The Time-Dependent Variational Principle for Matrix Product States

An implementation

Ashley Milsted

January 9, 2012

Abstract

Efficient algorithms for the application of the time-dependent variational principle (TDVP) to matrix product states (MPS) exist. This document describes the differences between the algorithm for generic MPS published by Haegeman et al. [1] and the one implemented in *evoMPS*, as well as novel parts of the *evoMPS* algorithm that have not been published elsewhere.

This document provides a little background to MPS and the TDVP, and then describes the key differences between the implementation of the TDVP algorithm for generic MPS in *evoMPS* and that described in [1].

Contents

1 Introduction

The matrix product state (MPS) representation of the state of a one-dimensional quantum spin chain is known for its usefulness in efficiently simulating dynamics, as well as in computing ground states (e.g. via imaginary time evolution), due to its ability to efficiently capture states that have a limited amount of entanglement [5], [4].

An efficient algorithm for the application of the time-dependent variational principle (TDVP) to MPS for the generic (not translationally invariant) case was first described by Haegeman et al. [1], who also set out and implemented an algorithm for the translationally invariant case (uniform MPS or uMPS). The former has now also been implemented, with some modifications with respect to the published version, in *evoMPS*.

1.1 The Matrix Product State Representation

The MPS representation of a state $|\Psi\rangle$, for a one-dimensional spin chain of finite length N with open boundary conditions, can be written as

$$|\Psi[A]\rangle = \sum_{s_1 \dots s_N = 1}^{q_1 \dots q_N} A_1^{s_1} \dots A_N^{s_N} |s_1, \dots, s_N\rangle \tag{1}$$

where each $A_n^{s_n}$ represents a $D_{n-1} \times D_n$ matrix, which will henceforth be abbreviated to A^{s_n}. D_n is called the *bond dimension* and q_n is the dimension of the Hilbert space $\mathcal{H}_n = \mathbb{C}^{q_n}$ of a site on the chain. The left and right boundary vectors have been absorbed into the first and last matrices respectively, so that $D_0 = D_N = 1$. The boundary vectors have been included in the end-site matrices A^{s_1} and A^{s_N}, which are therefore row and column vectors respectively.

For the calculation of expectation values, the norm of the state and other quantities, it is useful to define the left and right density matrices

$$l_n = \sum_{s_n} A^{s_n\dagger} l_{n-1} A^{s_n} \tag{2}$$

$$r_{n-1} = \sum_{s_n} A^{s_n} r_n A^{s_n\dagger} \tag{3}$$

where $l_0 = r_N = 1$. As can be seen by writing out the norm of (1), $l_N = r_0 = \langle\Psi[A]|\Psi[A]\rangle$.

1.2 Gauge Freedom

A state represented in the above MPS form is invariant under gauge transformations

$$A^{s_n} \to g_{n-1} A^{s_n} g_n^{-1} \tag{4}$$

where the g_n are $D_n \times D_n$ invertible matrices and $g_0 = g_N \in \mathbb{C}$, as can easily be verified.

1.3 The Time Dependent Variational Principle for Matrix Product States

The TDVP works by projecting the exact time evolution of a state

$$\mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t}\left|\Psi(t)\right\rangle = \hat{H}(t)\left|\Psi(t)\right\rangle \tag{5}$$

(with units such that $\hbar = 1$) onto a variational manifold within the Hilbert space belonging to the system. The class of MPS with fixed bond dimensions D_n defines such a variational manifold $\mathcal{M}_{\mathrm{MPS}\{D_n\}}$ where the D_n are generally chosen to be less than would be required to exactly represent any state of the system. In this way, the computational complexity of the problem can be reduced by sacrificing the ability to represent states with higher levels of entanglement [5].

Since the variational manifold is, for practical purposes, smaller than the Hilbert space, projecting $\hat{H}(t)\left|\Psi(t)\right\rangle$ onto the manifold involves finding a tangent vector $\left|\Phi\right\rangle$ on the manifold that best approximates the exact time evolution.

