[Tutorial] tDMRG and TDVP: Real-time evolution

Author: Seung-Sup Lee

Here we will simulate the real-time evolution of quantum states in one-dimensional spin system, driven by time-independent Hamiltonian, by using **tDMRG (time-dependent density-matrix renormalization group)** and **TDVP (time-dependent variational principle)** methods.

Exercise (a): Complete `tDMRG_Ex.m`

There is a function `tDMRG_Ex.m` which is zipped together with this tutorial material. This function is designed to perform the real-time evolution of the MPS via the two-site tDMRG method, and to measure the expectation values of local operators at each time step. **Complete the parts which are enclosed by the comments `TODO - Exercise (a).`**

XY spin chain: Real-time evolution of domain wall

We consider the XY spin-1/2 chain,

$$\begin{aligned}\hat{H}_{\mathrm{XY}} &= +\sum_{\ell=1}^{N-1}(\hat{S}^x_\ell \hat{S}^x_{\ell+1} + \hat{S}^y_\ell \hat{S}^y_{\ell+1}) \\ &= +\frac{1}{2}\sum_{\ell=1}^{N-1}(\hat{S}^+_\ell \hat{S}^-_{\ell+1} + \hat{S}^-_\ell \hat{S}^+_{\ell+1}).\end{aligned}$$

The XY chain model has been already treated in the previous tutorial on the DMRG method for ground state search. As mentioned there, the system can be mapped onto non-interacting spinless fermions in one dimension via the Jordan-Wigner transformation. Note that here the coupling strength between neighboring spins is +1, while in the previous tutorial a it was -1. This sign difference is not physically relevant, of course, since it can be absorbed into the phase factors of the local basis.

Consider a chain of even length N, and prepare the state at initial time $t = 0$ such that the left half of the chain is up-polarized and the right half is down-polarized:

$$|\Psi(t=0)\rangle = |\uparrow\rangle_1 |\uparrow\rangle_2 \cdots |\uparrow\rangle_{N/2-1} |\uparrow\rangle_{N/2} |\downarrow\rangle_{N/2+1} |\downarrow\rangle_{N/2+2} \cdots |\downarrow\rangle_{N-1} |\downarrow\rangle_N.$$

That is, there is a domain wall in between sites $N/2$ and $N/2+1$. And let the state to be evolved in real time, via the unitary operator $\exp(-i\hat{H}_{\mathrm{XY}}t)$. We expect that the domain wall will be blurred out and spread, since the interaction terms in the Hamiltonian $\hat{H}_{\mathrm{XY}}$ flip nearest-neighbor spins that are anti-aligned. In the limit of $N \to \infty$, there exists the exact solution of magnetization

$$\langle\Psi(t)|\hat{S}^z_\ell|\Psi(t)\rangle = \frac{-1}{2}\sum_{n=1-(\ell-N/2)}^{(\ell-N/2)-1}[J_n(t)]^2, \quad \text{for } j > N/2$$

and $\langle\Psi(t)|\hat{S}^z_\ell|\Psi(t)\rangle = -\langle\Psi(t)|\hat{S}^z_{N-\ell}|\Psi(t)\rangle$, where $J_n(t)$ is the Bessel function of the first kind; see Eq. (4) in D. Gobert et al, Phys. Rev. E **71**, 036102 (2005) or in its arXiv version.

Let's compute magnetization $\langle\Psi(t)|\hat{S}^z_\ell|\Psi(t)\rangle$ by using the tDMRG, and compare with the exact result.

```
clear

% system parameter
J = +1; % coupling strength
N = 50; % number of sites in a chain

% DMRG parameter
Nkeep = 20; % bond dimension
dt = 1/20; % discrete time step size
tmax = 20; % maximum time

% Local operators
[S,I] = getLocalSpace('Spin',1/2);

% XY model
Hs = cell(1,N-1);
Hs(:) = {J*contract(S(:,1:2,:),3,2,permute(conj(S(:,1:2,:)),[3 2 1]),3,2)};

% operator to measure magnetization
Sz = squeeze(S(:,3,:));

% initialize MPS: product state such that the left half of the chain is
% up-polarized and the right half is down-polarized.
M = cell(1,N);
for itN = (1:N)
    if itN <= (N/2)
        M{itN} = [1,0];
    else
        M{itN} = [0,1];
    end
end

% tDMRG
[ts,M,Ovals,EE,dw] = tDMRG_Ex (M,Hs,Sz,Nkeep,dt,tmax);
```

```
tDMRG : Real-time evolution with local measurements
N = 50, Nkeep = 20, dt = 0.05, tmax = 20 (400 steps)
21-07-22 12:42:36 | Transform the MPS into right-canonical form.
21-07-22 12:42:36 | Trotter steps: start
21-07-22 12:42:38 | #40/400 : t = 2/20
21-07-22 12:42:39 | #80/400 : t = 4/20
21-07-22 12:42:41 | #120/400 : t = 6/20
21-07-22 12:42:44 | #160/400 : t = 8/20
21-07-22 12:42:46 | #200/400 : t = 10/20
21-07-22 12:42:49 | #240/400 : t = 12/20
21-07-22 12:42:51 | #280/400 : t = 14/20
21-07-22 12:42:54 | #320/400 : t = 16/20
21-07-22 12:42:57 | #360/400 : t = 18/20
21-07-22 12:43:00 | #400/400 : t = 20/20
Elapsed time: 24.47s, CPU time: 99.49s, Avg # of cores: 4.067
```

```
21-07-22 12:43:00 | Memory usage : 3.49GiB
```

