Tampere University

MIKKO LEHTIMÄKI

Model Order Reduction for Modeling the Brain

Tampere University Dissertations 817

Tampere University Dissertations 817

MIKKO LEHTIMÄKI

Model Order Reduction for Modeling the Brain

ACADEMIC DISSERTATION
To be presented, with the permission of
the Faculty of Medicine and Health Technology
of Tampere University,
for public discussion in the auditorium FA032
of the Festia building, Korkeakoulunkatu 8, Tampere,
on 16 June 2023, at 13 o'clock.

ACADEMIC DISSERTATION
Tampere University, Faculty of Medicine and Health Technology
Finland

Responsible supervisor and Custos
Docent
Marja-Leena Linne
Tampere University
Finland

Supervisor
Associate Professor
Lassi Paunonen
Tampere University
Finland

Pre-examiners
Associate Professor
Marc de Kamps
University of Leeds
United Kingdom

Senior Research Fellow
Arnd Roth
University College London
United Kingdom

Opponent
Dr. Jennifer S. Goldman, Ph.D.
Ernst Strüngmann Institute for Neuroscience
Germany

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

Cover design: Roihu Inc.

ISBN 978-952-03-2941-9 (print)
ISBN 978-952-03-2942-6 (pdf)
ISSN 2489-9860 (print)
ISSN 2490-0028 (pdf)
http://urn.fi/URN:ISBN:978-952-03-2942-6

Carbon dioxide emissions from printing Tampere University dissertations have been compensated.

PunaMusta Oy – Yliopistopaino
Joensuu 2023

PREFACE

This thesis was completed at Tampere University, Finland, in the Faculty of Medicine and Health Technology (MET). The original publications of this thesis, as well as the supervision, were a joint effort by the Computational Neuroscience Research Group, headed by Principal Investigator, Docent Marja-Leena Linne (MET), and the Systems Theory Research Group, lead by Associate Professor Lassi Paunonen (Faculty of Information Technology and Communication Sciences).

I and Marja-Leena began the work in this thesis at the Department of Signal Processing of Tampere University of Technology, where the Computational Neuroscience Research Group was based. After a fruitful meeting with Professor Emeritus Seppo Pohjolainen and his research group at the Department of Mathematics, we started a collaboration that eventually resulted in this thesis.

The collaboration offered an exciting environment for scientific work, and I'm privileged to have been taught by the two best mentors I could imagine. It has been ten years since I first attended your lectures, yet you both remain an inspiration.

I sincerely thank Prof. Dr. Sacha van Albada, who provided her time and wisdom as a member of the steering group for my doctoral studies. I'm grateful to the pre-examiners Associate Professor Marc de Kamps and Senior Research Fellow Arnd Roth, who gave thoughtful and detailed feedback to the thesis manuscript. Your advice improved the thesis.

The Computational Neuroscience Research Group has been a great place to learn and grow, with the nicest people, thanks to every member of the group! The TUNI doctoral school and the MET doctoral program provided a unique opportunity for studies in a cross-disciplinary environment, for which I am thankful.

I have been incredibly lucky to be part of the European Union's Human Brain Project, an invaluable source of knowledge and through which I made many amazing friends. I wish the best to all of you.

I would like to acknowledge the financial support from the Tampere University

doctoral school, as well as from the Academy of Finland, EU FET Flagship Human Brain Project, and The Finnish Foundation for Technology Promotion.

Dedicated to my family.

Tampere 22.5.2023

ABSTRACT

In this thesis, we study the use of Model Order Reduction (MOR) methods for accelerating and reducing the computational burden of brain simulations. Mathematical modeling and numerical simulations are the primary tools of computational neuroscience, a field that strives to understand the brain by combining data and theories. Due to the complexity of brain cells and the neuronal networks they form, computer simulations cannot consider neuronal networks in biologically realistic detail. We apply MOR methods to derive lightweight *reduced order models* and show that they can approximate models of neuronal networks. Reduced order models may thus enable more detailed and large-scale simulations of neuronal systems.

We selected several mathematical models that are used in neuronal network simulations, ranging from synaptic signaling to neuronal population models, to use as reduction targets in this thesis. We implemented the models and determined the mathematical requirements for applying MOR to each model. We then identified suitable MOR algorithms for each model and established efficient implementations of our selected methods. Finally, we evaluated the accuracy and speed of our reduced order models. Our studies apply MOR to model types that were not previously reduced using these methods, widening the possibilities for use of MOR in computational neuroscience and deep learning. In summary, the results of this thesis show that MOR can be an effective acceleration strategy for neuronal network models, making it a valuable tool for building large-scale simulations of the brain.

MOR methods have the advantage that the reduced model can be used to reconstruct the original detailed model, hence the reduction process does not discard variables or decrease morphological resolution. We identified the Proper Orthogonal Decomposition (POD) combined with Discrete Empirical Interpolation Method (DEIM) as the most suitable tool for reducing our selected models. Additionally, we implemented several recent advanced variants of these methods. The primary obstacle of applying MOR in neuroscience is the nonlinearity of neuronal models, and

POD-DEIM can account for that complexity. Extensions of the Balanced Truncation and Iterative Rational Krylov Approximation methods for nonlinear systems also show promise, but have stricter requirements than POD-DEIM with regards to the structure of the original model.

Excellent accuracy and acceleration were found when reducing a high-dimensional mean-field model of a neuronal network and chemical reactions in the synapse, using the POD-DEIM method. We also found that a biophysical network, which models action potentials through ionic currents, benefits from the use of adaptive MOR methods that update the reduced model during the model simulation phase. We further show that MOR can be integrated to deep learning networks and that MOR is an effective reduction strategy for convolutional networks, used for example in vision research.

Our results validate MOR as a powerful tool for accelerating simulations of nonlinear neuronal networks. Based on the original publications of this thesis, we can conclude that several models and model types of neuronal phenomena that were not previously reduced can be successfully accelerated using MOR methods. In the future, integrating MOR into brain simulation tools will enable faster development of models and extracting new knowledge from numerical studies through improved model efficiency, resolution and scale.

TIIVISTELMÄ

Tässä väitöskirjassa tutkimme Model Order Reduction (MOR) -menetelmien käyttöä aivosimulaatioiden vaatimien laskentaresurssien pienentämiseksi ja laskenta-ajan nopeuttamiseksi. Matemaattinen mallintaminen ja numeeriset menetelmät, kuten simulaatiot, ovat tärkeimpiä työkaluja laskennallisessa neurotieteessä, jossa pyritään ymmärtämään aivojen toimintaa dataa ja teoriaa yhdistämällä. Aivosolujen ja niiden muodostamien soluverkostojen monimutkaisuudesta johtuen tietokonesimulaatiot eivät voi sisältää kaikkia biologisesti realistisia yksityiskohtia. MOR-menetelmiä käyttäen johdamme *redusoituja malleja* ja näytämme, että niillä on mahdollista approksimoida hermosoluverkostomalleja. Redusoidut mallit saattavat mahdollistaa entistä tarkempien tai suuren mittakaavan hermosoluverkostojen simulaatiot.

Valitsimme tähän tutkimukseen redusoinnin kohteiksi useita neurotieteessä relevantteja matemaattisia malleja, alkaen synaptisesta viestinnästä aivojen populaatiotason malleihin. Simuloimme malleja numeerisesti ja määritimme matemaattiset vaatimukset MOR-menetelmien soveltamiseksi jokaiseen malliin. Seuraavaksi tunnistimme kullekin mallille sopivat MOR-algoritmit ja toteutimme valitsemamme menetelmät laskennallisesti tehokkaalla tavalla. Lopuksi arvioimme redusoitujen mallien tarkkuutta ja nopeutta. Tutkimuksemme soveltavat MOR-menetelmiä mallityyppeihin, joita ei ole aiemmin tutkittu kyseisillä menetelmillä, laajentaen mahdollisuuksia MORin käyttöön laskennallisessa neurotieteessä sekä myös koneoppimisessa. Tutkimuksemme osoittavat, että MOR voi olla tehokas nopeutusstrategia hermosoluverkostomalleille ja keinotekoisille neuroverkoille, mikä tekee siitä arvokkaan työkalun aivojen laskennallisessa tutkimuksessa.

MOR-menetelmät ovat hyödyllisiä, sillä redusoidun mallin perusteella on mahdollista rekonstruoida alkuperäinen malli. Redusointi ei poista mallista muuttujia tai heikennä sen morfologista resoluutiota. Tunnistimme Proper Orthogonal Decomposition (POD) -menetelmän yhdistettynä Discrete Empirical Interpolation Method (DEIM) -algoritmiin sopivaksi menetelmäksi valitsemiemme mallien redusointiin.

Lisäksi otimme käyttöön useita viimeaikaisia edistyneitä muunnelmia näistä menetelmistä. Ensisijainen este MOR-menetelmien soveltamiselle neurotieteessä on hermosolumallien epälineaarisuus. POD-DEIM -menetelmää voidaan käyttää myös epälineaaristen mallien redusointiin. Balanced Truncation ja Iterative Rational Krylovin Approximation -menetelmien muunnelmat epälineaaristen mallien approksimointiin ovat myös lupaavia, mutta niiden käyttö vaatii redusoitavalta mallilta enemmän matemaattisia ominaisuuksia verrattuna POD-DEIM -menetelmiin.

Saavutimme erinomaisen approksimaatiotarkkuuden ja nopeutuksen redusoimalla moniulotteista hermosolupopulaatiomallia ja synapsin kemiallisia reaktioita kuvaavaa mallia käyttämällä POD-DEIM -menetelmää. Biofysikaalisesti tarkan verkostomallin, joka kuvaa aktiopotentiaalin muodostumista ionivirtojen kautta, redusoinnin huomattiin hyötyvän simulaation aikana redusoitua mallia päivittävien MOR-menetelmien käytöstä. Osoitimme lisäksi, että MOR voidaan integroida syväoppimisverkkoihin ja että MOR on tehokas redusointistrategia konvoluutioverkkoihin, joita käytetään esimerkiksi näköhermoston tutkimuksessa.

Tuloksemme osoittavat, että MOR on tehokas työkalu epälineaaristen hermosoluverkostojen simulaatioiden nopeuttamiseen. Tämän väitöskirjan osajulkaisujen perusteella voimme todeta, että useita neurotieteellisesti relevantteja malleja ja mallityyppejä, joita ei ole aiemmin redusoitu, voidaan nopeuttaa käyttämällä MOR-menetelmiä. Tulevaisuudessa MOR-menetelmien integrointi aivosimulaatiotyökaluihin mahdollistaa mallien nopeamman kehittämisen ja uuden tiedon luomisen numeeristen simulaatioiden tehokkuutta, resoluutiota ja mittakaavaa parantamalla.

CONTENTS

ABBREVIATIONS

ADEIM	Adaptive Discrete Empirical Interpolation Method
AdEx	Adaptive Exponential Integrate-and-Fire
AMPA	α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid
BT	Balanced Truncation
CNS	Computational Neuroscience
DAPOD	Discrete Adaptive Proper Orthogonal Decomposition
DEIM	Discrete Empirical Interpolation Method
FN	FitzHugh-Nagumo
FP	Fokker-Planck
HH	Hodgkin-Huxley
HR	Hindmarsh-Rose
IRKA	Iterative Rational Krylov Algorithm
LDEIM	Localized Discrete Empirical Interpolation Method
MIMO	Multiple-Input Multiple-Output
MOR	Model Order Reduction
NMDA	N-methyl-D-aspartate
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
POD	Proper Orthogonal Decomposition
PR	Pinsky-Rinzel
QB	Quadratic-Bilinear

QDEIM	QR-based Discrete Empirical Interpolation Method
RMS	Root Mean Square
ROM	Reduced Order Model
SISO	Single-Input Single-Output
SVD	Singular Value Decomposition

ORIGINAL PUBLICATIONS

Publication I M. Lehtimäki, L. Paunonen, S. Pohjolainen, and M.-L. Linne, "Order reduction for a signaling pathway model of neuronal synaptic plasticity", *IFAC-PapersOnLine, 2017 20th International Federation of Automatic Control (IFAC) World Congress*, 2017, pp. 7687–7692. DOI: https://doi.org/10.1016/j.ifacol.2017.08.1143.

Publication II M. Lehtimäki, L. Paunonen, and M.-L. Linne, "Projection-based order reduction of a nonlinear biophysical neuronal network model", *Proceedings of the 2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 1–6. DOI: https://doi.org/10.1109/CDC40024.2019.9029510.

Publication III M. Lehtimäki, I. Seppälä, L. Paunonen, and M.-L. Linne, "Accelerated simulation of a neuronal population via mathematical model order reduction", *Proceedings of the 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2020, pp. 118–122. DOI: https://doi.org/10.1109/aicas48895.2020.9073844.

Publication IV M. Lehtimäki, L. Paunonen, and M.-L. Linne, "Accelerating Neural ODEs using model order reduction", *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2022. DOI: https://doi.org/10.1109/TNNLS.2022.3175757.

Author's contribution

The author of this thesis was the first author in each of the original publications of this thesis.

Publication I — First author. The author implemented the mathematical model as well as chose and implemented the Model Order Reduction method. The author analysed the results and drafted the initial version of the manuscript. Lassi Paunonen formulated the mathematical presentations in Chapter 3 (Methods) of the publication and wrote the chapter together with the author. Marja-Leena Linne directed the study and wrote the abstract together with the author. Seppo Pohjolainen, Lassi Paunonen and Marja-Leena Linne provided feedback in all phases of the study.

Publication II — First author. The author implemented the mathematical model that was reduced. The author chose and implemented the Model Order Reduction methods and performed analysis of the results. The author drafted the initial version of manuscript. Lassi Paunonen verified the mathematical presentations. Lassi Paunonen and Marja-Leena Linne directed the study, participated in interpreting the results and provided feedback on the manuscript.

Publication III — First author. The author chose and implemented the mathematical model and the Model Order Reduction methods. The author drafted the manuscript. Ippa Seppälä and the author wrote the segment on mean-field modeling theory together. Lassi Paunonen and Marja-Leena Linne directed the study, participated in interpreting the results and provided feedback on the manuscript.

Publication IV — First author. The author conceived the study as well as chose and implemented the deep learning models, Model Order Reduction methods and benchmarked acceleration methods from the literature. The author drafted the manuscript. Lassi Paunonen verified the mathematical presentations. Lassi Paunonen and Marja-Leena Linne directed the study, participated in interpreting the results and provided feedback on the manuscript as

well as result visualisations. The author prepared and submitted the code as open source.

1 INTRODUCTION

Neural tissue is characterized by heterogeneous cell types and a myriad of interaction mechanisms between these cells. Observing neurons or neuronal networks operating in the living brain is a grand challenge. While large volumes of data can be collected using a variety of methods, tools to extract the essence of the data are needed to form a unified view of the brain. In computational neuroscience, mathematical models and theories attempt to capture this diversity [1].

Numerical simulation of mathematical models is the principal method of analyzing large, nonlinear, multi-scale models of neural systems [2]. In computational neuroscience, the sheer number of cells and connections in the brain make simulations of the brain a challenge. Simulations of models of biologically realistic detail and scale may need more compute resources than current supercomputers can offer. In this thesis, we focus on improving the computational efficiency of brain simulations. The aim of the thesis was to discover and apply new acceleration methods to systems that are relevant in modeling neuronal networks. We show that approximation via mathematical Model Order Reduction (MOR) can accelerate simulations by identifying low-dimensional subspaces where *reduced* models can be simulated efficiently by lowering the number of equations in the models. During and after simulation, an approximation of the original model can be reconstructed. The approach we propose is not specific to any given model type. Moreover, MOR addresses the primary shortcoming of existing simplification methods, namely MOR does not compromise biological detail of models. We demonstrate MOR using a collection of neuronal network models and a synapse model that have not been reduced using MOR earlier.

In recent years, several large-scale modeling studies of the brain have been conducted. In [3], a cortical microcircuit of 31 000 morphologically detailed neurons is reconstructed using scarce experimental data from multiple sources. The model is able to reproduce many emergent phenomena from *in vivo* and *in vitro* experiments. In [4], a mathematical model was built using 2.5 million simple phenomenological

spiking neurons. Although phenomenological models trade biological details for simplicity, the model is able to describe several anatomical areas of the brain and can perform nine different functional tasks. In [5], a model of the visual cortex was developed by combining a total of 45 thousand morphologically detailed neurons and spiking neurons. The model was able to recreate *in vivo* responses to visual stimuli in mice. These studies can be considered state-of-the-art among the current multi-scale brain models.

As an example of the computational challenges of simulation studies in computational neuroscience, each of the large-scale brain simulation studies of [3]–[5] had to make compromises in the biological scope and realism of their modeling approaches, firstly since compute resources are insufficient for even simplified biophysical whole-brain simulations and secondly because data of the brain is still sparse. In [3] the authors emphasize that despite using supercomputers, it was only reasonable to study a small portion of a single region of a young rat brain. In [4] the authors write that many details, neurons and neuron types are missing from the large-scale model, and that the modeled brain areas only perform a part of their natural functions. Moreover, in [4], a future rise in computing power is expected to allow the model to perform more advanced tasks. In [5] it was necessary to omit biophysical properties from dendrites of the compartmental neurons due to boundaries set by computational resources. Furthermore, parameter optimization was restricted to two stimuli and was done in part manually as automatic approaches were computationally intractable. All the studies mentioned their ambitions for studying larger-scale models, once they become computationally feasible. However, it is acknowledged that new methods may be needed for enabling and understanding such simulations [6]. A class of such methods is explored in this thesis.

While simplified models of neuronal phenomena have been successful, more can be done to enable large-scale simulations of the brain. At present, we require supercomputers to simulate fractions of biophysically detailed neural networks. The whole brain level can be coarsely simulated with neuronal population-level simplified mean-field models, but single neuron level of detail is out of reach in these studies, not to even mention including detailed synaptic mechanisms or non-neuronal cells such as glia in the simulations. The lack of sufficient computational power also forbids parameter fitting, optimization, as well as perturbation and sensitivity studies that rely on repeated simulations of the system. Hence, new acceleration approaches

are needed to enable more efficient and comprehensive simulations.

The objectives of this thesis are to find out which MOR methods are applicable to models of neuronal networks in computational neuroscience, what are the main obstacles faced by MOR in computational neuroscience, how to implement MOR methods efficiently and how effective results simulation speed and accuracy-wise can be achieved by using MOR. All original publications of this thesis consider these objectives using different mathematical models of neuronal phenomena and a variety of MOR methods. The main hypothesis of this thesis is that MOR methods can accelerate nonlinear neuronal network models directly as well as other network components, such as detailed synapse models, individually. For example, it could be rationalized that neuronal networks have redundancies in the network structure and cellular components that MOR methods could take advantage of. These questions were studied in this thesis in the scope of model types that are relevant for constructing large-scale mammalian brain models in computational neuroscience.

MOR is a promising tool, since it can address weaknesses of existing simplification methods. The primary drawback of simplification methods is that they discard variables from the detailed models. On the other hand, MOR methods are able to reconstruct an approximation of the dynamics of the original complex model, despite simulating a reduced model with a smaller number of equations. The approximation includes the complete spatial resolution of the original model. Additionally, the reduced model retains the same input signals and output or readout values as the original model. MOR methods are not specific to any given model type, as will be shown in this thesis. Moreover, MOR methods can be applied directly to large-scale nonlinear models or to already simplified models, making them particularly flexible tools. Finally, software designed for creating reduced models using MOR methods is starting to mature. These properties make MOR methods an interesting and relevant topic for computational neuroscientists.

With regards to performance of MOR methods, we obtained the following results. In **Publication I** it was found that MOR methods can accelerate a chemical reaction equation based synapse model. The result is useful for network simulations that can in the future be extended with more detailed models of synapses. In **Publication II** we found that networks made of simplified nonlinear biophysical neuron models can be reduced using recent adaptive MOR methods. In **Publication III** we successfully reduced a high-dimensional nonlinear Fokker-Planck neuronal popula-

tion model with several orders of magnitude acceleration, showing that MOR methods can enable more detailed mean-field modeling with complete state probability distributions. Finally, In **Publication IV** we compressed models generated using a new class of deep learning networks, continuous-time Neural Ordinary Differential Equations (Neural ODEs) [7], by leveraging our result that MOR operations can be implemented as layers of artificial neural networks. For convolutional Neural ODEs, MOR methods achieved better results than benchmarked artificial neural network pruning methods from the literature. With recurrently connected Neural ODEs, we found that an advanced MOR method for nonlinear reduction was required to obtain competitive performance with benchmark methods.

The publications of this thesis focus on nonlinear models for which MOR is still an open theoretical question. Nonlinearity is perhaps the most restricting factor when it comes to choosing MOR methods, and only a handful of MOR methods are currently capable of reducing general nonlinear systems [8]–[10]. However, in computational neuroscience, nonlinear systems are the primary class of models, and it is important to have tools for reducing them. Other characteristics of models in neuroscience include multiple inputs and outputs of models as well as mechanisms such as resets and thresholds, which are discontinuous and hence set constraints to model reduction performance. Our results show that these challenges can be overcome with the proper choice and implementation of MOR methods.

MOR research as well as earlier applications of MOR in reducing neuron models have focused on systems that stem from discretizing partial differential equations (PDEs) [11], [12]. PDE models typically contain structure that results in very large systems and fine space resolution, and MOR methods have been successful in reducing these systems [13]. In addition to a PDE model, the studies in this thesis consider models that are assembled from small systems of or individual ordinary differential equations (ODEs) directly, a common approach to creating models in computational neuroscience, which however does not automatically result in systems with similar spatial structure as found in discretized PDEs. In the broad scope of model reduction, this is an unconventional application of MOR methods. In the field of neuroscience, the results in this thesis provide new knowledge of the efficacy of MOR methods for these model types.

The impact of MOR methods in the field of computational neuroscience is based on the simulation acceleration offered by MOR methods without compromising bi-

ological detail in models. We can now suggest that before resorting into developing novel neuroscience-specific or even model-specific simplification methods, MOR options should be exhausted first, since their generic nature and strong theoretical background supports them. The selection of applicable methods, adjustable approximation dimensionality, capability to preserve input-output mappings and spatial structure, *a priori* error bounds, active research community and available software implementations should encourage researchers to apply MOR for their specific simulation acceleration needs. Whereas earlier MOR results in neuroscience were obtained for cable-equation based branching single neurons, the original publications in this thesis specifically establish MOR as a valuable approach for accelerating a variety of models of neuronal network systems, taking into account the specific needs of the field of computational neuroscience.

In Chapter 2 we give an introduction to mathematical models and MOR and brief summary of earlier results from MOR in computational neuroscience, covering the previously employed MOR methods as well as reduced model types. We further summarize other, perhaps more traditional, model acceleration approaches, namely those that are based on model simplification. Chapter 3 details the aims of this thesis. In Chapter 4 we explain the mathematical models and MOR methods that were used in the original publications of this thesis. The results towards each aim of the thesis are presented in Chapter 5 and critically discussed in Chapter 6. Finally, Chapter 7 concludes the thesis.

2 BACKGROUND

There are two main motivations for complexity reduction of nonlinear computational models in neuroscience. One is accelerating numerical simulations of models [5], [14]. The other is mechanistic understanding of functional contributions of the components of the model [15]–[17].

Focusing on acceleration is important, because the compute time required for simulations of neural systems is prohibitive of large-scale computational studies, especially those that require repeated simulations of a system. Such studies include determining the effects of different initial values, model parameters and stimulus functions. Furthermore, accelerated models are more accessible than their high-dimensional versions, as it may be possible to study an accelerated system on a laptop, whereas previously a compute cluster was needed. Finally, accelerated models allow including a larger and more heterogeneous set of biological entities in mathematical models, since the price of simulating a single entity is lowered.

Acceleration is the primary purpose of mathematical Model Order Reduction (MOR) methods [13]. The basic premise of MOR is determining a low-dimensional space, where the dynamics of an original, high-dimensional and computationally expensive model can be simulated with improved efficiency. From simulations of this reduced model, an approximation of the dynamics of the original system is then reconstructed. In this manner, all variables, parameters, cellular elements and spatial details of the high-dimensional mathematical model can be included in the model. Complexity reduction can also be pursued using *simplified* models. MOR contrasts model simplification, where the approach is to simulate a cartoon model from which spatial or mechanistic detail has been discarded.

In this chapter, a brief review of the current state-of-the-art methods for MOR and model simplification in computational neuroscience is given. First, essential terms and equations that are used throughout the thesis are introduced, with regards to Model Order Reduction and mathematical modeling in general. Second,

prior applications of MOR in computational neuroscience are presented. After this summary, the reader has an understanding of what MOR methods have been used in computational neuroscience and which models have been reduced using MOR methods outside of the publications in this thesis. Finally, a brief review of model simplification approaches is given.

2.1 Overview of Model Order Reduction

In this section, we explain the background knowledge needed for applying MOR methods. We start with differential equation-based mathematical modeling, and then move to formulating reduced models.

2.1.1 Theory

In this thesis, we consider time-invariant general nonlinear ordinary differential equations (ODEs) in *state-space form*

$$\begin{aligned} E\frac{\mathrm{d}x(t)}{\mathrm{d}t} &= Ax(t) + f(x(t),t) + Bu(t), \\ y(t) &= g(x(t),u(t)), \end{aligned} \tag{2.1}$$

that describe the time evolution of an n-dimensional state $x(t) \in \mathbb{R}^n$. In this equation, linear terms of the model are in the *state matrix* $A \in \mathbb{R}^{n\times n}$, time-dependent inputs in vector $u(t) \in \mathbb{R}^p$, linear input coefficients in *input matrix* $B \in \mathbb{R}^{n\times p}$ and nonlinear functions in the vector $f(x(t),t) \in \mathbb{R}^n$. The outputs or *readouts* or *observables* in $y(t) \in \mathbb{R}^o$ are a function of the state and inputs but are not modeled with differential equations. It is common to have only linear readouts $y(t) = Cx(t) + Du(t)$, with the *output matrix* $C \in \mathbb{R}^{o\times n}$ and *feedforward matrix* $D \in \mathbb{R}^{o\times p}$. The system is time-invariant since parameters of the matrices and functions do not change with time. In the publications of this thesis, the *mass matrix* $E \in \mathbb{R}^{n\times n}$ is the identity matrix I_n, although in general it need not be so. Notice that if E had rows of zeros, the system would be a *differential algebraic* system of equation, for which the following theory does not automatically hold. In the case that $p = o = 1$, Eq. (2.1) is called a single-output-single-input (SISO) system, which may permit easier analytical study and model reduction. In the general case, the system is multiple-input-multiple-output (MIMO), where the number of inputs and outputs are not constrained.

The ODE system in Eq. (2.1) is nonlinear if $f(x(t), t) \neq 0$. The state-space representation is generic, and all ODEs can be written in such a form. Moreover, partial differential equations (PDEs), after discretizing space variables, fit the same description, typically resulting in very large groups of ODEs. Accordingly, MOR methods have been traditionally developed and validated with PDE applications or even specific PDE discretization schemes in mind [13].

In the general nonlinear case Eq. (2.1) cannot be analytically solved. Hence, to determine a specific solution, the state of the system at a desired time, numerical methods are used to propage the state $x(t)$ forward from an initial state, according to the ODE. In these initial value problems, the beginning state is decided first, and the solution of the equation is specific to that selection. Given $x(0)$, numerical simulation is then used to reach $x(t)$. The primary tool for simulation is an ODE solver, which takes small steps in time until the end time or state-based condition is reached. A reliable *fixed-step* solver is the fourth order Runge-Kutta method [18], [19], while adaptive solvers that take varying steps in time can also be used.

Model order reduction (MOR) methods in this thesis are *projection-based* and can be called *subspace projection methods*. We seek k-dimensional basis matrices $V, W \in \mathbb{R}^{n \times k}$ so that $V^*W = I_k$ and $k < n$, onto which the system in Eq. (2.1) is projected. This yields the reduced state variables $\tilde{x} = V^*x \in \mathbb{R}^k$ that evolve in a k-dimensional subspace. An approximation of the original variables based on the reduced state can then be computed via the transformation $x(t) \approx W\tilde{x}(t)$. How accurate the reduced model is and how well it preservers properties of the original model, such as stability [20], passivity [21] and positivity [22], are determined by the basis matrices V and W.

MOR is very established for linear systems. For linear systems, MOR methods can provide error bounds, stability guarantees and positivity results. Linear reduced models can be computed using a variety of methods, some of which use simulated or real-world data and some that derive reduced models directly from the system matrices. On the other hand, for nonlinear systems, the choice of reduction methods is narrower, and it is challenging to derive results such as error bounds. Additionally, some reduction methods can take advantage of SISO features. Generally, MIMO systems are common, and these are more challenging to reduce accurately.

To obtain a *reduced order model* (ROM) that describes time-evolution in the k-dimensional space, Petrov-Galerkin projections of the terms in Eq. (2.1) are com-

puted as

$$\begin{aligned}
\tilde{E} &= V^* E W \in \mathbb{R}^{k\times k},\\
\tilde{A} &= V^* A W \in \mathbb{R}^{k\times k},\\
\tilde{B} &= V^* B \in \mathbb{R}^{k\times p},\\
\tilde{C} &= C W \in \mathbb{R}^{o\times k},\\
\tilde{E}\frac{\mathrm{d}\tilde{x}(t)}{\mathrm{d}t} &= \tilde{A}\tilde{x}(t) + V^* f(W\tilde{x}(t)) + \tilde{B}u(t),\\
y &= \tilde{C}\tilde{x}(t) + Du(t),
\end{aligned} \tag{2.2}$$

which illustrates that matrices $\tilde{E}, \tilde{A}, \tilde{B}, \tilde{C}$ are constant in time and can be computed before solving the equation, in the so called *offline phase* of model reduction. If the mass matrix $E = I$, then $V^*EW = V^*W = I$, and hence E can be ignored in the reduction process. Now the number of variables and equations in the system has been reduced, and solving the reduced ODE is expected to be faster than numerically solving the original initial value problem, resulting in accelerated numerical simulation.

In the linear case of $\tilde{x}'(t) = \tilde{A}\tilde{x}(t) + \tilde{B}u(t)$ (where x' denotes time derivative), the reduced model would now be completely independent of the original dimension n. However, evaluating the nonlinear term $V^* f(W\tilde{x}(t))$ of Eq. (2.2) requires that the n-dimensional state $x(t) \approx W\tilde{x}(t)$ is reconstructed and each n functions evaluated. Additionally, the resulting vector of values from the nonlinear functions must be projected to the k-dimensional reduced state. This hampers the efficiency of simulating the reduced model. *Evaluating the nonlinear term efficiently is the primary challenge of present day MOR methods.*

Certain nonlinear systems can be written in *quadratic-bilinear* (QB) format. There, the nonlinear term of the state evolution equation is composed only of quadratic nonlinearities and input functions that are multiplied with the state vector. QB systems are written as

$$E\frac{\mathrm{d}x(t)}{\mathrm{d}t} = Ax(t) + H(x(t)\otimes x(t)) + \sum_{i=1}^{p} N_i x(t) u_i(t) + Bu(t), \tag{2.3}$$

where $H \in \mathbb{R}^{n\times n^2}$ is a matrix of coefficients for quadratic nonlinearities and $N_i \in \mathbb{R}^{n\times n}$ are coefficients of bilinear inputs. The symbol $\otimes$ denotes the Kronecker product. The matrices H and N can be transformed to the reduced space to yield a reduced

model

$$\begin{aligned}\tilde{H} &= V^* H(W \otimes W) \in \mathbb{R}^{k \times k^2} \\ \tilde{N}_i &= V^* N_i W \in \mathbb{R}^{k \times k} \\ \tilde{E}\frac{\mathrm{d}\tilde{x}(t)}{\mathrm{d}t} &= \tilde{A}\tilde{x}(t) + \tilde{H}(\tilde{x}(t) \otimes \tilde{x}(t)) + \sum_{i=1}^{p} \tilde{N}_i \tilde{x}(t) u_i(t) + \tilde{B}u(t),\end{aligned} \tag{2.4}$$

which does not depend anymore on the original dimension n, even in the nonlinear term, although the dependency on k is quadratic [23], [24]. Similarly, in case of polynomial nonlinearities, forms corresponding to the QB Eq. (2.3) and Eq. (2.4) can be derived for higher order polynomials as well, but the H matrix quickly grows to inconvenient dimensions [25].

2.1.2 Algorithms

There are three MOR method families that stand out from the literature. In the following, we summarize them. More details are given in Chapter 4.

Proper Orthogonal Decomposition (POD) methods are a family of *empirical* MOR methods, introduced in [26]. They require data of system dynamics from the original system. This *snapshot* data can be collected from sensors that measure a real system or by simulating the original model. In POD, there is only a single reduction basis $V = W$. It is obtained via the Singular Value Decomposition (SVD) of this snapshot data matrix. The Discrete Empirical Interpolation Method (DEIM) [27] is a recent method that applies a similar process in order to evaluate nonlinear vector-valued functions in a reduced space. POD and DEIM are remarkable because they can be applied to nonlinear models in a straightforward manner.

Balanced Truncation (BT) methods compute a reduced model directly from the system matrices [20]. BT is based on *reachability* and *observability* of the system. Reachability and observability can be measured with the *reachability gramian* and the *observability gramian*, respectively. BT is a two-step process, where the system is first *balanced* so that states that are difficult to observe become difficult to reach. Second, the balanced state matrices are truncated in order to obtain a reduced model that can approximate the most important input-output characteristics of the original system. Finding the balanced realization using the gramians require that two

Lyapunov equations are solved. These are

$$\begin{aligned} AP + PA^* + BB^* &= 0, \\ A^*Q + QA + C^*C &= 0, \end{aligned} \tag{2.5}$$

where P and Q are the reachability and observability gramians, respectively. Note that P depends on the input matrix B and Q depends on the output matrix C, a fact that closely couples BT to input-output behavior of the system. The following must be considered when computing the gramians. The Lyapunov equation can only be guaranteed to have a unique solution if the matrix A only has eigenvalues that have negative real parts. In this case, the system is asymptotically stable and solutions tend towards zero as $t \to \infty$ when no input is present. Furthermore, determining gramians through Lyapunov equations for nonlinear systems is an open question. These matters mean that the original BT method is not applicable to general nonlinear systems, although it has extensions for quadratic bilinear systems and empirical variants for nonlinear models. However, the BT method preserves stability in the reduced model and it can compute an *a priori* error bound for the approximation.

Moment Matching (MM) methods aim to approximate a high-dimensional system through the transfer function. From the state-space format of Eq. (2.1), for a linear system, we can compute the transfer function

$$H(s) = D + C(sI + A)^{-1}B, \tag{2.6}$$

where s denotes the Laplace transform and $x(0) = 0$ [28]. The transfer function offers an additional approach into studying linear systems. A low-dimensional system is sought so that the transfer function approximates that of the original system. The *moments* of a linear function correspond to the coefficients of the Laurent series expansion of that function. The MM reduced system will match the first k terms of the Laurent series expansion of the transfer function. Different methods use Laurent expansions around different points in the complex plane (with zero and infinity being typical choices). However, directly and accurately computing the moments of a function is numerically challenging. An MM approximation can be computed iteratively with pure matrix-vector multiplications, without explicitly computing moments, which is a way to make MM methods numerically robust. Algorithms that find such MM approximations are known as Krylov methods, the most famous

implementations being the Arnoldi and the Lanczos methods [29].

2.2 Mathematical modeling in computational neuroscience

Mathematical models of neurons typically characterize three functional regions from single neurons [30], [31]; the dendrites, the soma and the axon. The dendrites form a tree-like structure and their cell membrane contains several types of ion channels. Ion channels are the main source of nonlinear (active) dynamics in neuron models. Ionic currents through ion channels generate *action potentials*, where the membrane voltage at a spatial location rises rapidly. These action potentials propagate through the neuron to pass information along the cell. The dendrites also receive synaptic inputs, which connect neurons to others via chemical transmission. The soma and dendrites integrate dendritic currents. The soma is typically modeled individually due to its shape that makes electrical properties different from other parts of the neurons. There are ion channels on the somatic membrane as well. The soma connects the dendrites to the axon of the neuron. The axon is insulated through myelination and has ion channels only at dedicated locations, at the Nodes of Ranvier. The axon propagates membrane potential changes to presynaptic terminals, where connections are made through synapses. Neural cells may also connect to each other through gap junctions, which is a direct channel between the intracellular domains of two cells. These core components form the basis of mathematical single neuron models.

Single neuron models cannot and do not aim to explain the functioning of the whole brain. Instead, models of higher levels of organization are needed [32], [33]. Higher on the level of organization is the network level. Network models connect single neuron models via synapses and attempt to generate emergent phenomena that are not seen at or explained by single neuron models alone [15], [33]. The synapses itself can be described on many levels of detail, from simple currents [15] to phenomenological systems [34] to detailed molecular reaction networks [35]. The increase in complexity from the single neuron models is considerable, leading modelers to resort to simplified single neuron and synapse models when creating network models.

Earlier MOR and model simplification efforts in computational neuroscience, prior to this thesis, have focused on accelerating morphologically detailed models of single neurons. In the following, we introduce the *cable equation* that is used

to model membrane voltage propagation in single neurons. With regards to this thesis, the cable equation is of historical importance, and the original publications of this thesis focus on model types that have not been previously reduced using MOR. However, neuronal network models can use models that are derived from the cable equation, such as in [15].

The basis of morphologically detailed neuron modeling is that voltage propagation in dendrites and axons can be described by the one-dimensional cable equation

$$\tau \frac{\partial V(x,t)}{\partial t} = \lambda^2 \frac{\partial^2 V(x,t)}{\partial x^2} - f(V(x,t),x,t) + I_\mathrm{e}(x,t) r_\mathrm{m}, \tag{2.7}$$

where V is membrane voltage, t is time, $x \in [0, L]$ is distance along the cable of length L, $\tau = c_\mathrm{m} r_\mathrm{m}$ is the membrane time constant, c_m is membrane capacitance (capacitance per unit length), r_m is membrane resistance (resistance of the cell membrane per unit length), $\lambda = \sqrt{r_\mathrm{m}/r_\mathrm{l}}$ is the characteristic length constant and r_l is resistance along the neurite per unit of length [30]. $I_\mathrm{e}(x,t)$ accounts for currents injected to the cable. The term $f(V(x,t),x,t)$ contains nonlinear voltage sources at location x. In the passive (linear) neuron case, this includes commonly only the linear leak voltage $f(V(x,t),x,t) = V(x,t) - E_\mathrm{l}$ with E_l being the leak reversal potential. In the case of an active, nonlinear neuron, $f(V(x,t),x,t)$ can include effects such as of ion channels and gap junctions.

The nonlinear Hodgkin-Huxley (HH) equations for ionic currents are often used in detailed models [36] for explaining excitability of single neurons. The equations for the i-th ionic current $I_i(t)$ have the form

$$\begin{aligned} I_i(t) &= g_i(V(t) - E_i) \prod_{q=1}^{n} m_q^{p_q}, \\ \frac{\mathrm{d}m_q(t)}{\mathrm{d}t} &= a_q(V(t))(1 - m_q) - b_q(V(t)) m_q, \\ a_q(V(t)) &= \frac{A_{a_q} + B_{a_q} V(t)}{C_{a_q} + H_{a_q} e^{\frac{V(t)+D_{a_q}}{F_{a_q}}}}, \end{aligned} \tag{2.8}$$

with $b_q(V(t))$ having the same format as $a_q(V(t))$. Here, $V(t)$ is membrane voltage, g_i is the maximum conductance and E_i the reversal potential of this channel type, while m_q are probabilities of ion channel gate activations [37]. For example the sodium channel has subunits m_1 and m_2 (commonly named m and h) that correspond

to the activation and inactivation gates of the channel. The constants A, B, C, D, F, H are parameters that can be fit to experimental data in order to model different channel types [38, p. 293].

When there are no nonlinear voltage sources, the cable equation has an analytical solution that allows studying the effect of parameters on membrane voltage, or inferring parameters based on measured voltages. With suitable boundary conditions, several cables can be connected to form a dendritic tree, so that realistic morphologies can be studied. Typically, membrane voltage dynamics are studied with numerical simulation as an initial value problem. For simulations, the PDE form of the cable equation is discretized into a system of ODEs using finite difference methods. This results in *compartmental* models where each compartment corresponds to a tiny segment of a neuronal cable. Each compartment can be described as a resistor-capacitor electrical circuit. It is possible to define a unique set of ion channels as well as synaptic inputs and outputs for each compartment. A very fine discretization leads to computationally expensive simulations, while a crude discretization is more efficient but yields inaccurate results.

Many of the following MOR and simplification approaches employ a linearization strategy where nonlinear Hodgkin-Huxley dynamics are approximated first by *quasi-active* dynamics [39]. Linearization is not always ideal, since it tends to lose accuracy when compared to the original nonlinear model. However, it does allow using reduction or simplification methods that are intended for linear systems. The original publications of this thesis use methods that do not need linearization, and this is an important difference to most prior MOR and simplification studies in computational neuroscience.

2.3 MOR in computational neuroscience

Early applications of MOR ideas in neuroscience studied the efficient simulation of the cable equation. In [40], a reduced model was compared to the compartmental modeling approach for simulating voltage propagation in a passive dendrite under synaptic stimulation. The idea was to use a global discretization of a neuronal cable, instead of the typical finite difference discretization using local basis functions that is also the starting place for the compartmental approximation. The global basis used in [40] was based on an eigenfunction expansion, where the basis functions are de-

fined for the full spatial domain. To derive low-order models with just a few basis functions, which can here be likened to number of compartments, singular perturbation methods were applied. Singular perturbation gives an approximation of the eigenbasis using a finite number of basis vectors, which correspond to spatial discretization points. The model was then evaluated using a varying number of spatial locations, with a small number corresponding to a low-dimensional model, and high number approaching the true solution. Compared to a compartmental model, it was shown that the MOR approximation was both more accurate at an equivalent low number of spatial discretization points and converged to the correct solution faster. The cost of the method is more work for dealing with boundary conditions imposed by the soma and cable branch points as well as synapses. The same approach was later used to accelerate simulations of voltage propagation in myelinated axons [41]. In [42], [43], similar studies were conducted for passive and active dendrites using Chebyshev polynomials as the basis functions.

It is worth noting that these early studies did not attempt to reduce a fully developed model, such as a morphologically detailed neuron. Rather, their aim was to find a sufficient but small dimension for the discretized cable equation. A discretization of the PDE form of the cable equation is necessary when simulating voltage propagation in neurons, and analytical solutions are not available. Hence, the goals of the above studies were about finding an efficient numerical solution to the cable equation, although they did include many of the important properties of MOR methods, such as choosing a good set of basis functions. In this sense, the approach in these studies was different than in the original publications of this thesis, where we seek reduced models that are used to reconstruct an approximation of the original model.

A line of applications of MOR methods for reducing morphologically detailed neuron models started with [44]. In this work, Kellems et al. considered a morphologically detailed neurons which were finely discretized into compartments. The models featured linearized conductances around resting potentials at dendritic trees, although with this simplification only subthreshold events were captured. By subthreshold, the authors mean events that do not trigger action potentials in the dendritic tree, as capturing those dynamics would require complete nonlinear dynamics. The models were stimulated to random synaptic locations using low amplitude step currents and oscillating current input in order to stay in the subthreshold regime. Reduced models were then computed using two different methods, BT and IRKA.

