–1047, 2007.

[32] Michael Hinze and Martin Kunkel. Discrete empirical interpolation in pod model order reduction of drift-diffusion equations in electrical networks. In *Scientific Computing in Electrical Engineering SCEE 2010*, pages 423–431. Springer, 2012.

[33] Philip Holmes, John L Lumley, and Gal Berkooz. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press, 1998.

[34] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[35] Arieh Iserles. *A first course in the numerical analysis of differential equations*. Number 44. Cambridge University Press, 2009.

[36] Kazufumi Ito and SS Ravindran. A reduced-order method for simulation and control of fluid flows. *Journal of computational physics*, 143(2):403–425, 1998.

[37] Petar V Kokotovic, RE O'malley, and P Sannuti. Singular perturbations and order reduction in control theoryan overview. *Automatica*, 12(2):123–132, 1976.

[38] Hyun Geun Lee, Jeong-Whan Choi, and Junseok Kim. A practically unconditionally gradient stable scheme for the n-component cahn–hilliard system. *Physica A: Statistical Mechanics and its Applications*, 391(4):1009–1019, 2012.

[39] Kyunghoon Lee and Dimitri N Mavris. A unifying least squares perspective for gappy proper orthogonal decomposition and probabilistic principal component analysis. In *39th AIAA Fluid Dynamics Conference*, page 3899, 2009.

[40] Michel Loève. *Probability Theory; Foundations, Random Sequences*. New York: D. Van Nostrand Company, 1955.

[41] John L. Lumley. The structure of inhomogeneous turbulence in atmospheric turbulence and radio wave propagation, am yaglom adn vi tatarski, eds, 1967.

[42] Volker Mehrmann and Tatjana Stykel. Balanced truncation model reduction for large-scale systems in descriptor form. In *Dimension Reduction of Large-Scale Systems*, pages 83–115. Springer, 2005.

[43] Nathan E Murray and Lawrence S Ukeiley. An application of gappy pod. *Experiments in fluids*, 42(1):79–91, 2007.

[44] Ngoc-Cuong Nguyen, Anthony T Patera, and Jaume Peraire. A best points interpolation method for efficient approximation of parametrized functions. *International journal for numerical methods in engineering*, 73(4):521–543, 2008.

[45] Ahmed K Noor and Jeanne M Peters. Reduced basis technique for nonlinear analysis of structures. *Aiaa journal*, 18(4):455–462, 1980.

[46] Saheed Olalekan Ojo, Stefano Grivet-Talocia, and Marco Paggi. Model order reduction applied to heat conduction in photovoltaic modules. *Composite Structures*, 119:477–486, 2015.

[47] Pablo A Parrilo, Sanjay Lall, Fernando Paganini, George C Verghese, Bernard C Lesieutre, and Jerrold E Marsden. Model reduction for analysis of cascading failures in power systems. 1999.

[48] Benjamin Peherstorfer, Daniel Butnaru, Karen Willcox, and Hans-Joachim Bungartz. Localized discrete empirical interpolation method. *SIAM Journal on Scientific Computing*, 36(1):A168–A192, 2014.

[49] Janet S Peterson. The reduced basis method for incompressible viscous flow calculations. *SIAM Journal on Scientific and Statistical Computing*, 10(4):777–786, 1989.

[50] Stephen Prajna. Pod model reduction with stability guarantee. 2003.

[51] Muruhan Rathinam and Linda R. Petzold. A new look at proper orthogonal decomposition. *SIAM Journal on Numerical Analysis*, 41(5):1893–1925, 2003.

[52] Azriel Rosenfeld and Avinash C Kak. Digital picture processing, 1982.

[53] Clancy Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(03):997–1013, 2005.

[54] Stanislav Y. Shvartsman and Panayotis G. Kevrekidis. Low-dimensional approximation and control of periodic solutions in spatially extended systems. *Physical Review E*, 58(1):361, 1998.

[55] Stanislav Y. Shvartsman, Konstantinos Theodoropoulos, Roberto Rico-Martınez, Panayotis G. Kevrekidis, Edriss S. Titi, and T.J. (Lakis) Mountziaris. Order reduction for nonlinear dynamic models of distributed reacting systems. *Journal of Process Control*, 10(2):177–184, 2000.

[56] Răzvan Ştefănescu and Ionel Michael Navon. Pod/deim nonlinear model order reduction of an adi implicit shallow water equations model. *Journal of Computational Physics*, 237:95–114, 2013.

[57] Lloyd N Trefethen. *Spectral methods in MATLAB*, volume 10. Siam, 2000.

[58] Lloyd N. Trefethen and David Bau, III, editors. *Numerical Linear Algebra*. 1997.

[59] Andrew A Wheeler, William J. Boettinger, and Geoffrey B. McFadden. Phase-field model for isothermal phase transitions in binary alloys. *Phys. Rev. A*, 45:7424–7439, May 1992.

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

[61] Karen Willcox and Jaime Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA journal*, 40(11):2323–2330, 2002.

[62] Dunhui Xiao, Fangxin Fang, Andrew G Buchan, Christopher C Pain, Ionel Michael Navon, Juan Du, and G Hu. Non-linear model reduction for the navier–stokes equations using residual deim method. *Journal of Computational Physics*, 263:1–18, 2014.

[63] Xiaofeng Yang. Error analysis of stabilized semi-implicit method of allen-cahn equation. *Discrete Contin. Dyn. Syst. Ser. B*, 11(4):1057–1070, 2009.

[64] Jian Zhang and Qiang Du. Numerical studies of discrete approximations to the allen-cahn equation in the sharp interface limit. *SIAM Journal on Scientific Computing*, 31(4):3042–3063, 2009.

[65] Jingzhi Zhu, Long-Qing Chen, Jie Shen, and Veena Tikare. Coarsening kinetics from a variable-mobility cahn-hilliard equation: Application of a semi-implicit fourier spectral method. *Physical Review E*, 60(4):3564, 1999.

BIOGRAPHY

Name	Mr. Chutipong Dechanubeksa
Date of birth	26 February 1991
Educational Attainment	Thammasat University, 2009-2012
	Bachelor of Science (Mathematics)

Publication

1. Chutipong Dechanubeksa, Saifon Chaturantabut. Model order reduction applied to Allen-Cahn equation. *Proceedings of The 20th Annual Meeting in Mathematics 2015 (AMM2015)* pp.334–340, 2015.

2. Chutipong Dechanubeksa, Saifon Chaturantabut. Model reduction for Allen-Cahn equation. *Proceedings of Annual Pure and Applied Mathematics 2015 (APAM2015)* pp.39–47, 2015.