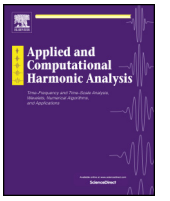
Appl. Comput. Harmon. Anal. 67 (2023) 101575

Contents lists available at ScienceDirect

Applied and Computational Harmonic Analysis

journal homepage: www.elsevier.com/locate/acha



Generative modeling via tensor train sketching

YoonHaeng Hur [a,*], Jeremy G. Hoskins [a], Michael Lindsey [b], E.M. Stoudenmire [c], Yuehaw Khoo [a]

[a] *Department of Statistics, University of Chicago, United States of America*
[b] *Department of Mathematics, Courant Institute of Mathematical Sciences, New York University, United States of America*
[c] *Center for Computational Quantum Physics, Flatiron Institute, United States of America*

ARTICLE INFO

Article history:
Received 9 May 2022
Received in revised form 23 June 2023
Accepted 10 July 2023
Available online 17 July 2023
Communicated by Jared Tanner

Keywords:
Tensor decompositions
Tensor train
Randomized algorithm
Generative modeling
High-dimensional function approximation

ABSTRACT

In this paper, we introduce a sketching algorithm for constructing a tensor train representation of a probability density from its samples. Our method deviates from the standard recursive SVD-based procedure for constructing a tensor train. Instead, we formulate and solve a sequence of small linear systems for the individual tensor train cores. This approach can avoid the curse of dimensionality that threatens both the algorithmic and sample complexities of the recovery problem. Specifically, for Markov models under natural conditions, we prove that the tensor cores can be recovered with a sample complexity that scales logarithmically in the dimensionality. Finally, we illustrate the performance of the method with several numerical experiments.

1. Introduction

Given independent samples from a probability distribution, learning a *generative model* [24] that can produce additional samples is a task of fundamental importance in machine learning and data science. The generative modeling of high-dimensional probability distributions has seen significant recent progress, particularly due to the use of neural-network based parametrizations within both old and new paradigms such as generative adversarial networks (GANs) [11], variational autoencoders (VAE) [16], and normalizing flows [22,30]. Among these three major paradigms, only normalizing flows furnish an analytic formula for the probability density function, and in all cases the computation of downstream quantities of interest can only be achieved via Monte Carlo sampling-based approaches with a relatively low order of convergence.

* Corresponding author.
E-mail addresses: yoonhaenghur@uchicago.edu (Y. Hur), jeremyhoskins@uchicago.edu (J.G. Hoskins), michael.lindsey@cims.nyu.edu (M. Lindsey), mstoudenmire@flatironinstitute.org (E.M. Stoudenmire), ykhoo@uchicago.edu (Y. Khoo).

https://doi.org/10.1016/j.acha.2023.101575
1063-5203/

More precisely, suppose we are given N independent samples

$$(y_1^{(1)}, \dots, y_d^{(1)}), \ \dots \ , (y_1^{(N)}, \dots, y_d^{(N)}) \sim p^\star$$

drawn from an underlying probability density $p^\star \colon \mathbb{R}^d \to \mathbb{R}$, our goal is to estimate $p^\star$ from the empirical distribution

$$\widehat{p}(x_1, \dots, x_d) = \frac{1}{N} \sum_{i=1}^{N} \delta_{(y_1^{(i)}, \dots, y_d^{(i)})}(x_1, \dots, x_d), \tag{1}$$

where $\delta_{(y_1,\dots,y_d)}$ is the δ-measure supported on $(y_1, \dots, y_d) \in \mathbb{R}^d$. In this paper, assuming that the underlying density $p^\star$ takes a low-rank tensor train (TT) [20] format (known as a matrix product state (MPS) in the physics literature [21,34]), we propose and analyze an algorithm that outputs a TT format of $\hat{p}$ to estimate $p^\star$. Such a TT ansatz has found applications in generative modeling; for instance, [14] (and its extension [6]) utilizes it to learn the distribution of handwritten digit images. In particular, the TT ansatz offers several benefits. First, generating independent and identically distributed (i.i.d.) samples can be done efficiently by applying conditional distribution sampling [9] to the obtained TT format; it can also be used for other downstream tasks, such as direct (deterministic) computation of the moments. However, in order to exploit these benefits, we need to be able to determine the TT representation efficiently. Our algorithm, which we name Tensor Train via Recursive Sketching (TT-RS), provides computationally/statistically efficient estimation of $p^\star$, making the following contributions.

- By a sketching technique, we can estimate the tensor components of the TT via a sequence of linear systems, with a complexity that is linear in both the dimension d and the sample size N.
- In the setting of a Markovian density with dimension-independent transition kernels, we prove that the tensor cores can be estimated from a number of samples that scales as $\log(d)$.

1.1. Prior work

In the literature, generally two types of input data are considered for the recovery of low-rank TTs. In the first case, one assumes that one has the ability to evaluate a d-dimensional function p at arbitrary points and seeks to recover p in a TT format with a limited (in particular, polynomial in d) number of evaluations. In this context, various methods such as TT-cross [19], DMRG-cross [25], and TT completion [27] have been considered. Furthermore, generalizations such as [15,31] have been developed to treat densities which have a tensor ring structure. In the second case, which is the case of this paper, one only has access to a fixed collection of empirical samples from the density. Importantly, one does not have access to the value of the density at the given samples. In this case, the ideas of the TT methods that we mentioned earlier cannot be applied directly.

In order to understand how the proposed method differs from the previous methods, we first show that in generative modeling, the nature of the problem is different. More precisely, we are mainly dealing with an estimation problem rather than an approximation problem, where we want to estimate the underlying density $p^\star$ that gives the empirical distribution $\hat{p}$, in terms of a TT. In such a generative modeling setting, suppose one designs an algorithm $\mathcal{A}$ that takes any d-dimensional function p and gives $\mathcal{A}(p)$ as a TT, then one would like such $\mathcal{A}$ to minimize the following differences

$$p^\star - \mathcal{A}(\hat{p}) = \underbrace{p^\star - \mathcal{A}(p^\star)}_{\text{approximation error}} + \underbrace{\mathcal{A}(p^\star) - \mathcal{A}(\hat{p})}_{\text{estimation error}}.$$

In generative modeling, $\hat{p}$ suffers from sample variance, which leads to variance in $\mathcal{A}(\hat{p})$ and hence the estimation error. Our focus is to reduce such an error so that there is no curse of dimensionality in estimating $p^\star$. While our method is inspired by sketching ideas from randomized linear algebra [17,23], which have found applications in the tensor computation field [3,8,29], there are several notable differences with the current literature.

- **In relation to TT-compression algorithms**: Algorithms based on singular value decomposition (SVD) [20] and randomized linear algebra [19,25,26] aim to compress the input function p as a TT such that $\mathcal{A}(p) \approx p$. If such a compression is successful, the above approximation error can be made small, that is, $p^\star - \mathcal{A}(p^\star) \approx 0$, and we also have $\mathcal{A}(\hat{p}) \approx \hat{p}$; accordingly, the estimation error becomes $\mathcal{A}(p^\star) - \mathcal{A}(\hat{p}) \approx p^\star - \hat{p}$. Such an estimation error, however, grows exponentially in d when having a fixed number of samples. In this paper, we focus on developing methods that reduce the estimation error due to sample variance such that there is no curse of dimensionality, and such a setting has not been considered in the previous TT-compression literature.
  A recent work [26] determines a TT from values of a high-dimensional function in a computationally distributed fashion. In particular, [26] forms an independent set of equations with sketching techniques from randomized linear algebra to determine the tensor cores in a parallel way. While our method has similarities with [26], our goal, which is to estimate a TT based on empirical samples of a density, is different from [26]. Therefore, the purpose and means of sketching are fundamentally different. We apply sketching such that each equation in the independent system of equations has size that is constant with respect to the dimension of the problem (unlike the case in [26]), and hence we can estimate the coefficient matrices of the linear system in a statistically efficient way. Furthermore, our use of parallelism in setting up the system is mainly to prevent error accumulation in the estimation of tensor cores.
- **In relation to optimization-based algorithms**: A more principled approach for estimating the underlying density $p^\star$ is to perform maximum likelihood estimation, i.e. minimizing the Kullback-Leibler (KL) divergence between the TT ansatz and the empirical distribution [2,14,18]. Although maximum likelihood estimation is statistically efficient in terms of having a low-variance estimator, due to the non-convex nature of the minimization, these methods can suffer from local minima. Furthermore, these iterative procedures require multiple passes over N data points. In contrast, the method described in this paper recovers the cores with a single sweep across all tensor cores.

1.2. Organization

The paper is organized as follows. First, we briefly describe the main idea of our algorithm in Section 2. Details of the algorithm are presented in Section 3 and conditions for the algorithm to work are discussed in Section 4. In Section 5, we examine how the conditions in Section 4 lead to exact and stable recovery of tensor cores under a Markov model assumption of the density. In Section 6, we illustrate the performance of our algorithm with several numerical examples. We conclude in Section 7.

1.3. Notations

For an integer $n \in \mathbb{N}$, we define $[n] = \{1, \dots, n\}$. Note that for $m, n \in \mathbb{N}$, a function $c\colon [m] \times [n] \to \mathbb{R}$ may also be viewed as a matrix of size $m \times n$. We alternate between these two viewpoints often throughout the paper. For any $a, b \in \mathbb{R}$, we define $a \vee b := \max(a, b)$ and $a \wedge b := \min(a, b)$. For $a, b \in \mathbb{N}$ where $b \geq a$, we may use the "MATLAB notation" $a : b$ to denote the set $\{a, a+1 \dots, b\}$.

Our primary objective in this paper is to obtain a TT representation of a d-dimensional function. Throughout the remainder of this paper, we fix a d-dimensional function $p\colon X_1 \times \cdots \times X_d \to \mathbb{R}$, where $X_1, \dots, X_d \subset \mathbb{R}$.

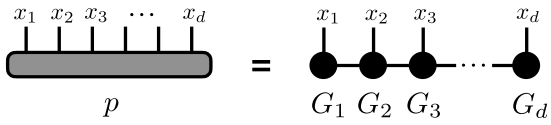


Fig. 1. Tensor diagram illustrating the TT representation of p (Definition 1). The variables x_i's correspond to the outward solid lines in both sides. The solid lines between two adjacent cores on the right-hand side depict the contraction. See [4] for a detailed introduction to tensor network diagram notation.

Unless stated otherwise, p may not be a density, that is, it can take negative values or its integral may not be 1. Whenever we are interested in a density, we will mention explicitly that p is a density or use $p^\star$ instead.

Definition 1. We say that p admits a TT representation of rank $(r_1, \dots, r_{d-1})$ if there exist $G_1 \colon X_1 \times [r_1] \to \mathbb{R}$, $G_k \colon [r_{k-1}] \times X_k \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $G_d \colon [r_{d-1}] \times X_d \to \mathbb{R}$ such that

$$p(x_1, \dots, x_d) = \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} G_1(x_1, \alpha_1) G_2(\alpha_1, x_2, \alpha_2) \cdots G_{d-1}(\alpha_{d-2}, x_{d-1}, \alpha_{d-1}) G_d(\alpha_{d-1}, x_d)$$

for all $(x_1, \dots, x_d) \in X_1 \times \cdots \times X_d$. In this case, we call $G_1, \dots, G_d$ the *cores* of p. For notational simplicity, in the following we often replace the right-hand side of the above equation (and similar expressions involving contractions of several tensors) with $G_1 \circ \cdots \circ G_d$, where '$\circ$' represents the contraction of the cores. We will also sometimes express the TT representation of p diagrammatically as shown in Fig. 1.

Remark 1. In Definition 1, the sets $X_1, \dots, X_d \subset \mathbb{R}$ may be infinite; in such a case, the representation in Definition 1 is also called a functional TT representation [1,12].

Finally, when working with high-dimensional functions, it is often convenient to group the variables into two subsets and think of the resulting object as a matrix. We call these matrices *unfolding matrices*. In particular, for $k = 1, \dots, d-1$, we define the k-th unfolding matrix by $p(x_1, \dots, x_k; x_{k+1}, \dots, x_d)$; namely, group the first k and the last $d-k$ variables to form rows and columns, respectively. In certain situations, for ease of exposition we write $x_{\mathcal{S}}$ to denote the joint variable $(x_{i_1}, \dots, x_{i_k})$, where $\mathcal{S} = \{i_1, \dots, i_k\}$ and $1 \le i_1 \le \cdots \le i_k \le d$. For example, we may write $p(x_1, \dots, x_k; x_{k+1}, \dots, x_d)$ as $p(x_{1:k}; x_{k+1:d})$.

2. Main idea of the algorithm

In this section, we sketch the main idea of the TT-RS algorithm. We start with the following simple observation in the discrete case, i.e., the case where $p \colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$ for $n_1, \dots, n_d \in \mathbb{N}$. Supposing that p is representable in a TT format with rank $(r, \dots, r)$, then the k-th unfolding matrix $p(x_{1:k}; x_{k+1:d})$ is low-rank. Indeed, we can write

$$p(x_{1:k}; x_{k+1:d}) = \sum_{\alpha_k=1}^{r} \Phi_k(x_{1:k}; \alpha_k) \Psi_k(\alpha_k; x_{k+1:d})$$

for some $\Phi_k \colon [n_1] \times \cdots \times [n_k] \times [r] \to \mathbb{R}$ and $\Psi_k \colon [r] \times [n_{k+1}] \times \cdots \times [n_d] \to \mathbb{R}$. On the other hand, the TT-format assumption on p implies that there exist $G_1, \dots, G_d$ such that

$$\Phi_k(x_{1:k}, \alpha_k) := \sum_{\alpha_1=1}^{r} \cdots \sum_{\alpha_{k-1}=1}^{r} G_1(x_1, \alpha_1) \cdots G_k(\alpha_{k-1}, x_k, \alpha_k),$$

$$\Psi_k(\alpha_k, x_{k+1:d}) := \sum_{\alpha_{k+1}=1}^{r} \cdots \sum_{\alpha_{d-1}=1}^{r} G_{k+1}(\alpha_k, x_{k+1}, \alpha_{k+1}) \cdots G_d(\alpha_{d-1}, x_d),$$

so that $p = G_1 \circ \cdots \circ G_d$. In other words, contractions of the first k and the last $d-k$ cores of $G_1, \ldots, G_d$ yield spanning vectors for the r-dimensional column and the row spaces, respectively, of the k-th unfolding matrix.

This observation motivates the following procedure to obtain the cores. Suppose that the rank of the k-th unfolding matrix of p is r. We consider $\Phi_k \colon [n_1] \times \cdots \times [n_k] \times [r] \to \mathbb{R}$ such that the column space of $\Phi_k(x_{1:k}; \alpha_k)$ is the same as that of the k-th unfolding matrix; for instance, a suitable Φ_k can be constructed by forming the SVD of the k-th unfolding matrix $p(x_{1:k}; x_{k+1:d})$ and setting $\Phi_k(x_{1:k}; \alpha_k)$ to be the matrix of left-singular vectors. Next, we attempt to find cores $G_1, \ldots, G_{d-1}$ such that

$$\Phi_k(x_{1:k}, \alpha_k) = \sum_{\alpha_1=1}^{r} \cdots \sum_{\alpha_{k-1}=1}^{r} G_1(x_1, \alpha_1) \cdots G_k(\alpha_{k-1}, x_k, \alpha_k) \tag{2}$$

for $k = 1, \ldots, d-1$. Equivalently, we let $G_1 = \Phi_1$ and solve the following equations for the cores $G_k \colon [r_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ for $k = 2, \ldots, d-1$:

$$\Phi_k(x_{1:k}, \alpha_k) = \sum_{\alpha_{k-1}=1}^{r} \Phi_{k-1}(x_{1:k-1}, \alpha_{k-1}) G_k(\alpha_{k-1}, x_k, \alpha_k). \tag{3}$$

The above discussion has also been studied in [7,26]. For completeness, we formally state it as follows.

Proposition 2. *For each $k = 1, \ldots, d-1$, suppose that the rank of the k-th unfolding matrix of p is r_k and define $\Phi_k \colon [n_1] \times \cdots \times [n_k] \times [r_k] \to \mathbb{R}$ so that the column space of $\Phi_k(x_{1:k}; \alpha_k)$ is the same as that of the k-th unfolding matrix of p. Consider the following d matrix equations with unknowns $G_1 \colon [n_1] \times [r_1] \to \mathbb{R}$, $G_k \colon [r_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ for $k = 2, \ldots, d-1$, and $G_d \colon [r_{d-1}] \times [n_d] \to \mathbb{R}$:*

$$\begin{aligned} G_1(x_1; \alpha_1) &= \Phi_1(x_1; \alpha_1), \\ \sum_{\alpha_{k-1}=1}^{r_{k-1}} \Phi_{k-1}(x_{1:k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) &= \Phi_k(x_{1:k-1}; x_k, \alpha_k) \quad k = 2, \ldots, d-1, \\ \sum_{\alpha_{d-1}=1}^{r_{d-1}} \Phi_{d-1}(x_{1:d-1}; \alpha_{d-1}) G_d(\alpha_{d-1}; x_d) &= p(x_{1:d-1}; x_d). \end{aligned} \tag{4}$$

Then, each equation of (4) has a unique solution, and the solutions $G_1, \ldots, G_d$ satisfy

$$p(x_1, \ldots, x_d) = \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} G_1(x_1, \alpha_1) \cdots G_d(\alpha_{d-1}, x_d). \tag{5}$$

Hence, by solving these equations we obtain a TT representation of p with cores $G_1, \ldots, G_d$. We call (4) the Core Determining Equations (CDEs) formed by $\Phi_1, \ldots, \Phi_{d-1}$.

Proposition 2, which we prove in Appendix A, implies that the cores G_k can be obtained by solving matrix equations. That said, it should be noted that the coefficient matrices of the CDEs, $\Phi_k(x_{1:k}; \alpha_k)$ for $k = 1, \ldots, d-1$, are exponentially sized in the dimension d.

In what follows, we take an approach that is similar in spirit to the "sketching" techniques commonly employed in the randomized SVD literature [13], which are used to dramatically reduce the computational

cost of computing the SVD of several broad classes of matrices. In this paper, however, sketching plays a fundamentally different role. Here, sketching is crucial for the stability of the algorithm, though it also yields an improvement in computational complexity. For our problem, i.e., to determine a TT from samples, the most important function of sketching is to reduce the size of CDEs such that the reduced coefficient matrices can be estimated efficiently with a small sample size N. Furthermore, the choice of sketches cannot be arbitrary (e.g., Gaussian random matrices) but must be chosen carefully to reduce the variance of the coefficient matrices as much as possible. The features and requirements of this sketching strategy are particularly apparent in the case of their application to Markov models, which is treated in Section 5. More concretely, in order to reduce the size of the CDEs, for some function $S_{k-1}\colon [m_{k-1}]\times[n_1]\times\cdots[n_{k-1}]\to\mathbb{R}$, contracting S_{k-1} against (4) (i.e., multiplying both sides by S_{k-1} and summing over $x_1,\dots,x_{k-1}$) we find:

$$\begin{aligned}&\sum_{\alpha_{k-1}=1}^{r_{k-1}}\left(\sum_{x_1=1}^{n_1}\cdots\sum_{x_{k-1}=1}^{n_{k-1}}S_{k-1}(\beta_{k-1};x_{1:k-1})\Phi_{k-1}(x_{1:k-1};\alpha_{k-1})\right)G_k(\alpha_{k-1};x_k,\alpha_k)\\&=\sum_{x_1=1}^{n_1}\cdots\sum_{x_{k-1}=1}^{n_{k-1}}S_{k-1}(\beta_{k-1};x_{1:k-1})\Phi_k(x_{1:k-1};x_k,\alpha_k).\end{aligned}\tag{6}$$

Note that the number of rows of the new coefficient matrix on the left-hand side of (6) is m_{k-1}. Hence, sketching in this way reduces the number of equations to $m_{k-1}n_kr_k$ when determining each G_k. Of course, one must be careful to choose suitable *sketch functions* S_{k-1}, as mentioned previously. As we shall see, Φ_k's are also obtained from some right sketching functions $T_k\colon [n_{k+1}]\times\cdots[n_d]\times[l_k]\to\mathbb{R}$ to be contracted with p over the variables $x_{k+1},\dots,x_d$.

