

Finite-temperature MPS methods for the 1D XY model

Author: Jheng-Wei Li

We demonstrate two MPS-based approaches: (1) purification and (2) exponential tensor renormalization group (XTRG), for simulating the one-dimensional XY model at finite temperatures,

$$H_{XY} = \sum_i S_i^x S_{i+1}^x + S_i^y S_{i+1}^y.$$

To approximate the thermal density matrix $\rho = e^{-\beta H}$ with $\beta = \frac{1}{T}$ being the inverse temperature, the first approach uses a *linear* discretization scheme ($\beta_N = \sum_1^N \beta$) to cool down the system from the infinite temperature limit via the imaginary time evolution, while the second approach use a *logarithmic* discretization scheme ($\beta_N = 2^N \beta$) starting from a very high temperature and exploiting the fact that $\rho(2\beta) = \rho(\beta) \times \rho(\beta)$. For comparison, we set the final temperature $\beta_N = 20$ and $N = 40$.

Exercise (a): Purification

Complete the function `ITE_ex.m` which is zipped together with this tutorial material.

Here we test on the XY model with $N_{\text{site}} = 8$ and the bond dimension $D = 50$:

```
clear all;
nsite = 8; % # of sites
niter = 40; % # of iterations
dkeep = 50; % the bond dimension
betaN = 20; % the final inverse temperature
beta0 = betaN/niter; % linear discretization

% generate the 2-site Trotter gate for e^(-0.5*beta0*H_{XY})
% index ordering: (lower left)-(upper left)-(lower right)-(upper right)
[S,I] = getLocalSpace('Spin',1/2);
ham = contract(S(:,1:2,:),3,2,permute(conj(S(:,1:2,:)),[3 2 1]),3,2);
[sH1,sH2,sH3,sH4] = size(ham);
gate = expm(-(beta0/2)*reshape(permute(ham,[1,3,2,4]),[sH1*sH3,sH2*sH4]));
gate = permute(reshape(gate,[sH1,sH3,sH2,sH4]),[1,3,2,4]);

% initialize the infinite temperature MPO
% index ordering: left-down-up-right
% ("down" is the physical d.o.f.)
% ("up" is the auxiliary d.o.f.)
mpo = cell(nsite,1);
mpo(:) = {permute(I,[3,1,2])};
% cast into a canonical MPS to truncate
mps = MPO2MPS(mpo);
[mps,fac1] = canonForm(mps,nsite);
[mps,fac2] = canonForm(mps,0,dkeep);
```

```
% cast the MPS back to a MPO
mpo = MPS2MPO(mps);
```

