[Tutorial T07b] QSpace: DMRG for ground-state search

Zoom session, 02.06.2020

Author: Seung-Sup Lee (this document), Jheng-Wei Li (codes)

Date: 27.05.2020

Note that this part of tutorial (T07b) is optional. Please keep in mind that, however, some of the final exam topics would be the implementation of some tensor network methods by using the QSpace library. For those who will work on such exam topics, this exercise will be a good practice!

Here we will study the single-site and two-site DMRG methods for ground-state search, which exploit the symmetries in the system. For this, we use the QSpace library. Thus we distribute the materials for T07b, both for exercise and solution, only via my exchange directory within the ASC cluster system, since the QSpace is not public yet.

There is a new sub-directory (`/home/s/S.Lee/X/QSpace_TNcourse/JhengWei.Li`) under my exchange directory. It contains the functions that are used for the one-site and two-site DMRG, written by Jheng-Wei Li (JW). JW's implementation of DMRG for T07b differs from Seung-Sup (SS)'s for T07a, at a few points.

- JW's uses QSpace while SS's uses MATLAB only. So the former respects the symmetry in a strict sense, which leads to more crucial difference between the one-site and two-site DMRG methods.
- JW's initializes the MPS with random numbers, while the default of SS's is to initialize with the result of the iterative diagonalization. Of course, one can implement the initialization with the iterative diagonalization result, for the QSpace codes.
- JW's uses the restarted Lanczos method to update the ket tensor, while SS's uses the basic Lanczos method.

Random number stream

Here we use random MPS as initial guess. MATLAB uses the same random number stream for the same session. So if you run the same tutorial material directly after starting the MATLAB session, you will get the same result. To reset the random number seed with the current time, one may use the following:

```
RandStream.setGlobalStream( ...
    RandStream('mt19937ar','Seed','shuffle'));
```

Note that the MATLAB documentation for `RandStream` reads: "*It is usually not desirable to (reset random number seed) more than once per MATLAB session as it may affect the statistical properties of the random numbers MATLAB produces.*"

One-site DMRG with QSpace

We use the one-site DMRG code (`JhengWei.Li/DMRG_1site.m`) to obtain the ground state of the spin-1/2 Heisenberg chain. The Hamiltonian of the model is

$$H = J \sum_{\ell=1}^{N-1} \hat{\vec{S}}_{\ell} \cdot \hat{\vec{S}}_{\ell+1}.$$

We can set $J = 1$ without loss of generality. (Note that this model has been studied with the iterative diagonalization in T04 and T06.)

Create the basic QSpace objects, and set the parameters for DMRG.

```
clear
%% Basic Operators
[ss, info]= getLocalSpace('Spin' ,1/2, '-v');
```

```
getL'>getLocal':1191 18:54:18   *  got { E } ops
getL'>getLocal':1238 18:54:18   *  1 S-op
```

```
eid = info.E;
inv = QCREATE(eid, '1', 'IN');
vac = getvac(eid, [1]);
ss_T = QSpace(permuteQS(ss, [3,2,1], 'conj'));

%% DMRG parameters
NSITE = 16;
DKEEP = 16;
ECONV = 1E-6;
SWEEP = 50;
```

Construct the MPO Hamiltonian.

```
% % Build MPO
% generate a MPO of form [I 0 0]
%                        [S 0 0]
%                        [0 S I]
%                      [3]
%                       |
% index ordering: [1] ---  [4]
%                       |
%
%                      [2]
z = realmin;
mat2 = [1, z, z;
        1, z, z;
        z, 1, 1;];
LL = GetMPO(mat2(end,:), ss, ss_T, [], []);
RR = GetMPO(mat2(:,1), ss, ss_T, [], []);
CC = GetMPO(mat2, ss, ss_T, [], []);

MPO = QSpace(NSITE,1);
MPO(1) = LL;
MPO(end) = RR;
if (NSITE>2)
 for isite = 2:NSITE-1
```

```
    MPO(isite) = CC;
  end
end
```

