Russ. J. Numer. Anal. Math. Modelling, Vol. 28, No. 2, pp. 149–172 (2013)
DOI 10.1515/rnam-2013-0009

TT-GMRES: solution to a linear system in the structured tensor format

S. V. DOLGOV*

Abstract — An adapted tensor-structured GMRES method for the TT format is proposed and investigated. The Tensor Train (TT) approximation is a robust approach to high-dimensional problems. One class of such problems involves the solution of a linear system. In this work we study the convergence of the GMRES method in the presence of tensor approximations and provide relaxation techniques to improve its performance. Several numerical examples are presented. The method is also compared with a projection TT linear solver based on the ALS and DMRG methods. On a particular SPDE (high-dimensional parametric) problem these methods manifest comparable performance, with a good preconditioner the TT-GMRES overcomes the ALS solver.

1. Introduction

Solving linear systems arising from problems with many dimensions is a computationally demanding task. Such problems are posed, for example, in quantum chemistry [21, 22], financial mathematics [30, 33] and other sciences. To work with d-dimensional arrays is a challenging problem due to the *curse of dimensionality* [2]: the number of elements of a tensor grows exponentially with the number of dimensions d, and so does the complexity of working with fully populated tensors. For d of an order of some tens or hundreds one needs other approaches, for example, special low-parametric representations or *formats*. As soon as such a format comes with fast linear

*Institute of Numerical Mathematics, Russian Academy of Sciences, Moscow 119333, Russia

This work was supported in part by RFBR grants 09-01-12058, 10-01-00757, 11-01-00549, RFBR/DFG grant 09-01-91332, Russian Federation Gov. contracts No. P1178, P1112 and P940, 14.740.11.0345 and Promotionstipendium of Max-Planck Gesellschaft. Part of this work was done during the stay of the author at Max-Plank Institute for Mathematics in Sciences, Leipzig, Germany.

algebra algorithms, such as additions, matrix-by-vector multiplications and scalar products, any of the classical iterative methods can be implemented straightforwardly. The first difficulty is that the *effective* amount of the unknowns, required to store the vectors involved in computations in a chosen format might grow arbitrarily during the solution process. Second, most of classical methods are proved to be convergent in the exact arithmetics, and their behaviour in approximate computations arising from the use of the formats is uncertain. The former issue depends essentially on a particular problem and has to be considered with respect to a particular application, the latter one can be described in a general case. As for the GMRES method, such consideration will be presented in this paper.

Several formats have been proposed to represent a tensor in a data-sparse way. They include the canonical and the Tucker formats, the two formats with well-established properties and application areas, see [17] for more details. Their drawbacks are known. To avoid these drawbacks, the development of new tensor formats began. In 2009 Hackbusch and Kuhn and later Grasedyck [7, 10] and Oseledets and Tyrtyshnikov [27] independently proposed two (slightly different) hierarchical schemes for the tensor approximation, the $\mathscr{H}$-Tucker and Tree Tucker formats. These formats depend on specially chosen *dimension trees* and require recursive procedures. To avoid the recursion, it has been proposed to use a simple matrix product form of the decomposition [23, 26], that is called the *Tensor Train format*, or simply the TT-format.

A tensor A is said to be in the TT-format if its elements are defined by a formula

$$A(i_1,\dots,i_d) = G_1(i_1)\dots G_d(i_d) \tag{1.1}$$

where $G_k(i_k)$ is an $r_{k-1}\times r_k$ matrix for each fixed $i_k,\ 1\leqslant i_k\leqslant n_k$. To make the matrix-by-matrix product in (1.1) scalar, the boundary conditions $r_0 = r_d = 1$ are imposed. The numbers r_k are called *TT-ranks* and $G_k(i_k)$ are the *cores* of the TT-decomposition of a given tensor. If $r_k\leqslant r, n_k\leqslant n$, then the storage of the TT-representation requires $\leqslant dnr^2$ memory cells. If r is small, this is much smaller than the storage of the full array, n^d.

One can go even further and introduce the *Quantized TT* (QTT) format [14, 24]: if the mode sizes are equal to 2^p, we can reshape a given tensor to the $2\times 2\times\dots\times 2$ tensor with a higher dimension, all mode sizes being equal to 2, and then apply the TT approximation to this new tensor. The storage in this case is estimated as $\mathscr{O}(dr^2\log n)$.

The TT-format is stable in the sense that the best approximation with bounded TT-ranks always exists and a quasioptimal approximation can be computed by a sequence of SVDs of auxiliary matrices [23, 26].

The TT-format comes with all basic linear algebra operations. Addition, matrix-by-vector product, or elementwise multiplication can be implemented in linear d and polynomial in r complexity producing a result also in the TT-format [23, 26]. The problem is that after such operations the TT-ranks grow. For example, the TT-ranks of the sum of two tensors are equal (formally) to the sum of the TT-ranks of the addends. In the case of the matrix-by-vector product the result is also in the TT-format with the TT-ranks of the matrix and the vector multiplied. After several iterations, the TT-ranks will become too large, therefore the tensor *rounding*, or truncation is needed: a given tensor A is approximated by another tensor B with the minimal possible TT-ranks with a prescribed accuracy ε (or a fixed maximal rank R):

$$B = \mathcal{T}_{\varepsilon,R}(A) \text{ so that } ||A-B||_F \leqslant \varepsilon ||A||_F \text{ and/or } \operatorname{rank}(B) \leqslant R.$$

The rounding procedure in the TT-format can be implemented in $\mathcal{O}(dnr^3)$ operations [23, 26].

In this work we implement an adapted GMRES solver using the TT arithmetics and truncations. It is worth mentioning previous papers devoted to Krylov methods with tensor computations: [18] (a FOM-like method in the case of a Laplace-like matrix), [19] (Richardson, PCG and BiCGStab methods with application to parametric and stochastic PDEs), and, the closest to our work, a projection method for linear systems in the $\mathscr{H}$-Tucker format which was proposed in [1]. Our GMRES method is slightly different. First, the $\mathscr{H}$-Tucker method from [1] uses the projectors with equal sizes:

$$Ax = b \to W_m^T A V_m y = W_m^T b, \quad V_m \in \mathbb{C}^{n\times m}, \quad W_m = AV_m.$$

A very important feature of the GMRES method is the rectangular Hessenberg matrix computed via the projections to subspaces with *different* dimensions:

$$\bar{H}_m = V_{m+1}^T A V_m.$$

In this sense, the above mentioned method is a certain realization of the *geometric* minimal residual method, the linear span of (nonorthogonal) vectors

W_m contains all Krylov vectors from V_{m+1} except the first one. Moreover, we provide the error and convergence analysis with respect to the tensor rounding using the theory of inexact GMRES and the relaxation strategies to reduce TT ranks and improve the performance. A convergence estimate has been also provided for the tensor CG-type method for eigenvalue problems in [20]. A part of our paper (Section 3) is devoted to the role of matrix-by-vector and preconditioner-by-vector multiplications in approximate computations, showing the differences between the GMRES and CG methods.

