ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

MATHICSE

Mathematics Institute of Computational Science and Engineering

School of Basic Sciences - Section of Mathematics

MATHICSE Technical Report

Nr. 17.2014

March 2014

Low-rank tensor methods for communicating Markow processes

Daniel Kressner, Francisco Macedo

http://mathicse.epfl.ch

Phone: +41 21 69 37648

Address:
EPFL - SB - MATHICSE (Bâtiment MA)
Station 8 - CH-1015 - Lausanne - Switzerland

Fax: +41 21 69 32545

Low-rank tensor methods for communicating Markov processes

Daniel Kressner[1] and Francisco Macedo[1,2]

[1] EPF Lausanne, SB-MATHICSE-ANCHP, Station 8, CH-1015 Lausanne, Switzerland, {daniel.kressner,francisco.macedo}@epfl.ch
[2] IST, Alameda Campus, Av. Rovisco Pais, 1, 1049-001 Lisbon, Portugal

Abstract. Stochastic models that describe interacting processes, such as stochastic automata networks, feature a dimensionality that grows exponentially with the number of processes. This state space explosion severely impairs the use of standard methods for the numerical analysis of such Markov chains. In this work, we discuss the approximation of solutions by matrix product states or, equivalently, by tensor train decompositions. Two classes of algorithms based on this low-rank decomposition are proposed, using either iterative truncation or alternating optimization. Our approach significantly extends existing approaches based on product form solutions and can, in principle, attain arbitrarily high accuracy. Numerical experiments demonstrate that the newly proposed algorithms are particularly well suited to deal with pairwise neighbor interactions.

1 Introduction

Markov processes featuring high-dimensional state spaces regularly arise when modelling processes that interact with each other. Termed communicating Markov processes in [4], this class includes queuing networks, stochastic automata networks, and stochastic Petri nets. The need for considering the joint probability distribution for a network of stochastic processes is responsible for the exponential growth of the state space dimension, which severely impairs the numerical analysis of such Markov processes. For example, all standard iterative solvers [2] for addressing the linear system or, equivalently, the eigenvalue problem needed for determining the stationary probability distribution have a complexity that scales at least linearly with the state space dimension.

The high dimensionality of the state space is usually coped with by either performing model reduction or by exploiting the rich structure of the transition rate matrix in the numerical solution procedure. Product form solutions represent a particularly popular reduction technique and have been successfully used in a wide range of applications. The basic idea of this reduction is to yield a system for which the stationary distribution factorizes into a product of distributions for the individual processes. This reduced system then allows for a much less expensive numerical treatment. General techniques for arriving at product form solutions are described, e.g., in [18, Ch. 6]. Extensive work has been done

on finding conditions under which such product form approach applies; see [10, 9] for some recent results. However, its practical range of applicability is still limited to specific subclasses. A rather different approach is based on the observation that the transition rate matrix of a communicating Markov process can often be represented by a short sum of Kronecker products [25]. This property can then be exploited when performing matrix-vector multiplications or constructing preconditioners [19, 20] to reduce the cost of iterative solvers significantly. Still, the complexity scales linearly with the state space dimension.

The approach proposed in this paper can be viewed as a combination of the two approaches above, performing (nonlinear) model reduction along with the iterative solution. For this purpose, we exploit the fact that a vector containing joint probabilities can be naturally rearranged into a tensor. This then allows us to use established low-rank tensor approximation techniques [12]. Such an approach has already been considered by Buchholz [6], using the so called CP decomposition. As explained below, this decomposition may not always be the best choice as it does not exploit the topology of interactions. We therefore propose the use of matrix product states (MPS) or, equivalently, TT decompositions. Note that MPS and related low-rank tensor decompositions have already been used for simulating stochastic systems [14, 15]. The novelty of our contribution consists of explicitly targeting the computation of the stationary distribution for a finite-dimensional communicating Markov process, developing and comparing different algorithmic approaches.

The rest of this work is organized as follows. In Section 2, we will introduce the low-rank tensor decompositions that will subsequently, in Section 3, be used to develop efficient algorithms for computing stationary distributions. Section 4 compares the performance of these algorithms for two popular examples.

2 Low-rank decompositions of tensors

In this section, we discuss low-rank decompositions for compressing vectors. When considering, for example, a network of d communicating finite state Markov processes, the vector π containing the stationary distribution has length $n_1 n_2 \cdots n_d$, where n_μ denotes the number of states in the μth process for $\mu = 1, \ldots, d$. Quite naturally, the entries of this vector can be rearranged into an $n_1 \times \cdots \times n_d$ array, defining a dth order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. The entries of $\mathcal{X}$ are denoted by

$$\mathcal{X}_{i_1, i_2, \ldots, i_d}, \quad 1 \le i_\mu \le n_\mu, \quad \mu = 1, \ldots, d.$$

The opposite operation is denoted by $\mathrm{vec}(\mathcal{X})$, which stacks the entries of $\mathcal{X}$ back into a long vector, so that the indices of $\mathcal{X}$ are sorted in lexicographical order.

For $d = 2$, $\mathcal{X}$ becomes a matrix and there is a unique notion of rank, which can be computed by the singular value decomposition (SVD) [11]. The extension of this concept to $d > 2$ is by no means unique, and different notions of low rank decompositions for tensors have been developed; see [16] for an overview.

The *CP decomposition* takes the form

$$\mathrm{vec}(\mathcal{X}) = \sum_{r=1}^{R} u_r^{(1)} \otimes u_r^{(2)} \otimes \cdots \otimes u_r^{(d)} = \sum_{r=1}^{R} \bigotimes_{\mu=1}^{d} u_r^{(\mu)}, \tag{1}$$

where each $u_r^{(\mu)}$ is a vector of length n_μ, and $\otimes$ denotes the usual Kronecker product. The tensor rank of $\mathcal{X}$ is the smallest R admitting such a decomposition. The individual entries of (1) are given by

$$\mathcal{X}_{i_1,i_2,\ldots,i_d} = \sum_{r=1}^{R} u_{r,i_1}^{(1)} u_{r,i_2}^{(2)} \cdots u_{r,i_d}^{(d)}, \quad 1 \le i_\mu \le n_\mu, \quad \mu = 1, \ldots, d.$$

This reveals that tensors of rank 1 are closely related to the concept of product form solutions.

