Electronic Transactions on Numerical Analysis.
Volume 46, pp. 215–232, 2017.

ISSN 1068–9613.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

ON THE APPROXIMATION OF FUNCTIONALS OF VERY LARGE HERMITIAN MATRICES REPRESENTED AS MATRIX PRODUCT OPERATORS*

MORITZ AUGUST†, MARI CARMEN BAÑULS‡, AND THOMAS HUCKLE†

Abstract. We present a method to approximate functionals $\mathsf{Tr} f(A)$ of very high-dimensional Hermitian matrices A represented as Matrix Product Operators (MPOs). Our method is based on a reformulation of a block Lanczos algorithm in tensor network format. We state main properties of the method and show how to adapt the basic Lanczos algorithm to the tensor network formalism to allow for high-dimensional computations. Additionally, we give an analysis of the complexity of our method and provide numerical evidence that it yields good approximations of the entropy of density matrices represented by MPOs while being robust against truncations.

Key words. tensor decompositions, numerical analysis, Lanczos method, Gauss quadrature, quantum physics

AMS subject classifications. 65F60, 65D15, 65D30, 65F15, 46N50, 15A69

1. Introduction.

Approximating functionals of very large matrices is an important problem in many fields of science, such as network analysis [3, 9, 11, 26] or quantum mechanics [33, 37]. In many cases, the respective matrices are Hermitian due to either the underlying physical properties of the systems they describe or the way they are constructed from, e.g., a graph. Naturally, as the dimensionality of the matrices becomes very high, i.e., several tens or hundreds of thousands and above, explicit methods of function evaluation, like exact diagonalization, break down and approximations must be made.

One paradigm for the approximation of high-dimensional matrices that has gained a lot of attention especially in the quantum information, condensed matter, and numerical linear algebra communities are tensor network representations [2, 15, 18, 27, 33, 37]. Among the class of tensor networks, matrix product states (MPS) and matrix product operators (MPO) count among the best established methods. These representations approximate large tensors by contractions of multiple low-rank tensors in a row and have been shown to yield efficient parametrizations of many relevant states of quantum many-body systems [17, 28, 35].

In this work, we introduce a novel method to approximate functionals of the form $\mathsf{Tr} f(A)$ where we assume $f : \mathbb{C}^N \times \mathbb{C}^N \to \mathbb{C}^N \times \mathbb{C}^N$ to be smooth and defined on the spectrum of A as well as A to be Hermitian and given in MPO format. For our method, we have reformulated a block version of the Lanczos algorithm in MPO/MPS-representation. This particular block Lanczos algorithm will be referred to as global Lanczos algorithm in the following and has already been used to approximate functionals of the given form for explicitly stored matrices [4, 8]. Rewriting it for the tensor network formalism, however, allows us to consider block vectors of size identical to A, which was previously prohibitive. Our method is thus able to approximate $\mathsf{Tr} f(A)$ for certain $f(A)$ requiring only one carefully selected starting block vector. This means that we get rid of the approximation error induced by the need to combine the results obtained for multiple different starting block vectors. At the same time, we find the numerical error induced by the MPS/MPO representation to be comparably small. Our method can be applied whenever A is efficiently approximated by an MPO. We will in the following, however, focus on the case where A has directly been defined as an MPO.

The rest of this work is structured as follows: after a basic introduction to matrix product states and operators in Section 2, we will introduce the block Lanczos method we employ in

*Received December 1, 2016. Accepted May 5, 2017. Published online on July 7, 2017. Recommended by G. Rodriguez.

†Department of Informatics, Technical University of Munich, 85748 Garching, Germany ({august, huckle}@in.tum.de).

‡Max Planck Institute for Quantum Optics, 85748 Garching, Germany (mari.banuls@mpq.mpg.de).

ETNA
Kent State University and
Johann Radon Institute (RICAM)

this work and explain its connection to Gauss quadrature in Section 3. Following this, we will then state our method in Section 4, show how we have reformulated the global Lanczos method in the tensor network formalism, and discuss its properties as well as give an analysis of its complexity. Finally, in Section 5 we will provide numerical evidence for the fast and robust convergence of our method for the case of the trace-norm and von-Neumann entropy of quantum mechanical density matrices. We conclude with a discussion of the results in Section 6.

2. Matrix product states and operators. In the area of tensor networks, MPS and MPOs form a well-established class of tensor decompositions that allow for efficient and stable approximation of very high-dimensional vectors and matrices, respectively. While they are commonly used in theoretical and numerical quantum many-body physics to model, e.g., ground and thermal states [10, 29, 33, 35, 36, 37, 38, 39, 41], they also have been independently introduced in the numerical mathematics community under the name of Tensor Trains (TT) [27] as a general approximation tool. Since our work is mainly motivated by applications in quantum physics, we will adapt the respective terminology in the following.

A matrix product state is a decomposition of a vector $v \in \mathbb{C}^N$ such that

$$v_i = v_{i_1 \dots i_L} = \mathsf{Tr} C_1^{i_1} C_2^{i_2} \cdots C_L^{i_L},$$

where the index i is split up into L sub-indices of dimensionality d, called physical indices, $i_1, \dots, i_L$. We will refer to $C_1, \dots, C_L \in \mathbb{C}^{d \times D \times D}$ as the sites or core tensors of the MPS and D is called the bond dimension. The superscripts i_j of the C_j correspond to the physical indices. The concept of splitting up the index i is the standard way to represent vectors and matrices in TT/MPS form and is also used for so-called quantized tensor trains (QTT) [21] in the numerical community. While in principle every site may have its own bond dimensions, as long as they allow for contraction with neighbouring sites, for simplicity and without loss of generality, we will assume all sites to have identical bond dimension D. The physical dimension d is likewise assumed to be identical for all sites. It is important to note that $N = L^d$, as this relation forms the basis for the ability of MPS/MPOs to represent vectors and matrices of very high dimensionality.

A slightly different representation can be chosen, where

$$v_i = v_{i_1 \dots i_L} = C_1^{i_1} C_2^{i_2} \cdots C_L^{i_L},$$

with $C_1 \in \mathbb{C}^{d \times 1 \times D}$ and $C_L \in \mathbb{C}^{d \times D \times 1}$. In physical terms, the former representation corresponds to systems with closed boundary conditions (CBC) whereas the latter assumes open boundary conditions (OBC). It is clear that OBC is a special case of CBC. For the remainder of this work, we assume open boundary conditions for the sake of simplicity, but the algorithm can be applied to the CBC case, albeit with a modified computational cost. Following this decomposition, a particular element of v is described by a chain of matrix multiplications, explaining the name of the representation.

Now, the whole vector v can be written as

$$\begin{aligned} v &= \sum_{i_1, \dots, i_L}^{d} \left(C_1^{i_1} C_2^{i_2} \cdots C_L^{i_L}\right) \left(e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_L}\right) \\ &= \sum_{k_2, \dots, k_{L-1}}^{D} \left(\sum_{i_1}^{d} C_{1,k_2}^{i_1} e_{i_1}\right) \otimes \left(\sum_{i_2}^{d} C_{2,k_2 k_3}^{i_2} e_{i_2}\right) \\ &\qquad \otimes \cdots \otimes \left(\sum_{i_L}^{d} C_{L,k_{L-1}}^{i_L} e_{i_L}\right) \end{aligned}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

$$= \sum_{k_2,\ldots,k_{L-1}}^{D} u_{1,k_2} \otimes u_{2,k_2k_3} \otimes \cdots \otimes u_{L,k_{L-1}},$$

where e_j denotes the jth column of the identity matrix and the subscripts k_j and k_{j+1} denote the row and column indices of the matrices $C_j^{i_j}$, respectively. This expression sheds some light on the underlying tensor product structure of MPS and facilitates comparisons with other tensor decomposition schemes.

