UNIVERSITY OF KIEL

MASTER THESIS

ALGORITHMIC OPTIMAL CONTROL
DEPARTMENT OF COMPUTER SCIENCE

Nonlinear Model Order Reduction of a marine ecosystem model using POD-DEIM

Author:
Neel Johann PETERS

Supervisors:
Prof. Dr. Thomas SLAWIG
Dr. Jaroslaw PIWONSKI

October 14, 2016

Declaration of Authorship

I, Neel Johann PETERS, declare that this thesis titled, "Nonlinear Model Order Reduction of a marine ecosystem model using POD-DEIM" and the work presented in it are my own. I confirm that:

- This work was done mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UNIVERSITY OF KIEL

Abstract

Algorithmic Optimal Control
Department of Computer Science

Master of Science

Nonlinear Model Order Reduction of a marine ecosystem model using POD-DEIM

by Neel Johann PETERS

The Discrete Empirical Interpolation Method (DEIM) is applied in conjunction with Proper Orthogonal Decomposition (POD) to construct nonlinear reduced-order models of marine ecosystem models. The POD approach is used to extract a low dimensional basis that optimally captures the dominant characteristics of the full-order model trajectory. This basis is used with a Galerkin projection to construct a reduced-order system. DEIM is applied to resolve the complexity issue in the nonlinear term of the POD reduced system. Numerical results demonstrate that the full-order model can be reduced with the POD-DEIM for a fixed parameter set. Creating reduced-order models with POD-DEIM approach for parameter studies have tuned out to be not practical with the marine ecosystem models.

Acknowledgements

I would like to thank the supervisors of my Master thesis, Jaroslaw Piwonski and Thomas Slawig. The meetings with them always were inspiring, motivating and, on a personal level, delightful.

Since the beginning of my studies, my parents and brothers and sisters were always there when I needed them and never ended their support. I am very thankful for their presence.

For the constructive advises to improve this thesis, I would like to thank Jenny and Lasse.

Also I would like to thank some people for the great time and distraction during the time working on this thesis, that help me to get new ideas and motivation. Doing acrobatic with the AkroKiel group, especially with Christian, Josi, Irene and Lena, was and is every time wonderful and jocular. In addition to that, I would like to thank Lasse for all the witty discussions about this thesis and everything else that popped up in our minds during cooking and eating.

Contents

Declaration of Authorship ii

Abstract iii

Acknowledgements iv

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Existing Methods . . . 2
- 1.3 Outline . . . 3

2 Nonlinear Model Reduction with POD-DEIM 4
- 2.1 Problem Formulation . . . 4
- 2.2 Proper Orthogonal Decomposition (POD) . . . 6
  - 2.2.1 Error Bound for POD . . . 7
- 2.3 Discrete Empirical Interpolation Method (DEIM) . . . 8
  - 2.3.1 Error Bound for DEIM . . . 10
- 2.4 Overview of POD-DEIM approach . . . 12

3 Application of the POD-DEIM approach to a marine ecosystem model 14
- 3.1 Introduction . . . 14
- 3.2 Reduce the System . . . 15
  - 3.2.1 Reduction of the nonlinear part . . . 16
- 3.3 Implementation . . . 17

4 Numerical Examples 18
- 4.1 Experiments with N-Model . . . 18
  - 4.1.1 Approximation error and CPU time . . . 21
  - 4.1.2 Behavior of the Reduced-Order Models . . . 23
- 4.2 Experiments with N-DOP-Model . . . 28

5 Experiments for Parameterized Reduced Order Models 33

6 Conclusion and Future Work **37**
6.1 Conclusion 37
6.2 Future Work 38

A Appendix **40**
A.1 Creating a ROM from a trajectory of one model year 40
A.2 Implementation Issue 43
A.3 Distribution of the CPU time of the main operations of the ROMs 44
A.4 Latin Hypercube samples 45
A.5 Plots of parametrized models 46

Bibliography **48**

List of Abbreviations

DEIM	**D**iscrete **E**mpirical **I**nterpolation **M**ethod
DOP	**D**issolved **O**rganic **P**hosphorus
EIM	**E**mpirical **I**nterpolation **M**ethod
FOM	**F**ull-**O**rder **M**odel
GCM	**G**eneral **C**irculation **M**odel
Metos3D	**M**arine **E**cosystem **T**oolkit for **O**ptimization and **S**imulation in **3-D**
MOR	**M**odel **O**rder **R**eduction
N	Phosphate $P0_4$
ODE	**O**rdinary **D**ifferential**E**quation
POD	**P**roper **O**rthogonal **D**ecomposition
ROM	**R**educed-**O**rder **M**odel
SVD	**S**ingular **V**alue **D**ecomposition
TMM	**T**ransport **M**atrix **M**ethod

List of Figures

2.1 An overview on the workflow of model reduction. 13

4.1 Distribution of singular values from a SVD of 12_3000, a matrix of snapshots distributed over the whole year. 18
4.2 Distribution of singular values from an SVD of 3_3000 a matrix of snapshots taken only in winter. 19
4.3 Relative error $\mathcal{E}$ during the course of a year for two different bases. . . . 20
4.4 Distribution of singular values from an SVD of different snapshot matrices. 21
4.5 The average relative error and the average CPU time of different ROMs of the N-Model. 22
4.6 The average relative error $\bar{\mathcal{E}}$ compared to the error bounds (2.8),(2.26) of the POD-DEIM approach. 23
4.7 Spin-up norm of different reduced-order models in comparison with the spin-up norm of the full-order model. 24
4.8 Relative error $\mathcal{E}$ of different reduced-order models. 25
4.9 Distribution of the singular values of 36_1000 from the N-DOP-Model. . 29
4.10 Spin-up norm of two ROMs in comparison with the spin-up norm of the FOM of N-DOP. 30
4.11 Relative error $\mathcal{E}$ of two ROMs of the N-DOP-Model. 31

5.1 Distribution of singular values of S and S_q. 34
5.2 The relative error between the ROM 36_3000P300D300 and the FOM for different sets of parameters. 35
5.3 The relative error between the ROM 36_3000P300D300 and the FOM solution of u_r. 36

A.1 Distribution of singular values from a SVD of 2880_1_{3000}, a matrix of snapshots of a full trajectory of the model year 3000. 40
A.2 Relative error $\mathcal{E}$ of the reduced-order model 2880_$1_{3000}P150D150$ initialized with an uniformly distributed concentration to the FOM. 41
A.3 Relative error $\mathcal{E}$ of the reduced-order model 2880_$1_{3000}P150D150$ initialized with the solution of model year 2000 of the FOM. 42

A.4 Spin-up norm of the reduced-order model $2880\_1_{3000}P150D150$ in comparison with the spin-up norm of the full-order model. 43
A.5 Absolute error of a model run with python and with PETSc 44
A.6 Distribution of the computational time among main operations of the N-Model during the computation of a model year. The BGCStep part indicates the time that is needed to evaluate the nonlinear function. . . . 44
A.7 Distribution of the computational time among main operations of the N-DOP-Model during the computation of a model year.The BGCStep part indicates the time that is needed to evaluate the nonlinear function. . . . 45
A.8 Relative error between different model runs and the FOM solution generated with u_{LHC_0}. 46
A.9 Relative error between different model runs and the FOM solution generated with $u_r + 50\%$. 47

List of Tables

2.1 Computational Complexity of POD-DEIM 13

4.1 Comparison of four ROMs from the N-Model 27
4.2 Comparison of two ROMs from the N-DOP-Model 32

A.1 10 Latin Hypercube samples used in Chapter 5. 45

Chapter 1

Introduction

1.1 Motivation

Numerical simulations of dynamical systems are extremely successful for the means of studying complex physical, biological or chemical phenomena. However the spatial and temporal dimensionality of those simulations often increase to the limits of computational requirements, storage and time. One possible approach to overcome this is model reduction.

Model reduction has become imported for many fields of physics as they offer the potential to simulate dynamical systems with a substantially increased computation efficiency. The technique is a mathematical approach to find a low-dimensional approximation for a system of ordinary differential equations. The goal is to determine a low-dimensional system that has the same solution characteristics as the original system, but needs less storage and evaluation time.

This approach is used in particular for models, that model the physics of the complex earths climate and have large spatial and temporal scales. Thus if they provide a certain accuracy, they need a lot of computation time. For these systems it is a balancing act between computation time and accuracy in time and space, and usually the computational resources are the limiting factor.

In addition these models are often controlled by a set of parameters and a common problem is to find a certain set of parameters. Which results in an, in some way, optimal solution, for example in a solution that is close to real measurements. This search for a set of parameters requires, in most cases, many evaluations of the model, so the computational cost is potentially higher. The idea is to use a reduced model for parts of this parameter search, to lower the computational time.

In this thesis, model reduction of a marine ecosystem model will be investigated. Marine ecosystem models play an imported role in research of the global carbon cycle as well as for questions on climate change. They have huge spatial scale if they provide a certain accuracy. The resulting partial differential equations are in there discrete form high in dimension. Also the temporal domain is large, because effects in the ocean take several thousand years to fully spread and thus these models need this time to approach a steady-state. This results from the circulation circle of the ocean. It is estimated that it can take up to 1000 years for a parcel of water to complete one circle. In addition to the ocean circulation a marine biogeochemistry model describes the marine chemistry, biology and the geological processes that occur in the ocean. These processes are controlled by a set of parameter of which some are not directly measurable. Thus, parameter optimization is needed to fit the model output to real observations. Therefore, there is a strong demand for model reduction for this kind of model.

1.2 Existing Methods

Model Reduction or Model Order Reduction (MOR) was developed in the context of linear control theory. It is today a wide field research in system and control theory and numerical mathematics such as signal analysis and fluid dynamics. Most work has been done for linear MOR and the generally used methods are discussed in [1]. Among them the so-called Proper Orthogonal Decomposition (POD) also known as Karhunen-Loeve decomposition or Principal Component Analysis. POD has been successfully applied to a wide field of research including pattern recognition [2, 3] , fluid dynamics [4, 5], turbulent flows [6, 7] and other numerical simulations [8, 9]. The POD method combined with Galerkin projection is a popular approach for constructing reduced-order models of PDEs. This approach has provided reduced-order models in many applications such as fluid dynamics, aerodynamics and optimal control. Also it has been used for large scale systems with a high dimensional parametric input space. According to [10], where also a historical overview of the POD approach can be found, the first usage of POD for MOR of a dynamical system was in the 1990s, cf. [11].

The POD-Galerkin method only reduces linear parts and therefore another method is needed to reduce the complexity of the nonlinear terms. Discrete Empirical Interpolation Method (DEIM) proposed by [12, 13] expands the POD approach to construct an interpolation of the nonlinear part. It is a discrete simplification of the Empirical Interpolation Method (EIM) introduced by [14]. The POD-DEIM approach has been applied to dynamical processes [15, 16, 17, 18] and has shown that it leads to a computational gain in complexity.

The marine ecosystem model, which will be used here, has been introduced by [19]. The simulation package Metos3D, that implements the model, is described briefly in

Chapter 3 and a full description is presented in [20].

1.3 Outline

The work in this thesis is structured as follows: In Chapter 2 the problem is formulated and the POD approach and its extension the DEIM is explained. The next chapter introduces into the model behind Metos3D and shows the theoretically application of the POD-DEIM approach on this model. Chapter 4 presents the results of the numerical experiments with some of the models Metos3D provides. It starts with an extensive analysis of the simplest model, called N-Model. Afterwards, a more complex model with two tracers is presented, the N-DOP-model. The results of the experiments for parameters studies with reduced-order models of the N-Model are content of Chapter 5. The last chapter presents a conclusion and an outlook on future research questions that had come up during this thesis, but were not considered.

Chapter 2

Nonlinear Model Reduction with POD-DEIM

This chapter presents a model reduction method for nonlinear ordinary differential equation (ODEs). First a general formulation of the problem is given and its reduction via Proper Orthogonal Decomposition (POD) with Galerkin projection is shown. Continued with a discussion of the complexity issue in the nonlinear part. To resolve this issue the Discrete Empirical Interpolation Method (DEIM) is introduced in 2.3. This method extends the orthogonal projection of POD in the nonlinear part with an interpolation of the nonlinear function. The DEIM was fist introduced by Saifon Chaturantabut in [12], where also the algorithm and the error bound analysis presented in 2.3 comes from.

2.1 Problem Formulation

Consider a system of nonlinear differential equations of the form

$$\frac{d}{dt}y(t) = Ay(t) + q(y(t)) \tag{2.1}$$

where $t \in [0, T]$ denotes the time, $y(t) = [y_1(t), \ldots, y_n(t)]^T \in \mathbb{R}^n$ is a vector of states with initial condition $y(0) = y_0 \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$ is the discrete approximation of the linear differential operator and $q : \mathbb{R}^n \longmapsto \mathbb{R}^n$ a nonlinear vector-based function. The complexity, both time and space, to solve the system is mostly depended on the dimension n of the system. Which could be very large, if high accuracy is required, thus, to reduce the complexity, we want to reduce the dimension.

In order to do this the system will be projected on a subspace spanned in $\mathbb{R}^n$ by a reduced basis of a dimension $k \ll n$. These projection-based techniques are commonly used for constructing a reduced-order system, that approximates the original system in a subspace. Here a Galerkin projection is used as the means for dimension reduction.

For that let $V \in \mathbb{R}^{n\times k}$ be a matrix whose columns are the orthonormal vectors of the reduced basis. Then projecting the system (2.1) onto V by replacing $y(t)$ with $V\hat{y}(t)$, where $\hat{y}(t) \in \mathbb{R}^k$, is the reduced system of the form

$$\frac{d}{dt}\hat{y}(t) = V^T AV\hat{y}(t) + V^T q(V\hat{y}(t)). \tag{2.2}$$

The quality of the approximation is clearly affected by the choice of the reduced basis vectors in V. The POD method constructs a set of basis vectors from a singular value decomposition (SVD) of a set of snapshots

$$S = \{y_1, \ldots, y_m\}, \tag{2.3}$$

which contains samples of trajectories $y(\cdot)$ (called snapshots) for a particular set of parameters, boundary condition and other system inputs. After a reduced model has been constructed from this basis, it may be used to compute approximate solutions for different initial conditions and parameter settings. If the snapshots are diverse enough it is expected that the approximate solution is near to the high dimensional one. The POD basis is optimal in the sense that a approximation error in relation to the snapshots is minimized. Therefore, the POD approach is used here for constructing the basis.

In the equation (2.2) the linear part is already reduced in dimension, because the pre-computation of $\hat{A} = V^T AV \in \mathbb{R}^{k\times k}$ supplies a system of dimension $k \ll n$. But the evaluation of the nonlinear function $q(y(t))$ is still in the dimension of n, i. e.

$$\frac{d}{dt}\hat{y}(t) = \underbrace{\hat{A}}_{k\times k}\hat{y}(t) + V^T q(\underbrace{V\hat{y}(t)}_{n}). \tag{2.4}$$

To reduce the nonlinear part as well, the DEIM will be used. It provides an approximation $\hat{q}(y(t))$ of the nonlinear function by approximating the function $q(y(t))$ in an low dimensional subspace spanned by another basis $U \in \mathbb{R}^{n\times m}$, which is constructed out of snapshots of the nonlinear function with the POD method. With coefficients $c(y(t))$ the approximation is of the form

$$q(y(t)) \approx \hat{q}(y(t)) = Uc(y(t)) \tag{2.5}$$

and the form of the reduced system is

$$\frac{d}{dt}\hat{y}(t) = \underbrace{\hat{A}}_{k\times k}\hat{y}(t) + \underbrace{V^T U}_{k\times m} c(V\hat{y}(t)). \tag{2.6}$$

The system of (2.1) is now completely reduced in dimension. The linear part is reduced to $k \ll n$ and the nonlinear part to $m \ll n$.

How the coefficients $c(y(t))$ will be determined, will be explained in Chapter 2.3, along with an explanation of the DEIM-algorithm and an short error analysis. How the two basis are constructed using the POD method will be explained in then next chapter.

2.2 Proper Orthogonal Decomposition (POD)

Consider a set of snapshots $S = \{y_1, \dots, y_{n_s}\} \subset \mathbb{R}^n$ and the corresponding matrix $Y = [y_1, \dots, y_{n_s}] \in \mathbb{R}^{n \times n_s}$. The POD method constructs an orthonormal basis in the space spanned by S. Let $r = rank\{Y\}$, then there exist $\{v_i\}_i^k \subset \mathbb{R}^n$ orthonormal basis vectors, for a $k \leq r$.

Definition 1 ***(SVD):*** *Let $\mathbb{A} \in \mathbb{R}^{m \times n}$ a matrix, then there exist two orthogonal matrices*

$$U = [\varphi_1 | \dots | \varphi_n] \in \mathbb{R}^{m \times m}, \qquad Z = [\Psi_1 | \dots | \Psi_m] \in \mathbb{R}^{n \times n}$$

such that

$$\mathbb{A} = U \Sigma Z^T, \; mit \; \Sigma = diag(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{m \times n} \tag{2.7}$$

and $\sigma_1 \geq \dots \geq \sigma_r > 0$, for r = min(m,n), $\mathbb{A}\Psi_i = \sigma_i \varphi_i$ and $\mathbb{A}^T \varphi_i = \sigma_i \Psi_i$ with $i = 1, \dots, r$.

The SVD of $Y = U \Sigma Z^T$ with orthogonal matrices $U = [\varphi_1 | \dots | \varphi_n] \in \mathbb{R}^{n \times n}$ and $Z = [\Psi_1 | \dots | \Psi_m] \in \mathbb{R}^{m \times m}$ and the diagonal matrix $\Sigma = diag(\sigma_1, \dots, \sigma_r)$ with singular values $\sigma_1 \geq \dots \geq \sigma_r$ provides these basis vectors. Note that the diagonal matrix Σ is not square, thus, $diag(\sigma_1, \dots, \sigma_r)$ provides a rectangular matrix with entries $a_{ij} = \sigma_i$ for $i = j$ and $a_{ij} = 0$ for $i \neq j$.