For MPS, a general tangent vector has the form

$$\left|\Phi[B;A]\right\rangle = \sum_{n=1}^{N}\sum_{s_n=1}^{q_n} B^{s_n}\frac{\partial}{\partial A^{s_n}}\left|\Psi[A]\right\rangle \tag{6}$$

The tangent vector best approximating $\hat{H}(t)\left|\Psi(t)\right\rangle$ can then be written as $\left|\Phi[B^*(t);A(t)]\right\rangle$ where

$$B^*(t) = \arg_B \min \|\left|\Phi[B;A(t)]\right\rangle - \hat{H}(t)\left|\Psi[A(t)]\right\rangle\|^2 \tag{7}$$

Here, A and B represent the set of all matrices A^{s_n} and B^{s_n}.

At first sight, calculation of the optimal parameters $B^{s_n *}$ involves evaluating $\langle\Phi[\bar{B}]|\Phi[B]\rangle$, which, as can be seen from (6), includes terms that mix the B^{s_n} for different sites. Finding B^* would therefore require an iterative algorithm that sweeps over the chain several times.

Conveniently, the gauge freedom in the MPS representation (4) can be exploited to eliminate the site-mixing terms in $\langle\Phi[\bar{B}]|\Phi[B]\rangle$ such that sweeping is no longer required. This is achieved by restricting the choice of B^{s_n}, using a parametrization, to those that alter only the physical degrees of freedom (those that maintain the current gauge choice). This leads to the efficient algorithm for the TDVP set out in [1].

2 Differences in the *evoMPS* Implementation of the TDVP for MPS

2.1 Parametrization of the Tangent Space

The gauge freedom inherent in the MPS representation can be exploited to greatly simplify the application of the TDVP. The set of tangent vectors $\mathbb{T}_{\text{MPS}}[A]$ at a given point A on the variational manifold $\mathcal{M}_{\text{MPS}\{\text{D}_\text{n}\}}$ can be parametrized in a way that eliminates all non-physical degrees of freedom and, simultaneously, eliminates all non-local (site-mixing) terms from $\langle\Phi[\bar{B}]|\Phi[B]\rangle$, which must be evaluated in a TDVP implementation.

A parametrization of the tangent space $B^{s_n}[x_n]$ in terms of matrices x_n is chosen so that each B^{s_n} satisfies a gauge-fixing condition. In [1], the *left* gauge-fixing condition

$$\sum_{s_n=1}^{q_n} A^{s_n\dagger} l_{n-1} B^{s_n} = 0 \quad \forall n \in 1 \dots N \tag{8}$$

is used, leading to a parametrization

$$B^{s_n}(x_n) = l_{n-1}^{-1/2} V_L^{s_n} x_n r_n^{-1/2} \tag{9}$$

where $[V_L^{s_n}]_{\alpha,\beta} = [V_{L,n}]_{(s_n,\alpha);\beta}$ is defined to contain an orthonormal basis for the null space of the matrix L_n ($L_n V_{L,n} = 0$ and $V_{L,n}^\dagger V_{L,n} = \mathbb{1}$) with

$$[L_n]_{\alpha;(s_n,\beta)} = [A^{s_n\dagger} l_{n-1}^{1/2}]_{\alpha,\beta} \tag{10}$$

In *evoMPS*, the *right* gauge-fixing condition

$$\sum_{s_n=1}^{q_n} B^{s_n} r_n A^{s_n\dagger} = 0 \quad \forall n \in 1 \dots N \tag{11}$$

is used instead, with the corresponding parametrization

$$B^{s_n}(x_n) = l_{n-1}^{-1/2} x_n V_R^{s_n} r_n^{-1/2} \tag{12}$$

Here, $[V_R^{s_n}]_{\alpha,\beta} = [V_{R,n}]_{\alpha;(s,\beta)}$ contains an orthonormal basis for the null space of R_n ($V_{R,n} R_n = 0$ and $V_{R,n} V_{R,n}^\dagger = \mathbb{1}$) defined as

$$[R_n]_{(s_n,\alpha);\beta} = [r_n^{1/2} A^{s_n\dagger}]_{\alpha,\beta} \tag{13}$$

That the parametrizations satisfy the respective gauge-fixing conditions can easily be confirmed.