The result `Ovals` is the matrix whose rows indicate the magnetization along the chain for every time steps. Since the time evolution involves complex numbers, `Oval` can be complex-valued in general. However, in this tutorial we measure magnetization, i.e., the expectation of the spin-z operator that is Hermitian, any imaginary part of `Ovals` here is numerical noise.

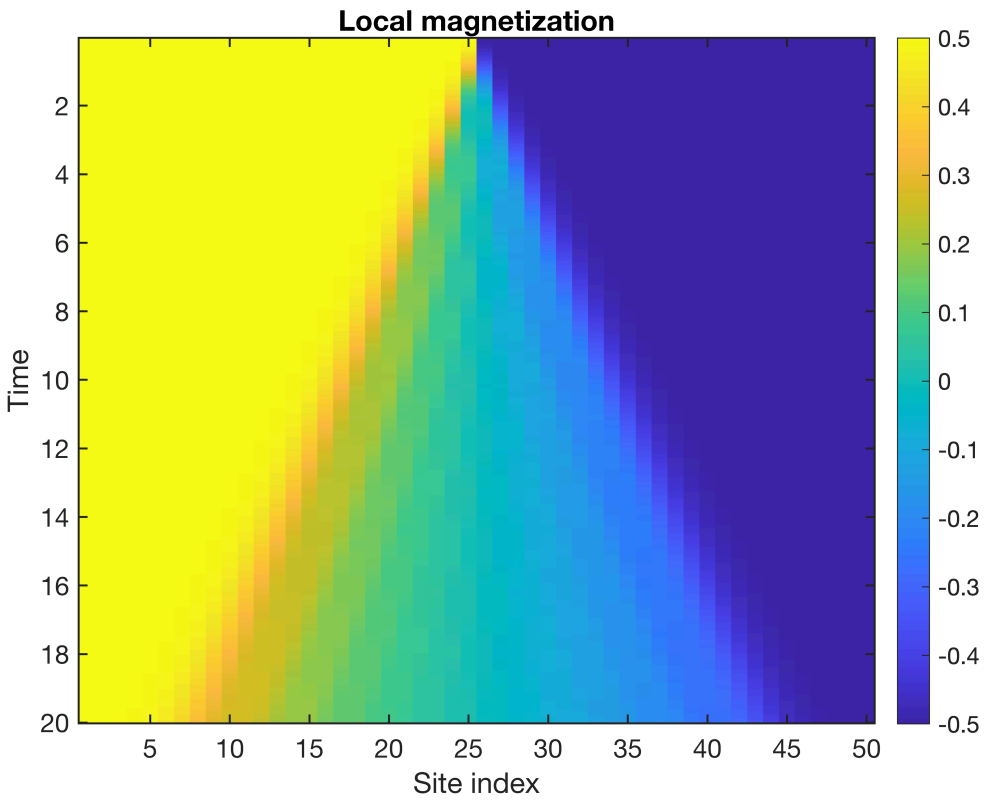
```
disp(mean(abs(imag(Ovals(:))))); % should be double precision noise
```

```
    2.1985e-17
```

```
Ovals = real(Ovals); % remove noise
```