We now turn our attention to the representation of operators and matrices. Abstractly, one can define an MPO as an operator with an MPS representation in a direct product basis of the operator linear space. More concretely, for the representation of a matrix $A \in \mathbb{C}^{N\times N}$ as an MPO, the approach presented above can easily be adapted to yield

$$A_{ij} = A_{i_1\ldots i_L j_1\ldots j_L} = C_1^{i_1 j_1} C_2^{i_2 j_2} \cdots C_L^{i_L j_L},$$

where i, j have been split up as before, resulting in two superscripts this time, and with the matrices $C_1, \ldots, C_L \in \mathbb{C}^{d\times d\times D\times D}$. In analogy to the case for a vector, we can write the whole matrix as

$$\begin{aligned} A = &\sum_{i_1,\ldots,i_L}^{d} \sum_{j_1,\ldots,j_L}^{d} (C_1^{i_1 j_1} C_2^{i_2 j_2} \cdots C_L^{i_L j_L}) \\ &\qquad \cdot (e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_L})(e_{j_1}^T \otimes e_{j_2}^T \otimes \cdots \otimes e_{j_L}^T) \\ = &\sum_{i_1,\ldots,i_L}^{d} \sum_{j_1,\ldots,j_L}^{d} \sum_{k_2,\ldots,k_{L-1}}^{D} (C_{1,1k_2}^{i_1 j_1} C_{2,k_2k_3}^{i_2 j_2} \cdots C_{L,k_{L-1}1}^{i_L j_L}) \\ &\qquad \cdot (e_{i_1} e_{j_1}^T) \otimes (e_{i_2} e_{j_2}^T) \otimes \cdots \otimes (e_{i_L} e_{j_L}^T) \\ = &\sum_{k_2,\ldots,k_{L-1}}^{D} U_{1,k_2} \otimes U_{2,k_2k_3} \otimes \cdots \otimes U_{L,k_{L-1}}, \end{aligned}$$

where e_j is again the jth column of the identity and e_j^T is its transpose. Note that this also holds true for other product bases, like for instance the Pauli basis. Making use of these formulations, it is easy to show that basic operations such as scalar multiplication, addition, and inner product as well as the multiplication of an MPS by an MPO or of two MPOs can be performed in the formalism. The addition and non-scalar multiplication, however, lead to an increase in the bond dimension D. For the addition of two MPS/MPOs with bond dimensions D and D', the new bond dimension is $D'' \leq D + D'$, and for the multiplication, $D'' \leq D \cdot D'$ [33]. This can again easily be verified.

It is obvious from the above explanation that the bond dimension is the decisive factor for the expressive power of the formalism. An exact representation of a vector (operator) as an MPS (MPO) is always possible if we allow the bond dimension D to be big enough, which may mean exponentially large in L, up to $d^{\lfloor N/2\rfloor}$ [38]. When the maximum value of D is limited to some fixed value (*truncated*) smaller than the one required for exactness, not all vectors or operators can be represented, which may give rise to approximation errors. We will in the following denote that some vector v or matrix A is approximated with bond dimension D by writing $v[D]$ and $A[D]$, respectively. Nevertheless, it has been found that for many physically relevant states and operators, MPS/MPO yield good approximations for $D \in \mathcal{O}(poly(L))$ [35, 36, 41] leading to the total number of parameters $LdD^2 \in \mathcal{O}(poly(L))$ as opposed to d^L or d^{2L} for the whole vector or matrix, respectively. This constitutes another main reason for their usefulness as an efficient approximation scheme.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Algorithm 1: Global Lanczos Algorithm.

Input : Matrix $A \in \mathbb{C}^{N\times N}$, Starting Matrix $U \in \mathbb{C}^{N\times M}$, Number of Dimensions K

$U_0 \leftarrow 0$;
$V_0 \leftarrow U$;
for $i \leftarrow 1; i \leq K$ **do**
  $\beta_i \leftarrow \|V_{i-1}\|_F$;
  if $\beta_i = 0$ **then**
    break ;
  end
  $U_i \leftarrow V_{i-1}/\beta_i$;
  $V_i \leftarrow AU_i - \beta_i U_{i-1}$;
  $\alpha_i \leftarrow \langle U_i, V_i\rangle$;
  $V_i \leftarrow V_i - \alpha_i U_i$;
end

Output : Orthonormal Basis $\mathbf{U}_K \in \mathbb{C}^{N\times KM}$, Tridiagonal Matrix $T_K \in \mathbb{R}^{KM\times KM}$

Naturally, many methods have been developed to find optimal and canonical representations for a given D both in the numerical and the quantum physics community. The most important algorithms for optimizing a given MPS/MPO with respect to some error function and bond dimension thereby rely on local updates of the individual C_i with all other sites being treated as constants, rather than considering all parameters simultaneously. These algorithms, starting with the left- or right-most site, generally sweep back and forth over the chain of sites updating one site per step until convergence. As all sites not considered in a given step are treated as fixed, this sweeping scheme allows for reusage of previously computed values in a dynamical programming fashion. As explaining the details and the complexity of these algorithms exceeds the scope of this section, we refer the interested reader to the overview articles [2, 15, 33, 37].

3. The global Lanczos algorithm and Gauss quadrature. The idea of employing variants of Krylov methods to solve various types of problems, for instance, solving linear systems [19, 22, 31], finding eigenvectors [1, 5, 23, 24] or approximating the action of an exponential operator onto a vector [12], is already well-established. To solve, e.g., linear systems with multiple right-hand sides and for reasons of efficiency, block versions of the originally vector-based Krylov algorithms have been developed. While there exist several block versions of the Lanczos algorithm [6, 8, 13, 16, 25], we will only consider the one presented in [4] as it does not require the columns of the basis blocks to be orthogonal, which would be prohibitive for very large matrices.

Starting from an initial matrix $U \in \mathbb{C}^{N\times M}$, the algorithm will build up a basis of matrices $\mathbf{U}_i = [U_1, \cdots, U_i]$ with $\mathbf{U}_i \in \mathbb{C}^{N\times iM}$. Now, we first need to state the inner product with respect to which the individual U_i must be orthonormal and define it to be

$$\langle U_i, U_j\rangle = \mathsf{Tr} U_i^* U_j,$$

where $U_i, U_j \in \mathbb{C}^{N\times M}$. This induces the well known Frobenius norm

$$\|U_i\|_F = \langle U_i, U_i\rangle^{1/2},$$

and hence this definition of the inner product is a straightforward generalization of the one used in the standard Lanczos algorithm. Naturally, one may also choose different inner products [8].

ETNA
Kent State University and
Johann Radon Institute (RICAM)

For this work, we do, however, choose the Frobenius norm as it can be efficiently computed for MPOs. Equipped with this definition, we can see that Algorithm 1 is in fact a direct generalization of the standard Lanczos algorithm to the matrix-case. As such we find that after i steps, the method has produced the reduction T_i of A given by

$$T_i = \begin{bmatrix} \alpha_1 & \beta_2 & & \mathbf{0} \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_i \\ \mathbf{0} & & \beta_i & \alpha_i \end{bmatrix}$$

and yields the partial global Lanczos decomposition

$$A\mathbf{U}_i = \mathbf{U}_i \tilde{T}_i + \beta_{i+1} U_{i+1} E_i^T,$$

where we define $\tilde{T}_i = T_i \otimes I_M \in \mathbb{R}^{iM \times iM}$ and $E_i^T = [\mathbf{0}, \cdots, \mathbf{0}, I_M] \in \mathbb{R}^{M \times iM}$. Furthermore, it holds that

$$\beta_{i+1} U_{i+1} = (A - \alpha_i I_N) U_i - \beta_i U_{i-1}$$

and

$$\mathbf{U}_i^* A \mathbf{U}_i = T_i,$$

if we apply the inner product defined previously. From now on, we will implicitly make use of this inner product whenever appropriate. Then, all other results obtained for the original Lanczos method carry over to the global Lanczos case.

