Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

Quantics-TT Approximation of Elliptic Solution Operators in Higher Dimensions

(revised version: January 2010)

by

Boris N. Khoromskij, and Ivan V. Oseledets

Preprint no.: 79 2009

Quantics-TT Approximation of Elliptic Solution Operators in Higher Dimensions

Boris N. Khoromskij[a], and Ivan V. Oseledets[b]
[a]Max-Planck-Institute for Mathematics in the Sciences,
Inselstr. 22-26, D-04103 Leipzig, Germany.{bokh}@mis.mpg.de
[b]INM RAS, Moscow, Russia. {ivan}@bach.inm.ras.ru

Abstract

In the present paper, we present the numerical analysis of the Quantics-Tensor-Train (QTT) methods for numerical solution of the elliptic equations in higher dimensions. The ε-accurate solutions in the Frobenius norm can be computed with the complexity $O(d \log^q \varepsilon^{-1})$, where $d \geq 2$ is the spatial dimension, and $q \geq 2$ is some fixed constant. This seems to be the nearly optimal asymptotical computational cost to be expected in the d-dimensional numerical simulations.

AMS Subject Classification: 65F50, 65F30, 46B28, 47A80
Key words: High dimensions, elliptic boundary value and spectral problems, tensor approximation, quantics and tensor train formats, matrix exponential.

1 Introduction

The construction of efficient and robust numerical methods for elliptic boundary value and spectral problems in higher dimensions gives rise to the challenging mathematical and algorithmic problems. The effective solution of arising computational problems can be accomplished based on the modern concept of tensor-structured multilinear algebra.

In the present paper our main goal is to demonstrate by extensive numerics that, based on the so-called quantics-TT tensor formats, the asymptotical complexity $\mathcal{O}(d \log^q \varepsilon^{-1}), q \geq 2$ can be really achieved for large dimensions d, for high accuracy $\varepsilon > 0$, and with the small constant in the front of $\mathcal{O}(\cdot)$. Another methods [3, 2, 9] can be based on the canonical decomposition [10, 4, 1]. The approximation in canonical format is known to be ill-posed and there are no robust numerical algorithms with guaranteed accuracy and cost. In three-dimensional case Tucker format [6, 5] is a good alternative for the canonical representation [15, 20, 16], but it is not directly applicable to high dimensional problems (cf. [15] for the combined Tucker-canonical model).

The *tensor-train (TT)* decomposition [21, 18] is free from the curse of dimensionality and can be computed by standard stable algorithms of linear algebra (SVD and QR decompositions). Therefore it is very promising for high-dimensional applications. The *Quantics-TT (QTT) format* [14, 19] combines the idea of "virtual levels" in auxiliary higher dimensions

with the tensor-train decomposition, and attains logarithmic complexity in the number of tensor entries. It is closely related to the approximation of functions by the sum of exponentials [11, 22], however it is much more general and the approximation has controlled error and scales linear in $\log_2 n$, where n is a number of vector entries.

In the present paper, we combine TT decomposition in d dimensions with QTT representation of "small" $n \times n$ matrices and vectors of length n for numerical solution of the elliptic boundary value and spectral problems in higher dimensions. The ε-accurate solutions in the Frobenius norm can be computed with the complexity $O(d \log^q \varepsilon^{-1})$, where $q \geq 2$ is some fixed constant. This result seems to be the nearly optimal in the sence asymptotical computational cost vs. accuracy, to be expected in the d-dimensional numerical simulations. The numerical algorithms are based on the idea of truncated preconditioned iteration [8, 12, 14] via the TT-approximation method [18, 21], and combined with the quantics reshaping of the arising multidimensional arrays [19, 14]. The asymptotical performance is demonstrated by numerous numerical examples.

The rest of the paper is organised as follows. Section describes the construction of QTT approximate elliptic operator inverse via truncated preconditioned iteration. The efficient implementation is essentially based on the new type of sinc-quadratures via recursive scheme proposed in §2.4. In §2.5, we discuss the representation of d-Laplacian inverse in the QTT and canonical-QTT matrix formats. Section 3 presents various numerical illustrations considering high dimensional Poisson and anisotropic Poisson equations, 3D convection-diffusion equation, as well as the 3D spectral problem for the Schrödinger equation for the hydrogen atom. The important issue is not the only featuring the linear dependence on the physical dimension, but also demonstration on the log-scaling in the grid size. The later allows really high resolution in FEM/FDM schemes. The future prospects of the QTT-based numerical methods are discussed in Section 4.

2 QTT approximation of elliptic operator inverse

2.1 Quantics-TT representation of tensors

The solution vectors arising in d-dimensional discretisations of elliptic equations are represented by the N-d tensors in $\mathbb{R}^{\mathcal{I}}$ that are real-valued arrays over the product index set $\mathcal{I} = I_1 \times ... \times I_d$ with $I_\ell = \{1, ..., n_\ell\}$. For the ease of presentation we set $N = n_\ell$, $\ell = 1, ..., d$.

In this paper, our favorable rank-structured representation of N-d tensors is based on the so-called TT format [21, 18]. *The rank-$(r_0, ..., r_d)$ tensor-train (TT) format* is defined in the spirit of Tucker model, but with essentially reduced "connectivity" constraints. As in the case of canonical format it scales linearly in both d and N. We describe a slight generalisation of the TT-format to the case of "periodic" index chain given by the following definition [14].

