Improved scaling of Time-Evolving Block-Decimation algorithm through Reduced-Rank Randomized Singular Value Decomposition

D. Tamascelli[1,2], R. Rosenbach[2], and M. B. Plenio[2]
[1] *Dipartimento di Fisica, Università degli Studi di Milano, Via Celoria 16, 20133 Milano-Italy*
[2] *Institut für Theoretische Physik & IQST, Albert-Einstein-Allee 11, Universität Ulm, Germany*

When the amount of entanglement in a quantum system is limited, the relevant dynamics of the system is restricted to a very small part of the state space. When restricted to this subspace the description of the system becomes efficient in the system size. A class of algorithms, exemplified by the Time-Evolving Block-Decimation (TEBD) algorithm, make use of this observation by selecting the relevant subspace through a decimation technique relying on the Singular Value Decomposition (SVD). In these algorithms, the complexity of each time-evolution step is dominated by the SVD. Here we show that, by applying a randomized version of the SVD routine (RRSVD), the power law governing the computational complexity of TEBD is lowered by one degree, resulting in a considerable speed-up. We exemplify the potential gains in efficiency at the hand of some real world examples to which TEBD can be successfully applied to and demonstrate that for those system RRSVD delivers results as accurate as state-of-the-art deterministic SVD routines.

I. INTRODUCTION

The description of the physics of quantum many-body systems suffers from the "curse of dimensionality", that is, the size of the parameter set that is required to achieve an exact description of the physical state of a quantum many-body system grows exponentially in the number of its subsystems. Therefore, the simulation of quantum-many-body systems by classical means appears to require an exponential amount of computational resources which, in turn, would impose severe limitations on the size of the quantum many-body systems that are amenable to classical simulation.

On the other hand, exactness of description may be traded for an approximate representations of the state and dynamics of a quantum many-body system for as long as the quality of the approximation can be controlled and increased at will and whenever such an approximate treatment results in a polynomial scaling of the resources with the system size. Clearly, this will not be possible for all system dynamics as it would imply the classical simulability of quantum computers which is generally not believed to be the case.

One specific setting of considerable practical importance that allows for efficient approximate description of quantum many-body systems concerns systems whose entanglement content is limited. Indeed, in pure states that factor, at all times, into a product of states; each involving a number of qubits that is bounded from above for all times by a constant [1] can be simulated efficiently on a classical device. Going beyond this, it is well-known that the states of 1-D quantum systems often obey an area law [2–6] which represents a severe limitation of their entanglement content. This can be made use of, as the state of such slightly entangled 1-D quantum many-body system in a pure state can be described efficiently by means of the Density Matrix Renormalization Group (DMRG) [7]. It was noticed early on [8] that DMRG amounts to the approximation of the state of the system by a matrix product state [9] which can be constructed systematically in terms of a consecutive Schmidt decomposition.

This approach can be extended to the dynamics of one-dimensional quantum many-body systems. The time-dependent DMRG (t-DMRG) [10], the time-dependent Matrix Product States (t-MPS) [11], and the Time-Evolving Block-Decimation (TEBD) [12, 13] are all algorithms based on the idea of evolving an MPS [9] in time. In settings in which only a restricted amount of entanglement is present in the system, these methods are very efficient and the cost of simulation scales merely polynomially in the system size. A very clear presentation of these three algorithms as well as a discussion of the main differences between them can be found in [14].

All these algorithms rely in an essential way on the singular value decomposition (SVD) which, due to its computational complexity, represents a bottleneck in their implementations. In this work we show how to address this issue within the TEBD framework by replacing the SVD which is used to restrict the Hilbert space of quantum states relevant to the system dynamics, with the Reduced-Rank Randomized SVD (RRSVD) proposed in [15]. We show that when the Schmidt coefficients decay exponentially, the polynomial complexity of the TEBD decimation step can be reduced from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ with n indicating the linear dimension of the square matrix to decompose. This results in a considerable speed-up for the TEBD algorithm for real world problems, enabling the access of larger parameter spaces, faster simulation times, and thus opening up new regimes that were previously inaccessible.

The paper is organized as follows: Section II provides a brief description of the TEBD algorithm for pure and mixed state dynamics in order to make the manuscript self-contained for the non-specialist and to identify and highlight the crucial step in which the RRSVD routine can be applied for considerable benefit. Section III introduces a specific and challenging application of TEBD, the TEDOPA scheme, which maps an open quantum system on a one-dimensional configuration, allowing for the efficient simulation of its dynamics. Section IV the proceeds with a description of the salient features of the RRSVD algorithm and a discussion of the speed-up that it provides over the standard SVD. Benchmarking in the TEBD context with applications to TEDOPA along with stability analysis are presented in section V. The last section is devoted to conclusions and outlook.

II. AN ALGORITHM FOR ONE-DIMENSIONAL QUANTUM SYSTEMS

The time evolving block decimation (TEBD) is an algorithm that generates efficiently an approximation to the time evolution of a one-dimensional system subject to a nearest-neighbor Hamiltonian. Under the condition that the amount of entanglement in the system is bounded a high fidelity approximation requires polynomially scaling computational resources. TEBD does so by dynamically restricting the exponentially large Hilbert space to its most relevant subspace whose size is scaling polynomially in the system size, thus rendering the computation feasible [13, 14].

TEBD is essentially a combination of an MPS description for a one-dimensional quantum system and an algorithm that applies two-site gates that are necessary to implement a Suzuki-Trotter time evolution. Together with MPS operations such as the application of measurements this yields a powerful simulation framework [9].

While originally formulated for pure states, an extension to mixed states is possible by introducing a matrix product operator (MPO) to describe the density matrix, in complete analogy to an MPS describing a state [16]. The simulation procedure remains unchanged, except for details such as a squaring of the local dimension on each site, the manner in which two-site gates are built, the procedures to achieve normalisation as well as the implementation of measurements [14, 16]. While standard MPO formulations cannot ensure positivity of the state efficiently, recent reformulations can account for this feature too [17]. Important for the present work are implications of these modifications on the two-site update - and while the numerical recipe does not change, its scaling *does*.

A. Introduction to MPS

The remarkable usefulness and broad range of applications of the MPS description for quantum states has been widely recognized early on in the development of DMRG algorithms [8]. To better understand the full extent of the presented work, we highlight the relevant key features of MPS, referring to references [9, 14] for a full account.

Let us introduce an MPS for a pure state of N sites. For simplicity we assume that each site has the same number of dimensions d, the extension to varying dimension is straight forward. The MPS then relates the expansion coefficients $c_{i_1 i_2 \ldots i_N}$ in the Fock basis to a set of $N \cdot d$ matrices Γ and $N-1$ matrices λ

$$| \psi \rangle = \sum_{i_1, i_2, \ldots i_N} c_{i_1 i_2 \ldots i_N} | i_1 i_2 \cdots i_N \rangle \tag{1}$$

$$= \sum_{i_1, i_2, \ldots i_N} \Gamma^{[1] i_1} \cdot \lambda^{[1]} \cdot \Gamma^{[2] i_2} \cdot \ldots \cdot$$
$$\ldots \lambda^{[N-1]} \Gamma^{[N] i_N} | i_1 i_2 \cdots i_N \rangle. \tag{2}$$

Each of the N sites is assigned a set of d matrices Γ which have dimension $\chi_l \times \chi_r$. The index k in square brackets denotes the corresponding site and the i_k the corresponding physical state. The diagonal $\chi_b \times \chi_b$ matrices λ are assigned to the bond k between sites k and $k+1$. The structure of the MPS is such that the matrices λ contain the Schmidt values for a bipartition at this bond. The matrices Γ and λ are related to the coefficients c by

$$c_{i_1 i_2 \ldots i_N} = \Gamma^{[1] i_1} \cdot \lambda^{[1]} \cdot \Gamma^{[2] i_2} \cdot \ldots \cdot \lambda^{[N-1]} \cdot \Gamma^{[N] i_N} \tag{3}$$

with matrix dimensions χ_l, χ_r, and χ_b for all Γ and λ such that building the product yields a number. The main reason for employing an MPS description is the reduction from d^N coefficients c to only $\mathcal{O}\left(dN\chi^2\right)$ when the matrices Γ and λ are at most of size $\chi \times \chi$. This description is efficient, provided that the matrix size χ (also known as the *bond dimension*) is restricted - which it is, if the amount of entanglement in the system is bounded. Further the MPS structure entails that local gates applied to the whole state only change the matrices of the sites they act on - thus updates are inexpensive, as opposed to a full state vector description.

B. Two-site gates

The crucial ingredient in this simulation scheme is the SVD. It is the solution to the question of how to apply a two-site gate to an MPS or MPO.