For a $k \leq m$ the POD-basis is defined as the set of the first k left singular vectors $\varphi_1, \dots, \varphi_k$ of U. Let $V = [\varphi_1, \dots, \varphi_k] \in \mathbb{R}^{n \times k}$ be the matrix of these POD-basis vectors. An approximation of a snapshot y_j in the span of Y is therefore Vc with coefficients $c \in \mathbb{R}^k$. The Galerkin orthogonality of the residual $y_j - Vc$ to the span of V gives $V^T(y_j - Vc) = 0$, hence $y_j \approx VV^T y_j$. The POD-basis provides a optimal orthogonal basis with regard to the sum of the quadratic error of the approximation of the snapshots.

2.2.1 Error Bound for POD

Theorm 1 ***(POD-error bound, [21, p. 125]):***

Let $\mathbb{V}_k = \{W \in \mathbb{R}^{n\times k} : W^T W = I_k\}$ be the set of all k-dimensional orthonormal bases and $V \in \mathbb{R}^{n\times k}$ the matrix of basis vectors provided by the POD-method related to the snapshots $Y = \{y_1, \dots, y_m\}$, then the following holds

$$\sum_{j=1}^{m} \| y_j - VV^T y_j \|_2^2 = \min_{W\in\mathbb{V}_k} \sum_{j=1}^{m} \| y_j - WW^T y_j \|_2^2 = \sum_{i=k+1}^{r} \sigma_i^2. \tag{2.8}$$

Theorm 2 ***(Schmidt-Eckart-Young, [21, p. 118]):*** *Let $A \in \mathbb{R}^{m\times n}$ be a matrix of rank p, then there exist two orthogonal matrices $U = [\varphi_1|\dots|\varphi_n] \in \mathbb{R}^{n\times n}$ and $Z = [\Psi_1|\dots|\Psi_m] \in \mathbb{R}^{m\times m}$, so that $A = U\Sigma Z^T$ with the diagonal-matrix $\Sigma = diag(\sigma_1, \dots, \sigma_r)$ and the singular values $\sigma_1 \geq \dots \geq \sigma_r \geq 0$ for r=min(m,n). The matrix*

$$A_k = \sum_{i=1}^{k} \sigma_i \varphi_i \Psi_i^T, \qquad 0 \leq k \leq p \tag{2.9}$$

satisfies the property

$$\|A - A_k\|_F = \min_{\substack{B\in\mathbb{R}^{m\times n}\\ rank(B)\leq k}} \|A - B\|_F = \sqrt{\sum_{i=k+1}^{p} \sigma_i^2}. \tag{2.10}$$

Proof of Theorem 1, [21, p. 125]: By Theorem 2, the best rank k approximation of Y is given by

$$Y_k = \sum_{i=1}^{k} \sigma_i \varphi_i \Psi_i^T.$$

With $\Psi_i = \frac{1}{\sigma_i} Y^T \varphi_i$, which results from the definition of the SVD, follows

$$Y_k = \sum_{i=1}^{k} \sigma_i \varphi_i (\frac{1}{\sigma_i} Y^T \varphi_i)^T = \sum_{i=1}^{k} \varphi_i \varphi_i^T Y = VV^T Y.$$

Moreover, for a $W \in \mathbb{V}_k$

$$\sum_{j=1}^{m} \| y_j - WW^T y_j \|_2^2 = \| Y - WW^T Y \|_F^2,$$

because of the definition of the Frobenius norm. And with (2.10) it is

$$\|Y - Y_k\|_F^2 = \min_{\substack{B\in\mathbb{R}^{m\times n}\\ rank(B)\leq k}} \|Y - B\|_F^2 \leq \min_{W\in\mathbb{V}_k} \| Y - WW^TY \|_F^2 .$$

As $Y_k = VV^TY$ follows

$$\| Y - VV^TY \|_F^2 = \min_{W\in\mathbb{V}_k} \| Y - WW^TY \|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2.$$

For the quality of the approximation of the POD reduced model the choice of the dimension k of th POD basis is importent. There exist no theoretical bound for the approximation error depending on k. According to [21] in practice the ratio of the first k singular values to all singular values, i.e

$$I(k) = \frac{\sum_{i=1}^{k}\sigma_i^2}{\sum_{i=1}^{r}\sigma_i^2} \tag{2.11}$$

is used as an error condition for selcting a dimension k heuristically. For a given approximation error ϵ_{POD}

$$I(k) = \frac{\sum_{i=1}^{k}\sigma_i^2}{\sum_{i=1}^{r}\sigma_i^2} \leq 1 - \epsilon_{POD}^2 \tag{2.12}$$

can be used to determine k.

2.3 Discrete Empirical Interpolation Method (DEIM)

In order to reduce the nonlinear part in (2.4), a low-dimensional approximation of a nonlinear function $q(t) : \mathbb{D} \longmapsto \mathbb{R}^n$ is needed, i.e.

$$q(t) \approx Uc(t), \tag{2.13}$$

where $c(t) \in \mathbb{R}^m$ is the corresponding coefficient vector and $U \in \mathbb{R}^{n\times m}$ is a matrix obtain from the application of the POD approach only on the nonlinear terms. Indeed, it is received from an SVD of a snapshots matrix of the nonlinearity. To determine the vector $c(t)$ m distinguished rows are selected from the over-determined system $q(t) = Uc(t)$. The selection of the rows can be done with a matrix

$$P = [e_{p1}, \ldots, e_{pm}] \in \mathbb{R}^{n\times m}, \tag{2.14}$$

where $e_{pi} = [0, \ldots 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^n$ is the pi-th unit vector, for $i = 1, \ldots, m$. By multiplying P^T on $q(t) = Uc(t)$ m rows are selected. If P^TU is non-singular, the vector $c(t)$ can be determined from

$$c(t) = (P^TU)^{-1}P^Tq(t) \tag{2.15}$$

and the approximation (2.13) becomes

$$q(t) \approx Uc(t) = U(P^TU)^{-1}P^Tq(t) \tag{2.16}$$

The DEIM thus requires to construct a projection basis U and interpolation indices $\{p_1, \ldots, p_m\}$ used for P. The projection basis U is constructed by using the POD on a nonlinear set of snapshots $Y_q = [q(t_1), \ldots, q(t_m)]$ obtained from the high order system. These snapshots are the evaluations of the nonlinear function $q(t)$ on $t_1, \ldots, t_{n_s}$. They can be obtained together with the snapshots for the full-system POD, because they are already computed there. Thus only the SVD has to be computed to obtain U.

The interpolation indices are iteratively selected by the greedy algorithm from the basis U (cf. Algorithm 1). The algorithm selects iteratively m indices $\{p_1, \ldots, p_m\}$, where $p_i \in \{1, \ldots, n\}$ for $i = 1, \ldots, m$. This minimizes the interpolation error over the snapshot set measured in the maximum norm. It starts by selecting the index of the biggest entry of the first column of U and thus the first basis vector, which corresponds to the most dominant singular value of the SVD. Further indices are selected in the way that they correlate to the index of the larges entry of the residual $r = u_i - Uc$. The residual r is the error between the basis vector u_i and its projected approximation Uc. Since the columns of U are linear independent, r is a nonzero vector in each step. Thus, an entry with the larges magnitude can always be selected.

Algorithm 1 DEIM , [12]

```
1: **Input:**$\{u_i\}_{i=1}^m \subset \mathbb{R}^n$
2: **Output:**$\vec{\varrho} = \{\varrho_1, \ldots, \varrho_m\}^T \in \mathbb{N}^m$
3: $\varrho_1 = argmax\{|u_1|\}$
4: $U = [u_1], P = [e_{\varrho 1}], \vec{\varrho} = [\varrho_1]$
5: **for** $i = 2$ to $m$ **do**
6:     **Solve** $(P^TU)c = P^Tu_i$
7:     $r = u_i - Uc$
8:     $\varrho_i = argmax\{|r|\}$
9:     $U \leftarrow [U|u_i], P \leftarrow [P|e_{\varrho i}], \vec{\varrho} \leftarrow \begin{bmatrix} \varrho \\ \varrho_i \end{bmatrix}$
10: **end for**
```

In line 6 of Algorithm 1 the linear system $(P^TU)c = P^Tu_i$ must be solved. This is only possible if P^TU is always non-singular, as is shown in [12]. The whole procedure has a complexity of $O(m^4 + mn)$ (cf. [12, Chapter 2.2.6]), where the dimension m of the basis U is small.

Formally the DEIM approximation is defined in [12] as follows.

Definition 2 *Let $f : D \longmapsto \mathbb{R}^n$ be a nonlinear vector valued function with $D \subset \mathbb{R}^d$ for a $d \in \mathbb{N}$. Let $\{u_l\}_{l=1}^m \subset \mathbb{R}^n$ be a set of linear independent vectors. For an $x \in D$ the DEIM approximation of order m in the space spanned by $\{u_l\}_{l=1}^m$ is given by:*

$$\hat{f}(x) := U(P^TU)^{-1}P^Tf(x) \tag{2.17}$$

where $U = [u_1|\dots|u_m] \in \mathbb{R}^{n\times m}$ is the basis, provided by the POD method. Moreover $P = [e_{p_1}|\dots|e_{p_m}] \in \mathbb{R}^{n\times m}$, where $\{p_1, \dots, p_m\}$ is the output of the DEIM algorithm with $\{u_l\}_{l=1}^m$ as input vectors.

It can be shown that $\hat{f}(x)$ is a interpolation of the original $f(x)$. It holds

$$P^T\hat{f}(x) = P^T(U(P^TU)^{-1}P^Tf(x)) = (P^TU)(P^TU)^{-1}P^Tf(x) = P^Tf(x).$$

The functions $\hat{f}(x)$ and $f(x)$ are exactly the same at the DEIM points $\{p_1, \dots, p_m\}$.

2.3.1 Error Bound for DEIM

Theorm 3 ***(DEIM-error),[12]:*** *Let $f \in \mathbb{R}^n$ be a vector, let $\{u_l\}_{l=1}^m \subset \mathbb{R}^n$ be a set of linear independent orthogonal vectors. The DEIM approximation of order $m \le n$ of f in the space spanned by $\{u_l\}_{l=1}^m$ is $\hat{f} = \mathbb{P}f$, where $\mathbb{P} = U(P^TU)^{-1}P^T$, $U = [u_1|\dots|u_m] \in \mathbb{R}^{n\times m}$ and $P = [e_{p_1}|\dots|e_{p_m}] \in \mathbb{R}^{n\times m}$, with $\{p_1, \dots, p_m\}$ being the output of the DEIM-algorithm with input $\{u_l\}_{l=1}^m$. Then the following holds*

$$\| f - \hat{f} \|_2 \le C_m \varepsilon(f), \tag{2.18}$$

where $C_m = \| (P^TU)^{-1} \|_2$ and $\varepsilon(f) = \| f - UU^Tf \|_2$.

Proof of Theorem 3, [12]: Let $\hat{f}(x)$ be the DEIM approximation. To find an error bound in the 2-norm for $\| f - \hat{f} \|_2$ look at the best approximation of f in the space spanned by U, in particular $f_* = UU^Tf$. Then it holds

$$f = (f - f_*) + f_* = w + f_* \qquad \text{with} \qquad w = f - f_* = f - UU^Tf \tag{2.19}$$

and also

$$\hat{f} = \mathbb{P}f = \mathbb{P}(w + f_*) = \mathbb{P}w + f_*, \tag{2.20}$$

because $\mathbb{P}f_* = f_*$. Therefore

$$\| f - \hat{f} \|_2 = \| w + f_* - (\mathbb{P}w + f_*) \|_2 \leq \| I - \mathbb{P} \|_2 \| w \|_2 \tag{2.21}$$

with $\| I - \mathbb{P} \|_2 = \| \mathbb{P} \|_2$ as shown in [22] for any projector-matrix $P \neq 0$. From 2.21 follows

$$\| f - \hat{f} \|_2 \leq \| I - \mathbb{P} \|_2 \| w \|_2 = \| \mathbb{P} \|_2 \| w \|_2 \tag{2.22}$$

and with $\| \mathbb{P} \|_2 = \| U(P^TU)^{-1}P^T \|_2 = \| (P^TU)^{-1} \|_2$. Because U and $\mathbb{P}$ are both orthogonal it holds

$$\| f - \hat{f} \|_2 \leq \| (P^TU)^{-1} \|_2 \| w \|_2 \tag{2.23}$$

Overall it holds

$$\| f - \hat{f} \|_2 \leq \| \mathbb{P} \|_2 \| w \|_2 = \| (P^TU)^{-1} \|_2 \| f - UU^Tf \|_2 = C_m \varepsilon(f). \tag{2.24}$$

The factor $(P^TU)^{-1}$ depends on the matrix P, which is constructed out of the DEIM indices $\{p_1, \ldots, p_m\}$. That is the reason the DEIM algorithm aims to select an index that limits the growth of $\| (P^TU)^{-1} \|_2$ and thus the error of $\| f - \hat{f} \|_2$. Chaturantabut shows in [12] a recursive formula for $(P^TU)^{-1}$. This proofs the minimization of this error and also the non-singularity of $(P^TU)^{-1}$, which is needed in the Algorithm 1. Moreover, it can be shown that C_m can be bound with

$$C_m \leq \frac{(1 + \sqrt{2n})^{m-1}}{\| u_1 \|_\infty}. \tag{2.25}$$

This bound is very pessimistic and it will grow more rapidly than the actual matrix $\| (P^TU)^{-1} \|_2$. In [12] it is advised to use the matrix norm as a posteriori error estimation, than the priori estimation C_m, because the matrix $(P^TU)^{-1}$ is usually small.

The term $\varepsilon(f) = \| f - UU^Tf \|_2$ depends on f. It changes for ever new f, hence it is expensive to compute. It is desirable to have an easily computable estimation. In [12], the approximation $\varepsilon(f) \approx \sigma_{m+1}$ is used, thus the singular value corresponding to the first left singular vector that has not been taken into the DEIM base matrix U. This estimate is purely heuristic and is not proved in general, but for the numerical examples in [12] it does seem to provide a reasonable qualitative estimate of the expected error. The DEIM error should be bound like this, if the trajectories are attracted to a low-dimensional subspace, this means they lie nearly in the space spanned by the snapshots. In this work the error bound approximation

$$\bar{\mathcal{E}}_* = \| f - \hat{f} \|_2 \leq C_m \varepsilon(f) \lessapprox \| (P^T U)^{-1} \|_2 \, \sigma_{m+1} \tag{2.26}$$

will be used for the numerical experiments.

2.4 Overview of POD-DEIM approach

To recall what the POD-DEIM approach requires to reduce a system like (2.1) and how the workflow will be, look at Figure 2.1. To see an overview over the complexity of the different parts look at Table 2.1. The generation of linear and nonlinear snapshots can usually be done in one model evaluation. The computation of the SVD for a large matrix is quite costly. Thus if the number of snapshots n_s grows, it is more efficient to compute a thin SVD and the singular vectors and the singular values iteratively and parallel with for example with ARPACK [23] or SLEPc [24]. The computational work in Table 2.1 has only to be done once, to get the reduced system. The matrices $\hat{A}$ and $\mathbb{P}$ are precomputed and stored and reused when solving the reduced system.

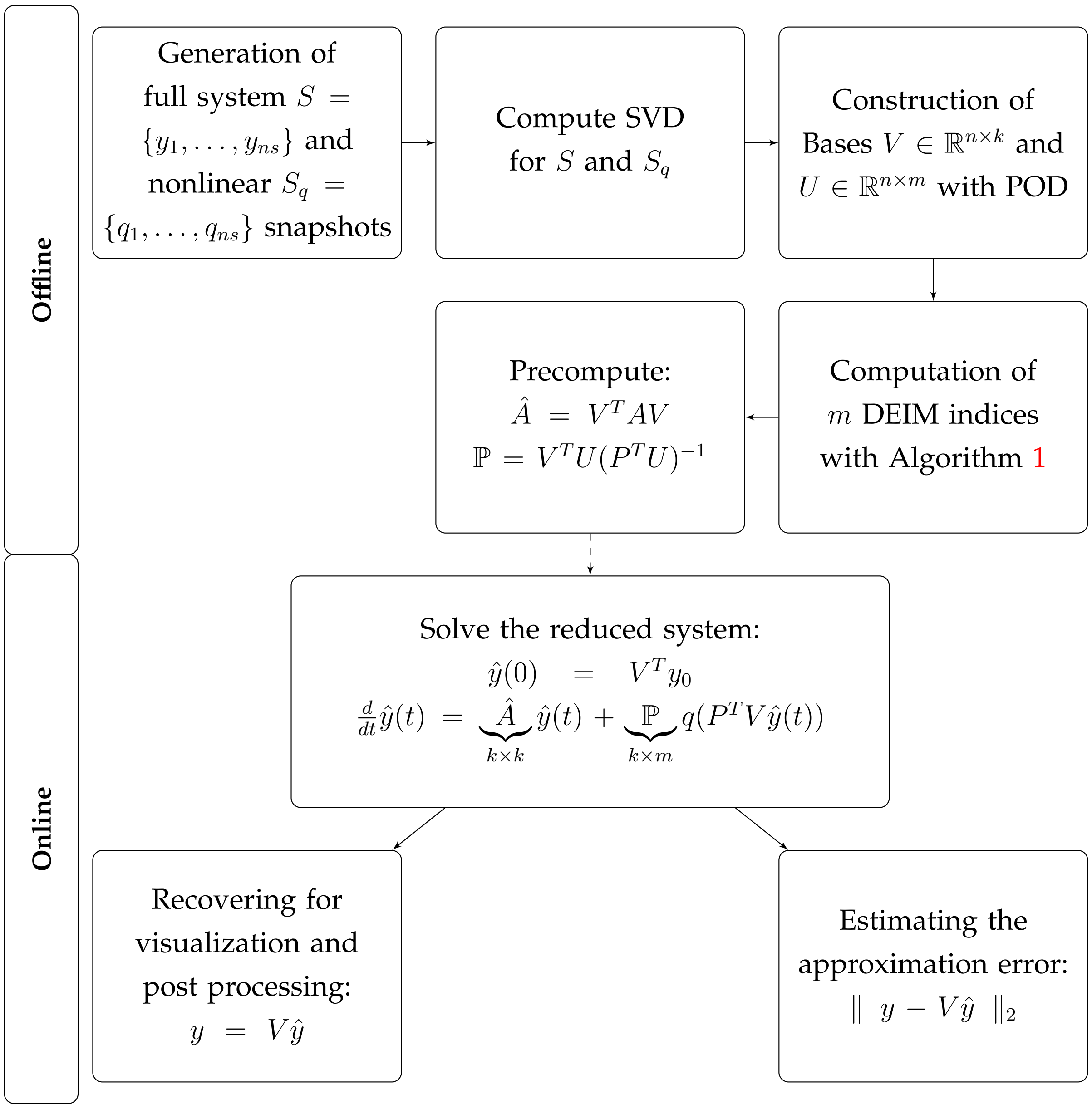

Offline
Online
Generation of full system $S = \{y_1, \dots, y_{ns}\}$ and nonlinear $S_q = \{q_1, \dots, q_{ns}\}$ snapshots
Compute SVD for S and S_q
Construction of Bases $V \in \mathbb{R}^{n \times k}$ and $U \in \mathbb{R}^{n \times m}$ with POD
Computation of m DEIM indices with Algorithm 1
Precompute: $\hat{A} = V^T A V$ $\mathbb{P} = V^T U (P^T U)^{-1}$
Solve the reduced system: $\hat{y}(0) = V^T y_0$ $\frac{d}{dt}\hat{y}(t) = \underbrace{\hat{A}}_{k \times k} \hat{y}(t) + \underbrace{\mathbb{P}}_{k \times m} q(P^T V \hat{y}(t))$
Recovering for visualization and post processing: $y = V\hat{y}$
Estimating the approximation error: $\| y - V\hat{y} \|_2$