Here we perform the imaginary time evolution and compute the partition function ($Z = \text{Tr}(e^{-\beta_m})$) and its relative error at each iteration m with the inverse temperature $\beta_m = \frac{m}{N}\beta_N$:

```
fac = log(fac1) + log(fac2);
for iter = 1 : niter
    time = tic;
    [mpo, fac1] = ITE_ex(mpo, gate, dkeep); % the imaginary time evolution
    % compute lnZ
    fac = fac + log(fac1);
    lnZ = 2*fac + log(TRACE(mpo));
    beta = iter*beta0;
    lnZ_ex = XY_lnZ(nsites, beta); % the exact solution
    err = abs(1 - lnZ/lnZ_ex);
    fprintf('Iter: %2d, beta= %10.3e, lnZ= %13.8f, err= %10.3e >>> %9.3f sec \n', ...
        iter, beta, lnZ, err, toc(time));
end
```

```
Iter:  1, beta=  5.000e-01, lnZ=    5.65426879, err=  8.486e-05 >>>  0.090 sec
Iter:  2, beta=  1.000e+00, lnZ=    5.97272792, err=  3.060e-04 >>>  0.024 sec
Iter:  3, beta=  1.500e+00, lnZ=    6.47705272, err=  5.888e-04 >>>  0.016 sec
Iter:  4, beta=  2.000e+00, lnZ=    7.13591602, err=  8.644e-04 >>>  0.016 sec
Iter:  5, beta=  2.500e+00, lnZ=    7.91708190, err=  1.094e-03 >>>  0.024 sec
Iter:  6, beta=  3.000e+00, lnZ=    8.79187857, err=  1.267e-03 >>>  0.015 sec
Iter:  7, beta=  3.500e+00, lnZ=    9.73698288, err=  1.387e-03 >>>  0.013 sec
Iter:  8, beta=  4.000e+00, lnZ=   10.73443138, err=  1.465e-03 >>>  0.016 sec
Iter:  9, beta=  4.500e+00, lnZ=   11.77082556, err=  1.512e-03 >>>  0.015 sec
Iter: 10, beta=  5.000e+00, lnZ=   12.83634291, err=  1.536e-03 >>>  0.015 sec
Iter: 11, beta=  5.500e+00, lnZ=   13.92383634, err=  1.543e-03 >>>  0.015 sec
Iter: 12, beta=  6.000e+00, lnZ=   15.02811113, err=  1.540e-03 >>>  0.015 sec
Iter: 13, beta=  6.500e+00, lnZ=   16.14537940, err=  1.529e-03 >>>  0.015 sec
Iter: 14, beta=  7.000e+00, lnZ=   17.27286188, err=  1.514e-03 >>>  0.014 sec
Iter: 15, beta=  7.500e+00, lnZ=   18.40850222, err=  1.496e-03 >>>  0.016 sec
Iter: 16, beta=  8.000e+00, lnZ=   19.55076352, err=  1.476e-03 >>>  0.014 sec
Iter: 17, beta=  8.500e+00, lnZ=   20.69848394, err=  1.455e-03 >>>  0.014 sec
Iter: 18, beta=  9.000e+00, lnZ=   21.85077402, err=  1.433e-03 >>>  0.013 sec
Iter: 19, beta=  9.500e+00, lnZ=   23.00694353, err=  1.412e-03 >>>  0.016 sec
Iter: 20, beta=  1.000e+01, lnZ=   24.16644910, err=  1.392e-03 >>>  0.014 sec
Iter: 21, beta=  1.050e+01, lnZ=   25.32885661, err=  1.371e-03 >>>  0.015 sec
Iter: 22, beta=  1.100e+01, lnZ=   26.49381393, err=  1.352e-03 >>>  0.019 sec
Iter: 23, beta=  1.150e+01, lnZ=   27.66103115, err=  1.333e-03 >>>  0.016 sec
Iter: 24, beta=  1.200e+01, lnZ=   28.83026606, err=  1.315e-03 >>>  0.015 sec
Iter: 25, beta=  1.250e+01, lnZ=   30.00131352, err=  1.298e-03 >>>  0.014 sec
Iter: 26, beta=  1.300e+01, lnZ=   31.17399753, err=  1.281e-03 >>>  0.015 sec
Iter: 27, beta=  1.350e+01, lnZ=   32.34816532, err=  1.265e-03 >>>  0.016 sec
Iter: 28, beta=  1.400e+01, lnZ=   33.52368290, err=  1.250e-03 >>>  0.015 sec
Iter: 29, beta=  1.450e+01, lnZ=   34.70043169, err=  1.235e-03 >>>  0.016 sec
Iter: 30, beta=  1.500e+01, lnZ=   35.87830591, err=  1.221e-03 >>>  0.014 sec
Iter: 31, beta=  1.550e+01, lnZ=   37.05721059, err=  1.208e-03 >>>  0.016 sec
Iter: 32, beta=  1.600e+01, lnZ=   38.23705998, err=  1.195e-03 >>>  0.015 sec
Iter: 33, beta=  1.650e+01, lnZ=   39.41777634, err=  1.183e-03 >>>  0.015 sec
Iter: 34, beta=  1.700e+01, lnZ=   40.59928889, err=  1.171e-03 >>>  0.015 sec
Iter: 35, beta=  1.750e+01, lnZ=   41.78153304, err=  1.160e-03 >>>  0.013 sec
Iter: 36, beta=  1.800e+01, lnZ=   42.96444966, err=  1.149e-03 >>>  0.025 sec
Iter: 37, beta=  1.850e+01, lnZ=   44.14798459, err=  1.139e-03 >>>  0.013 sec
Iter: 38, beta=  1.900e+01, lnZ=   45.33208810, err=  1.129e-03 >>>  0.015 sec
```

```

Iter: 39, beta= 1.950e+01, lnZ= 46.51671449, err= 1.120e-03 >>> 0.016 sec
Iter: 40, beta= 2.000e+01, lnZ= 47.70182174, err= 1.111e-03 >>> 0.015 sec