The fact that a gate G acting on the two sites k and $k+1$ only changes the matrices local to these sites can easily be seen from

$$G_{k,k+1} | \psi \rangle = \sum_{i_1 \ldots i_N} \Gamma^{[1] i_1} \cdot \lambda^{[1]} \cdot \ldots \cdot \Gamma^{[N] i_N} \tag{4}$$
$$| i_1 \ldots i_{k-1} \rangle G_{k,k+1} | i_k i_{k+1} \rangle | i_{k+2} \cdots i_N \rangle$$
$$= \sum_{i_1 \ldots i_N} \sum_{j_k, j_{k+1}} \Gamma^{[1] i_1} \cdot \lambda^{[1]} \cdot \ldots \cdot \Gamma^{[N] i_N}$$
$$\langle i_k i_{k+1} | G_{k,k+1} | j_k j_{k+1} \rangle | i_1 \ldots i_N \rangle. \tag{5}$$

Above we first introduced the completeness relation for j_k and j_{k+1}, followed by switching indices i_k with j_k and i_{k+1} with j_{k+1}. Identifying all terms related to j_k and j_{k+1} now defines a tensor Θ of fourth order

$$\Theta\left(i_k, i_{k+1}, \alpha, \beta\right) = \sum_{\gamma} \lambda_{\alpha}^{[k-1]} \Gamma_{\alpha,\gamma}^{[k]i_k} \cdot \lambda_{\gamma}^{[k]} \cdot \Gamma_{\gamma,\beta}^{[k+1]i_{k+1}} \cdot \lambda_{\beta}^{[k+1]} \tag{6}$$

which can be built at a numerical cost of $\mathcal{O}\left(d_k \cdot d_{k+1} \cdot \chi^3\right)$ basic operations. This tensor needs to be updated when applying the gate G. The update rule from Eq. (5) yields the relation

$$\tilde{\Theta}\left(i_k, i_{k+1}, \alpha, \beta\right) = \sum_{j_k, j_{k+1}} \langle\, i_k i_{k+1} \,| G_{k,k+1} |\, j_k j_{k+1} \,\rangle \cdot \\ \cdot\, \Theta\left(j_k, j_{k+1}, \alpha, \beta\right). \tag{7}$$

This sum is performed for all α and β - which in general run from 1 to χ. Thus there are $d_k \times d_{k+1}$ products, which gives the number of basic operations for the update of Θ as $\mathcal{O}\left(d_l^2 \cdot d_r^2 \cdot \chi^2\right)$ [12]. This formula however only enables the update of the complete matrix Θ and not of the single entities $\Gamma^{[k]}$, $\lambda^{[k]}$ and $\Gamma^{[k+1]}$. They still have to be extracted from these updated products. To do this, Θ is written in a blocked index form $\Theta_{(d_k\chi),(d_{k+1}\chi)}$ which then is singular value-decomposed [12]. The general singular value decomposition (SVD) scales as $\mathcal{O}\left(m \cdot n^2\right)$ for a $m \times n$-matrix, thus resulting in an overall computational cost of $\mathcal{O}\left(d_l \cdot d_r^2 \cdot \chi^3\right)$. This makes the SVD the real bottleneck in the simulation, consuming by far the most resources, and therefore the first target for improvements.

Here we stick to the (unofficial) standard notation for MPS parameters in the context of the TEBD algorithm and denote the diagonal matrices as well as their entries by λ. During our discussion of the singular value decomposition though we switch to the respective notational standards where the i'th singular value will be denoted by σ_i - which however is the same quantity as the λ_i in this section.

C. Error analysis

During a TEBD simulation, the two main error sources are the Trotter and the truncation error. Other small error sources, often depending on implementational or algorithmical choices, are neglected in the following analysis.

TEBD relies heavily on the nearest-neighbor structure of the underlying Hamiltonian to implement time evolution. Traditionally this is done by standard Suzuki-Trotter decompositions [18] where the total Hamiltonian is split into two terms $H = F + G$ with each F and G are the sum over all even and odd (respectively) terms of the Hamiltonian. This way all terms within F (G) commute with each other, incurring no error while applying an operator of the form $\exp\left(\alpha F\right)$ ($\alpha \in \mathbb{C}$). The most straight forward and illustrative example is the standard 3rd-order expansion

$$\exp\left(iH\delta t\right) = \exp\left(i\left(F+G\right)\delta t\right) = \\ = \exp\left(i\frac{1}{2}F\delta t\right) \cdot \exp\left(iG\delta t\right) \cdot \exp\left(i\frac{1}{2}F\delta t\right) + \mathcal{O}(\delta t)^3. \tag{8}$$

This leads to three sweeps ($\frac{1}{2}F$, G, $\frac{1}{2}F$) of non-overlapping gate applications, comprising one time step. Various higher-order schemes for such decompositions exists, differing in the number of sweeps and the order of the resulting error [19]. However, these kind of schemes may introduce non-orthogonal components since the order of gate applications is not successive. This can be circumvented by resorting to schemes that produce sweeps with ordered, successive gate applications [20].

A decomposition to order p introduces an error of order $\epsilon_{\delta t} = \left(\delta t\right)^{p+1}$. The error incurred in one time step in general scales linearly with the system size N. This is due to the nested commutators occurring in the error term of the Suzuki-Trotter decomposition as can be seen when applying the Baker-Campbell-Hausdorff formula. Since the number of time steps taken is the total time T divided by the number of time steps $T/\delta t$, the total Trotter error $\epsilon_{trotter}$ is of order $\mathcal{O}\left(\left(\delta t\right)^p NT\right)$ [21].

The second considered error source is the truncation error. It stems from the truncation of the Schmidt values during the application of a two-site gate. Employing the Schmidt decomposition, a bipartite state can be written as

$$|\,\psi\,\rangle = \sum_{i=1}^{\chi'} \lambda_i |\,\psi_i^{\text{left}}\,\rangle |\,\psi_i^{\text{right}}\,\rangle + \sum_{i=\chi'+1}^{\chi} \lambda_i |\,\psi_i^{\text{left}}\,\rangle |\,\psi_i^{\text{right}}\,\rangle \\ = |\,\psi_{\text{trunc}}\,\rangle + |\,\psi_{\perp}\,\rangle \tag{9}$$

where the left sum until χ' denotes the kept part $|\psi_{\text{trunc}}\rangle$ and the right sum starting from $\chi'+1$ denotes the discarded part $|\,\psi_{\perp}\,\rangle$. Due to the fact that the $|\,\psi_i^{\text{left}}\,\rangle$ are mutually orthogonal (as are those of the right subsystem), the discarded part is orthogonal to the retained. Given that the squared Schmidt values sum up to 1, this truncation leads to a deviation in the norm of the state

$$\langle\,\psi_{\text{trunc}}\,|\,\psi_{\text{trunc}}\,\rangle = 1 - \sum_{i=\chi'+1}^{\chi} \lambda_i^2 = 1 - w \tag{10}$$

where we defined the *discarded weight* $w = \sum_{i=\chi'+1}^{\chi} \lambda_i^2$. Thus when renormalizing $|\,\psi_{\text{trunc}}\,\rangle$ we pick up a factor of $1/\left(1-w\right)$. Thus upon n truncations we are off by a factor of about $\left(1-w\right)^{n_t}$ with n_t being the number of truncations performed. Truncating each bond in each time step results in $n_t \propto \frac{NT}{\delta t}$ and thus the truncation error is about

$$\epsilon_{\text{trunc}} = \left(1-w\right)^{\frac{NT}{\delta t}} = \exp\left(\frac{NT}{\delta t}\ln\left(1-w\right)\right) \tag{11}$$

Thus we end up with a careful balancing of the two errors, depending on the size of the time step δt. For smaller δt we have a smaller truncation error. Yet this requires more truncations due to the larger number of time steps taken and thus in a larger truncation error.

III. AN ADVANCED APPLICATION OF THE TEBD ALGORITHM

The TEBD algorithm is remarkably useful also in scenarios which at first seem to be quite different from quantum many-body systems. One such example is its usage in the time evolving density matrix using orthogonal polynomials algorithm (TEDOPA) capable of treating open quantum systems. We briefly present the TEDOPA scheme to show in which regimes RRSVD proves to be most useful and how to speed-up previous simulations and refer to [22–24] for a more detailed presentation of the algorithm.

TEDOPA is a certifiable and numerically exact method to treat open quantum system dynamics [22, 24, 25]. It acts upon a spin-boson model description of an open quantum system where a central spin interacts linearly with an environment modelled by harmonic oscillators. In a two-stage process TEDOPA then first employs a unitary transformation reshaping the spin-boson model into a one-dimensional configuration. In a second step this emerging configuration is treated by TEBD, exploiting its full simulation power for one-dimensional systems.

The total Hamiltonian is split into system, environment and interaction part

$$H = H_{\text{sys}} + H_{\text{env}} + H_{\text{int}}, \tag{12}$$

$$H_{\text{env}} = \int_0^{x_{\max}} dx\ g\left(x\right) a_x^\dagger a_x, \tag{13}$$

$$H_{\text{int}} = \int_0^{x_{\max}} dx\ h\left(x\right) \left(a_x^\dagger + a_x\right) A. \tag{14}$$

The bosonic creation and annihilation operators $a_x^\dagger$ and a_x fulfill the usual bosonic commutation relations for the environmental mode x. The function $g\left(x\right)$ can be identified with the environmental dispersion relation; the function $h\left(x\right)$ gives the system-environment coupling strength for mode x between its displacement $\left(a_x^\dagger + a_x\right)$ and the operator A acting on the system.

