UNIVERSITY of CALIFORNIA
Santa Barbara

Computational Tools for Large-Scale Linear Systems

A dissertation submitted in partial satisfaction of the
requirements for the degree of

Doctor of Philosophy

in

Mechanical Engineering

by

Michael D. Nip

Committee in charge:

João P. Hespanha, Chair
Professor Mustafa Khammash
Professor Bassam Bamieh
Professor Frederic Gibou

December 2014

The dissertation of Michael D. Nip is approved:

Mustafa Khammash

Bassam Bamieh

Frederic Gibou

João P. Hespanha, Committee Chair

December 2014

Computational Tools for Large-Scale Linear Systems

by

Michael D. Nip

To my parents, for your continuing support

Acknowledgements

Above all, I would like to thank my advisors Mustafa Khammash and João Hespanha who have provided countless opportunities and interesting problems throughout my time at UCSB and elsewhere. They have been more patient and encouraging with me than was really reasonable.

A substantial portion of this work would not have been possible without the collaboration of Vladimir Kazeev and Christoph Schwab of ETH-Züich. Working closely with them to apply the TT technology to the CME was both fruitful and fun.

I would like to thank the members of the Khammash group, in particular, Patrick Sheppard, Gabrielle Lillacci, Gentian Buzi, Corentin Briat, Ankit Gupta, and Bence Melykuti. I also wish to thank the many DCR lab members both past and present, in particular, Ryan Mohr, Michael Busch, Lina Kim, and Mitchell Craun for many interesting discussions and for helping me keep a healthy perspective over the years.

I would like to acknowledge all my friends both in Santa Barbara and elsewhere. Thank you, David and Nicky Painter, Marissa Beuhler, and Marissa Foster for listening and caring.

Last, but not least, I would like thank my entire family, but especially my parents Cara and Randall, and my sister Kari. Without their love and support I would not have made it through even the first day.

Curriculum Vitae

Education

2008 – PhD Candidate in Mechanical Engineering, University of California, Santa Barbara, USA

2003 – 2007 B.S., Engineering-Physics, Minor: Mathematics, University of California, Berkeley, USA

Relevant Experience

Research

Graduate Student Researcher *March 2009 – Present*

Advisors: Prof. Mustafa H. Khammash & Prof. João P. Hespanha

Dept. of Mech. Engineering, UCSB, Santa Barbara, CA 93106, USA

- **Advanced Computational Tools for Systems Biology**

  We investigate and develop strategies for large scale model-order reduction of stochastic models of gene regulatory networks. Our work strongly emphasizes preserving a faithful description of the statistical properties of the time-evolution of the original system. The goal of this work is to enable rapid prototyping of synthetic gene networks by significantly reducing the computational work necessary to perform stochastic simulations.

- **Low-Parametric Tensor Numerical Methods for Control System Design**

  We investigate the use of low-parametric tensor numerical linear algebra techniques in designing computationally efficient algorithms for control design for large-scale linear systems.

Mitarbeiter/Visiting Student *April 2012 – Oct. 2012*

Advisor: Prof. Mustafa H. Khammash

Dept. of Biosystems Science and Engineering, ETH-Zürich, 4058 Basel, Switzerland

- ***hp*-DG-QTT Solver for the Chemical Master Equation**

  We developed a direct numerical solver for the Chemical Master Equation that exploits *hp*-Discontinuous Galerkin semidiscretization in time and Quantized Tensor Train numerical linear algebra in space to lift the "Curse of Dimensionality". The resulting numerical method is highly robust and computationally efficient.

Undergraduate Research Assistant Jan. 2005 – May 2007

Advisors: Dr. Elke Arenholz & Dr. Soren Prestemon

Advanced Light Source, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

- **Technical and Computational Support for a Synchrotron Beamline**

  We provided mechanical engineering support for a synchrotron beamline used

to investigate the material properties of magnetic materials. Our primary focus was developing new instrumentation to enable beamline users to perform novel experiments. This work included elements of superconducting magnetic design using commercial finite element software, cooling and power system design, and nonlinear control design.

Teaching Experience

Teaching Assistant *Sept 2008 – Present*

Dept. of Mech. Engineering, UCSB, Santa Barbara, CA 93106, USA

- Directed laboratory sections, held office hours, graded assignments, and provided technical support for undergraduate senior projects for the following courses:
  - ME 125CH – LabView and Mechatronics
  - ME 170C – Introduction to Robotics
  - ENGR 3 – Introduction to Programming for Engineers
  - ME 104 – Mechatronics

Summer Research Mentor *Summer 2014*

Summer Sessions, UCSB, Santa Barbara, CA 93106, USA

- Oversaw and mentored high school student participating in a summer research

experience and outreach program. Each student wrote a research report, participated in a research symposium, and presented a poster on their project.

  - Jonathan Huang - DMRG-Based Fast Numerical Tensor Arithmetic in the TT Format
  - Michelle Lee - Sensitivity Analysis of a Stochastic Chemical Model Using Polynomial Chaos Expansions

Summer Research Mentor *Summer 2009*

Summer Sessions, UCSB, Santa Barbara, CA 93106, USA

- Oversaw and mentored a local high school student participating in a summer research experience and outreach program. The student wrote a research report in the style of a journal article and gave a presentation on his work at the end of the program.
  - Paul Park - Model Reduction of a Stochastic Epigenetic Switch Using Empirical Gramians and Balanced Truncation

Private Tutor *Sept. 2007 – May 2008*

Expanding Abilities Tutoring, Redondo Beach, CA 90277, USA

- Provided in-home tutoring in mathematics and science at the middle, high-school, and undergraduate levels.

Private Tutor *Sept. 2007 – May 2008*

PJ Test Prep, El Segundo, CA 90245, USA

- Provided in-home tutoring in mathematics and science at the middle, high-school, and undergraduate levels.

PROFESSIONAL SOCIETIES

Institute of Electrical and Electronics Engineers (IEEE), Student

Society of Industrial and Applied Mathematics (SIAM), Student

Tau Beta Pi National Engineering Honor Society, Member

Phi Beta Kappa National Honor Society, Member

Abstract

Computational Tools for Large-Scale Linear Systems

by

Michael D. Nip

While the theoretical analysis of linear dynamical systems with finite state-spaces is a mature topic, in situations where the underlying model has a large number of dimensions, modelers must turn to computational tools to better visualize and analyze the dynamic behavior of interest. In these situations, we are confronted with the Curse of Dimensionality: computational and storage complexity grows exponentially in the number of dimensions.

This doctoral project focuses on two main classes of large-scale linear systems which arise in system biology. The Chemical Master Equation (CME) is a Fokker-Planck equation which describes the evolution of the probability mass function of a countable state space Markov process. Each state of the CME is labelled with an ordered S-tuple corresponding to one configuration of a well-mixed chemical system, where S is the number of distinct chemical species of interest. Even in cases where one only considers a projection of the CME to a finite subset of the states, one still must contend with the Curse of Dimensionality: the computational complexity grows exponentially in the number of chemical species. This dissertation describes

a computational methodology for efficient solution of the CME which, in the best cases, will scale *linearly* in the number of chemical species.

The second main class of high-dimensional problems requiring computational tools are coupled linear reaction-diffusion equations. For this class of models, we focus primarily on the computation of certain high-dimensional matrices which describe in a quantitative sense the input-to-state and state-to-output relationships. We describe algorithms for extracting useful information stored in these matrices and use this information to efficiently compute both reduced order models and open-loop control laws for steering the full system. A key feature of this approach is that the method is completely simulation or experiment free, in fact, in our numerical experiments, the computation of a reduced model or open-loop control law is *an order of magnitude* faster on a laptop than simulation of the full system on a 32 core node of a high-performance cluster.

In both projects, the enabling computational technology is the recently proposed Tensor Train (TT) structured low-parametric representation of high-dimensional data. The TT-format effectively exploits low-rank structure of the "unfolding matrices" for compression and computational efficiency. Formally, the computational complexity of basic TT arithmetics scale linearly in the number of dimensions, potentially circumventing the curse of dimensionality. To demonstrate the effectiveness of this approach, we performed numerous numerical experiments whose results are reported here.

Contents

List of Figures

List of Tables

Chapter I

Introduction

I.1 Large-Scale Linear Time Invariant Systems in Systems Biology

The work in this dissertation is concerned with the efficient simulation and control of large-scale Linear Time Invariant (LTI) systems that arise in systems biology. The standard state-space model with state $x \in \mathbb{R}^n$, input $u \in \mathbb{R}^m$, and outputs $y \in \mathbb{R}^k$ is given by

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du, \end{aligned} \tag{I.1.1}$$

where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{k \times n}$, and $D \in \mathbb{R}^{k \times m}$.

The first class of systems we are concerned with are discrete stochastic mod-

els which are used to describe the biological phenomena within living cells. Each state of the stochastic model is labeled with an ordered S-tuple corresponding to one configuration of a well-mixed chemical system, where S is the number of distinct chemical species of interest. The Chemical Master Equation describes the evolution of the probability mass function conditioned on some initial probability distribution and can be formatted as a linear ODE with countable state-space $\dot{p} = Ap$. In this example we are concerned with the efficient numerical solution of the Initial Value Problem and leave questions concerning model reduction and control to future work. This problem is difficult to treat using standard reduced-basis methods since the computational complexity grows exponentially in the number of chemical species.

The second class of problems examined are linearized reaction-diffusion equations which describe the evolution of a spatial profile of biochemicals of interest. Here, the computational complexity of analyzing and solving these problems computationally originates in both the dimension of the spatial domain and the choice of discretization as well as the complexity of the chemical networks under consideration. In this case, we are interested in questions of efficient control and model reduction of the large-scale system. For example, is there an efficient method to compute a control policy which steers the system to a certain chemical profile by introducing small quantities of certain key reactants? How should one efficiently compute a reduced order model which sufficiently captures the input-output behavior of the original system?

We emphasize here that we will treat numerical solution of the CME *itself* of

great practical interest while linearized reaction-diffusion equations are an interesting source of *examples* as a test bed for our control algorithms.

I.1.1 The Chemical Master Equation

In spite of the success of continuous-variable deterministic models in describing many biological phenomena, discrete stochastic models are often necessary to describe biological phenomena inside living cells where random motion of reacting species introduces randomness in both the order and timing of biochemical reactions. Such random effects become more pronounced when one factors in the discrete nature of reactants and the fact that they are often found in low copy numbers inside the cell. Manifestations of randomness vary from copy-number fluctuations among genetically identical cells [Elo+02] to dramatically different cell fate decisions [MA97] leading to phenotypic differentiation within a clonal population. Characterizing and quantifying the effect of stochasticity and its role in the function of cells is a central problem in molecular systems biology.

In order to effectively capture this experimentally observed stochasticity, the evolution of the chemical species of interest are commonly modeled using jump Markov processes. Here, each state of the process corresponds to the copy number of one of the constituent species [Gil76]. Within this framework, the evolution of the probability density over the possible configurations of the reaction network is described by a Forward Kolmogorov Equation, frequently referred to as the Chemical

Master Equation (CME) within the chemical literature. While analytical solutions can be obtained under specific assumptions about the structure of the chemical network [JH07], these assumptions prove so restrictive as to exclude the vast majority of biologically relevant systems. In most cases, the CME cannot be solved explicitly and various numerical simulation techniques have been proposed to approximately solve the time-evolution problem.

A "well-stirred" solution of d chemically reacting molecules in thermal equilibrium can be described by a jump Markov process, where for each fixed time $t \geq 0$, $X(t) \in \mathbb{Z}^d_{\geq 0}$ is a random vector of nonnegative integers with each component representing the number of molecules of one chemical species present in the system. In [Kam92] and the references therein, it is shown that, given an initial condition $X(0) \in \mathbb{Z}^d_{\geq 0}$, the corresponding probability density function (PDF) $\mathbb{Z}^d_{\geq 0} \times [0, \infty) \ni (\underline{x}, t) \mapsto \boldsymbol{p}_{\underline{x}}(t)$ of the process solves the Chemical Master Equation (CME):

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\boldsymbol{p}_{\underline{x}}(t) = -\boldsymbol{p}_{\underline{x}}(t)\sum_{s=1}^{R}\omega^s(\underline{x}) + \sum_{s=1}^{R}\boldsymbol{p}_{\underline{x}-\underline{\eta}^s}(t)\,\omega^s\big(\underline{x}-\underline{\eta}^s\big) \tag{I.1.2}$$

where R is the number of reactions in the system, $\underline{\eta}^s \in \mathbb{Z}^d$ and ω^s are the stoichiometric vector and propensity function of the sth reaction, respectively. The CME is a system of coupled linear ordinary differential equations with one equation per state $X(t) = \underline{x} \in \mathbb{Z}^d_{\geq 0}$.

The CME describes the dynamics of probabilities of finding the chemical system in different states. In general the number of these different states is countably infinite,

as it is not unknown *a priori* the maximum number of copies that each species can take. While this gives rise to an infinite number of state variables, each indicating the probability of a given chemical state, the vast majority of these probabilities are vanishingly small. This has motivated approaches for truncating the infinite number of state variables in the CME in a way that results in a finite number of state variables corresponding to chemical states that are likely to have high probability mass. The truncated CME consists of a system of linear ODEs with finite state space, that can *in principle* be solved. One such truncation approach which we will follow here is the Finite State Projection method. This truncation approach has the advantage of yielding bounds on the error between the solution of the truncated finite system and the original infinite set of ODEs (the CME).

In practice, the truncation satisfying a given error tolerance may still require a very large number of states rendering a direct numerical solution of even the projected equation infeasible in many cases. Chapter IV describes a numerical approach to solving the CME which scales favorably with the number of chemical species, expanding the class of efficiently solvable CME problems.

I.1.2 Linearized Reaction-Diffusion Equations

For a reaction-diffusion system with S chemical species, a PDE describing the time evolution on the cube $D = (-\pi, \pi)^d$ with control u and output y, subject to

Dirichlet boundary conditions is given by:

$$\begin{cases} \partial_t \boldsymbol{q}(x,t) = \boldsymbol{D}\Delta\boldsymbol{q}(x,t) + \boldsymbol{R}(\boldsymbol{q},x,t) + \boldsymbol{F}(u(t),x,t), & x \in D. \\ \boldsymbol{q}(x,t) = 0, & x \in \partial D, \\ y(t) = \boldsymbol{H}(\boldsymbol{q}(x,t)). \end{cases} \tag{I.1.3}$$

where $\boldsymbol{q}(x,t)$ is a vector-valued function whose s-th entry $q_s(x,t)$ corresponds to the concentration of chemical species Q_s, $\boldsymbol{D}$ is a diagonal matrix of diffusion coefficients, $\boldsymbol{R}(\boldsymbol{q},x,t)$ accounts for the local reactions, $\boldsymbol{F}(u(t),x,t)$ describes the input to the system, and $\boldsymbol{H}(\boldsymbol{q}(x,t))$ describes the output.

If the system has R reaction channels and if the reaction rate $\boldsymbol{R}_\rho(\boldsymbol{q},x,t)$ for each channel ρ has the form

$$\boldsymbol{R}_\rho(\boldsymbol{q},x,t) = f^\rho(x)\boldsymbol{\mathcal{S}}_\rho\boldsymbol{\omega}_\rho\boldsymbol{q}(x,t), \tag{I.1.4}$$

then the term $\boldsymbol{R}(\boldsymbol{q},x,t)$ describing the reactions can be written as the sum

$$\boldsymbol{R}(\boldsymbol{q},x,t) = \sum_{\rho=1}^{R} \left(f^\rho(x)(\boldsymbol{\mathcal{S}}_\rho\boldsymbol{\omega}_\rho\boldsymbol{q})\right). \tag{I.1.5}$$

In equation (I.1.4), $\boldsymbol{\mathcal{S}}_\rho$ is the stoichiometric vector associated with reaction ρ that describes the change in molecule counts when the reaction fires, $\boldsymbol{\omega}_\rho\boldsymbol{q}(x,t)$ describes the rate of reaction ρ as a *linear* functional of the concentration of reactants, and $\boldsymbol{f}^\rho(\boldsymbol{x})$ describes the spatial dependence of the reaction.

Further assuming that $\boldsymbol{F}(u(t),x,t)$ is a linear function of $u(t)$ and time-invariant and that $\boldsymbol{H}(\boldsymbol{q}(x,t))$ is a linear function of $\boldsymbol{q}(x,t)$:

$$\boldsymbol{F}(u(t),x,t) \;=\; \boldsymbol{F}(x)u(t),$$

$$\boldsymbol{H}(\boldsymbol{q}(x,t)) \;=\; \boldsymbol{H}\boldsymbol{q}(x,t),$$

(V.1.3) simplifies to

$$\begin{cases} \partial_t \boldsymbol{q}(x,t) = \left(\boldsymbol{D}\Delta + \sum_{\rho=1}^{R} f^{\rho}(x)(\boldsymbol{\mathcal{S}}_{\rho}\boldsymbol{\omega}_{\rho})\right)\boldsymbol{q}(x,t) + \boldsymbol{F}(x)u(t), & x \in D. \\ \boldsymbol{q}(x,t) = 0, & x \in \partial D, \\ y(t) = \boldsymbol{H}\boldsymbol{q}(x,t). \end{cases} \tag{I.1.6}$$

Note that (V.4.1) can also be used to represent the small fluctuations about a solution profile of the full nonlinear system. Suppose $\boldsymbol{q}_0(x,t)$ is a solution of (V.1.3) under the constant control input $\boldsymbol{u}_0$. Consider a small perturbation in the control input

$$\boldsymbol{u}(t) = \boldsymbol{u}_0 + \varepsilon \boldsymbol{u}_1(t),$$

and the corresponding perturbation in the concentration profile

$$\boldsymbol{q}(x,t) = \boldsymbol{q}_0(x,t) + \varepsilon \boldsymbol{q}_1(x,t),$$

where $q_1(x,t) = 0$ on $x \in \partial D$ and ε is a small parameter. Assuming $\boldsymbol{R}(\boldsymbol{q},x,t)$ and $\boldsymbol{F}(u(t),x,t)$ can be expanded in Taylor series about $\boldsymbol{q}(x,t)$ and $\boldsymbol{u}_0$, we can expand (V.4.1) in powers of ε

$$\varepsilon\partial_t \boldsymbol{q}_1(x,t) = \varepsilon \boldsymbol{D}\Delta \boldsymbol{q}_1(x,t) + \varepsilon \nabla_q \boldsymbol{R}(\boldsymbol{q}_0,x,t)\boldsymbol{q}_1(x,t) + \varepsilon \nabla_u \boldsymbol{F}(\boldsymbol{u}_0,x,t)\boldsymbol{u}_1(t) + \mathcal{O}(\varepsilon^2).$$

which reduces to (V.4.1) after truncation of the terms that are $\mathcal{O}(\varepsilon^2)$ and higher.

For the numerical experiments, we consider a version of (V.4.1) that has been discretized in space using a finite difference scheme on a uniform tensor grid with

spacing h but no discretization in time (Method of Lines). Let $\hat{\boldsymbol{q}}(x,t)$ denote the discrete approximation of $\boldsymbol{q}(x,t)$. The time evolution of the discretized system is given by the finite-dimensional LTI system:

$$\begin{aligned} \partial_t \hat{\boldsymbol{q}}(x,t) &= \boldsymbol{A}\hat{\boldsymbol{q}}(t) + \boldsymbol{B}u(t), \\ y(t) &= \boldsymbol{C}\hat{\boldsymbol{q}}(t). \end{aligned} \tag{I.1.7}$$

where

$$\boldsymbol{A} = \frac{1}{h^2}(\boldsymbol{\Delta}_{dd} \otimes \boldsymbol{D}) + \sum_{\rho=1}^{R} \left(\mathtt{diag}(\hat{\boldsymbol{f}}^{\rho}) \otimes (\boldsymbol{\mathcal{S}}_{\rho}\hat{\boldsymbol{\omega}}_{\rho}) \right), \tag{I.1.8}$$

where $\boldsymbol{\Delta}_{dd}$ is the discrete Laplacian on a rectangular grid with Dirichlet boundary conditions, $\boldsymbol{D}$ is the diffusion tensor, $\hat{\boldsymbol{f}}^{\rho}$ is the discretization of $f^{\rho}(\boldsymbol{x})$ on the spatial grid, and $\boldsymbol{B}$ and $\boldsymbol{C}$ depend on the discretizations of $\boldsymbol{F}(x)$ and $\boldsymbol{H}$, respectively.

I.2 Structured Low-Parametric Representations of Multidimensional Arrays

A key feature of the two applications discussed previously is that while the problems maybe very difficult to treat computationally when using a naive discretization scheme, the operators involved and the solutions may have nice structure (e.g. separability) which may be used to reduce the complexity. A computational approach should use a structured formatting of the data so that both the storage required to represent the system and solutions is small and the number of floating point operations needed to perform basic arithmetic operations is reduced.

A formatting of the data that has proven quite advantageous, at least experimentally, in this dissertation work is the *Tensor Train* (TT) format developed by Oseledets and Tyrtyshnikov [Ose11; OT09]. We remark that the TT format has been known in theoretical chemistry as Matrix Product States or Linear Tensor Networks for at least two decades [Whi93]. Given a multidimensional array $\boldsymbol{X}(i_1, \ldots, i_d)$ with d separate indices, a TT decomposition of $\boldsymbol{X}$ could be

$$\boldsymbol{X}(i_1, \ldots, i_d) = X_1(i_1)X_2(i_2)\ldots X_d(i_d). \tag{I.2.1}$$

where each $X_k(i_k)$ is an indexed family of matrices of size $r_{k-1} \times r_k$. The collection of r_k's are called the *TT-ranks* of the decomposition and measure the structure in the data. Let n be the largest number of values that each index i_k may take and r an upper bound on the TT ranks. While a full-format representation of X would require storage of $\mathcal{O}(n^d)$ separate parameters, the storage of the TT decomposition scales as $\mathcal{O}(dnr^2)$. If the ranks are small, the storage savings may be quite significant. The structure of the TT format also allows fast basic arithmetic. Typically, whereas a full-format operation would require $\mathcal{O}(n^d)$ operations, the same operation when using TT formatted arrays will be $\mathcal{O}(dn \cdot \texttt{poly}(r))$. In some sense, the TT format trades the Curse of Dimensionality for the Curse of the Ranks.

While there are many other alternative structured low-parametric representations of multidimensional arrays, e.g. [Gra10a; Hit26; HK09], the TT format has a number of advantages. First, given an array in full format, the TT-ranks are well-defined and are easily computed. Second, there is a stable algorithm for computing a

TT decomposition from a full-format array. Third, the TT format admits a robust, stable, and fast tensor rounding procedure meaning that given a TT formatted array, there is a numerical procedure for finding an approximation of the array in a TT decomposition with smaller ranks, *without* reference to the full-format representation.

The combination of good compression, fast arithmetics, and numerical stability make the TT format a good choice for the representation of the data, in general. An important part of the dissertation will be characterizing the TT ranks of the objects involved since this will give a good indication of whether the TT approach will work well for the particular problem under consideration.

I.3 Structure of the Dissertation

This dissertation is structured as follows. Chapter II reviews important background information on modeling high-dimensional systems with tensor data-structures and, in particular, the Tensor Train representation. Chapter III discusses some extensions to the Tensor Train format which are based on alternate orderings of the spatial, quantization, or operator levels and derives algorithms for basic arithmetic operations in these formats including products with alternate formats. It is adapted from material published in[Kaz+14; NHK13]. Chapters IV and V describe applications of the preceding two sections to construct algorithms for large-scale LTI systems. Chapter IV describes the *hp*-DG-QTT numerical solver for the Chemical Master Equation which combines the QTT representation in the "species" space with the *hp*-

Discontinous Galerkin semidiscretization in time to efficiently simulate the evolution of the probability distribution and demonstrates its effectiveness on some examples from systems biology. The results of that chapter were first published in [Kaz+14], which was work done in collaboration with Mr. Vladimir Kazeev and Dr. Christoph Schwab of ETH-Zürich. Chapter V describes an algorithm which uses the TT format for solving Continous Time Algebraic Lyapunov Equations (CALEs), in particular, solving for the infinite-time horizon gramians of an LTI control system, and then using the TT structure of the solution to compute reduced models by projecting the dynamics onto the dominant subspaces of the gramians. This chapter also gives detailed description of numerical experiments which demonstrate the effectiveness of the approaches. The CALE solver and a portion of the numerical experiments were originally published in [NHK13]. Chapter VII summarizes some additional projects related to modeling and identification of stochastic gene regulatory networks. The section on reduced models of the CME contains work originally published in [NHK12].

Chapter II

High-Dimensional Modeling and the Tensor Train Format

This chapter reviews the basic numerical linear algebra on which the material of the three subsequent chapters are built, namely, basic finite-dimensional multilinear algebra and the Tensor Train structured representation of multidimensional arrays. This chapter is only meant to give the reader the bare essentials, i.e. we barely scratch the surface of algebraic tensor spaces and make no mention of topological tensor spaces. We refer the reader to [Hac12] and the references therein for a more thorough treatment of these subjects.

II.1 Tensors and Multidimensional Arrays

Here and in the subsequent chapters, the terms tensor and multi-dimensional array will be used synonymously. Suppose $\boldsymbol{x}(i_1, \ldots, i_d)$ is a multi-dimensional array with d separate indices i_k which each take n_k separate values for each k. We refer to $\boldsymbol{X}$ as a d-dimensional or d-level vector. Each parameter i_k taking values in a corresponding index set $\mathcal{I}_k$ indexes a *dimension* (alternatively referred to as a *level*, or *mode*) of the d-level vector. The ordered list of n_k's are referred to as the *mode sizes* of the array.

A few basic operations on tensors will be useful to us. The *contraction* of the modes $i_{k_1}, \ldots, i_{k_s}$ is a summation along the elements of the array for which those indices are equal. For example, the contraction of modes i_1 and i_2 of a 3-dimensional vector $\boldsymbol{v}$ is given by:

$$\sum_{i_1=i_2} v(i_1, i_2, i_3). \tag{II.1.1}$$

For the contraction to be well defined the summation must be over compatible indices, i.e. their index sets must be the same. The *tensor product* of two arrays $\boldsymbol{x}(i_1, \ldots, i_{d_x})$ and $\boldsymbol{v}(j_1, \ldots, j_{d_v})$ denoted $\boldsymbol{x} \otimes \boldsymbol{v}$ is a $(d_x + d_v)$-level vector with elements given by the following formula:

$$(\boldsymbol{x} \otimes \boldsymbol{v})(i_1, \ldots, i_{d_x}, j_1, \ldots, j_{d_v}) = \boldsymbol{x}(i_1, \ldots, i_{d_x}) \cdot \boldsymbol{v}(j_1, \ldots, j_{d_v}). \tag{II.1.2}$$

A linear transformation of d-level vectors may be encoded in a d-level matrix $\boldsymbol{A}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2})$. The *matrix-vector* multiplication is defined as a tensor prod-

uct of the matrix and vector followed by a contraction along the compatible indices

$$(\boldsymbol{A}\boldsymbol{x})(i_1,\ldots,i_{d_1}) = \sum_{j_1,\ldots,j_{d_2}} \boldsymbol{A}(i_1,\ldots,i_{d_1};j_1,\ldots,j_{d_2})\boldsymbol{x}(j_1,\ldots,j_{d_2}). \tag{II.1.3}$$

A major hurdle that must be overcome when working with high dimensional arrays is the so-called *curse of dimensionality* [Bel61], that is, the memory requirements and computational complexity of basic arithmetics grow exponentially in the number of dimensions.

One approach to circumventing the curse of dimensionality is inspired by the low-rank approximation of matrices by the SVD. Suppose $A \in \mathbb{R}^{m\times n}$ has rank k and we wish to find a lower rank matrix $R \in \mathbb{R}^{m\times n}$ that best approximates A in the Frobenius norm, that is, which minimizes $||A-R||_F$. Erhard Schmidt showed that the best choice of R can be expressed in terms of the SVD of A [Sch07]. Suppose A has a singular value decomposition given by:

$$A = U\Sigma V^T$$

Taking $s < \min\{m,n\}$, and take

$$R_s = U\Sigma_s V^T, \quad \text{with } (\Sigma_s)_{ij} = \begin{cases} \sigma_i & i=j\le s \\ 0 & \text{otherwise,} \end{cases} \tag{II.1.4}$$

Proposition II.1.1. *For $A \in \mathbb{R}^{m\times n}$, fix $r < k$ and define R_r as in* (II.1.4)*. R_r is the solution to the following two minimization problems:*

$$\min_{Rank(R)\le r} ||A-R||_F, \qquad \min_{Rank(R)\le r} ||A-R||_2,$$

with approximation error given by

$$||A - R||_2 = \sigma_{r+1}, \quad ||A - R||_F = \sqrt{\sum_{i=r+1}^{k} \sigma_i^2}.$$

The minimizer is unique if and only if $\sigma_r > \sigma_{r+1}$.

When the singular values of A decay rapidly, both storage and computational complexity can be reduced by using this structured representation. The storage of a rank-r matrix of size $m \times n$ in full-format requires mn parameters while the reduced SVD requires $r(m + n)$. Full-format matrix-vector multiplication has complexity $\mathcal{O}(mn)$, while the SVD representation of A requires $\mathcal{O}(r(m+n))$. Therefore when A can be well approximated by a low-rank matrix, the savings can be quite significant.

Many papers have attempted to address the curse of dimensionality by generalizing the SVD to higher dimensions [Cic+09; CV09; Gra10a; Hit26; HK09; LC09; Ose09a]. One commonly used approach is known variously as the *canonical polyadic* decomposition or *CANDECOMP/PARAFAC*, both abbreviated as *CP* [CC70; Hit26]. A CP decomposition of a d-level vector $\boldsymbol{x}$ is a sum of tensor products of 1-dimensional vectors:

$$\boldsymbol{x}(i_1, \ldots, i_d) = \sum_{k=1}^{R} x^1(i_1) \otimes \ldots \otimes x^d(i_d), \tag{II.1.5}$$

where the number of summands R is referred to as the *tensor rank* of $\boldsymbol{x}$. In applications, as long as the tensor rank of the arrays involved remain low, this approach can be very computationally efficient as basic arithmetics for tensors in the CP format scale *linearly* in the number of dimensions.

Naturally, a key challenge in applying the CP decomposition to practical problems is estimating or controlling the tensor ranks throughout the calculation since basic algebraic tensor operations such as addition and matrix-vector multiplication generally increase rank and hence computational cost. This can be quite difficult in practice. Unfortunately, the above definition implies that the approximation in Frobenius norm of a tensor with one of fixed tensor rank is ill-posed [SL08], and the numerical algorithms for computing an approximate representation may easily fail. Another obstacle is that the problem is NP-hard [Hås90; HL09] so that the complexity of known robust algorithms scale poorly.

Hence for practical applications, one desires a tensor format with three characteristics: (1) substantial compression of the vectors and matrices involved, (2) fast tensor arithmetics (ideally, scaling linearly in dimension), and (3) a fast and robust tensor approximation procedure. We make use of the Tensor Train format which makes good trade-offs between these three objectives.

II.2 Tensor Train (TT) Format

This doctoral project makes heavy use of the low-parametric representation of multi-dimensional arrays known as the *Tensor Train* (TT) format [Ose11; OT09]. The TT-format in its current form was recently developed by Oseledets and Tyrtyshnikov [Ose11; OT09], though, we remark that the Matrix Product States are an identical data format that has been known in theoretical chemistry for at least two

decades now [Vid03; VPC04; Whi93]). In addition, the TT-format is a special case of a data structure known as Tensor Network States where the associated graph is linear [VCM09].

Consider a d-dimensional $n_1 \times \ldots \times n_d$-vector $\boldsymbol{X}(i_1, \ldots, i_d)$ and assume that for the k-th dimension there is a collection of $r_{k-1} \times r_k$ matrices $X_k(i_k)$ indexed by $1 \leq i_k \leq n_k$ such that

$$\boldsymbol{X}(i_1, \ldots, i_d) = X_1(i_1) \times X_2(i_2) \times \ldots \times X_d(i_d). \tag{II.2.1}$$

We say that $\boldsymbol{X}$ is represented in the *Tensor Train (TT) format* with *TT-cores* $X_1(\cdot), \ldots, X_d(\cdot)$, where each TT-core is a one-parameter family of matrices $X_k(i_k)$. The matrix sizes $r_1, \ldots, r_{d-1}$ are referred to as the *TT-ranks* of the decomposition. Note that for some fixed index values, the product of the corresponding matrices is of size $r_0 \times r_d$ so we constrain $r_0 = r_d = 1$. See Figure II.1 for a schematic drawing.

For any d-dimensional vector $\boldsymbol{X}$ in full format, there is a robust procedure for computing a TT-decomposition by successively computing low-rank approximations of its *unfolding matrices*. For $k = 1, \ldots, d-1$ the kth unfolding matrix $\boldsymbol{X}^{(k)}$ consists of the entries

$$\boldsymbol{X}^{(k)}(i_1 \ldots i_k; i_{k+1} \ldots i_d) = \boldsymbol{X}(i_1, \ldots, i_d),$$

where $i_1 \ldots i_k$ and $i_{k+1} \ldots i_d$ are treated as multi-indices. The following example of a 3-dimensional array has two unfolding matrices.

Example II.2.1 (Unfolding of a tensor)**.** *Consider a tensor* $\boldsymbol{X}$ *of size* $3 \times 2 \times 2$*. It*

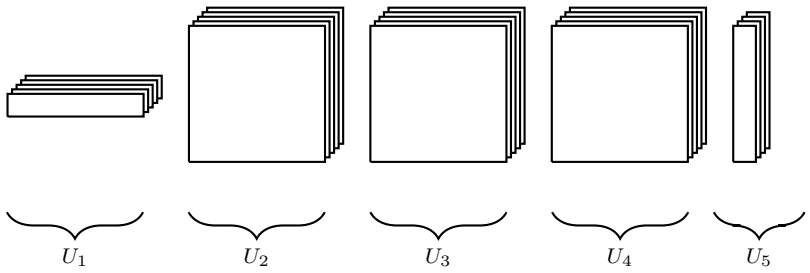


Figure II.1: **Schematic drawing of a TT decomposition of a five-dimensional array.** Each TT core can be visualized as a stack of matrices with the size of the stack equal to the corresponding mode size. The number of TT cores is equal to the number of dimensions of the array. Element $\boldsymbol{u}(j_1, \ldots, j_5)$ of the full array is given by the (matrix) product of matrix j_1 selected from core U_1, matrix j_2 from core U_2, etc. Note that the size of each matrix within a core must be the same, but may differ between distinct cores. Note also that the number of matrices in each core depends on the corresponding mode size of the full tensor and generally differs between cores. This graphical representation is widely used for the *Matrix Product States*, see [Vid03; VPC04; Whi93]

has two unfolding matrices $\boldsymbol{X}^{(1)}$ *and* $\boldsymbol{X}^{(2)}$ *given by*

$$\boldsymbol{X}^{(1)} = \begin{pmatrix} x_{111} & x_{121} & x_{112} & x_{122} \\ x_{211} & x_{221} & x_{212} & x_{222} \\ x_{311} & x_{321} & x_{312} & x_{322} \end{pmatrix} \quad \textit{and} \quad \boldsymbol{X}^{(2)} = \begin{pmatrix} x_{111} & x_{112} \\ x_{211} & x_{212} \\ x_{311} & x_{312} \\ x_{121} & x_{122} \\ x_{221} & x_{222} \\ x_{321} & x_{322} \end{pmatrix}.$$

While $\boldsymbol{X}$, $\boldsymbol{X}^{(1)}$, *and* $\boldsymbol{X}^{(2)}$ *are structured differently, all have the same entries and represent the same data.*

Note that the index ordering plays a crucial role in determining the numerical values of the TT-ranks and that alternate index orderings may significantly change the TT-ranks.

Unlike the CP decomposition, the compression (TT) ranks are readily computable since each is the matrix rank of the corresponding unfolding matrix, see Theorem 2.1 in [Ose11]. Once the TT-ranks of a full format vector are known the TT-cores of the decomposition can be computed using numerically stable linear algebra routines, e.g. QR and SVD.

We may also apply the TT format to multidimensional matrices. Consider a d-dimensional $(m_1 \times \ldots \times m_d) \times (n_1 \times \ldots \times n_d)$-matrix $\boldsymbol{A}(i_1, \ldots, i_d; j_1, \ldots, j_d)$ and assume that for the k-th dimension there is a collection of $r_{k-1} \times r_k$ matrices $A_k(i_k, j_k)$

indexed by (i_k, j_k) such that

$$\boldsymbol{A}(i_1, \ldots, i_d; j_1, \ldots, j_d) = A_1(i_1, j_1) \times A_2(i_2, j_2) \ldots \times A_d(i_d, j_d). \tag{II.2.2}$$

We say that $\boldsymbol{A}$ is represented in the *Tensor Train-Matrix (TTM) Format*. The same definitions and properties of the TT decomposition of vectors applies to the TTM format for matrices.

Basic tensor arithmetics with vectors and matrices in the TT format, such as addition, Hadamard and dot products, multi-dimensional contraction, matrix-vector multiplication, etc. are described in detail in [Ose11].

Note that the storage cost and complexity of many basic operations in the TT format are *linear* in d and polynomial in the TT-ranks. TT methods are therefore seen as a means of lifting the so-called Curse of Dimensionality [Bel61] in many applications [OT09]. We emphasize that the polynomial dependence on the TT-ranks means that it is crucial to characterize the growth of the TT-ranks whenever possible.

II.3 Tensor Rounding in the TT Format

Let $\boldsymbol{X}(i_1, \ldots, i_{d_1})$ be a multilevel array with TT decomposition

$$\boldsymbol{X}(i_1, \ldots, i_d) = X_1(i_1) \times \ldots \times X_d(i_d)$$

and suppose that it has suboptimal TT-ranks. A fast and robust tensor rounding procedure is available based on the QR and SVD algorithms for single matrices.

In this context, rounding is understood to mean finding a vector $\boldsymbol{Y}$ with smaller TT ranks close enough to satisfy a prescribed accuracy tolerance ε in the Frobenius norm [Ose11].

$$||\boldsymbol{X} - \boldsymbol{Y}||_F \leq \varepsilon ||\boldsymbol{X}||_F$$

In this section we briefly overview the relevant results.

The k-th unfolding matrix $\boldsymbol{X}^{(k)}$ may be written as the product

$$\boldsymbol{X}^k = U_k V_k^T,$$

where

$$\begin{aligned} U_k(i_1, \ldots, i_k; \alpha) &= X_1(i_1) \times \ldots \times X_k(i_k; \alpha), \\ V_k(i_{k+1}, \ldots, i_d; \alpha) &= X_{k+1}(\alpha; i_{k+1}) \times \ldots \times X_d(i_d). \end{aligned}$$

The singular value decomposition of $\boldsymbol{X}^{(k)}$ may be computed from the QR factorization of U_k and V_k in the following way. Suppose U_k and V_k have "economy" QR decompositions given by:

$$U_k = Q_{U_k} R_{U_k}, \quad V_k = Q_{V_k} R_{V_k},$$

where both Q_{U_k} and Q_{V_k} have r_k orthonormal columns and R_{U_k} and R_{V_k} are each $r_k \times r_k$ upper triangular matrices. Let $P = R_{U_k} R_{V_k}^T$, and compute its SVD:

$$P = U_P D V_P^T,$$

where U_P and V_P have orthonormal columns, and D is an $\hat{r}_c \times \hat{r}_c$ diagonal matrix. Let

$$\hat{U}_x = Q_U U_P, \quad \hat{V}_x = Q_V V_P.$$

Both $\hat{U}_x$ and $\hat{V}_x$ have orthonormal columns. Therefore,

$$\boldsymbol{X}^{(k)} = \hat{U}_x D \hat{V}_x^T,$$

is a singular value decomposition of $\boldsymbol{X}^{(k)}$ with singular vectors given by the columns of $\hat{U}_k$ and $\hat{V}_k$ and singular values listed in descending order along the diagonal of D.

Computing the SVD of P can be done directly since P is assumed to be small. Computing the QR decompositions of each factor U_k and V_k when they are in the TT format is less straightforward but can be done efficiently using the algorithm described in [Ose11]. It is based on the following lemma.

Lemma II.3.1 (Lemma 3.1 of [Ose11])**.** *If the tensors $\boldsymbol{U}_k$ and $\boldsymbol{V}_k$ are written as*

$$\begin{aligned} \boldsymbol{U}_k(i_1, \ldots, i_k; \alpha) &= Q_1(i_1) \ldots Q_k(i_k; \alpha), \\ \boldsymbol{V}_k(i_{k+1}, \ldots, i_d; \alpha) &= Q_{k+1}(\alpha; i_{k+1}) \ldots Q_d(i_d) \end{aligned}$$

where each $Q_s(i_s)$ is an $r_{s-1} \times r_s$ matrix, and satisfies one of the orthogonality conditions

$$\sum_{i_s} Q_s(i_s)^T Q_s(i_s) = I_{r_{s-1}} \quad s = 1, \ldots, k, \tag{II.3.1}$$

$$\sum_{i_s} Q_s(i_s) Q_s^T(i_s) = I_{r_s} \quad s = k+1, \ldots, d, \tag{II.3.2}$$

then $\boldsymbol{U}_k$ considered as an $\prod_{s=1}^k n_s \times r_k$ matrix U_k has orthonormal columns:

$$(U_k^T U_k)_{\alpha,\hat{\alpha}} = \sum_{i_1,\ldots,i_k} U_k^T(\alpha; i_1, \ldots, i_k) U_k(i_1, \ldots, i_k; \hat{\alpha}) = (I_{r_k})_{\alpha,\hat{\alpha}}, \tag{II.3.3}$$

and $\boldsymbol{V}_k$ considered as an $r_k \times \prod_{s=1}^k n_s$ matrix V_k^T has orthonormal rows:

$$(V_k^T V_k)_{\alpha,\hat{\alpha}} = \sum_{i_{k+1},\ldots,i_d} V_k^T(\alpha; i_{k+1}, \ldots, i_d) V_x(i_{k+1}, \ldots, i_d; \hat{\alpha}) = (I_{r_k})_{\alpha,\hat{\alpha}}, \tag{II.3.4}$$

We note that the statement of the lemma in [Ose11] only covers the case (**??**) corresponding to V_x. The proof for the dual case of U_x is exactly the same except for the exchange of transposes in the appropriate places. We leave the details to the reader.

For any multilevel array given in the TT format, there is an algorithm for transforming the first k cores into a format satisfying (II.3.1) and the last $d-k$ cores into a format satisfying (II.3.2) up to multiplication by matrices of size $r_k \times r_k$ [Ose11]. It is based on a successive reshaping and QR factorization of each core along with a contraction of the R factor with the succeeding core. The algorithm for the structured QR decomposition of $\boldsymbol{V}_k$ is given in [Ose11] and referred to as the the *Right-to-Left Orthogonalization (qr_rl)*. The *Left-to-Right Orthogonalization (qr_lr)* is a similar algorithm for $\boldsymbol{U}_k$ and is summarized in Algorithm 1.

Having both the Left to Right and Right to Left Orthogonalization algorithms implementing the structured QR factorizations in the TT format, the tensor rounding of an array in the TT format can be computed using the procedure described previously. The idea is to successively compute low-rank approximations of each of the unfolding matrices using the SVD. II.3.1 implies that it is unnecessary to expand the unfolding matrices to full format: each low-rank approximation can be obtained by considering just a single core at a time. The details are summarized in Algorithm 2.

Note that only a single run of the Right-to-Left Orthogonalization is required in practice. When starting at the left-most core, only the RL orthogonalization is

Algorithm 1 Left to Right Orthogonalization (qr_lr)

Require: Array $\boldsymbol{X}$ in the TT Format with cores $X_s(i_s)$ and with ranks $r_1, \ldots, r_d$. Unfolding number k. Implementation of the QR decomposition, qr, and a index reshape function, reshape.

Ensure: Matrix $\boldsymbol{Q}_U(i_1, \ldots, i_k; \hat{\alpha}) = Q_1(i_1)Q_2(i_2)\ldots Q_k(i_k; \hat{\alpha})$ with each Q_s satisfying (II.3.1) and matrix $\boldsymbol{R}_U(\hat{\alpha}, \alpha)$ such that $\boldsymbol{U}_x(i_1, \ldots, i_d; \alpha) = \sum_\alpha \boldsymbol{Q}_U(i_1, \ldots, i_k; \hat{\alpha})\boldsymbol{R}_U(\hat{\alpha}, \alpha)$.

$[Q_1(i_1; \hat{\alpha}_1), R_1(\hat{\alpha}_1; \alpha_1)] = \mathrm{qr}(X_1(i_1; \alpha_1))$ for the first unfolding $\boldsymbol{X}^{(1)}$,

for $s = 2$ to d **do**

$\hat{X}_s(\hat{\alpha}_{s-1}; i_s; \alpha_s) = \sum_{\alpha_1} R_{1,s-1}(\hat{\alpha}_{s-1}; \alpha_{s-1})X_s(\alpha_{s-1}; i_s; \alpha_s)$,

$\hat{X}_s(\hat{\alpha}_{s-1}, i_s; \alpha_s) = \mathrm{reshape}(\hat{X}_s(\hat{\alpha}_{s-1}; i_s; \alpha_s))$,

$[Q_s(\hat{\alpha}_{s-1}, i_s; \hat{\alpha}_s), R_s(\hat{\alpha}_s; \alpha_s)] = \mathrm{qr}(\hat{X}_s(\hat{\alpha}_{s-1}, i_s; \alpha_s))$,

$Q_s(\hat{\alpha}_{s-1}; i_s; \hat{\alpha}_s) = \mathrm{reshape}(Q_s(\hat{\alpha}_{s-1}, i_s; \hat{\alpha}_s))$,

end for

$\boldsymbol{R}_U = R_d$.

necessary for truncation of the first core. The orthogonality of V from the SVD can be used to ensure that the cores of $\boldsymbol{Y}$ are in the proper format for the succeeding rank-reduction.

Algorithm 2 TT-Rounding

Require: Tensor $\boldsymbol{X}$ in the TT-Format with cores $X_k(i_k)$ and with ranks $r_1, \ldots, r_{d-1}$, implementation of the qr_lr$(\cdot)$ algorithm as in [Ose11], the qr_lr$(\cdot)$ algorithm as in Algorithm 1, and the singular value decomposition, svd$_\delta(\cdot)$, required accuracy ε.

