tgEDMD: Approximation of the Kolmogorov Operator in Tensor Train Format

Marvin Lücke[1] [ID] · **Feliks Nüske**[2,3] [ID]

Abstract

Extracting information about dynamical systems from models learned off simulation data has become an increasingly important research topic in the natural and engineering sciences. Modeling the Koopman operator semigroup has played a central role in this context. As the approximation quality of any such model critically depends on the basis set, recent work has focused on deriving data-efficient representations of the Koopman operator in low-rank tensor formats, enabling the use of powerful model classes while avoiding over-fitting. On the other hand, detailed information about the system at hand can be extracted from models for the infinitesimal generator, also called Kolmogorov backward operator for stochastic differential equations. In this work, we present a data-driven method to efficiently approximate the generator using the tensor train (TT) format. The centerpiece of the method is a TT representation of the tensor of generator evaluations at all data sites. We analyze consistency and complexity of the method, present extensions to practically relevant settings, and demonstrate its applicability to benchmark numerical examples.

Communicated by Felix Otto.

Marvin Lücke and Feliks Nüske contributed equally to this work.

✉ Feliks Nüske
  nueske@mpi-magdeburg.mpg.de

1 Zuse Institute Berlin, Berlin, Germany

2 Institute of Mathematics, Paderborn University, Paderborn, Germany

3 Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany

🍀 Springer

1 Introduction

Learning models for complex dynamical systems based on simulation or measurement data has become a vibrant and successful research field, with applications in fluid dynamics, engineering, biophysics, economics, and many others. A significant body of work revolves around the statistical description of a dynamical system using the *Koopman operator*, see Koopman (1931), Dellnitz and Junge (1999), Schütte et al. (1999), Mezić (2005), Budišić et al. (2012), Klus et al. (2016, 2018b) and Mauroy et al. (2020). Major algorithmic frameworks originating from this line of research include, for example, *(extended) dynamic mode decomposition*, or (E)DMD (Schmid 2010; Williams et al. 2015a), *Markov state modeling* (Dellnitz and Junge 1999; Schütte et al. 1999; Prinz et al. 2011; Bowman et al. 2014), the *variational approach to conformational dynamics* (VAC) (Noé and Nüske 2013; Nüske et al. 2014) and its generalization *variational approach to Markov processes* (VAMP) (Wu and Noé 2020), and the SINDy approach to system identification (Brunton et al. 2016). At their core, most of these techniques aim at solving a linear regression problem, the solution of which consistently approximates the Koopman operator on a pre-selected model class in the infinite data limit. In order to reduce the influence of statistical noise when using finite data, the problem dimension is often reduced via a projection onto the leading components of a singular value decomposition (SVD) of the data matrix. This de-noising and whitening technique is also known as *AMUSE* algorithm (Tong et al. 1990; Klus et al. 2018b). Nonlinear machine learning techniques have also been incorporated into the Koopman framework, see Mardt et al. (2018), Lusch et al. (2018), and Otto and Rowley (2019) for deep learning approaches, and Williams et al. (2015b) and Klus et al. (2020a, b) for kernel-based methods.

This work combines two recent developments along these lines: the first is the use of a specific low-rank tensor representation, the *tensor train format* (TT format), in order to balance sufficient representational power of the model class and manageable computational complexity. The TT format has been known in various scientific fields for decades, especially in quantum chemistry (Affleck et al. 1987; Östlund and Rommer 1995; Szalay et al. 2015). It was introduced in the mathematical literature in Oseledets and Tyrtyshnikov (2009) and Oseledets (2011), and was applied specifically to the Fokker–Planck equation of an SDE in Chertkov and Oseledets (2021). Within the Koopman framework, it was first introduced in Nüske et al. (2016) and Klus and Schütte (2016). Building on results from Klus et al. (2018a), a multi-linear version of the AMUSE algorithm to be used with the TT format was presented by one of the authors in Nüske et al. (2021a). In particular, it was shown that, due to the TT structure of all quantities involved, one can efficiently compute a representation of the Koopman operator on a reduced subspace defined by a multi-linear SVD. Consistency of this representation for fixed TT ranks was also proven.

The second development we will build on is the extension of EDMD and related approaches to approximate the infinitesimal generator of the Koopman operator semi-group. The EDMD method can be adapted in a fairly direct manner for the purpose of generator approximation, as shown in Klus et al. (2020c). The resulting method, called *gEDMD*, can further be utilized to identify system parameters, to define coarse grained models, and for system control. For alternative approaches to generator approxima-

tion, please see Froyland et al. (2013), Kaiser et al. (2018, 2021), Giannakis (2019) and Mauroy and Goncalves (2019).

The core contribution of this paper is to combine these developments for systems driven by stochastic differential equations (SDE), where the Koopman generator is also called Kolmogorov backward operator. The result is a data-driven method to approximate the Kolmogorov operator on a multi-linear subspace of a tensor product function space. In more detail, we

1. Derive a TT representation formula for tensors which are given as a certain structured sum of rank-one tensors, extending a result from Kazeev et al. (2013). This representation is then used to deduce the TT structure of the tensor of generator evaluations at all data sites, which is the missing piece when translating existing methods to the approximation of the generator. We derive separate representations for reversible and general SDE systems.
2. Present the resulting data-driven algorithm, called tgEDMD, which combines the methods suggested in Refs. Klus et al. (2020c) and Nüske et al. (2021a). Further, we analyze its computational complexity and consistency.
3. Present extensions of the method to two practically relevant settings: first, we show how importance sampling ratios can be incorporated if the sampling distribution differs from the reference distribution for generator approximation. Second, we explain how basis sets defined on reduced variables can be used to approximate the generator of an effective dynamics in TT format, building on results from Refs. Zhang et al. (2016) and Klus et al. (2020c).
4. Illustrate the capabilities of the proposed methods by approximating the generator of a four-dimensional model system, and the effective generator for molecular dynamics of the deca-alanine peptide on a set of torsion angle coordinates.

The rest of the paper is structured as follows: basic concepts regarding the Koopman framework, data-driven approximation, the AMUSE algorithm, and tensor trains are introduced in Sect. 2. The general TT representation formula and the TT structures of the generator data tensor are derived in Sect. 3. The main algorithm, its analysis, and the extensions are presented in Sect. 4. Numerical examples follow in Sect. 5.

2 Prerequisites

2.1 Kolmogorov Backward Operator

In this paper, we are concerned with the numerical approximation of the *Kolmogorov backward operator*

$$\mathcal{L}\phi(x) = \frac{1}{2}a(x) : \nabla^2\phi(x) + b(x) \cdot \nabla\phi(x), \tag{1}$$

where $b \in \mathbb{R}^d$ is a vector field and $a \in \mathbb{R}^{d \times d}$ is a field of symmetric positive semi-definite matrices (that is, $v^T a(x) v \geq 0$ for all x and $v \in \mathbb{R}^d$). Moreover, $\nabla\phi(x) \in \mathbb{R}^d$ is the gradient of the scalar function ϕ, $\nabla^2\phi(x)$ is the Hessian matrix of its second-order derivatives, and the colon : denotes the Frobenius inner product, $A : B = \text{tr}(A^T B)$, between matrices.

The differential operator $\mathcal{L}$ is closely linked to the statistical description of dynamical systems which are driven by a stochastic differential equation (SDE). For $t \geq 0$, let X_t be a stochastic process on a state space $\mathbb{X} \subset \mathbb{R}^d$, subject to the governing equation

$$\mathrm{d}X_t = b(X_t)\mathrm{d}t + \sigma(X_t)\mathrm{d}W_t. \tag{2}$$

In this context, the field b is called the *drift*, while the matrix field σ is called *diffusion*, which must be such that $\sigma\sigma^T = a$. Furthermore, W_t in (2) is the Brownian motion in d-dimensional space. Equation (2) can be understood in the sense that the infinitesimal change of the process X_t is proportional to the drift field b, but also subject to the random fluctuations of the Brownian motion W_t, modulated by σ. See Ref. Oksendal (2013) for a rigorous introduction to the dynamics of type (2).

The statistics of the process X_t are then captured by a family of linear operators, called the *Koopman semigroup* (Koopman 1931; Mezić 2005). For $t \geq 0$ and a function ϕ on $\mathbb{X}$ (usually called an *observable* of the system), the linear operator $\mathcal{K}^t$ acts as

$$\mathcal{K}^t\phi(x) = \mathbb{E}^x[\phi(X_t)] = \mathbb{E}[\phi(X_t)|X_0 = x].$$

The Kolmogorov operator $\mathcal{L}$ arises as the *infinitesimal generator* of this semigroup, that is, for suitable functions ϕ, we have

$$\mathcal{L}\phi = \lim_{t \to 0} \frac{1}{t}(\mathcal{K}^t - \mathrm{Id})\phi. \tag{3}$$

As described in the introduction, knowledge of the Koopman semigroup, and especially its generator, can be utilized for a thorough statistical analysis of the dynamical system X_t. Before discussing its numerical approximation, let us point out that the precise mathematical properties of the differential operator $\mathcal{L}$ and its domain of definition can be quite intricate. As these technical details are mostly irrelevant for the present study, we just briefly lay out a standard setting to serve as a guideline for the following, see Oksendal (2013) and Bakry et al. (2014) for an in-depth discussion of this setting.

(**A1**) The coefficients b, σ of the SDE (2) are such that global existence and uniqueness of solutions with continuous paths are guaranteed.

(**A2**) There is a unique, everywhere positive invariant density ρ for (2), with associated probability measure μ. Invariance of the measure μ can be characterized by the identity, for all ϕ and $t \geq 0$:

$$\int_{\mathbb{X}} \mathcal{K}^t\phi(x)\,\mathrm{d}\mu(x) = \int_{\mathbb{X}} \phi(x)\,\mathrm{d}\mu(x).$$

It follows that the operators $\mathcal{K}^t$ form a strongly continuous contraction semigroup on the Hilbert space of square-integrable functions with respect to the invariant measure μ:

$$L^2_\mu(\mathbb{X}) = \left\{ \phi : \mathbb{X} \mapsto \mathbb{R} \ \mid \int_{\mathbb{X}} |\phi(x)|^2\,\mathrm{d}\mu(x) < \infty \right\}.$$

The inner product for $\phi, \tilde{\phi} \in L^2_\mu(\mathbb{X})$ is given by $\langle \phi, \tilde{\phi} \rangle_\mu = \int_{\mathbb{X}} \phi \tilde{\phi} \, d\mu$. The domain $\mathcal{D}(\mathcal{L})$ of functions where the derivative (3) exists is a dense subspace of $L^2_\mu(\mathbb{X})$. If in addition the dynamics (2) are *reversible* with respect to the invariant measure μ, that is the detailed balance condition

$$\mathbb{P}^\mu(X_0 \in A, \ X_t \in B) = \mathbb{P}^\mu(X_0 \in B, \ X_t \in A)$$

holds for all $A, B \subset \mathbb{X}$ and $t \geq 0$, then the generator $\mathcal{L}$ is self-adjoint on its domain. Moreover, the following important relation holds for all $\phi, \tilde{\phi} \in \mathcal{D}(\mathcal{L})$:

$$\langle \mathcal{L}\phi, \tilde{\phi} \rangle_\mu = -\frac{1}{2} \int_{\mathbb{X}} \nabla\phi(x)^T \, a(x) \, \nabla\tilde{\phi}(x) \, d\mu(x). \tag{4}$$

2.2 Galerkin Projection and Data-Driven Estimation

Next, we consider the approximation of $\mathcal{L}$ on a finite-dimensional subspace $\mathbb{V}$, spanned by twice continuously differentiable functions contained in the domain, i.e., $\mathbb{V} \subset \mathcal{D}(\mathcal{L}) \cap C^2(\mathbb{X})$. Choosing $\mathbb{V}$ in this way ensures that $\mathcal{L}\phi(x)$ can be evaluated in the form (1) for all $\phi \in \mathbb{V}$. If a basis set $\psi = [\psi_1, \ldots, \psi_n]^T$ for $\mathbb{V}$ is given, we will briefly write $\psi(x) = [\psi_1(x), \ldots, \psi_n(x)]^T \in \mathbb{R}^n$ for the evaluation of all basis functions at a point x, and similarly $\mathcal{L}\psi(x) \in \mathbb{R}^n$ for the application of $\mathcal{L}$ to all basis functions at once. Moreover, if $X = [x_1, \ldots, x_m] \in \mathbb{R}^{d \times m}$ is a collection of data points in $\mathbb{X}$, we write

$$\Psi(X) = [\psi(x_1), \ldots, \psi(x_m)] \in \mathbb{R}^{n \times m}, \quad \mathcal{L}\Psi(X) = [\mathcal{L}\psi(x_1), \ldots, \mathcal{L}\psi(x_m)] \in \mathbb{R}^{n \times m},$$

and call these the *data matrix* and *generator data matrix*, respectively.