Here the functions $g\left(x\right)$ and $h\left(x\right)$, together with the temperature, uniquely characterize an environment and define the spectral density $J\left(\omega\right)$ given by

$$J\left(\omega\right) = \pi h^2\left[g^{-1}\left(\omega\right)\right] \frac{dg^{-1}\left(\omega\right)}{d\omega}. \tag{15}$$

The interpretation of the quantity $\left(dg^{-1}\left(\omega\right)/d\omega\right)\delta\omega$ is the number of quantised modes with frequencies between ω and $\omega + \delta\omega$ (for $\delta\omega \to 0$). Further hold g the relations $g^{-1}\left[g\left(x\right)\right] = g\left[g^{-1}\left(x\right)\right] = x$.

Then new oscillators with creation and annihilation operators $b_n^\dagger$ and b_n can be obtained by defining the analytical transformation $U_n\left(x\right)$ as

$$U_n\left(x\right) = h\left(x\right) p_n\left(x\right), \tag{16}$$

$$b_n^\dagger = \int_0^{x_{\max}} dx U_n\left(x\right) a_x^\dagger. \tag{17}$$

It utilizes the orthogonal polynomials $p_n\left(x\right)$ defined with respect to the measure $d\mu\left(x\right) = h^2\left(x\right)dx$. While in certain cases it is possible to perform this transformation analytically, in general a numerically stable procedure is used [22, 23, 26].

This transformation yields a semi-infinite one-dimensional nearest-neighbor Hamiltonian

$$\begin{aligned} H =& H_{\text{sys}} + t_0 A\left(b_0 + b_0^\dagger\right) + \sum_{n=1}^{\infty} \omega_n b_n^\dagger b_n \\ &+ \sum_{n=1}^{\infty} t_n\left(b_n^\dagger b_{n+1} + b_n b_{n+1}^\dagger\right) \end{aligned} \tag{18}$$

whose nearest-neighbor geometry (which is necessary for the application of TEBD) as well as coefficients ω_n and t_n are directly related to the recurrence coefficients of the three-term recurrence relation defined by the orthogonal polynomials $p_n\left(x\right)$ with respect to the measure $d\mu\left(x\right) = h^2\left(x\right)dx$ [23].

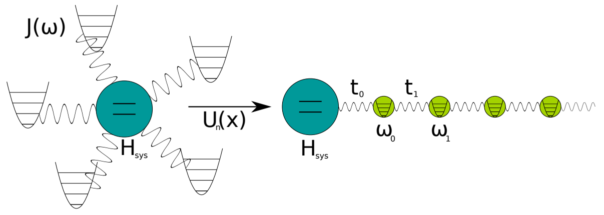


FIG. 1: Illustration of the spin-bonson model's transformation into a one-dimensional configuration where the system is only coupled to the environment's first site.

This transformation of the configuration is depicted in Fig. 1, from the spin-boson model on the left to a one-dimensional geometry on the right. In a last step it is necessary to adjust this configuration further to suit numerical needs. The number of levels for the environment's oscillator on site k can be restricted to $d_{k,\max}$ to reduce required computational resources. A suitable value for $d_{k,\max}$ is related to this site's average occupation, depending on the environment's structure and temperature. The number of sites that is required to faithfully represent the environment has to be sufficiently large to completely give the appearance of a "large" reservoir - one that avoids unphysical back-action on the system due to finite-size effects that lead to reflections at the system boundaries (see [27] for extensions of TEDOPA that can alleviate this problem considerably). It should be noted that these truncations, while feasible numerically and easily justifiable in a hand-waving kind of argument, in the end can also be *rigorously certified* by analytical bounds [25]. These adjustments yield an emerging system-environment configuration which is now conveniently accessible by TEBD.

IV. REDUCED-RANK RANDOMIZED SVD

The Singular Value Decomposition (SVD) is at the heart of the MPS representation and MPS-based algorithms, such as TEBD. The efficiency of TEBD comes from the possibility of approximating states living in an exponentially large Hilbert space with states defined by a number of parameters that grows only polynomially with the system size. In order to understand why the SVD plays such a crucial role, we introduce the following problem: given a complex $m \times n$ matrix A, provide the best rank-k ($k \leq n$) approximation of A. Without loss of generality we suppose $m \geq n$ and $rank(A) = n$. The solution to this problem is well known [28]: first compute the Singular Value Decomposition of $A = U\Sigma V^\dagger$, where $U = \left(U^{(1)}, U^{(2)}, \ldots, U^{(n)}\right)$, $V = \left(V^{(1)}, V^{(2)}, \ldots, V^{(n)}\right)$ are the left and right singular vectors of A respectively and $\Sigma = diag\left(\sigma_1, \sigma_2, \ldots, \sigma_n\right)$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$. Then retain the first k largest singular values $(\sigma_1, \sigma_2, \ldots, \sigma_k)$ and build the matrices $U_k = \left[U^{(1)}, U^{(2)}, \ldots, U^{(k)}\right]$ and $V_k = \left[V^{(1)}, V^{(2)}, \ldots, V^{(k)}\right]$. The matrix $\widetilde{A}_k = U_k \Sigma_k V_k^\dagger$ satisfies

$$||A - \widetilde{A}_k||_F = \sqrt{\sum_{i=k+1}^{n} \sigma_i^2} = \min_{rank(A')=k} ||A - A'||_F, \tag{19}$$

where $||\cdot||_F$ indicates the Frobenius norm

$$||A||_F = \sqrt{\sum_{i=1}^{n} \sigma_i^2}. \tag{20}$$

In other words, $\widetilde{A}_k$ provides the best rank-k approximation of A. This result justifies the application of SVD for the TEBD decimation step.

The computational complexity of the SVD of A is $\mathcal{O}(m \cdot n^2)$. For large matrices, the SVD can therefore require a significant amount of time. This is a crucial point since every single TEBD simulation step of an N-sites spin chain requires $\mathcal{O}(N)$ SVDs which usually consumes about 90% of the total simulation time.

As discussed in section II, the bond-dimension χ requires discarding of $n - \chi$ singular values (and corresponding left-/right- singular vectors). In the TEBD two-site update step, for example, we keep only χ singular values out of $n = d \cdot \chi$ (pure states) or $n = d^2 \cdot \chi$ (mixed states). Most of the singular values and left-/right-singular vectors are therefore discarded. It means that we are investing time and computational resources to compute information that is then wasted.

It is possible to avoid the *full* SVD of A and compute only the first k singular values and corresponding singular vectors by using *Truncated SVD* methods; such methods are standard tools in data-classification algorithms [29], signal-processing [30] and other research fields. The Implicitly Restarted Arnoldi Method [31, 32] and the Lanczos-Iteration [33] algorithms, both belonging to the Krylov-subspace iterative methods [34, 35], are two examples.

The Reduced-Rank Singular Value Decomposition (RRSVD), originally presented in by N. Halko *et al.* [15], is a *randomized* truncated SVD. It is particularly well suited to decompose structured matrices, such as those appearing in the TEBD simulation of non-critical quantum systems. Most interestingly, the algorithm is insensitive to the quality of the random number generator used, delivers highly accurate results and is, despite its random nature, very stable: the probability of failure can be made arbitrarily small with only minimal impact on the necessary computational resources.
In what follows we will describe the RRSVD algorithm and report the main results on stability and accuracy. For a full account on RRSVD we refer the reader to [15].

The RRSVD algorithm is a two-step procedure. The first step constructs an orthogonal matrix Q whose columns constitute a basis for the approximated range of the input matrix A. In the second step, the approximated SVD of A is computed by performing a singular value decomposition of $Q^\dagger A$.

The approximation of the range can be done either for fixed *error tolerance* ϵ, namely by finding a Q_ϵ such that

$$||A - Q_\epsilon Q_\epsilon^\dagger A|| \leq \epsilon, \tag{21}$$

or for a fixed rank k of the approximation, that is by finding Q_k such that

$$\min_{rank(X) \leq k} ||A - X|| \approx ||A - Q_k Q_k^\dagger A||. \tag{22}$$

The first one is known as the *fixed-precision approximation problem*, whereas the second is known as the *fixed-rank approximation problem.* Here and in what follows, we indicate by $||A||$ the operator norm, corresponding to the largest singular value of A.

If σ_j is the j-th largest singular value of A then

$$\min_{rank(X) \leq k(\epsilon)} ||A - X||_F = \\ = ||A - Q_{k(\epsilon)} Q_{k(\epsilon)}^\dagger A||_F = \sqrt{\sum_{j=k(\epsilon)+1}^{n} \sigma_j^2}. \tag{23}$$

where $\sigma_{k(\epsilon)}$ is the first singular value $\geq \epsilon$ and the columns of the rank-$k(\epsilon)$ matrix Q are the first $k(\epsilon)$ left singular vectors of A. However, this would require the knowledge of the first $k(\epsilon)$ singular values and vectors of A.