Ensure: Tensor $\boldsymbol{Y}$ in TT-format with TT-ranks $\hat{r}_1, \ldots, \hat{r}_{d-1}$ less than or equal to the ranks of the truncated unfoldings $\boldsymbol{X}^{(k)}$, where each truncation is computed to accuracy $\delta = \frac{\varepsilon}{\sqrt{d-1}}||\boldsymbol{X}||_F$. The computed approximation satisfies the relative accuracy requirement $||\boldsymbol{X} - \boldsymbol{Y}||_F \leq \varepsilon||\boldsymbol{X}||_F$.

Compute $\delta = \frac{\varepsilon}{\sqrt{d-1}}||\boldsymbol{X}||_F$,

$[\boldsymbol{Q}^T, \boldsymbol{R}^T] = \text{qr\_rl}(\boldsymbol{X})$,

$Y_1(i_1) = \boldsymbol{R}(i_1; \hat{\alpha})$,

for $k = 2$ to d **do**

$Y_k(i_k) = Q_k(i_k)$

end for

for $k = 1$ to $d - 1$ **do**

$[Y_k(\alpha_{k-1}, i_k; \gamma_k), D, V^T] = \text{svd}_\delta(Q_k(\alpha_{k-1} i_k; \alpha_k))$,

$Q_{k+1}(\alpha_k; i_{k+1}; \alpha_{k+1}) = \sum_{\gamma;\alpha_k} V(\alpha_k; \gamma) D(\gamma; \gamma) Q_{k+1}(\alpha_k; i_{k+1}; \alpha_{k+1})$,

$Y_k(\alpha_{k-1}; i_k; \gamma_k) = \text{reshape}(Y_k(\alpha_{k-1}, i_k; \gamma_k))$,

end for

II.4 Tensorization and Quantized Tensor Train Format

The close connection between the TT-ranks of a tensor and the ranks of its unfolding matrices leads to the informal interpretation of the ranks as being a measure of the "separability" of the data in each dimension. A route to further reduce the complexity is to take advantage of structure in each dimension, e.g. by expansion in a carefully selected reduced basis, etc. One approach to this complexity reduction in the modes is by "tensorization" or "quantization folding" of the array and applying the TT format to the resulting data structure. This approach leads to the *Quantized Tensor Train* (QTT) format [Kho11; Ose09b; Ose10a].

Suppose that each mode size n_k can be factorized as $n_k = n_{k,1} \cdot n_{k,2} \cdot \ldots \cdot n_{k,l_k}$ in terms of integral factors $n_{k,1}, \ldots, n_{k,l_k} \geq 2$. Quantization folding with respect to "physical" dimension k consists of artificially folding the tensor into these l_k "virtual" dimensions. Typically, one uses the finest quantization, i.e., $n_{k,\hat{k}} = 2$ for $\hat{k} = 1, \ldots, l_k$. For example, if $n_k = 2^{10}$, then the finest possible quantization would fold the k-th dimension into $l_k = 10$ dimensions with each corresponding index taking $n_{k,\hat{k}} = 2$ values. The folding preserves the number of entries in the vector (matrix) but, ideally, makes more structure in the data accessible to the TT compression. Applying the TT decomposition to a tensor whose "physical" dimensions have all been folded results in a *QTT decomposition* of the original vector. The ranks of this TT decomposition

are called *QTT-ranks* of the original vector.

If the natural ordering

$$\underbrace{i_{1,1},\ldots,i_{1,l_1}}_{\text{1st dimension}},\underbrace{i_{2,1},\ldots,i_{2,l_2}}_{\text{2nd dimension}},\ldots,\underbrace{i_{d,1},\ldots,i_{d,l_d}}_{d\text{th dimension}} \tag{II.4.1}$$

of the "virtual" indices is used, then the QTT-ranks are ordered as follows:

$$\underbrace{r_{1,1},\ldots,r_{1,l_1-1}}_{\text{1st dimension}},\hat{r}_1,\underbrace{r_{2,1},\ldots,r_{2,l_2-1}}_{\text{2nd dimension}},\hat{r}_2,\ldots,\hat{r}_{d-1},\underbrace{r_{d,1},\ldots,r_{d,l_d-1}}_{d\text{th dimension}},$$

where $\hat{r}_1,\ldots,\hat{r}_{d-1}$ are the TT ranks of the *original* tensor. That is, the folding preserves the TT-ranks.

Since the QTT decomposition *is* a TT decomposition, all the properties and algorithms available for the TT format carry over to the QTT format.

Quantization has been shown to be crucial in many practical applications for reducing the complexity. While a simple characterization of when a vector (matrix) will have low-rank QTT structure is unavailable QTT-rank bounds are available for many simple functions evaluated on tensor grids. Consider the following examples taken from the literature.

Example II.4.1 (Proposition 1.1 in [Kho11])**.** *Consider the one-dimensional vector* $\boldsymbol{u}$ *whose entries are given by evaluation of the exponential with base* $q > 0$ *on the nonnegative integers* $\{0,1,\ldots,2^l-1\}$*:* $\boldsymbol{u} = \left(1, q, \ldots, q^{2^l-1}\right)^\top$ *. Originally, there is only one dimension in this vector, and the elementwise representation requires storage of* 2^l *parameters since it does not exploit any structure in the data. However, if we use the quantization approach described above to split the single dimension into* l *virtual*

levels, the one-dimensional vector is transformed into l-dimensional one which exhibits a low-parametric structure. Indeed, in terms of the "virtual" indices it is a rank-one Kronecker product of l vectors with 2 components each:

$$\boldsymbol{u} = \begin{pmatrix} 1 \\ q^{2^{l-1}} \end{pmatrix} \otimes \begin{pmatrix} 1 \\ q^{2^{l-2}} \end{pmatrix} \otimes \ldots \otimes \begin{pmatrix} 1 \\ q \end{pmatrix},$$

*which implies both rank-*1 *CP and QTT decompositions of* $\boldsymbol{u}$*.*

Other important classes of functions for which explicit QTT decompositions have been found are univariate polynomials evaluated on uniform grids [Ose13a], as well as univariate asymptotically smooth functions [Gra]. See [KK12; KKT11; KRS13; Ose13a] for other explicit low-rank examples. When a uniform QTT-rank bound is available, this implies a *logarithmic* scaling of the storage and complexity of basic arithmetics with the mode size n_k [Ose09b].

II.5 Density Matrix Renormalization Group

For certain optimization problems formulated in the TT format, the Alternating Least Squares approach yields a useful set of optimization algorithms that work well at least locally [HRS12; RU12]. One such algorithm is based on the Density Matrix Renormalization Group approach [Vid03; VPC04; Whi93]. The basic idea is to consider a succession of local optimization problems in which all but two adjacent cores are fixed and to optimize over the parameters of these two cores together. Assuming that the optimization objective is a quadratic cost function, by contracting the two

cores into a so-called *supercore*, each local problem reduces to a linear least squares problem which may be solved precisely when small or approximately using Lanzcos or Arnoldi Iteration when large. The new supercore can then be re-split into the component cores in an optimal way, e.g. using SVD. By sweeping through the chain of cores sequentially one hopes that the method converges to a global minimum of the full optimization problem.

Practically, DMRG based algorithms have attractive properties. While they still lack a rigorous theoretical foundation, they prove to be highly efficient in many applications (including our experiments) where they converge quickly when the minimizer can be expressed in TT format with low TT-ranks. The iterative process is also completely rank-adaptive. When the ranks of the actual solution are large, the DMRG-based algorithms will automatically enlarge the ranks. When the ranks are small, it will truncate the excessive ranks so that calculations can be completed efficiently.

One DMRG based algorithm used frequently in the subsequent chapters is the solution of large-scale systems of linear equations in the TT-format [DO11]. By recasting the initial linear system

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},$$

as a residual minimization problem

$$||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}|| \rightarrow \text{ min}$$

one can then apply the DMRG methodology to compute an approximate solution of the original linear system.

Chapter III

Numerical Algorithms Using the TT-Format

As discussed in the previous chapter, the TT-ranks of a particular decomposition may depend critically on the particular index ordering or separation into levels chosen for the data. For example, consider the identity matrix I of size $2^N \times 2^N$:

$$I(i_1; j_1) = \underbrace{\left.\begin{pmatrix} 1 & 0 & & \\ 0 & \ddots & \ddots & \\ & \ddots & \ddots & 0 \\ & & 0 & 1 \end{pmatrix}\right\} 2^N}_{2^N}$$

using the finest possible quantizations $i_1 = i_{1,1} \dots i_{1,N}$ and $j_1 = j_{1,1} \dots j_{1,N}$. In the QTT-format for vectors, there is no mixing of the quantization levels of the row and column indices so that one of the unfolding matrices is just the identity matrix itself,

hence at least one of the QTT-ranks is 2^N. In the QTT Matrix format, however, the quantization levels are matched and the matrix can be written as a Kronecker product of N 2×2 matrices:

$$\begin{pmatrix} 1 & 0 & & \\ 0 & \ddots & \ddots & \\ & \ddots & \ddots & 0 \\ & & 0 & 1 \end{pmatrix} = \bigotimes_{k=1}^{N} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

which implies the existence of a decomposition with uniform QTT-ranks equal to 1.

It seems strange to assert the first index ordering over the second; after all, assuming the same level of accuracy, one should use an index structure which results in the most parsimonious representation of the data. This observation leads directly to two new TT-based data formats. The first is the Quantized and Transposed Tensor Train (QT3) format which can be summarized as selecting an index structure in which the "actual" dimensions are mixed at the different quantization levels. This format works well when there is strong positive correlation structure between dimensions in the data. The second is the TT-Vectorized Matrix (TTVM) format which is actually a low-rank SVD approximation of a linear operator.

The theoretical and algorithmic development of the QT3-format is largely the work of Vladimir Kazeev. Section III.1 gives an overview of the format and the main results necessary for the subsequent chapters. The interested reader should see [Kaz+14] for the detailed treatment. The TTVM format is my own contribution.

Section III.2 describes the format and describes basic fast tensor arithmetics that are possible. Section III.3 discusses numerical approaches to computing the eigenvalues and eigenvectors of matrices in the format as well as the approximation of inverses and matrix powers.

III.1 Quantized Transposed Tensor Trains (QT3)

The example from the beginning of the chapter demonstrates that the ordering of the indices plays a huge role in determining the compression ranks. In that example, there was more exploitable low-rank structure when ordering the dimensions by "quantization" levels, as opposed to the "physical" levels, meaning that the same separable structure was present at each quantization level across dimensions. The so-called Quantized and Transposed Tensor Train (QT3) format is based on this idea of shuffling the indicies from an appropriately quantized vector so that the corresponding quantization levels are adjacent. It was first applied to vectors in [Ose10b]. For example, for $l_1 = \ldots = l_d = l$ instead of

$$\underbrace{i_{1,1}, \ldots, i_{1,l_1}}_{\text{1st dimension}}, \underbrace{i_{2,1}, \ldots, i_{2,l_2}}_{\text{2nd dimension}}, \ldots, \underbrace{i_{d,1}, \ldots, i_{d,l_d}}_{d\text{th dimension}} \tag{III.1.1}$$

we use the ordering

$$\underbrace{i_{1,1}, \ldots, i_{d,1}}_{\text{1st level}}, \underbrace{i_{1,2}, \ldots, i_{d,2}}_{\text{2nd level}}, \ldots\ldots, \underbrace{i_{1,l}, \ldots, i_{d,l}}_{d\text{th level}}. \tag{III.1.2}$$

If $l_1, \ldots, l_d$ are not equal, we introduce trivial indices j_{k,m_k} with $n_{k,m_k} = 1$ for $l_k + 1 \leq m_k \leq \max_{1 \leq k' \leq d} l_{k'}$ and reorder the virtual indices to match (III.1.2). Since

$n_{k,m_k} = 1$, this does not affect the number of elements of the vector. Following the index reordering, the trivial indices are dropped.

The QT3 format is expected to outperform the basic QTT approach when the data has more exploitable low-rank structure at similar quantization levels rather than within a dimension. One example where this is the case is when the support of the data is tightly correlated across dimensions. This was the case in the example from the beginning of the chapter. In a later chapter, we see that the QT3 approach does well when at compressing probability density functions when the data is tightly correlated.

III.2 TT Vectorized Matrix Format

This section describes an alternate formatting of matrices called the Tensor Train Vectorized Matrix (TTVM) format. This format is rank-revealing meaning that it exploits the low-rank structure of the matrix for compression and efficient computation. We discuss the properties of this format and its interpretation as the Singular Value Decomposition of the matrix.

If $\boldsymbol{A}$ is a multilevel matrix and $\boldsymbol{v}$ is a multilevel vector then the multilevel matrix-by-vector product is defined by as:

$$(\boldsymbol{A}\boldsymbol{v})(i_1, \ldots, i_{d_1}) = \sum_{j_k} \boldsymbol{A}(i_1, \ldots, i_{d_1}, j_1, \ldots, j_{d_2})\boldsymbol{v}(j_1, \ldots, j_{d_2})$$

where we refer to the indices i_k as the *row indices* and the indices j_k as the *column*

indices of $\boldsymbol{A}$.

The TT-Matrix format requires the number of row indices to be equal to the number of column indices $d = d_1 = d_2$. It uses the index ordering

$$(i_1, j_1), (i_2, j_2), \ldots, (i_d, j_d)$$

and uses the Tensor Train compression with each core indexed by one of the ordered pairs (i_k, j_k) using the lexicographic ordering. A d-level matrix with $2d$-indices as an array, will have d cores.

In contrast, the TTVM format uses the simple index ordering

$$i_1, \ldots, i_{d_1}, j_1, \ldots, j_{d_2},$$

with a core indexed by each array index. That is, for a multilevel matrix with $d_1 + d_2$ indices as an array, a TTVM representation of the matrix will have $d_1 + d_2$ cores

$$\boldsymbol{A}(i_1, \ldots, i_{d_1}, j_1, \ldots, j_{d_2}) = A_{1,1}(i_1) \times \ldots \times A_{1,d_1}(i_{d_1}) A_{2,1}(j_1) \times \ldots \times A_{2,d_2}(j_{d_2}). \quad \text{(III.2.1)}$$

The TTVM-ranks of the decomposition can be enumerated as follows:

$$\text{TTVM\_rank}(\boldsymbol{A}) = r_{1,1}, \ldots, r_{1,d_1-1}, \hat{r}_c, r_{2,1}, \ldots, r_{2,d_2-1},$$

where the rank $\hat{r}_c$ corresponds to the (matrix) rank of the unfolding matrix

$$\boldsymbol{A}^{(c)} = \boldsymbol{A}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}).$$

That is, $\boldsymbol{A}$ as a linear operator has rank $\hat{r}_c$. Given a matrix in the TTVM format, the rank of the corresponding linear operator is immediately known. Note that any

permutation of the first (last) $d-1$ indices will not affect $\hat{r}_c$ since this corresponds to reordering the rows (columns) of the unfolding matrix.

In the next subsection we introduce the TTVM stack matrix which we will use when computing the SVD of a general TT Vectorized Matrix.

III.2.1 Stacking TT-formatted vectors into a TTVM-formatted Matrix

Given an ordered list of vectors $\left\{v^1, \ldots, v^N\right\}$ that all have the same size, a common procedure in matrix linear algebra is to construct matrices by stacking the vectors, that is, constructing matrices whose nth row or column is the n-th vector in the list:

$$\left\{v^1, \ldots, v^N\right\} \quad \longmapsto \quad V = \begin{bmatrix} v^1 & v^2 & \ldots & v^N \end{bmatrix}, \quad V^T = \begin{bmatrix} v^1 \\ v^2 \\ \vdots \\ v^N \end{bmatrix}.$$

A similar data structure can be constructed for a list of TT-formatted d-vectors with compatible mode sizes. The result is a TTVM formatted matrix.

Fix $d, N \in \mathbb{N}$. Suppose that for each $k \in \{1, \ldots, N\}$, $\boldsymbol{v}^k(i_1, \ldots, i_d)$ is a d-dimensional vector with TT decomposition given by,

$$\boldsymbol{v}^k(i_1, \ldots, i_d) = V_1^k(i_1) \times \ldots \times V_d^k(i_d), \tag{III.2.2}$$

and suppose further that the corresponding mode sizes are the same for every k. We

define the *stack matrix* $\boldsymbol{V}$ to be the $d+1$-dimensional array whose slices in the last index are the d-vectors $\boldsymbol{v}^k$. Its transpose $\boldsymbol{V}^T$ is similarly defined but with the slices along the first index. Explicit TTVM representations of each are given by

$$\begin{aligned}
\boldsymbol{V}(i_1,\ldots,i_d;i_s) &= \begin{bmatrix} V_1^1(i_1) & V_1^2(i_1) & \ldots & V_1^N(i_1) \end{bmatrix} \begin{bmatrix} V_2^1(i_2) & 0 & \cdots & 0 \\ 0 & V_2^2(i_2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & V_2^N(i_2) \end{bmatrix} \cdots \\
&\quad \cdots \begin{bmatrix} V_d^1(i_d) & 0 & \cdots & 0 \\ 0 & V_d^2(i_d) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & V_d^N(i_d) \end{bmatrix} \begin{bmatrix} \delta_1(i_s) \\ \delta_2(i_s) \\ \vdots \\ \delta_N(i_s) \end{bmatrix}, \\
\boldsymbol{V}^T(i_s,i_1,\ldots,i_d) &= \begin{bmatrix} \delta_1(i_s) & \delta_2(i_s) & \ldots & \delta_N(i_s) \end{bmatrix} \begin{bmatrix} V_1^1(i_1) & 0 & \cdots & 0 \\ 0 & V_1^2(i_1) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & V_1^N(i_1) \end{bmatrix} \cdots \\
&\quad \cdots \begin{bmatrix} V_{d-1}^1(i_{d-1}) & 0 & \cdots & 0 \\ 0 & V_{d-1}^2(i_{d-1}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & V_{d-1}^N(i_{d-1}) \end{bmatrix} \begin{bmatrix} V_d^1(i_d) \\ V_d^2(i_d) \\ \vdots \\ V_d^N(i_d) \end{bmatrix}, \quad \text{(III.2.3)}
\end{aligned}$$

where $\delta_l(i_s)$ denotes the Kronecker delta taking the value one at l and zero everywhere else, and 0 denotes the matrix of all zeros of the compatible sizes. In each case, we

refer to i_s as the *selection index*. In each case the explicit representation also provides an upper bound on the TT ranks of the corresponding vectorized matrix, though in both cases we emphasize that representations with significantly smaller ranks may be possible.

Proposition III.2.1. *If each vector $\boldsymbol{v}^k(i_1, \ldots, i_d)$ has TT ranks $r_1^k, \ldots, r_{d-1}^k$, then for the stack matrix $\boldsymbol{V}(i_1, \ldots, i_d, i_s)$ there exists a TTVM representation satisfying the rank bound*

$$TT\_Ranks(\boldsymbol{V}) \leq \sum_{n=1}^{N} r_1^n, \ldots, \sum_{n=1}^{N} r_{d-1}^n, N.$$

Similarly for the stack matrix $\boldsymbol{V}^T(i_s, i_1, \ldots, i_d)$ there is a TTVM decomposition satisfying the bound

$$TT\_Ranks(\boldsymbol{V}^T) \leq N, \sum_{n=1}^{N} r_1^n, \ldots, \sum_{n=1}^{N} r_{d-1}^n.$$

The stack matrix representation is commonly used in linear control theory. We will also use it in the next section to describe the singular value decomposition of a TTVM-formatted matrix.

III.2.2 Singular Value Decomposition of a TTVM-formatted matrix

The singular value decomposition of a linear operator in the TTVM format may be easily computed using a modification of the TT rounding procedure, discussed in the previous chapter.

Let $\boldsymbol{X}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2})$ be a multilevel matrix with TTVM decomposition

$$\boldsymbol{X}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}) = X_{1,1}(i_1) \times \ldots \times X_{1,d_1}(i_{d_1}) \times X_{2,1}(j_1) \times \ldots \times X_{2,d_2}(j_{d_2})$$

The unfolding matrix $\boldsymbol{X}^{(c)}$ may be written as the product

$$\boldsymbol{X}^{(c)} = U_x V_x^T,$$

where

$$\begin{aligned} U_x(i_1, \ldots, i_{d_1}; \alpha) &= X_{1,1}(i_1) \times \ldots \times X_{1,d_1}(i_{d_1}; \alpha), \\ V_x(j_1, \ldots, j_{d_2}; \alpha) &= X_{2,1}(\alpha; j_1) \times \ldots \times X_{2,d_2}(j_{d_2}). \end{aligned}$$

The singular value decomposition of $\boldsymbol{X}^{(c)}$ may be computed from the QR factorization of U_x and V_x in the following way. Both U_x and V_x have "economy" QR decompositions which can be computed using the Left-to-Right and Right-to-Left Orthogonalization algorithms described previously:

$$U_x = Q_U R_U, \quad V_x = Q_V R_V,$$

where both Q_U and Q_V have $\hat{r}_c$ orthonormal columns and R_U and R_V are each $\hat{r}_c \times \hat{r}_c$ upper triangular matrices. Let $P = R_U R_V^T$, and compute its SVD:

$$P = U_P D V_P^T,$$

where U_P and V_P have orthonormal columns, and D is an $\hat{r}_c \times \hat{r}_c$ diagonal matrix. Let

$$\hat{U}_x = Q_U U_P, \quad \hat{V}_x = Q_V V_P.$$

Both $\hat{U}_x$ and $\hat{V}_x$ have orthonormal columns. Therefore,

$$\boldsymbol{X}^{(c)} = \hat{U}_x D \hat{V}_x^T,$$

is a singular value decomposition of $\boldsymbol{X}^{(c)}$ with singular vectors given by the columns of $\hat{U}_x$ and $\hat{V}_x$ and singular values listed in descending order along the diagonal of D.

The TTVM-SVD takes a multilevel matrix $\boldsymbol{X}$ in the TTVM format and returns TTVM-formatted stack matrices $\hat{\boldsymbol{U}}_x(i_1, \ldots, i_{d_1}; \alpha)$, $\hat{\boldsymbol{V}}_x^T(\hat{\alpha}; i_1, \ldots, i_{d_2})$ containing the right and left singular vectors, respectively, of $\boldsymbol{X}$, and a diagonal matrix D listing the corresponding singular values. The details are summarized in Algorithm 3.

Since the TTVM format can be interpreted as the SVD of a matrix, it is possible to give a rough characterization of the TT ranks in terms of the rank of the matrix and the TT ranks of the singular vectors.

Proposition III.2.2. *Suppose the multilevel matrix $\boldsymbol{X}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2})$ given in TTVM format has singular value decomposition given by*

$$\boldsymbol{X}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}) = \sum_{k=1}^{\hat{r}_c} \sigma_k \times (\boldsymbol{u}_k(i_1, \ldots, i_{d_1}) \otimes \boldsymbol{v}_k(j_1, \ldots, j_{d_2})) \qquad \text{(III.2.4)}$$

where $\hat{r}_c$ is the rank of $\boldsymbol{X}$, σ_k's are the ordered singular values, and $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$ are the corresponding left and right singular vectors, respectively. If the TT ranks of $\boldsymbol{u}_k$ are given by

$$TT\_ranks(\boldsymbol{u}_k) = r_{1,1}^k, \ldots, r_{1,d_1-1}^k,$$

and the TT ranks of $\boldsymbol{v}_k$ are given by

$$TT\_ranks(\boldsymbol{v}_k) = r_{2,1}^k, \ldots, r_{2,d_2-1}^k,$$

Algorithm 3 TTVM-SVD

Require: Matrix $\boldsymbol{X}$ in the TTVM Format with cores $X_{1k}(i_k), X_{2k}(j_k)$ and with ranks $r_{1,1}, \ldots, r_{1,d_1-1}, \hat{r}_c, r_{2,1}, \ldots, r_{2,d_2-1}$, implementation of the qr_lr$(\cdot)$ algorithm as in [Ose11], the qr_lr$(\cdot)$ algorithm as in Algorithm 1, and the singular value decomposition, svd$(\cdot)$.

Ensure: Matrices $\hat{\boldsymbol{U}}_x(i_1, \ldots, i_{d_1}; \alpha)$, $\hat{\boldsymbol{V}}_x^T(\hat{\alpha}; j_1, \ldots, j_{d_2})$ in the TTVM-format with cores $\hat{U}_1(i_1), \ldots, \hat{U}_d(i_{d_1}), \hat{U}_{d_1+1}(\alpha)$, and $\hat{V}_0(\hat{\alpha}), \hat{V}_1(j_1), \ldots, \hat{V}_{d_2}(j_{d_2})$, respectively, and a diagonal matrix D in full format containing the singular values of $\boldsymbol{X}$, such that $\boldsymbol{X}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}) = \sum_{\alpha,\hat{\alpha}} \hat{\boldsymbol{U}}_x(i_1, \ldots, i_{d_1}; \alpha) D(\alpha, \hat{\alpha}) \hat{\boldsymbol{V}}_x^T(\hat{\alpha}; j_1, \ldots, j_{d_2})$.

$[\boldsymbol{Q}_2^T, \boldsymbol{R}_V^T] = \text{qr\_rl}(\boldsymbol{X})$,

$[\boldsymbol{Q}_1, \boldsymbol{R}_U] = \text{qr\_lr}(\boldsymbol{X})$,

$[U_P, D, V_P^T] = \text{svd}(\boldsymbol{R}_U \boldsymbol{R}_V^T)$,

$\hat{V}_0 = V_P^T$,

$\hat{U}_{d_1+1} = U_p$,

for $k = 1$ to d **do**

$\hat{U}_k(i_k) = Q_{1,k}(i_k)$,

$\hat{V}_k(j_k) = Q_{2,k}(j_k)$,

end for

then there is a TTVM decomposition of $\boldsymbol{X}$ *that satisfies the following bounds on the TT-ranks:*

$$TT\_ranks(\boldsymbol{X}) \leq \sum_{k=1}^{\hat{r}_c} r_{1,1}^k, \ldots, \sum_{k=1}^{\hat{r}_c} r_{1,d_1-1}, \hat{r}_c, \sum_{k=1}^{\hat{r}_c} r_{2,1}^k, \ldots, \sum_{k=1}^{\hat{r}_c} r_{2,d_2-1}.$$

Proof. We show that $\boldsymbol{X}$ has a particular representation in the TTVM format satisfying the rank bound. Suppose $\boldsymbol{X}$ is in the TTVM format with TTVM-SVD resulting from the application of Algorithm 3:

$$\boldsymbol{X}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}) = \sum_{\alpha,\hat{\alpha}} \hat{\boldsymbol{U}}_x(i_1, \ldots, i_{d_1}; \alpha) D(\alpha, \hat{\alpha}) \hat{\boldsymbol{V}}_x^T(\hat{\alpha}; i_1, \ldots, i_{d_2}). \quad \text{(III.2.5)}$$

Both $\hat{\boldsymbol{U}}_x$ and $\hat{\boldsymbol{V}}_x^T$ are stack matrices and therefore using Proposition III.2.1, their ranks satisfy the bounds

$$\text{TT\_Ranks}(\hat{\boldsymbol{U}}_x) \leq \sum_{k=1}^{\hat{r}_c} r_{1,1}^k, \ldots, \sum_{k=1}^{\hat{r}_c} r_{d_1-1}^k, \hat{r}_c, \quad \text{TT\_Ranks}(\hat{\boldsymbol{V}}_x^T) \leq \hat{r}_c, \sum_{k=1}^{\hat{r}_c} r_{1,1}^k, \ldots, \sum_{k=1}^{\hat{r}_c} r_{d_2-1}^k.$$

Now the contraction (III.2.5) can be computed as

$$\begin{aligned}
\boldsymbol{X} &= \sum_{\alpha,\hat{\alpha}} \hat{\boldsymbol{U}}_x(i_1, \ldots, i_{d_1}; \alpha) D(\alpha, \hat{\alpha}) \hat{\boldsymbol{V}}_x^T(\hat{\alpha}; i_1, \ldots, i_{d_2}) \\
&= \sum_{\alpha,\hat{\alpha}} (\hat{U}_1(i_1) \times \ldots \times \hat{U}_{d_1}(i_{d_1}) \times \hat{U}_{d_1+1}(\alpha)) D(\alpha, \hat{\alpha}) \times \ldots \\
&\qquad \ldots \times (\hat{V}_0(\hat{\alpha}) \times \hat{V}_1(j_1) \times \ldots \times \hat{V}_{d_2}(j_{d_2})) \\
&= \sum_{\alpha,\hat{\alpha},\beta_k,\hat{\beta}_k} \hat{U}_1(i_1, \beta_1) \ldots \hat{U}_d(\beta_{d_1-1}, i_{d_1}, \beta_{d_1}) \hat{U}_{d_1+1}(\beta_{d_1}, \alpha) D(\alpha, \hat{\alpha}) \times \ldots \\
&\qquad \ldots \times \hat{V}_0(\hat{\alpha}, \hat{\beta}_0) \hat{V}_1(\hat{\beta}_0, j_1, \hat{\beta}_1) \ldots \hat{V}_{d_2}(\hat{\beta}_{d_2-1}, j_{d_2}) \\
&= \sum_{\beta_k,\hat{\beta}_k} \hat{U}_1(i_1, \beta_1) \ldots \hat{U}_{d_1}(\beta_{d_1-1}, i_{d_1}, \beta_{d_1}) \left(\sum_{\alpha,\hat{\alpha}} \hat{U}_{d_1+1}(\beta_{d_1}, \alpha) D(\alpha, \hat{\alpha}) \hat{V}_0(\hat{\alpha}, \hat{\beta}_0) \right) \times \ldots \\
&\qquad \ldots \times \hat{V}_1(\hat{\beta}_0, j_1, \hat{\beta}_1) \ldots \hat{V}_{d_2}(\hat{\beta}_{d_2-1}, j_{d_2})
\end{aligned}$$

The contraction $\sum_{\alpha,\hat{\alpha}} \hat{U}_{d_1+1}(\beta_{d_1}, \alpha) D(\alpha, \hat{\alpha}) \hat{V}_0(\hat{\alpha}, \hat{\beta}_0)$ can be realized by matrix multiplication resulting in a matrix of size $\hat{r}_c \times \hat{r}_c$. The contraction of this matrix with the core $\hat{V}_1(\hat{\beta}_0, j_1, \hat{\beta}_1)$ can be realized by reshaping $\hat{V}_1$ into a matrix of size $\hat{r}_c \times n_{2,1} r_{2,1}$ and post-multiplying. The result is a matrix of size $\hat{r}_c \times n_{2,1} r_{2,1}$ which can be reshaped into a core $\hat{\hat{V}}_1(\beta_{d_1}, j_1, \hat{\beta}_1)$ of size $\hat{r}_c \times n_{2,1} \times r_{2,1}$. Hence, $\boldsymbol{X}$ has TTVM representation

$$\boldsymbol{X}(i_1, \ldots, i_d; j_1, \ldots, j_d) = \hat{U}_1(i_1) \ldots \hat{U}_{d_1}(i_{d_1}) \hat{\hat{V}}_1(j_1) \ldots \hat{V}_{d_2}(j_{d_2}),$$

with mode sizes $n_{1,1}, \ldots, n_{1,d_1}, n_{2,1}, \ldots, n_{2,d_2}$. The cores $\hat{U}_1(i_1), \ldots, \hat{U}_{d_1}(i_{d_1}), \hat{V}_2(j_2) \ldots \hat{V}_{d_2}(j_{d_2})$ are exactly the same as from the TTVM-SVD and therefore satisfy the same TT-rank bounds. $\hat{\hat{V}}_1$ as a core has size necessary for the matrix multiplications to be well-defined therefore the TT-ranks of $\boldsymbol{X}$ satisfy the bound

$$\text{TT\_ranks}(\boldsymbol{X}) \leq \sum_{k=1}^{\hat{r}_c} r_{1,1}^k, \ldots, \sum_{k=1}^{\hat{r}_c} r_{1,d_1-1}, \hat{r}_c, \sum_{k=1}^{\hat{r}_c} r_{2,1}^k, \ldots, \sum_{k=1}^{\hat{r}_c} r_{2,d_2-1},$$

as desired. □

We emphasize that the bound is not tight and furthermore that any TTVM formatted matrix may be compressed into a decomposition with ranks no larger than those provided by the bound by applying the TT-rounding procedure (without truncation of nonzero singular values). In addition, the TTVM-SVD make explicit the rank structure of a TTVM formatted matrix. The first $d_1 - 1$ ranks are determined by the TT compression possible of stack matrices of the left singular vectors, while the last $d_2 - 1$ ranks correspond similarly to the right singular vectors. The middle

rank $\hat{r}_c$ is the matrix rank of $\boldsymbol{X}$.

$$\text{TT\_Ranks}(\boldsymbol{X}) = \underbrace{r_{1,1}, \ldots, r_{1,d_1-1}}_{\text{Left Singular Vector Ranks}} \quad , \quad \underbrace{\hat{r}_c}_{\text{Matrix Rank}} \quad , \quad \underbrace{r_{2,1}, \ldots, r_{2,d_2-1}}_{\text{Right Singular Vector Ranks}}$$

III.2.3 Converting a TTVM decomposition to a TTM decomposition

We complete this section with a characterization of the TTM decomposition of a multilevel matrix in terms of its TTVM decomposition. This establishes TT rank bounds on any minimal rank TTM decomposition of the matrix.

Proposition III.2.3. *Suppose matrix $\boldsymbol{A}$ with d row and column indices is given in the TTVM-format with cores $A_{11}(i_1), \ldots, A_{1d}(i_d)$, $A_{21}(j_1), \ldots, A_{2d}(j_d)$ and TT ranks $r_{1,1}, \ldots, r_{1,d-1}, \hat{r}_c, r_{2,1}, \ldots, r_{2,d-1}$. A TTM-decomposition of the same multilevel matrix is given by:*

$$\boldsymbol{A}(i_1, \ldots, i_d; j_1, \ldots, j_d) = \hat{A}_1(i_1, j_1)\hat{A}_2(i_2, j_2) \ldots \hat{A}_{d-1}(i_{d-1}, j_{d-1})\hat{A}_d(i_d, j_d),$$

with cores given by:

$$\begin{aligned}
\hat{A}_1(i_1, j_1) &= A_{11}(i_1) \otimes \begin{bmatrix} A_{21}^1(j_1) & A_{21}^2(j_1) & \ldots & A_{21}^{\hat{r}_c}(j_1) \end{bmatrix}, \\
\hat{A}_d(i_d, j_d) &= \begin{bmatrix} A_{1d}^1(i_d) \\ A_{1d}^2(i_d) \\ \vdots \\ A_{1d}^{\hat{r}_c}(i_d) \end{bmatrix} \otimes A_{2d}(j_d),
\end{aligned}$$

$$\hat{A}_k(i_k, j_k) \;=\; I_{\hat{r}_c} \otimes (A_{1k}(i_k) \otimes A_{2k}(j_k))\,. \tag{III.2.6}$$

and $A^{\beta}_{21}(j_1)$ *is the* β*-th row of matrix* $A_{21}(j_1)$*,* $A^{\beta}_{1d}(i_d)$ *is the* β*-th column of matrix* $A_{1d}(i_d)$*, and* $I_{\hat{r}_c}$ *is a* $\hat{r}_c \times \hat{r}_c$ *identity matrix. This decomposition has TT ranks that are the product of the matrix rank* $\hat{r}_c$*, and the corresponding TTVM ranks* $r_{1,k}$ *and* $r_{2,k}$*:*

$$\hat{r}_c r_{1,1} r_{2,1}, \ldots, \hat{r}_c r_{1,d-1} r_{2,d-1}. \tag{III.2.7}$$

Proof. Starting from the TTVM decomposition of $\boldsymbol{A}$, the formula can be expanded as

$$\begin{aligned}
&\boldsymbol{A}(i_1, \ldots, i_d; j_1, \ldots, j_d) \\
&\quad = A_{11}(i_1) \times \ldots \times A_{1d}(i_d) \times A_{21}(j_1) \times \ldots \times A_{2d}(j_d) \\
&\quad = \sum_{\beta=1}^{\hat{r}_c} A_{11}(i_1) \times \ldots \times A_{1d}(i_d, \beta) \times A_{21}(\beta, j_1) \times \ldots \times A_{2d}(j_d) \\
&\quad = \sum_{\beta=1}^{\hat{r}_c} (A_{11}(i_1) \otimes A_{21}(\beta, j_1)) \times \ldots \times (A_{1d}(i_d, \beta) \otimes A_{2d}(j_d))
\end{aligned} \tag{III.2.8}$$

Making the identifications $A^{\beta}_{21}(j_1) := A_{21}(\beta, j)$, and $A^{\beta}_{1d}(i_d) := A_{1d}(i_d, \beta)$, (III.2.8) can be viewed as the sum of $\hat{r}_c$ TTM formatted matrices $\boldsymbol{A}^{\beta}(i_1, \ldots, i_d; j_1, \ldots, j_d)$:

$$\boldsymbol{A}(i_1, \ldots, i_d; j_1, \ldots, j_d) = \sum_{\beta=1}^{\hat{r}_c} \boldsymbol{A}^{\beta}(i_1, \ldots, i_d; j_1, \ldots, j_d),$$

with

$$\boldsymbol{A}^{\beta}(i_1, \ldots, i_d; j_1, \ldots, j_d) = \left(A_{11}(i_1) \otimes A^{\beta}_{21}(j_1)\right) \times \ldots \times \left(A^{\beta}_{1d}(i_d) \otimes A_{2d}(j_d)\right),$$

where $(A_{1k}(i_k) \otimes A_{2k}(j_k))$ is of size $(r_{1,k-1} r_{2,k-1}) \times (r_{1,k} r_{2,k})$ Using the result [Ose11]

for the TT decomposition of the sum of TT formatted matrices gives the desired TTM decomposition of $\boldsymbol{A}$ as well as the TT ranks. □

The conversion from the TTVM to TTM format can be implemented easily in MATLAB using the reshape() and kron() functions. The rank characterization is not tight, meaning that there may exist an alternate TTM decomposition of $\boldsymbol{A}$ with much smaller ranks. This result establishes an upper bound on the ranks of any minimal rank TTM decomposition of $\boldsymbol{A}$ in terms of the ranks of its TTVM decomposition, therefore a matrix with small ranks in the TTVM format will have small ranks in the TTM format.

III.2.4 Basic Arithmetic Operations

Many useful algorithms for tensor arithmetic such as matrix addition, matrix-vector products, and matrix-matrix products involving matrices in the TTVM can be formulated. These algorithms take advantage of the structured representation of the matrices involved for efficient computation, often lifting the curse of dimensionality. The algorithms for computing matrix-matrix products when one matrix is in the TTVM format and the other is in the TTM format will be used extensively in the following sections.

The details of the algorithms are summarized in Table III.1. As the derivations of the basic algorithms are rather repetitive, the uninterested reader may wish to skip to next subsection.

Operation	Inputs	Outputs	Complexity
VM Addition	$\boldsymbol{A}$,$\boldsymbol{B}$ in TTVM format	$\boldsymbol{A}+\boldsymbol{B}$ in TTVM format	-
VM-V Product	$\boldsymbol{A}$ in TTVM format, $\boldsymbol{x}$ in TTV format	$\boldsymbol{Ax}$ in TTV format	$\mathcal{O}(d_2 n r^4)$
VM-M Product	$\boldsymbol{A}$ in TTVM format, $\boldsymbol{B}$ in TTM format	$\boldsymbol{AB}$ in TTVM format	$\mathcal{O}\left(d_2 n^2 r^4\right)$
VM-M-VM	$\boldsymbol{A}$, $\boldsymbol{C}$ in TTVM format, $\boldsymbol{B}$ in TTM format	$\boldsymbol{ABC}$ in TTVM format	$\mathcal{O}\left(d_2 n^2 r^4 + d_2 n r^7\right)$

Table III.1: Algorithms involving TTVM formatted matrices

A multilevel matrix $\boldsymbol{A}$ in the TTVM format with d_1 row indices and d_2 column indices may alternatively be viewed as a multilevel vector in the TT format with $d_1 + d_2$ indices. An algorithm for the summation of two such vectors with compatible indices is given in [Ose11]. The result is a TT formatted vector whose ranks are the sums of the corresponding ranks of the addends. The assembly of the solution vector in TT format requires virtually no operations. The same algorithm can be used for TTVM formatted matrices and results in the same rank characterization.

III.2.4.1 Partial summations in the TT format

The partial summation of the tensor $\boldsymbol{X}(i_1, \ldots, i_d)$ over the index $i_\alpha \in \mathcal{I}_\alpha$ is a key building block in many algorithms for performing tensor arithmetic in the TT format. In some cases, $\mathcal{I}_\alpha$ may be a subset of the index set for i_α. The resulting tensor has one less mode:

$$\boldsymbol{X}(i_1, \ldots, i_{\alpha-1}, i_{\alpha+1}, i_d) = \sum_{i_\alpha \in \mathcal{I}_\alpha} X(i_1, \ldots, i_\alpha, \ldots, i_d) \tag{III.2.9}$$

When the tensor is given in the TT format it is possible to compute the partial summation by summing over the matrices in a single core. This fact was used extensively in [Ose11] but we highlight and prove a slightly more general version here.

Proposition III.2.4 (Partial Summation)**.** *Let $\boldsymbol{X}(i_1, \ldots, i_d)$ be a tensor of dimension d given in the TT-format with cores $X_k(i_k)$.*

$$\boldsymbol{X}(i_1, \ldots, i_d) = X_1(i_1) \times \ldots \times X_d(i_d).$$

The partial summation over $i_\alpha \in \mathcal{I}_\alpha$ results in a tensor $\hat{\boldsymbol{X}}(i_1, \ldots, i_{\alpha-1}, i_{\alpha+1}, \ldots, i_d)$ of dimension $d-1$ in the TT-format with cores $\hat{X}_k(i_k)$

$$\hat{\boldsymbol{X}}(i_1, \ldots, i_{\alpha-1}, i_{\alpha+1}, \ldots, i_d) = \hat{X}_1(i_1) \times \ldots \times \hat{X}_{\alpha-1}(i_{\alpha-1}) \times \hat{X}_{\alpha+1}(i_{\alpha+1}) \times \ldots \times \hat{X}_d(i_d),$$

where

$$\hat{X}_k(i_k) = \begin{cases} ZX_{\alpha+1}(i_{\alpha+1}), & \text{if } k = \alpha + 1, \\ X_k(i_k), & \text{otherwise,} \end{cases}$$

with $Z = \sum_{i_\alpha \in \mathcal{I}_\alpha} X_\alpha(i_\alpha)$.

Proof. This follows immediately from linearity of multiplying by fixed matrices. Indeed,

$$\begin{aligned} &\sum_{i_\alpha \in \mathcal{I}_\alpha} \boldsymbol{X}(i_1, \ldots, i_\alpha, \ldots, i_d) \\ &\quad = \sum_{i_\alpha \in \mathcal{I}_\alpha} X_1(i_1) \times \ldots \times X_{\alpha-1}(i_{\alpha-1}) \times X_\alpha(i_\alpha) \times X_{\alpha+1}(i_{\alpha+1}) \times \ldots \times X_d(i_d) \\ &\quad = X_1(i_1) \times \ldots \times X_{\alpha-1}(i_{\alpha-1}) \times \left(\sum_{i_\alpha \in \mathcal{I}_\alpha} X_\alpha(i_\alpha) \right) \times X_{\alpha+1}(i_{\alpha+1}) \times \ldots \times X_d(i_d). \end{aligned}$$

□

The construction of the matrix Z has computational complexity $\mathcal{O}(nr^2)$, while the matrix by core product has complexity $\mathcal{O}(nr^3)$ since it can be realized by multiplying a matrix of size $r \times r$ with a matrix of size $r \times nr$ after a suitable reshaping. Hence, the overall computational complexity is $\mathcal{O}(nr^3)$.

Many algorithms for tensor arithmetics can be formulated as a reformatting of the tensors involved into some intermediate TT format followed by a sequence of

partial summations. In many cases, it is possible to reduce the computational cost of the partial summations by exploiting special structure of the intermediate format.

III.2.4.2 TT-Vectorized-Matrix-Vector Product

The most important operation in linear algebra is probably the matrix-by-vector product. Oseledets and Tyrtyshnikov proposed an algorithm for computing the matrix-by-vector product $\boldsymbol{Ax}$ when $\boldsymbol{A}$ is a d-level matrix in the TT-Matrix format and $\boldsymbol{x}$ is a d-level vector in the TT format [Ose11]. We introduce a similar algorithm for the matrix-by-vector product when A is in the TTVM format.

Suppose that $\boldsymbol{A}$ is in the TTVM format with decomposition

$$\begin{aligned}&\boldsymbol{A}(i_1,\ldots,,i_{d_1};j_1,\ldots,j_{d_2}) = \\ &\quad A_{11}(i_1)\ldots A_{1d_1}(i_{d_1})A_{21}(j_1)\ldots A_{2d_2}(j_{d_2}),\end{aligned} \tag{III.2.10}$$

where $A_{1k}(i_k)$ and $A_{2k}(j_k)$ are $r_{1,k-1}\times r_{1,k}$ and $r_{2,k-1}\times r_{2,k}$ matrices, respectively. Suppose that $\boldsymbol{x}$ is in the TT format (II.2.1) with cores $X_k(l_k)$. The matrix-by vector product is the computation of the following sum,

$$\boldsymbol{y}(i_1,\ldots,i_{d_1}) = \sum_{j_1,\ldots,j_{d_2}} \boldsymbol{A}(i_1,\ldots,i_{d_1},j_1,\ldots,j_{d_2})\boldsymbol{x}(j_1,\ldots,j_{d_2}),$$

which is equivalent to the tensor product of $\boldsymbol{A}$ and $\boldsymbol{x}$ followed by partial summations over the corresponding indices.