Note that the particular choice of the TT format in this paper is not important for the general theory and is due to the simplicity, convenience and robustness of the TT format for a wide class of problems. The performance improvements considered below, arising from the inexact Krylov theory, have been also successfully applied for the tensor version of GMRES in the Tucker format by Dmitry Savostyanov. Numerical examples were presented at the Workshop on Tensor Decompositions and Applications (TDA 2010), Monopoli, Bari, Italy.

The rest of the paper is organized as follows. In the next section we briefly review the scheme of the GMRES method. In Section 3 we discuss the influence of preconditioners, especially in the case of errors caused by tensor roundings via SVD. In Section 4 we explain the inexact GMRES theory, provide the error analysis for the approximate TT-GMRES and the whole algorithm. And in Section 5 we present several numerical examples and compare two methods: TT-GMRES and the DMRG-ALS linear solver from [5].

2. Exact GMRES method in standard arithmetics

In this section we briefly recall the GMRES method following [28]. We are going to investigate how the errors arising from tensor roundings affect the convergence of the methods. Moreover, if we are solving discretized PDEs, we have to consider carefully which norm of the residual one shall use.

Let us present the basic properties of this method. Suppose we are going to solve a linear system

$$Ax = b, \quad A \in \mathbb{C}^{n\times n}, \quad x, b \in \mathbb{C}^n.$$

The method is based on the minimization of the residual functional $||b - Ax||$

on the Krylov subspaces:

$$\mathscr{K}_k = \{b, Ab, A^2b, \ldots, A^{k-1}b\}.$$

To build the Krylov basis, one uses the Arnoldi process (see Algorithm 2.1), which is nothing else but the Gramm-Shmidt orthogonalization method applied to the Krylov vectors. After k steps we have the orthonormal vectors V_{k+1} and $(k+1)\times k$ matrix $\bar{H}_k = [h_{i,j}]$. Now we have to obtain a correction to the solution.

The vectors v_i possess the following property:

$$AV_k = V_{k+1}\bar{H}_k. \tag{2.1}$$

Suppose the initial guess x_0 is given, the initial residual $r_0 = b - Ax_0$. We are to solve the least squares problem

$$\min_{z\in\mathscr{K}_k} ||f - A(x_0+z)|| = \min_{z\in\mathscr{K}_k} ||r_0 - Az||.$$

Now put $z = V_k y$, reformulate the functional for the vector y, which is small, if $k \ll n$:

$$J(y) = ||\beta v_1 - AV_k y||$$

where $\beta = ||r_0||$. Taking into account (2.1), we obtain

$$J(y) = ||V_{k+1}(\beta e_1 - \bar{H}_k y)|| = ||\beta e_1 - \bar{H}_k y||$$

since $||V_{k+1}|| = 1$ due to the orthogonality, e_1 is the first identity vector of size $k+1$. Now we write the correction to the solution:

$$x_k = x_0 + V_k y_k, \quad y_k = \arg\min_y ||\beta e_1 - \bar{H}_k y||.$$

In the Arnoldi process the number of basis vectors grows up to the size of the matrix, resulting in a prohibitive amount of memory and computational time required. To avoid this, one uses GMRES with *restarts*: every m steps the current solution is taken as the initial guess, and the algorithm restarts. The overall scheme of the GMRES(m) is given in Algorithm 2.1.

Algorithm 2.1 (GMRES(m)).
Require: Matrix A, right-hand side b, initial guess x_0, stopping tolerance ε.
Ensure: Approximate solution x_m : $||Ax_m - b|| \leqslant \varepsilon$.

1: Start: compute $r_0 = b - Ax_0$, $v_1 = r_0/||r_0||$.
2: Iterations:
3: **for** $j = 1, 2, \ldots, m$ **do**
4: $h_{i,j} = (Av_j, v_i)$, $i = 1, 2, \ldots, j$ {Arnoldi process}
5: $\hat{v}_{j+1} = Av_j - \sum_{i=1}^{j} h_{i,j} v_i$
6: $h_{j+1,j} = ||\hat{v}_{j+1}||$
7: $v_{j+1} = \hat{v}_{j+1}/h_{j+1,j}$.
8: **end for**
9: Assemble the matrix $\bar{H}_m = [h_{i,j}]$, $j = 1, \ldots, m$, $i = 1, \ldots, j+1$.
10: Compute the least-squares solution: $y_m = \arg\min_y ||\beta e_1 - \bar{H}_m y||$
11: $x_m = x_0 + V_m y_m$ {Update}
12: Restart: compute $r_m = b - Ax_m$. Stop if $||r_m|| \leqslant \varepsilon$.
13: Otherwise set $x_0 = x_m$, $v_1 = r_m/||r_m||$ and go to 2.

One of the nice properties following from (2.1) is a cheap way to compute the residual:

$$||\beta e_1 - \bar{H}_k y_k|| = ||b - A(x_0 + V_k y_k)||$$

so we can check the stopping criteria using only small vectors e_1, y_k and matrix $\bar{H}_k$.

For the *exact* GMRES the following property is shown in [28].

Theorem 2.1. *The solution x_j, obtained on the j-th GMRES step is exact if and only if $\hat{v}_{j+1} = 0 \Leftrightarrow h_{j+1,j} = 0$.*

Hereafter we will consider the *inexact* GMRES, for which this theorem cannot be applied.

3. Role of preconditioners and smoothers

A well-known way to improve the convergence of an iterative method is to use a preconditioner:

$$Ax = b \to MAx = Mb \text{ or } AMy = b,\ x = My$$

which may shrink the spectrum of a matrix to a small interval (for discretized PDE problems one usually requires a mesh-independent spectral interval), or make clusters of eigenvalues (see [3, 31]).

The first formula is called the *left* preconditioner, the second is the *right* one. The main difference between these approaches (at least, for symmetric matrices) is the residual to be computed and considered: the first one works with $||MAx - Mb||$, i.e. the *preconditioned* residual, the second one with the real residual $||Ax - b||$. Usually in the solution process the residual-based stopping criterion is used, and in the case of the left preconditioner the residual has to be computed explicitly (whereas the norm of the preconditioned residual can be computed rapidly in the GMRES method). However, if the preconditioner is close enough to the inversed matrix ($MA = I + E$, $||E|| \ll 1$), it might be worth using the preconditioned residual, since it provides information on the *solution error*, which is more important (and relevant) than the residual. Indeed,

$$MAx - Mb = (I+E)x - (I+E)A^{-1}b = (x - A^{-1}b) + Ex - EA^{-1}b \approx x - A^{-1}b.$$

In some cases the norm of E might be even greater than 1, but if it does not depend on the grid size, the preconditioned residual still gives relevant information on the error, considering that the constants of equivalence $c_1||x - A^{-1}b|| \leqslant ||MAx - Mb|| \leqslant c_2||x - A^{-1}b||$ do not depend on h. Moreover, for the usual scales of errors arising in tensor arithmetical roundings ($10^{-4} \div 10^{-6}$), the real residual might give no information on the convergence at all.