Definition 2.1 *(Tensor train/chain format). Given the rank parameter $\mathbf{r} = (r_0, ..., r_d)$, and the respective index sets $J_\ell = \{1, ..., r_\ell\}$ ($\ell = 0, 1, ..., d$), with the periodicity constraints $J_0 = J_d$. The rank-$\mathbf{r}$ tensor chain (TC) format contains all elements V in $\mathbb{W}_{\mathbf{n}} = \mathbb{R}^{\mathcal{I}}$ that can be represented as the chain of contracted products of 3-tensors over the d-fold product*

index set $J := \times_{\ell=1}^{d} J_\ell$,

$$V = \{\times_\ell\}_{\ell=1}^{d} G^{(\ell)} \quad \textit{with given 3-tensors} \quad G^{(\ell)} \in \mathbb{R}^{J_{\ell-1}\times I_\ell \times J_\ell}. \tag{2.1}$$

Denote this set of tensors by $TC[\mathbf{r}, d] \equiv TC[\mathbf{r}, \mathbf{n}, d] \subset \mathbb{W}_{\mathbf{n}}$. *The parameters* $d, \mathbf{n}$ *can be skept upon the context.*

In the case $J_0 = J_d = \{1\}$ *(disconnected chain), this construction coincides with the original definition of TT format in [21, 18], thus implying* $TT[\mathbf{r}, d] \subset TC[\mathbf{r}, d]$.

Quantics representation of N-d tensor with $N = 2^L$ is based on its binary reshaping (folding) to the auxiliary 2-D tensor of size $\underbrace{2 \times 2 \times ... \times 2}_{dL}$, leaving in higher dimensional tensor space $\mathbb{R}^{\{1,2\}^{\otimes D}}$ with "virtual" dimension $D = d \log N$ (cf. [14, 19]). The advantage of such a reshaping transform, $\mathcal{F} : \mathbb{W}_{\mathbf{n}} \to \mathbb{R}^{\{1,2\}^{\otimes D}}$, is the possibility for the low rank tensor structured representation in the D-dimensional quantics tensor space thus reducing dramatically the multilinear operational complexity from N^d to $O(d \log N)$. In particular, the exponential N-vector has quantics rank 1, hence, it can be represented by $2 \log N$ numbers as declared in the following lemma (cf. [14]).

Proposition 2.2 *For a given* $N = 2^L$, *with* $L \in \mathbb{N}_+$, *and* $c, z \in \mathbb{C}$, *the single exponential vector* $X := \{x_n := cz^{n-1}\}_{n=1}^{N} \in \mathbb{C}^N$, *can be reshaped by the successive dyadic folding to the rank-*1 $\underbrace{2 \times 2 \times ... \times 2}_{L}$*-tensor representation (shortly, to the rank-*1, 2-L *tensor),*

$$X \mapsto A = c \otimes_{p=1}^{L} \begin{bmatrix} 1 \\ z^{2^{p-1}} \end{bmatrix}, \quad A : \{1,2\}^{\otimes L} \to \mathbb{C}. \tag{2.2}$$

The number of representation parameters is reduced dramatically from N *to* $2 \log_2 N$.

In turn, the trigonometric N*-vector has the TT-rank equals to 2.*

Proposition 2.2 implies that the single exponential N-vector is exactly represented by the rank-1, 2-L tensor, hence the R-term sum of exponential vectors, $\{x_n := \sum_{k=1}^{R} c_k z_k^{n-1}\}_{n=1}^{N}$, can be exactly reconstructed by the rank-R, 2-L tensor with the storage cost $2R \log_2 N$. Clearly, the similar folding strategy can be applied to matrices and N-d tensors with $d \geq 3$.

In the case of a general vector/matrix/tensor, the resultant 2-dL folding tensor can be *approximated* in the low rank TT-format. We call such rank-structured tensor representation as the quantics-TT tensors or shortly, QTT-representation. The above mentioned gainful properties of exponential vectors indicate the class of functions to be well approximated in the QTT tensor format.

2.2 Truncated preconditioned iteration

We consider the model discrete elliptic problem of stationary type

$$\mathcal{L}U \equiv (\Delta_d + \mathcal{V})U = F, \quad U \in \mathbb{R}^{\mathcal{I}}, \quad \mathcal{I} = [1:n]^{\otimes d}, \tag{2.3}$$

where Δ_d stands for the finite difference negative Laplacian in $\mathbb{R}^d$, that allows the Kronecker rank-d representation,

$$\Delta_d = A \otimes I_n \otimes ... \otimes I_n + I_n \otimes A \otimes I_n ... \otimes I_n + ... + I_n \otimes I_n ... \otimes A, \; \Delta_d \in \mathbb{R}^{\mathcal{I}\otimes\mathcal{I}},$$

with $A = \Delta_1 = \mathrm{tridiag}\{-1, 2, -1\} \in \mathbb{R}^{n\times n}$, and I_n being the $n \times n$ identity. Notice that the numerical algorithm presented applies to the general case of non-equal grid size in different dimensions.

Here the matrix $\mathcal{V} \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ represents certain interaction potential, say, a sum of the Newton potentials centered at different spacial points in $\mathbb{R}^d$. The class of quasi-linear elliptic equations corresponding to the case $\mathcal{V} = \mathcal{V}_0 + \mathcal{V}(U)$ can be considered as well. In this case the TT-Cross approximation [21] can be applied.