FIGURE 2.1: An overview on the workflow of model reduction.

compute Snapshots	Problem dependent
SVD for POD-basis	$\mathcal{O}(nn_s^2)$
DEIM algorithm for m interpolation indices	$\mathcal{O}(m^4 + nm)$
Precompute: $\hat{A} = V^T A V$	$\mathcal{O}(NNZk + nk^2)$ for a sparse A
Precompute: $\mathbb{P} = V^T U (P^T U)^{-1}$	$\mathcal{O}(nkm + m^2 n + km^2 + m^3)$

TABLE 2.1: Computational Complexity of POD-DEIM

Chapter 3

Application of the POD-DEIM approach to a marine ecosystem model

3.1 Introduction

To apply the POD-DEIM approach on a marine ecosystem model, the Simulation Package Metos3D (a Marine Ecosystem Toolkit for Optimization and Simulation in 3-D) [20] is used. This package provides the possibility to simulate the ocean circulation coupled with biogeochemical processes. For the simulation of the ocean circulation the package uses transport matrices as described in [25]. The matrices are extracted from a general circulation model (GCM) and are capturing the complex three-dimensional advective diffusive transport of tracers in a GCM as a sparse matrix. The so called transport matrix method (TMM), demonstrated in [26], makes use of these pre-computed transport data to apply it on arbitrary variables. Monthly averaged diffusion and advection matrices, which are interpolated during computation, are used as a compromise between storage and accuracy. The effects, that are in longitudinal and latitudinal directions, are treated explicitly while the vertical effects are computed implicitly. Thus, there are twelve explicit and twelve implicit matrices.

In addition to this linear part, the package provides an interface for different biogeochemical models that vary in the number of tracers and types of modeled processes. They are controlled by a set of parameters u and describe processes like nutrient consumption as well as growing, dying and sinking of different tracer elements. For example, the N-Model, that is used in Chapter 4, calculates the uptake of inorganic nutrients by phytoplankton of an implicit pyhtoplankton concentration in the euphotic zone (cf. [27]). This uptake is limited by the amount available inorganic nutrients and by the light. The model uses phosphate (PO_4) as tracer element which governs the rate of growth of pyhtoplankton.

The surface grid has a resolution of $2.8125°$ in both directions, longitudinal and latitudinal. This results into a 128×64 grid with 4448 horizontal points in the oceans, if the grid points, where only land is, are left out. Each grid point represents a water column, called profile. Each profile has at most 15 vertical depth layers. Overall, there are $n_h = 52749$ grid points, which are held in one single vector for each tracer. The implicit and explicit matrices are extremely sparse as they have only 672k (0,024%) and 5407k (0,19%) non zero values. To solve a system with more than one tracer, the matrices and tracers are arranged in a block diagonal system. Thus, with each tracer the linear system grows.

The temporal resolution is $\Delta t = 1/2880$. Assuming a year with 360 days, Δt represent a step of three hours. The boundary data is used to have the ice cover and latitude information for each profile and a domain data set provides the model with the information of the depths and heights of the vertical layers of each profile.

The packed provides two solver for periodic steady-states one is a fixed point iteration (spin-up) and the other a Newton solver. In this thesis only the spin-up solver is used.

3.2 Reduce the System

As said, the full discrete representation of the iteration for n tracers is a block diagonal system. An iteration reads

$$y_{j+1} = A_{impj}(A_{expj}y_j + \Delta t q_j(y_j, u, b_j, d_j)), \tag{3.1}$$

where $y_j = (y_i(t_j))_{i=1}^{n}$ is a vector combined of all n tracer vectors and $t_j = t_0 + (j-1) * \Delta t$ is a point in time. A_{impj} and A_{expj} are the block diagonal matrices, interpolated for the time index j. The discrete boundary and domain data is represented by $b_j = (b_i(t_j))_{i=1}^{n_b}$ and $d_j = (d_i(t_j))_{i=1}^{n_d}$. The biogeochemical model q_j is described by $q_j(y_j, u, b_j, d_j) = (q_i(t_j, y_i, u, b_i, d_i))_{i=1}^{n}$.

To apply the POD-DEIM approach, let $n = 1$, so that there is only one tracer and let n_h be the length of the tracer vector. The basis matrices $V \in \mathbb{R}^{n_h \times k}$ and $U \in \mathbb{R}^{n_h \times m}$ are computed with the POD method. The projector $\mathbb{P} = U(P^T U)^{-1} P^T$ is computed with the result $P \in \mathbb{R}^{n_h \times m}$ of the DEIM-Algorithm with input U.

After projecting the system (3.1) onto V and applying the DEIM approximation onto the nonlinear part, it reads

$$\hat{y}_{j+1} = V^T A_{impj}(A_{expj} V \hat{y}_j + \Delta t \mathbb{P} q_j(V\hat{y}_j, u, b_j, d_j)), \tag{3.2}$$

where $\hat{y}_j = V^T y_j$.

To use the advantage of the reduction, $V^T A_{impj}$ is multiplied out and the system looks like this:

$$\begin{aligned} \hat{y}_{j+1} &= \underbrace{V^T A_{impj} A_{expj} V}_{A_{rj} \in \mathbb{R}^{k \times k}} \hat{y}_j + \Delta t \underbrace{V^T A_{impj} U (P^T U)^{-1}}_{\mathbb{P}_{rj} \in \mathbb{R}^{k \times m}} P^T q_j(V \hat{y}_j, u, b_j, d_j) \\ &= A_{rj} \hat{y}_j + \Delta t \mathbb{P}_{rj} P^T q_j(V \hat{y}_j, u, b_j, d_j). \end{aligned} \tag{3.3}$$

The matrices A_{rj} and $\mathbb{P}_{rj}$ are interpolated out of the twelve monthly average matrices A_{ri} and $\mathbb{P}_{ri}$ for $i = 1, ..., 12$. A_{ri} and $\mathbb{P}_{ri}$ are precomputed out of monthly averaged matrices A_{impi} and A_{expi} of the original system, the POD-basis V as well as U and the DEIM index matrix P. Thus, $A_{ri} = V^T A_{impi} A_{expi} V$ and $\mathbb{P}_{ri} = V^T A_{impi} U (P^T U)^{-1}$. They are dense, but quite small, because $k << n_h$ and $m << n_h$.

3.2.1 Reduction of the nonlinear part

The reduction of the nonlinear part is more complex. The best reduction would be possible if the nonlinear function could be evaluated componentwise. Then, only the selected points have to be computed and the reduction would be from n_h to m evaluations. But in Metos3D the biogeochemical model is evaluated independently for each profile.

Thus, to compute one point a whole profile has to be evaluated. The profiles that have to be evaluated can be calculated with the help of the DEIM-indices and the land-sea mask which contains the geometry information. From all the $p = 4448$ profiles of the full-order model (FOM), only m are computed. Thus, the dimension reduction is less than the reduction of the linear part but one evaluation of the biogeochemical model can be very expensive.

The following pseudo code describes the reduced evaluation of the nonlinear function.

Algorithm 2 Pseudo code for the reduction of a biogeochemical model

```
1: Input: $\vec{\varrho} = \{\varrho_1, .., \varrho_m\}^T \in \mathbb{N}^m$
2: Output: $\hat{q} \in \mathbb{R}^m$
3: for $i = 0$ to $m$ do
4:     $index$ = range of profile in tracer vector for DEIM-point $\varrho[i]$
5:     $y_p = V[index, :]\hat{y}$
6:     $b_p, d_p$ = select corresponding boundary and domain conditions
7:     $q_p = q(y_p, u, b_p, d_p)$
8:     $\hat{q}[i] = q_p[\varrho[i]]$
9: end for
```

This algorithm uses the DEIM-indices as input and provides an evaluation of the biogeochemical model at the desired points. In line 4 the range of the profile for a specific DEIM-point is determined, and then, this part of the high dimensional tracer is computed, through projection with the POD-basis V. After selecting the corresponding boundary and domain conditions the biogeochemical model function is evaluated for this profile. In line 8, the desired point in the profile is saved and the rest of the vector is discarded.

Overall, for one time step of the reduced model, the steps that are described by Algorithm 3 have to be done.

Algorithm 3 pseudo code for time step of reduced model

```
1: Input: point in time $t \in [0, 1)$, $\hat{y}_j \in \mathbb{R}^k$
2: Output: $\hat{y}_{j+1} \in \mathbb{R}^k$
3: compute $\hat{q}$ with Algorithm 2
4: interpolate $A_{ri}$ and $\mathbb{P}_{ri}$ in point $t$
5: $\hat{y}_{j+1} = A_{ri}\hat{y}_j + \mathbb{P}_{ri}\hat{q}$
```

For one time step, the desired parts of the biogeochemical model have to be evaluated, and then, two matrix vector multiplications have to be done.

3.3 Implementation

The code to generate the reduced-order models was mainly written in python using scipy and numpy [28]. The DEIM Algorithm and the precomputing of the system matrices were implemented with numpy. To exchange data with the PETSc library, used by Metos3D, IO routines for PETSc files were created in python.

To compute the SVD of the snapshot matrices, at first, numpy was used. As the matrices have become quite large during the progress of this thesis, and the SVD algorithm in numpy was not suitable for matrices of this size, the library SLEPc [24] was used. This provides the possibility to compute partial SVDs in parallel. The algorithms SLEPc uses are explained in [29], where a performance analysis is given as well.

The reduced model was implemented in python. The biogeochemical models used in Metos3D were included directly with F2PY (Fortran to Python interface generator).

The whole code is freely accessibly via

`https://github.com/neeljp/MasterThesis_POD-DEIM`

and can be used for further improvements or demonstration.

Chapter 4

Numerical Examples

In this chapter the results of numerical experiments with different biogeochemical models are presented. First an investigation of the one tracer N-Model will be presented. Starting with an overview about how many snapshots have to be taken into account to build an accurate reduced-order model (ROM). Afterwards, information about the accuracy and CPU time of ROMs from the N-Model with different POD-DEIM base dimension are given. In Chapter 4.1.2 different ROMs of the N-Model are compared to the full-order model (FOM) in terms of convergence, relative error and speed up. Building up on these results the behavior of ROMs for the N-DOP-Model is investigated and compared to the full-order N-DOP-Model, in Chapter 4.2.

4.1 Experiments with N-Model

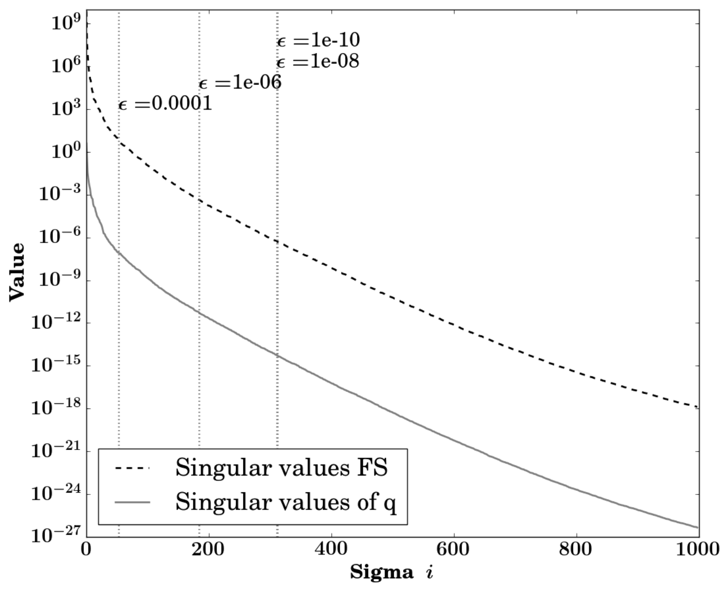


FIGURE 4.1: Distribution of singular values from a SVD of 12_3000, a matrix of snapshots distributed over the whole year.

The first experiments were done with a one tracer model, the so called N-model in Metos3D. To test and show the feasibility of the reduction a specific fixed parameter set $u_r = \{0.02, 2.0, 0.5, 30.0, 0.858\}$ was used to generate snapshots. These parameters are taken from [20], where they are given as the parameter set of a reference solution. A uniformly distributed start concentration of $2.17\frac{mmol}{m^3}$ was used. The time scale was set to 6000 model years and three model states in each month were saved. The reason to only save three states out of the 2880 in one year, was the size of the data. One state vector used $0.422KB$ and two have to be save for each state, one for the full system and the other one for the nonlinear part. Thus, storing 2880 snapshots would have taken $2.37GB$ per year. Therefor, it was tried to reduced the amount of snapshots taken. Different snapshot matrices were build from these saved states to try out how many must be taken to build an accurate reduced-order model (ROM).

Figure 4.1 shows the distribution of the first 1000 singular values from an SVD of a snapshot matrix $S = \{y^i_1, \ldots, y^i_{3000}\}$ for $i = 1, \ldots, 12$ (called 12_3000). These snapshots are distributed uniformly over the whole year and are taken at the end of each month, thus $12 \times 3000 = 36000$ snapshots were taken. The solid line are the singular values of the nonlinear snapshots and the dashed line are the ones from the snapshots of the full system. These two behave very similar, independently from the choice of the snapshot distribution. The vertical dotted lines point out the estimated dimension k of the base with a POD error ϵ, computed with (2.12). It shows that an $\epsilon < 10e-8$ does not change k anymore.

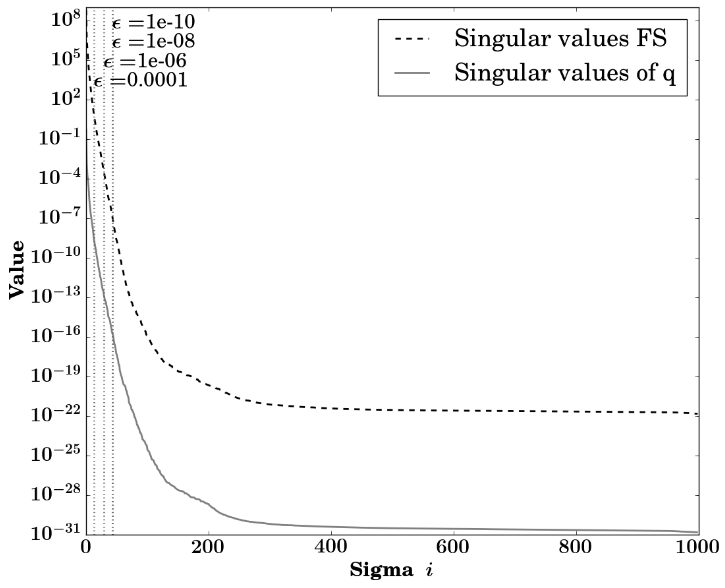


FIGURE 4.2: Distribution of singular values from an SVD of 3_3000 a matrix of snapshots taken only in winter.

In comparison with Figure 4.2 the singular values descend slowly. There the singular values of an SVD from $S = \{y_1^1, y_1^2, y_1^{12}, \dots, y_{3000}^1, y_{3000}^2, y_{3000}^{12}\}$ (called 3_3000) are plotted. In the literature [30, p. 261] it is written that fast decaying singular values are a requirement for constructing efficient ROMs and are an indicator for the quality of the ROM.

The snapshots for the singular values in Figure 4.2 were taken only in winter. The distribution seems to be more feasible for a ROM, but since only data in winter time were taken into account, the ROM resulting from the corresponding singular vectors is inaccurate in summer time, as shown in Figure 4.3. It shows that at least one snapshot in each month has to be used because otherwise the reduced model will leak accuracy in the months that are left out. There the relative error

$$\mathcal{E} = \| y_j^{FO} - y_j^{RO} \|_2 \; / \; \| y_j^{FO} \|_2 \tag{4.1}$$

over the period of a whole year from two different bases is plotted. The one, where only snapshots in winter time contributed to the base, (dashed line) has an increasing error in summer time, which leads to a drifting of the solution. The other one (dotted line), where the snapshots were distributed over the whole year, is more accurate and has nearly the same error over the course of the entire year.