So, consider the tensor rounding in the following form. Suppose a tensor u is given, and consider a low-rank approximation

$$\tilde{u} = u + \varepsilon.$$

Since the correction ε is composed from the last singular vectors of the TT-blocks of u, it generally contains harmonics with higher frequencies than $\tilde{u}$. Let us illustrate it taking the following example. Consider a 1D function u in the interval $[-1, 1]$ and its Fourier decomposition:

$$u(x) = \alpha_0 + \sum_{m=1}^{\infty} \alpha_m \cos(\pi m x) + \beta_m \sin(\pi m x).$$

Take a partial sum of this sequence as an approximation.

$$\tilde{u}(x) = \alpha_0 + \sum_{m=1}^{R} \alpha_m \cos(\pi m x) + \beta_m \sin(\pi m x).$$

From the Parseval's theorem it is known that if the coefficients are computed as follows:

$$\alpha_m = \frac{(u, \cos(\pi m x))}{(\cos(\pi m x), \cos(\pi m x))}, \quad \beta_m = \frac{(u, \sin(\pi m x))}{(\sin(\pi m x), \sin(\pi m x))}$$

the approximation with harmonic functions $\tilde{u}$ is *optimal* in the L_2-norm. The approximation error is written as the following sum:

$$u - \tilde{u} = \sum_{m=R+1}^{\infty} \alpha_m \cos(\pi m x) + \beta_m \sin(\pi m x)$$

i.e. it contains harmonics with frequencies greater than R. The singular value decomposition provides the optimal rank-r approximation to a matrix in the 2-norm, and the same phenomenon occurs.

Consider now the effect of the second derivative operator $\mathrm{d}^2/\mathrm{d}x^2$ on the approximated function $\tilde{u}$. It reads

$$\frac{\mathrm{d}^2 \tilde{u}}{\mathrm{d}x^2} = \frac{\mathrm{d}^2 u}{\mathrm{d}x^2} + \sum_{m=R+1}^{\infty} \pi^2 m^2 \alpha_m \cos(\pi m x) + \pi^2 m^2 \beta_m \sin(\pi m x)$$

and

$$\left\| \frac{d^2 \tilde{u}}{dx^2} - \frac{d^2 u}{dx^2} \right\| \geqslant \pi^2 R^2 \left\| \tilde{u} - u \right\|.$$

The approximation $\tilde{u}$ in the L_2 norm might provide a sufficient accuracy of ε, but the error in the second derivative is of the order $R^2\varepsilon$, which might be prohibitively large. The discretization matrix A of the second derivative operator has the norm $\mathcal{O}(1/h^2)$, so

$$||A\tilde{u} - Au|| \leqslant ||A||\varepsilon = \mathcal{O}\left(\frac{1}{h^2}\varepsilon\right) \gg \varepsilon.$$

Considering relative quantities, if the accuracy of the linear system solution is $||x - A^{-1}b||/||x|| = \varepsilon$, the residual norm might be $||Ax - b||/||b|| = \mathcal{O}(\mathrm{cond}(A)\ \varepsilon)$. If the tensor rounding accuracy is $\varepsilon = 10^{-5}$, and the problem is discretized on 1000 grid points, then $\mathrm{cond}(A) = \mathcal{O}(10^6)$ and $||Ax - b||/||b|| = \mathcal{O}(10)$. The L_2-norm accuracy of the order 10^{-5} might be sufficient, but one can not control it, as the residual norm is greater than 1.

So, for tensor iterative methods the use of a preconditioner is important not only for the convergence acceleration, but also to keep the equivalence

constants between the error and the residual of the order of 1. Moreover, it is important to use the left preconditioner when we multiply first the stiffness matrix by a vector and then the preconditioner, which reduces the errors introduced by tensor roundings (*smoother*). In this sense, the Conjugate Gradient method is not very efficient. Indeed, consider the PCG algorithm (it works in terms of the right preconditioner $AMM^{-1}x = b$, see e.g. [11]):

$$\begin{aligned} r_0 &= b - Ax_0, \quad p_1 = Mr_0 \\ \alpha_i &= (r_{i-1}, Mr_{i-1})/(Ap_i, p_i) \\ x_i &= x_{i-1} + \alpha_i p_i \\ r_i &= r_{i-1} - \alpha_i A p_i \\ \beta_i &= (r_i, Mr_i)/(r_{i-1}, Mr_{i-1}) \\ p_{i+1} &= Mr_i + \beta_i p_i. \end{aligned}$$

In the formulas for α_i and r_i the last operation before the scalar product or linear combination is the MatVec with the stiffness matrix Ap_i, which might be computed with a very large error, if p_i is rounded in the L_2 norm. As a result, new iterands are computed with the same large error, and the method diverges. In this sense, it is much more efficient to use the GMRES method with the left preconditioner, when the next Krylov vector is computed as $v_{i+1} = MAv_i$. Numerical experiments have shown that the roundings in this operation can be even applied sequentially:

$$v_{i+1} = \mathscr{T}_{\varepsilon,R}(M\mathscr{T}_{\varepsilon,R}(Av_i))$$

without corrupting the final result. It is especially important if the preconditioner is combined from several matrices, e.g. the preconditioner from [3], where the whole matrix MA can not be computed explicitly due to its very high ranks, and multiplications are applied during several successive implicit procedures, which provide approximate products (for example, the DMRG-based TT-MatVec, see [5, 15] for the DMRG schemes).

4. Relaxation strategies, inexact GMRES, and the TT-GMRES algorithm

The inexact Krylov methods theory [6, 29, 32] allows us to estimate the effect of the noise arising from tensor roundings on the GMRES convergence. Moreover, the performance of tensor methods essentially depends on the TT

ranks of the intermediate vectors. It turns out in practice that if we truncate all the vectors with the same accuracy, the ranks of the last Krylov vectors added to the Krylov basis increase from one iteration to another. The relaxation strategies proposed in the papers [6, 29, 32] allow us to truncate the latter Krylov vectors less accurately than the former ones, thus keeping the ranks at almost constant values, or even reducing them in the last iterations.

We can consider roundings as the usual MatVecs with the perturbed matrix:

$$\mathcal{T}_{\varepsilon,R}(Av) = \tilde{A}v = (A+E)v$$

where E is the error matrix, which, generally speaking, changes each time when the matrix-by-vector multiplication is computed, and moreover, might depend on v. Our goal is to estimate allowed values for $||E||$, which provide convergence to some desired accuracy. It can be proved that during the Krylov iterations the norm of the error matrix $||E||$ can be increased, i.e. the vector operations in the last iterations may be computed with worse accuracy.