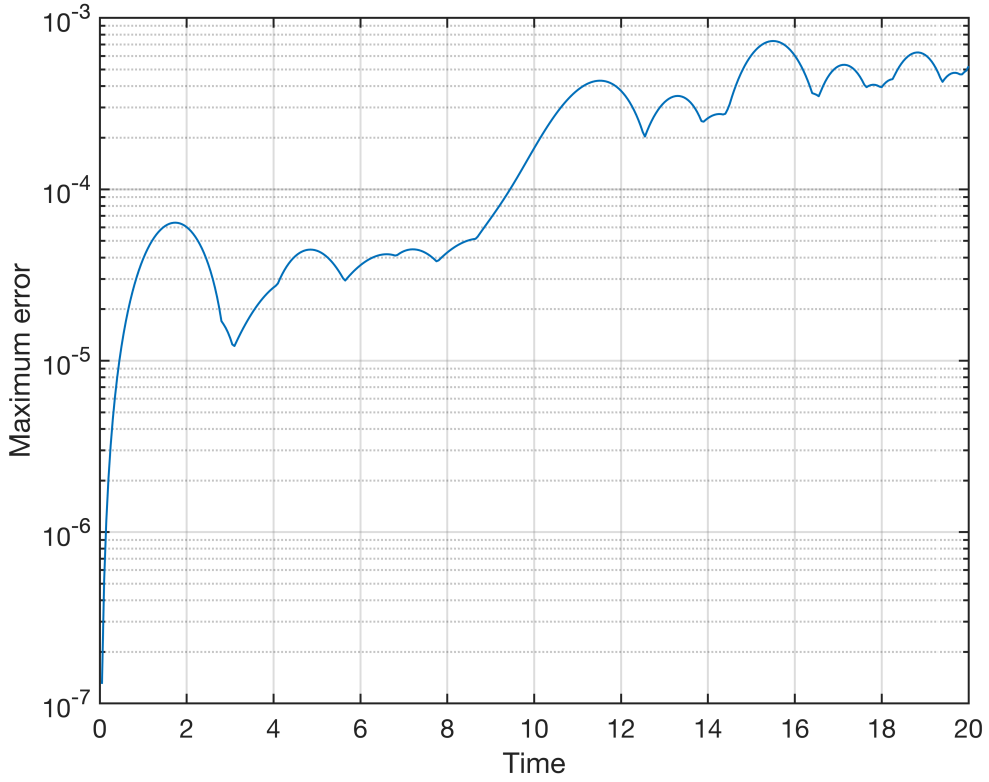
We see that the region in which the magnetization deviates from $\pm 1/2$ propagates linearly in time, and that the velocity is approximately 1. The "wavefronts" of the blurred domain wall (or, say, magnetization fluctuation) don't reach the ends of the system within this time window.

```
figure;
imagesc([1 N],[ts(1) ts(end)],real(Ovals));
colorbar;
set(gca,'FontSize',13,'LineWidth',1);
xlabel('Site index');
ylabel('Time');
title('Local magnetization');
```



Let's measure the error of the tDMRG result against the exact solution.

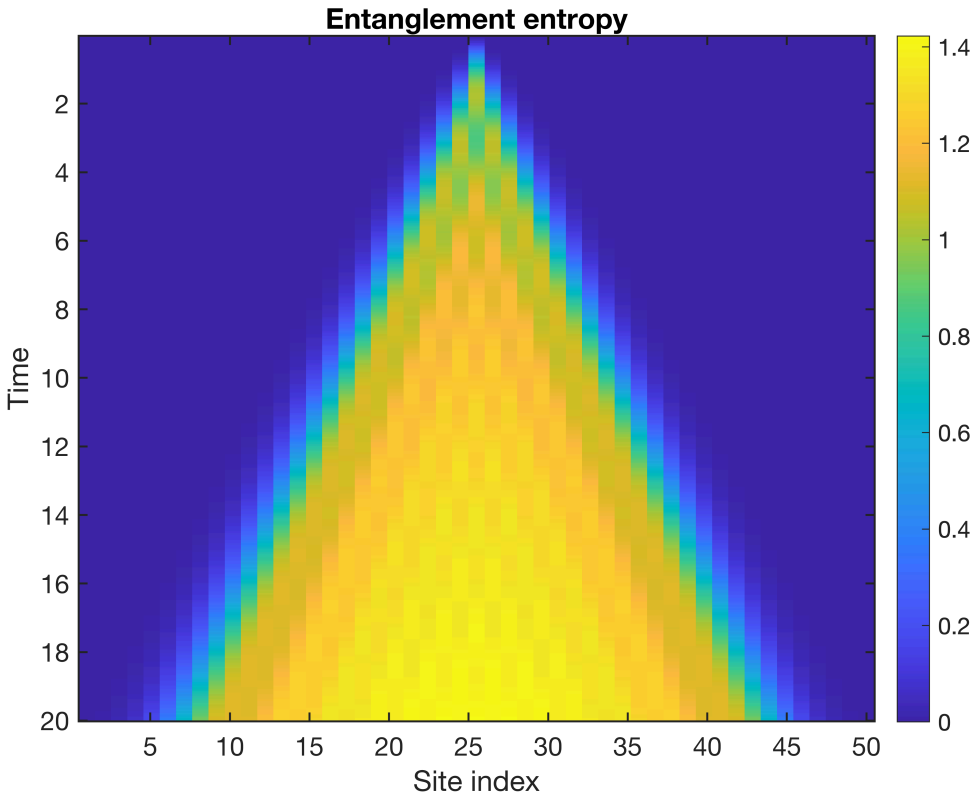
```
% % exact values of magnetization in the infinite chain limit
% cf. D. Gobert et al., Phys. Rev. E 71, 036102 (2005), Eq. (4)
fvals = zeros(numel(ts),N-1);
for it = (1:size(fvals,2))
    % Bessel function of the 1st kind
    fvals(:,it) = (besselj(it-(N/2),ts(:))).^2;
end
fvals = -0.5*fvals;

Oexact = zeros(numel(ts),N/2); % exact values
for it = (1:(N/2))
    Oexact(:,it) = sum(fvals(:,(N/2-it+1):(it+N/2-1)),2);
end
Oexact = [-fliplr(Oexact),Oexact];

% error between numerical and exact results
figure;
% maximum error along the chain at each time instance
plot(ts,max(abs(Ovals-Oexact),[],2),'LineWidth',1);
set(gca,'FontSize',13,'LineWidth',1,'YScale','log');
grid on;
xlabel('Time');
ylabel('Maximum error');
```