In the next section, we present the details of the proposed algorithm, TT-RS, which gives a set of equations of the form (6).

Remark 2. We pause here to comment on why we solve (2) in the form of (3). To solve (2), one can in principle determine $G_1,\dots,G_d$ successively, i.e. after determining $G_1,\dots G_{k-1}$, plug them into (2) to solve for G_k. In principle, this is the same as solving (3) where each $G_1,\dots,G_d$ is determined independently. But in practice, when Φ_k's contain noise, determining $G_1,\dots,G_d$ successively via substitutions leads to noise accumulation. As we will see later, solving the independent set of equations (3) is more robust against perturbations on the coefficients Φ_k's. We again remark that this independent set of equations is similar to the ones presented in a recent work [26]. However, as mentioned in Section 1.1, our main algorithm presented in the next section is designed to improve statistical estimation, where it is instrumental to reduce the size of the coefficients Φ_k's via the sketching using S_{k-1}'s, whereas equations in [26] are exponentially large.

3. Description of the main algorithm: TT-RS

In this section, we present the algorithm TT-RS (Algorithm 1 below) for the case of determining a TT representation of any discrete d-dimensional function p, where we assume $p\colon [n_1]\times\cdots\times[n_d]\to\mathbb{R}$ for some $n_1,\dots,n_d\in\mathbb{N}$. The stages of Algorithm 1 are depicted in Fig. 2.

Algorithm 1 is divided into four parts: SKETCHING (Algorithm 2), TRIMMING (Algorithm 3), SYSTEM-FORMING (Algorithm 4), and solving d matrix equations (7). As input, Algorithm 1 requires functions $T_2,\dots,T_d$ and $s_1,\dots,s_{d-1}$; we call them *right* and *left sketch functions*, respectively. SKETCHING applies these sketch functions to p so that $\widetilde{\Phi}_k$ resembles the right-hand side of the reduced CDEs (6). In particular, if l_k denotes the number of right sketches and we set $l_k=r_k$ for each k where $r_1,\dots,r_{d-1}$ are the target ranks of the TT, then one could in principle replace the right-hand side of (6) with $\widetilde{\Phi}_k$. In practice, we choose $l_k>r_k$, and use TRIMMING to generate suitable B_k's, to be defined below, from the corresponding $\widetilde{\Phi}_k$'s. These can in turn be used to form a right-hand side in the sense of (6). Lastly, based on $B_1,\dots,B_{d-1}$,

Algorithm 1 TT-RS for a discrete function p.

Require: $p\colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$ and target ranks $r_1, \ldots, r_{d-1}$.
Require: $T_k\colon [n_k] \times \cdots \times [n_d] \times [l_{k-1}] \to \mathbb{R}$ with $l_{k-1} \geq r_{k-1}$ for $k = 2, \ldots, d$.
Require: $s_1\colon [m_1] \times [n_1] \to \mathbb{R}$ and $s_k\colon [m_k] \times [n_k] \times [m_{k-1}] \to \mathbb{R}$ for $k = 2, \ldots, d-1$.
1: $\tilde{\Phi}_1, \ldots, \tilde{\Phi}_d \leftarrow$ SKETCHING$(p, T_2, \ldots, T_d, s_1, \ldots, s_{d-1})$.
2: $B_1, \ldots, B_d \leftarrow$ TRIMMING$(\tilde{\Phi}_1, \ldots, \tilde{\Phi}_d, r_1, \ldots, r_{d-1})$.
3: $A_1, \ldots, A_{d-1} \leftarrow$ SYSTEMFORMING$(B_1, \ldots, B_{d-1}, s_1, \ldots, s_{d-1})$.
4: Solve the following d matrix equations via least-squares for the variables $G_1\colon [n_1] \times [r_1] \to \mathbb{R}$, $G_k\colon [r_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ for $k = 2, \ldots, d-1$, and $G_d\colon [r_{d-1}] \times [n_d] \to \mathbb{R}$:

$$\begin{aligned} G_1 &= B_1, \\ \sum_{\alpha_{k-1}=1}^{r_{k-1}} A_{k-1}(\beta_{k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) &= B_k(\beta_{k-1}; x_k, \alpha_k) \quad k = 2, \ldots, d-1, \\ \sum_{\alpha_{d-1}=1}^{r_{d-1}} A_{d-1}(\beta_{d-1}; \alpha_{d-1}) G_d(\alpha_{d-1}; x_d) &= B_d(\beta_{d-1}; x_d). \end{aligned} \tag{7}$$

5: **return** $G_1, \ldots, G_d$

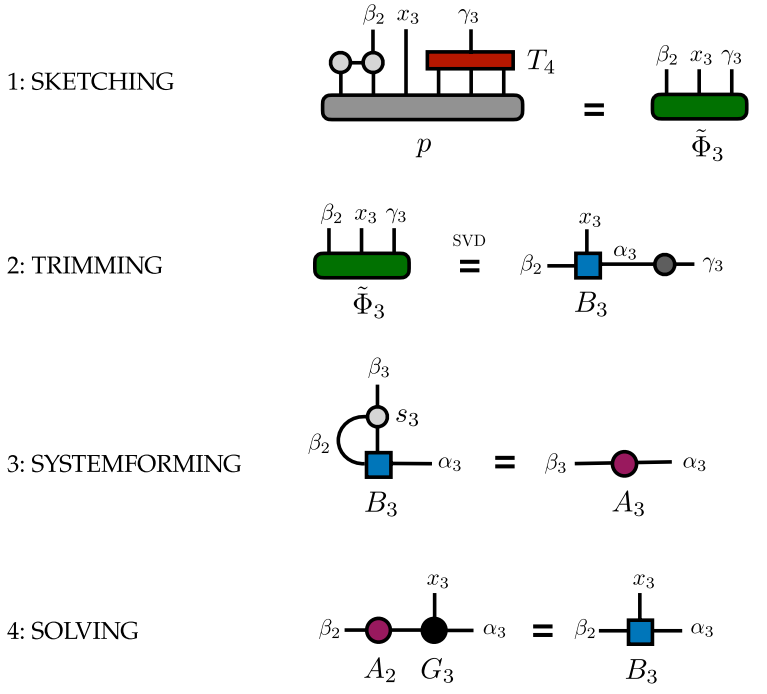


Fig. 2. Tensor diagrams illustrating the four steps of the TT-RS algorithm (Algorithm 1), explicitly showing the case of step $k = 3$ for a $d = 6$ dimensional distribution. SKETCHING produces $\tilde{\Phi}_k$ by applying sketch functions S_{k-1} and T_{k+1} to p. TRIMMING generates B_k from $\tilde{\Phi}_k$ using the SVD. SYSTEMFORMING outputs A_k based on B_k. Lastly, collecting the outputs A_k's and B_k's, we form (7) and solve for G_k's. See Section 3.1 for full details.

SYSTEMFORMING outputs $A_1, \ldots, A_{d-1}$, which resemble the coefficient matrices on the left-hand side of (6). Detailed descriptions of each subroutine are given in the following subsection. In what follows, we constantly refer back to Section 2 to motivate the algorithm.

Remark 3. The choice of sketch functions is based on two criteria: (i) When p actually has an underlying TT representation, solving the equations (7) should produce suitable cores $G_1, \ldots, G_d$. A proof of such an exact recovery property is given in Section 4, where we also discuss the conditions that the sketch functions have to satisfy. (ii) Let $\hat{G}_1, \ldots, \hat{G}_d$ be the results of TT-RS with $\hat{p}$ as input, where $\hat{p}$ is an empirical distribution constructed based on i.i.d. samples from some density $p^\star$. We would like to have $p^\star \approx \hat{G}_1 \circ \cdots \circ \hat{G}_d$ if $\hat{p}$ is a

good approximation to $p^\star$. This requires $A_1, \dots, A_{d-1}, B_1, \dots, B_d$ to have a small variance, and the variance of these objects depends on the choice of sketches. We discuss these considerations in Section 5 for Markov models.

Remark 4. The above algorithm is written only for the case where we consider densities p over a finite state space. However, if p is in $L_2(X_1) \times \cdots \times L_2(X_d)$ then one can pass to a suitable tensor product of orthonormal bases in each dimension and work with truncated coefficient tensors instead. We summarize the necessary modifications required for continuous functions in Appendix C. We call this "continuous" version of the algorithm TT-RS-Continuous (TT-RS-C) (Algorithm 9). There will, of course, be a new source of error associated with the choice of how to truncate the coefficients. Standard estimates from approximation theory can be used to relate the smoothness of p to the decay of coefficients in each dimension.

3.1. Details of the subroutines

In this section, we provide details of the three main subroutines used in TT-RS. First, SKETCHING (Algorithm 2) converts each unfolding matrix of p into a smaller matrix using sketch functions. For each $k = 2, \dots, d-1$, by contracting the k-th unfolding matrix of p with the right and left sketch functions, T_{k+1} and S_{k-1}, we obtain $\widetilde{\Phi}_k$, which can be thought of as a three-dimensional tensor of size $\mathbb{R}^{m_{k-1} \times n_k \times l_k}$ as in Step 1 of Fig. 2. This "sketched" version of the k-th unfolding matrix of p is no longer exponentially large in d. In SKETCHING, each $\bar{\Phi}_k$ plays the role of Φ_k in the left-hand side of (2), which captures the range of the k-th unfolding matrix of p. The extra "bar" in the notation for $\bar{\Phi}_k$ is used to distinguish this object from Φ_k, as $\bar{\Phi}_k(x_{1:k}; \gamma_k)$ has $l_k \geq r_k$ columns, while $\Phi_k(x_{1:k}; \alpha_k)$ only has r_k columns. Such "oversampling" [13] is standard in randomized linear algebra algorithms for capturing the range of a matrix effectively. Then, as in (6), left sketches S_k's are applied to further reduce $\bar{\Phi}_k$'s to $\widetilde{\Phi}_k$'s. As mentioned previously, $\widetilde{\Phi}_k$ resembles the right-hand side of (6), though the $\widetilde{\Phi}_k$'s need to be further processed by TRIMMING. An important remark here is that unlike the right sketch functions $T_2, \dots T_d$, the left sketch functions $S_2, \dots, S_{d-1}$ are constructed sequentially, i.e., S_k is obtained by contracting a small block s_k with S_{k-1}; hence, it is a sequential contraction of $s_1, \dots, s_k$. Such a design is necessary as is shown in SYSTEMFORMING. Another remark is that Algorithm 2 is presented in a modular fashion for the sake of clarity. In fact, many computations in Algorithm 2 can be re-used by leveraging the fact that S_k is obtained from the contraction of S_{k-1} and s_k. Hence, $\widetilde{\Phi}_k$ can be obtained recursively from $\widetilde{\Phi}_{k-1}$.

TRIMMING takes the outputs $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_{d-1}$ of SKETCHING and further process them to have the appropriate rank of the underlying TT using the SVD. This procedure is illustrated in Step 2 in Fig. 2. It should be noted that this procedure is not necessary if for any k, $l_k = r_k$. In this case, one should directly let $B_k = \widetilde{\Phi}_k$ for each k.

Finally, SYSTEMFORMING forms the coefficient matrices to solve for $G_1, \dots, G_d$ from the output $B_1, \dots, B_{d-1}$ of TRIMMING by contracting $s_1, \dots, s_{d-1}$ with them, which results in $A_1, \dots, A_{d-1}$, respectively, as in Step 3 of Fig. 2. The matrices $A_1, \dots, A_{d-1}$ play the role of the coefficient matrices appearing on the left-hand side of (6). As we see in the algorithm, the fact that the sketch functions $S_1, \cdots, S_{d-1}$ are obtained by successive contractions of $s_1, \dots, s_{d-1}$ allows A_k to be constructed from B_k. We stress that this is not merely for the sake of efficient computation. In fact, it is important for the correctness of the algorithm, as illustrated in the proof of recovery for Markov models in Section 5 below.

3.2. Complexity

As noted earlier, we are practically interested in the case where p is an empirical distribution $\hat{p}$ constructed from N i.i.d. samples from an underlying density $p^\star$. In such a case, $\hat{p}$ is N-sparse. The high-dimensional integrals within TT-RS can be efficiently computed in this case. To see this, suppose that the input p

Algorithm 2 SKETCHING.

Require: p, $T_2, \dots, T_d$, and $s_1, \dots, s_{d-1}$ as given in Algorithm 1.
for $k = 1$ to $d-1$ **do**
Right sketching: define $\bar{\Phi}_k \colon [n_1] \times \cdots \times [n_k] \times [l_k] \to \mathbb{R}$ as

$$\bar{\Phi}_k(x_{1:k}, \gamma_k) = \sum_{x_{k+1}=1}^{n_{k+1}} \cdots \sum_{x_d=1}^{n_d} p(x_{1:k}, x_{k+1:d}) T_{k+1}(x_{k+1:d}, \gamma_k).$$

if $k > 1$ **then**
Left sketching: define $\widetilde{\Phi}_k \colon [m_{k-1}] \times [n_k] \times [l_k] \to \mathbb{R}$ as

$$\widetilde{\Phi}_k(\beta_{k-1}, x_k, \gamma_k) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1}) \bar{\Phi}_k(x_{1:k-1}, x_k, \gamma_k).$$

Compute sketch function $S_k \colon [m_k] \times [n_1] \times \cdots \times [n_k] \to \mathbb{R}$ for the next iteration:

$$S_k(\beta_k, x_{1:k}) = \sum_{\beta_{k-1}=1}^{m_{k-1}} s_k(\beta_k, x_k, \beta_{k-1}) S_{k-1}(\beta_{k-1}, x_{1:k-1}).$$

else
Define

$$\widetilde{\Phi}_1(x_1, \gamma_1) = \bar{\Phi}_1(x_1, \gamma_1).$$

Define sketch function

$$S_1(\beta_1, x_1) = s_1(\beta_1, x_1).$$

end if
end for
Left sketching: define $\widetilde{\Phi}_d \colon [m_{d-1}] \times [n_d] \to \mathbb{R}$ as

$$\widetilde{\Phi}_d(\beta_{d-1}, x_d) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_{d-1}=1}^{n_{d-1}} S_{d-1}(\beta_{d-1}, x_{1:d-1}) p(x_{1:d-1}, x_d).$$

return $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$.

Algorithm 3 TRIMMING.

Require: $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$ from Algorithm 2.
Require: Target ranks $r_1, \dots, r_{d-1}$ as given in Algorithm 1.
for $k = 1$ to $d-1$ **do**
if $k = 1$ **then**
Compute the first r_1 left singular vectors of $\widetilde{\Phi}_1(x_1; \gamma_1)$ and define $B_1 \colon [n_1] \times [r_1] \to \mathbb{R}$ so that these singular vectors are the columns of $B_1(x_1; \alpha_1)$.
else
Compute the first r_k left singular vectors of $\widetilde{\Phi}_k(\beta_{k-1}, x_k; \gamma_k)$ and define $B_k \colon [m_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ so that these singular vectors are the columns of $B_k(\beta_{k-1}, x_k; \alpha_k)$.
end if
end for
Let $B_d(\beta_{d-1}, x_d) = \widetilde{\Phi}_d(\beta_{d-1}, x_d)$.
return $B_1, \dots, B_d$.

of Algorithm 1 is N-sparse, and let $n = \max_{1 \le k \le d} n_k$, $m = \max_{1 \le k \le d-1} m_k$, $l = \max_{1 \le k \le d-1} l_k$, and $r = \max_{1 \le k \le d-1} r_k$. Note that the complexity of SKETCHING is $O(mlNd)$ since each $\widetilde{\Phi}_k$ can be computed in $O(mlN)$ time. TRIMMING requires $O(mnl^2 d)$ operations as each B_k is computed using SVD in $O(mnl^2)$ times. Also, SYSTEMFORMING is achieved in $O(m^2 nrd)$ time. Lastly, the equations (7) can be solved in $O(mnr^3 d)$ time. In summary, the total computational cost of TT-RS with N-sparse input is

Algorithm 4 SystemForming.

Require: $B_1, \dots, B_{d-1}$ from Algorithm 2.
Require: $s_1, \dots, s_{d-1}$ as given in Algorithm 1.
  for $k = 1$ to $d-1$ **do**
    if $k = 1$ **then**
      Compute $A_1 \colon [m_1] \times [r_1] \to \mathbb{R}$:

$$A_1(\beta_1, \alpha_1) = \sum_{x_1=1}^{n_1} s_1(\beta_1, x_1) B_1(x_1, \alpha_1).$$

    else
      Compute $A_k \colon [m_k] \times [r_k] \to \mathbb{R}$:

$$A_k(\beta_k, \alpha_k) = \sum_{x_k=1}^{n_k} \sum_{\beta_{k-1}=1}^{m_{k-1}} s_k(\beta_k, x_k, \beta_{k-1}) B_k(\beta_{k-1}, x_k, \alpha_k).$$

    end if
  end for
  return $A_1, \dots, A_{d-1}$.

$$O(mlNd) + O(mnl^2 d) + O(m^2 nrd) + O(mnr^3 d).$$

Note that this cost is linear in both n and the dimension d of the distribution.

Remark 5. The term "recursive sketching" in the name TT-RS is due to the sequential contraction of the left sketch functions $s_1, \dots, s_{d-1}$. We remark that it is possible to design an algorithm without such "recursiveness", which we call TT-Sketch (TT-S); see Appendix B for the details.

4. Conditions for exact recovery for TT-RS

The main purpose of this section is to provide sufficient conditions for when TT-RS can recover an underlying TT if the input function p admits a representation by a tensor train. In particular, the following theorem provides a guideline for choosing the sketch functions in TT-RS.

Theorem 3. *Assume the rank (in exact arithmetic) of the k-th unfolding matrix of p is r_k for each $k = 1, \dots, d-1$. Suppose $T_2, \dots, T_d$ and $s_1, \dots, s_{d-1}$ of Algorithm 1 satisfy the following.*

(i) $\bar{\Phi}_k(x_{1:k}; \gamma_k)$ *and* $p(x_{1:k}; x_{k+1:d})$ *have the same column space for* $k = 1, \dots, d-1$.
(ii) $\widetilde{\Phi}_k(\beta_{k-1}, x_k; \gamma_k)$ *and* $\bar{\Phi}_k(x_{1:k}; \gamma_k)$ *have the same row space for* $k = 2, \dots, d-1$.
(iii) $A_k(\beta_k; \alpha_k)$ *is rank-*r_k *for* $k = 1, \dots, d-1$.

Then, each equation of (7) has a unique solution, and the solutions $G_1, \dots, G_d$ are cores of p.

We first present a lemma showing that Sketching and Trimming give rise to the right-hand side of (6) for determining the cores of p.

Lemma 4. *Under the assumptions of Theorem 3, consider the results $B_1, \dots, B_{d-1}$ produced by Algorithms 2 and 3. The column space of $B_1(x_1; \alpha_1)$ is the same as that of the first unfolding matrix of p. Also, for each $k = 2, \dots, d-1$, there exists a $\Phi_k \colon [n_1] \times \cdots \times [n_k] \times [r_k] \to \mathbb{R}$ such that the column space of $\Phi_k(x_{1:k}; \alpha_k)$ is the same as that of the k-th unfolding matrix and*

$$B_k(\beta_{k-1}, x_k, \alpha_k) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1}) \Phi_k(x_{1:k-1}, x_k, \alpha_k). \tag{8}$$

Proof. (i) implies that $\widetilde{\Phi}_1(x_1;\gamma_1) = \bar{\Phi}_1(x_1;\gamma_1)$ and $p(x_1;x_{2:d})$ have the same r_1-dimensional column space, which is the same as the column space of $B_1(x_1;\alpha_1)$ by the definition of B_1.