In the study, BT was more accurate at equivalent reduced model dimensions than IRKA, but computing the matrices of the reduced model with IRKA was more efficient, since BT requires solving the two Lyapunov equations. The usefulness of reduced models was demonstrated by stimulating the system at increasingly distant dendritic locations, and accurate results were obtained. In [45], similar linear and linearized models were considered. The authors compared their moment matching method to IRKA and BT proposed in [44] and obtained improved results. Moreover, the method of [45] included a simulation of a network constructed from reduced cells.

The work in [44] additionally raised an issue with phenomenological models of neural dynamics, specifically membrane voltage reset conditions in integrate-and-fire models. Reset conditions cannot be evaluated directly in the low-dimensional space where reduced models are simulated, and projecting the system to the original space, checking reset conditions, and finally projecting the updated variable back to the reduced space creates computational burden. In [44] a simple approach for including thresholded variables was studied. The thresholded variable, membrane voltage at the soma, was not considered as a variable and hence was not projected to the low-dimensional, reduced space. Instead, soma voltage was taken as an output (corresponding to $y(t)$ in Eq. (2.1)) of the system. A spike was recorded after a threshold-crossing membrane potential and a holding period of constant voltage was used to record a hyperpolarization period for the output variable without actual simulation.

An advantage of MOR methods is that a complete approximation of the original, detailed model can be recovered. However, it is true that in the reduced form, and when simulated in the low-dimensional subspace, the model does not automatically have a biophysical interpretation. This matter was investigated in [46] using passive and linearized single neuron models with current and conductance-based stimulation. They were able to show that a reduction basis with a biological interpretation can be developed using moment matching. The reduced models were interpretable as a collection compartments with a corresponding resistor-capacitor circuit. However, the reduced system may have negative conductances, each compartment will have an extra current term, and compartments of the reduced system will be fully connected instead of representing a neuronal morphology.

In [12] Kellems et al. reduced a morphologically detailed neuron model, with non-

linear HH ion channels. This work can be considered state-of-the-art for MOR of single neurons to this day. The nonlinear model could be reduced without linearization by combining the Proper Orthogonal Decomposition (POD) [47] method with the Discrete Empirical Interpolation Method (DEIM) [27]. POD can be used to identify least-squares optimal subspaces for reduced order modeling, while DEIM is used for interpolating nonlinear functions. The combination is critical, since POD alone cannot find *nonlinear* reduced models that are completely independent of the original model dimension. With the help of DEIM, this can be achieved, as the interpolation can be efficiently computed in the POD subspace. The timing of the work was critical, since DEIM had just been published. The main drawback of the POD-DEIM method is that it is *empirical*, meaning that it requires data from simulations (or experiments) of the original high-dimensional system. While IRKA, BT and moment matching methods do not have this requirement, they are not as versatile as POD. Based on the insights of this work, we evaluated the POD-DEIM method for new model types in the original publications of this thesis.

The results from efficient partitioning of HH neurons [48], reduction of cells with nonlinear ion channels [12] and reduction of linearized dendrites [46] were combined in the work by Du et al. [11]. They considered a model of a locust collision detection neuron that has weakly excitable distal dendrites that produce graded rather than all-or-none potentials. These weak channels were linearized and it was assumed sufficient accuracy of the original dynamics would be retained. Weak channels were reduced with the structure preserving moment-matching method from [46]. One dendritic branch and the soma were modeled with active HH dynamics and reduced with the POD-DEIM method. Under injected current stimulation, good accuracy of membrane voltage at the soma was found. Most recently, linear models of voltage propagation along myelinated axons were reduced in [49], while retaining nonlinear dynamics at the nodes of Ranvier.

In summary, the previous work on applying MOR in computational neuroscience has considered single neuron models based on the cable equation. Most work has used linear or linearized dynamics. Nonlinear HH dynamics have been successfully reduced using the POD-DEIM method. In later sections, we show how the original publications of this theses extend these results by reducing network and synapse models, as well as using newer, advanced POD-DEIM methods.

2.4 Model simplification

In this section, we introduce the most recent and well-known simplification methods for models in computational neuroscience. These methods can be seen as alternatives to MOR methods since they attempt to solve a similar problem. We present simplification methods here in order to give the reader a wide picture of the history of simulation acceleration methods in computational neuroscience. Moreover, it is possible to reduce the following simplified models further using MOR methods, which means that MOR and simplification methods can also complement each other. We will first cover morphological simplification methods that aim to simplify the structure of single neurons, and move to useful phenomenological descriptions of single cells and action potential generation mechanisms. Finally, we discuss mean-field approaches which lead into efficient neuronal population models.

2.4.1 Simplification of neuronal morphology

Morphology simplification is very established in neuroscience, and new methods are actively developed. Two prevailing approaches to morphology simplification can be distinguished. Either the aim is to derive a simpler model of a specific cell type from a given species, or to develop a cell type agnostic simplification method. Cell type specific methods may produce more specific biological knowledge, but generalistic approaches are easier to automate and distribute in software packages.

The earliest morphology simplification method is based on the equivalent cylinder model (ECM) [50], [51]. The ECM started with Wilfrid Rall, who wanted to simplify the morphologies of branching dendritic models in order to analyse them mathematically. The purpose of an ECM is to exactly recover the membrane somatic membrane voltage after a postsynaptic potential, while using a single cable model instead of a detailed dendritic tree. Moreover, the ECM preserves the dendritic membrane area and electrotonic length, so that it can be used to make predictions of the properties of complicated morphologies.

It was found that under certain (idealized) conditions, an entire passive model of a branching dendritic tree can be exactly described by an ECM [50], [51]. The conditions are as follows. First, all terminal branches need to be at the same electrotonic distance (length of the cylinder divided by the space constant) from the soma.

Second, the terminal branches should use the same boundary conditions. Third, axial and membrane resistivities must be uniform in the dendritic branches. Finally, branching must follow the 3/2 power law (also known as $d^{3/2}$ rule) [30], which states that if the diameters d of dendrites k branching from parent segment j follow

$$d_j^{3/2} = \sum_k d_{jk}^{3/2}, \tag{2.9}$$

then the k branches can be collapsed to a single equivalent cylinder connected to the segment j. The new cylinder has the diameter d_j. This law must hold at all branches. If these rules are obeyed, an ECM for the dendritic tree can be computed, making interesting geometries accessible for analytical study.

To study more realistic models where the ECM did not apply, cables were divided into small *compartments*, electrical circuits connected in series [17]. In the first application of compartmental modeling, the continuous cable equations were approximated by a series of 10 compartments. This traded the expensive fine resolution finite difference discretization of the continuous model of Eq. (2.7) into a crude discretization where homogeneity of properties was assumed within compartments instead of within the entire cable. The resulting group of ordinary differential equations could be analysed numerically. In this manner, geometries breaking the idealized conditions above could be studied more efficiently. Additionally, it was possible to include excitatory and inhibitory synaptic currents as well as other nonlinear voltage-gated currents in specific dendritic and axonal locations [52], [53]. The tools for practical modeling of detailed morphologies had been created. To this day, simplification of neuronal morphology equates to removing compartments from compartmental models in a principled manner.

Similar methods to the ECM that collapse branching dendrite or axon into an approximately equivalent representation have been developed. These approaches are often referred to as equivalent cable, equivalent profile and equivalent dendrite models. Most notably, these methods relax or generalize the strict requirements of the 3/2 power law, for example so that the diameters of the collapsed neurites can vary within the neurite [54]–[57]. In [58] ECM methods are further extended so that active ion channels and nonlinear synaptic currents can be formally considered in the approximation, whereas earlier work implemented active membrane currents in a more heuristic manner. Further simplification approaches have been studied and approaches to simplify neuronal morphology without aiming for equivalent descrip-

tions have been developed [59]–[61]. However, no general process of deriving an equivalent model for arbitrary morphologies exists [62], [63].

Meanwhile, improved numerical methods for solving the compartmentalized [64] and continuous [65], [66] neuronal cable equation were developed. Some took advantage of look-up tables for evaluating the Hodgkin-Huxley equations, while others were developed for passive cables only. The method of Holmes [65] is based on the Laplace transformation of the cable equation and has complexity that only scales with the number of branching points of the neuron model. These methods enabled compartmental models with improved spatial resolution and made it possible to study both passive and active single neurons models relatively efficiently. Methods based on [64] are used in the NEURON simulator to this day [67]. More recently in [48], the authors partition the discretized HH cables into branches that are described with tridiagonal matrices, and dynamically control ODE solver accuracy at sites where membrane voltage is predicted to change in order to accelerate simulation, since now small time steps are only used at active sites. The method also facilitates parallel computing of simulations.

An interesting line of work began in [68], where dendritic branches of rat pyramidal and Purkinje cells were assigned scores based on their Strahler's order, a measure of branching complexity, originally developed for analysing hydrology networks such as rivers. In this analogy, high score indicates a functionally significant dendritic branch. To create passive neuron models of Purkinje cells, later in [69] dendritic branches were scored based on their Strahler's order. Those branches with a low Strahler's order were excluded from the morphological model. The surface area of the parent dendritic segment was correspondingly increased to compensate for the lost segments.

A general strategy for simplifying morphologically detailed models with active synaptic conductances was developed in [70]. These results demonstrated the method using hippocampal pyramidal neurons by comparing action potential shape and interspike interval distribution at the soma. The simplification is based on the experimental observation that (six week old rat hippocampal) pyramidal neurons have nine functional regions into which each branch can be clustered [71]. The clusters are then modeled with a variable number of compartments. Passive properties of the reduced model and scaling parameters for maximum synaptic conductances are derived analytically. Moreover, synapses are mapped to the reduced model by con-

sidering axial path resistance. While the process does not use parameter fitting, it does require the original model to be simulated so that synapses can be redistributed. An open question is then how to choose correct stimulation locations, frequencies and amplitudes for this purpose, and how well does the simplified model generalize outside this regime.

The method of [70] was further developed in [72]. By clustering dendrites using Strahler's order, instead of the experimentally identified functional areas of the earlier method, a more general simplification process was developed. The process assigns a Strahler's order o to each branch, with terminal branches receiving a low number. The low number branches are then merged so that two order o branches create a branch of order $o+1$, whereas joining an order o branch to an $o+1$ branch keeps the order at $o+1$. This is continues until low number branches have been merged away, resulting in a small number of functional groups which each are assigned a compartment. Moreover, less scaling parameters were needed than in the original work. The method can be used to simplify neurons using active synapses and arbitrarily distributed ionic currents. However, the effects of these channels in the reduced model are considered at the soma, and nonlinear dendritic membrane dynamics are not directly mapped to specific locations in the reduced model [72]. In summary, at their time the two papers by Marasco et al. [70], [72] pushed the state-of-the-art of model simplification to new morphologies and dynamics and remained as the most flexible method for almost a decade.

At the time writing this thesis, the state-of-the-art in creating simplified morphologies in an automatic, analytical manner is the work by Amsalem et al. [73]. The method is based on mapping stem dendrites of a detailed model to single cylindrical cables, while preserving specific membrane resistivity, capacitance, and axial resistivity of the detailed dendrites. Synapses and nonlinear ion channels are mapped to the simplified model and merged if their parameters are identical. The resulting model has less compartments, less synapses and less ion channels, while maintaining biologically realistic characteristics. Notably, all the mappings are computed analytically and no parameter fitting is used, which makes the method very fast to apply. Additionally, this work was the first to provide an easily usable automated software for morphological simplification. Even with this method, local dendritic events are hard to replicate, although the simplified models achieve good membrane voltage reproduction at the soma.

Another interesting collection of research started with the work by Wybo et al. [74]. They proposed an extreme simplification of the dendritic tree. The method analytically calculates the transfer function between a synapse and the soma along a passive dendrite, and replaces the given function, and replaces the given dendritic segment with this function. The end result resembles a point neuron, since all spatial structure has been removed. While this method of simplification allows nonlinear synaptic conductances, dendritic voltage propagation is assumed to be linear along a uniformly parameterized dendrite. This way, complicated dendritic trees can be treated efficiently. However, the method scales exponentially in complexity in the number of synapses in the model, hence it should be used in simulations where the number of incoming synaptic connections is low. The scaling limitation was overcome in later work by Wybo et al. [75]. If the synapses are located favourably, the complexity of the updated method scales only linearly. The downside is that the synapses still cannot be placed randomly. While the runtime of the simplified models was improved in the later work [75], the cost was increased initialization time.

In a further study by Wybo et al. [76], the transfer function of a passive dendrite, as derived in [74], [75], was used to characterize how synaptic inputs to any site on the dendritic tree affect the membrane voltage and conductance-based dynamics at other synapses. In comparison, the older work considered only the transfer-function between a given synapse and the soma. The focus was on modeling the electrical behavior of dendrites, while chemical properties were not accounted for. The method in [76] considers membrane voltage as a superposition of voltage changes generated at individual synapses, which affect membrane voltage across the full dendritic tree. This way, an approximation of impedances for modeling the total synaptic input response in the dendritic tree can be created. The method allows model creation from experimental data. While in [76] the method was not used for simplifying models, it can be used to determine approximate impedance functions in order to accelerate simulations.

The other state-of-the-art morphology simplification method was published by Wybo et al. [77], building on top of the results of their earlier work using the transfer function approximation of passive dendritic trees [74], [75], as well as a prior study of dendritic compartmentalization [76]. In this recent work, simplification is based on retaining original compartments at user-chosen sites in the dendrites, which is a new approach. Unlike the other state-of-the-art method [73] that is based on an-

alytically derived simplifications, the method in [77] relies dominantly on parameter fitting at the retained compartments. Essentially, nonlinear current propagation between distant dendritic compartments is simplified to a single coupling conductance. Additionally, synapses are moved to the closest remaining compartment and given a scaling factor. To compensate for these changes, at the modeled compartments the leak and coupling conductances, capacitances, maximal conductances of ion channels and channel reversal potentials are fitted to data from the original model so that input response trajectories and resting membrane voltage match the original morphologically detailed model. The authors note that the resulting fitted parameters, particularly reversal potentials, may be out of physiological range, but the response to synaptic input will be matched least-squares optimally. In summary, the method is able to create simplified models that keep the original ion channels and input regions of interest while being flexible with the level of simplification. The authors also provide open-source software for their method.

2.4.2 Simplification of neuronal dynamics

Several studies have created compartmental single neuron models with a low number of compartments and selected ion channels, with the aim of providing an efficient but general model that reproduces qualitatively or quantitatively a set of important neuronal dynamics. These models do not aim to be exact reconstructions of neurons. For example the Traub model [16] and the Pinsky-Rinzel (PR) model [15] are classic examples of this approach, with [78] being a more recent one. In these studies, the aim has not been on developing a simplification method, but rather a generally usable and efficient model that captures features of morphologically detailed models. For example, the PR model does not include inhibitory currents in its original formulation, but it does recreate synchronized population bursting phenomena.

To lower the computational burden of simulating action potential dynamics, simplified phenomenological mechanisms are commonly used in place of the computationally expensive HH formalism seen in Eq. (2.8). Such models are sometimes collectively called *spiking neuron models*. For example, the FitzHugh-Nagumo (FN)

model [79], [80] is

$$\begin{aligned}\frac{\mathrm{d}v(t)}{\mathrm{d}t} &= v(t) - \frac{v(t)^3}{3} - w(t) + I_\mathrm{e}(t),\\ \frac{\mathrm{d}w(t)}{\mathrm{d}t} &= a(v(t) + b - cw(t)),\end{aligned} \tag{2.10}$$

where $v(t)$ represents membrane voltage, $I_\mathrm{e}(t)$ is stimulus current into the cell (or compartment) and $w(t)$ is recovery variable that has no measurable biological meaning. Moreover, a, b, c are parameters that can be changed in order to obtain spiking with different frequency and amplitude. This model has fewer equations to solve than even the most basic HH model, hence it is more efficient to use as a membrane voltage model when elementary dynamics are enough. However, due to the simple formulation, this model cannot recreate bursting dynamics, where a neuron rapidly fires action potentials for a short while. The Hindmarsh-Rose (HR) model [81] implements spiking dynamics using three variables

$$\begin{aligned}\frac{\mathrm{d}x(t)}{\mathrm{d}t} &= y(t) - ax(t)^3 + bx(t)^2 - z(t) + I_\mathrm{e}(t),\\ \frac{\mathrm{d}y(t)}{\mathrm{d}t} &= c - dx(t)^2 - y(t),\\ \frac{\mathrm{d}z(t)}{\mathrm{d}t} &= r(s(x(t) - x_\mathrm{R}) - z),\end{aligned} \tag{2.11}$$

where $x(t)$ represents membrane voltage while $y(t)$ models fast ionic currents and $z(t)$ adaptation to spiking. $I_\mathrm{e}(t)$ is a stimulus current and a, b, c, r, s and x_R are parameters of the model. Compared to the FN model, with the addition of a third variable this system is able create bursting spike dynamics. Other examples of phenomenological neuron models include the Izhikevich model [14] and the adaptive exponential integrate-and-fire (AdEx) model [82], among others. For a review of spiking neuron models, see [83].

In order to simplify models of neuronal networks, a successful strategy has been to model statistical characteristics of single neuron or network activity instead of modeling the detailed activity of individual cells. These *mean-field models* exist in different levels of complexity, and an extensive review is found in [84] while early work on statistical models of neuronal firing can be found e.g. in [85]. The simplest ones replace network models with an average firing rate in a point-like mass, while more detailed ones include spatial activity profiles and probability distributions of vari-

ables. Mean-field models can be derived analytically from stochastic single-neuron models by making assumptions about population level features. For example, an infinite population size and homogeneous connectivity or an asynchronous-irregular action potential firing regime [84], [86] are common choices. When interested in stationary states of networks, and if single-neuron time trajectories do not need to be considered, mean-field models are effective tools.

One example of the more detailed mean-field descriptions can be derived using the Fokker-Planck (FP) equation. When considering an infinitely large population of neurons, the activity of each cell becomes uncorrelated of other cells. Then, the FP equation is a partial differential equation for the time evolution of the joint probability distribution of the variables of a given neuron model [84]. This limits the use of FP equations use as nodes in whole-brain simulations and prevents them being derived from detailed single-cell models, since each variable of the underlying single-cell model leads into an additional dimension in the FP model. In [87], the FP equation is developed for networks of neurons modeled with FN or HH neurons. Including synaptic conductances means that even with the simple two-variable FN neuron, the FP equation will have three dimensions where discretization is needed. Numerically solving the system derived from FN equations requires using compute clusters [87], [88]. Solving the HH neuron FP system from [87] has not been demonstrated, due to insufficient compute resources.

3 AIMS OF THE STUDY

The goal of the thesis was to evaluate the suitability and efficacy of Model Order Reduction (MOR) methods for accelerating numerical simulations of mathematical models that are of interest in computational neuroscience. We selected nonlinear models for which MOR had not been studied previously. The selected models described a biophysical neuronal network, a Fokker-Planck mean-field network and chemical reactions in the synapse. Additionally, two artificial neural networks were chosen for study. See Table 4.1 for a summary of these models. These systems were used to obtain results towards the following aims of this thesis:

1. Determine the requirements that models used in computational neuroscience set for Model Order Reduction methods.
2. Find Model Order Reduction methods that are applicable for reducing our selected nonlinear mathematical models of neural activity.
3. Assess the suitability of Model Order Reduction for artificial neural networks used in machine learning.
4. Study how the chosen Model Order Reduction methods can be implemented efficiently for reducing the selected models.
5. Evaluate the performance of the studied Model Order Reduction methods for accelerating simulations of neuronal network models and network components that are relevant in computational neuroscience and deep learning.

4 MATERIALS AND METHODS

In this chapter, we first define the mathematical models of neuronal systems used in the original publications of this thesis. Then, we explain the Model Order Reduction (MOR) methods used in the publications.

4.1 Models of neuronal systems

Table 4.1 summarizes the models studied in each of the original publications of this thesis. We studied these models and evaluated suitable MOR methods for each reduction task. We implemented each model in either Python [89] or Matlab [90]. After obtaining ground truth values of model dynamics or other metrics by simulating the chosen systems, we derived multiple reduced models using different methods, described below, and simulated the reduced models. We then compared the metrics and computation time between the original and reduced models in order to assess the performance of the reduced models.

Table 4.1 Summary of model types studied in each original publication of the thesis.

Article	Type	System	Linearity	Input-output
Publication I	Synaptic chemical reaction network	ODE	Nonlinear	MIMO
Publication II	Biophysical neuronal network	ODE	Nonlinear	SIMO
Publication III	Fokker-Planck mean-field	PDE	Nonlinear	SIMO
Publication IV	Convolutional artificial neural network	ODE	Nonlinear	(MI)MO
Publication IV	Antisymmetric artificial neural network	ODE	Nonlinear	SIMO

4.1.1 Synaptic chemical reaction network model

A model of chemical reactions in the synapse was reduced in **Publication I**. This system contains 44 ODEs and two time-dependent stimuli according to [35]. It describes the chemical reactions underlying synaptic plasticity in a single dendritic spine in the striatum of basal ganglia of a mouse. The system is illustrated in Figure 4.1. Panel A of Figure 4.1 shows an overview of the modeled reaction chains for synaptic plasticity. Panel B of Figure 4.1 displays how the morphology of the dendritic spine is considered. These measures are needed to compute concentrations in the model. This level of detail would be impossible to include in large-scale simulations of the brain, and accelerating the system would be valuable for network models.

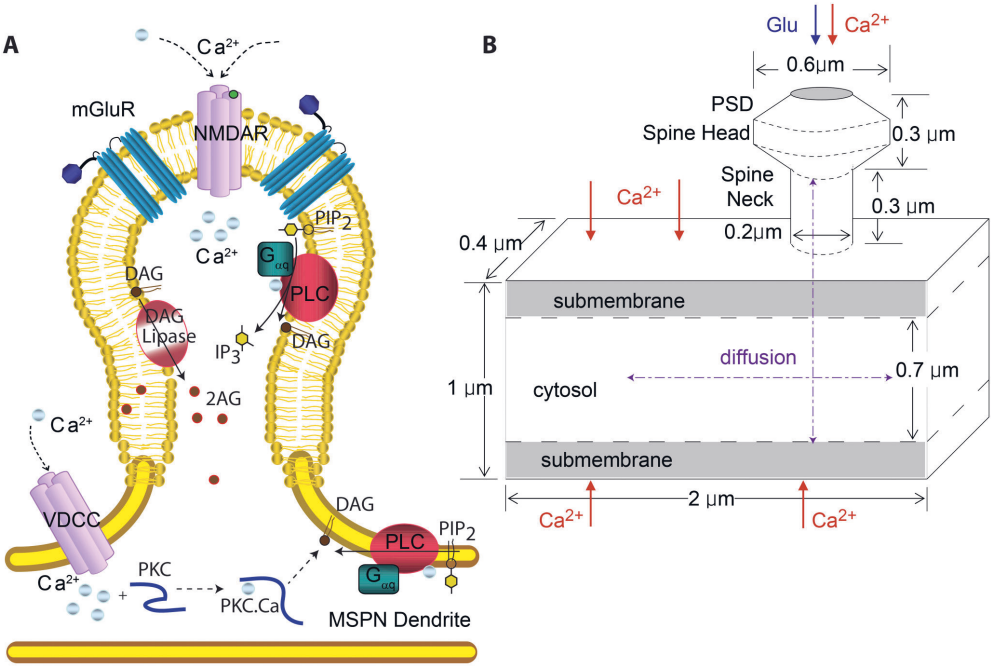

Figure 4.1 Description of the pathways included in the mathematical model of a neuronal spine studied in **Publication I**. A: Components of the model and their primary interactions. B: Morphology of the spine, used to estimate concentrations. Image from [35].

This model of synaptic dynamics has been validated against experimental data [35]. In state-space form, the system is quadratic-bilinear, as in Eq. (2.3). Quadratic nonlinearities stem from the chemical reaction equations that are modeled with Michaelis-Menten kinetics. Bilinearity rises from the time-dependent inputs of glu-

tamate and calcium that are multiplied with the state variables of the model. With these two inputs, we measure several molecular concentrations as outputs, indicating a MIMO system. We implement the model in Matlab [90] and simulate the system using ode15s, an adaptive ODE solver.

4.1.2 Biophysical neuronal network model

In **Publication II**, we studied a network of Pinsky-Rinzel (PS) neurons [15]. The PS neuron itself is a simplified system that models a neuron using two compartments, a dendritic compartment for receiving stimuli and a soma-axon compartment that sends input to other neurons. The compartments contain biophysical Hodgkin-Huxley (HH) currents as well as phenomenological thresholding mechanisms. The equation for the current in the somatic compartment is

$$\begin{aligned} C_{\mathrm{m}}\frac{\mathrm{d}V_{\mathrm{s}}(t)}{\mathrm{d}t} = &- I_{\mathrm{leak}}(V_{\mathrm{s}}(t)) - I_{\mathrm{Na}}(V_{\mathrm{s}}(t), m(t), h(t)) - I_{\text{K-DR}}(V_{\mathrm{s}}(t), n(t)) \\ &+ \frac{g_{\mathrm{c}}}{p}(V_{\mathrm{d}}(t) - V_{\mathrm{s}}(t)) + \frac{I_{\mathrm{s}}(t)}{p}, \end{aligned} \tag{4.1}$$

where $V_{\mathrm{s}}(t)$ is the membrane voltage in the soma.

For the current in the dendritic compartment the equation is

$$\begin{aligned} C_{\mathrm{m}}\frac{\mathrm{d}V_{\mathrm{d}}(t)}{\mathrm{d}t} = &- I_{\mathrm{leak}}(V_{\mathrm{d}}(t)) - I_{\mathrm{Ca}}(V_{\mathrm{d}}(t), s(t)) - I_{\text{K-AHP}}(V_{\mathrm{d}}(t), q(t)) \\ &- I_{\text{K-C}}(V_{\mathrm{d}}(t), Ca(t), c(t)) - \frac{I_{\mathrm{syn}}}{1-p} \\ &- I_{\mathrm{AMPA}}(V_{\mathrm{d}}(t), W(t)) - I_{\mathrm{NMDA}}(V_{\mathrm{d}}(t), S(t)) \\ &+ \frac{g_{\mathrm{c}}}{1-p}(V_{\mathrm{s}}(t) - V_{\mathrm{d}}(t)) + \frac{I_{\mathrm{d}}(t)}{1-p}, \end{aligned} \tag{4.2}$$

where $V_{\mathrm{d}}(t)$ is the membrane voltage of the dendritic compartment.

In Eq. (4.1) and Eq. (4.2), I_{leak} is a voltage-dependent leak current. Furthermore, I_{Na}, $I_{\text{K-DR}}$, I_{Ca}, $I_{\text{K-AHP}}$ and $I_{\text{K-C}}$ are voltage-dependent HH ionic currents and $h(t)$, $n(t)$, $s(t)$, $q(t)$ and $c(t)$ are their respective gating variables. C_{m} is membrane capacitance, g_{c} the eletrotonic coupling conductance between compartments, p is a ratio that indicates the size of the somatic compartment as a percentage of total cellular area and Ca is free intracellular calcium in the dendritic compartment. I_{s} and I_{d} are currents injected to the soma and dendritic compartments, respectively.

The synaptic currents I_{AMPA} and I_{NMDA} are used in network simulations. They have activation variables $W(t)$ and $S(t)$ that are functions of the somatic voltages of excitatory input cells. The NMDA activation of the i-th cell has the differential equation

$$\begin{aligned} \frac{\mathrm{d}S_i(t)}{\mathrm{d}t} &= \sum_{j=0}^{n} H(V_{\mathrm{s},j} - 10) - \frac{S_i}{150}, \\ H(x) &= \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{4.3}$$

where the sum is taken over all n cells that send synaptic input to the i-th cell. Additionally, $S_i(t)$ has a reset mechanism, so that if $S_i(t) > S_{max}$, then we set $S_i(t) = S_{max}$. For full details of the model equations and parameters refer to **Publication II** and [15].

When connecting a population of PS cells into a network, in **Publication II** the calcium conductance was drawn from the uniform distribution $\mathcal{U}_{[9,11]}$ in order to make the cells heterogeneous. We used 50 cells with each cell receiving synaptic stimuli from 20 randomly connected cells. The model allows injected current inputs into each cell and as outputs we take the somatic voltages of each cell, indicating a MIMO system. Our implementation follows the original paper [15] and injects a single current as an input. We simulate the system using a fourth order Runge-Kutta method [18], [19]. The PS network produces an important emergent phenomena, synchronized network bursting. Being able to reproduce this behavior using an efficient reduced model while maintaining an approximation of the original variables would be beneficial for using these networks in brain simulations. In **Publication II**, we qualitatively evaluated the capability of reduced models to reproduce the population bursting phenomenon.

4.1.3 Fokker-Planck mean-field model

The Fokker-Planck (FP) formalism for modeling neuronal networks provides the complete probability density function of the network state [84]. The state of the model is comprised of the variables of a single neuron model from which the population-level FP model is derived. In **Publication III**, we study the acceleration of an FP mean-field model of FitzHugh-Nagumo (FN) neurons with chemical synapses according to [87]. This FP model is a second-order partial differential equation with

three independent variables. It takes the form

$$
\begin{aligned}
&\partial_t p(t,V,W,Y) \\
&= \frac{1}{2}\sigma_{\mathrm{J}}^2 \bar{y}(t)^2 \frac{\partial^2}{\partial V^2}\left[(V-V_{\mathrm{rev}})^2 p(t,V,W,Y)\right] \\
&\quad + \frac{1}{2}\frac{\partial^2}{\partial Y^2}\left[\sigma_{\mathrm{Y}}^2(V,Y)p(t,V,W,Y)\right] \\
&\quad + \frac{1}{2}\sigma_{\mathrm{ext}}^2 \frac{\partial^2}{\partial V^2}\left[p(t,V,W,Y)\right] \\
&\quad - \frac{\partial}{\partial V}\left[\left(V - \frac{V^3}{3} - W + I_{\mathrm{ext}}(t) + \bar{J}(V-V_{\mathrm{rev}})\bar{y}(t)\right)p(t,V,W,Y)\right] \\
&\quad - \frac{\partial}{\partial W}\left[a(V+b-cW)p(t,V,W,Y)\right] \\
&\quad - \frac{\partial}{\partial Y}\left[\left(\alpha_r S(V)(1-Y) - \alpha_d Y\right)p(t,V,W,Y)\right],
\end{aligned}
\tag{4.4}
$$

where V and W correspond to the membrane voltage and recovery variable of the FN model, while the Y variable models excitatory synaptic currents. The joint probability density of these variables is given by $p(t,V,W,Y)$ as a function of time. The model can consider time-dependent bilinear stimuli through the term $I_{\mathrm{ext}}(t)$. The function $\bar{y}(t) = \iiint y p(t,v,w,y)\,dv\,dw\,dy$ depends on the current state of the density and when squared as in Eq. (4.4), introduces nonlinearity into the model. Other parameters are related to the single neuron and synaptic current model, as explained in [87].

As $I_{\mathrm{ext}}(t)$ is the only input, and as outputs we take the marginal distribution over Y, we get a single-input multiple-output (SIMO) system. For numerical simulations, we discretized the model using a fourth-order central difference scheme. The number of ODEs resulting from the discretization of Eq. (4.4) is $(n_V n_W n_Y)$ where n_i denotes the number of discretization points used in each dimension. We solve the system using a fourth-order Runge-Kutta method.

4.1.4 Artificial neural network models

Artificial Neural Networks (ANNs) are increasingly used as models for neuronal systems [91], [92]. Computationally these systems are demanding, and often they are executed using Graphics Processing Units (GPUs) to parallelize matrix multiplica-

tions. Recent results have made it possible to include ordinary differential equations as ANN building blocks [7]. These Neural ODEs bring ANNs closer to the models used in neuroscience, especially recurrent neural network models that have a long history in computational neuroscience. Hence, Neural ODEs may in the future act as a bridge for allowing the tools of deep learning to be more effectively applied in computational neuroscience. In **Publication IV** we show that Neural ODEs can be accelerated using MOR methods.

Typical *discrete* ANNs are assembled from layers of the form

$$x_{t+1} = f_t(x_t, \theta_t), \tag{4.5}$$

where x_t is the hidden state of the network at layer t and $f_t(x_t, \theta_t)$ defines a nonlinear transformation of the data with parameters θ_t to obtain the state at layer $t + 1$. In ANN *training*, the parameters θ are *learned* to make the network perform a task that minimizes a cost function. By viewing layers of an ANN as steps forward in time, Neural ODEs make it possible to include *continuous-time* differential equations as building blocks of ANNs. The data transformation applied by these layers is

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = f(x(t), t, \theta), \tag{4.6}$$

where $x(t)$ is the state of the network at time t and $f(x(t), t, \theta)$ defines a nonlinear function that maps the data $x(t)$ into $x(t + dt)$, with dt being very small. Such a block is said to be infinitely deep, since during training of the network and at the time of inference, we can propagate the data using ODE solvers for time $t = [0, T]$ of our choice, or until $x(t)$ reaches a desired state. In this format, the parameters θ are *tied*, being the same at every step. This makes Neural ODEs memory efficient since the number of unique parameters is typically lower than in deep discrete ANNs and additionally the algorithm for optimizing Neural ODEs requires less memory than the traditional backpropagation method [7].

In **Publication IV**, we compare our MOR approach to traditional ANN compression methods from the literature in two classification tasks, and implement a different Neural ODE model for each task. In the first task, the Neural ODE model implements convolutional transformations [93] as

$$\frac{\mathrm{d}}{\mathrm{d}x}(t)t = f(Ax(t) + b), \tag{4.7}$$

where A implements the convolutional operations in matrix form and b is a vector of constant *bias* terms applied to each neuron unit. The full state of the ODE block is taken as an output and propagated downstream in the neural network. If the bias vector b would apply linearly, this could be considered as a MIMO system, and now that the bias affects inside the nonlinear activation function $f(\cdot)$ we say this is a homogeneous system without inputs, hence we denote it (MI)MO in 4.1. This has implications for MOR, particularly in how POD-DEIM approximates the linear and nonlinear terms. We solve the ODE block of the convolutional Neural ODE using the fourth-order Runge-Kutta method.

In the second model, the Neural ODE model processes time-series data using an antisymmetric recurrent connection architecture [94]

$$\begin{aligned} A &= W - W^T - \gamma I, \\ \frac{\mathrm{d}}{\mathrm{d}x}(t)t &= \sigma(Ax(t) + b) + Zu(t), \end{aligned} \tag{4.8}$$

where A is an antisymmetric matrix of connection weights, $\theta = W, Z, b$ are trained parameters of the network, γ is a small positive parameter, σ is a hyperbolic tangent function and $u(t)$ is the input data into the network. We feed a single pixel of an input image to the ODE at a time, with $u(t)$, building a SIMO system. We solve the ODE block numerically using the forward Euler method.

All our ANN and Neural ODE models were implemented in Pytorch [95]. In each task and for each reduced model, we quantitatively evaluated the classification accuracy of the reduced model as a function of obtained acceleration.

4.2 Proper Orthogonal Decomposition

Proper Orthogonal Decomposition (POD) is a projection-based model reduction method that is applicable to general nonlinear MIMO systems. POD was introduced in [47], [96] and it resembles the Principal Component Analysis, a technique used for data analysis, and Kosambi-Karhunen-Loève [97], [98] theorem, a technique better known for analysing stochastic processes.

The POD method uses a Galerkin projection, where the projection basis is $V = W$ in Eq. (2.2), in place of a Petrov-Galerkin projection where $V \neq W$. The basis V can be calculated empirically using the method of *snapshots* (sometimes also called

strobes in the literature) [26], a flexible and efficient approach. The snapshots are states $\{x(t_1), \ldots, x(t_N)\} \subset \mathbb{R}$ of the system that are saved at discrete timesteps for further processing. Snapshots are collected either from real data that is generated by a process that the model describes, or by forward simulation of the original high-dimensional system. Moreover, the collection can be repeated over different initial values and model parameters to gather a representative set of states. These vectors are called snapshots, and together they form the snapshot matrix $X \in \mathbb{R}^{n \times s}$, where s can be smaller or greater than n. In case snapshot data from some variables is missing, a gappy POD procedure can be used [99], [100].

Finding the projection basis proceeds by computing the singular value decomposition (SVD) of the snapshots $\Phi\Sigma\Psi^T = X$, which from now on are assumed to be real-valued allowing us to ignore complex conjugates in matrix transposes. Here, columns of the matrices Φ, Ψ are called left and right singular vectors, respectively. Each has orthonormal columns, so that $\Phi^T\Phi = I$ and $\Psi^T\Psi = I$. The singular values $\Sigma = \mathrm{diag}(\sigma_0, \ldots \sigma_n)$ are the square roots of eigenvalues of X^*X. They are sorted into descending order and they are positive or zero by definition. Moreover, from the SVD of the snapshots, only n singular values need to be computed for model reduction purposes, corresponding to the original dimension of the system. To obtain a k-dimensional basis (or rank-k basis) for POD projection, we choose only the leading k left singular vectors from Φ and assemble the projection matrix $V = [\phi_0, \ldots, \phi_k]$. This choice of V given k is optimal as the least-squares reconstruction error between the snapshots and reconstructed reduced states is minimized [101]. The corresponding POD reduced matrices, state equation and output function are

$$\begin{aligned}
\tilde{E} &= V^T E V \in \mathbb{R}^{k \times k}, \\
\tilde{A} &= V^T A V \in \mathbb{R}^{k \times k}, \\
\tilde{B} &= V^T B \in \mathbb{R}^{k \times p}, \\
\tilde{C} &= C V \in \mathbb{R}^{o \times k}, \\
\tilde{E}\frac{\mathrm{d}\tilde{x}(t)}{\mathrm{d}t} &= \tilde{A}\tilde{x}(t) + V^T f(V\tilde{x}(t)) + \tilde{B}u(t), \\
y(t) &= \tilde{C}\tilde{x}(t) + Du(t),
\end{aligned} \tag{4.9}$$

which closely resembles Eq. (2.2).

The singular values are informative for choosing a suitable reduced dimension k. One interpretation is that their magnitude describes the energy captured by the

corresponding singular vector in Φ. To determine a suitable rank k for the reduced order model, one can look at the decay of the singular values. The decay of singular values of the snapshot matrices of a convolutional Neural ODE from **Publication IV** is illustrated in Figure 4.2. The figure shows the decay of singular values for POD and DEIM snapshots separately, on a logarithmic y-axis. In this case, the magnitude of the singular values decays exponentially, indicating that the snapshots can be approximated in a low-dimensional subspace.

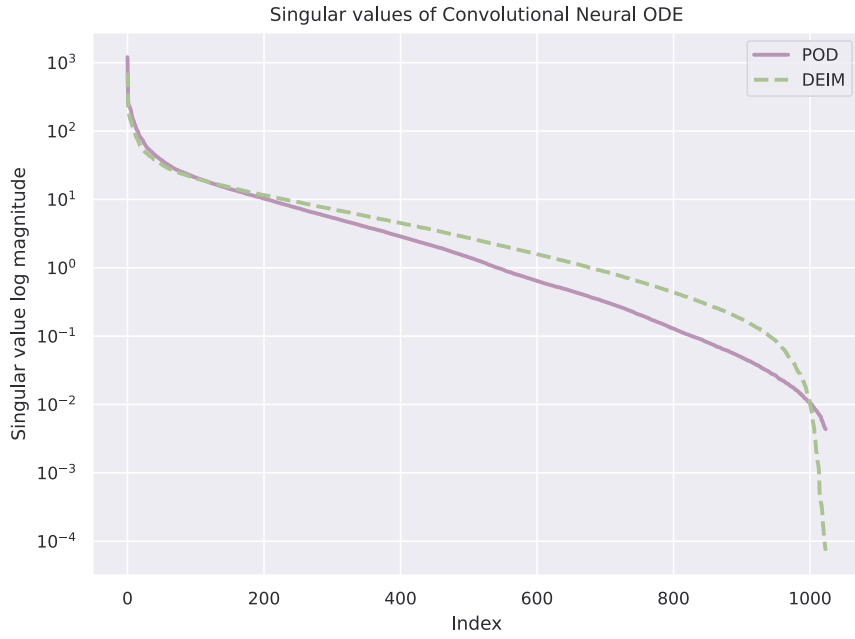

Figure 4.2 The decay of the singular values of the snapshot matrices from the full system (POD) and nonlinear function outputs (DEIM). Adapted from **Publication IV**.

By setting a percentage threshold $\varepsilon \in [0, 1]$, a cutoff point k is found so that $\sum_{i=0}^{k} \sigma_i / \sum_{i=0}^{\infty} \sigma_i > \varepsilon$ [26]. In practice, the decay of singular values is often exponential, and $k < n$ can be determined with this method. The SVD based basis selection can be summarized into

$$\min_{V \in \mathbb{R}^{n \times k}} \|X - VV^T X\|^2 = \sum_{i=k+1}^{\infty} \sigma_i \tag{4.10}$$

meaning that the least-squares optimal snapshot reconstruction error can be computed from the sum of truncated singular values [13], [101]. Naturally, how X is

chosen plays a large part in the overall quality of the reduced order model.

Some studies advocate for *centering*, in other words subtracting the mean of the snapshots from the snapshot set and using POD to approximate the dynamics around this mean [101], [102]. Likewise, some flow models have shown to benefit from snapshot *normalization* [103]. The purpose is to have the reduced model put equal weight into each phase of the flow. The approach has been used in situations where the reduction is targeting a PDE system with symmetries.

Snapshot collection is time intensive, making for the largest part of the offline cost of the POD method. While collecting snapshots at equal intervals of forward simulation of the original high-dimensional system may be a valid strategy, it could also result in redundant snapshots or a suboptimal projection basis, which wastes offline computation time. Additionally, in optimal control settings it is not desirable to use snapshots collected with arbitrary input functions. To this end, updating of snapshots and the computed projection basis is a topic of active research. Several strategies for updating the snapshots in optimal control settings, starting with arbitrary control or coarsely discretized PDEs have been proposed [104]–[106]. In [107]–[109], a method to determine an optimal snapshot set is developed. These approaches assume that new snapshot samples of the high-dimensional model can be generated. For iteratively constructing the reduction basis without access to new snapshot data, choosing or weighing the basis vectors based on expert knowledge has been explored in [110], [111].

The collected snapshots may require a lot of memory and it may not be feasible to store them in a single matrix. In this case, computing the SVD of the snapshots is challenging, and incremental methods that update the SVD as more snapshots become available or distributed methods that compute the SVD in a manner that does not use shared memory between processes are required. As an example of a distributed SVD algorithm that is not domain specific to model reduction, the method in [112] is useful for computing the POD basis. Moreover, the method in [113] may be used to incrementally build the SVD. For distributed basis computation in the POD framework, a method was first proposed in [114], with a more recent method algorithm given in [115]. A method capable of distributed and incremental basis computation was recently developed in [116].

As mentioned earlier, with general nonlinear systems the POD method does not decouple the reduced model from the original n-dimensional state, and thus cannot

guarantee accelerated simulations of nonlinear systems. In [25] a method of reducing quadratic-bilinear systems via POD is developed. It is shown that the usual POD basis V can be used to compute $\tilde{H} = V^T H(V \otimes V)$ and $\tilde{N}_i = V^T N_i V$ of Eq. (2.4), and all these matrices can be precomputed in the offline phase of the reduction. Now the QB reduced model no longer depends on the original dimension n.