```

Exercise (b): XTRG

Complete the function `MPOxMPO_ex.m` which is zipped together with this tutorial material.

```

clear all;
nsite = 8; % # of sites
niter = 40; % # of iterations
dkeep = 50; % the bond dimension
betaN = 20; % the final inverse temperature
beta0 = betaN/2^niter; % logarithmic discretization

% generate the initial MPO with the 1st-order Trotter decomposition at beta0
% index ordering: left-down-up-right
[S,I] = getLocalSpace('Spin',1/2);
ham = contract(S(:,1:2,:),3,2,permute(conj(S(:,1:2,:)),[3 2 1]),3,2);
mpo = initMPO(ham, beta0, nsite);

```

Here we perform the XTRG and compute the partition function ($Z = \text{Tr}(e^{-\beta_m})$) and its relative error at each iteration m with the inverse temperature $\beta_m = 2^{m-N}\beta_N$:

```

lnZ = 0;
fac = 0;
for iter = 1 : niter
    time = tic;
    % compute lnZ
    lnZ = 2*fac + log(TRACE(mpo));
    beta = beta0*2^iter;
    lnZ_ex = XY_lnZ(nsite,beta); % the exact solution
    err = abs(1-lnZ/lnZ_ex);

    % XTRG
    if iter~=niter
        mpo = MPOxMPO_ex(mpo); % double the MPO
        % cast into a canonical MPS to truncate
        mps = MPO2MPS(mpo);
        [mps,fac2] = canonForm(mps,nsite);
        [mps,fac1] = canonForm(mps,0,dkeep);
        % cast the MPS back to a MPO
        mpo = MPS2MPO(mps);
        fac = 2*fac + log(fac2) + log(fac1);
    end
    fprintf('Iter: %2d, beta= %10.3e, lnZ= %13.8f, err= %10.3e >>> %9.3f sec \n', ...
        iter, beta, lnZ, err, toc(time));
end

```

```

Iter: 1, beta= 3.638e-11, lnZ= 5.54517744, err= 4.441e-16 >>> 0.061 sec
Iter: 2, beta= 7.276e-11, lnZ= 5.54517744, err= 1.443e-15 >>> 0.008 sec
Iter: 3, beta= 1.455e-10, lnZ= 5.54517744, err= 2.776e-15 >>> 0.007 sec
Iter: 4, beta= 2.910e-10, lnZ= 5.54517744, err= 5.107e-15 >>> 0.003 sec
Iter: 5, beta= 5.821e-10, lnZ= 5.54517744, err= 1.077e-14 >>> 0.011 sec
Iter: 6, beta= 1.164e-09, lnZ= 5.54517744, err= 2.143e-14 >>> 0.003 sec

