Electronic Transactions on Numerical Analysis.
Volume 48, pp. 348–361, 2018.
Copyright © 2018, Kent State University.
ISSN 1068–9613.
DOI: 10.1553/etna_vol48s348

MULTIGRID METHODS COMBINED WITH LOW-RANK APPROXIMATION FOR TENSOR-STRUCTURED MARKOV CHAINS[*]

MATTHIAS BOLTEN[†], KARSTEN KAHL[†], DANIEL KRESSNER[‡], FRANCISCO MACEDO[‡§], AND SONJA SOKOLOVIĆ[†]

Abstract. Markov chains that describe interacting subsystems suffer from state space explosion but lead to highly structured matrices. In this work, we propose a novel tensor-based algorithm to address such tensor-structured Markov chains. Our algorithm combines a tensorized multigrid method with AMEn, an optimization-based low-rank tensor solver, for addressing coarse grid problems. Numerical experiments demonstrate that this combination overcomes the limitations incurred when using each of the two methods individually. As a consequence, Markov chain models of unprecedented size from a variety of applications can be addressed.

Key words. Multigrid method, SVD, Tensor Train format, Markov chains, singular linear system, alternating optimization

AMS subject classifications. 65F10, 65F50, 60J22, 65N55

1. Introduction.

This paper is concerned with the numerical computation of stationary distributions for large-scale continuous-time Markov chains. Mathematically, this task consists of solving the linear system

$$Ax = 0 \quad \text{with} \quad \mathbf{1}^T x = 1, \tag{1.1}$$

where A is the transposed generator matrix of the Markov chain and $\mathbf{1}$ denotes the vector of all ones. The matrix A is square, nonsymmetric, and satisfies $\mathbf{1}^T A = 0$. It is well known [4] that the irreducibility of A implies existence and uniqueness of the solution of (1.1).

We specifically consider Markov chains that describe d interacting subsystems. Assuming that the kth subsystem has n_k states, the generator matrix usually takes the form

$$A = \sum_{t=1}^{T} E_1^t \otimes E_2^t \otimes \cdots \otimes E_d^t, \tag{1.2}$$

where $\otimes$ denotes the Kronecker product and $E_k^t \in \mathbb{R}^{n_k \times n_k}$ for $k = 1, \ldots, d$. Consequently, A has size $n = n_1 n_2 \cdots n_d$, which reflects the fact that the states of the Markov chain correspond to all possible combinations of subsystem states. The exponential growth of n with respect to d is usually called state space explosion [10]. Applications of models described by (1.2) include queuing theory [11, 12, 16], stochastic automata networks [20, 28], analysis of chemical reaction networks [1, 21], and telecommunication [2, 27]. Equations of similar type arise in time stepping procedures for the chemical master equation [13, 17].

The tensor structure of (1.2) can be exploited to yield efficient matrix-vector multiplications in iterative methods for solving (1.1); see, e.g., [20]. However, none of the standard iterative solvers is computationally feasible for larger d because of their need to store vectors of length n. To a certain extent, this can be avoided by reducing each n_k with the tensorized

[*] Received May 18, 2016. Accepted April 19, 2018. Published online on Oktober 8, 2018. Recommended by Raf Vandebril.

[†] Fakultät für Mathematik und Naturwissenschaften, Bergische Universität Wuppertal, 42097 Wuppertal, Germany ({bolten,kkahl,sokolovic}@math.uni-wuppertal.de).

[‡] EPF Lausanne, SB-MATHICSE-ANCHP, Station 8, CH-1015 Lausanne, Switzerland ({daniel.kressner,francisco.macedo}@epfl.ch).

[§] IST, Alameda Campus, Av. Rovisco Pais, 1, 1049-001 Lisbon, Portugal.

multigrid method recently proposed in [5]. Still, the need for solving coarse subproblems of size 2^d or 3^d limits such an approach to modest values of d.

Low-rank tensor methods as proposed in [9, 18] can potentially deal with large values of d. The main idea is to view the solution x of (1.2) as an $n_1 \times n_2 \times \cdots \times n_d$ tensor and aim at an approximation in a highly compressed, low-rank tensor format. The choice of the format is crucial for the success and practicality of such an approach. In [9], the so called canonical decomposition was used, constituting a natural extension of the concept of product form solutions. Since this format aims at separating all subsystems at the same time, it cannot benefit from an underlying topology and thus often results in relatively large ranks. In contrast, low-rank formats based on tensor networks can be aligned with the topology of interactions between subsystems. In particular, it was demonstrated in [18] that the so called tensor train format [25] appears to be well suited. Alternating optimization techniques are frequently used to obtain approximate solutions within a low-rank tensor format. Specifically, [18] proposes a variant of the Alternating Minimal Energy method (AMEn) [14, 34]. In each step of alternating optimization, a subproblem of the form (1.1) needs to be solved. This turns out to be challenging, although these subproblems are much smaller than the original problem, they are often too large to allow for the solution by a direct method and too ill-conditioned to allow for the solution by an iterative method. It is not known how to design effective preconditioners for such problems.

In this paper, we combine the advantages of two methods. The tensorized multigrid method from [5] is used to reduce the mode sizes n_k and the condition number. This, in turn, benefits the use of the low-rank tensor method from [18] by reducing the size and the condition number of the subproblems. The same idea has been proposed by Ballani and Grasedyck [3] for linear systems from PDE discretizations using a different low-rank tensor format.

The rest of this paper is organized as follows. In Section 2 we briefly describe the tensor train format and explain the basic ideas of alternating least squares methods, including AMEn. The tensorized multigrid method is described in Section 3. Section 4 describes our proposed combination of the tensorized multigrid method with AMEn. In Section 5, the advantages of this combination are illustrated by a series of numerical experiments involving models from different applications.

2. Low-rank tensor methods.

A given vector $x \in \mathbb{R}^{n_1 \cdots n_d}$ is turned into a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ by setting

$$\mathcal{X}(i_1, \ldots, i_d) = x\big(i_1 + (i_2 - 1)n_1 + (i_3 - 1)n_1 n_2 + \cdots + (i_d - 1)n_1 n_2 \cdots n_{d-1}\big), \tag{2.1}$$

with $1 \leq i_k \leq n_k$ for $k = 1, \ldots, d$. In MATLAB, this corresponds to the command `X=reshape(x,n)` with `n=[n_1,n_2,...,n_d]`.