The approximation of stationary distributions with tensors of tensor rank $R > 1$ has been proposed by Buchholz [6], using a combination of greedy low-rank and alternating optimization schemes. Despite certain theoretical drawbacks [16], the CP decomposition has been observed to perform fairly well in practice. On the other hand, this decomposition aims at a simultaneous separation of all d processes. This ignores the topology of interactions between processes and may result in relatively high ranks.

2.1 TT decomposition / Matrix product states

Low-rank decompositions that benefit from the locality of interactions are well established in computational physics, in particular for simulating quantum systems [28, 30]. A *matrix product state* (MPS) takes the form

$$\mathcal{X}_{i_1,\ldots,i_d} = G_1(i_1) \cdot G_2(i_2) \cdots G_d(i_d), \qquad G_\mu(i_\mu) \in \mathbb{R}^{r_{\mu-1} \times r_\mu}, \tag{2}$$

where $r_0 = r_d = 1$. In the numerical analysis community, this decomposition was proposed in [23, 24] and termed *tensor train* (TT) decomposition. During the last few years, MPS/TT have been used in a wide range of applications; see [12] for a literature survey.

The decomposition (2) is closely connected to certain matricizations of $\mathcal{X}$. Let $X^{(1,\ldots,\mu)}$ denote the $(n_1 \cdots n_\mu) \times (n_{\mu+1} \cdots n_d)$ matrix obtained by reshaping the entries of $\mathcal{X}$ such that the indices $(i_1, \ldots, i_\mu)$ become column indices and $(i_{\mu+1}, \ldots, i_d)$ become row indices, sorted, again, in lexicographical order. This operation has an important interpretation for a network of d stochastic automata: Merging the automata $1, \ldots, \mu$ into one subsystem and the automata $\mu+1, \ldots, d$ into another subsystem yields a network with only two (aggregated) automata, see Figure 1. The tensor corresponding to this network has order 2 and coincides with the matrix $X^{(1,\ldots,\mu)}$. The following result is well known; we include its proof to illustrate the use of matricizations.

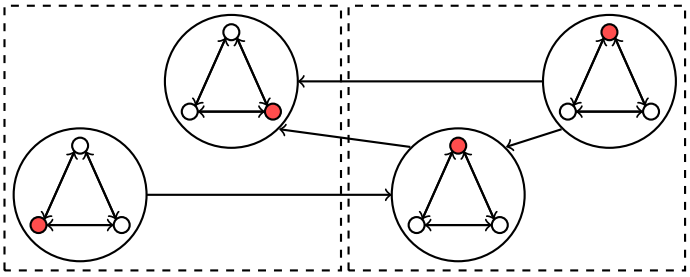

Fig. 1. Network of 4 interacting automata. The aggregation of automata into two disjoint subsystems (indicated by the dashed lines) corresponds to a matricization of the tensor.

Lemma 1. *Given a TT decomposition* (2)*, the matricization* $X^{(1,\dots,\mu)}$ *of* $\mathcal{X}$ *satisfies*

$$\operatorname{rank}\big(X^{(1,\dots,\mu)}\big) \leq r_\mu$$

for every $\mu = 1, \dots, d$.

Proof. Let us define the so called interface matrices

$$\begin{aligned} \mathbf{G}_{\leq \mu} &= \big[G_1(i_1) \cdots G_\mu(i_\mu)\big] \in \mathbb{R}^{(n_1 \cdots n_\mu) \times r_\mu} \\ \mathbf{G}_{\geq \mu+1} &= \big[G_{\mu+1}(i_{\mu+1}) \cdots G_d(i_d)\big]^T \in \mathbb{R}^{(n_{\mu+1} \cdots n_d) \times r_\mu} \end{aligned} \tag{3}$$

Then, by definition (2), we have

$$X^{(1,\dots,\mu)} = \mathbf{G}_{\leq \mu} \mathbf{G}_{\geq \mu+1}^T,$$

which implies the statement of the lemma since each of the factors has only r_μ columns. □

Motivated by Lemma 1, the tuple $\big(\operatorname{rank}\big(X^{(1,\dots,\mu)}\big)\big)_{\mu=1,\dots,d-1}$ is called the *TT rank* of $\mathcal{X}$. Following the proof of Lemma 1, successive low-rank factorizations can be used to compute the TT decomposition (2) of a given tensor with TT rank $r = (r_1, \dots, r_{d-1})$. More importantly, by using truncated SVD, we can truncate any tensor $\mathcal{X}$ to a tensor $\mathcal{X}_r$ of rank r verifying

$$\|\mathcal{X} - \mathcal{X}_r\|_2^2 \leq \sum_{\mu=1}^{d-1} \sum_{j=r_{\mu+1}}^{n_\mu} \sigma_j\big(X^{(1,\dots,\mu)}\big)^2, \tag{4}$$

where the 2-norm of a tensor is defined via its vectorization and $\sigma_j(\cdot)$ denotes the jth largest singular value of a matrix. In our algorithms, we will use a similar procedure to truncate a tensor in TT decomposition to lower TT rank; see [23] for more details. Properly implemented, this procedure takes $O(dNR^3)$ operations, where R and N are upper bounds on the ranks and the sizes of the tensor $\mathcal{X}$, respectively.

2.2 Example

One fundamental assumption in all low-rank techniques is that the data (in our case, the stationary probability distribution) can actually be well approximated by a low-rank tensor. According to (4), this can be quantified by considering the singular values of the matricizations – good accuracy can only be expected when these singular values decay sufficiently fast.