For $k = 2,\dots,d-1$, (i) and (ii) imply that $\widetilde{\Phi}_k(\beta_{k-1},x_k;\gamma_k)$ is still rank-r_k. Since the columns of $B_k(\beta_{k-1},x_k;\alpha_k)$ are the first r_k left singular vectors of $\widetilde{\Phi}_k(\beta_{k-1},x_k;\gamma_k)$, we may write

$$B_k(\beta_{k-1},x_k;\alpha_k) = \sum_{\gamma_k=1}^{l_k} \widetilde{\Phi}_k(\beta_{k-1},x_k;\gamma_k) q_{k+1}(\gamma_k;\alpha_k)$$

for some $q_{k+1}\colon [l_k]\times[r_k]\to\mathbb{R}$; here, the column space of $q_{k+1}(\gamma_k;\alpha_k)$ is the same as the row space of $\widetilde{\Phi}_k(\beta_{k-1},x_k;\gamma_k)$. Now, we define $\Phi_k\colon [n_1]\times\cdots[n_k]\times[r_k]\to\mathbb{R}$ by

$$\Phi_k(x_{1:k},\alpha_k) = \sum_{\gamma_k=1}^{l_k} \bar{\Phi}_k(x_{1:k},\gamma_k) q_{k+1}(\gamma_k,\alpha_k). \tag{9}$$

Next, we observe that (8) holds since

$$\begin{aligned} B_k(\beta_{k-1},x_k,\alpha_k) &= \sum_{\gamma_k=1}^{l_k} \widetilde{\Phi}_k(\beta_{k-1},x_k,\gamma_k) q_{k+1}(\gamma_k,\alpha_k) \\ &= \sum_{\gamma_k=1}^{l_k}\sum_{x_1=1}^{n_1}\cdots\sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1},x_{1:k-1})\bar{\Phi}_k(x_{1:k-1},x_k,\gamma_k) q_{k+1}(\gamma_k,\alpha_k). \end{aligned}$$

We claim that the column space of $\Phi_k(x_{1:k};\alpha_k)$ is the same as that of the k-th unfolding matrix. Indeed, due to (9), the column space of $\Phi_k(x_{1:k};\alpha_k)$ is contained in that of $\bar{\Phi}_k(x_{1:k};\gamma_k)$, which is the column space of the k-th unfolding matrix because of (i). Now, it suffices to prove that $\Phi_k(x_{1:k};\alpha_k)$ has full column rank. This is true because the column space of $q_{k+1}(\gamma_k;\alpha_k)$ is the same as the row space of $\widetilde{\Phi}_k(\beta_{k-1},x_k;\gamma_k)$ by construction, which is equivalent to the row space of $\bar{\Phi}_k(x_{1:k};\gamma_k)$ due to (ii). □

In Lemma 4, we showed that SKETCHING and TRIMMING give the right-hand sides of (6) (i.e., B_k in (8)), without forming the exponentially-sized Φ_k explicitly. Lastly, by combining SKETCHING and TRIMMING with SYSTEMFORMING, we have a well-defined system of equations for determining $G_1,\dots,G_d$, as in Algorithm 1. This is shown in the following proof for Theorem 3.

Proof of Theorem 3. Due to Lemma 4, there exists $\Phi_2,\dots,\Phi_{d-1}$ such that (8) holds; also, letting $\Phi_1 = B_1$, we have shown that $\Phi_k(x_{1:k};\alpha_k)$ and the k-th unfolding matrix have the same column space for $k = 1,\dots,d-1$. Hence, we can consider CDEs (4) formed by $\Phi_1,\dots,\Phi_{d-1}$. First, we verify that the equations in (7) are implied by (4), obtained by applying sketch functions to both sides of (4). The first equation $G_1 = \Phi_1$ is the same in both (7) and (4). For $k = 2,\dots,d-1$, if we apply S_{k-1} to both sides of the k-th equation of (4), then

$$\begin{aligned} &\sum_{x_1=1}^{n_1}\cdots\sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1},x_{1:k-1}) \sum_{\alpha_{k-1}=1}^{r_{k-1}} \Phi_{k-1}(x_{1:k-1},\alpha_{k-1}) G_k(\alpha_{k-1},x_k,\alpha_k) \\ &= \sum_{x_1=1}^{n_1}\cdots\sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1},x_{1:k-1})\Phi_k(x_{1:k},\alpha_k). \end{aligned} \tag{10}$$

Note that the right-hand side of (10) is simply $B_k(\beta_{k-1},x_k,\alpha_k)$, which is the right-hand side of the k-th equation of (7). We now want to show that the coefficient matrix on the left-hand side of (10) is the coefficient matrix $A_{k-1}(\beta_{k-1},\alpha_{k-1})$ of the k-th equation of (7), that is, we want to prove for $k = 2,\dots,d-1$,

$$\sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1})\Phi_{k-1}(x_{1:k-1}, \alpha_{k-1}) = A_{k-1}(\beta_{k-1}, \alpha_{k-1}), \tag{11}$$

This is implied by Algorithm 4. To see this, for $k = 2$, note that (11) amounts to

$$\sum_{x_1=1}^{n_1} s_1(\beta_1, x_1)B_1(x_1, \alpha_1) = A_1(\beta_1, \alpha_1),$$

which follows immediately from Algorithm 4. For $2 < k \leq d-1$, (11) holds because

$$\begin{aligned}
&\sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1})\Phi_{k-1}(x_{1:k-1}, \alpha_{k-1}) \\
&= \sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} \sum_{\beta_{k-2}=1}^{m_{k-2}} s_{k-1}(\beta_{k-1}, x_{k-1}, \beta_{k-2})S_{k-2}(\beta_{k-2}, x_{1:k-2})\Phi_{k-1}(x_{1:k-1}, \alpha_{k-1}) \\
&= \sum_{x_{k-1}=1}^{n_{k-1}} \sum_{\beta_{k-2}=1}^{m_{k-2}} s_{k-1}(\beta_{k-1}, x_{k-1}, \beta_{k-2})B_{k-1}(\beta_{k-2}, x_{k-1}, \alpha_{k-1}) \\
&= A_{k-1}(\beta_{k-1}, \alpha_{k-1}),
\end{aligned}$$

where the first equality holds since S_{k-1} is a contraction of s_{k-1} and S_{k-2}, the second equality holds because of (8), and the last equality is given in Algorithm 4. Hence, we have shown that for $k = 2, \ldots, d-1$, the k-th equation of (7) is indeed obtained by applying S_{k-1} to both sides of the k-th equation of (4). Similarly, the last equation of (7) is obtained by applying S_{d-1} to both sides of the last equation of (4). From this it is clear that solutions $G_1, \ldots, G_d$ of (4) formed by $\Phi_1, \ldots, \Phi_{d-1}$ satisfy (7). Now, we use condition (iii) in Theorem 3; this means that the coefficient matrices $A_1, \ldots, A_{d-1}$ have full column rank, and thus each equation of (7) must have a unique solution. Therefore, a unique set of solutions $G_1, \ldots, G_d$ of (4) formed by $\Phi_1, \ldots, \Phi_{d-1}$ discussed in Proposition 2 gives rise to a unique set of solutions of (7). Additionally, as in Proposition 2, $G_1, \ldots, G_d$ give a TT representation of p. □

5. Application of TT-RS to Markov model

In this section, we demonstrate how model assumptions on p can guide the choice of sketch functions $T_2, \ldots, T_d$ and $s_1, \ldots, s_{d-1}$ to guarantee that the conditions (i)-(iii) of Theorem 3 are satisfied. More precisely, we show that for Markov models, suitable sketch functions exist, and moreover, we give an explicit construction. In Section 5.2 we prove that the sketch functions we construct satisfy the requisite conditions. When working with an empirical distribution $\hat{p}$ which is constructed based on i.i.d. samples from some underlying density $p^\star$, TT-RS requires obtaining $B_1, \ldots, B_d$, $A_1, \ldots, A_{d-1}$ by taking expectations over the empirical distribution. Though the variance can be large, in Section 5.3, we show that under certain natural conditions, our choice of sketch functions does not suffer from the "curse of dimensionality" when estimating the cores from the empirical distribution.

Throughout this section, we assume that the input p of TT-RS (Algorithm 1) is a Markov model, that is, p is a probability density function and satisfies

$$p(x_1, \ldots, x_d) = p(x_1)p(x_2|x_1) \cdots p(x_d|x_{d-1}). \tag{12}$$

Here, by abuse of notation, for any $i < j$, we denote the marginal density of $(x_i, \ldots, x_j)$ as $p(x_i, \ldots, x_j)$. Depending on the situation, we also use

$$(\mathcal{M}_{\mathcal{S}}p)(x_{\mathcal{S}}) := p(x_{\mathcal{S}}), \quad \mathcal{S} \subset [d] \tag{13}$$

to denote the marginalization of p to the variables given by the index set $\mathcal{S}$, which is a $|\mathcal{S}|$-dimensional function. Also, $p(x_i, \dots, x_j|x_k)$ denotes the conditional density of $(x_i, \dots, x_j)$ given x_k. For a Markov model p, the conditional probabilities $p(x_2|x_1), \dots, p(x_d|x_{d-1})$ in (12) are referred to as the transition kernels.

5.1. Choice of sketch

We start with the following simple lemma that shows the low-dimensional nature of the column and row spaces of the unfolding matrices.

Lemma 5. *Suppose p is a Markov model. For any $i \le k < j$,*

(i) *$p(x_{i:k}; x_{k+1:j})$ and $p(x_{i:k}; x_{k+1})$ have the same column spaces,*
(ii) *$p(x_{i:k}; x_{k+1:j})$ and $p(x_k; x_{k+1:j})$ have the same row spaces.*

Proof. Since $x_{i:k} \perp x_{k+2:j} \mid x_{k+1}$ (conditional independence), we have that

$$p(x_{i:k}; x_{k+1:j}) = p(x_{i:k}|x_{k+1})p(x_{k+2:j}|x_{k+1})p(x_{k+1}),$$

which implies that the column space of $p(x_{i:k}; x_{k+1:j})$ is not affected by $x_{k+2:j}$. For the same reason, $x_{i:k-1} \perp x_{k+1:j} \mid x_k$ implies

$$p(x_{i:k}; x_{k+1:j}) = p(x_{i:k-1}|x_k)p(x_{k+1:j}|x_k)p(x_k),$$

and hence the row space of $p(x_{i:k}; x_{k+1:j})$ is not affected by $x_{i:k-1}$. □

An immediate consequence of Lemma 5 is that each unfolding matrix $p(x_{1:k}; x_{k+1:d})$ may be replaced by $p(x_{1:k}; x_{k+1})$ if our main focus is the column space. This motivates a specific choice of sketch functions for a Markov model. For each $k = 1, \dots, d-1$, let $l_k = n_{k+1}$ and define

$$T_{k+1}(x_{k+1:d}, \gamma_k) = I_{k+1}(x_{k+1}; \gamma_k), \tag{14}$$

where $I_{k+1} \colon [n_{k+1}] \times [n_{k+1}] \to \mathbb{R}$ such that $I_{k+1}(x_{k+1}; \gamma_k)$ is the identity matrix. This choice of T_{k+1} yields

$$\bar{\Phi}_k(x_{1:k}, \gamma_k) = (\mathcal{M}_{1:k+1}p)(x_{1:k}, \gamma_k). \tag{15}$$

In other words, contracting T_{k+1} with the k-th unfolding matrix amounts to marginalizing out variables $x_{k+2}, \dots, x_d$.

Similarly, we let $m_k = n_k$ for each $k = 1, \dots, d-1$, and define

$$s_1(\beta_1, x_1) = I_1(\beta_1; x_1), \quad s_k(\beta_k, x_k, \beta_{k-1}) = I_k(\beta_k; x_k), \tag{16}$$

where $I_1 \colon [n_1] \times [n_1] \to \mathbb{R}$ is defined so that $I_1(\beta_1; x_1)$ is the identity matrix, which gives rise to $S_k(\beta_k, x_{1:k}) = I_k(\beta_k, x_k)$ and

$$\widetilde{\Phi}_1 = \mathcal{M}_{\{1,2\}}p, \quad \widetilde{\Phi}_k = \mathcal{M}_{\{k-1,k,k+1\}}p, \ 2 \le k \le d-1, \quad \widetilde{\Phi}_d = \mathcal{M}_{\{d-1,d\}}p. \tag{17}$$

Again, this choice of left sketch functions leads to S_{k-1} that marginalizes out variables $x_1, \dots, x_{k-2}$.

In summary, with these sketch functions, SKETCHING outputs marginals of p. Now, it is obvious that Algorithm 3 and 4 can be done efficiently; it just performs an SVD on these small marginal matrices and computes both $A_1 = B_1$ and

$$A_k(x_k, \alpha_k) = \sum_{x_{k-1}=1}^{n_{k-1}} B_k(x_{k-1}, x_k, \alpha_k)$$

for $k \geq 2$.

Remark 6. For the situation where p is a function of continuous variables, in Appendix C.1 we discuss how to adapt TT-RS-C (Algorithm 9) to the Markov case.

5.2. Exact recovery for Markov models

In this subsection, we prove that if we use TT-RS (Algorithm 1) in conjunction with the sketches defined in (14) and (16), then the resulting algorithm enjoys the exact recovery property. Using Theorem 3, it suffices to check the choice of sketch functions mentioned in the previous subsection satisfies (i)-(iii) of Theorem 3.

Theorem 6. *Let p be a discrete Markov model such that the rank (in exact arithmetic) of the k-th unfolding matrix of p is r_k for each $k = 1, \dots, d-1$. With right and left sketches in (14), (16), Algorithm 1 returns $G_1, \dots, G_d$ as cores of p.*

Proof. It suffices to check that (i)-(iii) of Theorem 3 are satisfied. As noted earlier, for each $k = 1, \dots, d-1$, (15) holds. Hence, $\bar{\Phi}_k(x_{1:k}; \gamma_k)$ and $p(x_{1:k}; x_{k+1:d})$ have the same column space by Lemma 5. Thus, (i) of Theorem 3 holds. Similarly, for each $k = 2, \dots, d-1$, (17) holds, hence, $\widetilde{\Phi}_k(\beta_{k-1}, x_k; \gamma_k)$ and $\bar{\Phi}_k(x_{1:k}; \gamma_k)$ have the same row space. Thus, (ii) of Theorem 3 holds.

Lastly, we claim $A_k(x_k; \alpha_k)$ is rank-r_k for all $k = 1, \dots, d-1$ (condition (iii) of Theorem 3). Clearly, $A_1(x_1; \alpha_1) = B_1(x_1; \alpha_1)$ is rank-r_1 by definition. For $k = 2, \dots, d-1$, by definition of B_k, we can find $q_{k+1} \colon [n_{k+1}] \times [r_k] \to \mathbb{R}$ such that the column space of $q_{k+1}(x_{k+1}; \alpha_k)$ is the same as the row space of $p(x_{k-1}, x_k; x_{k+1})$ and

$$B_k(x_{k-1}, x_k, \alpha_k) = \sum_{x_{k+1}=1}^{n_{k+1}} p(x_{k-1}, x_k, x_{k+1}) q_{k+1}(x_{k+1}, \alpha_k).$$

Hence,

$$\begin{aligned}
A_k(x_k, \alpha_k) &= \sum_{x_{k-1}=1}^{n_{k-1}} B_k(x_{k-1}, x_k, \alpha_k) \\
&= \sum_{x_{k-1}=1}^{n_{k-1}} \sum_{x_{k+1}=1}^{n_{k+1}} p(x_{k-1}, x_k, x_{k+1}) q_{k+1}(x_{k+1}, \alpha_k). \\
&= \sum_{x_{k+1}=1}^{n_{k+1}} \left(\sum_{x_{k-1}=1}^{n_{k-1}} p(x_{k-1}, x_k, x_{k+1}) \right) q_{k+1}(x_{k+1}, \alpha_k) \\
&= \sum_{x_{k+1}=1}^{n_{k+1}} p(x_k, x_{k+1}) q_{k+1}(x_{k+1}, \alpha_k).
\end{aligned}$$

By Lemma 5, $p(x_{k-1}, x_k; x_{k+1})$ and $p(x_k; x_{k+1})$ have the same row space. Therefore, the column space of $q_{k+1}(x_{k+1}; \alpha_k)$ is the same as the row space of $p(x_k; x_{k+1})$, where both are rank-r_k. Thus, $A_k(x_k; \alpha_k)$ must be rank-r_k by construction. □

5.3. Stable estimation for Markov models

In this section, we present an informal result regarding the stability of the TT-RS algorithm when an empirical distribution $\hat{p}$ is provided as input instead of the true density $p^\star$. The precise statement of the theorem is deferred to Appendix D. If $\hat{p}$ is taken as the input of Algorithm 1, the results $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$ of SKETCHING have certain variances that get propagated to the final output $G_1, \dots, G_d$ via the coefficient matrices $A_1, \dots, A_{d-1}, B_1, \dots, B_d$. The variances of $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$ depend critically on the choice of sketch functions. In what follows, we show that the sketches (14) and (16) give a nearly dimension-independent error when estimating the tensor cores if $p^\star$ is a Markov model satisfying the following natural condition.

Condition 1. The transition kernels $p^\star(x_2|x_1), \dots, p^\star(x_d|x_{d-1})$ are independent of d.

Theorem 7 *(Informal statement of Theorem 19). Suppose $p^\star$ is a discrete Markov model that satisfies Condition 1 and admits a TT-representation with rank $(r_1, \dots, r_{d-1})$. Consider an empirical distribution $\hat{p}$ constructed based on N i.i.d. samples from $p^\star$. Let $\hat{G}_1, \dots, \hat{G}_d$ and $G_1^\star, \dots, G_d^\star$ be the results of TT-RS with $\hat{p}$ and $p^\star$ as input, respectively. Then, with high probability,*

$$\frac{\operatorname{dist}(\hat{G}_k, G_k^\star)}{|||G_k^\star|||} \le O\left(\frac{\sqrt{\log(d)}}{\sqrt{N}}\right) \quad \forall k = 1, \dots, d, \tag{18}$$

where the hidden constant in the "big-O" notation does not depend on the dimensionality d, $|||\cdot|||$ is some appropriate norm, and $\operatorname{dist}(\cdot,\cdot)$ is a suitable measure of distance between cores.

In Theorem 7, the errors in the cores show $\sqrt{\log(d)}$-dependence which grows very slowly in d; the term $\sqrt{\log(d)}$ is a consequence of the union bound required to derive a probabilistic bound on d objects (the cores) simultaneously. We remark, however, that near dimension-independent errors in the pairs $(G_k^\star, \hat{G}_k)$ do not necessarily imply such an error in approximating $p^\star$ by $\hat{G}_1 \circ \cdots \circ \hat{G}_d$, the results of TT-RS with $\hat{p}$ as input. Instead, we can derive an error that scales almost linearly in d, thereby avoiding the curse of dimensionality. The precise statement is deferred to Appendix D; here, we provide an informal statement summarizing this result.

Corollary 8 *(Informal statement of Theorem 20). In the setting of Theorem 7, with high probability,*

$$\frac{\|\hat{G}_1 \circ \cdots \circ \hat{G}_d - G_1^\star \circ \cdots \circ G_d^\star\|_\infty}{|||G_1^\star||| \dots |||G_d^\star|||} \le O\left(\frac{d\sqrt{\log(d)}}{\sqrt{N}}\right),$$

where $\|\cdot\|_\infty$ denotes the largest absolute value of the entries of a tensor.

In Section 6, we verify from the experiments that such $d\sqrt{\log(d)}$-dependence of the error indeed suggests near-linear dependence on the dimensionality d.

Remark 7. Extensive numerical experiments suggest that Theorem 7 and Corollary 20 are valid for a broad class of Markov models that may not necessarily satisfy Condition 1. See Remark 11 for details.

5.4. Higher-order Markov models

We conclude this section with a brief discussion on higher-order Markov models. For $m \in \mathbb{N}$, we call p an order-m Markov model if it is a density and satisfies

$$p(x_1, \dots, x_d) = p(x_1, \dots, x_m)p(x_{m+1}|x_1, \dots, x_m) \cdots p(x_d|x_{d-m}, \dots, x_{d-1}).$$

What we have presented so far, i.e., the case $m = 1$, can be generalized to any $m \in \mathbb{N}$ by a suitable replacement of the sketch functions $T_2, \dots, T_d$ and $s_1, s_2, \dots, s_{d-1}$. Recall that the sketch functions for the case where $m = 1$ are chosen based on Lemma 5, which can be properly generalized to any order-m Markov model. For instance, we can say that $p(x_{i:k}; x_{k+1:k+j})$ and $p(x_{i:k}; x_{k+1:k+m})$ have the same column space for any $j \geq m$. Based on this generalization, the choice of the sketch functions for general $m \in \mathbb{N}$ is straightforward: they are chosen such that

$$\bar{\Phi}_k = \mathcal{M}_{1:(k+m)\wedge d}p,$$
$$\widetilde{\Phi}_k = \mathcal{M}_{k-1:(k+m)\wedge d}p.$$

In particular, using such $\widetilde{\Phi}_k$'s as the input to TRIMMING and subsequently SYSTEMFORMING, we obtain an algorithm for a discrete order-m Markov density.