The orthogonal projection of $\mathcal{L}$ onto $\mathbb{V}$ can be represented by the matrix (w.r.t. the basis ψ):

$$L_{\mathbb{V}} := C(\psi)^{-1} A(\psi),$$
$$C(\psi)_{uv} = \langle \psi_u, \psi_v \rangle_\mu = \mathbb{E}^\mu[\psi_u \psi_v], \quad A(\psi)_{uv} = \langle \psi_u, \mathcal{L}\psi_v \rangle_\mu = \mathbb{E}^\mu[\psi_u \ \mathcal{L}\psi_v]. \tag{5}$$

As the integrals appearing in (5) are expectation values with respect to the probability measure μ, they can be approximated by Monte Carlo methods. From now on, we assume that $x_l, \ l \in \mathbb{N}$, is a sequence of random variables, such that for all integrable functions $\phi \in L^1_\mu(\mathbb{X})$ and almost all realizations of the x_l we have:

$$\lim_{m \to \infty} \frac{1}{m} \sum_{l=1}^{m} \phi(x_l) = \int_{\mathbb{X}} \phi(x) \, d\mu(x). \tag{6}$$

Equation (6) is of course satisfied if all x_l are i.i.d. with respect to μ. By the ergodic theorem (Chacon 1962), the same conclusion holds if x_l is taken as the l-th step of any

discretized trajectory of the dynamics (2), provided the initial condition is drawn from μ. As a consequence, for any finite realization $x_1, \ldots, x_m$ of the random variables x_l, we obtain an empirical Galerkin matrix $\hat{L}_{\mathbb{V}} = \hat{C}(\psi)^{-1} \hat{A}(\psi)$, where

$$\hat{C}(\psi) = \frac{1}{m} \sum_{l=1}^{m} \psi(x_l) \otimes \psi(x_l) = \frac{1}{m} \Psi(X) \Psi(X)^T, \tag{7}$$

$$\hat{A}(\psi) = \frac{1}{m} \sum_{l=1}^{m} \psi(x_l) \otimes \mathcal{L}\psi(x_l) = \frac{1}{m} \Psi(X) (\mathcal{L}\Psi(X))^T. \tag{8}$$

For the reversible case, the matrix $\hat{A}(\psi)$ can be calculated in a simpler form based on (4), requiring only first-order derivatives of the basis set. Define $\mathrm{d}\psi(x) = [\nabla\psi(x)\sigma(x)] \in \mathbb{R}^{n \times d}$, using the Jacobian $\nabla\psi \in \mathbb{R}^{n \times d}$. Then, introduce a modified generator data matrix $\mathrm{d}\Psi(X) = [\mathrm{d}\psi(x_1) \ldots \mathrm{d}\psi(x_m)] \in \mathbb{R}^{n \times dm}$. It follows from (4) that $A(\psi)$ can be approximated by

$$\hat{A}(\psi) = -\frac{1}{2m} \mathrm{d}\Psi(X) [\mathrm{d}\Psi(X)]^T, \tag{9}$$

noting that the summation extends over both the data size and the state space dimension. This conceptually simple way of approximating the Kolmogorov operator based on data using (7–9) has been labeled *generator extended dynamic mode decomposition (gEDMD)* in Klus et al. (2020c).

2.3 AMUSE Algorithm

The calculation of the Gramian matrix $\hat{C}(\psi)$ and its inverse may be avoided by means of the AMUSE Algorithm 1 (Tong et al. 1990; Klus et al. 2018b), which also serves as a tool for dimension reduction.

Algorithm 1 AMUSE

Input: data matrix $\Psi(X)$, generator data matrix $\mathcal{L}\Psi(X)$ or $\mathrm{d}\Psi(X)$.
Output: Reduced matrix representation $\hat{M}_r \in \mathbb{R}^{r \times r}$ or $\hat{M}_r^{\mathrm{rev}} \in \mathbb{R}^{r \times r}$ of $\mathcal{L}$, where $r \leq n$.

1: Compute rank-r SVD of $\Psi(X)$, i.e., $\Psi(X) \approx \hat{U}_r \, \hat{\Sigma}_r \, \hat{V}_r^{\top}$.
2: **(Non-reversible Case)**: Compute reduced matrix
    $\hat{M}_r = \hat{\Sigma}_r^{-1} \hat{U}_r^{\top} \Psi(X) \mathcal{L}\Psi(X)^T \hat{U}_r \, \hat{\Sigma}_r^{-1} = \hat{V}_r^{\top} \mathcal{L}\Psi(X)^T \hat{U}_r \, \hat{\Sigma}_r^{-1}$.
3: **(Reversible Case)**: Compute reduced matrix
    $\hat{M}_r^{\mathrm{rev}} = -\frac{1}{2} \hat{\Sigma}_r^{-1} \hat{U}_r^{\top} \mathrm{d}\Psi(X) \, \mathrm{d}\Psi(X)^T \hat{U}_r \, \hat{\Sigma}_r^{-1}$.

The rationale behind AMUSE is that, using the truncated SVD $\Psi(X) \approx \hat{U}_r \, \hat{\Sigma}_r \, \hat{V}_r^{\top}$ computed in line 1, the transformed basis $\hat{\eta}_r(x)^T := \psi(x)^T \hat{U}_r \hat{\Sigma}_r^{-1}$ is empirically orthonormal, hence computation of the Gramian matrix is no longer necessary. In order to compute a representation of the generator on the linear span of $\hat{\eta}_r$, the same

transformation just needs to be applied to $\hat{A}(\psi)$, which is what happens in lines 2 and 3 of Algorithm 1. The reduced matrices $\hat{M}_r$ and $\hat{M}_r^{\text{rev}}$ can also be computed by the following expressions, which do not require assembly of the full generator data matrices:

$$\hat{M}_r = \hat{V}_r^\top \left[\sum_{l=1}^m \mathcal{L}\psi(x_l) \otimes e_l \right]^T \hat{U}_r \, \hat{\Sigma}_r^{-1} = \sum_{l=1}^m (\hat{V}_r)_{l,:} \otimes \mathcal{L}\psi(x_l)^T \hat{U}_r \, \hat{\Sigma}_r^{-1}, \quad (10)$$

$$\hat{M}_r^{\text{rev}} = -\frac{1}{2} \sum_{l=1}^m \hat{\Sigma}_r^{-1} \hat{U}_r^\top \nabla\psi(x_l) \, a(x_l) \, \nabla\psi(x_l)^T \hat{U}_r \, \hat{\Sigma}_r^{-1}. \quad (11)$$

If one is interested in consistency of the reduced matrix for the infinite data limit, the spectral decomposition of the analytical Gramian needs to be considered. Denote this spectral decomposition by $C(\psi) = U\Sigma^2 U^T$, and for any $r \leq n$ we write $C(\psi) = U_r \Sigma_r^2 U_r^T + E_r$ for its truncation using r terms, with error term E_r. Now, if the reduced dimension r in Algorithm 1 is fixed, it can be shown that, unless the eigenvalue σ_r^2 of $C(\psi)$ is degenerate, the linear span of $\hat{\eta}_r$ converges almost surely to that of $\eta_r := \psi(x)^T U_r \Sigma_r^{-1}$, along with the projected generator on this space (see Nüske et al. 2021a [Proposition 3] for a proof). Alternatively, one can start with a fixed truncation parameter $\epsilon > 0$. Assume there is an $r(\epsilon)$ such that $\sigma_{r(\epsilon)} > \epsilon > \sigma_{r(\epsilon)+1}$ in the spectral decomposition of $C(\psi)$. Then, if the reduced rank in Algorithm 1 is chosen such that $\hat{\sigma}_r \geq \sqrt{m}\epsilon > \hat{\sigma}_{r+1}$, it follows by the same arguments that $r \to r(\epsilon)$ almost surely as $m \to \infty$, recalling that $\hat{C}(\psi) = \frac{1}{m}\Psi(X)\Psi(X)^T$. Hence, $\hat{M}_r$, $\hat{M}_r^{\text{rev}}$ provide consistent estimates of the projected generator on the linear span of $\eta_{r(\epsilon)}$. This data-dependent truncation strategy will also be used in the multi-linear case, see the discussion in Sect. 4.1.

2.4 Tensor Train Format

The choice of the finite-dimensional subspace $\mathbb{V}$ in Sect. 2.2 is of critical importance to the quality of the gEDMD approximation. A widely used approach to arrive at a large space with rich approximation properties is to consider all products of functions contained in a selection of elementary subspaces of moderate dimension. Many of the quantities introduced above will be endowed with a *tensor structure* in this case. For our purposes, it is sufficient to think of tensors as multi-dimensional arrays, that is, array entries are labeled by p different indices:

$$\mathbf{T} = \mathbf{T}_{u_1,\ldots,u_p}, \quad 1 \leq u_k \leq n_k, \quad 1 \leq k \leq p.$$

The number of indices p is called *order* of the tensor; the individual indices are often called *modes* with *mode sizes* n_k. A tensor can be *unfolded* into a matrix by grouping any selection $\mathrm{I} \subset \{1, \ldots, p\}$ of tensor indices along rows and all remaining tensor indices along columns. We denote this matrix by $\mathbf{T}|_\mathrm{I}$. Specifically, if I consists of the first k indices, we just write $\mathbf{T}|_k$ and call the resulting matrix the *mode-k unfolding*.

The *tensor product* of p vectors $v^{(k)} \in \mathbb{R}^{n_k}$ is defined by

$$\mathbf{T} = v^{(1)} \otimes \ldots \otimes v^{(p)}, \quad \mathbf{T}_{u_1,\ldots,u_p} = \prod_{k=1}^{p} v^{(k)}_{u_k},$$

generalizing the standard outer product between two vectors. A tensor of this type is a *rank-one* tensor. Every tensor can be written as a linear combination of finitely many rank-one tensors, but their number may be exponentially large. However, many tensors appearing in applications can be (approximately) represented by a much smaller number of parameters. Many different parametrizations, typically called *tensor formats*, have been put forward. Here, we focus exclusively on the tensor train format (TT format), introduced in Oseledets and Tyrtyshnikov (2009) and Oseledets (2011), where a tensor is represented as a contraction of multiple lower-order tensors:

Definition 2.1 A tensor $\mathbf{T}$ is said to be in the *TT format* if

$$\mathbf{T} = \sum_{s_0=1}^{r_0} \cdots \sum_{s_p=1}^{r_p} \bigotimes_{k=1}^{p} \mathbf{T}^{(k)}_{s_{k-1},:,s_k} = \sum_{s_0=1}^{r_0} \cdots \sum_{s_p=1}^{r_p} \mathbf{T}^{(1)}_{s_0,:,s_1} \otimes \cdots \otimes \mathbf{T}^{(p)}_{s_{p-1},:,s_p}. \quad (12)$$

The tensors $\mathbf{T}^{(k)} \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$ of order 3 are called *TT cores* and the numbers r_k are called *TT ranks*. It holds that $r_0 = r_p = 1$ and $r_k \geq 1$ for $k = 1, \ldots, p-1$. A TT core is *left-orthonormal* if

$$\left(\mathbf{T}^{(k)}|_2 \right)^{\top} \cdot \left(\mathbf{T}^{(k)}|_2 \right) = \mathrm{Id} \in \mathbb{R}^{r_k \times r_k}.$$

Remark 2.2 Sometimes, we will also allow the first or final rank to be greater than one, i.e., $r_0 > 1$ or $r_p > 1$. In this case, we understand $\mathbf{T}$ to be encoding multiple tensors, one for each pair of values of s_0 and s_p in (12).

The following formal representation for tensor trains has proven particularly useful. For a given tensor train $\mathbf{T}$ with cores $\mathbf{T}^{(k)} \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$, a single core is written as a two-dimensional matrix containing vectors as elements

$$[\![\mathbf{T}^{(k)}]\!] = \begin{bmatrix} \begin{bmatrix} \mathbf{T}^{(k)}_{1,:,1} & \cdots & \mathbf{T}^{(k)}_{1,:,r_k} \\ \vdots & \ddots & \vdots \\ \mathbf{T}^{(k)}_{r_{k-1},:,1} & \cdots & \mathbf{T}^{(k)}_{r_{k-1},:,r_k} \end{bmatrix} \end{bmatrix} \quad (13)$$

We then represent $\mathbf{T}$ by a formal matrix product, called *rank-core product* in Kazeev and Khoromskij (2012):

$$\mathbf{T} = [\![\mathbf{T}^{(1)}]\!] \otimes \cdots \otimes [\![\mathbf{T}^{(p)}]\!].$$

In this representation, we compute the tensor products of the matrix elements—which are vectors instead of scalars—and then sum over the columns and rows, just as for an ordinary matrix product. It is readily verified that the result is the same as (12). Further below, we will also require a tensor product for matrices, which is known as *Kronecker product*, and denoted by the same symbol. For matrices $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{r \times q}$, its block-wise definition is

$$A \otimes B \in \mathbb{R}^{mr \times nq} : \quad A \otimes B = \begin{bmatrix} A_{11}B & \dots & A_{1n}B \\ \vdots & \ddots & \vdots \\ A_{m1}B & \dots & A_{mn}B \end{bmatrix}. \tag{14}$$

The Kronecker product is equivalent to the standard tensor product applied to vectorized versions of the matrices A, B, but using a nonstandard ordering of their indices, see Hackbusch (2012) for details.