2.1. Tensor train format.

The *tensor train (TT) format* is a multilinear low-dimensional representation of a tensor. Specifically, a tensor $\mathcal{X}$ is said to be represented in TT format if each entry of the tensor is given by

$$\mathcal{X}(i_1, \ldots, i_d) = G_1(i_1) \cdot G_2(i_2) \cdots G_d(i_d). \tag{2.2}$$

The parameter-dependent matrices $G_k(i_k) \in \mathbb{R}^{r_{k-1} \times r_k}$ for $k = 1, \ldots, d$ are usually collected in $r_{k-1} \times n_k \times r_k$ tensors, which are called the *TT cores*. The integers $r_0, r_1, \ldots, r_{d-1}, r_d$, with $r_0 = r_d = 1$, determining the sizes of these matrices, are called the *TT ranks*. The complexity of storing $\mathcal{X}$ in the format (2.2) is bounded by $(d-2)\widehat{n}\widehat{r}^2 + 2\widehat{n}\widehat{r}$ if each $n_k \leq \widehat{n}$ and $r_k \leq \widehat{r}$.

For a matrix $A \in \mathbb{R}^{n_1 \cdots n_d \times n_1 \cdots n_d}$, one can define a corresponding *operator TT format* by mapping the row and column indices of A to tensor indices analogous to (2.1) and letting each

TABLE 2.1
Complexity of operations in the TT format for tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with TT ranks bounded by $\hat{r}_{\mathcal{X}}$ and $\hat{r}_{\mathcal{Y}}$, respectively, and a matrix $A \in \mathbb{R}^{(n_1 \times \cdots \times n_d) \times (n_1 \times \cdots \times n_d)}$ with operator TT ranks bounded by $\hat{r}_A$. All sizes n_k are bounded by $\hat{n}$.

Operation	Cost	Resulting TT ranks
Addition of two tensors $\mathcal{X} + \mathcal{Y}$	—	$\hat{r}_{\mathcal{X}} + \hat{r}_{\mathcal{Y}}$
Scalar multiplication $\alpha \mathcal{X}$	$\mathcal{O}(1)$	$\hat{r}_{\mathcal{X}}$
Scalar product $\langle \mathcal{X}, \mathcal{Y} \rangle$	$\mathcal{O}(d\hat{n} \max\{\hat{r}_{\mathcal{X}}, \hat{r}_{\mathcal{Y}}\}^3)$	—
Matrix-vector product $A\mathcal{X}$	$\mathcal{O}(d\hat{n}^2 \hat{r}_A^2 \hat{r}_{\mathcal{X}}^2)$	$\hat{r}_A \hat{r}_{\mathcal{X}}$
Truncation of $\mathcal{X}$	$\mathcal{O}(d\hat{n}\hat{r}_{\mathcal{X}}^3)$	prescribed

entry of A satisfy

$$A(i_1, \ldots, i_d; j_1, \ldots, j_d) = M_1(i_1, j_1) \cdot M_2(i_2, j_2) \cdots M_d(i_d, j_d), \tag{2.3}$$

with parameter-dependent matrices $M_k(i_k, j_k) \in \mathbb{R}^{r_{k-1} \times r_k}$ for $k = 1, \ldots, d$. The difference to (2.2) is that the cores now depend on two parameters instead of one. A matrix given as a sum of T Kronecker products as in (1.2) can be easily converted into an operator TT format (2.3) using, e.g., the techniques described in [23]. It holds that $r_k \leq T$ but often much smaller operator TT ranks can be achieved.

Assuming constant TT ranks, the TT format allows to perform certain elementary operations with a complexity linear (instead of exponential) in d. Table 2.1 summarizes the complexity for operations of interest, which shows that the cost can be expected to be dominated by the TT ranks. For a detailed description of the TT format and its operations, we refer to [23, 25, 26].

2.2. Alternating least squares. In this section, we describe the method of alternating least squares (ALS) from [18].

To incorporate the TT format, we first replace (1.1) by the equivalent optimization problem

$$\min \|Ax\| \qquad \text{subject to } \mathbf{1}^T x = 1, \tag{2.4}$$

where $\|\cdot\|$ denotes the Euclidean norm. We can equivalently view A as a linear operator on $\mathbb{R}^{n_1 \times \cdots \times n_d}$ and constrain (2.4) to tensors in TT format:

$$\min \|A\mathcal{X}\| \qquad \text{subject to } \langle \mathcal{X}, \mathbf{1} \rangle = 1, \ \mathcal{X} \text{ is in TT format (2.2)}, \tag{2.5}$$

where $\mathbf{1}$ now refers to the $n_1 \times \cdots \times n_d$ tensor of all ones.

Note that the TT format is linear in each of the TT cores. This motivates the use of an ALS approach that optimizes the kth TT core while keeping all other TT cores fixed. To formulate the subproblem that needs to be solved in each step of ALS, we define the interface matrices

$$G_{\leq k-1} = \big[G(i_1) \cdots G(i_k)\big] \in \mathbb{R}^{(n_1 \cdots n_k) \times r_{k-1}},$$
$$G_{\geq k+1} = \big[G(i_{k+1}) \cdots G(i_d)\big]^T \in \mathbb{R}^{(n_{k+1} \cdots n_d) \times r_k}.$$

Without loss of generality, we may assume that the TT format is chosen such that the columns of $G_{\leq k}$ and $G_{\geq k+1}$ are orthonormal; see, e.g., [19]. By letting $g_k \in \mathbb{R}^{r_{k-1} n_k r_k}$ contain the vectorization of the kth core and setting

$$G_{\neq k} = G_{\leq k-1} \otimes I_{n_k} \otimes G_{\geq k+1},$$

it follows that

$$\operatorname{vec}(\mathcal{X}) = G_{\neq k} g_k.$$

Inserting this relation into (2.5) yields

$$\min \|A G_{\neq k} g_k\| \qquad \text{subject to } \langle G_{\neq k} g_k, \mathbf{1}\rangle = 1,$$

which is equivalent to the linear system

$$\begin{bmatrix} G_{\neq k}^T A^T A G_{\neq k} & \tilde{\mathbf{e}} \\ \tilde{\mathbf{e}}^T & 0 \end{bmatrix} \begin{bmatrix} g_k \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{2.6}$$

with the Lagrange multiplier $\lambda \in \mathbb{R}$. The vector $\tilde{\mathbf{e}} = G_{\neq k}^T \mathbf{1}$ can be cheaply computed by tensor contractions. After (2.6) has been solved, the TT format of the tensor $\mathcal{X}$ is updated by reshaping g_k into its kth TT core. One full sweep of ALS consists of applying the described procedure first in a forward sweep over the TT cores $1, 2, \ldots, d$ followed by a backward sweep over the TT cores $d, d-1, \ldots, 1$. After each update of a core, an orthogonalization procedure [25] is applied to ensure the orthonormality of the interface matrices in the subsequent optimization step.