6. Numerical experiments

In this section, we illustrate the performance of our algorithm with concrete examples. More specifically, given i.i.d. samples of some ground truth density $p^\star$, we construct an empirical density $\hat{p}$ and apply TT-RS (or TT-RS-C) to it to obtain cores $G_1, \dots, G_d$ such that $p^\star \approx G_1 \circ \cdots \circ G_d =: q$.

6.1. Ginzburg-Landau distribution

We consider the following probability density defined on $[a, b]^d$:

$$p_{GL}(x_1, \dots, x_d) \propto \exp\left(-\beta \sum_{k=0}^{d} \left(\frac{\lambda}{2}\left(\frac{x_k - x_{k+1}}{h}\right)^2 + \frac{1}{4\lambda}(x_k^2 - 1)^2\right)\right),$$

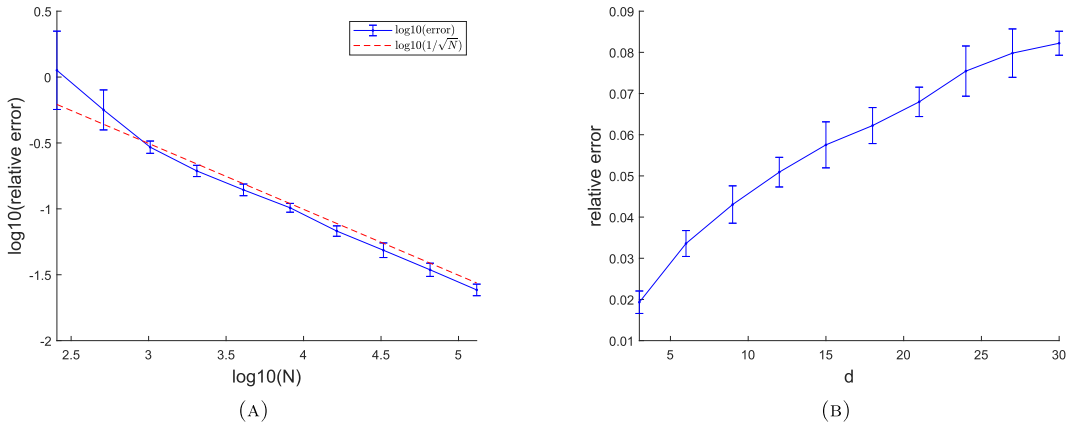
where $x_0 = x_{d+1} = 0$. This is the Boltzmann distribution of a Ginzburg-Landau potential, which is classically used to model phase transitions in physics and also more recently as a test case in generative modeling [10]. Throughout the section, we fix $[a, b] = [-4, 4]$ and $\beta = \lambda = h = 1$.

First, we consider a discretized version of p. To discretize p, we choose n uniform grid points of $[a, b]$, that is, $\mathcal{Z} = \left\{a + \frac{i}{n-1}(b - a)\right\}_{i=0}^{n-1}$, and define a discretized density $p_D \colon [n]^d \to \mathbb{R}$ as

$$p_D := [p_{GL}(x_1, \dots, x_d)]_{(x_1,\dots,x_d)\in\mathcal{Z}^d}.$$

Hence, p_D is essentially a multi-dimensional array of size n^d. Notice that p_{GL} is a Markov model, hence so is p_D. We obtain N i.i.d. samples from p_D using a Gibbs sampler and construct an empirical density based on these samples, which form the empirical measure $\hat{p}_D$. We apply TT-RS with sketches (14) and (16) to $\hat{p}_D$ and let $q_D := G_1 \circ \cdots \circ G_d$ be the contraction of the cores obtained by the algorithm. We compute the following relative l^2 error:

$$\frac{\|p_D - q_D\|_2}{\|p_D\|_2},$$



(A) (B)

Fig. 3. Relative l_2 errors for the discretized case. In (A), we fix $d = 8$ and change the sample size $N \in \{2^8, 2^9, \dots, 2^{17}\}$. In (B), we fix the sample size $N = 50000$ and change $d \in \{3, 6, \dots, 27, 30\}$. In both cases, we use the fixed number of grid points $n = 9$, and TT-RS (Algorithm 1) is applied with $r_1 = \cdots = r_d = 3$. Each error bar is centered at the average of 20 realizations, with the standard deviation as its vertical length.

where $\|f\|_2^2 := \sum_{x_1=1}^n \cdots \sum_{x_d=1}^n f(x_1, \dots, x_d)^2$ for any $f\colon [n]^d \to \mathbb{R}$. We see in Fig. 3(A) that the error decreases with rate $O\left(\frac{1}{\sqrt{N}}\right)$ as sample size N increases when we fix d. Furthermore, when we fix N and let d grow, we see a linear growth in the error (Fig. 3(B)).

Next, we repeat the same procedure with a continuous density p_{GL}. Now we obtain N i.i.d. samples from p_{GL} using the Metropolis-Hastings algorithm and construct an empirical density $\hat{p}$ based on them. Then, we apply Algorithm 13 to $\hat{p}$, where we choose the basis functions $\phi_1, \dots, \phi_M$ as Fourier basis functions on $[a, b]$. Recall that contraction of the resulting cores gives a function q such that

$$q(x_1, \dots, x_d) = \sum_{j_1=1}^{M} \cdots \sum_{j_d=1}^{M} \left(\sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} g_1(j_1, \alpha_1) \cdots g_d(\alpha_{d-1}, j_d) \right) \phi_{j_1}(x_1) \cdots \phi_{j_d}(x_d).$$

Then, we compute the relative L^2 error:

$$\mathrm{err}_t = \frac{\|p_{GL} - q\|_2}{\|p_{GL}\|_2},$$

where $\|f\|_2^2 = \int_{[a,b]^d} f(x_1, \dots, x_d)^2 \, dx_1 \cdots dx_d$ for any f defined on $[a, b]^d$. Since q is an element of the function space $\Pi_M := \{\phi_{j_1} \otimes \cdots \otimes \phi_{j_d} : j_1, \dots, j_d \in [M]\}$, we may decompose this L^2 error as follows using the orthogonality:

$$\mathrm{err}_t^2 = \underbrace{\left(\frac{\|p_{GL} - p_A\|_2}{\|p_{GL}\|_2} \right)^2}_{=:\mathrm{err}_a^2} + \underbrace{\left(\frac{\|p_A - q\|_2}{\|p_{GL}\|_2} \right)^2}_{=:\mathrm{err}_e^2},$$

where

$$p_A(x_1, \dots, x_d) = \sum_{j_1=1}^{M} \cdots \sum_{j_d=1}^{M} \nu(j_1, \dots, j_d) \phi_{j_1}(x_1) \cdots \phi_{j_d}(x_d),$$

$$\nu(j_1, \dots, j_d) = \int p_A(x_1, \dots, x_d) \phi_{j_1}(x_1) \cdots \phi_{j_d}(x_d) \, dx_1 \cdots dx_d.$$

Table 1
L^2 errors for Ginzburg-Landau Gibbs measure in the continuous case. Sample size N is fixed to 10^6, and Algorithm 13 is applied with $r_1 = \cdots = r_d = 3$. Each err_e is averaged over 20 realizations, and the number in the parentheses denotes the standard deviation.

	$d = 5$			$d = 10$			$d = 15$		
M	err_a	err_e	err_t	err_a	err_e	err_t	err_a	err_e	err_t
7	0.2693	0.0202 (0.0023)	0.2701	0.4144	0.0392 (0.0032)	0.4163	0.5104	0.0582 (0.0041)	0.5138
9	0.1617	0.0334 (0.0018)	0.1651	0.2511	0.0621 (0.0027)	0.2587	0.3142	0.0908 (0.0041)	0.3270
11	0.0867	0.0411 (0.0016)	0.0960	0.1365	0.0754 (0.0024)	0.1559	0.1722	0.1100 (0.0039)	0.2044
13	0.0400	0.0433 (0.0015)	0.0589	0.0655	0.0802 (0.0023)	0.1036	0.0837	0.1186 (0.0039)	0.1451
15	0.0201	0.0446 (0.0015)	0.0489	0.0330	0.0833 (0.0023)	0.0896	0.0421	0.1246 (0.0038)	0.1315

In other words, p_A is the approximation of p within the space Π_M spanned by the product basis, thus err_a represents an approximation error. Accordingly, we can think of err_e as an estimation error, where the resulting $g_1, \ldots, g_d$ can be thought of as approximate cores of ν. All the integrals above are approximated using the Gauss-Legendre quadrature rule with 50 nodes.

The resulting L^2 errors are shown in Table 1. As M increases, the approximation error err_a decreases quickly to 0. On the other hand, larger M leads to a larger estimation error err_e as one needs to estimate a larger size of coefficient tensor ν.

6.2. Ising-type model

For our next example we consider the following slight generalization of the one-dimensional Ising model. Define $p\colon \{\pm 1\}^d \to \mathbb{R}$ by

$$p_I(x_1, \ldots, x_d) \propto \exp\left(-\beta \sum_{i,j=1}^{d} J_{ij} x_i x_j\right), \tag{19}$$

where $\beta > 0$ and the interaction J_{ij} is given by

$$J_{ij} = \begin{cases} -(1 + |i-j|)^{-1} & |i-j| \le 2 \\ 0 & \text{otherwise.} \end{cases}$$

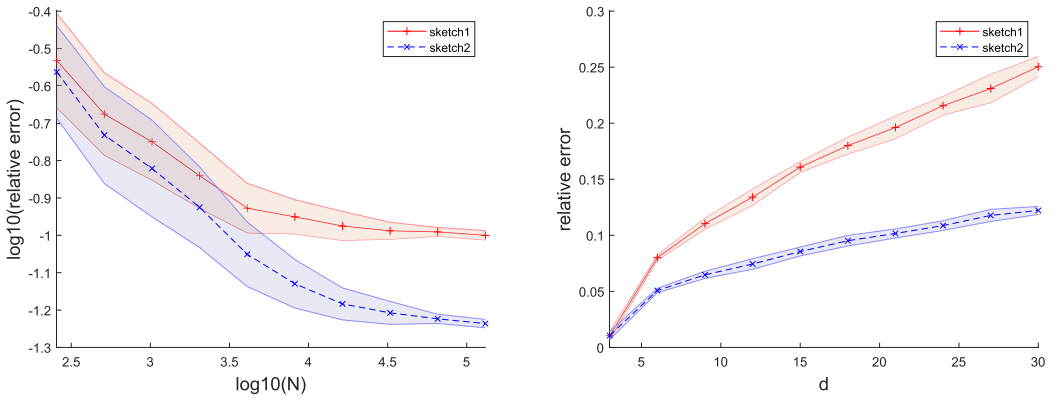
From this, we can easily see that p_I is an order-2 Markov model. For such a model we can apply TT-RS with the sketch functions described in Section 5.4.

As in the previous section, we obtain N i.i.d. samples from p_I using a Gibbs sampler and construct an empirical density based on them, $\hat{p}$. Then, we apply Algorithm TT-RS, with the sketch functions in Section 5.1 and with the modifications outlined in Section 5.4, to obtain the contraction of the resulting cores q_1 and q_2, respectively. Then, we compare the two relative l^2 errors:

$$\mathrm{err}_1 = \frac{\|p_I - q_1\|_2}{\|p_I\|_2} \quad \text{and} \quad \mathrm{err}_2 = \frac{\|p_I - q_2\|_2}{\|p_I\|_2}.$$

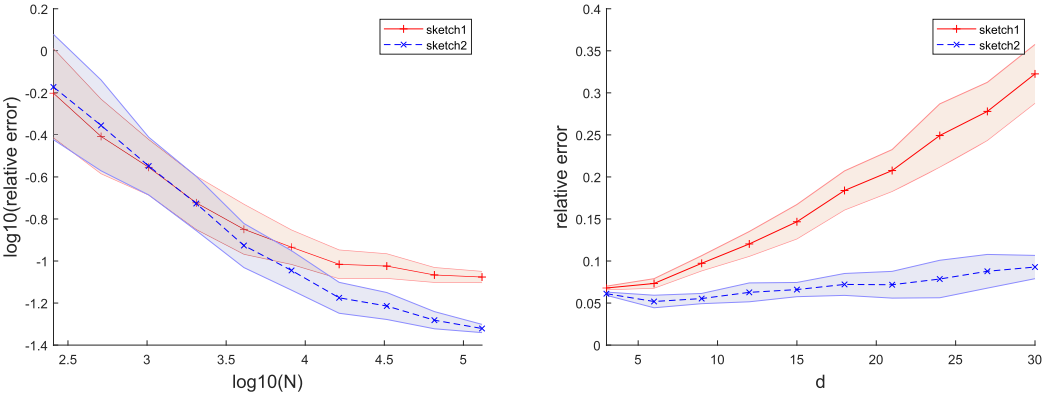
The errors are plotted in Fig. 4, in which the dashed curves denote the result err_1 of TT-RS with sketches as in Section 5.1 and the solid curves correspond to err_2 from TT-RS with the sketches as in Section 5.4. Clearly, as expected, the error is smaller when using the sketches from Section 5.4.

Lastly, we repeat the same procedure for p_I where $x_k \in \{-2, -1, 0, 1, 2\}$ for $k = 1, \ldots, d$ in (19). The results are shown in Fig. 5, which demonstrates that TT-RS with appropriate sketching yields small error in the case of higher-order Markov distributions.



(A) (B)

Fig. 4. Relative l_2 errors for the order-2 Ising model. In (A), we fix $d = 8$ and change the sample size $N \in \{2^8, 2^9, \ldots, 2^{17}\}$. In (B), we fix the sample size $N = 50000$ and change $d \in \{3, 6, \ldots, 27, 30\}$. In both cases, we use $\beta = 0.4$, and TT-RS (Algorithm 1) is applied with $(r_1, \ldots, r_d) = (2, 3, \ldots, 3, 2)$. Errors are shown as shaded regions, where both solid and dashed curves connect the averages of errors from 20 realizations, with the standard deviation as the vertical width.



(A) (B)

Fig. 5. Relative l_2 errors for the order-2 Ising model on $\{-2, -1, 0, 1, 2\}^d$. In (A), we fix $d = 8$ and change the sample size $N \in \{2^8, 2^9, \ldots, 2^{17}\}$. In (B), we fix the sample size $N = 50000$ and change $d \in \{3, 6, \ldots, 27, 30\}$. In both cases, we use $\beta = 0.2$, and TT-RS (Algorithm 1) is applied with $(r_1, \ldots, r_d) = (2, 3, \ldots, 3, 2)$. Each error bar is centered at the average of 20 realizations, with the standard deviation as its vertical length.

7. Conclusion

We have described an algorithm TT-RS which obtains a tensor train representation of a probability density from a collection of its samples. This is done by formulating a sequence of equations, one for each core, which can be solved independently. Additionally, in order to reduce the variance in the coefficient matrices of these equations (which are constructed from the empirical distribution) sketching is required. For Markov (and higher-order Markov) models we give explicit constructions of suitable sketches and provide guarantees on the accuracy of the resulting algorithm.

Lastly, we briefly mention several possible extensions for future research. First, we can apply TT-RS to more complicated models such as hidden Markov models. The ideas that we discussed based on (higher-order) Markov models can be generalized to various models by specifying concrete sketch functions for such models. More generally, future research could focus on adapting TT-RS to tree tensor networks, aiming at generalizing TT-RS to distributions with more general graphical structure. By designing sketch functions for a broader class of models, one can bring TT-RS closer to a wide range of applications and we leave this as future work.

Data availability

Data will be made available on request.

Acknowledgments

The work of YH and YK was partially supported by the National Science Foundation under Award No. DMS-211563 and the Department of Energy under Award No. DE-SC0022232. The work of ML was partially supported by the National Science Foundation under Award No. 1903031. The Flatiron Institute is a division of the Simons Foundation.

Appendix A. Validity of solving CDEs

In this section, we give the proof of Proposition 2.

Proof of Proposition 2. For $k = 2, \dots, d-1$, consider the k-th equation in (4):

$$\sum_{\alpha_{k-1}=1}^{r_{k-1}} \Phi_{k-1}(x_{1:k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) = \Phi_k(x_{1:k-1}; x_k, \alpha_k). \tag{20}$$

By definition, $\Phi_{k-1}(x_{1:k-1}; \alpha_{k-1})$ is the left factor in an exact low-rank factorization of p, so Φ_{k-1} has full column rank and the uniqueness of solutions is guaranteed. To prove a solution also exists, we need to show that columns of $\Phi_k(x_{1:k-1}; x_k, \alpha_k)$ are contained within the column space of $\Phi_{k-1}(x_{1:k-1}; \alpha_{k-1})$.

By the definition of Φ_{k-1} and Φ_k, we know there exists $\Psi_k \colon [r_{k-1}] \times [n_k] \times \cdots \times [n_d] \to \mathbb{R}$ and $\Psi_{k+1} \colon [r_k] \times [n_{k+1}] \times \cdots \times [n_d] \to \mathbb{R}$ such that

$$\begin{aligned} p(x_{1:k-1}; x_{k:d}) &= \sum_{\alpha_{k-1}=1}^{r_{k-1}} \Phi_{k-1}(x_{1:k-1}; \alpha_{k-1}) \Psi_k(\alpha_{k-1}; x_{k:d}), \\ p(x_{1:k}; x_{k+1:d}) &= \sum_{\alpha_k=1}^{r_k} \Phi_k(x_{1:k}; \alpha_k) \Psi_{k+1}(\alpha_k; x_{k+1:d}). \end{aligned} \tag{21}$$

Note that these are rank-r_{k-1} and rank-r_k decomposition of the $(k-1)$-th and k-th unfolding matrices, respectively. Defining $t_{k+1} \colon [n_{k+1}] \times \cdots \times [n_d] \times [r_k] \to \mathbb{R}$ so that $t_{k+1}(x_{k+1:d}; \alpha_k)$ is the pseudoinverse of $\Psi_{k+1}(\alpha_k; x_{k+1:d})$, we obtain

$$\Phi_k(x_{1:k}; \alpha_k) = \sum_{x_{k+1}=1}^{n_{k+1}} \cdots \sum_{x_d=1}^{n_d} p(x_{1:k}; x_{k+1:d}) t_{k+1}(x_{k+1:d}; \alpha_k).$$

Then, one can easily verify that the k-th equation (20) holds if we let

$$G_k(\alpha_{k-1}, x_k, \alpha_k) = \sum_{x_{k+1}=1}^{n_{k+1}} \cdots \sum_{x_d=1}^{n_d} \Psi_k(\alpha_{k-1}, x_k, x_{k+1:d}) t_{k+1}(x_{k+1:d}, \alpha_k)$$

along with (21). Thus, we have not only proved the existence of solutions to the k-th equation, but also obtained the exact form of the solution in terms of Ψ_k and t_{k+1}.

Similarly, we can show that the equation in (20) for G_d is well-defined. By construction, it then follows that (5) holds. □

Appendix B. Non-recursive TT-RS: TT-Sketch (TT-S)

B.1. Role of recursive left sketches and possibility of non-recursive sketches

In this subsection, we discuss the importance of forming the recursive right sketches $S_1, \dots, S_{d-1}$ from $s_1, \dots, s_d$, noting that for $T_2, \dots, T_d$ there is no such need. The requirement of "recursiveness" in the construction of the S_k's is a consequence of the TRIMMING step, which introduces an arbitrary projection matrix in the factorization of the k-th unfolding of p. To see this, consider first the case without TRIMMING, i.e., using sketches T_{k+1} with $l_k = r_k$. Then one can use $\bar{\Phi}_k$ (defined in Algorithm 2) in the CDEs (4), i.e. solve

$$\sum_{\alpha_{k-1}=1}^{r_{k-1}} \bar{\Phi}_{k-1}(x_{1:k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) = \bar{\Phi}_k(x_{1:k-1}; x_k, \alpha_k) \tag{22}$$

if each $\bar{\Phi}_k$ has rank r_k. To reduce the system size further one could simply apply *arbitrary* left sketches as

$$\begin{aligned} &\sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}; x_{1:k-1}) \sum_{\alpha_{k-1}=1}^{r_{k-1}} \bar{\Phi}_{k-1}(x_{1:k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) \\ &= \sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}; x_{1:k-1}) \bar{\Phi}_k(x_{1:k-1}; x_k, \alpha_k) \end{aligned} \tag{23}$$

so long as the reduced CDEs remain well-posed. In this case, one could set $B_k = S_{k-1}\bar{\Phi}_k$ and $A_{k-1} = S_{k-1}\bar{\Phi}_{k-1}$.