A linear system reduction to the Krylov subspace (2.1) for the inexact GMRES is written as:

$$\begin{bmatrix}(A+E_1)v_1 & \cdots & (A+E_m)v_m\end{bmatrix} = V_{m+1}\bar{H}_m. \tag{4.1}$$

Notice that V_m is no more a basis in the exact subspace $\mathscr{K}_m$. The algorithm remains the same, as for the exact GMRES, but the minimization of $||\beta e_1 - \bar{H}_m y_m||$ does not lead any more to the minimization of the real residual $||b - Ax_m||$. From (4.1) it follows that we are solving the following optimization problem:

$$\min_{q\in R(W_m)} ||r_0 - q||, \quad r_0 = b - (A+E_0)x_0$$

and $R(W_m)$ is the linear span of the vectors $W_m = V_{m+1}\bar{H}_m$. So in fact we are minimizing the *computed* approximate residual

$$||\tilde{r}_m|| = ||r_0 - q_m|| = |h_{m+1,m}e_m^T y_m|.$$

If the new Krylov vector $(A+E_m)v_m$ in some iteration appears to be almost linearly dependent on the basis V_m, the quantity $h_{m+1,m}$ is small, and the approximate residual $||\tilde{r}_m||$ is also small, which we can consider as the convergence of the method. But the real residual $||r_m||$ might be significantly larger, and the estimate of the difference $||r_m - \tilde{r}_m||$ will be considered below.

We formulate the main theorem, which comes from [29].

Theorem 4.1. *Suppose some $\varepsilon > 0$ is given, m GMRES iterations are conducted for the system $Az = r_0$, the computed residual $\tilde{r}_m$ is obtained. Then if for any $i \leqslant m$ we have*

$$||E_i|| \leqslant \frac{\sigma_m(\bar{H}_m)}{m} \frac{1}{||\tilde{r}_{i-1}||} \varepsilon$$

where $\sigma_m(\bar{H}_m)$ is the minimal singular value of the Hessenberg matrix of the reduced system, for the real residual $r_m = r_0 - Az_m$ the following estimate holds:

$$||r_m - \tilde{r}_m|| \leqslant \varepsilon.$$

We refer for the proof to [29].

So the accuracy of the MatVec computation can be relaxed inversely proportional to the current residual, and if the process is stopped (in the case of stagnation, or if the computed residual becomes smaller than the stopping tolerance), the real residual will differ quantitatively from the computed one not greater in the norm than ε, i.e. the convergence of the method is controlled.

In order to obtain scale-independent estimates (i.e. the same ones for the systems $Ax = b$ and $\alpha Ax = \alpha b$), one usually considers the *relative* residual and the corresponding stopping criteria, for example,

$$\frac{||r_i||}{||b||} \leqslant \varepsilon.$$

In the same way, one can consider the difference between the real and computed residuals: $||r_m - \tilde{r}_m||/||b|| \leqslant \varepsilon$. In this case the result of Theorem 4.1 can be reformulated for the relative quantities, taking into account that $\varepsilon = \varepsilon||b||$:

$$\frac{||E_i||}{||A||} \leqslant \frac{\sigma_m(\bar{H}_m)}{m||A||} \frac{1}{||\tilde{r}_{i-1}||/||b||} \varepsilon.$$

The minimal singular value of $\bar{H}_m$ can be estimated from the minimal singular value of A:

$$\sigma_m(\bar{H}_m) \geqslant \sigma_n(A) - \left\| \begin{bmatrix} E_1 v_1 & \cdots & E_m v_m \end{bmatrix} \right\|$$

so we formulate the following relaxation strategy for the MatVec error.

Corollary 4.1. Suppose m GMRES iterations are performed. If for any $i \leqslant m$ the relative error introduced in the matrix-by-vector multiplication is bounded by the following rule:

$$\frac{||E_i||}{||A||} \leqslant \frac{1}{m\,\mathrm{cond}(A)} \frac{1}{||\tilde{r}_{i-1}||/||b||}\varepsilon \tag{4.2}$$

the real relative residual and the computed one are connected with $||r_m||/||b|| \leqslant ||\tilde{r}_m||/||b|| + \varepsilon$.

With a good spectrally equivalent preconditioner $\mathrm{cond}(A) = \mathcal{O}(1)$ (notice that the matrix A is considered to be already left-preconditioned here), and $m = \mathcal{O}(1)$, in this case we can consider (4.2) in the following form: if

$$\frac{||E_i||}{||A||} \leqslant \frac{1}{||\tilde{r}_{i-1}||/||b||}\varepsilon$$

then the inexact GMRES will converge to the relative residual not greater than

$$m\,\mathrm{cond}(A)\varepsilon.$$

This approach will be used in the numerical experiments below.

Let us write the final algorithm of the tensor GMRES with relaxations. Notice also that in the Arnoldi process we have used the modified Gramm-Shmidt algorithm, which is more stable in the presence of the rounding errors. In addition, as the left preconditioner is used, we do not write it explicitly, but assume that the matrix A and the right-hand side b are already preconditioned.

It is important to choose when to perform tensor rounding, either after adding all the summands in the orthogonalization and correction steps, or after each addition. Formally, one can introduce the error only to the MatVec itself, but the orthogonality of V_m must be kept despite the perturbations in (4.1). Moreover, it is better to sum exactly the significantly different vectors $y_j(i)v_i$. The obvious drawback is the rank overhead, which can be m times larger than in the case of step-by-step truncations. So whenever possible (small number of iterations) it is worth to perform only the final truncation when the summation is ready (in the case of small mode sizes (Quantized TT) it can be easily done by the DMRG truncation (see next section) instead of the direct one from [27]).

Algorithm 4.1 (Relaxed TT-GMRES(m)).

Require: Right-hand side b, initial vector x_0 in the TT format, matrix A as a tensor MatVec procedure $y = \mathcal{T}_{\varepsilon,R}(Ax)$, accuracy ε and/or maximal TT rank R.

Ensure: Approximate solution x_j : $||Ax_j - b||/||b|| \leqslant \varepsilon$.

Start: compute $r_0 = \mathcal{T}_{\varepsilon,R}(b - Ax_0)$, $\beta = ||r_0||$ $v_1 = r_0/\beta$
Iterations:
for $j = 1, 2, \ldots, m$ **do**
  Compute the relaxed accuracy $\delta = \dfrac{\varepsilon}{||\tilde{r}_{j-1}||/\beta}$
  $w = \mathcal{T}_{\delta,R}(Av_j)$ {new Krylov vector}
  for $i = 1, 2, \ldots, j$ **do**
    $h_{i,j} = (w, v_i)$
    $w = w - h_{i,j}v_i$ {orthogonalization}
  end for
  $w = \mathcal{T}_{\delta,R}(w)$ {compression}
  $h_{j+1,j} = ||w||$, $v_{j+1} = w/h_{j+1,j}$
  Assemble matrix $\bar{H}_j = [h_{i,k}]$, $k = 1, \ldots, j$, $i = 1, \ldots, j+1$
  Compute a solution of the reduced system: $y_j = \arg\min_y ||\beta e_1 - \bar{H}_j y||$
  Check the residual $||\tilde{r}_j|| = ||\beta e_1 - \bar{H}_j y_j||$: if $||\tilde{r}_j||/||b|| \leqslant \varepsilon$, then break
end for
Update the solution: initialize $x_j = x_0$
for $i = 1, 2, \ldots, j$ **do**
  $x_j = x_j + y_j(i)v_i$ {correction}
end for
$x_j = \mathcal{T}_{\varepsilon,R}(x)$ {compression}
Restart: if $||\tilde{r}_j||/||b|| > \varepsilon$, then set $x_0 = x_j$, go to 1.