2.5 Basis Decomposition and Global SVD

We are now prepared to formulate the tensor-based approximation framework for the Kolmogorov operator. Suppose that for $1 \le k \le p$, $\mathbb{V}^k = \text{span}\{\psi_{k,1}, \dots, \psi_{k,n_k}\} \subset \mathcal{D}(\mathcal{L}) \cap C^2(\mathbb{X})$ is an n_k-dimensional subspace. Assume further the tensor product

$$\mathbb{V} = \bigotimes_{k=1}^{p} \mathbb{V}^k = \text{span}\{\psi_{1,u_1} \cdot \ldots \cdot \psi_{p,u_p} \mid 1 \le u_k \le n_k, \ 1 \le k \le p\}$$

is also contained in $\mathcal{D}(\mathcal{L})$. We denote the basis formed by all the products above by Ψ (assuming their linear independence). The evaluation of Ψ at $x \in \mathbb{X}$ is a rank-one tensor of order p, while the analogue of the data matrix $\Psi(X)$ is a tensor of order $p + 1$, which we call the *data tensor*:

$$\Psi(X)_{u_1,\dots,u_p,l} = \psi_{1,u_1}(x_l) \cdot \ldots \cdot \psi_{p,u_p}(x_l).$$

Tensor-based counterparts $\mathcal{L}\Psi(x)$, $\mathcal{L}\Psi(X)$, and $d\Psi(x)$, $d\Psi(X)$ of the generator data matrices can be defined the same way. Unless the n_k and p are very small, these data tensors cannot even be held in memory, let alone be used to compute the Galerkin matrices in (7–9). Therefore, we follow the lines of the AMUSE approach in Sect. 2.3, but notice that we cannot compute an SVD of the unfolded data tensor $\Psi(X)|_p$ either, necessitating further modifications.

As shown in Gelß et al. (2019), the data tensor is a sum of m rank-one tensors, and hence possesses a TT representation with all ranks equal to m, and the following TT cores (using the rank-core notation introduced in (13)):

$$\left[\!\left[\Psi(X)^{(1)} \right]\!\right] = \left[\!\left[\psi_1(x_1) \dots \psi_1(x_m) \right]\!\right], \qquad \left[\!\left[\Psi(X)^{(p+1)} \right]\!\right] = \left[\!\left[\begin{matrix} e_1 \\ \vdots \\ e_m \end{matrix} \right]\!\right],$$

$$\left[\!\!\left[\boldsymbol{\Psi}(X)^{(k)}\right]\!\!\right] = \left[\!\!\left[\begin{matrix} \psi_k(x_1) & & 0 \\ & \ddots & \\ 0 & & \psi_k(x_m) \end{matrix}\right]\!\!\right], \ 2 \le k \le p. \tag{15}$$

In the final core above, e_l denote the canonical unit vectors in $\mathbb{R}^m$. Moreover, by a sequence of SVDs applied to the cores of (15), a multi-linear approximation of the true SVD for $\boldsymbol{\Psi}(X)|_p$ can be computed, called global SVD (Klus et al. 2018a), replacing the first step in Algorithm 1. The result is an SVD-like decomposition

$$\boldsymbol{\Psi}(X)|_p \approx \hat{\mathbf{U}}|_p \hat{\Sigma} \hat{V}^T.$$

Here, $\hat{\mathbf{U}} \in \mathbb{R}^{n_1 \times \cdots \times n_p \times r_p}$ is a tensor train with TT ranks $r_1, \ldots, r_p$, where $r_p > 1$ is allowed (see Remark 2.2), and left-orthonormal cores. Moreover, $\hat{\Sigma} \in \mathbb{R}^{r_p \times r_p}$ is diagonal and nonnegative, and $\hat{V} \in \mathbb{R}^{m \times r_p}$ is orthogonal. As shown in Nüske et al. (2021a), the columns of $\hat{\mathbf{U}}|_p \hat{\Sigma}^{-1}$ encode an empirically orthonormal basis of an r_p-dimensional subspace of $\mathbb{V}$:

$$\hat{\eta}_{r_p}(x)^T = (\boldsymbol{\Psi}(x)|_p)^T \hat{\mathbf{U}}|_p \hat{\Sigma}^{-1}.$$

Analogous to the standard AMUSE algorithm, we can then compute an empirical representation of the generator on the linear span of $\hat{\eta}_{r_p}$, by forming the product [see (10 and 11)]:

$$\hat{M}_{r_p} = \sum_{l=1}^{m} \hat{V}_{l,:} \otimes (\mathcal{L}\boldsymbol{\Psi}(x_l)|_p)^T \hat{\mathbf{U}}|_p \hat{\Sigma}^{-1}, \tag{16}$$

$$\hat{M}_{r_p}^{\text{rev}} = -\frac{1}{2} \sum_{l=1}^{m} \hat{\Sigma}^{-1} (\hat{\mathbf{U}}|_p)^T \nabla\boldsymbol{\Psi}(x_l) \, a(x_l) \, \nabla\boldsymbol{\Psi}(x_l)^T \, \hat{\mathbf{U}}|_p \, \hat{\Sigma}^{-1}. \tag{17}$$

The computationally demanding part of (16, 17) is the multiplication of the unfolded tensors $(\mathcal{L}\boldsymbol{\Psi}(x_l)|_p)^T$ and $\hat{\mathbf{U}}|_p$ (or $(\nabla\boldsymbol{\Psi}(x_l)|_p)^T$ and $\hat{\mathbf{U}}|_p$). The orthonormal factor $\hat{\mathbf{U}}$ is always given in tensor train format by construction. The main topic of this study is therefore the derivation of a tensor train representation of the first factor in each case, containing the action of the generator $\mathcal{L}$ (see Sect. 3), and the efficient calculation of the reduced matrix by contraction of a suitable tensor network (see Sect. 4).

3 Tensor Representations of the Generator Data Tensor

The goal of this section is to determine a tensor train representation of the generator data tensors $\mathcal{L}\boldsymbol{\Psi}(x) \in \mathbb{R}^{n_1 \times \cdots \times n_p}$ and $\nabla\boldsymbol{\Psi}(x) \in \mathbb{R}^{n_1 \times \cdots \times n_p \times d}$, given entry-wise by

$$\mathcal{L}\boldsymbol{\Psi}(x)_{u_1,\ldots,u_p} = \mathcal{L}(\psi_{1,u_1} \cdot \ldots \cdot \psi_{p,u_p})(x), \quad \nabla\boldsymbol{\Psi}(x)_{u_1,\ldots,u_p,i} = \frac{\partial}{\partial x^i}(\psi_{1,u_1} \cdot \ldots \cdot \psi_{p,u_p})(x). \tag{18}$$

3.1 A General Representation Formula

As the Kolmogorov operator acts as a differential operator, its point-wise action on a product basis will be a sum of rank-one tensors, by the product rule. In anticipation of later results, we prove the following general representation formula, which is a straightforward extension of results shown in Kazeev et al. (2013):

Proposition 3.1 *Consider a tensor* $\mathbf{T} \in \mathbb{R}^{n_1 \times \cdots \times n_p}$ *defined by*

$$
\mathbf{T} = \sum_{k=1}^{p} \left(\mathbf{e}_1 \otimes \cdots \otimes \mathbf{f}_k \otimes \cdots \otimes \mathbf{e}_p \right)
$$
$$
+ \sum_{k_1=1}^{p-1} \sum_{k_2=(k_1+1)}^{p} \sum_{i=1}^{d} \left(\mathbf{e}_1 \otimes \cdots \otimes \mathbf{g}_{k_1,i} \otimes \cdots \otimes \mathbf{h}_{k_2,i} \otimes \cdots \otimes \mathbf{e}_p \right), \qquad (19)
$$

with vectors $\mathbf{e}_k, \mathbf{f}_k, \mathbf{g}_{k,i}, \mathbf{h}_{k,i} \in \mathbb{R}^{n_k}$ *for* $k = 1, \ldots, p$ *and* $i = 1, \ldots, d$. *Then,* $\mathbf{T}$ *can be written as a tensor train* $\mathbf{T} = \llbracket \mathbf{T}^{(1)} \rrbracket \otimes \cdots \otimes \llbracket \mathbf{T}^{(p)} \rrbracket$ *with all ranks equal to* $d + 2$, *and cores:*

$$
\llbracket \mathbf{T}^{(1)} \rrbracket = \llbracket \mathbf{e}_1 \ \mathbf{f}_1 \ \mathbf{g}_{1,1} \ \ldots \ \mathbf{g}_{1,d} \rrbracket \tag{20}
$$

$$
\llbracket \mathbf{T}^{(k)} \rrbracket = \llbracket \begin{array}{ccccc} \mathbf{e}_k & \mathbf{f}_k & \mathbf{g}_{k,1} & \ldots & \mathbf{g}_{k,d} \\ & \mathbf{e}_k & & & \\ & \mathbf{h}_{k,1} & \mathbf{e}_k & & \\ & \vdots & & \ddots & \\ & \mathbf{h}_{k,d} & & & \mathbf{e}_k \end{array} \rrbracket \qquad k = 2, \ldots, (p-1) \tag{21}
$$

$$
\llbracket \mathbf{T}^{(p)} \rrbracket = \llbracket \mathbf{f}_p \ \mathbf{e}_p \ \mathbf{h}_{p,1} \ \ldots \ \mathbf{h}_{p,d} \rrbracket^T . \tag{22}
$$

Proof Define auxiliary tensors $\mathbf{A}^{(k)}, \mathbf{B}_i^{(k)} \in \mathbb{R}^{n_k \times \cdots \times n_p}$ for $k = 1, \ldots, (p-1)$ and $i = 1, \ldots, d$ by:

$$
\mathbf{A}^{(k)} = \sum_{l=k}^{p} \left(\mathbf{e}_k \otimes \cdots \otimes \mathbf{f}_l \otimes \cdots \otimes \mathbf{e}_p \right)
$$
$$
+ \sum_{l_1=k}^{p-1} \sum_{l_2=(l_1+1)}^{p} \sum_{i=1}^{d} \left(\mathbf{e}_k \otimes \cdots \otimes \mathbf{g}_{l_1,i} \otimes \cdots \otimes \mathbf{h}_{l_2,i} \otimes \cdots \otimes \mathbf{e}_p \right),
$$

and

$$
\mathbf{B}_i^{(k)} = \sum_{l=k}^{p} \left(\mathbf{e}_k \otimes \cdots \otimes \mathbf{h}_{l,i} \otimes \cdots \otimes \mathbf{e}_p \right).
$$

We also define $\mathbf{A}^{(p)} = \mathbf{f}_p$ and $\mathbf{B}_i^{(p)} = \mathbf{h}_{p,i}$. Note that $\mathbf{A}^{(k)}$ is similar to $\mathbf{T}$, but all sums start from k instead of 1. In particular, we have that $\mathbf{A}^{(1)} = \mathbf{T}$. The following relation holds for $k = 1, \ldots, (p-1)$, and is central for the proof:

$$\mathbf{A}^{(k)} = \mathbf{e}_k \otimes \mathbf{A}^{(k+1)} + \left(\mathbf{f}_k \otimes \mathbf{e}_{k+1} \otimes \cdots \otimes \mathbf{e}_p\right) + \sum_{i=1}^{d} \mathbf{g}_{k,i} \otimes \mathbf{B}_i^{(k+1)}. \quad (23)$$

The recursion (23) can be verified directly by expanding $\mathbf{e}_k \otimes \mathbf{A}^{(k+1)}$, and noting that the remaining terms in (23) are exactly what is missing to complete the expression for $\mathbf{A}^{(k)}$. We also obtain a recursive relation for $\mathbf{B}_i^{(k)}$, for $k = 1, \ldots, (p-1)$:

$$\mathbf{B}_i^{(k)} = \mathbf{h}_{k,i} \otimes \mathbf{e}_{k+1} \otimes \cdots \otimes \mathbf{e}_p + \mathbf{e}_k \otimes \mathbf{B}_i^{(k+1)}. \quad (24)$$

Using (23 - 24), we can verify that for $k = 2, \ldots, p-1$:

$$\left[\!\left[\mathbf{T}^{(k)}\right]\!\right] \otimes \left[\!\left[\mathbf{A}^{(k+1)} \; \mathbf{e}_{k+1} \otimes \cdots \otimes \mathbf{e}_p \; \mathbf{B}_1^{(k+1)} \; \ldots \; \mathbf{B}_d^{(k+1)}\right]\!\right]^T$$

$$= \left[\!\left[\begin{matrix} \mathbf{e}_k & \mathbf{f}_k & \mathbf{g}_{k,1} & \ldots & \mathbf{g}_{k,d} \\ & \mathbf{e}_k & & & \\ & \mathbf{h}_{k,1} & \mathbf{e}_k & & \\ & \vdots & & \ddots & \\ & \mathbf{h}_{k,d} & & & \mathbf{e}_k \end{matrix}\right]\!\right] \otimes \left[\!\left[\begin{matrix} \mathbf{A}^{(k+1)} \\ \mathbf{e}_{k+1} \otimes \cdots \otimes \mathbf{e}_p \\ \mathbf{B}_1^{(k+1)} \\ \vdots \\ \mathbf{B}_d^{(k+1)} \end{matrix}\right]\!\right] \quad (25)$$

$$= \left[\!\left[\mathbf{A}^{(k)} \; \mathbf{e}_k \otimes \cdots \otimes \mathbf{e}_p \; \mathbf{B}_1^{(k)} \; \ldots \; \mathbf{B}_d^{(k)}\right]\!\right]^T. \quad (26)$$