2.2 Norm-Preserving Dynamics

The TDVP can be modified so that the projection onto the variational manifold explicitly preserves the norm of the the state. In [1], this norm-preserving form of the TDVP was described, with (7) becoming

$$B^*(t) = \arg_B \min \| \,|\Phi[B; A(t)]\rangle - \left[\hat{H}(t) - H(t)\right] |\Psi[A(t)]\rangle \,\|^2 \tag{14}$$

In [1], $\hat{H}(t) - H(t)$ is present in the final TDVP algorithm for generic MPS. However, this is not necessary when using a tangent space that excludes changes to the norm ($\langle \Phi[B; A(t)] | \Psi[A] \rangle = 0$), as is the case for the gauge-fixing parametrizations (9) and (12). In this case, (14) and (7) are equivalent, as can easily be seen.

In *evoMPS*, this simplification is used in the implementation of the TDVP algorithm.

2.3 Inclusion of a Single-Site Hamiltonian Term

A single-site Hamiltonian term can be included in the nearest-neighbour term (making it non-uniform, if it was uniform before), thus allowing an algorithm designed for a non-uniform nearest-neighbour Hamiltonian to be used in this case too. However, it is also straightforward to introduce single-site terms into the algorithm for the TDVP explicitly, which has been done in *evoMPS*. See the next subsection for an example.

2.4 Implications for the Algorithm

Apart from the obvious differences in the parametrization of the tangent vectors, the choice of left or right gauge fixing determines how terms like $\langle \Phi[\bar{B}] | \, \hat{H} \, | \Psi[A] \rangle$ are calculated, since the left and right gauge fixing choices eliminate different terms in the resulting sum.

The lack of a need to use the explicitly norm-preserving form of the TDVP (14) means, compared to [1], merely changing $(\hat{H} - H)$ to $\hat{H}$ or, for a nearest-neighbour Hamiltonian $\hat{H} = \sum_n \hat{h}_{n,n+1}$ changing $(\hat{h}_{n,n+1} - h_{n,n+1})$ to $\hat{h}_{n,n+1}$, where $\hat{h}_{n,n+1}$ is the nearest neighbour term for sites n and $n+1$.

To illustrate the required changes to the algorithm published by Haegeman et al., the matrices C^{s_n,t_n} defined on page 14 of [1] become, in *evoMPS*,

$$C^{s_n,t_{n+1}} = \sum_{u_n,v_{n+1}}^{q_n,q_{n+1}} \langle s_n, t_{n+1} | \hat{h}_{n,n+1} | u_n, v_{n+1} \rangle \, A^{u_n} A^{v_{n+1}} \tag{15}$$

and the matrices K_n used to evaluate the Hamiltonian as required for the calculation of (7) become

$$K_n = \theta(n < N) \left[\sum_{s_n,t_{n+1}}^{q_n,q_{n+1}} C^{s_n,t_{n+1}} r_{n+1} {A^{t_{n+1}}}^\dagger {A^{s_n}}^\dagger + \sum_{s_n}^{q_n} A^{s_n} K_{n+1} {A^{s_n}}^\dagger \right] + \sum_{s_n,t_n}^{q_n,q_n} \langle s_n | \hat{h}_n^{\text{s.s.}} | t_n \rangle A^{t_n} r_n {A^{s_n}}^\dagger \tag{16}$$

where $K_{N+1} = 0$ and the recursion now runs from right to left due to the decision to use right gauge-fixing. θ is the Heaviside step function. $\hat{h}_n^{\text{s.s.}}$ is the single-site Hamiltonian term for site n. Since the H term from (14) was eliminated, $K_1 = \langle \Psi[A] | \hat{H} | \Psi[A] \rangle$ and gives us the energy expectation value (for a normalized state).

The $F_n = x_n^*$ matrices in [1] that represent the projection of the exact time evolution onto the MPS tangent space are, of course, also calculated differently in *evoMPS* due to right gauge-fixing and due to the inclusion of an extra term for the single-site part of the Hamiltonian.

3 Additional Numerical Integration Methods

Some features have been added to the algorithm for *evoMPS* that are not described, at least for the non-uniform case, in [1]. For example, Haegeman et al. describe a time-symmetrical implicit method for numerically integrating the TDVP flow equations for the translationally invariant case (uniform MPS). A modified version has been implemented for the non-uniform case in *evoMPS*. Additionally, the explicit fourth-order Runge-Kutta method has been implemented.