2.3. AMEn. The method AMEn proposed in [14] for linear systems enriches the TT cores locally by gradient information, which potentially yields faster convergence than ALS and allows for rank adaptivity. It is sufficient to consider $d = 2$ for illustrating the extension of this procedure to (2.5). The general case $d > 2$ then follows, analogously to [14, 19], by applying the case $d = 2$ to neighbouring cores.

For $d = 2$, the TT format corresponds to a low-rank factorization $\mathcal{X} = G_1 G_2^T$ with $G_1 \in \mathbb{R}^{n_1 \times r_1}$, $G_2 \in \mathbb{R}^{n_2 \times r_2}$. Suppose that the first step of ALS has been performed and G_1 has been optimized. We then consider a low-rank approximation of the negative gradient of $\frac{1}{2}\|A\mathcal{X}\|^2$:

$$\mathcal{R} = -A\mathcal{X} \approx R_1 R_2^T.$$

In practice, a rank-2 or rank-3 approximation of $\mathcal{R}$ is used. Then the method of steepest descent applied to minimizing $\frac{1}{2}\|A\mathcal{X}\|^2$ would compute

$$\mathcal{X} + \alpha\mathcal{R} \approx \begin{bmatrix} G_1 & R_1 \end{bmatrix} \begin{bmatrix} G_2 & \alpha R_2 \end{bmatrix}^T$$

for some suitably chosen scalar α. We now fix (and orthonormalize) the first augmented core $\begin{bmatrix} G_1 & R_1 \end{bmatrix}$. However, instead of using $\begin{bmatrix} G_2 & \alpha R_2 \end{bmatrix}$, we apply the next step of ALS to obtain an optimized second core via the solution of a linear system of the form (2.6). As a result we obtain an approximation $\mathcal{X}$ that is at least as good as the one obtained from one forward sweep of ALS without augmentation and, when ignoring the truncation error in $\mathcal{R}$, at least as good as one step of steepest descent. The described procedure is repeated by augmenting the first core and optimizing the second core and so on. In each step, the rank of $\mathcal{X}$ is adjusted by performing low-rank truncation. This rank adaptivity is one of the major advantages of AMEn.

3. Multigrid.

In this section, we recall the multigrid method from [5] for solving (1.1) with a matrix A having the tensor structure (1.2). Special care has to be taken in order to preserve the tensor structure within the multigrid hierarchy. We first introduce the generic components of a multigrid method before explaining the tensor specific construction.

A multigrid approach has the following ingredients: the smoothing scheme, the set of coarse variables, transfer operators (the interpolation operator and the restriction operator), and the coarse grid operator.

Algorithm 1: Multigrid V-cycle.

```
$v_\ell = \text{MG}(b_\ell, v_\ell)$
if coarsest grid is reached then
    Solve coarse grid equation $A_\ell v_\ell = b_\ell$
else
    Update $v_\ell$ by $\nu_1$ smoothing steps for $A_\ell v_\ell = b_\ell$
    Compute coarse right-hand side $b_{\ell+1} = Q_\ell(b_\ell - A_\ell v_\ell)$
    $e_{\ell+1} = \text{MG}(b_{\ell+1}, 0)$
    $v_\ell = v_\ell + P_\ell e_{\ell+1}$
    Update $v_\ell$ by $\nu_2$ smoothing steps for $A_\ell v_\ell = b_\ell$
end
```

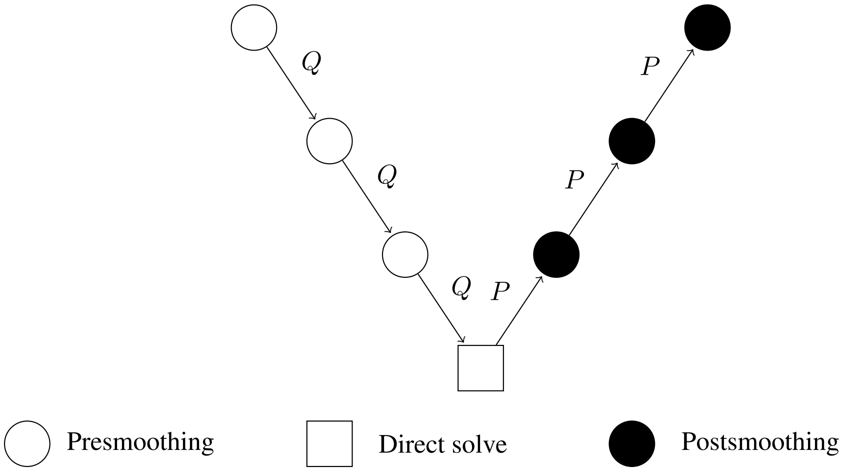


FIG. 3.1. *Multigrid V-cycle: on each level, a presmoothing iteration is performed before the problem is restricted to the next coarser grid. On the smallest grid, the problem is typically solved exactly by a direct solver. When interpolating back to the finer grids, postsmoothing iterations are applied on each level.*

Algorithm 1 is a prototype of a V-cycle and includes the mentioned ingredients. For a detailed description we refer the reader to [29, 32]. In particular, for a two-grid approach, i.e., $\ell = 1, 2$, one can describe the realization as follows: the method performs a certain number ν_1 of smoothing steps using an iterative solver that can be, for instance, a weighted Jacobi, a Gauss-Seidel, or a Krylov subspace method like GMRES [30, 31]; the residual of the current iterate is computed and restricted by a matrix-vector multiplication with the restriction matrix $Q \in \mathbb{R}^{n \times n_c}$; the operator $A_1 = A$ is restricted via a Petrov-Galerkin construction to obtain the coarse-grid operator, $A_2 = QA_1P \in \mathbb{R}^{n_c \times n_c}$, where $P \in \mathbb{R}^{n_c \times n}$ is the interpolation operator. Then we have a recursive call where we solve the coarse grid equation, which is the residual equation. Then the error is interpolated and again some smoothing iterations are applied. Instead of stopping at the second grid because the matrix may still be too large, one can approximate the solution of the residual equation again via a two-grid approach. By this recursive construction one obtains the V-cycle displayed in Figure 3.1. This V-cycle is performed repeatedly until a certain accuracy of the residual is reached or a maximum number of V-cycles have been applied.