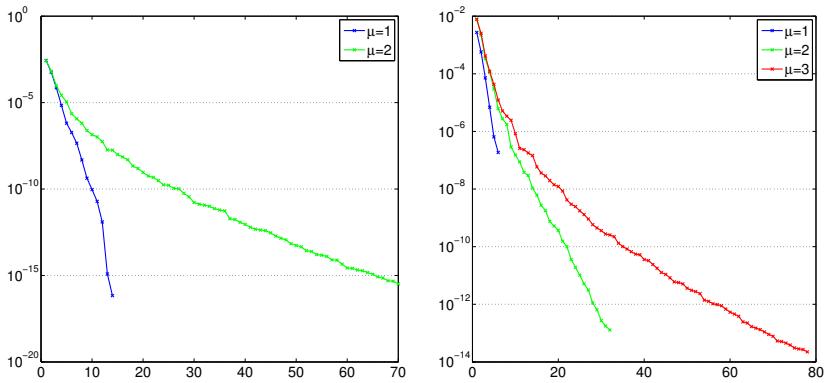

Fig. 2. Singular values of $X^{(1,\dots,\mu)}$ for the stationary distribution of the large overflow model, for $d = 4$ (left plot) and $d = 6$ (right plot).

Figure 2 displays the singular values for the relevant matricizations of the large overflow model, one of the queuing networks studied in Section 4. We consider both cases $d = 4$ with $n_\mu = 20$ states for each queue; and $d = 6$ with $n_\mu = 6$ for each queue. Note that only the first half of the matricizations are considered, as the singular values of the second half display a similar behavior. It turns out that the singular values have a very fast decay, showing that this model can be well approximated with very low TT ranks.

2.3 Operator TT decomposition / matrix product operator

Our algorithms will require the repeated application of the transition matrix $\mathbf{Q} \in \mathbb{R}^{(n_1 \cdots n_d) \times (n_1 \cdots n_d)}$ to a vector in TT decomposition. It is therefore important to represent $\mathbf{Q}$ in a form that allows to perform this operation efficiently. In principle, a sum of T Kronecker products

$$\mathbf{Q} = \sum_{i=1}^{T} \bigotimes_{\mu=1}^{d} Q_\mu^{(i)} \tag{5}$$

would be suitable for this purpose. However, the application of such a $\mathbf{Q}$ will increase all TT ranks by the factor T. Taking into account that low TT ranks are

decisive to attain efficiency, we prefer to work with a more suitable representation of $\mathbf{Q}$.

An *operator TT decomposition*, also called *matrix product operator* (MPO), takes the form

$$\mathbf{Q}_{(i_1,\ldots,i_d),(j_1,\ldots,j_d)} = A_1(i_1,j_1)\cdot A_2(i_2,j_2)\cdots A_d(i_d,j_d), \tag{6}$$

where $A_\mu(i_\mu,j_\mu)\in\mathbb{R}^{t_{\mu-1}\times t_\mu}$, and $t_0=t_d=1$. This is simply the TT decomposition (2) applied to $\mathrm{vec}(\mathbf{Q})$, merging each index pair (i_μ,j_μ) in lexicographical order into a single index ranging from 1 to n_μ^2. The tensor $\widetilde{\mathcal{X}}$ resulting from a matrix-vector product with $\mathbf{Q}$, $\mathrm{vec}(\widetilde{\mathcal{X}})=\mathbf{Q}^T\mathrm{vec}(\mathcal{X})$, has a simple TT decomposition. The updated cores $\widetilde{G}_\mu$ are given by

$$\widetilde{G}_\mu(i_\mu)=\sum_{j_\mu=1}^{n_\mu} A_\mu(j_\mu,i_\mu)\otimes G_\mu(j_\mu)\in\mathbb{R}^{r_{\mu-1}t_{\mu-1}\times r_\mu t_\mu},\quad i_\mu=1,\ldots,n_\mu, \tag{7}$$

which shows that the TT ranks multiply.

The advantage of (6) over (5) is that the ranks t_μ are often much lower than T, especially in the case of pairwise neighbor interactions. To see this, let us consider a typical example. The transition rate matrix has the general representation

$$\mathbf{Q}=\mathbf{Q}_L+\mathbf{Q}_I,$$

with $\mathbf{Q}_L$ representing local transitions and $\mathbf{Q}_I$ representing interactions. The local part always takes the form

$$\mathbf{Q}_L=\sum_{\mu=1}^{d} I_{n_1}\otimes\cdots\otimes I_{n_{\mu-1}}\otimes L_\mu\otimes I_{n_{\mu+1}}\otimes\cdots\otimes I_{n_d},$$

where $L_\mu\in\mathbb{R}^{n_\mu\times n_\mu}$ contains the local transitions in the μth system. For the large overflow model considered in Section 4.1, L_μ contains arrival and departure rates. In this example, the matrix $\mathbf{Q}_I$ is given by

$$\sum_{1\le\mu_1<\mu_2\le d} I_d\otimes\cdots\otimes I_{\mu_2+1}\otimes B_{\mu_2}\otimes C_{\mu_2-1}\otimes\cdots\otimes C_{\mu_1+1}\otimes D_{\mu_1}\otimes I_{\mu_1-1}\otimes\cdots\otimes I_1$$

for some matrices $B_\mu,C_\mu,D_\mu\in\mathbb{R}^{n_\mu\times n_\mu}$. Hence, the CP-like decomposition (5) of the operator $\mathbf{Q}$ requires $T=\frac{d(d+1)}{2}$ terms. On the other hand, by direct calculation, it can be shown that $\mathbf{Q}$ admits an operator TT decomposition (6) with the cores

$$A_1(i_1,j_1)=\begin{bmatrix}L_1(i_1,j_1) & B_1(i_1,j_1) & I_1(i_1,j_1)\end{bmatrix},\quad A_d(i_d,j_d)=\begin{bmatrix}I_d(i_d,j_d)\\ D_d(i_d,j_d)\\ L_d(i_d,j_d)\end{bmatrix},$$

and

$$A_\mu(i_\mu,j_\mu)=\begin{bmatrix}I_\mu(i_\mu,j_\mu) & 0 & 0\\ C_\mu(i_\mu,j_\mu) & B_\mu(i_\mu,j_\mu) & 0\\ L_\mu(i_\mu,j_\mu) & D_\mu(i_\mu,j_\mu) & I_\mu(i_\mu,j_\mu)\end{bmatrix},\quad \mu=2,\ldots,d-1.$$

In particular, the corresponding TT ranks t_μ are 3, for *any* value of d.

