The time-dependent DMRG and its applications

Adrian Feiguin
Boulder Summer School 2010
Some literature

• G. Vidal, PRL 93, 040502 (2004)
• S.R. White and AEF, PRL 93, 076401 (2004)
• AEF and S.R. White, PRB 020404 (2005)
• U. Schollwoeck and S.R. White, arXiv:cond-mat/0606018
The density matrix projection

Solution: The optimal states are the eigenvectors of the reduced density matrix

\[ \rho_{ii'} = \sum_j \psi^*_i \psi_{i'j} \quad \text{Tr } \rho = 1 \]

with the \( m \) largest eigenvalues \( \omega_\alpha \).
We start from a small superblock with 4 sites/blocks, each with a dimension $m_i$, small enough to be easily diagonalized.

DMRG: The Algorithm

How do we build the reduced basis of states?
We grow our basis systematically, adding sites to our system at each step, and using the density matrix projection to truncate.
The finite size algorithm

We add one site at a time, until we reach the desired system size.
The finite size algorithm

We sweep from right to left

...Until we converge
The wave-function transformation

When we add a site to the left block we represent the new basis states as:

\[ |\alpha_{l+1}\rangle = \sum_{s_{l+1},\alpha_i} L^{l+1}_{s_{l+1},\alpha_i} |\alpha_i\rangle \otimes |s_{l+1}\rangle \]

Similarly for the right block:

\[ |\beta_{l+3}\rangle = \sum_{s_{l+1},\alpha_i} R^{l+3}_{s_{l+1},\alpha_i} |\beta_{l+4}\rangle \otimes |s_{l+3}\rangle \]
The wave-function transformation

Before the transformation, the superblock state is written as:

\[ |\psi\rangle = \sum_{\alpha_i, s_{l+1}, s_{l+2}, \beta_{l+3}} \psi(\alpha_i, s_{l+1}, s_{l+2}, \beta_{l+3}) |\alpha_i\rangle \otimes |s_{l+1}\rangle \otimes |s_{l+2}\rangle \otimes |\beta_{l+3}\rangle \]

After the transformation, we add a site to the left block, and we “spit out” one from the right block

\[ |\psi\rangle = \sum_{\alpha_{l+1}, s_{l+2}, s_{l+3}, \beta_{l+4}} \psi(\alpha_{l+1}, s_{l+2}, s_{l+3}, \beta_{l+4}) |\alpha_{l+1}\rangle \otimes |s_{l+2}\rangle \otimes |s_{l+3}\rangle \otimes |\beta_{l+4}\rangle \]

After some algebra, and assuming \( \sum |\alpha_i\rangle \langle \alpha_i | \approx 1 \), one readily obtains:

\[ \psi(\alpha_{l+1}, s_{l+2}, s_{l+3}, \beta_{l+4}) \approx \sum_{\alpha_i, s_{l+1}, \beta_{l+3}} L^{l+1}[s_{l+1}]_{\alpha_{l+1}, \alpha_i} \psi(\alpha_i, s_{l+1}, s_{l+2}, \beta_{l+3}) R^{l+3}[s_{l+3}]_{\beta_{l+3}, \beta_{l+4}} \]
Targeting states in DMRG

Our DMRG basis is only guaranteed to represent targeted states, and those only after enough sweeps!

\[ \rho = \sum_t w_t |\psi_t \rangle \langle \psi_t| \]

If we target the ground state only, we cannot expect to have a good representation of excited states (dynamics).

If the error is strictly controlled by the DMRG truncation error, we say that the algorithm is “quasiexact”.

Non quasiexact algorithms seem to be the source of almost all DMRG “mistakes”. For instance, the infinite system algorithm applied to finite systems is not quasiexact.
Solving the t-d Schrödinger Equation

\[ i\hbar \frac{\partial}{\partial t} \left| \Psi(t) \right\rangle = H \left| \Psi(t) \right\rangle \rightarrow \left| \Psi(t) \right\rangle = e^{-itH} \left| \Psi(t = 0) \right\rangle \]

Let us assume we know the eigenstates of \( H \)

\[ \left| \Psi(t = 0) \right\rangle = \sum_n c_n \left| \psi_n \right\rangle \]

\[ \left| \Psi(t) \right\rangle = e^{-itH} \left| \Psi(t = 0) \right\rangle \rightarrow \left| \Psi(t) \right\rangle = \sum_n c_n e^{-itE_n} \left| \psi_n \right\rangle \]