No details have yet been provided on how to choose n_c and how to obtain the weights for the interpolation and restriction operators P and Q. The value n_c is obtained by specifying coarse variables. Geometric coarsening [32] or compatible relaxation [6, 7] are methods which split the given n variables into fine variables $\mathcal{F}$ and coarse variables $\mathcal{C}$ so that $n = |\mathcal{C}| + |\mathcal{F}|$. If such a splitting is given, $n_c = |C|$, the operators are defined as

$$Q : \mathbb{R}^{|\mathcal{C} \cup \mathcal{F}|} \to \mathbb{R}^{|\mathcal{C}|}, \quad P : \mathbb{R}^{|\mathcal{C}|} \to \mathbb{R}^{|\mathcal{C} \cup \mathcal{F}|}.$$

To obtain the entries for these operators, one can use methods like linear interpolation [32] or direct interpolation [29, 32], among others. Another approach for choosing a coarse grid is aggregation [8], where one defines a partition of the set of variables and each subset of this partition is associated with one coarse variable.

In this work we focus on the V-cycle strategy. Other strategies, for example W- or F-cycles [32], can be applied in a straightforward fashion.

3.1. Tensorized multigrid.

In order to make Algorithm 1 applicable to a tensor-structured problem, one has to ensure that the tensor structure is preserved along the multigrid hierarchy. In this, we follow the approach taken in [5] and define interpolation and restriction in the following way.

PROPOSITION 3.1. *Let the matrix A of the form (1.2) be given with $E_k^t \in \mathbb{R}^{n_k \times n_k}$. Let $P = \bigotimes_{k=1}^d P_k$ and $Q = \bigotimes_{k=1}^d Q_k$ with $P_k \in \mathbb{R}^{n_k \times n_k^c}$ and $Q_k \in \mathbb{R}^{n_k^c \times n_k}$, where $n_k^c < n_k$. Then the corresponding Petrov-Galerkin operator satisfies*

$$QAP = \sum_{t=1}^{T} \bigotimes_{k=1}^{d} Q_k E_k^t P_k.$$

Thus, the task of constructing interpolation and restriction operators becomes a "local" task, i.e., each part P_k of the interpolation $P = \bigotimes_{k=1}^d P_k$ coarsens the kth subsystem. In particular, this implies $n_k^{(c)} < n_k$, and the entries of P_k depend largely on the local part of the tensorized operator.

Another important ingredient of the multigrid method is the smoothing scheme. In our setting, it has to satisfy two main requirements: it should

(i) be applicable to non-symmetric, singular systems;
(ii) admit an efficient implementation in the TT format.

Requirement (ii) basically means that only the operations listed in Table 2.1 should be used by the smoother, as most other operations are far more expensive. In this context, one logical choice is GMRES [30, 31] (which also fulfills requirement (i)), which consists of matrix-vector products and orthogonalization steps (i.e., inner products and vector addition). Note that GMRES is a non-stationary method and so an atypical choice for a smoother in a multigrid context. However, the typically used stationary smoothers like Gauss-Seidel or Jacobi can not be implemented efficiently in a TT format; see [5]. GMRES as smoother is discussed, e.g., in [33], and was already successfully applied to tensor-structured Markov chains in [5].

Parameters of the SVD truncation. We apply the TT-SVD algorithm from [25] to keep the TT ranks of the iterates in the tensorized multigrid method under control. Except for the application of restriction and interpolation, which both have operator TT rank one by construction, all operations of Algorithm 1 lead to an increase of the rank of the current iterate.

In particular, truncation has to be performed after line 6 and line 8 of Algorithm 1. Concerning the truncation of the restricted residual in line 6, we have observed that we do not need a very strict accuracy to obtain convergence of the global scheme and thus set the value to 10^{-1}. As for the truncation of the updated iterates v_ℓ after line 8, we note that they have highly different norms on the different levels so that the accuracy for their truncation should depend on the level. Additionally, a dependency on the cycle, following the idea in [18] in which such an adaptive scheme is applied to the sweeps of AMEn, is also included. Precisely, the accuracy depends on the residual norm after the previous cycle. This is motivated by the fact that truncations should be more accurate as we get closer to the desired approximation, while this is not needed far away from it. Summarizing, the accuracy of the truncation of the different v_ℓ is thus taken as the norm of v_ℓ divided by v_1 (dependency on the level) times the residual norm after the previous cycle (dependency on the quality of the current approximate solution) times a default value of 10. This "double" adaptivity is also used within the GMRES smoother to truncate the occurring vectors.

We also impose a restriction on the maximum TT rank allowed after each truncation. This maximum rank is initially set to 15 and grows by a factor of $\sqrt{2}$ after each cycle for which the reduction of the residual norm is observed to be smaller than a factor of $\frac{9}{10}$, signalling stagnation.

4. Multigrid-AMEn.

In Sections 2 and 3 we have discussed two independent methods for solving (1.1). In this section we first discuss the limitations of these two methods and then describe a novel combination that potentially overcomes these limitations.

4.1. Limitation of AMEn.

Together with orthogonalization and low-rank truncation, one of the computationally most expensive parts of AMEn is the solution of the linear system (2.6), which has size $r_{k-1}r_kn_k + 1$. A direct solver applied to this linear system has complexity $\mathcal{O}(\hat{r}^6\hat{n}^3)$ and can thus only be used in the presence of small ranks and mode sizes.

Instead of a direct solver, an iterative solver such as MINRES [15, 30] can be applied to (2.6). The Kronecker structure of $G_{\neq k}^T A^T A G_{\neq k}$ inherited by the low operator TT rank of A allows for efficient matrix-vector multiplications despite the fact that this matrix is not sparse. Unfortunately, we have observed for all the examples considered in Section 5 that the condition number of the reduced problem (2.6) grows rapidly as the mode sizes n_k increase. In turn, the convergence of MINRES is severely impaired, often leading to stagnation. It is by no means clear whether it is possible to turn a preconditioner for the original problem into an effective preconditioner for the reduced problem. So far, this has only been achieved via a very particular construction for Laplace-like operators [19], which is not relevant for the problems under consideration.

4.2. Limitations of tensorized multigrid.

The described tensorized multigrid method is limited to modest values of d, simply because of the need for solving the problem on the coarsest grid. The size of this problem grows exponentially in d. Figure 4.1 illustrates the coarsening process if one applies full coarsening to each E_j^t in an overflow queueing problem with mode sizes 9 as described, e.g., in [5, Section 5.1]; see also Section 5.1 of this paper. In the case of three levels, a problem of size 3^d would need to be addressed by a direct solver on the coarsest grid. Due to the nature of the problem it is not possible to coarse the problem to a single variable in each dimension.