Following [12, 14], we make use of the so-called $\mathcal{S}$-truncated preconditioned iteration. Specifically, we choose the manifold $\mathcal{S}$ of rank structured QTT tensors, and then perform the truncated iterations over $\mathcal{S}$ by projection of the current iterand onto this manifold.

For the linear system (2.3) the truncated preconditioned iteration takes the form

$$\widetilde{U}_{m+1} = U_m - \mathcal{B}^{-1}(\mathcal{L}U_m - F), \quad U^{(m+1)} := T_{\mathcal{S}}(\widetilde{U}^{(m+1)}),$$

where the truncation operator $T_{\mathcal{S}} : \mathbb{R}^{\mathcal{I}} \to \mathcal{S}$ is defined by the nonlinear approximation procedure that "projects" the respective vectors onto the manifold $\mathcal{S}$ by using SVD/QR-based algorithm in [18]. The initial guess is chosen as an element $U_0 \in \mathcal{S}$. Given the quadratue parameter $M \in \mathbb{N}_+$, the preconditioner $\mathcal{B}^{-1}$ can be specified as a rank-structured approximate inverse of the d-Laplacian, $\mathcal{B}^{-1} = \mathcal{B}_M$, that will be represented explicitly by low TT-rank expansion [14], using the family of matrix exponentials, $\exp(-t_k\Delta_1)$, $t_k > 0$,

$$\mathcal{B}_M := \sum_{k=-M}^{M} c_k \bigotimes_{\ell=1}^{d} \exp(-t_k\Delta_1), \quad t_k = e^{k\mathfrak{h}}, \quad c_k = \mathfrak{h}t_k, \quad \mathfrak{h} = \pi/\sqrt{M}. \tag{2.4}$$

In the following, to specify the explicit dependence on a sequence $\{t_k\}$, we call the respective representation as the t_k-quadrature. The numerical illustrations on the practical efficiency of this quadrature in higher dimensions are given in Section 3.

In the present paper, we also consider the truncated Green function iteration (cf. [13] and refrences therein) for solving the spectral problem,

$$(\Delta_d + \mathcal{V})U = EU, \quad \|U\| = 1, \quad U \in \mathbb{R}^{\mathcal{I}}. \tag{2.5}$$

This iteration takes the form

$$\widetilde{U}_{m+1} = (\Delta_d - E_m I)^{-1}\mathcal{V}U_m, \quad U_{m+1} := T_{\mathcal{S}}(\widetilde{U}_{m+1}), \quad U_{m+1} := \frac{U_{m+1}}{||U_{m+1}||},$$

and E_{m+1} is recomputed at each step as a Rayleigh quotient:

$$E_{m+1} = \langle \mathcal{L}U_{m+1}, U_{m+1}\rangle.$$

The particular numerical illustrations are presented in Section 3 for the Schrödinger equation describing the hydrogen atom.

2.3 Computation of matrix exponentials: basic schemes

There are several ways to compute the family of the matrix exponentials $\exp(-t_k\Delta_1)$ in the QTT format. For "small" one-dimensional mode sizes n ($n \le 256$ on our current-state

computational environment) such computation can be performed fast by using standard approaches, even separately for different $k = -M, ..., M$. For the symmetric case, the matrix Δ_1 can be diagonalized ones and for all at $\mathcal{O}(n^3)$ cost and then all exponents are computed cheaply (only the exponents of diagonal matrices have to be computed).

For the nonsymmetric case, as in the convection-diffusion equations, the Schur decomposition can be used to reduce the matrix in question to triangular form and compute exponent of the triangular matrix by known methods. This would yield $\mathcal{O}(n^3)$-algorithm which is nominally high but practically viable for moderate n. Finally, the conversion of a full $n \times n$ matrix into QTT format takes $\mathcal{O}(n^2)$ time, using the algorithm of [19].

For large n another approach is needed. A good idea is to use some truncated iteration for the matrix exponential. First, it is easy to convert the Laplace operator Δ_1 to the QTT format (it can be done by simple algebraic manipulations). For example, standard 3-point stencil for the second derivative gives a matrix with QTT ranks not larger than 3. For the (approximate) computation of the matrix exponential scaling-and-squaring method can be used, which is the most suitable for rank-structured format. The matrix $B = -t_k \Delta_1$ is first scaled by $\frac{1}{2^s}$ with $s \approx \log_2 ||B||$, so that

$$\frac{1}{2^s}||B|| \leq 1.$$

Then for a scaled matrix $C = \frac{1}{2^s}B$ the exponent is computed by Taylor series

$$\exp(C) \approx \sum_{k=0}^{N} \frac{C^k}{k!},$$

in N multiplications using a Horner rule:

$$C_k = \frac{1}{k} C_{k+1} C + I, \; k = N-1, N-2, \ldots, 1, 0,$$

and $C_N = I$. Of coarse, all multiplications are performed in a structured format (QTT format here) and the compression is done at each step. After $E_N = C_0$ is computed, the final exponent is obtained by squaring:

$$\exp(B) = \exp(C)^{2^s}$$

in $s \approx \log_2 ||B||$ steps. This iteration requires $N + s$ matrix-by-matrix multiplications with compression, and if we assume that approximate ranks are bounded by r at each step, then the complexity is

$$\mathcal{O}(r^6 \log n)$$

operations, since after multiplication of two QTT matrices with compression ranks r the result has compression ranks r^2, and recompression requires $\mathcal{O}(r^6 \log n)$ operations. It is worth to note that the recompression step dominates over the multiplication step, since all mode sizes are equal to 4. The computation scheme for a matrix exponential of a QTT matrix is given in Algorithm 2.1.