For the sake of simplicity, let us focus initially on the fixed-rank approximation problem. In order to determine Q_k, we resort to randomness. More precisely, we use a sample of $k+p$ random vectors $\omega^{(i)}$, whose components are independently drawn form a standard Gaussian distribution $\mathcal{N}_{(\mu=0,\sigma=1)}$. The set $\left\{\omega^{(i)}\right\}_{i=1}^{k+p}$ will be with *very high probability* a set of independent vectors. The parameter p determines the amount of *oversampling* needed to make the rank-k approximation of A more precise. The $m \times (k+p)$ matrix

$$Y = A\Omega,$$

where $\Omega = (\omega^{(1)}, \omega^{(2)}, \ldots, \omega^{(k+p)})$, will therefore have full rank $(k+p)$. By re-orthogonalizing the columns of the matrix Y we obtain a basis for the rank-$(k+p)$ approximation of the range of A. The re-orthogonalization can be done by using the QR-decomposition $Y = QR$. If $(k+p) < n$, the computational cost of the first step of this algorithm is dominated by the matrix multiplication $A\Omega$: this operation has an asymptotic complexity $\mathcal{O}(mn(k+p))$, whereas the QR decomposition of the $m \times (k+p)$ matrix Y has asymptotic complexity $\mathcal{O}(m(k+p)^2)$.

When the input matrix A is very large, the singular vectors associated with small singular values may interfere with the calculation. In order to reduce their weight relative to the dominant singular values it is expedient to take powers of the original matrix A. So, instead of computing $Y = A\Omega$ we compute

$$Z = B\Omega = (AA^\dagger)^q A\Omega. \tag{24}$$

The singular vectors of $B = (AA^\dagger)^q A$ are the same as the singular vectors of A; for the singular values of B, on the other hand, it holds:

$$\sigma_j(B) = (\sigma_j(A))^{2q+1}, \tag{25}$$

which leads to the desired reduction of the influence on the computation of the singular vectors associated to small singular values. This "stabilizing" step, also referred to as *Power Iteration* step (PI), increases the computational cost of the first step of the RRSVD algorithm by a constant factor $(2q+1)$.

A side effect of the PI is the extinction of all information pertaining to singular vectors associated to small singular values due to the finite size of floating point number representation. In order to avoid such losses, we use intermediate re-orthogonalizations (see Algorithm 1). We point out that in the typical context where TEBD is successfully applied, the singular values of the according matrices decay very fast, so the PI scheme must be applied.

Since the RRSVD is a randomized algorithm, the quality of the approximation comes in the form of expectation values, standard deviations and failure probabilities. We report pertinent results that can be found (including proofs) in [15], as well as some particular results tuned to cases of specific relevance to applications for TEBD.

Theorem 1. *(Corollary 10.10 in [15]) Given a target rank $k \geq 2$ and an oversampling parameter $p \geq 2$, let Q_Z be the orthogonal $m \times (k+p)$ matrix consisting of the fist $k+p$ left singular vectors of the matrix Z defined in (24). We define $P_Z = Q_Z^\dagger Q_Z$. Then*

$$\mathbb{E}\left(||\left(\mathbb{I} - P_Z\right)A||\right) \leq \tag{26}$$

$$\leq \left[\left(1+\sqrt{\frac{k}{p-1}}\right)\sigma_{k+1}^{2q+1} + \frac{e\sqrt{k+p}}{p}\left(\sum_{j>k}\sigma_j^{2(2q+1)}\right)^{\frac{1}{2}}\right]^{\frac{1}{2q+1}}$$

$$\leq \left[1+\sqrt{\frac{k}{p-1}} + \frac{e\sqrt{k+p}}{p}\sqrt{n-k}\right]^{1/(2q+1)} \sigma_{k+1}.$$

The application of the PI scheme reduces the average error exponentially in q. In order to quantify the deviations from the expected estimation error we use the following facts:

$$||\left(\mathcal{I} - P_Z\right)A||^{2q+1} \leq ||\left(\mathcal{I} - P_Z\right)B|| \tag{27}$$

(Theorem 9.2 [15]) and

$$||\left(\mathcal{I} - P_Y\right)A|| \leq \tag{28}$$

$$1+6\sqrt{(k+p)\cdot p\log(p)}\sigma_{k+1} + 3\sqrt{k+p}\left(\sum_{j>k}\sigma_j^2\right)^{\frac{1}{2}},$$

with probability greater or equal to $1-\frac{3}{p^p}$ (Corollary 10.9 [15]). We can now state the following

Corollary 1. *Under the same hypotheses of Theorem 1 it is*

$$P\left(||\left(\mathcal{I} - P_Z\right)A|| \leq \alpha^{\frac{1}{2q+1}}\sigma_{k+1} + \beta^{\frac{1}{2q+1}}\sum_{j>k}\sigma_j\right) \geq 1 - \frac{3}{p^p}, \tag{29}$$

with $\alpha = (1+6\sqrt{(k+p)\cdot p\log(p)})$ *and* $\beta = 3\sqrt{k+p}$.

Proof. By applying (28) to $B = (AA^\dagger)^q A$, and using (25) we have

$$P\left(||\left(\mathcal{I} - P_Z\right)B|| \leq \alpha\sigma_{k+1}^{2q+1} + \beta\left(\sum_{j>k}(\sigma_j^{2q+1})^2\right)^{\frac{1}{2}}\right) \geq 1 - \frac{6}{p^p}.$$

Using the relation (27) we have that

$$||\left(\mathcal{I} - P_Z\right)A|| \leq \left(\alpha\sigma_{k+1}^{2q+1} + \beta\left(\sum_{j>k}(\sigma_j^{2q+1})^2\right)^{\frac{1}{2}}\right)^{1/(2q+1)}$$

Algorithm 1: **Randomized SVD with Power Iterations**

```
Require: $m \times n$ matrix $A$; integers $l = k + p$ (rank of the
    approximation) and $q$ (number of iterations).
 1: Draw an $n \times l$ Gaussian matrix $\Omega$.
 2: Form $Y_0 = A\Omega$.
 3: Compute the QR factorization $Y_0 = Q_0R_0$.
 4: for $j = 1, 2, \ldots, q$ do
 5:     Form $\widetilde{Y}_j = A^\dagger Q_{j-1}$.
 6:     Compute the QR factorization $\widetilde{Y}_j = \widetilde{Q}_j\widetilde{R}_j$.
 7:     Form $Y_j = A\widetilde{Q}_j$.
 8:     Compute the QR factorization $Y_j = Q_jR_j$.
 9: return $Q = Q_q$.
```

since the function $f(q) = a^{1/x}, a > 0, x > 0$ is convex, it holds

$$
\begin{aligned}
&\left(\alpha\ \sigma_{k+1}^{2q+1} + \beta\left(\sum_{j>k}(\sigma_j^{2q+1})^2\right)^{\frac{1}{2}}\right)^{1/(2q+1)}\\
&\leq \left(\alpha\ \sigma_{k+1}^{2q+1}\right)^{\frac{1}{(2q+1)}} + \left(\beta\sqrt{\sum_{j>k}(\sigma_j^{2q+1})^2}\right)^{\frac{1}{2q+1}}\\
&\leq \alpha^{\frac{1}{2q+1}}\sigma_{k+1} + \beta^{\frac{1}{2q+1}}\sqrt{\sum_{j>k}\sigma_j^2}\\
&= \alpha^{\frac{1}{2q+1}}||A - \widetilde{A}_k|| + \beta^{\frac{1}{2q+1}}||A - \widetilde{A}_k||_F. \qquad (30)
\end{aligned}
$$

□

The results provided by the algorithm are usually closer to the average value than those estimated by the bound (26), which therefore seems to be not tight. However, the important message of the preceding results is that by applying PI we obtain much better approximations of A than those provided by the original scheme while error expectations and deviations are under full control. It would be most convenient to have some means to check how close $QQ^\dagger A$ is to the original input matrix A. With such a tool, we could check the quality of the approximation; moreover, we would be able to solve the fixed-error approximation problem (21). In the TEBD setting, this would allow us to determine the bond dimension χ for an assigned value ϵ of the truncation error. The solution to this problem comes from this result:

Theorem 2. *(equation 4.3 supported by Lemma 4.1 [15]) With $M = (I - QQ^\dagger)A$*

$$
P\left(||M|| \leq 10\sqrt{\frac{2}{\pi}}\max_{i=1,2,\ldots,r}||M\omega^{(i)}||\right) \geq 1 - 10^{-r}, \qquad (31)
$$

where $\omega^i, i = 1, \ldots, r$ are standard normal random vectors.