Unfortunately, a complication arises when we use sketches T_{k+1} with $l_k > r_k$. In this case we cannot simply solve (22) or (23) as it gives TT with excessively large rank. We then need to apply a suitable further projection $q_k \in \mathbb{R}^{l_{k-1} \times r_{k-1}}, q_{k+1} \in \mathbb{R}^{l_k \times r_k}$ in (22) and (23)

$$\bar{\Phi}_{k-1} \to \bar{\Phi}_{k-1} q_k, \quad \bar{\Phi}_k \to \bar{\Phi}_k q_{k+1} \tag{24}$$

treating $\bar{\Phi}_{k-1}$ and $\bar{\Phi}_k$ as matrices of size $n_1 \cdots n_{k-1} \times l_{k-1}$ and $n_1 \cdots n_k \times l_k$, respectively. This is the idea behind TRIMMING. However, rather than explicitly applying the projection q_k, TRIMMING performs the projections implicitly, i.e., it directly gives

$$B_k(\beta_{k-1}, x_k, \alpha_k) = \sum_{x_1, \dots, x_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1}) \sum_{\gamma_k} \bar{\Phi}_k(x_{1:k-1}, x_k, \gamma_k) q_{k+1}(\gamma_k, \alpha_k) \tag{25}$$

via an SVD without obtaining the q_k's. This presents a complication: in order to solve (23), one needs to form

$$A_{k-1}(\beta_{k-1}, \alpha_{k-1}) = \sum_{x_1, \dots, x_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1}) \sum_{\gamma_{k-1}} \bar{\Phi}_{k-1}(x_{1:k-1}, \gamma_{k-1}) q_k(\gamma_{k-1}, \alpha_{k-1}), \tag{26}$$

but all we have access to is B_{k-1} which contains q_k implicitly (note that A_{k-1} is not B_{k-1}). It is unclear how to obtain A_{k-1} without knowing q_k explicitly.

There are two remedies for this. The first one is recursive left sketching, and the second one is to obtain projections (the q_k's) directly. The second remedy is more complicated than the first, though it does not require recursive sketching. In this paper, we have focused primarily on the recursive left sketching approach, which allows us to obtain A_k's directly from B_k's. In the next subsection, we provide details of the second remedy in the following subsection.

B.2. Non-recursive sketches

Suppose now $S_1, \dots, S_{d-1}$ in Algorithm 2 are arbitrary sketches that are *non-recursive*, meaning that they are not in the form of

$$S_{k-1}(\beta_{k-1}, x_{1:k-1}) = \sum_{\beta_{k-2}} s_{k-1}(\beta_{k-1}, x_{k-1}, \beta_{k-2}) S_{k-2}(\beta_{k-2}, x_{1:k-2}). \tag{27}$$

Evidently, TRIMMING gives the following expression for B_k in terms of the sketched unfolding matrix (in Algorithm 3) and some "gauge" q_{k+1}

$$B_k(\beta_{k-1}, x_k; \alpha_k) = \sum_{\gamma_k} \widetilde{\Phi}_k(\beta_{k-1}, x_k; \gamma_k) q_{k+1}(\gamma_k, \alpha_k) \tag{28}$$

where

$$q_{k+1} = V_k \Sigma_k^{-1} \tag{29}$$

and

$$\widetilde{\Phi}_k \approx U_k \Sigma_k V_k^\top, \quad U_k \in \mathbb{R}^{m_{k-1} n_k \times r_k},\ \Sigma_k \in \mathbb{R}^{r_k \times r_k},\ V_k \in \mathbb{R}^{l_k \times r_k} \tag{30}$$

being the best rank-r_k approximation of $\widetilde{\Phi}_k \in \mathbb{R}^{m_{k-1} n_k \times l_k}$ (defined in Algorithm 3) obtained via the SVD. Now, after obtaining the q_k's in this manner, we can use them to construct the A_k's in (26). In this case, we do not need to use B_k's to obtain A_k's, as in the case when using recursive sketches.

In what follows, we summarize this approach in TT-S (Algorithm 5) which removes the necessity of recursive sketching. The main difference between TT-S and TT-RS is that TT-S keeps track of the projection matrices $q_2, \dots q_d$ in (26) obtained via Algorithm 7 when performing TRIMMING-TT-S and uses them in Algorithm 8. In this way, one eliminates the need for obtaining the A_k's via the B_k's from recursive sketching.

Algorithm 5 TT-S for a discrete function p.

Require: $p\colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$ and target ranks $r_1, \dots, r_{d-1}$.
Require: $T_k\colon [n_k] \times \cdots \times [n_d] \times [l_{k-1}] \to \mathbb{R}$ with $l_{k-1} \geq r_{k-1}$ for $k = 2, \dots, d$.
Require: $S_k\colon [m_k] \times [n_1] \times \cdots \times [n_k] \to \mathbb{R}$ for $k = 1, \dots, d-1$.
1: $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d, \bar{\Phi}_1, \dots, \bar{\Phi}_{d-1} \leftarrow$ SKETCHING-TT-S$(p, T_2, \dots, T_d, S_1, \dots, S_{d-1})$.
2: $B_1, \dots, B_d, q_2, \dots, q_d \leftarrow$ TRIMMING-TT-S$(\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d, r_1, \dots, r_{d-1})$.
3: $A_1, \dots, A_{d-1} \leftarrow$ SYSTEMFORMING-TT-S$(\bar{\Phi}_1, \dots, \bar{\Phi}_{d-1}, q_2, \dots, q_d, S_1, \dots, S_{d-1})$.
4: Solve the following d matrix equations via least-squares for the variables $G_1\colon [n_1] \times [r_1] \to \mathbb{R}$, $G_k\colon [r_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $G_d\colon [r_{d-1}] \times [n_d] \to \mathbb{R}$:

$$\begin{aligned} G_1 &= B_1, \\ \sum_{\alpha_{k-1}=1}^{r_{k-1}} A_{k-1}(\beta_{k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) &= B_k(\beta_{k-1}; x_k, \alpha_k) \quad k = 2, \dots, d-1, \\ \sum_{\alpha_{d-1}=1}^{r_{d-1}} A_{d-1}(\beta_{d-1}; \alpha_{d-1}) G_d(\alpha_{d-1}; x_d) &= B_d(\beta_{d-1}; x_d). \end{aligned} \tag{31}$$

5: **return** $G_1, \dots, G_d$.

Algorithm 6 SKETCHING-TT-S.

Require: p, $T_2, \dots, T_d$, and $S_1, \dots, S_{d-1}$ as given in Algorithm 5.
for $k = 1$ to $d-1$ **do**
Right sketching: define $\bar{\Phi}_k \colon [n_1] \times \cdots \times [n_k] \times [l_k] \to \mathbb{R}$ as

$$\bar{\Phi}_k(x_{1:k}, \gamma_k) = \sum_{x_{k+1}=1}^{n_{k+1}} \cdots \sum_{x_d=1}^{n_d} p(x_{1:k}, x_{k+1:d}) T_{k+1}(x_{k+1:d}, \gamma_k).$$

if $k > 1$ **then**
Left sketching: define $\widetilde{\Phi}_k \colon [m_{k-1}] \times [n_k] \times [l_k] \to \mathbb{R}$ as

$$\widetilde{\Phi}_k(\beta_{k-1}, x_k, \gamma_k) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_{k-1}=1}^{n_{k-1}} S_{k-1}(\beta_{k-1}, x_{1:k-1}) \bar{\Phi}_k(x_{1:k-1}, x_k, \gamma_k).$$

else
Define

$$\widetilde{\Phi}_1(x_1, \gamma_1) = \bar{\Phi}_1(x_1, \gamma_1).$$

end if
end for
Left sketching: define $\widetilde{\Phi}_d \colon [m_{d-1}] \times [n_d] \to \mathbb{R}$ as

$$\widetilde{\Phi}_d(\beta_{d-1}, x_d) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_{d-1}=1}^{n_{d-1}} S_{d-1}(\beta_{d-1}, x_{1:d-1}) p(x_{1:d-1}, x_d).$$

return $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d, \bar{\Phi}_1, \dots, \bar{\Phi}_{d-1}$.

Algorithm 7 TRIMMING-TT-S.

Require: $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$ from Algorithm 6.
Require: Target ranks $r_1, \dots, r_{d-1}$ as given in Algorithm 5.
for $k = 1$ to $d-1$ **do**
if $k = 1$ **then**
Let $U_1 \Sigma_1 V_1^\top$, where $U_1 \in \mathbb{R}^{n_1 \times r_1}, V_1 \in \mathbb{R}^{l_1 \times r_1}, \Sigma_1 \in \mathbb{R}^{r_1 \times r_1}$, be the best rank-$r_1$ approximation to the matrix $\widetilde{\Phi}_1(x_1; \alpha_1)$ via SVD. Define $B_1 \colon [n_1] \times [r_1] \to \mathbb{R}$ where $B_1(x_1; \alpha_1) = U_1(x_1; \alpha_1)$. Furthermore, let $q_2 = V_1 \Sigma_1^{-1}$.
else
Let $U_k \Sigma_k V_k^\top$, where $U_k \in \mathbb{R}^{m_{k-1} n_k \times r_k}, V_k \in \mathbb{R}^{l_k \times r_k}, \Sigma_k \in \mathbb{R}^{r_k \times r_k}$, be the best rank-$r_k$ approximation to the matrix $\widetilde{\Phi}_k(\beta_{k-1}, x_k; \gamma_k)$ via SVD. Define $B_k \colon [m_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ where $B_k(\beta_{k-1}, x_k; \alpha_k) = U_k(\beta_{k-1}, x_k; \alpha_k)$. Furthermore, let $q_{k+1} = V_k \Sigma_k^{-1}$.
end if
end for
Let $B_d(\beta_{d-1}, x_d) = \widetilde{\Phi}_d(\beta_{d-1}, x_d)$.
return $B_1, \dots, B_d, q_2, \dots, q_d$.

Algorithm 8 SYSTEMFORMING-TT-S.

Require: $\bar{\Phi}_1, \dots, \bar{\Phi}_{d-1}$ from Algorithm 6
Require: $q_2 \dots, q_d$ from Algorithm 7.
Require: $S_1, \dots, S_{d-1}$ as given in Algorithm 5.
for $k = 1$ to $d-1$ **do**
Compute $A_k \colon [m_k] \times [r_k] \to \mathbb{R}$:

$$A_k(\beta_k, \alpha_k) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_k=1}^{n_k} S_k(\beta_k, x_{1:k}) \sum_{\gamma_k=1}^{l_k} \bar{\Phi}_k(x_{1:k}, \gamma_k) q_{k+1}(\gamma_k, \alpha_k).$$

end for
return $A_1, \dots, A_{d-1}$.

Appendix C. Continuous TT-RS

In this section, we consider a general function $p\colon X_1 \times \cdots \times X_d \to \mathbb{R}$, where $X_1, \dots, X_d \subset \mathbb{R}$. It turns out that everything presented in previous sections is still valid if we replace every discrete quantity with its continuous counterpart; concretely, we replace $[n_k]$, $[m_k]$, and $[l_k]$ with X_k, $\mathcal{B}_k$, and $\mathcal{C}_k$, respectively, where $\mathcal{B}_k$ and $\mathcal{C}_k$ are appropriate domains that can be chosen by model assumptions. Accordingly, we also replace all the summation over these sets with appropriate integration; for instance, replace $\sum_{x_k=1}^{n_k}$ and $\sum_{\beta_k=1}^{m_k}$ with $\int_{X_k} dx_k$ and $\int_{\mathcal{B}_k} d\beta_k$, respectively. As a result, we obtain Algorithms 9, 10, 11, and 12 as continuous counterparts of Algorithms 1, 2, 3, and 4.

Algorithm 9 TT-RS-C for a continuous function p.

Require: $p\colon X_1 \times \cdots \times X_d \to \mathbb{R}$ and target ranks $r_1, \dots, r_{d-1}$.
Require: $T_k\colon X_k \times \cdots \times X_d \times \mathcal{C}_{k-1} \to \mathbb{R}$ for $k = 2, \dots, d$.
Require: $s_1\colon \mathcal{B}_1 \times X_1 \to \mathbb{R}$ and $s_k\colon \mathcal{B}_k \times X_k \times \mathcal{B}_{k-1} \to \mathbb{R}$ for $k = 2, \dots, d-1$.
1: $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d \leftarrow$ SKETCHING-C$(p, T_2, \dots, T_d, s_1, \dots, s_{d-1})$.
2: $B_1, \dots, B_d \leftarrow$ TRIMMING-C$(\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d, r_1, \dots, r_{d-1})$.
3: $A_1, \dots, A_{d-1} \leftarrow$ SYSTEMFORMING-C$(B_1, \dots, B_{d-1}, s_1, \dots, s_{d-1})$.
4: Solve the following d matrix equations via least-squares for the variables $G_1\colon X_1 \times [r_1] \to \mathbb{R}$, $G_k\colon [r_{k-1}] \times X_k \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $G_d\colon [r_{d-1}] \times X_d \to \mathbb{R}$.

$$\begin{aligned} G_1 &= B_1, \\ \sum_{\alpha_{k-1}=1}^{r_{k-1}} A_{k-1}(\beta_{k-1}; \alpha_{k-1}) G_k(\alpha_{k-1}; x_k, \alpha_k) &= B_k(\beta_{k-1}; x_k, \alpha_k) \quad k = 2, \dots, d-1, \\ \sum_{\alpha_{d-1}=1}^{r_{d-1}} A_{d-1}(\beta_{d-1}; \alpha_{d-1}) G_d(\alpha_{d-1}; x_d) &= B_d(\beta_{d-1}; x_d). \end{aligned} \tag{32}$$

5: **return** $G_1, \dots, G_d$.

First, note that the main algorithm for the continuous case, TT-RS-C (Algorithm 9), has equations (32) which are exactly the same as (7) of Algorithm 1. Now, (32) are infinite-dimensional matrix equations, that is, coefficients and cores are functions. Also, the sketching algorithm for a continuous density (Algorithm 10), which we call SKETCHING-C, is simply a modification of SKETCHING by replacing all the summations with integrals properly. We modify TRIMMING similarly to obtain its continuous counterpart TRIMMING-C. In this case, TRIMMING-C should be done by applying functional SVD [28,35] to $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_{d-1}$ to obtain $B_1, \dots, B_{d-1}$, respectively. We demonstrate how such a functional SVD works in the next subsection with a concrete example.

C.1. Applying TT-RS-C to the Markov case

In this subsection, we assume p is a continuous Markov model, that is, p is a continuous density and satisfies (12). For simplicity, we assume $X_1 = \cdots = X_d = [a, b]$ and $(\phi_n)_{n\in\mathbb{N}}$ be a countable orthonormal basis of $L^2([a, b])$ such that ϕ_1 is a constant function, say $\phi_1(x) \equiv c$. Due to orthogonality,

$$\int_a^b \phi_n(x)\, dx = 0$$

for all $n \geq 2$. Suppose each marginal density of p is well approximated using the first M basis functions $\phi_1, \dots, \phi_M$. Based on Lemma 5, we now show that we can choose concrete sketch functions $T_2, \dots, T_d$ and $s_1, \dots, s_{d-1}$ so that Algorithm 9 exactly recovers the cores of p, when provided with $\hat{p} = p$.

Algorithm 10 SKETCHING-C.

Require: p, $T_2, \dots, T_d$, and $s_1, \dots, s_{d-1}$ as given in Algorithm 9.
for $k = 1$ to $d-1$ **do**
Right sketching: define $\bar{\Phi}_k \colon X_1 \times \cdots \times X_k \times \mathcal{C}_k \to \mathbb{R}$ as

$$\bar{\Phi}_k(x_{1:k}, \gamma_k) = \int p(x_{1:k}, x_{k+1:d}) T_{k+1}(x_{k+1:d}, \gamma_k)\, dx_{k+1} \cdots dx_d.$$

if $k > 1$ **then**
Left sketching: define $\widetilde{\Phi}_k \colon \mathcal{B}_{k-1} \times X_k \times \mathcal{C}_k \to \mathbb{R}$ as

$$\widetilde{\Phi}_k(\beta_{k-1}, x_k, \gamma_k) = \int S_{k-1}(\beta_{k-1}, x_{1:k-1}) \bar{\Phi}_k(x_{1:k-1}, x_k, \gamma_k)\, dx_1 \cdots dx_{k-1}.$$

Compute $S_k \colon \mathcal{B}_k \times X_1 \times \cdots \times X_k \to \mathbb{R}$ for the next iteration:

$$S_k(\beta_k, x_{1:k}) = \int s_k(\beta_k, x_k, \beta_{k-1}) S_{k-1}(\beta_{k-1}, x_{1:k-1})\, d\beta_{k-1}.$$

else
Define

$$\widetilde{\Phi}_1(x_1, \gamma_1) = \bar{\Phi}_1(x_1, \gamma_1).$$

Define sketch function

$$S_1(\beta_1, x_1) = s_1(\beta_1, x_1).$$

end if
end for
Left sketching: define $\widetilde{\Phi}_d \colon \mathcal{B}_{d-1} \times X_d \to \mathbb{R}$ as

$$\widetilde{\Phi}_d(\beta_{d-1}, x_d) = \int S_{d-1}(\beta_{d-1}, x_{1:d-1}) p(x_{1:d-1}, x_d)\, dx_1 \cdots dx_{d-1}.$$

return $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$.

Algorithm 11 TRIMMING-C.

Require: $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$ from Algorithm 10.
Require: Target ranks $r_1, \dots, r_{d-1}$ as given in Algorithm 9.
for $k = 1$ to $d-1$ **do**
if $k = 1$ **then**
Compute the first r_1 left singular vectors of $\widetilde{\Phi}_1(x_1; \gamma_1)$ and define $B_1 \colon X_1 \times [r_1] \to \mathbb{R}$ so that these singular vectors are the columns of $B_1(x_1; \alpha_1)$.
else
Compute the first r_k left singular vectors of $\widetilde{\Phi}_k(\beta_{k-1}, x_k; \gamma_k)$ and define $B_k \colon \mathcal{B}_{k-1} \times X_k \times [r_k] \to \mathbb{R}$ so that these singular vectors are the columns of $B_k(\beta_{k-1}, x_k; \alpha_k)$.
end if
end for
Let $B_d(\beta_{d-1}, x_d) = \widetilde{\Phi}_d(\beta_{d-1}, x_d)$.
return $B_1, \dots, B_d$.

First, let $\mathcal{B}_k = \mathcal{C}_k = [M]$ for $k = 1, \dots, d-1$, where $r_1, \dots, r_{d-1} \leq M$. Then, we define $T_{k+1} \colon X_{k+1} \times \cdots \times X_d \times \mathcal{C}_k \to \mathbb{R}$ as

$$T_{k+1}(x_{k+1:d}, \gamma_k) = \phi_{\gamma_k}(x_{k+1})$$

which gives

$$\bar{\Phi}_k(x_{1:k}, \gamma_k) = \int p(x_{1:k}, x_{k+1}) \phi_{\gamma_k}(x_{k+1})\, dx_{k+1}.$$

Algorithm 12 SystemForming-C.

Require: $B_1, \dots, B_{d-1}$ from Algorithm 11.
Require: $s_1, \dots, s_{d-1}$ as given in Algorithm 9.
for $k = 1$ to $d-1$ **do**
if $k = 1$ **then**
Compute $A_1 \colon \mathcal{B}_1 \times [r_1] \to \mathbb{R}$:

$$A_1(\beta_1, \alpha_1) = \int s_1(\beta_1, x_1) B_1(x_1, \alpha_1)\, dx_1.$$

else
Compute $A_k \colon \mathcal{B}_k \times [r_k] \to \mathbb{R}$:

$$A_k(\beta_k, \alpha_k) = \int s_k(\beta_k, x_k, \beta_{k-1}) B_k(\beta_{k-1}, x_k, \alpha_k)\, dx_k d\beta_{k-1}.$$

end if
end for
return $A_1, \dots, A_{d-1}$.

