Exercise 1.2 solution

Decomposing a state into an MPS Computing wave function from an MPS

Contents


```
% Do not forget to set the path to the minclude folder, e.g.:
%addpath ../minclude

function ex1_2(L)
```

parameters:

```
if nargin==0
    L = 16; %default value
end
```

first get the ground state of the Heisenberg model:

```
H2 = getHeisenberg(1);
HL = getHL(H2,L);
% compute ground state:
[V,e0] = eigs(HL,1,'SA');
TV = reshape(V,ones(1,L)*2);
```

Test creation of an MPS without compression

```
maxD = inf; %no truncation
mps = getMPS(TV, maxD);
vstate = getstate_fromMPS(mps);
% recompute energy:
en = vstate'*HL*vstate / (vstate'*vstate);
disp0('Energy difference:',e0-en);
```

```
Energy difference:8.8818e-16
```

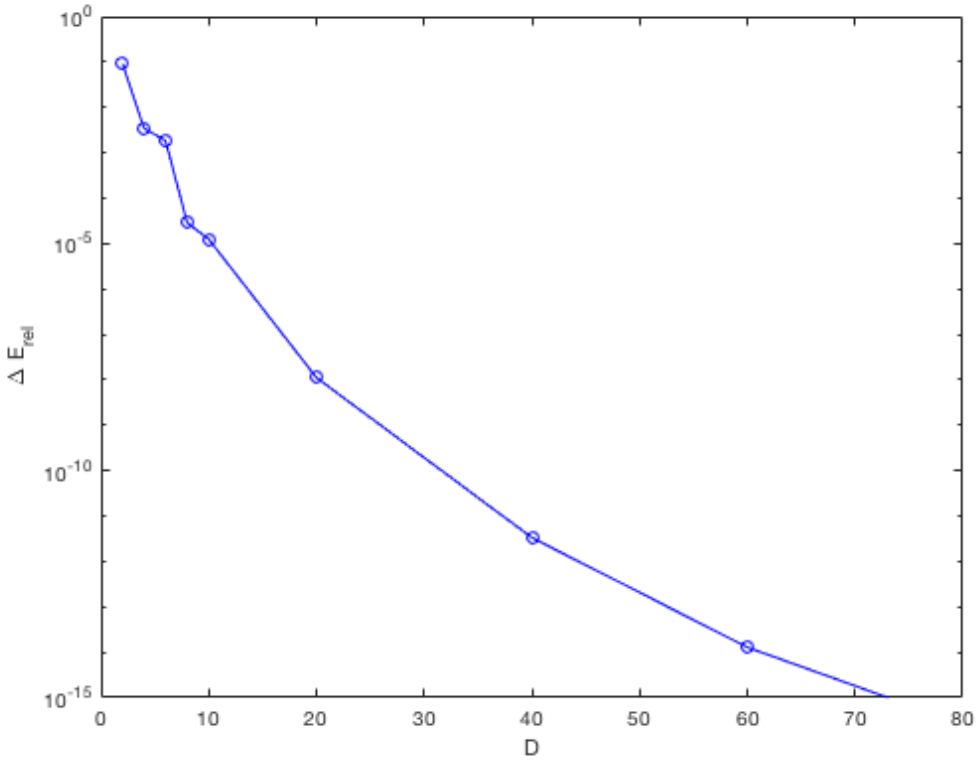
Now part 3

plot the relative error of the energy of the two states as a function of the number of singular values (states) D kept.

```
vD = [2 4 6 8 10 20:20:100];
% store results in:
energies = zeros(numel(vD),1);
```

```
%
for k=1:numel(vD)
    maxD = vD(k); %current number of states to keep
    mps = getMPS(TV, maxD);
    truncstate = getstate_fromMPS(mps);
    % compute energy of truncated state
    energies(k) = truncstate' * HL * truncstate / (truncstate' * truncstate);
end

% plot
figure;
set(gca,'FontSize',15);
semilogy(vD,(energies-e0)/abs(e0),'bo-');
hold on;
ylim([1e-15 1]);
xlabel('D');
ylabel('\Delta E_{rel}')
```




```
end

function mps = getMPS(state, maxD)
    % returns a cell array with MPS inside
    % legs are: 1: physical leg, 2: left auxiliary leg, 3: right auxiliary leg
    L = numel(size(state));
    % mps in a cell
    mps = cell(1,L);

    % do tensorsvd, absorbing singular values on the right in V
    [U,s,V] = tensorsvd(state, [1], [2:L], maxD, 'r');
    mps{1} = U;
    % first leg of U is physical leg, second leg is right auxiliary leg
```

```
    % last leg of V is left auxiliary leg

    for k=2:L-1
        Vnlegs = numel(size(V));    % number of legs of V
        [U,s,V] = tensorsvd(V, [1 Vnlegs], [2:Vnlegs-1], maxD, 'r');
        mps{k} = U;
    end
    mps{L} = V;
end

function vstate = getstate_fromMPS(mps)
    % returns a state by contracting all MPS tensors
    % start on the left:
    state = mps{1};

    for k=2:numel(mps)
        % contract leg k with leg 2 of next MPS tensor
        state = tcontract(state, mps{k}, k, 2);
    end
    % create a vector
    vstate = reshape(state,[numel(state) 1]);
end
```

Published with MATLAB® R2017a