Given the TT decompositions of both $\boldsymbol{A}$ and $\boldsymbol{x}$, the tensor product $\boldsymbol{A}\otimes\boldsymbol{x}$ can

be written in TT format as

$$\boldsymbol{A} \otimes \boldsymbol{x} = A_{11}(i_1) \dots A_{1d_1}(i_{d_1}) A_{21}(j_1) \dots A_{2d}(j_{d_2}) X_1(l_1) \dots X_{d_2}(l_{d_2}).$$

Alternatively, after a permutation and combination of the indices, it may also be represented in the following format:

$$\boldsymbol{A} \otimes \boldsymbol{x} = A_{11}(i_1) \dots A_{1d_1}(i_{d_1}) \left(A_{21}(j_1) \otimes X_1(l_1)\right) \dots \left(A_{2d_2}(j_{d_2}) \otimes X_{d_2}(l_{d_2})\right), \quad \text{(III.2.11)}$$

where each $j_k l_k$ is interpreted as a multiindex. The matrix-vector product is given by the contraction over the corresponding indices, that is, the partial sums over the index sets with $j_k = l_k$. The resulting tensor will also be in the TT-format. Indeed,

$$\begin{aligned} \boldsymbol{y}(i_1, \dots, i_{d_1}) &= \sum_{j_1, \dots, j_{d_2}} \left(A_{11}(i_1) \dots A_{1d_1}(i_{d_1})\right) \left(A_{21}(j_1) \otimes X_1(j_1)\right) \times \dots \\ &\qquad \times \left(A_{2d_2}(j_{d_2}) \otimes X_{d_2}(j_{d_2})\right) \\ &= \left(A_{11}(i_1) \dots A_{1d_1}(i_{d_1})\right) Z_1 \dots Z_{d_2}, \end{aligned}$$

where

$$Z_k = \sum_{j_k} \left(A_{2k}(j_k) \otimes X_k(j_k)\right).$$

Taking

$$Y_k(i_k) = \begin{cases} A_{1k}(i_k), & k \neq d_1 \\ A_{1k}(i_k) Z_1 \dots Z_{d_2}, & k = d_1 \end{cases},$$

we have that

$$\boldsymbol{y}(i_1, \dots, i_{d_1}) = Y_1(i_1) \dots Y_{d_1}(i_{d_1}),$$

and the product is in the TT-format. A formal description of the algorithm is presented in Algorithm 1.

Algorithm 4 TT-Vectorized-Matrix-by-Vector Product

Require: Matrix $\boldsymbol{A}$ in the TTVM-Format with cores $A_{1k}(i_k), A_{2k}(j_k)$, and vector x in the TT-format with cores $X_k(j_k)$.

Ensure: Vector $\boldsymbol{y} = Ax$ in the TT-format with cores $Y_k(i_k)$.

for $k = 1$ to d_2 **do**

$Z_k = \sum_{j_k}(A_{2k}(j_k) \otimes X_k(j_k))$.

end for

for $k = 1$ to d_1 **do**

if $k \neq d_1$ **then**

$Y_k(i_k) = A_{1k}(i_k)$.

else

$Y_{d_1}(i_{d_1}) = A_{1d_1}(i_{d_1})Z_1 \dots Z_{d_2}$.

end if

end for

The assembly of the factors Z_k can be realized by multiplication of a matrix of size $r^2 \times n$ by a matrix of size $n \times r^2$ (after a suitable reshaping). The computational complexity of assembly all factors is $\mathcal{O}(d_2 n r^4)$. Since Z_{d_2} is a column vector, evaluating $Y_{d_1}(i_{d_1})$ reduces to the computation of matrices Z_k and evaluating $d_2 + 1$ matrix-by-vector products giving an overall computational complexity of $\mathcal{O}(d_2 n r^4)$. Note that the first $d_1 - 1$ cores of the resulting tensor are precisely the first $d_1 - 1$ cores of the matrix $\boldsymbol{A}$. The d_1-th core is the d_1-th core of the matrix A post-multiplied by the matrices Y_k. For this formulation of the matrix-by-vector product, the TT-ranks of the product are exactly the TT-ranks of the original matrix $\boldsymbol{A}$. This is in sharp contrast with the matrix-by-vector product for matrices in TTM-format where the ranks of the product are at worst bound from above by the products of the corresponding ranks and generally the product has ranks larger than either of the factors.

III.2.4.3 TT VM-M Product

We formulate a similar algorithm for computing a matrix-matrix product when one matrix $\boldsymbol{A}$ is in the TTVM format and the other matrix $\boldsymbol{B}$ is in the TTM format and the number of column indices of $\boldsymbol{A}$ is equal to the number of row indices of $\boldsymbol{B}$. The product is given in the TTVM format.

Let multilevel matrix $\boldsymbol{A}$ have TTVM decomposition given by

$$\boldsymbol{A}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}) = A_{11}(i_1) \ldots A_{1d_1}(i_{d_1}) A_{21}(j_1) \ldots A_{2d_2}(j_{d_2}),$$

and multilevel matrix $\boldsymbol{B}$ have TTM decomposition given by

$$\boldsymbol{B}(\hat{i}_1, \ldots, \hat{i}_{d_2}; \hat{j}_1, \ldots, \hat{j}_{d_2}) = B_1(\hat{i}_1, \hat{j}_1) \ldots B_{d_2}(\hat{i}_{d_2}, \hat{j}_{d_2}).$$

The tensor product $\boldsymbol{A} \otimes \boldsymbol{B}$ can be represented in the TT format by:

$$\begin{aligned} & (\boldsymbol{A} \otimes \boldsymbol{B})(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}; \hat{i}_1, \ldots, \hat{i}_{d_2}; \hat{j}_1, \ldots, \hat{j}_{d_2}) \\ = \quad & A_{11}(i_1) \ldots A_{1d_1}(i_{d_1}) A_{21}(j_1) \ldots A_{2d_2}(j_{d_2}) B_1(\hat{i}_1, \hat{j}_1) \ldots B_{d_2}(\hat{i}_{d_2}, \hat{j}_{d_2}), \end{aligned}$$

which after rearrangement can be written in the TT format as

$$= A_{11}(i_1) \ldots A_{1d_1}(i_{d_1}) \left(A_{21}(j_1) \otimes B_1(\hat{i}_1, \hat{j}_1) \right) \ldots \left(A_{2d_2}(j_{d_2}) \otimes B_{d_2}(\hat{i}_{d_2}, \hat{j}_{d_2}) \right).$$

The matrix vector product $\boldsymbol{Y} = \boldsymbol{AB}$ can then be written as the contraction over the tensor product cores with $j_k = \hat{i}_k$ for each $k = 1, \ldots d_2$ which using III.2.4 results in a TTVM formatted matrix which can be written as:

$$\boldsymbol{Y}(i_1, \ldots, i_{d_1}; \hat{j}_1, \ldots, \hat{j}_{d_2}) = Y_{11}(i_1) \ldots Y_{1d_1}(i_{d_1}) Y_{21}(\hat{j}_1) \ldots Y_{2d_2}(\hat{j}_{d_2}),$$

where

$$Y_{lk}(\cdot) = \begin{cases} A_{1k}(\cdot), & \text{if } l = 1, \\ \sum_{j_k = \hat{i}_k} \left(A_{2k}(j_k) \otimes B_k(\hat{i}_k, \cdot) \right), & \text{otherwise.} \end{cases}$$

The assembly of the factors Y_{2k} can be realized by multiplication of a matrix of size $r^2 \times n$ by a matrix of size $n \times nr^2$ (after a suitable reshaping). The computational complexity of assembly all factors is $\mathcal{O}\left(d_2 n^2 r^4\right)$. Again, the first $d_1 - 1$ cores of the resulting tensor are precisely the first $d_1 - 1$ cores of the matrix $\boldsymbol{A}$ so this procedure does not result in increased ranks for these cores, though the last $d_2 - 1$ cores will

generally have ranks larger than the cores of the two factors. The details of the algorithm are summarized in Algorithm 5.

Algorithm 5 Tensor Train VM-by-M Product

Require: Matrix $\boldsymbol{A}$ in the TTVM-format with cores $A_{1k}(i_k), A_{2k}(j_k)$, and matrix $\boldsymbol{B}$ in the TTM-format with cores $B_k(i_k, j_k)$.

Ensure: Matrix $\boldsymbol{Y} = \boldsymbol{AB}$ in the TTVM-format with cores $Y_{1k}(i_k), Y_{2k}(j_k)$.

for $k = 1$ to d_1 **do**

$Y_{1k}(i_k) = A_{1k}(i_k)$.

end for

for $k = 1$ to d_2 **do**

$Y_{2k}(j_k) = \sum_{\alpha_k}(A_{2k}(\alpha_k) \otimes B_k(\alpha_k, j_k))$.

end for

III.2.4.4 TT VM-M-VM Product

We discuss algorithms for computing the matrix product $\boldsymbol{Y} = \boldsymbol{ABC}$ where $\boldsymbol{A}$ is a TTVM formatted matrix with d_1 row indices and d_2 column indices, $\boldsymbol{B}$ is a TTM formatted matrix with d_2 row and column indices, and $\boldsymbol{C}$ is a TTVM formatted matrix with d_2 row indices and d_3 column indices and all the corresponding mode sizes are the same so that the contractions are all well-defined.

Let multilevel matrix $\boldsymbol{A}$ have TTVM decomposition given by

$$\boldsymbol{A}(i_1, \ldots, i_{d_1}; j_1, \ldots, j_{d_2}) = A_{11}(i_1) \ldots A_{1d_1}(i_{d_1}) A_{21}(j_1) \ldots A_{2d_2}(j_{d_2}),$$

multilevel matrix $\boldsymbol{B}$ have TTM decomposition given by

$$\boldsymbol{B}(\hat{i}_1,\ldots,\hat{i}_{d_2};\hat{j}_1,\ldots,\hat{j}_{d_2}) = B_1(\hat{i}_1,\hat{j}_1)\ldots B_{d_2}(\hat{i}_{d_2},\hat{j}_{d_2}),$$

multilevel matrix $\boldsymbol{C}$ have TTVM decomposition given by

$$\boldsymbol{C}(\hat{\hat{i}}_1,\ldots,\hat{\hat{i}}_{d_2};\hat{\hat{j}}_1,\ldots,\hat{\hat{j}}_{d_3}) = A_{11}(\hat{\hat{i}}_1)\ldots A_{1d_2}(\hat{\hat{i}}_{d_2})A_{21}(\hat{\hat{j}}_1)\ldots A_{2d_3}(\hat{\hat{j}}_{d_3}).$$

The tensor product $\boldsymbol{A}\otimes\boldsymbol{B}\otimes\boldsymbol{C}$ can be represented in the TT format by:

$$\begin{aligned}
&(\boldsymbol{A}\otimes\boldsymbol{B}\otimes\boldsymbol{C})(i_1,\ldots,i_{d_1};j_1,\ldots,j_{d_2};\hat{i}_1,\ldots,\hat{i}_{d_2};\hat{j}_1,\ldots,\hat{j}_{d_2};\hat{\hat{i}}_1,\ldots,\hat{\hat{i}}_{d_2};\hat{\hat{j}}_1,\ldots,\hat{\hat{j}}_{d_3})\\
&= A_{11}(i_1)\ldots A_{1d_1}(i_{d_1})A_{21}(j_1)\ldots A_{2d_2}(j_{d_2})B_1(\hat{i}_1,\hat{j}_1)\ldots B_{d_2}(\hat{i}_{d_2},\hat{j}_{d_2})C_{11}(\hat{\hat{i}}_1)\times\\
&\quad\ldots\times C_{1d_2}(\hat{\hat{i}}_{d_2})C_{21}(\hat{\hat{j}}_1)\ldots A_{2d_3}(\hat{\hat{j}}_{d_3}),
\end{aligned}$$

which after rearrangement can be written in the TT format as

$$\begin{aligned}
= \quad & A_{11}(i_1)\ldots A_{1d_1}(i_{d_1})\left(A_{21}(j_1)\otimes B_1(\hat{i}_1,\hat{j}_1)\otimes C_{11}(\hat{\hat{i}}_1)\right)\times\\
&\ldots\times\left(A_{2d_2}(j_{d_2})\otimes B_{d_2}(\hat{i}_{d_2},\hat{j}_{d_2})\otimes C_{1d_2}(\hat{\hat{i}}_{d_2})\right)C_{21}(\hat{\hat{j}}_1)\ldots A_{2d_3}(\hat{\hat{j}}_{d_3}).
\end{aligned}$$

The matrix vector product $\boldsymbol{Y}=\boldsymbol{ABC}$ can then be written as the contraction over the tensor product cores with $j_k=\hat{i}_k, \hat{j}_l=\hat{\hat{i}}_l$ for each $k=1,\ldots d_2, l=1,\ldots,d_3$ which using III.2.4 results in a TTVM formatted matrix which can be written as:

$$\boldsymbol{Y}(i_1,\ldots,i_{d_1};\hat{\hat{j}}_1,\ldots,\hat{\hat{j}}_{d_2}) = Y_{11}(i_1)\ldots Y_{1d_1}(i_{d_1})Y_{21}(\hat{\hat{j}}_1)\ldots Y_{2d_3}(\hat{\hat{j}}_{d_3}),$$

where

$$Y_{lk}(\cdot)=\begin{cases} A_{1k}(\cdot), & \text{if } l=1,\\ C_{2k}(\cdot), & \text{if } l=2, k\neq 1\\ Z_1\ldots Z_{d_2}C_{21}(\cdot), & \text{otherwise,}\end{cases}$$

where the matrices Z_k are given by:

$$Z_k = \sum_{j_k=\hat{i}_k} \sum_{\hat{j}_k=\hat{\hat{i}}_k} \left(A_{2k}(j_k) \otimes B_k(\hat{i}_k, \hat{j}_k) \otimes C_{1k}(\hat{\hat{i}}_k) \right).$$

The assembly of the factors Z_k can be realized by multiplication of a matrix of size $r^2 \times n$ by a matrix of size $n \times nr^2$ and then multiplication of a matrix of size $r^4 \times n$ by a matrix of size $n \times r^2$ (after suitable reshapings). The computational complexity of assembly all factors is $\mathcal{O}\left(d_2 n^2 r^4 + d_2 n r^6\right)$. The construction of the factor $Y_{21}(\cdot)$ requires d_2 matrix multiplications of matrices of size $r^3 \times r^3$ by a matrix of size $r^3 \times nr$ resulting in a computational complexity of $\mathcal{O}\left(d_2 n r^7\right)$. The overall computational complexity is therefore $\mathcal{O}\left(d_2 n^2 r^4 + d_2 n r^7\right)$. Again, the first d_1 cores of the resulting tensor are precisely the first d_1 cores of the matrix $\boldsymbol{A}$ while the last $d_3 - 1$ cores are precisely the last $d_3 - 1$ cores of the matrix $\boldsymbol{C}$ so this procedure does not result in increased ranks. The details of the algorithm are summarized in Algorithm 6.

In practice, we find that explicitly assembling the factors Z_k is unnecessary and very expensive in terms of storage when the ranks of $\boldsymbol{A}$, $\boldsymbol{B}$, or $\boldsymbol{C}$ are large. Since the factors Z_k are multiplied together, we find it more efficient to only explicitly represent one row of Z_k at a time during the matrix multiplication. The does not affect the estimates of the computational complexity in terms of the number of arithmetic operations performed but it significantly reduces the storage required to compute the product.

Algorithm 6 Tensor Train VM-M-VM Product

Require: Matrix $\boldsymbol{A}$ with d_1 row indices and d_2 column indices in the TTVM-format with cores $A_{1k}(i_k), A_{2k}(j_k)$, matrix $\boldsymbol{B}$ with d_2 row and column indices in the TTM-format with cores $B_k(i_k, j_k)$, and a matrix $\boldsymbol{C}$ with d_2 row indices and d_3 column indices in the TTVM-format with cores $C_{1k}(i_k), C_{2k}(j_k)$.

Ensure: Matrix $\boldsymbol{Y} = \boldsymbol{ABC}$ in the TTVM-format with cores $Y_{1k}(i_k), Y_{2k}(j_k)$.

for $k = 1$ to d_1 **do**

$Y_{1k}(i_k) = A_{1k}(i_k)$.

end for

$Y_{21}(j_1) = C_{21}(j_1)$.

for $k = 1$ to d_2 **do**

$Y_{21}(j_1) = \left(\sum_{\alpha_k, \beta_k} (A_{2k}(\alpha_k) \otimes B_k(\alpha_k, \beta_k) \otimes C_{1k}(\beta_k))\right) Y_{21}(j_1)$.

end for

for $k = 2$ to d_3 **do**

$Y_{2k}(j_k) = C_{2k}(j_k)$.

end for

III.3 Diagonalization of TT-VM Formatted Matrices

In this section we discuss the diagonalization of symmetric positive semi-definite matrices approximated in the TTVM format. The first section describes an approach based on the Lanczos Method for computing the dominant eigenvalues and corresponding eigenvectors of a Hermitian matrix which may be done cheaply whenever the compression ranks of the matrix are small. The second section observes that whenever the SVD is a good approximation of the diagonalization of the matrix one should compute that since the SVD of a TTVM formatted matrix may be computed cheaply and accurately using the TT rounding procedure.

III.3.1 TT-Lanczos Diagonalization

The Lanczos iteration is a powerful method for quickly estimating dominant eigenvalues and the corresponding eigenvectors of Hermitian matrices, for instance, finding the most controllable/observable modes of a linear system from the controllability/observability Gramian. Given a Hermitian matrix H and a predetermined number of iterations m, it performs an efficient Krylov iteration to construct a change of basis transforming H into a tri-diagonal matrix T_{mm} that can be diagonalized efficiently, for example, using the QR algorithm. Once the eigensystem of T_{mm} is known, it is straightforward to reconstruct the dominant eigenvectors of H, see for

example [Saa03]. The Lanczos iteration is attractive computationally since the only large-scale linear operation is matrix-by-vector multiplication.

We use a version of the Lanczos iteration described in [Hac+12] for low-rank tensor formats incorporating the Vectorized Matrix-by-Vector product introduced in the previous section. A key challenge of such methods is controlling the rank through the orthogonalization step, as adding or subtracting tensors generally increases rank. The more iterations, the larger the ranks. We use the TT-rounding procedure to re-compress the tensor after each orthogonalization step to help control this growth. The formal description of the procedure is listed in Algorithm 2. If the Hermitian matrix H were instead given in the (Q)TT-Matrix format, the standard (Q)TT Matrix-by-Vector product could be substituted in the proposed algorithm but the TT-rounding procedure should also be performed after every matrix-by-vector product to help control the TT-ranks.

While it can be proved that with exact arithmetic, the Lanczos iteration constructs a unitary transformation and that the eigenvalues/vectors computed are good approximations to those of the original matrix, it is well understood that when using floating point arithmetic the orthogonality may be quickly lost. An overzealous use of the TT-rounding procedure has the same effect and we observe this in our numerical experiments.

Algorithm 7 (Q)TT-Lanczos Iteration

Require: Matrix $\boldsymbol{A}$ in the Vectorized-(Q)TT-Matrix format and number of iterations m.

Ensure: Sequences of values $\{\alpha_k\}_{k=1}^m$, $\{\beta_k\}_{k=1}^m$ and Lanczos vectors $\{\boldsymbol{v}_k\}_{k=1}^m$ in the (Q)TT-format.

$\boldsymbol{v_1}$ = random (Q)TT-vector with norm 1.

$\boldsymbol{v_0}$ = zero vector.

$\beta_1 = 0$.

for $k = 1$ to m **do**

$\boldsymbol{w_j} = A\boldsymbol{v_j}$, {Matrix-by-Vector Product}

$\alpha_j = \boldsymbol{w_j} \cdot \boldsymbol{v_j}$,

$\boldsymbol{w_j} = \boldsymbol{w_j} - \alpha_j \boldsymbol{v_j} - \beta_j \boldsymbol{v_{j-1}}$, {Orthogonalization}

TT-Round $\boldsymbol{w_j}$ to accuracy ε,

$\beta_{j+1} = ||\boldsymbol{w_j}||$,

$\boldsymbol{v_{j+1}} = \boldsymbol{w}_j/\beta_{j+1}$

end for

III.3.2 TTVM-SVD Diagonalization

We first remark that computing the SVD of a symmetric, positive semidefinite matrix automatically reveals its nonzero eigenvalues and associated eigenvectors since powers and roots of a matrix can be computed in a straightforward manner once it is diagonalized.

Proposition III.3.1. *Let P be a positive semidefinite matrix with singular value decomposition $P = U\Sigma V^T$. If v is a column vector of V (right-singular vector of P) then it is also an eigenvector of P whose associated singular value is also its associated eigenvalue. Furthermore, v is also a column of U.*

Proof. Since P is positive semidefinite symmetric, there exists an orthonormal basis for $\mathbb{R}^n$ of eigenvectors $\{w_k\}_{k=1}^n$ of P with associated nonnegative eigenvalues λ_k. Recall that the nonzero singular values of P are positive roots of the nonzero eigenvalues of P^*P and that the right-singular vectors of P (the columns of V^T) are eigenvectors of P^*P. Suppose v is a right-singular vector of P and therefore an eigenvector of P^*P with associated eigenvalue μ. Then v can be expressed as a linear combination of the basis elements w_k: $v = \sum_{k=1}^n \alpha_k w_k$, where each $\alpha_k \in \mathbb{R}$. Computing the matrix vector product in terms of the expansion in the basis elements:

$$\begin{aligned} P^*Pv &= \mu v, \\ \sum_{k=1}^{n} \alpha_k \lambda_k^2 w_k &= \mu \sum_{k=1}^{n} \alpha_k w_k. \end{aligned}$$

Computing the inner product of each side with each element of the orthonormal basis

$\{w_k\}_{k=1}^n$ yields:

$$\alpha_k \lambda_k^2 = \mu \alpha_k, \quad k = 1, \ldots, n.$$

For each k, either $\alpha_k = 0$ or $\mu = \lambda_k^2$, but at least one $\alpha_k \neq 0$ since otherwise $v = 0$ and it is not an eigenvector of P^*P as was assumed. Instead, there is some subset $K \subset \{1, \ldots, n\}$, where the $\alpha_{\hat{k}} \neq 0$ for each $\hat{k} \in K$. $\lambda_{\hat{k}} = +\sqrt{\mu}$ for every $\hat{k}$ in this index set. This means that for each $\hat{k} \in K$, the associated eigenvectors $v_{\hat{k}}$ all belong to the same eigenspace of P. Hence, dropping all the terms of the expansion whose coefficients $\alpha_k = 0$, v can be written as a linear combination of elements all belonging to the same eigenspace and is therefore itself an eigenvector of P.

To see that the associated singular value is also an eigenvalue suppose v_k is the k-th column vector of V corresponding to a nonzero singular value and is therefore also an eigenvector with associated eigenvalue ν:

$$\begin{aligned} Pv_k &= \nu v_k \\ U\Sigma V^T v_k &= \nu v_k \\ \sigma_k u_k &= \nu v_k \end{aligned}$$

where u_k is the k-th column vector of U and σ_k is the k-th singular value of P. Since the above equation shows that they differ by a scalar multiple, they are collinear. Since the singular vectors u_k and v_k are assumed to have unit norm, $\sigma_k > 0$ since it is a nonzero singular value, and $\nu \geq 0$ since P is positive semidefinite, $u_k = v_k$, and therefore $\nu = \sigma_k$. Therefore, σ_k is the associated eigenvalue of v_k and $u_k = v_k$ so that v_k is also the k-th column of U. □

The proposition when applied to TTVM formatted positive semidefinite matrices says that by computing the TTVM-SVD, we are also computing a diagonalization of the matrix.

III.4 Matrix Powers

The controls applications described in the following chapters require easily computable matrix inverses and square roots. While in general, even for positive definite matrices, some of these objects may not be unique, simple formulas can be stated and computed in terms of the diagonalization of the matrix in question. We first state a formula for computing the (positive) matrix power.

Definition III.4.1. *Suppose W is positive semidefinite and has rank $\hat{r}_c$ and eigenvectors $\{w_k\}_{k=1}^{\hat{r}_c}$ with associated eigenvalues $\{\lambda_k\}_{k=1}^{\hat{r}_c}$. For $p > 0$, the p-th power of W is given by the formula:*

$$(W)^p = \sum_{k=1}^{\hat{r}_c} (\lambda_k)^p (w_k \otimes w_k). \tag{III.4.1}$$

Inverse powers will not be defined if W is singular. At best we may consider the restriction of positive semidefinite W to the orthogonal complement of its nullspace, $W|_{\mathcal{N}_W^\perp}$. We may also describe some of its inverse powers in terms of its eigendecomposition.

Definition III.4.2. *Suppose W is positive semidefinite and has rank $\hat{r}_c$ and eigenvectors $\{w_k\}_{k=1}^{\hat{r}_c}$ with associated eigenvalues $\{\lambda_k\}_{k=1}^{\hat{r}_c}$. For $p \geq 0$, the p-th inverse power*

of $W|_{\mathcal{N}_W^\perp}$ *is given by the formula:*

$$\left(W|_{\mathcal{N}_W^\perp}\right)^{-p} = \sum_{k=1}^{\hat{r}_c} (\lambda_k)^{-p} (w_k \otimes w_k). \tag{III.4.2}$$

Since the Lyapunov Equation solver described previously outputs gramians in the TTVM format, their square roots and inverses are easily computable in the TTVM format assuming the gramian has been computed with high enough accuracy to be symmetric. The process requires a computation of the TTVM-SVD, and calculating $\hat{r}_c$ powers of positive numbers. The details of the procedure for computing the p-th power are summarized in Algorithm 8. The generalization to inverse powers on the restricted domain is straightforward.

Algorithm 8 Tensor Train Vectorized Matrix Power

Require: Positive definite matrix $\boldsymbol{X}$ in the TTVM-format with cores $X_{1k}(i_k), X_{2k}(j_k)$ and with ranks $r_{1,1}, \ldots, r_{1,d_1-1}, \hat{r}_c, r_{2,1}, \ldots, r_{2,d_2-1}$, implementation of the qr_lr$(\cdot)$ algorithm as in [Ose11], the qr_lr$(\cdot)$ algorithm, and the singular value decomposition, svd$(\cdot)$.

Ensure: Matrix $\boldsymbol{Y}$ in the TTVM-format with cores $Y_{1k}(i_k), Y_{2k}(j_k)$ such that $\boldsymbol{Y} = \boldsymbol{X}^2$.

$[\boldsymbol{Q}_2^T, \boldsymbol{R}_V^T] = \text{qr\_rl}(\boldsymbol{X})$,

$[\boldsymbol{Q}_1, \boldsymbol{R}_U] = \text{qr\_lr}(\boldsymbol{X})$,

$[U_P, D, V_P^T] = \text{svd}(\boldsymbol{R}_U \boldsymbol{R}_V^T)$,

$Y_P = U_P D^2 V_P^T$

for $k = 1$ to d **do**

$\hat{Y}_{1,k}(i_k) = Q_{1,k}(i_k)$,

$\hat{Y}_{2,k}(j_k) = Q_{2,k}(j_k)$,

end for

$\hat{Y}_{1,d_1}(i_{d_1}) = Y_{1,d_1}(i_{d_1}) Y_P$.

Chapter IV

hp-DG-QTT Solver for the Chemical Master Equation

This chapter describes the *hp*-DG-QTT approach to the numerical solution of the Chemical Master Equation. It combines three main technologies. First, employ the Finite State Projection (FSP) to reduce the formally countably infinite state-space of the Markov Process to one with finitely many states. The FSP reduces the CME to a large but structured system of ODEs. Second, use the Quantized Tensor Train format to express all vectors and matrices involved. We compute upper bounds on the QTT ranks of the CME operator. While this work does not provide rank bounds on the solution of the CME we refer the reader to the related work of Kazeev and Schwab [KS13] which computes rank bounds for the *stationary* solutions of reaction networks with Deficiency Zero in the sense of Feinberg [Fei79]. Lastly, we employ the

hp-Discontinous Galerkin time discretization to convert the system of ODEs into a sequence of QTT structured linear systems which are solved at each time step using the DMRG based solver. The resulting numerical method is adaptive both in time and in the QTT compression. In the best cases, it demonstrates *linear* complexity scaling in the number of chemical species.

IV.1 CME Truncation

A "well-stirred" solution of d chemically reacting molecules in thermal equilibrium can be described by a jump Markov process, where for each fixed time $t \geq 0$, $X(t) \in \mathbb{Z}^d_{\geq 0}$ is a random vector of nonnegative integers with each component representing the number of molecules of one chemical species present in the system. In [Kam92] and the references therein, it is shown that, given an initial condition $X(0) \in \mathbb{Z}^d_{\geq 0}$, the corresponding probability density function (PDF) $\mathbb{Z}^d_{\geq 0} \times [0, \infty) \ni (\underline{x}, t) \mapsto \boldsymbol{p}_{\underline{x}}(t)$ of the process solves the Chemical Master Equation (CME):

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\boldsymbol{p}_{\underline{x}}(t) = -\boldsymbol{p}_{\underline{x}}(t)\sum_{s=1}^{R}\omega^s(\underline{x}) + \sum_{s=1}^{R}\boldsymbol{p}_{\underline{x}-\underline{\eta}^s}(t)\,\omega^s\big(\underline{x}-\underline{\eta}^s\big) \tag{IV.1.1}$$

where R is the number of reactions in the system, $\underline{\eta}^s \in \mathbb{Z}^d$ and ω^s are the stoichiometric vector and propensity function of the sth reaction, respectively. The CME is a system of coupled linear ordinary differential equations with one equation per state $X(t) = \underline{x} \in \mathbb{Z}^d_{\geq 0}$.

Munsky and Khammash [MK06] rewrote the right-hand side of the CME (IV.1.1) as the action of a linear operator $\boldsymbol{A}$ on the probability density at the current time:

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\boldsymbol{p}(t) = \boldsymbol{A}\boldsymbol{p}(t) \tag{IV.1.2}$$

Throughout this paper, we refer to $\boldsymbol{A}$ as the *CME operator*.

Hegland and Garcke introduced an explicit representation of the CME operator as sums and compositions of a few elementary linear operators [HG11]: let $\boldsymbol{S}_{\underline{\eta}}$ be the spatial shift of a probability density by a vector $\underline{\eta} \in \mathbb{Z}^d$ and let $\boldsymbol{M}_{\omega}$ be multiplication by a real-valued function ω:

$$\Big(\boldsymbol{S}_{\underline{\eta}}\,\boldsymbol{p}\Big)_{\underline{x}} = \boldsymbol{p}_{\underline{x}-\underline{\eta}}, \quad \big(\boldsymbol{M}_{\omega}\,\boldsymbol{p}\big)_{\underline{x}} = \omega(\underline{x}) \cdot \boldsymbol{p}_{\underline{x}}\,.$$

Then the CME operator can be written as follows, with Id denoting the identity operator:

$$\boldsymbol{A} = \sum_{s=1}^{R} \Big(\boldsymbol{S}_{\underline{\eta}^s} - \mathrm{Id}\Big) \circ \boldsymbol{M}_{\omega^s}\,. \tag{IV.1.3}$$

To simplify the exposition, we assume that all propensity functions are *rank-one separable*, i.e. they are of the form

$$\omega^s(\underline{x}) = \prod_{k=1}^{d} \omega_k^s(x_k)\,, \quad \underline{x} \in \mathbb{Z}^d_{\geq 0}, \tag{IV.1.4}$$

for $1 \leq s \leq R$, where each $\omega_k^s(x_k)$ is a nonnegative function in the single variable x_k. Considering rank-one separable propensity functions is sufficient for all elementary reactions which occur as building blocks in more complicated reaction kinetics.

The CME (IV.1.2) is posed on the (countably) infinite space $\mathbb{Z}^d_{\geq 0}$ of states. In this form, the CME (IV.1.1) is an infinite-dimensional coupled evolution problem

which necessitates truncation prior to numerical discretization. In the case of a particular class of monomolecular reactions, Jahnke and Huisinga were able to construct an explicit solution in terms of convolutions of products of Poisson and multinomial distributions [JH07].

In order to address more complex systems computationally, Munsky and Khammash proposed the Finite State Projection Algorithm (FSP) [MK06] which seeks to truncate the countably infinite dimensional space $\mathbb{Z}^d_{\geq 0}$ of states of the process to its finite subset

$$\Omega^{\underline{n}} = \left\{ \underline{x} \in \mathbb{Z}^d_{\geq 0} : 0 \leq x_k < n_k \quad \text{for} \quad 1 \leq k \leq d \right\} \subset \mathbb{Z}^d_{\geq 0} , \tag{IV.1.5}$$

associated with a multi-index $\underline{n} = (n_1, n_2, ..., n_d) \in \mathbb{N}^d$, so that the dynamics over $\Omega^{\underline{n}}$ are close to those of the original system.

Theorem IV.1.1 (Finite State Projection, Theorem 2.2 in [MK06])**.** *Consider a Markov process with state space $\mathbb{Z}^d_{\geq 0}$ whose probability density evolves according to the initial value ODE: given an initial state $\boldsymbol{p}_0 \in [0,1]^{\mathbb{Z}^d_{\geq 0}}$, find $\boldsymbol{p}(t) \in [0,1]^{\mathbb{Z}^d_{\geq 0}}$ such that*

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{p}(t) = \boldsymbol{A}\boldsymbol{p}(t) \quad 0 \leq t < \infty , \qquad \boldsymbol{p}(0) = \boldsymbol{p}_0$$

where the operator $\boldsymbol{A} : [0,1]^{\mathbb{Z}^d_{\geq 0}} \mapsto [0,1]^{\mathbb{Z}^d_{\geq 0}}$ can be interpreted as a semi-infinite matrix.

Let $\boldsymbol{A}^{\underline{n}}$ denote the restriction of $\boldsymbol{A}$ to $\Omega^{\underline{n}}$ defined by (IV.1.5) *and assume that $\boldsymbol{p}_0$ is supported in $\Omega^{\underline{n}}$, i.e. that $\boldsymbol{p}_0 = 0$ in $\mathbb{Z}^d_{\geq 0} \backslash \Omega^{\underline{n}}$. Denote by $\hat{\boldsymbol{p}}.(\cdot) \in [0,1]^{\Omega^{\underline{n}}}$ the*

solution of the truncated system with dynamics given by the linear ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\hat{\boldsymbol{p}}(t) = \boldsymbol{A}^{\underline{n}}\hat{\boldsymbol{p}}(t)\ , \qquad 0 \le t < \infty \tag{IV.1.6}$$

with initial condition $\hat{\boldsymbol{p}}_{\underline{x}}(0) = \boldsymbol{p}_{\underline{x}}(0) = \boldsymbol{p}_0(\underline{x})$. *If for some* $\varepsilon > 0$ *and* $\tau \ge 0$

$$\sum_{\underline{x}\in\Omega^{\underline{n}}} \hat{\boldsymbol{p}}_{\underline{x}}(\tau) \ge 1 - \varepsilon \tag{IV.1.7}$$

then

$$\hat{\boldsymbol{p}}_{\underline{x}}(\tau) \le \boldsymbol{p}_{\underline{x}}(\tau) \le \hat{\boldsymbol{p}}_{\underline{x}}(\tau) + \varepsilon \tag{IV.1.8}$$

for every $\underline{x} \in \Omega^{\underline{n}}$ *and*

$$\left\| \hat{\boldsymbol{p}}(\tau) - \boldsymbol{p}(\tau)\,|_{\Omega^{\underline{n}}} \right\|_1 \le \varepsilon. \tag{IV.1.9}$$

Assuming that a truncation satisfying (IV.1.7) can be found, then (IV.1.9) gives an explicit certificate of the accuracy of the approximate solution.

In practice, the truncation satisfying a given error tolerance may still require a very large number of states: the dimension of the FSP vector $\hat{\boldsymbol{p}}$ equals $\mathrm{card}(\Omega^{\underline{n}}) = \prod_{k=1}^{d} n_k$ rendering a direct numerical solution of even the projected equation (IV.1.6) infeasible in many cases. The remainder of the chapter presents a novel approach for the numerical solution of such FSP truncated systems that retain large numbers of states. For notational convenience, we drop the superscripts $\underline{n}$ and the hat from $\hat{\boldsymbol{p}}$ indicating the FSP since we will only consider systems which have already been truncated. Similarly, we now use the shift and multiplication operators in (IV.1.3) restricted to the truncated state space without change of notation.

Assuming that a FSP has been performed, we henceforth treat $\boldsymbol{p}_{\underline{x}}(t)$ as a d-dimensional $n_1 \times \ldots \times n_d$-vector, i.e. as an array indexed by $\Omega^{\underline{n}}$ which we identify with *ordered* d-tuples of indices $i_k \in \{0, 1, 2, \ldots, n_k - 1\}$, where k ranges from 1 to d. Each dimension k (alternatively referred to as a *mode* or *level*) has a corresponding *mode size* n_k, that is, the number of values which the index for that dimension can take. For our chemically reacting system, $n_k - 1$ corresponds to the maximum number of copies of the kth species that is considered.

For the same ordering of $\underline{x}$, consider the corresponding d-dimensional $n_1 \times \ldots \times n_d$-vectors $\boldsymbol{\omega}^s$, $1 \leq s \leq R$, containing the values of the propensities on $\Omega^{\underline{n}}$ to which we shall refer as *propensity vectors*:

$$\boldsymbol{\omega}^s{}_{\underline{x}} = \omega^s(\underline{x}) \quad \text{for all} \quad \underline{x} \in \Omega^{\underline{n}} . \tag{IV.1.10}$$

Within the projected CME (IV.1.6), the operators corresponding to weighting by the propensity functions, involved in (IV.1.3), are finite matrices: $\boldsymbol{M}_{\omega^s} = \operatorname{diag} \boldsymbol{\omega}^s$. Then, under the rank one separability assumption (IV.1.4), with $(\boldsymbol{\omega}_k^s)_{x_k} = \omega_k^s(x_k)$ for $0 \leq x_k \leq n_k$, $1 \leq k \leq d$ there holds

$$\boldsymbol{\omega}^s = \boldsymbol{\omega}_1^s \otimes \ldots \otimes \boldsymbol{\omega}_d^s , \qquad 1 \leq s \leq R . \tag{IV.1.11}$$

IV.2 Representation of the CME in QTT Format

In this section we consider the Finite State Projection of the CME, projected onto a rectangular collection of states as described previously. We take the finest

possible quantization and representing both the PDF $\boldsymbol{p}$ of the truncated model and the CME operator $\boldsymbol{A}$ from (IV.1.3) in the QTT format. We first establish upper bounds on the QTT ranks of $\boldsymbol{A}$.

Theorem IV.2.1. *Consider the projected CME operator $\boldsymbol{A}$ defined by* (IV.1.3)*. Assume that for every $s = 1, \ldots, R$ and $k = 1, \ldots, d$ the one-dimensional vector $\boldsymbol{\omega}_k^s$ from* (IV.1.10)–(IV.1.11) *is given in a QTT decomposition of ranks bounded by r_k^s; and that $\eta_k^s = 0$ implies $r_k^s = 1$. Then the CME operator $\boldsymbol{A}$ admits a QTT decomposition of ranks*

$$q_1, \ldots, q_1, \hat{q}_1, q_2, \ldots, q_2, \hat{q}_2, \ldots, \ldots, \hat{q}_{d-1}, q_d, \ldots, q_d$$

with $\hat{q}_k = 2R$ for $1 \leq k \leq d-1$ and

$$q_k = \sum_{\substack{s=1,\ldots,R: \\ \eta_k^s = 0}} 2 \quad + \sum_{\substack{s=1,\ldots,R: \\ \eta_k^s \neq 0}} 3 r_k^s$$

for $1 \leq k \leq d$.

Proof. The specific details of the proof is the work of Vladimir Kazeev and is given in the Supplementary Text of [Kaz+14]. The sketch of the proof is to establish bounds on each of the matrices in the explicit representation of the CME operator (IV.1.3), and to use the sum and product rules for TT-formatted arrays to establish an upper bound on the ranks of $\boldsymbol{A}$. □

A simpler (but cruder) upper bound on the QTT ranks of the CME operator is $3 \cdot R \cdot r$, where $r = \max_{s,k} r_k^s$. Note that when all the reactions are elementary, the

propensities can be expressed as polynomials in which case if p is the order of the highest order polynomial then, $r = p + 1$, see [Gra10b, Corollary 13] and [Ose13b, Theorem 6]. We emphasize, however, that the bound IV.2.1 holds equally well when the reactions are not elementary. In particular, our numerical experiments (e.g. the toggle switch example) show that the QTT ranks of vectors corresponding to rational propensity functions are also small.

At various points of this chapter, we assume that the propensities are separable in each species. We emphasize that this is only to simplify the exposition of the algorithmic details and relaxation of this condition poses no mathematical difficulties. The QTT rank bound for $\boldsymbol{A}$ in Theorem IV.2.1, does not rely on any assumption of separability of the propensity vectors.

IV.2.0.1 Structure of the CME operator in the transposed QTT format

This subject of this section is the work of Vladimir Kazeev and is included to provide background for one of the numerical experiments. The interested reader should see [Kaz+14] for details. Similarly to Theorem IV.2.1, we can bound the ranks of the CME operator in the transposed QTT format relying on the ordering (III.1.2) of "virtual" indices.

Theorem IV.2.2. *Consider the projected CME operator* $\boldsymbol{A}$ *defined by* (IV.1.3). *Assume that for every* $s = 1, \ldots, R$ *and* $k = 1, \ldots, d$ *the one-dimensional vector* $\boldsymbol{\omega}_k^s$ *from* (IV.1.10)–(IV.1.11) *is given in a QTT decomposition of ranks bounded by* r_k^s*;*

and that $\eta_k^s = 0$ *implies* $r_k^s = 1$. *Then the CME operator* $\boldsymbol{A}$ *admits a QT3 decomposition of ranks bounded by*

$$\sum_{s=1}^{R} \left(1 + \prod_{k \in \mathcal{K}^s} 2\right) \left(\prod_{k \in \mathcal{K}^s} r_k^s\right),$$

where $\mathcal{K}^s = \{k \in \mathbb{N} : 1 \leq k \leq d \text{ and } \eta_k^s \neq 0\}$.

Proof. The proof is the work of Vladimir Kazeev and is given in the Supplementary Text of [Kaz+14] □

We observe in the enzymatic futile cycle example below that the QT3 ranks of the CME operator may be significantly lower than the bound of Theorem IV.2.1.

Time Integration of the CME: hp-Discontinuous Galerkin Discretization

Consider the truncated CME (IV.1.6) with a state space $X = \mathbb{R}^{n_1 \times \ldots \times n_d}$ on a finite time interval $J = (0, T)$. The Initial Value Problem (IVP) with initial data $\boldsymbol{p}_0 \in X$ is find a continuously differentiable function $\boldsymbol{p} : \overline{J} \to X$ such that

$$\begin{cases} \dot{\boldsymbol{p}}(t) &= \boldsymbol{A} \cdot \boldsymbol{p}(t) \quad \text{for} \quad t \in \overline{J}, \\ \boldsymbol{p}(0) &= \boldsymbol{p}_0 \,. \end{cases} \tag{IV.2.1}$$

The theoretical solution of the IVP is given by the *matrix exponential*, $\boldsymbol{p}(t) = \exp(t\boldsymbol{A}) \cdot \boldsymbol{p}_0$ for $t \in \overline{J}$, but computing the numerical value at each time by traditional methods suffers from the curse of dimensionality.

We use *hp*-discontinous Galerkin (*hp*-DG) discretization in time as the time-stepping scheme [SS00] to solve the truncated ODE. Given a time mesh, the *hp*-DG method finds an approximate solution to the initial value problem that is a polynomial when restricted to each subinterval of the time mesh and possibly discontinuous at each mesh point. This method allows adaptation of the size of each time step (*h*-adaptation), allowing good resolution of fast transients, as well as the order of the approximating polynomial on each time step (*p*-adaptation), or both simultaneously (*hp*-adaptation). For linear finite-dimensional ODE initial value problems like the projected CME, the solution is time-analytic and the *hp*-DG approach achieves *exponential rates of convergence* to the classical solution with respect to the number of temporal degrees of freedom [SS00]. Assuming the problem has been expressed in QTT format, *hp*-DG discretization in time reduces the projected CME evolution problem to a sequence of systems of QTT structured linear equations that must be solved at each time step [KRS12]. Our computational method then exploits an algorithm available for solving linear systems in this format that is based on Density Matrix Renormalization Group (DMRG) methods from quantum chemistry.

To discuss the tensor structure of the *hp*-DG-QTT approach, we revisit the following definitions from [Kaz+14].

Denote the space of polynomials of degree at most ρ and with coefficients from X defined on a finite interval I by $\mathcal{P}^{\rho}(I, X)$. Let $\mathcal{M} = \{J_m\}_{m=1}^{M}$ be a partition of the time interval J into subintervals $J_m = (t_{m-1}, t_m)$, $1 \leq m \leq M$, and $\underline{\rho} \in \mathbb{N}_{\geq 0}^{M}$.

Consider the space

$$\mathcal{P}^{\underline{\rho}}(\mathcal{M}, X) = \left\{ \boldsymbol{p} : J \to X : \ \boldsymbol{p}|_{J_m} \in \mathcal{P}^{\rho_m}(J_m, X) \quad \text{for} \quad 1 \le m \le M \right\}$$

of functions, which are polynomials of degree at most ρ_m on J_m for all m. For all $\boldsymbol{q} \in \mathcal{P}^{\underline{\rho}}(\mathcal{M}, X)$ let $\boldsymbol{q}_m^+ = \lim_{t \downarrow t_m} \boldsymbol{q}(t)$ and $\boldsymbol{q}_m^- = \lim_{t \uparrow t_m} \boldsymbol{q}(t)$ for all feasible m.

Definition IV.2.3. *The hp-DG formulation of* (IV.2.1)*, corresponding to the partition $\mathcal{M}$ of the time interval and the vector $\underline{\rho}$ of polynomial degrees, reads as follows: find $\boldsymbol{p} \in \mathcal{P}^{\underline{\rho}}(\mathcal{M}, X)$ such that*

$$\sum_{m=1}^{M} \int_{J_m} \langle \dot{\boldsymbol{p}} - \boldsymbol{A}\boldsymbol{p},\, \boldsymbol{q} \rangle \, \mathrm{d}t + \sum_{m=1}^{M} \left\langle \boldsymbol{p}_{m-1}^+ - \boldsymbol{p}_{m-1}^- \,,\, \boldsymbol{q}_{m-1}^+ \right\rangle = 0 \tag{IV.2.2}$$

for all $\boldsymbol{q} \in \mathcal{P}^{\underline{\rho}}(\mathcal{M}, X)$, where $\boldsymbol{p}_0^-$ stands for the initial value $\boldsymbol{p}_0$.