3 Low-rank algorithms

In terms of the transition rate matrix $\mathbf{Q}$, the computation of the stationary distribution $\mathbf{x}$ requires the solution of the problem

$$\mathbf{Q}^T\mathbf{x} = 0, \quad \mathbf{e}^T\mathbf{x} = 1, \tag{8}$$

where $\mathbf{e}$ is the vector of all ones. Clearly, a solution of (8) also solves the equivalent constrained least-squares problem

$$\min_{\mathbf{x}}\{\|\mathbf{Q}^T\mathbf{x}\|_2^2 : \mathbf{e}^T\mathbf{x} = 1\}. \tag{9}$$

Remark 1. For an irreducible ergodic Markov Chain, each of the problems (8) and (9) admits a unique solution [26, Ch. 4]. Reducible Markov Chains appear quite frequently, e.g., this is the case for the Kanban control model considered in Section 4.2. In this example, the states are combinations from 0 to the maximum capacity of the queue for all types of customers. Some states are not well-defined due to restrictions on the total number of customers. To address this, one can eliminate combinations not verifying the restriction by ordering and filtering the states in a specific way described in [5].

3.1 Truncated power method

A time discretization with step size $\Delta t > 0$ results in the matrix

$$\mathbf{P} = I + \Delta t\mathbf{Q},$$

and (8) becomes equivalent to the eigenvalue problem

$$\mathbf{P}^T\mathbf{x} = \mathbf{x}, \quad \mathbf{e}^T\mathbf{x} = 1. \tag{10}$$

As explained in [29, Ch. 1], $\Delta t > 0$ needs to be chosen sufficiently small to ensure that $\mathbf{P}$ is a non-periodic stochastic matrix. More specifically, it is required that

$$0 < \Delta t < (\max_i |q_{ii}|)^{-1}.$$

In our experiments, we chose $\Delta t = 0.9999 \times (\max_i |q_{ii}|)^{-1}$. Note that $\mathbf{Q}$ is only given implicitly, in terms of the low-rank Kronecker representations (5) or (6). For large values of d, it may not be feasible to evaluate all diagonal entries of $\mathbf{Q}$. In such cases, we use an upper bound on $\max_i |q_{ii}|$ instead. For example, when a Kronecker product representation of the form (5) is available, an inexpensive upper bound is given by

$$\prod_{\mu=1}^{d} \left|\left[Q_\mu^{(1)}\right]_{ii}\right| + \cdots + \prod_{\mu=1}^{d} \left|\left[Q_\mu^{(T)}\right]_{ii}\right|.$$

Originally proposed in [3], in combination with the CP decomposition, the truncated power method is probably the simplest low-rank tensor method for

solving the eigenvalue problem (10). Starting with a random vector $\mathbf{x}_0$ of rank 1, the jth iteration consists of computing

$$\widetilde{\mathbf{x}}_{j+1} = \mathsf{trunc}\big(\mathbf{x}_j + \mathsf{trunc}\big(\Delta t \mathbf{Q}^T \mathbf{x}_j\big)\big), \quad \mathbf{x}_{j+1} = \widetilde{\mathbf{x}}_{j+1}/(\mathbf{e}^T \widetilde{\mathbf{x}}_{j+1}).$$

Note that all iterates $\mathbf{x}_j$ are represented in terms of their TT decomposition. Since both the application of $\mathbf{Q}$ and the addition of two tensors increase the TT ranks, it is mandatory to repeatedly use the operation trunc, which truncates the tensor back to lower TT ranks within a specified tolerance; see also Section 2.1. This truncation destroys the property $\mathbf{e}^T \widetilde{\mathbf{x}}_{j+1} = 1$, which would otherwise be preserved by the power method. The required inner product $\mathbf{e}^T \widetilde{\mathbf{x}}_{j+1}$ to restore this normalization is inexpensive since $\mathbf{e}$ corresponds to a tensor of rank 1 [23].

3.2 Alternating least-squares (ALS)

A rather different approach consists of constraining the optimization problem (8) further to the manifold $\mathcal{M}_r$ of tensors having fixed TT ranks $r = (r_1, \ldots, r_{\mu-1})$:

$$\min_{\mathbf{x} \in \mathcal{M}_r} \{ \|\mathbf{Q}^T \mathbf{x}\|_2^2 : \mathbf{e}^T \mathbf{x} = 1 \}. \tag{11}$$

Due to the nonlinear nature of $\mathcal{M}_r$, the solution of this optimization problem is by no means simple. On the other hand, each individual core $G_\mu(\cdot)$ enters the TT decomposition (2) linearly and therefore the optimization with respect to a single core (while keeping all other cores fixed) should pose no problem. To discuss the resulting alternating least-squares (ALS) method, we require some additional notation.

Recalling the definition (3) of interface matrices for a given TT decomposition, let

$$\mathbf{G}_{\neq \mu} = \mathbf{G}_{\leq \mu-1} \otimes I_{n_\mu} \otimes \mathbf{G}_{\geq \mu+1}.$$

Then

$$\mathbf{x} = \mathbf{G}_{\neq \mu} \cdot \mathbf{g}_\mu, \quad \text{where} \quad \mathbf{g}_\mu = \operatorname{vec}\big(\big[G_\mu(1), \ldots, G_\mu(n_\mu)\big]\big) \in \mathbb{R}^{n_\mu r_{\mu-1} r_\mu}.$$

We additionally assume that the columns of $\mathbf{G}_{\leq \mu-1}$ and $\mathbf{G}_{\geq \mu+1}$ are orthonormal, which can always be achieved by the orthogonalization procedure described in [23]. In turn, the optimization of the μth core in ALS is performed by minimizing

$$\|\mathbf{Q}^T \mathbf{x}\|_2^2 = \big\|\mathbf{Q}^T \big(\mathbf{G}_{\neq \mu} \cdot \mathbf{g}_\mu\big)\big\|_2^2 = \mathbf{g}_\mu^T \big(\mathbf{G}_{\neq \mu}^T \mathbf{Q} \mathbf{Q}^T \mathbf{G}_{\neq \mu}\big) \mathbf{g}_\mu$$

among all $\mathbf{g}_\mu$ satisfying

$$\mathbf{e}^T \mathbf{G}_{\neq \mu} \mathbf{g}_\mu = (\mathbf{G}_{\neq \mu}^T \mathbf{e})^T \mathbf{g}_\mu = 1.$$