To establish the link between the global Lanczos method and Gauss quadrature, we start by observing that

$$u^* f(A) u = u^* V_A f(\Lambda_A) V_A^* u = \sum_{i=1}^{N} f(\lambda_i) \mu_i^2 = \int_a^b f(\lambda) \mathrm{d}\mu(\lambda),$$

with $V_A \Lambda_A V_A^*$ being the spectral decomposition of A, $\mu_i = e_i^T V_A^* u$ and

$$\mu(\lambda) = \begin{cases} 0 & \text{if } \lambda < \lambda_1 = a \\ \sum_{i=1}^{j} \mu_i^2 & \text{if } \lambda_j \leq \lambda < \lambda_{j+1} \\ \sum_{i=1}^{N} \mu_i^2 & \text{if } b = \lambda_N \leq \lambda \end{cases}$$

being a piecewise constant and nondecreasing distribution function. Here we assume the eigenvalues of A to be ordered ascendingly. We can use this result to obtain

$$\begin{aligned} \mathcal{I} f &\coloneqq \mathsf{Tr}(U^* f(A) U) = \sum_{i=1}^{N} e_i^* U^* V_A f(\Lambda_A) V_A^* U e_i = \sum_{i=1}^{N} \int_a^b f(\lambda) \mathrm{d}\mu_i(\lambda) \\ &= \int_a^b f(\lambda) \mathrm{d} \sum_{i=1}^{N} \mu_i(\lambda) = \int_a^b f(\lambda) \mathrm{d}\mu(\lambda) \end{aligned}$$

for a matrix U like the initial matrix of the global Lanczos method, where we define $\mu_i(\lambda)$ analogously to the case above and $\mu(\lambda) \coloneqq \sum_{i=1}^{N} \mu_i(\lambda)$. This Riemann-Stieltjes integral can now be tackled via Gauss-type quadrature, the most general formulation of which is given by

$$\mathcal{G} f \coloneqq \sum_{k=1}^{K} \omega_k f(\theta_k) + \sum_{m=1}^{M} \nu_m f(\tau_m),$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

where θ_k and τ_m are called the nodes and ω_k and ν_m the weights of the quadrature. In this work we only consider the case where $M = 0$.

It is well known that in order to determine the ω_k and θ_k that satisfy this property, one can construct a sequence of polynomials $\{p_0, \cdots, p_K\}$ that are orthonormal in the sense that

$$\int_a^b p_i(\lambda)p_j(\lambda)\mathrm{d}\mu(\lambda) = \delta_{ij}$$

and that satisfy a recurrence relation given by

$$\beta_i p_i(\lambda) = (\lambda - \alpha_{i-1})p_{i-1}(\lambda) - \beta_{i-1}p_{i-2}(\lambda), \tag{3.1}$$

where $p_{-1}(\lambda) \equiv 0$ and $p_0(\lambda) \equiv 1$. Then, the roots of p_K can be shown to be the optimal θ_k [7, 14]. Now, the recurrence relation yields a recurrence matrix T_K defined by

$$T_K = \begin{bmatrix} \alpha_1 & \beta_2 & & \mathbf{0} \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_K \\ \mathbf{0} & & \beta_K & \alpha_K \end{bmatrix},$$

whose eigenvalues are the zeros of $p_K(\lambda)$ and consequentially the θ_k of $\mathcal{G}f$ [14]. The ω_k are given by the squared first elements of the normalized eigenvectors of T_K and so,

$$\mathcal{G}f = e_1^T f(T_K)e_1 = e_1^T V_T f(\Lambda_T)V_T^* e_1,$$

where $V_T\Lambda_T V_T^*$ is the spectral decomposition of T_K.

Now, the U_i from the global Lanczos method can be expressed by

$$U_i = p_{i-1}(A)U,$$

with p_{i-1} being some polynomial of degree $i-1$. Then, it is clear that

$$\langle p_{i-1}(A)U, p_{j-1}(A)U\rangle = \langle U_i, U_j\rangle = \delta_{ij}$$

and taking into account the above derivations

$$\begin{aligned}\langle p_{i-1}(A)U, p_{j-1}(A)U\rangle &= \mathsf{Tr}(U^* p_{i-1}(A)^* p_{j-1}(A)U) \\ &= \int_a^b p_{i-1}(\lambda)p_{j-1}(\lambda)\mathrm{d}\mu(\lambda).\end{aligned}$$

Hence, the global Lanczos method produces orthonormal polynomials [24] that in addition satisfy the recurrence relation stated in equation (3.1) by construction as we have seen above. The T_i obtained by the global Lanczos algorithm is thus the recurrence matrix needed to perform a Gauss quadrature with i nodes. If we choose $U \in \mathbb{C}^{N\times N}$ to be unitary, it follows that

$$\mathsf{Tr}f(A) = \mathsf{Tr}(U^* f(A)U) = \int_a^b f(\lambda)\mathrm{d}\mu(\lambda) \approx e_1^T f(T_i)e_1.$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Algorithm 2: Approximation Algorithm.

Input : MPO $A[D_A] \in \mathbb{C}^{N\times N}$, Starting orthogonal MPO $U[D_{init}] \in \mathbb{C}^{N\times N}$, Number of Dimensions K, Maximal Bond-Dimension D_{max}, Stopping Criteria $\mathcal{S}$

$U_0 \leftarrow 0$;
$V_0 \leftarrow U$;
$D \leftarrow D_{init}$;
for $i \leftarrow 1; i \leq K$ **do**
  $\beta_i \leftarrow \sqrt{\texttt{contract}(V_{i-1}, V_{i-1})}$;
  if $\beta_i = 0$ **then**
    break ;
  end
  $U_i \leftarrow \texttt{multiplyScalar}(1/\beta_i, V_{i-1})$;
  $D \leftarrow \min(D_{max}, D \cdot D_A)$;
  $V_i \leftarrow \texttt{multiplyAndOptimize}(A, U_i, D)$;
  $D \leftarrow \min(D_{max}, D + D_{U_{i-1}})$;
  $V_i \leftarrow \texttt{sumAndOptimize}(V_i, -\beta_i U_{i-1}, D)$;
  $\alpha_i \leftarrow \texttt{contract}(U_i, V_i)$;
  $D \leftarrow \min(D_{max}, D + D_{U_i})$;
  $V_i \leftarrow \texttt{sumAndOptimize}(V_i, -\alpha_i U_i, D)$;
  $V_T \Lambda_T V_T^* \leftarrow \texttt{spectralDecomposition}(T_i)$;
  $\mathcal{G}f \leftarrow \beta_1^2 e_1^T V_T f(\Lambda_T) V_T^* e_1$;
  if $\texttt{checkStop}(\mathcal{G}f, \Lambda_T, \mathcal{S})$ **then**
    break ;
  end
end

Output : Approximation $\mathcal{G}f$ of $\mathsf{Tr}f(A)$

4. Assembling the parts. Now that we have reviewed the relevant theoretical aspects, we will proceed by showing how we put together the pieces to obtain our algorithm. The whole algorithm is presented in Algorithm 2.

Since the global Lanczos method is based on matrix-matrix multiplications, additions of matrices, and multiplications of matrices by scalars, these operations have to be formulated for the MPO case. As the bond dimension of the basis-MPOs grows with the number of multiplications and additions, we need to keep track of the bond dimensions and perform projections onto lower bond dimensions whenever necessary. Thus, the input parameters of our method are the MPO $A[D_A] \in \mathcal{C}^{N\times N}$, an orthogonal starting MPO $U[D_{init}] \in \mathcal{C}^{N\times N}$, the maximal Krylov-dimension K, a set of stopping criteria $\mathcal{S}$, and finally the maximal bond dimension D_{max} of the U_i.

It should be stressed that we assume $U[D_{init}]$ to be unitary and of the same dimension as $A[D_A]$. This allows us to replace the approximation that had to be made previously by combining the estimations for several starting matrices by the exact computation since now it holds that $\mathsf{Tr}f(A) = \mathsf{Tr}U^* f(A)U$ if we assume U to be normalized without loss of generality. This is only possible due to the fact that we translate the Lanczos method to the MPO formalism.