5. Fast and accurate TT arithmetics in high dimensions

An interesting class of high-dimensional problems are the multiparametric problems arising in the discretized Karhunen–Loeve model for the PDEs with stochastic data. In such problem, the number of parameters is usually of the order of tens, and after the tensorisation (Quantisation) the number of dimensions is in the order of hundreds. Even for an 1D physical problem, the QTT ranks may usually depend on the number of parameters almost lin-

early, thus keep the values 50–100. In this case, the multiply-and-compress strategy fails, because of the prohibitive complexity $\mathcal{O}(dnr^6)$. A better alternative is to use direct minimization methods, based on the alternating directions approach, the ALS and DMRG (also known as MALS) schemes. There are several papers on linear- and eigenvalue solvers using the DMRG scheme [5, 12, 15]. The simple approximation issue is discussed in these articles as well, and now there is the new one [25] concerning specifically the approximate matrix-by-vector product.

Unfortunately, the main disadvantage of all presented TT-DMRG methods is the tendency to underestimate the ranks in essentially high-dimensional problems. Recall briefly the main procedure of the approximation via the DMRG (MALS) scheme:

1. Suppose a functional $J(x)$ to minimize is given (e.g. $J(x) = ||x-y||^2$).
2. Initial guess for x in the TT format is given: $x = X_1(i_1)\cdots X_d(i_d)$.
3. Choose two neighbouring cores and convolve a *supercore*: $X_k(i_k)X_{k+1}(i_{k+1}) \to W_k(i_k, i_{k+1})$.
4. Solve the reduced optimization problem for the elements of W_k: $\hat{W}_k = \arg\min_{W_k} J(x)$.
5. Recover the TT structure (e.g. via SVD): $\hat{W}_k \approx \hat{X}_k(i_k)\hat{X}_{k+1}(i_{k+1})$.
6. Consider the next pair of cores, and so on.

The rank is determined adaptively at step 5. The ranks are not known in general, and we usually start from a low-rank initial guess, subsequently increasing them during the DMRG iterations. The problem is that if we are using the fixed ε-truncation of singular values, the ranks determined become underestimated, as the dimension increases. There are two factors. First, the worst-case error accumulation in the whole tensor is $d\varepsilon$, if the local errors in each block are bounded by ε [27]. Second, instead of the direct compression routine from [27], where the fixed cores are the cores of the initial tensor, here we are working with a *projection* to some tensor with blocks, which are far from a good approximation (in early iterations), and moreover, have insufficient ranks. To get rid of this, in this work we have used the algorithms modified as follows:

- First, set the accuracy for the local truncation to $\varepsilon_{\text{loc}} = \varepsilon/d$.

- Second, after the rank is truncated according to ε_{loc}, artificially add more singular vectors (thus obtaining the truncation with increased accuracy and rank). This additional rank can even be determined adaptively, depending on the convergence of the current supercore, by comparison with the approximation from the previous iteration.

The approximation computed this way might have overestimated ranks. To reduce them to the proper values, it is sufficient to conduct the last iteration with the standard truncation without including additional singular vectors (in fact, it performs like a direct compression routine, as the proper approximation is already achieved at this step, but the complexity is now $\mathcal{O}((r+r_{\text{add}})^3)$ instead of $\mathcal{O}(r^6)$, and the additional rank is usually significantly smaller than r).

For the DMRG-solve routine, we will show the role of the increased-rank truncation in the next section. But for the approximations and MatVecs in the TT-GMRES, we always keep it on.

6. Numerical experiments

The TT-GMRES method and the numerical experiments have been implemented using the routines from TT Toolbox 2.1 (see URL http://spring.inm.ras.ru/osel/) in the MATLAB R2009b and conducted on a Linux x86-64 machine with Intel Xeon 2.00GHz CPU in the sequential mode.

6.1. Convection–diffusion (Table 1, Figures 1–7)

The first example is a 3D diffusion-convection problem with the recirculating wind

$$\begin{cases} -\alpha\Delta u + 2y(1-x^2)\dfrac{\partial u}{\partial x} - 2x(1-y^2)\dfrac{\partial u}{\partial y} = 0 \quad \text{in } \Omega = [-1,1]^3 \\ u_{y=1} = 1, \quad u_{\partial\Omega\setminus\{y=1\}} = 0 \end{cases}$$

discretized using the central-point finite difference scheme:

$$-\Delta \;\to\; -\Delta_h = (-\Delta_h^1)\otimes I\otimes I + I\otimes(-\Delta_h^1)\otimes I + I\otimes I\otimes(-\Delta_h^1)$$

$$\frac{\partial u}{\partial x} \rightarrow \nabla_h^x = \nabla_h^1 \otimes I \otimes I, \qquad \frac{\partial u}{\partial y} \rightarrow \nabla_h^y = I \otimes \nabla_h^1 \otimes I$$

$$-\Delta_h^1 = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & & \\ -1 & 2 & -1 & 0 & \\ 0 & -1 & 2 & -1 & 0 \\ & & \ddots & \ddots & \ddots \\ & & & -1 & 2 \end{bmatrix}, \quad \nabla_h^1 = \frac{1}{h}\begin{bmatrix} 0 & 0.5 & 0 & & \\ -0.5 & 0 & 0.5 & 0 & \\ 0 & -0.5 & 0 & 0.5 & 0 \\ & & \ddots & \ddots & \ddots \\ & & & -0.5 & 0 \end{bmatrix}$$

where $h = 1/(n+1)$ is a grid size. The scalar parameter α (diffusion scale) varies from 1 to $1/50$ in the numerical tests below.

We use the TT data representation (without the QTT structure), so the TT ranks of the stiffness matrix

$$\begin{aligned} -\alpha\Delta_h + \left(\operatorname{diag}\left(1-x^2\right) \otimes \operatorname{diag}\left(2y\right) \otimes I\right) \cdot \nabla_h^x \\ + \left(\operatorname{diag}\left(-2x\right) \otimes \operatorname{diag}\left(1-y^2\right) \otimes I\right) \cdot \nabla_h^y \end{aligned}$$

are bounded by 4 (the ranks of $-\Delta_h$ are all equal to 2, see [13]).