This problem can be addressed by solving the linear system

$$\begin{bmatrix} \mathbf{G}_{\neq \mu}^T \mathbf{Q} \mathbf{Q}^T \mathbf{G}_{\neq \mu} & \widetilde{\mathbf{e}} \\ \widetilde{\mathbf{e}}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{g}_\mu \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \tag{12}$$

where $\widetilde{\mathbf{e}} = \mathbf{G}_{\neq\mu}^T\mathbf{e}$ is computed by performing contractions.

After (12) has been solved, the μth core of $\mathbf{x}$ is updated with $\mathbf{g}_\mu$. The ALS method continues with a left-orthonormalization of this core, followed by the optimization of the $(\mu+1)$th core. A half sweep of ALS consists of processing all cores from the left to the right until reaching $\mu = d$. Similarly, the second half sweep of ALS consists of processing all cores from the right to the left until reaching $\mu = 1$. Two subsequent half sweeps constitute a full sweep of ALS.

Remark 2. For small ranks, the reduced problem (12) is solved by explicitly forming the matrix $\mathbf{L} = \mathbf{G}_{\neq\mu}^T\mathbf{Q}\mathbf{Q}^T\mathbf{G}_{\neq\mu}$ of size $n_\mu r_{\mu-1} r_\mu$ and utilizing a direct solver. When this becomes infeasible due to large TT ranks, we should resort to an iterative solver. As explained in [17, Sec. 3.3] the matrix $\mathbf{L}$ can be represented by a short sum of Kronecker products which greatly speeds up matrix-vector multiplications with $\mathbf{L}$. Moreover, by a standard manipulation [21, Ch. 16], the symmetric indefinite linear system (12) can be transformed into a symmetric positive definite linear system. This allows us to apply the conjugate gradient (CG) method to (12). In our experiments, we sometimes observed the CG method to suffer from slow convergence. However, the construction of effective preconditioners to accelerate convergence for such reduced problems appears to be a rather challenging task.

3.3 Local enrichment with residuals

An obvious drawback of ALS is that all TT ranks of $\mathbf{x}$ need to be chosen *a priori*. This can be addressed by starting with very low ranks, say $r_\mu \equiv 1$, and subsequently increasing the ranks by enriching the cores with additional information. For the optimization problem (11), we can mimic gradient descent by incorporating information from the residual

$$\mathbf{r} = \mathbf{Q}^T\mathbf{x}.$$

Such an approach has been suggested by White [31] for eigenvalue problems, and extended to linear systems by Dolgov and Savostyanov [7, 8].

As explained in Section 2.3, the tensor corresponding to $\mathbf{r}$ is again in TT decomposition, with the updated cores $\widetilde{G}_\mu(\cdot)$ defined in (7). The basic idea of enrichment is to augment the μth core of $\mathbf{x}$ with $\widetilde{G}_\mu(\cdot)$. However, augmenting all cores in each microstep of ALS would increase the TT ranks way too quickly and result in an inefficient algorithm. To avoid this, this procedure is modified as follows:

1. Before the enrichment, the residual is truncated to low TT ranks, within a specified accuracy.
2. In the μth microstep of the first half sweep, only the cores $G_\mu(\cdot)$ and $G_{\mu+1}(\cdot)$ are augmented. In the μth microstep of the second half sweep, only the cores $G_\mu(\cdot)$ and $G_{\mu-1}(\cdot)$ are augmented. This local enrichment benefits the next microstep while it avoids that all TT ranks increase simultaneously. The rank of the enrichment is fixed to 3.

We refrain from a more detailed discussion of these points and refer to [8, 17] for more details. Following [8], the resulting variant of ALS that allows for rank adaptation is called AMEn.

4 Numerical Experiments

We have implemented all methods described in Section 3 – truncated power method, ALS, and AMEn – in MATLAB version 2013b. These methods are based on the TT decomposition and we make use of functionality from the TT-Toolbox [22]. For reference, we have also implemented the ALS method for the CP decomposition [6], based on functionality from the tensor toolbox [1]. To distinguish between the two different ALS methods, we will denote them by ALS-TT and ALS-CP.

We nearly always used a direct solver for addressing the reduced problems (12) in ALS-TT and AMEn. Only for one example, we applied the CG method with an adaptive stopping criterion. The CG method is terminated when the residual norm has been decreased by a factor of 100 relative to the current residual norm or when the number of iterations exceeds 1000, which was frequently the case.

The truncated power method and AMEn rely on repeated low-rank truncations to prevent excessive rank growth. If not stated otherwise, the tolerance for performing this truncation is set adaptively to $\|\mathbf{r}\|_2$ for the power method and to $\|\mathbf{r}\|_2/100$ for AMEn. The TT ranks used in ALS-TT are all set to the same value, the maximum of the TT ranks of the approximate solution produced by AMEn. In ALS-CP, we extensively tried different tensor ranks, choosing the one exhibiting the best performance.

All computations were performed on a 12-core Intel Xeon CPU X5675 3.07 GHz with 192 GB RAM, under 64-Bit Linux version 2.6.32.

4.1 Large overflow model

This model was used to test ALS-CP in [6]. It consists of a queuing network where customers can arrive in each of $d = 6$ ordered queues according to independent Poisson processes and, in case the queue of arrival is full, they get blocked and try to enter the next one, until finding one that is not full. If all queues turn out to be full, the customer gets lost. The corresponding interactions are thus associated with functional transitions, where the arrival rates of the queues depend on the state of the previous ones (being full or not). Services follow an exponential distribution with mean 1. The arrival rates for queues 1 to 6 are given by 1.2, 1.1, 1.0, 0.9, 0.8 and 0.7, respectively. Choosing a maximum capacity of 10 for each queue leads to a total of $11^6 = 1771561$ states. As the capacity of each system is not infinite and the arrival rates depend on the states of other queues, this model does not constitute a Jackson network [13] and therefore the product-form approach cannot be applied.