```

Iter: 7,	beta= 2.328e-09,	lnZ= 5.54517744,	err= 4.297e-14 >>>	0.002 sec
Iter: 8,	beta= 4.657e-09,	lnZ= 5.54517744,	err= 8.549e-14 >>>	0.003 sec
Iter: 9,	beta= 9.313e-09,	lnZ= 5.54517744,	err= 1.715e-13 >>>	0.004 sec
Iter: 10,	beta= 1.863e-08,	lnZ= 5.54517744,	err= 3.433e-13 >>>	0.002 sec
Iter: 11,	beta= 3.725e-08,	lnZ= 5.54517744,	err= 6.865e-13 >>>	0.002 sec
Iter: 12,	beta= 7.451e-08,	lnZ= 5.54517744,	err= 1.373e-12 >>>	0.002 sec
Iter: 13,	beta= 1.490e-07,	lnZ= 5.54517744,	err= 2.747e-12 >>>	0.002 sec
Iter: 14,	beta= 2.980e-07,	lnZ= 5.54517744,	err= 5.494e-12 >>>	0.002 sec
Iter: 15,	beta= 5.960e-07,	lnZ= 5.54517744,	err= 1.100e-11 >>>	0.002 sec
Iter: 16,	beta= 1.192e-06,	lnZ= 5.54517744,	err= 2.201e-11 >>>	0.003 sec
Iter: 17,	beta= 2.384e-06,	lnZ= 5.54517744,	err= 4.403e-11 >>>	0.002 sec
Iter: 18,	beta= 4.768e-06,	lnZ= 5.54517744,	err= 8.806e-11 >>>	0.002 sec
Iter: 19,	beta= 9.537e-06,	lnZ= 5.54517744,	err= 1.761e-10 >>>	0.002 sec
Iter: 20,	beta= 1.907e-05,	lnZ= 5.54517744,	err= 3.522e-10 >>>	0.003 sec
Iter: 21,	beta= 3.815e-05,	lnZ= 5.54517744,	err= 7.045e-10 >>>	0.002 sec
Iter: 22,	beta= 7.629e-05,	lnZ= 5.54517744,	err= 1.409e-09 >>>	0.004 sec
Iter: 23,	beta= 1.526e-04,	lnZ= 5.54517744,	err= 2.818e-09 >>>	0.004 sec
Iter: 24,	beta= 3.052e-04,	lnZ= 5.54517745,	err= 5.636e-09 >>>	0.003 sec
Iter: 25,	beta= 6.104e-04,	lnZ= 5.54517754,	err= 1.127e-08 >>>	0.004 sec
Iter: 26,	beta= 1.221e-03,	lnZ= 5.54517797,	err= 2.254e-08 >>>	0.004 sec
Iter: 27,	beta= 2.441e-03,	lnZ= 5.54517980,	err= 4.509e-08 >>>	0.013 sec
Iter: 28,	beta= 4.883e-03,	lnZ= 5.54518738,	err= 9.019e-08 >>>	0.007 sec
Iter: 29,	beta= 9.766e-03,	lnZ= 5.54521817,	err= 1.804e-07 >>>	0.011 sec
Iter: 30,	beta= 1.953e-02,	lnZ= 5.54534233,	err= 3.610e-07 >>>	0.014 sec
Iter: 31,	beta= 3.906e-02,	lnZ= 5.54584098,	err= 7.228e-07 >>>	0.026 sec
Iter: 32,	beta= 7.812e-02,	lnZ= 5.54783924,	err= 1.448e-06 >>>	0.043 sec
Iter: 33,	beta= 1.562e-01,	lnZ= 5.55583510,	err= 2.904e-06 >>>	0.073 sec
Iter: 34,	beta= 3.125e-01,	lnZ= 5.58775213,	err= 5.822e-06 >>>	0.333 sec
Iter: 35,	beta= 6.250e-01,	lnZ= 5.71415888,	err= 1.158e-05 >>>	0.688 sec
Iter: 36,	beta= 1.250e+00,	lnZ= 6.20062798,	err= 2.202e-05 >>>	1.004 sec
Iter: 37,	beta= 2.500e+00,	lnZ= 7.90814236,	err= 3.622e-05 >>>	1.479 sec
Iter: 38,	beta= 5.000e+00,	lnZ= 12.81606528,	err= 4.662e-05 >>>	1.698 sec
Iter: 39,	beta= 1.000e+01,	lnZ= 24.13164730,	err= 5.046e-05 >>>	1.568 sec
Iter: 40,	beta= 2.000e+01,	lnZ= 47.64644734,	err= 5.153e-05 >>>	0.002 sec