Algorithm 1 Proper Orthogonal Decomposition

INPUT: Snapshots $X \in \mathbb{R}^{n \times s}$, matrices $E, A, B, H, \{N\}^p, C$, threshold $\varepsilon \in [0, 1]$, initial value $x(0)$

OUTPUT: $V, \tilde{A}, \tilde{B}, \tilde{H}, \{\tilde{N}\}^p, \tilde{C}, \tilde{x}(0)$

1: Compute $\Phi\Sigma\Psi^T = X$
2: determine k s.t. $\sum_{i=0}^{k} \sigma_i / \sum_{i=0}^{\infty} \sigma_i > \varepsilon$
3: $V = [\phi_1, \ldots, \phi_k]$
4: $\tilde{E} = V^T E V, \tilde{A} = V^T A V, \tilde{B} = V^T B, \tilde{H} = V^T H(V \otimes V), \{V^T N V\}^p, \tilde{C} = CV$
5: $\tilde{x}(0) = V^T x(0)$

Algorithm 1 summarizes the POD process. The inputs to the algorithm are the snapshot set X, matrices of the n-dimensional state-space system in Eq. (2.1) and the reduction threshold ε. As output, the algorithm gives the $k < n$-dimensional reduction basis as well as the reduced system matrices. Notice that is is possible to provide a large threshold, which will result in relatively large k, then further truncate the columns of V to derive lower dimensional reduced models without computing the expensive SVD of the snapshot matrix repeatedly. Overall, the POD method is flexible, since it imposes practically no limitations on the systems to be reduced. Computationally, the biggest burden is gathering snapshots of the high-dimensional system. The snapshots play a large role in the quality of the reduced model, and one of the challenges of POD reduced models is their accuracy at approximating states outside the snapshot set.

POD is the most flexible MOR method for reducing nonlinear systems, and forms the basis of the reduced model in all of the original publications of this thesis.

4.3 Advanced POD variants

The accuracy of the POD approximation will suffer if the reduced model enters a state which was not covered by the original set of snapshots. In these cases, adaptive POD methods can be used to update the basis vectors and rank during the model

simulation phase. Adaptive methods use a relatively small snapshot set, so that updates are efficient. The update will reduce the efficiency of the reduced model, but the gain in accuracy is expected to be worth the cost. In [117] an adaptive scheme was proposed, where the snapshot set was updated with new data during the simulation while old snapshots were removed. Later work included a modification where snapshots were eliminated based on importance scores [118]. However, these methods often lead into oscillating basis size and basis vectors that are not orthogonal, which reduces approximation quality.

In [119], the above issues are remedied. A new method, called Discrete Adaptive POD (DAPOD), considers both the novelty and importance of new snapshots simultaneously. It updates the reduction bases during the simulation phase while exhibiting less oscillations in basis size compared to earlier methods. In **Publication III**, we demonstrate the efficacy of DAPOD in reducing a biophysical neuronal network model.

4.4 Discrete Empirical Interpolation Method

An approach to reducing the complexity of the nonlinear term $f(x(t), t)$ in Eq. (2.1) was proposed in [27] under the name Discrete Empirical Interpolation Method (DEIM). When combined with POD, the POD-DEIM model reduction approach is state-of-the-art for deriving nonlinear reduced order models. The core idea is to separate the approximation of the state of the system, done by POD, from the approximation of the nonlinear function of the state equation, which will be the responsibility of DEIM. The two methods will work together seamlessly, so that the reduced model can be independent of the dimension of the full model. The DEIM method is closely based on the Empirical Interpolation Method from [120]. Other methods employing similar ideas are found in [100], [121], [122].

As the name implies, DEIM aims to interpolate the nonlinear n-component function using $m < n$ points in combination with approximation by projection in the n-dimensional basis. Since the projection is a linear operation, it can be efficiently computed directly in the POD subspace. DEIM finds an approximation

$$f(x(t), t) \approx U(P^T U)^{-1} \tilde{f}(x(t)) \tag{4.11}$$

where $U \in \mathbb{R}^{n \times m}$ is the DEIM projection basis and $P \in \mathbb{R}^{n \times m}$ is a sampling map

that indicates which indices of the original n-dimensional nonlinear function are used for interpolation. Here it is important to notice that $\tilde{f}(\cdot)$ (sometimes $P^T f(\cdot)$ in the literature [27]) indicates that the components dictated by P are extracted from the vector-valued nonlinear function. In other words, $f(x(t), t)$ is only evaluated at m locations. Moreover, the matrices U, P are chosen so that the approximation of $f(x(t), t)$ is exact at the m sampling points. In the case that the i-th component of $f(\cdot)$ depends on the i-th component of $x(t)$ alone, then only components $x_i(t)$ are needed in the evaluation. This requires a specific structure (one that can be exploited in artificial neural networks, as in **Publication IV**). The components of $f(\cdot)$ may depend on $x(t)$ in some other sparse manner as well. Under these assumptions, the nonlinear term can be approximated without needing the full n-dimensional state of the system.

The DEIM projection basis U is computed using the same process as the POD basis. However, snapshots are now collected from the nonlinear function only, in order to construct a matrix $F = [f(x(t_1), t), \ldots f(x(t_s), t)]$. The SVD of F is computed up to n singular values, $\Phi\Sigma\Psi^T = F$, and the m leading left singular vectors are chosen as the projection basis so that $U = [\phi_1, \ldots, \phi_m]$ (note that s and the SVD results are not to be confused with those in the POD). As in POD, a threshold value can be set to determine a suitable m. Once the DEIM basis has been computed, the interpolation indices can be determined.

Algorithm 2 finds interpolation indices φ and sampling matrix P in a greedy fashion. As input, the algorithm takes linearly independent vectors, which are commonly the columns of U. The first interpolation point is the index of the maximum absolute value of the first column of U, u_1. The following steps compute a residual r between the i-th basis vector and its current approximation using the $i-1$ determined interpolation points. Where the absolute error of the approximation is largest, a new interpolation point is inserted, until m have been chosen. In this manner, a sampling matrix $P = [e_{\varphi_1} \ldots e_{\varphi_m}]$ is constructed so that each column is a standard basis vector of $\in \mathbb{R}^n$ with the order of vectors determined by the indices in φ. The matrix P is a map that picks elements from the high-dimensional nonlinear vector into the m-dimensional interpolation space. The algorithm guarantees that $P^T U$ is nonsingular and its inverse is well defined [27].

To include the DEIM approximation of the nonlinear function into the POD

Algorithm 2 Discrete Empirical Interpolation Method

INPUT: $\{u_l\}_{l=1}^{m} \subset \mathbb{R}^n$ linearly independent
OUTPUT: $\vec{p} = [p_1, \ldots, p_m], P \in \mathbb{R}^{n \times m}$

1: $p_1 = \text{argmax}(|u_1|)$
2: $U = [u_1], P = [e_{p_1}], \vec{p} = [p_1]$
3: **for** l = 2 to m **do**
4: solve $(P^T U)c = P^T u_l$ for c
5: $p_l = \text{argmax}(|u_l - Uc|)$
6: $U \leftarrow [U\ u_l], P \leftarrow [P\ e_{p_l}], \vec{p} \leftarrow [\vec{p}\ p_l]$
7: **end for**

reduced model, the interpolation operation is projected onto the POD basis

$$M = V^T U (P^T U)^{-1} \tag{4.12}$$

which can be precomputed in the offline phase of the reduction. The dimensions are $M \in \mathbb{R}^{k \times m}$, which confirms that from m evaluated nonlinear functions, an interpolation approximation is given directly in the $k < n$ dimensional space. The combined POD-DEIM reduced model can then be written as

$$\tilde{E} \frac{\mathrm{d}\tilde{x}(t)}{\mathrm{d}t} = \tilde{A}\tilde{x}(t) + M\tilde{f}(\tilde{V}\tilde{x}(t,t)) + \tilde{B}u(t), \tag{4.13}$$

where $\tilde{V}\tilde{x}(t)$ indicates that we only pick the rows of V that reconstruct those variables of the original system that are needed by the nonlinear functions in $\tilde{f}$, and the system is completely independent of the original dimension n.

4.5 Advanced DEIM variants

An improvement to the DEIM algorithm was proposed in [123]. Their algorithm, Localized DEIM (LDEIM), uses an unsupervised clustering algorithm to group the snapshots. For each group, a DEIM basis and interpolation points are computed. Each group is assigned an indicator, which can be computed efficiently from the value of the system's nonlinear function. During simulation, the DEIM approximation given by the snapshot group closest to the most recent indicator value is used. We used the LDEIM algorithm in **Publication II** and **Publication III**.

A method that adapts the DEIM basis and interpolation points online was pro-

posed in [124]. This ADEIM, unlike LDEIM, can find a new DEIM basis in the online phase. The authors term this as nonlinear approximation. ADEIM has theoretically promising features for accurately approximating systems outside the collected snapshot space. This adaptivity comes at a relatively high computational cost, as we show in **Publication II**.

An improved *a priori* error bound for DEIM was given in [125] using the Q-DEIM algorithm. There, interpolation points are selected using a QR factorization instead of the original greedy iteration. This selection process can be used in place of the original method, for example in combination with LDEIM and ADEIM. In the publications of this thesis, we used Q-DEIM in **Publication III**.

An approach called oversampled DEIM (ODEIM) that uses regression instead of interpolation in the DEIM algorithm was given in [126]. In vanilla DEIM, the number of DEIM basis vectors and nonlinear sampling points are equal. The ODEIM algorithm uses more sampling points than basis vectors, and the authors show this leads to more stable reduced models. We employed this algorithm in **Publication IV**.

5 RESULTS

In this chapter, the main results towards each aim of Section 3 are given. We selected biologically relevant mathematical models that are of interest to computational neuroscientists, and the focus was in network models and components of neuronal networks. These models were not reduced earlier using model order reduction (MOR) methods, and hence the publications provide value to both fields, MOR and neuroscience. Our models are summarized in Table 4.1.

The primary results of the thesis are:

1. The dynamics of several nonlinear neuronal network model types and models were successfully reduced using MOR.

2. From the literature on model order reduction methods, Proper Orthogonal Decomposition (POD) with the Discrete Empirical Interpolation Method (DEIM) was found to be the most suitable MOR approach for reducing neuronal network models, as it can handle nonlinear multiple-input-multiple-output systems even with conditions such as membrane voltage resets.

3. Methods that adapt the reduction bases were shown to improve accuracy of reduced models in comparison to the original POD-DEIM method, at the cost of simulation time.

4. Our studies showed that reduced models can be computed efficiently with parallelization and distributed computation, and can benefit from graphics processing units.

5. MOR was incorporated in continuous-time deep learning models and reduced models with competitive classification accuracy versus speed-up trade-offs were obtained.

In the following, the results of the thesis are described in more detail.

5.1 Requirements for MOR in computational neuroscience

The first aim of the thesis was to determine which MOR methods are applicable for accelerating mathematical models that are relevant in computational neuroscience. The importance of this aim is based on the observation that MOR methods may have strict requirements about the properties of the mathematical model that is to be reduced, for example with regards to the type of nonlinearity in the system.

Mathematical models in neuroscience are typically nonlinear, as in all the original publications of this thesis. Models in the field also commonly have multiple inputs or multiple outputs (MIMO), as in **Publication I**. The models we implemented in **Publication II**, **Publication III** and **Publication IV** also has multiple outputs. In our studies, sources for nonlinearity in neural networks were the action potential mechanism, firing rate functions, chemical reactions and synaptic coupling, which cannot be described accurately using linear equations. Additionally, there are typically many outputs of interest, such as the membrane voltage of each cell in a network. Models may also contain multiple inputs, such as synaptic currents affecting different compartments, external currents injected into the cells and molecular perturbations of synapses. Nonlinearity as well as the number of inputs and outputs greatly affect the choice of available model reduction methods. As discussed in Chapter 2, MOR is most established for linear SISO systems. For nonlinear MIMO systems, the choice of suitable MOR methods is narrower, although most MOR methods have been extended to MIMO systems as well.

An important additional characteristic of models in computational neuroscience is that they are often assembled from a large number of small ODE systems that are discrete in space but continuous in time, as in **Publication I**, **Publication II** and **Publication IV**. First, not all MOR methods can be applied to this semi-discrete form of dynamical systems. Second, much of the MOR literature is focused on accelerating the simulation of discretized PDEs. The spatially discretized form of a PDE tends to contain a large number of ODEs in a rigid structure, depending on the PDE and the used discretization method, which MOR methods can then exploit. This case was studied in **Publication III**. However, generally in models in computational neuroscience, such structure is missing.

The choice of applicable MOR methods becomes clear when writing the system in the state-space format as in Eq. (2.1). In the general nonlinear case, the system will

have a nonlinear term $f(x(t), t)$, and MOR methods that can be applied to general nonlinear systems must be chosen. However, it is worth looking for additional structure in the model, in order to be able to apply a wider range of MOR methods, perhaps with different properties.

When the nonlinear term is defined by $H(x(t) \otimes x(t))$, the system has only quadratic nonlinearities, increasing the number of applicable MOR methods. When the inputs interact linearly with the state of the system via the term $\sum_{i=1}^{p} N_i x(t) u_i(t)$, methods that work for bilinear systems may be considered. There exist methods that can lift nonlinear systems to quadratic-bilinear form shown in Eq. (2.3), for example [24]. If the system can be written without the nonlinear and bilinear terms, then the system is linear and can be reduced with multiple methods. If the vectors $u(t)$ and $y(t)$ have only a single element, the system is SISO, again widening the choice of MOR methods.

Finally, it may be necessary to study the spectral properties of the state matrix A in Eq. (2.1). Spectal properties are deduced from the eigenvalues of the matrix. By investigating the eigenvalues of A we can understand the stability properties of the system. Some methods, particularly those based on balancing transformations (see Eq. (2.5)), may require that the eigenvalues of A have strictly negative real parts. With these considerations in mind, proper choice of MOR methods can be made.

5.2 Applicable MOR methods for the selected models

With the general requirements of MOR at hand, we made a thorough literature search and analysis of MOR methods. We then selected methods that fulfilled the general requirements and were suitable for the models chosen as reduction targets in this thesis. This section details these findings.

We identified the Proper Orthogonal Decomposition (POD) with the Discrete Empirical Interpolation Method (DEIM)[27] to be the most promising MOR methods for computational neuroscience. In computational neuroscience, POD-DEIM has been used earlier to accelerate the discretized form of the cable equation PDE. In POD-DEIM, POD finds a subspace for describing the dynamics of the system, while DEIM approximates the nonlinear part using interpolation. POD-DEIM creates reduced models that aim to approximate the state of the system, and the reduction process does not rely on the number of inputs or outputs, making the method ap-

plicable to both SISO and MIMO systems.

Furthermore, POD-DEIM allows us to control the dimensionality of the POD and DEIM approximations separately. As an example, this is illustrated in Figure 5.1, where reduced models are created using different POD and DEIM dimensions independent of each other. In the figure, the left column shows simulation time while the right column shows approximation error, computed as the root mean square (RMS) of the difference between the time series of the original and reduced models. POD dimension is indicated by the x-axis, while different colors show DEIM dimensions. The original model (from **Publication I**, here using a shorter simulation time) has 44 equations. In this case, the simulation time is affected linearly by both the POD and DEIM dimensions. The approximation error is more challenging to quantify; here it appears that at POD or DEIM dimension 10 the approximation quality deteriorates, while POD dimensions greater than 15 affect the accuracy quite linearly. Analyses like this example are useful for evaluating the suitability of POD-DEIM and determining useful approximation dimensions.

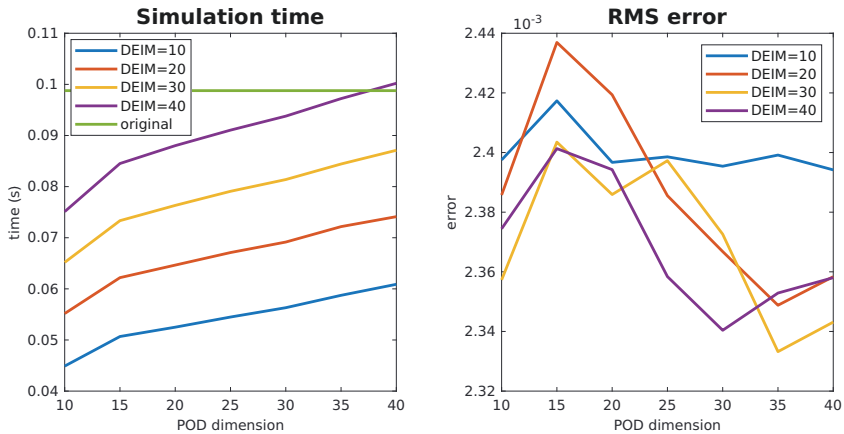

Figure 5.1 The effect of different POD and DEIM dimension combinations on the reduced model. Left column shows simulation time, right column shows root mean square error of the approximation. POD dimension changes according to the x-axis, while DEIM dimension is indicated with different colors. Adapted from **Publication I**.

POD-DEIM is used in all the publications in this thesis. Additionally, the publications in this thesis study the efficacy of advanced POD-DEIM methods that have not been used in neuroscience earlier. Specifically, the Discrete Adaptive POD (DAPOD) [119], Localized DEIM (LDEIM) [123], Adaptive DEIM (ADEIM) [124], Q-DEIM [125] and Oversampled DEIM (ODEIM) [126] were evaluated in the pub-

lications of this thesis.

Other potentially useful methods for nonlinear reduction were also found. For example, the Gauss-Newton with Approximative Tensors (GNAT) method [122], [127] which works on systems that are discrete in space and time. Earlier methods that attempt to reduce nonlinear systems include empirical gramians [128], Gappy POD [100], Missing Point Estimation [121] and Best Point Interpolation method [129]. However, the DEIM algorithm is the most generally applicable out of these methods. It scales to large systems unlike gramian-based approaches, provides a way to decouple the approximation of linear and nonlinear terms, and at the time of starting this study, it also had a number of advanced variants developed. Notably, all these methods, including POD-DEIM, are empirical, in other words they require data of the system behavior for reduction.

In unpublished work (Appendix A), we evaluated the quadratic-bilinear (QB) versions of Balanced Truncation (BT) [130] and Iterative Rational Krylov Approximation (IRKA) [10] methods for accelerating a network of Hindmarsh-Rose (HR) neurons connected with phenomenological Tsodyks-Markram (TM) synapses. This system is nonlinear and has multiple inputs and outputs. BT and IRKA are interesting since they differ from POD-DEIM methods in that the reduced model is derived directly from system matrices of the full model and no empirical snapshot collection phase is required. Both methods are applicable to MIMO systems. However, The HR+TM model is nonlinear and hence neither BT or IRKA can be used to directly reduce it. In Appendix A, we have derived a quadratic bilinear *lifted* form of this model through a lifting transformation. The lifted model has state variables that are equivalent to the original model, while also having auxiliary variables modeled with differential equations. Hence, the dimensionality of the lifted model is higher than the original model, but it contains a purely quadratic-bilinear structure that can be taken advantage of by model reduction algorithms such as the advanced BT and IRKA variations we used. While the lifting transformation is not unique and finding a QB representation is not possible for every nonlinear system, we expect this approach to find many uses in computational neuroscience in the future.

5.3 Efficient implementation of the MOR procedure

As the third aim the thesis, we wanted to determine how each step of the model order reduction workflow can be implemented efficiently. The reduced model is computed in the offline-stage of model reduction. The cost of this step, which can involve multiple algorithms, can be significant. The cost of the offline-stage may determine if the reduction method is feasible for applications that require repeated computation of the reduced model, such as parameter optimization studies.

An efficient implementation of MOR methods should account for both offline costs where the reduced model is computed, and online costs where the reduced model is simulated. On the one hand, when we wish to use a reduced model for simulation studies where only the initial conditions and inputs to the system are varied, it is enough to compute a reduced model once. An example of such a situation could be that many copies of the reduced model are deployed in a large or multi-scale simulation setting, modeling nodes in a large-scale brain simulation. For this purpose, a computationally heavy offline phase may be allowed. On the other hand, if a ROM is used to study the parameter sensitivity of a system with a set of parameters, such as varying ion channel conductances, steps of the offline phase may have to be executed repeatedly, in order to derive a reduced model for each permutation of parameters. In this case, the selection of MOR method greatly impacts the cost of the offline phase.

The POD-DEIM method is an empirical method that needs samples of the dynamics of high-dimensional original model that is to be reduced. To obtain reduced models that perform well in different initial value and parameter settings, such snapshots should be available when computing the bases and interpolation points for POD and DEIM. This requires several simulations of the high-dimensional model. Although the snapshot collection is a time-consuming process, snapshots of different initial conditions, parameters and input functions can be collected efficiently in parallel, as done for initial conditions and time-dependent inputs in **Publication IV** using batch computing. The snapshot collection step is *embarrassingly parallel* and no shared memory between processes is required, thus software implementations for parallelizing the data collection readily exist (for example through Message Passing Interface (MPI) or tensor algebra as done in **Publication IV** using Pytorch).

The next step in the POD-DEIM algorithm is singular value decomposition

(SVD) over all the snapshot data. This step will find the basis vectors onto which the high-dimensional model is projected for reduction. This operation is first and foremost very memory intensive, since a naive implementation will attempt to load all snapshots in memory for decomposing with SVD. However, algorithms for distributed [112], [114], [115] and adaptive [113] SVD do exist and should be leveraged, as we demonstrate in **Publication IV**. In addition, it may be possible to remove redundant snapshots and consider coarser sampling to alleviate memory requirements in the SVD step. In the publications of this thesis, it was never necessary to save snapshots at every discrete simulation timestep to obtain good results. Moreover, methods like Discrete Adaptive POD [119] are designed to work with a relatively small initial snapshot set. Once the SVD over the snapshots has been done, it is possible to create reduced models with different parameters using the same basis. This is due to the fact that the projection basis is not computed from the system matrix, which changes when model parameters are changed. Hence POD-DEIM is efficient in parameter optimization and sensitivity analysis studies.

For Balanced Truncation methods, solving the Lyapunov equations of Eq. (2.5) in order to obtain the reduction basis is the heaviest part of the offline phase. This phase is computationally more demanding than taking the SVD of the snapshots in the POD-DEIM algorithm, which limits the usability of BT methods for extremely large systems. Moreover, the solution of the Lyapunov equations depends on the parameterization of the high-dimensional model that we wish to reduce, hence this step must be repeated if the parameters change. An efficient and numerically robust implementation of a Lyapunov equation solver will not attempt to find the full gramian matrices X, but instead solves directly a low-rank Cholesky factorization $ZZ^* = X$, never forming the full matrix X [131]. The complete BT algorithm can then be computed using the Cholesky factors of controllability and observability gramians [132], even in the nonlinear (quadratic-bilinear) BT method [130].

Continuous-time recurrent neural networks, with simple nonlinearities, can be implemented on deep learning frameworks such as Python's Pytorch or Julia's DiffEqFlux. Using these libraries, the networks can then be efficiently simulated on graphics processing units (GPUs). In **Publication IV** we show how the model reduction workflow can utilize existing computation tools of deep learning. Specifically, we showed that the reduced model can be both computed and simulated using deep learning software. Moreover, we determined that all steps of the POD-DEIM

algorithm can also be implemented on GPUs, greatly streamlining model reduction of artificial neural networks.

Reset conditions, which are seen for example in phenomenological neuroscience models, introduce complexity to the model reduction process. In principle, it is possible to ignore the reset condition and attempt to derive a low-order model using data-driven model such as POD-DEIM, but the accuracy of such a reduction would be poor [133]. An additional transformation of the reduced model to the original high-dimensional space and back to the low-dimensional space is needed at every timestep to accurately implement reset conditions in the reduced model, as we did in **Publication II**. This is a slowing factor for the simulation time of of the reduced model. In the MOR literature, these systems are researched as *jump systems* or *hybrid systems* [134]. However, the reset moment can be used to ensure that other variables have stayed in a biologically meaningful range, at a low additional computational cost. For example, at the same time as reset conditions are set, it could be checked that variables describing concentrations have remained strictly non-negative. This can improve accuracy of the reduced model with minimal further computational cost.

Since the implementation and simulation of state-space models and reduced order models uses software for linear algebra (such as ScaLAPACK or MAGMA), the MOR approach is scalable to large-scale systems [13], [125]. This means that reduced models itself can leverage parallel computing, distributed computing clusters and graphics processing units. Moreover, the implementation of MOR is not dependent on any specific hardware, which contributes to the flexibility of MOR methods.

5.4 Integration of MOR in artificial neural network models

Machine learning models, such as deep neural networks, are increasingly used as model systems in computational neuroscience. These models are computationally demanding. In this aim, we wanted to find if MOR can be performed within the machine learning workflow to accelerate deep neural networks, and hence facilitate their use in models of the brain as well as in real-time or low-power applications.

Machine learning is struggling with similar computational bottlenecks as neuroscience; the best models are very large and resource-expensive to run [135]. Novel architectures, specifically Neural Ordinary Differential Equations (Neural ODEs) [7],

have considerable similarities with neural networks that are used in computational neuroscience. For example convolutional neural networks have been used as models in studies of the visual system [91]. Machine learning models focus on *predicting* outputs from unseen inputs. Neural ODEs improve the capability of deep learning systems to also consider time-dynamics of the modeled system, which is close to the modeling paradigm in computational neuroscience. It then makes sense to ask whether MOR methods are applicable for accelerating deep learning models and if they are, how well do they perform in prediction tasks. In **Publication IV**, we answer these questions.

Neural ODEs offer a natural setting for POD-DEIM model reduction. To create a Neural ODE model, a large collection of training data is required. Once the model is trained, the same training data can be used to collect snapshots for the POD-DEIM method, as we demonstrate in **Publication IV**. It is important, for both the neural network and POD-DEIM, that the training data set gives a varying and accurate description of the data that the model will encounter in a real-world setting. Using data that has not been employed for model training or reduction, it is possible to validate the performance of the full and reduced models. Hence the data sets that are present for machine learning purposes, can be used as is for model reduction.

In **Publication IV** we show how the POD-DEIM method can be entirely implemented for Neural ODEs by manipulating weight matrices, pruning activation functions and introducing projection and interpolation layers in the artificial neural network. For some architectures, such as fully-connected Neural ODEs receiving time-dependent input, this requires that the network is analysed in the state-space format. For architectures using convolutional layers, we propose in **Publication IV** to write convolution operations as Toeplitz matrices, in order to enable POD-DEIM reduction in the state-space format. In this manner, the reduced model also becomes a deep neural network. Then, the entire model reduction workflow can utilize existing computation tools of deep learning and the reduced model can be used as a plug-in replacement of the original network.

5.5 Performance of reduced models

In order to measure the performance of MOR methods, we computed reduced models of several systems of neuroscientific interest, and evaluated their accuracy and

speed with regards to the original, high dimensional systems as well as other reduction methods. The results towards this aim should indicate the situations where MOR methods show the most promise, as well as identify those models that are challenging to reduce using MOR.

In the publications collected into this thesis, we found model order reduction methods to be an effective and versatile tool for accelerating simulations of nonlinear models of neuronal populations and synaptic dynamics. Depending on the type and size of the original model, the achieved acceleration with low approximation error varied from a few percents to orders of magnitude. The approximation error is always a function of the desired speedup, although the dependence is rarely linear. In **Publication I**, **Publication III** and **Publication IV** it was possible to locate a speedup value after which the approximation error started increasing fast. Figure 5.2 illustrates this behavior for a convolutional and an antisymmetric recurrent deep learning models from **Publication IV**. Here, model accuracy is shown on the y-axis, with value of 1.0 indicating identical performance to the original model, while the x-axis indicates achieved speedup as the computation time of the original model divided by that of the reduced model. Each dot indicates a reduced model dimension. Reduction for the convolutional model succeeds very well, and the drop in accuracy happens after dimension 600, when the original model has 1000 ODEs. For the antisymmetric network model, the point of accuracy drop is seen around dimension 400, whereas the original model has 512 equations. The point just before rapid accuracy drop was found to be a good choice for the dimension of the reduced model, although larger speedups can be obtained if the application of the reduced model allows the resulting change in accuracy. Overall, with the correct choice of algorithms and reduction parameters, the model reduction process can be controlled to be suitable for many different model types.

With the POD-DEIM method [27], we were able to create well performing reduced models of chemical reactions in the synapse in **Publication I**, Fokker-Planck-type mean-field models in **Publication III** and convolutional continuous-time neural networks in **Publication IV**. In **Publication II**, we established advanced versions of the POD-DEIM method as necessary tools accelerating this network of biophysically detailed compartmental models.

Specifically, the DAPOD method [119], [136] together with the DEIM algorithm was found to be efficient and able to derive approximations with more ac-

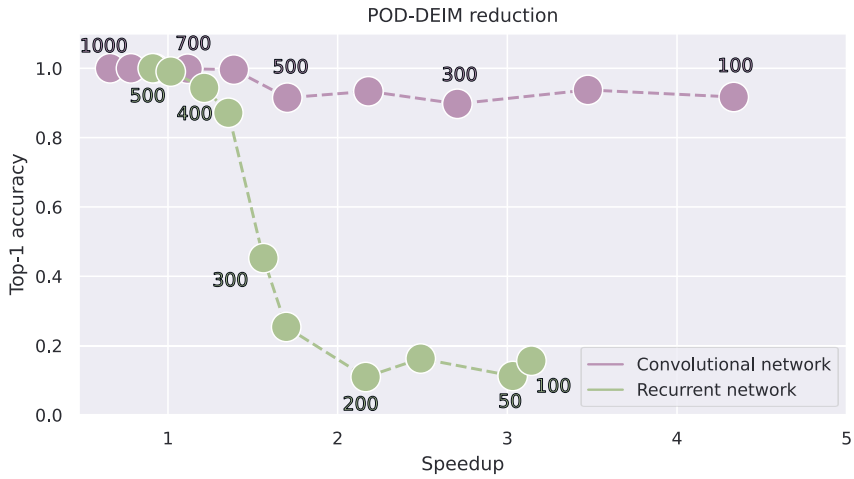

Figure 5.2 Model reduction results using POD-DEIM to reduce a convolutional and an antisymmetric neural network model. The x-axis shows model acceleration as factor of achieved speedup, while the y-axis shows the accuracy of the reduced model relative to the original model. Each dot indicates a different reduced model dimension. Adapted from **Publication IV**.

curate dynamics than the original DEIM method. Moreover in the same study, the ADEIM algorithm [124], which adapts the DEIM basis during the simulation, showed favourable approximation capability when compared to DEIM. Without these adaptive MOR methods, the reduced model showed unstable behavior in the form of residual network activity which the original model did not have. This is illustrated in Figure 5.3, where the number of neurons spiking as a function of time is compared between the original model and approximations from DEIM and ADEIM methods. The DEIM approximation shows continued network activity until the end of the simulation, whereas the ADEIM approximation correctly returns to zero activity.

The performance of QDEIM [125] and LDEIM [123] were evaluated in **Publication II** and **Publication III**. In these studies, the adaptive approach of LDEIM was not found to achieve enough improvement in accuracy over DEIM when the increased cost of computation time was considered. Hence, more favourable results in terms of accuracy versus speedup were achieved with other methods, including the original DEIM. The QDEIM algorithm improves the interpolation point selection process of the DEIM method. In the online phase, QDEIM has the same cost as the

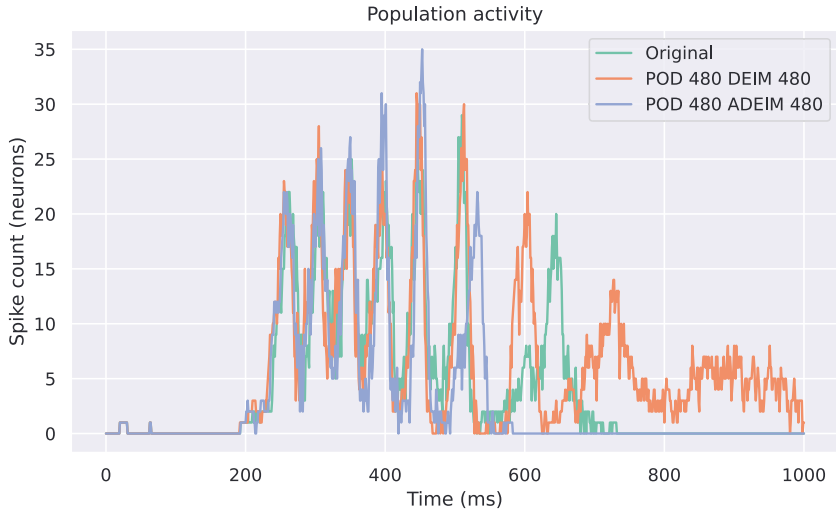

Figure 5.3 Number of neurons spiking in the biophysical neuronal network model. The x-axis shows time and y-axis the spiking neuron count. The original model, DEIM reduced model and ADEIM reduced model are shown. Adapted from **Publication II**.

original DEIM. This method was not found to deliver a difference in accuracy in **Publication III**, compared to DEIM.

The QDEIM interpolation point selection process is a recommended part of the Oversampled DEIM algorithm, which achieved good acceleration results in **Publication IV**. In **Publication IV** we found the Oversampled DEIM (sometimes called Gappy POD) to outperform the DEIM algorithm for accelerating an antisymmetric recurrent neural network that receives time-dependent input. Taken together, the fact that MOR methods keep developing at a rapid pace is leading towards algorithms that are increasingly robust, even for reducing nonlinear models.

With regards to achieved acceleration results, different amounts of reduction were feasible for every neural system model of the original publications in this thesis. In **Publication I**, the steady-state reduction error of the synapse model was mostly dependent on POD dimension, while speedup was determined by DEIM dimension. It was possible to find a reduced model that halved the simulation time. In **Publication II** a reduction of 5 %, from a system of 500 equations to 470-480 equations, was feasible to still reproduce the original population level behavior of compartmental neurons in the network. At this level of reduction, MOR does not lead into acceleration compared to the original model. This is primarily due to the reset mechanisms

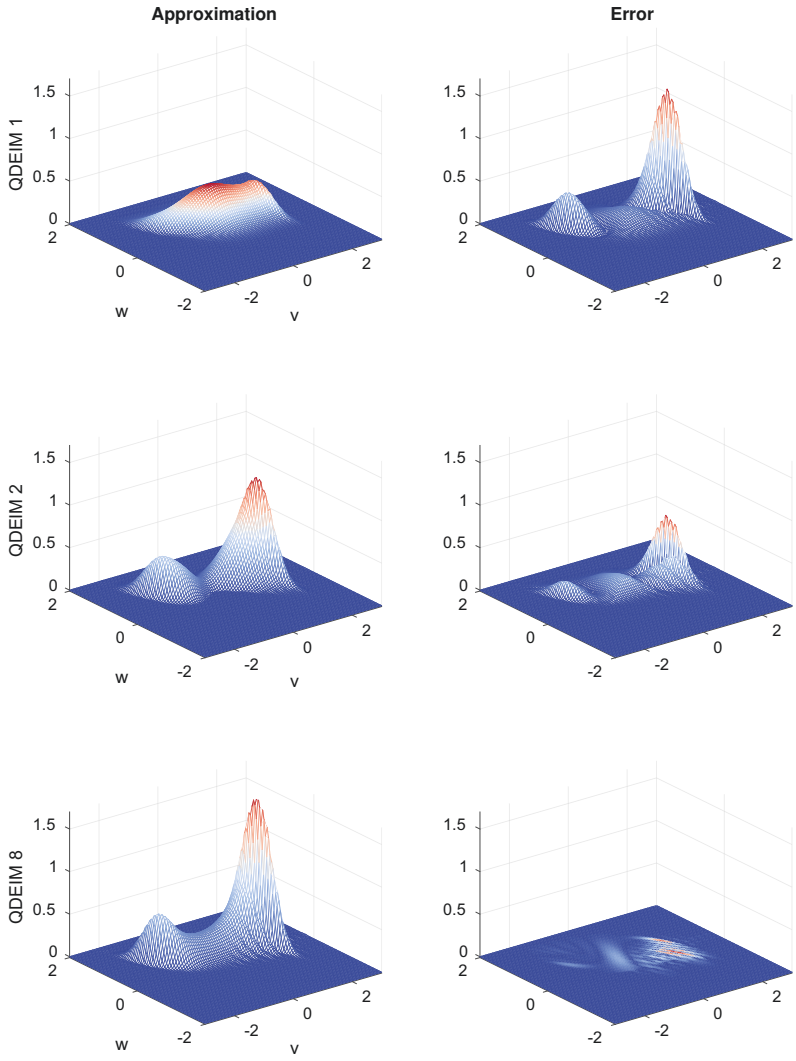

Figure 5.4 Model reduction results using QDEIM to reduce a Fokker-Planck mean-field model describing an infinitely large population of FitzHugh-Nagumo neurons. The left column illustrates approximation from the QDEIM MOR algorithm, while the right column shows approximation error. Each row is a different dimension reduced model, from 1 to 8, indicated on the left. POD and QDEIM dimensions are equal here. The original model has dimension 50^3. Adapted from **Publication III**.

of the single neuron model causing computational overhead into the ROM. In **Publication III**, where a discretized PDE mean-field model was reduced, an acceleration of three orders of magnitude was possible with great accuracy. In **Publication IV**, it was possible to retain close to 90 % of the classification accuracy of a convolutional neural network while requiring only 25 % of the computation time of the original model, a 4-fold speedup. For a more densely connected antisymmetric recurrent neural network, a reduced model with 90 % accuracy required about 80 % computation time of the original model, reaching a speedup factor of 1.2. The reported simulation times in all studies have been obtained on CPUs. The original and reduced models were given equal compute resources. No model-specific optimization, other than efficient matrix multiplications, were implemented.

The most impressive reduction was obtained for the Fokker-Planck mean-field model, possibly due to the structure that results from the discretization of the PDE model. Figure 5.4 illustrates the dramatic performance of MOR for reducing model. The left column illustrates approximation from the QDEIM MOR algorithm, while the right column shows approximation error. Each row is a different dimension reduced model. The original model has dimension 50^3. For this model, MOR converges to the correct solution with dimensions less than 10.

What remains challenging is the model order reduction of densely connected neuronal networks that receive time-dependent input. In **Publication II** the acceleration achieved in reducing a network of compartmental neurons was not sufficient when considering the approximation error introduced in the process. In **Publication IV**, the POD-DEIM method found a great balance between speedup gained and accuracy lost when reducing a convolutional continuous-time network, improving on existing neural network pruning methods from the literature, but did not lead into better results than benchmark methods when reducing an all-to-all antisymmetric network with time-dependent input. In **Publication II**, the inputs were included in the snapshot collection phase, whereas in **Publication IV** the inputs are novel to the reduced model. Future studies are needed to quantify the effect of time-dependent inputs to reduced models in more detail.

In Appendix A a lifting transformation of a nonlinear model into QB format is shown. While the lifting transformation is very appealing for bringing new models into the realm of data-free model reduction algorithms, two problems can arise. Numerically the lifted system may become stiff, which negatively affects simulation

time. Additionally, the size of the matrix of nonlinear coefficients, H in Eq. (2.3), scales quadratically with lifted system size, and this leads to large memory requirements in the MOR methods. While it was possible to compress the high-dimensional, lifted form of the HR+TM network using BT and IRKA, as of yet the gain in faster simulation time was not worth the cost of increased approximation error and complexity of the approach.

6 DISCUSSION

In this thesis we have presented results towards the validity and efficacy of model order reduction (MOR) methods for accelerating nonlinear neuronal network models in computational neuroscience. Despite advances in computing power, large-scale simulations of the mammalian brain using biophysically detailed models are not practical [6]. To this end, several methods of *simplifying* single neuron and neuronal population models have been developed. While these approaches and the resulting models have been tremendously useful, their primary shortcoming is that spatial resolution of neuronal models is compromised [63] and details from the morphology, network structure or cellular mechanisms are removed. To maintain high fidelity in reduced models, the publications included in this thesis use MOR methods to accelerate simulations. Specifically, we focused on methods that are based on projecting a high-dimensional model onto a low-dimensional subspace for the simulation phase. The resulting reduced models are efficient to simulate and an approximation of the dynamics of the original system variables can be constructed at any time. Moreover, these reduced models maintain the original outputs of the system, inputs to the system and all parameters of the original model, hence spatial resolution is preserved and interactions between variables can be studied. For this reason, the reduced models can serve as plug-in replacement of computationally expensive systems, for example in multi-scale simulations.

6.1 Impact of MOR in computational neuroscience

The full impact of MOR will be realized once these methods can be incorporated in the toolkit of computational neuroscientists, for example integrated into neuronal simulator software. Neural simulation tools may not always store models in the state-space format in which MOR is applied. It may be laborious to extract the high-dimensional models from neural simulators in a format that could be di-

rectly reduced using MOR. While the state-space formulation is standard in fields like control-theory, neural simulators tend to use their own internal representations of models. This means that MOR methods and reduced models do not integrate with existing neural simulators automatically. On the positive side, in **Publication IV** we have demonstrated integration of MOR with deep learning software and artificial neural networks, which is a promising direction for MOR and a practical step for integrating MOR into a task-specific tool.

To integrate MOR into neuronal simulators, the reduced model should be linked to the simulators via inputs and outputs of the original full-order model in the format of Eq. (2.1). Whether this can be implemented in software differs from simulator to simulator. The Virtual Brain [137] is an interesting simulation tool in this regard, since users can write new equations to use as nodes in whole-brain simulations. With this level of modularity, reduced models can be defined as nodes in brain simulations. The NEST [138] simulator is another tool where users can define network components such as neurons and synapses as code, making it possible to replace components with their reduced version. These simulators seem good targets for initial MOR integration work. In general, open source software is a great way to facilitate this integration, hence we have made the code of our integration of MOR with deep learning for **Publication IV** available with the publication.

Previous work by others has demonstrated applicability of MOR for reducing morphologically detailed single neurons with passive, linearized and nonlinear dynamics [11], [12], [22]. Moreover, MOR has been shown to be effective for accelerating simulations of electrophysiological activity in cardiac monodomain and bidomain models, although their dynamics are diffusion-driven and hence simpler than neuronal network activity stemming from action potentials and synaptic processing [139], [140]. This thesis establishes MOR as a viable method for reducing models of chemical reactions in the synapse (**Publication I**), network models of biophysically detailed compartmental neurons (**Publication II**), high-dimensional Fokker-Planck-type mean-field models (**Publication III**) and artificial neural networks (**Publication IV**). By studying these classes of models as new targets for MOR, we expand the range of scenarios where MOR can be useful. We also provide new insights for the MOR community, where our models are an unconventional reduction target.

The MOR approach does not depend on numerical or manual tuning of parame-

ters of the reduced model, and all parameters of the original model are kept intact and adjustable. This is valuable for parameter tuning and sensitivity studies. In morphologically simplified models, starting with the earliest equivalent cylinders and cables, biophysical parameters are commonly rescaled or optimized after simplifying the morphology. The rescaling is done for the purpose of maintaining features such as transfer impedance [73] and electrotonic properties [72] of the original model, which should improve the accuracy of simplified models. In doing so, the values of adjusted parameters may eventually be out of biologically realistic ranges [77]. Although the models still reproduce experimentally observed phenomena, the contribution of parameters in the original and simplified models may not not comparable. Hence it is questionable to use such models in parameter sensitivity studies, for example. The MOR approach may be more theoretically suitable for such use cases. However, not all MOR methods are computationally efficient for parameter sensitivity studies. If the reduced model needs to be recomputed after each parameter change, methods such as Balanced Truncation will become computationally expensive in the offline phase.

With regards to other desirable properties, MOR can have a priori bounds [13] for error, guarantees of asymptotic stability [13], preservation of non-negativity [22], positivity [141] and passivity [45], [142], [143]. Extensions to specific stochastic systems have also been considered [144], [145]. All these qualities are actively researched for both linear and nonlinear systems. That is not to say that *ad hoc* simplification methods or approaches based purely on numerical optimization could not maintain biological laws or give error estimates, since it is possible with proper regularization and heuristic rules. However, it could be challenging to guarantee these properties for *ad hoc* methods, and discussion of these questions is missing in morphological simplification literature. The more properties that can be carried from the original, complex models to the accelerated models, the higher quality of reduced systems can be promised.