Table 1. Obtained execution times (in seconds) and corresponding ranks for the large overflow model with respect to different accuracies.

Accuracy	1×10^{-5}		4.6×10^{-6}		2.2×10^{-6}		1×10^{-6}	
	time	rank	time	rank	time	rank	time	rank
ALS-CP	169.0	120	430.8	180	1330.3	240	3341.4	300
Power method	35.9	12.03	47.5	13.67	61.6	15.09	78.8	17.32
ALS-TT	10.7	13.92	11.4	13.92	22.8	15.67	35.9	16.84
AMEn	4.9	12.96	4.9	12.96	9.3	14.85	12.5	15.85
AMEn (CG)	266.5	11.65	–		–		–	

We have applied ALS-CP to the large overflow model for tensor ranks ranging from 120 to 300. The obtained accuracy (in terms of the residual norm) is displayed in the first row of Table 1. We then iterate the truncated power method, ALS-TT, and AMEn until the same accuracy is reached. The obtained results are shown in rows 4–7 of Table 1. For TT decompositions, the rank refers to the effective rank r_{eff}. Following [27], r_{eff} is determined as the solution of the equation

$$\text{memory}(r_1, \ldots, r_{d-1}) = \text{memory}(r_{\mathsf{eff}}, \ldots, r_{\mathsf{eff}}),$$

where $\text{memory}(r_1, \ldots, r_{d-1})$ is the amount of storage needed by a TT decomposition with TT ranks $r_1, \ldots, r_{d-1}$. All algorithms attained the target accuracy except for AMEn (CG), which refers to AMEn with the reduced problems solved by the CG method. In this case, convergence stagnated and AMEn was stopped after 10 full sweeps, reaching an accuracy of 3.622×10^{-5}. Due to the lack of an effective preconditioner for CG, the reduced problems could not be solved to sufficient accuracy. This renders the use of the CG method unattractive and we therefore not consider it in the following experiments.

The execution times of ALS-CP are much higher compared to the times of the TT-based algorithms. This is mainly due to the fact that the number of terms in the Kronecker product representation (5) of the operator is $T = \frac{d(d+1)}{2} = 21$, while the TT ranks of the operator TT decomposition (6) are 3, see Section 2.3.

Among the TT-based algorithms, AMEn is clearly the best and the power method appears to offer the poorest performance. This picture changes, however, when demanding higher accuracies. This results in higher TT ranks and consequently makes the solution of the reduced problems in AMEn and ALS-TT more expensive. For example, when demanding an accuracy of 10^{-8}, the truncated power method requires 275 seconds, ALS-TT 1120 seconds, and AMEn 367 seconds.

The left plot of Figure 3 shows how the residual norm evolves for AMEn and ALS-TT during the microiterations. Note that one microiteration corresponds to the optimization of one core in the TT decomposition. Not surprisingly, ALS-TT converges faster as it uses the maximal TT ranks right from the first sweep. This picture changes significantly when considering the evolution with respect to the

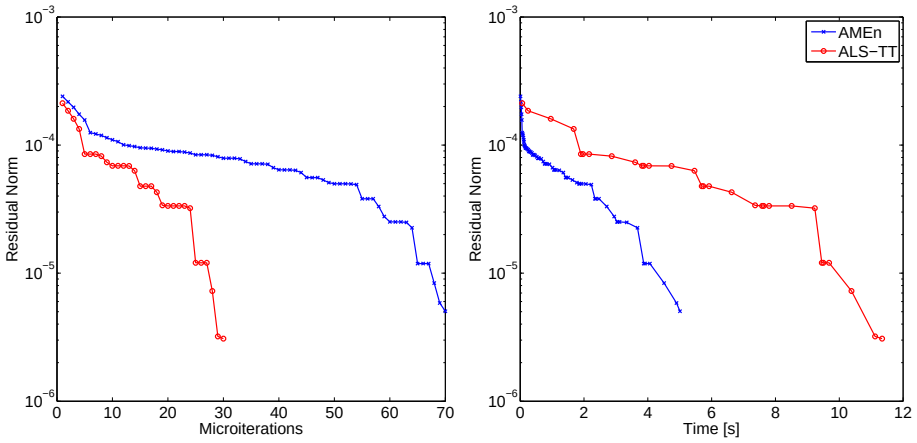

Fig. 3. Evolution of the residual norm with respect to the number of microiterations and with respect to the accumulated execution time (in seconds) for ALS-TT and AMEn applied to the large overflow model.

accumulated execution time in the right plot of Figure 3. AMEn operates with much smaller TT ranks during the first sweeps, making them less expensive and resulting in a smaller total execution time despite the fact that the total number of sweeps is higher.

Reduced arrival rates. We now divide the arrival rates by 2. This significantly decreases the execution times. To attain an accuracy of about 10^{-8}, the truncated power method now requires only 14 seconds, ALS-TT requires 11 seconds, and AMEn 4.9 seconds. With 32 seconds, ALS-CP is still the slowest method.

Exploring AMEn. Since the experiments above clearly reveal the advantages of AMEn, we investigate its performance for high-dimensional problems in more detail. For this purpose, we reduce the maximum capacity to 2 customers in each queue, that is $n_\mu \equiv 3$, and target an accuracy of 10^{-5}. We vary the number of queues from $d = 7$ to $d = 24$. Service rates are 1 for all queues while the arrival rates have been adjusted to $\frac{12-0.1\times i}{8}$ for the ith queue. To avoid convergence problems, the tolerance for performing low-rank truncations has been decreased to $\|\mathbf{r}\|_2/1000$.