Unfortunately, the number of outer iterations (i.e. s) can be large. For example, for $M = 30$, that usually gives accuracy of order 10^{-7} for the quadrature, t_k can be large (of order 10^7), and the number of additional iterations can be rather high. Thus, a special quadrature is needed.

Algorithm 2.1 Computation of a matrix exponential in QTT format

Require: Matrix A in QTT format, approximation parameter ε, parameter N (for Gorner rule).

Ensure: $\Phi = \exp A$ in QTT format

Estimate $||A||_2$ by structured QTT power iteration:
Scale: $s = \lceil \log_2 B \rceil$, $B = \frac{1}{2^s} A$.
{Compute $\exp B$ by Horner rule:}
$C_N = I$.
for $k = N-1$ to 1 step -1 **do**
    $C_k = \frac{1}{k} C_{k+1} C + I$
    $C_k = \mathrm{TT\_COMPRESS}(C_k, \varepsilon)$
end for
{Compute $\exp A$}
$\Phi := C_0$
for $k = 2$ to s **do**
    $\Phi := \mathrm{TT\_COMPRESS}(\Phi^2, \varepsilon)$
end for

2.4 A new quadrature via recursive scheme

Suppose we want to have a simple recursion that connects previously computed exponents $\exp(-t_p \Delta_1)$, $p < k$, with the new one for the number k. Denote these matrices by Φ_k,

$$\Phi_k = \exp(-t_k \Delta_1).$$

The simplest possible recursion is

$$\Phi_k = \Phi_{k-1}^2,$$

corresponding to

$$t_k = 2t_{k-1}.$$

This is possible by choosing M such that

$$e^{\mathfrak{h}} = 2,$$

or equivalently,

$$\mathfrak{h} = \log 2 = \frac{\pi}{\sqrt{M}},$$

therefore

$$M = \left(\frac{\pi}{\log 2} \right)^2 \approx 20.54.$$

Since M should be integer, so we select $M = 21$ or $M = 20$ and slightly modify $\mathfrak{h}$ ($\mathfrak{h} = \log 2$ to make the recursion exact. This yields a new quadrature formula with

$$t_k = 2^k, \; c_k = 2^k \log 2, \; k = -M, \ldots, M. \tag{2.6}$$

This quadrature will be called 2^k-*quadrature*.

The accuracy of quadrature formula (2.6) depends on the interval where it is considered (i.e. on the spectrum of Δ_1), but it always gives an excellent approximate inverse to serve as preconditioner (with relative accuracy not smaller than 10^{-3}). The special structure of the quadrature nodes allows the computation of all exponents fast, in $2M+1$ multiplications — in exact arithmetic. However, in the approximate case, the error may accumulate during the squaring (since for $k=-M$ the exponents are close to the identity matrix), and a mixed scheme is more preferable. Up to some k_0 the exponents are computed by scaling and squaring method, and after that they are just the squares of the previously computed exponents.

A similar approach can be adopted to obtain a more accurate quadrature. Another possible recurrence relation is

$$\Phi_k = \Phi_{k-1}\Phi_{k-2},$$

or

$$t_k = t_{k-1} + t_{k-2}.$$

If we denote by a the exponent $e^{\mathfrak{h}}$, then a should satisfy the quadratic equation

$$a^2 - a - 1 = 0,$$

therefore a is a *golden ratio*:

$$a = \rho = \frac{1+\sqrt{5}}{2} \approx 1.6180.$$

The corresponding M is larger:

$$M = \left(\frac{\pi}{\log \rho}\right)^2 \approx 42.62,$$

so one can choose $M=42$ or $M=43$. The corresponding quadrature weights and nodes are

$$t_k = \rho^k,\ c_k = \rho^k \log \rho.$$

This quadrature formula is around 1000 times more accurate than 2^k-quadrature. The number of quadrature points can be slightly decreased, since the norm of several first and last summands is negligible.

There are other possible recursions:

$$\Phi_k = \Phi_{k-2}^2,\ \text{ i.e. } t_k = 2t_{k-2}, \quad \text{and} \quad \Phi_k = \Phi_{k-2}\Phi_{k-3},\ \text{ i.e. } t_k = t_{k-2} + t_{k-3},$$

and so on, which lead to increased M and increased accuracy, yielding a whole family of "cheap" quadrature rules.

A scheme for computing the family of matrix exponentials $\exp(-t_k\Delta_1)$ for 2^k-quadrature is given in Algorithm 2.2. All other cheap quadrature formulae are implemented in the same fashion.

Remark 2.3 *The number of matrix-by-matrix multiplications in Algorithm 2.2 is*

$$N(M+k_0)N + (M-k_0-1),$$

and it seems that choosing k_0 to be as close to $-M$ as possible gives lower complexity. However, it may lead to considerable loss of accuracy. No theoretical estimates are available yet, but numerical experiments confirm that $k_0=0$ is generally a good choice, at least for $A=\Delta_1$.