4.3. Combination of the two methods.

Instead of using a direct method for solving the coarsest-grid system in the tensorized multigrid method, we propose to use AMEn. Due to the fact that the mode sizes on the coarsest grid are small, we expect that it becomes much simpler to solve the reduced problems (2.6) within AMEn.

Note that the problem to be solved on the coarsest grid constitutes a correction equation and thus differs from the original problem (1.1) in having a nonzero right-hand side and incorporating a different linear constraint. To address this problem, we apply AMEn [14] to the normal equations and ignore the linear constraint. The linear constraint is fixed only at the end of the cycle by explicitly normalizing the obtained approximation as in [5].

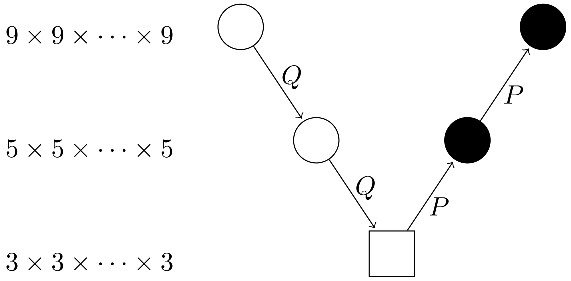

$9 \times 9 \times \cdots \times 9$
$5 \times 5 \times \cdots \times 5$
$3 \times 3 \times \cdots \times 3$
Q
Q
P
P

FIG. 4.1. *Coarsening process for a problem with mode sizes* 9.

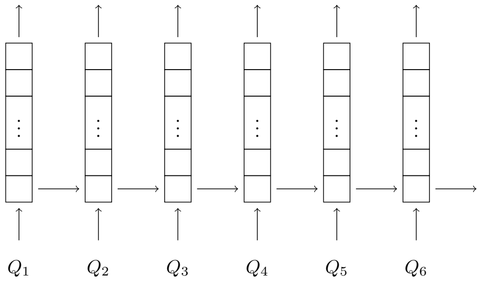


FIG. 5.1. *Structure of the model* overflow.

Parameters of AMEn for the coarsest grid problem. AMEn targets an accuracy that is at the level of the residual from the previous multigrid cycle, and we stop AMEn once this accuracy is reached or, at the latest, after 5 sweeps. A rank-3 approximation of the negative gradient, obtained by ALS as suggested in [14], is used to augment the cores within AMEn. Reduced problems (2.6) are addressed by a direct solver for size up to 1000; otherwise MINRES (without a preconditioner) is used.

Initial approximation of the solution. All algorithms are initialized with the tensor that results from solving the coarsest grid problem, using the variant of AMEn described in Section 2.3, and then bringing it up to the finest level using interpolation as in [5].

5. Numerical experiments.

In this section, we illustrate the efficiency of our newly proposed algorithm from Section 4. All tests have been performed in MATLAB version 2013b, using functions from the *TT-Toolbox* [24]. The execution times have been obtained on a 12-core Intel Xeon CPU X5675, 3.07GHz with 192 GB RAM running 64-Bit Linux version 2.6.32.

5.1. Model problems.

All benchmark problems used in this paper are taken from the benchmark collection [22], which not only provides a detailed description of the involved matrices but also MATLAB code. In total, we consider six different models, which can be grouped into three categories.

Overflow queuing models. The first class of benchmark models consists of the well-known overflow queuing model and two variations thereof. The structure of the model is depicted in Figure 5.1. The arrival rates are chosen as $\lambda_k = 1.2 - (k-1) \cdot 0.1$ and the service rates as $\mu_k = 1$ for $k = 1, \ldots, d$, as suggested in [9]. The variations of the model differ in the interaction between the queues:

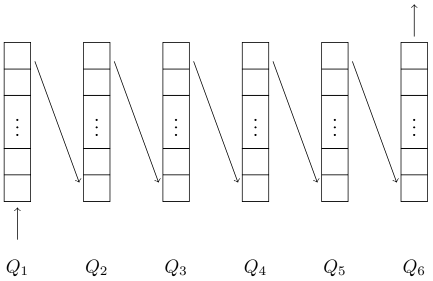


FIG. 5.2. *Structure of the model* kanbanalt2.

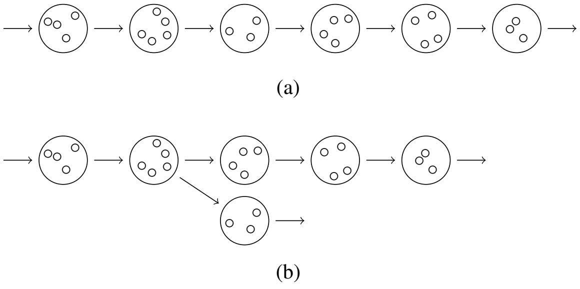


FIG. 5.3. *Structure of the models* directedmetab *(a) and* divergingmetab *(b).*

- overflow: customers which arrive at a full queue try to enter subsequent queues until they find one that is not full; after trying the last queue, they leave the system.
- overflowsim: as overflow, but customers arriving at a full queue try only one subsequent queue before leaving the system.
- overflowpersim: as overflowsim, but when the last queue is full, a customer arriving there tries to enter the first queue instead of immediately leaving.

For these models, as suggested in [5], we choose the interpolation operator P_k as direct interpolation based on the matrices describing the local subsystems and the restriction operator as its transpose.

*Simple tandem queuing network (*kanbanalt2*).* A number d of queues has to be passed through by customers one after the other. Each queue k has its own service rate, denoted by $dep(k)$, and its own capacity, denoted by $cap(k)$. For our tests we choose $dep(k) = 1$ for all $k = 1, \dots, d$. The service in queue k can only be finished if queue $k + 1$ is not full so that the served customer can immediately enter the next queue. Customers arrive only at the first queue with an arrival rate of 1.2. Figure 5.2 illustrates this model.

As only the subsystems corresponding to the first and last dimensions have a non-trivial "local part" and the one for the last dimension is associated with a subdiagonal matrix, we construct only P_1 via direct interpolation (as in the overflow models) and use linear interpolation for $P_2, \dots, P_d$.