Besides the case of very large matrices that can be explicitly stored but are too large for multiplications with equally sized matrices, this is especially important for the case where the respective matrices are only given in MPO format and N is of the order of several millions, as

ETNA
Kent State University and
Johann Radon Institute (RICAM)

in the case of quantum many-body systems. While we introduce some approximation error by using MPOs, we will show in Section 5 that these errors can be comparably small already for low bond dimensions in cases of practical interest. In the following, we will omit the declaration of the bond dimension of an MPO whenever it increases clarity.

While in principle every orthogonal MPO can serve as a starting point, in this work we choose $U[D_{init}]$ to be the identity matrix because it has a minimal MPO formulation of $C_i^{jk} = \delta_{jk}$. This allows us to start from the minimal bond dimension $D_{init} = 1$ and thus maximizes the amount of relevant information that we can store for a given D_{max}. In certain cases it might however be possible to choose a better starting MPO, e.g., when A is very close to being diagonal. Note that starting with the full identity matrix does not imply convergence in one step as the identity is not a basis of the space implied by the Frobenius inner product. For the implementation of the inner product and norm used in the global Lanczos algorithm, we observe that

$$\langle U_i, U_j \rangle = \sum_{k=1}^{N} \sum_{l=1}^{N} U_{i,kl} U_{j,lk} = U_{i,vec}^{*} U_{j,vec} \,,$$

where $U_{i,vec}$ and $U_{j,vec}$ are the vectorized versions of U_i and U_j, respectively. This allows us to make use of an efficient, exact way of computing the inner product of MPS [33] by rewriting the $C_i^{j_i k_i}$ as $C_i^{j_i'}$ with $\dim j_i' = \dim j_i \cdot \dim k_i$ and hence vectorizing the MPO. This functionality is implemented in `contract()`. The implementation of the scalar multiplication `multiplyScalar()` is straightforward as it corresponds to multiplying an arbitrary C_i—we choose C_1 for simplicity—of the respective MPO by the scalar at hand.

A bit more care has to be taken when implementing the functions for the multiplication and summation of MPOs, `multiplyAndOptimize()` and `sumAndOptimize()`, respectively. One possibility would be to first perform the respective operation exactly, i.e., use the bond dimension required for the exact result, and to project the resulting MPO onto the current D via the singular value decomposition (SVD) of its C_i in a second step. It has however been found that performing the projection simultaneously to the multiplication or summation at the level of the individual C_i yields superiour results; see [33, 37, 40]. In case of the multiplication, we implement this strategy by solving the optimization problem

$$\min_{\tilde{C}_i} \| A U_j[D_{old}] - \tilde{U}_j[D_{new}] \|_F^2 \,,$$

where D_{old} is the bond dimension used previous to the multiplication and D_{new} is the bond dimension used for the optimization. $\tilde{U}_j[D_{new}]$ denotes the result of the multiplication of A on U_j and the $\tilde{C}_i$ are its tensors. The implementation hence performs the multiplication of the MPOs at tensor level and directly optimizes the resulting tensors for the chosen bond dimension by employing the sweeping scheme sketched in Section 2. In order to apply this algorithm to the case of MPO-MPO multiplication, we rewrite AU_j as

$$(I \otimes A) U_{j,vec} = \begin{bmatrix} A & 0 & & \mathbf{0} \\ 0 & A & \ddots & \\ & \ddots & \ddots & 0 \\ \mathbf{0} & & 0 & A \end{bmatrix} \begin{bmatrix} U_{j,1} \\ U_{j,2} \\ \vdots \\ U_{j,N} \end{bmatrix} ,$$

with $U_{j,k}$ being the kth column vector of U_j. Due to technical reasons, we do in fact use $U_j \otimes I$ and A_{vec}. For the summation, we apply the same strategy and solve

$$\min_{\tilde{C}_i} \| \left(U_j[D_{old}] + \sum_k \gamma_k U_k[D_{old}^k] \right) - \tilde{U}_j[D_{new}] \|_F^2 \,,$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

where D_{old} is the bond dimension used before the addition, D_{old}^k are some other previously used bond dimensions, D_{new} is the bond dimension to be used for the optimization, and $\gamma_k \in \mathbb{C}$ are some scalars. $\tilde{U}_j[D_{new}]$, similarly as before, represents the outcome of the summation and the $\tilde{C}_i$ are its tensors.

As it can be seen in Algorithm 2, we allow for exact multiplication and summation as long as the resulting bond dimension does not grow beyond D_{max}. This, however, happens quickly since D can grow exponentially with the number of iterations, and so most of the U_i will be represented based on D_{max}. This underlines the importance of D_{max} for the accuracy of the approximation.

After the algorithm has completed one iteration of the global Krylov method, the spectral decomposition of T_i is performed and the current approximation is computed. Based on the approximation and the eigenvalues of T_i, the algorithm then determines if it should be stopped in `checkStop`. Here we have to account for several factors.

Firstly, we know that $\mathcal{G}f$ converges to the correct value in absence of approximation errors. So, the algorithm can terminate when the distance between the previous and current $\mathcal{G}f$ becomes smaller than some ϵ.

Secondly, it is clear that the projection of the generated MPOs down to D_{max} introduces an approximation error. While it is possible to obtain the error of the optimization problems described above, it is not clear how the accumulated error influences $\mathcal{G}f$ precisely. However, a possible way of detecting when the approximation error has become too large is to check for the violation of some theoretical constraints. For instance, in case of a positive A, we know that the Ritz-values of A must be positive as well. If T_i starts to develop significant negative eigenvalues relative to the allowed numerical precision, we thus know that the total approximation error has reached a level that leads to unreasonable results. The same reasoning could be applied for other intervals in which one knows the spectrum of A to be in.

It is well known that, depending on the sign of the derivative of f in (λ_1, λ_N), $\mathcal{G}f$ can yield an upper/lower bound and that it converges to the true value. Based on this it is possible to show that $\mathcal{G}_M < \mathcal{G}_{M+1}$ for the lower-bound case and $\mathcal{G}_M > \mathcal{G}_{M+1}$ for the case of an upper bound [14]. This provides another stopping-criterion.

As the accumulation of truncation errors can lead to unreasonable results even before the violation of the above property, we propose to keep a moving average of the last k approximations and employ the 3σ-rule to detect improbable results. To dismiss unlikely results, the 3σ-rule makes use of Chebyshev's inequality, according to which the probability for a sample from a probability distribution with finite expected value and variance to be farther away from the expected value than three times the variance is roughly 11%. This heuristic is justified by the guaranteed convergence in the absence of numerical errors.

After having explained the algorithm, a few remarks are in order:

(i) In this version of the algorithm, we only consider the Gauss quadrature. This is mainly due to the fact that obtaining good lower or upper bounds on the spectrum of A is in general not possible because of the size of its dimensions. Analogously to [4], our algorithm can nevertheless be adapted to perform Gauss-Radau or Gauss-Lobatto approximations.

(ii) To improve numerical stability and prevent 'ghost' eigenvalues from occurring, it could be beneficial to perform reorthogonalization. Due to the MPO representation, this would, however, be very costly and not necessarily result in a large improvement. Thus, we do not consider this extension. It can, however, be easily added to the algorithm.