Suppose that we have completed the first three steps of Algorithm 1. Set $Q = Q_0$, with $rank(Q) = l = k + p$ and choose the size r of the sample. The *Accuracy Check* algorithm (Algorithm 2) takes in input A, Q, r and ϵ and returns a new matrix Q' that satisfies the accuracy bound with probability $1 - 10^{-r}$.

Algorithm 2: **Accuracy check**

```
Require: $m \times n$ matrix $A$; rank-$k + p$ projector $P_Q = QQ^\dagger$;
    integer $r$; tolerance $\epsilon$.
 1: do
 2:     Set $l = k + p$.
 3:     Draw an $n \times r$ Gaussian matrix $\Omega_r$.
 4:     Compute $B = A\Omega_r$.
 5:     Compute $D = (I - P_Q)A\Omega_r = \left(d^{(1)}, d^{(2)}, \ldots, d^{(r)}\right)$.
 6:     Set MAX= $\max\left\{d^{(i)}\right\}_{i=1}^r$.
 7:     if (MAX $> \epsilon$) then
 8:         Build $\widetilde{Q} = Q|B$.
 9:         Set $l = l + r$.
10:         Compute the QR decomposition $\widetilde{Q} = \widetilde{Q}'\widetilde{R}'$.
11:         Set $Q = \widetilde{Q}'$.
12: while (MAX $> \epsilon$ and $l \leq n - r$ )
13: return $Q$.
```

The computational cost of the Accuracy Check depends on different parameters. The cost of each iteration step is $\mathcal{O}(m \cdot l^2)$. Then we have to consider the iterations. If we take r too small (e.g. $r = 1$), then if the starting rank-l approximation is not good enough, we might need many iteration to converge to the desired accuracy. As a rule of thumb, we suggest to double the rank of the approximation at each time. This will likely lead to oversampling, but still delivers a good performance balance.

Since the reference metric in TEBD is the Frobenius norm eq. (20), some estimate of the Frobenius norm via the operator norm is required. To this end we observe that the TEBD is successful when the correlations in the simulated system are sufficiently short-ranged, i.e. the "singular values decay fast enough". If the entanglement between distant parts of the system is non-vanishing, the decimation step will lead to an important loss of relevant information. As an example let us consider a spin-chain of size n and a bipartition $A = \{1, 2, \ldots l\}$, $B = \{l + 1, \ldots, n\}$ of the chain. Let $|k\rangle_A$ and $|j\rangle_B$ be orthonormal bases for subsystems A and B respectively. Then the spin-chain state $|\,\psi\,\rangle$ can be Schmidt-decomposed as

$$
|\,\psi\,\rangle = \sum_{k,j} c_{k,j}|\,k\,\rangle_A|\,j\,\rangle_B = \sum_i \sigma_i|\,i\,\rangle_A|\,i\,\rangle_B,
$$

where in the last equality we used the SVD decomposition of the matrix $C = (c_{j,k}) = U\Sigma V^\dagger$ to perform the Schmidt decomposition of $|\,\psi\,\rangle$. The Schmidt coefficients σ_i are the singular values of C, i.e. the diagonal elements of Σ [36]. The number of non-zero singular values is the Schmidt number. The amount of entanglement between the subsystems A and B can be quantified by the von Neumann, or entanglement, entropy [37] of the reduced density matrices ρ_A and ρ_B

$$S(\rho_A) = -\sum_i \sigma_i^2 \log(\sigma_i^2) = S(\rho_B). \tag{32}$$

The decay rate of the singular value is therefore directly related to the amount of entanglement shared between two parts of a system. If the system is highly entangled, the singular values will decay "slowly"; in the limiting case where the system is maximally entangled, the reduced density matrices will describe completely mixed states and the singular values will be all equal to each other. When the system is only slightly entangled, the singular values will decay very fast; in particular, if the state $|\,\psi\,\rangle$ is separable, the Schmidt number is equal to 1 (and $\sigma_1 = 1$). The behavior of the entanglement entropy in systems at, or close to, the critical regime has been thoroughly studied (see [4] and references therein) together with its dependence on the decay rate of the eigenvalues of the reduced density matrices $\rho_{A,B}$ [5, 6, 38, 39]. By oversimplifying the problem, we observe that if the singular values decay as $\sigma_j = 1/\sqrt{j}$ the entanglement entropy shows, when the system size $n \to \infty$, a divergence $\log^2(n)$, whereas if they decay as $\sigma_j = 1/j$ the entanglement entropy does converge, since an area law holds [4]. When the singular values decay as $1/j$, however, the truncation error $||A - \widetilde{A_k}||_F$ decreases slowly in k and the TEBD scheme becomes inefficient since any decimation will lead to considerable approximation errors (see Eq. (11)). For these reasons we consider the case of linearly decreasing singular values as an extremal case for the range of applicability of the TEBD scheme.

This observation provides a useful tool to estimate the Frobenius norm, which plays a central role in TEBD, through the operator norm. For any matrix A it is:

$$||A|| \leq ||A||_F \leq \sqrt{rank(A)}||A||.$$

This result holds in general. The inequalities are saturated, in particular, when $rank(A) = 1$. The upper bound on the Frobenius norm is the tighter the closer the singular values of A are to each other. If the singular values decay at least as $1/j$, on the other hand, we have the following result.

Fact 1. *Given a rank-n matrix A with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$ with $\sigma_j \leq \sigma_1/j, j = 1, \ldots, n$, it holds:*

$$||A||_F \leq \frac{\pi}{\sqrt{6}}||A|| = \sigma_1 \frac{\pi}{\sqrt{6}} \tag{33}$$

where $||A||$ indicates the operator norm.

Proof.

$$||A||_F = \sqrt{Tr\,(A^\dagger A)} = \sqrt{\sum_{i=1}^n \sigma_i^2} \leq \sigma_1 \sqrt{\sum_{i=1}^n \frac{1}{i^2}}. \tag{34}$$

The last term is upper bounded by

$$\sigma_1 \lim_{n\to\infty} \sqrt{\sum_{i=1}^n \frac{1}{i^2}} = \sigma_1 \frac{\pi}{\sqrt{6}}.$$

□

This result finds application in the Accuracy Check routine. If we set $P_Q = QQ^\dagger$ we find

$$\begin{aligned} ||(I - P_Q)A||_F &= \sqrt{Tr\left((A - P_Q A)^\dagger\,(A - P_Q A)\right)} \\ &= \sqrt{Tr\,(A^\dagger A) - (A^\dagger P_Q A)} \\ &= \sqrt{||A||_F^2 - ||A^\dagger Q||_F^2} \\ &\leq ||A||_F. \end{aligned} \tag{35}$$

Therefore

$$\begin{aligned} ||(I - P_Q)A||_F &\leq \frac{\pi}{\sqrt{6}}||A||_F \\ &\leq \frac{10}{3}\pi \max_{i=1,2,\ldots,r} ||(I - QQ^\dagger)A\omega^{(i)}|| \end{aligned} \tag{36}$$

with probability $1 - 10^{-r}$.

We point out that it is not really necessary to *estimate* the Frobenius norm of the error: given Q we can compute $||(I - QQ^\dagger)A||_F$ directly. However one should note that this computation of $QQ^\dagger A$ requires $2(m \cdot n \cdot k)$ float operations instead of the $2(m \cdot k \cdot r) + m \cdot n \cdot r$ operations required to get the error estimate.

Now that we have the orthogonal matrix Q whose columns constitute a basis for the approximated range of the input matrix A, we can directly compute the approximate SVD of A by:

i) Form the matrix $B = Q^\dagger A$.

ii) Compute the SVD of B: $B = \widetilde{U}\Sigma V^\dagger$.

iii) Form the orthonormal matrix $U = Q\widetilde{U}$.

The product $Q^\dagger A$ requires $\mathcal{O}((k+p) \cdot n \cdot m)$ floating point operations. The SVD of B, under the reasonable assumption that $k + p \leq n$, requires $\mathcal{O}(n \cdot (k+p)^2)$ operations. The product $Q\widetilde{U}$ requires $\mathcal{O}(m \cdot (k+p)^2)$ operations.

We conclude this section by presenting some results on the computational cost of the whole procedure and on some more technical aspects related to the implementation of the algorithm. The asymptotic computational cost of partial steps has been given at various places in this section. In summary, the real bottleneck of the complete RRSVD algorithm is the first matrix multiplication

$Y = A\Omega$ which has complexity $\mathcal{O}(m \cdot n \cdot (k+p))$. The value of p can be set in advance or determined by the Accuracy Check method described above. All remaining operations, such as QR decompositions and error estimation, have smaller computational complexity. If the TEBD scheme is applied in a non-adaptive way, i.e. the bond dimension is kept fixed at a given value χ, we use RRSVD to solve the fixed-rank problem. In this case RRSVD has complexity $\mathcal{O}(m \cdot n \cdot \chi)$. If the bond dimension χ is set independently of the input matrix size, the replacement of the standard SVD by RRSVD will therefore result in a speed-up linear in n. On the other hand, if we use an adaptive TEBD simulation where the bond dimension is set such that some bound on the approximation error is satisfied, the cost of RRSVD will (strongly) depend on the structural properties of the input matrices. If the singular values decay exponentially (short-range correlations), then the expected speed-up is roughly the same as for the non-adaptive scheme. If the simulated system presents long-range correlations, then the speed-up provided RRSVD will be less then linear in n, possibly even vanishing. However, TEBD itself is not the ideal tool to deal with systems exhibiting long-range correlations, so this is only a minor limitation.