By noting that

$$\left[\!\left[\mathbf{T}^{(p)}\right]\!\right] = \left[\!\left[\mathbf{A}^{(p)} \; \mathbf{e}_p \; \mathbf{B}_1^{(p)} \; \ldots \; \mathbf{B}_d^{(p)}\right]\!\right]^T,$$

and repeated application of (26), we find

$$\left[\!\left[\mathbf{T}^{(1)}\right]\!\right] \otimes \cdots \otimes \left[\!\left[\mathbf{T}^{(p)}\right]\!\right] = \left[\!\left[\mathbf{e}_1 \; \mathbf{f}_1 \; \mathbf{g}_{1,1} \; \ldots \; \mathbf{g}_{1,d}\right]\!\right] \otimes \left[\!\left[\begin{matrix} \mathbf{A}^{(2)} \\ \mathbf{e}_2 \otimes \cdots \otimes \mathbf{e}_p \\ \mathbf{B}_1^{(2)} \\ \vdots \\ \mathbf{B}_d^{(2)} \end{matrix}\right]\!\right] = \mathbf{T}.$$

$\square$

Remark 3.2 Proposition 3.1 differs from the setting in Kazeev et al. (2013) by adding the sums over i in the second line of (19). On the other hand, additional weights a_{k_1,k_2} in each term were considered in Kazeev et al. (2013), as well as an additional sum over

the terms in the second line, but with $\mathbf{g}_{k,i}$ and $\mathbf{h}_{k,i}$ swapped. The latter modification, without weights, is easily incorporated into (20 - 22). However, this result is not needed in the following:

Corollary 3.3 *If the definition of* $\mathbf{T}$ *additionally contains the terms:*

$$\sum_{k_1=1}^{p-1} \sum_{k_2=(k_1+1)}^{p} \sum_{i=1}^{d} \left(\mathbf{e}_1 \otimes \cdots \otimes \mathbf{h}_{k_1,i} \otimes \cdots \otimes \mathbf{g}_{k_2,i} \otimes \cdots \otimes \mathbf{e}_p \right),$$

then $\mathbf{T}$ *can be represented with cores of ranks* $2d + 2$ *similar to* (20–22). *In detail,* $\mathbf{T}^{(1)}$ *is extended by appending all terms* $\mathbf{h}_{1,i}$, *while all* $\mathbf{g}_{p,i}$ *are appended to* $\mathbf{T}^{(p)}$. *For* $2 \leq k \leq p-1$, *all terms* $\mathbf{h}_{k,i}$ *are added to the first row of* $\mathbf{T}^{(k)}$, *while all* $\mathbf{g}_{k,i}$ *are added to the second column. The diagonal is just filled up with copies of the vector* $\mathbf{e}_k$.

3.2 Representations of the Generator Data Tensor

Based on Proposition 3.1, we can now derive tensor train representations of the data tensors required for generator EDMD in TT format. We introduce the following notation: for a vector-valued function $\psi : \mathbb{X} \mapsto \mathbb{R}^n$, we write $\frac{\partial \psi}{\partial x^i}$ for the vector of partial derivatives of all functions ψ_u with respect to coordinate x^i. The associated Jacobian matrix is denoted by $\nabla \psi = \left(\frac{\partial \psi_u}{\partial x^i} \right)_{u,i} \in \mathbb{R}^{n \times d}$, and the corresponding Hessian tensor by $\nabla^2 \psi \in \mathbb{R}^{n \times d \times d}$.

3.2.1 Reversible Case

We consider the reversible generator $\mathcal{L}$ first, since only first-order derivatives are required for its data-driven approximation (11).

Lemma 3.4 *For any* $x \in \mathbb{X}$, *the tensor* $\nabla \mathbf{\Psi}(x) \in \mathbb{R}^{n_1 \times \cdots \times n_p \times d}$ *can be written in the form* (19) *with:*

$$\mathbf{e}_k = \begin{cases} \psi_k(x) & 1 \leq k \leq p \\ 0 & k = p+1 \end{cases}, \quad \mathbf{f}_k = 0, \quad \mathbf{g}_{k,i} = \frac{\partial \psi_k}{\partial x^i}(x), \quad \mathbf{h}_{k,i} = \begin{cases} 0 & 2 \leq k \leq p \\ e_i & k = p+1. \end{cases}$$

Proof The representation follows directly from the product rule:

$$\nabla \mathbf{\Psi}(x) = \sum_{i=1}^{d} \sum_{k=1}^{p} \psi_1(x) \otimes \cdots \otimes \frac{\partial \psi_k}{\partial x^i}(x) \otimes \cdots \otimes \psi_p(x) \otimes e_i,$$

which is already (19) with the parameters given by the Lemma. $\square$

Corollary 3.5 *The tensor train representation of* $\nabla \mathbf{\Psi}(x)$, *with ranks equal to* $r_1 = (d+1), \ldots, r_{p-1} = (d+1), r_p = d$, *is given by the cores:*

$$\left[\!\left[\nabla \mathbf{\Psi}^{(1)}(x) \right]\!\right] = \left[\!\left[\psi_1(x) \; \frac{\partial \psi_1}{\partial x^1}(x) \; \dots \; \frac{\partial \psi_1}{\partial x^d}(x) \right]\!\right]$$

$$\left[\!\left[\nabla \mathbf{\Psi}^{(k)}(x) \right]\!\right] = \left[\!\left[\begin{array}{cccc} \psi_k(x) & \frac{\partial \psi_k}{\partial x^1}(x) & \dots & \frac{\partial \psi_k}{\partial x^d}(x) \\ & \psi_k(x) & & \\ & & \ddots & \\ & & & \psi_k(x) \end{array} \right]\!\right], \quad 2 \le k \le p-1,$$

$$\left[\!\left[\nabla \mathbf{\Psi}^{(p)}(x) \right]\!\right] = \left[\!\left[\begin{array}{ccc} \frac{\partial \psi_p}{\partial x^1}(x) & \dots & \frac{\partial \psi_p}{\partial x^d}(x) \\ \psi_p(x) & & \\ & \ddots & \\ & & \psi_p(x) \end{array} \right]\!\right],$$

$$\left[\!\left[\nabla \mathbf{\Psi}^{(p+1)}(x) \right]\!\right] = \left[\!\left[e_1 \; \dots \; e_d \right]\!\right]^T.$$

Proof The result follows from Lemma 3.4, and by observing that the second row and column of each core in Proposition 3.1 can be omitted in this case, as they do not contribute to the final tensor. Moreover, the first column of the pre-last core $\nabla \mathbf{\Psi}^{(p)}(x)$ can also be omitted, as it would be multiplied by a zero in $\nabla \mathbf{\Psi}^{(p+1)}(x)$. $\qquad \square$

Remark 3.6 A TT representation of $\mathrm{d}\mathbf{\Psi}(x)$ can be obtained based on Corollary 3.5 by absorbing the additional factor σ into the final core. As $\mathrm{d}\mathbf{\Psi}(X) = \sum_{l=1}^{m} \mathrm{d}\mathbf{\Psi}(x_l) \otimes e_l$, where e_l is the canonical unit vector in $\mathbb{R}^m$, we can also obtain a TT decomposition of $\mathrm{d}\mathbf{\Psi}(X)$ using the summation rule for tensor trains (Oseledets 2011).

3.2.2 Non-reversible Case

Next, we consider the generator $\mathcal{L}$ for a general (not necessarily reversible) SDE. The point-wise evaluation of $\mathcal{L}$ can be written as a sum of rank-one tensors as follows:

Lemma 3.7 *For any $x \in \mathbb{X}$, the tensor $\mathcal{L}\mathbf{\Psi}(x) \in \mathbb{R}^{n_1 \times \dots \times n_p}$ can be written in the form (19) with:*

$$\mathbf{e}_k = \psi_k(x), \qquad \mathbf{f}_k = \mathcal{L}\psi_k(x), \qquad \mathbf{g}_{k,i} = \mathbf{h}_{k,i} = \nabla\psi_k(x) \cdot \sigma_{:,i}(x),$$

where $\sigma_{:,i}$ denotes the i-th column of the diffusion σ.

Proof The first and second derivatives of the tensor $\mathbf{\Psi}(x)$ are given by (omitting the argument x):

$$\frac{\partial \mathbf{\Psi}}{\partial x^i} = \sum_{k=1}^{p} \psi_1 \otimes \dots \otimes \frac{\partial \psi_k}{\partial x^i} \otimes \dots \otimes \psi_p,$$

$$\frac{\partial^2 \mathbf{\Psi}}{\partial x^i \partial x^j} = \sum_{k=1}^{p} \psi_1 \otimes \dots \otimes \frac{\partial^2 \psi_k}{\partial x^i \partial x^j} \otimes \dots \otimes \psi_p + \sum_{k_1 \neq k_2} \psi_1 \otimes \dots \otimes \frac{\partial \psi_{k_1}}{\partial x^i} \otimes \dots \otimes \frac{\partial \psi_{k_2}}{\partial x^j} \otimes \dots \otimes \psi_p.$$

Therefore, application of the generator yields:

$$\mathcal{L}\mathbf{\Psi} = \sum_{i=1}^{d} b_i \frac{\partial \mathbf{\Psi}}{\partial x^i} + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij} \frac{\partial^2 \mathbf{\Psi}}{\partial x^i \partial x^j}$$

$$= \sum_{k=1}^{p} \psi_1 \otimes \cdots \otimes \left(\nabla \psi_k \cdot b + \frac{1}{2} \nabla^2 \psi_k : a \right) \otimes \cdots \otimes \psi_p$$

$$+ \frac{1}{2} \sum_{i,j=1}^{d} a_{ij} \sum_{k_1 \neq k_2} \psi_1 \otimes \cdots \otimes \frac{\partial \psi_{k_1}}{\partial x^i} \otimes \cdots \otimes \frac{\partial \psi_{k_2}}{\partial x^j} \otimes \cdots \otimes \psi_p.$$

The k-th term in the first line is just $\mathcal{L}\psi_k$, while in the second line, we can write $a_{ij} = \sum_{s=1}^{d} \sigma_{is} \sigma_{js}$, to obtain:

$$\mathcal{L}\mathbf{\Psi} = \sum_{k=1}^{p} \psi_1 \otimes \cdots \otimes \mathcal{L}\psi_k \otimes \cdots \otimes \psi_p$$

$$+ \frac{1}{2} \sum_{s=1}^{d} \sum_{k_1 \neq k_2} \psi_1 \otimes \cdots \otimes (\nabla \psi_{k_1} \cdot \sigma_{:,s}) \otimes \cdots \otimes (\nabla \psi_{k_2} \cdot \sigma_{:,s}) \otimes \cdots \otimes \psi_p$$

$$= \sum_{k=1}^{p} \psi_1 \otimes \cdots \otimes \mathcal{L}\psi_k \otimes \cdots \otimes \psi_p$$

$$+ \sum_{s=1}^{d} \sum_{k_1=1}^{p-1} \sum_{k_2=k_1+1}^{p} \psi_1 \otimes \cdots \otimes (\nabla \psi_{k_1} \cdot \sigma_{:,s}) \otimes \cdots \otimes (\nabla \psi_{k_2} \cdot \sigma_{:,s}) \otimes \cdots \otimes \psi_p.$$

In the last step, we used that the second term is symmetric in k_1 and k_2. This is the desired result. □

The tensor train representation of $\mathcal{L}\mathbf{\Psi}(x)$ is then obtained from Proposition 3.1:

Corollary 3.8 *The tensor train representation of $\mathcal{L}\mathbf{\Psi}(x)$, with ranks all equal to $(d+2)$, is given by the cores:*

$$\left[\!\left[\mathcal{L}\mathbf{\Psi}^{(1)}(x) \right]\!\right] = \left[\!\left[\psi_1(x) \; \mathcal{L}\psi_1(x) \; \nabla\psi_1(x) \cdot \sigma_{:,1}(x) \; \ldots \; \nabla\psi_1(x) \cdot \sigma_{:,d}(x) \right]\!\right]$$

$$\left[\!\left[\mathcal{L}\mathbf{\Psi}^{(k)}(x) \right]\!\right] = \left[\!\left[\begin{matrix} \psi_k(x) & \mathcal{L}\psi_k(x) & \nabla\psi_k(x) \cdot \sigma_{:,1}(x) & \ldots & \nabla\psi_k(x) \cdot \sigma_{:,d}(x) \\ & \psi_k(x) & & & \\ & \nabla\psi_k(x) \cdot \sigma_{:,1}(x) & \psi_k(x) & & \\ & \vdots & & \ddots & \\ & \nabla\psi_k(x) \cdot \sigma_{:,d}(x) & & & \psi_k(x) \end{matrix} \right]\!\right]$$

$$\left[\!\left[\mathcal{L}\mathbf{\Psi}^{(p)}(x) \right]\!\right] = \left[\!\left[\mathcal{L}\psi_p(x) \; \psi_p(x) \; \nabla\psi_p(x) \cdot \sigma_{:,1}(x) \; \ldots \; \nabla\psi_p(x) \cdot \sigma_{:,d}(x) \right]\!\right]^T.$$

Remark 3.9 As $\mathcal{L}\Psi(X) = \sum_{l=1}^{m} \mathcal{L}\Psi(x_l) \otimes e_l$, where e_l is the canonical unit vector in $\mathbb{R}^m$, we can also obtain a TT decomposition of $\mathcal{L}\Psi(X)$ using the summation rule for tensor trains (Oseledets 2011).