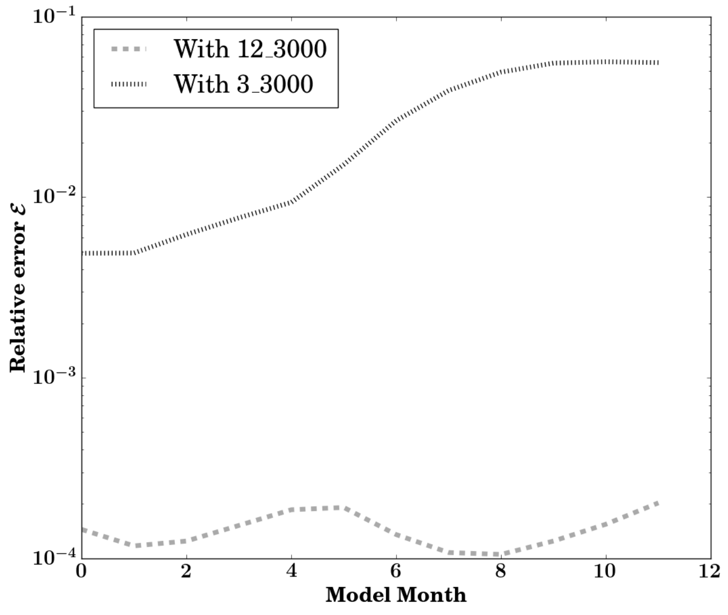
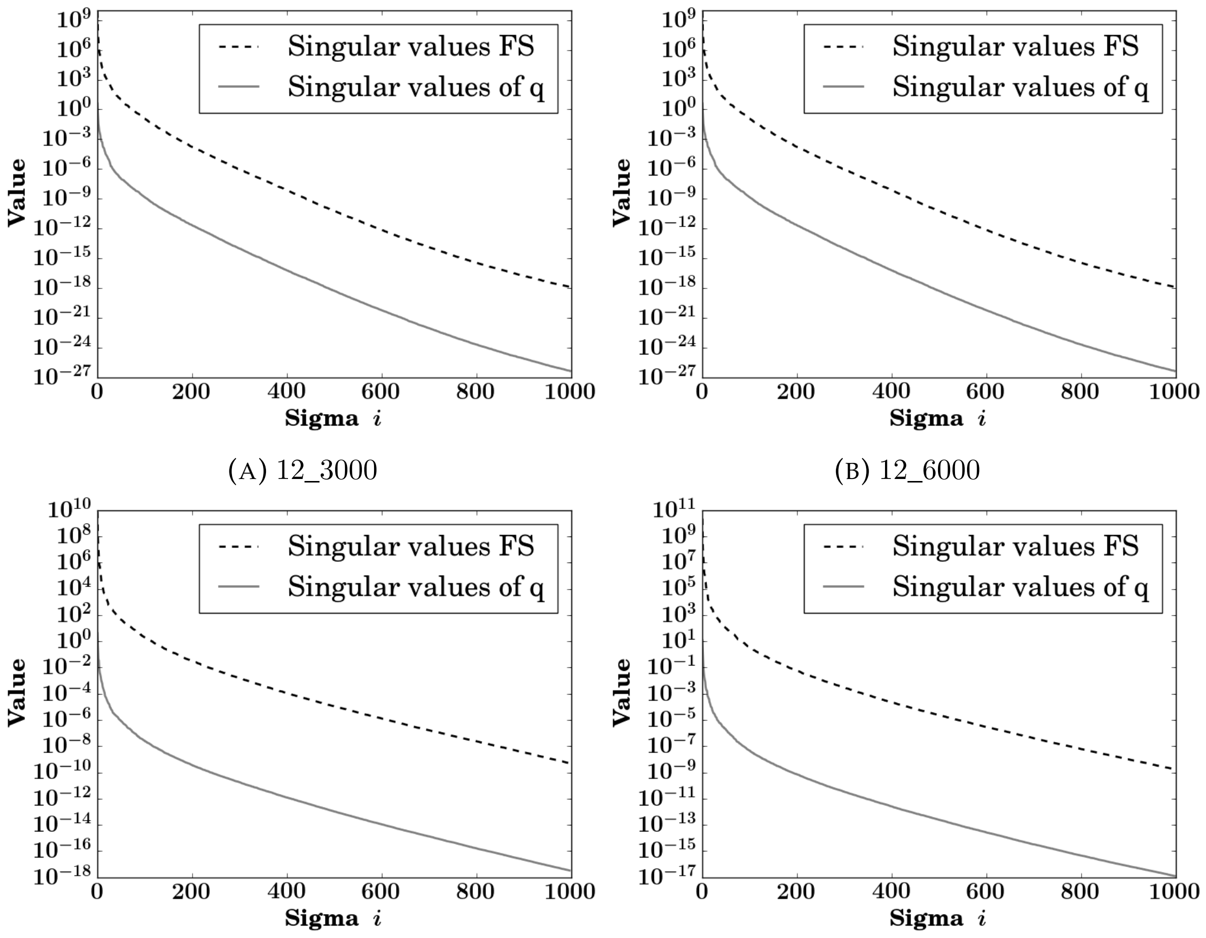


FIGURE 4.3: Relative error $\mathcal{E}$ during the course of a year for two different bases. One where the Snapshots where taken in winter only and the other with snapshots distributed over the whole year.

Comparing the plots of the different singular values in Figure 4.4 with the relative error of the derived ROMs in Figure 4.8, it is not possible to see the correlation of the quality of the ROM with the singular values as proposed in the literature. Interesting is that the singular values of 12_3000 and 12_6000 are very similar and the behavior of the derived ROMs is similar as well, as shown in Figure 4.7 and 4.8. This indicates that taking snapshots after 3000 model years into the snapshot set does not change the ROM. Thus, it is sufficient to take only snapshots from the first 3000 model years, but it is better to take more snapshots during the course of the model years.



(A) 12_3000 (B) 12_6000

(C) 36_1000 (D) 36_3000

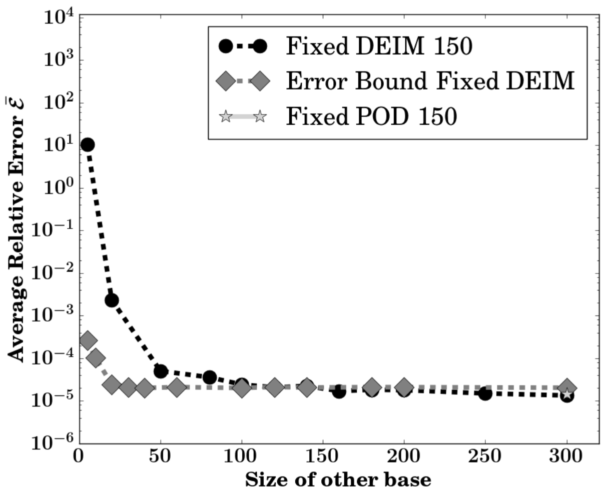
FIGURE 4.4: Distribution of singular values from an SVD of different snapshot matrices.

4.1.1 Approximation error and CPU time

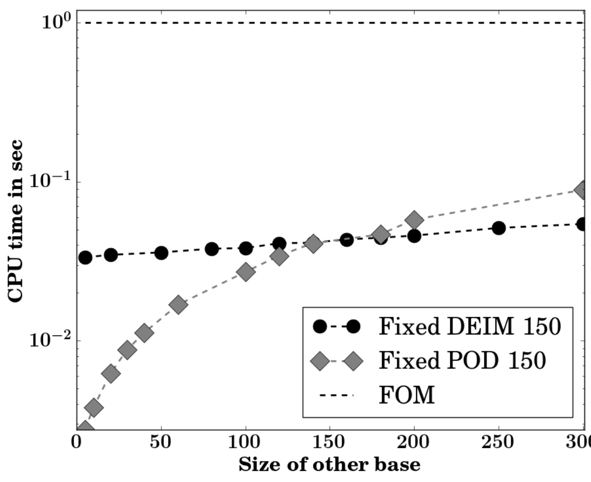
To find out how to choose the dimension of the POD and the DEIM basis, a row of experiments were made. The singular vectors for these bases come from the SVD of uniformly distributed snapshots over 3000 model years (12_3000), see Figure 4.1 for the corresponding singular values. In these experiments the size of one base was fixed and the size of the other one was changed. The size of a base marks the number of left singular vectors of the SVD that are taken to build the base. In Figure 4.5 the behavior of the average relative error

$$\bar{\mathcal{E}} = \frac{1}{n_t} \sum_{j=0}^{n_t} \parallel y_j^{FO} - y_j^{RO} \parallel_2 / \parallel y_j^{FO} \parallel_2 \tag{4.2}$$

with $n_t = 1000$ is shown on the left. Either the DEIM dimension is fixed (black with dots) at $m = 150$ or the POD dimension $k = 150$ is fixed (gray with diamonds). It can be seen that both lines are limited to the same accuracy and also the error of a ROM with both bases dimension of 300 (gray star) does not lead to a better approximation. The fixed POD ROMs reach that point as the dimension of the DEIM base is larger than 40. Below that the approximation gets worse. The fixed DEIM ROMs converge slower at a POD base size of 130. Thus, chose the size of the POD base around 130 and the DEIM base around 40 could be sufficient for the accuracy of the ROM. Note that, in this case, only information from the first 1000 model years are used and for longer model runs this might change.



(A) Relative error



(B) Average scaled CPU time

FIGURE 4.5: The average relative error (4.2) of different ROMs is shown on the left side and average CPU time for these ROMs on the right.

On the right side of Figure 4.5 the average CPU time for one model year of these ROMs with different base sizes is shown. The CPU time was taken over 1000 model years and scaled with the time of the full-order model (FOM). The black line with dots, where the size of the POD base changes, indicates that a large POD base does not lead to a strong increment of CPU time for the whole system. whereas changing the size of the DEIM base leads to a noticeable change in CPU time, as shown by the gray line with diamonds. This was made with a one tracer model and will change with more tracers since the amount of matrix vector multiplications in (3.3) will grow linear with the number of tracers. Overall a speed up between 10 and 100 is possible. For the minimal size of the bases, derived from the left side in Figure 4.5, the speed up would be close to 100. In Figure 4.6 the real average relative error of the bases is compared to the approximated error bound of the POD-DEIM approach from Chapter 2. The bounds are quite pessimistic, but they show a similar behavior as the real error, if the base size changes. There it is even better to see that the error of the POD part is governing the error of the whole system.

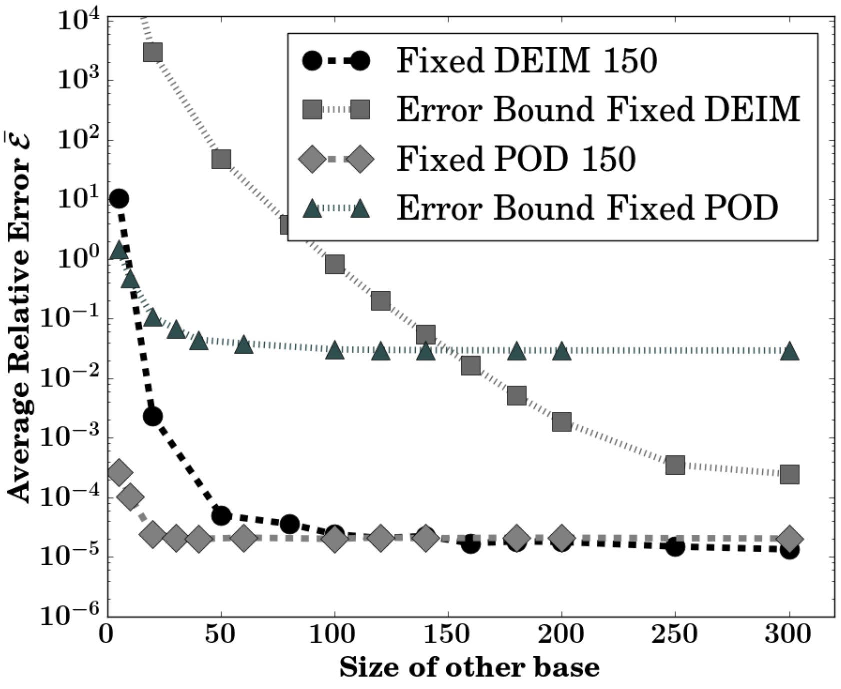


FIGURE 4.6: The average relative error $\bar{\mathcal{E}}$ compared to the error bounds (2.8),(2.26) of the POD-DEIM approach.

In conclusion, increasing the DEIM base size over the size of the POD base does not lead to a better approximation, but this increases the CPU time a lot. The increase of size of the POD base over the size of the DEIM instead does lead to a decreasing approximation error. Therefore, the size of the POD base should be chosen larger than the size of the DEIM base or at least they should have the same size.

4.1.2 Behavior of the Reduced-Order Models

In order to explore the convergence and the error behavior of different bases long simulations were made. The base names begin with the name of the SVD, i.e. 12_3000, which means 12 snapshots per year and 3000 years of data. Then follows a pattern like $P100D50$, where $P100$ means a POD basis of dimension $k = 100$ and $D50$ means a

DEIM basis of dimension $m = 50$. At first, the convergence towards a periodic steady state was tested and compared to the convergence behavior of the full-order model. In Figure 4.7 the spin-up norm

$$S_{norm} = \parallel y_l - y_{l-1} \parallel_2 \tag{4.3}$$

is plotted, which is the norm of the difference between the initial states of consecutive model years l. It shows that none of the ROMs reaches a periodic steady state. The full-order model reaches a periodic steady state after about 7000 model years, as shown in Figure 2 in [20].

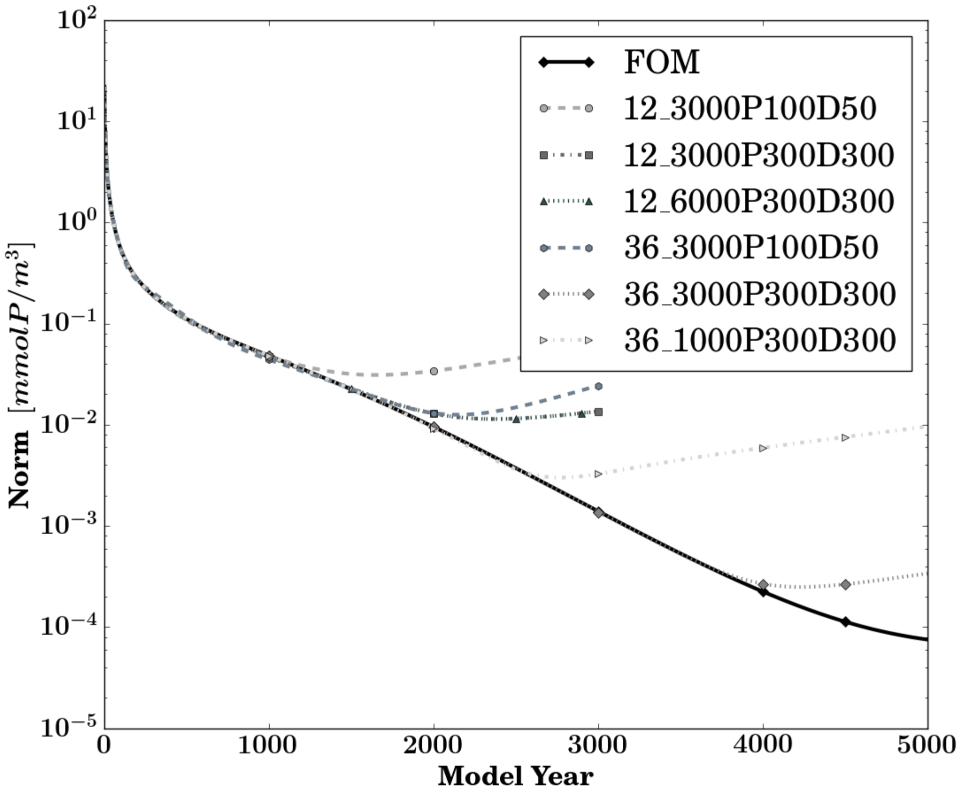


FIGURE 4.7: Spin-up norm of different reduced-order models in comparison with the spin-up norm of the full-order model.

All ROMs diverge at some point. The ROM out of $12\_3000P100D50$ starts to deviate first, at about 1000 model years, form the black line, which shows the spin-up behavior of the full-order model. The ROMs $12\_3000P300D300$, $12\_6000P300D300$ and $36\_3000P100D50$ depart after 1500 model years and begin to diverge. The ROM $36\_3000P100D50$ increases faster than the other two because of the smaller size of the bases. The ROM derived from the longer timeline of snapshots behaves in the same way as the base that is derived from only snapshots out of 3000 model years. The ROM $36\_3000P300D300$ is the last one that departs from the behavior of the FOM, after about

3800 model years. This is far away from the other ones which show similar behavior. The ROM derived from only snapshots of 1000 model years diverges at around 2500 model years, this is later than the 12_3000P300D300 ROM. This indicates again that taking more snapshots during the year is more important than the total number of years that are taken into account.

Figure 4.8 shows the relative error (4.1) of the same bases. The error was measured over 3000 model years and was computed the end of each year. All ROMs show an increasing error over the 3000 years. Again the ROMs 12_3000P300D300 and 12_6000P300D300 show a similar behavior and the ROM out of 12_3000P100D50 has the largest error and the fastest increase.