Algorithm 2.2 Computations of a family of matrix exponentials for 2^k-quadrature

Require: Matrix A is in QTT format, accuracy parameter $\varepsilon > 0$, parameter $-M \le k_0 \le M$, number of quadrature points M

Ensure: Approximation to matrices $\Phi_k = \exp(-2^k A)$, $k = -M, \ldots, M$

1: **for** $k = -M$ to k_0 **do**
2: {Compute $\Phi_k = \exp(-2^k A)$ using Algorithm 2.1}
3: **end for**
4: **for** $k = k_0 + 1$ to M **do**
5: $\Phi_k :=$ TT_COMPRESS$(\Phi_{k-1}^2, \varepsilon)$
6: **end for**

2.5 Laplacian inverse and Canonical-to-QTT conversion

As the first example of the proposed technique consider the discretization of d-dimensional Laplace operator on uniform grid with $n = 2^D$ points in QTT format and its inverse via sinc quadrature. The simplest but interesting case (up to a scaling factor) is

$$\triangle_1 = \text{tridiag}[1, -2, 1],$$

which is the standard discretization of the one-dimensional Laplace operator. The compression of the representation (2.4) to QTT format is performed in two steps. Each individual exponent $\exp(-t_k \Delta_1)$ is computed in QTT format, then each summand is computed by taking Kronecker products of "one-dimensional" exponents (this can be done simply by concatenation of corresponding cores, [19], so no work to be done). Then the new tensor is added to the current approximation, which is compressed with some accuracy parameter ε to avoid unnecessary rank growth. It is observed numerically that during this procedure the rank do not grow. This procedure is valid not only for the particular case of the inverse to the Laplacian operator, but also for the compression of arbitrary tensor in the canonical format to QTT format, so we give it here in the general setting.

The whole method is summarized in Algorithm 2.3. The computational complexity (provided that all matrix exponents are already computed in QTT format) is estimated as $\mathcal{O}(RDr^3)$, where r is the typical TT rank in the computations, and $R = 2M + 1$ is the number of summands (initial canonical rank). Note that the mode size $n = 2^D$ enters the estimate only logarithmically.

After the representation for the inverse is computed, it can be stored and applied to any QTT vector very cheap. Therefore, the first step, i.e. compression prpcedure, can be considered as a precomputation step, since usually the inversion of the Laplace operator is required many times during the iteration process. Hence, the approximate inverses can be precomputed for a range of acceptable values of d and n and stored.

Algorithm 2.3 Canonical-to-QTT compression

Require: d-dimensional tensor $\mathbf{A}$ in canonical format with factor matrices $U_1, U_2, \ldots, U_d$ of sizes $2^D \times R$, required accuracy $\varepsilon > 0$.

Ensure: Approximation $\mathbf{B}$ in QTT format of $\mathbf{A}$.

{Rank-one update QTT}
Set $\mathbf{B}$ to a zero TT tensor with dimension dD and mode sizes 2.
for $k = 1$ to R **do**
  {Compress rank-one term}
  Set TT to a zero TT tensor with dimension dD and mode sizes 2.
  for $i = 1$ to d **do**
    $V := U_i(:, k)$, compress V to QTT with parameter ε.
    TT := TT $\times$ V (by merging the cores)
  end for
  $\mathbf{B} := \mathbf{B} +$ TT, in TT format (the core sizes are doubled).
  $\mathbf{B} :=$ TT_COMPRESS$(\mathbf{B}, \varepsilon)$.
end for

3 Numerical experiments

3.1 Poisson equation in high dimension

As a first example we consider approximate solution of Poisson equation. In Table 3.1 we present numerical results for $d = 3$ and grid sizes up to 2^{10}. The right-hand side is a vector of all ones. We present timings (in seconds) for the construction of matrix exponentials ("Step 1"), for the computation of the Canonical-to-QTT conversion (QTT). This can be considered as a precomputational step. The computed QTT approximate inverse can be stored. Time required for a solution of a system with a single right-hand side is also given. The quality of the solution can be measured in two ways. The simplest is the relative residue,

$$||Ax - f||/||f||.$$

However, in the as the order of the matrix grows, so does the condition number, and we can have a small relative error in the solution itself, but large residue. Usually, the case with known analytical solution is considered, and the L_2 error is computed. However, in our case we can provide a simple and cheap error estimator for the solution:

$$||x - \widehat{x}|| \approx ||X(Ax - f)||,$$

where X is the computed approximate inverse.

n	Step 1	Step 2	Time for sol	Residue	Relative L_2 error
2^5	2.12	0.46	0.03	1.5e-04	7.2e-06
2^6	4.83	0.98	0.07	2.0e-04	7.5e-06
2^7	7.34	1.66	0.11	7.4e-03	6.3e-06
2^8	10.01	2.67	0.19	2.1e-04	8.3e-06
2^9	12.43	7.68	0.36	2.0e-04	1.0e-05
2^{10}	18.71	27.89	0.49	1.7e-03	1.8e-05

Table 3.1: Numerics for 3D Poisson equation, using 2^k-quadrature

The analogous results are presented for $d = 10$ in Table 3.2.

n	Step 1	Step 2	Time for sol	Residue	Relative L_2 error
2^5	1.90	2.29	0.14	3.34e-05	8.79e-06
2^6	2.77	4.28	0.34	5.07e-05	8.23e-06
2^7	4.68	8.56	0.58	1.18e-02	7.35e-05
2^8	6.94	12.98	0.78	4.42e-01	7.81e-04
2^9	9.99	24.12	1.17	5.97e+00	1.75e-02
2^{10}	13.32	45.37	1.48	1.0774e+00	6.23e-01