Another crucial observation, related to the implementation, is that all the operations required by RRSVD are standard functions of either the Basic Linear Algebra Subprograms (BLAS) [40] or the Linear Algebra PACKage (LAPack [41]) libraries. Both libraries are heavily optimized and are available for single-core, multi-core (e.g. Intel Math Kernel Library (MKL) [42]) and Kilo-processor architectures (e.g. CuBLAS [43] and CULA [44]). Since TEBD simulations are usually run on many cores (on current cluster architectures often 8 or 16), RRSVD can take full advantage of the optimized libraries.

V. RRSVD CASE-STUDY AND PROFILING OF REAL TEBD SIMULATIONS

We start by showing that, despite its random nature, RRSVD produces very accurate results with surprisingly small fluctuations. To this end we test the described algorithm on a sample of relatively small structured matrices extracted from pure-state TEBD simulations of our standard system from section III, subsequently continuing to larger matrices from mixed-state TEBD simulations of the same system. We analyze the accuracy of RRSVD and its time performances, concluding the section by presenting how RRSVD impacts full TEBD simulations.

A. Stability analysis

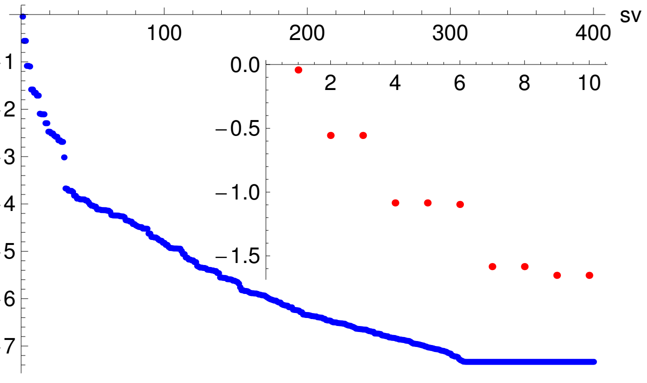
In order to benchmark the stability of the RRSVD algorithm we consider a set of 7 diagonal 750×750 matrices $\Sigma_m,\ m = 1, 2, \ldots, 7$. The matrices Σ_m are extracted from an actual pure-state TEBD simulation of our benchmark system described in section III. For every Σ_m we generate a set of $n_A = 20$ random matrices $\{A_{m,i}\}_{i=1}^{n_A}$ by randomly generating 1500×750 random orthonormal matrices $U_{m,i}$ and 750×750 random orthonormal matrices $V_{m,k}$; each $A_{m,i}$ has dimensions 1500×750. In this way we check for the dependence of both LaPack SVD and RRSVD on different matrices with the same structural properties. In order to take the random nature of RRSVD into account, for *each* $A_{m,i}$ we perform $n_R = 20$ executions of RRSVD. In this first benchmark we provide a rank-50 approximation of the rank-750 original matrices $A_{m,i}$ and show how the accuracy is related to the number of subspace iterations q. Motivated by the theorems reported in the previous section, we set $p = 50$. We compare the accuracy and timing results for different values of the iteration parameter $q = 2, 4, 6$. The *accuracy-check* part of the algorithm is not included here: we do not estimate the difference between the original matrix A and its projection on the reduced space.

By running the RRSVD on a set $\{A_{k,i}\}_{i=1}^{n_A}$ random realizations of matrices exhibiting the same structure, i.e. same singular values Σ_k, we check that the accuracy of RRSVD depends only, for fixed numbers of iterations q, approximation-rank k and oversampling parameter p, on the structural properties of the matrices. Therefore, in what follows we present an analysis referring to the instance corresponding to the random realization $A_{2,1}$, which is, in every respect, a good representative of the kind of matrices we deal with when performing a TEBD simulations of a pure quantum system far from the critical regime. In figure 2(a) we plot the singular values of $A_{2,1}$. It is important to notice that some of the largest singular values are very close to each other. This is a typical situation in which truncated-SVD methods belonging to the family of Krylov-subspace iterative methods are likely to require more iterations in order to accurately resolve the singular values. RRSVD, on the other hand, is completely insensitive to this peculiarity of the spectrum and provides very good results for the whole range of retained singular values ($k = 50$) starting from $q = 4$: the approximation error is comparable to that of the state-of-the-art LAPack SVD routine. Most noticeably, none of the n_R executions of RRSVD on the instance $A_{2,1}$ presents *outliers*, that is to say singular values that are computed with anomalous inaccuracy (figure 2(b)). The behavior of the approximation error as a function of q is compatible with the theoretical results stated in the previous section. In Table I we show the speed-up t_{SVD}/t_{RRSVD} when both, MKL-SVD and MKL-based RRSVD (see section V B for more information about the implementation), are executed on an Intel Xeon X5570@2.93GHz by 8 concurrent threads. The speed-up over the MKL-SVD is obviously decreasing as q and k increase: for $k = p = 100$ and $q = 6$ almost no advantage remains in applying RRSVD in-

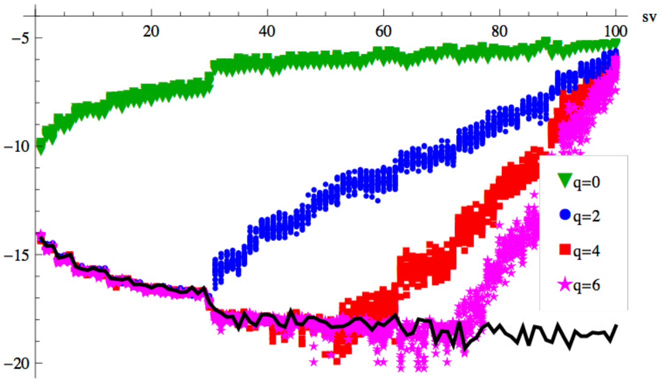
k/q	0	2	4	6	8	10
50	11.6	5.4	3.5	2.6	2.04	1.7
100	4.7	2.3	1.5	1.1	0.89	0.69

TABLE I: $A_{2,1}$: RRSVD Speed-up t_{SVD}/t_{RRSVD}. LAPack SVD time: 3.84 s

stead of the standard SVD. It is worth stressing here that RRSVD is meant to deliver a substantial advantage only for very large matrices and a comparatively small number of retained dimensions, as we will show later. We now turn our attention to the TEBD extremal case discussed previously. We consider an $m = 1500, n = 750$ random matrix A_c generated, as described at the beginning of this subsection, starting from singular values $\Sigma_c = diag(\sigma_1^c, \sigma_2^c, \dots, \sigma_n^c)$ with $\sigma_i^c = 1/i$. As shown in figure 3(a), in order to provide the same accuracy delivered by LAPack on the first k singular values, we need to increase the number of iterations. For $k = 50$ and $q = 10$ RRSVD is still able to provide some speed-up over the LAPack SVD. But the real problem is the truncation error: for $k = 50$ we have a truncation error of $\epsilon \approx 10^{-1}$. But in order to achieve $\epsilon < 10^{-2}$, about 650 singular values need to be retained. This results in a major loss of efficiency of the TEBD simulation scheme.

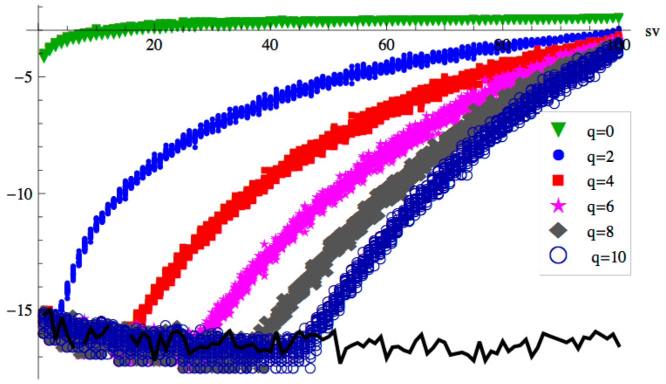


(a)



(b)

FIG. 2: Instance $A_{2,1}$: $m = 1500, n = 750$; $k = p = 50$. Discarded weight: $w = 4 \cdot 10^{-4}$ (a)Base-10 logarithmic plot of the singular values Σ_2: the decay appears roughly exponential. The inset shows the first 10 singular values: some of these SVs are very close to each other (about 10^{-5}) (b) The errors $\log_{10}(|\sigma_i - \sigma_i^{RRSVD}|)$ of the RRSVD for each singular value and for all $n_R = 20$ executions of RRSVD on the same instance matrix $A_{2,1}$ for different values of the iteration number q. The standard MKL SVD routine errors are shown as a thick black line.