In reality, we work in some arbitrary basis

\[ \left| \Psi(t = 0) \right\rangle = \sum_k d_k \left| \varphi_k \right\rangle \]

\[ \rightarrow \left| \Psi(t) \right\rangle = \sum_k d_k e^{-itH} \left| \varphi_k \right\rangle \]

\[ = \sum_k d_k \sum_n a_{kn} e^{-itE_n} \left| \psi_n \right\rangle \]

Mixture of excited states with oscillating terms with different frequencies

Typically we avoid high freq. oscillations by adding a phase \( e^{-itH} \rightarrow e^{-it(H-E_0)} \)
Time evolution and DMRG: First attempts

- Cazalilla and Marston, PRL 88, 256403 (2002). Use the infinite system method to find the ground state, and evolved in time using this fixed basis without sweeps. This is not quasiexact. However, they found that works well for transport in chains for short to moderate time intervals.

\[ |\psi(t = 0)\rangle \quad t=0 \quad t=\tau \quad t=2\tau \quad t=3\tau \quad t=4\tau \quad |\psi(t)\rangle \]

- Luo, Xiang and Wang, PRL 91, 049901 (2003) showed how to target correctly for real-time dynamics. They target

\[ \psi(t=0), \psi(t=\tau), \psi(t=2\tau), \psi(t=3\tau) \ldots \]

This is quasiexact as $\tau \to 0$ if you add sweeping.

The problem with this idea is that you keep track of all the history of the time-evolution, requiring large number of states $m$. It becomes highly inefficient.
Adaptive Time-dependent DMRG:

In a truncated basis:

We need to "follow" the state in the Hilbert space adapting the basis at every step.

We would feel tempted to do something like:

\[ e^{-i\tau H} = e^{-i\tau(H_1+H_2+H_3+H_4...)} \approx e^{-i\tau H_1} e^{-i\tau H_2} e^{-i\tau H_3} e^{-i\tau H_4} ... \]

But it turns out that \[ e^{-i\tau(H_1+H_2)} \neq e^{-i\tau H_1} e^{-i\tau H_2} \] because \[ [H_1, H_2] \neq 0 \]

This actually would give you an error of the order of \( \tau^2 \), similar to a 1\(^{st}\) order S-T expansion...
Suzuki-Trotter approach

\[ |\psi(t = 0)\rangle \xrightarrow{e^{-i\tau H}} |\psi(t)\rangle \]

\[ H = H_1 + H_2 + H_3 + H_4 + H_5 + H_6 \]

\[ H_A = H_2 + H_4 + H_6 \]

\[ H_B = H_1 + H_3 + H_5 \]

\[ e^{-i\tau(H_A+H_B)} = e^{-i\tau H_A} e^{-i\tau H_B} e^{i\frac{\tau^2}{2}[H_A,H_B]} = e^{-i\tau H_A} e^{-i\tau H_B} e^{O(\tau^2)} \]
Suzuki-Trotter expansions

We want to write

\[ e^{(A+B)h+C_2h^2+C_3h^3+C_4h^4+O(h^5)} = \prod_{p=1}^{P} e^{a_pAh} e^{b_pBh} \]

with

\[ C_2 = \alpha(\{a_p, b_p\}) A, B \]
\[ C_3 = \beta(\{a_p, b_p\}) \{A, A, B\} + \gamma(\{a_p, b_p\}) \{B, B, A\} \]

We want to choose the \( a \)'s and \( b \)'s such that they kill the first \( K \) coefficients \( C_K \), minimizing the number of factors \( P \) for a given order, to obtain

\[ e^{(A+B)h+O(h^{K+1})} = \prod_{p=1}^{P} e^{a_pAh} e^{b_pBh} \]

We will impose the conditions that the operators enter symmetrically in the decomposition and \( \sum_p a_p = \sum_p b_p = 1 \).

Suzuki-Trotter expansions

First order:
\[ e^{(A+B)h+O(h^2)} = e^{Ah} e^{Bh} \]

Second order:
\[ e^{(A+B)h+\alpha(a,b)[A,B]h^2+O(h^3)} = e^{aAh} e^{bBh} e^{(1-a)Ah} e^{(1-b)Bh} \]
\[ e^{aAh} e^{bBh} e^{(1-a)Ah} e^{(1-b)Bh} \approx e^{(aA+bB)h} e^{((1-a)A+(1-b)B)h} \]
\[ \approx e^{(A+B)h+\frac{1}{2}ab[A,B]h^2+\frac{1}{2}(1-a)(1-b)[A,B]h^2+\frac{1}{2}a(1-b)[A,B]h^2-\frac{1}{2}(1-a)b[A,B]h^2} \]
\[ = e^{(A+B)h+(ab-b+1/2)[A,B]h^2} \]