4 Algorithmic Realization

In this section, we discuss the algorithmic realization of the tensor-based version of gEDMD. We first present the tgEDMD algorithm in Sect. 4.1 and briefly analyze its complexity and consistency for infinite data. Subsequently, we present two extensions of the method in Sect. 4.2.

4.1 The tgEDMD Algorithm

Based on the results of Sect. 3, the tensor-based counterpart of the AMUSE Algorithm 1, which we call tgEDMD, is given in Algorithm 2. Its output is a matrix representation $\hat{M}_{r_p}$ or $\hat{M}_{r_p}^{\text{rev}}$ of the generator, on the linear span of the functions $\hat{\eta}_{r_p}$. This matrix representation can then be used for further analysis of the generator, e.g., for spectral analysis, prediction, or model reduction.

Algorithm 2 tgEDMD

Input: Data $x_1, \ldots, x_m \in \mathbb{X}$, basis sets $\{\psi_{k,u_k}\}_{u_k=1}^{n_k}$, $1 \le k \le p$.
Output: Reduced matrix representation $\hat{M}_{r_p} \in \mathbb{R}^{r_p \times r_p}$ or $\hat{M}_{r_p}^{\text{rev}} \in \mathbb{R}^{r_p \times r_p}$ of $\mathcal{L}$.

1: Compute global SVD of $\Psi(X)$, i.e., $\Psi(X)|_p \approx \hat{U}|_p \hat{\Sigma} \hat{V}^T$, where $\hat{U}$ is in tensor train format with ranks $r_1, \ldots, r_p$ and left-orthonormal cores.
2: Set up TT representations of $\mathcal{L}\Psi(x_l)$ or $\nabla\Psi(x_l)$ for $1 \le l \le m$, according to Corollaries 3.8 or 3.5.
3: Compute reduced matrices $\hat{M}_{r_p}$ or $\hat{M}_{r_p}^{\text{rev}}$, according to (16 - 17).

Computational Effort The computational effort required by Algorithm 2 can be broken down into three major contributions.

The first is the global SVD of the data tensor $\Psi(X)$. The method consists of a series of matrix SVDs, each matrix being of size $r_{k-1}n_k \times m$. As is well known, the cost for these can be estimated as $\mathcal{O}(\min\{r_{k-1}^2 n_k^2 m, r_{k-1}n_k m^2\})$. The cost of updating the next core in each step can be neglected in comparison.

Secondly, the reduced matrices $\hat{M}_{r_p}$ or $\hat{M}_{r_p}^{\text{rev}}$ need to be computed. To this end, for each data point x_l, the contraction of the orthonormal tensor train $\hat{U}$ with either $\mathcal{L}\Psi(x_l)$ or $\nabla\Psi(x_l)$ in (16 and 17) must be carried out. This can be achieved by Algorithm 3, see Oseledets (2011) for details. The tensor product in lines 1 and 3 is the Kronecker product between matrices defined in (14). Of course, Algorithm 3 can be applied to any pair of general tensors of compatible dimensions.

As the cores of $\mathcal{L}\Psi(x_l)$ and $\nabla\Psi(x_l)$ are structured and sparse, we obtain the following result:

Lemma 4.1 *The cost of applying Algorithm 3 can be estimated by* $\mathcal{O}(\sum_{k=1}^{p} n_k d r_{k-1} r_k)$.

Algorithm 3 Tensor Network Contraction

Input: Tensor trains $\mathbf{T}$, $\hat{\mathbf{U}}$ of order p with ranks $s_1, \ldots, s_p$ and $r_1, \ldots, r_p$, respectively,
  where either $\mathbf{T} = \mathcal{L}\boldsymbol{\Psi}(x_l)$ or $\mathbf{T} = \nabla\boldsymbol{\Psi}(x_l)$.
Output: Contraction $\mathbf{T} \cdot \hat{\mathbf{U}} \in \mathbb{R}^{s_p \times r_p}$, defined by $(\mathbf{T} \cdot \hat{\mathbf{U}})_{u,v} = \sum_{u_1,\ldots,u_p} \mathbf{T}_{u_1,\ldots,u_p;u} \, \hat{\mathbf{U}}_{u_1,\ldots,u_p;v}$.

1: $v = \sum_{u_1} \mathbf{T}^{(1)}_{:,u_1,:} \otimes \hat{\mathbf{U}}^{(1)}_{:,u_1,:}$.
2: **for** $k = 2, \ldots, p$ **do**
3:   $v \leftarrow v \left[\sum_{u_k} \mathbf{T}^{(k)}_{:,u_k,:} \otimes \hat{\mathbf{U}}^{(k)}_{:,u_k,:} \right]$.
4: **end for**
5: Re-shape v to shape $s_p \times r_p$.
6: **return** v

Proof We obtain the result by recalling that the TT ranks of $\mathcal{L}\boldsymbol{\Psi}(x_l)$ and $\nabla\boldsymbol{\Psi}(x_l)$ are essentially equal to d by the results obtained in Sect. 3, and by observing that the sparsity patterns of their cores are invariant under Kronecker products. Taking $\mathcal{L}\boldsymbol{\Psi}(x_l)$ as an example, we have by (14) for all $2 \leq k \leq p - 1$:

$$
\mathcal{L}\boldsymbol{\Psi}(x_l)^{(k)}_{:,u_k,:} \otimes \hat{\mathbf{U}}^{(k)}_{:,u_k,:} =
\begin{bmatrix}
B_{1,1} & B_{1,2} & B_{1,3} & \cdots & B_{1,d+2} \\
 & B_{2,2} & & & \\
 & B_{3,2} & B_{3,3} & & \\
 & \vdots & & \ddots & \\
 & B_{d+2,2} & & & B_{d+2,d+2}
\end{bmatrix},
$$

where each block $B_{u,v} \in \mathbb{R}^{r_{k-1} \times r_k}$ is given by $\mathcal{L}\boldsymbol{\Psi}(x_l)^{(k)}_{u,u_k,v} \hat{\mathbf{U}}^{(k)}_{:,u_k,:}$. The Kronecker product $\mathcal{L}\boldsymbol{\Psi}(x_l)^{(k)}_{:,u_k,:} \otimes \hat{\mathbf{U}}^{(k)}_{:,u_k,:}$ can therefore be evaluated in $\mathcal{O}(dr_{k-1}r_k)$ operations, while multiplication by a row vector from the left also costs $\mathcal{O}(dr_{k-1}r_k)$ operations. The full cost of applying Algorithm 3 is therefore $\mathcal{O}(\sum_{k=1}^{p} n_k dr_{k-1}r_k)$. $\square$

The remaining matrix multiplications appearing in the computation of the reduced matrices are again not relevant in comparison, leaving us with the estimate $\mathcal{O}(m \sum_{k=1}^{p} n_k dr_{k-1}r_k)$. Finally, if eigenvalues of the Kolmogorov operator are required, an additional cost of $\mathcal{O}(r_p^3)$ for the diagonalization of the reduced matrix must be allowed. In summary, the total cost for the approximation of eigenvalues of the Kolmogorov operator can be estimated as

$$
\mathcal{O}\left(\sum_{k=1}^{p} \min\{r_{k-1}^2 n_k^2 m, r_{k-1}n_k m^2\} \right) + \mathcal{O}\left(m \sum_{k=1}^{p} n_k dr_{k-1}r_k \right) + \mathcal{O}(r_p^3). \quad (27)
$$

It is instructive to compare these findings to the cost of applying the standard AMUSE Algorithm 1 in conjunction with the full product basis $\boldsymbol{\Psi}$. Denoting the product of all mode sizes by $N = \prod_{k=1}^{p} n_k$, and adding up the operation counts for the SVD of the data tensor $\boldsymbol{\Psi}(X)$, the assembly of the reduced matrix, and diagonalization of the latter, we end up with the estimate

$$
\mathcal{O}(Nm^2) + \mathcal{O}(m(Nr + r^2)) + \mathcal{O}(r^3). \quad (28)
$$

In all but the smallest examples, the cost is dominated by the exponential dependence on the dimension p through N.

Consistency The discussion on consistency of the AMUSE algorithm, see Sect. 2.3, carries over to the tensor case, based on the analysis in Ref. Nüske et al. (2021a). Therein, it was shown that if all SVDs in the global SVD are truncated according to a fixed vector of TT ranks $\mathbf{r} = [r_1, \ldots, r_p]$, one can construct a recursive sequence of subspaces $\mathbb{G}^{:k}$, $1 \leq k \leq p$ (called multi-linear spectral subspaces), such that the linear span of the functions $\hat{\eta}_{r_p}$ consistently approximates the last of these subspaces $\mathbb{G}^{:p}$, along with the projected Koopman generator. This result required non-degeneracy of a series of eigenvalues of recursively constructed Gramian matrices. Analogous to the matrix case, we can also impose a vector of truncation parameters $[\epsilon_1, \ldots, \epsilon_p]$ instead and choose the reduced rank of each SVD in the global SVD method such that $\hat{\sigma}_{r_k}^k \geq \sqrt{m}\epsilon_k > \hat{\sigma}_{r_k+1}^k$, $1 \leq k \leq p$. Given similar non-degeneracy conditions, the same arguments as in Nüske et al. (2021a) can be used to show that there are limiting ranks $r_k(\epsilon_k)$ such that $r_k \to r_k(\epsilon_k)$, $1 \leq k \leq p$, and the linear span of $\hat{\eta}_{r_p}$ consistently approximates $\mathbb{G}^{:p}$ for the TT ranks $[r_1(\epsilon_1), \ldots, r_p(\epsilon_p)]$.

4.2 Extensions

We conclude the methodological part of this paper by briefly discussing two practically relevant settings which require modifications of the proposed methods.

Importance Sampling First, obtaining data sampling the invariant measure μ is often a major challenge, for instance due to metastability of the dynamics (2). In principle, this difficulty can be circumvented as long as appropriate re-weighting factors can be calculated. Assume the data are generated according to a probability distribution with density ϑ on $\mathbb{X}$ and we are able to calculate the *importance sampling ratio* $w(x) = Z\frac{\rho(x)}{\vartheta(x)}$ at any point $x \in \mathbb{X}$, where Z is a possibly unknown constant. This setting reflects a rather typical scenario where both the invariant density ρ and the sampling density ϑ can be evaluated up to an unknown normalization constant. Note, however, that a poor choice of the sampling density ϑ can drastically increase the variance of all estimators involved, but this problem is beyond the scope of the present work.

To see how this affects the proposed methods, consider the matrix case first. Introducing the diagonal weighting matrix $W = \mathrm{diag}[w(x_1), \ldots, w(x_m)]$, the empirical estimators (7, 8) change according to:

$$\hat{C}(\psi) = \frac{1}{m}(\Psi(X)W^{1/2})(\Psi(X)W^{1/2})^T, \quad \hat{A}(\psi) = \frac{1}{m}(\Psi(X)W^{1/2})(\mathcal{L}\Psi(X)W^{1/2})^T.$$

We need to modify line 1 of Algorithm 1 to the computation of the rank-r SVD $(\Psi(X)W^{1/2}) \approx \hat{U}_r \hat{\Sigma}_r \hat{V}_r^T$. The reduced matrices are then obtained as

$$\hat{M}_r = \hat{\Sigma}_r^{-1} \hat{U}_r^\top (\Psi(X)W^{1/2})(\mathcal{L}\Psi(X)W^{1/2})^T \hat{U}_r \hat{\Sigma}_r^{-1} = \hat{V}_r^\top W^{1/2} \mathcal{L}\Psi(X)^T \hat{U}_r \hat{\Sigma}_r^{-1},$$

$$\hat{M}_r^{\mathrm{rev}} = -\frac{1}{2}\hat{\Sigma}_r^{-1} \hat{U}_r^\top \, \mathrm{d}\Psi(X) \, W \, \mathrm{d}\Psi(X)^T \hat{U}_r \hat{\Sigma}_r^{-1}.$$

Note that both estimators are indeed invariant with respect to the constant Z. In the tensor case, the following modifications must be made as a result:

1. The TT representation (15) of $\Psi(X)$ is modified by scaling each unit vector in the last core by the corresponding re-weighting factor:

$$\left[\!\!\left[\Psi(X)^{(p+1)}\right]\!\!\right] = \left[\!\!\left[\begin{array}{c} \sqrt{w(x_1)}e_1 \\ \vdots \\ \sqrt{w(x_m)}e_m \end{array}\right]\!\!\right].$$