The error increases gradually with oscillations. Note that the exact solution holds for the limit $N \to \infty$: the exact solution describes the system **before** the wavefronts of magnetization fluctuation reach the ends of the system.

The spread of magnetization fluctuation is accompanied with the growth of the entanglement in the quantum state. We plot the entanglement entropy of the MPS with respect to the bipartition at individual bonds, for different times.

```
figure;
% every third rows, since each row is a decomposed Trotter
% steps applied to only part of the system
imagesc([1 N],[ts(1) ts(end)],EE(3:3:end,:));
colorbar;
set(gca,'FontSize',13,'LineWidth',1);
xlabel('Site index');
ylabel('Time');
title('Entanglement entropy');
```

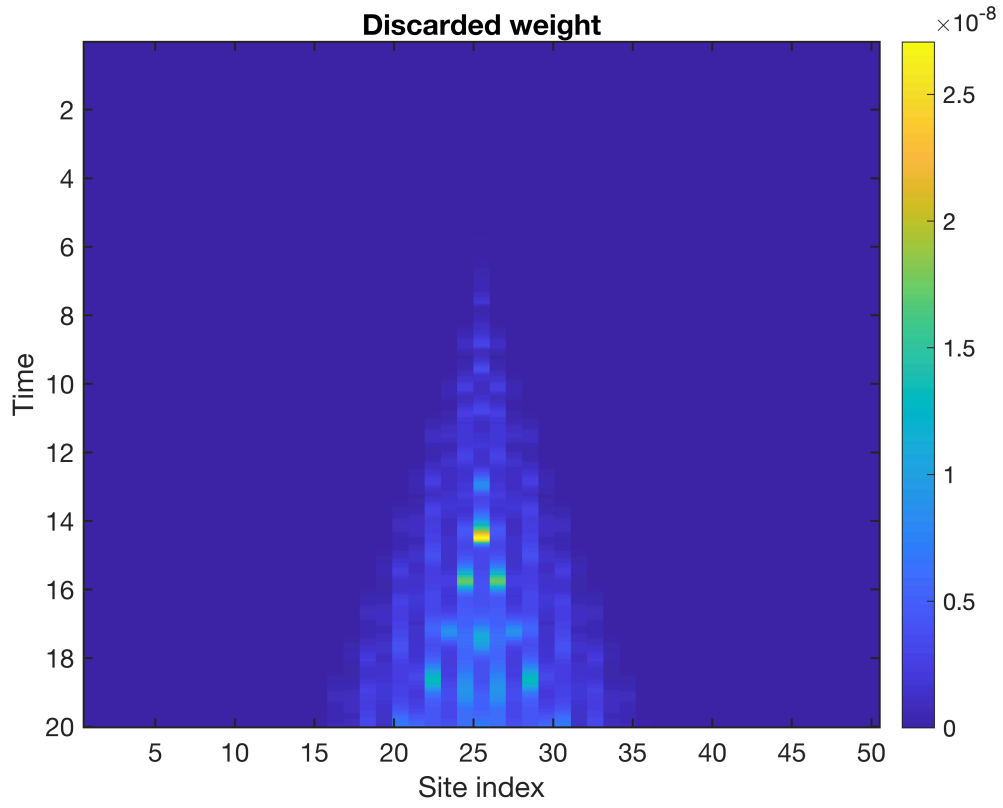


The entanglement entropy is the highest in the central region, and overall increases with time. As the entanglement grows, the bond dimension should also increase to fully describe the MPS. However, for the reason of computational cost, we typically fix the maximum bond dimension, as in this tutorial. Thus the smallest singular values need to be discarded after each singular value decomposition (SVD). The discarded weight is the sum of the squares of such discarded singular values after the SVD (See the documentation of `Tensor/svdTr.m` also). That is, the discarded weight measures the truncation error.

Let's plot the discarded weight. They become finite in the region where the entanglement becomes larger. But within this time window, the discarded weight are negligibly small.

```
figure;
% sum over every three (decomposed) Trotter steps, to obtain
```

```
% the value for every full time step by dt
imagesc([1 N],[ts(1) ts(end)], ...
    squeeze(sum(reshape(dw,[3, size(dw,1)/3, size(dw,2)]),1)));
colorbar;
set(gca,'FontSize',13,'LineWidth',1);
xlabel('Site index');
ylabel('Time');
title('Discarded weight');
```



Exercise (b): Error analysis

How the error changes with increasing/decreasing `Nkeep` and `dt`, with fixed `tmax = 20`?

(Hint : Refer to Sec. IV in D. Gobert et al, Phys. Rev. E **71**, 036102 (2005) or in its arXiv version)

Exercise (c): Longer time evolution

Perform the tDMRG calculation for longer time, say, `tmax = 80`. How would the results (magnetization, entanglement entropy, and discarded weight) look like?

Exercise (d): Different initial state where only one spin is up

Considere a different initial state,

$$|\Psi(t=0)\rangle = \hat{S}_n^+ |\downarrow, \ldots, \downarrow\rangle,$$