MOR has an active research and developer base. Existing MOR methods are being continuously improved and new modifications studied. As shown in **Publication IV** and **Publication II**, extensions to the original DEIM algorithm have improved the accuracy and reliability of the method. Once a neuroscientific model is implemented in suitable state-space format for MOR, reducing the model with a variety of algorithms becomes possible. Likewise, there are tools being developed to apply MOR

methods with little mathematical or programming effort required [146]–[149]. Since these tools are more general than any given neuron simplification method, a large researcher and developer community is incentivized to participate in the development. It would be beneficial for neuroscience to embrace this opportunity to leverage theoretically validated and professionally developed software. In summary, the infrastructure that enables model order reduction is large and growing.

Despite the above advantages, knowledge of MOR methods in the computational neuroscience community appears limited. For example, earlier state-of-the-art MOR work in neuroscience by Kellems et al. [12], [44] or Du et al. [11] is not acknowledged in some of the latest high-profile simplification publications. Others who do cite MOR work may have misunderstood the characteristics of projection-based reduction, claiming that reduced models do not preserve the biophysical interpretation of the models. However, the physiological variables can always be reconstructed while inputs and outputs of the model remain unchanged. We hope that this thesis expands the knowledge and facilitates the use of MOR in neuroscience.

MOR does not remove the possibility of using other, more traditional simplification methods to derive efficient models. The model reduction process could begin with applying morphological simplification methods or network pruning, using any of the existing tools or expert knowledge. This would lessen the number of parameters in the model, and it should be agreed that some loss of spatial resolution or model variables is acceptable. The next step would be to apply MOR to the simplified model, as was done in **Publication II**, in order to take advantage of any redundancies that were not removed in the simplification step. This "best of both worlds" approach could potentially lead to very efficient models, but needs to be studied further in future work.

6.2 Open questions

During the studies of this thesis, the following topics were identified as areas that could benefit from additional exploration and research. While we were able to find initial solutions for some of these questions, more rigorous effort could be directed towards each topic.

The offline cost of MOR methods can be high. Empirical methods such as POD-DEIM require the collection of samples of system dynamics, which entails simulation

of the high-dimensional system. Moreover, this snapshot collection may have to be repeated with several initial values or system parameters. For other methods such as Balanced Truncation, solving the Lyapunov methods, even in low-rank factorized form, is a challenging task that requires extensive computing power, time and memory. Depending on the application, such offline costs may be acceptable, for example when repeatedly simulating the reduced model or if distributing it online for others to use. The same can be said for simplification methods that are based partially or entirely on numerical optimization and fitting on experimental data. So even with MOR methods at hand, sufficient computing capacity should be available for the offline phase where reduced models are computed.

Phenomenological models of neural dynamics commonly contain reset or threshold mechanisms where a variable is set manually to a reference value after reaching a specific level [15]. In **Publication II**, such a mechanisms was used to model excitatory synaptic currents. From a projection-based model reduction perspective, the reset mechanism has two implications. First, checking the reset mechanism forces the reconstruction of a variable from the original high-dimensional space. Since this is generally a nonlinear operation, the full state of the reduced model must be reconstructed, as complete state information is required by projection back into the reduced space. This has a negative effect on the runtime of the reduced model since thresholding conditions are commonly checked at every timestep (but not at intermediate ODE solver steps). Second, reset conditions can introduce large jumps in the time-trajectory of the model, and the precise timing of the jumps may be offset by approximation errors from the reduced model. Over time, these errors can accumulate and lead into unexpected behavior of the reduced model. Whether thresholding operations could be efficiently implemented, perhaps so that the full state of the model need not be reconstructed at each time step, remains an open question [133].

Fokker-Planck-type (FP) mean-field models describe the time evolution of the probability density of variables in large-scale neuronal network models [84], [87]. Mathematically the FP models guarantee that the probability density at any point is equal to or greater than zero and that total probability integrates to one. This holds as long as discretization is done accurately, with a sufficiently fine grid, although this does not carry over to reduced order models [150]. The POD-DEIM method, cannot ensure either of these conditions natively [22]. Despite this shortcoming, in **Publication III** we found the POD-DEIM method to achieve several orders of

magnitude acceleration and good accuracy when reducing a FP system modeling a network of FitzHugh-Nagumo neurons.

In a similar manner, the POD-DEIM method does not automatically guarantee positivity of concentrations or rate variables such as those in the synapse model of **Publication I** and biophysical network model in **Publication II**. The Non-negative DEIM [22] is a step towards such guarantees, although at present requires a specific PDE discretization scheme to be applicable [151].

In neuroscience, mathematical models may contain phenomena from different time-scales. Accounting for this information could improve reduction accuracy. One MOR method that separates fast and slow time-scales is the Singular Perturbation Approximation, a balancing-related approach [152]. There is no extension of time-scale aware methods to general nonlinear systems, but bilinear systems have been studied in [153].

The nonlinear model reduction methods used in the publications of this thesis do not guarantee asymptotic stability of the reduced models. In general, global stability of nonlinear systems cannot be determined analytically. For this reason, stability in an initial value and input regime of interest is commonly verified using numerical methods. In **Publication II**, the biophysical network model receives stimulation and after a period of synchronized network activity returns to resting membrane voltages. We found that POD-DEIM and LDEIM reduced models of the system contained instabilities that were not present in the original system. The instabilities manifested as spontaneous or prolonged activity of the network. Using adaptive model reduction methods such as DAPOD and ADEIM, it was possible to derive stable reduced order models. While the adaptive methods improve accuracy, they are not as efficient as methods that use constant reduction bases.

When modeling neuronal behavior using universal function approximators such as recurrent neural networks, the dynamics are usually connected to a downstream linear or nonlinear readout layer that determines model performance. The readout may be used for classification or regression purposes, as in **Publication IV**. In these architectures, the POD-DEIM model order reduction method is not aware of the overall performance of the model. The focus is purely on approximating the dynamics of the system, as we show in **Publication IV**. Hence, connecting the reduction method into downstream network outputs could improve the reduction result on the full network level. Methods such as BT and transfer function interpolation are

more faithful in preserving the input-output response of the reduced model, but using them for accelerating nonlinear neural networks has not been shown. At present it is not possible to establish a relationship between model reduction methods and general neural network outputs.

An additional question concerns the applicability of MOR methods in parameter sensitivity studies. The POD-DEIM algorithm can in principle be employed for this purpose, as a computed basis can be used to reduce a system where parameters or initial values have changed. The BT algorithm needs to be executed from the beginning every time parameters of the model are changed, as the solution of the Lyapunov equations will change accordingly. The same is true for transfer function interpolation methods that compute the reduced model directly from the system matrices. Hence, BT and IRKA may not be practical for parameter sensitivity studies. All three methods can be used for studying the effects of initial value perturbations.

The POD-DEIM method relies on collecting snapshots of the dynamics of the high-dimensional system. Due to large number of possible states of a neuronal system, it is practically impossible, due to time and memory constraints, to collect a snapshot set that exhaust all possible system dynamics. Quantifying the generalization capability to states for which snapshots were not gathered is a relevant question in all POD-DEIM model reduction studies. In **Publication IV**, we found that POD-DEIM can compute reduced models that work outside the snapshot set. In **Publication IV** this was demonstrated in a machine learning task on data that was not used for reduction. We found that MOR methods can perform better than two benchmarked acceleration methods, one data-driven and one data-independent. In **Publication I** we concluded that the reduced model needed higher dimensionality when studying longer time scales and that generalizing outside the snapshot set increased approximation error. While data-driven, empirical methods such as POD-DEIM typically generate accurate reduced models for simulations in the snapshots space, we conclude that data-dependence remains a challenge for the generalization capability of reduced models.

To use the BT reduction method, nonlinear models of neuroscience tend to require a quadratic-bilinear lifting transformation of the system. This transformation typically does not yield a linear weight matrix that has eigenvalues with strictly negative real parts, a requirement for solving the Lyapunov equations uniquely during the BT method. In this case, it is necessary to artificially stabilize the lifted system

by an eigenvalue shift [130], [150], [154]. The eigenvalue shift may allow the BT method to find a reduced model of a system that is *close* to the original, but it unavoidably introduces an additional source of error into the reduced model. Whether models could be designed with stability in mind, or if MOR methods can be developed to better cope with unstable systems, remains a question for future studies in computational neuroscience and MOR method research.

6.3 Future directions

Optimizing the performance of MOR methods is an interesting future research topic. There are a number of hyperparameters in the MOR process. For example, the POD-DEIM method requires choosing initial values and parameters for snapshot generation, snapshot sampling algorithm (e.g. adaptively update the set or fixed step sampling), and dimensions of POD and DEIM bases separately. Advanced variations like the Localized DEIM add choices such as clustering method and distance function. For BT, the algorithms for finding factorized gramians are many, each with their own parameters. Different system stabilization schemes are available and the input and output matrices can be augmented to guide the BT algorithm. A study for optimizing these hyperparameters should be interesting, although it would have to be completed on a specific model and method basis.

Several directions can be taken to optimize the speed of reduced models, which was not a goal of the present thesis. For example, choosing the best differential equation solver for the reduced model is not trivial, as the reduced model may be stiff even if the original system is not, requiring different numerical treatment. Efficient distributed and parallel computation of reduced models are interesting topics and highly dependent on the used MOR method. In neuroscience, implementation of reduced models on neuromorphic hardware, such as the SpiNNaker system [155], is an exciting research direction.

A current trend is employing more and more machine learning to derive low-order models [156]. In neuroscience, it has been shown that a deep convolutional neural network can mimic the action potential patterns of a cortical neuron [92]. The authors provide an interesting analysis of how the learned weights map into properties of the original model. Although it was not a primary objective of their work, the authors observed a speedup of many orders of magnitude when using

their deep neural network model. While the method in [92] does not attempt to reconstruct the original model like MOR does, several lines of research are looking at combining machine learning and model reduction more tightly. For example, non-intrusive methods attempt to learn ROMs without knowing the equations of the original system [157]. Furthermore, machine learning has been used to reconstruct the original system from a low-dimensional model [158], a clever approach for adding nonlinearity to the MOR process. Additionally, there are promising methods for learning ODEs from data using gradient-based optimization by leveraging the adjoint method [7], [159], [160]. Notably these methods depend heavily on training data, even more so than the regular POD-DEIM reduction method.

It has been observed in both neuroscience (e.g. [161]) and machine learning [162] that task-specific dynamics, such as neuronal activity during navigation, of neural networks seem to occupy low-dimensional manifolds. The concept of using low-dimensional subspaces for model reduction bears many similarities to the manifold hypothesis. Future work should explore the link between model reduction and low-dimensional population dynamics further. Being able to connect the reduction subspace to the physical world, similarly to how low-dimensional manifolds are currently found in high-dimensional neural networks [162], [163], would provide a completely novel use case for MOR in neuroscience, although at the moment this idea has not been explored in the literature.

A very promising future direction for model reduction is the lifting of general nonlinear models into structured polynomial nonlinear form [8], [24], [25], [154]. The lifting transformation can lead into an equivalent quadratic-bilinear (QB) form of a nonlinear state-space model, where the original variables are augmented with complementary variables. However, there is no guarantee of the existence or uniqueness of the lifted form, meaning that the process may not find a quadratic-bilinear format for every model. Additionally, it is challenging to automate the lifting process. In Appendix A we demonstrate a lifted form for a network of Hindmarsh-Rose cells connected by Tsodyks-Markram synapses. The QB form is interesting, as some MOR methods that were previously only applicable to linear systems have been extended to the QB case [10], [130]. In the future, cubic nonlinearities may hopefully be in the scope of these methods, as a lifting transformation into cubic nonlinearities is easier to find than the QB form, further extending the choice of MOR methods for nonlinear systems.

With regards to the publications of this thesis, we want the highlight the following possibilities for future research questions.

For extending the work in **Publication I**, we propose further evaluations of the reduced model under different stimulus conditions. Especially quantifying the performance under more stimuli that were not present in the snapshot collection phase would be an important step for validating the accuracy of the reduced model.

With the network model reduced in **Publication II**, it would be interesting to explore model reduction with constraints. In our study, we reduced the complete network as a single system. Alternatively, it could be possible to use the DEIM algorithm to indicate the importance of nodes in the network, instead of individual variables in the system. MOR could then be targeted only to the least important nodes. In this manner, it may be possible to improve accuracy of the reduced system. The main challenge is determining the important nodes, as the DEIM algorithm does not have a notion of "aggregate" importance of multiple variables. Hence the reduction would be suboptimal from the perspective of the POD-DEIM method, but there may be interesting practical implications.

The model studied in **Publication III** can be forced into QB form if linear methods are used to compute the triple integrals over the probability density function. In follow-up studies, model reduction could be attempted with the Balanced Truncation or Iterative Rational Krylov Approximation methods that have been extended to QB systems. These methods would have the advantage that the snapshots of the high-dimensional system would not be required prior to reduction.

With regards to **Publication IV**, where deep learning models were reduced, next steps should focus on understanding the contribution of the (connectivity) weight matrix to the reduction result. We observed that a network with a densely connected weight matrix with an antisymmetric pattern was challenging to reduce. Follow-up work could ask how the structure of the weight matrix affects reducibility, while controlling for factors such as inputs to the system. An additional line of research would look at connecting the classification or regression accuracy of the complete network to the reduction process.

7 CONCLUSIONS

Computational methods are necessary for understanding the brain [1], but the currently available hardware is not powerful enough for conducting large-scale simulation studies in neuroscience [3]. The hypothesis stated in the introduction of this thesis was that MOR methods could accelerate nonlinear neuronal network models and network model components used in computational neuroscience. Based on the original publications of this thesis, we can conclude that several neuroscientifically relevant models and model types that were not previously studied can be successfully reduced using MOR methods. Additionally, we integrated MOR into the deep learning workflow. These new applications of MOR are expected to facilitate large-scale simulations of the mammalian brain.

MOR is interesting, as these methods are able to approximate the complete morphology and connectivity of the high-dimensional model. Earlier MOR studies in computational neuroscience had studied the acceleration of cable equation-based single neuron models [11], [12]. We first selected new targets for MOR in neuroscience; a model of chemical reactions in the synapse [35], a biophysical network model [15], a high-dimensional mean-field model [87] and two continuous-time artificial neural network models. These models were then implemented in the state-space format and their mathematical properties were analysed in order to choose suitable MOR methods for accelerating the models.

We found that Proper Orthogonal Decomposition (POD) with Discrete Empirical Interpolation Method (DEIM) [27] was the appropriate approach for accelerating our selected models. For quadratic bilinear (QB) systems the QB extensions of the Iterative Rational Krylov Approximation (IRKA) method [10] and Balanced Truncation (BT) method [130] appear promising. QB systems are significant because through lifting transformations [8], [24] some nonlinear systems may be brought to this form. Moreover, we used several advanced POD and DEIM algorithms that had not been applied in neuroscience earlier. These results should prove interesting for

both the neuroscience and the MOR communities.

The largest acceleration results were obtained with the Fokker-Planck mean-field model, where simulation times were improved by orders of magnitude while retaining satisfactory approximation accuracy. The result is very encouraging for including additional detail to mean-field based whole-brain simulations. For modeling chemical reactions in the synapse, the simulation speed was halved while maintaining accurate results. This result is meaningful for facilitating network simulations with realistic synapse models. For networks using compartmental neurons we found that adaptive MOR methods are beneficial, yet more work needs to be done in order to enable accurate and efficient low-dimensional simulation of these models. For accelerating nonlinear artificial neural networks with convolution operations, used for example in vision research, we found that MOR methods performed better than two established benchmark methods from the literature. For antisymmetric neural networks receiving time-dependent input, the Oversampled DEIM [126] was needed to obtain results on par with benchmark methods. These findings establish the usefulness of MOR for computational neuroscience.

Application of MOR is not always straightforward. In the present moment, general nonlinear models can only be reduced using empirical methods such as POD-DEIM, while extensions of non-empirical methods such as BT are still being developed. Additionally, phenomenological mechanisms used in models in computational neuroscience still restrict the computational speedup that can be obtained using MOR. Nevertheless, the benefits of MOR methods, for example the capability to reconstruct the high-dimensional model in full after low-dimensional simulation and the fact that MOR is not limited to specific model types, outweigh the challenges.

The future of MOR in computational neuroscience will benefit from the continued development of POD-DEIM methods. Additionally, the lifting of general nonlinear models into more structured polynomial forms will enable the use of MOR methods that are not dependent on snapshot data like the POD-DEIM is. These methods include the BT method and transfer function interpolation methods such as IRKA. We are also excited about the possibilities offered by progress in deep learning, both as a reduction target and a tool towards identifying reduced models. The largest impact of MOR methods will be achieved when the methods can be seamlessly integrated into neuronal simulation software. At that point, MOR methods can push the boundaries of multi-scale model simulations beyond the current levels.

REFERENCES

[1] W. Gerstner, H. Sprekeler, and G. Deco, "Theory and simulation in neuroscience", *Science*, vol. 338, no. 6103, pp. 60–65, 2012.

[2] G. Einevoll, A. Destexhe, M. Diesmann, *et al.*, "The scientific case for brain simulations", *Neuron*, vol. 102, no. 4, pp. 735–744, 2019.

[3] H. Markram *et al.*, "Reconstruction and simulation of neocortical microcircuitry", *Cell*, vol. 163, no. 2, pp. 456–492, 2015.

[4] C. Eliasmith, T. C. Stewart, X. Choo, *et al.*, "A large-scale model of the functioning brain", *Science*, vol. 338, no. 6111, pp. 1202–1205, 2012.

[5] A. Arkhipov, N. W. Gouwens, Y. N. Billeh, *et al.*, "Visual physiology of the layer 4 cortical circuit in silico", *PLoS Computational Biology*, vol. 14, no. 11, e1006535, 2018.

[6] A. E. Urai, B. Doiron, A. M. Leifer, and A. K. Churchland, "Large-scale neural recordings call for new insights to link brain and behavior", *Nature Neuroscience*, pp. 1–9, 2022.

[7] R. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural Ordinary Differential Equations", *Advances in Neural Information Processing Systems*, vol. 31, 2018, pp. 6571–6583.

[8] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, "Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems", *Physica D: Nonlinear Phenomena*, vol. 406, p. 132 401, 2020.

[9] K. Lu, Y. Jin, Y. Chen, *et al.*, "Review for order reduction based on proper orthogonal decomposition and outlooks of applications in mechanical systems", *Mechanical Systems and Signal Processing*, vol. 123, pp. 264–297, 2019.

[10] P. Benner, P. Goyal, and S. Gugercin, "$\mathcal{H}_2$-quasi-optimal model order reduction for quadratic-bilinear control systems", *SIAM Journal on Matrix Analysis and Applications*, vol. 39, no. 2, pp. 983–1032, 2018.

[11] B. Du, D. Sorensen, and S. J. Cox, "Model reduction of strong-weak neurons", *Frontiers in Computational Neuroscience*, vol. 8, p. 164, 2014.

[12] A. R. Kellems, S. Chaturantabut, D. C. Sorensen, and S. J. Cox, "Morphologically accurate reduced order modeling of spiking neurons", *Journal of Computational Neuroscience*, vol. 28, no. 3, pp. 477–494, 2010.

[13] P. Benner, S. Gugercin, and K. Willcox, "A survey of projection-based model reduction methods for parametric dynamical systems", *SIAM Review*, vol. 57, no. 4, pp. 483–531, 2015.

[14] E. M. Izhikevich, "Simple model of spiking neurons", *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[15] P. F. Pinsky and J. Rinzel, "Intrinsic and network rhythmogenesis in a reduced Traub model for CA3 neurons", *Journal of Computational Neuroscience*, vol. 1, no. 1, pp. 39–60, 1994.

[16] R. D. Traub, R. K. Wong, R. Miles, and H. Michelson, "A model of a CA3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances", *Journal of Neurophysiology*, vol. 66, no. 2, pp. 635–650, 1991.

[17] W. Rall, "Theoretical significance of dendritic trees for neuronal input-output relations", *Neural Theory and Modeling*, pp. 73–97, 1964.

[18] C. Runge, "Über die numerische Auflösung von Differentialgleichungen", *Mathematische Annalen*, vol. 46, no. 2, pp. 167–178, 1895.

[19] W. Kutta, "Beitrag zur näherungsweisen Integration totaler Differentialgleichungen", *Zeitschrift für Angewandte Mathematik und Physik*, vol. 46, pp. 435–453, 1901.

[20] B. Moore, "Principal Component Analysis in linear systems: Controllability, observability, and model reduction", *IEEE Transactions on Automatic Control*, vol. 26, no. 1, pp. 17–32, 1981.

[21] R. W. Freund, "Krylov-subspace methods for reduced-order modeling in circuit simulation", *Journal of Computational and Applied Mathematics*, vol. 123, no. 1-2, pp. 395–421, 2000.

[22] D. Amsallem and J. Nordström, "Energy stable model reduction of neurons by Nonnegative Discrete Empirical Interpolation", *SIAM Journal on Scientific Computing*, vol. 38, no. 2, B297–B326, 2016.

[23] C. Gu, "QLMOR: A new projection-based approach for nonlinear model order reduction", *2009 IEEE/ACM International Conference on Computer-Aided Design-Digest of Technical Papers*, IEEE, 2009, pp. 389–396.

[24] C. Gu, "QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 9, pp. 1307–1320, 2011.

[25] B. Kramer and K. E. Willcox, "Nonlinear model order reduction via lifting transformations and Proper Orthogonal Decomposition", *AIAA Journal*, vol. 57, no. 6, pp. 2297–2307, 2019.

[26] L. Sirovich, "Turbulence and the dynamics of coherent structures. i. coherent structures", *Quarterly of Applied Mathematics*, vol. 45, no. 3, pp. 561–571, 1987.

[27] S. Chaturantabut and D. C. Sorensen, "Nonlinear model reduction via discrete empirical interpolation", *SIAM Journal on Scientific Computing*, vol. 32, no. 5, pp. 2737–2764, 2010.

[28] A. C. Antoulas, *Approximation of large-scale dynamical systems*. SIAM, 2005.

[29] Z. Bai, "Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems", *Applied Numerical Mathematics*, vol. 43, no. 1-2, pp. 9–44, 2002.

[30] W. Rall, "Branching dendritic trees and motoneuron membrane resistivity", *Experimental Neurology*, vol. 1, no. 5, pp. 491–527, 1959.

[31] C. Koch and I. Segev, "The role of single neurons in information processing", *Nature Neuroscience*, vol. 3, no. 11, pp. 1171–1177, 2000.

[32] T. J. Sejnowski, C. Koch, and P. S. Churchland, "Computational neuroscience", *Science*, vol. 241, no. 4871, pp. 1299–1306, 1988.

[33] A. Buchin, R. de Frates, A. Nandi, *et al.*, "Multi-modal characterization and simulation of human epileptic circuitry", *Cell Reports*, vol. 41, no. 13, p. 111 873, 2022.

[34] M. Tsodyks, K. Pawelzik, and H. Markram, "Neural networks with dynamic synapses", *Neural Computation*, vol. 10, no. 4, pp. 821–835, 1998.

[35] B. Kim, S. L. Hawes, F. Gillani, L. J. Wallace, and K. T. Blackwell, "Signaling pathways involved in striatal synaptic plasticity are sensitive to temporal pattern and exhibit spatial specificity", *PLoS Computational Biology*, vol. 9, no. 3, e1002953, 2013.

[36] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve", *The Journal of Physiology*, vol. 117, no. 4, pp. 500–544, 1952.

[37] O. J. Walch and M. C. Eisenberg, "Parameter identifiability and identifiable combinations in generalized Hodgkin–Huxley models", *Neurocomputing*, vol. 199, pp. 137–143, 2016.

[38] M. E. Nelson, *Databasing the brain: from data to knowledge*. Wiley Hoboken, NJ, USA, 2004.

[39] C. Koch, "Cable theory in neurons with active, linearized membranes", *Biological Cybernetics*, vol. 50, no. 1, pp. 15–33, 1984.

[40] B. Woo, D. Shin, D. Yang, and J. Choi, "Reduced model and simulation of neuron with passive dendritic cable: An eigenfunction expansion approach", *Journal of Computational Neuroscience*, vol. 19, no. 3, pp. 379–397, 2005.

[41] B. Woo and J. Choi, "Reduced model and simulation of myelinated axon using eigenfunction expansion and singular perturbation", *Computers in Biology and Medicine*, vol. 37, no. 8, pp. 1148–1154, 2007.

[42] D. Shin, D. R. Yang, and J. Choi, "On the use of pseudo-spectral method in model reduction and simulation of active dendrites", *Computers in Biology and Medicine*, vol. 39, no. 4, pp. 340–345, 2009.

[43] A. Omurtag and W. W. Lytton, "Spectral method and high-order finite differences for the nonlinear cable equation", *Neural Computation*, vol. 22, no. 8, pp. 2113–2136, 2010.

[44] A. R. Kellems, D. Roos, N. Xiao, and S. J. Cox, "Low-dimensional, morphologically accurate models of subthreshold membrane potential", *Journal of Computational Neuroscience*, vol. 27, no. 2, pp. 161–176, 2009.

[45] B. Yan and P. Li, "Reduced order modeling of passive and quasi-active dendrites for nervous system simulation", *Journal of Computational Neuroscience*, vol. 31, no. 2, pp. 247–271, 2011.

[46] K. R. Hedrick and S. J. Cox, "Structure-preserving model reduction of passive and quasi-active neurons", *Journal of Computational Neuroscience*, vol. 34, no. 1, pp. 1–26, 2013.

[47] J. L. Lumley, "The structure of inhomogeneous turbulent flows", *Atmospheric Turbulence and Radio Wave Propagation*, 1967.

[48] M. J. Rempe and D. L. Chopp, "A predictor-corrector algorithm for reaction-diffusion equations associated with neural activity on branched structures", *SIAM Journal on Scientific Computing*, vol. 28, no. 6, pp. 2139–2161, 2006.

[49] D. Ioan, R. Bărbulescu, L. M. Silveira, and G. Ciuprina, "Reduced order models of myelinated axonal compartments", *Journal of Computational Neuroscience*, vol. 47, no. 2, pp. 141–166, 2019.

[50] W. Rall, "Theory of physiological properties of dendrites", *Annals of the New York Academy of Sciences*, vol. 96, no. 4, pp. 1071–1092, 1962.

[51] W. Rall, "Electrophysiology of a dendritic neuron model", *Biophysical Journal*, vol. 2, no. 2 Pt 2, p. 145, 1962.

[52] J. Cooley and F. Dodge Jr, "Digital computer solutions for excitation and propagation of the nerve impulse", *Biophysical Journal*, vol. 6, no. 5, pp. 583–599, 1966.

[53] W. Rall and G. M. Shepherd, "Theoretical reconstruction of field potentials and dendrodendritic synaptic interactions in olfactory bulb", *Journal of Neurophysiology*, vol. 31, no. 6, pp. 884–915, 1968.

[54] J. W. Fleshman, I. Segev, and R. Burke, "Electrotonic architecture of type-identified alpha-motoneurons in the cat spinal cord", *Journal of ceurophysiology*, vol. 60, no. 1, pp. 60–85, 1988.

[55] J. D. Clements and S. J. Redman, "Cable properties of cat spinal motoneurones measured by combining voltage clamp, current clamp and intracellular staining", *The Journal of Physiology*, vol. 409, no. 1, pp. 63–87, 1989.

[56] A. K. Schierwagen, "A non-uniform equivalent cable model of membrane voltage changes in a passive dendritic tree", *Journal of Theoretical Biology*, vol. 141, no. 2, pp. 159–179, 1989.

[57] R. R. Poznanski, "A generalized tapering equivalent cable model for dendritic neurons", *Bulletin of Mathematical Biology*, vol. 53, no. 3, pp. 457–467, 1991.

[58] M. Ohme and A. Schierwagen, "An equivalent cable model for neuronal trees with active membrane", *Biological Cybernetics*, vol. 78, no. 3, pp. 227–243, 1998.

[59] T. B. Kepler, L. Abbott, and E. Marder, "Reduction of conductance-based neuron models", *Biological Cybernetics*, vol. 66, no. 5, pp. 381–387, 1992.

[60] P. C. Bush and T. J. Sejnowski, "Reduced compartmental models of neocortical pyramidal cells", *Journal of Neuroscience Methods*, vol. 46, no. 2, pp. 159–166, 1993.

[61] A. Destexhe, "Simplified models of neocortical pyramidal cells preserving somatodendritic voltage attenuation", *Neurocomputing*, vol. 38, pp. 167–173, 2001.

[62] R. Burke, "Comparison of alternative designs for reducing complex neurons to equivalent cables", *Journal of Computational Neuroscience*, vol. 9, no. 1, pp. 31–47, 2000.

[63] E. B. Hendrickson, J. R. Edgerton, and D. Jaeger, "The capabilities and limitations of conductance-based compartmental neuron models with reduced branched or unbranched morphologies and active dendrites", *Journal of Computational Neuroscience*, vol. 30, no. 2, pp. 301–321, 2011.

[64] M. Hines, "Efficient computation of branched nerve equations", *International Journal of Bio-medical Computing*, vol. 15, no. 1, pp. 69–76, 1984.

[65] W. Holmes, "A continuous cable method for determining the transient potential in passive dendritic trees of known geometry", *Biological Cybernetics*, vol. 55, no. 2, pp. 115–124, 1986.

[66] J. van Pelt, "A simple vector implementation of the laplace-transformed cable equations in passive dendritic trees", *Biological Cybernetics*, vol. 68, no. 1, pp. 15–21, 1992.

[67] M. L. Hines and N. T. Carnevale, "The NEURON simulation environment", *Neural Computation*, vol. 9, no. 6, pp. 1179–1209, 1997.

[68] T. Hollingworth and M. Berry, "Network analysis of dendritic fields of pyramidal cells in neocortex and Purkinje cells in the cerebellum of the rat", *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, vol. 270, no. 906, pp. 227–264, 1975.

[69] D. Shelton, "Membrane resistivity estimated for the Purkinje neuron by means of a passive computer model", *Neuroscience*, vol. 14, no. 1, pp. 111–131, 1985.

[70] A. Marasco, A. Limongiello, and M. Migliore, "Fast and accurate low-dimensional reduction of biophysically detailed neuron models", *Scientific Reports*, vol. 2, no. 1, pp. 1–7, 2012.

[71] M. Migliore, M. Ferrante, and G. A. Ascoli, "Signal propagation in oblique dendrites of CA1 pyramidal cells", *Journal of Neurophysiology*, vol. 94, no. 6, pp. 4145–4155, 2005.

[72] A. Marasco, A. Limongiello, and M. Migliore, "Using Strahler's analysis to reduce up to 200-fold the run time of realistic neuron models", *Scientific Reports*, vol. 3, no. 1, pp. 1–7, 2013.

[73] O. Amsalem, G. Eyal, N. Rogozinski, *et al.*, "An efficient analytical reduction of detailed nonlinear neuron models", *Nature Communications*, vol. 11, no. 1, pp. 1–13, 2020.

[74] W. A. Wybo, K. M. Stiefel, and B. Torben-Nielsen, "The Green's function formalism as a bridge between single-and multi-compartmental modeling", *Biological Cybernetics*, vol. 107, no. 6, pp. 685–694, 2013.

[75] W. A. Wybo, D. Boccalini, B. Torben-Nielsen, and M.-O. Gewaltig, "A sparse reformulation of the Green's function formalism allows efficient simulations of morphological neuron models", *Neural Computation*, vol. 27, no. 12, pp. 2587–2622, 2015.

[76] W. A. Wybo, B. Torben-Nielsen, T. Nevian, and M.-O. Gewaltig, "Electrical compartmentalization in neurons", *Cell Reports*, vol. 26, no. 7, pp. 1759–1773, 2019.

[77] W. A. Wybo, J. Jordan, B. Ellenberger, U. M. Mengual, T. Nevian, and W. Senn, "Data-driven reduction of dendritic morphologies with preserved dendro-somatic responses", *eLife*, vol. 10, e60936, 2021.

[78] R. Naud, B. Bathellier, and W. Gerstner, "Spike-timing prediction in cortical neurons with active dendrites", *Frontiers in Computational Neuroscience*, vol. 8, p. 90, 2014.

[79] R. FitzHugh, "Impulses and physiological states in theoretical models of nerve membrane", *Biophysical Journal*, vol. 1, no. 6, pp. 445–466, 1961.

[80] J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon", *Proceedings of the IRE*, vol. 50, pp. 2061–2070, 1962.

[81] J. L. Hindmarsh and R. Rose, "A model of neuronal bursting using three coupled first order differential equations", *Proceedings of the Royal society of London. Series B. Biological Sciences*, vol. 221, no. 1222, pp. 87–102, 1984.

[82] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity", *Journal of Neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.

[83] R. Brette, M. Rudolph, T. Carnevale, *et al.*, "Simulation of networks of spiking neurons: A review of tools and strategies", *Journal of Computational Neuroscience*, vol. 23, no. 3, pp. 349–398, 2007.

[84] G. Deco, V. K. Jirsa, P. A. Robinson, M. Breakspear, and K. Friston, "The dynamic brain: From spiking neurons to neural masses and cortical fields", *PLoS Computational Biology*, vol. 4, no. 8, e1000092, 2008.

[85] R. Stein, "A theoretical analysis of neuronal variability", *Biophysical Journal*, vol. 5, no. 2, pp. 173–194, 1965.

[86] M. Carlu, O. Chehab, L. Dalla Porta, *et al.*, "A mean-field approach to the dynamics of networks of complex neurons, from nonlinear integrate-and-fire to Hodgkin–Huxley models", *Journal of Neurophysiology*, vol. 123, no. 3, pp. 1042–1051, 2020.

[87] J. Baladron, D. Fasoli, O. Faugeras, and J. Touboul, "Mean-field description and propagation of chaos in networks of Hodgkin-Huxley and FitzHugh-Nagumo neurons", *The Journal of Mathematical Neuroscience*, vol. 2, no. 1, pp. 1–50, 2012.

[88] J. B. Pezoa, D. Fasoli, and O. Faugeras, "Three applications of GPU computing in neuroscience", *Computing in Science & Engineering*, vol. 14, no. 3, pp. 40–47, 2011.

[89] U. S. Python Software Foundation DE, *Version 3.6*.

[90] U. S. The MathWorks Inc. MA, *Matlab, version r2018b*.

[91] G. W. Lindsay, "Convolutional neural networks as a model of the visual system: Past, present, and future", *Journal of Cognitive Neuroscience*, vol. 33, no. 10, pp. 2017–2031, 2021.

[92] D. Beniaguev, I. Segev, and M. London, "Single cortical neurons as deep artificial neural networks", *Neuron*, vol. 109, no. 17, pp. 2727–2739, 2021.

[93] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series", *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10, p. 1995, 1995.

[94] B. Chang, M. Chen, E. Haber, and E. H. Chi, "AntisymmetricRNN: A dynamical system view on recurrent neural networks", *International Conference on Learning Representations*, 2019.

[95] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library", *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[96] G. Berkooz, P. Holmes, and J. L. Lumley, "The Proper Orthogonal Decomposition in the analysis of turbulent flows", *Annual Review of Fluid Mechanics*, vol. 25, no. 1, pp. 539–575, 1993.

[97] D. Kosambi, "Statistics in function space", *Journal of the Indian Mathematical Society*, vol. 7, pp. 76–88, 1943.

[98] K. Karhunen, *Ueber lineare Methoden in der Wahrscheinlichkeitsrechnung*. Soumalainen Tiedeakatemia, 1947.

[99] R. Everson and L. Sirovich, "Karhunen–Loeve procedure for gappy data", *JOSA A*, vol. 12, no. 8, pp. 1657–1664, 1995.

[100] K. Willcox, "Unsteady flow sensing and estimation via the gappy Proper Orthogonal Decomposition", *Computers & Fluids*, vol. 35, no. 2, pp. 208–226, 2006.

[101] M. Rathinam and L. R. Petzold, "A new look at Proper Orthogonal Decomposition", *SIAM Journal on Numerical Analysis*, vol. 41, no. 5, pp. 1893–1925, 2003.

[102] S. Glavaski, J. E. Marsden, and R. M. Murray, "Model reduction, centering, and the Karhunen-Loeve expansion", *Proceedings of the 37th IEEE Conference on Decision and Control*, IEEE, vol. 2, 1998, pp. 2071–2076.

[103] M. Fogleman, J. Lumley, D. Rempfer, and D. Haworth, "Application of the Proper Orthogonal Decomposition to datasets of internal combustion engine flows", *Journal of Turbulence*, vol. 5, no. 1, p. 023, 2004.

[104] S. Ravindran, "Adaptive reduced-order controllers for a thermal flow system using Proper Orthogonal Decomposition", *SIAM Journal on Scientific Computing*, vol. 23, no. 6, pp. 1924–1942, 2002.

[105] K. Kunisch and S. Volkwein, "Proper Orthogonal Decomposition for optimality systems", *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 42, no. 1, pp. 1–23, 2008.

[106] A. Alla, C. Gräßle, and M. Hinze, "A residual based snapshot location strategy for POD in distributed optimal control of linear parabolic equations", *IFAC-PapersOnLine*, vol. 49, no. 8, pp. 13–18, 2016.

[107] K. Kunisch and S. Volkwein, "Optimal snapshot location for computing POD basis functions", *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 44, no. 3, pp. 509–529, 2010.

[108] O. Lass and S. Volkwein, "Adaptive POD basis computation for parametrized nonlinear systems using optimal snapshot location", *Computational Optimization and Applications*, vol. 58, no. 3, pp. 645–677, 2014.

[109] T. Braconnier, M. Ferrier, J.-C. Jouhaud, M. Montagnac, and P. Sagaut, "Towards an adaptive POD/SVD surrogate model for aeronautic design", *Computers & Fluids*, vol. 40, no. 1, pp. 195–209, 2011.

[110] E. A. Christensen, M. Brøns, and J. N. Sørensen, "Evaluation of Proper Orthogonal Decomposition–based decomposition techniques applied to parameter-dependent nonturbulent flows", *SIAM Journal on Scientific Computing*, vol. 21, no. 4, pp. 1419–1434, 1999.

[111] K. Willcox, O. Ghattas, B. van Bloemen Waanders, and B. Bader, "An optimization framework for goal-oriented, model-based reduction of large-scale systems", *Proceedings of the 44th IEEE Conference on Decision and Control*, IEEE, 2005, pp. 2265–2271.

[112] F. Liang, R. Shi, and Q. Mo, "A split-and-merge approach for Singular Value Decomposition of large-scale matrices", *Statistics and its Interface*, vol. 9, no. 4, p. 453, 2016.

[113] M. Brand, "Fast low-rank modifications of the thin Singular Value Decomposition", *Linear Algebra and its Applications*, vol. 415, no. 1, pp. 20–30, 2006.

[114] C. A. Beattie, J. Borggaard, S. Gugercin, and T. Iliescu, "A domain decomposition approach to POD", *Proceedings of the 45th IEEE Conference on Decision and Control*, IEEE, 2006, pp. 6750–6756.

[115] Z. Wang, B. McBee, and T. Iliescu, "Approximate partitioned method of snapshots for POD", *Journal of Computational and Applied Mathematics*, vol. 307, pp. 374–384, 2016.

[116] C. Himpe, T. Leibner, and S. Rave, "Hierarchical approximate Proper Orthogonal Decomposition", *SIAM Journal on Scientific Computing*, vol. 40, no. 5, A3267–A3292, 2018.

[117] A. Varshney, S. Pitchaiah, and A. Armaou, "Feedback control of dissipative PDE systems using adaptive model reduction", *AIChE Journal*, vol. 55, no. 4, pp. 906–918, 2009.

[118] D. B. Pourkargar and A. Armaou, "Modification to adaptive model reduction for regulation of distributed parameter systems with fast transients", *AIChE Journal*, vol. 59, no. 12, pp. 4595–4611, 2013.

[119] M. Yang and A. Armaou, "Revisiting APOD accuracy for nonlinear control of transport reaction processes: A spatially discrete approach", *Chemical Engineering Science*, vol. 181, pp. 146–158, 2018.

[120] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, "An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations", *Comptes Rendus Mathematique*, vol. 339, no. 9, pp. 667–672, 2004.

[121] P. Astrid, S. Weiland, K. Willcox, and T. Backx, "Missing point estimation in models described by Proper Orthogonal Decomposition", *IEEE Transactions on Automatic Control*, vol. 53, no. 10, pp. 2237–2251, 2008.

[122] K. Carlberg, C. Bou-Mosleh, and C. Farhat, "Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations", *International Journal for Numerical Methods in Engineering*, vol. 86, no. 2, pp. 155–181, 2011.

[123] B. Peherstorfer, D. Butnaru, K. Willcox, and H.-J. Bungartz, "Localized Discrete Empirical Interpolation Method", *SIAM Journal on Scientific Computing*, vol. 36, no. 1, A168–A192, 2014.

[124] B. Peherstorfer and K. Willcox, "Online adaptive model reduction for nonlinear systems via low-rank updates", *SIAM Journal on Scientific Computing*, vol. 37, no. 4, A2123–A2150, 2015.

[125] Z. Drmac and S. Gugercin, "A new selection operator for the Discrete Empirical Interpolation Method—improved a priori error bound and extensions", *SIAM Journal on Scientific Computing*, vol. 38, no. 2, A631–A648, 2016.

[126] B. Peherstorfer, Z. Drmac, and S. Gugercin, "Stability of discrete empirical interpolation and gappy Proper Orthogonal Decomposition with randomized and deterministic sampling points", *SIAM Journal on Scientific Computing*, vol. 42, no. 5, A2837–A2864, 2020.

[127] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem, "The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows", *Journal of Computational Physics*, vol. 242, pp. 623–647, 2013.

[128] J. Hahn and T. F. Edgar, "An improved method for nonlinear model reduction using balancing of empirical gramians", *Computers & Chemical Engineering*, vol. 26, no. 10, pp. 1379–1397, 2002.

[129] N. C. Nguyen, A. T. Patera, and J. Peraire, "A 'best points' interpolation method for efficient approximation of parametrized functions", *International Journal for Numerical Methods in Engineering*, vol. 73, no. 4, pp. 521–543, 2008.

[130] P. Benner and P. Goyal, "Balanced truncation model order reduction for quadratic-bilinear control systems", *arXiv preprint arXiv:1705.00160*, 2017.

[131] S. D. Shank, V. Simoncini, and D. B. Szyld, "Efficient low-rank solution of generalized Lyapunov equations", *Numerische Mathematik*, vol. 134, no. 2, pp. 327–342, 2016.

[132] A. Varga, "Minimal realization procedures based on balancing and related techniques", *International Conference on Computer Aided Systems Theory*, Springer, 1991, pp. 733–761.

[133] N. Sarna and P. Benner, "Data-driven model order reduction for problems with parameter-dependent jump-discontinuities", *Computer Methods in Applied Mechanics and Engineering*, vol. 387, p. 114 168, 2021.

[134] E. Mazzi, A. S. Vincentelli, A. Balluchi, and A. Bicchi, "Hybrid system reduction", *Proceedings of the 47th IEEE Conference on Decision and Control*, IEEE, 2008, pp. 227–232.

[135] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges", *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018.

[136] M. Yang and A. Armaou, "Dissipative distributed parameter systems online reduction and control using DEIM/APOD combination", *2018 Annual American Control Conference (ACC)*, IEEE, 2018, pp. 2557–2562.