Metabolic pathways. The next model problems we consider come from the field of chemistry describing stochastic fluctuations in metabolic pathways. In Figure 5.3(a) each node of the given graph describes a metabolite. A flux of substrates can move along the nodes being converted by means of several chemical reactions (an edge between node k and ℓ in the graph means that the product of reaction k can be converted further by reaction ℓ). The rate at which

TABLE 5.1
Execution time (in seconds), number of iterations, and maximum rank of the computed approximations for overflow *with mode size 17 and varying dimension d. The symbol — indicates that the desired accuracy could not be reached within* 3 600 *seconds.*

	AMEn			Multigrid			MultigridAMEn		
d	time	iter	rank	time	iter	rank	time	iter	rank
4	4.5	7	16	4.6	13	13	4.2	13	13
5	36.3	9	23	6.4	11	20	7.0	11	20
6	239.4	12	28	24.7	17	29	20.4	17	29
7	1758.4	14	36	252.4	24	29	38.3	24	29
8	—	—	—	—	—	—	98.4	28	41
9	—	—	—	—	—	—	214.8	36	57
10	—	—	—	—	—	—	718.8	40	80
11	—	—	—	—	—	—	2212.2	45	113

the kth reaction happens is given by

$$\frac{v_k m_k}{m_k + K_k - 1},$$

divergingmetab is a variation of this model. Now, one of the metabolites in the reaction network can be converted into two different metabolites, meaning that the reaction path splits into two paths which are independent of each other as shown in Figure 5.3(b).

The interpolation and restriction operators for these models are chosen in the same way as for kanbanalt2.

5.2. Numerical results. In this section, we report the results of the experiments we performed on the models from Section 5.1 in order to compare our proposed method, called "MultigridAMEn", to the existing approaches "AMEn" and "Multigrid".

Throughout all experiments, we stop an iteration when the residual norm $\|Ax\|$ is two orders of magnitude smaller than the residual norm of the tensor of all ones (scaled so that the sum of its entries is one). This happens to be our initial guess for AMEn, but it does not correspond to the initial guesses of Multigrid and MultigridAMEn. Note that this requirement also implicitly defines the accuracy requirement for the solution tensor. We do not study the relation between the ranks and the accuracy requirement here. For the model problem overflow such a study was performed in [5].

For both multigrid methods, three pre- and postsmoothing steps are applied on each grid. The number of levels is chosen such that the coarsest grid problem has mode size 3.

Scaling with respect to the number of subsystems. In order to illustrate the scaling behaviour of the three methods, we first choose in all models a capacity of 16 in each subsystem (i.e., mode sizes 17) and vary d, the number of subsystems. Figure 6.1 displays the obtained execution times.

To provide more insight into the results depicted in Figure 6.1, we also give the number of iterations and the maximum rank of the computed approximation for the overflow model in Table 5.1. For the other models, the observed behaviour is similar and we therefore refrain from providing more detailed data.

In Figure 6.1, we observe that Multigrid and MultigridAMEn behave about the same up to $d = 6$ subsystems. For larger d, the cost of solving the coarsest grid problem of size 3^d by a direct method becomes prohibitively large within Multigrid. MultigridAMEn is almost always faster than AMEn even for $d = 4$ or $d = 5$. To which extent MultigridAMEn is faster depends on the growth of the TT ranks of the solution with respect to d, as these have the largest influence on the performance of AMEn.

TABLE 5.2

Execution time (in seconds), number of iterations and maximum rank of the computed approximations for overflow *with $d = 6$ and varying mode sizes. The symbol — indicates that the desired accuracy could not be reached within* 3 600 *seconds.*

	AMEn			Multigrid			MultigridAMEn		
n	time	iter	rank	time	iter	rank	time	iter	rank
5	0.7	4	13	5.9	8	15	6.2	8	15
9	3.8	6	19	6.1	8	15	3.9	8	15
17	239.4	12	28	24.8	17	29	19.5	17	29
33	—	—	—	102.9	17	41	104.6	17	41
65	—	—	—	882.1	20	57	904.1	20	57

Note that the choice of levels in MultigridAMEn is not optimized; it is always chosen such that the coarsest grid mode sizes are three. We sometimes observed that choosing a larger mode size leads to better performance, but we have not attempted to optimize this choice.

The TT format is a degenerate tree tensor network and thus perfectly matches the topology of interactions in the models overflowsim, kanbanalt2, and directedmetab. Compared to overflowsim, the performance is slightly worse for kanbanalt2 and directedmetab, possibly because they contain synchronized interactions, that is, interactions associated with a simultaneous change of state in more than one subsystem. In contrast, overflowsim as well as overflow and overflowpersim only have functional interactions, that is, the state of some subsystems determines the rates associated with other subsystems. This seems to be an important factor as the second best performance is observed for overflowpersim, which contains a cycle in the topology of the network and thus does not match the TT format. This robustness with respect to the topology is also reflected by the results for divergingmetab; recall Figure 5.3(b).

The maximum problem size that is considered is $17^{13} \approx 9.9 \times 10^{15}$. MultigridAMEn easily deals with larger d, but this is the largest configuration for which an execution time below 3 600 seconds is obtained.

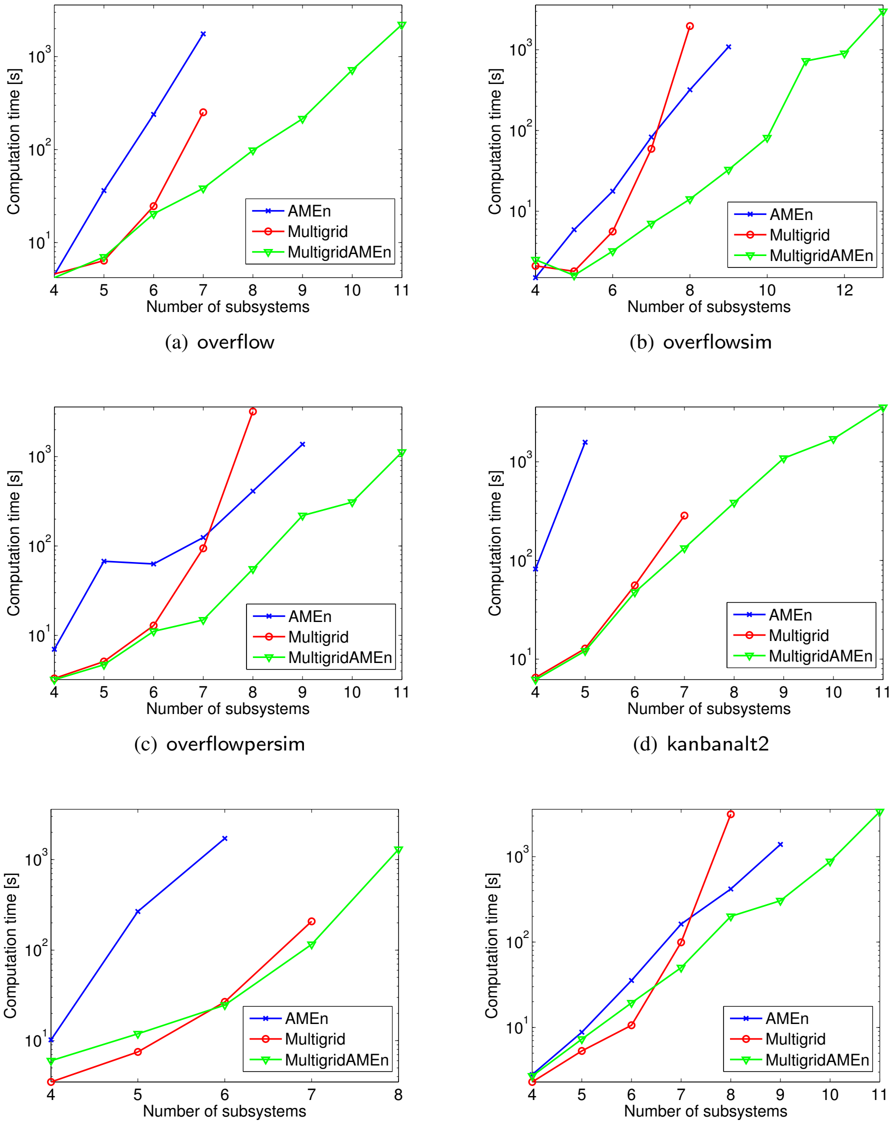
Scaling with respect to the mode sizes. To also illustrate how the methods scale with respect to increasing mode sizes, we next perform experiments where we fix all models to $d = 6$ subsystems and vary their capacity. The execution times for all models are presented in Figure 6.2, while more detailed information for the overflow model is given in Table 5.2.