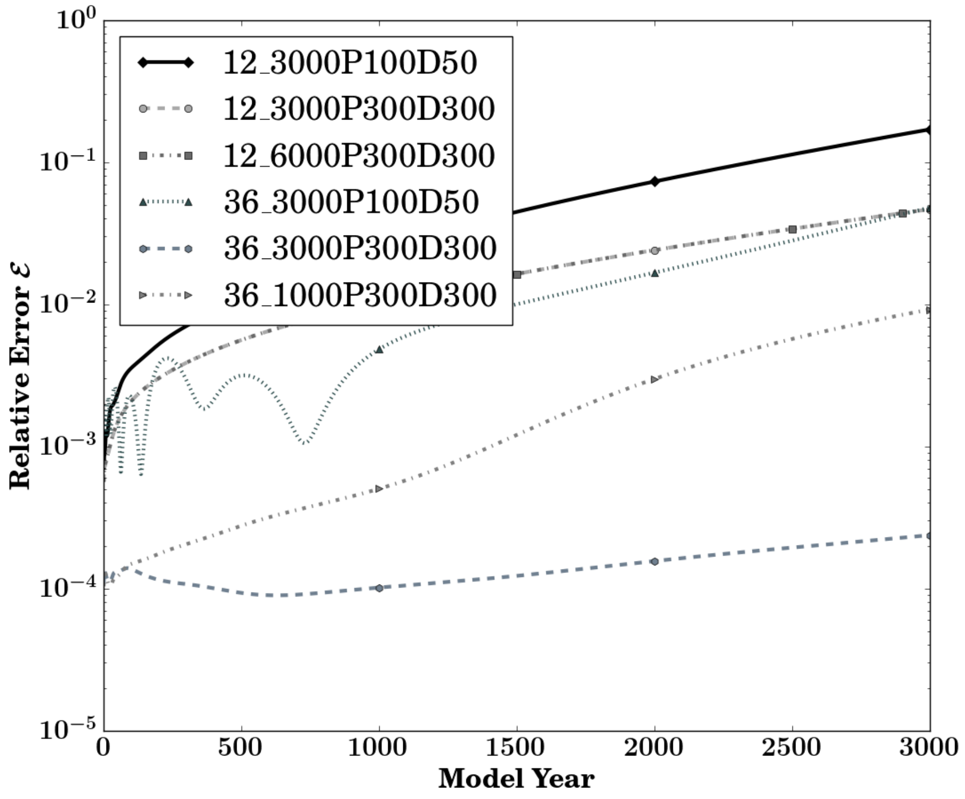


FIGURE 4.8: Relative error $\mathcal{E}$ of different reduced-order models.

The error of all ROMs oscillates at the beginning, usually only the first 100 model years, but the error of 36_3000P100D50 has a strong oscillation until 1000 model years. Afterwards, it shows the same increase as the black line of 12_3000P100D50. The ROM 36_3000P300D300 shows the slowest error increment as well as a very different behavior in the first 500 model years. At first, it oscillates as well, but only a couple of model years, after that it slightly decreases until around 600 model years. Then the increment starts, but it is much slower than that of the other ROMs. The base 36_1000P300D300, where only snapshots from the first 1000 model years were taken, starts with the same error as $36\_3000P300D300$, but shows a stronger increment over the model years.

In conclusion, large bases have a better convergence behavior and a smaller approximation error. Taking snapshots throughout a longer time period than 3000 model years does not change the convergence behavior nor the relative error. Indeed, changing the amount of snapshots per year influences both.

To show some more quantities of the ROMs, four bases have been picked out and compared in Table 4.1. The bases are generated out of two different snapshot sets, one is the set 36_3000, the other is the set 36_1000. For each of them two bases are selected with different POD and DEIM base sizes. For each of them the following quantities have been measured:

- t_{setup} - The time which is required to pre-compute the SVD, the DEIM indices and the matrices A_{ri} and P_{ri}. Note that this only has to be performed once. The SVD was computed with SLEPc on an Intel® Sandy Bridge EP architecture with Intel Xeon® E5-4640 that consist of 8 cores running at 2.4 GHz. The DEIM indices and the base matrices with Intel Core® i7-4750HQ that consist of 8 cores running at 2 GHz. They where computed sequentially and the SVD was computed with 96 cores in parallel.

- S_P - The speed up for a model year, which is the ratio of the computational time for a model year of the FOM and the ROM. They were both computed sequentially on the i7-4750HQ.

- M_a - The relative mass change of the model after a years, scaled with the total mass of the tracer at $t = 0$, i.e. $\frac{\|V_{y_a}\|_1 - \|V_{y_0}\|_1}{\|V_{y_0}\|_1}$, where V_{y_a} is a vector of the masses of the tracer y_a.

- $\bar{\mathcal{E}}_a$ - The average relative error (4.2) over a years.

- $\bar{\mathcal{E}}_*$ - The approximated error bound of the POD-DEIM approach as defined in Chapter 2.

- R_a - The number of spin-ups (model years) that the FOM needs to get to the solution of 3000 model years, if initialized with the output of the ROM at model year a.

SVD	36_3000		36_1000		–
Base	$P100D50$	$P300D300$	$P100D50$	$P300D300$	FOM
$t_{setup}[s]$	$(9676, 10, 39)$	$(9676, 870, 327)$	$(2071, 9, 38)$	$(2071, 989, 350)$	–
S_P	78.29	7.35	73.00	9.08	–
M_{1000}	0.00227	0.00019	0.00102	0.00056	0.00013
M_{3000}	0.03727	0.00056	0.25539	0.00725	0.00039
$\bar{\mathcal{E}}_{1000}$	0.00252	0.00010	0.00156	0.00029	–
$\bar{\mathcal{E}}_{3000}$	0.01409	0.00014	0.06661	0.00250	–
$\bar{\mathcal{E}}_{*}$	77.35	0.11475	44.72	0.05972	–
R_{1000}	2028	1995	2114	1996	–
R_{3000}	1711	40	2755	758	–

TABLE 4.1: Comparison of four ROMs from the N-Model

The time to set up the ROMs is dominated by the computation of the SVD, which is only possible because only a truncated SVD was computed in parallel. In the 9676 seconds (2.6h) and 2071 seconds (0.57h) two SVDs were computed, one for the POD method and one for DEIM. The time to compute the DEIM indices grows fast with the size of the DEIM base because of the complexity of the Algorithm 1.

The relative mass change for the $P100D50$ ROMs is much higher than that of the $P300D300$. It is increasing from 1000 to 3000 model years for all of the ROMs. The 36_3000$P300D300$ has nearly the same mass change as the FOM. The average relative error is increasing a lot from 1000 to 3000 model years, except for the 36_3000$P300D300$ ROM. The approximated error bound $\bar{\mathcal{E}}_*$ is not accurate and roughly three orders of magnitude away from the real error. It indicates that the 36_1000 ROMs should be more accurate than the 36_3000 ROMs. This is due to the fact that the smaller snapshot matrix has smaller singular values and the approximated error bound relies mostly on the singular values.

If the output solution of one of the ROMs after 1000 model years were taken and the FOM is initialized with them, the FOM needs nearly the rest of the 3000 model years to get to the original solution. If the output of the ROMs after 3000 model years is taken only the ROM 36_3000$P300D300$ supplies a solution that is near the one after 3000 model years of the FOM. The other are far away, even further away as the solution at 1000 model years as for the ROM 36_1000$P100D50$, which takes 2755 model years in the FOM. If it is done like this, i.e. combining one ROM with the FOM, all the systems will give an overall speed up of approximated 1.4, if supplied with the ROM solution of 1000 model years. The system supplied with the solution after 3000 model years of the ROM $36\_3000P300D300$ will only need time equivalent to $\frac{3000}{7.35} + 40 = 447$ model years of the FOM to get to the original solution of 3000 model years. This leads to an overall speed up of $\frac{3000}{447} \approx 6.71$. Note that here CPU times of different implementations are compared, the FOM is implemented in C with PETSc and the ROM is implemented in python with numpy and both were run sequentially. It is expected to get a better speed up if the ROM will be implemented in C as well.

In conclusion, the creation of a reduced-order model of the N-Model is feasible and the POD-DEIM leads to an adequate reduction in computational cost. As long as the snapshots and reduced matrices only have to be computed once and then the reduced system is solved many times with them. This leads to the question whether it is possible using one reduced model for different sets of parameters and get accurate solutions. This problem will be described in Chapter 5. In the next chapter the reduction of a two tracer model is explained and the results of experiments with it are presented.

4.2 Experiments with N-DOP-Model

The model used in these experiments uses two tracer elements. It extends the N-model, which considers only inorganic phosphate (PO_4 called N here) with dissolved organic phosphorus (called DOP). All uptake of phosphate is shifted either to export production or to DOP in this model. It is further described in [27]. The model uses one vector for each tracer, and thus, there are two solution vectors. 36 uniformly distributed snapshots per year from the first 1000 model years were generated. For this the parameter set $u = \{0.02, 2.0, 0.5, 30.0, 0.67, 0.5, 0.858\}$ was used and the start concentration was set to $2.17\frac{mmol}{m^3}$ and $1e^{-4}\frac{mmol}{m^3}$, for N and DOP respectively. There are two strategies how to create a ROM for this model, one is to create one snapshot matrix out of the combined tracers vectors, i.e.

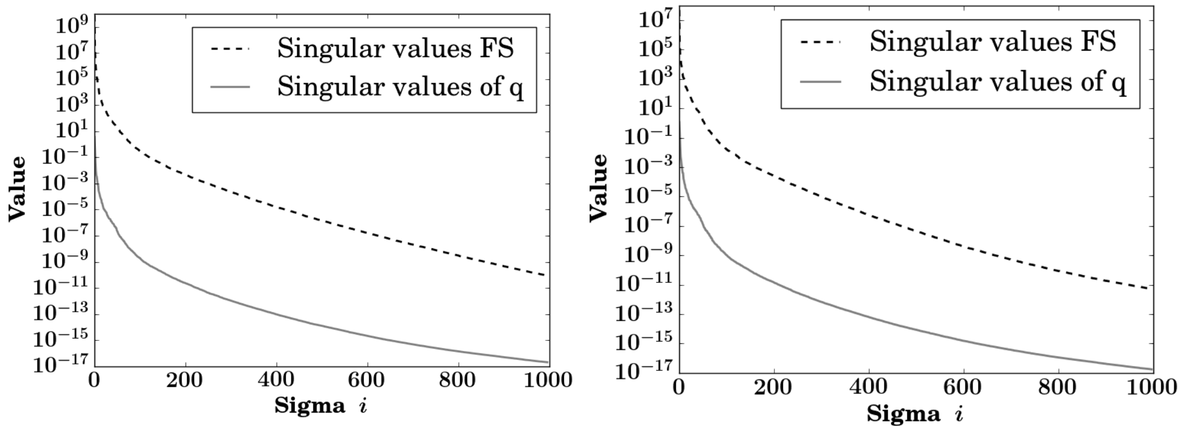
$$S = \begin{bmatrix} | & & | \\ y_1^N & \dots & y_{ns}^N \\ | & & | \\ | & & | \\ y_1^{DOP} & \dots & y_{ns}^{DOP} \\ | & & | \end{bmatrix}.$$

The other option would be to create two independent matrices

$$S_N = \begin{bmatrix} y_1^N & \dots & y_{ns}^N \end{bmatrix}, \quad S_{DOP} = \begin{bmatrix} y_1^{DOP} & \dots & y_{ns}^{DOP} \end{bmatrix}$$

and create one POD-DEIM base for each tracer. The first one has not lead to a usable ROM because the DOP solution vector was tremendously inaccurate. A possible reason might be the fact that the DEIM algorithm has picked out indices only in the upper part of the combined vector, thus, in the y^N part. Consequently, for the nonlinear part only information from the solution of one tracer were taken into account. Thus, the part of other tracer were only bad approximated.

In the following, the results of the second, decoupled approach are presented. Therefore, a matrix with 36 equally distributed snapshots for each model year over 1000 model years was created and the SVD was computed. The singular values of both tracers in Figure 4.9 look similar to those from the N-Model. Thus, it is expected that the resulting ROM will give similar results as well.



(A) Singular values of S_N (B) Singular values of S_{DOP}

FIGURE 4.9: Distribution of the singular values of 36_1000 from the N-DOP-Model.

In Figure 4.10 the spin-up norm (4.3) of two ROMs of the N-DOP-Model is shown for each tracers. The ROM with the smaller base matrices shows a very different spin-up behavior than the FOM. Both tracer concentrations are oscillating during the first 500 model years. Afterwards, the N concentration is changing a lot from model year to model year, but the change does not increase. The DOP concentration converges to a state with only small changes each model year. The other ROM, that is shown in Figure 4.10, has nearly the same change of concentration over the 3000 model years as the FOM. This is interesting because the ROM was only created with information of the first 1000 model years and such a ROM of the N-Model diverges stronger from the FOM, compare Figure 4.7.

The relative error (4.1) over 3000 model years for each tracer of these ROMs is presented in Figure 4.11. First of all, note that in this case it relates to a relative error

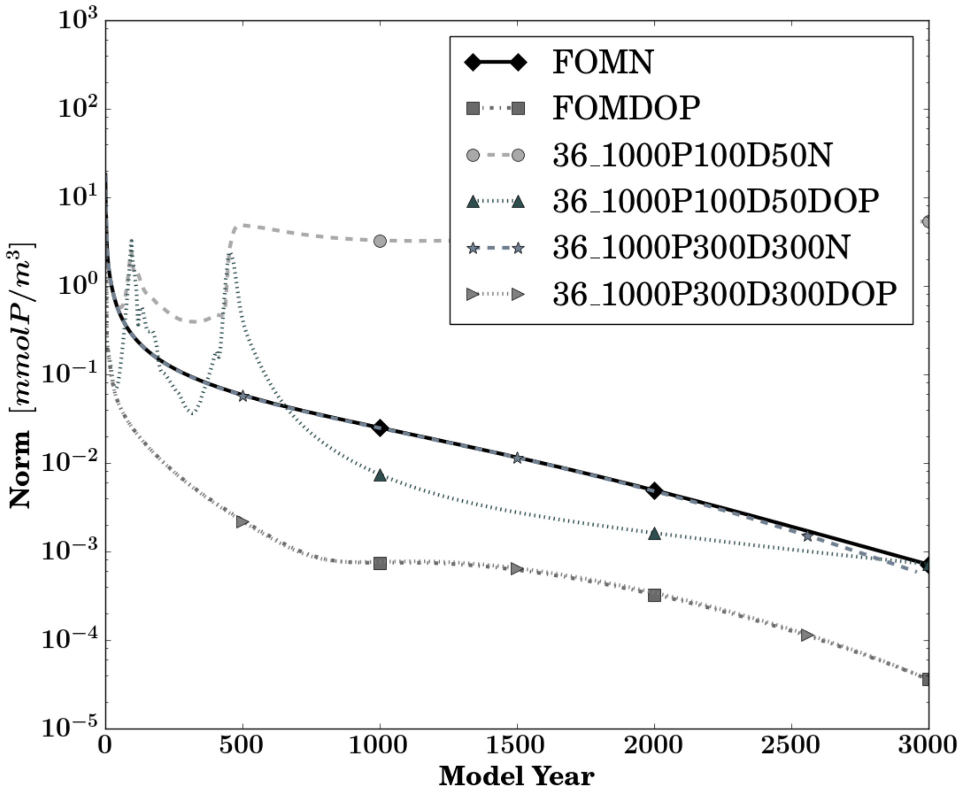


FIGURE 4.10: Spin-up norm of two ROMs in comparison with the spin-up norm of the FOM of N-DOP.

and values higher than one mean that the error is larger than the 2-norm of the original solution vector of the FOM. Thus, the error of both concentrations (N,DOP) of the smaller ROM is large compared to the overall concentration in the FOM, and the N part shows a strong increasing over the model years. The DOP part is nearly constant after one peak at about 500 model years. Since the relative error of the ROM $36\_1000P100D50$ increases far beyond one after 500 model years, the solution of this ROM is not close to the solution of the FOM. The relative errors of both concentrations of the ROM $36\_1000P300D300$ are increasing over the 3000 model years, at first, stronger and later they stay nearly constant. The are both very similar in each model year and in an order of 10^{-3}.

Table 4.2 shows the same information as Table 4.1 for the N-DOP-Model. The setup times of the ROMs are roughly doubled because each tracer has its own bases, and thus, its own setup process. Due to this fact the speed up is much smaller as well. Since each tracer has its own and mostly different DEIM indices, the biogeochemical model has to be evaluated nearly twice the number of DEIM indices. In the ROM $36\_1000P300D300$ only 52 DEIM indices of the two tracers were identical, this is less then 10%. In addition, the number of matrices that must be interpolated is doubled as

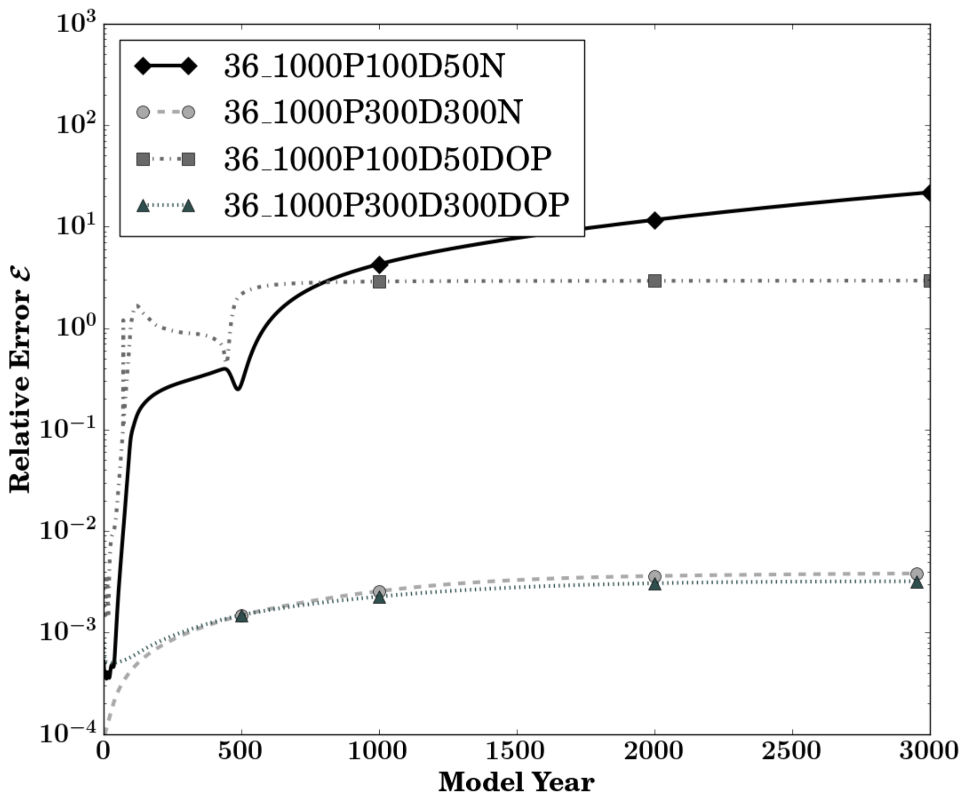


FIGURE 4.11: Relative error $\mathcal{E}$ of two ROMs of the N-DOP-Model.

well because each tracer has its own POD base. For a comparison of the CPU time of the main operations during one time step of the N-Model and the N-DOP-Model look in Appendix A.3.