Equation (IV.2.2) can be understood as a time-stepping method: if for all m from 1 up to $\ell - 1$ the polynomial $\boldsymbol{p}|_{J_m} \in \mathcal{P}^{\rho_m}(J_m, X)$ is known through $\rho_m + 1$ coefficients from X, then $\boldsymbol{p}|_{J_\ell} \in \mathcal{P}^{\rho_\ell}(J_\ell, X)$ can be found as the solution to

$$\int_{J_\ell} \langle \dot{\boldsymbol{p}} - \boldsymbol{A}\boldsymbol{p},\, \boldsymbol{q} \rangle \, \mathrm{d}t + \left\langle \boldsymbol{p}_{\ell-1}^+ - \boldsymbol{p}_{\ell-1}^- \,,\, \boldsymbol{q}_{\ell-1}^+ \right\rangle = 0. \tag{IV.2.3}$$

For $1 \le m \le M$ let $\{\varphi_j\}_{j=0}^{\rho_m}$ be a basis in $\mathcal{P}^{\rho_m}((-1,1), X)$, then the corresponding temporal shape functions on J_m are $\varphi_j \circ F_m^{-1}$, $0 \le j \le \rho_m$, where the affine map $F_m \colon (-1,1) \to J_m$ is defined by $t = F_m(\tau) = \frac{1}{2}(t_m + t_{m-1}) + \frac{1}{2}(t_m - t_{m-1})\tau$ for $\tau \in (-1,1)$. If $\boldsymbol{p}|_{J_m} = \sum_{j=0}^{\rho_m} (\boldsymbol{P}_m)_j \cdot (\varphi_j \circ F_m^{-1})$, where $\boldsymbol{P}_m \in X^{\rho_m+1} \simeq \mathbb{R}^{(\rho_m+1)\times n_1 \times \ldots \times n_d}$, then (IV.2.3) yields the following linear system on the coefficients:

$$(\boldsymbol{C}_m \otimes \mathrm{Id} - \boldsymbol{G}_m \otimes \boldsymbol{A}) \cdot \boldsymbol{P}_m = \boldsymbol{\varphi}_{m-1} \otimes \boldsymbol{p}_{m-1}^- \,, \tag{IV.2.4}$$

where $(\boldsymbol{C}_m)_{\substack{i\\j}} = \int_{-1}^{1} \varphi_j'(\tau)\,\varphi_i(\tau)\,\mathrm{d}\tau + \varphi_j(-1)\,\varphi_i(-1)$ and $(\boldsymbol{G}_m)_{\substack{i\\j}} = \int_{-1}^{1} \varphi_j(\tau)\,\varphi_i(\tau)\mathrm{d}\tau$ for $0 \leq i, j \leq \rho_m$, while $\left(\boldsymbol{\varphi}_{m-1}\right)_i = \varphi_i(-1)$ for $0 \leq i \leq \rho_m$.

In order to represent the system while preserving the tensor structure (IV.2.4), we attach the index corresponding to the orthogonal polynomials of the temporal discretization as a single (unquantized) dimension to the first "virtual" spatial index.

Theorem IV.2.4. *Assume that $\boldsymbol{A}$ is represented in the QTT or QT3 format in terms of D cores with ranks $r_1, \ldots, r_{D-1}$. Then the matrix of system* (IV.2.4) *can be represented in the corresponding format in terms of $D+1$ cores with ranks $2, r_1 + 1, \ldots, r_{D-1} + 1$.*

Proof. The proof is the work of Vladimir Kazeev and is given in the Supplementary Text of [Kaz+14]. □

IV.2.1 Algorithm Summary

Assuming we have a finite state projection of the CME, we summarize our approach to the CME solution by outlining the two main algorithms we propose for its subsequent efficient solution. Given a reaction network and a finite state projection Algorithm 9 approximates the CME operator in QTT format. Algorithm 10 then describes the time-stepping procedure for computing the solution. Note that the integrals in Algorithm 10 may be pre-computed depending on the choice of temporal basis functions. E.g. if one chooses the Legendre polynomials as the basis, then there are explicit solutions of the integrals involved.

Algorithm 9 Assemble Projected CME Operator in QTT Matrix Format

Require: Rank-1 separable propensity functions $\omega^s(\underline{x})$, stoichiometric vectors $\underline{\eta}^s$, rectangular FSP truncation $[0,\ldots,2^{l_1}-1]\times\ldots\times[0,\ldots,2^{l_d}-1]$, propensity QTT compression tolerance $\varepsilon_{\text{prop}}$, a QTT approximation subroutine `QTT_Approx` implementing [Ose11, Algorithm 1] for quantized vectors.

Ensure: Projected CME operator $\boldsymbol{A}$ in QTT matrix format

Initialize $\boldsymbol{A}=0$;

for $s=1,\ldots,R$ **do**

$\boldsymbol{S}_{\underline{\eta}^s} = \boldsymbol{S}^{(l_1)}_{\eta^s_1}\otimes\ldots\otimes\boldsymbol{S}^{(l_d)}_{\eta^s_d}$;

for $k=1,\ldots,d$ **do**

$\boldsymbol{\omega}^s_k = \texttt{QTT\_Approx}(\boldsymbol{\omega}^s_k(0,\ldots,2^{l_k}-1))$ with tolerance $\varepsilon_{\text{prop}}$;

end for

$\boldsymbol{\omega}^s = \boldsymbol{\omega}^s_1\otimes\ldots\otimes\boldsymbol{\omega}^s_d$;

$\boldsymbol{M}_{\omega^s} = \operatorname{diag}\boldsymbol{\omega}^s$;

$\boldsymbol{A} = \boldsymbol{A} + \left(\boldsymbol{S}_{\underline{\eta}^s} - \mathrm{Id}\right)\circ\boldsymbol{M}_{\omega^s}$;

end for

Algorithm 10 *hp*-DG-QTT CME Solver

Require: Projected CME operator A in QTT format, time mesh $\mathcal{M} = \{J_m\}_{m=1}^{M}$, polynomial orders $\underline{\rho} \in \mathbb{N}_{\geq 0}^{M}$, basis of temporal shape functions $\{\varphi_j\}_{j=0}^{\infty}$, DMRG-solver tolerance `RES`

Ensure: Approximate solution $\boldsymbol{p} \in \mathcal{P}^{\underline{\rho}}(\mathcal{M}, X)$ of the evolution $\dot{p} = Ap$

for $m = 1, \ldots, M$ **do**

for $i, j = 0, \ldots, \rho_m$ **do**

$(\boldsymbol{C}_m)_{\substack{i \\ j}} = \int_{-1}^{1} \varphi_j'(\tau)\, \varphi_i(\tau)\, \mathrm{d}\tau + \varphi_j(-1)\, \varphi_i(-1)$;

$(\boldsymbol{G}_m)_{\substack{i \\ j}} = \int_{-1}^{1} \varphi_j(\tau)\, \varphi_i(\tau) \mathrm{d}\tau$;

end for

Solve $(\boldsymbol{C}_m \otimes \mathrm{Id} - \boldsymbol{G}_m \otimes \boldsymbol{A}) \cdot \boldsymbol{P}_m = \boldsymbol{\varphi}_{m-1} \otimes \boldsymbol{p}_{m-1}^{-}$, for $\boldsymbol{P}_m$ using DMRG-solver with tolerance `RES`;

$\boldsymbol{p}_m = \sum_{j=1}^{\rho_m} \boldsymbol{P}_{m,j}\, \varphi_j(1)$;

end for

IV.2.2 Comparison to Krylov Subspace Methods

The solution at a particular time of a finite state projection of the CME is given analytically by the matrix exponential, but the numerical computation of such solutions for large $\boldsymbol{A}$ is often expensive. When $\boldsymbol{A}$ is sparse, however, the Krylov subspace method [Saa92; SS86] is one approach for performing the computation for the CME as described in [Bur+06]. The method uses the Arnoldi iteration to compute the Krylov subspace up to some order of accuracy then computes the matrix exponential in that smaller space (by diagonal Padé approximation). The publicly available Expokit Toolbox by Sidje [Sid98] provides an implementation of the algorithm.

It is important to note that the algorithm steps incrementally in time rather than jumping to the desired time step. In the context of the CME, this means that the faster the support of the pdf fills the set of reachable states, the more expensive this algorithm becomes to compute. When there is reason to believe the support of the pdf remains small, then the algorithm can be expected to compute efficiently over large time intervals. Generically, however,the support of the pdf quickly fills the set of reachable states which may include every state retained in the projection. This renders the Arnoldi iteration computationally expensive at each time step.

The QTT method effectively circumvents this problem by storing the computed solution at each time step in the QTT format and exploiting the fast algorithms for basic tensor arithmetic available in this format. While it is unknown whether a given reaction network and initial probability distribution will produce an evolution

that can be represented well by a QTT formatted tensor with low QTT ranks, our numerical experiments find this often is the case and that the savings over using traditional sparse representations of vectors and matrices may be quite substantial.

Below we compare our method to the Krylov subspace approach in the toggle switch example which does not exhibit any pronounced structure favoring either one of the methods (rank-one separability and sparse structure respectively).

Algorithm 11 Krylov-Based Matrix Exponential from MacNamara, *et al.* [Bur+06]

Require: Matrix $\mathbf{A}$ in sparse format, vector $\mathbf{v}$ in sparse format, time interval τ, dimension of Krylov subspace m, local error tolerance ε.

Ensure: $\exp(\tau\mathbf{A})\mathbf{v}$

for $k = 0, 1, 2, \ldots$ until $t_k = t_f$ **do**

$[\mathbf{V_{m+1}}, \mathbf{H_{m+1}}]$:=Arnoldi$(\mathbf{A}, \mathbf{v_k(t_k)}, \mathbf{m})$;

while err$\leq 1.2\varepsilon$ **do**

τ_k:= step size;

$\mathbf{v_k(t_k)} = \beta \mathbf{V_{m+1}} \exp(\tau_\mathbf{k} \mathbf{H_{m+1}}) \mathbf{e_1}$;

err := numerical error estimate;

end while

$t_k + 1 := t_k + \tau_k$

end for

IV.3 Numerical Experiments

IV.3.1 Implementation Details

In order to solve the IVP (IV.1.2), we exploit the *hp*-DG-QTT algorithm proposed in [KRS12], adapted to the CME as described above, and implemented in MATLAB. It uses an exponentially convergent spectral time discretization scheme which reduces the solution of the IVP to a sequence of QTT structured linear system solves in the "species space". We exploit a DMRG optimization-based linear system solver [Vid03; VPC04; Whi93] and elaborated on in the context of the TT format in [DO11] and available as the function `dmrg_solve3` of the TT Toolbox.

The temporal discretization requires three elements: specification of a basis of orthogonal polynomials, a mesh of time points to solve on, and a schedule specifying the polynomial orders of the discretization over each subinterval. For the numerical experiments, we use normalized Legendre polynomials as the polynomial basis for the temporal discretization. We use a time mesh that is split into three regions $[0, h]$, $[h, T_1]$, and $[T_1, T]$, where each region is chosen heuristically to most efficiently reproduce the behavior of the solution in that time region. In the first region, $[0, h]$ we initialize the algorithm with $M_0 = 10$ steps increasing geometrically with the factor $\sigma_0 = \frac{t_{m-1}}{t_m} = 0.5$. In the region $[h, T_1]$ where the solution exhibits fast transient behavior, we use a uniform time mesh of width h. In the last region, $[T_1, T]$, when the solution is close to stationary, we increase the mesh width geometrically width

grading factor $\sigma_2 = \frac{t_{m-1}}{t_m} = 1 - \frac{h}{T_1}$. On every time interval we use polynomial spaces of degree $\rho = 3$ to discretize (IV.1.2). Note that in the reported results we are forced to use rather small time steps and to restrict the polynomial spaces to low degree since the DMRG solver, like any other available tensor-structured solver, converges only locally. Thus the DMRG solver prevents us from taking full advantage of the exponential convergence of the *hp*-DG time discretization.

While the "DMRG" solver still lacks a rigorous theoretical foundation, it proves to be highly efficient in many applications, including our experiments. In [RU12] a closely related *Alternating Least Squares (ALS)* approach was mathematically analyzed and shown to converge locally. More on the mathematical ideas behind the ALS and DMRG optimization in the TT format can be found in [HRS12].

The DMRG solver requires specification of the following parameters: the relative tolerance of the residual `RES` in the Frobenius norm for the linear system, the maximum allowed number of DMRG sweeps `SWP`, the maximum number `RST` of restarts for the GMRES solution of the DMRG local optimization problems, the max number `ITR` of iterations before restarting the GMRES procedure, the maximum allowed rank `RMX`, and the rank of random components added to the solution to avoid stopping in local minima `KCK`. The DMRG procedure loops until either the number of iterations reach `SWP` or the residual tolerance is met. In each simulation run in each numerical experiment, these parameters are held fix for every time step.

Once a time step is successfully computed, since the elements of $\boldsymbol{P}_m$ are the

coefficients of the expansion in the temporal basis, we contract over the temporal indices to evaluate $\boldsymbol{p}_m^-$. Since the ranks of $\boldsymbol{p}_m^-$ are generally suboptimal we recompress in the TT format with relative ℓ_2-accuracy `EPS`.

We evaluate the accuracy of the approach in each experiment by comparison to reference data. In the first example, the problem setup leads to solutions that are symmetric and independent so the marginal of each species may be computed separately and combined to form the full PDF. For this problem, we solve each subproblem using the standard MATLAB solver `ode15s` in the sparse format to obtain the univariate factor of a reference solution. In other examples we used SPSens beta 3.4, a massively parallel package for the stochastic simulation of chemical networks (`http://sourceforge.net/projects/spsens/`) [SRK13], to construct reference PDFs. The stochastic simulations were carried out on up to 1500 cores of Brutus, a high-performance cluster of ETH Zürich (`https://www1.ethz.ch/id/services/list/comp_zentral/cluster/index_EN`).

Due to the high dimensionality of some problems, it is actually computationally difficult to make a comparison using the entire PDFs; in these cases we only compare marginal distributions. Δ_{ℓ_p} denotes the ℓ_p-norm of the difference in each case. Interestingly, the reference solution can be realistically obtained with a certain level of accuracy which cannot be reduced arbitrarily so in some cases the *hp*-DG-QTT solution appears more accurate.

Since our solution sometimes appears more accurate than the reference data,

in the first and last experiments we reapproximate the solution again with relative ℓ_2-accuracy $\alpha \cdot \frac{\Delta_{\ell_2}}{\|\bar{\boldsymbol{p}_m}\|}$, with $\alpha = 0.05$ for the first example and $\alpha = 0.01$ for the second. We refer to the reapproximation procedure as *truncation* and the result the *truncated solution*. The procedure results in a solution which has minimal QTT ranks needed to represent the data at a similar level of accuracy as the reference data. The relative approximation tolerance ensures that the relative discrepancy in the ℓ_2-norm grows by at most a factor of $1 + \alpha$.

Our Matlab implementation relies on the TT toolbox, publicly available at `http://spring.inm.ras.ru/osel` and `http://github.com/oseledets/TT-Toolbox`, for the object oriented implementation of the TT data structure and arithmetic operations as well as the DMRG linear system solver. For consistency, we use the GitHub version of July 12, 2012 in all examples below. We run the *hp*-DG-QTT solver in MATLAB 7.12.0.635 (R2011a) on a laptop with a 2.7 GHz dual-core processor and 4 GB RAM, and report the computational time in seconds.

IV.3.2 d Independent Birth-Death Processes

As a first example we consider a system composed of d chemical species with $\{X_1, \ldots, X_d\}$ a vector of random variables representing the species count of each. The dynamics of the random vector are governed by independent birth-death processes. For the k-th species, the corresponding reactions are given by

$$\emptyset \underset{d_k}{\overset{b_k}{\rightleftharpoons}} \mathrm{X}_k$$

	Direct Approach		**Proposed Approach**					
run	solution	operator	solution		truncated solution		operator	
	`Mem`	`Mem`	`Mem`	ratio	`Mem`	ratio	`Mem`	ratio
d independent birth-death processes								
$d=1$	$4.10_{\mathbf{3}}$	$1.68_{\mathbf{7}}$	736	$1.80_{\mathbf{-1}}$	264	$6.45_{\mathbf{-2}}$	992	$5.91_{\mathbf{-5}}$
$d=2$	$1.68_{\mathbf{7}}$	$2.82_{\mathbf{14}}$	3858	$2.30_{\mathbf{-4}}$	528	$3.15_{\mathbf{-5}}$	2852	$1.01_{\mathbf{-11}}$
$d=3$	$6.87_{\mathbf{10}}$	$4.72_{\mathbf{21}}$	7742	$1.13_{\mathbf{-7}}$	898	$1.31_{\mathbf{-8}}$	4800	$1.02_{\mathbf{-18}}$
$d=4$	$2.81_{\mathbf{14}}$	$7.90_{\mathbf{28}}$	12176	$4.33_{\mathbf{-11}}$	1432	$5.09_{\mathbf{-12}}$	6748	$8.52_{\mathbf{-26}}$
$d=5$	$1.15_{\mathbf{18}}$	$1.32_{\mathbf{36}}$	16262	$1.41_{\mathbf{-14}}$	1946	$1.69_{\mathbf{-15}}$	11032	$8.30_{\mathbf{-33}}$
genetic toggle switch								
only	$3.36_{\mathbf{7}}$	$1.12_{\mathbf{15}}$	65264	$1.95_{\mathbf{-3}}$	–	–	10988	$9.76_{\mathbf{-12}}$
enzymatic futile cycle								
(A)	$4.19_{\mathbf{6}}$	$1.76_{\mathbf{13}}$	18396	$4.39_{\mathbf{-3}}$	8472	$2.02_{\mathbf{-3}}$	25848	$1.47_{\mathbf{-9}}$
(D)			360332	$8.59_{\mathbf{-2}}$	290144	$6.92_{\mathbf{-2}}$	5584	$3.17_{\mathbf{-10}}$

Table IV.1: Overview of the QTT compression of the storage needed for solutions (maximum throughout the time stepping) and CME operators. For details on "truncated solution" see **Numerical experiments. Common details**. Solution `Mem` in the Direct Approach is the number of states taken into account in the FSP, which is equal to the number of entries, N, in the solution vector. For the CME operator, `Mem` is N^2, the number of entries in the matrix. In the Proposed QTT Approach, *ratio* indicates the memory storage compression ratio, i.e. the ratio of `Mem` in the Proposed QTT Approach to that in the Direct Approach. In the sparse representation of the CME operator the number of nonzero entries would be $\mathcal{O}(N)$ rather than N^2. The exponents are given in boldface for the base 10.

where b_k is the spontaneous creation rate and d_k is the destruction rate for species X_k. This problem is perfectly separable in the sense that the dynamics of any one chemical species of this system is independent of the dynamics of all others. Given the initial condition $X_k(0) = \xi_k$ for each k, the marginal distribution for any one species X_k at time t is given by:

$$p_k(x_k; t) = \mathcal{P}(x_k, \lambda_k(t)) \star_{x_k} \mathcal{M}(x_k, \xi_k, p^{(k)}(t)), \quad x_k \in \mathbb{Z}_{\geq 0}$$

where $\mathcal{P}(\cdot, \lambda_k(t))$ is the Poisson distribution with parameter $\lambda_k(t)$, $\star_{x_k}$ indicates the discrete convolution in variable x_k, $\mathcal{M}\big(x_k, \xi_k, p^{(k)}(t)\big)$ the multinomial distribution with parameter $p^{(k)}(t)$, and the parameters $p^{(k)}$ and λ_k evolve according to the reaction rate equations

$$\tfrac{\mathrm{d}}{\mathrm{d}t}\, p^{(k)}(t) = -d_k p^{(k)}(t)\,, \quad \tfrac{\mathrm{d}}{\mathrm{d}t}\, \lambda_k(t) = b_k - d_k \lambda_k(t),$$

$$p^{(k)}(0) = 1, \qquad\qquad \lambda_k(t) = 0.$$

See [JH07, Theorem 1] for details. Since $X_1, \ldots, X_k$ are mutually independent, the joint PDF at time t, $\boldsymbol{p}(t)$, is the product of the marginals:

$$\boldsymbol{p}(t) = \prod_{k=1}^{d} p_k(t)$$

that is, this system has an explicit formula for the solution regardless of the number of chemical species involved. We can, therefore, evaluate the accuracy and observe the complexity scaling of the *hp*-DG-QTT solver as the number of chemical species increases.

For numerical simulations we assume $b_k = 1000$ and $d_k = 1$ for $1 \leq k \leq d$ and consider the FSP with $l_k = 12$. We solve the corresponding projected CME

d	N	$\frac{\|\boldsymbol{A}\boldsymbol{p}_0\|_2}{\|\boldsymbol{p}_0\|_2}$	$\frac{\|\boldsymbol{A}\boldsymbol{p}_M^-\|_2}{\|\boldsymbol{p}_M^-\|_2}$	r_{eff}	Δ_{ℓ_2}	TIME
1	2^{12}	$1.4_{+\mathbf{3}}$	$1.0_{-\mathbf{3}}$	3.53	$1.9_{-\mathbf{5}}$	87
2	2^{24}	$2.4_{+\mathbf{3}}$	$1.4_{-\mathbf{3}}$	3.42	$2.3_{-\mathbf{5}}$	704
3	2^{36}	$3.5_{+\mathbf{3}}$	$1.8_{-\mathbf{3}}$	3.38	$3.5_{-\mathbf{5}}$	1548
4	2^{48}	$4.5_{+\mathbf{3}}$	$2.0_{-\mathbf{3}}$	3.37	$3.6_{-\mathbf{5}}$	2516
5	2^{60}	$5.5_{+\mathbf{3}}$	$2.3_{-\mathbf{3}}$	3.36	$3.5_{-\mathbf{5}}$	3544

Table IV.2: d independent birth-death processes: $r_{\text{eff}} = r_{\text{eff}}\left[\boldsymbol{p}_M^-\right]$, $\Delta_{\ell_2} = \Delta_{\ell_2}\left[\boldsymbol{p}_M^-\right]$, computational TIME in seconds; $r_{\max}\left[\boldsymbol{p}_M^-\right] = 6$ for all d. N is the number of states taken into account in the FSP. The exponents are given in boldface for the base 10.

for $d = 1, 2, 3, 4, 5$ to check that in all these cases the hp-DG-QTT method using the ordering (III.1.1) without transposition is capable of revealing the same low-rank QTT structure of the solution. For the CME operator we have $r_{\max}[\boldsymbol{A}] \leq 8$ up to accuracy $5 \cdot 10^{-15}$. We compute the evolution of the PDF of the system for the zero initial value through $M = 569$ time steps till $T = 10$. In this experiment, $T_1 = 10^{-1}$ and $h = 10^{-3}$. The settings of the DMRG solver are: RES $= 2 \cdot 10^{-6}$, SWP $= 2$, RMX $= 20$, ITR $= 100$, RST $= 1$, KCK $= 1$. The evaluation accuracy is EPS $= 10^{-8}$.

The results, which are presented in Figure IV.1 and Table IV.2, show that the same low-rank structure of the solution is adaptively reconstructed by the algorithm for all d considered.

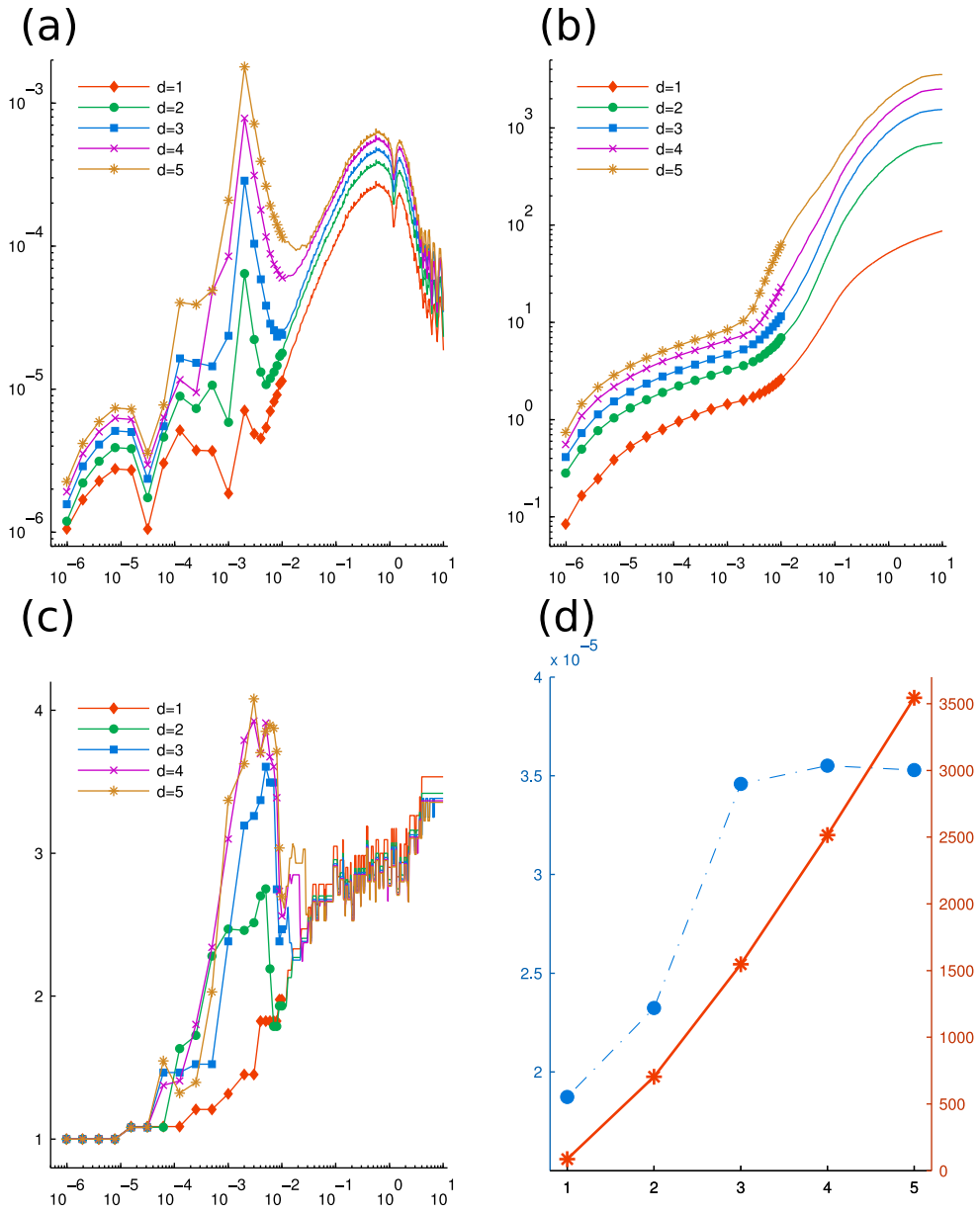


Figure IV.1: d **independent birth-death processes.** The maximum QTT ranks of the solutions, $r_{\max}\left[\boldsymbol{p}_M^-\right] = 6$ for each d. Markers are omitted for $t_m > 10^{-2}$ in (a)–(c). (a) Relative discrepancy $\Delta_{\ell_2}\left[\boldsymbol{p}_m^-\right] / \left\|\boldsymbol{p}_M^-\right\|$ (after truncation) vs. t_m. (b) Cumulative computation time (in seconds) vs. t_m. (c) Effective QTT rank $r_{\text{eff}}\left[\boldsymbol{p}_m^-\right]$ (after truncation) vs. t_m. (d) Relative discrepancy $\Delta_{\ell_2}\left[\boldsymbol{p}_M^-\right] / \left\|\boldsymbol{p}_M^-\right\|$ (blue) and total computation time (red) vs. d.

IV.3.3 Toggle Switch

The next example models a synthetic gene-regulatory circuit designed to produce bistability over a wide range of parameter values [GCC00]. The network consists of two promoters constructed in a mutually inhibitory configuration that implement a double negative feedback loop, causing the network to exhibit robust bistable behavior (see Figure IV.2). If the concentration of one repressor is high, this lowers the production rate of the other repressor, keeping its concentration low. This allows a high rate of production of the original repressor, thereby stabilizing its high concentration.

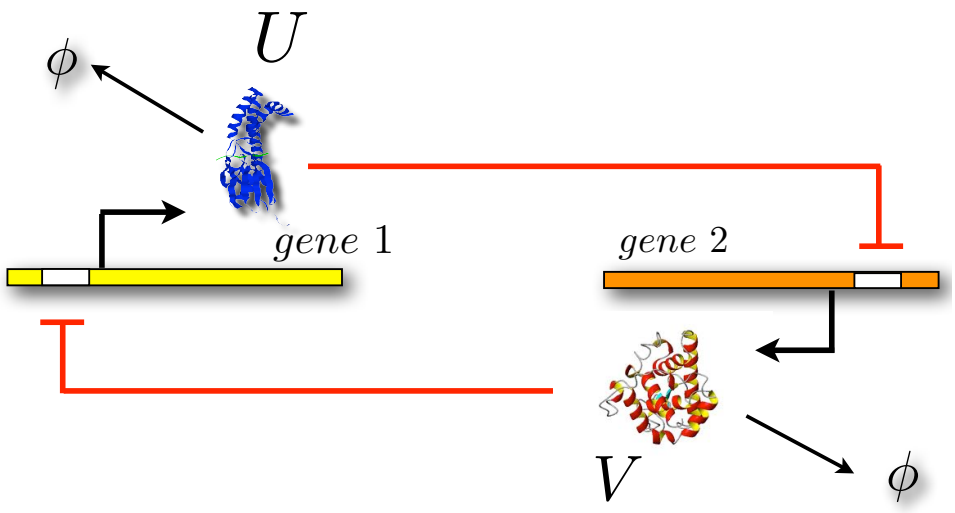


Figure IV.2: **Toggle Switch consisting of double negative feedback loop.** Species U represses the production of species V and vice versa. Photo courtesy of Mustafa Khammash.

A stochastic model of the toggle switch was considered in [MK08] and consists

of the following four reactions:

$$\emptyset \xrightarrow{\frac{\alpha_1}{1+V^\beta}} \mathrm{U}, \quad \emptyset \xrightarrow{\frac{\alpha_2}{1+U^\gamma}} \mathrm{V}$$

$$\mathrm{U} \xrightarrow{\delta_1} \emptyset, \quad \mathrm{V} \xrightarrow{\delta_2} \emptyset$$

where U and V represent the two repressors. Denote the species counts of each by U and V, respectively. The stochastic model admits a bimodal stationary distribution over a wide range of values of the rate constants. We consider the set of parameters from [MK08] which were selected to test the efficiency of using available numerical algorithms to calculate matrix exponentials to solve low dimensional FSP approximations of the CME. We then scaled the parameters so that a larger set of states would be required to guarantee an FSP truncation with low approximation error. While a different set of parameters were considered in [Deu+08; SLE09], which required a larger FSP truncation, this choice of values renders the system symmetric under interchange of the roles of U and V. This situation is less biologically relevant than what we consider here.

For this numerical example we assume $\alpha_1 = 5000$, $\alpha_2 = 1600$, $\beta = 2.5$, $\gamma = 1.5$, $\delta_1 = \delta_2 = 1$. We consider the FSP with $l_U = 13$, $l_V = 12$, which allows to take into account 2^{25} states. The initial value is zero. We use the ordering (III.1.1) without transposition. For the CME operator we have $r_{\max}[\boldsymbol{A}] = 14$ and $r_{\text{eff}}[\boldsymbol{A}] = 10.89$ up to accuracy 10^{-14}. We compute the evolution of the PDF up to time $T = 100$ with $M = 1111$ time steps. In this experiment $T_1 = 10$ and $h = 0.03$. The settings of the DMRG solver are: `RES` $= 10^{-6}$, `SWP` $= 3$, `RMX` $= 200$, `ITR` $= 100$, `RST` $= 2$, `KCK` $= 2$.

The evaluation accuracy is $\texttt{EPS} = 10^{-8}$.

The results are presented in Figure IV.3. At the terminal time T we have $\texttt{ERR}_{\Sigma}\left[\boldsymbol{p}_M^-\right] = 3.17 \cdot 10^{-5}$. The overall computation time is 14728 seconds. The validation with the PDF based on 816 million Monte Carlo simulations (every 1000 draws taking on average over 360 seconds, adding up to the overall CPU time over $3 \cdot 10^8$ seconds), indicates $\Delta_{\ell_1}\left[\boldsymbol{p}_M^-\right] = 8.34 \cdot 10^{-4}$, and for the 2- and Chebyshev norms we have $\Delta_{\ell_2}\left[\boldsymbol{p}_M^-\right] / \left\|\boldsymbol{p}_M^-\right\|_2 = 6.62 \cdot 10^{-4}$ and $\Delta_{\ell_\infty}\left[\boldsymbol{p}_M^-\right] = 5.50 \cdot 10^{-6}$. As for the ranks, $r_{\text{eff}}\left[\boldsymbol{p}_M^-\right] = 8.74$ and $r_{\max}\left[\boldsymbol{p}_M^-\right] = 13$. Figure IV.3 (c) shows that after $t \approx 20$ the norm of the time derivative stagnates at approximately 10^{-5} determined by the accuracy parameters chosen, and the following time steps require negligible computational effort. At the same time, as we see in Figure IV.3 (b), all QTT ranks stabilize under 15, but the transient phase preceding that moment involves far higher ranks. Figure IV.4 (a) presents a snapshot of the distribution.

Comparison to the Krylov subspace approach. We compared the performance of our proposed method to that of the Krylov subspace approach implemented in Sidje's Expokit [Sid98]. In order to make the comparison as fair as possible we further restricted the FSP truncation used by the Krylov approach to a *hyperbolic cross*, that is, we only kept states with indices (j_U, j_V) satisfying the condition $(j_U+1)\cdot(j_V+1) < 9216000$. Effectively, this reduces the states in the truncation from 2^{25} to 21120695, a reduction of about a third. A similar truncation was used for this model in [MK08].

We emphasize that formulating this hyperbolic cross truncation requires special

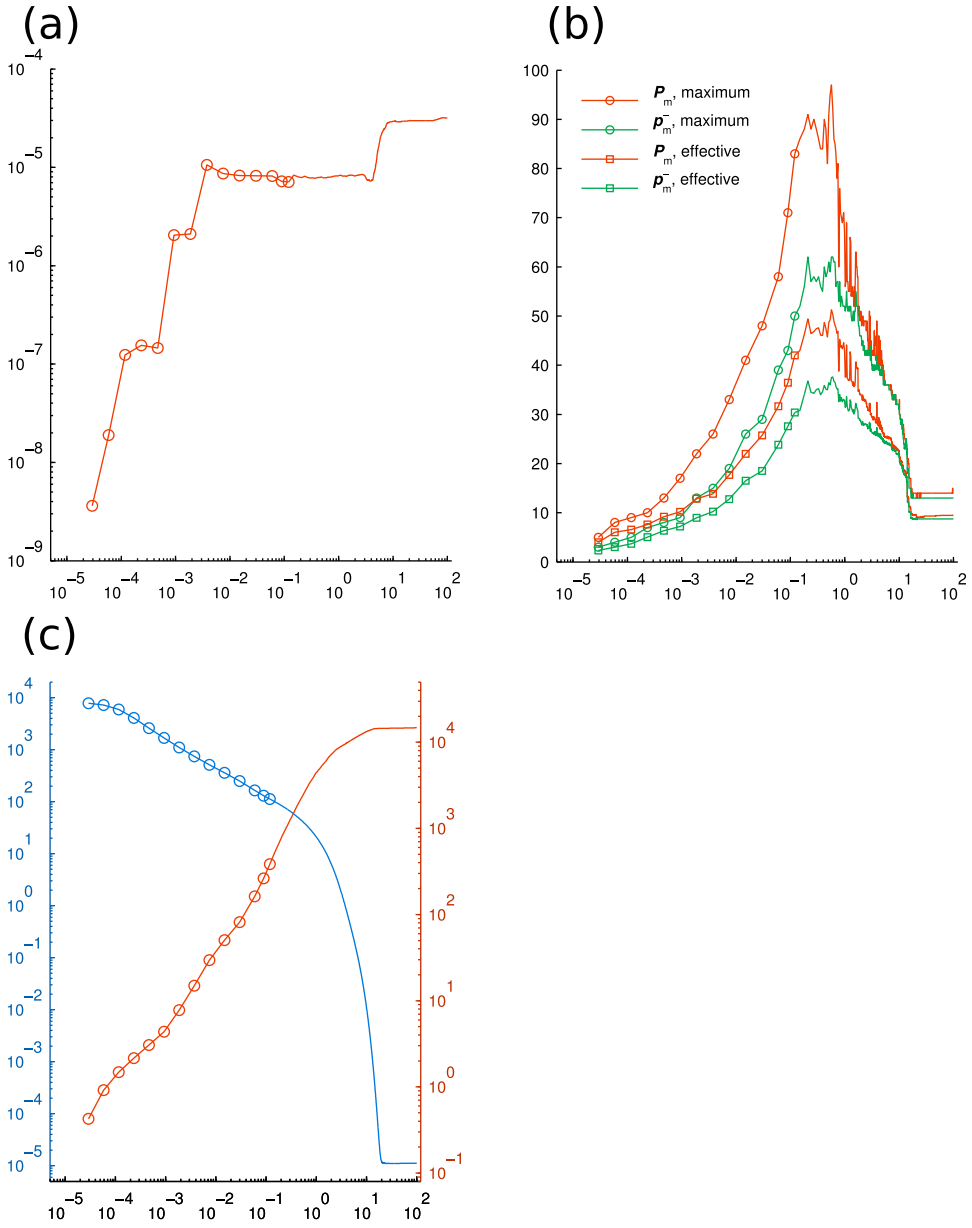


Figure IV.3: **Genetic toggle switch.** The values are given vs. t_m. Markers are omitted for $t_m > 10^{-1}$. (a) Probability deficiency $\mathtt{ERR}_\Sigma\,[\boldsymbol{p}_m^-]$. (b) Maximum and effective QTT ranks of the computed solution. (c) Relative norm $\frac{\left\|\boldsymbol{A}\boldsymbol{p}_m^-\right\|_2}{\left\|\boldsymbol{p}_m^-\right\|_2}$ of the derivative (blue) and cumulative computation time (red, sec.)

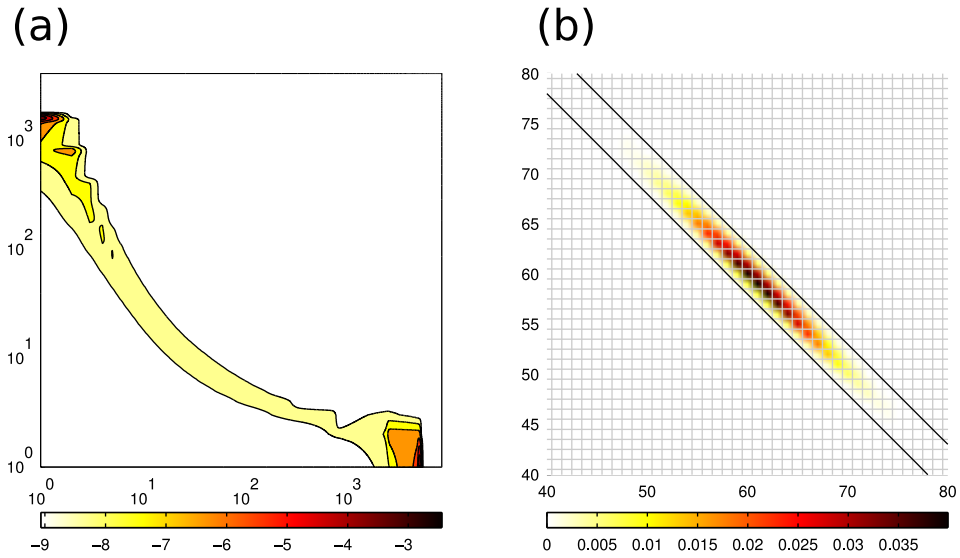


Figure IV.4: **Snapshots of solutions.** (a) Genetic toggle switch. The PDF for $m = 350$, $t_m \approx 10.18$, U (hor.) vs. V (vert.). As the process evolves, the probability mass becomes concentrated in two distinct regions. Contour coloring is logarithmically scaled with base 10. (b) Enzymatic futile cycle. The marginal PDF for $m = 20$, $t_m = 5{\cdot}10^{-3}$, X (vert.) vs. X* (hor.). Black diagonal lines delimit the states reachable from the initial condition. The transposed QTT format automatically exploits this sparsity pattern *of the full PDF* for compression without special input from the user.

insight into the problem on the part of the modeler. In constrast, our proposed method is completely naive in this respect, instead relying on the adaptivity of the QTT compression.

For the FSP with 2^{25} states considered we reach $t \approx 1$ with the first 43 time steps of our method in 4385 seconds; with the Krylov subspace method restricted to the hyperbolic cross, in 10333 seconds. For the discrepancy between the two solutions obtained we have $\Delta_{\ell_1} = 4.04 \cdot 10^{-5}$ and $\Delta_{\ell_\infty} = 9.64 \cdot 10^{-8}$.

At approximately $t = t_{43} \approx 1$, the decay of the relative norm of the solution becomes exponential; see Figure IV.3 (c). That is exploited by our method in two ways. On the one hand, we adjust the time mesh manually, which reduces the overall number of time steps needed to reach $t_{1111} = T$ from t_{43}: we take 1068 steps intead of approximately 3307 we would need if we had used a uniform time mesh for the long-term dynamics. On the other hand, what is more significant, the adaptive QTT representation used at each step yields a substantial speedup of the solution of linear systems, which is possible due to the rapid convergence of the solution to a stationary distribution. The Krylov subspace solver adapts the time mesh on its own, but employs no self-adaptivity for efficient storage of numerically computed states. As a result, the performance (in terms of the computational time vs. physical time of the system) decays much slower for the Krylov subspace solver, and our method excels even more in modelling the long-term dynamics. For example, our method achieves $t \approx 30$, when $\|\dot{\boldsymbol{p}}\|_2 \,/\, \|\boldsymbol{p}\|_2$ reaches $1.1 \cdot 10^{-5}$, with the overall computation time 14541

seconds compared to 126530 seconds of the Krylov subspace solver, i.e. approximately 8.7 times faster. For larger terminal times the advantage of our method becomes even more pronounced.

IV.3.4 Enzymatic Futile Cycle

Futile cycles are composed of two metabolic or signaling pathways that work in opposite directions so that the products of one pathway are the precursors of the other and vice versa, see Figure IV.5.

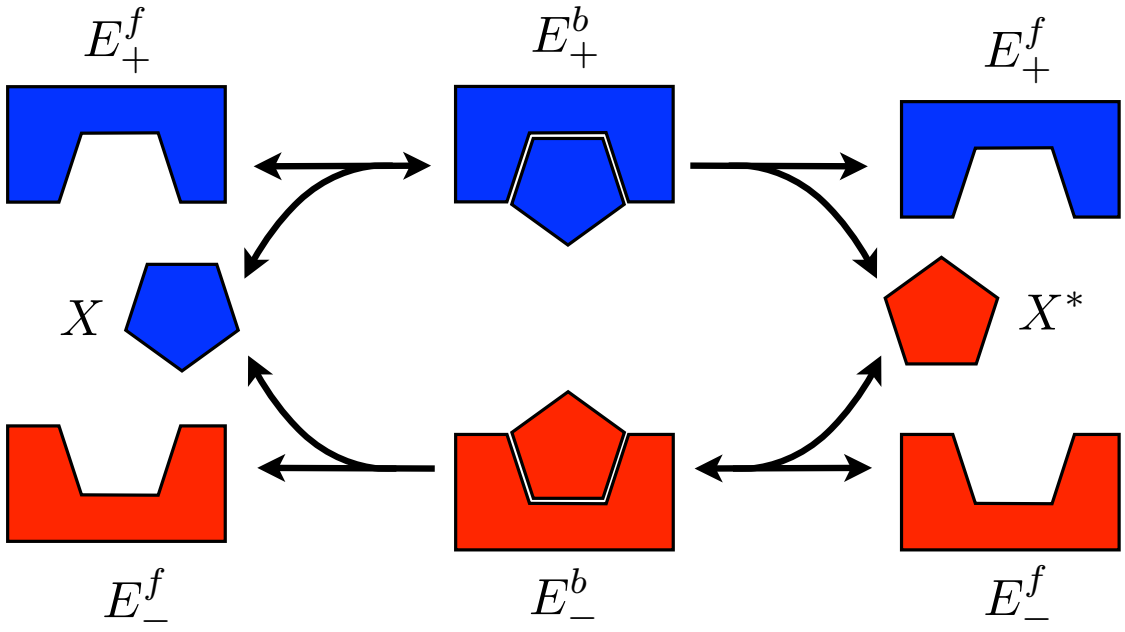


Figure IV.5: **Enzymatic Futile Cycle.** X is transformed into X^* and vice versa by enzymes E_+ and E_-, respectively. Photo courtesy of Mustafa Khammash

This biochemical network structure results in no net production of molecules and often results only in the dissipation of energy as heat [SOS04]. Nevertheless,

there is an abundance of known pathways that use this motif and it is thought to provide a highly tunable control mechanism with potentially high sensitivity [SOS04; SPA05].

[SPA05] introduced a stochastic version of the model with just the essential network components required to model the dynamics. The stochastic model consists of six chemical species and six reactions:

$$\mathrm{X} + \mathrm{E}_+^{\mathrm{f}} \underset{k_{+2}}{\overset{k_{+1}}{\rightleftharpoons}} \mathrm{E}_+^{\mathrm{b}}, \quad \mathrm{X}^* + \mathrm{E}_-^{\mathrm{f}} \underset{k_{-2}}{\overset{k_{-1}}{\rightleftharpoons}} \mathrm{E}_-^{\mathrm{b}},$$
$$\mathrm{E}_+^{\mathrm{b}} \xrightarrow{k_{+3}} \mathrm{E}_+^{\mathrm{f}} + \mathrm{X}^*, \quad \mathrm{E}_-^{\mathrm{b}} \xrightarrow{k_{-3}} \mathrm{E}_-^{\mathrm{f}} + \mathrm{X},$$

$\{\mathrm{X}, \mathrm{X}^*\}$ represent the forward substrate and product, $\{\mathrm{E}_+, \mathrm{E}_-\}$ denote the forward and reverse enzymes, respectively. Note that this system is closed meaning that particles are neither created nor destroyed. We denote the random variables representing the molecule count of each species with italics.

For the particular set of initial conditions considered in [SPA05] the number of states that are reachable is large enough to render a direct numerical solution of the CME impractical. The authors instead used the Gillespie Direct SSA to generate a large number of sample paths to estimate the distribution. The authors also applied a diffusion approximation to their model which resulted in a SDE which produced qualitatively similar dynamics.

To the authors' knowledge, no attempt has been made so far towards the direct numerical solution of the CME for this system.