(a)

FIG. 3: Instance A_c: $m = 1500, n = 750$; $k = p = 50$. The errors $\log_{10}(|\sigma_i - \sigma_i^{RRSVD}|)$ of the RRSVD as in figure 2(b) but referring to the singular values of the matrix A_c and for $q \in \{0, 2, 4, 6, 8, 10\}$. The discarded weight w for the chosen value of k is $w = 1 \cdot 10^{-1}$.

Therefore we can claim that RRSVD is indeed a fast and reliable method, able to successfully replace the standard SVD in the TEBD algorithm *in all situations* where TEBD can successfully be applied.

B. Performance on larger matrices and TEBD profiling

Now that the basic properties of the algorithm are established, we test it on larger matrices sampled from mixed-state TEBD simulation of our benchmark system from section III. Given the bond dimension χ and the local dimension d of the sites involved in the two-site-update, the size of the matrix given as input to the SVD is $d^2\chi \times d^2\chi$. In the following example we set $k = \chi = 100$ and the dimension of the local oscillators to $3, 4, 5, 6$ and 7 respectively. We therefore present results for matrices of dimensions $d_3 = 900 \times 900, d_4 = 1600 \times 1600, d_5 = 2500 \times 2500$, $d_6 = 3600 \times 3600$ and $d_7 = 4900 \times 4900$.

The structural properties of the test matrices considered are similar to the non-critical instances considered in the previous subsection.

We first analyzed the results provided by the RRSVD routine on a large sample of matrices (200 instances for each dimension). We determined that the RRSVD reaches the LAPack accuracy for a number $q = 2$ of PI steps.

We have developed three versions of the RRSVD algorithm: BL-, MKL- and GPU-RRSVD; each one uses a different implementation of the BLAS and LAPack libraries. BL-RRSVD is based on a standard single-thread implementation (CBLAS [45], LAPACKE [46]); MKL-RRSVD uses the Intel® implementation Math Kernel Library (MKL [42]); GPU-RRSVD exploits CUBLAS[43] and CULA [44], i.e. the BLAS and LAPack libraries for Nvidia® Graphics Processing Units (GPUs). RRSVD is available for single/double precision real/complex matrices in each version. We refer the reader to [47] for more details about our RRSVD implementations. During the completion of this work, another implementation of RRSVD from one of the authors of [15] has been reported in [48]. There the authors present three variants of RRSVD (essentially RRSVD with and without the PI and the Accuracy check) and discuss their performance on large (up to 6000×12000) real matrices with slowly decaying singular values. The implementation described in [48] is available for single-multi and Kilo processor architectures, as ours, but is limited to double precision real matrices. We are currently working on a full comparison between our and this other version of RRSVD.

In Table II we show the time required to perform the SVD/RRSVD of d_3, d_4, d_5,d_6 and d_7 double-precision complex matrices for the three implementations.

RRSVD provides a speed-up (figure 4) which is consistent with the predictions except in the 16-MKL case, there it grows stronger than linearly for matrix sizes in the range $d_3 - d_6$. This peculiar behavior is due to the low impact of the initial matrix multiplication $Y = A\Omega$ (and subsequent ones) on matrices of such sizes: this operation is heavily optimized for multi-threaded execution. Then the operations that determine the computational cost are the QR and final SVD decompositions, which have complexity $\mathcal{O}\left(m \cdot (k+p)^2\right)$. Since $k + p$ is kept constant, we have a quadratic speed-up. This justification is supported, for example, by the speed-up scaling of the 1-MKL case. However, as the matrix size increases the speed-up will tend to be linear (see the d_7 case). The performance on RRSVD provided by GPUs are comparable those delivered by the 16-MKL . Indeed, the standard SVD is faster on the GPU starting from size d_6. This behavior was expected, since our test matrices are still too small to take full advantage of the Kilo-processor architecture.

At last, we perform full TEBD simulations: for each dimension d_i, $i = 3, 4, 5, 6, 7$ we executed one TEBD

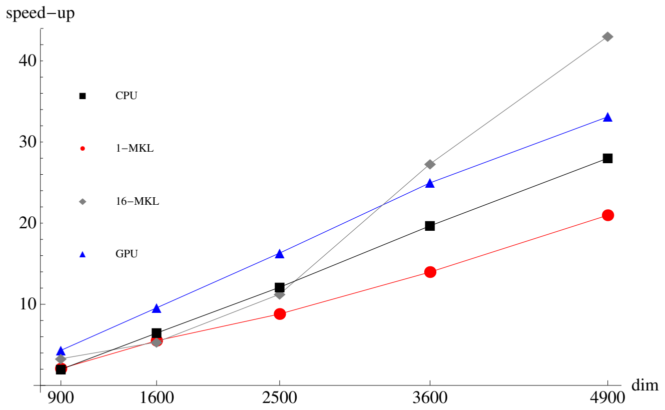


FIG. 4: $k = p = 100$; $q = 2$. Scaling of the SVD/RRSVD speed-up for different platforms as a function of the matrix size, as obtained from the data reported in Table II.

simulation with the standard SVD routine, and another TEBD with RRSVD. We ran all the jobs on the same cluster node, equipped with two Xeon X5570@2.93GHz with 8 cores each, as to assure a fair comparison. In figure 5 we show the average time required by the TEBD two-site update when standard SVD and RRSVD are used. The overall speed-up of the two-site update (inset of figure 5) agrees with the Amdahl law [49]. The average is taken over all maximum-sized two-site updates performed in the TEBD simulation (more than 1500 for every dimension). When the standard SVD is applied, it takes more than 90% of the overall computation time; however if RRSVD is employed, SVD-times reduce drastically. Its computational cost is of the same order as that of the building of the $\widetilde{\Theta}$ matrix (cf. figure 5). The fluctuations around the average RRSVD time are due to the action of the Accuracy-Check: from time to time the bond dimension χ must be increased in order to keep the approximation error below the threshold value $\epsilon = 10^{-3}$ required in the simulation. According to Corollary 1, for the choice $p = 100$ and $q = 2$, such an increase is motivated only by the decay rate of the singular values: the failure probability is smaller than $1/10^{100}$. This is confirmed by an *a posteriori* analysis of the matrices produced during our test TEBD simulations that required an extension of the bond dimension. The rank of the approximation proposed by RRSVD, for assigned tolerance ϵ, can therefore be used to detect either an entanglement build-up in the system or an (unlikely) anomalous behavior of the RRSVD itself.

VI. CONCLUSIONS AND OUTLOOK

The TEBD-algorithm, an important tool for the numerical description of one-dimensional quantum many-

	1-BL-SVD	1-BL-RRSVD	1-MKL-SVD	1-MKL-RRSVD	16-MKL-SVD	16-MKL-RRSVD	GPU-SVD	GPU-RRSVD
d_3	13.27	6.62	1.47	0.71	0.46	0.14	1.08	0.25
d_4	262.47	21.38	9.31	1.69	1.92	0.36	3.92	0.41
d_5	449.97	37.19	31.87	3.62	6.07	0.54	9.95	0.61
d_6	1464.67	74.48	97.37	6.97	22.93	0.84	21.97	0.88
d_7	1973.23	99.9	241.01	11.49	61.51	1.43	49.00	1.48

TABLE II: Execution time, in seconds, for the SVD of matrices of different sizes with LAPack SVD and RRSVD. The parameters for RRSVD are $q = 2$, $k = p = 100$. 1-BL: single-thread CBLAS-LAPACKE-based implementation, executed on a Intel i7@2.66GHz processor. MKL: MKL-based implementation; 1-MKL: with one MKL thread, 16-MKL: with 16 MKL-threads, executed on one and two 8-core Xeon X5570@2.93GHz respectively. GPU: CUBLAS-CULA implementation executed on a NVIDIA K20s; the timing in this case includes host-to-device and device-to-host memory transfers.

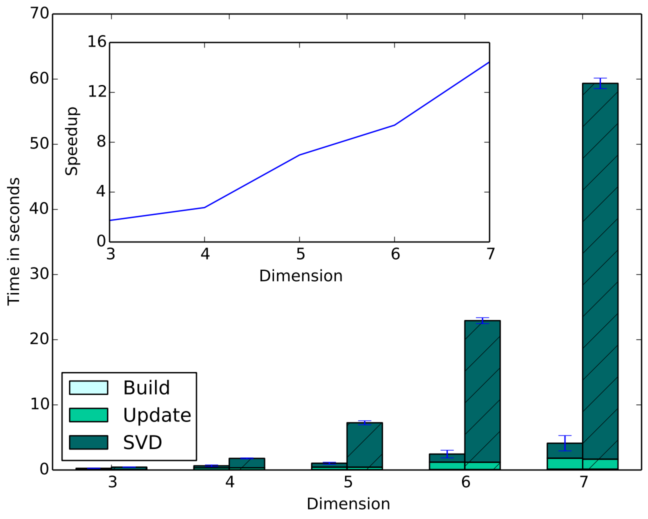


FIG. 5: Profile of the average time required by the SVD during a TEBD update step; left bars are for the RRSVD, right bars for the standard SVD. The data refers to the MKL implementation executed with 16 MKL-threads. The (unbiased) standard deviation of the execution time for the singular value decomposition part of the update step is shown as an error bar on top of each bar. The inset presents the overall speed-up for one complete TEBD two-site update achieved by the RRSVD.

body systems, depends essentially on the SVD which, in current implementations represents the bottleneck. We have demonstrated that this bottleneck can be addressed by the successful application of the RRSVD algorithm to TEBD simulations. The block decimation step of the TEBD update procedure is now approximately one order of magnitude faster than with the standard SVD without incurring additional accuracy losses. We note that in our test cases we have always chosen the RRSVD parameters such that we obtain singular values (and vectors) which were as precise as those provided by the standard (LAPack) SVD routine. By relaxing this requirement, the speed-up can be increased further. Moreover, by augmenting RRSVD with the Accuracy Check feature we are able not only to detect any very unlikely deviations from the average behavior, but also to understand whether the system is experiencing an entanglement build-up which would require an increasing of the retained Hilbert space.