We kill the second order term by choosing \( a=1/2; \ b=1 \)
\[ e^{(A+B)h+O(h^3)} = e^{Ah/2} e^{Bh} e^{Ah/2} \]
Suzuki-Trotter expansions

Fourth order:

\[
e^{(A+B)h+O(h^5)} = e^{a_1Ah} e^{b_1Bh} e^{a_2Ah} e^{b_2Bh} e^{a_2Ah} e^{b_3Bh} e^{(1-a_1-a_2-a_3)Ah} e^{(1-b_1-b_2-b_3)Bh}
\]

One solution (the most convenient expression) has the form (Forest-Ruth formula)

\[
e^{(A+B)h+O(h^5)} = e^{Ah\theta/2} e^{B\theta h} e^{(1-\theta)Ah/2} e^{(1-2\theta)Bh} e^{(1-\theta)Ah/2} e^{\theta Bh} e^{\theta Ah/2}
\]

with \( \theta = 1/(2 - \frac{3}{2}) \)
**Evolution using Suzuki-Trotter**

1\textsuperscript{st} order Suzuki-Trotter decomposition:

\[ e^{-i\mathcal{H}} \approx e^{-i\mathcal{H}_A} e^{-i\mathcal{H}_B} \]

where \( \mathcal{H} = \mathcal{H}_A + \mathcal{H}_B \). Here we make \( \mathcal{A} \) the even bonds and \( \mathcal{B} \) the odd, 1D only. The individual link-terms \( \exp(-i\tau \mathcal{H}_j) \) (coupling sites \( j \) and \( j + 1 \)) within \( \mathcal{H}_A \) or \( \mathcal{H}_B \) commute. Thus

\[ e^{-i\mathcal{H}_B} \equiv e^{-i\mathcal{H}_1} e^{-i\mathcal{H}_3} e^{-i\mathcal{H}_5} \ldots \]

No error introduced!

So the time-evolution operator is a product of individual link terms. Each link term only involves two-sites interactions => small matrix, easy to calculate!
The two-site evolution operator

Example: Heisenberg model (spins)

\[ H = \sum_i \vec{S}_i \cdot \vec{S}_{i+1} \quad \text{with} \quad \vec{S}_i \cdot \vec{S}_{i+1} = S_i^z S_{i+1}^z + \frac{1}{2} (S_i^+ S_{i+1}^- + S_{i+1}^+ S_i^-) \]

The two-site basis is given by the states

\[ |\sigma \sigma'\rangle = \{|\uparrow \uparrow\rangle; |\uparrow \downarrow\rangle; |\downarrow \uparrow\rangle; |\downarrow \downarrow\rangle\} \]

We can easily calculate the Hamiltonian matrix:

\[ H = \begin{pmatrix}
1/4 & 0 \\
-1/4 & 1/2 \\
1/2 & -1/4 \\
0 & 1/4 \\
\end{pmatrix} \]

**Exercise:** Exponentiate (by hand) the matrix by following these steps:
1. Diagonalize the matrix and calculate eigenvalues and eigenvectors
2. Calculate the exponential of \( H \) in the diagonal basis
3. Rotate back to the original basis
Evolving the wave-function

We want to apply the evolution operator between the two central sites:

\[ e^{-i\tau H_{ij}} \]

As we've seen before, the link evolution operator can be written as

And the wave function after the transformation:
tDMRG: The algorithm


We turn off the diagonalization and start applying the evolution operator:

$e^{-i\tau H_{ij}}$
**tDMRG: The algorithm**


Depending on the S-T break-up, a few sweeps evolve a time step

Each link term only involves two-sites interactions: small matrix, easy to calculate! Much faster than Lanczos!
**Time-step targeting method**

What if we don’t have a “nice” Hamiltonian, and S-T cannot be applied

The time-evolution can be implemented in various ways:

1) Krylov basis: Calculate Lanczos (tri-diagonal) matrix, and exponentiate. *(time consuming)*
2) Runge-Kutta. *(non-unitary!)*

- We target one time step accurately, then we move to the next step.
- We keep track of intermediate points between $t$ and $t+\tau$

AEF and S. R.White, PRB (05). See also P. Schmitteckert, PRB 70, 121302(2004)
Recall the fourth order Runge-Kutta method for integrating \( y'(t) = f(y, t) = f(y) \):

\[
k_1 = \tau f(y); \quad k_2 = \tau f(y + k_1/2); \quad k_3 = \tau f(y + k_2/2); \quad k_4 = \tau f(y + k_3);
\]

Then

\[
y(t + \tau) \approx y(t) + \frac{1}{6}(k_1 + 2(k_2 + k_3) + k_4)
\]

Using Mathematica, we find that to \( O(\tau^4) \),

\[
y(t + \tau/3) \approx y(t) + \frac{1}{162}(31k_1 + 14(k_2 + k_3) - 5k_4)
\]

\[
y(t + 2\tau/3) \approx y(t) + \frac{1}{81}(16k_1 + 20(k_2 + k_3) - 2k_4)
\]

The recipe is:

- Each half-sweep is one time step. At each step of the half-sweep, do one RK step, but without advancing \( t \to t + \tau \).