At time t, let $X^{\mathrm{T}}(t)$ denote the total amount of both free and bound substrate,

and $E_+^{\mathrm{T}}(t)$ and $E_-^{\mathrm{T}}(t)$ the total forward and reverse enzymes, respectively. We observe the following conservation relations:

$$E_+^{\mathrm{f}}(t) + E_+^{\mathrm{b}}(t) = E_+^{\mathrm{T}}(t) = E_+^{\mathrm{T}}(0)$$

$$E_-^{\mathrm{f}}(t) + E_-^{\mathrm{b}}(t) = E_-^{\mathrm{T}}(t) = E_-^{\mathrm{T}}(0)$$

$$X(t) + X^*(t) + E_+^{\mathrm{b}}(t) + E_-^{\mathrm{b}}(t) = X^{\mathrm{T}}(t) = X^{\mathrm{T}}(0)$$

Using the above, one can establish an upper and lower bound relating the species count of $\mathrm{X}(t)$ to $\mathrm{X}^*(t)$ that depends only on the total initial amount of substrate and the total initial amount of enzymes in the system

$$X^{\mathrm{T}}(0) - X^*(t) \geq X(t) \geq X^{\mathrm{T}}(0) - X^*(t) - \left(E_+^{\mathrm{T}}(0) + E_-^{\mathrm{T}}(0)\right).$$

Assuming that the initial quantity of enzymes $E_+^{\mathrm{T}}(0) + E_-^{\mathrm{T}}(0)$ is small, for a given copy number of $X^*(t)$, $X(t)$ may take at most $E_+^{\mathrm{T}}(0) + E_-^{\mathrm{T}}(0)$ different values. Since $X^{\mathrm{T}}(t)$ is a conserved quantity, this means that $X(t)$ and $X^*(t)$ will be strongly anti-correlated with the set of reachable states having an affine structure. Under these circumstances, we find in our numerical experiments that the transposed QTT format is better suited than the standard QTT to efficiently represent the corresponding PDF, since the transposed format better utilizes the sparsity pattern of the full PDF for compression.

Following [SPA05], we consider $k_{+1} = 40$, $k_{+2} = 10^4$, $k_{+3} = 10^4$, $k_{-1} = 200$, $k_{-2} = 100$, $k_{-3} = 5000$. For initial value we take $E_\pm^f = 2, E_\pm^b = 0, X = 30, X^* = 90$. We consider the FSP projection with $l_{E_\pm^{b,f}} = 2$ and $l_X = l_{X^*} = 7$, i.e. with 2^{22}

states. We present 4 runs: (A), (B) and (C) use the transposed QTT format, and (D), the standard QTT. Theorems IV.2.2 and IV.2.1 bound the exact QTT ranks of the CME operator by 216 and 21 respectively, and numerically for accuracy 10^{-14} we have $r_{\max}[\boldsymbol{A}] = 38$, $r_{\text{eff}}[\boldsymbol{A}] = 17.93$ in (A)–(C) and $r_{\max}[\boldsymbol{A}] = 11$, $r_{\text{eff}}[\boldsymbol{A}] = 8.30$ in (D). We compute the evolution of the PDF up to time $T = 1$ with $M = 1332$ time steps. In this experiment $T_1 = 0.3$ and $h = 5 \cdot 10^{-4}$.

For (A) and (D), which differ in the format, we keep the same accuracy parameters: `RES` $= 10^{-6}$ and `EPS` $= 10^{-8}$. On the other hand, (B) and (C) use the same format as (A), but different accuracy parameters. In (B) they are `RES` $= 10^{-8}$ and `EPS` $= 10^{-10}$; in (C), `RES` $= 10^{-4}$ and `EPS` $= 10^{-6}$. We set `RMX` $= 200$ in (A)–(C) and `RMX` $= 400$ in (D). Other parameters of the DMRG solver are the same for all 4 runs: `SWP` $= 5$, `ITR` $= 50$, `RST` $= 2$, `KCK` $= 2$.

For the runs (A) and (D), which differ in the format, we keep the same accuracy parameters. The runs (B) and (C) use the same format as (A), but different accuracy parameters, so that they yield, respectively, a more accurate and a cruder solution as compared to (A).

This experiment shows, in particular, that lower ranks of the operator do not necessarily lead to lower ranks of the solution, and that in this example the transposed QTT format actually ensures smaller ranks of the solution than the QTT format without transposition does and than Theorem IV.2.2 suggests. As for the solution, we observe that $\max_{0 \le t_m \le 0.1} r_{\max}[\boldsymbol{P}_m]$ reaches 51 for (A) and 359 for (D).

For every m, we validate our solution $\boldsymbol{p}_m^-$ by comparing its marginal distribution $\sum_{E_\pm^{\mathrm{b,f}}} \boldsymbol{p}_m^-$ to that based on $18.6 \cdot 10^9$ Monte Carlo simulations (every 10000 draws taking at least 110 seconds, amounting to the overall CPU time over $2 \cdot 10^8$ seconds). The discrepancy $\Delta_{\ell_p} = \Delta_{\ell_p} \left[\sum_{E_\pm^{\mathrm{b,f}}} \boldsymbol{p}_m^- \right]$ in the marginal distribution with respect to X and X^* is reported for $p = 1$ in Figure IV.6 (a) and Table IV.3. With $p = 2$ we use it for the discrepancy-based truncation, which, as Figure IV.6 (b) shows, does not affect the probability deficiency significantly.

Figure IV.6 (a) shows that the refined run (B) yields the smallest discrepancy, which suggests that the reference distribution is sufficiently accurate to allow for the discrepancy to represent the actual error in the results of (A), (B) and (C). As we can see from Figure IV.6 (d), in all 4 runs the time derivative stagnates after $t \approx 0.1$, at lower levels for more accurate runs. Let us note that at that stage in (A)–(C) it exhibits relatively strong oscillations compared to (D), which happens due to different effect of the addition of random components in the DMRG solver in the presence and absence of the transposition. On the other hand, compared to (A), the run (D) yields a less accurate solution and reaches $t = 0.1$ almost 9 times later, the accuracy settings being the same in these two runs. In all, the transposition appears to make the QTT format far more efficient in this experiment, and we expect it to be even more so in larger systems of such type.

The results are given in Figures IV.6 and IV.7 and in Table IV.3. Figure IV.4 (b) presents a snapshot of the marginal distribution.

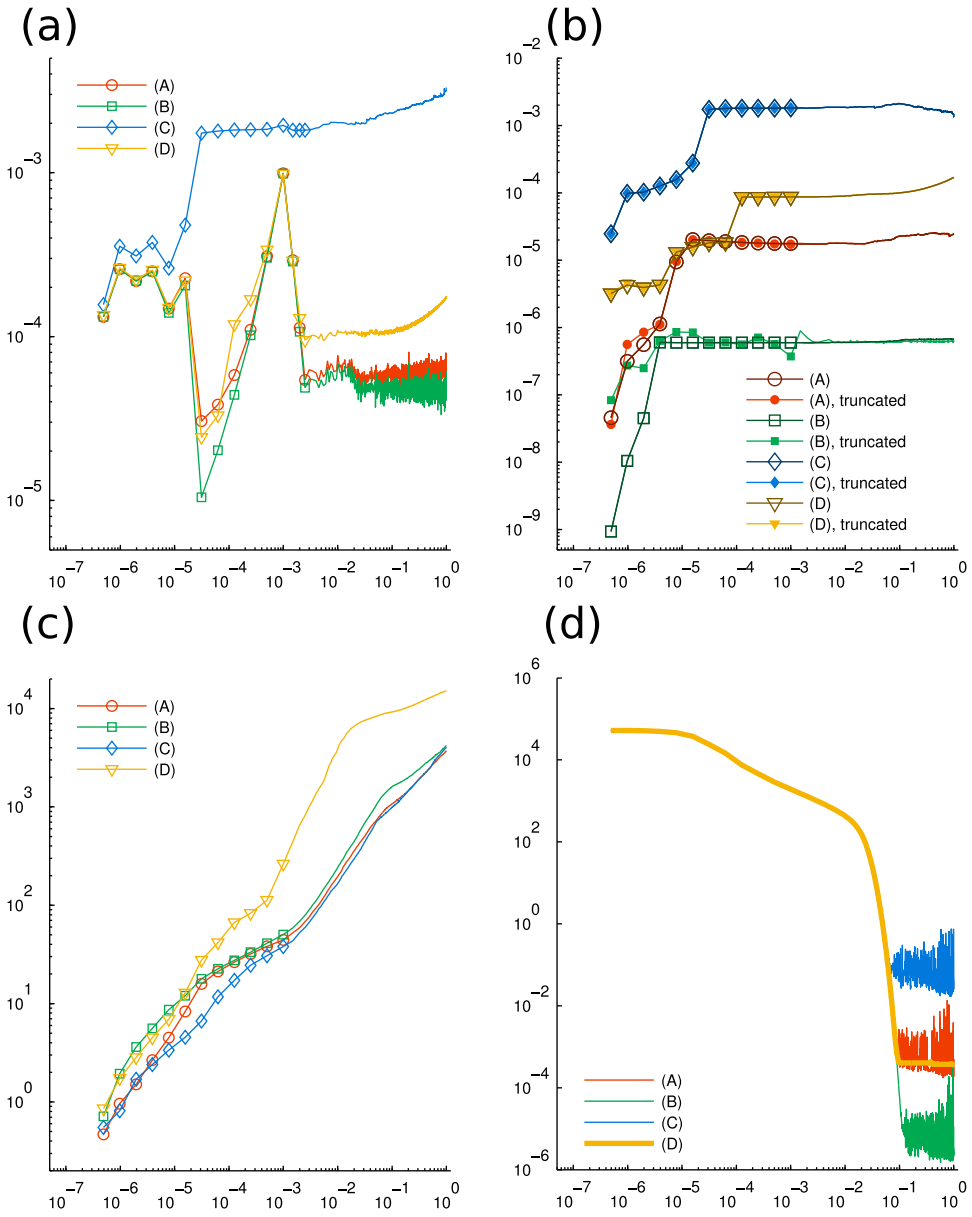


Figure IV.6: **Enzymatic futile cycle.** The values are given vs. t_m. Markers are omitted for $t_m \geq 2 \cdot 10^{-3}$ in (a)–(c). (a) Discrepancy Δ_{ℓ_1} (before truncation) from the marginal PDF based on Monte Carlo simulations. (b) Probability deficiency $\texttt{ERR}_{\Sigma}[\boldsymbol{p}_m^-]$. (c) Cumulative computation time (sec.) (d) Relative norm $\frac{\|\boldsymbol{A}\boldsymbol{p}_m^-\|_2}{\|\boldsymbol{p}_m^-\|_2}$ of the derivative.

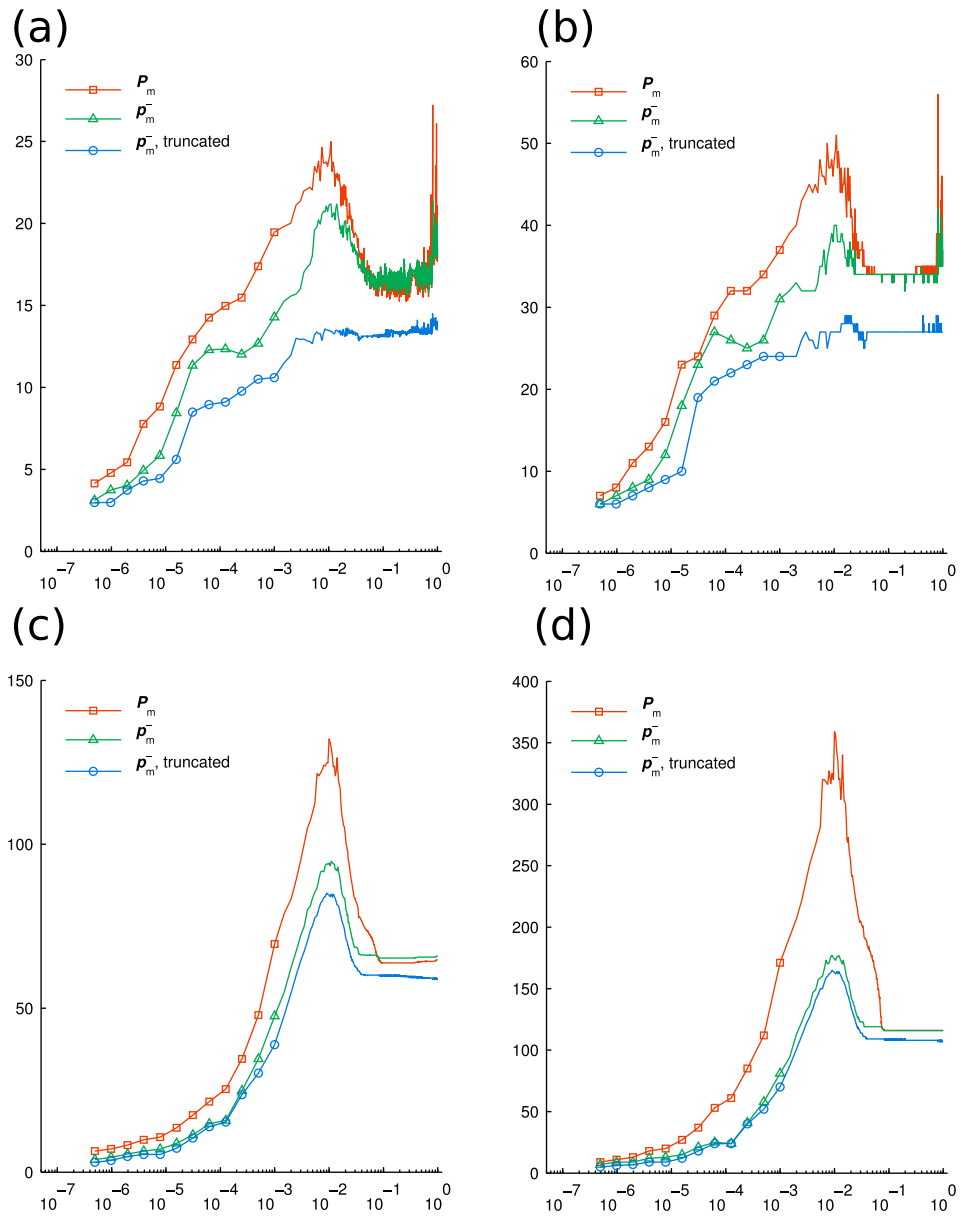


Figure IV.7: **Enzymatic futile cycle.** QTT ranks of the solution. The values are given vs. t_m. Markers are omitted for $t_m \geq 2 \cdot 10^{-3}$. (a) Effective QTT ranks r_{eff} for parameter set (A). (b) Maximum QTT ranks $r_{\max}$ for parameter set (A). (c) Effective QTT ranks r_{eff} for parameter set (D). (d) Maximum QTT ranks $r_{\max}$ for parameter set (D).

run	$\frac{\|\boldsymbol{A}\boldsymbol{p}_m^-\|_2}{\|\boldsymbol{p}_m^-\|_2}$	r_{eff}	$r_{\max}$	Δ_{ℓ_1}	$\texttt{ERR}_\Sigma$	$\texttt{TIME}$
$m = 210,\ t_m = 0.1$						
(A)	$3.5_{\mathbf{-4}}$	13.17	27	$5.7_{\mathbf{-5}}$	$2.3_{\mathbf{-5}}$	$1.07_{\mathbf{3}}$
(B)	$6.5_{\mathbf{-5}}$	12.14	25	$4.6_{\mathbf{-5}}$	$6.1_{\mathbf{-7}}$	$1.60_{\mathbf{3}}$
(C)	$1.3_{\mathbf{-1}}$	12.16	24	$2.3_{\mathbf{-3}}$	$2.1_{\mathbf{-3}}$	$9.87_{\mathbf{2}}$
(D)	$4.1_{\mathbf{-4}}$	60.06	109	$1.1_{\mathbf{-4}}$	$1.0_{\mathbf{-4}}$	$9.23_{\mathbf{3}}$
$m = M = 1332,\ t_m = T = 1$						
(A)	$1.8_{\mathbf{-4}}$	13.66	27	$7.2_{\mathbf{-5}}$	$2.5_{\mathbf{-5}}$	$3.70_{\mathbf{3}}$
(B)	$1.1_{\mathbf{-5}}$	12.06	25	$5.7_{\mathbf{-5}}$	$6.2_{\mathbf{-7}}$	$4.21_{\mathbf{3}}$
(C)	$2.5_{\mathbf{-2}}$	12.85	24	$3.3_{\mathbf{-3}}$	$1.3_{\mathbf{-3}}$	$4.03_{\mathbf{3}}$
(D)	$3.7_{\mathbf{-4}}$	58.97	107	$1.7_{\mathbf{-4}}$	$1.7_{\mathbf{-4}}$	$1.52_{\mathbf{4}}$

Table IV.3: Enzymatic futile cycle: $r_{\text{eff}} = r_{\text{eff}}[\boldsymbol{p}_m^-]$, $r_{\max} = r_{\max}[\boldsymbol{p}_m^-]$, $\Delta_{\ell_1} = \Delta_{\ell_1}\left[\sum_{E_\pm^{\text{b,f}}} \boldsymbol{p}_m^-\right]$, $\texttt{ERR}_\Sigma = \texttt{ERR}_\Sigma[\boldsymbol{p}_m^-]$ are given for the truncated solution $\boldsymbol{p}_m^-$; computational $\texttt{TIME}$ is given in seconds; $\frac{\|\boldsymbol{A}\boldsymbol{p}_0\|_2}{\|\boldsymbol{p}_0\|_2} = 5.2 \cdot 10^4$. The exponents are given in boldface for the base 10.

IV.4 Conclusion

This chapter presented a new computational method for the numerical solution of the Chemical Master equation and described numerical experiments that demonstrated its efficiency in comparison to state of the art Monte Carlo simulations. The method combines the *hp*-DG time-stepping scheme with the low-rank Quantized Tensor Train representation of the "species" space. The*hp*-DG time-stepping is both step-size and order adaptive in time forming a method that is both exponentially convergent in the number of temporal parameters and unconditionally stable. The TT representation of the "species" space allows automatic rank-adaptation of the solution, ensuring that the QTT manifold is sufficiently rich to represent the dynamics well but also as small as possible to ensure efficient computation. In this sense, this approach is superior to fixed reduced basis methods such as those described in [Deu+08; Eng09; HL07; Jah10]. While this method does particularly well when both the CME operator and the solution have a high degree of separability in both the "physical" and "virtual" levels, meaning low TT-ranks, the method does not assume or require the ranks to be small. If the solution requires large TT-ranks to represent well, the adaptivity built into the method allows this to be discovered at run-time. While the presence of large TT-ranks will decrease the efficiency of the method, other methods will do poorly for these problems as well.

While the discussion in this chapter assumed propensity functions that are monomials of the chemical species, the computational method extends easily to

propensities that can be represented in the QTT format with low ranks. For example, separable propensity function arising from stochastic mass-action and Michaelis-Menten kinetics are considered in [KS13] where QTT rank bounds are given in each case. The case where the propensities are rational functions of the chemical species was shown to be experimentally feasible here. Also, the assumption of separability of the propensities can similarly be relaxed. This requires slight (but straightforward) modification of Algorithm 9.

The speed of the method relies essentially on the efficient solution of TT-structured linear systems of equations. In this case, the local nature of the DMRG optimization-based algorithm limited the feasible step-sizes and polynomial orders of the temporal discretization. Hence, the method is unable to take full advantage of the exponential convergence of the time-stepping scheme. However, the subject of tensor-structured linear system solvers is a rapidly advancing research field and our method will become more efficient as the solvers do.

We compared our numerical results to those obtained from a state-of-the-art, massively parallel stochastic simulation package demonstrating the tremendous increase in efficiency allowed by our new approach. Using a MATLAB implementation running on a notebook, the QTT approach outperformed the Monte Carlo simulations running on 1500 cores of a high-performance cluster in the wall-clock time while computing to similar levels of accuracy in the solution.

Chapter V

QTT-Gramian-Based Model Reduction and Control of Linear Systems

Consider the continuous-time LTI system with state-space realization (A, B, C, D):

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du, \end{aligned} \tag{V.0.1}$$

where $A \in \mathbb{R}^{n\times n}$ is Hurwitz, $B \in \mathbb{R}^{n\times k}$, $C \in \mathbb{R}^{m\times n}$, and $D \in \mathbb{R}^{m\times k}$. We restrict our discussion to the case where the number of state variables greatly exceeds both the number of control inputs and the number of outputs, that is, $n >> m, k$.

The infinite-horizon observability and controllability gramians of the LTI system

are given by the formulas:

$$\begin{aligned} P_o &= \int_0^\infty e^{A^*\tau} C^* C e^{A\tau} d\tau, \\ P_c &= \int_0^\infty e^{A\tau} B B^* e^{A^*\tau} d\tau, \end{aligned} \tag{V.0.2}$$

respectively. The observability and controllability gramians are positive semidefinite matrices whose eigenspaces characterize which directions in the state-space are more observable/controllable in the L^2 sense. That is, the eigenvalues of the observability gramian measure how large the output signal of the system will be with a given initial condition and zero input, while the eigenvalues of the controllability gramian measure the minimum energy needed to drive the system from zero to a specified final state in infinite time. Since the gramians provide quantitative and geometric characterizations of the input and output behavior of the system, they are useful in many practical applications such as open-loop optimal control, optimal state-reconstruction in the presence of certain-types of measurement noise, and for model reduction.

Denote the transfer function of (A, B, C, D) by

$$G(s) := C(sI - A)^{-1}B + D. \tag{V.0.3}$$

The $\mathcal{H}_2$-norm of $G(s)$ is given by

$$||G(s)||_{\mathcal{H}_2}^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} \text{trace}[G^*(j\omega)G(j\omega)]d\omega, \tag{V.0.4}$$

while the $\mathcal{H}_\infty$-norm of $G(s)$ is given by

$$||G(s)||_{\mathcal{H}_\infty} = \sup_{\omega \in \mathbb{R}} |G(j\omega)|_2. \tag{V.0.5}$$

Both system norms will be used in the following sections as a measure of the distance between two LTI systems. Recall that the $\mathcal{H}_\infty$-norm is an induced norm while the $\mathcal{H}_2$-norm is not.

A common approach to model reduction is to reduce the dimensionality of a state-space system by restricting the dynamics to some subspace which captures most of the interesting behavior. Consider the real matrices $\mathcal{V}$ and $\mathcal{W}$ which satisfy the biorthogonality condition

$$\mathcal{W}^T\mathcal{V} = I,$$

and the system

$$\begin{aligned} \dot{z} &= \mathcal{W}^T A\mathcal{V}z + \mathcal{W}^T Bu, \\ y &= C\mathcal{V}z + Du. \end{aligned} \tag{V.0.6}$$

Suppose the state variables z can be partitioned as

$$z = \begin{bmatrix} z_1 \\ \\ z_2 \end{bmatrix},$$

and that we wish to reduce the system only to the states z_1. Partitioning the matrices $\mathcal{V}, \mathcal{W}$ conformally:

$$\mathcal{V} = \begin{bmatrix} \mathcal{V}_1 & \mathcal{V}_2 \end{bmatrix}, \quad \mathcal{W} = \begin{bmatrix} \mathcal{W}_1 & \mathcal{W}_2 \end{bmatrix},$$

results in the following equations for the state space system

$$\dot{z}_1 = \mathcal{W}_1^T A\mathcal{V}_1 z_1 + \mathcal{W}_1^T A\mathcal{V}_2 z_2 + \mathcal{W}_1^T Bu,$$

$$\begin{aligned} \dot{z}_2 &= \mathcal{W}_2^T A \mathcal{V}_1 z_1 + \mathcal{W}_2^T A \mathcal{V}_2 z_2 + \mathcal{W}_2^T B_2 u, \\ y &= C\mathcal{V}_1 z_1 + C\mathcal{V}_2 z_2 + Du. \end{aligned} \tag{V.0.7}$$

One then hopes by picking $\mathcal{W}_1$ and $\mathcal{V}_1$ "well", that z_2 is essentially zero for all time and

$$\begin{aligned} \dot{z}_1 &= \mathcal{W}_1^T A \mathcal{V}_1 z_1 + \mathcal{W}_1^T B u, \\ y &= C\mathcal{V}_1 z_1 + Du. \end{aligned} \tag{V.0.8}$$

is a good approximation of the original system V.0.1 meaning that it has approximately the same input-to-state or input-output behavior, whichever is most appropriate for the particular task.

How should one choose $\mathcal{W}_1$ and $\mathcal{V}_1$ "well"?

The *Dominant Subspace Projection* method projects the dynamics onto a subspace spanned by the eigenvectors of the controllability (observability) gramian corresponding to the largest eigenvalues. It essentially projects the system onto the most controllable (observable) modes as these are the most important in preserving the input-to-state (state-to-output) characteristics of the original system. In this case, $\mathcal{V}_1$ is chosen to be an orthonormal basis for the subspace of most controllable (observable) modes, and $\mathcal{W}_1$ is chosen to satisfy the biorthogonality condition. For example, one could choose $\mathcal{W}_1 = \mathcal{V}_1$ to obtain an orthogonal projection onto the dominant subspace.

The *Balanced Truncation* method constructs a coordinate transformation of the original state-space so that the gramians are both *equal* and *diagonal* and then trun-

cates to the directions corresponding to the dominant eigenspaces of the gramian [Moo81]. This is possible whenever the control system is both controllable and observable [DP00, Corollary 4.8]. It essentially seeks to truncate directions which are not important in describing the input-output behavior of the system. In this case, $\mathcal{W}_1$ and $\mathcal{V}_1$ both change the coordinates and truncate to the useful modes and can be computed provided one has both the controllability and observability gramians, and can compute matrix square roots (e.g. Cholesky) and singular value decompositions.

In both cases, the model reduction procedure requires the calculation of a gramian for the system. This is by far the most difficult task computationally. Selecting which time horizon to compute the gramian for depends on the application.

One approach to computing the infinite-horizon gramians involves solving certain linear matrix equations. The observability gramian solves the Continuous-Time Algebraic Lyapunov Equation (CALE) given by:

$$A^* P_o + P_o A + C^* C = 0, \tag{V.0.9}$$

while the controllability gramian solves the dual equation:

$$A P_c + P_c A^* + B B^* = 0. \tag{V.0.10}$$

Analytic solutions of these Lyapunov equations are available when the Jordan Decomposition of A is known, but this is completely impractical for most engineering systems. More practically, exact computational algorithms have been available for at least four decades now [BS72; GNVec; HAM82]. Unfortunately, these approaches

exhibit cubic scaling of the computational scaling rendering them impractical for very large-scale problems.

In the case of large-scale systems where n is very large, the previously mentioned direct numerical methods may prove computationally infeasible on available hardware. However, in many of these cases it is unnecessary to compute the full gramian; often times a few dominant singular values and singular vectors represent a good enough approximation of the gramian for the analysis or control design task at hand. Indeed, when A is Hurwitz and B or C is low-rank, the singular values of the solution of the underlying Lyapunov equation can be shown to exhibit exponential decay suggesting that an accurate low-rank approximation of the solution is possible [Gra04].

Many computational methodologies have been developed over the last two decades that seek to exploit this fact by looking for solutions in *rank-revealing* matrix formats, e.g. Cholesky, SVD. By restricting the search space to matrices of low-rank expressed in one of these formats, many proposed algorithms have achieved significant computational savings [GH07; Pen99].

This chapter describes an approach to finding low-rank approximations of the gramians using the TT-format and the DMRG based linear system solver and their use in model reduction of LTI systems. Section V.1 describes a (CALE) solver that takes advantage of the TT and DMRG technologies and discusses the TT-rank structure of the solution. Section V.2 describes the Dominant Subspace Projection Open Loop Control (DSPOLC) procedure which is a computationally tractable method for

steering and LTI system from the origin to a desired target state. Section V.3 describes an implementation of the well-known Balanced Truncation procedure using TT-formatted arrays. Section V.4 describes numerical experiments which demonstrate the feasibility of these approaches.

V.1 Solution of Lyapunov Equations in the TT-Format

This section describes a numerical solver for CALE exploiting the TT-format and the DMRG Linear System Solver. Subsection V.1.1 describes the tensor structure of the Lyapunov operator and computes an upper bound on its TT-ranks. Subsection V.1.2 describes the DMRG-based solver. Subsection V.1.1 applies the solver to an example of a controlled reaction-diffusion system and tests the efficiency of the diagonalization procedures described in the previous chapter.

V.1.1 Tensor Structure of Lyapunov Equations

Let $\boldsymbol{A}$ be a d-level matrix. The *Lyapunov Operator* $\boldsymbol{\mathcal{L}_A}$ corresponding to $\boldsymbol{A}$ is given by

$$\boldsymbol{\mathcal{L}_A}(\boldsymbol{X}) = \boldsymbol{AX} + \boldsymbol{XA^*}$$

It is a multilinear operator on the space of d-level matrices.

By considering the vectorization of (V.1.1), we can express the left-hand side in

terms of a matrix-vector product of a sum of tensor products and the $2d$-level vector $\boldsymbol{X}$:

$$(\boldsymbol{A} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{A})\boldsymbol{X} = -\boldsymbol{Q}, \tag{V.1.1}$$

where $\boldsymbol{I}$ is the d-level identity matrix with mode sizes identical to $\boldsymbol{A}$. We therefore have that the $2d$-level matrix representation of $\boldsymbol{\mathcal{L}_A}$ is given by

$$\boldsymbol{\mathcal{L}_A} = \boldsymbol{A} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{A}. \tag{V.1.2}$$

While the process of generating the vectorized equation (V.1.1) is often a first step in analyzing the properties of the equation and solution [HAM82] , numerically computing the solutions by directly solving the linear system is generally a very difficult task in full format and is universally avoided. For a simple LTI system with n states, $\boldsymbol{A}$, $\boldsymbol{Q}$, and $\boldsymbol{X}$ will have n^2 entries, while the full matrix representation of $\boldsymbol{\mathcal{L}_A}$ has n^4 terms. Even when $\boldsymbol{A}$ and $\boldsymbol{Q}$ are sparse or have special structure, it is usually not the case that the solution $\boldsymbol{X}$ also has the special structure. However, by representing the vectors and matrices involved in a low-parametric format with fast arithmetics (like QTT) we can formulate efficient numerical algorithms using this vectorized equation. The use of an efficient representation format allows previously intractable approaches to problems to be fast and scalable.

Assuming a characterization of the TT-ranks of $\boldsymbol{A}$ is available, the following theorem describes the TT-ranks of $\boldsymbol{\mathcal{L}_A}$ in the TTM format.

Theorem V.1.1. *Suppose $\boldsymbol{A}$ has a TTM decomposition with TT-ranks $r_1, \ldots, r_{d-1}$.*

The Lyapunov Operator has a TTM decomposition with ranks bound from above by

$$r_1 + 1, \ldots, r_{d-1} + 1, 2, r_1 + 1, \ldots, r_{d-1} + 1.$$

Proof. The identity matrix has an explicit rank-1 TTM decomposition. From (V.1.1) we see that $\boldsymbol{\mathcal{L}}$ is the sum of two parts: $\boldsymbol{A} \otimes \boldsymbol{I}$ which has TTM-ranks

$$r_1, \ldots, r_d, 1, \ldots, 1,$$

and $\boldsymbol{I} \otimes \boldsymbol{A}$ which has QTT matrix ranks

$$1, \ldots, 1, r_1, \ldots, r_d,$$

The ranks of the sum are bound above by the sum of the ranks which completes the proof. □

When the ranks of $\boldsymbol{A}$ are small in the TTM format, then so are the ranks of $\boldsymbol{\mathcal{L}_A}$. In the next subsection, we discuss the TT structure of the solution using this vectorized approach.

V.1.2 Solution of Lyapunov Equations in the TT-Format

We propose using the DMRG-based linear system solver [DO11] to solve the TT-structured linear system corresponding to (V.1.1). Given a system in which $\boldsymbol{\mathcal{L}_A}$ is expressed in TTM format and $\boldsymbol{Q}$ is expressed in TTVM format and a relative

tolerance on the Frobenius norm of the residual error, it returns a TTVM formatted solution of the Lyapunov equation.

A key problem in designing optimization based solvers for low parametric tensor formats is determining the sizes of the compression ranks needed for an accurate solution. When the ranks are too small, it may be impossible to achieve a given error tolerance. When the ranks are larger than necessary, then solving the local problems in the optimization are more computationally expensive than necessary. The DMRG based solver effectively balances these two problems by adaptively growing and shrinking the TT ranks of the candidate solution by merging and then splitting cores based on the low rank approximation of matrices [DO11].

While a convergence theory for the DMRG solver is still lacking, it has proven highly efficient in practical applications where the TT-ranks of the solution are small. In the case where $\boldsymbol{A}$ is Hurwitz and $\boldsymbol{Q}$ has low rank, [Gra04] has shown that the singular values of $\boldsymbol{X}$ decay exponentially. In terms of the controllability and observability gramians, this would correspond to LTI systems that are not very controllable or observable. One also desires that the singular vectors may be represented with low TT-ranks.

V.1.3 Numerical Experiments

Consider the following reaction-diffusion equation on the hypercube $D = (-\pi, \pi)^d$ with point control and Dirichlet boundary conditions:

$$\begin{cases} \partial_t \boldsymbol{\psi}(x,t) = c_1 \Delta \boldsymbol{\psi}(x,t) - c_2 \boldsymbol{\psi}(x,t) + \delta(x) u(t), \\ \qquad\qquad x \in D, \\ \boldsymbol{\psi}(x,t) = 0, \quad x \in \partial D \end{cases} \tag{V.1.3}$$

where $c_1, c_2 > 0$, $u(t)$ is a control input and $\delta(x)$ is the Dirac delta function centered at zero. This equation models a d-dimensional reaction vessel in which the only reaction is spontaneous degradation. $\boldsymbol{\psi}(x,t)$ describes the concentration of a particular chemical reactant at spatial coordinate x and at time t. The control signal allows the injection or removal of the substance at the point $x = 0$.

In this section, we consider versions of (V.1.3) that have been discretized using a finite difference scheme in space on a uniform tensor grid with spacing h but no discretization in time (Method of Lines). We use a second-order central difference to approximate the Laplacian and approximate the Dirac delta as a Kronecker delta function at the nearest grid point with unit mass. Let $\hat{\boldsymbol{\psi}}(x,t)$ denote the discrete approximation of $\boldsymbol{\psi}(x,t)$. The time evolution of the discretized system is given by the finite-dimensional LTI system:

$$\partial_t \hat{\boldsymbol{\psi}}(x,t) = A\hat{\boldsymbol{\psi}}(x,t) + Bu(t), \tag{V.1.4}$$

with

$$A = \frac{c_1}{h^2}\Delta_{dd} - c_2 I, \quad B = \frac{1}{h^d}\hat{\delta}(x),$$

where Δ_{dd} is the discrete Laplacian on a rectangular grid with Dirichlet boundary conditions and $\hat{\delta}(x)$ is a vector that is zero everywhere except at the entry corresponding to the grid point closest to the origin.

Kazeev, *et al.* showed that using the finest possible quantization, Δ_{dd} has an explicit QTT representation with all QTT ranks < 4 [KK12]. Hence, A has QTT-ranks < 5. Also, B is rank 1 separable after quantization so that all the matrices and vectors involved have low QTT-ranks.

The controllability Gramian of the discretized system is expected to have low QTT rank for two reasons. First, the matrix B has rank one so the singular values of the Gramian can be expected to decay rapidly (the system is not very controllable). A low-rank approximation using only the first few singular values and vectors can be expected to approximate the solution well. Secondly, the first few singular vectors of the Gramian of the original system can be expected to have a high degree of regularity so that at fine levels of discretization, they can be well approximated by low order polynomials. Hence, the singular vectors of the approximate Gramian can be expected to have low QTT ranks. While it is possible to use another compression scheme other than quantization (e.g. Galerkin projection onto tensorized Legendre polynomials), this would require *a priori* a choice of grid, polynomial orders, etc for the compression. By using the QTT numerical linear algebra, the compression is performed *automatically and on the fly*.

In the following, we implemented our proposed algorithms in MATLAB using

the *TT Toolbox* implementation of the TT and QTT tensor formats, publicly available at `http://spring.inm.ras.ru/osel`. All calculations were performed in MATLAB 7.11.0.584 (R2010b) on a laptop with a 2.7 GHz dual-core processor with 12 GB RAM.

Our implementation relies crucially on the DMRG Linear System Solver available as **dmrg_solve2** in the *TT Toolbox*. While a rigorous convergence analysis of the DMRG solver is still missing, we find in our examples that it can be highly efficient.

V.1.3.1 Testing the DMRG Solver

We used the proposed DMRG-based solver to compute a low-rank approximate Gramian $\hat{W}_c$ and compare it to a solution computed in full format using the MATLAB Control System Toolbox's **lyap** function which we treat as the true solution. In each case, we ran the DMRG-based solver until the following accuracy condition was met:

$$||\hat{P}_c - P_c||_2 < 10^{-9}$$

where $||\cdot||_2$ denotes the induced 2-norm. Practically, this required careful tuning of the DMRG-based linear system solver residual error tolerance to produce solutions of the desired accuracy (and no more). In each case, the DMRG solver was initialized with a random rank-2 QTT vector, though it is possible in practical problems to initialize in a smarter way.

Figure V.1 plots the computation time required to compute the Gramian in both the full and the QTT formats for the 1-D version of the problem over a range

of discretization levels. Compute time in the full format scales linearly with the number of discretization points while compute time for the QTT version does not. Instead, the QTT based algorithm depends critically on the ranks of the solution. Interestingly, on finer grids, the compute time actually decreases since the solution can be well represented with tensors having much smaller QTT-ranks. While the full format solution is faster for coarse levels of discretization, we emphasize that the *scaling* of the QTT approach is much more favorable.

Figure V.2 plots the computation time required to compute the Gramian in both the full and the QTT formats where the discretization level in each dimension is kept uniform but the number of dimensions is scaled. In each case, 2^4 discretization points are used in each dimension. The compute time for the full format grows rapidly with the number of dimensions while the compute time for the QTT version remains bounded. Again, the QTT based algorithm depends critically on the QTT-ranks of the solution. Three dimensions was the maximum number that our compute hardware could handle for the full format due to the exponential increase in storage requirements but dimensions as high as ten were successfully computed in the QTT format using a residual tolerance $< 10^{-9}$ in less than 5 minutes for each problem.

V.1.3.2 A Large Scale Problem

We next tested our proposed methods on a large-scale version of the problem. In 2-D we took $2^{10} = 1024$ grid points in each direction resulting in a discretized version

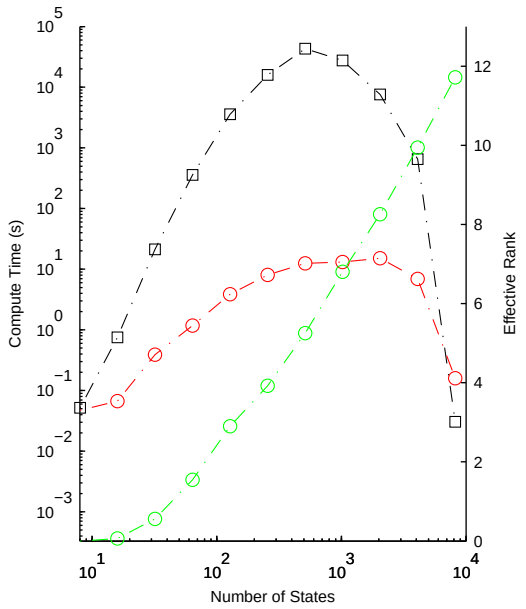


Figure V.1: DMRG Compute Time and Effective Rank vs. Number of States for discretized 1-D reaction-diffusion model. The compute time for the full format solution (green) scales linearly with the number of states, while it remains small for the proposed method (red). Black indicates the effective rank of the solution obtained by the proposed method. At finer discretizations, approximations with lower effective QTT-rank approximate the full solution to the same accuracy tolerance.

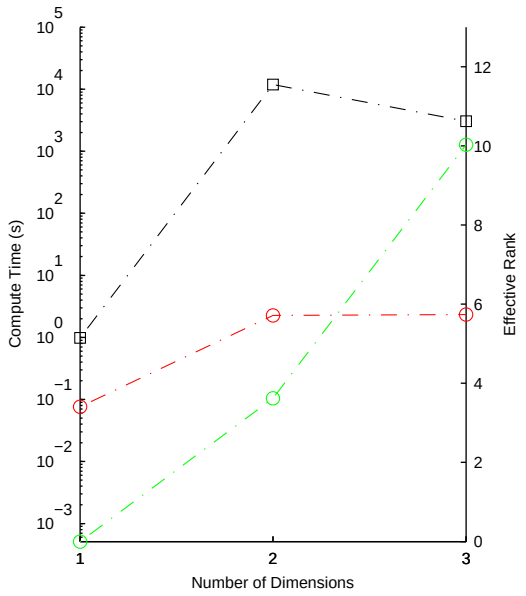


Figure V.2: DMRG Compute Time and Effective Rank vs. Number of Dimensions for discretized reaction-diffusion models where the number of physical dimensions scales. The compute time for the full format solution (green) increases rapidly with the number of dimensions, while it increases less quickly for the proposed method (red). Effective QTT rank of the approximate solution appears in black.

of the problem with $2^{20} > 1.04$ million states. In full format, the corresponding A matrix has 2^{40} entries, though it is highly structured with only 5.24 million nonzero elements. However, even when A has a lot of sparse structure, it is rare that the Controllability Gramian inherits this structure. In this example the Gramian would require storage of 2^{40} entries or over 3 TB of storage using 32-bit floating precision. However, the rank adaptiveness of the DMRG solver combined with the high degree of QTT structure in the solution means that it is no problem for our proposed approach.

Using a residual tolerance smaller than 10^{-9} for the DMRG solver and allowing it to compute as many iterations as needed for convergence, the solver took 67.5 sec to converge to a solution in QTT-VM format with effective rank=29.09 and only 67,680 parameters needed to specify it (a compression ratio of seven orders of magnitude). The (matrix) rank of the approximate Gramian was 16.

We then used the QTT Lanczos algorithm to compute approximate dominant eigenvalues. We performed 30 iterations and accepted eigenvectors with a high degree of symmetry under exchange of the two physical dimensions due to the symmetry in the problem, see Figure V.5. Figure V.4 plots the eigenvalues produced by the algorithm. Figure V.3 plots the cumulative compute time for the QTT Lanzcos Iteration as well as the time needed to reassemble the eigenvectors of $\hat{P}_c$ from those of T_{mm}. The total run time of the QTT Lanczos Algorithm was 2.79 hours. Orthogonality of the iterates was lost very quickly so that the algorithm produced multiple copies of these eigenvectors. The algorithm also produced several badly corrupted vectors

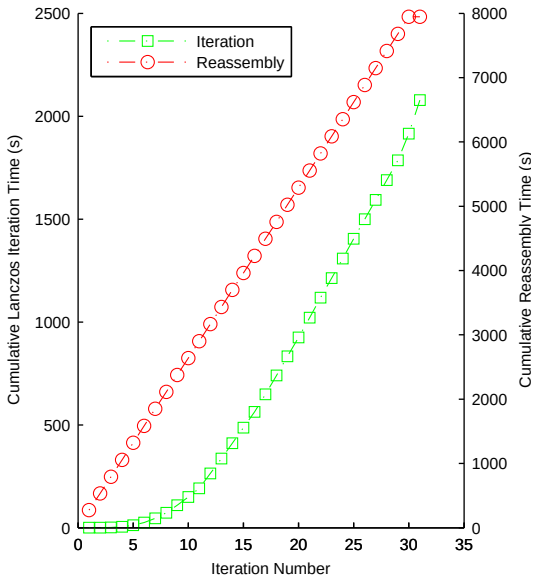


Figure V.3: Compute time for the proposed QTT Lanczos Algorithm for the large-scale problem. 30 iterations were performed.

that lacked the symmetry; especially in the last few iterations, see Figure V.6. These effects are typical of the classical Lanczos Algorithm. The former problem could be mitigated using modern heuristics such as random restarts, better orthogonalization procedures, etc. and the latter issue can be resolved by performing more iterations.