Initialize the MPS. Since the MPS consists of QSpace objects that have symmetry information, we generate the isometries by using `getIdentityQS` successively. JW's `QCREATE` routine is a wrap-up for `getIdentityQS`.

```
% % Init MPS
% [1] --- [3]
%       |
%      [2]
MPS = QSpace(NSITE, 1);
MPS(1) = QCREATE(vac, '1', eid, '1', 'OUT');
for isite = 2 : ceil(NSITE/2)
  tmp = QCREATE(MPS(isite-1), '3', eid, '1', 'OUT');
  [MPS(isite), ~, ~] = QSVD(tmp, [1,2], -1, DKEEP);
end
```

```
./mpsortho.cc:767    18:54:18  WRN readjusting norm by factor 1.109
./mpsortho.cc:767    18:54:18  WRN readjusting norm by factor 1.472
./mpsortho.cc:767    18:54:18  WRN readjusting norm by factor 1.386
```

```
for isite = 1 : floor(NSITE/2)
  MPS(NSITE-isite+1) = contract(conj(MPS(isite)), [2], inv,[2], [2,3,1]);
end
```

Then replace the reduced matrix tensors, i.e., `.data` of the QSpace objects, with random numbers.

```
% Randomize
for isite = 1 : NSITE
  MPS(isite) = QRAND(MPS(isite), -1.0, 1.0);
end
```

Bring the MPS into left-canonical form.

```
[MPS, ~] = CANONICAL(MPS, [], 0, NSITE, -1, DKEEP); %% left normalize
L_eff = QCREATE(MPS(1), '1', MPO(1), '1*', 'OUT');
R_eff = conj(L_eff);
```