(iii) In the presented algorithm, we stick to the canonical way of orthogonalizing the new matrix against the old matrices individually. In the case of exactly stored

ETNA
Kent State University and
Johann Radon Institute (RICAM)

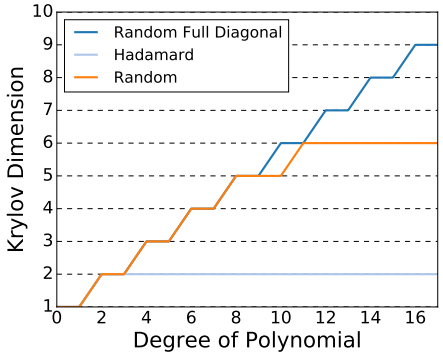


FIG. 4.1. *The number of Krylov dimensions needed to converge to the correct value of* $\mathrm{Tr}\sum_{i=0}^{g}(A)$ *over the degree of the polynomial for the exact version of the algorithm with* $L = \log N = 10$.

matrices/vectors, this scheme increases the numerical stability. Since we now employ approximations of the exact matrices, it might, however, be worth considering to compute α_i first and then optimize the sum containing both U_i and U_{i-1}. The advantage of being able to optimize the whole expression at once might outweigh the disadvantage of orthogonalizing against both matrices simultaneously. On the other hand, computing α_i first might lead to different and possibly worse results.

(iv) As we have stated above, it is possible to obtain approximation errors from both `multiplyAndOptimize` and `sumAndOptimize`. But these errors naturally only refer to the current optimization and do not allow for strict bounds on the overall error. One could of course try to increase the bond dimension for each individual optimization until its error converges to make sure the partial result is close to exact. The problem here is that due to the possibly exponential growth of the bond dimension needed for exactness, D_{max} is typically reached within very few iterations. From this point on, it is not possible to increase D any more and so, the information about the error provides little useful information. This is why we have resorted to the approach of checking for the violation of theoretically guaranteed behaviour.

(v) From the above explanations it is clear that K and D_{max} are the parameters that control the accuracy of the approximation. For the algorithm to be of use for very high-dimensional matrices, we must impose the restriction that $K, D_{max} \in \mathcal{O}(poly(L))$. This property is particularly relevant for quantum mechanical simulations where N grows exponentially with the number of particles.

While it is very difficult to rigorously analyze the convergence behaviour of our method when facing truncation errors introduced by the MPO/MPS representation, we are able to make a statement for the case of exact arithmetic without truncations.

THEOREM 4.1. *For* $f : \mathbb{C}^N \times \mathbb{C}^N \to \mathbb{C}^N \times \mathbb{C}^N$ *being a polynomial of degree* g *or a smooth function that is arbitrarily well approximated by its power series expansion up to* g *and exact arithmetic without truncations, Algorithm 2 converges to the exact value in* $\min\{g^*, \lfloor g/2 \rfloor + 1\}$ *steps, where* g^* *is the degree of the minimal polynomial of* A.

Proof. 1.) It follows from the fact that we perform a Gauss quadrature, which for i nodes is exact for all polynomials up to degree $2i - 1$, and employ the full identity matrix as starting matrix for our algorithm that our method requires maximally $\lfloor g/2 \rfloor + 1$ steps to converge to the exact value. 2.) Furthermore, it follows from the underlying Lanczos algorithm that our method will converge in maximally g^* steps, where g^* is the degree of the minimal polynomial of A. From 1.) and 2.), it directly follows that our method will converge in $\min\{g^*, \lfloor g/2 \rfloor + 1\}$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 4.1
A listing of the complexity of the subfunctions of Algorithm 2. $L = \log N$ is the number of tensors of the MPOs, d is the physical dimension. For simplicity, all U_i are assumed to have bond dimension D_{max} and T_i is assumed to be in $\mathbb{R}^{K\times K}$.

Function	Complexity
`contract`	$\mathcal{O}(LD_{max}^3 d^2)$
`multiplyAndOptimize`	$\mathcal{O}(LD_{max}^3 D_A d^2)$
`sumAndOptimize`	$\mathcal{O}(LD_{max}^3 d^2)$
`multiplyScalar`	$\mathcal{O}(D_{max}^2 d^2)$
`spectralDecomposition`	$\mathcal{O}(K^3)$
`checkStop`	$\mathcal{O}(1)$

steps in the exact case. □

It is worth noting that this result would also hold for the global Lanczos method as introduced in [4] if the authors would not explicitly restrict themselves to starting matrices of significantly smaller dimension than that of A, as that restriction prevents the Gauss quadrature from converging to the exact value. While truncation errors introduced by the tensor network representation will deteriorate the convergence behaviour and the statement above will therefore not be directly applicable to practical applications in general, it is still instructive to understand the behaviour of the algorithm in the ideal case.

To illustrate the result, Figure 4.1 depicts the number of steps needed by the algorithm without truncations to converge to the correct result with a relative error of 10^{-12} for increasing degrees of the polynomials of the form $f(A) = \sum_{i=0}^{g} A^i$ and $N = 1024$. For a full diagonal matrix with entries randomly sampled from the uniform distribution over $[-1, 1]$, the algorithm in fact needs as many steps as required by the Gauss quadrature to reach exactness. On the other hand, for the Hadamard matrix, which only has two distinct eigenvalues, the method always converges in two steps. A full symmetric matrix with entries uniformly sampled from $[0, 1]$ typically only has a few dominant eigenvalues, which corresponds to the number of steps needed to converge being six in this case.

We will conclude this section with an analysis of the complexity of our algorithm. The complexities of the subfunctions of Algorithm 2 are listed in Table 4.1. For the analysis of `multiplyAndOptimize`, we have assumed D_A to be smaller or of the same order as D_{max}. If it were significantly larger, the complexity would change to $\mathcal{O}(LD_{max}^2 D_A^2 d^2)$. Note that this analysis does not extend to the number of sweeps necessary for the optimizations to converge. For the spectral decomposition, we have for simplicity assumed all T_i to be of size $K \times K$. Combining all the different results, we thus find that the overall complexity of Algorithm 2 is $\mathcal{O}(KLD_{max}^3 D_A d^4)$ with $L = \log N$ and since we require $K, D_{max} \in \mathcal{O}(\log N)$, this is equivalent to $\mathcal{O}(poly(L))$.

5. Numerical results. In this section we present numerical results obtained for a challenging problem of relevance in quantum many-body physics. Our goal thereby is to study the convergence of the results with increasing K and D_{max}. The problem we consider is the approximation of the von Neumann entropy. For a quantum state ρ[1], the von Neumann entropy is given by $S = -\mathsf{Tr}\rho \log \rho$. In the following, we will focus on the case of states of the form $\rho = e^{-\beta H} / \mathsf{Tr} e^{-\beta H}$, i.e., thermal equilibrium states for a Hamiltonian H at a certain

[1] ρ is a positive operator with unit trace, representing an ensemble of pure states.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

inverse temperature β[2]. Here, we assume H to be the Ising Hamiltonian with open boundary conditions that is given by

$$H = J \sum_{i=1}^{L-1} \sigma_i^x \sigma_{i+1}^x + g \sum_{i=1}^{L} \sigma_i^z + h \sum_{i=1}^{L} \sigma_i^x,$$

where $\sigma^{\{x,y,z\}}$ are the Pauli matrices

$$\sigma^x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \sigma^y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \sigma^z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Note, however, that here by $\sigma_i^{\{x,y,z\}}$ we actually denote the tensor product $I_1 \cdots \otimes I_{i-1} \otimes \sigma_i^{\{x,y,z\}} \otimes I_{i+1} \otimes \cdots \otimes I_L$ and analogously for $\sigma_i^{\{x,y,z\}} \sigma_{i+1}^{\{x,y,z\}}$ for simplicity of notation. The Hamiltonian describes a chain of spin particles with nearest neighbour interactions and two magnetic fields acting only on the individual particles. This choice of H has the advantage that it is exactly solvable for $h = 0$, a case commonly known as 'transverse Ising', and thus opens the possibility to obtain a reference solution for system sizes for which it could otherwise not be obtained [20, 32, 34]. Hence, in the following we will assume $h = 0$.