where only one spin at site n is up-polarized and the rest is down-polarized. Perform tDMRG calculation of local magnetization, and compare with the analytic result:

$$\langle\Psi(t)|\hat{S}^z_\ell|\Psi(t)\rangle = \left|\sum_{m=1}^{N}\frac{2}{N+1}\sin\left(\frac{\pi m}{N+1}n\right)\sin\left(\frac{\pi m}{N+1}\ell\right)e^{-it\cos\left(\frac{\pi m}{L+1}\right)}\right|^2 - \frac{1}{2}\,.$$

Actually, with this initial state, the tDMRG simulation can go over longer time window with smaller error, compared with the demonstrated example of the "half-half" initial state. It can be seen by analyzing the entanglement entropy and the discarded weight. **Can you explain why?**

Exercise (e): Complete `TDVP_1site_Ex.m`

There is a function `TDVP_1site_Ex.m` which is zipped together with this tutorial material. This function is designed to perform the real-time evolution of the MPS via the **one-site TDVP** method, and to measure the expectation values of local operators at each time step. **Complete the parts which are enclosed by the comments `TODO - Exercise (e).`** There are several remarks:

- The one-site TDVP algorithm for real-time evolution is quite similar to the one-site DMRG algorithm for ground-state search. So one may **recycle many lines from `DMRG/DMRG_1site.m`**! One can also follow the algorithm described in Appendix B of [J. Haegeman, C. Lubich, I. Oseledets, B. Vandereycken, and F. Verstraete, Phys. Rev. B **94**, 165116 (2016)] or its arXiv version.
- Due to the similarity with the one-site DMRG, it requires the input of Hamiltonian as the matrix product operator (MPO). Again, one may copy-and-paste the lines from the tutorial on the one-site DMRG.
- The `TDVP_1site_Ex.m` contains subfunctions `TDVP_1site_expHA` and `TDVP_1site_expHC` for the Lanczos method for local time evolution. Complete these subfunctions as well, by adapting from `DMRG/eigs_1site`.
- This is one-site update method, so the discarded weights are trivially zero. Of course, it does **not** mean that it does not suffer with entanglement growth with time. Indeed, it is worse, since one cannot dynamically change the bond dimension in response to the entanglement growth. This problem can be remedied by considering the two-site TDVP. The above mentioned paper by J. Haegeman *et al.* also describes how to implement the two-site TDVP method.
- Moreover, since it is one-site update method, **the initial MPS should have the bond dimension of `Nkeep`, and the TDVP function should not truncate the bond space.**

After completing the `TDVP_1site_Ex.m`, run the function for the XY spin chain as demonstrated above.