To solve this problem efficiently, we use the inversed discrete Laplacian $-\Delta_h^{-1}$ as a preconditioner (although this is not the optimal preconditioner, and the convergence depends significantly on α, the problem is tractable within our range of Reynolds numbers). To implement the inversed Laplacian in the TT format we use the quadrature from [8, 9]: if

$$\Delta_h = \Delta_h^1 \otimes I \otimes \cdots \otimes I + \cdots + I \otimes \cdots \otimes I \otimes \Delta_h^1$$

then

$$\Delta_h^{-1} \approx \sum_{k=-M}^{M} c_k \bigotimes_{p=1}^{d} \exp(-t_k \Delta_h^1)$$

where $t_k = \mathrm{e}^{k\eta}$, $c_k = \eta t_k$, $\eta = \pi/\sqrt{M}$, with the accuracy $\mathcal{O}(\mathrm{e}^{-\pi\sqrt{M}})$, so that $r_{\Delta^{-1}} = \mathcal{O}(\log^2(1/\varepsilon))$. In practice this formula can be accelerated (giving the complexity $\mathcal{O}(n \log n)$) by using Fast Trigonometric transforms (in our case of the Dirichlet boundary conditions the appropriate transform is DST-I) with all TT ranks equal to 1 [4], compressing only the diagonal matrix with inversed eigenvalues.

The timings of the TT solver are compared with the standard full-vector GMRES solver, with the same preconditioner implemented in the full format using the trigonometric transforms as well, of the complexity $\mathcal{O}(n^3 \log n)$.

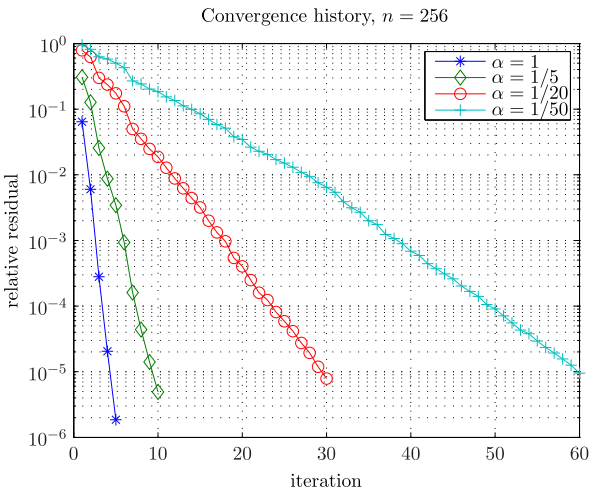

Figure 1. Convergence history for the convection example.

Table 1.
Number of iterations versus the grid size (n) and diffusion scale (α).

n \ α	1	1/2	1/5	1/10	1/20	1/50
64	5	6	10	17	30	60
256	5	6	10	17	30	60

The tensor rounding accuracy for the solution is fixed to $\varepsilon = 10^{-5}$, and the accuracy for the Krylov vectors is determined according to the relaxation strategies.

First, we check the convergence properties of the preconditioner (see Table 1 and Fig. 1).

The number of iterations is stable with respect to the grid size, but grows approximately linearly with the Reynolds number. The convergence histories for different α and $n = 256$ are given in Fig. 1.

The behaviour of the TT ranks during the iterations (we measure here the highest rank $\max_{i=1,\ldots,d-1} r_i$) of the Krylov vectors and the solution is presented on Fig. 2 and 3, respectively. The solution rank grows from 1 (zero tensor) to its stable value with a weak (approx. logarithmic) dependence on the grid size. The Krylov vector rank has its maximum at the middle iterations on the finer grids (it is also important, that it grows slightly with

the grid size, it will be reflected in the computational time), but near the end of the process, it begins to decrease due to the relaxed accuracy.

Now, consider the computational time of the TT-GMRES solver and the standard full GMRES method in MATLAB with the same Fourier-based preconditioner. The CPU time of the TT solver is presented in Fig. 4, and the log-log scale plot is in Fig. 5. The linear fitting on the log-log plot gives the experimental complexity rate $n^{1.4}$. The overhead with respect to the true linear complexity appears from the additional logarithmic terms in the Fourier transforms and approximately logarithmic grow of the TT ranks of the Krylov vectors, see Fig. 2.

The full solver manifests the complexity rate $n^{3.4}$, which is in correspondence with its theoretical estimate $\mathcal{O}(n^3 \log n)$ (see Fig. 6 for the CPU time, and 7 for the log-log scale). Notice that the full solver timings are presented only for grid sizes not larger than 256. We were not been able to perform the calculations on the grid 512^3 due to insufficient memory resources. Nevertheless, the extrapolation via the linear fit from Fig. 7 gives an estimate $2^{14} \sim 20000$ seconds for that experiment, which is about 15 times larger than the corresponding times of the TT solver.

6.2. 1D stochastic (parametric) PDE (Tables 2–5, Figures 8, 9)

In this example we consider a 1D stochastic (multiparametric) equation from [16]:

$$-\frac{\partial}{\partial x} a(x,\mathbf{y}) \frac{\partial u(x,\mathbf{y})}{\partial x} = f(x) = 1 \text{ in } \Omega \times Y = [-1,1] \times [-1,1]^d \qquad (6.1)$$

Dirichlet boundary conditions on $\partial\Omega$, and the coefficient is given as a Karhunen-Loeve expansion:

$$a(x,\mathbf{y}) = a_0(x) + \sum_{j=1}^{d} \sqrt{\lambda_j} a_j(x) y_j$$

$$a_0(x) = 1, \quad \sqrt{\lambda_j} = \frac{1}{2(j+1)^2}, \quad a_j(x) = \sin(\pi j x).$$

The problem is $(d+1)$-dimensional, and is not tractable in the full format. It is again discretized using the FD scheme with the collocation method in the parameters on uniformly distributed points. We use the preconditioner [3]:

$$P_2 = \Delta^{-1} \Gamma(1/a) \Delta^{-1}$$

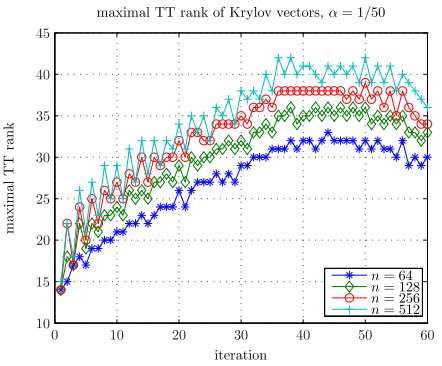

Figure 2. Maximal TT rank of the last Krylov vector, convection example.

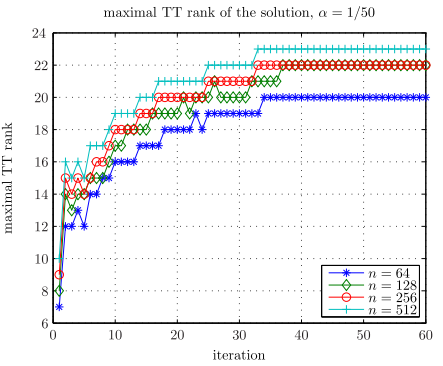

Figure 3. Maximal TT rank of the solution, convection example.