It is possible to find an MPO approximation to the thermal equilibrium state ρ by means of standard MPS-techniques [12, 36, 41]. It is customary to use a *purification* ansatz for this purpose, where $\rho(\beta/2)$ is approximated by standard imaginary time evolution, and the whole state is then written as $\rho \propto \rho(\beta/2)^* \rho(\beta/2)$. In the context of our algorithm, nevertheless, applying exactly this ρ involves a larger cost and worse numerical condition. Instead, we apply the method as described above to $\rho(\beta/2)$ and absorb the necessary squaring into the function that is to be approximated. In our case this means that instead of computing $f(\lambda_i) = \lambda_i \log \lambda_i$ for each Ritz-value, we can compute $f'(\lambda_i) = \lambda_i^2 \log \lambda_i^2$ for λ_i corresponding to $\rho(\beta/2)$. This allows us to apply the algorithm to a possibly much more benign input at the cost of an only slightly more complicated function. Due to truncation errors, the operator $\rho(\beta/2)$ may not be exactly Hermitian. This can be easily accounted for by taking its Hermitian part, $\frac{1}{2}[\rho(\beta/2)^* + \rho(\beta/2)]$, which is an MPO with at most twice the original bond dimension. In our experiments we, however, did not find this to be necessary.

Apart from the entropy, another interesting function to examine would have been the trace norm given by $||\rho||_1 = \mathsf{Tr}\sqrt{\rho^* \rho}$, i.e., the sum of the singular values. But as we only consider positive matrices in this scenario, this sum is equal to the trace which we know to be equal to 1 due to the normalization of the thermal state. Directly related to this, we find that α_1 as computed by our algorithm is given by

$$\alpha_1 = \mathsf{Tr} U_1^* V_1 = \mathsf{Tr} U_1^* \rho U_1 = \frac{1}{\beta_1^2} \mathsf{Tr} U^* \rho U = \frac{1}{\beta_1^2} \mathsf{Tr} \rho.$$

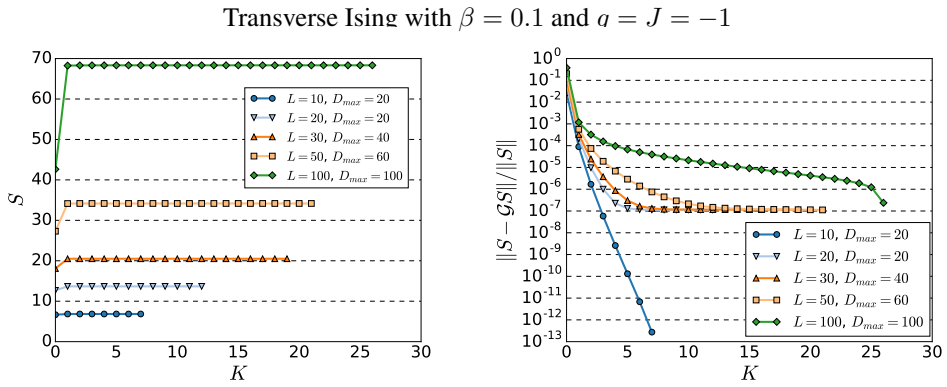
So, our algorithm computes the trace of the input MPO A in one step. We verified this result numerically and found it to hold for all considered cases. This means that the algorithm also computes the trace norm of $\rho(\beta)$ in one step in this case.

It is well known that if its sign is constant over the considered interval, then the $2K$th derivative of f determines whether $\mathcal{G}f$ poses a lower or upper bound of the true value. In our case and for $K > 1$, it is given by

$$\frac{d^{2K} - \lambda_i^2 \ln \lambda_i^2}{d^{2K} \lambda_i} = \frac{4(2K-3)!}{\lambda_i^{2K-2}},$$

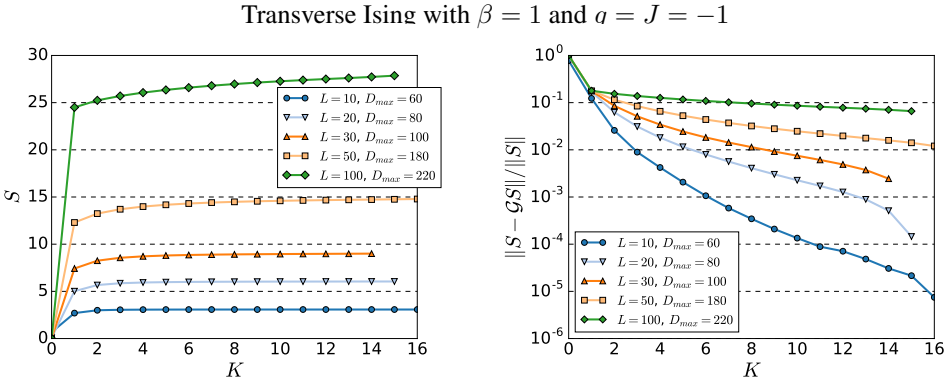
[2]Note that β is not related to the β_i computed by our algorithm.

ETNA
Kent State University and
Johann Radon Institute (RICAM)



(a) Entropy over K. (b) Relative error in the entropy over K.

FIG. 5.1. *Convergence behavior of the algorithm for $L \in \{10, 20, 30, 50, 100\}$, $\beta = 0.1$ and varying Krylov-dimension K. In (a), the convergence of the approximation is depicted. In (b), the convergence of the relative error is shown.*



(a) Entropy over K. (b) Relative error in the entropy over K.

FIG. 5.2. *Convergence behavior of the algorithm for $L \in \{10, 20, 30, 50, 100\}$, $\beta = 1$, and varying Krylov-dimension K. In (a), the convergence of the approximation is depicted. In (b), the convergence of the relative error is shown.*

with λ_i being the ith eigenvalue of ρ. Hence, we can expect our algorithm to provide increasingly tight lower bounds for the correct value. We use the violation of this property as a stopping criterion to account for the situation when truncation errors become too large. Additionally, we keep the average of the last three or four—depending on β—approximations and employ the aforementioned 3σ-rule. In case these stopping criteria are not met, we terminate the algorithm when the absolute difference between successive approximations is below 10^{-10}.

In our experiments we considered systems of size $L \in \{10, 20, 30, 50, 100\}$, Hamiltonian parameters $J = g = 1$, and inverse temperatures $\beta \in \{0.1, 1.0\}$. The bond dimension used to obtain $\rho(\frac{\beta}{2})$ was set to 20 for all cases. The convergence of the approximation as well as the relative error for $\beta = 0.1$ and $\beta = 1$ are shown in Figure 5.1 and Figure 5.2, respectively. Note that in order to compute the relative error, we used numerical diagonalization for $L = 10$ and the analytical solution [32] for $L > 10$. In all cases except for $L = 10$ and $\beta = 0.1$, where the algorithm was stopped when the distance between successive approximations reached the threshold, the algorithm was stopped when meeting the 3σ stopping criterion. Table 5.1

ETNA
Kent State University and
Johann Radon Institute (RICAM)

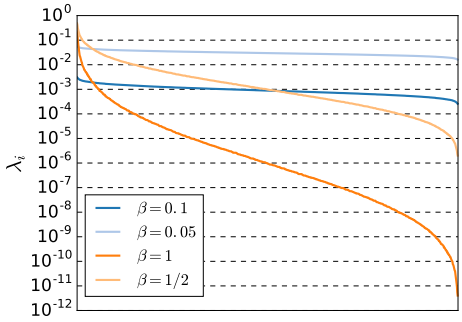


FIG. 5.3. *The spectra of $\rho(\beta)$ and $\rho(\frac{\beta}{2})$ of the transverse Ising Hamiltonian with $J = g = -1$ for $L = 10$ and $\beta \in \{0.1, 1\}$.*

shows the change of the relative error of the final approximations with growing D_{max} for $L \in \{50, 100\}$ and $\beta \in \{0.1, 1\}$.

For the case of $\beta = 0.1$ we observe fast convergence to good approximations in K and D_{max} as shown in Figure 5.1. The maximal bond dimension required for good convergence only grows mildly with L allowing our method to scale very well with the size of the input. The plots in Figure 5.1b show a plateau in the relative error at 10^{-7}. This corresponds to the non-vanishing difference between the exact solution and the numerical MPO used as input. Note that for $L = 10$, where the input is exact, the method is able to achieve a smaller error.