T_{k+1} marginalizes out $x_{k+2}, \dots, x_d$ as in the discrete case and it replaces the variable x_{k+1} with the index $\gamma_k \in [M]$ based on the fact that the marginal density can be approximated well by the fist M basis functions.

Similarly, define $S_1 \colon \mathcal{B}_1 \times X_1 \to \mathbb{R}$ such that $\mathcal{B}_1 = X_1$

$$s_1(\beta_1, x_1) = \phi_{\beta_1}(x_1), \quad s_k(\beta_k, x_k, \beta_{k-1}) = \phi_{\beta_k}(x_k)\delta(\beta_{k-1} - 1),$$

where δ is the Dirac delta function. Then,

$$S_k(\beta_k, x_{1:k}) = \phi_{\beta_k}(x_k)\phi_1(x_{k-1}) \cdots \phi_1(x_1) = c^{k-1}\phi_{\beta_k}(x_k),$$

thus

$$\begin{aligned}
\widetilde{\Phi}_k(\beta_{k-1}, x_k, \gamma_k) &= \int S_{k-1}(\beta_{k-1}, x_{1:k-1}) \bar{\Phi}_k(x_{1:k-1}, x_k, \gamma_k)\, dx_1 \cdots dx_{k-1} \\
&= \int c^{k-2} \phi_{\beta_{k-1}}(x_{k-1}) p(x_{1:k}, x_{k+1}) \phi_{\gamma_k}(x_{k+1})\, dx_{k+1} dx_1 \cdots dx_{k-1} \\
&= c^{k-2} \int \phi_{\beta_{k-1}}(x_{k-1}) p(x_{k-1}, x_k, x_{k+1}) \phi_{\gamma_k}(x_{k+1})\, dx_{k+1} dx_{k-1}.
\end{aligned}$$

and

$$\begin{aligned}
\widetilde{\Phi}_d(\beta_{d-1}, x_d) &= \int S_{d-1}(\beta_{d-1}, x_{1:d-1}) p(x_{1:d-1}, x_d)\, dx_1 \cdots dx_{d-1} \\
&= c^{d-2} \int \phi_{\beta_{d-1}}(x_{d-1}) p(x_{d-1}, x_d)\, dx_{d-1}.
\end{aligned}$$

In other words, S_{k-1} marginalizes out variables $x_1, \dots, x_{k-2}$ as in the discrete case; furthermore, it replaces the variable x_{k-1} with the index β_{k-1} by integration against basis functions.

Using the results $\widetilde{\Phi}_1, \dots, \widetilde{\Phi}_d$ from Sketching-C, we now explain how to implement Trimming-C via functional SVD. The idea is to use basis expansion with respect to each node $x_k \in X_k$ and then apply SVD. For instance, consider $\widetilde{\Phi}_1(x_1, \gamma_1)$. For large enough $M \in \mathbb{N}$, we have

$$\widetilde{\Phi}_1(x_1, \gamma_1) \approx \sum_{\beta_1=1}^{M} \nu_1(\beta_1, \gamma_1) \phi_{\beta_1}(x_1),$$

where $\nu_1 \colon [M] \times [M] \to \mathbb{R}$ is given as

$$\nu_1(\beta_1, \gamma_1) = \int \widetilde{\Phi}_1(x_1, \gamma_1)\phi_{\beta_1}(x_1)\, dx_1.$$

Now, we can apply SVD to a matrix $\nu_1(\beta_1; \gamma_1)$; compute the first r_k left singular vectors of $\nu_1(\beta_1; \gamma_1)$ and define $\widetilde{B}_1 \colon [M] \times [r_1] \to \mathbb{R}$ so that these singular vectors are the columns of $\widetilde{B}_1(\beta_1; \alpha_1)$. Then, we define $B_1 \colon X_1 \times [r_1] \to \mathbb{R}$ as

$$B_1(x_1, \alpha_1) := \sum_{\beta_1=1}^{M} \widetilde{B}_1(\beta_1, \alpha_1)\phi_{\beta_1}(x_1).$$

Then,

$$A_1(\beta_1, \alpha_1) = \int B_1(x_1, \alpha_1)\phi_{\beta_1}(x_1)dx_1 = \widetilde{B}_1(\beta_1, \alpha_1).$$

Similarly, for $k = 2, \dots, d-1$, we have

$$\widetilde{\Phi}_k(\beta_{k-1}, x_k, \gamma_k) \approx \sum_{j_k=1}^{M} \nu_k(\beta_{k-1}, j_k, \gamma_k)\phi_{j_k}(x_k),$$

where $\nu_k \colon [M] \times [M] \times [M] \to \mathbb{R}$ is given as

$$\nu_k(\beta_{k-1}, j_k, \gamma_k) = \int \widetilde{\Phi}_k(\beta_{k-1}, x_k, \gamma_k)\phi_{j_k}(x_k)\, dx_k.$$

We compute the first r_k left singular vectors of $\nu_k(\beta_{k-1}, j_k; \gamma_k)$ and define $\widetilde{B}_k \colon [M] \times [M] \times [r_k] \to \mathbb{R}$ so that these singular vectors are the columns of $\widetilde{B}_k(\beta_{k-1}, j_k; \alpha_k)$. Then, we define $B_k \colon [M] \times X_k \times [r_k] \to \mathbb{R}$ as

$$B_k(\beta_{k-1}, x_k, \alpha_k) = \sum_{j_k=1}^{M} \widetilde{B}_k(\beta_{k-1}, j_k, \alpha_k)\phi_{j_k}(x_k),$$

which yields $A_k \colon [M] \times [r_k] \to \mathbb{R}$ as

$$\begin{aligned} A_k(\beta_k, \alpha_k) &:= \int \sum_{\beta_{k-1}=1}^{M} s_k(\beta_k, x_k, \beta_{k-1})B_k(\beta_{k-1}, x_k, \alpha_k)\, dx_k \\ &= \int \phi_{\beta_k}(x_k)B_k(1, x_k, \alpha_k)\, dx_k \\ &= \sum_{j_k=1}^{M} \widetilde{B}_k(1, j_k, \alpha_k) \int \phi_{\beta_k}(x_k)\phi_{j_k}(x_k)\, dx_k \\ &= \widetilde{B}_k(1, \beta_k, \alpha_k). \end{aligned}$$

Lastly, we apply basis expansion to B_d as well. Define

$$\widetilde{B}_d(\beta_{d-1}, j_d) = \int B_d(\beta_{d-1}, x_d)\phi_{j_d}(x_d)\, dx_d$$

so that

$$B_d(\beta_{d-1}, x_d) \approx \sum_{j_d=1}^{M} \widetilde{B}_d(\beta_{d-1}, j_d)\phi_{j_d}(x_d).$$

Now, one can easily verify that solving (32) for $G_1, \dots, G_d$ amounts to solving

$$\begin{aligned} g_1 &= \widetilde{B}_1, \\ \sum_{\alpha_{k-1}=1}^{r_{k-1}} A_{k-1}(\beta_{k-1}, \alpha_{k-1}) g_k(\alpha_{k-1}, j_k, \alpha_k) &= \widetilde{B}_k(\beta_{k-1}, j_k, \alpha_k) \quad k = 2, \dots, d-1, \\ \sum_{\alpha_{d-1}=1}^{r_{d-1}} A_{d-1}(\beta_{d-1}, \alpha_{d-1}) g_d(\alpha_{d-1}, j_d) &= \widetilde{B}_d(\beta_{d-1}, j_d) \end{aligned} \tag{33}$$

for the variables $g_1 \colon [M] \times [r_1] \to \mathbb{R}$, $g_k \colon [r_{k-1}] \times [M] \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $g_d \colon [r_{d-1}] \times [M] \to \mathbb{R}$ and letting

$$\begin{aligned} G_1(x_1, \alpha_1) &= \sum_{j_1=1}^{M} g_1(j_1, \alpha_1)\phi_{j_1}(x_1), \\ G_k(\alpha_{k-1}, x_k, \alpha_k) &= \sum_{j_k=1}^{M} g_k(\alpha_{k-1}, j_k, \alpha_k)\phi_{j_k}(x_k) \quad k = 2, \dots, d-1, \\ G_d(\alpha_{d-1}, x_d) &= \sum_{j_d=1}^{M} g_d(\alpha_{d-1}, j_d)\phi_{j_d}(x_d). \end{aligned}$$

In this case, the resulting TT-format is

$$\begin{aligned} &\sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} G_1(x_1, \alpha_1) \cdots G_d(\alpha_{d-1}, x_d) \\ &= \sum_{j_1=1}^{M} \cdots \sum_{j_d=1}^{M} \left(\sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} g_1(j_1, \alpha_1) \cdots g_d(\alpha_{d-1}, j_d) \right) \phi_{j_1}(x_1) \cdots \phi_{j_d}(x_d). \end{aligned}$$

We summarize the case of specializing Algorithm 9 to the case of Markov density in Algorithm 13. We note that one should be able to prove a result similar to Theorem 6 under mild assumptions.

Appendix D. Perturbation results

This section provides perturbation results of Algorithm 1. First, we prove that small perturbation on the coefficients and the right-hand sides of (7) of Algorithm 1 leads to small perturbations of the cores. Using this result we show that Algorithm 1 with sketches (14) and (16) is robust against small perturbations for a discrete Markov density p. From this, we prove that Algorithm 1 with sketches (14) and (16) applied to the empirical density $\hat{p}$, which is constructed based on N i.i.d. samples from a discrete density $p^\star$, recovers $p^\star$ with high probability given N is large enough; a concrete sample complexity is then derived.

D.1. Preliminaries

In what follows, for a given vector x we let $\|x\|$ and $\|x\|_\infty$ denote its Euclidean norm and its supremum norm, respectively. For a matrix A, we denote its spectral norm, Frobenius norm, and the r-th singular

Algorithm 13 Main algorithm for a continuous Markov density.

Require: $p\colon [a,b]^d \to \mathbb{R}$ and target ranks $r_1, \dots, r_{d-1}$.
Require: Orthonormal functions $\phi_1, \dots, \phi_M$ in $L^2([a,b])$ with $\phi_1 \equiv c$ and $M \geq r_1, \dots, r_{d-1}$.
for $k = 1$ to $d-1$ **do**
  if $k = 1$ **then**
    Define $\nu_1\colon [M] \times [M] \to \mathbb{R}$ as

$$\nu_1(\beta_1, \gamma_1) = \iint p(x_1, x_2)\phi_{\beta_1}(x_1)\phi_{\gamma_1}(x_2)\, dx_1 dx_2.$$

    Compute the first r_1 left singular vectors of $\nu_1(\beta_1; \gamma_1)$ and define $\widetilde{B}_1\colon [M] \times [r_1] \to \mathbb{R}$ so that these singular vectors are the columns of $\widetilde{B}_1(\beta_1; \alpha_1)$.
    Define $A_1\colon [M] \times [r_1] \to \mathbb{R}$ so that

$$A_1 = \widetilde{B}_1.$$

  else if $k < d$ **then**
    Define $\nu_k\colon [M] \times [M] \times [M] \to \mathbb{R}$ as

$$\nu_k(\beta_{k-1}, j_k, \gamma_k) = c^{k-2} \int p(x_{k-1}, x_k, x_{k+1})\phi_{\beta_{k-1}}(x_{k-1})\phi_{j_k}(x_k)\phi_{\gamma_k}(x_{k+1})\, dx_{k-1} dx_k dx_{k+1}.$$

    Compute the first r_k left singular vectors of $\nu_k(\beta_{k-1}, j_k; \gamma_k)$ and define $\widetilde{B}_k\colon [M] \times [M] \times [r_k] \to \mathbb{R}$ so that these singular vectors are the columns of $\widetilde{B}_k(\beta_{k-1}, j_k; \alpha_k)$.
    Define $A_k\colon [M] \times [r_k] \to \mathbb{R}$ so that

$$A_k(\beta_k, \alpha_k) = \widetilde{B}_k(1, \beta_k, \alpha_k).$$

  else
    Define $\widetilde{B}_d\colon [M] \times [M] \to \mathbb{R}$ as

$$\widetilde{B}_d(\beta_{d-1}, j_d) = c^{d-2} \int \phi_{\beta_{d-1}}(x_{d-1})p(x_{d-1}, x_d)\phi_{j_d}(x_d)\, dx_{d-1} dx_d.$$

  end if
end for
Solve the following d matrix equations via least-squares for the variables $g_1\colon [M] \times [r_1] \to \mathbb{R}$, $g_k\colon [r_{k-1}] \times [M] \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $g_d\colon [r_{d-1}] \times [M] \to \mathbb{R}$.

$$\begin{aligned} g_1 &= \widetilde{B}_1, \\ \sum_{\alpha_{k-1}=1}^{r_{k-1}} A_{k-1}(\beta_{k-1}; \alpha_{k-1}) g_k(\alpha_{k-1}; j_k, \alpha_k) &= \widetilde{B}_k(\beta_{k-1}; j_k, \alpha_k) \quad k = 2, \dots, d-1, \\ \sum_{\alpha_{d-1}=1}^{r_{d-1}} A_{d-1}(\beta_{d-1}; \alpha_{d-1}) g_d(\alpha_{d-1}; j_d) &= \widetilde{B}_d(\beta_{d-1}; j_d). \end{aligned} \tag{34}$$

return $G_1, \dots, G_d$ by letting

$$\begin{aligned} G_1(x_1, \alpha_1) &= \sum_{j_1=1}^{M} g_1(j_1, \alpha_1)\phi_{j_1}(x_1), \\ G_k(\alpha_{k-1}, x_k, \alpha_k) &= \sum_{j_k=1}^{M} g_k(\alpha_{k-1}, j_k, \alpha_k)\phi_{j_k}(x_k) \quad k = 2, \dots, d-1, \\ G_d(\alpha_{d-1}, x_d) &= \sum_{j_d=1}^{M} g_d(\alpha_{d-1}, j_d)\phi_{j_d}(x_d). \end{aligned}$$

value by $\|A\|$, $\|A\|_F$, and $\sigma_r(A)$, respectively. With some abuse of notation, we also let $\|A\|_\infty$ denote the largest absolute value of the entries of A. Lastly, the orthogonal group in dimension r is denoted by $O(r)$.

We also introduce the following norms for 3-tensors.

Definition 9. For any 3-tensor $G \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, or equivalently, $G\colon [n_1] \times [n_2] \times [n_3] \to \mathbb{R}$, we define the norm

$$|||G||| := \max_{i_2 \in [n_2]} \|G(\cdot, i_2, \cdot)\|.$$

Here, $G(\cdot, i_2, \cdot) \in \mathbb{R}^{n_1 \times n_2}$ denotes a matrix, and $\|G(\cdot, i_2, \cdot)\|$ denotes its spectral norm. Also, we define $\|G\|_\infty$ by

$$\|G\|_\infty = \max_{(i_1,i_2,i_3) \in [n_1]\times[n_2]\times[n_3]} |G(i_1, i_2, i_3)|.$$

Remark 8. Such a norm $|||\cdot|||$ is useful for bounding the norm of a contraction of cores. Throughout the section, we will analyze cores obtained by our algorithm: $G_1\colon [n_1] \times [r_1] \to \mathbb{R}$, $G_k\colon [r_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $G_d\colon [r_{d-1}] \times [n_d] \to \mathbb{R}$. For ease of exposition, for the specific matrices G_1 and G_d produced by the algorithm (and any perturbations of them), set $|||G_1||| = \max_{x_1 \in [n_1]} \|G(x_1, \cdot)\|$ and $|||G_d||| = \max_{x_d \in [n_d]} \|G(\cdot, x_d)\|$. Then, one can easily verify that

$$\|G_1 \circ \cdots \circ G_d\|_\infty \le |||G_1||| \cdots |||G_d|||,$$

where $\|G_1 \circ \cdots \circ G_d\|_\infty$ denotes the supremum norm of the function $(G_1 \circ \cdots \circ G_d)\colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$. In summary, the supremum norm of the contraction is easily bounded by the product of $|||\cdot|||$'s.

We start with the following basic perturbation result on a linear system $Ax = b$.

Lemma 10 *(Theorem 3.48 of [33]). For $A \in \mathbb{R}^{m\times n}$, suppose $\operatorname{rank}(A) = n \le m$. Let $\Delta A \in \mathbb{R}^{m\times n}$ be a perturbation such that $\|A^\dagger\|\|\Delta A\| < 1$. Then, $\operatorname{rank}(A + \Delta A) = n$. Moreover, let x and $x + \Delta x$ be least-squares solutions to linear systems $Ax = b$ and $(A + \Delta A)x = b + \Delta b$, respectively. Then,*

$$\frac{\|\Delta x\|}{\|x\|} \le \frac{\|A\|\|A^\dagger\|}{1 - \|A^\dagger\|\|\Delta A\|} \left[\frac{\|\Delta A\|}{\|A\|} \left(1 + \kappa(A) \frac{\|Ax - b\|}{\|A\|\|x\|} \right) + \frac{\|\Delta b\|}{\|A\|\|x\|} \right].$$

Using this we prove the following lemma which bounds the perturbation of solutions of the tensor equation $A \circ X = B$, where A is a matrix, and both X and B are three-tensors. The contraction here is performed over the second index of A and the first index of X.

Lemma 11. *For $A \in \mathbb{R}^{m\times n}$ suppose $\operatorname{rank}(A) = n \le m$. Let $\Delta A \in \mathbb{R}^{m\times n}$ be a perturbation such that $\|A^\dagger\|\|\Delta A\| < 1$. Then, $\operatorname{rank}(A + \Delta A) = n$. Let $B \in \mathbb{R}^{m\times l_1\times l_2}$ and ΔB be its perturbation. Also, let $X \in \mathbb{R}^{n\times l_1\times l_2}$ and $X + \Delta X$ be least-squares solutions to the tensor equations $A \circ X = B$ and $(A + \Delta A) \circ X = B + \Delta B$, respectively. Suppose the column space of $B \in \mathbb{R}^{m\times l_1\times l_2}$ is contained in that of A, then*

$$|||\Delta X||| \le \frac{\sqrt{2ml_2}\|A^\dagger\|}{1 - \|A^\dagger\|\|\Delta A\|} \left(\|\Delta A\|\,|||X||| + \|\Delta B\|_\infty \right).$$

In particular, if $|||X||| \ge \chi > 0$ for some constant χ, and ΔA satisfies $\|A^\dagger\|\|\Delta A\| \le 1/2$, then

$$\frac{|||\Delta X|||}{|||X|||} \le \sqrt{8ml_2}\|A^\dagger\| \left(\|\Delta A\| + \|\Delta B\|_\infty \chi^{-1} \right).$$

Proof. For any $i = (i_1, i_2)$ with $1 \le i_1 \le l_1$ and $1 \le i_2 \le l_2$, we set $x_i = X(\cdot, i_1, i_2)$ and $b_i = B(\cdot, i_1, i_2)$ to be "columns" of X and B respectively. For each equation, since b_i is contained in the column space of A, the previous lemma implies that

$$\|\Delta x_i\| \le \frac{\|A\|\|A^\dagger\|}{1 - \|A^\dagger\|\|\Delta A\|} \left(\frac{\|\Delta A\|}{\|A\|}\|x_i\| + \frac{\|\Delta b_i\|}{\|A\|} \right) = \underbrace{\frac{\|A^\dagger\|}{1 - \|A^\dagger\|\|\Delta A\|}}_{=:C} \left(\|\Delta A\|\|x_i\| + \|\Delta b_i\| \right).$$

Now, for each $1 \le i_1 \le l_1$,

$$\begin{aligned}
\|\Delta X(\cdot, i_1, \cdot)\|_F &= \sum_{j=1}^{n} \sum_{i_2=1}^{l_2} |\Delta X(j, i_1, i_2)|^2 \\
&= \sum_{i_2=1}^{l_2} \|\Delta x_{(i_1,i_2)}\|^2 \\
&\le \sum_{i_2=1}^{l_2} C^2 (\|\Delta A\|\, \|x_{(i_1,i_2)}\| + \|\Delta b_{(i_1,i_2)}\|)^2 \\
&\le \sum_{i_2=1}^{l_2} 2C^2 (\|\Delta A\|^2 \|x_{(i_1,i_2)}\|^2 + \|\Delta b_{(i_1,i_2)}\|^2) \\
&= 2C^2 (\|\Delta A\|^2 \|X(\cdot, i_1, \cdot)\|_F^2 + \|\Delta B(\cdot, i_1, \cdot)\|_F^2) \\
&\le 2C^2 (l_2 \|\Delta A\|^2 \|X(\cdot, i_1, \cdot)\|^2 + m l_2 \|\Delta B\|_\infty^2).
\end{aligned}$$

Thus,

$$\begin{aligned}
|||\Delta X||| &= \max_{i_1} \|\Delta X(\cdot, i_1, \cdot)\| \\
&\le \max_{i_1} \|\Delta X(\cdot, i_1, \cdot)\|_F \\
&\le \left(2C^2 (l_2 \|\Delta A\|^2 \max_{i_1} \|X(\cdot, i_1, \cdot)\|^2 + m l_2 \|\Delta B\|_\infty^2) \right)^{1/2} \\
&= \left(2C^2 (l_2 \|\Delta A\|^2 |||X|||^2 + m l_2 \|\Delta B\|_\infty^2) \right)^{1/2} \\
&\le \sqrt{2 m l_2} C \left(\|\Delta A\| |||X||| + \|\Delta B\|_\infty \right),
\end{aligned}$$

from which the rest of the result follows immediately. □

Lemma 12. *Let $G_1 \colon [n_1] \times [r_1] \to \mathbb{R}$, $G_k \colon [r_{k-1}] \times [n_k] \times [r_k] \to \mathbb{R}$ for $k = 2, \dots, d-1$, and $G_d \colon [r_{d-1}] \times [n_d] \to \mathbb{R}$. Denote their corresponding perturbations by ΔG_k. Suppose that there exist $\delta_k > 0$, $k = 1, \dots, d$ such that $|||\Delta G_k||| \le \delta_k |||G_k|||$ for all $k = 1, \dots, d$. Set*

$$\Delta(G_1 \circ \cdots \circ G_d) := (G_1 + \Delta G_1) \circ \cdots \circ (G_d + \Delta G_d) - G_1 \circ \cdots \circ G_d.$$

Then

$$\|\Delta(G_1 \circ \cdots \circ G_d)\|_\infty \le |||G_1||| \cdots |||G_d||| \left(\sum_{k=1}^{d} \delta_k \right) \exp\left(\sum_{k=1}^{d} \delta_k \right).$$

The following corollary is an immediate consequence of the previous lemma.