[137] P. Sanz Leon, S. A. Knock, M. M. Woodman, *et al.*, "The Virtual Brain: A simulator of primate brain network dynamics", *Frontiers in Neuroinformatics*, vol. 7, p. 10, 2013.

[138] M.-O. Gewaltig and M. Diesmann, "NEST (neural simulation tool)", *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.

[139] H. Yang and A. Veneziani, "Efficient estimation of cardiac conductivities via POD-DEIM model order reduction", *Applied Numerical Mathematics*, vol. 115, pp. 180–199, 2017.

[140] M. Mordhorst, T. Strecker, D. Wirtz, T. Heidlauf, and O. Röhrle, "POD-DEIM reduction of computational EMG models", *Journal of Computational Science*, vol. 19, pp. 86–96, 2017.

[141] P. Li, J. Lam, Z. Wang, and P. Date, "Positivity-preserving h∞ model reduction for positive systems", *Automatica*, vol. 47, no. 7, pp. 1504–1511, 2011.

[142] A. C. Antoulas, "A new result on passivity preserving model reduction", *Systems & Control Letters*, vol. 54, no. 4, pp. 361–374, 2005.

[143] D. C. Sorensen, "Passivity preserving model reduction via interpolation of spectral zeros", *Systems & Control Letters*, vol. 54, no. 4, pp. 347–360, 2005.

[144] U. B. Desai and D. Pal, "A realization approach to stochastic model reduction and balanced stochastic realizations", *Proceedings of the 21st IEEE Conference on Decision and Control*, IEEE, 1982, pp. 1105–1112.

[145] A. Doostan, R. G. Ghanem, and J. Red-Horse, "Stochastic model reduction for chaos representations", *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 37-40, pp. 3951–3966, 2007.

[146] C. Poussot-Vassal and P. Vuillemin, "Introduction to MORE: A MOdel REduction toolbox", *2012 IEEE International Conference on Control Applications*, IEEE, 2012, pp. 776–781.

[147] R. Milk, S. Rave, and F. Schindler, "pyMOR–generic algorithms and interfaces for model order reduction", *SIAM Journal on Scientific Computing*, vol. 38, no. 5, S194–S216, 2016.

[148] C. Himpe, "emgr—the EMpirical Gramian framework", *Algorithms*, vol. 11, no. 7, p. 91, 2018.

[149] M. Drohmann, B. Haasdonk, S. Kaulmann, and M. Ohlberger, "A software framework for reduced basis methods using DUNE-RB and RBmatlab", *Advances in DUNE*, Springer, 2012, pp. 77–88.

[150] P. Benner, T. Breiten, C. Hartmann, and B. Schmidt, "Model reduction of controlled Fokker–Planck and Liouville–von Neumann equations", *Journal of Computational Dynamics*, vol. 7, no. 1, p. 1, 2020.

[151] D. Amsallem and J. Nordström, "High-order accurate difference schemes for the Hodgkin–Huxley equations", *Journal of Computational Physics*, vol. 252, pp. 573–590, 2013.

[152] K. Fernando and H. Nicholson, "Singular perturbational model reduction of balanced systems", *IEEE Transactions on Automatic Control*, vol. 27, no. 2, pp. 466–468, 1982.

[153] C. Hartmann, B. Schafer-Bung, and A. Thons-Zueva, "Balanced averaging of bilinear systems with applications to stochastic control", *SIAM Journal on Control and Optimization*, vol. 51, no. 3, pp. 2356–2378, 2013.

[154] B. Kramer and K. E. Willcox, "Balanced truncation model reduction for lifted nonlinear systems", *arXiv preprint arXiv:1907.12084*, 2019.

[155] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker project", *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[156] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics", *Nature Communications*, vol. 9, no. 1, pp. 1–10, 2018.

[157] B. Peherstorfer and K. Willcox, "Data-driven operator inference for nonintrusive projection-based model reduction", *Computer Methods in Applied Mechanics and Engineering*, vol. 306, pp. 196–215, 2016.

[158] R. Swischuk, L. Mainini, B. Peherstorfer, and K. Willcox, "Projection-based model reduction: Formulations for physics-based machine learning", *Computers & Fluids*, vol. 179, pp. 704–717, 2019.

[159] S. Greydanus, M. Dzamba, and J. Yosinski, "Hamiltonian neural networks", *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[160] C. Rackauckas, Y. Ma, J. Martensen, *et al.*, "Universal differential equations for scientific machine learning", *arXiv preprint arXiv:2001.04385*, 2020.

[161] R. Chaudhuri, B. Gerçek, B. Pandey, A. Peyrache, and I. Fiete, "The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep", *Nature Neuroscience*, vol. 22, no. 9, pp. 1512–1520, 2019.

[162] U. Cohen, S. Chung, D. D. Lee, and H. Sompolinsky, "Separability and geometry of object manifolds in deep neural networks", *Nature Communications*, vol. 11, no. 1, pp. 1–13, 2020.

[163] F. Mastrogiuseppe and S. Ostojic, "Linking connectivity, dynamics, and computations in low-rank recurrent neural networks", *Neuron*, vol. 99, no. 3, pp. 609–623, 2018.

APPENDIX A LIFTING OF A NETWORK MODEL WITH SYNAPTIC PLASTICITY

Due to the nonlinear nature of mathematical models in computational neuroscience, MOR approaches have been limited to either linearization-based or empirical methods like the POD-DEIM. Recent results [10], [130] extend the Balanced Truncation (BT) and Iterative Rational Krylov Approximation (IRKA) to quadratic-bilinear (QB) systems that are a group of nonlinear systems with a structure as

$$\begin{aligned} x'(t) =& Ax(t) + H(x \otimes x) + \sum_{i=0}^{n} N_i x(t) u_i(t) + Bu(t) \\ y(t) =& Cx(t), \end{aligned} \tag{A.1}$$

where $H \in \mathbb{R}^{n \times n^2}$ is a matrix of coefficients of quadratic nonlinearities and $N_i \in \mathbb{R}^{n \times n}$ is a matrix of bilinear coefficients. For these methods, the outputs $y(t) = Cx(t)$ must be linear in $x(t)$. While models of this form are scarce in neuroscience, it may be possible to apply a *lifting transformation* that lifts a general nonlinear system into an equivalent QB form. Notice that the transformation is not guaranteed to exist and may not be unique if it exists.

The following work is not published. As a case study, we consider the Hindmarsh-Rose (HR) bursting neuron model [81] and the Tsodyks-Markram (TM) short-term synaptic plasticity model [34] that form a large-scale network. The HR single neuron model is

$$\begin{aligned} x'(t) &= y - ax^3 + bx^2 - z + SE + I_{\text{ext}}(t) \\ y'(t) &= c - dx^2 - y \\ z'(t) &= rsx - rsx_{\text{R}} - rz, \end{aligned} \tag{A.2}$$

where x describes neuronal voltage and y, z are phenomenological adaptation variables, $I_{\text{ext}}(t)$ is injected current, E is synaptic current and S is a matrix of synaptic

weights. Parameters $a, b, c, dr, s, x_{\mathrm{R}}$ are constant. The TM model is

$$\begin{aligned} E'(t) &= -E\tau_{\mathrm{TM}}^{-1} + Aux_{\mathrm{TM}}f(v_{\mathrm{pre}}) \\ x'_{\mathrm{TM}}(t) &= \tau_D^{-1}(1-x) - ux_{\mathrm{TM}}f(v_{\mathrm{pre}}) \\ u'(t) &= -\tau_F^{-1}(u-U) + U(1-u)f(v_{\mathrm{pre}}), \end{aligned} \tag{A.3}$$

where x_{TM} is portion of available synaptic resources, u is synaptic release probability, $f(\cdot)$ maps the membrane voltage to a positive value mimicking a synaptic release event and $\tau_{\mathrm{TM}}, \tau_F, \tau_D, A$ are constants. We choose $f(\cdot) \equiv \exp^{(x-x_{\max})}$ with $x_{\max}$ the value of an action potential at its peak.

Next, we introduce auxiliary variables

$$\begin{aligned} p &= x^2 \\ h &= xp = x^3 \\ q &= e^{x-x_{\max}} \\ l &= uq, \end{aligned} \tag{A.4}$$

and following [24] calculate their time derivatives to obtain a QB system

$$\begin{aligned} x'(t) &= y - ah + bp - z + SE + u(t) \\ y'(t) &= c - dp - y \\ z'(t) &= rsx - rsx_{\mathrm{R}} - rz \\ E'(t) &= -E/\tau_{TM} + Alx_{TM} \\ x'_{TM,i}(t) &= (1-x_{\mathrm{TM}})/\tau_D - x_{\mathrm{TM}}l \\ u'(t) &= U(q-l) - (u-U)/\tau_F \\ p'(t) &= 2xx' \\ &= 2(xy - ap^2 + bh - zx + SEx + u(t)x) \\ h'(t) &= 3x^2x' \\ &= 3(py - aph + bp^2 - zp + SEp + u(t)p) \\ q'(t) &= qx' \\ &= yq - ahq + bpq - zq + SEq + u(t)q \\ l'(t) &= qu' + uq' \\ &= lx' + U(q-l)q - (l-Uq)\tau_F^{-1} \end{aligned} \tag{A.5}$$

that is equivalent to the original nonlinear model in the original state variables, but also includes ODEs for the lifted variables. Additionally, the inputs and outputs of the lifted system have not changed, so the lifted system admits the original controls and readouts. In [24], [25] it is also discussed how to arrive at a system of differential algebraic equations, however these systems are more challenging to simulate and reduce than ODEs. The dimension of the lifted system has grown, but the QB format yields itself to model reduction algorithms that were not applicable in the original nonlinear form, such as [10], [130].

The system matrix A of the lifted model has zero eigenvalues. This means that BT methods may not yield reduced models if applied to this system directly. A solution is to apply artifical stabilization to the state matrix. We may write

$$A_s = A - \sigma I, \tag{A.6}$$

where σ is a small constant. By subtracting from the diagonal of A, we create a new matrix A_s that has eigenvalues shifted towards negative real parts. After obtaining the reduction matrices using A_s, the actual reduced system may be computed from the original system matrix A. This introduces minor error into the reduced model, but we can now apply BT as in [130]. With this result, we are a step closer to reducing biophysical neuronal networks in a data independent manner.

PUBLICATIONS

PUBLICATION
I

Order reduction for a signaling pathway model of neuronal synaptic plasticity

M. Lehtimäki, L. Paunonen, S. Pohjolainen, and M.-L. Linne

IFAC-PapersOnLine, 2017 20th International Federation of Automatic Control (IFAC) World Congress, 2017, pp. 7687–7692
DOI: https://doi.org/10.1016/j.ifacol.2017.08.1143

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online
CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 7687–7692

Order reduction for a signaling pathway model of neuronal synaptic plasticity

Mikko Lehtimäki [*,***] **Lassi Paunonen** [**] **Seppo Pohjolainen** [**] **Marja-Leena Linne** [*,***]

[*] *Computational Neuroscience Group, BioMediTech Institute and Faculty of Biomedical Sciences and Engineering, Tampere University of Technology, FI-33720 Tampere, Finland (email: lehtimaki.mikko@gmail.com, marja-leena.linne@tut.fi)*
[**] *Mathematics, Faculty of Natural Sciences, Tampere University of Technology, Tampere, Finland, FI-33720 Tampere, Finland (email: lassi.paunonen@tut.fi, seppo.pohjolainen@tut.fi)*
[***] *Department of Signal Processing, Tampere University of Technology, FI-33720 Tampere, Finland*

Abstract: In this study a nonlinear mathematical model of plasticity in the brain is reduced using the Proper Orthogonal Decomposition and Discrete Empirical Interpolation Method. Such methods are remarkably useful for connecting reduced small scale models via the inputs and outputs to form optimally performing large scale models. Novel results were obtained as mathematical model order reduction has not been applied in neuroscience without linearization of the mathematical model and never to the model presented here. The reduced order model consumes considerably less computational resources than the original while maintaining a low root mean square error between the original and reduced model.

Keywords: model reduction, nonlinear models, Proper Orthogonal Decomposition, Discrete Empirical Interpolation Method, cell signaling, synaptic plasticity

1. INTRODUCTION

Dimensionality reduction is a commonly used method in engineering sciences, such as control theory, in improving computational efficiency of simulations of complex nonlinear mathematical models. In the field of neuroscience, there is a great demand to incorporate molecular and cellular level detail in large-scale models of the brain in order to reproduce phenomena such as learning and behavior. This cannot be achieved with the computing power available today, since the detailed models are complex and often computationally too demanding for large-scale network or system level simulations.

In the field of systems biology, models are typically simplified by completely eliminating variables, such as molecular entities, from the system, and making assumptions of the system behavior, for example regarding the steady state of the chemical reactions. However, this approach is not suitable for the current trend in neuroscience, in which multiple physical scales of the brain are incorporated in simulations and the consequent analysis of neural phenomena. Instead comprehensive models with full system dynamics are needed in order to increase understanding of different actors in one brain area.

The information loss typically induced by eliminating variables of the system can be avoided by mathematical reduction methods that strive to approximate the entire system with a smaller number of dimensions compared to the original system. Here we demonstrate the effectiveness of mathematical model order reduction methods in approximating the behavior of all the variables in the original system by simulating a model with a radically reduced dimension.

In this study, mathematical model reduction is applied in the context of an experimentally verified signaling pathway model of plasticity. This nonlinear chemical equation based data-driven model was published in Kim et al. (2013), and it describes the biochemical calcium signaling steps required for plasticity, and hence for learning, in the subcortical area of the brain. In addition to nonlinear characteristics, the model includes time-dependent terms, which pose an additional challenge both computational efficiency and reduction wise. The chosen biophysical model is one of the most comprehensive models out of those that are currently able to explain aspects of plasticity on the molecular level with chemical interactions and the law of mass action. The original model is too detailed for utilization in large-scale network simulations, which serves as motivation for the present study. Moreover, with this case study the aim is to demonstrate that the behavior of the model can be analysed faster yet with satisfactory accuracy by using a reduced order model.

The model order reduction method employed in this study is Proper Orthogonal Decomposition with Discrete Empirical Interpolation Method (POD+DEIM), a subspace projection method for reducing the dimensionality of nonlinear systems. By applying these methods, the simulation time of the plasticity model is radically compressed al-

10.1016/j.ifacol.2017.08.1143

though approximation errors are present if the model is reviewed on large time scales. The tolerated amount of approximation error depends on the final application of the model. Based on these promising results, POD+DEIM are recommended for dimensionality reduction in computational neuroscience.

Algorithms to achieve the elimination-type reduction for nonlinear neuronal models have been proposed for instance in Woo et al. (2005), Sorensen and DeWeerth (2006) and later in Shin et al. (2009). However the studies rely on several assumptions of the model structure and are only suitable in very specific scenarios. Recently, a variable elimination strategy was used to reduce a model of astrocyte metabolism in Diekman et al. (2013). Additionally, mathematical reduction of neuronal dendrite using a linearization approach has been performed in Kellems et al. (2009) and a nonlinear model discretized from partial differential equations has been reduced in Du et al. (2014).

An empirical interpolation method for reducing the complexity of nonlinear functions was first proposed by Barrault et al. (2004). The discrete version Discrete Empirical Interpolation Method (DEIM) was then introduced in Chaturantabut and Sorensen (2010). The previous five years have seen DEIM developed further with localized, adaptive and stability conserving variants in Peherstorfer et al. (2014); Peherstorfer and Willcox (2015); Amsallem and Nordström (2016) as well as a monotonicity preserving variation for reaction diffusion systems in Chaturantabut (2016). DEIM is a method that complements POD by reducing the nonlinear term so that together the two arrive at a reduced model which no longer depends on the original dimension of the system. Alternatively, DEIM can be used for standalone reduction of nonlinear functions.

2. PLASTICITY MODEL

We study a mathematical model of signaling pathways in striatal synaptic plasticity by Kim et al. (2013). The model is specific for the basal ganglia area of the brain and it describes how certain molecules in intercellular information transfer points of neurons, synapses, are responsible for plasticity, which is presumably a prerequisite for learning in the brain. It is a biophysicochemical model that is based on experimental data. Originally the model was employed in studying the effects of different stimuli to the synapse and how they could direct plasticity. Additionally, the predictions from the model have been verified experimentally and the model itself is based on validated experimental data.

The model is based on chemical reactions of the molecules in the synapse. The stoichiometric equations obey the law of mass action, which leads to a deterministic system of ordinary differential equations. Our implementation of the model contains $n = 44$ ordinary differential equations.

The model has two external stimulus variables, calcium ion (Ca) and neurotransmitter glutamate (Glu). The state-space model is of the form

$$\begin{aligned}\dot{x}(t) =& A(t)x(t) + F(x(t)) + B \cdot Glu(t)\\ =& (A_0 + A_1 Ca(t) + A_2 Ca(t)^2 + A_3 Glu(t))x(t) + \\ & F(x(t)) + B \cdot Glu(t)\end{aligned} \quad (1)$$

and the system is nonautonomous due to a Ca stimulus being part of $A(t)$. In our simulations, both Ca and Glu stimuli are fixed functions in the model reduction and testing phases. If Glu and Ca are considered inputs to the system, the result is a nonlinear control system, which additionally has bilinear characteristics. The five first equations of the model are

$$\begin{aligned}\dot{x}_1 =& k_{prodAG_c} \cdot x_4 - k_{degAG_f} \cdot x_1\\ \dot{x}_2 =& k_{PMCA_c} \cdot x_{15} + k_{NCX_c} \cdot x_{11} - k_{Leak_f} \cdot x_2 \cdot x_{36} + \\ & k_{Leak_b} \cdot x_{10}\\ \dot{x}_3 =& k_{buffer_f} \cdot Ca(t) \cdot x_{19} - k_{buffer_b} \cdot x_3\\ \dot{x}_4 =& k_{prodAG_f} \cdot x_{21} \cdot x_7 - k_{prodAG_b} \cdot x_4 - k_{prodAG_c} \cdot x_4\\ \dot{x}_5 =& k_{DAG_3c} \cdot x_8 - k_{DAG_4f} \cdot x_5,\end{aligned} \quad (2)$$

and they illustrate the nonlinearity of the system in equation of $\dot{x}_2$, and the time-dependence of the system in the equation of $\dot{x}_3$. In Equation (2), terms k_n represent constants and x_n chemical species. This nonlinear system has a sparse linear part and includes a time dependent stimulus. In the numerical implementation of the model, the linear coefficients, nonlinear function and external inputs of the system are separated.

For the following analysis five biologially interesting species included in the model were chosen as outputs of the system to be studied in more detail. These were 2-arachidonoylglycerol ($\mathrm{Ag_{post}}$), external calcium ($\mathrm{Ca_{ext}}$), diacylglycerol ($\mathrm{DAG_{post}}$), G protein with α, β and γ subunits ($\mathrm{Gabg_{post}}$) and phospholipase C ($\mathrm{PLC_{post}}$). In the present model these species are also included as state variables. Their behavior is significant as these substances can connect the current model to a larger, even more detailed model and they are known to be active influencers in the two forms of plasticity, LTP (long term potentiation) and LTD (long term depression) (see Manninen et al. (2010); Hellgren-Kotaleski and Blackwell (2010)).

3. MODEL REDUCTION USING POD AND DEIM

In this section we outline the Proper Orthogonal Decomposition (POD) (see Lumley et al. (1993); Sirovich (1987); Kellems et al. (2009, 2010); Benner et al. (2015)) and Discrete Empirical Interpolation Method (DEIM) (see Chaturantabut and Sorensen (2010)) that are used to reduce the order of the quadratic and nonautonomous model discussed in Section 2. POD is a well-known method that is used in model reduction of various types of differential equations, partial differential equations and dynamical systems.

The underlying idea of the POD method is to project the system (1) onto a subspace so that the reduced system approximates the dynamical behaviour of (1) in the best possible way in the sense of least squares. The POD reduction procedure is begun by simulating the full system (1) and choosing "snapshots" $x(t_j)$ of the state of the system at equally spaced time instances $(t_j)_{j=1}^{N_s} \subset [0, T]$ where $T > 0$ is the length of the time interval (see Sirovich (1987)). The POD reduction replaces the system (1) with an approximate system on the space spanned by the first $1 \leq k \leq n$ singular vectors of the matrix $S = [x(t_1), \ldots, x(t_{N_s})] \in \mathbb{R}^{n \times N_s}$. In particular, if

$S = V\Sigma W^*$ is the singular value decomposition of S and V_k consists of the first k columns of V, then the POD reduced order model of (1) has state $x_k(t) = V_k^* x(t)$, and its dynamics are determined by the Galerkin projection

$$\dot{x}_k(t) = V_k^* A(t) V_k x_k(t) + V_k^* F(V_k x_k(t)) + V_k^* B u(t), \quad (3)$$

where $u(t)$ is the input vector.

The main drawback of the reduced model (3) in terms of computational efficiency is that the function $F(V_k x_k(t))$ appearing in the nonlinear term needs to be evaluated for a full-sized vector $V_k x_k(t) \in \mathbb{R}^{n\times n}$ with $n = 44$. The computational cost of evaluating the nonlinear term can be reduced by approximating the function F using the DEIM procedure.

DEIM extends POD with an interpolation step for nonlinear terms of the model, while also maintaining a subspace projection approach. The construction of the DEIM approximation begins with the construction of the so-called *DEIM modes*, vectors $U_m = [u_1, ..., u_m] \in \mathbb{R}^{n\times m}$. The matrix U_m consists of the first $m \leq n$ left-singular vectors of the matrix $[F(x(t_0)), \ldots, F(x(t_{N_s}))]$ (these columns of the DEIM projection matrix can be collected during the generation of the snapshots in the POD reduction process to minimize offline computational burden of DEIM).

In the second step of the DEIM procedure we define

$$P = [e_{\wp_1}, ..., e_{\wp_m}] \in \mathbb{R}^{n\times m}, \quad (4)$$

where $e_{\wp_j} \in \mathbb{R}^n$ are the columns of the identity matrix $I \in \mathbb{R}^{n\times n}$ and where $\{\wp_1, ..., \wp_m\}$ is a set of *interpolation indices*. The indices $\{\wp_1, ..., \wp_m\}$ are chosen based on the columns of U_m using the algorithm presented in (Chaturantabut and Sorensen, 2010, Algorithm 1). By construction the matrix $P^T U_m \in \mathbb{R}^{m\times m}$ is nonsingular.

The function $F : \mathbb{R}^n \to \mathbb{R}^n$ is of the form $F(x) = [f_1(x), \ldots, f_n(x)]^T$, where $f_j : \mathbb{R}^n \to \mathbb{R}$ for $j \in \{1, \ldots, n\}$. We define $F_m : \mathbb{R}^n \to \mathbb{R}^m$ such that $F_m(x) = [f_{\wp_1}(x), \ldots, f_{\wp_m}(x)]^T$. Finally, the DEIM approximation of the nonlinear term $F(V_k x_k(t))$ in the POD approximation is given by

$$F_{(k,m)}(V_k x_k(t)) = V_k^T U_m (P^T U_m)^{-1} F_m(V_k x_k(t)), \quad (5)$$

where the matrix $V_k^T U_m (P^T U_m)^{-1} \in \mathbb{R}^{k\times m}$ can be computed in the offline stage. The computational savings of the DEIM approximation result from the fact that in the function $F_{(k,m)}(\cdot)$ we only need to evaluate m component functions of the original nonlinear function $F(\cdot)$.

The final reduced order form of the system (1) becomes

$$\dot{x}_k(t) = A_k(t) x_k(t) + F_{(k,m)}(V_k x(t)) + B_k u(t),$$

where $x_k(t) = V_k^* x(t)$, $A_k(t) = V_k^* A(t) V_k$, $B_k = V_k^* B$, and $F_{(k,m)}$ is defined in (5).

The orders k and m of the POD and DEIM model reductions can be chosen independently of each other.

4. SIMULATION RESULTS

In order to compare the original model versus POD+DEIM reduced models the simulation speed and error of several subspace dimensions were measured. The original and reduced ordinary differential equation systems were simulated in Matlab for time span $t = [0, 10000]$ using the variable time step ODE15S solver for stiff differential equations. For each POD dimension $k = 2 : 2 : 40$ (Matlab notation), DEIM dimension $m = 5 : 5 : 30$ reduced models were calculated. For each combination, 20 simulations were performed and their average computation times and system solutions at each time step were stored.

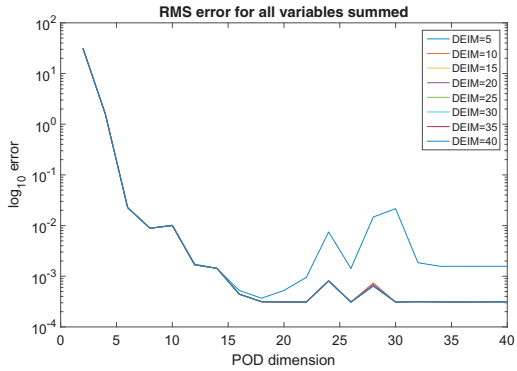

Fig. 1. Root mean square error between the original model and reduced order solutions. X-axis shows POD dimension and y-axis the error on logarithmic scale. Each distinctly colored plot corresponds to different dimensions used in DEIM.

Figure 1 shows the root mean square (RMS) error between the full dimension model and different reduced models of each POD+DEIM combination. The y-axis contains the error values on a logarithmic scale, while x-axis indicates the number of POD dimensions. Each line in the plot corresponds to a DEIM dimension. RMS error was calculated by

$$e_{RMS} = \sqrt{\frac{1}{k}\sum_{n=1}^{k}(Y - \tilde{Y})^2} \quad (6)$$

where Y is the matrix of solutions of the original system, $\tilde{Y} = V_k Y_{reduced}$ is the matrix of reduced order simulation results transformed back into the original space and k is the number of elements in the matrices.

From Figure 1 it is seen that regardless of the DEIM mode, or the nonlinear dimensionality, the error decays exponentially until POD dimension 15 is reached. This suggests that more than 15 POD dimensions is not necessary beneficial for a reduced order model, since the accuracy will not improve with additional dimensions. Depending on the application, as little as five to ten dimensions could be sufficient for simulating this model while keeping the error tolerable. Moreover the RMS error is seen to not depend radically on the DEIM dimension. Increasing the DEIM modes from 5 to 10 reduces the error if the POD mode is already over 15. This suggests that the linear part of the model that is reduced with POD is dominant in terms of approximation error and that the interpolation approach to reducing the nonlinear complexity is effective.

Figure 2 displays the relative computational advantage gained from the reduced model in terms of simulation speed. In the figure, the simulation time of the original full dimension model is plotted as a straight red line. From Figure 2 it is seen that the simulation time is

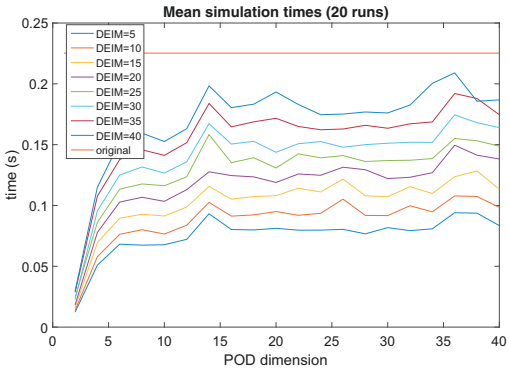

Fig. 2. Mean simulation times of 20 executions of each POD+DEIM reduced model compared to the original model, plotted as a straight red line. The y-axis shows the simulation time in seconds, and x-axis shows the POD dimension. Each colored plot corresponds to a DEIM dimension. Simulation time interval was $t = [0, 10000]$.

approximately halved by using 20 DEIM modes, which corresponds to halving the original dimension (44) of the nonlinear term. The simulation times depend on POD reduction only when less than 15 modes are chosen. In summary, for this model, the nonlinear term is the largest computational burden, since reducing it has the largest effect on simulation times.

Figure 3 displays how the dynamics of selected output species given by the reduced order model (red line) compared to the original model (blue line) in the first 5000 seconds. Here y-axis shows the concentration of each substance and x-axis the time. Analyzing the solutions in this format is important, for the absolute error measured earlier does not take into account how the error as a function of time is affected by dimension reduction. In the context of neural models, it is important that the dynamics are preserved. For example, information transmission via calcium signaling between astrocytes and neurons has been demonstrated to be amplitude and frequency modulated in Wade et al. (2011), so even a slight defect might cause the higher level behavior of the model to change.

The concentrations of molecular species participating in specific signaling pathways are difficult, or sometimes impossible to measure, which emphasizes the importance of modeling the dynamics of signaling pathways. The experimental challenge is related to measurement techniques: to this day there is no direct way to estimate the exact concentrations of molecular entities in as a function of time nor the possible variability of molecular entities. The changes in concentrations are measured, as an example, using fluorescent Ca^{2+} indicators which do not directly give absolute concentrations. For some of the variables described in this study, such as the calcium, some estimates of measures can however be obtained from theoretical studies. The average volume of a spine is 1 fl and resting level concentration of Ca^{2+} is 0.1 μM, which means that there are about 60 calcium ions in one spine. Moreover, it has been estimated that when 100 calcium ions enter into the spine head, it increases the calcium concentration in the spine from 100 to 300 nM (depending on the volume of the spine), which corresponds to the physiological range of increases (see Majewska et al. (2000); Holcman et al. (2004)).

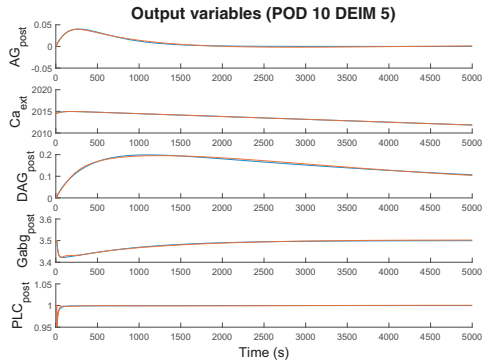

Fig. 3. Solutions of the dynamics of the selected biologically interesting output variables. Five species were tracked and their behavior plotted as a function of time. The y-axis displays the concentration of each ion/molecule. Blue line is the original model, and red is the approximation from the reduced order model with 10 POD and 5 DEIM modes.

However, the reduced order is not able to predict the behavior of the system when a simulation time longer than the training time is used. In order to test whether a low number of POD modes would be able to perform a near-correct approximation for a very long time span if the snapshots were also taken from a prolonged simulation, new reduced models were generated. The employed simulation time was $5 * 10^9$ seconds. Figure 4 shows the approximation with 10 POD and 5 DEIM modes and it is seen that $Gabg_{post}$ and PLC_{post} significantly different from the correct solution. However, a very good approximation was obtained with 30 POD and 10 DEIM modes, which is seen in Figure 5, while almost maintaining a simulation time of one third of the original model. The reduced model has gained more pronounced oscillations, although their amplitude is extremely low. Moreover, the steady state concentrations are physiologically very close to the original and in an acceptable range considering the inherent errors a deterministic model such as the one studied here always has. Whether the errors seen here would affect the behavior of a multi-scale model remains a question for another study.

The magnitude of the errors with a long simulation time was further studied using the absolute and relative errors between the original model and the 30 POD 10 DEIM reduced model. The errors are visualized in Figure 6 and Figure 7. The absolute errors are small, with the size being less than 10^{-6} for all species except calcium, where the range is approximately 10^{-2}. The relative errors for PLC and $Gabg$ confirm that the observed variation is extremely small. The relative errors for AG, Ca_{ext} and DAG on the other hand display oscillations in the reduced order model and additionally, are in a different magnitude than

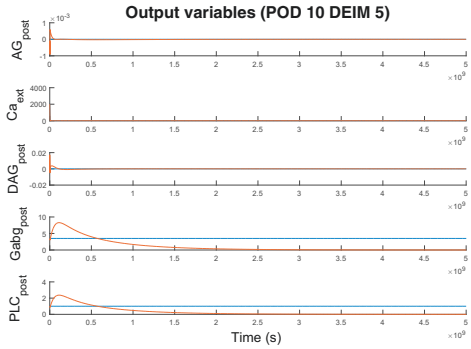

Fig. 4. Behavior the reduced order model in a long duration simulation using ten POD and five DEIM modes. Blue line is the original model and red is the reduced model for each output variable. The simulation time was $5 * 10^9$ seconds.

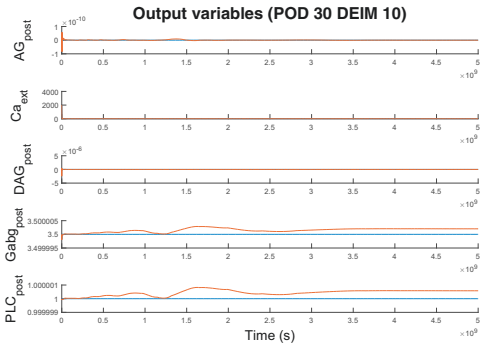

Fig. 5. Behavior the reduced order model in a long duration simulation using 30 POD and 10 DEIM modes. Blue line is the original model and red is the reduced model for each output variable. The simulation time was $5 * 10^9$ seconds.

the two other output species, achieving 10^8. However, the magnitude can be explained by numerical inaccuracy of the denominator, since the true concentration reaches zero at all points where a high error is seen, except at the very beginning of the simulation where stimulus is applied. Moreover, the relative error indicates that these three species correctly predict the steady state concentration, eventually, seen as the error degrading to zero.

To conclude, good results are achieved when the reduced order model is trained in a matching time interval to the final use case. The greatest challenge for the present method is generalizing the reduced model to longer time intervals. This issue is possibly solved by more recent improvements of the DEIM algorithm introduced in Peherstorfer et al. (2014) and Peherstorfer and Willcox (2015).

5. CONCLUSIONS

In this study Proper Orthogonal Decomposition and Discrete Empirical Interpolation Method (POD+DEIM) was

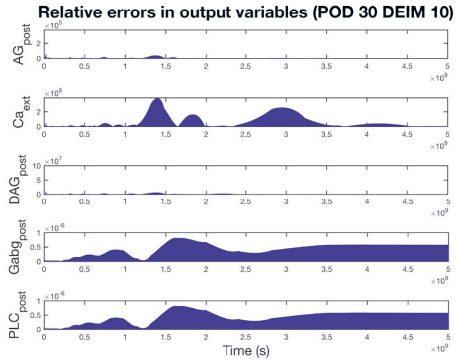

Fig. 6. Relative error between the 30 POD and 10 DEIM modes reduced model and the original model at every 10^6 seconds when simulated for $5 * 10^9$ seconds.

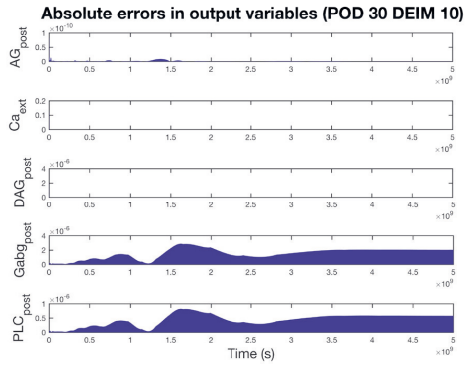

Fig. 7. Absolute error between the 30 POD and 10 DEIM modes reduced model and the original model at every 10^6 seconds when simulated for $5 * 10^9$ seconds.

applied to a data-driven biological model of plasticity in the brain. Five important molecules and ions were chosen for analysis, since these species have the greatest potential to link the model to a larger system comprising more brain areas and features of the multi-scale neural system.

Model order reduction is an essential process for improving the scale and quality of future computational models of the brain. Moreover, reduction methods will become increasingly important when models representing other mammalian species, such as rat and mouse, will be extended into human models. Although many methods of model reduction exist, subspace projection methods show most promise for they can be automatically applied, have adjustable error bounds and scale to virtually any size of system without compromising variables in the model. Additionally, they are applicable to nonlinear systems, either directly or via linearization, which greatly increases their applicability to complex models in neuroscience.

Model reduction with POD+DEIM was found to significantly reduce the simulation time. An additional benefit is that the approximation can be tuned by adjusting the POD and DEIM dimensionality independently. However,

the reduced order model did not achieve a perfect reproduction of the solutions of the original model in long time intervals and the steady states also had slight deviations from the original model. Whether the observed error is tolerable depends on the final purpose of the model.

DEIM has already been developed further and future studies are needed to test the effectiveness of these new variations of the algorithm. The recently published Localized DEIM looks extremely promising for maintaining a low number of approximation modes for widely varying model parameters, given that the conditions were present in the offline training phase of POD+DEIM (Peherstorfer et al. (2014)). Moreover, the adaptive version ADEIM is able to react to unanticipated behavior on the online stage of a simulation by efficiently querying the original system (Peherstorfer and Willcox (2015)). All in all, subspace projection methods seem suitable for reducing the dimensionality of signaling pathway models in neuroscience.

ACKNOWLEDGEMENTS

This project has received funding awarded to M.-L.L from the European Union Seventh Framework Programme (FP7) under grant agreement no 604102 (HBP RUP) and the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 720270 (HBP SGA1).

L.P. is funded by the Academy of Finland, Grant No. 298182.

REFERENCES

Amsallem, D. and Nordström, J. (2016). Energy stable model reduction of neurons by nonnegative discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 38(2), B297–B326. doi:10.1137/15M1013870.

Barrault, M., Maday, Y., Nguyen, N., and Patera, A. (2004). An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematicue*, 339(9), 667–672. doi:10.1016/j.crma.2004.08.006.

Benner, P., Gugercin, S., and Willcox, K. (2015). A survey of projection-based model reduction methods for parametric dynamical systems. *Society for Industrial and Applied Mathematics Review*, 57(4), 483–531. doi:10.1137/130932715.

Chaturantabut, S. (2016). Nonlinear reduced-order modeling with monotonicity property. In *American Institute of Physics Conference Proceedings*, 2849–2856. AIP Publishing. doi:10.1063/1.4965383.

Chaturantabut, S. and Sorensen, D. (2010). Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal of Scientific Computing*, 32(5), 2737–2764. doi:10.1137/090766498.

Diekman, C., Fall, C., Lechleiter, J., and Terman, D. (2013). Modeling the neuroprotective role of enhanced astrocyte mitochondrial metabolism during stroke. *Biophysical Journal*, 104(8), 1635–1838.

Du, B., Sorensen, D., and Cox, S. (2014). Model reduction of strong-weak neurons. *Frontiers in Computational Neuroscience*, 8(164).

Hellgren-Kotaleski, J. and Blackwell, K. (2010). Modelling the molecular mechanisms of synaptic plasticity using systems biology approaches. *Nature Reviews Neuroscience*, 11, 239–251. doi:10.1038/nrn2807.

Holcman, D., Schuss, Z., and Korkotian, E. (2004). Calcium dynamics in dendritic spines and spine motility. *Biophysical Journal*, 87(1), 81–91. doi:10.1529/biophysj.103.035972.

Kellems, A., Chaturantabut, S., Sorensen, D., and Cox, S. (2010). Morphologically accurate reduced order modeling of spiking neurons. *Journal of Computational Neuroscience*, 28(3). doi:10.1007/s10827-010-0229-4.

Kellems, A., Roos, D., Xioa, N., and Cox, S. (2009). Low-dimensional, morphologically accurate models of subthreshold membrane potential. *Journal of Computational Neuroscience*, 27(2), 161–176.

Kim, B., Hawes, S., Gillani, F., Wallace, L., and Blackwell, K. (2013). Signaling pathways involved in striatal synaptic plasticity are sensitive to temporal pattern and exhibit spatial specificity. *PLoS Computational Biology*, 9(3). doi:10.1371/journal.pcbi.1002953.

Lumley, J., Berkooz, G., and Holmes, P. (1993). The Proper Orthogonal Decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25, 539–575. doi:10.1146/annurev.fl.25.010193.002543.

Majewska, A., Brown, E., Ross, J., and Yuste, R. (2000). Mechanisms of calcium decay kinetics in hippocampal spines: role of spine calcium pumps and calcium diffusion through the spine neck in biochemical compartmentalization. *Journal of Neuroscience*, 20(5), 1722–1734.

Manninen, T., Hituri, K., Hellgren-Kotaleski, J., Blackwell, K., and Linne, M. (2010). Postsynaptic signal transduction models for long-term potentiation and depression. *Frontiers in Computational Neuroscience*, 4(152), 1–29. doi:10.3389/fncom.2010.00152.

Peherstorfer, B., Butnaru, D., Willcox, K., and Bungartz, H. (2014). Localized discrete empirical interpolation method. *SIAM Journal of Scientific Computing*, 36(1). doi:10.1137/130924408.

Peherstorfer, B. and Willcox, K. (2015). Online adaptive model reduction for nonlinear systems via low-rank updates. *SIAM Journal of Scientific Computing*, 37(4). doi:10.1137/140989169.

Shin, D., Yang, D., and Choi, J. (2009). On the use of pseudo-spectral method in model reduction and simulation of active dendrites. *Computers in Biology and Medicine*, 39(4), 340–345.

Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. I-III. *Quarterly of Applied Mathematics*, 45(3), 561–590.

Sorensen, M. and DeWeerth, S. (2006). An algorithmic method for reducing conductance-based neuron models. *Biological Cybernetics*, 95(2), 185–192.

Wade, J.J., McDaid, L.J., Harkin, J., Crunelli, V., and Kelso, J.A.S. (2011). Bidirectional coupling between astrocytes and neurons mediates learning and dynamic coordination in the brain: A multiple modeling approach. *PLOS ONE*, 6(12), 1–24. doi:10.1371/journal.pone.0029445.

Woo, B., Shin, D., Yang, D., and Choi, J. (2005). Reduced model and simulation of neuron with passive dendritic cable: an eigenfunction expansion approach. *Journal of Computational Neuroscience*, 19(3), 379–397.

PUBLICATION
II

Projection-based order reduction of a nonlinear biophysical neuronal network model

M. Lehtimäki, L. Paunonen, and M.-L. Linne

Proceedings of the 2019 IEEE 58th Conference on Decision and Control (CDC), 2019, pp. 1–6
DOI: https://doi.org/10.1109/CDC40024.2019.9029510

Projection-based order reduction of a nonlinear biophysical neuronal network model

Mikko Lehtimäki, Lassi Paunonen and Marja-Leena Linne

Abstract— **In this study mathematical model order reduction is applied to a nonlinear model of a network of biophysically realistic heterogeneous neurons. The neuron model describes a pyramidal cell in the hippocampal CA3 area of the brain and includes a state-triggered jump condition. The network displays synchronized firing of action potentials (spikes), a fundamental phenomenon of sensory information processing in the brain. Simulation of the system is computationally expensive, which limits network size and hence biological realism. We reduce the network using advanced variations of Proper Orthogonal Decomposition and Discrete Empirical Interpolation Method. The reduced models should recreate the original spiking activity. We show that reduction methods with online adaptivity achieve the most accurate reduction results. Some of the reduced models consume less computational resources than the original, at the cost of changes in population activity of the tested network model.**

I. INTRODUCTION

In the field of neuroscience, there is a great demand to incorporate molecular and cellular level detail in large-scale models of the brain in order to recreate phenomena such as learning and behavior [9]. This cannot be achieved with the computing power available today, since detailed models are complex and often computationally too demanding for large-scale network or system level simulations. Model order reduction (MOR) is a mathematical method for improving computational efficiency of simulations of mathematical models. However, in computational neuroscience the use of MOR is not common. Instead, efficient models are typically derived by eliminating variables and making assumptions of system behavior.