- At each step, target \( \psi(t) \), \( \psi(t + \tau/3) \), \( \psi(t + 2\tau/3) \), and \( \psi(t + \tau) \).

- At the last step, when the basis fully represents the states of the time step, advance to \( t + \tau \) more accurately using 10 RK steps with step \( \tau/10 \).
Sources of error

- **Suzuki-Trotter error**: Can be controlled by using higher order expansions, or smaller time-steps.

- **Truncation error**: In principle it can be controlled by keeping more DMRG states as the entanglement grows. Caveat: only works for “well-behaved” problems, since typically the entanglement grows uncontrollably.

- **Runge-Kutta/Krylov**: the error is dominated by the truncation error.

**Recipe**: instead of fixing the number of states for the simulation, we fix the truncation error, and we let the algorithm determine the optimal number of states... until the basis grows too large and the simulation breaks down. Hopefully this will enable us to go to large times...
$S=1$ Heisenberg chain ($L=32; t=8$)

$$E(t) = \sqrt{\frac{1}{L} \sum_{x=1}^{L} (S^z(x,t) - S^z_{Exact}(x,t))^2},$$

![Graph showing error vs. m for time targeting+RK and 1st and 4th order S-T methods.](image)
FIG. 3: Error $E(t = 8)$ for the Haldane chain for different time steps $\tau$: a) 1st, 2nd, and 4th order Suzuki-Trotter break-ups and $m = 160$; b) Runge-Kutta and $m = 100$. For smaller time-step we need more iterations→accumulation of error.
Fixed error, variable number of states

FIG. 4: Number of states required to keep a truncation error of $10^{-8}$, as a function of time. The results correspond to a R-K simulation of a Haldane chain with $L = 32$. 
Comparing S-T and time step targeting

- S-T is fast and efficient for one-dimensional geometries with nearest neighbor interactions
- S-T error depends strongly on the Trotter error but it can be reduced by using higher order expansions.
- Time step targeting (Krylov,RK) can be applied to ladders and systems with long range interactions
- It has no Trotter error, you can use a larger time-step, but it is more time consuming and you need more DMRG states.
- In RK simulations it is a good practice to do an intermediate sweep without evolving in time to improve the basis.
- Time evolution using RK is non-unitary (dangerous!). Krylov expansion is the right choice.
MPS representation of the time-evolution

A MPS wave-function is written as

$$|\psi\rangle = \sum_{\{s_i\}} \text{Tr} \left( A_1[s_1]A_2[s_2]...A_N[s_N] \right) s_1, s_2, ..., s_N$$

$$= \sum_{\{s_i\}} \sum_{\{\alpha_j\}} A_1[s_1]_{\alpha_1\alpha_2} A_2[s_2]_{\alpha_3\alpha_4} ... A_N[s_N]_{\alpha_N\alpha_1} |s_1, s_2, ..., s_N\rangle$$

The matrices can be represented diagramatically as

$$A[s]_{\alpha\beta} \equiv \alpha \begin{array}{c} S \end{array} \beta$$

And the contractions (coefficients), as:

$$\alpha_1 | S_1 \alpha_2 | S_2 \alpha_3 | S_3 \alpha_4 | S_4 \alpha_N | S_N \alpha_1$$
MPS representation of the time-evolution

The two-site time-evolution operator will act as:

\[ U \]

Which translates as:

\[ \sum_{s_4',s_5'} A_4[s_4']_\alpha_4 \alpha_5 U^{s_4',s_5'} A_5[s_5']_\alpha_5 \alpha_6 \]
**Swap gates**

In the MPS representation is easy to exchange the states of two sites by applying a "swap gate"

$$S[i, j] = \begin{array}{c}
S_i \\
S'_i \end{array} \begin{array}{c}
S_j \\
S'_j \end{array}$$

And we can apply the evolution operator between sites far apart as:

---

E.M Stoudenmire and S.R. White, NJP (2010)
Applications

1. Transport in nano-structures
2. Spectral properties, optical conductivity…
4. Time-dependent Hamiltonians.
5. Decoherence: Free induction decay, Hahn echo, Rabi oscillations, pulse sequences…

...