Figure 6.2 shows that the AMEn method outperforms the two multigrid methods (except for kanbanalt2) for small mode sizes. Depending on the model, the multigrid algorithms start to be faster for mode sizes 9 or 17 as the subproblems to be solved in AMEn become too expensive at this point. The bad performance of AMEn for kanbanalt2 can be explained by the fact that the steady state distribution of this model has rather high TT ranks already for small mode sizes.

Concerning the comparison between the two multigrid methods, no significant difference is visible in Figure 6.2; we have already seen in Figure 6.1 that $d = 6$ is not enough to let the coarsest grid problem solver dominate the computational time in Multigrid. In fact, Figure 6.2 nicely confirms that using AMEn for solving the coarsest grid problem does not have an adverse effect on the convergence of multigrid. The maximum problem size addressed in Figure 6.2 is $129^6 \approx 4.6 \times 10^{12}$.

6. Conclusion.

We have proposed a novel combination of two methods, AMEn and Multigrid, for computing the stationary distribution of large-scale tensor-structured Markov chains. Our numerical experiments confirm that this combination truly combines the advantages of both methods. As a result, we can address a much wider range of problems in terms of number of subsystems and subsystem states. Also, our experiments demonstrate that the TT format is capable of dealing with a larger variety of applications and topologies compared to what has been previously reported in the literature.



(a) overflow

(b) overflowsim

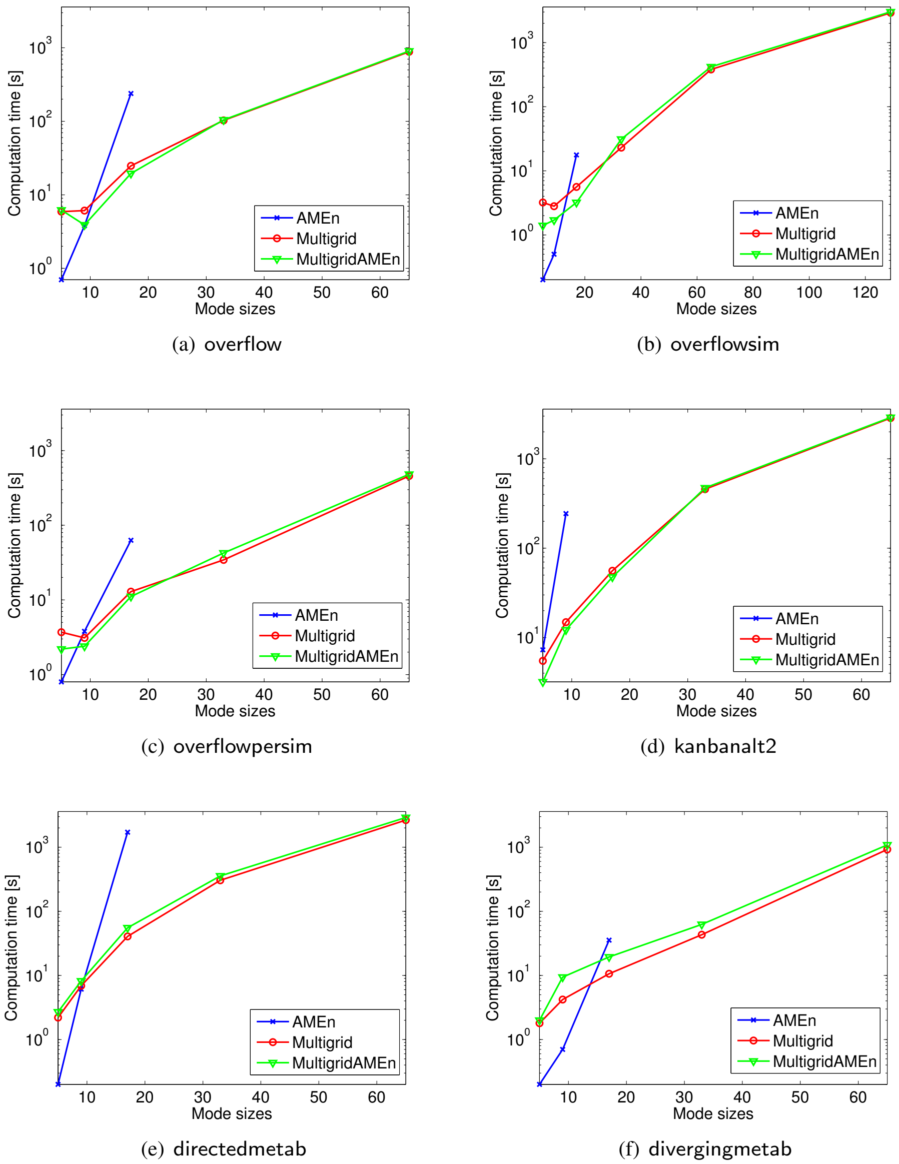
(c) overflowpersim

(d) kanbanalt2

(e) directedmetab

(f) divergingmetab

FIG. 6.1. *Execution time (in seconds) needed to compute an approximation of the steady state distribution for the benchmark models from Section 5.1. All mode sizes are set to* 17.