2. Application of the global SVD of $\Psi(X)$ must be preceded by right-orthonormalization of $\Psi(X)^{(p+1)}$, see Klus et al. (2018a).
3. Reduced matrices are calculated according to

$$\hat{M}_{r_p} = \sum_{l=1}^{m} w(x_l)^{1/2} \hat{V}_{l,:} \otimes (\mathcal{L}\Psi(x_l)|_p)^T \hat{U}|_p \hat{\Sigma}^{-1},$$

$$\hat{M}_{r_p}^{\mathrm{rev}} = -\frac{1}{2} \sum_{l=1}^{m} w(x_l) \hat{\Sigma}^{-1} (\hat{U}|_p)^T \nabla\Psi(x_l) \, a(x_l) \, \nabla\Psi(x_l)^T \hat{U}|_p \, \hat{\Sigma}^{-1}.$$

Projection onto Reduced Variables Secondly, we discuss the use of basis functions defined on a set of reduced variables. For high-dimensional systems, it is often convenient to employ basis functions which are not directly defined on the state variables x^i, $1 \leq i \leq d$, but on the image of some (possibly) nonlinear descriptors

$$\xi : \mathbb{X} \mapsto \mathbb{Y} \subset \mathbb{R}^D, \; D \leq d.$$

Now, if $\mathbb{V} = \mathrm{span}\{\psi_u(y)\}_{u=1}^n$ is a finite-dimensional space of functions defined on the lower-dimensional state space $\mathbb{Y}$, the resulting Galerkin approximation to the generator possesses a useful interpretation, as discussed in great detail in Ref. Zhang et al. (2016). Consider the space of functions in L_μ^2 which are in fact functions of ξ, i.e.,

$$\mathbb{V}^\xi = \{\bar\phi \in L_\mu^2(\mathbb{X}) : \quad \exists \phi : \mathbb{Y} \mapsto \mathbb{R}, \; \bar\phi = \phi \circ \xi\}.$$

After identification of $\bar\phi$ with ϕ, $\mathbb{V}^\xi$ equals the weighted Lebesgue space $L_\nu^2(\mathbb{Y})$, where ν is the marginal probability distribution of μ along ξ. Using the μ-orthogonal projector $\mathcal{P}^\xi$ onto $L_\nu^2(\mathbb{Y})$, one can introduce the projected generator $\mathcal{L}^\xi = \mathcal{P}^\xi \mathcal{L} \mathcal{P}^\xi$. Given some technical assumptions (Zhang et al. 2016), the projected generator $\mathcal{L}^\xi$ is again the generator of a stochastic differential equation (2), with effective drift and diffusion coefficients. Importantly, the following identities relating the generators $\mathcal{L}$, $\mathcal{L}^\xi$ and their invariant measures μ, ν, hold for all $\phi, \tilde\phi \in L_\nu^2(\mathbb{Y})$:

$$\left\langle \phi, \tilde\phi \right\rangle_\nu = \left\langle \phi \circ \xi, \tilde\phi \circ \xi \right\rangle_\mu, \qquad \left\langle \phi, \mathcal{L}^\xi \tilde\phi \right\rangle_\nu = \left\langle \phi \circ \xi, \mathcal{L}(\tilde\phi \circ \xi) \right\rangle_\mu.$$

As a consequence, any Galerkin approximation to $\mathcal{L}$ on a subspace $\mathbb{V} \subset L_\nu^2(\mathbb{Y})$ is simultaneously an approximation to $\mathcal{L}^\xi$. Conversely, the empirical estimators (7–9) can be used in conjunction with simulation data of the full process, sampling μ, in order to approximate the reduced generator $\mathcal{L}^\xi$ (Klus et al. 2020c). When applying the full generator $\mathcal{L}$ to a basis set on $\mathbb{Y}$ as above, derivatives of ξ with respect to the full state variables x^i must be taken using the chain rule.

If $\mathbb{V} \subset \mathcal{D}(\mathcal{L}^\xi)$ is a tensor subspace, the following adjustments need to be made when applying tgEDMD:

1. In the reversible case, it suffices to replace $a(x_l)$ in (17) by $a^\xi(x_l) = \nabla\xi(x_l)a(x_l)\nabla\xi(x_l)^T$. All derivatives appearing in Corollary 3.5 can then be taken simply with respect to the reduced variables y.
2. In the non-reversible setting, the terms $\nabla\psi_k(x) \cdot \sigma_{:,i}(x)$ in Corollary 3.8 must be replaced by $\nabla_y\psi_k(y) \cdot \nabla\xi(x) \cdot \sigma_{:,i}(x)$, where $y = \xi(x)$. Moreover, each of the terms $\mathcal{L}\psi_k(x)$ must be evaluated as

$$\mathcal{L}\psi_k(x) = \nabla_y\psi_k(y) \cdot \nabla\xi(x) \cdot b(x) + \frac{1}{2}\nabla_y^2\psi_k(y) : a^\xi(x) + \frac{1}{2}\nabla_y\psi_k(y)(\nabla^2\xi(x) : a(x)).$$

5 Numerical Results

Below, we illustrate the capabilities of the proposed methods by means of two examples. The first is a diffusion in a four-dimensional model potential (Sect. 5.1), the second is a data set of molecular dynamics simulation of the deca-alanine peptide (Sect. 5.2). For both examples, we mainly focus on re-producing the dominant spectrum of the generator. In the setting outlined in Sect. 2.1, the generator $\mathcal{L}$ is nonnegative; hence, any eigenvalue κ_k must satisfy $\mathcal{R}(\kappa_k) \leq 0$, where $\mathcal{R}(\cdot)$ is the real part. If the system is reversible in addition, all eigenvalues lie on the negative real half-line. For a broad class of systems, the largest eigenvalue $\kappa_0 = 0$ is non-degenerate, possibly followed by a number of real eigenvalues close to zero and a subsequent spectral gap, that is

$$0 = \kappa_0 > \kappa_1 \geq \ldots \geq \kappa_c \gg R,$$

where $R < 0$ is an upper bound for the remaining spectrum. This pattern, which we will encounter in both examples, is the signature of metastability, i.e., the existence of long-lived states such that interstate transitions are rare-events (Davies 1982).

By the spectral mapping theorem (Pazy 2012), any generator eigenvalue κ_k corresponds to an eigenvalue $\lambda_k(t) = e^{\kappa_k t}$ of the Koopman operator. If $\mathcal{R}(\kappa_k) < 0$, the Koopman eigenvalue is thus decaying exponentially with the time lag t. The characteristic decay time scale associated to $\lambda_k(t)$, also called *implied time scale*, is

$$t_k := -\frac{1}{\mathcal{R}(\kappa_k)} = -\frac{t}{\log|\lambda_k(t)|},$$

where $|\cdot|$ is the modulus of a complex number. Implied time scales have frequently been used as a metric in order to compare different approximate models for the Koopman operator or generator (Prinz et al. 2011; Bowman et al. 2014).

Our software implementation of the tgEDMD algorithm closely follows the discussion in Sect. 4.1. It is publicly available as part of the scikit-tt package,[1] including the numerical examples presented below. The data required to re-produce these examples is also available online.[2]

5.1 Lemon Slice Potential

Description The first system we study is defined by the reversible SDE (2) on $\mathbb{X} = \mathbb{R}^4$, with drift given by the negative gradient $b = -\nabla V$ of a scalar potential V and the diffusion equal to a constant multiple of the identity, $\sigma \equiv \sqrt{2}\text{Id}$. The potential V is a sum of the two-dimensional Lemon Slice potential V_{LS} along the first two state variables and harmonic potentials V_h along the last two variables, that is

$$V = V_{\text{LS}}(x^1, x^2) + V_h(x^3) + V_h(x^4), \tag{29}$$

$$V_{\text{LS}}(x^1, x^2) = \cos(4\varphi) + \frac{1}{\cos(0.5\varphi)} + 10\,(r-1)^2 + \frac{1}{r}, \quad V^h(x^i) = 5(x^i)^2. \tag{30}$$

Above, r, φ are two-dimensional polar coordinates derived from x^1, x^2. A contour plot of the Lemon slice potential, which has also been used in a number of previous publications (Bittracher et al. 2018; Nüske et al. 2021b), is shown in Fig. 1A. The dynamics along each of the variables x^3, x^4 is independent from that along the remaining ones and equilibrates quickly. The dynamics along x^1, x^2 on the other hand, is metastable with four long-lived states corresponding to the minima of the potential V_{LS}. Consequently, a Markov state model analysis using a box discretization in (x^1, x^2)-space yields an estimate of these eigenvalues as $\kappa_0 = 0$, $\kappa_1 = -0.42$, $\kappa_2 = -1.19$, $\kappa_3 = -1.54$, which translate into implied timescales $t_1 = 2.38$, $t_2 = 0.84$, $t_3 = 0.65$. This simple example mainly serves to illustrate the importance of choosing TT ranks appropriately, and also to illustrate the use of the re-weighting technique described in Sect. 4.2.

Methods For tgEDMD, we generate ten independent long simulations of the SDE (2) by the Euler-Maruyama scheme at integration time step $\Delta_t = 10^{-3}$, each spanning $3 \cdot 10^5$ time steps. These simulations are downsampled to $m = 3000$ data points each. Moreover, we generate ten independent data sets of the same size, drawn from a three component Gaussian mixture distribution (GMM) shown in Fig. 1B, in order to test the re-weighting method. The parameters of the GMM were chosen in such a way that samples will correspond to physically relevant states with high probability, while drastically shifting the relative weights of the four metastable states.

We define elementary subspaces $\mathbb{V}^k$, $1 \leq k \leq 4$, each spanned by Gaussian basis functions depending on x^k alone. For the first two variables, we use seven Gaussians

---

[1] https://github.com/PGelss/scikit_tt/.
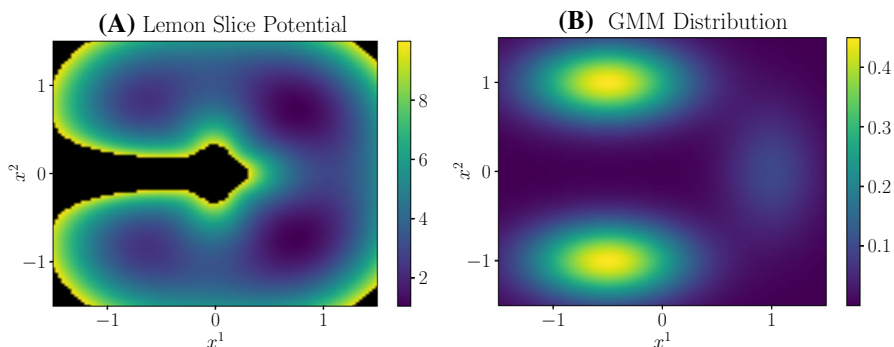
[2] https://doi.org/10.5281/zenodo.6367143.

Fig. 1 **A** Contour plot of the two-dimensional Lemon Slice potential Eq. (29). **B** Contour of the Gaussian mixture distribution of the data used in combination with the re-weighting scheme

centered equidistantly in $[-1.2, 1.2]$, each with bandwidth $\sqrt{0.4}$, while for the remaining variables we only use five Gaussians centered in $[-1.0, 1.0]$ with bandwidth $\sqrt{0.5}$. For each data set, we calculate the reduced matrix $\hat{M}_{r_p}^{\text{rev}}$ by Algorithm 2. The truncation threshold for the global SVD is set to $\sqrt{m}\epsilon$, where ϵ varies between $\epsilon = 10^{-8}$ and $\epsilon = 10^{-2}$. We calculate the first four dominant eigenvalues and implied time scales of the reduced matrix, along with their eigenvectors and the evaluation of their associated eigenfunctions at all data sites. Finally, these eigenfunction trajectories are passed to the PCCA method (Deuflhard and Weber 2005) to generate a clustering of the data into four metastable states. As a comparison, we also apply the standard AMUSE Algorithm 1 to the full product basis, using the same truncation parameters. We record the resulting ranks and implied time scales.