In this paper we focused on the TEBD algorithm and its application to the one-dimensional system obtained through a TEDOPA mapping of an open quantum system (section III). In this context, RRSVD makes it possible to increase the dimension of the local oscillators with a much reduced impact on the computational time, thus allowing for the efficient simulation of the system at higher temperatures. However, all MPS algorithms relying on the SVD to select the relevant Hilbert space can greatly benefit from the use of the RRSVD routine, as long as the ratio between the number of retained and total singular values is sufficiently small.

The real scope and impact of this new computational tool is still to be understood fully. To this end, we prepared the *RRSVD-Package*, a library that provides the RRSVD routine for single-/double-precision and real-/complex-matrices. The package includes all the different implementations (BL, MKL, GPU) of RRSVD and has been designed to be plugged into existing codes through very minor code modifications: in principle, it suffices to replace each call to SVD by a call to RRSVD. This should allow for a very quick test of RRSVD in different simulation codes and scenarios. The library is written in C++; a Fortran wrapper, that allows to call of the RRSVD routine from Fortran code, is included as well. Some of the matrices used for the analysis of RRSVD in this paper are available, together with some stand-alone code that exemplifies the use of RRSVD and how to reproduce some of the results reported in this paper. The RRSVD-Package is freely available at [47] .

The results obtained for the GPU implementation are rather promising: for the largest matrices considered (d_5, d_6, d_7) the GPU performs as well as the cluster node.

Preliminary results on even larger matrices show that a GPU can become a valid alternative means to perform TEBD simulations of system with high local dimensions, or when the number of retained dimensions must be increased because of larger correlation lengths. Moreover, if operated in the right way, a GPU can act as a large cluster of machines [50] without the difficulties stemming from the need of distributing the workload among different computational nodes (Message Passing Interface (MPI)). A full GPU version of TEBD can make the access to super-computing facilities superfluous: a typical laboratory workstation equipped with one or two GPUs would be sufficient. We are currently re-analyzing the TEBD algorithm to expose further options for parallelization, as for example in the construction of the $\widetilde{\Theta}$ matrix. It could be computed by an ad-hoc designed CUDA-kernel and is a valid target for improvement now that its computational complexity is similar to that of the SVD.

ACKNOWLEDGEMENTS

This work was supported by an Alexander von Humboldt-Professorship, the EU Integrating project SIQS, the EU STREP projects PAPETS and EQUAM, and the ERC Synergy grant BioQ. The simulations were performed on the computational resource bwUniCluster funded by the Ministry of Science, Research and Arts and the Universities of the State of Baden-Württemberg, Germany, within the framework program bwHPC.

[1] R. Jozsa and N. Linden, Proc. R. Soc. Lond. A **459**, 2011 (2003).

[2] K. Audenaert, J. Eisert, M. B. Plenio, and R. F. Werner, Phys. Rev. A **66**, 042327 (2002).

[3] M. B. Plenio, J. Eisert, J. Dreißig, and M. Cramer, Phys. Rev. Lett. **94**, 060503 (2005).

[4] J. Eisert, M. Cramer, and M. B. Plenio, Rev. Mod. Phys. **82**, 277 (2010).

[5] F. G. S. L. Brandão and M. Horodecki, Nat. Phys. **9**, 721 (2013).

[6] F. G. S. L. Brandão and M. Horodecki, Comm. Math. Phys. **333**, 761 (2015).

[7] S. R. White, Phys. Rev. Lett. **69**, 2863 (1992).

[8] S. Rommer and S. Östlund, Phys. Rev. B **55**, 2164 (1997).

[9] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, Quantum Info. Comput. **7**, 401 (2007).

[10] S. R. White and A. E. Feiguin, Phys. Rev. Lett. **93**, 076401 (2004).

[11] J. J. Garcìa-Ripoll, New J. Phys. **8**, 305 (2006).

[12] G. Vidal, Phys. Rev. Lett. **91**, 147902 (2003).

[13] G. Vidal, Phys. Rev. Lett. **93**, 040502 (2004).

[14] U. Schollwöck, Ann. Phys. **326**, 96 (2011).

[15] N. Halko, P. Martinsson, and J. Tropp, SIAM Review **53**, 217 (2011).

[16] M. Zwolak and G. Vidal, Phys. Rev. Lett. **93**, 207205 (2004).

[17] A. H. Werner, D. Jaschke, P. Silvi, T. Calarco, J. Eisert, and S. Montangero, E-print arXiv:1412.5746 (2014).

[18] M. Suzuki, Phys. Lett. A **146**, 319 (1990).

[19] N. Hatano and M. Suzuki, in *Quantum Annealing and Other Optimization Methods*, Lecture Notes in Physics, Vol. 679, edited by A. Das and B. K. Chakrabarti (Springer Berlin Heidelberg, 2005) pp. 37–68.

[20] A. T. Sornborger and E. D. Stewart, Phys. Rev. A **60**, 1956 (1999).

[21] D. Gobert, C. Kollath, U. Schollwöck, and G. Schütz, Phys. Rev. E **71**, 036102 (2005).

[22] J. Prior, A. W. Chin, S. F. Huelga, and M. B. Plenio, Phys. Rev. Lett. **105**, 050404 (2010).

[23] A. W. Chin, A. Rivas, S. F. Huelga, and M. B. Plenio, J. of Math. Phys. **51**, 092109 (2010).

[24] M. P. Woods, R. Groux, A. W. Chin, S. F. Huelga, and M. B. Plenio, J. Math. Phys. **55** (2014).

[25] M. P. Woods, M. Cramer, and M. B. Plenio, E-print arXiv:1504.0xxxx (2015).

[26] W. Gautschi, J. Trans. Math. Softw. **20**, 21 (1994).

[27] R. Rosenbach, J. Cerrillo, S. F. Huelga, J. Cao, and M. B. Plenio, E-print arXiv:1504.0xxxx (2015).

[28] G. H. Golub and C. F. Van Loan, *Matrix computations*, 3rd ed. (Johns Hopkins University Press, 1996).

[29] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. (Springer-Verlag, 2009).

[30] E. J. Candes and B. Recht, Found. Comput. Math. **9**, 717 (2009).

[31] D. Sorensen, *Deflation for implicitly restarted Arnoldi methods*, Tech. Rep. (CAAM at Rice University, 1998).

[32] D. Sorensen, Acta Numer. , 519 (2002).

[33] R. M. Larsen, *Lanczos bidiagonalization with partial reorthogonalization*, Tech. Rep. DAIMI PB-357 (Department of Computer Science, Aarhus University, 1998).

[34] Y. Saad, SIAM J. Numer. Anal. **29**, 209 (1992).

[35] M. Hochbruck and C. Lubich, SIAM J. Numer. Anal. **34**, 1911 (1997).

[36] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 10th ed. (Cambridge University Press, New York, NY, USA, 2011).

[37] M. B. Plenio and S. Virmani, Quant. Inf. Comp. **7**, 1 (2007).

[38] P. Calabrese and A. Lefevre, Phys. Rev. A **78**, 032329 (2008).

[39] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, Phys. Rev. Lett. **100**, 030504 (2008).

[40] "http://www.netlib.org/blas/,".

[41] "http://www.netlib.org/lapack/," ().

[42] "https://software.intel.com/en-us/intel-mkl,".

[43] "https://developer.nvidia.com/cublas,".

[44] "http://www.culatools.com,".

[45] "http://www.netlib.org/blas/cblas,".

[46] "http://www.netlib.org/lapack/lapacke.html," ().

[47] “https://github.com/kindaguy/rrsvd.git,”.
[48] S. Voronin and P. Martinsson, arXiv:1502.05366 [math.NA] (2015).
[49] G. M. Amdahl, AFIPS Conf. Proc. **30**, 483 (1967).
[50] D. Tamascelli, F. S. Dambrosio, R. Conte, and M. Ceotto, J. Chem. Phys. **140**, 174109 (2014).