3.1 Implicit Midpoint Method

In [1], a numerical integration scheme is described for the uniform MPS case which is time-symmetric, and has a lower order error than the simple forward Euler method. It is intended to produce better results when simulating real time evolution, where errors accumulate over time.

It is an implicit midpoint method, where a midpoint $|\Psi[A(t + dt/2)]\rangle$ between $|\Psi[A(t)]\rangle$ and $|\Psi[A(t + dt)]\rangle$ is first determined via the backward Euler method by iteratively solving an implicit equation. From the midpoint, a further forward Euler step of $dt/2$ is taken to produce the point $|\Psi[A(t + dt)]\rangle$.

This method is complicated by the gauge freedom (4) because a finite step will, despite using a gauge-fixing parametrization of the tangent space, generally alter A along the gauge degrees of freedom by a small amount.

For this reason, iteratively solving the implicit equation to find the midpoint requires a gauge transformation to ensure that only physical degrees of freedom are compared.

For *evoMPS*, the midpoint method for uMPS has been adapted to generic MPS, where the implicit equation for the midpoint $\tilde{A}^{s_n} := A^{s_n}(t + dt/2)$ becomes

$$g_{n-1}^{-1} A^{s_n} g_n = \tilde{A}^{s_n} - \mathrm{i}\frac{dt}{2} B^{s_n}(\tilde{x}_n) \tag{17}$$

where $A^{s_n} = A^{s_n}(t)$ and g_{n-1} is computed by solving

$$\sum_{s_n} A^{s_n} g_n \tilde{r}_n \tilde{A}^{s_n} = g_{n-1} \tilde{r}_{n-1} \tag{18}$$

Starting from an initial forward Euler estimate

$$\tilde{A}_0^{s_n} = A^{s_n} + \mathrm{i}\frac{dt}{2} B^{s_n}(x_n) \tag{19}$$

where the subscript 0 indicates the current iteration, the equation can be solved iteratively for $A^{s_n}(t + dt/2)$ by setting

$$\tilde{A}_1^{s_n} = \tilde{A}_0^{s_n} + d\tilde{A}_0^{s_n} \tag{20}$$

where

$$d\tilde{A}_0^{s_n} = g_{n-1}^{-1} A^{s_n} g_n - \tilde{A}_0^{s_n} + \mathrm{i}\frac{dt}{2} B^{s_n}(\tilde{x}_{n,0}) \tag{21}$$

To quantify convergence, the size of the correction $d\tilde{A}^{s_n}$ for site n can be used

$$\zeta_n = \| d\tilde{A}^{s_n} \frac{\partial}{\partial A^{s_n}} |\Psi[\tilde{A}]\rangle \| = \mathrm{tr}\left[\tilde{l}_{n-1} \sum_{s_n} d\tilde{A}^{s_n} \tilde{r}_n d\tilde{A}^{s_n\dagger} \right]^{1/2} \tag{22}$$

and an overall error measure can be defined as

$$\zeta = \sum_n \zeta_n \tag{23}$$

To solve (17) for the whole state:

1. Evaluate (19) for all s_n to get the initial estimate for $\tilde{A}$.

2. Set $n \leftarrow N$ and $g_n \leftarrow 1$.

3. Solve (18) to find g_{n-1}.

4. Set $i \leftarrow 0$ and evaluate (21) to get $d\tilde{A}_i^{s_n}$.

5. Use the result and (20) to get an improved estimate $\tilde{A}_{i+1}^{s_n}$.

6. Set $i \leftarrow i + 1$ so that $\tilde{A}_i^{s_n} \leftarrow \tilde{A}_{i+1}^{s_n}$.

7. Repeat the previous three steps until the error ζ_n (22) is small enough.
8. Set $n \leftarrow n-1$ (and $g_n \leftarrow g_{n-1}$) and go to step 3.
9. Having individually optimized the A^{s_n} for each n, go to step 2 and repeat until the overall error (23) is small enough.

Note that the current estimate for the midpoint $\tilde{A}$ is always updated as soon as a new $\tilde{A}^{s_n}$ is generated and is used for the generation of the next B.