Table 3.2: Numerics for 10D Poisson equation, using 2^k-quadrature

However, when we go to larger d, then the method that uses Canonical-to-QTT compression does not work at all. It is interesting to explain why it is happening, since we have a very good approximation to the inverse matrix in Frobenius norm:

$$||X - \widehat{X}|| \leq \varepsilon ||X||_F.$$

The problem is that a good approximation of the inverse matrix in the Frobenius does not guarantee that the solution Xf is good enough. The problem is that the d-dimensional Laplace operator (and it inverse) has eigenvalues with high multiplicity. Indeed, the eigenvalues of the inverse to the d-dimensional discretization of the Laplace operator are

$$\Lambda(k_1, k_2, \ldots, k_d) = \frac{1}{\lambda_{k_1} + \lambda_{k_2} + \ldots + \lambda_{k_d}}, \quad 1 \leq k_\ell \leq n,$$

where $\lambda_k, k = 1, \ldots, n$ are the eigenvalues of $\triangle_1$. If k_ℓ are all distinct, then the corresponding eigenvalue has multiplicity $d!$. For example, minimal and maximal eigenvalues have multiplicity 1. The approximation in Frobenius norm "captures" only eigenvalues of high multiplicity, for large d, whereas eigenvalues that lie closer to the border of the spectrum are not approximated. Even setting the approximation parameter ε to machine precision 10^{-15} will not help, since the required resolution is of order $\mathcal{O}(1/d!)$. This explains the deterioration of the error for ten-dimensional equation compared to three-dimensional one. The conclusion is that *approximation in Frobenius norm is not well suited for high-order matrices.* However, the approximation can be applied for construction of $O(d \log N)$-preconditioner. Next table demonstrate the effect of preconditioning in the case of huge grids.

n	Step 1	Step 2	Time for sol	Residue	Relative L_2 error
2^{11}	21.87	38.25	0.80	9.5e-03	4.3e-05
2^{12}	28.08	47.30	0.80	2.5e-02	1.4e-04
2^{13}	33.60	53.90	1.05	1.9e-01	5.3e-04
2^{14}	45.15	60.44	1.21	5.4e-00	2.1e-03
2^{15}	47.73	63.02	1.45	2.0e+01	8.3e-03

Table 3.3: Numerics for 3D Poisson equation, using 2^k-quadrature on large grids, $n \geq 2^{11}$.

For accurate approximation some other norm is needed. The quadrature formula gives uniform approximation of the spectrum, i.e. good approximation in the spectral norm:

$$||\Delta_d^{-1} - B||_2 \leq \varepsilon ||B||_2.$$

Further approximation of B in Frobenius norm destroys this property, therefore an good approximation by QTT format is needed in *spectral norm*:

$$C = \arg\min_{D \in \mathcal{S}} ||B - D||_2.$$

However, we only have fast approximation procedure in the Frobenius norm, as well as some linear algebra operations, like multiplication of matrices in QTT format. Based on these ingredients, it is possible to design a heuristic iterative algorithm for best approximation in spectral norm by structured matrices [17]. It requires few matrix-by-matrix multiplications at each step and a best approximation to a certain matrix in Frobenius norm which are readily available.

However, this approach is quite expensive, and a more simple remedy is to used mixed Can-QTT (or simply CQTT) format, when we approximate each individual factor in QTT format but do not assemble the full QTT matrix. It is easy to design an algorithm for matrix and vector operations in such format, using algorithms from TT Toolbox. The results for $d = 100$, given in Table 3.4 confirm that no collapses occur with this mixed format. Now, as a precomputation, we have time for evaluation of matrix exponentials. As a pay off, the solution time is now slightly higher, but still it is linear in d and logarithmic in grid size n.

n	Precomp	Time for sol	Residue	Relative L_2 error
2^5	1.70	1.57	9.1e-06	8.9e-06
2^6	2.56	1.98	7.8e-06	7.3e-06
2^7	4.19	2.45	7.3e-06	7.1e-06
2^8	6.14	2.98	6.6e-06	7.0e-06
2^9	8.37	3.52	8.7e-06	7.0e-06
2^{10}	10.81	4.02	9.4e-06	7.0e-06

Table 3.4: Numerics for 100D Poisson equation, using 2^k-quadrature

3.2 Anisotropic diffusion

The next numerical examples concern Poisson equation with different diffusion coefficients along each mode:

$$\Delta_d = a_1 A \otimes I_n \otimes ... \otimes I_n + a_2 I_n \otimes A \otimes I_n ... \otimes I_n + ... + a_n I_n \otimes I_n ... \otimes A,$$

where $A = \triangle_1$. The same approach is used to construct the approximate inverse, however we have to compute the following matrix exponentials:

$$\exp(-a_\ell t_k \triangle_1), \quad \ell = 1, \dots, d, \;\; k = -M, \dots, M,$$

i.e. there are $d(2M+1)$ exponents to compute, so the precomputation step is considerably slower, but still free from the exponential dependence on d. We use mixed Can-QTT format for the computations in 3D case, however, full QTT format is faster). The results for $d = 3$ are given in Table 3.5

n	Precomp	Time for sol	Residue	Relative L_2 error
2^5	16.48	0.07	8.6e-06	1.1e-05
2^6	30.39	0.08	8.3e-06	9.5e-06
2^7	52.85	0.13	8.4e-06	7.8e-06
2^8	74.41	0.15	8.4e-06	6.3e-06
2^9	93.88	0.19	8.4e-06	5.2e-06
2^{10}	103.80	0.19	8.4e-06	4.7e-06