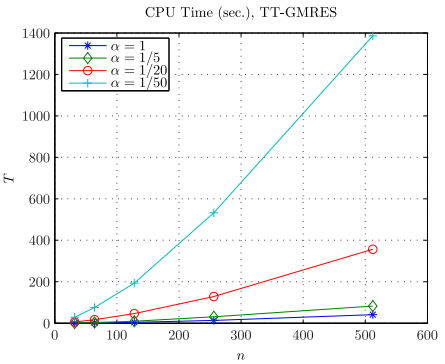

Figure 4. CPU time (sec.) of the TT solver, convection example.

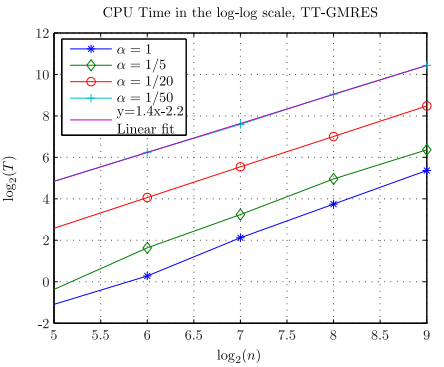

Figure 5. CPU time (sec.) of the TT solver in the log-log scale, convection example.

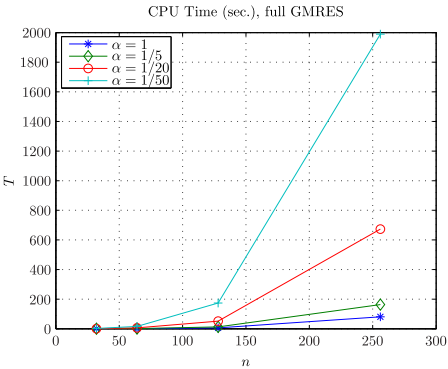

Figure 6. CPU time (sec.) of the full solver, convection example.

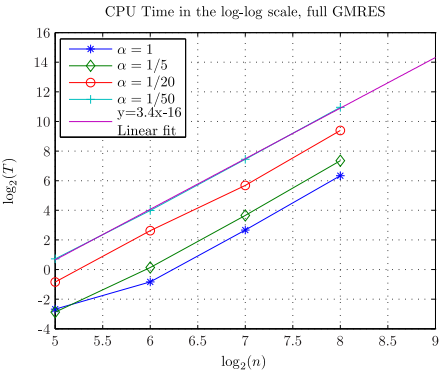

Figure 7. CPU time (sec.) of the full solver in the log-log scale, convection example.

Table 2.
$d = 20$, $\varepsilon = 10^{-5}$.

n_x	n_y	it.	T (sec.)	resid.	rank
128	64	3	129.2	3.19e-6	28
256	64	3	124.1	2.93e-6	28
128	128	3	133.8	4.64e-6	27
128	256	3	148.3	4.68e-6	28

Table 3.
$d = 40$, $\varepsilon = 10^{-5}$.

n_x	n_y	it.	T (sec.)	resid.	rank
128	64	3	413.7	2.13e-5	33
256	64	3	409.8	1.93e-5	33
128	128	3	334.7	1.51e-5	36
128	256	3	456.3	1.90e-5	33

Table 4.
$d = 80$, $\varepsilon = 10^{-5}$.

n_x	n_y	it.	T (sec.)	resid.	rank
128	64	3	1187	1.71e-5	37
256	64	3	1280	1.70e-5	36
128	128	3	1122	1.82e-5	35
128	256	3	1336	2.03e-5	33

Table 5.
Dependence on the rounding accuracy ε. Problem sizes $n_x = 128$, $n_y = 64$, $d = 20$.

ε	10^{-3}	10^{-4}	10^{-5}	10^{-6}
it.	2	2	3	3
T, sec.	7.31	16.98	129.2	384.41
resid.	2.05e-4	7.90e-5	3.19e-6	7.31e-8
rank	8	13	28	46

where $\Gamma(a)$ is a stiffness matrix of the discretized elliptic operator (6.1) with the coefficient a. The parametric inversed Laplacian reads just $\Delta_x^{-1} \otimes I_{y_1} \otimes \cdots \otimes I_{y_d}$. Moreover, we used the *QTT* format in this example, with the explicit analytic QTT representation of the 1D Δ_x^{-1} from [13]. To compute the reciprocal coefficient, we used the TT-structured Newton iterations. In the

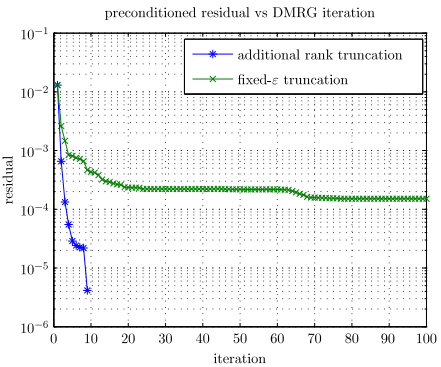

Figure 8. Relative residuals, sPDE, DMRG solvers.

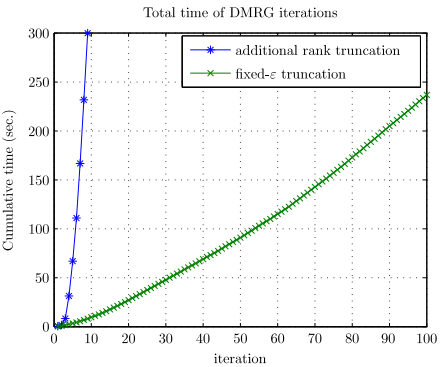

Figure 9. Cumulative times, sPDE, DMRG solvers.

following, unless specially noted (see Table 5), we fix the tensor rounding accuracy to $\varepsilon = 10^{-5}$.

This example is essentially high-dimensional, with large ranks of the solution, and what is more important, of the coefficients. Hence we have to use the DMRG compression routines. The increased-rank truncation strategy allows to keep the accuracy, while increasing the time respectively. But without it, one might get no relevant solution at all. We will demonstrate it in a comparison with the DMRG-solve algorithm.

We show in Tables 2–4 the number of iterations (it.), solution time (T, sec.), stabilized preconditioned residual (resid.) and the maximal TT rank versus the spacial n_x and parametric n_y grid sizes and the number of parameters d.

Consider the dependence on the tensor rounding accuracy ε in the case $n_x = 128$, $n_y = 64$, $d = 20$. As in the previous tables, we show the number of iterations, solution time, stabilized residual and maximal TT rank (see Table 5). With increasing accuracy, the computational time increases drastically, as it depends both on the number of iterations and on the TT ranks.

Now, consider the TT-DMRG-solver from [5] applied to the same problem $P_2\Gamma(a)u = P_2 f$ with $n_x = 128$, $n_y = 64$ and $d = 20$. Following Section 5 we compare two variants of rank truncations in superblock splitting: with a fixed ε and additional rank increasing. The convergence histories are shown in Fig. 8, and the cumulative times in Fig. 9, respectively.

We see that increasing truncation ranks improves the convergence significantly, despite the fact that the time of each iteration is larger than in the fixed-accuracy truncation.