Figure 4 reveals how the execution time and the effective TT rank grow as d increases. The ranks appear to grow less than linearly, while the times seem to grow proportionally with d^4. This is due to the need for solving the reduced problems, which require $O(r^6)$ operations when using a direct solver. Note that the largest Markov chain is associated with a total of $3^{24} \approx 2.82 \times 10^{11}$ states, which is clearly infeasible for standard solvers. In contrast, AMEn requires less than 2000 seconds to obtain a good approximation of the solution.

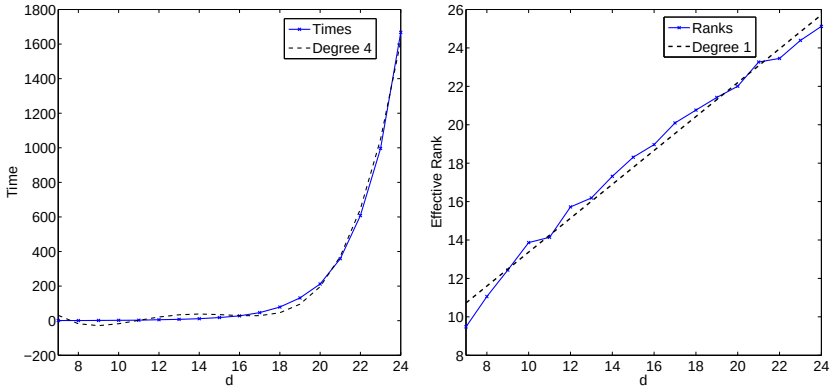

Fig. 4. Execution time in seconds (left plot) and effective rank (right plot) for AMEn applied to the large overflow model with $d = 7, 8, \ldots, 24$ queues.

4.2 Kanban control model

We now consider the Kanban control model [5], where customers arrive in the first queue, being then served in sequence until the last queue, leaving the network afterwards. We assume an infinite source – there is automatically a new arrival when the first queue is not full. A customer that finishes the service and experiences that the next queue is full needs to wait in the current queue.

We choose $d = 12$ queues, each with a maximum capacity of one customer. Services follow an exponential distribution with mean 1. The time spent traveling from one queue to the next is also exponential, with expected value $\frac{1}{10}$. For the queues 2 to 11, one needs to distinguish the type of customer (already served, waiting to move to the next queue, or no customer), so that there are 3 possible states. We therefore have, in total, $2 \times 3^{10} \times 2 = 236196$ states. The resulting matrix operator has TT rank 4 and CP rank $3d - 2 = 34$. The tolerance for performing low-rank truncations is again decreased to $\|\mathbf{r}\|_2/1000$ for AMEn.

Table 2. Obtained execution times (in seconds) and corresponding ranks for the Kanban control model.

	Accuracy	time	rank
ALS-CP	2×10^{-3}	3024.7	200
Power method	1×10^{-5}	314.5	20.95
ALS-TT	1×10^{-5}	624.9	39.51
AMEn	1×10^{-5}	200.8	28.02

Table 2 shows that the execution times for the Kanban control model are clearly higher than the ones for the large overflow model, see Table 1, despite the fact that there is a smaller total number of states. For ALS-CP, we needed to stop the algorithm before reaching the target accuracy of 10^{-5}, to avoid an excessive consumption of computational resources. In contrast to the large overflow model, this model features interactions between non-consecutive queues, as a customer has to go through all queues before leaving the system. This non-neighbor interaction can be expected to lead to the observed higher TT ranks. The largest TT rank of the approximate solution for this model is 47, while the TT ranks for the large overflow model never exceed 30, even for $d = 24$.

5 Conclusions and future work

We have proposed three algorithms for approximating stationary distributions by low-rank TT decompositions: the truncated power method, ALS and AMEn. Preliminary numerical experiments with stochastic automata networks that feature a fairly simple topology of interactions demonstrate that these methods, in particular AMEn, can perform remarkably well for very high-dimensional problems. They clearly outperform an existing approach based on CP decompositions.

Having obtained approximate stationary distributions in low-rank TT decomposition, it is comparably cheap to compute statistics of the solution. For example, the marginal probabilities of a particular queue are obtained by partial contractions with the vector of all ones, involving a cost that is negligible compared to the rest of the computation.

A bottleneck of ALS-TT and AMEn is that they use a direct solver for the reduced problems, which becomes rather expensive for larger TT ranks. Our future work will therefore focus on the further development of preconditioned iterative methods to address this issue. Being able to deal with larger TT ranks will then also allow us to study networks with a more complicated topology of interactions.

While our algorithms are designed to guarantee that the entries of the approximate solution sum up to one, the preservation of nonnegativity is another crucial aspect that remains to be addressed.

Acknowledgments. We thank Peter Buchholz for helpful discussions.

References

1. B. W. Bader, T. G. Kolda, et al. Matlab tensor toolbox version 2.5. Available from `http://www.sandia.gov/~tgkolda/TensorToolbox/`, January 2012.
2. R. Barrett, M. Berry, T. F. Chan, J. W. Demmel, J. Donato, J. J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM, Philadelphia, PA, 1994.