Corollary 13. *Under the same assumptions as the previous lemma, let* $\epsilon \in (0,1)$ *be given. If* $\delta := \max_{1\le k\le d}\delta_k \le \epsilon/(3d)$ *then*

$$\frac{\|\Delta(G_1\circ\cdots\circ G_d)\|_\infty}{|||G_1|||\cdots|||G_d|||}\le\epsilon.$$

Proof of Lemma 12. For ease of exposition, we set $G_k^* = G_k + \Delta G_k$ for $k = 1,\dots,d$. Next, we observe that

$$\begin{aligned}\Delta(G_1\circ\cdots\circ G_d) =&(G_1^*\circ\cdots\circ G_d^*) - (G_1\circ G_2^*\circ\cdots\circ G_d^*)\\ &+(G_1\circ G_2^*\circ\cdots\circ G_d^*) - (G_1\circ G_2\circ G_3^*\circ\cdots\circ G_d^*)\\ &+\dots\\ &+(G_1\circ\cdots\circ G_{d-1}\circ G_d^*) - (G_1\circ\cdots\circ G_d).\end{aligned}\tag{35}$$

The first line on the right-hand side of the previous equation reduces to $\Delta G_1\circ G_2^*\circ\dots G_d^*$. As in Remark 8,

$$\|\Delta G_1\circ G_2^*\circ\cdots\circ G_d^*\|_\infty\le|||\Delta G_1|||\,|||G_2^*|||\cdots|||G_d^*|||.$$

Furthermore, for $k = 1,\dots,d$,

$$|||G_k+\Delta G_k|||\le|||G_k|||+|||\Delta G_k|||\le(1+\delta_k)|||G_k|||,$$

and hence

$$\|\Delta G_1\circ G_2^*\circ\cdots\circ G_d^*\|_\infty\le\delta_1\prod_{k=2}^d(1+\delta_k)|||G_1|||\cdots|||G_d|||.$$

The other lines on the right-hand side of (35) can be bounded similarly. Thus, summing over all the terms on the right-hand side of (35), we find

$$\|\Delta(G_1\circ\cdots\circ G_d)\|_\infty\le|||G_1|||\cdots|||G_d|||\left(\sum_{k=1}^d\delta_k\right)\exp\left(\sum_{k=1}^d\delta_k\right),$$

where we have used the fact that $1+x<\exp(x)$. □

Remark 9. We note that in the previous lemma, the bounds we obtain are quite pessimistic, since they do not account for possible cancellations in contractions of multiple G_k's. The product of $|||G_k|||$'s could instead be replaced by the more cumbersome, but sharper, expression

$$\max_k\max_{\sigma_l,\sigma_r\in\{0,1\}}\max_{x_1,\dots,x_d}\|G_1^{\sigma_l}\circ\cdots\circ G_{k-1}^{\sigma_l}\|\cdot|||G_k|||\cdot\|G_{k+1}^{\sigma_r}\circ\cdots\circ G_d^{\sigma_r}\|,$$

where $G_k^0 = G_k$ and $G_k^1 = G_k^*$.

D.2. Perturbation results

We have seen from Theorem 3 that under certain mild assumptions Algorithm 1 produces a well-defined set of matrix equations (7). The following result shows that small perturbations of the coefficients and the right-hand sides of (7) result in small perturbations of the output of Algorithm 1.

Lemma 14. *Under the assumptions of Theorem 3, let $G_1, \dots, G_d$ be the solutions to (7). Given $\delta \in (0,1)$, suppose that the coefficients and right-hand sides of (7) are perturbed such that*

$$\|\Delta A_1\|, \dots, \|\Delta A_{d-1}\|, \|\Delta B_1\|_\infty, \dots, \|\Delta B_d\|_\infty \le \delta\beta =: \delta \left(\sqrt{8r \max(m,n)} c_A \left(1 + \frac{1}{c_G} \right) \right)^{-1}$$

where the constants are defined as follows:

- $r = \max_{1 \le k \le d-1} r_k$,
- $m = \max_{1 \le k \le d-1} m_k$,
- $n = \max_{1 \le k \le d} n_k$,
- $c_G = \min_{1 \le k \le d} |||G_k|||$,
- $c_A = 1 \vee \max_{1 \le k \le d-1} \|A_k^\dagger\|$.

Then, the perturbed version of (7) has $G_k + \Delta G_k$ as least-squares solutions such that

$$\frac{|||\Delta G_k|||}{|||G_k|||} \le \delta.$$

Proof. First, we compute a perturbation bound for the solution of the first equation: notice that $\sqrt{8r \max(m,n)} c_A \ge \sqrt{nr}$, which implies $\beta \le \frac{c_G}{\sqrt{nr}}$, hence

$$\frac{|||\Delta G_1|||}{|||G_1|||} = \frac{|||\Delta B_1|||}{|||G_1|||} \le \frac{\sqrt{nr} \|\Delta B_1\|_\infty}{|||G_1|||} \le \frac{\sqrt{nr} \beta \delta}{c_G} \le \frac{\sqrt{nr}}{c_G} \frac{c_G}{\sqrt{nr}} \delta = \delta.$$

Next, we observe that

$$\|\Delta A_k\| \le \beta \le \frac{1}{2c_A} \le \frac{1}{2\|A_k^\dagger\|},$$

from which it follows that $\|\Delta A_k\| \, \|A_k^\dagger\| \le 1/2$ for all $k = 1, \dots, d-1$, and therefore we may apply Lemma 11. In particular,

$$\begin{aligned} \frac{|||\Delta G_k|||}{|||G_k|||} &\le \sqrt{8 m_{k-1} r_k} \|A_{k-1}^\dagger\| \left(\|\Delta A_{k-1}\| + \frac{\|\Delta B_k\|_\infty}{c_G} \right) \\ &\le \sqrt{8mr} c_A \left(1 + \frac{1}{c_G} \right) \beta \delta \\ &\le \delta. \quad \square \end{aligned}$$

Next, we analyze the effect of a perturbation Δp of the input p of Algorithm 1. Having established Lemma 14, it suffices to quantify ΔA_k and ΔB_k in terms of Δp. First, the perturbation on $\widetilde{\Phi}_k$ from SKETCHING is obvious; we may roughly say $\Delta \widetilde{\Phi}_k \approx S_{k-1} \circ \Delta p \circ T_{k+1}$. Now that B_k is obtained as the left singular vectors of $\widetilde{\Phi}_k$ in TRIMMING, we invoke Wedin's theorem [32] to quantify ΔB_k in terms of $\Delta \widetilde{\Phi}_k$. To this end, we first introduce the following distance comparing two 3-tensors up to rotation, which is common in spectral analysis of linear algebra, see Chapter 2 of [5].

Definition 15. For any 3-tensors $\hat{G}, G \in \mathbb{R}^{r_1 \times n \times r_2}$, we define

$$\operatorname{dist}(\hat{G}, G) := \min_{R_1 \in O(r_1), R_2 \in O(r_2)} |||\hat{G} - R_1 \circ G \circ R_2|||.$$

Here, $R_1 \circ G \circ R_2$ denotes a 3-tensor formed by contracting the second index of R_1 and the first index of G and contracting the first index of R_2 and the third index of G.

Using this distance, we compare the $\hat{G}_1, \dots, \hat{G}_d$, the which result from applying Algorithm 1 to $\hat{p} = p + \Delta p$ as input, with $G_1, \dots, G_d$, the results of Algorithm 1 with p as input. We will restrict our analysis to the case where p is a Markov model and Algorithm 1 is implemented with sketches (14) and (16) as in Section 5.

Remark 10. As in Remark 8, we define $\mathrm{dist}(\cdot,\cdot)$ for the first and last cores as well. Accordingly, we set

$$\mathrm{dist}(\hat{G}_1, G_1) = \min_{R \in O(r_1)} |||\hat{G}_1 - G_1 R|||,$$
$$\mathrm{dist}(\hat{G}_d, G_d) = \min_{R \in O(r_{d-1})} |||\hat{G}_d - R G_d|||,$$

where $G_1, \hat{G}_1 \colon [n_1] \times [r_1] \to \mathbb{R}$ and $G_d, \hat{G}_d \colon [r_{d-1}] \times [n_d] \to \mathbb{R}$ are the first and last cores produced by the algorithm, respectively. Here $|||\cdot|||$ on the right-hand sides of the previous equations are the norms defined for the first and last cores introduced in Remark 8.

Proposition 16. *Under the assumptions of Theorem 6, let $G_1, \dots, G_d$ be the cores of p obtained as solutions to (7). Suppose we apply Algorithm 1 to the perturbed input $\hat{p} = p + \Delta p$ with sketches (14) and (16) as in Theorem 6; the results are denoted as $\hat{G}_1, \dots, \hat{G}_d$. Suppose further that for some fixed $\delta \in (0,1)$,*

$$\begin{aligned} &\|\Delta p(x_1; x_2)\|_\infty, \|\Delta p(x_1, x_2; x_3)\|_\infty, \dots, \|\Delta p(x_{d-2}, x_{d-1}; x_d)\|_\infty \|\Delta p(x_{d-1}; x_d)\|_\infty \\ &\le \frac{c_P}{2n^2(1+c_P)} \left(\sqrt{8rn} c_A \left(1 + \frac{1}{c_G}\right)\right)^{-1} \delta =: \gamma\delta \end{aligned} \tag{36}$$

where the constants are defined as follows:

- $n = \max_{1 \le k \le d} n_k$,
- $c_P = \sigma_{r_1}(p(x_1; x_2)) \wedge \min_{k=2,\dots,d-1} \sigma_{r_k}(p(x_{k-1}, x_k; x_{k+1}))$,
- $c_G = \min_{1 \le k \le d} |||G_k|||$,
- $c_A = 1 \vee \max_{1 \le k \le d-1} \|A_k^\dagger\|$.

Then, for $k = 1, \dots, d$,

$$\frac{\mathrm{dist}(\hat{G}_k, G_k)}{|||G_k|||} \le \delta.$$

Proof. We apply Algorithm 1 to p and $\hat{p}$ with sketches (14) and (16) as in Theorem 6; the resulting coefficient matrices and right-hand sides of (7) are

$$A_1, \dots, A_{d-1}, B_1, \dots, B_{d-1}, p(x_{d-1}, x_d) \quad \text{and} \quad \hat{A}_1, \dots, \hat{A}_{d-1}, \hat{B}_1, \dots, \hat{B}_{d-1}, \hat{p}(x_{d-1}, x_d),$$

respectively. Our goal is to quantify their differences.

Recall that B_1 and $\hat{B}_1$ are the first r_1 left singular vectors of $p(x_1; x_2)$ and $\hat{p}(x_1; x_2)$, respectively. We apply Wedin's theorem presented in Theorem 2.9 of [5]; if $\|\Delta p(x_1; x_2)\| < \sigma_{r_1}(p(x_1; x_2))$, we can find $R_1 \in O(r_1)$ such that

$$\hat{B}_1(x_1; \alpha_1) = \sum_{a_1=1}^{r_1} B_1(x_1; a_1) R_1(a_1; \alpha_1) + E_1(x_1; \alpha_1),$$

and

$$\|E_1(x_1;\alpha_1)\| \le \frac{\sqrt{2}\|\Delta p(x_1;x_2)^\top B_1(x_1;a_1)\|}{\sigma_{r_1}(p(x_1;x_2)) - \|\Delta p(x_1;x_2)\|}.$$

In particular, if $\|\Delta p(x_1;x_2)\| \le (1-1/\sqrt{2})\sigma_{r_1}(p(x_1;x_2))$, using $\|B_1(x_1;a_1)\| = 1$, we have

$$\|E_1(x_1;\alpha_1)\| \le \frac{2\|\Delta p(x_1;x_2)\|}{\sigma_{r_1}(p(x_1;x_2))}.$$

Similarly, for $k = 2,\dots,d-1$, if $\|\Delta p(x_{k-1},x_k;\alpha_k)\| \le (1-1/\sqrt{2})\sigma_{r_k}(p(x_{k-1},x_k;x_{k+1}))$, we can find $R_k \in O(r_k)$ such that

$$\hat{B}_k(x_{k-1},x_k;\alpha_k) = \sum_{a_k=1}^{r_k} B_k(x_{k-1},x_k;a_k)R_k(a_k;\alpha_k) + E_k(x_{k-1},x_k;\alpha_k),$$

and

$$\|E_k(x_{k-1},x_k;\alpha_k)\| \le \frac{2\|\Delta p(x_{k-1},x_k;x_{k+1})\|}{\sigma_{r_k}(p(x_{k-1},x_k;x_{k+1}))}.$$

Accordingly, for $k = 2,\dots,d-1$,

$$\hat{A}_k(x_k;\alpha_k) = \sum_{a_k=1}^{r_k} A_k(x_k;a_k)R_k(a_k;\alpha_k) + \sum_{x_{k-1}=1}^{n_{k-1}} E_k(x_{k-1},x_k;\alpha_k).$$

Conceptually speaking, we see that the perturbation in the coefficients and right-hand sides of equations (7) for $G_1,\dots,G_d$ consist of two parts: a rotation and an additive error. We will see that though the rotations affect the individual G_k's, they do not change the final contraction $G_1 \circ \cdots \circ G_d$, and hence do not directly contribute to the pointwise error in the compressed representation of the density. To that end, we define the rotated quantities Φ_1^*, A_k^*, and B_k^* as follows:

$$\begin{aligned}
B_1^*(x_1,\alpha_1) &:= \sum_{a_1=1}^{r_1} B_1(x_1,a_1)R_1(a_1,\alpha_1) =: A_1^*(x_1,\alpha_1),\\
B_k^*(x_{k-1},x_k,\alpha_k) &:= \sum_{a_k=1}^{r_k} B_k(x_{k-1},x_k,a_k)R_k(a_k,\alpha_k),\\
A_k^*(x_k,\alpha_k) &:= \sum_{x_{k-1}=1}^{n_{k-1}} B_k^*(x_{k-1},x_k,\alpha_k) = \sum_{a_k=1}^{r_k} A_k(x_k,a_k)R_k(a_k,\alpha_k).
\end{aligned}$$

Now, consider the following equations:

$$\begin{aligned}
G_1^* &= B_1^*,\\
\sum_{\alpha_{k-1}=1}^{r_{k-1}} A_{k-1}^*(x_{k-1},\alpha_{k-1})G_k^*(\alpha_{k-1},x_k,\alpha_k) &= B_k^*(x_{k-1},x_k,\alpha_k) \quad k = 2,\dots,d-1\\
\sum_{\alpha_{d-1}=1}^{r_{d-1}} A_{d-1}^*(x_{d-1},\alpha_{d-1})G_d^*(\alpha_{d-1},x_d) &= p(x_{d-1},x_d).
\end{aligned} \tag{37}$$

These equations can be viewed as the rotated version of the original equations for $G_1, \dots, G_d$. In fact, the solutions are also simply rotated from the original solutions $G_1, \dots, G_d$ as follows[1]:

$$\begin{aligned} G_1^*(x_1, a_1) &= \sum_{a_1=1}^{r_1} G_1(x_1, a_1) R_1(a_1, \alpha_1), \\ G_k^*(\alpha_{k-1}, x_k, \alpha_k) &= \sum_{a_{k-1}=1}^{r_{k-1}} \sum_{a_k=1}^{r_k} R_{k-1}(a_{k-1}, \alpha_{k-1}) G_k(a_{k-1}, x_k, a_k) R_k(a_k, \alpha_k) \quad k = 2, \dots, d-1, \\ G_d^*(\alpha_{d-1}, x_d) &= \sum_{a_{d-1}=1}^{r_{d-1}} R_{d-1}(a_{d-1}, \alpha_{d-1}) G_d(a_{d-1}, x_d). \end{aligned}$$

By definition, it is obvious that $|||G_k||| = |||G_k^*|||$ for all $k = 1, \dots, d$ and $G_1 \circ \cdots \circ G_d = G_1^* \circ \cdots \circ G_d^*$.

We now address the effect of the additive error. As a result of the above discussion, running our algorithm with input $\hat{p}$ amounts to a perturbed version of (37), where the coefficients and the right-hand sides are perturbed as follows:

$$\begin{aligned} &\hat{B}_k = B_k^* + \Delta B_k^*, \quad \hat{A}_k = A_k^* + \Delta A_k^* \quad k = 1, \dots, d-1, \\ &\hat{p}(x_{d-1}; x_d) = p(x_{d-1}; x_d) + \Delta p(x_{d-1}; x_d). \end{aligned}$$

By construction, $\Delta B_1^* = \Delta A_1^* = E_1$,

$$\Delta B_k^* = E_k, \quad \Delta A_k^*(x_k; \alpha_k) = \sum_{x_{k-1}=1}^{n_{k-1}} E_k(x_{k-1}, x_k; \alpha_k) \quad k = 2, \dots, d-1.$$

We now look for suitable bounds on $\hat{G}_k - G_k^*$ for $k = 1, \dots, d$. In light of Lemma 14, it suffices to construct suitable bounds for $\|\Delta A_1^*\|, \dots, \|\Delta A_{d-1}^*\|$, $\|\Delta B_1^*\|_\infty, \dots, \|\Delta B_{d-1}^*\|_\infty$, and $\|\Delta p(x_{d-1}; x_d)\|_\infty$. In particular, we claim

$$\|E_1\|, \|\Delta A_2^*\| \dots, \|\Delta A_{d-1}^*\|, \|E_1\|_\infty, \dots \|E_{d-1}\|_\infty, \|\Delta p(x_{d-1}; x_d)\|_\infty \le \beta\delta, \tag{38}$$

where β is as in Lemma 14, namely,

$$\beta = \left(\sqrt{8rn} c_A \left(1 + \frac{1}{c_G} \right) \right)^{-1}.$$

Here, we use the fact that $m_k = n_k$ and $\max_{1 \le k \le d-1} r_k \le n$. Essentially, we have

$$\gamma = \frac{c_P}{2n^2(1 + c_P)} \beta.$$

By definition of G_k^* and A_k^*, it is obvious that $c_G = \min_{1 \le k \le d} |||G_k||| = \min_{1 \le k \le d} |||G_k^*|||$ and $c_A = \max_{1 \le k \le d-1} \|A_k^\dagger\| = \max_{1 \le k \le d-1} \|(A_k^*)^\dagger\|$. Hence, by Lemma 14, it suffices to check (38) to prove that for $k = 1, \dots, d$,

$$\frac{|||\hat{G}_k - G_k^*|||}{|||G_k^*|||} \le \delta. \tag{39}$$

[1] More simply, $G_1^* = G_1 R_1$, $G_k^* = R_{k-1}^\top \circ G_k \circ R_k$ for $k = 2, \dots, d-1$, and $G_d^* = R_{d-1}^\top G_d$.