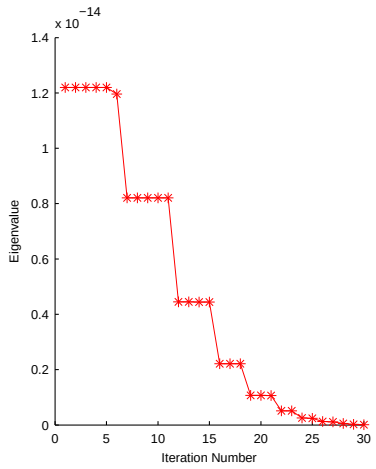
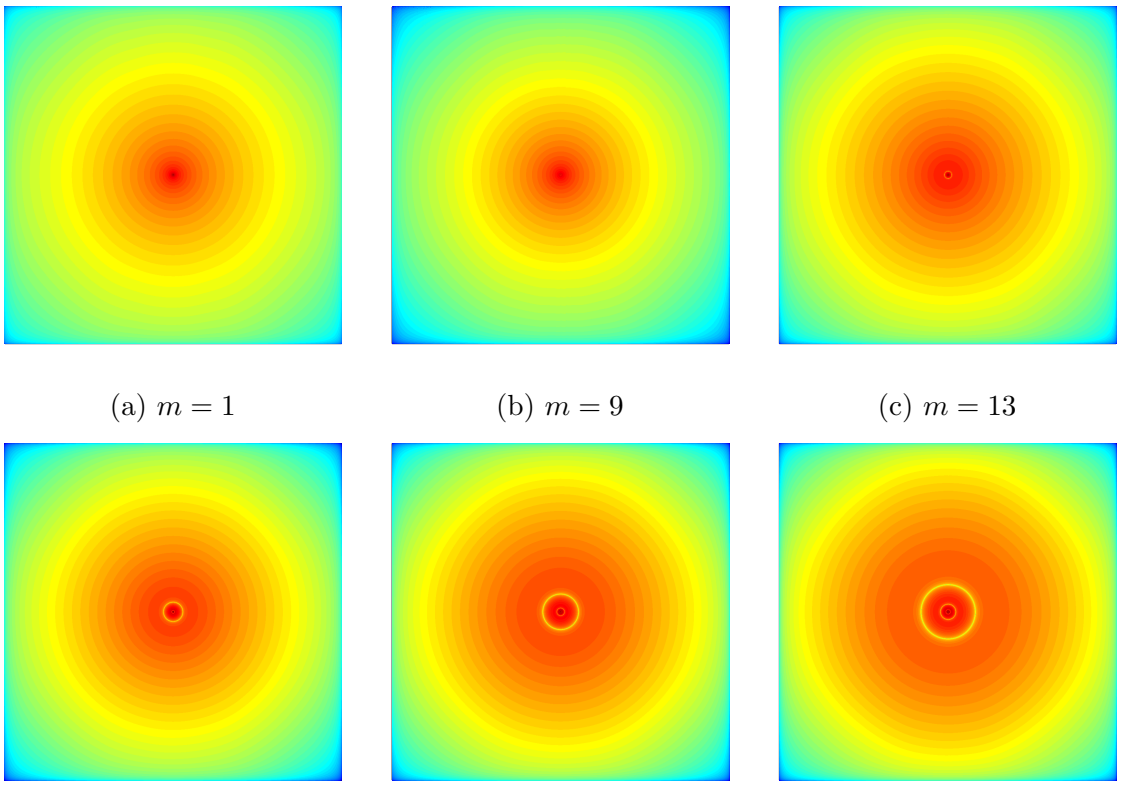


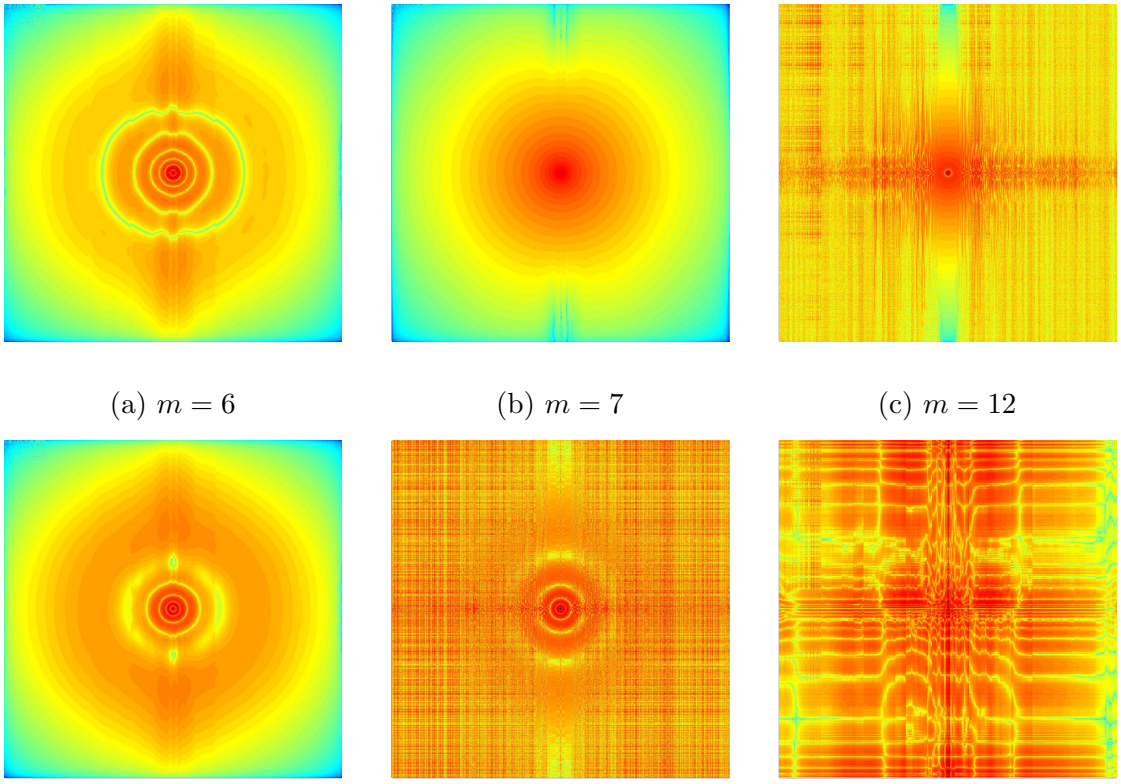
Figure V.4: Eigenvalues produced by the proposed QTT Lanczos Algorithm for the large-scale problem. 30 iterations were performed. Many eigenvalues repeat, especially in the first twenty iterations. This reflects the loss of orthogonality through the iteration process.



(a) $m = 1$ (b) $m = 9$ (c) $m = 13$

(d) $m = 16$ (e) $m = 19$ (f) $m = 22$

Figure V.5: Approximate eigenvectors of the Controllability Gramian produced by QTT Lanczos Iteration for the 2-D discretized reaction-diffusion equation with point control.



(a) $m = 6$ (b) $m = 7$ (c) $m = 12$

(d) $m = 26$ (e) $m = 28$ (f) $m = 29$

Figure V.6: Spurious eigenvectors produced by QTT Lanczos Iteration for the 2-D discretized reaction-diffusion equation with point control.

V.2 DSPOLC

In this section we describe a computationally efficient method for computing open loop control laws for steering a large-scale LTI system from the origin to a point in the controllable subspace known as Dominant Subspace Projection Open Loop Control (DSPOLC). The computational methodology makes heavy use of the TT numerical linear algebra discussed in III.2.4 though for the sake of clarity the details will be suppressed.

Suppose the controllability gramian P_c is invertible. The optimal input to steer the system to x_0 is given by

$$u_{opt} := \Psi_c^* P_c^{-1} x_0 = \begin{cases} B^* e^{-A^* t} P_c^{-1} x_0 & \text{for } t \leq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{V.2.1}$$

in the sense that this signal has the minimum energy of all such signals that drive the system to the desired point.

The bilateral Laplace transform of this input signal is given by

$$H(s) = -B^*(sI + A^*)^{-1} P_c^{-1} x_0, \qquad Re(s) < -Re(\sigma(A^*)) \tag{V.2.2}$$

with Region of Convergence including the $j\omega$ axis as long as A is Hurwitz. In the packed notation for LTI systems:

$$H(s) = \left[\begin{array}{c|c} -A^* & P_c^{-1} x_0 \\ \hline -B^* & 0 \end{array}\right]. \tag{V.2.3}$$

Now, we are interested in cases where we only have access to the SVD of a $\hat{r}_c$-rank approximation of the controllability gramian $\hat{V}_c \hat{\Sigma}_c \hat{V}_c^T = \hat{P}_c$. The columns of $\hat{V}_c$ form

an orthonormal basis for the $\hat{r}_c$-dimensional dominant subspace of the controllable subspace. We wish to design a control law that only requires computation on this reduced subspace.

Define the following projected open loop control law:

$$\hat{u}(t) := \Psi_c^* \hat{P}_c^{-1} x_0 = \begin{cases} B^* \hat{V}_c e^{-(\hat{V}_c^T A^* \hat{V}_c)t} P_c^{-1} x_0 & \text{for } t \leq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{V.2.4}$$

which is based on projecting the dynamics onto the $\hat{r}_c$-dimensional dominant subspace. The bilateral Laplace transform:

$$\hat{H}(s) = -B^* \hat{V}_c (sI + \hat{V}_c^T A^* \hat{V}_c)^{-1} P_c^{-1} x_0, \qquad Re(s) < -Re(\sigma(V_c^T A^* V_c)) \tag{V.2.5}$$

has a Region of Convergence which contains the $j\omega$-axis as long as $V_c^T A V_c$ is Hurwitz. In the packed notation:

$$\hat{H}(s) = \left[\begin{array}{c|c} -\hat{V}_c^T A^* \hat{V}_c & \hat{V}_c^T \hat{P}_c^{-1} x_0 \\ \hline -B^* \hat{V}_c & 0 \end{array}\right]. \tag{V.2.6}$$

Equation (V.2.4) is computationally advantageous since, for each target state x_0, $\hat{u}(t)$ may be computed by simulation of a linear system with at most $\hat{r}_c$ states. Define the discrepancy between the optimal input signal and the projected input signal in the frequency domain by

$$E_u(s) = H(s) - \hat{H}(s). \tag{V.2.7}$$

Proposition V.2.1 (Lemma 2 in [SS05]). *Suppose $\hat{P}_c$ is the $\hat{r}_c$-rank approximation of the controllability gramian with SVD $\hat{P}_c = \hat{V}_c\hat{\Sigma}_c\hat{V}_c^T$. Then the discrepancy between the optimal and projected inputs as defined by* (V.2.7) *satisfies the following:*

$$E_u(s) = L(s)(I - \hat{V}_c\hat{V}_c^T)F(s), \tag{V.2.8}$$

where $L(s) = -B^*(sI - A^*)^{-1}$ and $F(s) = -(A^*\hat{V}_c(sI + \hat{V}_c^T A^*\hat{V}_c)^{-1}\hat{V}_c^T\hat{P}_c^{-1} - P_c^{-1})x_0$.

Proof. Considering $E_u^T(s)$ in the packed notation,

$$E_u^T(s) = \left[\begin{array}{cc|c} -\hat{V}_c^T\bar{A}\hat{V}_c & 0 & \hat{V}_c^T\bar{B} \\ 0 & -\bar{A} & -\bar{B} \\ \hline x_0^T\hat{P}_c^{-1}\hat{V}_c & x_0^T P_c^{-1} & 0 \end{array}\right]. \tag{V.2.9}$$

where $\bar{A}, \bar{B}, \bar{C}$ are the matrices whose entries are the complex conjugates of A, B, C, respectively. Using the coordinate transformation

$$\begin{bmatrix} I & \hat{V}_c^T \\ 0 & I \end{bmatrix},$$

leads to

$$E_u^T(s) = \left[\begin{array}{cc|c} -\hat{V}_c^T\bar{A}\hat{V}_c & -\hat{V}_c^T\bar{A}(I - \hat{V}_c\hat{V}_c^T) & 0 \\ 0 & -\bar{A} & -\bar{B} \\ \hline x_0^T\hat{P}_c^{-1}\hat{V}_c & x_0^T(P_c^{-1} - \hat{P}_c^{-1}\hat{V}_c\hat{V}_c^T) & 0 \end{array}\right].$$

Observing that $\hat{P}_c^{-1}\hat{V}_c\hat{V}_c^* = P_c^{-1}\hat{V}_c\hat{V}_c^*$, this transfer function has a state-space realization

$$\dot{\eta} \;=\; -\hat{V}_c^T\bar{A}\hat{V}_c\eta - \hat{V}_c^T\bar{A}(I - \hat{V}_c\hat{V}_c^T)\xi$$

$$\begin{aligned}\dot{\xi} &= -\bar{A}\xi - \bar{B}u \\ e &= x_0^T \hat{P}_c^{-1}\eta + x_0^T P_c^{-1}(I - \hat{V}_c\hat{V}_c^T)\xi\end{aligned}$$

By inspection, this system also admits the transfer function

$$E_u^T(s) = F^T(s)(I - \hat{V}_c\hat{V}_c^T)L^T(s) \tag{V.2.10}$$

where

$$L^T(s) = -(sI + \bar{A})^{-1}\bar{B}, \qquad F^T(s) = -x_0^T(\hat{P}_c^{-1}\hat{V}_c(sI + \hat{V}_c^T\bar{A}\hat{V}_c)^{-1}\hat{V}_c^T\bar{A} - P_c^{-1}). \tag{V.2.11}$$

Transposing both sides of equation (V.2.10) yields the desired result. □

The previous lemma allows the estimation of $||E_u||_{\mathcal{H}_2}$ in terms of the truncated singular values of the controllability gramian.

Proposition V.2.2 (Theorem 1 in [SS05])**.** *Suppose $\hat{V}_c^T A\hat{V}_c$ is Hurwitz.*

$$||E_u||_{\mathcal{H}_2} \leq ||F||_{\mathcal{H}_\infty}\left(\sum_{i=\hat{r}_c+1}^{n}\sigma_i\right)^{1/2}, \tag{V.2.12}$$

where $F(s)$ is given by (V.2.11)*.*

Proof. The $\mathcal{H}_\infty$-norm of $E_u(s)$ is given by

$$\begin{aligned}||E_u||_{\mathcal{H}_2}^2 &= \frac{1}{2\pi}\int_{-\infty}^{\infty}\text{trace}[E_u(j\omega)E_u^*(j\omega)]d\omega \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty}\text{trace}[E_u^*(j\omega)E_u(j\omega)]d\omega \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty}\text{trace}[F^T(-j\omega)(I - \hat{V}_c\hat{V}_c^T)L^T(-j\omega)L(j\omega)(I - \hat{V}_c\hat{V}_c^T)F(j\omega)]d\omega\end{aligned}$$

$$\leq \quad ||F||^2_{\mathcal{H}_\infty} \frac{1}{2\pi} \int_{-\infty}^{\infty} \text{trace}[(I - \hat{V}_c \hat{V}_c^T) L^T(-j\omega) L(j\omega)(I - \hat{V}_c \hat{V}_c^T)] d\omega$$

since $(I - \hat{V}_c \hat{V}_c^T) L^T(-j\omega) L(j\omega)(I - \hat{V}_c \hat{V}_c^T)$ is Hermitian, positive-semidefinite and F has finite $\mathcal{H}_\infty$-norm since $\hat{V}_c^T A \hat{V}_c$ is Hurwitz. Using linearity of the trace:

$$\begin{aligned} ||E_u||^2_{\mathcal{H}_2} &\leq ||F||^2_{\mathcal{H}_\infty} \text{tr} \left[(I - \hat{V}_c \hat{V}_c^T) \left(\frac{1}{2\pi} \int_{-\infty}^{\infty} L^T(-j\omega) L(j\omega) d\omega \right) (I - \hat{V}_c \hat{V}_c^T) \right] \\ &= ||F||^2_{\mathcal{H}_\infty} \text{tr} \left[(I - \hat{V}_c \hat{V}_c^T) P_c (I - \hat{V}_c \hat{V}_c^T) \right] \\ &= ||F||^2_{\mathcal{H}_\infty} \left(\sum_{i=\hat{r}_c+1}^{n} \sigma_i \right) \end{aligned}$$

□

Formally, the projected open loop control law defined by (V.2.4) may have unbounded support meaning that it would take an infinite amount of time to steer the system to the desired target state. However, as long as $\hat{V}_c^T A \hat{V}_c$ is Hurwitz, the exponential stability ensures that we can find a good approximation with bounded support, i.e., a control law that can be implemented in finite time. Define the truncated projected control law by:

$$\hat{u}_T(t) := \Psi_c^* \hat{P}_c^{-1} x_0 = \begin{cases} B^* \hat{V}_c e^{-(\hat{V}_c^T A^* \hat{V}_c)t} P_c^{-1} x_0 & \text{for } t \in [-T, 0], \\ 0 & \text{otherwise.} \end{cases} \tag{V.2.13}$$

where $T > 0$ should be chosen sufficiently large so as to capture most of the control energy. The bilateral Laplace transform of (V.2.13) is given by

$$\hat{H}_T(s) = -B^* \hat{V}_c (sI + \hat{V}_c^T A^* \hat{V}_c)^{-1} \left(I - e^{(sI + \hat{V}_c^T A^* \hat{V}_c)T} \right) P_c^{-1} x_0.$$

Define the difference between the projected and truncated projected control laws to be

$$E_{\hat{u}} := \hat{H}(s) - \hat{H}_T(s). \tag{V.2.14}$$

By taking T to be sufficiently large, the $\mathcal{H}_2$-norm of $E_{\hat{u}}$ can be made as small as desired.

Proposition V.2.3. *Assume $\hat{V}_c^T A \hat{V}_c$ is Hurwitz. For any $\varepsilon > 0$, there exists a $T_\varepsilon > 0$ such that $||E_{\hat{u}}||_{\mathcal{H}_2}^2 < \varepsilon$.*

Proof. Fix $\varepsilon > 0$. Using Parseval's identity, the $\mathcal{H}_2$-norm can be rewritten in the time-domain as

$$\begin{aligned} ||E_{\hat{u}}||_{\mathcal{H}_2}^2 &= \int_{-\infty}^{0} (\hat{u}(t) - \hat{u}_T(t))^*(\hat{u}(t) - \hat{u}_T(t))dt \\ &= \int_{-\infty}^{-T} |B^* \hat{V}_c e^{-(\hat{V}_c^T A^* \hat{V}_c)t} P_c^{-1} x_0|^2 dt \end{aligned}$$

Since $\hat{V}_c^T A \hat{V}_c$ is Hurwitz, there exists constants $c > 0$, $\lambda > 0$ such that for any $v \in \mathbb{R}^n$, $|e^{\hat{V}_c^T A^* \hat{V}_c t} v| \leq c e^{-\lambda t}|v|$, for all $t \geq 0$.

$$\begin{aligned} ||E_{\hat{u}}||_{\mathcal{H}_2}^2 &\leq \int_{T}^{\infty} ||B^* \hat{V}_c||^2 |P_c^{-1} x_0|^2 c^2 e^{-2\lambda t} dt \\ &= \frac{c^2 ||B^* \hat{V}_c||^2 |P_c^{-1} x_0|^2}{2\lambda} e^{-2\lambda T} \end{aligned}$$

Taking

$$T_\varepsilon = -\frac{1}{\lambda} \ln \left(\frac{\varepsilon \sqrt{2\lambda}}{c ||B^* \hat{V}_c|| |P_c^{-1} x_0|} \right),$$

ensures the desired inequality. □

For simplicity, we will refer to the truncated projected open loop control law as the approximate control law. The details of the TT-DSPOLC computational procedure are summarized in Algorithm 12.

Algorithm 12 TT-DSPOLC

Require: LTI control system $(\boldsymbol{A}, \boldsymbol{B})$ with matrix $\boldsymbol{A}$ in the TTM format, matrix $\boldsymbol{B}$ in the TTVM format, and target state $\boldsymbol{x}_0$.

Ensure: Approximate control law $\boldsymbol{u}(t)$ which drives the system from $\boldsymbol{x} = 0$ near $\boldsymbol{x} = \boldsymbol{x}_0$.

Compute approximate infinite-time horizon gramian $\hat{\boldsymbol{W}}_c$ using TT-DMRG CALE solver,

$[\hat{\boldsymbol{U}}_c, \hat{\boldsymbol{D}}_c, \hat{\boldsymbol{V}}_c^T] = \text{TT\_SVD}(\hat{\boldsymbol{W}}_c)$,

$\hat{\boldsymbol{W}}_c^{-1} = \hat{\boldsymbol{U}}_c \hat{\boldsymbol{D}}_c^{-1} \hat{\boldsymbol{V}}_c^T$,

$\hat{\boldsymbol{A}} = \hat{\boldsymbol{V}}_c^T \boldsymbol{A} \hat{\boldsymbol{V}}_c$, $\hat{\boldsymbol{B}} = \hat{\boldsymbol{V}}_c \boldsymbol{B}$

Compute solution $\psi(t)$ to Initial Value Problem: $\dot{\boldsymbol{\psi}} = \hat{\boldsymbol{A}}\psi$, $\boldsymbol{\psi(0)} = \hat{\boldsymbol{W}}_c^{-1} \boldsymbol{x}_0$ up to time T,

$\boldsymbol{u}(t) = \hat{\boldsymbol{B}}_c^T \boldsymbol{\psi}(-t)$.

We can now state both pointwise and $\mathcal{L}_2$ error estimates for the trajectory resulting from the DSPOLC procedure. In particular, the pointwise estimate gives an error bound on how far the system ends up from the target state in terms of the sum of the truncated singular values of the controllability gramian.

Let $G(j\omega) = (j\omega I - A)^{-1} B$ be the transfer function of the full system.

Theorem V.2.4 (Error Estimates for the Trajectory Resulting From the Approximate Control Law)**.** *Suppose A and $\hat{V}_c^T A \hat{V}_c$ are Hurwitz and that $||E_{\hat{u}}||_{\mathcal{H}_2}^2 < \varepsilon$. The difference between the optimal trajectory and the approximate trajectory satisfies the following pointwise and $\mathcal{L}_2$ error estimates*

(a)

$$|e_x(t)| \leq \frac{1}{\sqrt{2\pi}} ||G||_{\mathcal{H}_\infty} \left(||F||_{\mathcal{H}_\infty} \left(\sum_{i=\hat{r}_c+1}^{n} \sigma_i \right)^{1/2} + \varepsilon \right), \qquad t \leq 0, \qquad \text{(V.2.15)}$$

(b)

$$||e_x(t)||_{\mathcal{L}_2} \leq ||G||_{\mathcal{H}_\infty} \left(||F||_{\mathcal{H}_\infty} \left(\sum_{i=\hat{r}_c+1}^{n} \sigma_i \right)^{1/2} + \varepsilon \right), \qquad \text{(V.2.16)}$$

where $F(s)$ is as defined in (V.2.11)*.*

Proof. Since A and $\hat{V}_c^T A \hat{V}_c$ are Hurwitz, the Fourier transform of the error is given by:

$$E_x(j\omega) = G(j\omega)(E_u(j\omega) + E_{\hat{u}}(j\omega)),$$

For part V.2.4, take the inverse Fourier transform and compute the magnitude:

$$\begin{aligned} |e_x(t)|^2 &= \left| \frac{1}{2\pi} \int_{-\infty}^{\infty} G(j\omega)(E_u(j\omega) + E_{\hat{u}}(j\omega)) e^{j\omega t} d\omega \right|^2 \\ &\leq \left(\frac{1}{2\pi} \right)^2 \int_{-\infty}^{\infty} \left| G(j\omega)(E_u(j\omega) + E_{\hat{u}}(j\omega)) e^{j\omega t} \right|^2 d\omega \end{aligned}$$

by Jensen's inequality. Using the submultiplicative property of the induced 2-norm

$$|e_x(t)|^2 \leq \left(\frac{1}{2\pi} \right)^2 \int_{-\infty}^{\infty} |G(j\omega)|^2 \, |E_u(j\omega) + E_{\hat{u}}(j\omega)|^2 \, d\omega.$$

Since A is Hurwitz, the $\mathcal{H}_\infty$-norm of $G(j\omega)$ is finite. Taking the supremum of $|G(j\omega)|$ over all ω:

$$\begin{aligned}
|e_x(t)|^2 &\leq \left(\frac{1}{2\pi}\right)^2 ||G(j\omega)||^2_{\mathcal{H}_\infty} \int_{-\infty}^{\infty} |E_u(j\omega) + E_{\hat{u}}(j\omega)|^2 \, d\omega \\
&= \frac{1}{2\pi} ||G(j\omega)||^2_{\mathcal{H}_\infty} ||E_u(j\omega) + E_{\hat{u}}(j\omega)||^2_{\mathcal{H}_2} \\
&\leq \frac{1}{2\pi} ||G(j\omega)||^2_{\mathcal{H}_\infty} \left(||E_u(j\omega)||_{\mathcal{H}_2} + ||E_{\hat{u}}(j\omega)||_{\mathcal{H}_2}\right)^2
\end{aligned}$$

yielding the desired result.

For part b, use Parseval's identity to express the $\mathcal{L}_2$-norm of the error as

$$\begin{aligned}
||e_x(t)||^2_{\mathcal{L}_2} &= \int_{-\infty}^{\infty} e_x^T(\tau) e_x(\tau) d\tau \\
&= \frac{1}{2\pi} \int_{-\infty}^{\infty} E_x^*(j\omega) E_x(j\omega) d\omega \\
&= \frac{1}{2\pi} \int_{-\infty}^{\infty} (E_u(j\omega) + E_{\hat{u}}(j\omega))^* G^*(j\omega) G(j\omega) (E_u(j\omega) + E_{\hat{u}}(j\omega)) d\omega.
\end{aligned}$$

Since A is Hurwitz, the $\mathcal{H}_\infty$-norm of $G(j\omega)$ is finite. Applying the submultiplicative property of the norm, the last expression can be bounded by

$$\begin{aligned}
||e_x(t)||^2_{\mathcal{L}_2} &\leq \frac{1}{2\pi} \int_{-\infty}^{\infty} |T(j\omega)|^2 |E_u(j\omega) + E_{\hat{u}}(j\omega)|^2 d\omega \\
&\leq ||G(j\omega)||^2_{\mathcal{H}_\infty} ||E_u(j\omega) + E_{\hat{u}}(j\omega)||^2_{\mathcal{H}_2} \\
&\leq ||G(j\omega)||^2_{\mathcal{H}_\infty} \left(||F||_{\mathcal{H}_\infty} \left(\sum_{i=\hat{r}_c+1}^{n} \sigma_i \right)^{1/2} + \varepsilon \right)^2,
\end{aligned}$$

obtaining the desired result. □

We emphasize that the error estimates assume projection onto the dominant eigenvectors of the *exact* gramian. We leave a characterization in terms of the approximate gramian to future work.

V.3 Balanced Truncation

In the balanced coordinates, the controllability and observability gramians are equal and diagonal with eigenvalues

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n > 0,$$

which are equal to the *Hankel singular values* of the system. The Hankel singular values relate the energy of the input signal to the energy of the output signal. The larger the singular values, the greater the amplification from the input signal to the output. They may be calculated as the singular values of $P_o^{\frac{1}{2}} P_c^{\frac{1}{2}}$. The balanced truncation approach seeks to truncate the directions in the state space corresponding to relatively small Hankel singular values, i.e., those that are relatively unimportant in describing the input-output behavior of the system while keeping those corresponding to the largest Hankel singular values. It can be shown that the distance between the original and truncated systems in the $\mathcal{H}_\infty$-norm is bound from above by twice the sum of the truncated Hankel singular values [DP00; Moo81].

The original formulation of the balanced truncation algorithm may be difficult to perform for large scale systems. One must first obtain both the controllability and observability gramians, either by solving the corresponding Lyapunov equations directly or approximately or by building up low-rank approximate gramians, e.g. from snapshot data from experiments [Row05]. Then, in principle, one can compute the balancing transformation using numerically stable but possibly expensive algorithms, e.g. SVD, Cholesky [DP00; Moo81].

In order to perform the balanced truncation efficiently, we obtain the gramians in TTVM format from our TT-based Lyapunov equation solver, then use a modified version known as Balanced Proper Orthogonal Decomposition [Row05] to balance and truncate in one step. First compute approximate gramians in TTVM format and use Algorithm 8 to compute their matrix square roots $P_c^{\frac{1}{2}}$ and $P_o^{\frac{1}{2}}$. Then compute the TTVM-SVD of the product (truncating the singular values below some threshold).

$$P_o^{\frac{1}{2}} P_c^{\frac{1}{2}} = U\Sigma V^T. \tag{V.3.1}$$

Take the TTVM products defined by

$$\mathcal{V}_1^T = \Sigma^{-\frac{1}{2}} V^T P_c^{\frac{1}{2}}, \quad \mathcal{W}_1^T = P_o^{\frac{1}{2}} U \Sigma^{-\frac{1}{2}}, \tag{V.3.2}$$

as the transformation and truncation. This transformation is balancing.

$$\begin{aligned} \mathcal{W}_1^T P_c \mathcal{W}_1 &= (P_o^{\frac{1}{2}} U \Sigma^{-\frac{1}{2}})^T P_c P_o^{\frac{1}{2}} U \Sigma^{-\frac{1}{2}} = \Sigma^{-\frac{1}{2}} U^T P_o^{\frac{1}{2}} P_c P_o^{\frac{1}{2}} U \Sigma^{-\frac{1}{2}} = \Sigma, \\ \mathcal{V}_1^T P_o \mathcal{V}_1 &= (P_c^{\frac{1}{2}} V \Sigma^{-\frac{1}{2}})^T P_o P_c^{\frac{1}{2}} V \Sigma^{-\frac{1}{2}} = \Sigma^{-\frac{1}{2}} V^T P_c^{\frac{1}{2}} P_o P_c^{\frac{1}{2}} V \Sigma^{-\frac{1}{2}} = \Sigma. \end{aligned}$$

The result is summarized in the following theorem.

Theorem V.3.1 ([Row05])**.** *Suppose $P_o^{\frac{1}{2}} P_c^{\frac{1}{2}}$ has rank $r = n$. Then with $\mathcal{V}_1$ and $\mathcal{W}_1^T$ defined above, $\mathcal{V}_1$ is a balancing transformation, and $\mathcal{W}_1^T$ is its inverse. That is,*

$$\mathcal{W}_1^T P_c \mathcal{W}_1 = \mathcal{V}_1^T P_o \mathcal{V}_1 = \Sigma.$$

Since the outputs of the TT-based Lyapunov solvers are low-rank approximations of the gramians, only a fraction of the Hankel singular values and their corresponding directions are available in (V.3.1). As a result, the balancing transformation

in (V.3.2) is not full rank, meaning that they perform the truncation to those Hankel singular values. Once the balancing and truncating transformation has been computed the matrices of the reduced system (V.0.8) can be computed using the TT VM-M-VM product and the TT VM-VM product. The results are summarized in Algorithm 13.

Algorithm 13 TT Balanced Truncation

Require: LTI control system (A, B, C, D) with matrix A in the TTM-format, matrix B in the TTVM-format, C in the TTVM-format and target state x_0 in the TTV format.

Ensure: Reduced system $A_{red}, B_{red}, C_{red}, D_{red}$.

Compute approximate infinite-time horizon gramians $\hat{P}_c$ and $\hat{P}_o$ using TT-DMRG CALE solver,

$[\hat{U}_c, \hat{\Sigma}_c, \hat{V}_c^T] = \text{TT\_SVD}(\hat{P}_c)$, $[\hat{U}_o, \hat{\Sigma}_o, \hat{V}_o^T] = \text{TT\_SVD}(\hat{P}_o)$,

$[\hat{U}, \hat{\Sigma}, \hat{V}^T] = \text{TT\_SVD}(\hat{U}_o \hat{\Sigma}_o^{1/2} \hat{V}_o^T \hat{U}_c \hat{\Sigma}_c^{1/2} \hat{V}_c^T)$,

$\mathcal{V}_1^T = \Sigma^{-1/2} V^T \hat{P}_c^{1/2}$, $\mathcal{W}_1^T = \hat{P}_o^{1/2} U \Sigma^{-1/2}$,

$A_{red} = \mathcal{W}_1^T A \mathcal{V}_1$, $B_{red} = \mathcal{W}_1^T B$,

$C_{red} = C\mathcal{V}_1$, $D_{red} = D$,

V.4 Numerical Experiments

Recall the linear reaction-diffusion equation

$$\begin{cases} \partial_t \boldsymbol{q}(x,t) = \left(\boldsymbol{D}\Delta + \sum_{\rho=1}^{R} f^\rho(x)(\boldsymbol{\mathcal{S}}_\rho \boldsymbol{\omega}_\rho)\right)\boldsymbol{q}(x,t) + \boldsymbol{F}(x)u(t), & x \in D. \\ \boldsymbol{q}(x,t) = 0, & x \in \partial D, \\ y(t) = \boldsymbol{H}\boldsymbol{q}(x,t). \end{cases} \tag{V.4.1}$$

For the numerical experiments, we consider a version of (V.4.1) that has been discretized in space using a finite difference scheme on a uniform tensor grid with spacing h but no discretization in time (Method of Lines). Let $\hat{\boldsymbol{q}}(x,t)$ denote the discrete approximation of $\boldsymbol{q}(x,t)$. The time evolution of the discretized system is given by the finite-dimensional LTI system:

$$\begin{aligned} \partial_t \hat{\boldsymbol{q}}(x,t) &= \boldsymbol{A}\hat{\boldsymbol{q}}(t) + \boldsymbol{B}u(t), \\ y(t) &= \boldsymbol{C}\hat{\boldsymbol{q}}(t). \end{aligned} \tag{V.4.2}$$

where

$$\boldsymbol{A} = \frac{1}{h^2}(\boldsymbol{\Delta}_{dd} \otimes \boldsymbol{D}) + \sum_{\rho=1}^{R} \left(\mathtt{diag}(\hat{\boldsymbol{f}}^\rho) \otimes (\boldsymbol{\mathcal{S}}_\rho \hat{\boldsymbol{\omega}}_\rho)\right), \tag{V.4.3}$$

where $\boldsymbol{\Delta}_{dd}$ is the discrete Laplacian on a rectangular grid with Dirichlet boundary conditions, $\boldsymbol{D}$ is the diffusion tensor, $\hat{\boldsymbol{f}}^\rho$ is the discretization of $f^\rho(\boldsymbol{x})$ on the spatial grid, and $\boldsymbol{B}$ and $\boldsymbol{C}$ depend on the discretizations of $\boldsymbol{F}(x)$ and $\boldsymbol{H}$, respectively.

Using the finest possible quantization for the spatial coordinates, $\hat{\boldsymbol{q}}(t)$ has the following index structure:

$$\underbrace{i_{1,1},\ldots,i_{1,l_1}}_{\text{1st dimension}},\underbrace{i_{2,1},\ldots,i_{2,l_2}}_{\text{2nd dimension}},\ldots,\underbrace{i_{d,1},\ldots,i_{d,l_d}}_{d\text{th dimension}},\underbrace{i_S}_{\text{reactants}}.$$

#	Reaction	Reaction Rate	Spatial Dependence
1.	$Q_1 \longrightarrow \emptyset$	$k_1 q_1(\boldsymbol{x})$	1

Table V.1: Reaction Network 1.

Using the sum and product estimates for TT-ranks, (V.4.3) gives an estimate the QTT-ranks of $\boldsymbol{A}$ (and using Theorem V.1.1, those of $\boldsymbol{\mathcal{L}}$, as well), assuming estimates of the ranks of each of the various terms are available. Let $r^{\boldsymbol{\Delta}_{dd}}_{k,m_k}$ and $r^{\hat{\boldsymbol{f}}^\rho}_{k,m_k}$ denote the QTT ranks corresponding to spatial dimension k and quantization level m_k of $\boldsymbol{\Delta}_{dd}$ and $\hat{\boldsymbol{f}}^\rho$, respectively, and let $r^{\boldsymbol{D}}$ denote the matrix rank of $\boldsymbol{D}$. Then the QTT ranks of $\boldsymbol{A}$ are bound from above by

$$\underbrace{r^{\boldsymbol{\Delta}_{dd}}_{1,1} + \sum_{\rho=1}^{R} r^{\hat{\boldsymbol{f}}^\rho}_{1,1}, \ldots, r^{\boldsymbol{\Delta}_{dd}}_{1,l_1-1} + \sum_{\rho=1}^{R} r^{\hat{\boldsymbol{f}}^\rho}_{1,l_1-1}}_{\text{1st dimension}}, \underbrace{r^{\boldsymbol{\Delta}_{dd}}_{1} + \sum_{\rho=1}^{R} r^{\hat{\boldsymbol{f}}^\rho}_{1}}_{\text{TT-rank}}, \ldots$$
$$, \underbrace{r^{\boldsymbol{\Delta}_{dd}}_{d-1} + \sum_{\rho=1}^{R} r^{\hat{\boldsymbol{f}}^\rho}_{d-1}}_{\text{TT-rank}}, \underbrace{r^{\boldsymbol{\Delta}_{dd}}_{d,1} + \sum_{\rho=1}^{R} r^{\hat{\boldsymbol{f}}^\rho}_{d,1}, \ldots, r^{\boldsymbol{\Delta}_{dd}}_{d,l_d-1} + \sum_{\rho=1}^{R} r^{\hat{\boldsymbol{f}}^\rho}_{d,l_d-1}}_{d\text{th dimension}}, R+1, \underbrace{r^{\boldsymbol{D}} + R}_{\text{reactants}},$$

and an upper bound for the QTT ranks of the corresponding $\boldsymbol{\mathcal{L}}_{\boldsymbol{A}}$ are given by Theorem V.1.1.

V.4.0.1 Example 1: 2D Multiple Input System

Consider the controlled reaction-diffusion system with a single chemical specie Q_1 on a square domain $D = [-\pi, \pi]^2$, subject to the single degradation reaction listed in Table V.1 and with four separate control channels $u_1(t), \ldots, u_4(t)$ corresponding

to injection or removal of Q_1 at four separate points

$$\begin{aligned} x_1 &= [1/3, 1/3], \\ x_2 &= [-1/3, 1/3], \\ x_3 &= [-1/3, -1/3], \\ x_4 &= [1/3, -1/3]. \end{aligned}$$

Under the finite difference discretization in space, this system can be modeled by equation (V.4.2) with

$$\boldsymbol{A} = \frac{D}{h^2}\boldsymbol{\Delta}_{dd} - k_1\mathbf{Id}, \quad \boldsymbol{B} = \frac{1}{h^2}\left[\hat{\boldsymbol{\delta}}_1(x) \quad \hat{\boldsymbol{\delta}}_2(x) \quad \hat{\boldsymbol{\delta}}_3(x) \quad \hat{\boldsymbol{\delta}}_4(x)\right]$$

where $\hat{\boldsymbol{\delta}}_m(x)$ is a vector that is zero everywhere except at the entry corresponding to the grid point closest to x_m, and we have suppressed the reactants mode since there is only one chemical specie.

We express $\boldsymbol{A}$ in the QTTM format and $\boldsymbol{B}$ in the QTTVM format. Kazeev, *et al.* showed that using the finest possible quantization, $\boldsymbol{\Delta}_{dd}$ has an explicit QTT matrix representation with all QTT ranks < 4 [KK12]. Using Equation (V.4.4), both $\boldsymbol{A}$ and $\boldsymbol{\mathcal{L}}_{\boldsymbol{A}}$ have QTT ranks < 5. Each $\hat{\boldsymbol{\delta}}_m(x)$ has QTT rank 1, so $\boldsymbol{B}$ has QTTVM representation with ranks at most 4.

For the numerical experiments, we used a uniform tensor product mesh with $2^{10} = 1024$ points in each direction, and parameter values $D = 1$ and $k_1 = 1$. Table V.2 lists the number of parameters required to represent the various matrices.

Matrix	Full Format	Sparse	QTTM
$\boldsymbol{A}$ 2D System	1.100e12	5.243e6	1456
$\boldsymbol{\mathcal{L}_A}$ 2D System	1.100e12	5.243e6	4388
$\boldsymbol{A}$ 3D System	1.038e19	3.114e10	8699
$\boldsymbol{\mathcal{L}_A}$ 3D System	1.077e38	1.003e20	2.159e4

Table V.2: Number of parameters needed to represent matrices for the 2D and 3D systems in various formats.

V.4.0.2 Example 2: 3D-SISO System

We next describe a very large scale system. Consider the reaction-diffusion system with three chemical species Q_1, Q_2, Q_3 and the set of reactions listed in Table V.3. Reaction 1 is localized about the point $\boldsymbol{x}^* = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^T$ with spatial dependence described by the function

$$f_{\boldsymbol{x}^*}(\boldsymbol{x}) = (8\pi)^{-3/2} e^{-2(\boldsymbol{x}-\boldsymbol{x}^*)^T(\boldsymbol{x}-\boldsymbol{x}^*)}.$$

The remaining reactions have no spatial dependence. The reaction network describes a spatiotemporal signaling cascade. A signal molecule Q_1 binds to a localized substrate (whose concentration is far in excess of that of Q_1) to produce a molecule Q_2. Q_2 then catalyzes the production of molecule Q_3 which can take place anywhere in

#	Reaction	Reaction Rate	Localized?	Spatial Dependence
1.	$Q_1 \longrightarrow Q_1 + Q_2$	$k_1 q_1(\boldsymbol{x})$	x	$f_{\boldsymbol{x}^*}(\boldsymbol{x})$
2.	$Q_2 \longrightarrow Q_2 + Q_3$	$k_2 q_2(\boldsymbol{x})$		1
3.	$Q_1 \longrightarrow \emptyset$	$\gamma_1 q_1(\boldsymbol{x})$		1
4.	$Q_2 \longrightarrow \emptyset$	$\gamma_2 q_2(\boldsymbol{x})$		1
5.	$Q_3 \longrightarrow \emptyset$	$\gamma_3 q_3(\boldsymbol{x})$		1

Table V.3: Reaction network for the example problem.

the reaction volume. The input to the system is described by

$$\boldsymbol{F}(u(t), \boldsymbol{x}, t) = \begin{bmatrix} \delta(\boldsymbol{x} - \boldsymbol{x}_u)u(t) \\ 0 \\ 0 \end{bmatrix},$$

which models injection or removal of the signaling molecule Q_1 at the point $\boldsymbol{x}_u = (\pi, 0, 0)$. We take the output of the system to be the total amount of Q_3 produced:

$$y(t) = \boldsymbol{H}(\boldsymbol{q}(x, t)) = \int_D q_1(x, t)dt.$$

We use a second-order central difference to approximate the Laplacian and approximate the Dirac delta as a Kronecker delta function at the nearest grid point with unit mass.

$$\boldsymbol{A} = \frac{1}{h^2}(\boldsymbol{\Delta}_{dd} \otimes \boldsymbol{D}) + \sum_{r=1}^{5} \left(\mathtt{diag}(\hat{\boldsymbol{f}}^r) \otimes (\boldsymbol{S}_r \hat{\boldsymbol{\omega}}_r) \right), \quad \boldsymbol{B} = \frac{1}{h^3} \left(\hat{\boldsymbol{\delta}}(x) \otimes \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \right),$$

$$\boldsymbol{C} = \mathbf{1} \otimes \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},$$

where $\boldsymbol{\Delta}_{dd}$ is the discrete Laplacian on a rectangular grid with Dirichlet boundary conditions, $\boldsymbol{D}$ is the diffusion tensor, $\hat{\boldsymbol{f}}^r$ is the discretization of $f^r(\boldsymbol{x})$ on the spatial grid, $\hat{\boldsymbol{\delta}}(x)$ is a vector that is zero everywhere except at the entry corresponding to the grid point closest to $\boldsymbol{x}_u$.

For reactions two through five, $f^r(\boldsymbol{x}) = 1$ (no spatial dependence), $\mathtt{diag}(1) = \mathrm{Id}$, and $\boldsymbol{A}$ can be rewritten as

$$\boldsymbol{A} = \frac{1}{h^2}(\boldsymbol{\Delta}_{dd} \otimes \boldsymbol{D}) + \left(\mathtt{diag}(\hat{\boldsymbol{f}}^1) \otimes (\boldsymbol{S}_1 \hat{\boldsymbol{\omega}}_1) \right) + \left(\mathrm{Id} \otimes (\sum_{r=2}^{5} \boldsymbol{S}_r \hat{\boldsymbol{\omega}}_r) \right)$$

We represent $\boldsymbol{A}$ in the TTM format while $\boldsymbol{B}$ and $\boldsymbol{C}$ are represented in the TTVM format. We use the explicit QTT matrix representation for Δ_{dd} with all QTT ranks < 4 [KK12], while $\boldsymbol{D}$, $\mathtt{diag}(\hat{\boldsymbol{f}}^1) \otimes (\boldsymbol{S}_1 \hat{\boldsymbol{\omega}}_1)$, $\sum_{r=2}^{5} \boldsymbol{S}_r \hat{\boldsymbol{\omega}}_r$ each have TTM rank 1. If the QTT ranks of $\hat{\boldsymbol{f}}^1$ are bound from above by r_{f^1}, then so are the ranks of $\mathtt{diag}(\hat{\boldsymbol{f}}^1) \otimes (\boldsymbol{S}_1 \hat{\boldsymbol{\omega}}_1)$, similarly $\left(\mathrm{Id} \otimes (\sum_{r=2}^{5} \boldsymbol{S}_r \hat{\boldsymbol{\omega}}_r) \right)$ is rank 1 since Id has QTT representation with rank 1. Hence, $\boldsymbol{A}$ has QTT-ranks $< 5 + r_{f^1}$. Using Theorem V.1.1, the ranks of $\boldsymbol{\mathcal{L}_A}$ are bounded from above by $6 + r_{f^1}$. Also, both $\boldsymbol{B}$ and $\boldsymbol{C}$ are rank 1 in the TTVM after

quantization since each is a stack matrix with one vector that is rank 1 separable. As a result, both $\boldsymbol{Q}_C$ and $\boldsymbol{Q}_B$ are rank 1 separable as well. Therefore, all the matrices and vectors involved have low QTT-ranks.

For the numerical experiments, we used a uniform tensor product mesh with $2^{10} = 1024$ points in each direction, and parameter values $D = 1$ and $k_1 = 1$. Table V.2 lists the number of parameters required to represent the various matrices.

V.4.1 Implementation Details

In the following, we implemented our proposed algorithms in MATLAB using the *TT Toolbox* implementation of the TT and QTT tensor formats, publicly available at `http://spring.inm.ras.ru/osel`. All TT-based calculations (computation of gramians, control laws, reduced models) were performed in MATLAB 8.2.0.701 (R2013b) on a laptop with a 2.7 GHz dual-core processor with 12 GB RAM.

When testing the TT-DSPOLC algorithm, we computed the evolution of the full-format system under the approximate control law using one DL580 node of the High-Performance Cluster Knot: `http://csc.cnsi.ucsb.edu/clusters/knot`. This node is equipped with 4 Intel X7550 eight core processors and 512GB shared RAM. We gratefully acknowledge the Center for Scientific Computing at UCSB and NSF Grant CNS-0960316 for making use of this system possible.

V.4.2 TT-DSPOLC

We demonstrate the efficacy of the TT-DSPOLC method by using it to steer a 2D reaction-diffusion system to various states. For each steering problem we report the accuracy of the control law as the relative error in the Euclidean norm between the target state and the final state of the full system

$$\varepsilon_x = \frac{|x_0 - \hat{x}(0)|}{|x_0|}$$

as well as the expansion coefficients of the final state in the basis of the left singular vectors of the controllability gramian obtained by orthogonal projection onto the subspace spanned by those vectors

$$c_\alpha = \sum_{i_1,\ldots,i_d} \hat{V}_c^T(\alpha; i_1, \ldots, i_d) x(i_1, \ldots, i_d).$$

We also report the computation times of both the QTT-DSPOLC and the simulation time for the full system.

Figure V.7 plots the matrix singular values and QTT-ranks of the approximate gramians, as well as the computation time for the DMRG-based solver for various residual tolerances. As the tolerance becomes tighter, the ranks corresponding to the compression of the singular vectors increases faster than the operator rank, meaning that the DMRG-based solver allocates the extra computational effort to better resolving the dominant singular vectors as opposed to finding more of them.

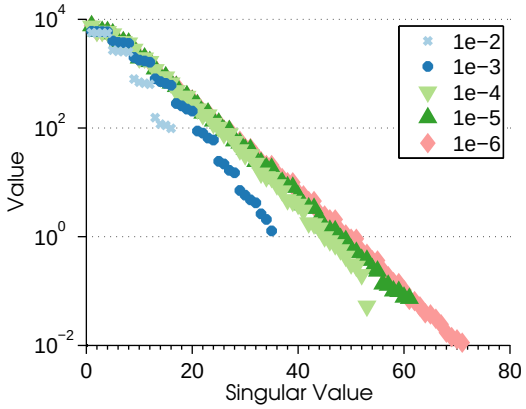
Figure V.8 plots a subset of the right singular vectors of $\hat{W}_c$ obtained from the DMRG-based solver with residual tolerance $\varepsilon_{\mathcal{L}} = 1e^{-}6$. In each case, the majority

of the variation in the singular vector is tightly concentrated about the points of actuation.