Results In Fig. 2, we show estimates for the leading three implied time scales t_i, $i = 1, 2, 3$, obtained from the eigenvalues of the reduced matrix, averaged over ten independent data sets, using both simulation data of the SDE (2) (panel A) and data sampled from the GMM (panel B). Time scales obtained by the standard AMUSE algorithm are shown in panel C as a comparison. We observe that using both the simulation data and the GMM data, we can robustly identify all three time scales within statistical uncertainty by the tgEDMD method, for a range of truncation parameters between $\epsilon = 10^{-6}$ and $\epsilon = 10^{-3}$. If a larger threshold is used, approximation quality starts deteriorating, especially for the slowest time scale. We also verify in panel E that a PCCA analysis of the eigenfunction trajectories computed by tgEDMD yields the correct decomposition of the data set into four metastable states. As shown in panel D of Fig. 2, the maximal rank employed by tgEDMD varies between roughly 200 for $\epsilon = 10^{-3}$ and about 800 for $\epsilon = 10^{-6}$. These ranks are approximately equal to the truncation rank achieved by standard AMUSE if the same thresholds are used. In summary, these results confirm that tgEDMD can robustly identify suitable reduced subspaces for spectral analysis of the Koopman generator. Due to the limited size of this example, there is no significant gain in terms of computational effort however. In fact, we see in panel F that estimators Eqs. (27, 28) yield about the same computational cost for tgEDMD and standard gEDMD for all truncation ranks. One can
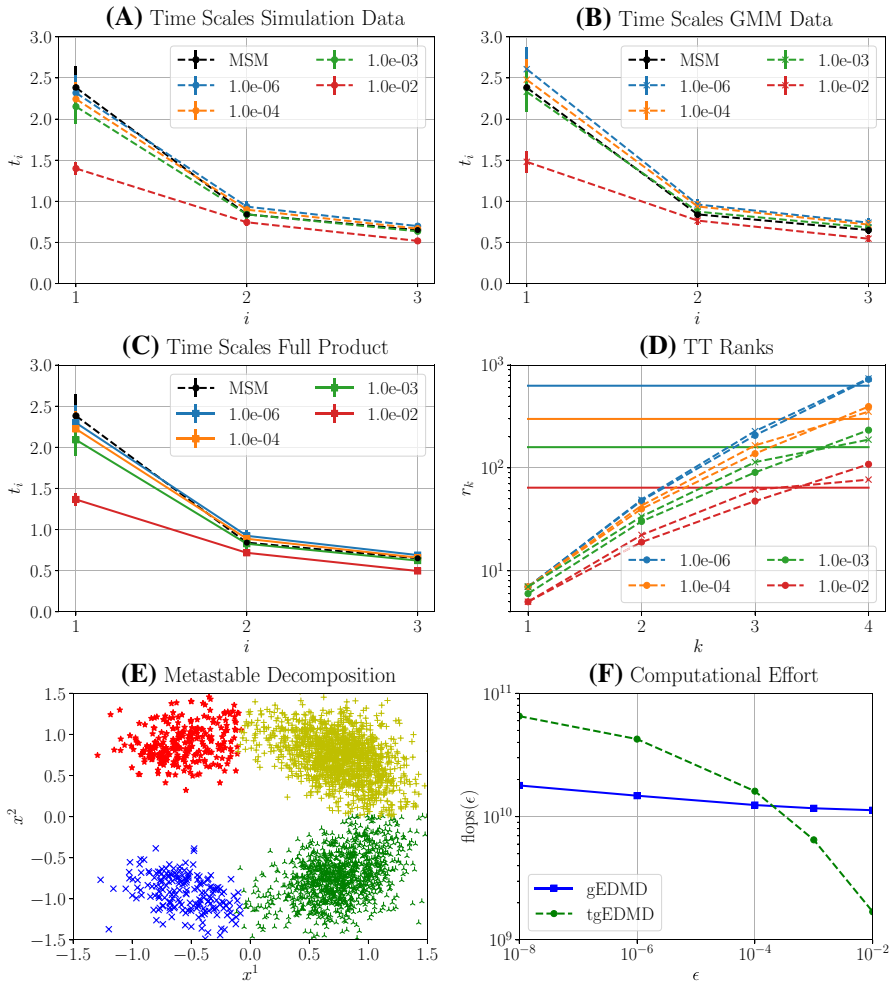
Fig. 2 Results for four-dimensional diffusion subject to the Lemon Slice potential Eq. (29) **A** Estimates of the leading implied time scales t_i, $i = 1, 2, 3$, obtained by applying the tgEDMD Algorithm 2 with Gaussian basis functions, using different values of the truncation threshold ϵ. MSM-based reference values are shown in black, errorbars were computed over ten independent simulations. **B** The same if the data are generated using a three component Gaussian mixture model, combined with the re-weighting method outlined in Sect. 4.2. **C** The same if time scales are calculated using the standard AMUSE Algorithm 1 and the SDE data. **D** TT ranks r_k, $k = 1, \ldots, 4$, for different values of the truncation threshold ϵ. Circles indicate results from the SDE trajectories, crosses those obtained using the GMM data. Solid horizontal lines indicate the rank obtained by applying standard AMUSE to the SDE data with truncation threshold indicated by the color. **E** Metastable decomposition projected onto (x^1, x^2), obtained by applying PCCA to the eigenfunction trajectories for one of the ten SDE trajectories and $\epsilon = 10^{-3}$. **F** Computational cost for tgEDMD and standard gEDMD as estimated by Eqs. (27, 28)

observe, however, that the cost for tgEDMD depends more strongly on the truncation rank, whereas the cost of gEDMD is dominated by the basis set size $N = \prod_{k=1}^{p} n_k$, as expected.

5.2 Deca-Alanine

Description The second example is a data set resulting from equilibrium molecular dynamics simulations of the ten residue peptide deca-alanine. The simulation setup can be found in Nüske et al. (2016), we only note that the data include both positions and momenta of all atoms, and the velocity re-scaling thermostat was used to control temperature. The (downsampled) data set consists of $m = 3 \cdot 10^5$ frames at a time spacing of 10 ps. The system has been analyzed in several previous publications, e.g., Nüske et al. (2014, 2016, 2021a) and Vitalini et al. (2015). It is well known that the system possesses two major metastable conformations, namely the folded (helical) and unfolded state. These can be distinguished by a collective transition in the space of backbone dihedral angles φ, ψ. Representing each (φ, ψ)-pair by its Ramachandran plane, the folded state is (approximately) characterized by $\varphi \leq 0° \vee \varphi \geq 130°$ and $\psi \in [-115°, 80°]$. We will refer to this region in the Ramachandran plane as the α-region. In the unfolded state, we expect to see a mixture of angle pairs in the α-region and in the β-region, which is defined by $\varphi \leq 0° \vee \varphi \geq 130°$ and $\psi \leq -115° \vee \psi \geq 80°$ (see the upper left plot of Fig. 3C). The slowest implied time scale, corresponding to the folding process, was determined as $t_1 \approx 7.5$ ns by an MSM analysis in Nüske et al. (2021a). Two additional time scales were estimated as $t_2 \approx 3.7$ ns, $t_3 \approx 3.4$ ns.

Methods As the physically relevant dynamics of deca-alanine can be described by dihedral angle coordinates, we use elementary subspaces of functions defined on ten internal dihedral angles for tgEDMD, the outermost dihedrals being more flexible. The elementary subspaces $\mathbb{V}^k$ are the same as in Nüske et al. (2021a), they are spanned by periodic Gaussian functions

$$\psi_{k,u_k}(x) = \exp\left[-\frac{1}{2s_{k,u_k}} \sin^2(0.5(x_k - c_{k,u_k})) \right].$$

Note that $\psi_{k,u_k}(x)$ only depends on the k-th coordinate of the state x. If the k-th coordinate refers to a φ angle, we choose the basis set comprised of the constant function and two periodic Gaussians with $s_{k,u_k} = 0.8, 0.5$ and $c_{k,u_k} = -2.0, 1.0$. If the k-th coordinate refers to a ψ angle, we choose the basis set as the constant function and three periodic Gaussians with $s_{k,u_k} = 0.8, 4.0, 0.8$ and $c_{k,u_k} = -0.5, 0.0, 2.0$. Thus, the full tensor space $\mathbb{V}$ consists of $3^5 \cdot 4^5 \approx 2.5 \cdot 10^5$ basis functions.

As the basis set is defined in terms of reduced coordinates, the effect of this projection needs to be taken into account as described in Sect. 4.2. The full state generator is then, in principle, given as the generator of the molecular dynamics engine on full position and momentum space. For several variants of thermostatted MD, including Langevin dynamics and velocity re-scaling, it is well known that the dynamics on the atomic position space can be approximated by a reversible SDE with constant diffusion on long time scales (see Nüske et al. 2021b for a detailed discussion). However, this

approximation induces a re-scaling of time by a typically unknown factor. Still, we make use of this approximation here by employing the reversible tgEDMD algorithm. Besides the basis functions and their derivatives, the only additional quantity we require is the Jacobian of all dihedral angles with respect to the atomic positions, which can be calculated analytically.

We then compute the reduced matrix $\hat{M}_{r_p}^{\text{rev}}$ by Algorithm 2 using downsampled trajectories of either $m = 3000$ or $m = 30,000$ snapshots. In this example, the dominant singular values of the TT-cores are rather large and differ significantly in size between the cores. Therefore, we employ a relative SVD truncation threshold in contrast to the absolute cut-off used before, i.e., the truncation threshold is set to $\sqrt{m}\epsilon s_1$, where s_1 denotes the largest singular value, and ϵ varies between $\epsilon = 10^{-7}$ and $\epsilon = 10^{-4}$. In addition, we cap the maximally allowed SVD rank at 500. As before, we calculate the dominant eigenvalues and implied time scales of the reduced matrix. In order to account for the change of time units, we calculate the ratio of the reference slowest time scale t_1 and our estimate and re-scale all time scales by that ratio. Of course, this enforces all tgEDMD estimates for t_1 to match the reference exactly. However, by also comparing the next few estimated time scales to their reference values, we can verify whether or not our estimates really match the reference up to a re-scaling of time. Finally, we also compute a clustering of the data into metastable states using the PCCA method (Deuflhard and Weber 2005).

Results In panel A of Fig. 3, we show estimates for the leading three implied time scales t_i, $i = 1, 2, 3$, obtained from the eigenvalues of the reduced matrix. We find that these time scales can be reliably approximated across the range of parameters and data sizes we tested. A ten to fifteen per cent relative error margin is only exceeded for the largest SVD threshold $\epsilon = 10^{-4}$.

The ranks of the associated TT representations of the data tensor are shown in panel B of Fig. 3. The maximal TT rank of 500 is only attained for the smallest threshold $\epsilon = 10^{-7}$. The displayed ranks are similar to the results presented in Nüske et al. (2021a) and we also confirm their finding that TT ranks on the order of 50 to 100 are sufficient to satisfactorily approximate the slow dynamics of the peptide.

In panel C, we verify that the folded and unfolded states were correctly identified by tgEDMD. To this end, we separately compute histogram densities of all data points that were assigned to each metastable state, projected onto each of the five Ramachandran planes. The results obtained for parameters $m = 30,000$, $\epsilon = 10^{-7}$, are shown in the first two rows. These results are consistently reproduced for all thresholds and data sizes we tested. As a reference, we show the same densities based on the clustering obtained from the eigenfunctions computed by tensor-based EDMD as in Nüske et al. (2021a) (last two rows of panel C). We observe that both the folded and unfolded states were identified correctly, up to a slight discrepancy of the relative weights of the alpha- and beta-regions of the first two planes for the unfolded state.

In this ten-dimensional example, the computational benefit of employing the presented tgEDMD method compared to the standard AMUSE algorithm (gEDMD) is already substantial, as Fig. 4 illustrates. For the parameters we tested, the computational effort of tgEDMD is orders of magnitude smaller than for gEDMD. For instance, we see that by setting $\epsilon = 10^{-5}$ and $m = 30,000$, accurate results can be achieved (cf. Fig. 3) at computational cost reduced by more than three orders of magnitude. Moreover, we
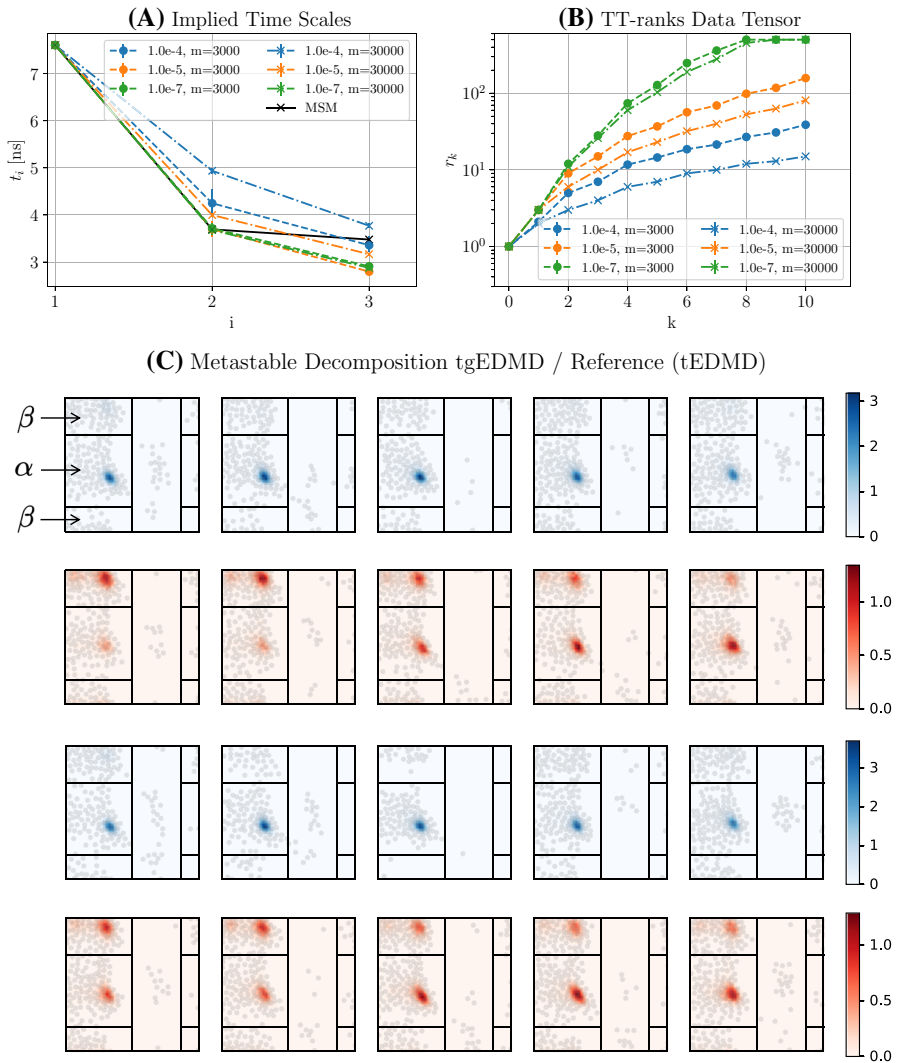
Fig. 3 Results for deca-alanine peptide. **A** Estimates of the leading three implied time scales, obtained by applying the tgEDMD Algorithm 2 with Gaussian basis functions, using different values of the truncation threshold ϵ and different data sizes. Errorbars were generated by selecting ten different subsets of the total simulation data set, which comprises $3 \cdot 10^5$ points. MSM-based reference values are shown in black. **B** TT-ranks for different values of the truncation threshold ϵ and different data sizes. **C** Metastable decomposition of the state space, obtained by applying PCCA to the tgEDMD eigenfunction trajectories ($m = 30,000$ and $\epsilon = 10^{-7}$). First row: density of data points assigned to the first PCCA state (folded, blue) along the five Ramachandran planes. Second row: the same for the second PCCA state (unfolded, red). Third and fourth row: densities of points assigned to the folded and unfolded states by applying PCCA to the eigenfunctions obtained by tensor-based EDMD as in Nüske et al. (2021a)
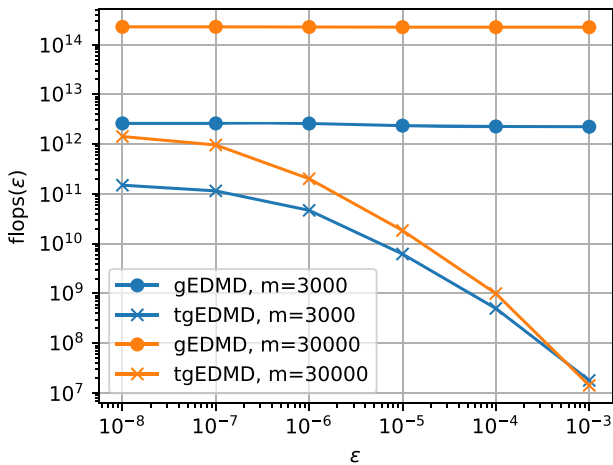
Fig. 4 Computational cost of tgEDMD compared to standard AMUSE (gEDMD), as estimated by Eqs. (27, 28), for the deca-alanine example