Let us verify (38). First,

$$\|\Delta p(x_{d-1}; x_d)\|_\infty \le \gamma\delta \le \beta\delta.$$

Moreover, as we showed above,

$$\|E_1\|_\infty \le \|E_1\| \le \frac{2\|\Delta p(x_1; x_2)\|}{\sigma_{r_1}(p(x_1; x_2))} \le \frac{2n\|\Delta p(x_1; x_2)\|_\infty}{\sigma_{r_1}(p(x_1; x_2))} \le \frac{2n}{c_P}\gamma\delta \le \beta\delta.$$

For $k = 2, \dots, d-1$, we verify $\|\Delta A_k^*\| \le n^{1/2}\|E_k(x_{k-1}, x_k; \alpha_k)\|$. Note that $\Delta A_k^* = P_k E_k(x_{k-1}, x_k; \alpha_k)$; here $P_k \in \mathbb{R}^{n_k \times n_k n_{k-1}} = [I_k, \dots, I_k]$, where $I_k \in \mathbb{R}^{n_k \times n_k}$ is the identity matrix. Hence, $\|\Delta A_k^*\| \le \|P_k\|\|E_k(x_{k-1}, x_k; \alpha_k)\| \le n^{1/2}\|E_k(x_{k-1}, x_k; \alpha_k)\|$ because $\|P_k\| = \sqrt{n_{k-1}} \le n^{1/2}$. Therefore,

$$\begin{aligned}
\|\Delta A_k^*\|, \|E_k\|_\infty &\le n^{1/2}\|E_k(x_{k-1}, x_k; \alpha_k)\| \\
&\le \frac{2n^{1/2}\|\Delta p(x_{k-1}, x_k; x_{k+1})\|}{\sigma_{r_k}(p(x_{k-1}, x_k; x_{k+1}))} \\
&\le \frac{2n^2\|\Delta p(x_{k-1}, x_k; x_{k+1})\|_\infty}{\sigma_{r_k}(p(x_{k-1}, x_k; x_{k+1}))} \\
&\le \frac{2n^2\gamma}{c_P}\delta \le \beta\delta.
\end{aligned}$$

Hence, (38) is satisfied, thus (39) holds. By definition of $\mathrm{dist}(\cdot,\cdot)$ and $|||\cdot|||$, we have for $k = 1, \dots, d$,

$$\frac{\mathrm{dist}(\hat{G}_k, G_k)}{|||G_k|||} \le \frac{|||\hat{G}_k - G_k^*|||}{|||G_k|||} = \frac{|||\hat{G}_k - G_k^*|||}{|||G_k^*|||} \le \delta. \quad \square$$

The following result on the error of the contraction follows immediately from the previous Proposition, combined with Corollary 13.

Theorem 17. *Under the assumptions of Theorem 6, let $G_1, \dots, G_d$ be the cores of p obtained as solutions to (7). Suppose we apply Algorithm 1 to the perturbed input $\hat{p} = p + \Delta p$ with sketches (14) and (16) as in Theorem 6; the results are denoted as $\hat{G}_1, \dots, \hat{G}_d$. Suppose further that for some fixed $\epsilon \in (0, 1)$,*

$$\begin{aligned}
&\|\Delta p(x_1; x_2)\|_\infty, \|\Delta p(x_1, x_2; x_3)\|_\infty, \dots, \|\Delta p(x_{d-2}, x_{d-1}; x_d)\|_\infty \|\Delta p(x_{d-1}; x_d)\|_\infty \\
&\le \frac{c_P}{6dn^2(1 + c_P)} \left(\sqrt{8rn}c_A \left(1 + \frac{1}{c_G}\right)\right)^{-1} \epsilon,
\end{aligned}$$

where the constants n, c_P, c_G, c_A are as in Proposition 16. Then,

$$\frac{\|\hat{G}_1 \circ \cdots \circ \hat{G}_d - G_1 \circ \cdots \circ G_d\|_\infty}{|||G_1||| \dots |||G_d|||} \le \epsilon.$$

D.3. Estimation error analysis

Lastly, we present a precise version of Theorem 7. Recall that our main interest is to apply Algorithm 1 to an empirical density $\hat{p}$ constructed based on N i.i.d. samples from some underlying density $p^\star$; letting $\hat{G}_1, \dots, \hat{G}_d$ be the results of Algorithm 1 applied to $\hat{p}$, we hope to claim $p^\star \approx \hat{G}_1 \circ \cdots \circ \hat{G}_d$.

Using the previous perturbation result (Proposition 16), we will quantify a difference between $\hat{G}_k$ and $G_k^\star$, where $G_1^\star, \dots, G_d^\star$ are the results of Algorithm 1 applied to $p^\star$. The only technicality here is that the

perturbed input is not arbitrary, but given as an empirical density. Therefore, the perturbation $\hat{p} - p^\star$ can be represented in terms of the sample size N. The following lemma derives a concrete bound on $\hat{p} - p^\star$ using simple concentration inequalities.

Lemma 18. *Let $p^\star \colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$ be a density. Suppose $\hat{p}$ is an empirical density based on N i.i.d. samples from $p^\star$. Let $\Delta p^\star = \hat{p} - p^\star$ and $n = \max_{1 \le k \le d} n_k$, then for any $\eta \in (0,1)$, the following inequalities hold with probability at least $1 - \eta$:*

$$\begin{aligned} \|\Delta p^\star(x_1; x_2)\|_\infty &\le \sqrt{\frac{\log(2n^2 d/\eta)}{2N}}, \\ \|\Delta p^\star(x_{k-1}, x_k; x_{k+1})\|_\infty &\le \sqrt{\frac{\log(2n^3 d/\eta)}{2N}} \quad k = 2, \dots, d-1, \\ \|\Delta p^\star(x_{d-1}; x_d)\|_\infty &\le \sqrt{\frac{\log(2n^2 d/\eta)}{2N}}. \end{aligned}$$

Proof. Since $N\hat{p}$ is the sum of N independent Bernoulli random variables, concentration inequalities imply that for any fixed $x_1 \in [n_1]$ and $x_2 \in [n_2]$ and $t \ge 0$,

$$\mathbb{P}(|\Delta p^\star(x_1, x_2)| > t) \le 2e^{-2Nt^2}.$$

Due to the union bound, $\|\Delta p^\star(x_1; x_2)\|_\infty \le t$ holds with probability at least $1 - 2n^2 e^{-2Nt^2}$. Equivalently,

$$\|\Delta p^\star(x_1; x_2)\|_\infty \le \sqrt{\frac{\log(2n^2/\eta)}{2N}}$$

holds with probability at least $1 - \eta$. Similarly, for $k = 2, \dots, d-1$,

$$\|\Delta p^\star(x_{k-1}, x_k; x_{k+1})\|_\infty \le \sqrt{\frac{\log(2n^3/\eta)}{2N}}$$

holds with probability at least $1 - \eta$. Due to the union bound,

$$\begin{aligned} \|\Delta p^\star(x_1; x_2)\|_\infty &\le \sqrt{\frac{\log(2n^2 d/\eta)}{2N}}, \\ \|\Delta p^\star(x_{k-1}, x_k; x_{k+1})\|_\infty &\le \sqrt{\frac{\log(2n^3 d/\eta)}{2N}} \quad k = 2, \dots, d-1, \\ \|\Delta p^\star(x_{d-1}; x_d)\|_\infty &\le \sqrt{\frac{\log(2n^2 d/\eta)}{2N}} \end{aligned}$$

hold with probability at least $1 - \eta$. □

Hence, we have proved that the perturbation $\hat{p} - p^\star$ is bounded above by $O(1/\sqrt{N})$. Now, by comparing this bound with the right-hand sides of (36), we obtain a complexity. Again, we will restrict our analysis to the case where $p^\star$ is a Markov model and Algorithm 1 is implemented with sketches (14) and (16) as in Section 5.

Theorem 19. *Let $p^\star \colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$ be a Markov density satisfying Condition 1 such that the rank of the k-th unfolding matrix of $p^\star$ is r_k for each $k = 1, \dots, d-1$. Let $G_1^\star, \dots, G_d^\star$ be the cores of $p^\star$ obtained by applying Algorithm 1 to $p^\star$ with sketches (14) and (16) as in Theorem 6; $A_1^\star, \dots, A_{d-1}^\star$ are the resulting coefficient matrices in (7).*

Now, let $\hat{p}$ be an empirical density based on N i.i.d. samples from $p^\star$. Let $\hat{G}_1, \dots, \hat{G}_d$ be the results of applying Algorithm 1 to $\hat{p}$ with sketches (14) and (16) as in Theorem 6. Given $\delta \in (0,1)$ and $\eta \in (0,1)$, suppose

$$N \geq 16 c_A^2 \left(1 + \frac{1}{c_G}\right)^2 \left(1 + \frac{1}{c_P}\right)^2 \frac{n^5 r \log(2n^3 d/\eta)}{\delta^2}, \tag{40}$$

where

- $n = \max_{1 \leq k \leq d} n_k$,
- $c_P = \sigma_{r_1}(p^\star(x_1; x_2)) \wedge \min_{k=2,\dots,d-1} \sigma_{r_k}(p^\star(x_{k-1}, x_k; x_{k+1}))$,
- $c_G = \min_{1 \leq k \leq d} |||G_k^\star|||$,
- $c_A = 1 \vee \max_{1 \leq k \leq d-1} \|(A_k^\star)^\dagger\|$.

Then,

$$\frac{\operatorname{dist}(\hat{G}_k, G_k^\star)}{|||G_k^\star|||} \leq \delta \quad \forall k = 1, \dots, d$$

with probability at least $1 - \eta$.

Proof. Due to Proposition 16, it suffices to show that N satisfies

$$\sqrt{\frac{\log(2n^3 d/\eta)}{2N}} \leq \frac{c_P}{2n^2(1 + c_P)} \left(\sqrt{8rn} c_A \left(1 + \frac{1}{c_G}\right)\right)^{-1} \delta,$$

which is equivalent to (40). □

In addition, using Proposition 17, we obtain the following sample complexity for bounding the error of the contraction.

Theorem 20. *Let $p^\star \colon [n_1] \times \cdots \times [n_d] \to \mathbb{R}$ be a Markov density satisfying Condition 1 such that the rank of the k-th unfolding matrix of $p^\star$ is r_k for each $k = 1, \dots, d-1$. Let $G_1^\star, \dots, G_d^\star$ be the cores of $p^\star$ obtained by applying Algorithm 1 to $p^\star$ with sketches (14) and (16) as in Theorem 6; $A_1^\star, \dots, A_{d-1}^\star$ are the resulting coefficient matrices in (7).*

Now, let $\hat{p}$ be an empirical density based on N i.i.d. samples from $p^\star$. Let $\hat{G}_1, \dots, \hat{G}_d$ be the results of applying Algorithm 1 to $\hat{p}$ with sketches (14) and (16) as in Theorem 6. Given $\epsilon \in (0,1)$ and $\eta \in (0,1)$, suppose

$$N \geq 144 c_A^2 \left(1 + \frac{1}{c_G}\right)^2 \left(1 + \frac{1}{c_P}\right)^2 \frac{d^2 n^5 r \log(2n^3 d/\eta)}{\epsilon^2},$$

where

- $n = \max_{1 \leq k \leq d} n_k$,
- $c_P = \sigma_{r_1}(p^\star(x_1; x_2)) \wedge \min_{k=2,\dots,d-1} \sigma_{r_k}(p^\star(x_{k-1}, x_k; x_{k+1}))$,
- $c_G = \min_{1 \leq k \leq d} |||G_k^\star|||$,
- $c_A = 1 \vee \max_{1 \leq k \leq d-1} \|(A_k^\star)^\dagger\|$.

Then,

$$\frac{\|\hat{G}_1 \circ \cdots \circ \hat{G}_d - G_1^\star \circ \cdots \circ G_d^\star\|_\infty}{|||G_1^\star||| \ldots |||G_d^\star|||} \leq \epsilon$$

with probability at least $1 - \eta$.

Remark 11. In Theorems 19 and 20, notice that the constants c_P, c_G, c_A are independent of d; to see this, observe that they are determined by the marginals of $p^\star$, namely, $p^\star(x_1; x_2)$ and $p^\star(x_{k-1}, x_k; x_{k+1})$, which are independent of d under Condition 1. Therefore, we obtain Theorem 7 and Corollary 8, where the upper bounds hide those constants under the "big-O" notation as they are independent of d. Meanwhile, notice that Theorems 19 and 20 are valid for $p^\star$ that may not satisfy Condition 1; in such a case, the constants c_P, c_G, c_A may depend on d in principle. Extensive numerical experiments, however, suggest that the constants c_P, c_G, c_A are often nearly independent of d for a broad class of Markov models that may not satisfy Condition 1, such as the Ginzburg-Landau model used in Section 6.

References

[1] Daniele Bigoni, Allan P. Engsig-Karup, Youssef M. Marzouk, Spectral tensor-train decomposition, SIAM J. Sci. Comput. 38 (4) (2016) A2405–A2439.
[2] Tai-Danae Bradley, E. Miles Stoudenmire, John Terilla, Modeling sequences with quantum states: a look under the hood, Mach. Learn.: Sci. Technol. 1 (3) (2020) 035008.
[3] Maolin Che, Yimin Wei, Randomized algorithms for the approximations of Tucker and the tensor train decompositions, Adv. Comput. Math. 45 (1) (2019) 395–428.
[4] Yian Chen, Jeremy Hoskins, Yuehaw Khoo, Michael Lindsey, Committor functions via tensor networks, arXiv preprint, arXiv:2106.12515, 2021.
[5] Yuxin Chen, Yuejie Chi, Jianqing Fan, Cong Ma, Spectral methods for data science: a statistical perspective, Found. Trends Mach. Learn. 14 (5) (2021) 566–806.
[6] Song Cheng, Lei Wang, Tao Xiang, Pan Zhang, Tree tensor networks for generative modeling, Phys. Rev. B. 99 (15) (2019) 155131.
[7] Hussam Al Daas, Grey Ballard, Peter Benner, Parallel algorithms for tensor train arithmetic, SIAM J. Sci. Comput. 44 (1) (2022) C25–C53.
[8] Hussam Al Daas, Grey Ballard, Paul Cazeaux, Eric Hallman, Agnieszka Miedlar, Mirjeta Pasha, Tim W. Reid, Arvind K. Saibaba, Randomized algorithms for rounding in the tensor-train format, arXiv preprint, arXiv:2110.04393, 2021.
[9] Sergey Dolgov, Karim Anaya-Izquierdo, Colin Fox, Robert Scheichl, Approximation and sampling of multivariate probability distributions in the tensor train decomposition, Stat. Comput. 30 (2020) 603–625.
[10] Marylou Gabrié, Grant M. Rotskoff, Eric Vanden-Eijnden, Adaptive Monte Carlo augmented with normalizing flows, arXiv preprint, arXiv:2105.12603, 2021.
[11] Ian Goodfellow, Jean Pouget-Abadie Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative adversarial nets, Advances in Neural Information Processing Systems (2014).
[12] Alex Gorodetsky, Sertac Karaman, Youssef Marzouk, A continuous analogue of the tensor-train decomposition, Comput. Methods Appl. Mech. Eng. 347 (2019) 59–84.
[13] Nathan Halko, Per-Gunnar Martinsson, Joel A. Tropp, Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions, SIAM Rev. 53 (2) (2011) 217–288.
[14] Zhao-Yu Han, Jun Wang, Heng Fan, Lei Wang, Pan Zhang, Unsupervised generative modeling using matrix product states, Phys. Rev. X. 8 (2018) 031012.
[15] Yuehaw Khoo, Jianfeng Lu, Lexing Ying, Efficient construction of tensor ring representations from sampling, Multiscale Model. Simul. 19 (3) (2021) 1261–1284.
[16] Diederik P. Kingma, Max Welling, Auto-encoding variational Bayes, arXiv preprint, arXiv:1312.6114, 2013.
[17] Yuji Nakatsukasa, Joel A. Tropp, Fast & accurate randomized algorithms for linear systems and eigenvalue problems, arXiv preprint, arXiv:2111.00113, 2021.
[18] Georgii S. Novikov, Maxim E. Panov, Ivan V. Oseledets, Tensor-train density estimation, Uncertainty in Artificial Intelligence (2021) 1321–1331.
[19] Ivan Oseledets, Eugene Tyrtyshnikov, TT-cross approximation for multidimensional arrays, Linear Algebra Appl. 432 (1) (2010) 70–88.
[20] Ivan V. Oseledets, Tensor-train decomposition, SIAM J. Sci. Comput. 33 (5) (2011) 2295–2317.
[21] D. Perez-Garcia, F. Verstraete, M.M. Wolf, J.I. Cirac, Matrix product state representations, Quantum Info. Comput. 7 (5) (2007) 401–430.
[22] Danilo Rezende, Shakir Mohamed, Variational inference with normalizing flows, in: International Conference on Machine Learning, 2015, pp. 1530–1538.

[23] Vladimir Rokhlin, Mark Tygert, A fast randomized algorithm for overdetermined linear least-squares regression, Proc. Natl. Acad. Sci. 105 (36) (2008) 13212–13217.
[24] Lars Ruthotto, Eldad Haber, An introduction to deep generative modeling, GAMM-Mitteilungen 44 (2) (2021) e202100008.
[25] Dmitry Savostyanov, Ivan Oseledets, Fast adaptive interpolation of multi-dimensional arrays in tensor train format, The 2011 International Workshop on Multidimensional (nD) Systems (2011) 1–8.
[26] Tianyi Shi, Maximilian Ruth, Alex Townsend, Parallel algorithms for computing the tensor-train decomposition, arXiv preprint, arXiv:2111.10448, 2021.
[27] Michael Steinlechner, Riemannian optimization for high-dimensional tensor completion, SIAM J. Sci. Comput. 38 (5) (2016) S461–S484.
[28] Endre Süli, David F. Mayers, An Introduction to Numerical Analysis, Cambridge University Press, 2003.
[29] Yiming Sun, Yang Guo, Charlene Luo, Joel Tropp, Madeleine Udell, Low-rank Tucker approximation of a tensor from streaming data, SIAM J. Math. Data. Sci. 2 (4) (2020) 1123–1150.
[30] Esteban G. Tabak, Eric Vanden-Eijnden, Density estimation by dual ascent of the log-likelihood, Commun. Math. Sci. 8 (1) (2010) 217–233.
[31] Wenqi Wang, Vaneet Aggarwal, Shuchin Aeron, Efficient low rank tensor ring completion, International Conference on Computer Vision (2017) 5697–5705.
[32] Per-Åke Wedin, Perturbation bounds in connection with singular value decomposition, BIT Numer. Math. 12 (1) (1972) 99–111.
[33] Holger Wendland, Numerical Linear Algebra: An Introduction, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2017.
[34] Steven R. White, Density matrix formulation for quantum renormalization groups, Phys. Rev. Lett. 69 (1992) 2863–2866.
[35] Wenjing Yang, Hans-Georg Müller, Ulrich Stadtmüller, Functional singular component analysis, J. R. Stat. Soc. Ser. B: Stat. Methodol. 73 (3) (2011) 303–324.