Run the DMRG sweeps to find the ground state.

```
[MPS, ene] = DMRG_1site(MPS, L_eff, R_eff, MPO, DKEEP, ECONV, SWEEP);
```

```
*** SWEEP:   1, R2L @  16; E =     -4.0788650433741065 >>>       0.01 sec
*** SWEEP:   1, R2L @  15; E =     -4.7851137361480740 >>>       0.03 sec
*** SWEEP:   1, R2L @  14; E =     -5.0036562480468278 >>>       0.08 sec
*** SWEEP:   1, R2L @  13; E =     -5.2734774528875628 >>>       0.36 sec
*** SWEEP:   1, R2L @  12; E =     -5.8068508635850229 >>>       1.02 sec
*** SWEEP:   1, R2L @  11; E =     -6.7255663118731448 >>>       1.05 sec
*** SWEEP:   1, R2L @  10; E =     -6.7439734261255202 >>>       0.82 sec
*** SWEEP:   1, R2L @   9; E =     -6.7439734261253523 >>>       0.02 sec
*** SWEEP:   1, R2L @   8; E =     -6.7478211421144092 >>>       0.28 sec
*** SWEEP:   1, R2L @   7; E =     -6.7478211421143328 >>>       0.02 sec
*** SWEEP:   1, R2L @   6; E =     -6.7498651973758124 >>>       0.73 sec
```

```
*** SWEEP:   1, R2L @   5; E =     -6.7498651973758088 >>>       0.03 sec
*** SWEEP:   1, R2L @   4; E =     -6.7498651973758079 >>>       0.02 sec
*** SWEEP:   1, R2L @   3; E =     -6.7498651973758044 >>>       0.02 sec
*** SWEEP:   1, R2L @   2; E =     -6.7498651973758044 >>>       0.02 sec
*** SWEEP:   1, L2R @   1; E =     -6.7498651973758044 >>>       0.01 sec
*** SWEEP:   1, L2R @   2; E =     -6.7498651973758017 >>>       0.01 sec
*** SWEEP:   1, L2R @   3; E =     -6.7498651973758070 >>>       0.02 sec
*** SWEEP:   1, L2R @   4; E =     -6.7498651973758070 >>>       0.02 sec
*** SWEEP:   1, L2R @   5; E =     -6.7498651973757999 >>>       0.02 sec
*** SWEEP:   1, L2R @   6; E =     -6.7498651973758026 >>>       0.02 sec
*** SWEEP:   1, L2R @   7; E =     -6.7498651973758008 >>>       0.02 sec
*** SWEEP:   1, L2R @   8; E =     -6.7498651973758026 >>>       0.02 sec
*** SWEEP:   1, L2R @   9; E =     -6.7498651973758026 >>>       0.02 sec
*** SWEEP:   1, L2R @  10; E =     -6.7498651973758026 >>>       0.02 sec
*** SWEEP:   1, L2R @  11; E =     -6.7498651973754544 >>>       0.26 sec
*** SWEEP:   1, L2R @  12; E =     -6.7498651973758221 >>>       0.02 sec
*** SWEEP:   1, L2R @  13; E =     -6.7498651973758212 >>>       0.02 sec
*** SWEEP:   1, L2R @  14; E =     -6.7498651973758195 >>>       0.02 sec
*** SWEEP:   1, L2R @  15; E =     -6.7498651973758212 >>>       0.01 sec
*** SWEEP:   2, R2L @  16; E =     -6.7498651973758177 >>>       0.01 sec
*** SWEEP:   2, R2L @  15; E =     -6.7498651973758221 >>>       0.01 sec
*** SWEEP:   2, R2L @  14; E =     -6.7498651973758195 >>>       0.02 sec
*** SWEEP:   2, R2L @  13; E =     -6.7498651973758230 >>>       0.02 sec
*** SWEEP:   2, R2L @  12; E =     -6.7498651973758257 >>>       0.02 sec
*** SWEEP:   2, R2L @  11; E =     -6.7498651973758266 >>>       0.02 sec
*** SWEEP:   2, R2L @  10; E =     -6.7498651973758257 >>>       0.02 sec
*** SWEEP:   2, R2L @   9; E =     -6.7498651973758221 >>>       0.02 sec
*** SWEEP:   2, R2L @   8; E =     -6.7498651973758230 >>>       0.02 sec
*** SWEEP:   2, R2L @   7; E =     -6.7498651973758186 >>>       0.02 sec
*** SWEEP:   2, R2L @   6; E =     -6.7498651973758221 >>>       0.02 sec
*** SWEEP:   2, R2L @   5; E =     -6.7498651973758230 >>>       0.02 sec
*** SWEEP:   2, R2L @   4; E =     -6.7498651973758257 >>>       0.02 sec
*** SWEEP:   2, R2L @   3; E =     -6.7498651973758230 >>>       0.02 sec
*** SWEEP:   2, R2L @   2; E =     -6.7498651973758266 >>>       0.02 sec
*** SWEEP:   2, L2R @   1; E =     -6.7498651973758257 >>>       0.01 sec
*** SWEEP:   2, L2R @   2; E =     -6.7498651973758257 >>>       0.02 sec
*** SWEEP:   2, L2R @   3; E =     -6.7498651973758221 >>>       0.02 sec
*** SWEEP:   2, L2R @   4; E =     -6.7498651973758239 >>>       0.02 sec
*** SWEEP:   2, L2R @   5; E =     -6.7498651973758239 >>>       0.02 sec
*** SWEEP:   2, L2R @   6; E =     -6.7498651973758204 >>>       0.02 sec
*** SWEEP:   2, L2R @   7; E =     -6.7498651973758248 >>>       0.02 sec
*** SWEEP:   2, L2R @   8; E =     -6.7498651973758212 >>>       0.02 sec
*** SWEEP:   2, L2R @   9; E =     -6.7498651973758212 >>>       0.02 sec
*** SWEEP:   2, L2R @  10; E =     -6.7498651973758248 >>>       0.02 sec
*** SWEEP:   2, L2R @  11; E =     -6.7498651973758275 >>>       0.02 sec
*** SWEEP:   2, L2R @  12; E =     -6.7498651973758284 >>>       0.02 sec
*** SWEEP:   2, L2R @  13; E =     -6.7498651973758266 >>>       0.02 sec
*** SWEEP:   2, L2R @  14; E =     -6.7498651973758221 >>>       0.02 sec
*** SWEEP:   2, L2R @  15; E =     -6.7498651973758239 >>>       0.02 sec
****** DMRG CONVERGED ******                                        5.86 sec
```

```
fprintf('E = %23.16f \n\n', ene);
```

```
E =     -6.7498651973758239
```

DMRG is converged, but the result is not good. We can compare the DMRG result of the ground-state energy with the Bethe Ansatz result, which is exact. Below is the list of the Bethe Ansatz solutions of the ground-state energy for finite chain length N:

N	Ground-state energy
16	-6.9117371455749
24	-10.4537857604096
32	-13.9973156182243
48	-21.0859563143863
64	-28.1754248597421

(Source: https://dmrg101-tutorial.readthedocs.io/en/latest/infinite_heisenberg.html)

The error here is about 3%, which is not that different from the error of the pure iterative diagonalization result.

The exploitation of the symmetry improves the accuracy and performance for most of cases, but here it is the opposite. The single-site DMRG with symmetry typically gets stuck to local minima, while the one without symmetry is less susceptible! This issue of the single-site DMRG can be overcome by using the two-site update. The detailed explanation of this will be given during lectures.

Exercise: Two-site DMRG by using QSpace

Implement the two-site DMRG by using QSpace. Compute the ground-state energies of the spin-1/2 Heisenberg chains, and compare the DMRG results with the Bethe Ansatz solutions.