Neuronal activity can be modeled in detail with the Hodgkin-Huxley (HH) formalism [10]. Less detailed, simplified neuron models are motivated by computationally efficient large scale simulations [11]. Morphologically accurate models of branching neurons have been simplified algorithmically to derive efficient models [15], [16]. However, the simplification approach is not suitable for the current trend in neuroscience, in which multiple physical scales of the brain are incorporated in simulations and the consequent analysis of neural phenomena. Instead comprehensive models with full system dynamics are needed in order to increase understanding of different actors in one brain area.

In this paper, we study the effectiveness of MOR methods to reduce a nonlinear biophysical network model describing synchronized population bursting behavior of heterogeneous pyramidal neurons in the brain [21]. Modeling studies in computational neuroscience are typically interested in the spatial and temporal evolution of the membrane voltage of neurons. Neurons communicate by swiftly changing their membrane voltage to create action potentials (spikes) that propagate from cell to cell. Spiking is the fundamental method of sensory information processing in the brain, and synchronized spiking is an emergent property of biological neuronal networks. MOR should preserve this network behavior. Here we reduce the network model with four variations of Proper Orthogonal Decomposition (POD) [14] and Discrete Empirical Interpolation Method (DEIM) [4] that are developed to reduce nonlinear systems. These methods are DEIM, Localized DEIM (LDEIM) [18], Discrete Adaptive POD (DAPOD) [28] and Adaptive DEIM [20]. DEIM and the variations are used here in combination with POD.

The neurons in our network model feature a state-triggered jump condition. Such *resets* are often used in simplified neuron models to efficiently simulate the spiking behavior of a neuron [11] as an alternative to explicitly modeling ion channel kinetics. In the present model, the jump condition implements a biological boundary to ions that are in limited supply in the neural matter.

Many versions of DEIM have been developed. Unassembled DEIM improves the reducibility of finite element models [24]. Matrix DEIM improves the efficiency of evaluating the Jacobian matrix [26]. Localized DEIM uses machine learning to compute multiple reduced bases offline and choose between them in the online phase appropriately [18]. Temporally Localized DEIM, a recent approach to rapidly changing local subspaces was introduced in [3]. Adaptive DEIM with online updates to the DEIM basis and interpolation points has been developed to better handle unseen states in the simulation phase [20]. Non-negative DEIM adds structure preservation guarantees to the reduced model [1]. Furthermore, several recent studies address the performance of DEIM in noisy environments and propose randomized oversampling and QR-decomposition inspired basis computation methods [6], [19]. Finally, an algorithm that adapts the POD basis online, Discrete Adaptive POD, has been published [29]. It can be combined with many of the DEIM algorithms. Our reduction methods in choice are described in detail in Section III.

M. Lehtimäki: Computational Neuroscience, Faculty of Medicine and Health Technology, Tampere University. Supported by TUNI Graduate School. mikko.lehtimaki@tunif.i

L. Paunonen: Mathematics, Faculty of Information Technology and Communication Sciences, Tampere University. Supported by Academy of Finland grants 298182 and 310489. lassi.paunonen@tuni.fi

M-L. Linne: Computational Neuroscience, Faculty of Medicine and Health Technology, Tampere University. Supported by Academy of Finland grant 297893 and Human Brain Project (785907). marja-leena.linne@tunif.i

Mathematical MOR of nonlinear systems in neuroscience has been studied before in limited settings. Kellems et al. (2010) reduced a partial differential equation (PDE) model that described a single branching neuron with HH dynamics [12]. They compared how an action potential travels from stimulus injection point along the neuron in the original and reduced models, using POD-DEIM as the MOR method. Du et al. (2014) built on the work of Kellems et al. by including linearization of weakly excitable (passive) parts of the neuron [7]. Amsallem and Nordström (2016) studied a branching neuron and introduced stability and nonnegativity properties to the reduced basis [1]. Finally, Lehtimäki et al. (2017) reduced a chemical reaction based nonlinear model of synaptic plasticity [13] using POD-DEIM.

Nonlinear MOR studies of population activity of neurons have been conducted to model the cardiac and muscular systems. A model of electrical properties of the cardiac system has been reduced in [2] via POD and in [27] via POD-DEIM. Both studies used a simple phenomenological neuron model. Additionally, POD-DEIM has been employed in reduction of bidomain and monodomain electromyography models describing muscle fibers in [17], [8], where the more biophysically accurate HH formalism was employed.

In Section II we describe the biophysics and construction of the network model. Section III explains our MOR approach and methods in more detail. Our results are presented in IV and we conclude by discussing the significance of our work and ideas for future studies in sections V and VI, respectively.

II. BIOPHYSICAL NETWORK MODEL

The biophysical network model describes pyramidal neurons from the CA3 area of the hippocampus [21]. The morphology of the cell is considered by a spatial modeling approach, where the cell is split into compartments, and the membrane voltage of each compartment is coupled with the voltage of adjacent compartments via electrotonic coupling as described by the neuronal cable theory [22]. Each neuron is modeled in a biophysical manner, so that the membrane voltage of one compartment behaves according to ionic currents in HH formalism [10]. Ion channels model the *active* propagation, and cable theory models the *passive* propagation of membrane potential along the cell. The single-cell model itself is a simplified version of an originally 19 compartment model of the same cell type [25].

A network of these cells is formed by coupling them in a random, directed graph manner. The interesting property of the modeled network is its capability to bring about and sustain periodic, oscillating population level activity, where neurons spike in their somas in a synchronized manner. We wish to determine whether this behavior is preserved throughout the MOR process.

Each single cell model consists of ten ordinary differential equations (ODEs). The ODE of the somatic membrane potential V_s is

$$\begin{aligned} C_m \frac{dV_s}{dt} =& - I_{leak}(V_s) - I_{Na}(V_s, m, h) - I_{K-DR}(V_s, n) \\ &+ \frac{g_c}{p}(V_d - V_s) + \frac{I_s}{p}, \end{aligned} \tag{1}$$

where V_d is the voltage of the dendritic compartment, I_{leak} is a leak current, I_s is an injected current, I_{Na} is a sodium current, I_{K-DR} is a potassium delayed rectifier current, m, h and n are HH-type voltage gated ion channel activation variables of sodium and potassium (delayed rectifier), g_c is the electrotonic coupling conductance between the two compartments and p is the relative size of the soma compartment.

The ODE of the dendritic compartment is comparable to the somatic compartment, however in place of sodium currents and potassium delayed rectifier current the voltage of the dendritic compartment depends on calcium, calcium-activated potassium and afterhyperpolarization potassium currents. Their respective activation variables are s, c and q. Additionally, excitatory synaptic currents from NMDA (S) and AMPA (W) are included. Moreover, Ca in the dendritic compartment as well as the synaptic AMPA and NMDA concentrations are modeled with their respective ODEs. [21]

The kinetics of the gating variables h, n, s, c and q are modeled by ODEs of the form

$$\frac{dy}{dt} = (y_\infty(U(t)) - y)/\tau_y(U(t)), \tag{2}$$

where $U(t)$ is either the somatic or dendritic membrane voltage or calcium (Ca) concentration at time t, depending on the gating variable, and

$$y_\infty = \alpha_y/(\alpha_y + \beta_y) \tag{3}$$

and

$$\tau_y = 1/(\alpha_y + \beta_y) \tag{4}$$

where α_y and β_y are distinct for every gating variable m, h, n, s, c. For example, for gating variable n we have

$$\begin{aligned} \alpha_n =& \frac{0.016(35.1 - V_s)}{e^{(35.1-V_s)/5-1}}, \\ \beta_n =& 0.25e^{0.5-0.025V_s}, \end{aligned}$$

hence the model contains very fast nonlinear dynamics. The sodium activation gate m is instantaneous and is modeled only with Equation 3.

The ODE of the NMDA concentration is particularly interesting as it is connected to a reset condition that keeps the value of S bounded so that

$$\begin{aligned} \frac{dS}{dt} =& \sum jH(V_{s,j} - 10) - S/150, \\ S(t) =& \min(S(t), 125), \end{aligned} \tag{5}$$

where $V_{s,j}$ is the somatic voltage of the synaptic connection from cell j and $H(x) = 1$ if $x \geq 0$ and 0 otherwise. The constants keep $S(t)$ in a biologically justified range. From this equation the nonlinear nature of synaptic connections is also apparent. For full details of the single cell model,

see [21] and for an implementation that accounts for the errata of the original publication, see [5].

Our network model consists of heterogeneous neurons. The calcium conductance in the dendritic compartment can vary by 10%, with the amount drawn from the uniform distribution. Each cell receives synaptic input from 20 other randomly chosen cells. We use in total 50 cells, obtaining an ODE system of 500 variables. To study the population behavior we use numerical integration with fixed step 4th order Runge-Kutta method. One cell is stimulated with a current pulse at $t = 5\,\mathrm{ms}$ for $t = 50\,\mathrm{ms}$ and the simulation is executed for $1000\,\mathrm{ms}$ with a timestep of $dt = 0.02\,\mathrm{ms}$. To measure population behavior, at each time instance the number of cells that are spiking is counted. An action potential (membrane voltage spike) is considered to occur when a threshold level is exceeded. Here, the threshold is $V_t = -40\,\mathrm{mV}$, as in [21].

The network model we study is nonlinear with

$$x'(t) = Ax(t) + f(x(t)) + Bu(t), \tag{6}$$

where A is the state matrix, B is the input matrix, $u(t)$ is a vector of time dependent inputs and $f(x(t))$ is a vector of nonlinear functions. $A \in \mathbb{R}^{10\nu\times 10\nu}$ and $B \in \mathbb{R}^{10\nu\times 4\nu}$ are block diagonal matrices composed of the state and input matrices of the cells in the network, and $f(x(t)) \in \mathbb{R}^{10\nu}$, where ν is the number of cells in the network. The system has a stable steady state where each cell in the network is at resting potential, thus no cell is spiking, and if the system is stimulated with current injection every cell will eventually return to the resting potential after the stimulus has stopped.

III. MODEL ORDER REDUCTION

We create reduced order models (ROMs) with variations of POD [14] and DEIM [4]. These methods are applicable to general nonlinear systems and their suitability to models with Hodgkin-Huxley type ion channel kinetics has been studied before in [1], [7], [8], [12], [17]. Furthermore, we wish to avoid linearizations, since for the present system it is very challenging to determine robust linearization points. These methods also allow the implementation of reset conditions that are a part of the studied model, similarly as in [2].

POD is a projection based MOR method that approximates the original n dimensional system in a reduced order linear subspace. A reduced basis with orthonormal column vectors $V_k \in \mathbb{R}^{n\times k}$ where $k < n$ is determined using singular value decomposition (SVD). This POD basis is constructed from snapshots $Y = [y_1, y_2, \cdots, y_s]$ that are a set of solutions to the original system [23]. By setting $x(t) \approx V_k\tilde{x}(t)$ and projecting the system described in Equation 6 onto V_k by Galerkin projection, a reduced system

$$\tilde{x}'(t) = \underbrace{V_k^T A V_k}_{\tilde{A}}\tilde{x}(t) + V_k^T f(V_k\tilde{x}(t)) + \underbrace{V_k^T B}_{\tilde{B}}u(t) \tag{7}$$

is obtained. In Equation 7 $\tilde{A}$ and $\tilde{B}$ can be precomputed before the online (simulation) phase. However, while POD itself can be applied to nonlinear systems, there is no guarantee of computational savings as the nonlinear part of the system must be evaluated in the original space.

Efficient evaluation of the nonlinear term can be achieved with DEIM [4]. DEIM extends the subspace projection approach of POD with an interpolation step for nonlinear functions. To construct a reduced order approximation of the nonlinear vector, the algorithm determines

$$\tilde{f}(x,t) \approx U_m(P_m^T U_m)^{-1} P_m^T f(x,t), \tag{8}$$

where the DEIM basis $U_m = [u_1, u_2, \cdots, u_m], m < n$ is determined via SVD of snapshots of nonlinear function outputs, $P_m^T f(x,t) := f_m(x,t)$ is a nonlinear function with m components chosen from f according to DEIM determined interpolation points $p_1, \cdots, p_m$ and $P_m = [e_{p_1}, e_{p_2}, \cdots, e_{p_m}]$ with e_{p_i} being the standard basis vector i of $\mathbb{R}^n$. Note that the POD dimension k and DEIM dimension m do not need to be equal, although empirically it has been observed that $k = m$ leads to most accurate reduced models [8], [19]. Together POD and DEIM form a ROM

$$\tilde{x}'(t) = \tilde{A}\tilde{x}(t) + \underbrace{V_k^T U_m (P^T U_m)^{-1}}_{N} f_m(V_k\tilde{x}(t)) + \tilde{B}u(t), \tag{9}$$

where N can be precomputed in the offline phase. Thus in the online phase it remains to compute $V_k\tilde{x}$ so that f_m can be evaluated at the m interpolation points.

Each cell in the network model has a state determined jump condition, as seen in Equation 5. This condition must be checked at every evaluation of the state of the system. To determine the jump condition, the reduced state vector $\tilde{x}$ must be projected to the original space. Numerically we implement this check before every evaluation of the nonlinear vector $f_m(x,t)$, that also requires the state in the original space. However, after resolving jump conditions the state needs to be projected back to the reduced space in order to evaluate $\tilde{A}\tilde{x}$, which creates an extra computational step that would not be otherwise required. The cost of this step depends on the chosen POD dimension k.

A. Versions of the Discrete Empirical Interpolation Method

In addition to DEIM as described in [4] we test the efficacy of Localized Discrete Empricial Interpolation Method (LDEIM) [18]. In LDEIM, a clustering algorithm is employed in the offline phase to group solution snapshots before DEIM basis generation. Several bases and interpolation points $(U_{m_1}, P_{m_1}), \cdots, (U_{m_p}, P_{m_p})$ are computed to obtain a set of local bases, one from each cluster of snapshots, in contrast to the global basis used in DEIM. Each local basis has the same dimension, and LDEIM should achieve a similar error estimate as a global basis but with a smaller dimension. In the online phase a local precomputed DEIM basis is chosen adaptively. The features used to derive and choose the bases are a subset of outputs from the nonlinear function. LDEIM requires the number of clusters and features as user defined parameters, and the size of the feature vector is a decision between classification power and computational efficiency. The premise of LDEIM is

to use multiple smaller yet accurate reduced subspaces to compensate for the extra online computation time that is needed for basis selection.

Another further development of DEIM that we use is Adaptive DEIM (ADEIM) [20]. In ADEIM, the DEIM basis and interpolation points are updated online. Initially, (U_m, P_m) are computed offline, and at step s of the simulation (U_s, P_s) are determined. The adaptivity is performed via low-rank updates to (U_{s-1}, P_{s-1}). A random set of size s_p of unique additional sampling points of the nonlinear function is drawn from the uniform distribution and added to the set of total sampling points, then the nonlinear function is evaluated at these points in length w window of past solutions. The resulting online snapshots are used to compute an updated basis and sampling points. This online adaptivity does not change the dimension of the reduced subspace. The algorithm involves considerable online computation, but is more capable of reducing models with trajectories that were not sampled in the offline phase.

Finally, we implement Discrete Adaptive POD (DAPOD) [29], [28] with DEIM. DAPOD adapts the dimension and structure of the POD basis V online. In the online phase, new snapshots are incorporated and existing ones eliminated from the snapshot ensemble based on adaptivity criteria that weigh importance and age of snapshots. The basis size is determined with an error bound parameter ϵ decided by the user. A smaller ϵ corresponds to a tighter error bound, which results in a larger POD dimension and greater run time. DAPOD can be combined with DEIM to reduce nonlinear models more efficiently. However, as the POD basis V now changes online, the DEIM projection matrix N of Equation 9 cannot be precomputed, which adds some additional online computational burden.

IV. RESULTS

The network displays a temporally synchronized activity pattern, where the majority of the neurons spike at similar times, which replicates the behavior from [21]. This is seen as oscillations in the number of neurons spiking at any given time. Figure 1 illustrates the trajectory of the somatic membrane voltage of the stimulated neuron (top) along with network level activity (bottom) as a raster plot. In the raster plot, a red dot is marked at every time instance on the x-axis if the somatic voltage of a neuron in the y-axis is greater than a voltage threshold $V_t = -40\,\mathrm{ms}$. The stimulated neuron is at index 0 in the raster plot.

The oscillations of the population activity are an important phenomenon of the model and one that the ROM should recreate. The pattern is illustrated in Figure 2, which shows the number of neurons spiking as a function of time. A qualitative comparison of the original model to reduced order models of several parameters, with many reduction methods, is provided in Figure 2. The behavior of the original model is shown in every plot, and each row corresponds to a different reduction method. Combinations of dimensions or reduction method specific parameters are displayed in different colors.

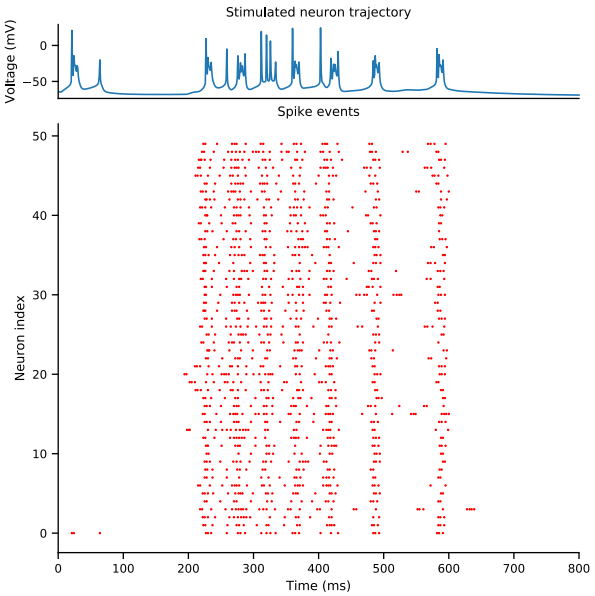

Fig. 1. Top: trajectory of the somaticmembrane voltage of the stimulated neuron. Bottom: raster plot of spike events of each neuron as a function of time.

In Figure 2 it is seen that each reduction method has their strengths and weaknesses in recreating the original simulation. With the present model, a 5-10% reduction, depending on the method, causes numerical overflow errors or flat trajectories. The overflow errors end the simulation immediately, and a ROM with a flat trajectory is not useful. For this reason, the analysis considers DEIM dimensions 480 and 470.

The top most plot in Figure 2 presents the performance of the standard DEIM method from [4]. Notably, the reduced order models display a slight temporal shift in population activity around $t = 600\,\mathrm{ms}$ already with 4% reduction. Moreover, an additional burst of spikes at $t = 750\,\mathrm{ms}$ is detected and subsequent residual network activity is observed in the DEIM reduced models, although the original network silences itself after the last population burst. This method has the lowest computational burden and even with the current slight reduction is faster to simulate than the original model.

The second plot from the top in Figure 2 shows results obtained with the LDEIM method from [18]. The trajectories are very similar to original DEIM, although LDEIM produces the early network activity more accurately. Both methods show a shift in the times of occurrence and magnitudes of synchronized spike events as the simulation progresses. Residual network activity is also observed here. The simulation time of LDEIM is greater than DEIM, since with the present model LDEIM does not achieve a smaller dimension than DEIM and the overhead of online basis changes reduces the computational efficiency of the reduced models.

Results with the DAPOD-DEIM method from [29], [28] are shown in the third plot from the top in Figure 2. DAPOD

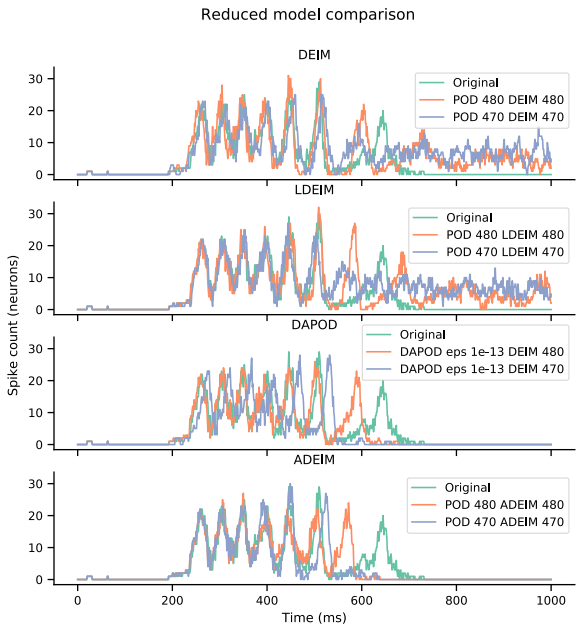

Fig. 2. Comparison of population behavior with different reduction methods and parameters. Methods from top to bottom are DEIM, LDEIM, DAPOD-DEIM and ADEIM. In all plots, x-axis is time in milliseconds and y-axis is number of neurons spiking.

is unique in the set of tested methods because it allows the POD basis to change online phase of reduction. We keep the error controlling value ϵ constant while lowering the DEIM dimension. In our simulations, this online adaptivity allows DAPOD to use a smaller number of POD dimensions than what is achieved by other reduction methods. With $\epsilon = 1e^{-13}$ the POD dimension is in the range $[462, 456]$. It is seen that at 470 DEIM dimensions, DAPOD-DEIM fails to recreate the last spike in population activity, whereas DEIM set to 480 displays it, although with a temporal and magnitudinal shift. Remarkably, DAPOD-DEIM simulations do not suffer from the residual activity seen in DEIM simulations. The simulation time of DAPOD-DEIM with the present parameters is greater than the original model.

Finally, the bottommost plot in Figure 2 displays MOR results from the ADEIM method [20]. We used a look-back window length of $w = 25$ and $s_p = 10$ additional random sampling points in our simulations. The results resemble those of the DAPOD method, as the residual activity toward the end of simulation is correctly absent. The last activity peak occurs too early, and some of the earlier peaks have a smaller magnitude than the original simulation. The first population activity bursts are recreated accuracy similar to LDEIM. With regards to simulation time, ADEIM is the heaviest to compute, having a runtime of over tenfold the original model.

V. DISCUSSION

There are several differences between prior MOR studies in neuroscience and our present work. We make the following comparison to publications [1], [7], [12], since those considered a biophysically detailed and morphologically complex neuron model. First, the present network model is built by coupling heterogeneous ODE systems, in comparison to reducing discretized PDEs as in the other studies. Second, in terms of system dynamics the above studies used a model of a single neural cell where an action potential was propagating as a result of current injection, whereas we now include multiple neurons that receive and process stimulus from several sources asynchronously. Third, biophysically our cells include more complexity due to a larger number of distinct ionic currents, heterogenic parameters and functionally specialized compartments over copying identical compartments to create the single cell model. To make a final distinction on network topology, we view the single cell models of [1], [7], [12] as networks of compartments; then, their connectivity is restricted to neighbouring compartments. On the other hand, our network of compartmental neurons allows random connectivity between any number of cells.

A neuron model with a reset condition was reduced in [2] using POD. There, the reset condition was used to create spiking behavior in a simple phenomenological model, whereas in our study a state-dependent jump condition was employed to implement a biophysical threshold to the synaptic NMDA current. In [2] it was concluded that significant offline efforts were needed to derive a reliable reduced model. Our implementation of the state-dependent jump is explained in detail in Section III. We found that the jump condition reduces the computational efficiency of our reduced models. It can also be a major source of reduction error, if the jumps in reduced models occur at different timesteps than in the original model.

With the present model, reduction error grows rapidly as POD and DEIM dimensions get lower, which then prevents the reduction methods from achieving low dimensions with meaningful results. However, the reduction methods described in this study were able to replicate the emergent synchronized population activity seen in our original network model. This is an encouraging result, especially as these methods have originally been reported in the context of discretized PDE systems [4]. In comparison, our model is based on nonlinearly coupled heterogeneous neurons described by nonlinear ODEs, making the model reduction setting different from those in the MOR method publications.

Based on our study, DAPOD and ADEIM perform best in preserving the spiking activity of the original network model. However, ADEIM is too slow to be practically usable, as the present model does not allow low enough dimensions to offset the computational costs of online adaptivity. DAPOD is able to find a lower dimensional POD basis online than the other methods find offline, and has runtime close to the original model. We deem DAPOD especially useful if the system has ”quiet” and ”active phases”, where the slowly evolving system could be approximated with a relatively small POD basis and when system-wide activity starts, POD dimension can increase to maintain a low error.

A shift in oscillation frequency, magnitude or phase of

population activity is a phenomenon seen in the reduced order models we presented in Section IV, Figure 2. It is difficult to exactly quantify the significance of this reduction error. However, with the relatively small computational efficiency increases seen in this study, the error is difficult to justify. The additional or missing population bursts in some of the reduced models are of greater significance. From the perspective of neuroscience, such behavior could be caused by intracellular or extracellular factors. The reduced models would not allow the study of these conditions, if they recreate an incorrect number of bursts of spikes.

We found residual network activity in DEIM and LDEIM models, seen as continued spiking activity towards the end of the simulation when the original model is no longer spiking. This reduction artifact could be caused by noise amplification as described in [19]. Especially the delicate ion channel kinetics are sensitive to approximation errors and noise. Interestingly, the two methods that adapt the reduced basis online, DAPOD-DEIM and ADEIM, do not display the same residual activity seen in DEIM and LDEIM reduced models with the same dimension. This improvement in accuracy does come at a cost in simulation speed.

VI. CONCLUSIONS AND FUTURE STUDIES

We constructed a network model of biophysically detailed compartmental neurons modeled with nonlinear ordinary differential equations, implemented several projection-based model order reduction methods and qualitatively evaluated model reduction results. When the model is stimulated with a current pulse, it displays synchronized population activity. The model was reduced with DEIM, LDEIM, DAPOD-DEIM and ADEIM. The reduced models had challenges in recreating the desired population behavior with low dimensions, possibly due to delicate ion channel kinetics or the inclusion of state-dependent jump conditions. Simulation was most efficient with DEIM, although DAPOD-DEIM and ADEIM capture the behavior of the model more accurately.

Future work will compare these results to reduced models obtained with TLDEIM [3] and the methods presented in [19].

REFERENCES

[1] D. Amsallem and J. Nordström. Energy stable model reduction of neurons by nonnegative discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 38(2):B297–B326, 2016.

[2] M. Boulakia, E. Schenone, and J-F. Gerbeau. Reduced-order modeling for cardiac electrophysiology. Application to parameter identification. *International Journal for Numerical Methods in Biomedical Engineering*, 28(6-7):727–744, 2012.

[3] S. Chaturantabut. Temporal localized nonlinear model reduction with a priori error estimate. *Applied Numerical Mathematics*, 119:225–238, 2017.

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

[5] E. Chmiel, J. Birgiolas, P. Gleeson, and W. Lytton. Pinsky and Rinzel 1994 CA3 neuron model. `http://www.opensourcebrain.org/projects/pinskyrinzelmodel,https://senselab.med.yale.edu/ModelDB/ShowModel.cshtml?model=35358`. Accessed 2019-02-19.

[6] Z. Drmac and S. Gugercin. A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM Journal on Scientific Computing*, 38(2):A631–A648, 2016.

[7] B. Du, D. Sorensen, and S.J. Cox. Model reduction of strong-weak neurons. *Frontiers in Computational Neuroscience*, 8(164), 2014.

[8] N. Emamy, P. Litty, T. Klotz, M. Mehl, and O. Röhrle. POD-DEIM model order reduction for the monodomain reaction-diffusion submodel of the neuro-muscular system. In J. Fehr and B. Haasdonk, editors, *IUTAM Symposium on Model Order Reduction of Coupled Systems, Stuttgart, Germany, May 22–25, 2018*, pages 177–190, Cham, 2020. Springer International Publishing.

[9] W. Gerstner, H. Sprekeler, and G. Deco. Theory and simulation in neuroscience. *Science*, 338, 2012.

[10] A. Hodgkin and A. Huxley. A quantitave description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, 117(4):500–544, 1952.

[11] E. M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, 2003.

[12] A. Kellems, S. Chaturantabut, D. Sorensen, and S. Cox. Morphologically accurate reduced order modeling of spiking neurons. *Journal of Computational Neuroscience*, 28(3), 2010.

[13] M. Lehtimäki, L. Paunonen, S. Pohjolainen, and M-L. Linne. Order reduction for a signaling pathway model of neuronal synaptic plasticity. *IFAC-PapersOnLine*, 50(1):7687–7692, 2017.

[14] J. Lumley, G. Berkooz, and P. Holmes. The Proper Orthogonal Decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[15] A. Marasco, A. Limongiello, and M. Migliore. Fast and accurate low-dimensional reduction of biophysically detailed neuron models. *Scientific Reports*, 2:928, 2012.

[16] A. Marasco, A. Limongiello, and M. Migliore. Using Strahler's analysis to reduce up to 200-fold the run time of realistic neuron models. *Scientific Reports*, 3:2934, 2013.

[17] M Mordhorst, T. Strecker, D. Wirtz, T. Heidlauf, and O. Röhrle. Pod-deim reduction of computational emg models. *Journal of Computational Science*, 19:86–96, 2017.

[18] B. Peherstorfer, D. Butnaru, K. Willcox, and H.J. Bungartz. Localized discrete empirical interpolation method. *SIAM Journal on Scientific Computing*, 36(1), 2014.

[19] B. Peherstorfer, Z. Drmač, and S. Gugercin. Stabilizing discrete empirical interpolation via randomized and deterministic oversampling. *arXiv preprint arXiv:1808.10473*, 2018.

[20] B. Peherstorfer and K. Willcox. Online adaptive model reduction for nonlinear systems via low-rank updates. *SIAM Journal on Scientific Computing*, 37(4), 2015.

[21] P. F. Pinsky and J. Rinzel. Intrinsic and network rhythmogenesis in a reduced traub model for CA3 neurons. *Journal of Computational Neuroscience*, 1(1-2):39–60, 1994.

[22] W. Rall. Theory of physiological properties of dendrites. *Annals of the New York Academy of Sciences*, 96(4):1071–1092, 1962.

[23] L. Sirovich. Turbulence and the dynamics of coherent structures. I-III. *Quarterly of Applied Mathematics*, 45(3):561–590, 1987.

[24] P. Tiso and D. J. Rixen. Discrete empirical interpolation method for finite element structural dynamics. In *Topics in Nonlinear Dynamics, Volume 1*, pages 203–212. Springer, 2013.

[25] R. D Traub, R. K. Wong, R. Miles, and H. Michelson. A model of a CA3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances. *Journal of Neurophysiology*, 66(2):635–650, 1991.

[26] D. Wirtz, D. C. Sorensen, and B. Haasdonk. A posteriori error estimation for deim reduced nonlinear dynamical systems. *SIAM Journal on Scientific Computing*, 36(2):A311–A338, 2014.

[27] H. Yang and A. Veneziani. Efficient estimation of cardiac conductivities via pod-deim model order reduction. *Applied Numerical Mathematics*, 115:180–199, 2017.

[28] M. Yang and A. Armaou. Dissipative distributed parameter systems on-line reduction and control using DEIM/APOD combination. In *2018 Annual American Control Conference (ACC)*, pages 2557–2562. IEEE, 2018.

[29] M. Yang and A. Armaou. Revisiting APOD accuracy for nonlinear control of transport reaction processes: A spatially discrete approach. *Chemical Engineering Science*, 181:146–158, 2018.

PUBLICATION III

Accelerated simulation of a neuronal population via mathematical model order reduction

M. Lehtimäki, I. Seppälä, L. Paunonen, and M.-L. Linne

Proceedings of the 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), 2020, pp. 118–122
DOI: https://doi.org/10.1109/aicas48895.2020.9073844

Accelerated Simulation of a Neuronal Population via Mathematical Model Order Reduction

Mikko Lehtimäki*, Ippa Seppälä*, Lassi Paunonen†, Marja-Leena Linne*
*Faculty of Medicine and Health Technology
†Faculty of Information Technology and Communication Sciences
Tampere University, Tampere, Finland.
Email: mikko.lehtimaki@tuni.fi, ippa.seppala@tuni.fi, lassi.paunonen@tuni.fi, marja-leena.linne@tuni.fi

***Abstract*—Mathematical modeling of biological neuronal networks is important in order to increase understanding of the brain and develop systems capable of brain-like learning. While mathematical analysis of these comprehensive, stochastic, and complex models is intractable, and their numerical simulation is very resource intensive, mean-field modeling is an effective tool in enabling the analysis of these models. The mean-field approach allows the study of populations of biophysically detailed neurons with some assumptions of the mean behaviour of the population, but ultimately requires numerical solving of high-dimensional differential equation systems. Mathematical model order reduction methods can be employed to accelerate the analysis of high-dimensional nonlinear models with a purely software-based approach. Here we compare state-of-the-art methods for improving the simulation time of a neuronal mean-field model and show that a nonlinear Fokker-Planck-McKean-Vlasov model can be accurately approximated in low-dimensional subspaces with these methods. Using Proper Orthogonal Decomposition and different variations of the Discrete Empirical Interpolation Method, we improved the simulation time by over three orders of magnitude while achieving low approximation error.**

I. Introduction

The human brain possesses unmatched information processing and generalization capabilities. Hence, machine learning has drawn inspiration from neuroscience and understanding the mechanisms of learning in biological neuronal networks of the brain continues to be of utmost importance for developing efficient and effective machine learning algorithms [1]. Some interesting properties of these networks are their natural capability to process data with a temporal dimension (see Reservoir Computing) [2], use of an energy efficient continuous computation paradigm which can be imitated with neuromorphic hardware [3], [4], and diverse synaptic plasticity rules [5]. Utilizing these features together with brain-like computation units could result in improved performance in classification and regression tasks [6]. Indeed, by understanding how the brain implements biological intelligence, progress can also be made in deep learning [7], [8]. To that end, insight into biological neuronal networks can be gained with mean-field methods [9]. Here, we show how the simulation of a computationally expensive mean-field model can be accelerated with mathematical model order reduction (MOR) methods [10].

Biological systems are naturally noisy and rarely behave identically between repeated measurements [11]. This stochasticity, be it intrinsic or external, is believed to serve a purpose such as enhancing detection of weak signals [12]. The effects of noise in neuronal signaling have been studied with stochastic models of neuronal systems in the molecular [13], single cell [14] and network [15] levels. However, the added biological relevance achieved with stochastic modeling comes with the burden of mathematical intractability and increased computation time, as specialized numerical methods are needed for simulating stochastic systems and analysis of system dynamics relies on Monte Carlo methods. Understanding network plasticity in the presence of this randomness is one step towards more brain-like machine learning algorithms.

The dynamics of stochastic neuronal network circuits can be studied with mean-field models that use deterministic descriptions of the underlying network [9]. The mean-field framework encompasses multiple methods that result in models of different levels of complexity, ranging from simple firing rate equations to probability density functions to models with multiple spatial dimensions. In this work, we will study a Fokker-Planck-McKean-Vlasov-type mean-field model that describes the time evolution of the probability density function of the state of a large neuronal population of stochastic FitzHugh-Nagumo (FN) neurons [16]–[18]. The model is mathematically intractable, and studying it requires numerical solving methods.

We show that numerical simulation of a mean-field model can be made significantly faster by employing *reduced order models*, created with mathematical MOR methods. MOR methods require no simplifications of the modeled system and allow every variable therein to be reconstructed at any time. When choosing MOR methods appropriately, no linearization is required, which allows more dynamics to be retained in the reduced model [19]. Hence, these methods can be applied directly to nonlinear systems, which in the field of computational neuroscience is highly advantageous. Moreover, the chosen MOR methods allow approximating linear and nonlinear components independently, in order to approximate the original system accurately.

In Section II, the model we studied is introduced together with mean-field theory, and the MOR methods employed here are described. In Section III, numerical simulation results of the original and reduced models are shown. Finally, in Section IV the significance of our results is discussed together with directions for future research.

II. METHODS

Mean-field approximation was originally used to describe the spin of electrons in theoretical physics. In the field of computational neuroscience the method can be used to model the behaviour of populations of neurons [9]. The single neuron model, which will be taken to the mean-field limit, is in itself stochastic. These neurons can prove to be challenging to incorporate into models, and are often preprocessed by introducing tractable randomness in the form of Markov chains. In such a Markov chain the transition probabilities of neuronal states obey the *master equation*, from which the *Fokker-Planck equation* can be derived. In general, the mean-field approximation consists of dividing neurons into statistically similar populations, in which the population behaviour is uncorrelated. This is true when the population size is theoretically infinite.

In this work we examine a mean-field model derived in [16], representing a population of FN neurons. This second-order nonlinear partial differential equation (PDE) has three independent variables and describes the time evolution of the probability density function $p(t, V, W, Y)$ of the state of the neuron population. It gives a deterministic description of the underlying stochastic system. The model is

$$\begin{aligned}
&\partial_t p(t,V,W,Y) \\
&= \frac{1}{2}\sigma_J^2 \bar{y}^2(t) \frac{\partial^2}{\partial V^2}\left[(V - V_{rev})^2 p(t,V,W,Y)\right] \\
&\quad + \frac{1}{2}\frac{\partial^2}{\partial Y^2}\left[\sigma_Y^2(V,Y) p(t,V,W,Y)\right] \\
&\quad + \frac{1}{2}\sigma_{ext}^2 \frac{\partial^2}{\partial V^2}\left[p(t,V,W,Y)\right] \\
&\quad - \frac{\partial}{\partial V}\left[\left(V - \frac{V^3}{3} - W + I_{ext} + \bar{J}(V - V_{rev})\bar{y}(t)\right)\right. \\
&\quad \left. \times\, p(t,V,W,Y)\right] \\
&\quad - \frac{\partial}{\partial W}\left[a(V + b - cW) p(t,V,W,Y)\right] \\
&\quad - \frac{\partial}{\partial Y}\left[\left(\alpha_r S(V)(1 - Y) - \alpha_d Y\right) p(t,V,W,Y)\right],
\end{aligned} \tag{1}$$

where $\bar{y}(t) = \iiint y p(t, v, w, y)\, dv\, dw\, dy$, V is the neuronal membrane voltage, W is the recovery variable of the FN model, Y is the synaptic conductance of the neurons in this population, and I_{ext} is external current stimulus. For additional details of the model, see [16].

Equation (1) must be discretized in space prior to numerical simulation. The discretization results in a system of ordinary differential equations of dimension (number of equations to solve) ν^3 with ν being the number of discretization points in one variable of the PDE, assuming an equal amount of discretization points in every variable. A fine discretization grid is required so that fast dynamics of the model are captured, making ν a large integer.

Spatial discretization of Equation (1) is carried out with a fourth-order central difference scheme and the triple integral $\bar{y}(t)$ is evaluated with the Newton-Cotes method of order six. After discretization, we write the system in state-space format

$$x'(t) = Ax(t) + f(x(t)), \tag{2}$$

where $x \in \mathbb{R}^n$ is the current state of the system, $A \in \mathbb{R}^{n \times n}$ is the state matrix with linear coefficients, $f(x(t)) \in \mathbb{R}^n$ is a vector of nonlinear functions, $n = \nu^3$, and $\nu = 50$. For numerical simulations we use parameters from [16].

We construct reduced order models (ROMs) with the Discrete Empirical Interpolation Method (DEIM) [20] and two of its advanced variants, namely LDEIM and QDEIM. DEIM is a MOR method that is used in conjunction with the Proper Orthogonal Decomposition (POD) [21] and is based on the method from [19]. These methods are applicable to general nonlinear systems such as the model used here.

POD is a projection based MOR method that approximates the original system of dimension n in a reduced linear subspace. A reduced basis with orthonormal column vectors $V_k \in \mathbb{R}^{n \times k}$ where $k < n$ is computed using singular value decomposition (SVD). This POD basis is constructed from snapshots $Y = [y_1, y_2, \cdots, y_s]$ that are a set of solutions to the original system [22], collected for example with numerical simulation. Then, a reduced state vector $V_k^T x(t) = \tilde{x}(t) \in \mathbb{R}^k$ is obtained by a linear transformation. Projecting the system described in Equation (2) onto V_k by Galerkin projection results in a reduced system

$$\tilde{x}'(t) = \underbrace{V_k^T A V_k}_{\tilde{A}} \tilde{x}(t) + V_k^T f(V_k \tilde{x}(t)) \tag{3}$$

where $\tilde{A}$ can be precomputed before the online (simulation) phase. At any point, an approximation of the original, full-dimensional state vector can be computed with $x(t) \approx V_k \tilde{x}(t)$.

However, while POD itself can be applied to nonlinear systems, there is no guarantee of computational savings as the nonlinear part $f(V_k \tilde{x}(t))$ of the reduced system must be evaluated in the original space. Efficient evaluation of the nonlinear term can be achieved with DEIM [19], [20]. DEIM extends the subspace projection approach with an interpolation step for nonlinear functions. To construct an approximation of the nonlinear term, the algorithm gives

$$\tilde{f}(x,t) \approx U_m (P_m^T U_m)^{-1} P_m^T f(x,t), \tag{4}$$

where the DEIM basis $U_m = [u_1, u_2, \cdots, u_m], m < n$ is computed via SVD of the snapshots of the nonlinear function outputs, $P_m^T f(x,t) := f_m(x,t)$ is a nonlinear function with m components chosen from f according to DEIM determined interpolation points $p_1, \cdots, p_m$ and $P_m = [e_{p_1}, e_{p_2}, \cdots, e_{p_m}]$ with e_{p_i} being the standard basis vector i of $\mathbb{R}^n$. Together POD and DEIM form a ROM

$$\tilde{x}'(t) = \tilde{A}\tilde{x}(t) + \underbrace{V_k^T U_m (P^T U_m)^{-1}}_{N} f_m(V_k \tilde{x}(t)), \tag{5}$$

where $N \in \mathbb{R}^{n \times m}$ can be precomputed in the offline phase. Thus in the online phase only m nonlinear functions are evaluated using $N f_m(V_k \tilde{x}(t))$. Note that the dimension k of

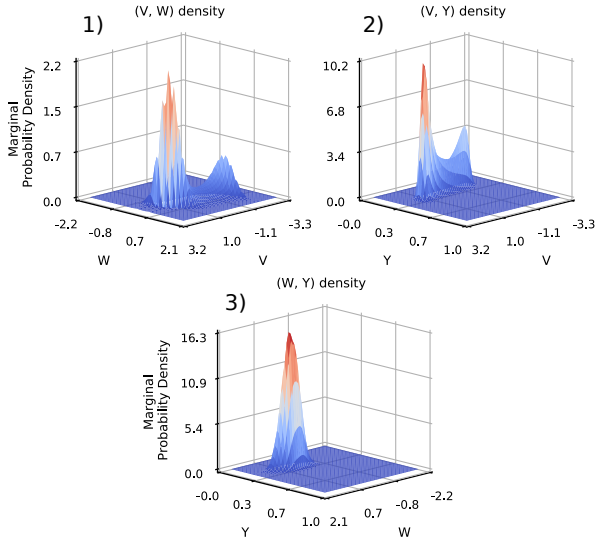

Fig. 1. Original model simulated for $t = 2.2\,\mathrm{s}$. The probability density function is integrated over each of the three independent variables, Y, W and V (plots 1) to 3) respectively) for visualization purposes.

the linear component of the reduced system does not need to equal the dimension m of the nonlinear component.

Another algorithm we use to reduce the mean-field model is QDEIM [23]. In DEIM, the computed basis functions U_m and interpolation points P_m depend on the sequence in which snapshots were collected. The idea of QDEIM is to find P_m independent of this specific sequence, and to compute a more numerically stable basis M_m that replaces U_m. P_m and M_m are obtained with QR factorization of column pivoted U_m^*. Efficient implementations of these steps are readily available in high performance computing software packages.