The relative mass change in the DOP concentration is tremendously high in all models as well as in the FOM, but the mass change of the ROM $36\_1000P300D300$ is nearly equal to that of the FOM. Note that the start concentration of DOP is $1e^{-4}\frac{mmol}{m^3}$, and thus, it has a really low amount of substance compared to the start concentration of N. The N part of the model $36\_1000P300D300$ has less mass change than the FOM. This mass change becomes even smaller after 3000 model years than after 1000, although, the mass change usually increases over the course of the model years.

The relative error of the ROM $36\_1000P100D50$ is higher than one as well after 1000 as after 3000 model years and for both tracers. Supplying the solution of this ROM into the FOM, it does not come near the the original solution of 3000 model years within the time of 3000 model years. Basically this results because of the poor approximation of the ROM, the resulting high relative mass change of the system, as well as the occurrence of a lot negative concentration values. For DOP$\approx 75\%$ and for N$\approx 1\%$ compared to approximately 47% and 0% negative values in the solution of the FOM after 3000 model years. The approximated error bounds for the ROMs are much closer to the actual error than in the N-Model. The approximated error bound for the DOP tracer is lower than the actual error in both ROMs. Since the tracers interact with each other, it is a question that has to be answered, whether the error bound of each tracer gives information about the error of the whole system.

The error of the ROM $36\_1000P300D300$ is similar in both tracers, and it is comparable to the error of the same sized ROM of the N-Model. The solution of this model is usable to produce a solution with the FOM that is close to the original solution. The FOM takes 2011 and 495 spin-ups to get to the original solution and with this setup a speed up of 1.7 is possible. This is less than the speed up of the N-Model which depends mostly on the fact that a separate POD-DEIM base is used for each tracer. Overall, the POD-DEIM approach can be applied to the N-DOP-Model and the other multi tracer models, if a separate POD-DEIM base is used for each tracer.

Base	$36\_1000P100D50$		$36\_1000P300D300$		FOM	
$t_{setup}[s]$	$3965, 19, 69$		$3965, 1989, 646$		$-$	
S_P	14.47323		2.42209		$-$	
Tracer	N	DOP	N	DOP	N	DOP
M_{1000}	2.607	1039.21	0.00652	184.282	0.00854	183.912
M_{3000}	18.192	1073.76	0.00550	186.397	0.00874	185.631
$\bar{\mathcal{E}}_{1000}$	1.293	1.803	0.00143	0.00142	$-$	$-$
$\bar{\mathcal{E}}_{3000}$	8.325	2.543	0.00279	0.00243	$-$	$-$
$\bar{\mathcal{E}}_{*}$	6.517	0.38633	0.00862	0.00044	$-$	$-$
R_{1000}	$-$		2011		$-$	
R_{3000}	$-$		495		$-$	

TABLE 4.2: Comparison of two ROMs from the N-DOP-Model

In the next chapter the results of experiments with ROMs of the N-Model for parameter studies are presented.

Chapter 5

Experiments for Parameterized Reduced Order Models

The previous examples showed the feasibility to use the POD-DEIM approach on marine ecosystem models with a fixed set of parameters. In [15], the authors propose to use the POD-DEIM approach for parameter studies of dynamical systems. In this case, it means to simulate the full-order model for multiple parameter sets and use all the snapshots to create a reduced-order model. To investigate this the N-Model with one tracer is used again, but this time with different parameter sets. Since the parameter space is quite large, Latin Hypercube sampling, as proposed in [21], is used to generate ten samples (cf. Table A.1) within the bounds that are given in Table 2 in [20]. In Chapter 4, the amount of snapshots for the smallest setup of viable ROM was $36 \cdot 1000 = 36000$ snapshots. Thus, for every parameter sample 36000 snapshots are used. Since a SVD must be computed of the whole snapshot matrix, there exist a soft upper bound for the number of snapshots, and thus, for the number of parameter samples. The combined snapshot matrices are

$$S = [Y_1| \dots |Y_9], \quad S_q = [Q_1| \dots |Q_9],$$

where Y_i and Q_i are the snapshot matrices of the solution corresponding to the Latin Hypercube sample i. These are in total 360000 snapshots, thus, the matrices S and S_q have the dimension 52749×360000. The singular values of this SVD are visible in Figure 5.1. They decrease much slower than the ones in Figure 4.4 from the N-Model. The singular values of the full system matrix decent only to a value near 10^0, this is much higher than the previously showed singular values. The approximated error bound of a ROM, created out of the corresponding left singular vectors, is tremendously high, in an order of 10^4. It was not possible to create a ROM that generates a solution that is close to the original for any of the parameter sets listed in Table A.1, with which the model was created. The relative error is always over 1 and it is oscillating a lot in the course of 3000 model years, cf. Figure A.8. With a new parameter set that was not used

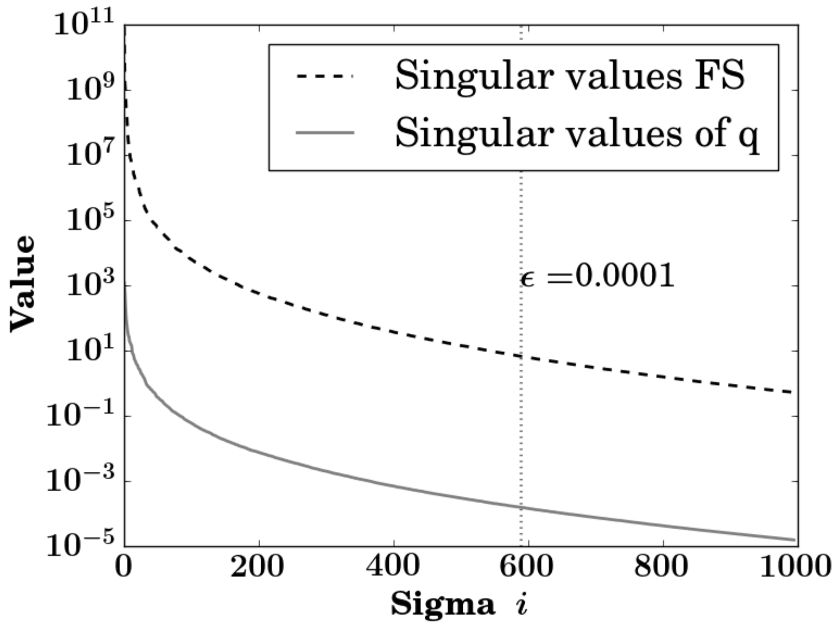


FIGURE 5.1: Distribution of singular values of S and S_q.

to create the model, for example $1.5 \cdot u_r = u_r + 50\% = \{0.03, 3., 0.75, 45., 1.287\}$, the result is similar. The ROM, created out of multiple solution trajectories, is very inaccurate, cf. Figure A.9. Thus, the approach to create one ROM out of multiple trajectories seems not to lead to an accurate and useful one. One possible explanation for this is the diversity of the solutions that the FOM generates for different parameter sets. Together, they can not be well approximated within the same low dimensional subspace. Therefore, the subspace, generated by the POD method, is only a poor approximation.

Another idea is to create multiple ROMs for multiple parameter sets. And then for a new parameter set use the ROM that was created with the parameter set that is the closest to the new one. This approach should be possible if a ROM can produce an acceptable solution also for a set of parameters that is different from the parameter set the ROM has been created with. To try this out, four parameter sets are taken. The parameter sets are deviations from the reference parameter set $u_r + x\%$ with $x \in \{5, 10, 20, 50\}$. Using the ROM $36\_3000P300D300$, from Chapter 4.1, which was create with the parameter set u_r, simulation are made. The relative error between these spin-ups and the corresponding original solution of the FOM is shown in Figure 5.2. In the figure it is visible that the further away the parameter set from u_r is, the larger becomes the relative error of the solution of the ROM to the original solution of the FOM. All spin-ups show a decrease of the error in the first half of the 1500 model years and an increase in the second half of the 3000 model years. The further away the parameter set is, the earlier the relative error begins to increase. The relative error is in the order of 10^{-1}, thus, the solution that the ROM generates for a different set of parameters, seems to be not usable. The dashed lines with the left arrow show the relative error between the two FOM spin-ups. One spin-up was executed with parameter set u_r and the other spin-up was run with one of the deviations $u_r + x\%$. Plotted in order 5, 10, 20, 50 from bottom to top. Compared to corresponding relative error of the FOM, the solutions of the ROMs are most of the time less accurate. And it seems that carrying out a ROM simulation with a different set of parameters does not lead to a significant change in the solution. The change is larger, the further away the set of parameters is, but then, there is a stronger divergence after around 1000 model years.

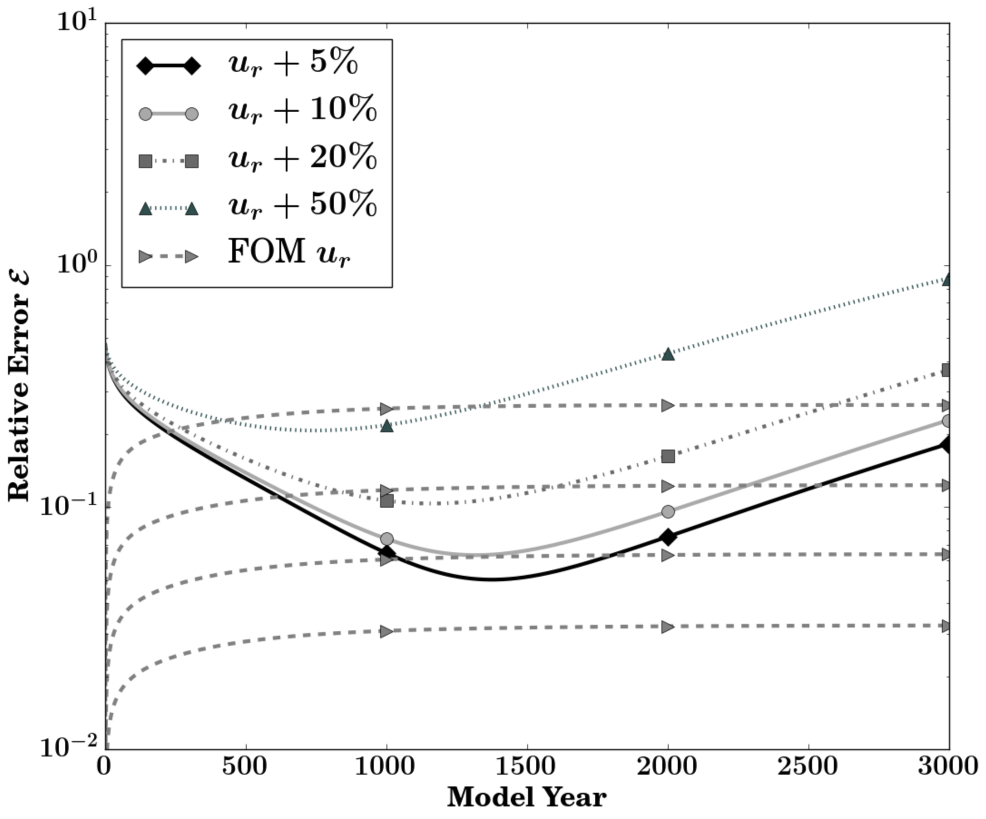


FIGURE 5.2: The relative error between the ROM 36_3000P300D300 and the FOM for different parameter sets $u_r + x\%$, where $x \in \{5, 10, 20, 50\}$. The dashed lines with the left arrow show the relative error between the FOM spin-up with parameter set u_r and one of the deviations $u_r + x\%$. Plotted in order 5,10,20,50 from bottom to top.

In Figure 5.3, the relative error between the ROMs spin-ups for the parameter sets $u_r + x\%$ with $x \in \{5, 10, 20, 50\}$ and the FOM solution of u_r is visible. The error is large, which means that the spin-ups of the ROM are not close to the original solution of the FOM using the parameter set u_r. Thus, the ROM results in a completely different solution if it is executed with a different parameter sample than it was created with and they start to diverge early.

In conclusion, executing a ROM with a different set of parameter than it was created with does not lead to a usable solution. Thus, the idea of creating multiple ROMs and using them for parameter studies seems not to be feasible. Overall, it has not been possible to find a ROM or multiple ROMs that can be used for new sets of parameters and resulting in an acceptable solution.

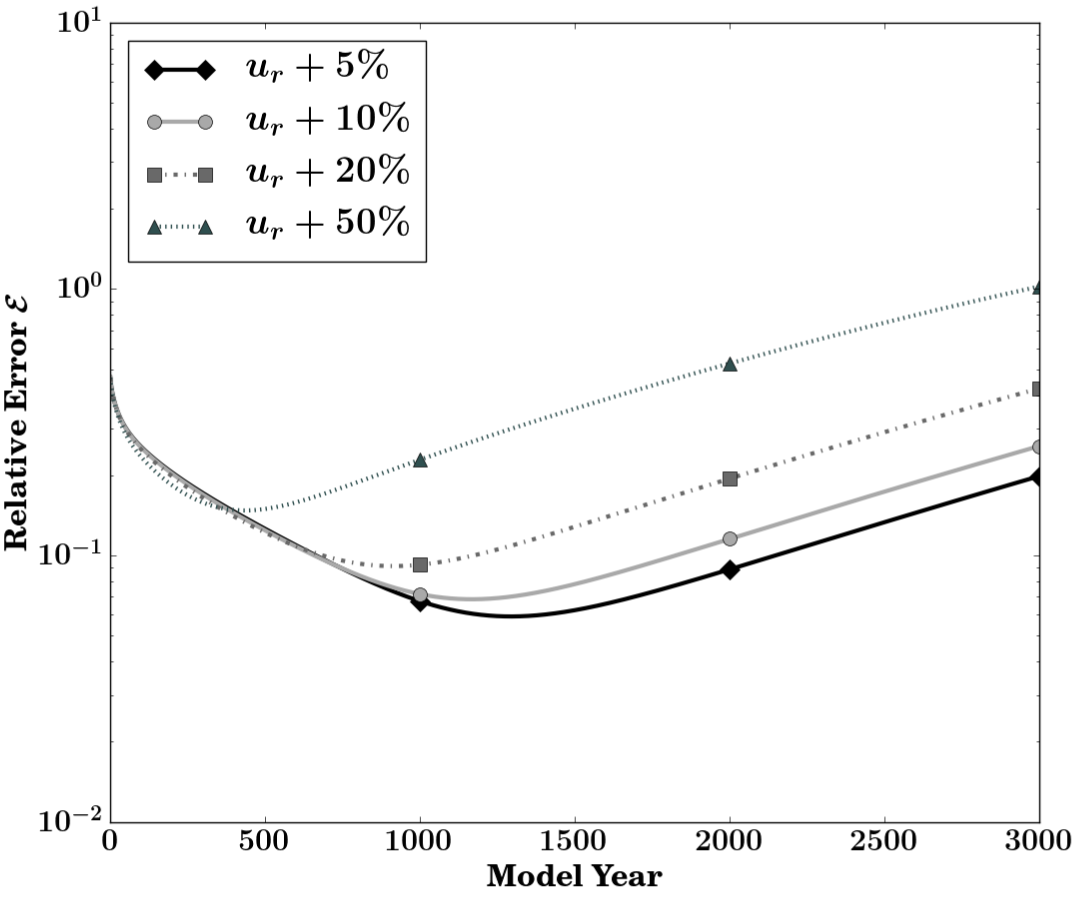


FIGURE 5.3: The relative error between the ROM $36\_3000P300D300$ for the parameter sets $u_r + x\%$ and the FOM solution of u_r, where $x \in \{5, 10, 20, 50\}$.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, the nonlinear MOR method POD-DEIM has been applied to three different marine ecosystem models of the software package Metos3D. Therefore, the reduced model equations (3.3) were established and implemented. Then, an analysis of the one tracer model, the N-Model, was made, in which the feasibility was tested. For that the manner of how many information from the full-order model are needed for an accurate reduced-order model was investigated. Moreover, it was examined how the size of the reduced bases affect the accuracy and CPU time of the reduced-order model. On the basis of these results six reduced-order models of the N-Model were compared to the full-order model. The approach was extended on the two tracer model, the N-DOP-Model, and two reduced-order models were compared to the full-order N-DOP-Model. Afterwards, it was tried to create reduced-order models of the N-Model, which could be used for parameter studies. For that two approaches were tested. The first one was to create a reduced-order model out of multiple trajectories created with different parameter sets. And the other one was to use one reduced-order model for multiple parameter sets.

One main goal of this thesis was to demonstrate the feasibility of the nonlinear MOR method POD-DEIM on marine ecosystem models with respect to accuracy and computational speed up. The numerical experiments in Chapter 4 have demonstrated the feasibility for a fixed set of parameters. The POD-DEIM approach leads to a significant speed up and a solution that is close to that of the full-order model measured in the 2-norm. For the two tracer N-DOP-Model the speed up is smaller because of the fact that a base for each tracer has to be used to get an accurate reduced-order model.

The idea of using the POD-DEIM approach to get a reduced-order model for parameter studies has turned out to be not practical. The main reason for that is diversity of the solutions for different sets of parameters, and thus, the approximation within a low dimensional subspace is very inaccurate.

6.2 Future Work

At this point, some directions for future research, that might build up on the work of this thesis, are presented.