3. G. Beylkin and M. J. Mohlenkamp. Algorithms for numerical analysis in high dimensions. *SIAM J. Sci. Comput.*, 26(6):2133–2159, 2005.
4. P. Buchholz. A framework for the hierarchical analysis of discrete event dynamic systems, 1996. Habilitationsschrift, Fachbereich Informatik, Universität Dortmund.
5. P. Buchholz. An adaptive decomposition approach for the analysis of stochastic petri nets. In *Proceedings of the 2002 International Conference on Dependable Systems and Networks*, DSN '02, pages 647–656, Washington, DC, USA, 2002. IEEE Computer Society.
6. P. Buchholz. Product form approximations for communicating Markov processes. *Performance Evaluation*, 67(9):797–815, 2010.
7. S. V. Dolgov and D. V. Savostyanov. Alternating minimal energy methods for linear systems in higher dimensions. Part I: SPD systems. *arXiv preprint arXiv:1301.6068*, 2013.
8. S. V. Dolgov and D. V. Savostyanov. Alternating minimal energy methods for linear systems in higher dimensions. Part II: Faster algorithm and application to nonsymmetric systems. *arXiv preprint arXiv:1304.1222*, 2013.
9. J. M. Fourneau. Product form steady-state distribution for stochastic automata networks with domino synchronizations. In Nigel Thomas and Carlos Juiz, editors, *EPEW*, volume 5261 of *Lecture Notes in Computer Science*, pages 110–124. Springer, 2008.
10. J. M. Fourneau, B. Plateau, and W. J. Stewart. An algebraic condition for product form in stochastic automata networks without synchronizations. *Perform. Eval.*, 65(11-12):854–868, November 2008.
11. G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.
12. L. Grasedyck, D. Kressner, and C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitt.*, 36(1):53–78, 2013.
13. J. R. Jackson. Jobshop-like queueing systems. *Manage. Sci.*, 50(12 Supplement):1796–1802, December 2004.
14. T. H. Johnson, S. R. Clark, and D. Jaksch. Dynamical simulations of classical stochastic systems using matrix product states. *Phys. Rev. E*, 82:036702, Sep 2010.
15. V. Kazeev, M. Khammash, M. Nip, and C. Schwab. Direct solution of the chemical master equation using quantized tensor trains. Technical Report 2013-04, Seminar for Applied Mathematics, ETH Zürich, 2013.
16. T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.
17. D. Kressner, M. Steinlechner, and A. Uschmajew. Low-rank tensor methods with subspace correction for symmetric eigenvalue problems. MATHICSE preprint 40.2013, EPF Lausanne, Switzerland, 2013.
18. V. G. Kulkarni. *Introduction to modeling and analysis of stochastic systems*. Springer Texts in Statistics. Springer, New York, second edition, 2011.
19. A. N. Langville and W. J. Stewart. The Kronecker product and stochastic automata networks. *J. Comput. Appl. Math.*, 167(2):429–447, 2004.
20. A. N. Langville and W. J. Stewart. A Kronecker product approximate preconditioner for SANs. *Numer. Linear Algebra Appl.*, 11(8-9):723–752, 2004.
21. J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 2nd edition, 2006.
22. I. V. Oseledets. MATLAB TT-Toolbox Version 2.2, May 2011. Available at `http://spring.inm.ras.ru/osel/?page_id=24`.

23. I. V. Oseledets. Tensor-train decomposition. *SIAM J. Sci. Comput.*, 33(5):2295–2317, 2011.
24. I. V. Oseledets and E. E. Tyrtyshnikov. Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM J. Sci. Comput.*, 31(5):3744–3759, 2009.
25. B. Plateau, J.-M. Fourneau, and K.-H. Lee. PEPS: A package for solving complex Markov models of parallel systems. In R. Puigjaner and D. Potier, editors, *Modeling Techniques and Tools for Computer Performance Evaluation*, pages 291–305. Springer US, 1989.
26. S. M. Ross. *Introduction to probability models*. Harcourt/Academic Press, Burlington, MA, seventh edition, 2000.
27. D. V. Savostyanov. QTT-rank-one vectors with QTT-rank-one and full-rank Fourier images. *Linear Algebra Appl.*, 436(9):3215–3224, 2012.
28. U. Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Ann. Physics*, 326:96–192, 2011.
29. W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, Princeton, NJ, 1994.
30. S. R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, 1992.
31. S. R. White. Density matrix renormalization group algorithms with a single center site. *Phys. Rev. B*, 72:180403, Nov 2005.

Recent publications:

MATHEMATICS INSTITUTE OF COMPUTATIONAL SCIENCE AND ENGINEERING
Section of Mathematics
Ecole Polytechnique Fédérale
CH-1015 Lausanne

04.2014 TAKAHITO KASHIWABARA, CLAUDIA M. COLCIAGO, LUCA DEDÈ, ALFIO QUARTERONI:
Numerical Well-posedness, regularity, and convergence analysis of the finite element approximation of a generalized Robin boundary value problem

05.2014 BJÖRN ADLERBORN, BO KAGSTRÖM, DANIEL KRESSNER:
A parallel QZ algorithm for distributed memory HPC systems

06.2014 MICHELE BENZI, SIMONE DEPARIS, GWENOL GRANDPERRIN, ALFIO QUARTERONI:
Parameter estimates for the relaxed dimensional factorization preconditioner and application to hemodynamics

07.2014 ASSYR ABDULLE, YUN BAI:
Reduced order modelling numerical homogenization

08.2014 ANDREA MANZONI, FEDERICO NEGRI:
Rigorous and heuristic strategies for the apporximation of stability factors in nonlinear parametrized PDEs

09.2014 PENG CHEN, ALFIO QUARTERONI:
A new algorithm for high-dimensional uncertainty quantification problems based on dimension-adaptive and reduced basis methods

10.2014 NATHAN COLLIER, ABDUL-LATEEF HAJI-ALI, FABIO NOBILE, ERIK VON SCHWERIN, RAÚL TEMPONE:
A continuation multilevel Monte Carlo algorithm

11.2014 LUKA GRUBISIC, DANIEL KRESSNER:
On the eigenvalue decay of solutions to operator Lyapunov equations

12.2014 FABIO NOBILE, LORENZO TAMELLINI, RAÚL TEMPONE:
Convergence of quasi-optimal sparse grid approximation of Hilbert-valued functions: application to random elliptic PDEs

13.2014 REINHOLD SCHNEIDER, ANDRÉ USCHMAJEW:
Convergence results for projected line-search methods on varieties of low-rank matrices via Lojasiewicz inequality

14.2014 DANIEL KRESSNER, PETAR SIRKOVIC:
Greedy law-rank methods for solving general linear matrix equations

15.2014 WISSAM HASSAN, MARCO PICASSO:
An anisotropic adaptive finite element algorithm for transonic viscous flows around a wing

16.2014 ABDUL-LATEEF HAJI-ALI, FABIO NOBILE, ERIK VON SCHWERIN, RAÚL TEMPONE:
Optimization of mesh hierarchies in multilevel Monte Carlo samplers

17.2014 DANIEL KRESSNER, FRANCISCO MACEDO:
Low-rank tensor methods for communicating Markov processes