Table 3.5: Numerics for 3D anisotropic diffusion, diffusion coefficients $1, 0.01, 0.0001$

The results for $d = 100$ are given in Table 3.6

n	Precomp	Time for sol	Residue	Relative L_2 error
2^5	234.54	1.52	8.1e-06	9.1e-06
2^6	327.47	1.82	8.1e-06	8.7e-06
2^7	499.01	2.27	8.4e-06	8.3e-06
2^8	576.82	2.71	8.3e-06	8.1e-06
2^9	710.18	2.93	8.4e-06	7.9e-06
2^{10}	757.36	3.37	8.4e-06	8.8e-06

Table 3.6: Numerics for 100D anisotropic diffusion, diffusion coefficients $c_i = 0.5^i, i = 0, \dots, 99$

As we can see, the timings depend from d at most linearly, and from the grid size – logarithmically, as expected.

3.3 Convection-diffusion problem

We can also treat problems convection-diffusion problems which lead to systems with non-symmetric matrices. As a model example, we consider the equation in form

$$Au = f,$$

where

$$A = -\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2} + \sum_{i=1}^{d} c_i \frac{\partial}{\partial x_i},$$

with zero boundary conditions, considered on d-dimensional unit cube. The discretization with second-order Fromm's scheme [7] give rise to the matrix of form

$$B = \sum_{i=1}^{d} B_i \equiv A_1 \otimes I_n \otimes I_n + I_n \otimes A_2 \otimes I_n + I_n \otimes I_n \otimes A_3, \tag{3.1}$$

where each B_i acts along the i-th mode, and corresponding "one-dimensional" matrices A_i are of form

$$A_i = \frac{1}{h^2}\triangle_1 + \frac{c_i}{h} \begin{pmatrix} \frac{3}{4} & -\frac{5}{4} & \frac{1}{4} & & \\ \frac{1}{4} & \frac{3}{4} & -\frac{5}{4} & \frac{1}{4} & \\ \ddots & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \frac{1}{4} & \\ & & \frac{1}{4} & \frac{3}{4} & \end{pmatrix},$$

where h is a mesh size. The the same quadrature approach to construct the quantics-inverse is used, since B_i commute and the computation reduces to the computation of several matrix exponential from "one-dimensional matrices" $t_k A_i$. Each A_i is easily transformed into the QTT format, and the exponentials are computed via the Algorithm 2.2. However, since matrices A_i are non-symmetric, the quadrature performs worse, than in the case with zero convection terms. The numerical results are presented in Table 3.7. We used the same quadrature with $M = 20$ and Can-QTT matrix format.

n	Precomp	Time for sol	Residue	Relative L_2 error
2^5	7.60	0.07	8.3e-04	3.0e-03
2^6	13.16	0.08	3.0e-04	2.8e-03
2^7	20.47	0.13	1.1e-04	2.6e-03
2^8	27.62	0.15	4.2e-04	2.5e-03
2^9	32.55	0.16	1.7e-05	2.3e-03
2^{10}	35.90	0.20	9.9e-06	2.2e-03

Table 3.7: Numerics for 3D convection-diffusion problem, convection coefficients $c_1 = c_2 = c_3 = 10$

3.4 Schrödinger equation for the hydrogen atom

Another example is the Schrödinger equation for the hydrogen atom. It has form

$$(-\frac{1}{2}\Delta + V(r))\psi = E\psi, \quad \psi \in H_0^1(\mathbb{R}^3),$$

where $V(r) = \frac{1}{r}$, with r being the Euclidean distance in $\mathbb{R}^3$, and the eigenfunction related to minimal eigenvalue is sought. The solution is known: $E = \frac{1}{2}$ and $\psi = Ce^{-r}$, so the accuracy can be controled. First, the infinite region is restricted to some cube $[-20, 20]^3$, and a tensor uniform grid is introduced. To avoid singularity at 0, the grid is shifted by half of the step size. The discretization of the Laplacian operator is given by Δ_3 with zero boundary conditions (since the exponential decay of the solution). The potential $V(r)$ is discretised by collocation scheme leading to the diagonal matrix V consisting of the values of the function $\frac{1}{r}$ at grid points. For this example the iterative recompression procedure is required, since the matrix is only approximately separable.

We apply Green iterations to compute the approximate eigenvalue:

$$\psi_{k+1} = (\frac{1}{2}\Delta_3 - \widehat{E}I)^{-1}V\psi_k, \quad \psi_{k+1} := T_{\mathcal{S}}(\psi_{k+1}), \quad \psi_{k+1} := \frac{\psi_{k+1}}{||\psi_{k+1}||},$$

and $\widehat{E}$ is recomputed at each step as a Rayleigh quotient:

$$\widehat{E} = (\mathcal{L}\psi_{k+1}, \psi_{k+1}).$$

For simplicity we take E as an exact eigenvalue, and compute the inverse to the shifted Laplacian by the same 2^k-quadrature.

Note, that the shift E here improves the conditioning of the matrix and the quality of the approximation to the inverse.