Selection of snapshots

In Chapter 4.1 the selection of snapshots for the SVD is not optimal and could be further improved. Since there is no theoretical approach to optimize the selection, it is only possible with numerical experiments. A problem in that area is the soft upper bound of the total number of snapshots that is given by the SVD algorithm. For that the Algorithm 5.1 in [15] may be interesting. This algorithm uses the Gram-Schmidt QR factorization with a reorthogonalization algorithm from [32]. The algorithm detects and eliminates redundant snapshots as well as prevents the loss of orthogonality of the POD basis due to the numerical instability of the Gram–Schmidt process.

Creating a reduced-order model for parameter studies

For parameter studies it could be tried to create a reduced-order model only for a subset of the parameters, thus, changing only some parameters and leave the others fixed. This could lead to a usable model since the space spanned by the solution should be smaller. Another idea is to create a reduced-order model for a subspace of the parameter space, thus, use all parameters but only create a model for parameters in the range of $u_r \pm x\%$. The idea is the same, the reduction of the solution space, and thus, a better low dimensional approximation.

Create combined ROMs

After the experiments that are showed in Appendix A.1, two ideas came up. Since the convergence of the ROM presented in Appendix A.1 is much faster than the convergence of the ROMs presented in Chapter 4.1.2, the first idea is to increase the overall speed up by using multiple of these ROMs. Each ROM has to be created with a full trajectory of a different model year, i.e. 500, 1000, 2000, 3000. Then, the first model is initialized with a starting concentration and simulated until it has reached the point, where it starts to diverge. Afterwards, the next ROM is initialized with the solution of the previous one and simulated until they have reached their point of divergence. This could lead to an increase of the overall speed up and could increase the accuracy as well, if enough ROMs are used.

The other idea is to use a ROM, as in Appendix A.1, in the opposite way it is used in Chapter 4.1.2. There, the ROMs were executed first. Thereafter the solution it has generated was improved with the full-order model. It could be feasible to run the FOM until 1000 or 2000 model years, and then, supply this solution into a ROM as in Appendix A.1. Being initialized with a solution from that point the ROM converges much faster than the FOM. After executing the ROM the solution could also be improved with the FOM.

The benefit of these two approaches is that a lot of spin-ups are “skipped”, and thus, the overall speed up would be much higher than it is presented in Chapter 4.1.2.

Appendix A

Appendix

A.1 Creating a ROM from a trajectory of one model year

During this thesis the idea came up to create a reduced-order model (ROM) using only a trajectory of one of the later model years. The aim of the reduced-order model is to get a solution that is close to the periodic steady state of the full-order model. Thus, a model that is created with only information of that steady state could maybe converge much faster to this state. A ROM out of snapshots of a full trajectory of the N-Model in model year 3000 was created. A full trajectory means that after every time step a snapshots was taken. Thus, there are 2880 snapshots. The first 500 singular values of the SVD of the matrix of these snapshots are visible in Figure A.1. They decay fast and to a very small value of 10^{-27}. The relative error of a run with a ROM, created out of the

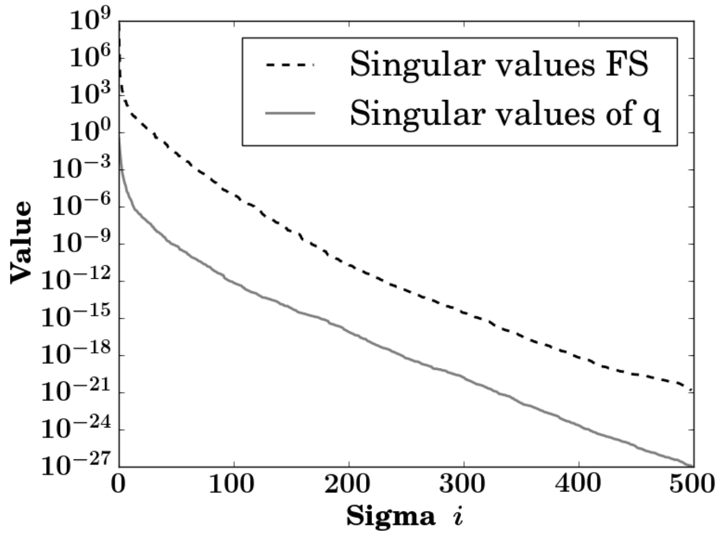


FIGURE A.1: Distribution of singular values from a SVD of 2880_1_{3000}, a matrix of snapshots of a full trajectory of the model year 3000.

corresponding left singular vectors, over 3000 model years is visible in Figure A.2. The ROM $2880\_1_{3000}P150D150$ was initialized with an uniformly distributed concentration of $2.17\frac{mmol}{m^3}$. The black line shows the relative error to the corresponding model year of the FOM. The gray dashed line shows the relative error to state of the FOM in the model year 3000. Both are similar, very high and increasing most of the 3000 model years. Thus, starting with a uniformly distribution does not lead to a usable behavior of the ROM. This is because the POD base projects the initial concentration vector on a low dimensional subspace, i.e. $V^T y_0 = \hat{y}$, but the vector can not be well approximated in that subspace.The approximation error $\| y_0 - VV^T y_0 \|_2 = 234.3$ is very high. Thus, the initial projection leads to a large error of the ROM. Initializing the ROM with a different

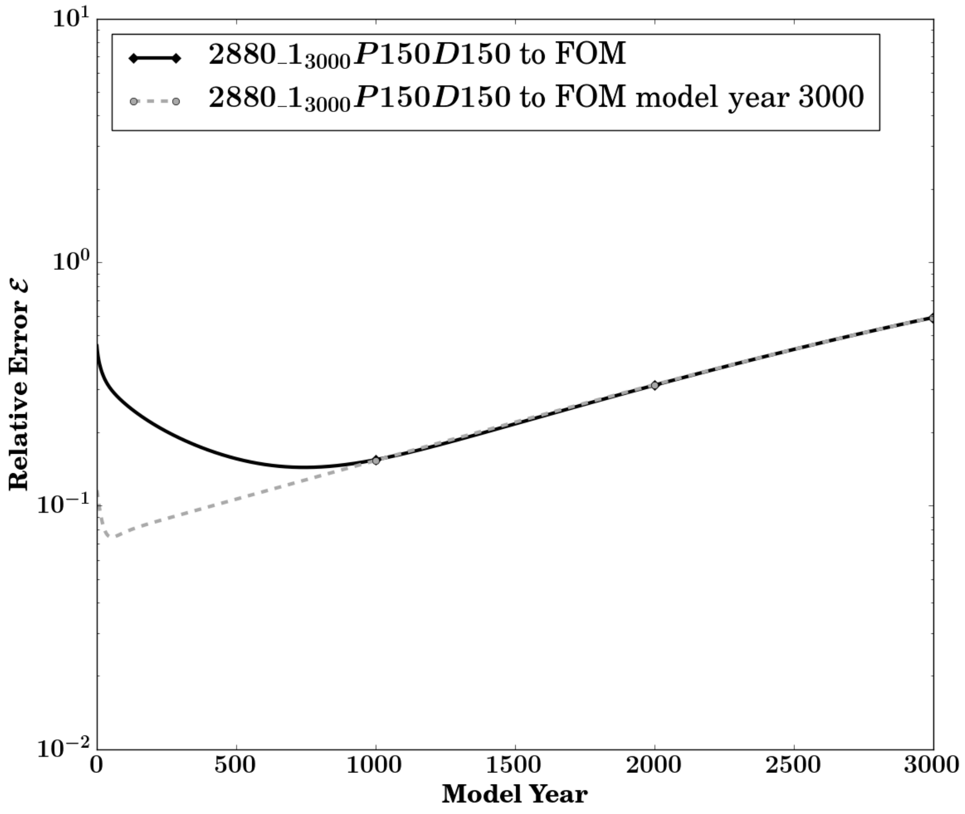


FIGURE A.2: Relative error $\mathcal{E}$ of the reduced-order model $2880\_1_{3000}P150D150$ initialized with an uniformly distributed concentration of $2.17\frac{mmol}{m^3}$ to the FOM. The black line shows the relative error to the corresponding model year of the FOM. The gray dashed line shows the relative error to state of the model year 3000 of the FOM.

concentration, which can be better approximated in the subspace spanned by the POD base, could lead to a better result. Therefore, the ROM was initialized with the solution of the FOM in model year 2000. The approximation error $\| y_{2000} - VV^T y_{2000} \|_2 = 1.573$ is much smaller. The relative error to the state of the FOM in model year 3000 is shown by the dashed gray line in Figure A.3. It is clearly visible that the ROM converges in the first 150 model years to the desired solution and diverges after that. Thus, it would be possible to use the solution of the ROM at that point, where the relative error has its minimum. Since the relative error can only be computed if the original solution is known, it would be more practical to select that point out of other information. The

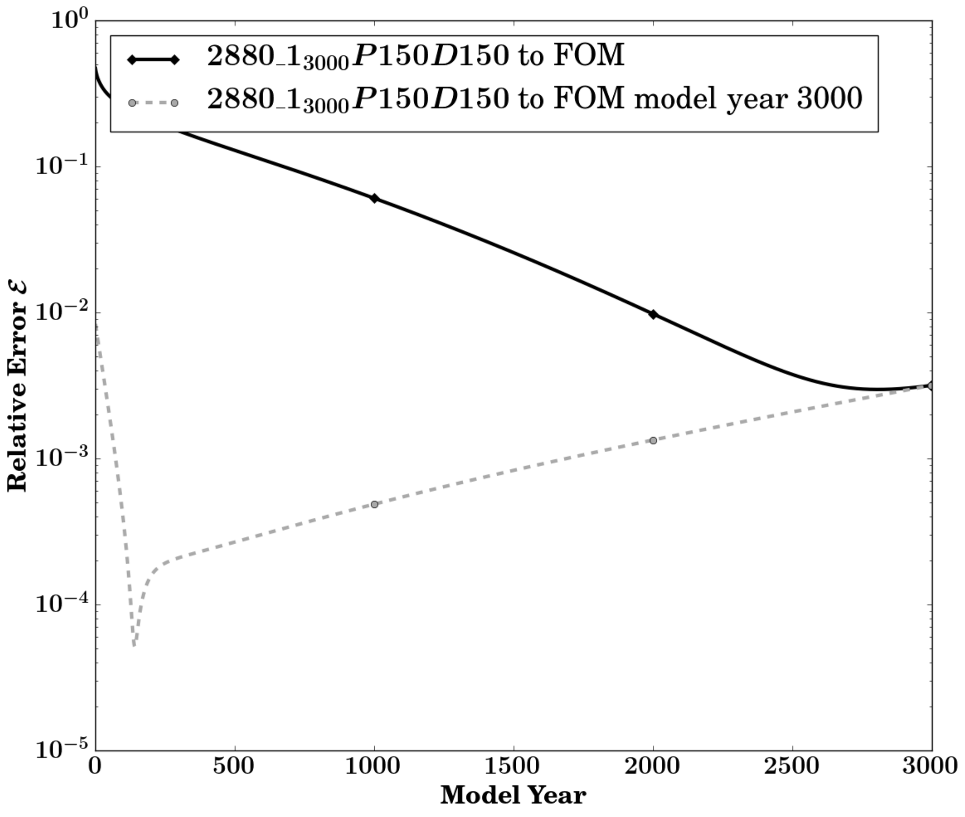


FIGURE A.3: Relative error $\mathcal{E}$ of the reduced-order model 2880_$1_{3000}P150D150$ initialized with the solution of model year 2000 of the FOM. The black line shows the relative error to the corresponding model year of the FOM. The gray dashed line shows the relative error to state of the model year 3000 of the FOM.

spin-up norm of this ROM shows an minimum at that point as well, visible in Figure A.4. It is not exactly the same point but it is quite close. The minimum of the relative error occurs in model year 141 and the minimum of the spin-up norm in model year 231. The information of the spin-up norm could be used to stop the ROM at the right model year to get an usable solution. Thus, a ROM created only with the trajectory of on model year can be used to generate a usable solution. It only has to be initialized with a concentration distribution that is closer to the one of the year the ROM was created with.

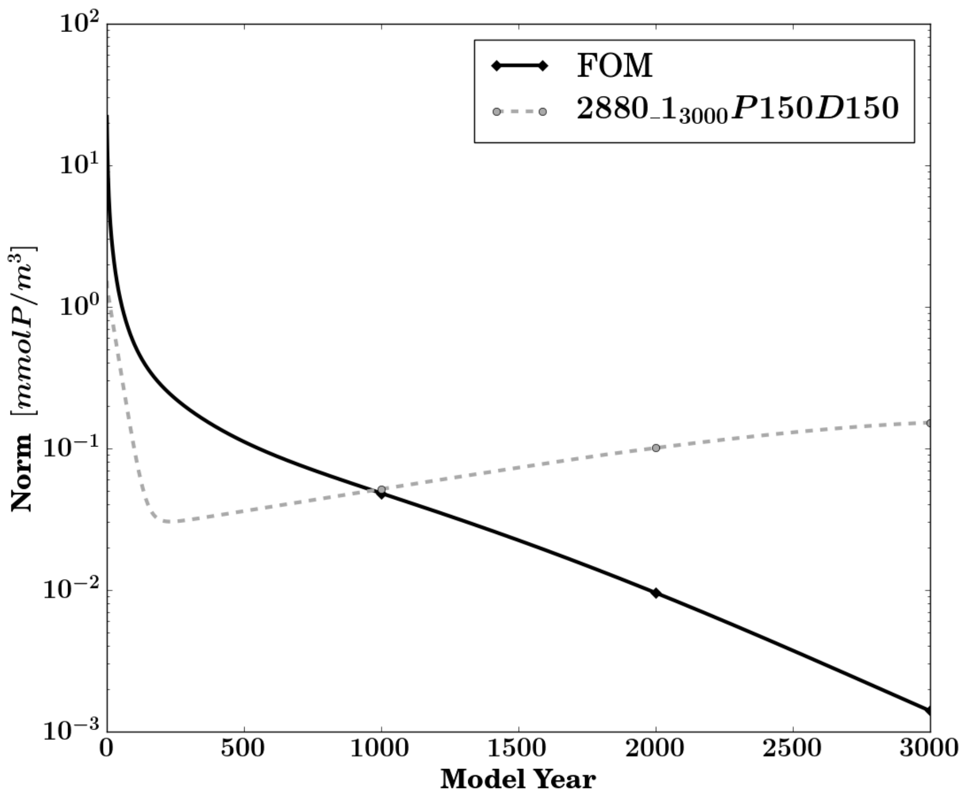


FIGURE A.4: Spin-up norm of the reduced-order model $2880\_1_{3000}P150D150$ in comparison with the spin-up norm of the full-order model.

A.2 Implementation Issue

In the progress of this thesis, tests have shown that there is a small numerical difference in the matrix vector multiplication implemented in numpy and PETSc. The same matrix vector operation has different results. This difference has been tracked down to the different order in which the addition in the matrix multiplication is performed. There, with some matrices vector combinations a difference of $4.44e^{-16}$ occurred, which is double the machine epsilon $\epsilon = 2.22e^{-16}$ for double precision floating point numbers. This happens because floating point additions are not necessarily associative. In Figure A.5 the absolute error between a model run in python and Metos3D is plotted. The example code and data can be found via

`https://github.com/neeljp/matMult_error_python_petsc`.

and can be used for further investigations at any time.

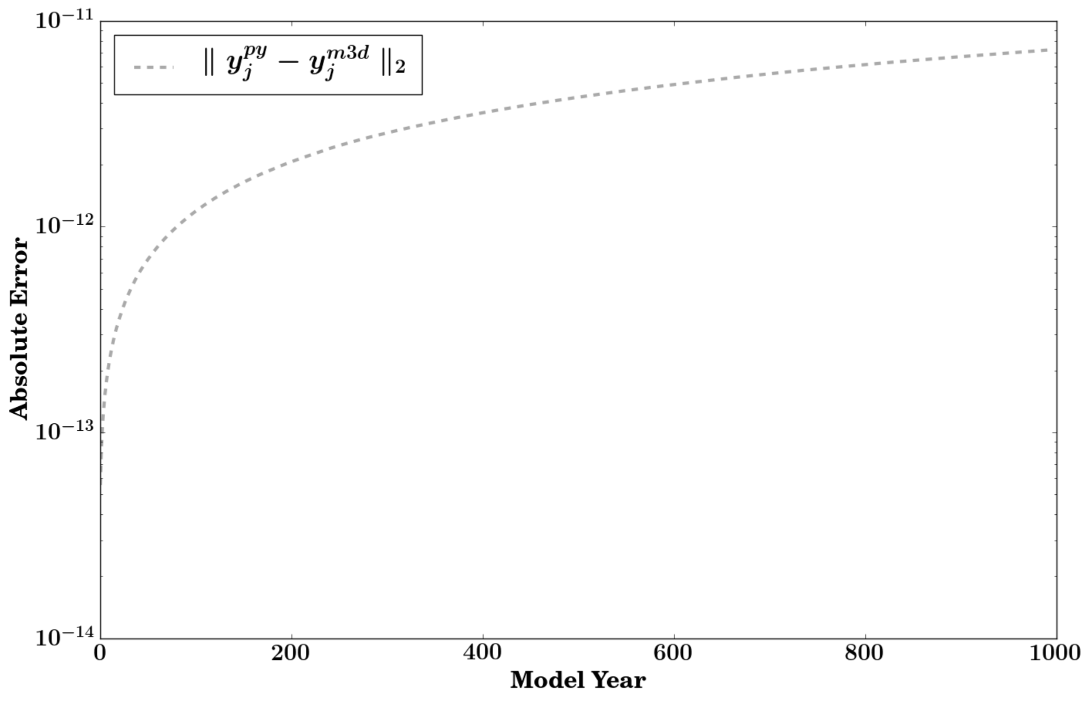
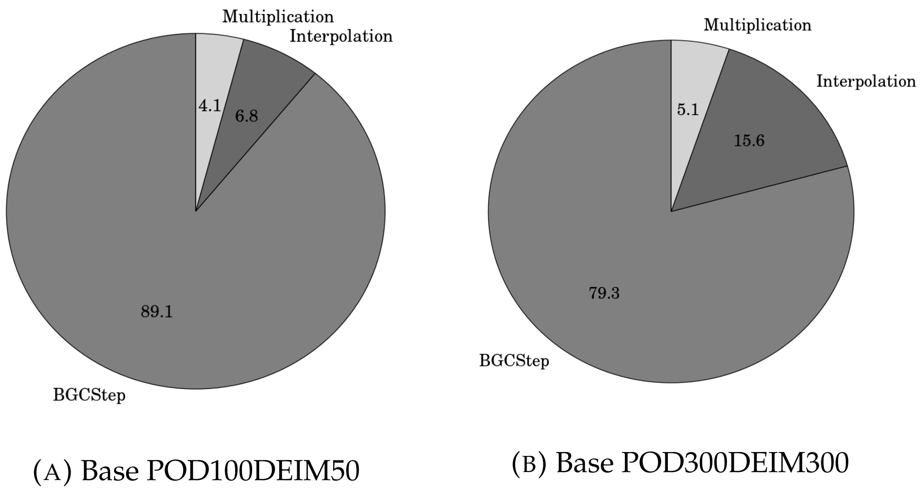


FIGURE A.5: Absolute error $\| y_j^{py} - y_j^{m3d} \|_2$ of a model run with python and with PETSc over 1000 model years.