(a) overflow

(b) overflowsim

(c) overflowpersim

(d) kanbanalt2

(e) directedmetab

(f) divergingmetab

FIG. 6.2. *Execution time (in seconds) needed to compute an approximation of the steady state distribution for the benchmark models from Section 5.1. All models have $d = 6$ subsystems.*

REFERENCES

[1] D. F. ANDERSON, G. CRACIUN, AND T. G. KURTZ, *Product-form stationary distributions for deficiency zero chemical reaction networks*, Bull. Math. Biol., 72 (2010), pp. 1947–1970.

[2] N. ANTUNES, C. FRICKER, P. ROBERT, AND D. TIBI, *Analysis of loss networks with routing*, Ann. Appl. Probab., 16 (2006), pp. 2007–2026.

[3] J. BALLANI AND L. GRASEDYCK, *A projection method to solve linear systems in tensor format*, Numer. Linear Algebra Appl., 20 (2013), pp. 27–43.

[4] A. BERMAN AND R. J. PLEMMONS, *Nonnegative Matrices in the Mathematical Sciences*, SIAM, Philadelphia, 1994.

[5] M. BOLTEN, K. KAHL, AND S. SOKOLOVIĆ, *Multigrid methods for tensor structured Markov chains with low rank approximation*, SIAM J. Sci. Comput., 38 (2016), pp. A649–A667.

[6] A. BRANDT, *General highly accurate algebraic coarsening*, Electron. Trans. Numer. Anal., 10 (2000), pp. 1–20. http://etna.ricam.oeaw.ac.at/vol.10.2000/pp1-20.dir/pp1-20.pdf

[7] J. J. BRANNICK AND R. D. FALGOUT, *Compatible relaxation and coarsening in algebraic multigrid*, SIAM J. Sci. Comput., 32 (2010), pp. 1393–1416.

[8] M. BREZINA, T. MANTEUFFEL, S. MCCORMICK, J. RUGE, AND G. SANDERS, *Towards adaptive smoothed aggregation* (αSA) *for nonsymmetric problems*, SIAM J. Sci. Comput., 32 (2010), pp. 14–39.

[9] P. BUCHHOLZ, *Product form approximations for communicating Markov processes*, Perform. Eval., 67 (2010), pp. 797–815.

[10] P. BUCHHOLZ AND T. DAYAR, *On the convergence of a class of multilevel methods for large sparse Markov chains*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 1025–1049.

[11] R. CHAN, *Iterative methods for overflow queueing networks. I*, Numer. Math., 51 (1987), pp. 143–180.

[12] ———, *Iterative methods for overflow queueing networks. II*, Numer. Math., 54 (1988), pp. 57–78.

[13] S. DOLGOV AND B. KHOROMSKIJ, *Simultaneous state-time approximation of the chemical master equation using tensor product formats*, Numer. Linear Algebra Appl., 22 (2015), pp. 197–219.

[14] S. V. DOLGOV AND D. V. SAVOSTYANOV, *Alternating minimal energy methods for linear systems in higher dimensions*, SIAM J. Sci. Comput., 36 (2014), pp. A2248–A2271.

[15] A. GREENBAUM, *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia, 1997.

[16] L. KAUFMAN, *Matrix methods for queuing problems*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 525–552.

[17] V. KAZEEV, M. KHAMMASH, M. NIP, AND C. SCHWAB, *Direct solution of the chemical master equation using quantized tensor trains*, PLoS Comput. Biol., 10 (2014), e1003359 (19 pages).

[18] D. KRESSNER AND F. MACEDO, *Low-rank tensor methods for communicating Markov processes*, in Quantitative Evaluation of Systems, G. Norman and W. Sanders, eds., vol. 8657 of Lecture Notes in Computer Science, Springer, Cham, 2014, pp. 25–40.

[19] D. KRESSNER, M. STEINLECHNER, AND A. USCHMAJEW, *Low-rank tensor methods with subspace correction for symmetric eigenvalue problems*, SIAM J. Sci. Comput., 36 (2014), pp. A2346–A2368.

[20] A. N. LANGVILLE AND W. J. STEWART, *The Kronecker product and stochastic automata networks*, J. Comput. Appl. Math., 167 (2004), pp. 429–447.

[21] E. LEVINE AND T. HWA, *Stochastic fluctuations in metabolic pathways*, Proc. Natl. Acad. Sci. USA, 104 (2007), pp. 9224–9229.

[22] F. MACEDO, *Benchmark problems on stochastic automata networks in tensor train format*, Tech. Rep., MATHICSE, EPF Lausanne, Lausanne, 2015.

[23] I. V. OSELEDETS, *Approximation of* $2^d \times 2^d$ *matrices using tensor decomposition*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2130–2145.

[24] ———, *MATLAB TT-Toolbox Version 2.2*, 2011.
Available at http://spring.inm.ras.ru/osel/?page_id=24.

[25] ———, *Tensor-Train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[26] I. V. OSELEDETS AND S. V. DOLGOV, *Solution of linear systems and matrix inversion in the TT-format*, SIAM J. Sci. Comput., 34 (2012), pp. A2718–A2739.

[27] B. PHILIPPE, Y. SAAD, AND W. J. STEWART, *Numerical methods in Markov chain modelling*, Oper. Res., 40 (1996), pp. 1156–1179.

[28] B. PLATEAU AND W. J. STEWART, *Stochastic automata networks*, in Computational Probability, W. K. Grassmann, ed., Kluwer, Norwell, 2000, pp. 113–152.

[29] J. RUGE AND K. STÜBEN, *Algebraic multigrid*, in Multigrid Methods, S. McCormick, ed., Frontiers in Applied Mathematics, 3, SIAM, Philadelphia, 1987, pp. 73–130.

[30] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.

[31] Y. SAAD AND M. H. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[32] U. TROTTENBERG, C. OSTERLEE, AND A. SCHÜLLER, *Multigrid*, Academic Press, San Diego, 2001.

[33] W. VANROOSE, B. REPS, AND H. B. ZUBAIR, *GMRES-based multigrid for the complex scaled preconditoner for the indefinite Helmholtz equation*, Preprint on arXiv, 2013.
https://arxiv.org/abs/1012.5379.

[34] S. R. WHITE, *Density matrix renormalization group algorithms with a single center site*, Phys. Rev. B, 72 (2005), 180403 (4 pages).