The results for the larger inverse temperature $\beta = 1$ paint a slightly different picture. While the overall behavior of our method remains the same and Figure 5.2a depicts good convergence especially for $L < 100$, Figure 5.2b shows that the relative error achieved is noticeably worse than for the case of $\beta = 0.1$. It also seems that larger values of D_{max} are required to achieve reasonable results. This phenomenon naturally becomes more pronounced with larger L.

We conjecture that the difference in the performance observed for the two considered values of β has two main reasons. Firstly, the bond dimension required for a good approximation of ρ grows with larger β. This might in turn increase the value of D_{max} required for good accuracy, and, correspondingly, increase the approximation error incurred by ρ, so that the computed function will be farther from the analytical solution. Secondly, the spectral properties of the obtained MPOs for the two considered cases are significantly different. In Figure 5.3, we show the spectra of ρ for both values of β and $\beta/2$ for $L = 10$, respectively. It is clearly visible that $\beta = 0.1$ poses a much more benign case. This is underlined by the condition numbers, which are roughly 11.9, $5.68 \cdot 10^{10}$, 3.5 and $2.38 \cdot 10^{5}$ for $\beta = 0.1$, $\beta = 1$, $\beta = 0.05$ and $\beta = 0.5$, respectively. They show that $\beta = 1$ in fact yields highly ill-conditioned MPOs, functions of which are hard to approximate. These considerations also make it clear that by absorbing the necessary squaring of the eigenvalues into the function, we obtain much more well-conditioned input MPOs of lower bond dimension. Hence, we can conclude that our method, while being influenced by both aforementioned factors, is relatively robust and even works reasonably well for very difficult cases.

Table 5.1 illustrates that while our method achieves a low error for $\beta = 0.1$ and a moderate error for $\beta = 1$ even for a small D_{max}, it profits from an increase of the maximal bond dimension. This effect is more pronounced for the lower β which is likely due to the reasons mentioned above. For instance, for $L = 100$ and $\beta = 0.1$, the error decreases by two orders of magnitude from 10^{-5} to about 10^{-7} when D_{max} is raised from 20 to 180,

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 5.1
Relative error in the entropy of the transverse Ising Hamiltonian with $J = -1$ for $L \in \{100, 50\}$, $g \in \{1, 0.1\}$ and increasing values of the maximal bond dimension D_{max}.

D_{max}	$L = 100, \beta = 0.1$	$L = 100, \beta = 1$	$L = 50, \beta = 0.1$	$L = 50, \beta = 1$
20	$1.14 \cdot 10^{-05}$	$9.75 \cdot 10^{-02}$	$2.73 \cdot 10^{-07}$	-
40	$3.96 \cdot 10^{-06}$	$9.14 \cdot 10^{-02}$	$8.04 \cdot 10^{-08}$	$2.75 \cdot 10^{-02}$
60	$1.67 \cdot 10^{-06}$	$9.32 \cdot 10^{-02}$	$1.12 \cdot 10^{-07}$	$1.84 \cdot 10^{-02}$
80	$1.41 \cdot 10^{-06}$	$7.95 \cdot 10^{-02}$	$1.15 \cdot 10^{-07}$	$1.89 \cdot 10^{-02}$
100	$2.38 \cdot 10^{-07}$	$7.81 \cdot 10^{-02}$	-	$1.26 \cdot 10^{-02}$
120	$4.01 \cdot 10^{-07}$	$7.37 \cdot 10^{-02}$	-	$1.11 \cdot 10^{-02}$
140	$2.77 \cdot 10^{-07}$	$7.04 \cdot 10^{-02}$	-	$1.05 \cdot 10^{-02}$
160	$2.29 \cdot 10^{-07}$	$7.11 \cdot 10^{-02}$	-	$9.56 \cdot 10^{-03}$
180	$6.43 \cdot 10^{-08}$	$7.03 \cdot 10^{-02}$	-	$9.06 \cdot 10^{-03}$

TABLE 5.2
Comparison of the runtime in seconds between SciPy's `expm` function and Algorithm 2 for $\mathsf{Tr}\, exp(A)$ and increasing values of L. Lower runtimes are printed bold.

L	`scipy.expm`	`Algorithm 2`
10	**1**	23
11	**3**	23
12	**20**	31
13	129	**43**
14	994	**50**

which still constitutes a strong truncation. It is conceivable that a further increase of the maximal bond dimension would improve the accuracy. For $L = 50$ and $\beta = 0.1$, the error still decreases when D_{max} is raised from 20 to 40, but a further increase shows now effect due to the aforementioned small error already introduced by $\rho(\beta)$. We also found that D_{max} limits the number of basis MPOs that can be successfully orthogonalized and therefore effectively controls the maximally reachable K. Hence, D_{max} can be regarded as the decisive parameter of our method.

We do not provide a proper comparison to other methods at this point because of the simple reason that to the best of the authors knowledge there is no other algorithm that can solve the considered kind of problem for $L \gg 20$, but for $L \leq 20$ we expect the existing highly optimized methods to outperform our method in terms of runtime. However, from our analysis of the complexity we know that our method scales as $\mathcal{O}(\log N)$ with the matrix dimension N when all other parameters are kept constant. This is in contrast to methods based on full diagonalization, which scales as $\mathcal{O}(N^3)$. One can therefore expect our method to eventually outperform such approaches as well as approaches with super-logarithmic scaling in general for growing N when all other parameters are kept fixed. In Table 5.2, we provide a small comparison of the runtime of our algorithm to that of the `expm` function from the SciPy package for Python which implements a squaring-and-scaling approach. We approximated $\mathsf{Tr}\exp(A)$ where A is the transverse Ising Hamiltonian with $L \in \{10, 11, 12, 13, 14\}$. We set $D_{max} = 30$ which yielded good approximations and $K = 100$ to prevent the method from reaching the maximal Krylov dimension before convergence. We then let the algorithm run until the relative error between successive approximations was smaller than 10^{-8}. The results were obtained on an Intel i7-4790 CPU with 32GB RAM. While for smaller matrices up to $L = 12$ our method is significantly slower, it does not suffer from the same drastic increase in runtime with growing matrix size and hence outperforms `expm` for $L > 12$. We

ETNA
Kent State University and
Johann Radon Institute (RICAM)

found the `expm` function to break down due to memory requirements for $L > 14$. The results show that for the given parameters, the runtime of our method remains small for all considered matrix sizes. However, depending on D_{max} the runtime can significantly increase to several hours. Note though that there exist several ways to speed up computations in the MPO/MPS representation via exploitation of symmetries which we did not consider in our implementation.

6. Discussion. In this work, we have introduced a method to approximate functionals of the form $\mathsf{Tr} f(A)$ for matrices of dimension much larger than 2^{20}. We started by giving an overview of the mathematical and algorithmic ideas behind the method. Following this, a detailed description of the algorithm together with an analysis of its complexity was provided. We then presented numerical results for a challenging problem in quantum many-body physics. These results indicate that our method is able to produce good approximations for a number of Krylov steps and a maximal bond dimension logarithmic in the size of the matrix as long as the matrix exhibits some structure that can be expressed well in the MPO/MPS-formalism and is moderately well-conditioned. It was also shown that the maximal allowed bond dimension is the decisive parameter of the algorithm.

There are several ways to build upon this work. Firstly, an investigation of preconditioning methods suitable for our method could be fruitful. Secondly, a more thorough analysis of the effect of the approximation error introduced by the tensor network formalism on the approximation error of the Gauss quadrature would be an interesting addition. Thirdly, the connection of the approximability of a matrix by an MPO to the convergence behavior of our method could provide deeper understanding. Fourthly, it could be investigated which of the many improvements over the normal Gauss quadrature, as for instance [30], can be incorporated into our algorithm to make better use of the expensive information obtained in the Krylov iteration. Finally, the method naturally could be applied to solve practical problems of interest.