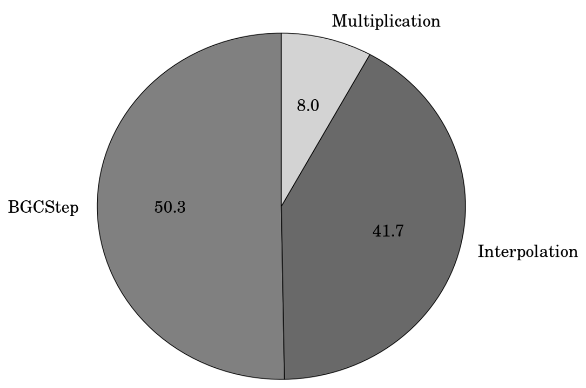
A.3 Distribution of the CPU time of the main operations of the ROMs



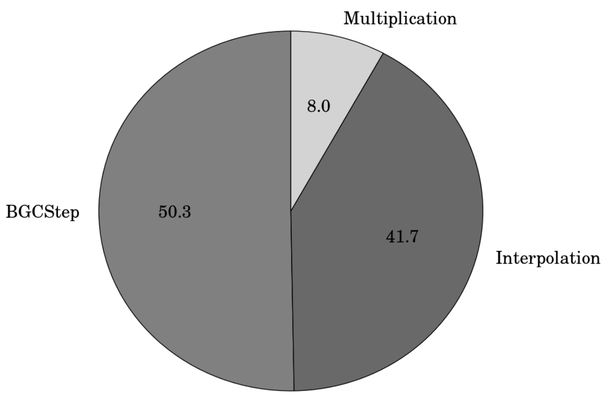
(A) Base POD100DEIM50

(B) Base POD300DEIM300

FIGURE A.6: Distribution of the computational time among main operations of the N-Model during the computation of a model year. The BGC-Step part indicates the time that is needed to evaluate the nonlinear function.



(A) Base POD100DEIM50



(B) Base POD300DEIM300

FIGURE A.7: Distribution of the computational time among main operations of the N-DOP-Model during the computation of a model year.The BGCStep part indicates the time that is needed to evaluate the nonlinear function.

A.4 Latin Hypercube samples

u_{LHC_0}	$\{0.0470, 140.86, 1.305, 17.078, 0.891\}$
u_{LHC_1}	$\{0.0254, 39.37, 0.710, 34.632, 0.766\}$
u_{LHC_2}	$\{0.0405, 175.83, 0.392, 19.928, 1.251\}$
u_{LHC_3}	$\{0.0450, 137.58, 1.154, 39.281, 1.066\}$
u_{LHC_4}	$\{0.0289, 113.77, 0.594, 26.857, 1.488\}$
u_{LHC_5}	$\{0.0215, 184.91, 0.294, 45.932, 1.321\}$
u_{LHC_6}	$\{0.0332, 58.38, 1.473, 24.204, 1.363\}$
u_{LHC_7}	$\{0.0373, 98.79, 1.052, 11.707, 1.133\}$
u_{LHC_8}	$\{0.0121, 63.83, 0.787, 46.016, 0.828\}$
u_{LHC_9}	$\{0.0162, 14.21, 0.933, 30.877, 0.959\}$

TABLE A.1: 10 Latin Hypercube samples used in Chapter 5.

A.5 Plots of parametrized models

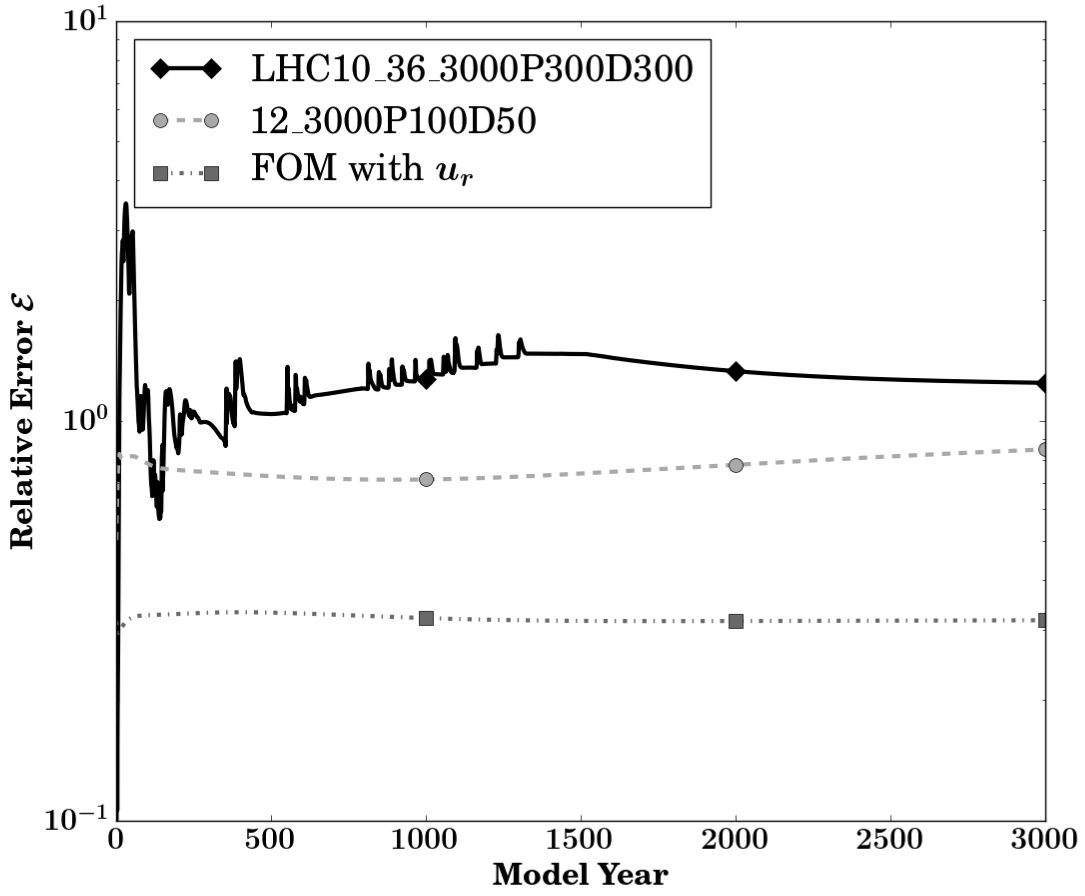


FIGURE A.8: Relative error between different model runs and the FOM solution generated with u_{LHC_0}. The black line is the relative error of the ROM generated with the ten Hypercube samples, run with u_{LHC_0}. The gray line with dots shows the relative error of the ROM out of Chapter 4 generated with the parameters set u_r, run with u_{LHC_0}. The dotted line with squares represents the relative error to the original solution of the FOM, run with parameter set u_r.

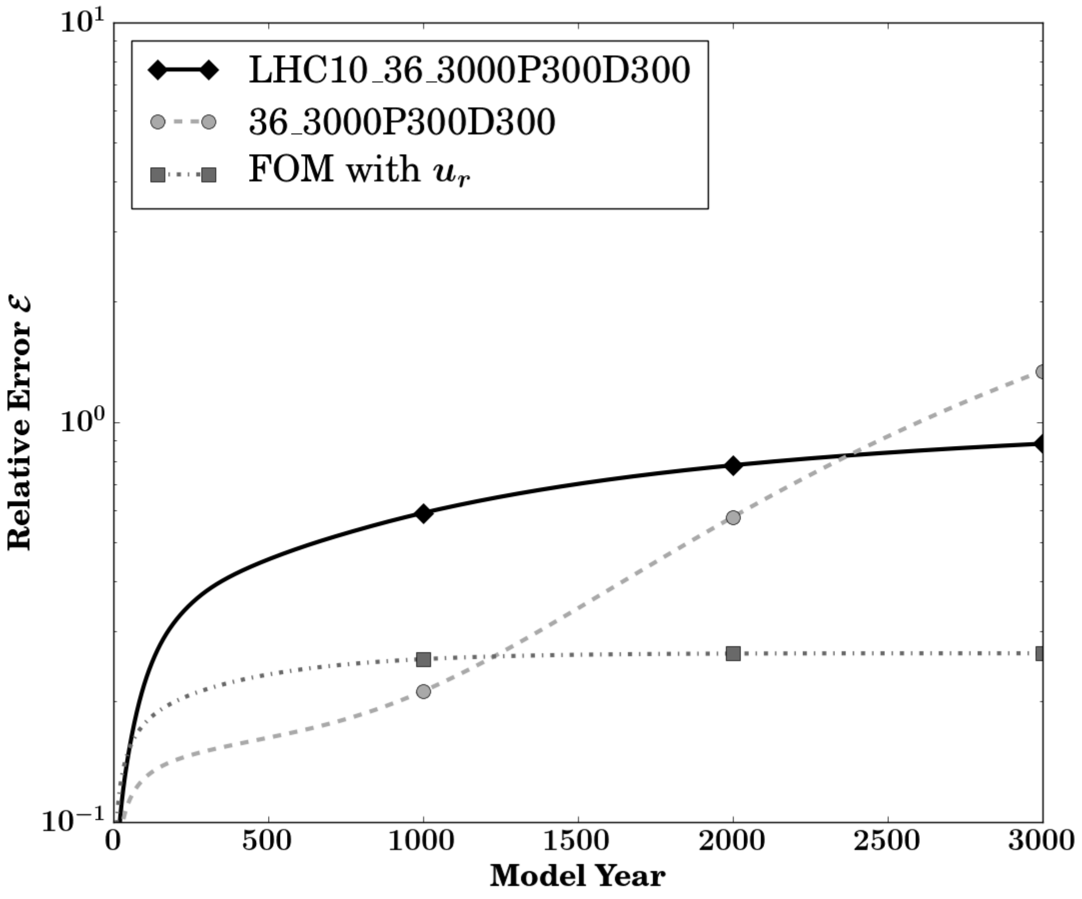


FIGURE A.9: Relative error between different model runs and the FOM solution generated with $u_r + 50\%$. The black line is the relative error of the ROM generated with the ten Hypercube samples, run with $u_r + 50\%$. The gray line with dots shows the relative error of the ROM 36_$3000P300D300$ out of Chapter 4 generated with the parameters set u_r, run with $u_r + 50\%$. The dotted line with squares represents the relative error to the original solution of the FOM, run with parameter set u_r.

Bibliography

[1] Wilhelmus H. A. Schilders, Henk A. van der Vorst, and Joost Rommes. *Model Order Reduction*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, p. 471.

[2] C. Naveena, V. N. Manjunath Aradhya, and S. K. Niranjan. "The Study of Different Similarity Measure Techniques in Recognition of Handwritten Characters". In: *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*. ICACCI '12. ACM, 2012, pp. 781–787.

[3] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition (Second Edition)*. Second Edition. Boston: Academic Press, 1990, p. 591.

[4] Zhendong Luo A et al. *An optimizing reduced order FDS for the tropical Pacific Ocean reduced gravity model*. 2006.

[5] Alexander Lozovskiy et al. "POD-based Model Reduction for Stabilized Finite Element Approximations of Shallow Water Flows". In: *J. Comput. Appl. Math.* 302.C (2016), pp. 50–70.

[6] Zhu Wang et al. "Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison". In: *Computer Methods in Applied Mechanics and Engineering* 237–240 (2012), pp. 10 –26.

[7] Gal Berkooz, Philip Holmes, and John L. Lumley. "The proper orthogonal decomposition in the analysis of turbulent flows". In: *Annual Rev. Fluid Mech* (1993), pp. 539–575.

[8] Zhendong Luo and Junqiang Gao. "A {POD} reduced-order finite difference time-domain extrapolating scheme for the 2D Maxwell equations in a lossy medium". In: *Journal of Mathematical Analysis and Applications* 444.1 (2016), pp. 433 –451.

[9] A.G. Buchan et al. "A {POD} reduced order model for resolving angular direction in neutron/photon transport problems". In: *Journal of Computational Physics* 296 (2015), pp. 138 –157.

[10] G. Kerschen et al. "The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview". In: *Nonlinear Dynamics* 41 (2005), pp. 147–169.

[11] P.M. Fitzsimons and C. Rui. “Determining low dimensional models of distributed systems”. In: *Advances in Robust and Nonlinear Control Systems* 53 (1993).

[12] Saifon Chaturantabut and Danny C. Sorensen. “Nonlinear Model Reduction via Discrete Empirical Interpolation”. In: *SIAM Journal on Scientific Computing* 32.5 (2010), pp. 2737–2764.

[13] S. Chaturantabut. “Nonlinear Model Reduction via Discrete Empirical Interpolation”. PhD thesis. Rice University, 2011.

[14] Maxime Barrault et al. “An ‘empirical interpolation’ method: application to efficient reduced-basis discretization of partial differential equations”. In: *Comptes Rendus Mathematique* 339.9 (2004), pp. 667 –672.

[15] S. Chaturantabut and D. Sorensen. “Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media”. In: *Mathematical and Computer Modelling of Dynamical Systems* 17 (2010), pp. 337–353.

[16] R. Ştefănescu and I.M. Navon. “POD/DEIM nonlinear model order reduction of an {ADI} implicit shallow water equations model”. In: *Journal of Computational Physics* 237 (2013), pp. 95 –114.

[17] Gabriel Dimitriu and Ionel M. Navon. “Application of a POD-DEIM Approach for Dimension Reduction of a Diffusive Predator-Prey System with Allee effect”. In: *In Lecture Notes in Computer Science, Large-Scale Scientific Computing, Lirkov et al.(Eds.), volume 42 of LNCS 8353*. Springer Verlag, 2014, pp. 373–381.

[18] D. Xiao et al. “Non-linear model reduction for the Navier–Stokes equations using residual {DEIM} method”. In: *Journal of Computational Physics* 263 (2014), pp. 1 – 18.

[19] J. Piwonski and T. Slawig. “The Idea and Concept of Metos3D: A Marine Ecosystem Toolkit for Optimization and Simulation in 3D”. In: (Oct. 2014).

[20] J. Piwonski and T. Slawig. “Metos3D: a marine ecosystem toolkit for optimization and simulation in 3-D – Simulation Package v0.2”. In: *Geoscientific Model Development Discussions* 8 (2015), pp. 4401–4451.

[21] Andrea Manzoni Alfio Quarteroni and Federico Negri. *Reduced Basis Method for Partial Differential Equations, An Introduction*. Springer International Publishing, 2016, p. 296.

[22] Daniel B. Szyld. “The many proofs of an identity on the norm of oblique projections”. In: *Numerical Algorithms* 42 (2006), pp. 309–323.

[23] Danny Sorensen Rich Lehoucq Kristi Maschhoff and Chao Yang. *ARPACK, Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. 1997. URL: http://www.caam.rice.edu/software/ARPACK/.

[24] Jose E. Roman Vicente Hernandez and Vicente Vidal. "SLEPc: A Scalable and Flexible Toolkit for the Solution of Eigenvalue Problems". In: *ACM Trans. Math. Software* 31.3 (2005), pp. 351–362.

[25] Samar Khatiwala, Martin Visbeck, and Mark A. Cane. "Accelerated simulation of passive tracers in ocean circulation models". In: *Ocean Modelling* 9.1 (2005), pp. 51 –69.

[26] Samar Khatiwala. "A computational framework for simulation of biogeochemical tracers in the ocean". In: *Global Biogeochemical Cycles* 21.3 (2007).

[27] I. Kriest, A. Oschlies, and S. Khatiwala. "Sensitivity analysis of simple global marine biogeochemical models". In: *Global Biogeochemical Cycles* 26.2 (2012).

[28] Eric Jones, Travis Oliphant, Pearu Peterson, et al. *SciPy: Open source scientific tools for Python*. [Online; accessed 2016-09-06]. 2001–. URL: http://www.scipy.org/.

[29] V. Hernández, J. E. Román, and A. Tomás. "A robust and efficient parallel SVD solver based on restarted Lanczos bidiagonalization". In: *Electron. Trans. Numer. Anal.* 31 (2008), pp. 68–85.

[30] Alfio Quarteroni and Gianluigi Rozza. *Reduced Order Methods for Modeling and Computational Reduction*. Springer International Publishing, 2014, p. 332.

[31] Jeffery Scott Stephanie Dutkiewicz Andrei P. Sokolov and Peter H. Stone. "A three-dimensional ocean-seaice-carbon cycle model and its coupling to a two-dimensional atmospheric model: Uses in climate change studies". In: vol. 122. MIT Joint Program on the Science and Policy of Global Change, 2005.

[32] J W Daniel et al. "Reorthogonalization and Stable Algorithms for Updating the Gram-Schmidt QR Factorization". In: 2007.