Additionally, we reduce the model with Localized Discrete Empirical Interpolation Method (LDEIM) [24]. In LDEIM, a clustering algorithm is employed in the offline phase to group solution snapshots before DEIM basis generation. Several basis and interpolation point pairs $(U_{m_1}, P_{m_1}), \cdots, (U_{m_p}, P_{m_p})$ are computed to obtain a set of local bases, one from each cluster of snapshots, in contrast to the global basis used in DEIM. In the online phase a local precomputed DEIM basis is chosen with nearest neighbor classification. LDEIM requires the number of clusters and features as user defined parameters, and the size of the feature vector is a decision between classification power and computational efficiency. The premise of LDEIM is to use multiple smaller yet accurate reduced subspaces to compensate for the extra online computation time that is needed for basis selection.

In general, hardware requirements of our approach are decided by the model that is reduced. The reduction algorithms need additional memory proportionally to n, m and k. To simulate the models we use four Intel Xeon E5-2680 v3 cores and 350GB RAM. The original model requires a considerable amount of memory in the state space format as there are $n^2, n = \nu^3$ floats in A, here $\sim$ 125Gb for A. However, we did not exploit the sparsity of A to reduce memory load, thus

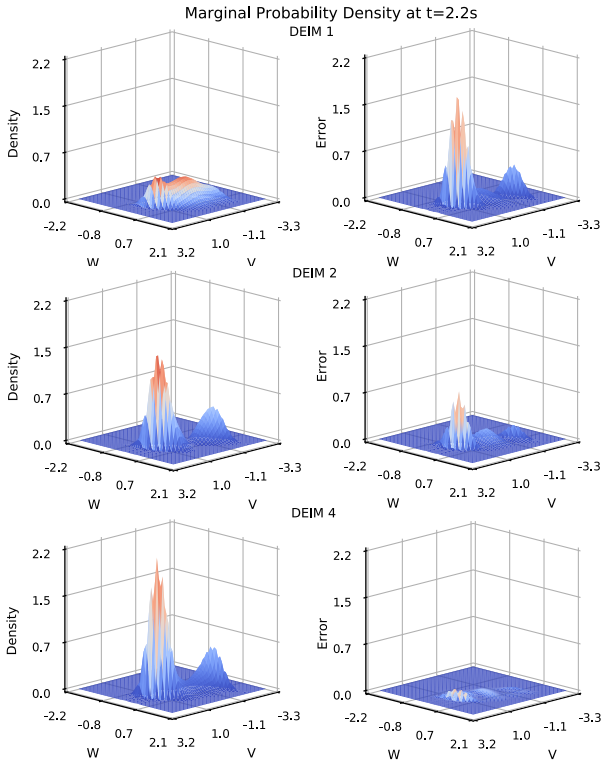

Fig. 2. Reduced models constructed with DEIM. Left column shows the approximation and right column the absolute difference between the state seen in Figure 1 in (V, W) space. Lower rows indicate approximation of higher dimension.

a more economical implementation is achievable. The models take advantage of multiple cores through differential equation solvers. We use the 4$^{\text{th}}$ order Runge-Kutta method with fixed step of $dt = 0.01\,\mathrm{s}$.

III. RESULTS

The original model from Equation (1) was simulated for $t = 2.2\,\mathrm{s}$ with the Gaussian distribution as the initial state. On average, this takes 200 minutes. The state of the model after simulation is seen in Figure 1. As the modeled probability density function has a 3-dimensional domain, for visualization purposes integration over one variable is required. In plot 1) of Figure 1, integration is done over the Y-variable, in plot 2) over W-variable and finally over V-variable in plot 3). The reduced models should reach the same state with minimal error. From this point forward, only the (V, W) space will be visualized.

Figure 2 shows reduction results with the DEIM method, after an approximation of the original system is reconstructed using the low-dimensional model. Left column shows the state of the reduced model integrated over the Y-variable, and right column shows the absolute difference between the original and reduced model at every point in the (V, W) space. Dimension of the reduced model grows in each row, with dimensions 1, 2 and 4 illustrated. It can be seen how the reduced model rapidly converges to the same solution as the original model.

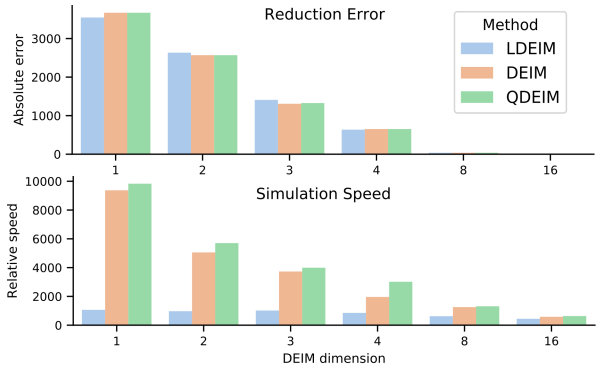

Fig. 3. Top row: approximation error from reduced models in (V, W) space. Bottom row: factor of acceleration gained using the reduced models. Reduced model dimension grows to the right and color indicates reduction method.

In Figure 3, upper row shows approximation error as the sum over point-wise absolute differences between the original and each reduced model in (V, W) space. Lower row indicates achieved speedup as median simulation time of each reduced model divided by the median time of the original model, with data from 20 simulations per model. Hue indicates MOR method. Error decreases with increasing dimension using any method, while the gained acceleration also declines. DEIM and QDEIM are seen to be equally accurate and fast, while LDEIM is slower due to the online adaptivity. The cost of adaptivity diminishes as the dimension of the reduced model increases. With the present model, LDEIM does not achieve an increase in accuracy compared to DEIM and QDEIM of similar dimensions.

IV. DISCUSSION AND CONCLUSIONS

Mathematical model order reduction (MOR) of nonlinear systems in neuroscience has been studied before in limited settings. In [25]–[27] a model of a branching neuron was reduced. A chemical reaction based model of synaptic plasticity was reduced in [28]. MOR studies of nonlinear neuronal populations have been conducted in a hippocampal network [29], in cardiac and muscular systems [30], [31] and in mono- and bidomain electromyography models [32], [33]. Mean-field models have been reduced in fluid dynamics in the context of e.g. water flow and flame behaviour [34], [35]. The authors are not aware of previous applications of mathematical MOR to neuronal Fokker-Planck mean-field models.

We have shown how the numerical simulation of a high-dimensional neuronal mean-field model can be accelerated significantly by the use of mathematical MOR methods. We achieved an improvement of over three orders of magnitude in simulation time, with low approximation error. Performing this type of numerical approximation does not render the model any less biologically relevant, as happens with simplification approaches that remove variables and make assumptions about the dynamics of the system. We note that the magnitude of acceleration gained using reduced models depends on the number of discretization points required to numerically solve the partial differential equation. To accurately reach a steady-state solution a fine grid is required, and the potential speedup is greater. The main bottleneck is the rapidly growing memory consumption of the original model.

Improving the simulation time of computational models of neural populations is important as the number of neural cells, and hence variables in the model, must be large in order to reach satisfactory levels of biological realism. While Fokker-Planck-type mean-field models make the study of stochastic networks easier, solving them requires spatial discretization over every variable of the neuron model, resulting in a rapidly growing number of dimensions and hence long computation times. In [36] the present model was simulated efficiently with graphical processing units (GPUs). The alternative approach described in this study does not rely on increasing hardware resources and instead improves simulation time with mathematical methods based on low-dimensional subspace approximation.

The present, purely software-based implementation of mathematical MOR methods is especially interesting in terms of integration into neuronal simulators such as NEURON [37], NEST [38] and The Virtual Brain [39]. In these simulators, approximated models could be made readily available as components for network or compartmental cell simulations. Alternatively, the simulator software could compute reduced models during long simulations and finish the simulation efficiently using the low-dimensional model. Integration is feasible because MOR methods do not require any special hardware, and software only needs to support matrix arithmetic and differential equation solvers.

Neuromorphic hardware is state-of-the-art in low energy computation and is used in neuroscience, robotics and artificial intelligence research [3], [4]. Neuromorphic chips have parallels with MOR methods in striving for accelerated simulation of (neuronal) models. Additionally, implementing reduced models on neuromorphic hardware, such as the SpiNNaker system [3], could enable the study of neuronal networks in even larger scales than before. However, not all neuromorphic chips can be combined with MOR methods.

Based on our results, we suggest MOR methods to be applied to other mean-field models to see whether we can reproduce such results with different populations of neurons. An excellent candidate would be the Hodgkin-Huxley neuronal mean-field model discussed in [16], for which applying the present method is straightforward. Follow-up studies should also address the situation when simulation parameters differ from parameters used in the snapshot collection phase. Additionally, the benefit gained from the methods introduced in [40] should be addressed in the context of computational neuroscience.

ACKNOWLEDGMENTS

Funding: 298182, 297893, 310489 (Academy of Finland; M.-L.L., L.P), 785907 (EU FET Flagship Human Brain Project; M.-L.L.), TUNI Graduate School (M.L.).

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[2] D. Buonomano and W. Maass, "State-dependent computations: spatiotemporal processing in cortical networks," *Nature Reviews Neuroscience*, vol. 10, no. 2, p. 113, 2009.

[3] S. Furber, F. Galluppi, S. Temple, and L. Plana, "The SpiNNaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[4] S. van Albada, A. Rowley, J. Senk, M. Hopkins, M. Schmidt, A. Stokes, D. Lester, M. Diesmann, and S. Furber, "Performance comparison of the digital neuromorphic hardware SpiNNaker and the neural network simulation software NEST for a full-scale cortical microcircuit model," *Frontiers in Neuroscience*, vol. 12, p. 291, 2018.

[5] F. Zenke, E. Agnes, and W. Gerstner, "Diverse synaptic plasticity mechanisms orchestrated to form and retrieve memories in spiking neural networks," *Nature Communications*, vol. 6, p. 6922, 2015.

[6] J. Guerguiev, T. P. Lillicrap, and B. Richards, "Towards deep learning with segregated dendrites," *ELife*, vol. 6, p. e22901, 2017.

[7] A. Marblestone, G. Wayne, and K. Kording, "Toward an integration of deep learning and neuroscience," *Frontiers in Computational Neuroscience*, vol. 10, p. 94, 2016.

[8] W. Maass, "Searching for principles of brain computation," *Current Opinion in Behavioral Sciences*, vol. 11, pp. 81–92, 2016.

[9] G. Deco, V. Jirsa, P. Robinson, M. Breakspear, and K. Friston, "The dynamic brain: from spiking neurons to neural masses and cortical fields," *PLoS Computational Biology*, vol. 4, no. 8, p. e1000092, 2008.

[10] P. Benner, S. Gugercin, and K. Willcox, "A survey of projection-based model reduction methods for parametric dynamical systems," *Society for Industrial and Applied Mathematics Review*, vol. 57, no. 4, pp. 483–531, 2015.

[11] M. Churchland and K. Shenoy, "Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex," *Journal of Neurophysiology*, vol. 97, no. 6, pp. 4235–4257, 2007.

[12] K. Wiesenfeld and F. Moss, "Stochastic resonance and the benefits of noise: from ice ages to crayfish and SQUIDs," *Nature*, vol. 373, no. 6509, p. 33, 1995.

[13] T. Manninen, M.-L. Linne, and K. Ruohonen, "Developing Itô stochastic differential equation models for neuronal signal transduction pathways," *Computational Biology and Chemistry*, vol. 30, no. 4, pp. 280–291, 2006.

[14] A. Saarinen, M.-L. Linne, and O. Yli-Harja, "Stochastic differential equation model for cerebellar granule cell excitability," *PLoS computational biology*, vol. 4, no. 2, p. e1000004, 2008.

[15] A. Destexhe and D. Contreras, "Neuronal computations with stochastic network states," *Science*, vol. 314, no. 5796, pp. 85–90, 2006.

[16] J. Baladron, D. Fasoli, O. Faugeras, and J. Touboul, "Mean-field description and propagation of chaos in networks of Hodgkin-Huxley and FitzHugh-Nagumo neurons," *The Journal of Mathematical Neuroscience*, vol. 2, no. 1, p. 10, 2012.

[17] R. FitzHugh, "Impulses and physiological states in theoretical models of nerve membrane," *Biophysical Journal*, vol. 1, no. 6, pp. 445–466, 1961.

[18] J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon," *Proceedings of the IRE*, vol. 50, pp. 2061–2070, 1962.

[19] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, "An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations," *Comptes Rendus Mathematique*, vol. 339, no. 9, pp. 667–672, 2004.

[20] S. Chaturantabut and D. Sorensen, "Nonlinear model reduction via discrete empirical interpolation," *SIAM Journal on Scientific Computing*, vol. 32, no. 5, pp. 2737–2764, 2010.

[21] J. Lumley, G. Berkooz, and P. Holmes, "The Proper Orthogonal Decomposition in the analysis of turbulent flows," *Annual Review of Fluid Mechanics*, vol. 25, pp. 539–575, 1993.

[22] L. Sirovich, "Turbulence and the dynamics of coherent structures. I-III." *Quarterly of Applied Mathematics*, vol. 45, no. 3, pp. 561–590, 1987.

[23] Z. Drmac and S. Gugercin, "A new selection operator for the Discrete Empirical Interpolation Method—improved a priori error bound and extensions," *SIAM Journal on Scientific Computing*, vol. 38, no. 2, pp. A631–A648, 2016.

[24] B. Peherstorfer, D. Butnaru, K. Willcox, and H. Bungartz, "Localized discrete empirical interpolation method," *SIAM Journal on Scientific Computing*, vol. 36, no. 1, 2014.

[25] A. Kellems, S. Chaturantabut, D. Sorensen, and S. Cox, "Morphologically accurate reduced order modeling of spiking neurons," *Journal of Computational Neuroscience*, vol. 28, no. 3, 2010.

[26] B. Du, D. Sorensen, and S. Cox, "Model reduction of strong-weak neurons," *Frontiers in Computational Neuroscience*, vol. 8, no. 164, 2014.

[27] D. Amsallem and J. Nordström, "Energy stable model reduction of neurons by nonnegative discrete empirical interpolation," *SIAM Journal on Scientific Computing*, vol. 38, no. 2, pp. B297–B326, 2016.

[28] M. Lehtimäki, L. Paunonen, S. Pohjolainen, and M.-L. Linne, "Order reduction for a signaling pathway model of neuronal synaptic plasticity," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7687–7692, 2017.

[29] M. Lehtimäki, L. Paunonen, and M.-L. Linne, "Projection-based order reduction of a nonlinear biophysical neuronal network model," in *2019 Proceedings of the IEEE Conference on Decision and Control (CDC)*. IEEE, accepted.

[30] M. Boulakia, E. Schenone, and J.-F. Gerbeau, "Reduced-order modeling for cardiac electrophysiology. Application to parameter identification," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 28, no. 6-7, pp. 727–744, 2012.

[31] H. Yang and A. Veneziani, "Efficient estimation of cardiac conductivities via pod-deim model order reduction," *Applied Numerical Mathematics*, vol. 115, pp. 180–199, 2017.

[32] M. Mordhorst, T. Strecker, D. Wirtz, T. Heidlauf, and O. Röhrle, "Pod-deim reduction of computational emg models," *Journal of Computational Science*, vol. 19, pp. 86–96, 2017.

[33] N. Emamy, P. Litty, T. Klotz, M. Mehl, and O. Röhrle, "POD-DEIM model order reduction for the monodomain reaction-diffusion sub-model of the neuro-muscular system," in *IUTAM Symposium on Model Order Reduction of Coupled Systems, Stuttgart, Germany, May 22–25, 2018*, J. Fehr and B. Haasdonk, Eds. Cham: Springer International Publishing, 2020, pp. 177–190.

[34] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza, "Model order reduction in fluid dynamics: challenges and perspectives," in *Reduced Order Methods for modeling and computational reduction*. Springer, 2014, pp. 235–273.

[35] S. Mowlavi and T. P. Sapsis, "Model order reduction for stochastic dynamical systems with continuous symmetries," *SIAM Journal on Scientific Computing*, vol. 40, no. 3, pp. A1669–A1695, 2018.

[36] J. Pezoa, J. Baladron, D. Fasoli, and O. Faugeras, "Three applications of GPU computing in neuroscience," *Computing in Science & Engineering*, vol. 14, no. 3, pp. 40–47, 2011.

[37] M. Hines and N. Carnevale, "The neuron simulation environment," *Neural Computation*, vol. 9, no. 6, pp. 1179–1209, 1997.

[38] M.-O. G. and M. D., "Nest (neural simulation tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.

[39] P. Sanz Leon, S. Knock, M. Woodman, L. Domide, J. Mersmann, A. McIntosh, and V. Jirsa, "The Virtual Brain: a simulator of primate brain network dynamics," *Frontiers in Neuroinformatics*, vol. 7, p. 10, 2013.

[40] B. Peherstorfer, Z. Drmač, and S. Gugercin, "Stabilizing discrete empirical interpolation via randomized and deterministic oversampling," *arXiv preprint arXiv:1808.10473*, 2018.

PUBLICATION
IV

Accelerating Neural ODEs using model order reduction

M. Lehtimäki, L. Paunonen, and M.-L. Linne

IEEE Transactions on Neural Networks and Learning Systems, pp. 1–13
DOI: https://doi.org/10.1109/TNNLS.2022.3175757

Accelerating Neural ODEs Using Model Order Reduction

Mikko Lehtimäki, Lassi Paunonen, *Senior Member, IEEE*, and Marja-Leena Linne, *Member, IEEE*

***Abstract*— Embedding nonlinear dynamical systems into artificial neural networks is a powerful new formalism for machine learning. By parameterizing ordinary differential equations (ODEs) as neural network layers, these Neural ODEs are memory-efficient to train, process time series naturally, and incorporate knowledge of physical systems into deep learning (DL) models. However, the practical applications of Neural ODEs are limited due to long inference times because the outputs of the embedded ODE layers are computed numerically with differential equation solvers that can be computationally demanding. Here, we show that mathematical model order reduction (MOR) methods can be used for compressing and accelerating Neural ODEs by accurately simulating the continuous nonlinear dynamics in low-dimensional subspaces. We implement our novel compression method by developing Neural ODEs that integrate the necessary subspace-projection and interpolation operations as layers of the neural network. We validate our approach by comparing it to neuron pruning and singular value decomposition (SVD)-based weight truncation methods from the literature in image and time-series classification tasks. The methods are evaluated by acceleration versus accuracy when adjusting the level of compression. On this spectrum, we achieve a favorable balance over existing methods by using MOR when compressing a convolutional Neural ODE. In compressing a recurrent Neural ODE, SVD-based weight truncation yields good performance. Based on our results, our integration of MOR with Neural ODEs can facilitate efficient, dynamical system-driven DL in resource-constrained applications.**

***Index Terms*— Acceleration, compression, discrete empirical interpolation method (DEIM), neural ordinary differential equations (Neural ODEs), proper orthogonal decomposition (POD), reduced order model (ROM).**

I. Introduction

DEEP learning (DL) is reaching and surpassing human performance in domain-specific applications [1]. Accordingly, there is increased demand for including DL-based algorithms into consumer devices that may contain only limited computational capacity and may be battery-powered, making resource efficiency of DL-based algorithms important. An interesting new development in DL research is artificial neural networks (ANNs) that employ dynamical systems, replacing traditional discrete layers with a continuous-time layer in the form of ordinary differential equations (ODEs) [2]–[5]. In these Neural ODEs, the continuous formalism allows flexible processing of time-series and irregularly sampled data, such as medical and physiological signals [6]. Moreover, Neural ODEs are useful for resource-constrained and embedded applications because they are very memory and parameter efficient [5]. The ODE layer also facilitates engineering physical details, such as energy conservation laws or spectral properties, into neural networks [4]. However, often, a big computational bottleneck in Neural ODEs is the dynamical system layer since propagating data through the system requires many evaluations using numerical ODE solvers. Reducing the computational cost of the ODE layer is the main motivation of our work.

In this work, we show that Neural ODEs can be accelerated by compressing them using model order reduction (MOR) methods. The MOR approach is based on projecting dynamical systems onto low-dimensional subspaces. Here, we develop MOR methods that are integrated into Neural ODEs. In this manner, we lower the required storage size, memory consumption, multiply–adds, and nonlinear activation count needed to compute predictions from input data. The resulting compressed Neural ODEs can be deployed for real-time computing and devices where energy efficiency is paramount. In order to validate the performance of our MOR method, we compare it to two established compression methods from the literature. Our results demonstrate that MOR is a theoretically grounded and effective method for accelerating Neural ODEs because it achieves a favorable, adjustable and extensible balance between speedup and accuracy of compressed models. We show this result in two different classification tasks that use different Neural ODE architectures: a convolutional and a recurrent neural network (RNN).

Compressing ANNs is one of the principal ways of converting high-performing trained networks into more efficient networks since good accuracy can often be recovered without long training times, while the size of the compressed network can be chosen in a flexible manner. Several neural network compression methods have been proposed for accelerating ANNs [7]–[9]. These include singular value decomposition (SVD)-based weight truncation [10], [11] and neuron pruning

Manuscript received May 28, 2021; revised February 23, 2022; accepted May 5, 2022. The work of Mikko Lehtimäki was supported in part by the Tampere University Graduate School and in part by the Finnish Foundation for Technology Promotion. The work of Lassi Paunonen was supported by the Academy of Finland under Grant 310489. The work of Marja-Leena Linne was supported by the Academy of Finland under Grant 297893. This work was supported by the European Union's Horizon 2020 Framework Program for Research and Innovation under the Specific Grant 785907 (Human Brain Project SGA2) and Grant 945539 (Human Brain Project SGA3). *(Corresponding author: Mikko Lehtimäki.)*

Mikko Lehtimäki and Marja-Leena Linne are with the Faculty of Medicine and Health Technology, Tampere University, 33100 Tampere, Finland (e-mail: mikko.lehtimaki@tuni.fi).

Lassi Paunonen is with the Faculty of Information Technology and Communication Sciences, Tampere University, 33100 Tampere, Finland.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3175757.

Digital Object Identifier 10.1109/TNNLS.2022.3175757

with an importance score based on zero activations [9], which we implement for comparison to our MOR method. Many prominent existing approaches rely on assigning importance scores to neurons based on their connection weights or output values. However, we argue that such methods are nonoptimal for compressing Neural ODEs, as it is difficult to quantify the importance of neurons in the ODE layer based on criteria such as the output of the layer alone. This is because the final state of the nonlinear ODE system gives no information about the dynamics of the actual computations. In contrast, MOR methods are designed precisely for the approximation of state trajectories and input–output relationships in dynamical systems, and hence, they can overcome the shortcomings of existing compression methods when aiming to accelerate Neural ODEs.

Our approach to model acceleration is inspired by computational neuroscience. Due to high computational burden, modeling studies of the brain in realistic scales are limited to simulating small fractions of the brain using supercomputers [12]. To overcome computational resource challenges, MOR methods have been adapted successfully for reducing single neuron [13], [14], synapse [15], and neuronal population [16], [17] models. Moreover, the similarities in models of the brain as well as ANNs, which include connectivity patterns and nonlinear computation units, make us hypothesize that MOR methods may be a principled approach for accelerating Neural ODEs.

Our MOR approach for compressing Neural ODEs is based on the proper orthogonal decomposition (POD) [18] with the discrete empirical interpolation method (DEIM) [19], a variant of the empirical interpolation method [20]. Using the POD-DEIM method, we derive reduced order models (ROMs) that can be simulated efficiently in low-dimensional subspaces, even in the presence of nonlinear activation functions. In the context of Neural ODEs, our method compresses the ODE block by projecting it onto a low-dimensional subspace using POD, which reduces the number of state variables in the ODE system. Linear operations are compressed by transformation into this subspace. When nonlinear activation functions are present, using DEIM, we determine the most informative output neurons based on their time dynamics and prune the other neurons. This reduces the dimensionality of the nonlinear operation and removes rows from the weight matrix since connection weights of pruned neurons are discarded. We interpolate an approximate response for the pruned neurons directly in the low-dimensional subspace. In convolutional layers, the reduced model computes convolutions only at the selected interpolation points so that every kernel has its own set of evaluation coordinates.

We integrate the subspace-projection and interpolation steps of POD-DEIM into the Neural ODEs as layers, and this introduces additional operations into the neural network (see Fig. 2). In some network architectures, these steps actually further reduce the overall number of multiply–adds in the neural network. The POD-DEIM reduction is applied after training the model and the reduced model can be fine-tuned for increased accuracy. In summary, Neural ODEs allow us to bridge a gap between ANN research and control theory research so that a substantial amount of previously unemployed knowledge in model reduction can be utilized in ANN compression and acceleration. For example, analytical optimality results and error bounds exist for our MOR algorithms [18], [21].

In Section II, we review previous work in compressing and accelerating neural networks in general and Neural ODEs specifically. In Section III, we present our proposed MOR approach and show how to formulate it in the context of continuous neural networks and introduce two established acceleration methods that we compare our method to in benchmark problems. In Section IV, we provide theoretical compression ratios for the chosen methods and show actual accuracy and wall-time metrics of compressed Neural ODEs in two different classification tasks: one using a convolutional and the other using a recurrent ODE architecture. We discuss the success of the model reduction approach and the significance of our work in Section V with future suggestions and conclude in Section VI.

II. Related Work

Several studies have reported that ANNs contain structures and redundancies that can be exploited for reducing memory and power usage [7]. Here, we focus on structural acceleration approaches that modify trained networks to achieve a more efficient architecture and leave efforts, such as low precision algebra, quantization of weights [22], binarization [23], hashing [24], vectorization [25], frequency space evaluation [26], and adjusting ODE solver tolerances or step size [5] out of the present study, since those are complementary to the approaches presented here. Moreover, several hardware accelerators have been proposed (e.g., [27]), and those will not be addressed here. Furthermore, computational bottlenecks in ANNs have also been addressed by first reducing data dimension and then training simpler models. These techniques range from feature engineering to data dimensionality reduction. However, in this work, our focus is on compressing trained networks, and hence, data compression is considered complementary to our approach. Structural compression methods can be further grouped into several categories.

a) Pruning weights: Prior work has addressed weight pruning [28], [29] and enforcing weight sparsity [30], [31] during training to obtain weight matrices that require less storage space and memory than dense weight matrices. Several methods to evaluate weight importance have been proposed, see, for example, a recent review addressing 81 pruning studies [7] and compares the achieved compression rates and speedups for several ANN models. It is common to include pruning in the training loop because altering weights after training leads to accuracy loss. In order to achieve significant acceleration with weight-pruning methods, the use of special software, masking strategies, sparse algebra, or hardware accelerators is recommended [32], [33].

b) Decomposition: Decompositions and low-rank matrix factorizations have been used for compressing network weights so that linear operations in a layer are computed efficiently [10], [11], [34]–[36]. Decomposition is based on the observation that weight matrices, especially as their size

increases, are seldom full rank. The idea is that a fully connected weight matrix or a tensorized convolutional kernel can be decomposed into compact low-rank matrices that require less storage space, memory, and multiply–adds during training and testing. An approximation of the original operation is then obtained as a product of the low-rank tensors. In addition to compressing linear operations, prior work has also addressed decomposition by taking the nonlinear response into account [37]. The cost of decomposition approaches is that a single layer is replaced with multiple smaller ones, which trades parallelism for forward operations, canceling out some of the obtained acceleration.

c) Pruning neurons: Weight pruning and decomposition approaches maintain the structure of the compressed network in which the number of nonlinear activation functions is not changed. Eliminating activation functions leads to acceleration as an entire row of weights from a fully connected layer can be removed [8], [38]. As neuron pruning changes the number of neurons in a layer, the next layer must take this into account, for example, by deleting columns (inputs) from the weight matrix in the case of fully connected layers. Alternatively, interpolation can be used to approximate the original response, possibly introducing additional computation. In fully connected architectures, large compression rates in storage space and memory are obtained when rows or columns are deleted from fully connected layers. Importance scores, such as percent of zero activations, have been developed for identifying prunable neurons [9]. Pruning and quantization have been combined with Huffman coding into a compression framework that delivers memory and energy efficiency [39]. An optimization approach has been used to enforce sparse columns in a fully connected layer so that the corresponding input neurons can be pruned [38].

d) Pruning filters: Convolutional neural networks (CNNs) have attracted a lot of attention in the compression literature. This is not surprising given their good rate of success in real-life tasks and their high computational cost. Convolutional operations make the bulk of modern image and video processing networks and are used in many other applications, such as sequence processing. It is possible to approach CNN compression by analyzing either the convolutional filters or feature maps obtained by applying the filters on input data [40]–[42]. The filters can be compressed with decomposition methods, similar to fully connected layers. Alternatively, entire kernels can be pruned from filters. Pruning kernels is very effective as intermediate feature maps are eliminated. The number of methods and criteria available for identifying pruning targets in CNNs is very high [42]. In convolutional networks, neuron pruning corresponds to skipping a kernel evaluation at a single spatial location. However, such an operation is rarely supported by DL frameworks, and pruning in convolutional networks has focused on eliminating parts of or entire filters or feature maps.

In the context of Neural ODEs, a few studies have addressed the acceleration of inference and training times. One approach focuses on learning simple dynamics that do not burden ODE solvers as much as stiff systems [43]–[45]. In addition, training times have been reduced by further improving the adjoint method, for example, by relaxing error criteria [46], while inference times have been accelerated by introducing hypersolvers—neural networks that solve Neural ODEs [47]. These methods are complementary to ours since we aim at compressing the learned architecture with MOR methods. Our approach is comparable to pruning neurons based on an importance score [8], [9] so that a number of rows from the weight matrix and nonlinear activation functions can be removed altogether. The POD-DEIM method improves on the existing methods of determining neuron importance, as it accounts for the complete dynamics of the ODE block. In addition, POD-DEIM can be combined with many of the prior methods such as quantization of weights.

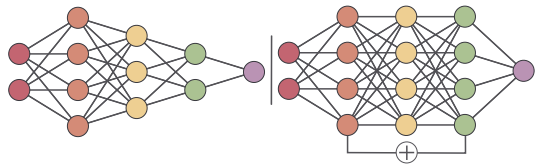

Fig. 1. Plain discrete feedforward network on the left and a residual network on the right. The defining feature of residual networks is that the output from an earlier layer skips layers and is added directly to a later layer.

III. Methods

In this section, we present the theory of the continuous interpretation of ANNs. We then describe our MOR method and show how to formulate it in the ANN and Neural ODE context. Finally, we present two other ANN acceleration methods from the literature, which we use as benchmarks to our method, and their implementation in the Neural ODE setting.

A. Continuous Networks

Since the success of residual neural networks (ResNets) [48], a continuous interpretation of neural networks has gained traction. The ResNet architecture utilizes skip connections to deal with the problem of vanishing gradients, allowing the training of extremely deep neural networks. In a ResNet block, the output is the sum of the usual feedforward operations on the data and the unprocessed data entering the block. The ResNet architecture is shown in Fig. 1 (right), with a plain feedforward network on the left. If the hidden layers of the plain network implement a nonlinear function $x_{k+1} = f(x_k)$, the skip connection of the ResNet implement $x_{k+1} = x_k + f(x_k)$. By introducing a constant $h = 1$ to obtain $x_{k+1} = x_k + hf(x_k)$, the resemblance of the skip connection to Euler's formula for solving ODEs is seen [2]–[4]. This has led to the continuous-time dynamical system interpretation of ANNs.

In the continuous interpretation of ANNs, a set of layers is replaced by one layer that parameterizes a dynamical system as a group of ODEs with state variable $x(t)$. We assume that this ODE system has the form

$$x'(t) = f(x(t), u(t), \theta) \tag{1}$$

in general and in our models specifically

$$x'(t) = f(Ax(t) + b) + Zu(t) \tag{2}$$

where $x(t) \in \mathbb{R}^n$ is the state of the ODE at time t, $x'(t)$ is the time derivative of $x(t)$, $A \in \mathbb{R}^{n\times n}$ is a weight matrix, $b \in \mathbb{R}^n$ is a bias vector, $u(t) \in \mathbb{R}^i$ is a time-dependent input, $Z \in \mathbb{R}^{n\times i}$ is an input matrix (may include bias), and $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ is a vector-valued function $f(\chi) = [f_a(\chi_1), f_a(\chi_2), \ldots, f_a(\chi_n)]^T$. In Neural ODEs, the activation function $f_a(\chi_i)$ is commonly the hyperbolic tangent, although any differentiable activation function, or a combination of different activation functions, can be used. Here, $\theta = (A, b, Z)$ are learned parameters of the ODE. In feedforward Neural ODEs, there are typically no time-dependent inputs and, hence, $Z = 0$. On the other hand, in RNNs where $x'(t)$ is the hidden state, the input data enter the system through $Z \neq 0$. The output of the layer is $x(t_{end})$, which is the state of the dynamical system at the user-specified final time t_{end}. Initial values of the ODE system can be obtained in two ways, either as the output of the layer preceding the ODE or set explicitly. The former is more common in feedforward architectures and the latter in RNNs that receive time-dependent input data. The output $x(t_{end})$ of the ODE layer is computed using numerical methods, taking discrete or adaptive steps with an ODE solver to solve an initial value problem. Neural ODEs can use the adjoint method of calculating gradients [5] and have enabled parameterizing ODEs as several different ANN operations or chains of them. The primary restriction is that the output size of the ODE layer must match the input size of the layer. Overall, it is possible to train a variety of ANN architectures for different tasks as continuous networks [5].

Training Neural ODEs with the adjoint method is memory efficient compared to deep discrete networks [5], as the backpropagation algorithm does not need to store intermediate ODE states to calculate gradients on the loss function. However, the use of the adjoint method is not required for training ODE networks. Parameterizing an ODE in place of several discrete layers may also lead to parameter efficiency and correspondingly require less storage space and memory since the continuous layer can replace several individually parameterized discrete layers. Other benefits of Neural ODEs include using ODE solvers for speed versus accuracy tuning and enabling ANNs to flexibly process continuous and irregularly sampled time-series data [5], [6]. Neural ODEs have also improved on existing ANN-based density estimation models [5]. However, the dynamics learned by Neural ODEs may be unnecessarily complex [44], [45] and result in stiff systems [49]. Often, the ODE block is the most computationally demanding part of the network since many numerical evaluations of the state of the ODE are needed to obtain the output. Hence, Neural ODEs require more time to evaluate data, both in training and testing, compared to traditional discrete networks [44].

It is possible to parameterize the ODE block so that the model in training has favorable properties that facilitate learning. Antisymmetric networks are a step toward this direction since they guarantee that the state $x(t)$ of the ODE system does not diverge far from or decay to the origin as $t \to \infty$ [4]. This prevents the gradient of the loss function from vanishing or exploding and makes training the network well-posed. Antisymmetric networks use an antisymmetric weight matrix $A = W - W^T$ that by definition has eigenvalues λ_i so that for all $i, \mathrm{Re}(\lambda_i(A)) = 0$. Furthermore, a small shift of the eigenvalues by γ may be applied so that $\mathrm{Re}(\lambda_i(A - \gamma I)) = -\gamma < 0$, which improves the behavior of the ODE system in the presence of noisy data [4]. Notice that in practical applications, t is finite, and hence, a small γ value will not make the gradients of the loss function vanish. In [50], it is demonstrated that the shifted antisymmetric weight matrix $A - \gamma I$ gives the hidden state of RNNs a favorable property of long-term dependence on the inputs to the system, which helps classifying data with temporal relationships. In this work, we implement an ODE-RNN as

$$\begin{aligned} A &= W - W^T - \gamma I \\ x'(t) &= \tanh(Ax(t) + b) + Zu(t) \end{aligned} \tag{3}$$

where the weight matrix A gives the network the desired properties that facilitate learning, and during training, the parameters W are learned. Other parameters are similar to (2). In Section IV, we will demonstrate the compression and acceleration of this ODE-RNN. Such networks make an interesting model compression target since their architecture gives the system temporal memory capacity that the compressed model must retain.

B. Model Order Reduction

A key contribution of our work is the formalization of MOR in the context of Neural ODEs and the realization that the necessary MOR operations can be expressed as layers of ANNs. A powerful method for MOR of general nonlinear systems is POD [18] coupled with the DEIM [19], [20], developed in the fields of systems and control theory. In order to compress Neural ODEs, we construct ROMs of the ODE block in trained Neural ODEs with the POD-DEIM method.

Both POD and DEIM utilize the method of *snapshots* [51]. In POD, snapshots are values of the state x_t of the ODE system at discrete times t. A snapshot matrix $X = [x_1, x_2, \ldots, x_s]$ is collected using numerical simulation of the ODE block with different initial values and time-dependent input data. ANNs provide a very natural setting for gathering snapshots since we have access to training data that can be propagated through the trained network and the ODE block of Neural ODEs. However, it is important that the snapshots are collected after the model is trained so that the snapshots reflect true learned dynamics of the ODE layer. Moreover, the snapshots are not used for optimization or model training. The following explains the purpose of the snapshots.

POD approximates the original system of dimension n via projection using a low-dimensional subspace. A k-dimensional POD basis with orthonormal column vectors $V_k \in \mathbb{R}^{n\times k}$, where $k < n$, is computed using SVD of snapshots $V\Sigma\Psi^T = \mathrm{SVD}(X)$ [51]. V_k is then the first k left singular vectors of the snapshot matrix, equaling the first k columns of V. A reduced state vector $V_k^T x(t) = \tilde{x}(t) \in \mathbb{R}^k$ is obtained by a linear

Algorithm 1 Discrete Empirical Interpolation Method

INPUT: $\{u_l\}_{l=1}^{m} \subset \mathbb{R}^n$ linearly independent
OUTPUT: $\vec{p} = [p_1, \ldots, p_m]$, $P \in \mathbb{R}^{n \times m}$

$p_1 = \text{argmax}(|u_1|)$
$U = [u_1]$, $P = [e_{p_1}]$, $\vec{p} = [p_1]$
for l = 2 to m **do**
  solve $(P^T U)c = P^T u_l$ for c
  $p_l = \text{argmax}(|u_l - Uc|)$
  $U \leftarrow [U\ u_l]$, $P \leftarrow [P\ e_{p_l}]$, $\vec{p} \leftarrow [\vec{p}\ p_l]$
end for

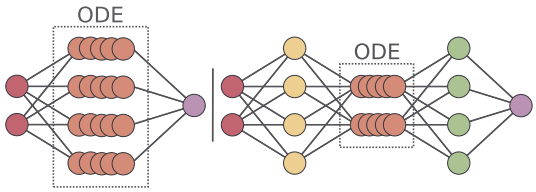

Fig. 2. Neural ODE on the left with the discretized differential equation block in orange color. A POD-DEIM reduced network on the right illustrates that the ODE block is evaluated in a low-dimensional subspace, with linear transformations around the ODE block. The networks have equal inputs and approximately equal outputs.

transformation, and at any time t, an approximation of the original, full-dimensional state vector can be computed with $x(t) \approx V_k \tilde{x}(t)$. Projecting the system in (2) onto V_k by the Galerkin projection results in a reduced system

$$\tilde{x}'(t) = V_k^T f(AV_k\tilde{x}(t) + b) + V_k^T Z u(t) \tag{4}$$

where we can precompute the transformation of the weight and input matrices into matrices $\tilde{A} = AV_k$ and $\tilde{Z} = V_k^T Z$ that are used in place of the original weight matrices in the reduced models. The reduced POD model approximates the original system optimally in the sense that the POD subspace has minimum snapshot reconstruction error [18]. However, the nonlinear form of the equation prevents computational savings as the number of neurons has not been reduced. The size of A is still $n \times k$ and $f(\cdot)$ is computed in the original dimension n. This is known as the lifting bottleneck in reducing nonlinear models.

Efficient evaluation of the nonlinear term can be achieved with DEIM [19], [20]. DEIM extends the subspace projection approach with an interpolation step for general nonlinear functions. To construct an interpolated approximation of the nonlinear term, we use

$$\tilde{f}(x,t) \approx U_m \left(P_m^T U_m\right)^{-1} P_m^T f(x,t) \tag{5}$$

where the DEIM basis vectors $U_m = [u_1, u_2, \ldots, u_m] \in \mathbb{R}^{n \times m}$, $m < n$ are the first m left singular vectors of the snapshot matrix of nonlinear vector-valued function outputs $F = [f(x_1,t_1), f(x_2,t_2), \ldots, f(x_s,t_s)]$ computed via SVD as $U_m \Sigma_U \Psi_U^T = F$, and $P_m^T f(x,t) := f_m(x,t)$ is a nonlinear function with m components chosen from f according to DEIM determined interpolation points $\vec{p} = p_1, \ldots, p_m$ and $P_m = [e_{p_1}, e_{p_2}, \ldots, e_{p_m}]$ with e_{p_i} being the standard basis vector i of $\mathbb{R}^n$. Together POD and DEIM form an ROM

$$\tilde{x}'(t) = \underbrace{V_k^T U_m \left(P^T U_m\right)^{-1}}_{N} f_m\left(\tilde{A}\tilde{x}(t) + b\right) + \tilde{Z}u(t) \tag{6}$$

where $N \in \mathbb{R}^{k \times m}$ can be precomputed. Now, in the prediction phase, only m nonlinear functions are evaluated. Due to the structure of $f_m : \mathbb{R}^m \to \mathbb{R}^m$, we can then further compress $\tilde{A}$ so that only m rows at indices $\vec{p}$ remain. Correspondingly, we only select m elements from the bias vector at indices $\vec{p}$. Thus, we have obtained the ROM

$$\tilde{x}'(t) = N f_m\left(\tilde{A}_m \tilde{x}(t) + b_m\right) + \tilde{Z}u(t) \tag{7}$$

where $\tilde{A}_m \in \mathbb{R}^{m \times k}$ and $b_m \in \mathbb{R}^m$. In (6), the dimension k of the POD projection subspace does not need to equal the number m of the DEIM interpolation points, as N can be computed even if $k \neq m$, although in practice, it is common to choose $m = k$. This means that m and k can be chosen either smaller or larger with respect to each other to reflect the complexity of linear and nonlinear functions of the model.

Algorithm 1 shows the process of determining the DEIM interpolation points and matrix P. The ordered linearly independent basis vector set given as input is the basis U_m that is obtained from snapshots of the nonlinear function. The argmax function returns the index of the largest value in a given vector and e_i is the ith standard basis vector. For more details, see the original reference [19], where an error bound for the DEIM approximation in the Euclidean space is also given. For an error estimate of the reduced state in POD-DEIM models specifically, see [21].

An essential part of our work is building the POD-DEIM subspace-projection operations V_k^T and V_k, reduced matrices $\tilde{A}_m$ and $\tilde{Z}$, and low-dimensional interpolation N of (7) into the Neural ODE. Consider a Neural ODE with an input layer, an ODE block, and a readout layer, as on the left of Fig. 2. We add two new layers: one for projecting the ODE block into a low-dimensional subspace and one for transforming the result of the ODE block back to the original space, as shown in the right of Fig. 2. These layers implement the computations $V_k^T x(0)$ and $V_k \tilde{x}(t_{end})$. The cost of these operations is offset by the cheaper evaluation of the ODE layer, which typically makes the bulk of the compute time of Neural ODEs. In the ODE layer, the reduced matrices $\tilde{A}_m$ and $\tilde{Z}$ replace the corresponding original large weight matrices and a new linear layer is added to implement the low-dimensional interpolation N. Furthermore, if the operations around the ODE block are linear, V_k^T and V_k can be computed into those existing layers, resulting in even cheaper online evaluation. Finally, we emphasize that this POD-DEIM process is widely applicable to different model architectures, as long as (1) defines the ODE block.