confirm once again that the computational effort for tgEDMD depends primarily on the TT ranks, thus offering an efficient means of controlling the computational cost. On the other hand, the cost of gEDMD is dominated by the basis set size N and is essentially independent of the truncation rank.

6 Conclusions

We have presented tgEDMD, a data-driven method to approximate the Kolmogorov backward operator, or generator, of a dynamical system driven by a stochastic differential equation using the tensor train format. The centerpiece of the method is a TT representation of the generator data tensor. We have derived this representation for reversible and non-reversible SDEs. We have shown that tgEDMD consistently approximates the generator on a multi-linear subspace of the full tensor space and also analyzed the computational complexity. We have shown how importance sampling methods can be incorporated and how the method can be used to compute an effective generator on a reduced state space. Finally, we have presented successful applications of the method to a low-dimensional test system and to a benchmarking molecular dynamics data set.

References

Affleck, I., Kennedy, T., Lieb, E.H., Tasaki, H.: Rigorous results on valence-bond ground states in antiferromagnets. Phys. Rev. Lett. **59**(7), 799–802 (1987)

Bakry, D., Gentil, I., Ledoux, M.: Analysis and Geometry of Markov Diffusion Operators, vol. 103. Springer, Berlin (2014)

Bittracher, A., Koltai, P., Klus, S., Banisch, R., Dellnitz, M., Schütte, C.: Transition manifolds of complex metastable systems. J. Nonlinear Sci. **28**(2), 471–512 (2018)

Bowman, G.R., Pande, V.S., Noé, F. (eds.): An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation, vol. 797 of Advances in Experimental Medicine and Biology. Springer, Heidelberg (2014)

Brunton, S.L., Proctor, J.L., Kutz, J.N.: Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc. Natl. Acad. Sci. USA **113**, 3932–3937 (2016)

Budišić, M., Mohr, R., Mezić, I.: Applied Koopmanism. Chaos **22**(4), 047510 (2012)

Chacon, R.V.: An ergodic theorem for operators satisfying norm conditions. J. Appl. Math. Mech. **11**(1), 165–172 (1962)

Chertkov, A., Oseledets, I.: Solution of the Fokker-Planck equation by cross approximation method in the tensor train format. Front. Artif. Intell. **4**, 668215 (2021)

Davies, E.B.: Metastable states of symmetric Markov semigroups II. J. Lond. Math. Soc. **2**(3), 541–556 (1982)

Dellnitz, M., Junge, O.: On the approximation of complicated dynamical behavior. SIAM J. Numer. Anal. **36**(2), 491–515 (1999)

Deuflhard, P., Weber, M.: Robust Perron cluster analysis in conformation dynamics. Linear Algebra Appl. **398**, 161–184 (2005)

Froyland, G., Junge, O., Koltai, P.: Estimating long term behavior of flows without trajectory integration: the infinitesimal generator approach. SIAM J. Numer. Anal. **51**(1), 223–247 (2013)

Gelß, P., Klus, S., Eisert, J., Schütte, C.: Multidimensional approximation of nonlinear dynamical systems. J. Comput. Nonlinear Dyn. **14**(6), 061006 (2019)

Giannakis, D.: Data-driven spectral decomposition and forecasting of ergodic dynamical systems. Appl. Comput. Harmon. Anal. **47**(2), 338–396 (2019)

Hackbusch, W.: Tensor Spaces and Numerical Tensor Calculus. Springer, Berlin (2012)

Kaiser, E., Kutz, J.N., Brunton, S.L.: Discovering conservation laws from data for control. In: 2018 IEEE Conference on Decision and Control (CDC), pp. 6415–6421. IEEE (2018)

Kaiser, E., Kutz, J.N., Brunton, S.L.: Data-driven discovery of Koopman eigenfunctions for control. Mach. Learn. Sci. Technol. **2**(3), 035023 (2021)

Kazeev, V.A., Khoromskij, B.N.: Low-rank explicit QTT representation of the Laplace operator and its inverse. SIAM J. Matrix Anal. Appl. **33**(3), 742–758 (2012)

Kazeev, V., Reichmann, O., Schwab, C.: Low-rank tensor structure of linear diffusion operators in the TT and QTT formats. Linear Algebra Appl. **438**(11), 4204–4221 (2013)

Klus, S., Schütte, C.: Towards tensor-based methods for the numerical approximation of the Perron-Frobenius and Koopman operator. J. Comput. Dyn. **3**(2), 139–161 (2016)

Klus, S., Koltai, P., Schütte, C.: On the numerical approximation of the Perron-Frobenius and Koopman operator. J. Comput. Dyn. **3**(1), 51–79 (2016)

Klus, S., Gelß, P., Peitz, S., Schütte, C.: Tensor-based dynamic mode decomposition. Nonlinearity **31**(7), 3359 (2018a)

Klus, S., Nüske, F., Koltai, P., Wu, H., Kevrekidis, I., Schütte, C., Noé, F.: Data-driven model reduction and transfer operator approximation. J. Nonlinear Sci. **28**(3), 985–1010 (2018b)

Klus, S., Schuster, I., Muandet, K.: Eigendecompositions of transfer operators in reproducing kernel Hilbert spaces. J. Nonlinear Sci. **30**(1), 283–315 (2020a)

Klus, S., Nüske, F., Hamzi, B.: Kernel-based approximation of the Koopman generator and Schrödinger operator. Entropy **22**(7), 722 (2020b)

Klus, S., Nüske, F., Peitz, S., Niemann, J.-H., Clementi, C., Schütte, C.: Data-driven approximation of the Koopman generator: model reduction, system identification, and control. Phys. D **406**, 132416 (2020c)

Koopman, B.O.: Hamiltonian systems and transformation in Hilbert space. Proc. Natl. Acad. Sci. USA **17**(5), 315 (1931)

Lusch, B., Kutz, J.N., Brunton, S.L.: Deep learning for universal linear embeddings of nonlinear dynamics. Nat. Commun. **9**(1), 1–10 (2018)

Mardt, A., Pasquali, L., Wu, H., Noé, F.: VAMPnets for deep learning of molecular kinetics. Nat. Commun. **9**(1), 5 (2018)

Mauroy, A., Goncalves, J.: Koopman-based lifting techniques for nonlinear systems identification. IEEE Trans. Autom. Control **65**(6), 2550–2565 (2019)

Mauroy, A., Mezić, I., Susuki, Y. (eds.): The Koopman Operator in Systems and Control: Concepts, Methodologies, and Applications, vol. 484. Springer, Berlin (2020)

Mezić, I.: Spectral properties of dynamical systems, model reduction and decompositions. Nonlinear Dyn. **41**(1), 309–325 (2005)

Noé, F., Nüske, F.: A variational approach to modeling slow processes in stochastic dynamical systems. Multiscale Model. Simul. **11**, 635–655 (2013)

Nüske, F., Keller, B.G., Pérez-Hernández, G., Mey, A.S.J.S., Noé, F.: Variational approach to molecular kinetics. J. Chem. Theory Comput. **10**, 1739–1752 (2014)

Nüske, F., Schneider, R., Vitalini, F., Noé, F.: Variational tensor approach for approximating the rare-event kinetics of macromolecular systems. J. Chem. Phys. **144**(5), 054105 (2016)

Nüske, F., Gelß, P., Klus, S., Clementi, C.: Tensor-based computation of metastable and coherent sets. Phys. D **427**, 133018 (2021a)

Nüske, F., Koltai, P., Boninsegna, L., Clementi, C.: Spectral properties of effective dynamics from conditional expectations. Entropy **23**(2), 134 (2021b)

Oksendal, B.: Stochastic Differential Equations: An Introduction with Applications. Springer, Berlin (2013)

Oseledets, I.: Tensor-train decomposition. SIAM J. Sci. Comput. **33**, 2295–2317 (2011)

Oseledets, I., Tyrtyshnikov, E.: Breaking the curse of dimensionality, or how to use SVD in many dimensions. SIAM J. Sci. Comput. **31**(5), 3744–3759 (2009)

Östlund, S., Rommer, S.: Thermodynamic limit of density matrix renormalization. Phys. Rev. Lett. **75**(19), 3537 (1995)

Otto, S.E., Rowley, C.W.: Linearly recurrent autoencoder networks for learning dynamics. SIAM J. Appl. Dyn. Syst. **18**(1), 558–593 (2019)

Pazy, A.: Semigroups of Linear Operators and Applications to Partial Differential Equations, vol. 44. Springer, Berlin (2012)

Prinz, J.-H., Wu, H., Sarich, M., Keller, B., Senne, M., Held, M., Chodera, J.D., Schütte, C., Noé, F.: Markov models of molecular kinetics: generation and validation. J. Chem. Phys. **134**, 174105 (2011)

Schmid, P.J.: Dynamic mode decomposition of numerical and experimental data. J. Fluid Mech. **656**, 5–28 (2010)

Schütte, C., Fischer, A., Huisinga, W., Deuflhard, P.: A direct approach to conformational dynamics based on hybrid Monte Carlo. J. Comput. Phys. **151**, 146–168 (1999)

Szalay, S., Pfeffer, M., Murg, V., Barcza, G., Verstraete, F., Schneider, R., Legeza, Ö.: Tensor product methods and entanglement optimization for ab initio quantum chemistry. Int. J. Quantum Chem. **115**(19), 1342–1391 (2015)

Tong, L., Soon, V.C., Huang, Y.F., Liu, R.: AMUSE: a new blind identification algorithm. In: IEEE International Symposium on Circuits and Systems, pp. 1784–1787. IEEE (1990)

Vitalini, F., Mey, A.S.J.S., Noé, F., Keller, B.G.: Dynamic properties of force fields. J. Chem. Phys. **142**(8), 084101 (2015)

Williams, M.O., Kevrekidis, I.G., Rowley, C.W.: A data-driven approximation of the Koopman operator: extending dynamic mode decomposition. J. Nonlinear Sci. **25**(6), 1307–1346 (2015a)

Williams, M.O., Rowley, C.W., Kevrekidis, I.G.: A kernel-based method for data-driven Koopman spectral analysis. J. Comput. Dyn. **2**(2), 247 (2015b)

Wu, H., Noé, F.: Variational approach for learning Markov processes from time series data. J. Nonlinear Sci. **30**(1), 23–66 (2020)

Zhang, W., Hartmann, C., Schütte, C.: Effective dynamics along given reaction coordinates, and reaction rate theory. Faraday Discuss. **195**, 365–394 (2016)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.