Notice the difference in time between the GMRES and DMRG solver. To achieve the same residual $4 \cdot 10^{-6}$ the GMRES spent 129 sec., whereas DMRG (with increased ranks only) took about 300 sec. This shows the advantage of rapidly converging GMRES, provided a good preconditioner is given. It is natural that the DMRG-based approximate MatVecs (which are in fact just the DMRG truncations, up to additional structure of TT blocks, provided by their construction as matrix-by-vector multiplications) are also cheaper than the linear system solutions.

Conclusion

The adapted GMRES method in the TT format for a linear system solution has been proposed and investigated. For the method presented the error analysis has been carried out and performance improvements have been obtained with the aid of the inexact Krylov methods theory. The numerical experiments show that the method provides linear complexity with respect to the grid size in the case of TT approximation, and even logarithmic complexity with the QTT format. The method is compared with the direct ALS/DMRG-type minimization solver for the TT format. These methods manifest comparable timings and accuracies, and the GMRES method might be recommended in the cases where a good preconditioner is available.

References

1. J. Ballani and L. Grasedyck, A projection method to solve linear systems in tensor format: DFG-SPP1324 Preprint 46. Marburg, Philipps-Univ., 2010.
2. G. Beylkin and M. J. Mohlenkamp, *Algorithms for numerical analysis in high dimensions*. SIAM J. Sci. Comp. (2005) **26**, No. 6, 2133–2159.
3. S. Dolgov, B. N. Khoromskij, I. V. Oseledets, and E. E. Tyrtyshnikov, A reciprocal preconditioner for structured matrices arising from elliptic problems with jumping coefficients. *Linear Algebra Appl.* (2011) **436**, No. 9, 2980-3007.
4. S. V. Dolgov, B. N. Khoromskij, and D. V. Savostyanov, Multidimensional Fourier transform in logarithmic complexity using QTT approximation. *Preprint No. 18*. MPI MIS, Leipzig, 2011.
5. S. V. Dolgov and I. V. Oseledets, Solution of linear systems and matrix inversion in the TT-format. *Preprint No. 19*. MPI MIS, Leipzig, 2011.
6. L. Giraud, S. Gratton, and J. Langou, *A note on relaxed and flexible GMRES*. Technical report, 2004.

7. L. Grasedyck, Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl.* (2010) **31**, No. 4, 2029–2054.

8. W. Hackbusch and B. N. Khoromskij, Low-rank Kronecker-product approximation to multi-dimensional nonlocal operators, I. Separable approximation of multi-variate functions. *Computing* (2006) **76**, No. 3–4. 177–202.

9. W. Hackbusch and B. N. Khoromskij, Low-rank Kronecker-product approximation to multi-dimensional nonlocal operators, II. HKT representation of certain operators. *Computing* (2006) **76**, No. 3–4. P. 203–225.

10. W. Hackbusch and S. Kühn, A new scheme for the tensor representation. *J. Fourier Anal. Appl.* (2009), **15**, No. 5, 706–722.

11. A. L. Hageman and D. M. Young, *Applied Iterative Methods*. Academic Press, New York, 1981.

12. S. Holtz, T. Rohwedder, and R. Schneider, The alternating linear scheme for tensor optimization in the tensor train format. *SIAM J. Sci. Comp.* (2012) **34**, No. 2, A683–A713.

13. V. Kazeev and B. N. Khoromskij, Explicit low-rank QTT representation of Laplace operator and its inverse, *Preprint No. 75*. MPI MIS, Leipzig, 2010.

14. B. N. Khoromskij, $\mathcal{O}(d\log N)$–Quantics approximation of N–d tensors in high-dimensional numerical modelling. *Constr. Appr.* (2011) **34**, No. 2, 257–280.

15. B. N. Khoromskij and I. V. Oseledets, DMRG+QTT approach to computation of the ground state for the molecular Schrödinger operator. *Preprint No. 69*. MPI MIS, Leipzig, 2010.

16. B. N. Khoromskij and I. V. Oseledets, Quantics-TT collocation approximation of parameter-dependent and stochastic elliptic PDEs. *Comput. Meth. Appl. Math.* (2010) **10**, No. 4, 376–394.

17. T. G. Kolda and B. W. Bader, Tensor decompositions and applications. *SIAM Review* (2009) **51**, No. 3, 455–500.

18. D. Kressner and C. Tobler, Krylov subspace methods for linear systems with tensor product structure. *SIAM J. Matrix Anal. Appl.* (2010) **31**, 1688–1714.

19. D. Kressner and C. Tobler, Low-rank tensor Krylov subspace methods for parametrized linear systems. *Research report*. Seminar for applied mathematics, ETH Zurich, 2010.

20. O. S. Lebedeva, Block tensor conjugate gradient-type method for Rayleigh quotient minimization in two-dimensional case. *Comp. Math. Math. Phys.* (2010) **50**, No. 5, 749–765.

21. C. Lubich, *From Quantum to Classical Molecular Dynamics: Reduced Models and Numerical Analysis*. EMS, Zürich, 2008.

22. H. D. Meyer, F. Gatti, and G. A. Worth, Multidimensional Quantum Dynamics: MCTDH Theory and Applications. Wiley-VCH, Weinheim, 2009.

23. I. V. Oseledets, Compact matrix form of the d-dimensional tensor decomposition. *Preprint No. 2009-01*. INM RAS, Moscow, 2009.

24. I. V. Oseledets, Approximation of $2^d \times 2^d$ matrices using tensor decomposition. *SIAM J. Matrix Anal. Appl.* (2010) **31**, No. 4, 2130–2145.

25. I. V. Oseledets, DMRG approach to fast linear algebra in the TT–format. *Comput. Meth. Appl. Math*. (2011) **11**, No. 3, 382–393.

26. I. V. Oseledets, Tensor-train decomposition. *SIAM J. Sci. Comput.* (2011) **33**, No. 5, 2295–2317.

27. I. V. Oseledets and E. E. Tyrtyshnikov, Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM J. Sci. Comput.* (2009) **31**, No. 5, 3744–3759.

28. Y. Saad and M. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems.*SIAM J. Sci. Stat. Comput.* (1986) **7**, No. 3, 856–869.

29. V. Simoncini and D. B. Szyld, Theory of inexact Krylov subspace methods and applications to scientific computing. *SIAM J. Sci. Comput.* (2003) **25**, 454–477.

30. I. Sloan and H. Wozniakowski, When are quasi-Monte Carlo algorithms efficient for high dimensional integrals. *J. Complexity* (1998) **14**, No. 1, 1–33.

31. E. E. Tyrtyshnikov, A unifying approach to some old and new theorems on distribution and clustering. *Linear Algebra Appl.* (1996), No. 323, 1–43.

32. J. van den Eshof and G. L. G. Sleijpen, Inexact Krylov subspace methods for linear systems. *SIAM J. Matrix Anal. Appl.* (2004) **26**, No. 1, 125–153.

33. X. Wang and I. H. Sloan, Why are high-dimensional finance problems often of low effective dimension? *SIAM J. Sci. Comp.* (2006) **27**, No. 1, 159–183.