While our method was tested for a quantum mechanical problem, it is of course general in nature and can be applied to any case where the matrix in question can be formulated as an MPO or well approximated by one. Especially for matrices of dimension larger than 2^{10} that however can still be explicitly stored, it might be interesting to consider computing the desired function for the MPO representation.

Acknowledgements. This work was partly funded by the *Elite Network of Bavaria* (ENB) via the doctoral programme *Exploring Quantum Matter*.

REFERENCES

[1] W. E. ARNOLDI, *The principle of minimized iteration in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.
[2] M. BACHMAYR, R. SCHNEIDER, AND A. USCHMAJEW, *Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations*, Found. Comput. Math., 16 (2016), pp. 1423–1472.
[3] J. BAGLAMA, C. FENU, L. REICHEL, AND G. RODRIGUEZ, *Analysis of directed networks via partial singular value decomposition and Gauss quadrature*, Linear Algebra Appl., 456 (2014), pp. 93–121.
[4] M. BELLALIJ, L. REICHEL, G. RODRIGUEZ, AND H. SADOK, *Bounding matrix functionals via partial global block Lanczos decomposition*, Appl. Numer. Math., 94 (2015), pp. 127–139.
[5] D. CALVETTI, L. REICHEL, AND D. C. SORENSEN, *An implicitly restarted Lanczos method for large symmetric eigenvalue problems*, Electron. Trans. Numer. Anal., 2 (1994), pp. 1–21. http://etna.ricam.oeaw.ac.at/vol.2.1994/pp1-21.dir/pp1-21.pdf
[6] J. CULLUM AND W. DONATH, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace of large, sparse, real symmetric matrices*, in 1974 IEEE Conference on Decision and Control, IEEE, New York 1974, pp. 505–509.
[7] P. J. DAVIS AND P. RABINOWITZ, *Methods of Numerical Integration*, 2nd ed., Dover, Mineola, 2007.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[8] L. ELBOUYAHYAOUI, A. MESSAOUDI, AND H. SADOK, *Algebraic properties of the block gmres and block Arnoldi methods*, Electron. Trans. Numer. Anal., 33 (2009), pp. 207–220. http://etna.ricam.oeaw.ac.at/vol.33.2008-2009/pp207-220.dir/pp207-220.pdf

[9] E. ESTRADA AND D. J. HIGHAM, *Network properties revealed through matrix functions*, SIAM Rev., 52 (2010), pp. 696–714.

[10] M. FANNES, B. NACHTERGAELE, AND R. F. WERNER, *Finitely correlated states on quantum spin chains*, Comm. Math. Phys., 144 (1992), pp. 443–490.

[11] C. FENU, D. MARTIN, L. REICHEL, AND G. RODRIGUEZ, *Block Gauss and anti-Gauss quadrature with application to networks*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1655–1684.

[12] J. J. GARCÍA-RIPOLL, *Time evolution of matrix product states*, New J. Phys., 8 (2006), Art. no. 305, 22 pages.

[13] G. H. GOLUB, F. T. LUK, AND M. L. OVERTON, *A block Lanczos method for computing the singular values of corresponding singular vectors of a matrix*, ACM Trans. Math. Software, 7 (1981), pp. 149–169.

[14] G. H. GOLUB AND G. MEURANT, *Matrices, Moments and Quadrature with Applications*, Princeton University Press, Princeton, 2010.

[15] L. GRASEDYCK, D. KRESSNER, AND C. TOBLER, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitt., 36 (2013), pp. 53–78.

[16] R. G. GRIMES, J. G. LEWIS, AND H. D. SIMON, *A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 228–272.

[17] M. B. HASTINGS, *An area law for one-dimensional quantum systems*, J. Stat. Mech. Theory Exp., (2007), pp. P08024, 14.

[18] T. HUCKLE, K. WALDHERR, AND T. SCHULTE-HERBRÜGGEN, *Computations in quantum tensor networks*, Linear Algebra Appl., 438 (2013), pp. 750–781.

[19] I. C. F. IPSEN AND C. D. MEYER, *The idea behind Krylov methods*, Amer. Math. Monthly, 105 (1998), pp. 889–899.

[20] D. KAREVSKI, *Ising quantum chains*, Preprint on arXiv, 2006. https://arxiv.org/abs/cond-mat/0611327

[21] B. KHOROMSKIJ, *O(d log n)-quantics approximation of $N - d$ tensors in high-dimensional numerical modeling*, Constr. Approx., 34 (2011), pp. 257–280.

[22] D. KRESSNER AND C. TOBLER, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1288–1316.

[23] A. KRYLOV, *On the numerical solution of the equation by which the frequency of small oscillations is determined in technical problems*, Izv. Akad. Nauk SSSR Ser. Fiz.-Mat, 4 (1931), pp. 491–539.

[24] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Research Nat. Bur. Standards, 45 (1950), pp. 255–282.

[25] P. L. MONTGOMERY, *A block Lanczos algorithm for finding dependencies over* $\mathrm{GF}(2)$, in Advances in Cryptology — EUROCRYPT '95, L. C. Guillou and J.-J. Quisquater, eds., vol. 921 of Lecture Notes in Comput. Sci., Springer, Berlin, 1995, pp. 106–120.

[26] M. E. J. NEWMAN, *Networks: An Introduction*, Oxford University Press, Oxford, 2010.

[27] I. OSELEDETS, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[28] D. PEREZ-GARCIA, F. VERSTRAETE, M. M. WOLF, AND J. I. CIRAC, *Matrix product state representations*, Quantum Inf. Comput., 7 (2007), pp. 401–430.

[29] B. PIRVU, V. MURG, J. I. CIRAC, AND F. VERSTRAETE, *Matrix product operator representations*, New J. Phys., 12 (2010), pp. 025012, 13.

[30] L. REICHEL, M. M. SPALEVIĆ, AND T. TANG, *Generalized averaged Gauss quadrature rules for the approximation of matrix functionals*, BIT, 56 (2016), pp. 1045–1067.

[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] S. SACHDEV, *Quantum Phase Transitions*, Wiley Online Library, New York, 2007.

[33] U. SCHOLLWÖCK, *The density-matrix renormalization group in the age of matrix product states*, Ann. Physics, 326 (2011), pp. 96–192.

[34] S. SUZUKI, J.-I. INOUE, AND B. K. CHAKRABARTI, *Quantum Ising Phases and Transitions in Transverse Ising Models*, 2nd ed., vol. 862 of Lecture Notes in Physics, Springer, Heidelberg, 2013.

[35] F. VERSTRAETE AND I. CIRAC, *Matrix product states represent ground states faithfully*, Phys. Rev. B, 73 (2006), Art. no., 094423, 8 pages.

[36] F. VERSTRAETE, J. GARCÍA-RIPOLL, AND I. CIRAC, *Matrix product density operators: simulation of finite-temperature and dissipative systems*, Phys. Rev. Lett., 93 (2004), Art. no. 207204, 4 pages.

[37] F. VERSTRAETE, V. MURG, AND I. CIRAC, *Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems*, Adv. in Phys, 57 (2008), pp. 143–224.

[38] F. VERSTRAETE, D. PORRAS, AND I. CIRAC, *Density matrix renormalization group and periodic boundary conditions: a quantum information perspective*, Phys. Rev. Lett., 93 (2004), Art. no. 227205, 4 pages.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[39] G. VIDAL, *Efficient classical simulation of slightly entangled quantum computations*, Phys. Rev. Lett.,, 91 (2003), Art. no., 147902, 4 pages.
[40] K. WALDHERR, *Numerical Linear and Multilinear Algebra in Quantum Control and Quantum Tensor Networks*, Verlag Dr. Hut, Munich, 2014.
[41] M. ZWOLAK AND G. VIDAL, *Mixed-state dynamics in one-dimensional quantum lattice systems: a time-dependent superoperator renormalization algorithm*, Phys. Rev. Lett., 93 (2004), Art. no. 207205, 4 pages.