Also the potential $V(r)$ is cast into the TT format by applying a suitable quadrature rule for the representation

$$\frac{1}{r} = \frac{2}{\sqrt{\pi}} \int_0^\infty e^{-t^2r^2} dt.$$

The results of numerical experiments are given in Table 3.8. The QTT representation for ψ is truncated at accuracy $\varepsilon = 10^{-10}$ at each iteration. The time for one iteration is now higher than in other examples, since the QTT ranks of involved vectors are considerably larger. We still observe logarithmic scaling in n.

n	Time for 1 iteration	Iter	Eigenvalue error
2^7	8.5	8	6.1e-03
2^8	13	8	1.5e-03
2^9	18	8	4.0e-04
2^{10}	25	8	1.0e-04

Table 3.8: Schrödinger equation for hydrogen atom

4 Conclusions and future work

In this paper, we showed by numerical experiments that QTT format can be applied to different high-dimensional problems discretized on tensor grids with n^d elements (n up to 2^{10}, d up to 100). For fixed accuracy $\varepsilon > 0$, it provides almost optimal complexity $\mathcal{O}(d \log^{\alpha} \varepsilon^{-1})$ with small computational times even for large d and prototype MATLAB implementation.

The proposed approach is applicable for various large scale problems, not only in high dimensions but also for small "physical" dimension (1, 2, 3), but enormous grid size (up to $n = 2^{15}$). The method is very promising for different applications including anisotropic and convection-diffusion elliptic equations, the Stokes and Navier-Stokes problems, stochastic PDEs and equations of density functional theory, yielding a log-linear complexity for all of them. This is a subject of future works.

References

[1] B. Bader and T. Kolda, *Tensor decompositions and applications*, SIAM Review, 51 (2009). to appear.

[2] G. Beylkin and M. Mohlenkamp, *Algorithms for numerical analysis in high dimensions*, SIAM Journal on Scientific Computing, 26 (2005), pp. 2133–2159.

[3] G. Beylkin and M. J. Mohlenkamp, *Numerical operator calculus in higher dimensions*, Proc. Nat. Acad. Sci. USA, 99 (2002), pp. 10246–10251.

[4] P. Comon and B. Mourrain, *Decomposition of quantics in sums of powers of linear forms*, Signal Processing, 53 (1996), pp. 93–107.

[5] L. de Lathauwer, B. de Moor, and J. Vandewalle, *A multilinear singular value decomposition*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1253–1278.

[6] ——, *On best rank-1 and rank-($R_1, R_2, ..., R_N$) approximation of high-order tensors*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1324–1342.

[7] J. Fromm, *A method for reducing dispersion in convective difference schemes(Difference schemes implementation through fractional time step and phase error methods reduce dispersion)*, Journal of Computational Physics, 3 (1968), pp. 176–189.

[8] W. Hackbusch, B. N. Khoromskij, S. A. Sauter, and E. E. Tyrtyshnikov, *Use of Tensor Formats in Elliptic Eigenvalue Problems*, Preprint 78, MIS MPI, Leipzig, 2008.

[9] W. Hackbush, B. N. Khoromskij, and E. E. Tyrtyshnikov, *Hierarchical Kronecker tensor-product approximations*, J. Numer. Math., 13 (2005), pp. 119–156.

[10] F. L. Hitchcock, *The expression of a tensor or a polyadic as a sum of products*, J. Math. Phys, 6 (1927), pp. 164–189.

[11] T. Jiang, N. Sidiropoulos, and J. ten Berge, *Almost-sure identifiability of multidimensional harmonic retrieval*, IEEE Transactions on Signal Processing, 49 (2001), pp. 1849–1859.

[12] B. Khoromskij, *Tensor-structured Preconditioners and Approximate Inverse of Elliptic Operators in* $\mathbb{R}^d$, J. Constr. Approx, 30 (2009), pp. 599–620.

[13] B. N. Khoromskij, *On tensor approximation of green iterations for kohn-sham equations*, Computing and visualization in science, 11 (2008), pp. 259–271.

[14] ——, $\mathcal{O}(d \log N)$*-quantics approximation of N-d tensors in high-dimensional numerical modelling*, Preprint 40, MPI MIS, 2009.

[15] B. N. Khoromskij and V. Khoromskaia, *Multigrid accelerated tensor approximation of function related multidimensional arrays*, SIAM Journal on Scientific Computing, 31 (2009), pp. 3002–3026.

[16] B. N. Khoromskij, V. Khoromskaia, and H. J. Flad, *Numerical solution of the Hartree-Fock equation in multilevel tensor-structured format*, Preprint 44, MPI MIS, 2009.

[17] I. V. Oseledets, *Approximation by low-rank matrices in nonstandard norms.* in progress, 2009.

[18] ——, *Compact matrix form of the d-dimensional tensor decomposition*, Preprint 1, INM RAS, Mar. 2009.

[19] ——, *Tensors inside of matrices give logarithmic complexity*, Preprint 5, INM RAS, 2009.

[20] I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov, *Tucker dimensionality reduction of three-dimensional arrays in linear time*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 939–956.

[21] I. V. Oseledets and E. E. Tyrtyshnikov, *Breaking the curse of dimensionality, or how to use SVD in many dimensions*, SIAM J. Sci. Comp., 31 (2010), pp. 3744–3759.

[22] J. M. Papy, L. De Lathauwer, and S. Van Huffel, *Exponential data fitting using multilinear algebra: the single-channel and multi-channel case*, Numer. Linear Algebra Appl., 12 (2005), pp. 809–826.