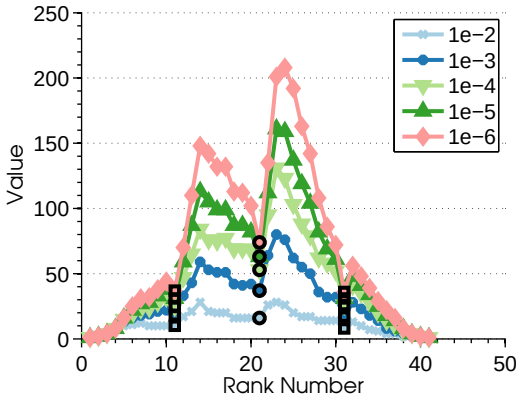
We first used the QTT-DSPOLC algorithm to steer the full-order system to a Gaussian profile $\frac{1}{2\pi}e^{-\frac{|x|^2}{2}}$, where the width of the profile was chosen to be large enough that no single left singular vector would work well as an approximation of the target profile. Even though each of the singular vectors of the Gramian has the majority of the variation concentrated about the actuation points, the QTT-DSPOLC algorithm can effectively steer the system to a profile that is qualitatively different, provided the resolution of the dominant subspace is sufficiently high.

Figure V.10 displays the final state of full system under the QTT-DSPOLC algorithm for various levels of approximation of the controllability gramian. As the approximation of the dominant subspace becomes more accurate, the full system can be driven closer to the target state. Figure V.9a plots the coefficients of the final state under each control law expanded in the basis of left singular vectors from the most accurately computed gramian and the coefficients of the target profile, while Figure V.9b plots the relative error in the final state and computation time of the control law with respect to the DMRG residual tolerance. Table V.5 gives a breakdown of the compute times for each step in generating the QTT-DSPOLC: t_{W_c} is the compute time of the DMRG-based solver, $t_{\hat{A}}$ is the compute time of the dominant subspace projection system, and t_{sim} is the compute time of the adjoint simulation.

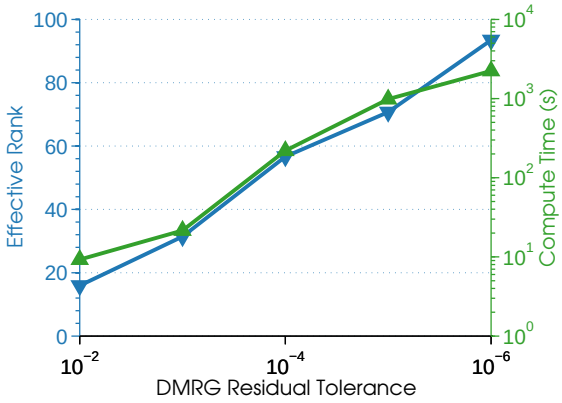
Next we tested the fidelity with which the DSPOLC algorithm steers the full



(a) Approximate Gramian Singular Values

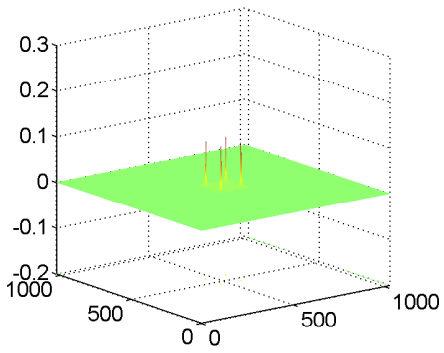


(b) Gramian Ranks

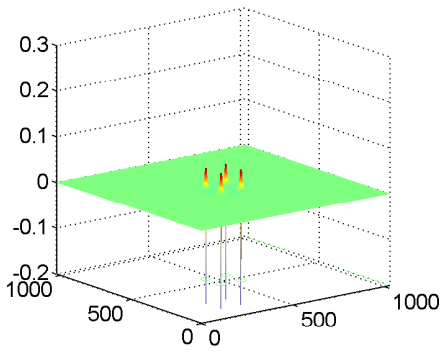


(c) Effective Rank and DMRG Compute Time

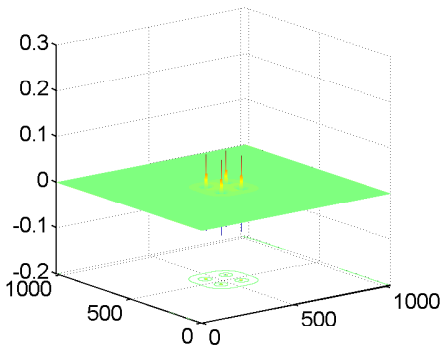
Figure V.7: Approximate controllability gramians for the multi-input system.



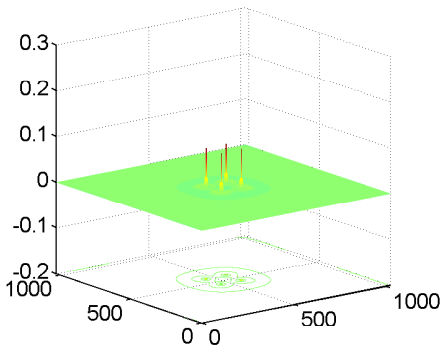
(a) $k = 1$



(b) $k = 9$

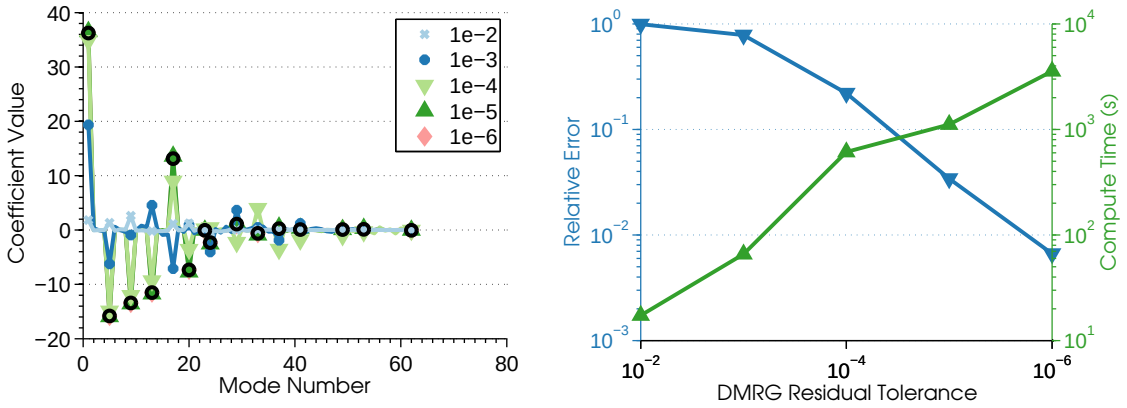


(c) $k = 20$



(d) $k = 29$

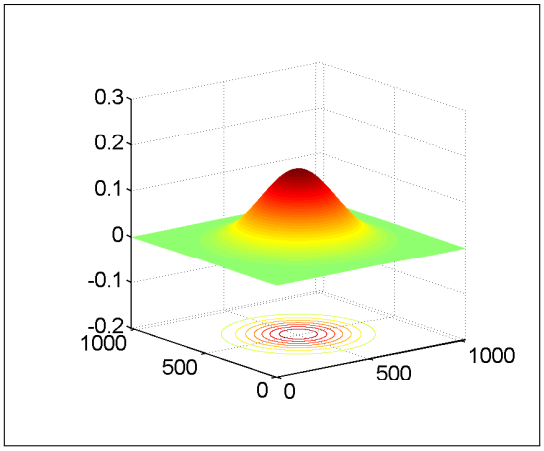
Figure V.8: Left singular vectors of $\hat{W}_c$ with $\varepsilon_{\mathcal{L}} = 1e^{-}6$ for the multi-input system.



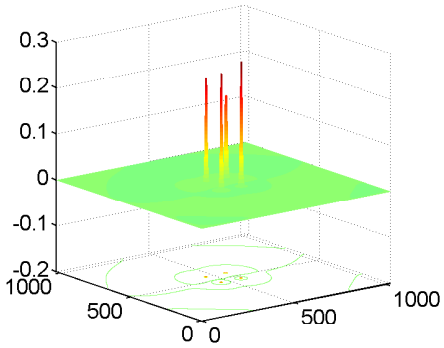
(a) Expansion Coefficients

(b) Relative Error and Total Compute Time

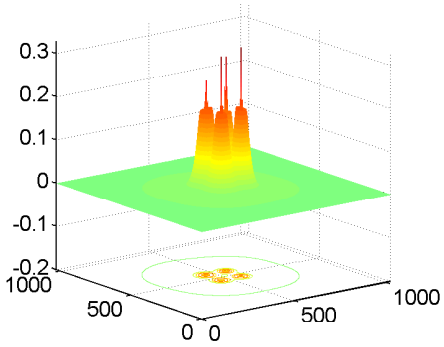
Figure V.9: Results for Gaussian profile steer problem for the multi-input system with respect to varying DMRG solver accuracy. V.9a plots the expansion coefficients of the final state with respect to the projection basis computed with DMRG tolerance set to $1e-6$. The black circles denote the expansion coefficients of the target state. V.9b plots the relative error of the final state under the open loop control as well as the total compute time of the control law.



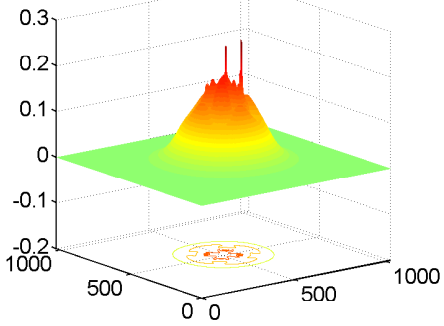
(a) Target State



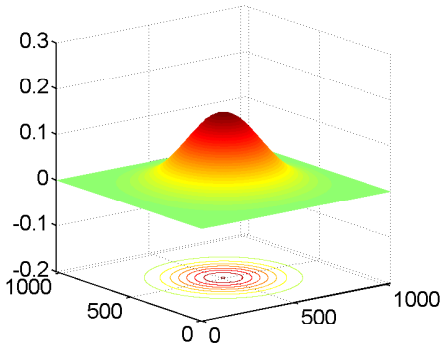
(b) 1e-2



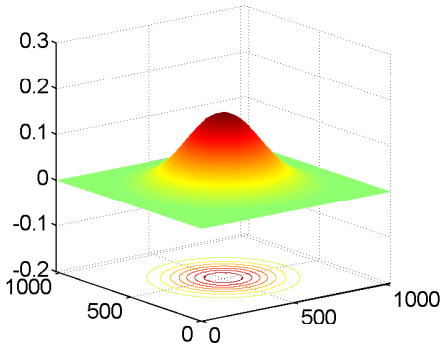
(c) 1e-3



(d) 1e-4



(e) 1e-5



(f) 1e-6

Figure V.10: Target and final states.

Matrix	Full Format	Sparse	$\hat{r}_c$-rank Approximation	QTT
$\hat{P}_c, \varepsilon_{\mathcal{L}} = 1e^{-}2$	1.100e12	1.100e12	1.678e7	1.900e4
$\hat{P}_c, \varepsilon_{\mathcal{L}} = 1e^{-}3$	1.100e12	1.100e12	3.670e7	1.016e5
$\hat{P}_c, \varepsilon_{\mathcal{L}} = 1e^{-}4$	1.100e12	1.100e12	5.557e7	2.467e5
$\hat{P}_c, \varepsilon_{\mathcal{L}} = 1e^{-}5$	1.100e12	1.100e12	6.396e7	3.939e5
$\hat{P}_c, \varepsilon_{\mathcal{L}} = 1e^{-}6$	1.100e12	1.100e12	7.445e7	6.775e5

Table V.4: Number of parameters needed to represent approximate controllability gramians for the multi-input system in various formats. The $\hat{r}_c$-rank approximation refers to an eigensystem approximation with the same rank. In all cases, the additional storage savings allowed by the QTT compression is signifcant.

system in each direction of the approximate controllable subspace by attempting to steer the system in the direction of each left singular vector of $\hat{P}_c$ resulting from $\varepsilon_{\mathcal{L}} = 1e^{-}5$. Figure V.11 displays the squared coefficients of the final state expanded in the left singular vectors of $\hat{P}_c$.

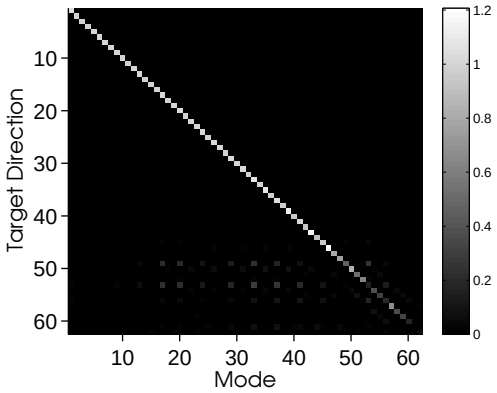
The QTT-DSPOLC algorithm steers the system to the desired direction with high fidelity for the dominant directions. As may be expected, the directions corresponding to smaller singular values of the controllability gramian are more difficult to steer to both in the sense that magnitude of the final state is smaller than desired and the coefficients of the expansion are nonzero for undesired modes.

Figure V.12 compares the total computation times for the control law and full

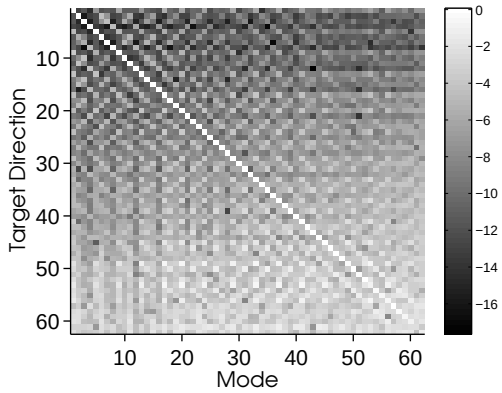
$\varepsilon_{\mathcal{L}}$	t_{P_c}	$t_{\hat{A}}$	t_{sim}
$1e^{-}2$	15.4s	1.7s	0.0337s
$1e^{-}3$	45.4s	20.4s	0.0664s
$1e^{-}4$	339.3s	273.8s	0.3299s
$1e^{-}5$	727.4s	386.1s	0.1034s
$1e^{-}6$	1887.9s	1671.5s	0.1253s

Table V.5: Compute time breakdown for various levels of accuracy of the controllability gramian for the multi-input system.

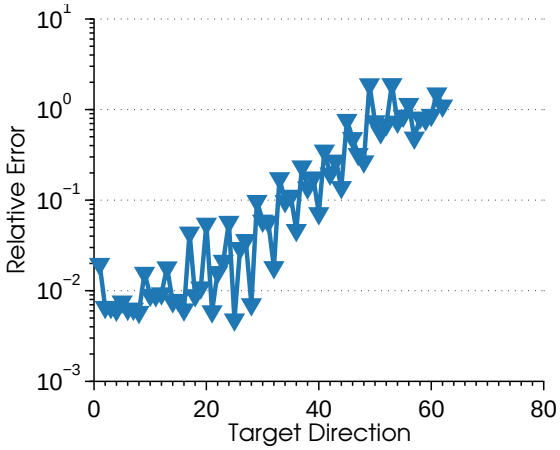
model simulation when steering in each direction. Compute time of the control law on a dual core laptop was nearly an order of magnitude faster than a simulation of the full system on a 32 core cluster node. We emphasize that the reported compute times are overestimated since each takes into account the time taken for both the solution of the Lyapunov equation and the projection onto the dominant subspace. In fact, the approximation of the dominant subspace and assembly of the projected system need only be done once to compute every control policy. In each case, the simulation time of the full system compared to the simulation time for the adjoint system differed by at least five orders of magnitude.



(a) Energy Content Per Mode vs Target Direction (Linear Scale)

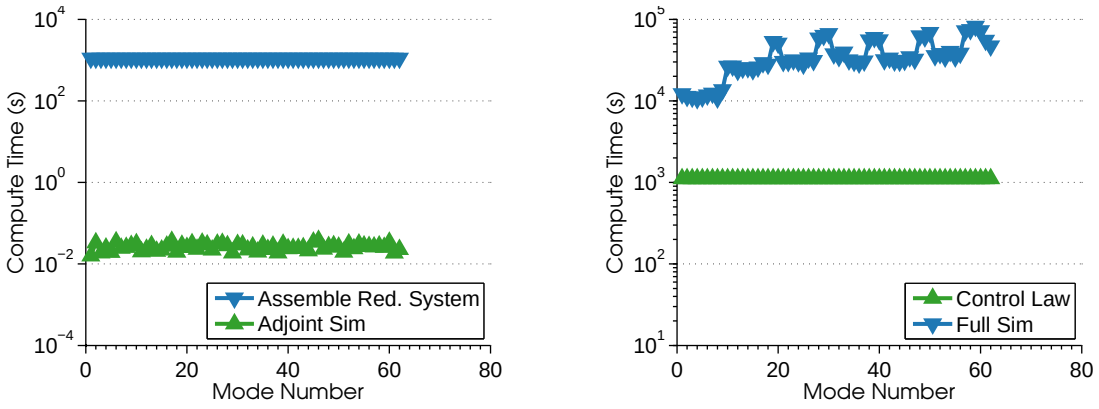


(b) Energy Content Per Mode vs Target Direction (Log Scale)



(c) Relative Error vs Target Direction

Figure V.11: Accuracy of DSPOLC Algorithm



(a) Control Law Compute Time Breakdown

(b) Control Law and Full Sim Compute Times

Figure V.12: Computation times for Example 1 when steering to a single mode. Compute time of the control law on a dual core laptop was nearly an order of magnitude faster than a simulation of the full system on a 32 core cluster node. We emphasize that the reported time for the computation of the control law includes both the time taken for the solution of the Lyapunov equation and the projection onto the dominant subspace, which only need to be done once to compute every control law.

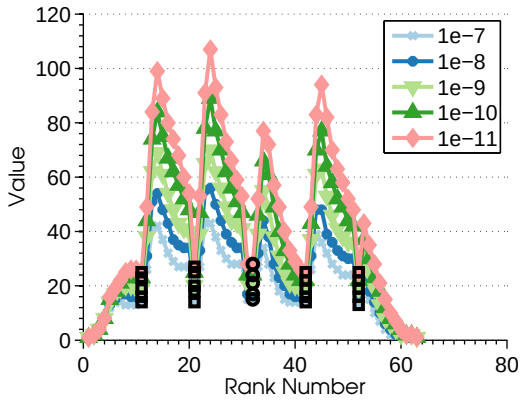
V.4.3 TT-Balanced Truncation

In order to demonstrate the efficacy of the Tensor Train approach, we computed computed reduced order models of the 3D SISO system using the Balanced Truncation summarized in Algorithm 13.

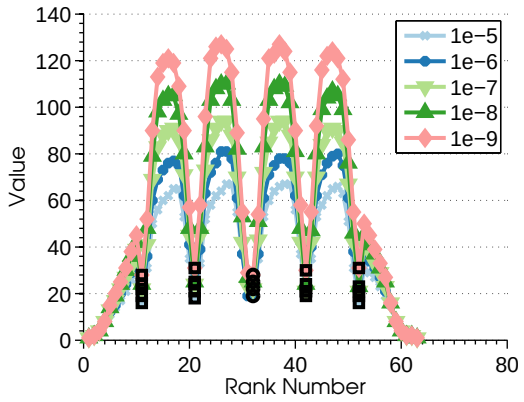
The results of the CALE solver are summarized in Figure V.13. For tighter solver tolerances, the TT ranks of the approximation must grow to satisfy the tolerances. Note that, due to the structure of the problem, the singular vectors of the gramians have a high degree of spatial separability corresponding to low values of the corresponding TT ranks. As the solver tolerances are tightened, the ranks corresponding to the separation in quantization levels grow much faster than those corresponding to both the matrix ranks of the gramians and the ranks separating the spatial dimensions of the singular vectors. The solver does more work in better *resolving* the singular vectors instead of increasing the *number* of singular vectors in the approximation.

The QTT compression significantly reduces the number of parameters required to represent the matrices and vectors involved. Table V.7 summarizes this data using the most accurate approximations. In each case, the QTT compression reduces the number of parameters required to represent the data by several orders of magnitude compared to sparse or simple low-rank formats.

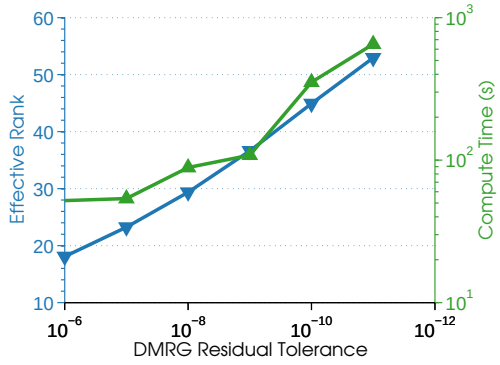
We then applied Algorithm 13 to compute reduced order models of the input-output system using six different combinations of controllability and observability



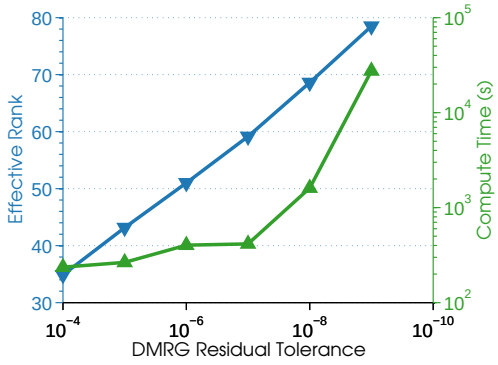
(a) Controllability Gramian TT-ranks



(b) Observability Gramian TT-ranks



(c) Controllability Gramian: Effective Rank and Compute Time vs. DMRG tolerance

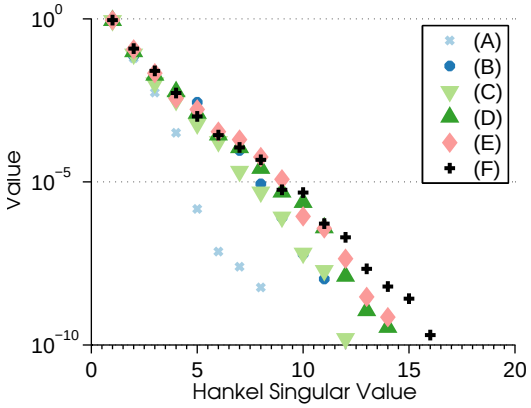


(d) Observability Gramian: Effective Rank and Compute Time vs. DMRG tolerance

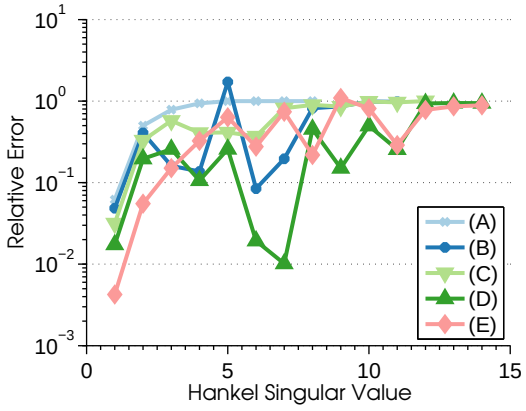
Figure V.13: Controllability and observability gramians computed using the DMRG-based solver for various tolerances in the residual error. TT ranks of the approximate controllability gramians (V.13a) and observability gramians (V.13b) for various tolerances. The matrix rank of each approximation of the operator is highlighted by a black circle, while ranks separating spatial dimensions in the singular vectors are highlighted by black squares. Effective ranks and compute times in seconds for various values of the residual tolerance for the controllability gramian (V.13c) and the observability gramian (V.13d).

gramians resulting from varying the DMRG residual levels. The combinations are listed in Table V.6 in order of increasing accuracy. Figure V.14 summarizes the results of the balanced truncation algorithm applied to each combination of gramians. All relative error calculations are taken with respect to the most accurately computed gramians. As the tolerances on the DMRG solver are tightened, more singular values of the Hankel matrix are included in the approximation with the largest singular values converging fastest. Note that while runs (D) and (E) result in Hankel matrices with the same number of singular values, the TT ranks corresponding to the singular vectors for (E) are significantly larger resulting in a longer compute time to obtain the reduced model. In each of the numerical experiments, the majority of the compute time was used by the DMRG solver finding the approximate gramians.

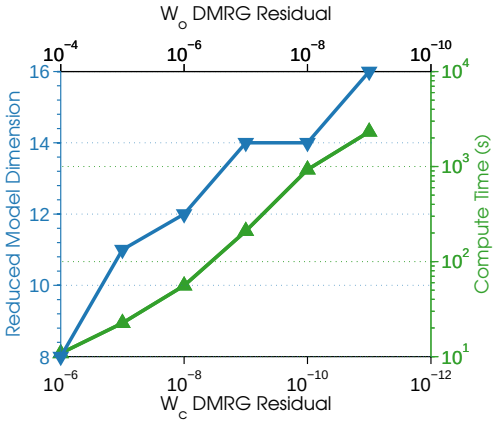
Even in the case of the most accurately computed gramians the model order reduction is massive with a reduction in the number of states by a factor by *nine orders of magnitude*.



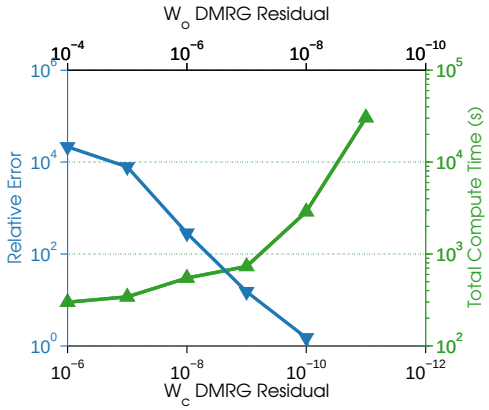
(a) Hankel Singular Values



(b) Relative Error in Hankel Singular Values



(c) Algorithm Compute Time



(d) $\mathcal{H}_\infty$ norm distance and total computation time

Figure V.14: Balanced truncation results. Each data set corresponds to a combination of DMRG residual tolerances listed in Table V.6. Relative error are computed with respect to the most accurate solutions of the Lyapunov equations (F). (V.14a) Singular values of each approximate Hankel matrix. (V.14b) Relative error in the singular values. (V.14c) Dimension of the resulting reduced models and the compute time of Algorithm 13 versus the residual tolerances. (V.14d) Relative errors of the transfer functions in the $\mathcal{H}_\infty$-norm of the reduced models and total computation time

Label	W_c DMRG Residual Tolerance	W_o DMRG Residual Tolerance
(A)	1e-6	1e-4
(B)	1e-7	1e-5
(C)	1e-8	1e-6
(D)	1e-9	1e-7
(E)	1e-10	1e-8
(F)	1e-11	1e-9

Table V.6: DMRG Tolerance Combinations

	Full Format	Sparse Format	Truncated SVD	QTT Format
$\boldsymbol{A}$	1.038e19	3.114e10	–	8699
$\boldsymbol{\mathcal{L}_A}$	1.077e38	1.003e20	–	2.159e4
$\boldsymbol{P_c}$	1.077e38	1.077e38	1.804e11	3.531e5
$\boldsymbol{P_o}$	1.077e38	1.077e38	1.804e11	7.762e5

Table V.7: Number of parameters required to represent matrices in full, sparse, truncated SVD, and QTT formats. $\boldsymbol{P_c}$ and $\boldsymbol{P_o}$ are the approximate gramians computed using the DMRG solver with tolerances (F) given in Table V.6. The number of parameters listed for the QTT format assume $\boldsymbol{A}$ and $\boldsymbol{\mathcal{L}_A}$ are represented in the QTT Matrix format while the gramians are represented in the Vectorized Matrix format. The truncated SVD representation refers to the low rank approximation of the gramians by SVD with the same number of singular values and vectors as is encoded in the QTTVM format.

V.5 Conclusions

We introduced a new approach to solving CALEs based on the Tensor Train numerical linear algebra. A key feature of the approach is that it adapts the matrix rank of the solution to capture the essential features of interest but no more. This parsimonious representation of the vectors and matrices involved allows substantial savings both in memory resources and overall compute time.

We remark that low-parametric tensor formats and linear system solvers in these formats is an active area of research. As solver technology improves we expect a further increase in the efficiency of this approach.

Based on the new approach to solving CALEs we proposed Gramian-based model reduction and open-loop control algorithms entirely within the TT format. A key feature of this approach is that all elements of the calculations, e.g. computation of dominant subspaces, coordinate transformations, subspace projections, etc., could be completed *without* ever referencing the full-format system.

We demonstrated the efficacies of these new approaches on challenging large-scale numerical examples of controlled reaction-diffusion equations.

Chapter VI

Conclusions

We presented a series of computational algorithms for the simulation, analysis, model reduction, and control of high-dimensional LTI systems from systems biology based on the Tensor Train numerical linear algebra. Overall, the TT approach to the design of numerical algorithms allows the solution of problems which classically required high-performance compute hardware on a notebook to the same or better levels of accuracy. In this chapter we summarize the conclusions of the doctoral project and describe some continuing work and possible future directions.

The *hp*-DG-QTT numerical solver for the CME produces numerically accurate results in a computationally efficient manner due to the combination of the efficient *hp*-DG time-stepping and the rank adaptivity of the QTT approach. Unlike most reduced basis methods, the solver automatically adapts the approximation "basis" of the solution. However, it is unclear how to efficiently and automatically adapt the

temporal mesh of for the *hp*-DG temporal semi-discretization. While the simulation results in Chapter IV look quite promising, they neglect to report the amount of time and effort required by the practitioner to tune the temporal mesh for both speed and accuracy. Elaboration of a simple and computationally efficient scheme to automatically generate the temporal meshes is a necessary prerequisite for wide-spread adoption of the method by non-specialists. This is a currently under investigation.

Both the *hp*-DG-QTT solver for the CME and the model reduction and control algorithms for LTI systems essentially rely on the efficient solution of certain tensor structured linear systems using the DMRG-based solver. In each case the local nature of the solver is a limiting factor in the efficiency of the approach. For the *hp*-DG-QTT solver, this restricts both the feasible step-size that can be taken and the polynomial order of the temporal discretization. For the CALE solver, computing additional singular values and vectors of the solution can require significant computational work. We remark that ongoing research in TT-structured linear system solvers promises substantial increases in efficiency for both approaches. We mention a family alternating minimal energy methods was recently announced in [DS13].

We observe that the efficiency of the TT approaches depend on the data involved having low rank in the format and that the particular choice of index orderings might affect this critically. We remark that the TT-format is a special case of the more general *tensor network states*. In real world applications it may be more appropriate to represent the data as some other tensor network in order to compute efficiently.

The development of methods for determining a good tensor network data structure for a particular problem is ongoing work. A general overview of tensor network states and their use in numerical computations can be found in [CV09; VCM09].

Chapter VII

Appendix

This chapter describes two projects related to the Chemical Master Equation research. The first describes the construction of reduced basis models of the CME without using the Finite State Projection to first reduce the countable state space problem to a finite one. This method is aimed at reduction of stochastic models of gene regulatory networks which combine chemical species which can take (formally) unconstrained copy numbers (mRNA, proteins) with species that are only present in fixed quantities (DNA) within a cell. The second section describes an approach to computing Wasserstein pseudometrics by solving certain large-scale Linear Programming problems and applies this to model discrimination of two stochastic models with identical stationary distributions.

VII.1 $\mathcal{L}_1$ - Reduced Models of CME for Gene Regulatory Networks

VII.1.1 Gene Regulatory Networks

We assume that the state of our system of chemical reactions is expressed by a pair (x, g) where g takes values in a finite set $\mathcal{Q} = \{1, \ldots, Q\}$ and x is a vector of integers that can potentially be unbounded. We shall refer to the g component of the state as the *low count species* and to x as the *high count species*. In a generic regulatory circuit, the subcomponent g could correspond to the configuration of occupied/vacant binding sites for a transcription factor, whereas x could be a vector with molecule counts of mRNA, protein, and/or transcription factors.

For a stochastic chemical reaction network with R reaction channels, for the kth reaction, let $\omega_k(x, g)$ denote the reaction propensity, and η_k^x and η_k^g the components of the stoichiometric vector affecting the high count and low count species, respectively, when channel k fires. The Chemical Master Equation describing the time evolution of the probability density function is given by:

$$\begin{aligned} \dot{p}(x, g; t) &= -p(x, g; t) \sum_{k=1}^{R} \omega_k(x, g) \\ &+ \sum_{k=1}^{R} p(x - \eta_k^x, g - \eta_k^g; t) \omega_k(x - \eta_k^x, g - \eta_k^g) \end{aligned} \tag{VII.1.1}$$

where each $p(x, g; t)$ is nonnegative, less than 1, and the sum of all $p(x, g; t)$ over

all possible values of x and g is equal to one [Kam92]. In general, this describes an infinite dimensional coupled linear system of differential equations since for each pair (x, g), equation (1) specifies one differential equation.

The state space on which the process evolves is $\mathbb{Z}_{\geq 0}^{n_x} \times \mathcal{Q}$ and, for each fixed time t, $p(x, g; t)$ is an ℓ^1 function over the index-set $\mathbb{Z}_{\geq 0}^{n} \times \mathcal{Q}$.

VII.1.2 Projection

Let $\{\psi_\alpha(x, g) : \alpha \in A\}$ be a (Schauder) basis for $\ell^1(\mathbb{Z}_{\geq 0}^{n} \times \mathcal{Q})$ where A is some infinite (but countable) index set. Expanded in this basis, the chemical master equation reads:

$$\begin{aligned}
&\sum_{\alpha \in A} \dot{c}_\alpha(t)\psi_\alpha(x, g) \\
&= -\sum_{k=1}^{R} \sum_{\alpha \in A} c_\alpha(t)\psi_\alpha(x, g)\omega_k(x, g) \\
&+ \sum_{k=1}^{R} \sum_{\alpha \in A} c_\alpha(t)\psi_\alpha(x - \eta_k^x, g - \eta_k^g)\omega_k(x - \eta_k^x, g - \eta_k^g)
\end{aligned} \tag{VII.1.2}$$

$\forall (x, g) \in \mathbb{Z}_{\geq 0}^{n} \times \mathcal{Q}$ where each $c_\alpha(t)$ is the *spectral coefficient* of the expansion corresponding to basis function ψ_α. Since the basis set is fixed for all time t, the spectral coefficients capture the time-dependence of the solution. Our goal is to develop an approximation to the equation (2) that accurately captures biologically meaningful quantities that can be computed from $p(x, g; t)$. Many such quantities can be obtained by computing functionals on the state space $\ell^1(\mathbb{Z}_{\geq 0}^{n} \times \mathcal{Q})$ that correspond to

inner products. For example, the probability of the system occupying some subset S of the state space is a linear functional given by the inner product of the distribution with the indicator function $\mathbf{1}_S$ taking the value one on S and zero elsewhere. The expected value of any function $f(x, g)$ can be expressed as a linear functional on $\ell^1(\mathbb{Z}^n_{\geq 0} \times \mathcal{Q})$ that can be written as the inner product of the distribution with a vector whose entries are the values of $f(x, g)$ evaluated at the corresponding points of the state space. For simplicity of the mathematical treatment, we only consider functionals of this type. In view of this, we take as given a collection $\{\varphi_\beta : \beta \in B\}$ of linear functionals on ℓ^1 parameterized by an index set B, with the understanding that, while we may be willing to accept errors in the probability density $p(x, g; t)$, we want our approximation to the CME to accurately capture the evolution of the values of each functional φ_β along solutions to the CME, which is given by the following set of equations:

$$\begin{aligned} & \sum_\alpha \dot{c}_\alpha(t)\varphi_\beta(\psi_\alpha(x,g)) \\ = & \sum_\alpha (-\sum_{k=1}^{R} c_\alpha(t)\varphi_\beta(\psi_\alpha(x,g)\omega_k(x,g)) \\ + & \sum_{k=1}^{R} c_\alpha(t)\varphi_\beta(\psi_\alpha(x-\eta_k^x, g-\eta_k^g)\omega_k(x-\eta_k^x, g-\eta_k^g))) \end{aligned} \tag{VII.1.3}$$

with one differential equation for each φ_β. In order for the set of coefficients $\{c_\alpha(t)\}$ to solve equation (2) they must necessarily solve the equation (3) for each functional φ_β, $\beta \in B$.

Our construction of the lower dimensional approximation to the CME relies on the premise that the distribution $p(x, g; t)$ lies close to a subspace of $\ell^1(Z^n_{\geq 0} \times \mathcal{Q})$ generated by a finite subset $\{\varphi_\alpha : \alpha \in \hat{A}\}$, $\hat{A} \subset A$ of the original basis for $\ell^1(Z^n_{\geq 0} \times \mathcal{Q})$. To determine appropriate equations for the evolution of the coefficients $\{c_\alpha : \alpha \in \hat{A}\}$, we require that the equations (3) that express the evolution of the values of the functionals $\{\varphi_\beta : \beta \in B\}$ holds for every $\beta \in B$.

VII.1.2.1 Projection onto the Reduced Basis

Choosing the truncated basis functions $\{\psi_\alpha : \alpha \in \hat{A}\}$ to form an orthogonal basis for a subspace of a Hilbert space H and selecting the functionals $\{\varphi_\beta : \beta \in B\}$ to be of the form $\varphi_\beta(z) = \langle \psi_\beta, z \rangle$, $\forall z \in H$, $\forall \beta \in A = B$, each equation (3) would directly give us the evolution of one of the coefficients (due to the orthogonality of the basis). However, this is undesirable for two reasons: (1) it excludes the possibility to work with truncated bases that, while not orthogonal, have a better chance to approximate well $p(x, g; t)$ and (2) it typically prevents the use of functionals φ_β that provide biologically useful information, e.g. moments or other expectation values of the distribution. The tradeoff for these desirable qualities is that the system of equations (3) does not directly lead to one equation for each coefficient c_α. However, by stacking the coefficient $\{c_\alpha : \alpha \in A\}$ into a column vector $\mathbf{c}$, we can write (3) as the following system of linear differential equations

$$\mathbf{Q}\dot{\mathbf{c}} = \tilde{\mathbf{A}}\mathbf{c} \tag{VII.1.4}$$

where $\mathbf{Q}$ and $\tilde{\mathbf{A}}$ denote semi-infinite matrices with one row for each element of the finite set B and one column for each element of the infinite set A. Similarly, by stacking all the probabilities $p(x, g; t)$ in an infinite vector $\mathbf{p}(t)$ we can write

$$\mathbf{p} = \mathbf{B}\mathbf{c} \tag{VII.1.5}$$

where $\mathbf{B}$ is the infinite matrix with one column for each element of the set $\mathbb{Z}^n_{\geq 0} \times \mathcal{Q}$ and one row for each element of A. The CME can be expressed as

$$\dot{\mathbf{p}} = \mathbf{A}\mathbf{p} \tag{VII.1.6}$$

for some infinite matrix with one row/column for each element of $Z^n_{\geq 0} \times Q$. Finally, we can also stack the values of all functionals $\{\varphi_\beta \in B\}$ into a (finite) vector $\mathbf{d}$ with one entry for each element of B and write

$$\mathbf{d} = \mathbf{D}\mathbf{p}$$

where the semi-infinite matrix $\mathbf{D}$ has one row for each entry of the finite set B and one column for each element of the infinite set $\mathbb{Z}^n_{\geq 0} \times \mathcal{Q}$. Given a truncation of the corresponding basis elements, $\mathbf{B}$ can be partitioned into submatrices:

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_r & \mathbf{B}_\infty \end{bmatrix} \tag{VII.1.7}$$

and $\mathbf{c}$ can be partitioned into subvectors:

$$\mathbf{c} = \begin{bmatrix} \mathbf{c}_r & \mathbf{c}_\infty \end{bmatrix}^T \tag{VII.1.8}$$

Equation (VII.1.4) can then be rewritten as

$$\mathbf{D}\mathbf{B}_r\dot{\mathbf{c}}_r + \mathbf{D}\mathbf{B}_\infty\dot{\mathbf{c}}_\infty = \mathbf{D}\mathbf{A}\mathbf{B}_r\mathbf{c}_r + \mathbf{D}\mathbf{A}\mathbf{B}_\infty\mathbf{c}_\infty$$

Truncation of the basis is equivalent to assuming that $\mathbf{c}_\infty = 0$ for all time resulting in the reduced system:

$$\mathbf{DB}_r\dot{\mathbf{c}}_r = \mathbf{DAB}_r\mathbf{c}_r \tag{VII.1.9}$$

We refer to equation (VII.1.9) as the reduced dynamics. These equations can be viewed as a Method of Lines approximation of the dynamics. If $\mathbf{DB}_r$ is invertible, then equation (VII.1.9) can be multiplied on both sides by the inverse leading to a set of coupled ODEs which can be solved using numerical ODE integrators.

VII.1.2.2 Approximating Stationary Distributions

In many applications it is desirable to obtain an estimate of the stationary distribution. For example, many hybrid algorithms assume that a subset of the dynamics is sufficiently fast and reaches stationarity to obtain a reduced model by averaging the fast dynamics. Such methods require an estimate of the stationary distribution of the fast subset. However, both Monte Carlo methods like SSA and the FSP are not particularly adequate for determining stationary distributions. SSA requires the generation of a large number of long sample paths, with the required time span of each simulation difficult to estimate *a priori.* In the FSP, the stationary distribution of the truncated process typically corresponds to all probability assigned to an artificial absorbing state. In this section we propose a heuristic for finding an approximation of the stationary distribution using the framework previously discussed.

Assuming that the system has a stationary distribution $\hat{\mathbf{p}}$ it must satisfy the

equation:

$$0 = \mathbf{A}\hat{\mathbf{p}} \tag{VII.1.10}$$

A candidate $\bar{\mathbf{p}}$ for the closest approximation of the stationary distribution in the reduced subspace is given by

$$\begin{aligned} \bar{\mathbf{p}} = \arg \quad & \inf \quad ||\mathbf{A}\mathbf{p}|| \\ & ||\mathbf{p}||_1 = 1, \\ & \mathbf{p} \in R(\mathbf{B}_r) \end{aligned} \tag{VII.1.11}$$

which identifies the distribution that exhibits the slowest change. Since the subspace is finite dimensional it is closed and the infimum is achieved. The infinite vector $\mathbf{Ap}$ is typically impossible to compute, but it is possible to compute the truncated vector $\mathbf{DAB}_r\mathbf{c_r}$ so we instead solve the related minimization problem:

$$\begin{aligned} \textit{minimize} \quad & ||\mathbf{DAB}_r\mathbf{c}_r|| \\ \textit{subject to} \quad & ||\mathbf{B}_r\mathbf{c}_r||_1 = 1 \end{aligned} \tag{VII.1.12}$$

where the optimization variables are the coefficients $\mathbf{c}_r$ of the expansion. For any choice of norm in the objective function, the optimization is convex. When it is chosen to be the 2-norm, this is equivalent to finding the eigenvector corresponding to the smallest eigenvalue of the positive semi-definite matrix

$$(\mathbf{DAB}_r)^*(\mathbf{DAB}_r) \tag{VII.1.13}$$

and then normalizing with respect to the 1-norm.

While it is well-understood in the numerical linear algebra literature that a small residual does not imply a small error in the approximate solution [**numLinAlg**], our numerical experiments show this method to be highly efficient and accurate even with heuristically chosen basis functions and functionals.

VII.1.2.3 Choice of Basis Functions

While many choices of bases in (2) are admissible, some will better capture the dynamics when truncating to a finite number of elements. [Eng09] proposes the use of discrete Charlier functions, which are mutually orthogonal with respect to the Poisson distributions. [Deu+08] suggests the use of discrete Chebychev polynomials with a suitable scaling and translation to construct a tensor basis, the elements of which are mutually orthogonal with respect to the standard inner product.

In this work we are especially interested in systems with state spaces of the form $\mathbb{Z}^n_{\geq 0} \times Q$, where the component of the state in the finite set Q typically represents the dynamics of transcription factor binding sites. A natural choice of basis for such system is obtained by taking the tensor product between vectors in a basis of $\ell^1(\mathbb{Z}^n_{\geq 0})$ with indicator functions on the set $\mathcal{Q}$. Specifically, we work with basis vectors for $\ell^1(\mathbb{Z}^n_{\geq 0} \times \mathcal{Q})$ of the form

$$\{\psi_j \otimes e_k\}_{j,k} \tag{VII.1.14}$$

where ψ_j, $j \in \mathbb{Z}^n_{\geq 0}$ is an element of the basis of $l_1(\mathbb{Z}^n_{\geq 0})$ and e_k, $k \in \mathcal{Q}$, is the indicator function $e_k(g) = 1$ if $g = k$ and $e_k(g) = 0$ if $g \neq k$.

This choice of basis is motivated by the observation that by expressing the probability distribution $p(x, g; t) = P(X(t) = x, G(t) = g)$ as a linear combination of the vectors (14), we are implicitly expanding the distribution of $X(t)$ conditioned to $G(t)$ as a linear combination of the vectors ψ_j since:

$$\begin{aligned} & P(X(t) = x, G(t) = g) \\ = \;& \alpha(t) P(X(t) = x | G(t) = g) \\ = \;& \alpha(t) \sum_{j,k} c_{j,k} (\psi_j \otimes e_k)(x, g) \\ = \;& \alpha(t) \sum_{j} c_{j,g} \psi_j(x) \end{aligned} \tag{VII.1.15}$$

where $\alpha(t) = P(G(t) = g)$. In view of this, the choice of the basis functions ψ_j should be dictated by the conditional distribution of $X(t)$ given $G(t)$.

In many systems, the binding configuration determines the qualitative dynamics of the mRNA, proteins, etc. For instance, the binding of a transcription factor to a promoter can drastically increase the transcription rate of the associated gene. In the next section, we describe a simple system where this is precisely the case and including the binding dynamics is crucial to obtaining an accurate model of the system.

VII.1.3 Case Study: Negative Feedback Circuit Exhibiting Bimodality

Consider the negative-feedback gene regulatory circuit modeled by

$$G_{on} \overset{\frac{k_1 X}{T} G_{on}}{\longrightarrow} G_{off}$$

$$
\begin{aligned}
G_{off} &\xrightarrow{\frac{k_2}{T} G_{off}} G_{on} \\
G_{on} &\xrightarrow{k_3 G_{on}} G_{on} + X \\
\emptyset &\xrightarrow{k_4} X \\
X &\xrightarrow{k_5 X} \emptyset
\end{aligned}
$$

G acts as a promoter that randomly switches between the "on" and "off" states and the species X is produced at a high rate when G is "on" and at a lower basal rate when G is "off". The likelihood of G switching from "on" to "off" is influenced by the concentration of X: the higher the concentration, the more likely G will transition to "off". T is a time-scale parameter that scales the flip rate between the two states of the gene, that is, decreasing T decreases the mean waiting time between switching from "on" to "off" and vice versa.