Having carried out the above, we have the midpoint $|\Psi[A(t + dt/2)]\rangle$ and can take a further forward step $dt/2$ to reach $|\Psi[A(t + dt)]\rangle$.

This algorithm is a simple extension of the algorithm for uniform MPS in [1], where the key difference is the need, in general, to sweep over the chain more than once to obtain a good solution. This is due to the dependence of B^{s_n}, for a given n, on the entire state, which is clear because the exact time evolution $\hat{H}\,|\Psi\rangle$ also depends of the entire state.

The current *evoMPS* implementation of this algorithm appears to have problems reaching the desired accuracy. The exact reason is currently unknown to the author.

3.2 Runge-Kutta Method

The well known explicit fourth order Runge-Kutta method for numerical integration [3] has also been implemented in *evoMPS*. Due to its explicit nature, there is no need to solve an equation when using this method and therefore no requirement to handle variation along the gauge degrees of freedom.

For cases tested by the author, this integration method seems to perform better for real time evolution than the midpoint method described in the previous subsection, achieving a better accuracy with fewer operations for the same step size.

4 Appendix

4.1 Obtaining a Canonical Form

A (right) canonical form for generic MPS with open boundary conditions is described in [2], which also includes details as to how the canonical form can be obtained from a general MPS using a gauge transformation (4).

The canonical form used in *evoMPS* is largely the same and can be written as

$$\sum_{s_n} A^{s_n} A^{s_n \dagger} = \mathbb{1} \quad \forall n \iff r_n = \mathbb{1} \quad \forall n \tag{24}$$

which leads to the entanglement spectrum for a cut between sites n and $n+1$ being contained in the left density matrices l_n (2). In *evoMPS*, these matrices are not diagonalized.

To obtain this form, essentially the same method is used as in [2], except that a Cholesky decomposition of $M = \sum_{s_n} A^{s_n} A^{s_n \dagger}$ is performed in place of the singular value decomposition, where M is taken to be positive definite.

4.2 Restoration of the Norm

Since, for a finite time step, the norm of the state may drift, it must be restored to 1 from time to time. This can be done by simply multiplying the vectors A^{s_1} or A^{s_N} by $(\langle\Psi|\Psi\rangle)^{-1/2}$, since $\langle\Psi|\Psi\rangle = l_N = r_0$ with r_n and l_n defined in (2) and (3).

In principle, one could modify any of the matrices A^{s_n} (for fixed n and $\forall s_n = 1 \dots q_n$). It is, however, more efficient to modify one of the end matrices because fewer steps are then required to calculate the new norm using (2) or (3). This is required since, in general, a single attempt to restore the norm in the above way will not succeed due to numerical inaccuracy. To solve this problem, the above operation is carried out repeatedly until the state is normalized to the desired precision.

References

[1] Jutho Haegeman, J. Ignacio Cirac, Tobias J. Osborne, Iztok Pižorn, Henri Verschelde, and Frank Verstraete. Time-Dependent variational principle for quantum lattices. *Physical Review Letters*, 107:70601, August 2011. Available from: `http://adsabs.harvard.edu/abs/2011PhRvL.107g0601H`.

[2] D. Perez-Garcia, F. Verstraete, M. M Wolf, and J. I Cirac. Matrix product state representations. *arXiv:quant-ph/0608197*, August 2006. Quantum Inf. Comput. 7, 401 (2007). Available from: `http://arxiv.org/abs/quant-ph/0608197`.

[3] WH Press, SA Teukolsky, WT Vetterling, and BP Flannery. Section 17.1 Runge-Kutta method. In *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, New York, 3rd edition. Available from: `http://www.nrbook.com`.

[4] F. Verstraete, V. Murg, and J.I. Cirac. Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Advances in Physics*, 57(2):143–224, 2008. Available from: `http://www.tandfonline.com/doi/abs/10.1080/14789940801912366`.

[5] Guifre Vidal. Efficient classical simulation of slightly entangled quantum computations. *arXiv:quant-ph/0301063*, January 2003. Phys. Rev. Lett.

91, 147902 (2003). Available from: `http://arxiv.org/abs/quant-ph/0301063`.