The DEIM method is known to exhibit lower accuracy in the presence of noise or perturbations in the snapshots. The input function $Zu(t)$ in ODE-RNNs can cause such nonsmooth behavior and a large number of timesteps taken are sensitive to instabilities. Hence, for compressing ODE-RNNs, we use an oversampling strategy that stabilizes the method as shown

in [52]. In this oversampled DEIM (ODEIM), the number of nonlinear sampling points in $\vec{p}$ becomes $m + o$ and is decoupled from the number of basis vectors m of the DEIM subspace. The matrix P then has size $n \times (m + o)$ and the matrix inverse of (6) is replaced with the Moore–Penrose pseudoinverse $(P^T U_m)^\dagger$. In ODEIM, interpolation becomes approximation via regression. During model evaluation, $m+o$ activation functions are evaluated. In our results, we evaluate $m+1$ nonlinear activation functions, which compared to vanilla DEIM has a slightly negative effect on model acceleration but a positive effect on accuracy of the compressed model.

The POD-DEIM method has adjustable parameters that affect the performance of the reduced model. The dimension k of the linear projection matrix and the number of interpolation points m are typically set according to the decay of the singular values of the snapshot matrices X and F, but both can be adjusted independently. The number of snapshots can be controlled in two ways: the number of samples of training data to use and the number of snapshots of model dynamics to save per sample. Computing the SVD of the snapshot matrix is the most computationally demanding step of the MOR pipeline, with memory being the typical bottleneck.

C. Benchmark Compression Methods

We compare the performance of the POD-DEIM method to two other ANN compression methods presented in the literature. To the best of our knowledge, no other studies using these methods to compress Neural ODEs have been published before.

The first method compresses the weight matrix of the ODE block into two low-rank matrices [10], [11] using SVD-based truncation. This reduces the overall number of multiply–adds needed to evaluate the layer but, unlike POD-DEIM, does not change the number of activation functions. Given a fully connected layer with activation $f(Ax + b)$ and $A \in \mathbb{R}^{n\times l}$, we apply SVD to the weight matrix to obtain $\Phi\Sigma\Psi^T = A$. Then, we only keep the first $k < n$ singular values by truncation so that $\Sigma_k = \text{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k\times k}$ and the first k left and right singular vectors so that $\Phi_k = \{\phi_1, \ldots, \phi_k\} \in \mathbb{R}^{n\times k}$ and $\Psi_k = \{\psi_1, \ldots, \psi_k\} \in \mathbb{R}^{l\times k}$. An approximation to A is then obtained as $\tilde{A} = \Phi_k \Sigma_k \Psi_k^T$. In ANNs, we can implement the approximation in two consequent fully connected layers that only use activation and bias after the last layer. The connection weights of the first layer are $\Sigma_k \Psi_k^T \in \mathbb{R}^{k\times l}$ and the second weights are Φ_k, which changes the parameter count from nl to $k(n+l)$. Notice that in a single-layer ODE block, we have $l = n$. This method is theoretically sound for compressing the ODE block. It retains most parameters out of the compression methods we test, meaning that it is expected to yield low-to-moderate acceleration. Since the weight truncation only uses the learned weights as inputs, the method does not depend on any other data, unlike POD-DEIM which requires gathering snapshots of the ODE dynamics.

The second method we used for comparison is based on computing an importance score for neurons, called average percent of zeros (APoZ) method, following [9], where it is suggested to prune neurons that have the largest number of zero activation values over the training or testing dataset. The corresponding rows of the weight, bias, and input matrices are then removed altogether. Moreover, with Neural ODEs specifically, for each removed row of the weight matrix, we also remove the corresponding column. The importance score of the cth neuron in a layer is defined as

$$\text{APoZ}_c = \frac{\sum_k^N \sum_j^M f\big(O_{c,j}(k) = 0\big)}{NM} \tag{8}$$

where N is the number of examples used to calculate the score and M is the dimension of the output feature map O. In Neural ODEs, it is common to use hyperbolic tangent activations, and hence, we adapt the scoring so that neurons with the lowest absolute activation magnitudes are eliminated. We compute the score after the last timestep of the ODE block, as this is the value that gets propagated to the following layers in the network. We only use training data for computing the score. This method does not account for the dynamics of the ODE block, as scoring is computed only at the last timestep of the ODE block, and hence, we evaluate the performance in detail here. Overall, this method applies the largest amount of compression out of the methods we test here.

IV. Results

We implemented two Neural ODE models and trained them on different classification tasks: one Neural ODE with a convolutional ODE block to classify digits of the MNIST dataset and one ODE-RNN to classify digits in the pixel-MNIST task, where the network is given one pixel at a time as input. We trained both models using data augmentation with random rotations and affine translations in order to prevent overfitting, using a decaying learning rate starting from 0.04, stochastic gradient descent optimizer, and cross-entropy loss function on image labels. After training, the POD-DEIM snapshots were collected by feeding each image in the training data to the network and saving the ODE state and activation function values at every second timestep. POD and DEIM bases were computed using a memory-efficient partitioning strategy [53]. Next, we compressed the models using POD-DEIM, APoZ trimming, and weight truncation and evaluated their performance directly after compression, after three epochs of fine-tuning and after excessive fine-tuning on training data. The performance of each compressed model was compared to the respective original model using Top-1 accuracy, Top-3 accuracy, and wall time as metrics. We restrict fine-tuning to the layers following the ODE block in order to get a better view of how well the compressed blocks maintain their computational capacity. Our results below have been implemented using the PyTorch machine learning framework [54]. To implement Neural ODEs, we additionally used the TorchDiffEq [5] and Torchdyn [55] python packages. Execution times are true wall times realized on Intel Xeon E5-2680 v3 2.5-GHz processors and measured as the time it takes to classify every item in the test dataset. For the convolutional network, we use a single core, and for the recurrent network, we use four processor cores.

In Table I, we show the number of parameters in a theoretical Neural ODE, and then, it has been compressed with

TABLE I
THEORETICAL LAYER SIZES

Model	ODE weights	ODE activations	Preceding weights	Following weights
Original model	n^2	n	ni	on
POD-DEIM, best case	$2k^2$	k	ki	ok
POD-DEIM, worst case	$2k^2$	k	$n(i+k)$	$n(o+k)$
APoZ, best case	k^2	k	ki	ok
APoZ, worst case	k^2	k	ki	$n(o+1)$
SVD	$2kn$	n	ni	on

n: ODE block dimension, k: compressed ODE dimension, i: neurons in the layer preceding the ODE block, o: neurons in the layer following the ODE block.

each compression method. We consider a nonlinear ODE block of the form $x'(t) = f(Ax(t))$ where the weight matrix has dimensions $A \in \mathbb{R}^{n\times n}$, and hence, there are n activation functions. The ODE block is preceded by a layer with weight matrix $W_1 \in \mathbb{R}^{n\times i}$ and followed by a layer with weight matrix $W_2 \in \mathbb{R}^{o\times n}$. In Table I, k denotes the dimension of the compressed ODE block and $n \times i$ and $o \times n$ are the number of weights in the layers preceding and following the ODE block, assuming that those layers exist. With regard to the model speed, the complexity of the ODE layer is significantly more important than the sizes of the surrounding layers. This is because the ODE function is evaluated multiple times, at least once for each intermediate time point in $[t_0, t_{end}]$. The architecture of the original ANN affects the achieved compression. The POD-DEIM and APoZ methods can reduce the ODE layer to a dimension that is independent of the original size n, although the interpolation in POD-DEIM requires an additional k^2 operations compared to APoZ. Moreover, for POD-DEIM, the best case is realized when the preceding and following operations are linear, whereas the APoZ method reaches the best case even if they are nonlinear. With the APoZ method, some nonlinear operations, such as max pooling, require that we maintain a lookup table of pooled indices or insert the k compressed values to their respective indices in an n size vector, which is the worst case result for the method. SVD truncation is not affected by the surrounding layers.

A. Convolutional Neural ODE

We illustrate the performance of reduced models on the MNIST dataset of handwritten digits [56]. For our first task, we train a convolutional Neural ODE that uses 16 convolutional kernels and tanh activation in the ODE block, with $Z = 0$ (see (2)). Before the ODE block, we use a convolutional layer with 3×3 kernels and rectified linear unit (ReLU) activations that outputs 16 channels into 3×3 max pooling. After the ODE block, we use 3×3 max pooling into a linear readout layer with ReLU activations. The ODE block propagates the data for $t = [0, 1]$ with timestep $dt = 0.1$ using a fourth-order fixed-step Runge–Kutta solver. The initial value of the ODE is the output from the prior layer. We use the adjoint method for training the ODE weights [5]. In order to implement MOR and model compression of the convolutional ODE block, the convolution operator is needed in the matrix form. After training, we replaced the original convolution operator with an equivalent operation implemented as a linear layer. The conversion yields a sparse Toeplitz matrix. With this operation, 16 convolution channels correspond to 1024 neurons and activation functions. All reported wall times are obtained with this conversion and compressed models are compared to these times. The reported wall times are medians of ten consecutive evaluations of the entire validation dataset. The trained network achieves 97.9% top-1 and 99.7% top-3 accuracies on held-out test dataset of 10 000 MNIST images and it takes 11.2s to run the model on the test dataset on CPU.

In order to determine the suitability of our chosen compression methods, we looked at several metrics of the trained Neural ODEs, namely, decay of singular values in the POD-DEIM snapshot matrices X and F and the weight matrix A. Moreover, we looked at the distribution of APoZ scores and their relation to DEIM interpolation indices. Fig. 3 shows these metrics for the convolutional Neural ODE. The left plot shows sorted singular values of the solution snapshot matrix X (POD) and F (DEIM) of the ODE block, gathered from training data, in purple and green, respectively. The logarithmic y-axis shows the magnitude of a given singular value divided by the sum of all singular values. Both POD and DEIM singular values collapse rapidly, indicating that a small number of singular vectors can span a space where the majority of dynamics are found. This is an important indicator for the suitability of the POD-DEIM method. In the same plot, the yellow plot shows the singular values of the weight matrix A of the ODE block. These values are useful for determining a rank for the SVD-based truncation method that we compare against POD-DEIM. Singular values of A also decay exponentially, meaning that a good low-rank approximation to A can likely be found. The right of Fig. 3 shows APoZ scores for each neuron in the ODE block, with low-scoring neurons being the least important. In addition, we have indicated the first 100 interpolation points p_i chosen by the DEIM method in green. All neurons seem to contribute to the model, with only a small number of relatively low-scoring neurons seen, which could make the APoZ method challenging to apply in the Neural ODE context. Some consensus between the methods is seen, as neurons with low APoZ scores are not in the top DEIM selections either.

We wanted to find out how the POD-DEIM method compares to existing model compression methods when compressing only the ODE block of a convolutional network. For POD-DEIM dimensions, we have chosen $k = m$ and use dimension k compressed models, which corresponds to the number of activation functions to dimension k APoZ trimmed models. The results on the MNIST dataset are shown in Fig. 4, where y-axis, model accuracy, is computed as reduced model top-1 accuracy divided by the original model result, and x-axis, achieved speedup, is computed as original model test time divided by reduced model test time. Each dot represents a compressed model dimension, which decreases in the direction of increased speedup. While the dimensions are not comparable between methods, the graphs indicate how much accuracy is maintained at a given acceleration amount. Table II presents the absolute performance values of compressed models without fine-tuning, with three epochs

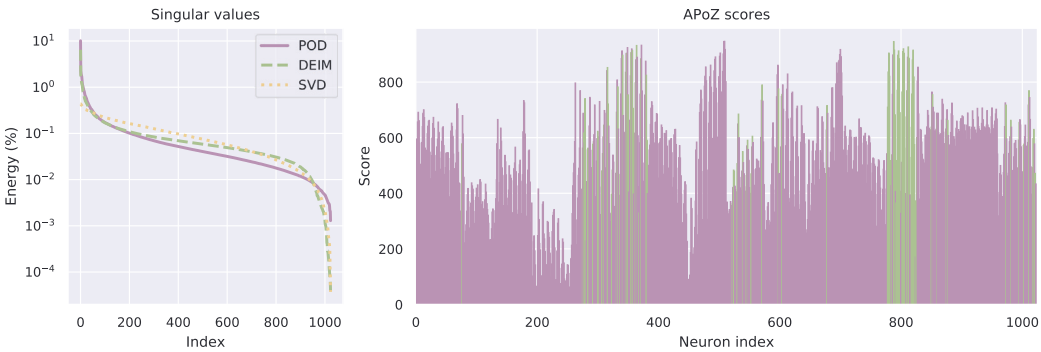

Fig. 3. Metrics for the convolutional Neural ODE. Left: singular values POD (purple) and DEIM (green) snapshot matrices and singular values of the weight matrix of the ODE block (yellow). Right: histogram of APoZ scores, with the first 100 DEIM indices indicated in green.

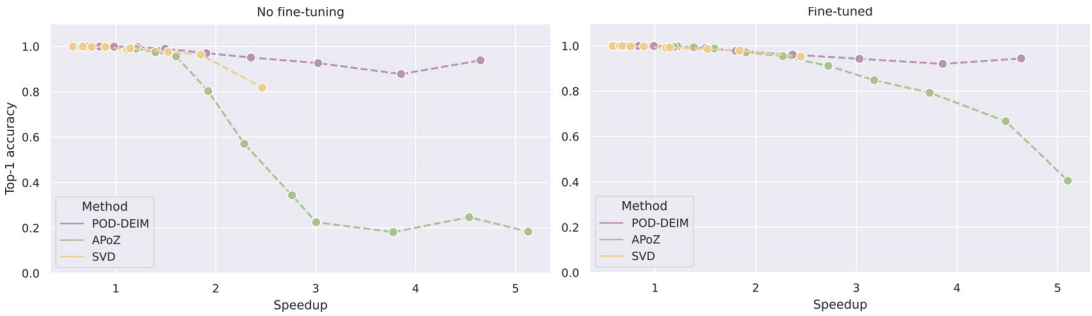

Fig. 4. Performance of reduced convolutional Neural ODEs on the MNIST dataset, relative to the original model. Left plot shows the results without fine-tuning, and right plot shows the results with three epochs of tuning. The x-axis shows achieved acceleration, while the y-axis shows reduced model top-1 accuracy divided by full model accuracy. Reduced model dimension decreases in the direction of improving speedup. Absolute performance can be seen in Table II.

of fine-tuning after model compression, and with retraining (30 epochs of training after compression).

Fig. 4 shows that the POD-DEIM method retains the highest accuracy when compared to similar speedups from other methods. Our POD-DEIM reaches over fourfold speedup in run time in this task. The SVD method is also accurate but cannot reach speedups greater than two times the original run time, explained by the method having to evaluate all the original nonlinear activation functions. The APoZ method reaches the fastest run times but loses the most amount of accuracy. After three epochs of fine-tuning, the APoZ method improves in accuracy, but not to the level of performance that the other methods show even without tuning. Overall, the POD-DEIM method achieves the best profile in terms of accuracy retained for speedup gained. A big contributor is the interpolation matrix N, which makes the method slower compared to APoZ, but the cost is justified by the better accuracy. When looking at absolute performance values, Table II shows that at dimension 350, the POD-DEIM method is two times faster to run than the original model while achieving 93.1% top-1 accuracy. At dimension 50, the POD-DEIM model is over four times faster than the original model, still reaching 91.1% top-1 accuracy without fine-tuning.

B. Recurrent Neural ODE

For our second study, we designed a task in which Neural ODEs have an advantage over discrete networks. ODE-RNNs [6] and antisymmetric RNNs [4], [50] are one such example. Using a continuous hidden state, ODE-RNNs enable flexible inference on continuous-time data even with irregularly sampled or missing values and antisymmetric models have spectral properties that are favorable for RNNs, as described in Section III. We implemented a shifted antisymmetric ODE-RNN for the pixel MNIST task. In this task, the network sees a pixel of a handwritten digit at a time and after iterating over each pixel outputs the class of the image. The length of the input sequence is 784 steps, corresponding to a flattened MNIST image, and the network sees each pixel for $dt = 0.1$s. Following [50], we initialize the ODE weights from zero mean, unit variance normal distribution, as well as train and test the network with the Euler discretization method.

TABLE II

ABSOLUTE PERFORMANCE OF REDUCED CONVOLUTIONAL NEURAL ODES

Dim.	Top-1 (%)			Top-3 (%)			Runtime (s)		
Original Model									
1024	97.9			99.7			11.2		
DEIM									
50	91.9	92.5	92.9	98.6	98.9	99.0	2.4	2.4	2.0
150	85.9	90.1	90.9	97.2	98.4	98.7	2.9	2.9	2.4
250	90.7	92.3	92.8	98.2	98.8	99.0	3.7	3.7	3.1
350	93.1	94.1	94.4	99.0	99.1	99.2	4.8	4.7	4.0
450	95.0	95.7	95.9	99.3	99.5	99.5	5.9	6.2	5.3
550	96.9	97.0	96.9	99.7	99.7	99.6	7.5	7.4	6.3
650	97.7	97.6	97.6	99.7	99.7	99.6	9.2	9.1	10.1
750	97.8	97.8	97.8	99.7	99.8	99.7	11.4	11.3	13.4
850	97.8	97.8	97.9	99.8	99.7	99.7	13.3	13.3	16.8
950	97.9	97.9	97.9	99.7	99.7	99.7	15.8	16.0	19.9
APoZ									
50	17.9	39.8	43.6	31.2	75.5	79.9	2.2	2.2	2.2
150	24.2	65.3	68.7	48.8	88.5	90.9	2.5	2.5	2.5
250	17.7	77.6	81.3	76.3	95.0	95.6	3.0	3.0	2.9
350	22.0	83.0	85.4	83.4	96.6	97.2	3.7	3.5	3.1
450	33.7	89.2	91.2	84.2	98.4	98.5	4.0	4.1	3.6
550	55.8	93.4	94.5	97.1	99.2	99.2	4.9	4.9	4.2
650	78.6	95.1	95.7	97.7	99.5	99.4	5.8	5.9	5.8
750	93.6	96.7	97.0	99.2	99.6	99.6	7.0	7.0	7.1
850	95.4	97.3	97.3	99.4	99.6	99.6	8.0	8.1	8.8
950	96.9	97.4	97.5	99.6	99.7	99.6	9.3	9.4	10.4
SVD									
50	80.1	93.2	93.7	97.9	99.2	99.3	4.5	4.6	4.6
150	94.3	95.9	96.1	99.2	99.5	99.4	6.1	6.1	5.3
250	95.5	96.5	96.5	99.3	99.7	99.6	7.3	7.3	6.5
350	96.6	97.0	97.1	99.6	99.7	99.6	10.0	10.1	9.0
450	97.0	97.2	97.3	99.7	99.7	99.6	9.8	9.7	8.7
550	97.7	97.6	97.7	99.7	99.8	99.7	12.5	12.6	11.2
650	97.7	97.7	97.7	99.7	99.8	99.6	14.7	14.7	13.3
750	97.8	97.8	97.9	99.7	99.7	99.7	17.2	17.1	17.4
850	97.9	97.8	97.9	99.7	99.8	99.7	16.7	16.5	16.9
950	97.9	97.8	97.8	99.7	99.8	99.7	19.6	19.3	20.1

Each column of Top-1, Top-3 and Runtime shows the results for the models directly after reduction, after three fine-tuning epochs and after retraining (30 epochs), in that order, separated by vertical lines.

TABLE III

ABSOLUTE PERFORMANCE OF REDUCED ODE-RNNS

Dim.	Top-1 (%)			Top-3 (%)			Runtime (s)		
Original Model									
512	96.2			99.6			48.1		
DEIM									
25	17.8	45.5	47.2	38.5	74.7	76.0	12.3	12.1	12.4
75	5.4	39.8	48.7	24.9	65.3	69.8	15.6	15.3	15.8
125	12.0	54.4	61.5	33.2	81.8	86.4	16.3	16.2	16.3
175	11.7	55.1	61.6	33.4	83.2	87.5	21.0	21.2	21.1
225	10.2	59.2	62.7	32.4	86.3	88.6	24.9	25.7	26.2
275	38.9	80.2	85.1	71.6	96.0	97.4	30.3	29.8	30.6
325	54.6	84.3	87.7	83.9	97.0	98.0	32.0	32.5	32.4
375	92.1	94.5	95.2	99.2	99.4	99.4	39.4	37.8	40.5
425	94.6	95.5	95.8	99.4	99.5	99.5	45.2	45.3	48.5
475	96.0	96.2	96.2	99.6	99.6	99.6	52.4	52.5	52.8
APoZ									
25	7.5	36.5	40.2	25.9	69.1	72.2	9.6	9.6	9.3
75	18.8	62.7	69.4	42.2	87.9	91.0	12.5	12.4	12.2
125	18.6	70.7	76.1	44.0	92.5	94.9	12.7	12.8	12.7
175	17.5	74.2	80.2	46.6	93.9	96.1	16.1	16.2	16.0
225	19.1	74.7	81.1	43.7	94.0	96.5	18.4	18.2	18.8
275	22.4	77.4	82.4	58.4	95.1	97.0	20.9	20.9	20.6
325	36.5	80.8	84.5	74.5	96.3	97.5	23.1	23.2	23.1
375	44.4	82.8	86.4	76.3	97.2	98.0	26.6	26.6	26.0
425	53.9	84.7	88.2	83.1	97.8	98.3	30.7	29.2	30.5
475	76.7	89.3	91.7	96.5	98.7	98.9	35.2	35.3	33.1
SVD									
25	28.5	80.4	87.2	54.2	95.9	97.8	20.6	20.8	20.7
75	74.4	93.4	94.4	94.8	99.0	99.3	23.9	24.3	24.2
125	94.5	95.6	95.9	99.5	99.6	99.5	26.6	25.7	25.6
175	95.6	96.0	96.2	99.6	99.6	99.6	33.8	32.9	32.7
225	96.0	96.1	96.2	99.6	99.6	99.7	35.3	36.4	36.9
275	96.1	96.2	96.2	99.6	99.7	99.7	38.6	39.1	39.2
325	96.1	96.3	96.3	99.6	99.7	99.6	40.3	42.5	42.3
375	96.2	96.3	96.3	99.6	99.7	99.6	46.3	46.2	46.2
425	96.2	96.3	96.3	99.6	99.7	99.6	48.1	48.5	47.8
475	96.2	96.3	96.3	99.6	99.7	99.6	50.5	50.3	50.4

Each column of Top-1, Top-3 and Runtime shows the results for the models directly after reduction, after three fine-tuning epochs and after retraining (30 epochs), in that order, separated by vertical lines.

We use 512 hidden units with a hyperbolic tangent nonlinearity and the shifted antisymmetric weight matrix as shown in (3). The initial value of the ODE is set to $x(0) = 0$. The ODE block is followed by a linear readout layer. Compared to the convolutional network, this model is smaller in neuron count but needs to incorporate the time-dependent input data. The trained network achieves 96.2% accuracy on held-out MNIST test data in 48.1s.

Fig. 5 shows the compression metrics for the ODE-RNN. The left plot shows the sorted relative singular value magnitude of the solution snapshot matrices X (POD) and F (DEIM) and the weight matrix A of the ODE block in purple, green, and yellow. For this model, the singular values of the snapshot matrices as well as the weight matrix also decay rapidly. However, with regard to POD-DEIM, the singular values of the snapshot matrices do not decay as fast as those obtained from the convolutional model and there is a considerable amount of total energy contained in the singular values until rank 500. This indicates that the dynamics of the ODE block are more difficult to approximate in low dimensions than those learned by the convolutional model. Such a result is expected because the ODE-RNN has a time-dependent input contributing to the hidden state and the forward propagations are computed for a longer time span than in our earlier example, allowing for richer dynamics. On the other hand, singular values of the shifted antisymmetric weight matrix decay more rapidly than in the earlier task, indicating suitability of SVD-based truncation of weights. The APoZ scores on the right of Fig. 5 show the same pattern as earlier, where it is not straightforward to detect the most important neurons conclusively. This is expected, given that in the context of ODE blocks, the output activity at the last timestep is not a conclusive measure of neuron importance.

We then evaluated the selected compression methods on the pixel MNIST task using our ODE-RNN. The results relative to the performance of the original model are shown in Fig. 6. Absolute results of compressed models without fine-tuning, with three epochs of fine-tuning, and with exhaustive tuning are shown in Table III. This task is more challenging for the reduction methods than the convolutional network. Possible factors are the dense structured antisymmetric weight matrix or the time-dependent input. Here, the SVD truncation method maintains the most accuracy. As predicted by the singular values of the snapshot matrices, the overall performance of the POD-DEIM method on this model is not as satisfying as with the convolutional model. On this model, the POD-DEIM method is faster than the original model until dimension 425, at which point the accuracy is 94.6%, while the APoZ method remains faster at all times. With regard to the tradeoff between run time and accuracy, the SVD method maintains

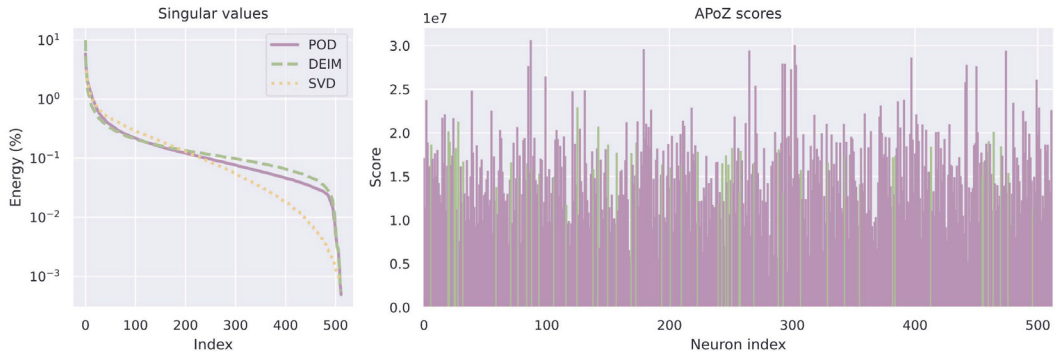

Fig. 5. Metrics of the ODE-RNN. Left: singular values of POD (purple) and DEIM (green) snapshot matrices and singular values of the weight matrix of the ODE block (yellow). Right: histogram of APoZ scores, with the first 50 DEIM indices indicated in green.

the most accuracy for speed gained. POD-DEIM shows good accuracy at high dimensions, although accuracy drops rapidly so that the overall performance is level with the APoZ method. With exhaustive fine-tuning, the APoZ method may be fit for compressing the ODE-RNN to a low dimension since APoZ yields the best absolute acceleration. The APoZ compressed model also seems easier to fine-tune than the POD-DEIM version since more accuracy is recovered in three fine-tuning epochs.

V. Discussion

In this work, we have studied the compression of ANNs that contain ODEs as layers (Neural ODEs). After training and compressing the networks, we evaluated their accuracies as well as execution speeds. Although in the literature many compression methods are applied in the training loop, the training of Neural ODEs is slower than discrete networks to begin with, making it attractive to compress the network in a separate step after training. We also assessed fine-tuning of the compressed network to see how much accuracy can be recovered. We focused on the run time and classification accuracy of compressed models since Neural ODEs are already very memory and parameter efficient by design.

Here, we showed how to formulate the POD-DEIM into compressing Neural ODEs. This method has been developed for accurately approximating dynamical systems in low dimensions and the continuous-time network formalism of Neural ODEs makes it possible to use POD-DEIM in deep learning applications. In addition, we compared the POD-DEIM and APoZ methods to SVD-based truncation. The performance was measured with two different architectures, a convolutional Neural ODE and a recurrent ODE architecture, which were trained on the MNIST digit classification task.

We hypothesized that some existing neuron pruning methods that measure neuron importance based on the final output of the layer only, such as the APoZ approach, are not optimal for compressing Neural ODEs. The reason is that these methods do not account for the dynamics of the ODE block when identifying pruning targets. Based on our results, the proposed approach for Neural ODE compression depends on the model architecture. We found the POD-DEIM method to achieve the most favorable continuum between accuracy and speedup on compressing the convolutional architecture, as shown in Fig. 4. The SVD-based weight truncation is the simplest of the studied methods and it performed well on both tasks, although it yielded less absolute acceleration than the other methods. The ODE-RNN was challenging to accelerate, and here, the weight truncation method is our favored choice based on Fig. 6. In this task, the APoZ method performed better or equally to POD-DEIM. While the POD-DEIM method is based on approximating the time trajectory in a low-dimensional subspace, there are also MOR methods, such as the balanced truncation, which are designed to approximate input–output behavior of high-dimensional systems. Such a method could be a good tool for compressing RNN models with time-dependent input, once the applicability of these methods to nonlinear systems develops further [57].

In this study, we only fine-tuned the layers following the ODE block, keeping results more indicative of raw performance of the methods. If the entire model were tuned, POD-DEIM and SVD truncation have more parameters available for fine-tuning than the APoZ method, allowing for greater accuracy. On the other hand, the small number of parameters is what makes APoZ compressed models the fastest out of the three methods, and hence, it should be favored if there are no restrictions on fine-tuning and retraining time. Moreover, compared to POD-DEIM, the APoZ method is more data and memory efficient since it does not need snapshots of trajectories of ODE dynamics nor does it compute large matrix decompositions. Overall, we found that POD-DEIM can achieve good acceleration and accuracy of compressed models, it has a solid theoretical foundation and several extensions while also being the most adjustable and optimizable approach of the methods used in this study.

We expect that our results with the POD-DEIM method motivate machine learning researchers to look into the possible applications of other MOR methods developed in control theory and related fields. For example, we assessed here only the

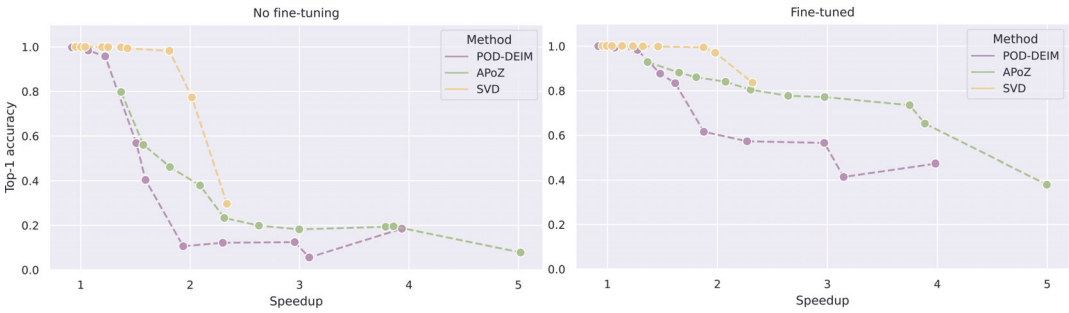

Fig. 6. Relative performance of reduced ODE-RNNs on the pixel MNIST task. Left plot shows results without fine-tuning, and right plot shows results with three epochs of tuning. The x-axis shows achieved acceleration, while the y-axis shows reduced model top-1 accuracy divided by full model accuracy. Reduced model dimension decreases in the direction of improving speedup. Absolute values can be seen in Table III.

original DEIM [19] and the ODEIM [52] methods, although in recent years, many advanced versions have been developed. These include novel interpolation point selection methods [58] and methods with adaptive basis and interpolation point selection capabilities [59]–[61] that allow the model to change its dimension or projection subspace online. It is worth noting that the POD-DEIM method aims at approximating the dynamics of the original system and is not aware of classification or other performance indicators of the Neural ODE. A potential future development could explore connecting the low-dimensional dynamics to downstream the network performance.

A limitation of the present work is that we could only study relatively small models, due to known challenges with training very large Neural ODEs [45]. In small models, the expected gains from compression methods are lesser than in large models. Moreover, we only discussed computation times on CPUs, as on GPUs the differences between original and compressed models were minor due to the small size of the models. While the computational speedup from POD-DEIM needs to be verified on GPUs and extremely large models, theoretical reductions in parameter, and activation function count, comparable results from existing literature as well as our results on the CPU indicate that the method should perform equally well.

Our results were obtained with fixed-step ODE solvers. We found that compressing Neural ODEs sometimes induces stiffness to the low-rank models (data not shown). This could cause adaptive ODE solvers to be slower, mitigating some of the speedup from compression. However, a systematic study of this behavior was not sought here and fixed-step ODE solvers were chosen to quantitatively compare the methods across dimensions. A future study is needed to quantify the effects of stiffness that may rise when compressing Neural ODEs.

VI. Conclusion

In this study, we addressed speeding up and compressing ANNs with continuous layers. Our POD-DEIM method, which has not previously been used to compress Neural ODEs, provides attractive results for accelerating convolutional Neural ODEs and equal results to neural pruning methods for accelerating recurrent Neural ODEs. POD-DEIM properly considers the trajectory of the ODE layer in an efficient manner. In addition, POD-DEIM has adjustable parameters and advanced variations that can be used to optimize the speed versus accuracy tradeoff for each model.

We expect our acceleration method to be useful when Neural ODEs are deployed on low-power hardware or battery-powered devices, for example, wearable devices, medical monitoring instruments, or smartphones. The dynamical system formalism for neural networks also allows other MOR methods to be applied in DL, which will provide for interesting future studies.

Acknowledgment

Neural network illustrations generated partly with http://alexlenail.me/NN-SVG/index.html.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, May 2015.
[2] E. Weinan, "A proposal on machine learning via dynamical systems," *Commun. Math. Statist.*, vol. 5, no. 1, pp. 1–11, 2017.
[3] Y. Lu, A. Zhong, Q. Li, and B. Dong, "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 3276–3285.
[4] E. Haber and L. Ruthotto, "Stable architectures for deep neural networks," *Inverse Problems*, vol. 34, no. 1, Jan. 2018, Art. no. 014004.
[5] R. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6571–6583.
[6] Y. Rubanova, R. T. Chen, and D. K. Duvenaud, "Latent ordinary differential equations for irregularly-sampled time series," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 5320–5330.
[7] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" in *Proc. 2nd Int. Conf. Mach. Learn. Syst.*, vol. 2, I. Dhillon, D. Papailiopoulos, and V. Sze, Eds. 2020, pp. 129–146. [Online]. Available: https://proceedings.mlsys.org/
[8] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1135–1143.
[9] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," 2016, *arXiv:1607.03250*.
[10] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1269–1277.
[11] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[12] H. Markram *et al.*, "Reconstruction and simulation of neocortical microcircuitry," *Cell*, vol. 163, no. 2, pp. 456–492, 2015.

[13] A. R. Kellems, S. Chaturantabut, D. C. Sorensen, and S. J. Cox, "Morphologically accurate reduced order modeling of spiking neurons," *J. Comput. Neurosci.*, vol. 28, no. 3, pp. 477–494, Jun. 2010.

[14] B. Du, D. Sorensen, and S. J. Cox, "Model reduction of strong-weak neurons," *Frontiers Comput. Neurosci.*, vol. 8, p. 164, Dec. 2014.

[15] M. Lehtimäki, L. Paunonen, S. Pohjolainen, and M.-L. Linne, "Order reduction for a signaling pathway model of neuronal synaptic plasticity," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7687–7692, Jul. 2017.

[16] M. Lehtimaki, L. Paunonen, and M.-L. Linne, "Projection-based order reduction of a nonlinear biophysical neuronal network model," in *Proc. IEEE 58th Conf. Decis. Control (CDC)*, Dec. 2019, pp. 1–6.

[17] M. Lehtimaki, I. Seppala, L. Paunonen, and M.-L. Linne, "Accelerated simulation of a neuronal population via mathematical model order reduction," in *Proc. 2nd IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Aug. 2020, pp. 118–122.

[18] G. Berkooz, P. Holmes, and J. L. Lumley, "The proper orthogonal decomposition in the analysis of turbulent flows," *Annu. Rev. Fluid Mech.*, vol. 25, no. 1, pp. 539–575, 1993.

[19] S. Chaturantabut and D. C. Sorensen, "Nonlinear model reduction via discrete empirical interpolation," *SIAM J. Sci. Comput.*, vol. 32, no. 5, pp. 2737–2764, 2010.

[20] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, "An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations," *Comp. Rendus Mathematique*, vol. 339, no. 9, pp. 667–672, 2004.

[21] S. Chaturantabut and D. C. Sorensen, "A state space error estimate for POD-DEIM nonlinear model reduction," *SIAM J. Numer. Anal.*, vol. 50, no. 1, pp. 46–63, 2012.

[22] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on CPUs," in *Proc. Deep Learn. Unsupervised Feature Learn. Workshop, NIPS*, 2011.

[23] M. Courbariaux, Y. Bengio, and J.-P. David, "BinaryConnect: Training deep neural networks with binary weights during propagations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3123–3131.

[24] W. Chen, J. Wilson, S. Tyree, K. Weinberger, and Y. Chen, "Compressing neural networks with the hashing trick," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2285–2294.

[25] J. Ren and L. Xu, "On vectorization of deep convolutional neural networks for vision tasks," in *Proc. AAAI Conf. Artif. Intell.*, 2015, vol. 29, no. 1, pp. 1840-1846.

[26] M. Mathieu, M. Henaff, and Y. LeCun, "Fast training of convolutional networks through FFTs," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, 2014.

[27] T. Chen *et al.*, "Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," *ACM SIGARCH Comput. Archit. News*, vol. 42, no. 1, pp. 269–284, Feb. 2014.

[28] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 598–605.

[29] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Proc. Adv. Neural Inf. Process. Syst.*, 1993, pp. 164–171.

[30] D. Yu, F. Seide, G. Li, and L. Deng, "Exploiting sparseness in deep neural networks for large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2012, pp. 4409–4412.

[31] J. Xue, J. Li, and Y. Gong, "Restructuring of deep neural network acoustic models with singular value decomposition," in *Proc. Interspeech*, Aug. 2013, pp. 2365–2369.

[32] S. Han *et al.*, "EIE: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Comput. Archit. News*, vol. 44, no. 3, pp. 243–254, 2016.

[33] A. Parashar *et al.*, "SCNN: An accelerator for compressed-sparse convolutional neural networks," *ACM SIGARCH Comput. Archit. News*, vol. 45, no. 2, pp. 27–40, Jun. 2017.

[34] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2754–2761.

[35] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *Proc. Brit. Mach. Vis. Conf.*, 2014.

[36] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," 2014, *arXiv:1412.6115*.

[37] X. Zhang, J. Zou, K. He, and J. Sun, "Accelerating very deep convolutional networks for classification and detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 1943–1955, Oct. 2015.

[38] H. Van Nguyen, K. Zhou, and R. Vemulapalli, "Cross-domain synthesis of medical images using efficient location-sensitive deep network," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, Munich, Germany. Switzerland: Springer, Oct. 2015, pp. 677–684.

[39] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and Huffman coding," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds. 2016.

[40] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1389–1397.

[41] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, 2017.

[42] V. Lebedev and V. Lempitsky, "Speeding-up convolutional neural networks: A survey," *Bull. Polish Acad. Sci. Tech. Sci.*, vol. 66, no. 6, pp. 799–810, 2018.

[43] E. Dupont, A. Doucet, and Y. W. Teh, "Augmented neural ODEs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019.

[44] J. Kelly, J. Bettencourt, M. J. Johnson, and D. K. Duvenaud, "Learning differential equations that are easy to solve," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associates, 2020, pp. 4370–4380.

[45] C. Finlay, J.-H. Jacobsen, L. Nurbekyan, and A. Oberman, "How to train your neural ODE: The world of Jacobian and kinetic regularization," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 3154–3164.

[46] P. Kidger, R. T. Q. Chen, and T. Lyons, "Hey, that's not an ODE': Faster ODE adjoints via seminorms," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 5443–5452.

[47] M. Poli, S. Massaroli, A. Yamashita, H. Asama, and J. Park, "Hypersolvers: Toward fast continuous-depth models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 21105–21117.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[49] S. Massaroli, M. Poli, M. Bin, J. Park, A. Yamashita, and H. Asama, "Stable neural flows," 2020, *arXiv:2003.08063*.

[50] B. Chang, M. Chen, E. Haber, and E. H. Chi, "AntisymmetricRNN: A dynamical system view on recurrent neural networks," in *Proc. 9th Int. Conf. Learn. Represent. (ICLR)*, New Orleans, LA, USA, 2019.

[51] L. Sirovich, "Turbulence and the dynamics of coherent structures. I-III," *Quart. Appl. Math.*, vol. 45, no. 3, pp. 561–590, 1987.

[52] B. Peherstorfer, Z. Drmać, and S. Gugercin, "Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points," *SIAM J. Sci. Comput.*, vol. 42, no. 5, pp. A2837–A2864, Jan. 2020.

[53] F. Liang, R. Shi, and Q. Mo, "A split-and-merge approach for singular value decomposition of large-scale matrices," *Statist. Interface*, vol. 9, no. 4, p. 453, 2016.

[54] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035.

[55] M. Poli, S. Massaroli, A. Yamashita, H. Asama, and J. Park, "TorchDyn: A neural differential equations library," 2020, *arXiv:2009.09346*.

[56] Y. LeCun. (1998). *The MNIST Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[57] P. Benner and P. Goyal, "Balanced truncation model order reduction for quadratic-bilinear control systems," 2017, *arXiv:1705.00160*.

[58] Z. Drmać and S. Gugercin, "A new selection operator for the discrete empirical interpolation method–improved *a priori* error bound and extensions," *SIAM J. Sci. Comput.*, vol. 38, no. 2, pp. A631–A648, Jan. 2016.

[59] B. Peherstorfer, D. Butnaru, K. Willcox, and H.-J. Bungartz, "Localized discrete empirical interpolation method," *SIAM J. Sci. Comput.*, vol. 36, no. 1, pp. A168–A192, Jan. 2014.

[60] B. Peherstorfer and K. Willcox, "Online adaptive model reduction for nonlinear systems via low-rank updates," *SIAM J. Sci. Comput.*, vol. 37, no. 4, pp. A2123–A2150, Jan. 2015.

[61] S. Chaturantabut, "Temporal localized nonlinear model reduction with *a priori* error estimate," *Appl. Numer. Math.*, vol. 119, pp. 225–238, Sep. 2017.

Mikko Lehtimäki received the M.Sc. degree in computational biology from Tampere University of Technology, Tampere, Finland, in 2016. He is currently pursuing the Ph.D. (Tech) degree with Tampere University, Tampere, with a focus on computational neuroscience.

In addition to biology, his studies focused on signal processing and computer science. His research is centered on accelerating the simulation of nonlinear neuron and neural network models, and his topics of interest range from cell biology to artificial intelligence. He has extensive engineering and design experience from the software and robotics industries.

Lassi Paunonen (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in mathematics from Tampere University of Technology, Tampere, Finland, in 2007 and 2011, respectively.

From 2016 to 2019, he held the position of an Academy of Finland Post-Doctoral Researcher. He is currently an Associate Professor and the leader of the Systems Theory Research Group at Tampere University, Tampere. His main research interests include mathematical control theory, analysis of partial differential equations, and the theory of strongly continuous operator semigroups.

Marja-Leena Linne (Member, IEEE) received the M.Sc. degree in electrical engineering and the Ph.D. degree in computer science and theoretical neuroscience from Tampere University of Technology, Tampere, Finland, in 1993 and 2001, respectively.

She is currently a Principal Investigator at Tampere University, Tampere, where she leads the Computational Neuroscience Group at the Faculty of Medicine and Health Technology. She develops computational and mathematical methods for brain modeling and analysis of signals and images obtained *in vitro* and *in vivo*. She is a core member of the EU FET Flagship Human Brain Project. She has authored over 80 refereed interdisciplinary publications, as the first and last author both in engineering and biological topics. The focus of her research in neuroscience is on understanding the roles of neuronal and glial cells in neuronal circuits as well as in brain signals recorded from humans in health and disease.