This system can be made to exhibit a bimodal stationary distribution for species X, which is difficult to capture by methods that rely on a continuous approximation of the dynamics. In the regime of slowly switching gene, the peaks of the distributions of X conditioned on G will be offset from one another as the system will slowly switch between two subsystems that are effectively isolated from each other. The stationary distribution of X for the isolated subsystems is Poisson with parameter $\frac{k_4}{k_5}$ for the subsystem conditioned to "off" and with parameter $\frac{(k_3+k_4)}{k_5}$ for the subsystem conditioned to "on". For k_3 large enough compared to k_4, the marginal distribution of X will then have two distinct peaks corresponding to the peak of each conditional. Note that in this parameter regime, it is essential to include the binding/unbinding

dynamics of the promoter in the Markov model. Assuming an equilibrium model for the state of the binding site through an averaging argument destroys the bimodality of the full stationary distribution.

We encoded the state space of the process as the ordered pair (x, g) with x counting the number of molecules of X present and $g \in \{0, 1\}$ with $g = 0$ corresponding to the "off" state of the gene and $g = 1$ corresponding to the "on" state. For this example, we used the tensor basis:

$$\left\{ e^{-\frac{\left(x-x_j\right)^2}{2\sigma^2}} \otimes e_k \right\}_{j,k} \tag{VII.1.16}$$

where the first term of the product are elements of a basis for $\ell^1(\mathbb{Z}_{\geq 0})$ and e_k, $k \in \{0, 1\}$ is the indicator function of the event $g = k$. The different basis vectors for $\ell^1(\mathbb{Z}_{\geq 0})$ are obtained using Gaussian distributions with means $x_j \in \Omega$, a finite subset of $\mathbb{Z}_{\geq 0}$ and standard deviation σ. The use of Gaussian distributions for the basis functions is motivated by the observation that the state distribution typically resemble Gaussians and therefore one hopes that a low-order representation can be achieved with such basis functions. For the linear functionals we used the evaluation mappings:

$$\varphi_{(x,g)}(f) = f(x, g) \tag{VII.1.17}$$

for any $f \in \ell^1(\mathbb{Z}^n_{\geq 0} \times \mathcal{Q})$ and with $g \in \{0, 1\}$ and $x = x_j$ corresponding to the peak of each Gaussian. That is, for each pair $(x, g) \in B$, $B = \{(x, g) : x \in \Omega, g \in \{0, 1\}\}$, $\varphi_{(x,g)}$ is the linear functional that evaluates the distribution at (x, g). This corresponds to being able to identify whether or not the binding site is occupied (since we included

both e_0 and e_1) and being able to count the number of transcription factors but with limited resolution.

Equations (16) and (17) can be translated into the language of semi-infinite matrices in the following way. Assuming the lexicographic ordering of the state space and using the corresponding matrix $\mathbf{A}$, the matrices encoding the choice of basis functions and linear functionals have the block diagonal form:

$$\mathbf{B}_r = \begin{bmatrix} \mathbf{B}_r^0 & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_r^1 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} \mathbf{D}^0 & \mathbf{0} \\ \mathbf{0} & \mathbf{D}^1 \end{bmatrix} \tag{VII.1.18}$$

with the columns of $\mathbf{B}_r^0$ and $\mathbf{B}_r^1$ representing the basis for the distribution conditioned on G_{off} and G_{on}, respectively, and with $\mathbf{D}^0$ and $\mathbf{D}^1$ representing measurements of the amount of X present in each of the gene states:

$$\mathbf{B}_r^0 = \mathbf{B}_r^1 = \begin{bmatrix} | & | & \\ e^{-\frac{(x-0)^2}{2\sigma^2}} & e^{-\frac{(x-5)^2}{2\sigma^2}} & \cdots \\ | & | & \end{bmatrix} \tag{VII.1.19}$$

$$\mathbf{D}^0 = \mathbf{D}^1 = \begin{bmatrix} e_0 \\ e_5 \\ \vdots \end{bmatrix} \tag{VII.1.20}$$

The reduced model is given by equation (VII.1.9). $\mathbf{DB}_r$ in this case happens to be invertible so we can directly apply conventional numerical ODE solvers.

Figure 1 compares the time evolution of the marginal distribution for X using the approximation method with $\sigma = 15$ and $\Omega = \{0, 5, 10, \ldots, 200\}$ with the distri-

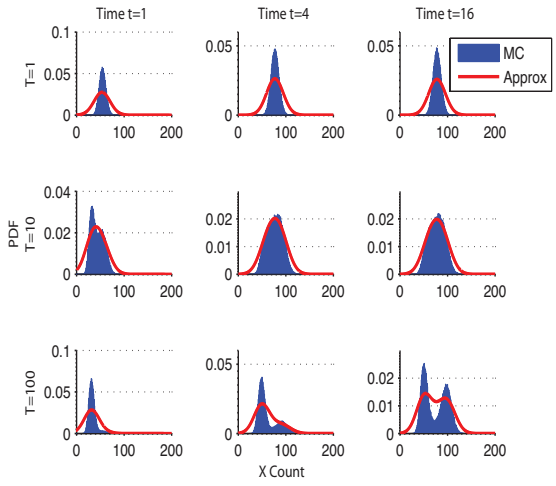


Figure VII.1: Comparison of the time evolutions of the marginal PDF of species X generated by the approximation method with 10^5 Monte Carlo simulations. In the case of the slow binding/unbinding dynamics ($T = 100$), the spectral method successfully captures the bimodality of the solution.

bution estimated by 10^4 SSA sample paths. SSA realizations were initialized with $(x, g) = (0, 0)$, while the approximation method was initialized with all probability concentrated in the function $e^{-\frac{x^2}{2\sigma^2}} \otimes e_0$, the basis function that is qualitatively the closest. Even though each system is initialized differently the spectral approximation produces qualitatively similar dynamics to the SSA with good agreement of the centering of the peaks of each mode of the distribution. Note, however, that the values of the peaks for the approximation are smaller than those produced by Monte Carlo and that the approximation distribution decays much more slowly. This lack of fast decay is a consequence of using Gaussians in the choice of basis since the decay rate of the approximation is bound below by the decay rate of any one of the Gaussians used.

Figure 2 compares the approximate stationary distribution generated by the heuristic using the same σ and Ω with SSA sample paths of simulated time length 100. The approximate solution was found by selecting the 2-norm for the objective function and performing the corresponding eigenvalue/eigenvector computation. The spectral approximation is qualitatively similar to the SSA estimate. Note here that this solution suffers from the same inaccuracies as the transient solution: peak values are smaller, and the distribution does not decay as rapidly.

Figure 3 compares the error produced at time $t = 100$ by time evolution of the approximate system for various values of σ and different uniform spacings between the peaks of each Gaussian. The error is calculated with respect to a solution of the FSP

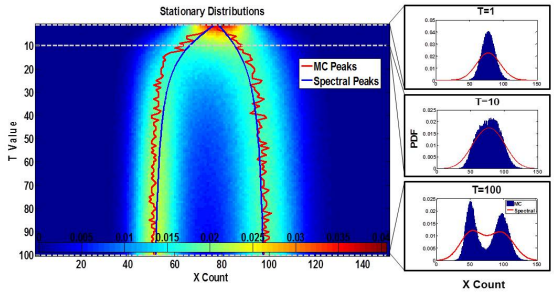


Figure VII.2: Comparison of estimates of the stationary marginal PDF of species X generated by the spectral approximation with 10^5 SSA realizations. The pseudo-color plot shows PDFs estimated by SSA over a range of binding/unbinding speeds. The red curves show the peaks of the distribution estimated by SSA, while the blue curves show the peaks estimated by the spectral method. At right, comparisons of the stationary distribution predicted by the spectral method with SSA estimates. The spectral method successfully captures the bimodality of the solution in the slow switching regime.

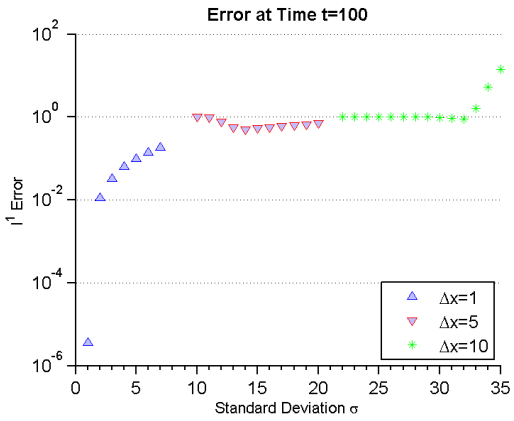


Figure VII.3: Comparison of error at time $t = 100$ for various values of peak spacing and width. For each peak spacing there is a value of the variance which minimizes the error of the approximation. Errors are calculated with respect to an FSP solution with the same initial conditions. The value of $t = 100$ was chosen as each system is approximately stationary after this amount of time.

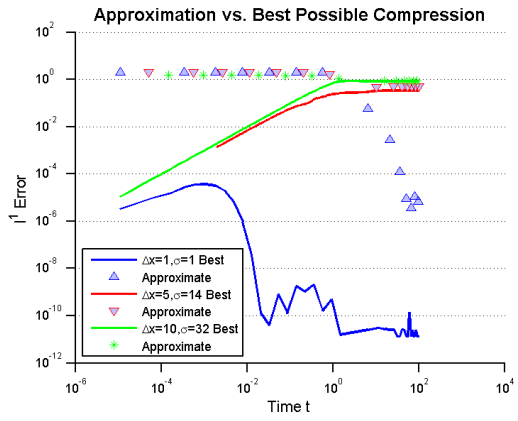


Figure VII.4: Comparison of approximate solutions with the best possible compression with the same basis functions. Solid lines indicate the best possible compression while markers indicate the approximations. In each case, the approximate solution may have a relatively large error during the initial transient phase, while the dynamics appear to minimize the errors in the stationary phase. Error values plotted below 10^{-7} are likely innacurate since these are of the same order as the tolerances used in the numerical optimization.

initialized with one of the basis functions. FSP reference solutions were calculated with an error certificate of $\varepsilon < 10^{-9}$ in each state at the final time. We used FSP to generate the reference solutions rather than SSA as the computation time to do so to the desired accuracy is significant. As expected, the more basis functions allowed, the lower the error that can be achieved.

For each minimal error spacing and variance combination shown in Figure 3, Figure 4 compares the ℓ^1 error of the approximation with the error of the best possible approximation for the same choice of basis over time. The best possible approxima-

tions were obtained by minimizing the ℓ^1 error of the FSP reference solution over the same choice of basis functions by numerically solving the corresponding Linear Program. While the spectral approximations may have large errors during the initial fast transient phase, the error gaps close when approaching steady state.

VII.1.4 Conclusions/Future Research

In this paper we considered a class of models of gene regulatory networks that explicitly include the full binding dynamics of transcription factors. Here we assumed that the binding dynamics occur on a time scale that is too slow to be averaged out by the rest of the system and are therefore important to the overall function of the network. We demonstrated that general spectral methods have characteristics well suited to solving CMEs of this type assuming a good choice of basis functions. We also introduced a novel method for approximating the stationary distribution based on the spectral framework.

Establishing error bounds for the approximations proposed here is the topic of ongoing research.

VII.2 Descrimination of Stochastic Models Using Wasserstein Pseudometrics and MultiCriterion Optimization

VII.2.1 Introduction

Quantifying the differences between two models or between experimental data and a candidate model is a fundamental procedure in system identification, model selection, and model reduction. Wasserstein pseudometrics have been suggested as a useful measure of the differences between stochastic models since they do not require strong statistical assumptions about the noise processes and can be tailored to capture and quantify essential elements of the output properties of the processes [TK08]. Previous work has considered computation of Wasserstein pseudometrics arising from a single comparison criteria [Koe+10; TK08; TK10]. In this project, we consider generalizing this framework to allow model comparisons based on multiple criteria. We show that for a particular choice of pseudometrics, approximation of their values can be accomplished by solving a single large-scale Linear Programming problem. We use the computational method to select the appropriate stochastic model from data generated by stochastic realizations.

VII.2.2 Background

Let $\Omega_X = \{f : \mathbb{R}_{\geq 0} \to X\}$. A Markov process X_t induces a σ-algebra $\mathcal{F}_X$ and probability measure μ_X on Ω_X. Given an observation function $g : X \to Y$ with sufficiently nice properties, this also induces a σ-algebra $\mathcal{F}_Y$ and probability measure μ_Y on a subset of the space $\Omega_Y = \{\omega : \mathbb{R}_{\geq 0} \to Y\}$, that is, it generates a new process, generally not Markov, Y_t parameterized by $t \in \mathbb{R}_{\geq 0}$ and with state space Y. The basic idea is that there is some underlying Markov process that has state space X but that we are only able to make observations of the process on a smaller space Y. For simplicity, take $Y = \mathbb{R}^m$.

Suppose we wish to compare two processes Y_t and $\hat{Y}_t$ defined on the same state space Y that are each generated by a Markov process and choice of observation function (X_t, g) and $(\hat{X}_t, \hat{g})$ as above, respectively. We can calculate a distance between the two measures, μ_Y and $\hat{\mu}_Y$, corresponding to each process. In general, there are many different choices of distance measures. Here we restrict our attention as in [1] to the Wasserstein pseudometric.

Let d be a pseudometric on Ω_Y. The Wasserstein pseudometric W_d between two probability measures μ_1 and μ_2 on Ω_Y is given by

$$W_d(\mu_1, \mu_2) = \inf_{Q \in J(\mu_1, \mu_2)} \mathrm{E}_Q[d(\omega_1, \omega_2)] \tag{VII.2.1}$$

Where the infimum is taken over all joint distributions that have μ_1 and μ_2 as

marginals. As in [TK10] we consider pseudodistances of the form

$$d(\omega_1, \omega_2) = ||Z(\omega_1) - Z(\omega_2)|| \tag{VII.2.2}$$

where $Z : \Omega_Y \to \mathbb{R}$ is a reporter random variable defined to capture some feature of the trajectories, and $||\cdot||$ is a norm on $\mathbb{R}$. Many choices of Z are possible (it only has to be measurable with respect to both σ-algebras). Some possible choices of Z are the copy number of a protein at time t or the first time instant that a transcriptor factor binds to a promoter site.

Let F_1 and F_2 be the probability distributions for Z under measures μ_1 and μ_2, respectively. Suppose that under both probability measures, the essential range of Z is discrete and has finite cardinality, that is, Z is a discrete random variable taking finitely many values. Then equation (1) reduces to:

$$W_d(\mu_1, \mu_2) = \inf_{Q \in J(F_1, F_2)} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} ||Z_{1i} - Z_{2j}|| Q_{ij} \tag{VII.2.3}$$

where n_1 and n_2 are the maximum number of distinct values that correspond to the preimage under Z having positive measure under F_1 and F_2, respectively, Z_{1i} and Z_{2j} are the ith and jth value taken by Z under F_1 and F_2, and Q_{ij} is the probability with which both events occur. (The insistence on Z taking finitely many values is not so essential since we can always compose Z with a suitable projection map.) If approximations of the marginal distributions can be obtained then an approximation of the Wasserstein pseudometric can be calculated by solving the linear program

$$\text{minimize} \quad \textstyle\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} ||Z_{1i} - Z_{2j}|| Q_{ij}$$

$$\begin{aligned} \text{subject to} \qquad & \textstyle\sum_{j=1}^{n} Q_{ij} = P(Z_i) \\ & \textstyle\sum_{i=1}^{n} Q_{ij} = P(Z_j) \\ & Q_{ij} \geq 0 \\ & \textstyle\sum_{ij} Q_{ij} = 1 \end{aligned}$$

where Q_{ij} are the optimization variables. The last two constraints require Q to be a probability measure, while the first two require that Q be a joint distribution with μ_1 and μ_2 as marginals. [TK08; TK10] do this by using kinectic monte carlo simulations to obtain approximate distributions for Z. [Koe+10] does this by numerical integration of the Chemical Master Equations corresponding to the underlying Markov processes.

VII.2.3 Multicriterion Comparisons

Often it may be desirable to compare two models based on more than one criterion. For instance, if we want to model a very complicated process with a much simpler one, it is often desirable for the reduced model to have both similar asymptotic and transient behavior to the original process. In this case, a single reporter random variable will be insufficient to capture information about both critera.

One method to incorporate multiple criteria into a single pseudometric is to use multiple reporter variables and to choose a corresponding family of pseudometrics:

Theorem VII.2.1. *Let $\{Z^k\}_{k=1}^m$ be a collection of random variables $Z^k : \Omega_Y \to \mathbb{R}$.*

For each k let

$$d_k(\omega_1, \omega_2) = ||Z^k(\omega_1) - Z^k(\omega_2)|| \tag{VII.2.4}$$

be the pseudometric associated with Z^k. Then the function $Z : \Omega_Y \to \mathbb{R}^m$ given by

$$Z(\omega) = \begin{bmatrix} Z^1(\omega) \\ \vdots \\ Z^m(\omega) \end{bmatrix} \tag{VII.2.5}$$

is a vector-valued random variable. In addition, every convex combination

$$d^\lambda(\omega_1, \omega_2) = \sum_{k=1}^{m} \lambda_k d_k(\omega_1, \omega_2) \tag{VII.2.6}$$

of the set $\{d_k\}_{k=1}^m$ is a pseudometric on Ω_Y with associated Wasserstein distance

$$W_D^\lambda(\mu_1, \mu_2) = \inf_{Q \in J(\mu_1, \mu_2)} \sum_{k=1}^{m} \lambda_k E_Q[d_k(\omega_1, \omega_2)] \tag{VII.2.7}$$

Proof. Measurability of Z follows from closure of $\mathcal{F}_Y$ under countable intersection. d^λ is a pseudometric can be shown by checking each of the axioms and using the fact that each d_k is a pseudometric. (VII.2.7) follows from the definition and by using linearity of the expectation. □

We consider the family of convex combinations of $\{d_k\}_{k=1}^m$ since we may be agnostic as to the relative importance of each criterion. If each Z^k is discrete and takes finitely many values under both measures then (VII.2.7) can be written explicitly as

$$W_D^\lambda(\mu_1, \mu_2) = \min_{Q \in J(\mu_1, \mu_2)} \sum_{k=1}^{m} \lambda_k \sum_{i=1}^{n_1^k} \sum_{j=1}^{n_2^k} ||Z_{1i}^k - Z_{2j}^k|| Q_{ij}^k \tag{VII.2.8}$$

where n_1^k, n_2^k are the maximum number of distinct values that Z^k takes under measure μ_1, μ_2, respectively. (i and j are best interpreted in this context as multi-indices).

As before, if the marginal distributions can be obtained then we can calculate the Wasserstein distance by solving the following linear program:

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{k=1}^{m} \lambda_k \sum_{i=1}^{n_1^k} \sum_{j=1}^{n_2^k} ||Z_{1i}^k - Z_{2j}^k|| Q_{ij}^k \\ \text{subject to} \quad & \textstyle\sum_{j=1}^{n} Q_{ij}^k = P(Z_i^k) \\ & \textstyle\sum_{i=1}^{n} Q_{ij}^k = P(Z_j^k) \\ & Q_{ij}^k \geq 0 \\ & \textstyle\sum_{ijk} Q_{ij}^k = 1 \end{aligned}$$

where the Q_{ij}^k's are the optimization variables. Note that the last two constraints require Q to be a probability measure and the first two require Q to have marginals μ_1 and μ_2. Now by sweeping through various λ's we assign different weights to each criterion.

VII.2.4 Example: A Gene Expression Network

Consider the following simple stochastic model of gene expression:

$\cdot \underset{k_2 T}{\overset{k_1 T}{\rightleftharpoons}}$ mRNA

mRNA $\xrightarrow{k_3 T}$ mRNA + P

$\mathrm{P} \xrightarrow{k_4 T} \cdot$

where the k_i's are rate parameters determining the relative probability of production and degradation events and T is a scaling parameter that determines the time-scale that characterizes the dynamics of the reaction network. In this example, we assume that we are able to measure the amount of proteins P present in the system for each time but are unable to measure the amount of $mRNA$.

VII.2.4.1 Motivation

Suppose that for some values of the k_i's and some value of T we have a collection of realizations of the process. Suppose further that through some other means, we are able to obtain the values of the k_i's used to generate the process. We can then construct a set of candidate models of the process parameterized by the unknown parameter T.

Now, if we are only able to make measurements once the process has reached stationarity then by simply measuring the amount of protein present at one time instant will be insufficient to distinguish between different values of T. This is because if $\hat{\pi}$ is a stationary distribution for one value of T, then it is a stationary distribution for all values of T. Consider the Chemical Master Equation:

$$\dot{p} = Ap \tag{VII.2.9}$$

If $A\hat{\pi} = 0$ then for every scaled CME $TA\hat{\pi} = 0$. As a result, if we only measure the amount of protein at a single fixed time, then the protein distributions generated by each process will be the same.

Now, if we are able to take a second measurement we can do better since we will then have statistical information about how the protein counts are correlated in time. For small values of T the system will evolve slowly and the two measurements will be tightly correlated for a longer amount of time. For larger values of T the system will evolve quickly and the time correlation will be weaker.

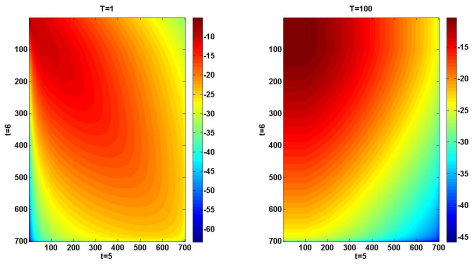


Figure VII.5: Log plot of the joint distributions of Z for processes corresponding to $T = 1$ and $T = 100$. Note that for the slower time-scale $T = 1$ the protein counts are tightly correlated while for $T = 100$ protein counts are less strongly correlated. Since the Wasserstein pseudometrics compare properties of probability distributions, these differences should be reflected in the Wasserstein distances.

VII.2.4.2 Implementation

For this example I chose the parameter values $k_1 = k_2 = k_4 = 1, k_3 = 100$ for the rate constants and chose the value of $T = 10$ as the unknown parameter for the

observed process. Realizations of the corresponding process were generated using the Gillespie's Stochastic Simulation Algorithm and sampled for protein counts at time points $t = 5, 6$ to generate an approximate joint distribution. Candidate models for the observed process were then generated using values of $T \in \{1, 2, 3, 4, 5, 10, 100\}$ and the Finite State Projection Algorithm was used to obtain approximate joint distributions for the same sampling procedure. Once the approximate joint distributions were obtained, a further state aggregation was performed to bin protein count values into one of 10 distinct values. This was done to reduce the number of optimization variables so that solving the linear programs would be tractable. This is also reflective of the resolution of protein count data that is obtained from flourescence experiments.

For this example, two reporter random variables were used: $Z^1(\omega)$ measured the amount of protein present at time $t = 5$ and then performed the state aggregation while $Z^2(\omega)$ measured at time $t = 6$ and performed the state aggregation. A family of multicriterion Wasserstein distances was then calculated using CVX. Calculating each multicriterion Wasserstein distance involved solving a linear program with 10^4 optimization variables, 10^4 inequality constraints, and 201 equality constraints. For each of the 7 candidate models, a paramter sweep through 50 different values of λ was calculated.

VII.2.4.3 Results

Figure VII.6 plots tradeoff curves corresponding to the parameter sweep through λ for different values of T. As T increases to to the value used for the observed process, the value of each pseudometric decreases to a minimal value corresponding to the correct parameter value. After this point, further increasing of T does not make any change to the pseudometric values. According to these comparison criteria, there is no distinguishing processes with mixing times beyond some threshhold. The threshhold is most likely determined by the time separation of the two samples that are taken, though without further data, I can't assert this conclusively.

Note that in the limits $\lambda_1 \to 0$ or $\lambda_2 \to 0$ the pseudometric values for each T (except for $T = 1$) go to the same value. This is because in these limits, the multicriterion comparison approaches the single criterion comparison using just a single reporter random variable. The fact that the pseudometric corresponding to $T = 1$ does not conform to this limit most likely indicates that the process has not reached the stationary distribution by the first time sample.

VII.2.5 Conclusions/Future Work

This project has shown that it is possible to do model comparison using multiple comparison criteria using Wasserstein pseudometric. A simple parameter identification problem was investigated using multiple reporter random variables.

Further theoretical work that could be done would be to quantify the error in

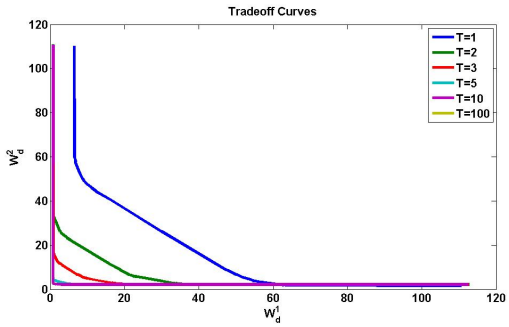


Figure VII.6: Tradeoff curves for various values of T. As T increases to the correct value, the value of each of the pseudometrics decreases, that is, the observed process and candidate process become more similar with respect to the comparison criteria. Increasing T beyond the actual value does not change the pseudometric values showing that under these comparison criteria, processes with fast mixing times are indistinguishable.

the pseudometric estimate due to use of sampled data to approximate the distributions. [TK10] showed that as the number of samples approaches infinity, then the approximate pseudometrics in their framework would approach the true value almost surely. A similar result can probably be stated here. In addition it would be useful to characterize the rate of convergence; my guess is that a central limit theorem argument is probably applicable. [TK08] uses bootstrap confidence intervals to quantify the uncertainty in the estimate. A tighter estimate might be possible.

Bibliography

Papers and Theses

[BS72] R. H. Bartels and G. W. Stewart. "Solution of the matrix equation AX + XB = C [F4]". In: *Commun. ACM* 15.9 (Sept. 1972), pp. 820–826. ISSN: 0001-0782. DOI: 10.1145/361573.361582 (cit. on p. 83).

[Bur+06] K. Burrage et al. "A Krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems". In: *Proceedings of the Markov 150th Anniversary Conference*. Boson Books, Raleigh, NC. 2006, pp. 21–38 (cit. on pp. 57, 60).

[CC70] J. D. Caroll and J. J. Chang. "Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition". In: *Psychometrika* 35 (3 1970), pp. 283–319. ISSN: 0033-3123. DOI: 10.1007/BF02310791 (cit. on p. 11).

[CV09] J. I. Cirac and F. Verstraete. “Renormalization and tensor product states in spin chains and lattices”. In: *Journal of Physics A: Mathematical and Theoretical* 42.50 (2009), p. 504004 (cit. on pp. 11, 125).

[Deu+08] P. Deuflhard et al. “Adaptive Discrete Galerkin Methods Applied to the Chemical Master Equation”. In: *Sci. Comput.* 30.6 (2008), pp. 2990–3011 (cit. on pp. 67, 78, 132).

[DO11] S. V. Dolgov and I. V. Oseledets. *Solution of linear systems and matrix inversion in the TT-format (submitted to SISC)*. Preprint 19. Max-Planck-Institut für Mathematik in den Naturwissenschaften, 2011 (cit. on pp. 22, 60, 86).

[DS13] S. V. Dolgov and D. V. Savostyanov. *Alternating minimal energy methods for linear systems in higher dimensions. Part I: SPD systems*. arXiv preprint 1301.6068. Jan. 2013 (cit. on p. 125).

[Elo+02] M. Elowitz et al. “Stochastic Gene Expression in a Single Cell”. In: *Nature* 297.5584 (2002), pp. 1183–1186 (cit. on p. 2).

[Eng09] S. Engblom. “Spectral approximation of solutions to the chemical master equation”. In: *Journal of Computational and Applied Mathematics* 229.1 (2009), pp. 208–221. ISSN: 0377-0427. DOI: 10.1016/j.cam.2008.10.029 (cit. on pp. 78, 132).

[Fei79] M. Feinberg. “Lectures on chemical reaction networks”. In: *Notes of lectures given at the Mathematics Research Center, University of Wisconsin* (1979) (cit. on p. 49).

[GCC00] T. S. Gardner, C. R. Cantor, and J. J. Collins. “Construction of a genetic toggle switch in Escherichia coli”. In: *Nature* 403.6767 (2000), pp. 339–342. DOI: http://dx.doi.org/10.1038/35002131 (cit. on p. 65).

[GH07] L. Grasedyck and W. Hackbusch. “A Multigrid Method to Solve Large Scale Sylvester Equations”. In: *SIAM Journal on Matrix Analysis and Applications* 29.3 (2007), pp. 870–894. DOI: 10.1137/040618102. eprint: http://epubs.siam.org/doi/pdf/10.1137/040618102 (cit. on p. 84).

[Gil76] D. T. Gillespie. “A general method for numerically simulating the stochastic time evolution of coupled chemical reactions”. In: *Journal of Computational Physics* 22.4 (1976), pp. 403–434. ISSN: 0021-9991. DOI: 10.1016/0021-9991(76)90041-3 (cit. on p. 3).

[GNVec] G. Golub, S. Nash, and C. Van Loan. “A Hessenberg-Schur method for the problem AX + XB= C”. In: *Automatic Control, IEEE Transactions on* 24.6 (Dec), pp. 909–913. ISSN: 0018-9286. DOI: 10.1109/TAC.1979.1102170 (cit. on p. 83).

[Gra] L. Grasedyck. “Polynomial Approximation in Hierarchical Tucker Format by Vector-Tensorization. 2010”. In: *Hackbusch, Tensorisation of Vectors*

and their Efficient Convolution, W. Hackbusch, L. Grasedyck, R. Schneider, Multilevel tensorisation (in preparation) () (cit. on p. 21).

[Gra04] L. Grasedyck. "Existence of a low rank or fffdfffdfffdfffdfffd?-matrix approximant to the solution of a Sylvester equation". In: *Numerical Linear Algebra with Applications* 11.4 (2004), pp. 371–389. ISSN: 1099-1506. DOI: 10.1002/nla.366 (cit. on pp. 83, 86).

[Gra10a] L. Grasedyck. "Hierarchical Singular Value Decomposition of Tensors". In: *SIAM Journal on Matrix Analysis and Applications* 31.4 (2010), pp. 2029–2054. DOI: 10.1137/090764189 (cit. on pp. 7, 11).

[Gra10b] L. Grasedyck. *Polynomial Approximation in Hierarchical Tucker Format by Vector-Tensorization*. Preprint 308. Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Apr. 2010 (cit. on p. 54).

[Hac+12] W. Hackbusch et al. "Use of tensor formats in elliptic eigenvalue problems". In: *Numerical Linear Algebra with Applications* 19.1 (2012), pp. 133–151. ISSN: 1099-1506. DOI: 10.1002/nla.793 (cit. on p. 44).

[HAM82] S. J. HAMMARLING. "Numerical Solution of the Stable, Non-negative Definite Lyapunov Equation Lyapunov Equation". In: *IMA Journal of Numerical Analysis* 2.3 (1982), pp. 303–323. DOI: 10.1093/imanum/2.3.303. eprint: http://imajna.oxfordjournals.org/content/2/3/303.full.pdf+html (cit. on pp. 83, 85).

[Hås90] J. Håstad. "Tensor rank is NP-complete". In: *Journal of Algorithms* 11.4 (1990), pp. 644–654. ISSN: 0196-6774. DOI: 10.1016/0196-6774(90)90014-6 (cit. on p. 12).

[HG11] M. Hegland and J. Garcke. "On the numerical solution of the chemical master equation with sums of rank one tensors". In: *ANZIAM Journal* 52 (2011). ISSN: 1446-8735 (cit. on p. 50).

[Hit26] F. L. Hitchcock. "The expression of a tensor or a polyadic as a sum of products". In: *Journal of Mathematics and Physics* 6.1 (Nov. 1926), pp. 164–189 (cit. on pp. 7, 11).

[HK09] W. Hackbusch and S. Kühn. "A New Scheme for the Tensor Representation". In: *Journal of Fourier Analysis and Applications* 15.5 (2009). 10.1007/s00041-009-9094-9, pp. 706–722. ISSN: 1069-5869. DOI: 10.1007/s00041-009-9094-9 (cit. on pp. 7, 11).

[HL07] A. Hellander and P. Lötstedt. "Hybrid method for the chemical master equation". In: *Journal of Computational Physics* 227.1 (2007), pp. 100–122. ISSN: 0021-9991. DOI: DOI:10.1016/j.jcp.2007.07.020 (cit. on p. 78).

[HL09] C. Hillar and L.-H. Lim. "Most tensor problems are NP hard". In: *arXiv* abs/0911.1393 (2009) (cit. on p. 12).

[HRS12] S. Holtz, T. Rohwedder, and R. Schneider. "The alternating linear scheme for tensor optimization in the Tensor Train format". In: *SIAM Journal on Scientific Computing* 34.2 (2012), A683–A713. DOI: 10.1137/100818893 (cit. on pp. 22, 61).

[Jah10] T. Jahnke. "An adaptive wavelet method for the chemical master equation". In: *SIAM Journal on Scientific Computing* 31 (2010), p. 4373 (cit. on p. 78).

[JH07] T. Jahnke and W. Huisinga. "Solving the chemical master equation for monomolecular reaction systems analytically". In: *Journal of mathematical biology* 54.1 (2007), pp. 1–26 (cit. on pp. 3, 51, 64).

[Kaz+14] V. Kazeev et al. "Direct Solution of the Chemical Master Equation Using Quantized Tensor Trains". In: *PLoS Comput Biol* 10.3 (Mar. 2014), e1003359. DOI: 10.1371/journal.pcbi.1003359 (cit. on pp. 8, 24, 53–57).

[Kho11] B. N. Khoromskij. "$\mathcal{O}(d \log N)$-Quantics Approximation of N-d Tensors in High-Dimensional Numerical Modeling". In: *Constructive Approximation* 34.2 (2011). 10.1007/s00365-011-9131-1, pp. 257–280. ISSN: 0176-4276. DOI: 10.1007/s00365-011-9131-1 (cit. on pp. 20, 21).

[KK12] V. A. Kazeev and B. N. Khoromskij. "Low-rank explicit QTT representation of the Laplace operator and its inverse". In: *SIAM Journal on Ma-*

trix Analysis and Applications 33.3 (2012), pp. 742–758. DOI: 10.1137/100820479 (cit. on pp. 21, 87, 107, 109).

[KKT11] V. A. Kazeev, B. N. Khoromskij, and E. E. Tyrtyshnikov. *Multiulevel Toeplitz matrices generated by tensor-structured vectors and convolution with logarithmic complexity*. Preprint 36. Max-Planck-Institut für Mathematik in den Naturwissenschaften, 2011 (cit. on p. 21).

[Koe+10] H. Koeppl et al. "Probability metrics to calibrate stochastic chemical kinetics". In: *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. IEEE. 2010, pp. 541–544 (cit. on pp. 139, 141).

[KRS12] V. Kazeev, O. Reichmann, and C. Schwab. *hp-DG-QTT solution of high-dimensional degenerate diffusion equations*. Research Report 11. Seminar for Applied Mathematics, ETH Zürich, 2012 (cit. on pp. 56, 60).

[KRS13] V. Kazeev, O. Reichmann, and C. Schwab. "Low-rank tensor structure of linear diffusion operators in the TT and QTT formats". In: *Linear Algebra and its Applications* (2013). DOI: 10.1016/j.laa.2013.01.009 (cit. on p. 21).

[KS13] V. Kazeev and C. Schwab. *Tensor approximation of stationary distributions of chemical reaction networks*. Tech. rep. 2013-18. Switzerland: Seminar for Applied Mathematics, ETH Zürich, 2013 (cit. on pp. 49, 78).

[LC09] L.-H. Lim and P. Comon. "Nonnegative approximations of nonnegative tensors". In: *Journal of Chemometrics* 23.7-8 (2009), pp. 432–441. ISSN: 1099-128X. DOI: 10.1002/cem.1244 (cit. on p. 11).

[MA97] H. H. McAdams and A. Arkin. "Stochastic mechanisms in gene expression". In: *PNAS* 94.3 (1997), pp. 814–819. DOI: 10.1073/pnas.94.3.814 (cit. on p. 2).

[MK06] B. Munsky and M. Khammash. "The finite state projection algorithm for the solution of the chemical master equation". In: *The Journal of Chemical Physics* 124.4, 044104 (2006), p. 044104. DOI: 10.1063/1.2145882 (cit. on pp. 50, 51).

[MK08] B. Munsky and M. Khammash. "The Finite State Projection Approach for the Analysis of Stochastic Noise in Gene Networks". In: *Automatic Control, IEEE Transactions on* 53.Special Issue (Jan. 2008), pp. 201–214. ISSN: 0018-9286. DOI: 10.1109/TAC.2007.911361 (cit. on pp. 67, 68).

[Moo81] B. Moore. "Principal component analysis in linear systems: Controllability, observability, and model reduction". In: *Automatic Control, IEEE Transactions on* 26.1 (1981), pp. 17–32 (cit. on pp. 82, 103).

[NHK12] M. Nip, J. Hespanha, and M. Khammash. "A spectral methods-based solution of the Chemical Master Equation for gene regulatory networks". In: *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on.*

Dec. 2012, pp. 5354–5360. DOI: 10.1109/CDC.2012.6425804 (cit. on p. 8).

[NHK13] M. Nip, J. P. Hespanha, and M. Khammash. "Direct numerical solution of algebraic Lyapunov equations for large-scale systems using Quantized Tensor Trains". In: *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. Dec. 2013, pp. 1950–1957. DOI: 10.1109/CDC.2013.6760167 (cit. on p. 8).

[Ose09a] I. Oseledets. "A new tensor decomposition". In: *Doklady Mathematics* 80 (1 2009), pp. 495–496. ISSN: 1064-5624. DOI: 10.1134/S1064562409040115 (cit. on p. 11).

[Ose09b] I. Oseledets. "Approximation of matrices with logarithmic number of parameters". In: *Doklady Mathematics* 80 (2 2009), pp. 653–654. ISSN: 1064-5624. DOI: 10.1134/S1064562409050056 (cit. on pp. 20, 21).

[Ose10a] I. V. Oseledets. "Approximation of $2^d \times 2^d$ matrices using tensor decomposition". In: *SIAM Journal on Matrix Analysis and Applications* 31.4 (2010), pp. 2130–2145. DOI: 10.1137/090757861 (cit. on p. 20).

[Ose10b] I. V. Oseledets. *QTT decomposition of the characteristic function of a simplex*. Personal communication. Sept. 2010 (cit. on p. 24).

[Ose11] I. V. Oseledets. "Tensor Train decomposition". In: *SIAM Journal on Scientific Computing* 33.5 (2011), pp. 2295–2317. ISSN: 10648275. DOI: 10.1137/090752286 (cit. on pp. 6, 12, 14–17, 19, 30, 34, 35, 37, 48, 58).

[Ose13a] I. V. Oseledets. "Constructive representation of functions in tensor formats". In: *Constructive Approximation* 37 (2013), pp. 1–18 (cit. on p. 21).

[Ose13b] I. Oseledets. "Constructive Representation of Functions in Low-Rank Tensor Formats". English. In: *Constructive Approximation* 37.1 (2013), pp. 1–18. ISSN: 0176-4276. DOI: 10.1007/s00365-012-9175-x (cit. on p. 54).

[OT09] I. V. Oseledets and E. E. Tyrtyshnikov. "Breaking the curse of dimensionality, or how to use SVD in many dimensions". In: *SIAM Journal on Scientific Computing* 31.5 (Oct. 2009), pp. 3744–3759. DOI: 10.1137/090748330 (cit. on pp. 6, 12, 15).

[Pen99] T. Penzl. "A Cyclic Low-Rank Smith Method for Large Sparse Lyapunov Equations". In: *SIAM Journal on Scientific Computing* 21.4 (1999), pp. 1401–1418. DOI: 10.1137/S1064827598347666. eprint: http://epubs.siam.org/doi/pdf/10.1137/S1064827598347666 (cit. on p. 84).

[Row05] C. Rowley. "Model reduction for fluids, using balanced proper orthogonal decomposition". In: *International Journal of Bifurcation and Chaos* 15.03 (2005), pp. 997–1013 (cit. on pp. 103, 104).

[RU12] T. Rohwedder and A. Uschmajew. *Local convergence of alternating schemes for optimization of convex problems in the TT format*. Preprint 112. DFG-Schwerpunktprogramm 1324, Aug. 2012 (cit. on pp. 22, 61).

[Saa92] Y. Saad. "Analysis of Some Krylov Subspace Approximations to the Matrix Exponential Operator". In: *SIAM Journal on Numerical Analysis* 29.1 (1992), pp. 209–228. DOI: 10.1137/0729014. eprint: http://dx.doi.org/10.1137/0729014 (cit. on p. 57).

[Sch07] E. Schmidt. "Zur Theorie der linearen und nicht linearen Integralgleichungen Zweite Abhandlung". German. In: *Mathematische Annalen* 64.2 (1907), pp. 161–174. ISSN: 0025-5831. DOI: 10.1007/BF01449890 (cit. on p. 10).

[Sid98] R. B. Sidje. "Expokit: A Software Package for Computing Matrix Exponentials". In: *ACM Trans. Math. Softw.* 24.1 (Mar. 1998), pp. 130–156. ISSN: 0098-3500. DOI: 10.1145/285861.285868 (cit. on pp. 59, 68).

[SL08] V. de Silva and L.-H. Lim. "Tensor Rank and the Ill-Posedness of the Best Low-Rank Approximation Problem". In: *SIAM Journal on Matrix Analysis and Applications* 30.3 (2008), pp. 1084–1127. DOI: 10.1137/06066518X (cit. on p. 12).

[SLE09] P. Sjfffdfffdberg, P. Lfffdfffdtstedt, and J. Elf. "Fokkerfffdfffdfffd Planck approximation of the master equation in molecular biology". In: *Comput-*

ing and Visualization in Science 12 (1 2009). 10.1007/s00791-006-0045-6, pp. 37–50. ISSN: 1432-9360 (cit. on p. 67).

[SOS04] J. Schwender, J. Ohlrogge, and Y. Shachar-Hill. "Understanding flux in plant metabolic networks". In: *Current Opinion in Plant Biology* 7.3 (2004), pp. 309–317. ISSN: 1369-5266. DOI: 10.1016/j.pbi.2004.03.016 (cit. on pp. 71, 72).

[SPA05] M. Samoilov, S. Plyasunov, and A. P. Arkin. "Stochastic amplification and signaling in enzymatic futile cycles through noise-induced bistability with oscillations". In: *Proceedings of the National Academy of Sciences of the United States of America* 102.7 (2005), pp. 2310–2315. DOI: 10.1073/pnas.0406841102. eprint: http://www.pnas.org/content/102/7/2310.full.pdf+html (cit. on pp. 72, 73).

[SRK13] P. W. Sheppard, M. Rathinam, and M. Khammash. "SPSens: a software package for stochastic parameter sensitivity analysis of biochemical reaction networks". In: *Bioinformatics* 29.1 (2013), pp. 140–142. DOI: 10.1093/bioinformatics/bts642. eprint: http://bioinformatics.oxfordjournals.org/content/29/1/140.full.pdf+html (cit. on p. 62).

[SS00] D. Schötzau and C. Schwab. "An hp a priori error analysis of the DG time-stepping method for initial value problems". English. In: *Calcolo* 37

(4 2000), pp. 207–232. ISSN: 0008-0624. DOI: 10.1007/s100920070002 (cit. on p. 55).

[SS05] G. Shi and C.-J. Shi. "Model-order reduction by dominant subspace projection: Error bound, subspace computation, and circuit applications". In: *Circuits and Systems I: Regular Papers, IEEE Transactions on* 52.5 (2005), pp. 975–993 (cit. on pp. 97, 98).

[SS86] Y. Saad and M. Schultz. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems". In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (1986), pp. 856–869. DOI: 10.1137/0907058. eprint: http://dx.doi.org/10.1137/0907058 (cit. on p. 57).

[TK08] D. Thorsley and E. Klavins. "Model reduction of stochastic processes using Wasserstein pseudometrics". In: *American Control Conference, 2008.* June 2008, pp. 1374–1381. DOI: 10.1109/ACC.2008.4586684 (cit. on pp. 139, 141, 146).

[TK10] D. Thorsley and E. Klavins. "Approximating stochastic biochemical processes with wasserstein pseudometrics". In: *Systems Biology, IET* 4.3 (May 2010), pp. 193–211. ISSN: 1751-8849. DOI: 10.1049/iet-syb.2009.0039 (cit. on pp. 139–141, 146).

[VCM09] F. Verstraete, J. I. Cirac, and V. Murg. *Matrix Product States, Projected Entangled Pair States, and variational renormalization group methods for*

quantum spin systems. arXiv preprint 0907.2796. July 2009 (cit. on pp. 12, 125).

[Vid03] G. Vidal. "Efficient Classical Simulation of Slightly Entangled Quantum Computations". In: *Phys. Rev. Lett.* 91.14 (Oct. 2003), p. 147902. DOI: 10.1103/PhysRevLett.91.147902 (cit. on pp. 12, 13, 22, 60).

[VPC04] F. Verstraete, D. Porras, and J. I. Cirac. "Density Matrix Renormalization Group and Periodic Boundary Conditions: A Quantum Information Perspective". In: *Phys. Rev. Lett.* 93.22 (Nov. 2004), p. 227205. DOI: 10.1103/PhysRevLett.93.227205 (cit. on pp. 12, 13, 22, 60).

[Whi93] S. R. White. "Density-matrix algorithms for quantum renormalization groups". In: *Phys. Rev. B* 48.14 (Oct. 1993), pp. 10345–10356. DOI: 10.1103/PhysRevB.48.10345 (cit. on pp. 6, 12, 13, 22, 60).

Books

[Bel61] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton, NJ: Princeton University Press, 1961 (cit. on pp. 10, 15).

[Cic+09] A. Cichocki et al. *Nonnegative matrix and tensor factorizations: application to exploratory multi-way data analysis and blind separation*. Wiley, 2009. ISBN: 978-0-470-74728-5 (cit. on p. 11).

[DP00] G. E. Dullerud and F. Paganini. *A Course in Robust Control Theory: A Convex Approach*. Springer, 2000 (cit. on pp. 82, 103).

[Hac12] W. Hackbusch. *Tensor Spaces and Numerical Tensor Calculus*. Vol. 42. Springer Series in Computational Mathematics. Springer, 2012. DOI: 10.1007/978-3-642-28027-6 (cit. on p. 9).

[Kam92] N. G. van Kampen. *Stochastic Processes in Physics and Chemistry*. Amsterdam and New York: North-Holland, 1992 (cit. on pp. 3, 50, 127).

[Saa03] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2003 (cit. on p. 44).