QSpace application: Iterative diagonalization

Author: Seung-Sup Lee

Iterative diagonalization of spin-1/2 Heisenberg chain

We demonstrate how to write an iterative diagonalization code by using the QSpace library to exploit a non-Abelian symmetry. For this, we revisit a familiar example of the spin-1/2 Heisenberg chain. The Hamiltonian for the Heisenberg spin chain of length N is given by:

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

Here we set the coupling strength $J = 1$ without loss of generality.

We first create the QSpace objects that represent operators of a spin site and respect the SU(2) spin symmetry of the system.

```
clear

[S,I] = getLocalSpace('Spin',1/2);
S % spin operator
```

```
S =
     Q:  1x [1 1 1]  having 'SU2',   operator,   { , *, * }
  data:  3-D double (112 B)      1 x 1 x 1 => 2 x 2 x 3

     1.  1x1        | 2x2x3      [ 1 ; 1 ; 2 ]      -1.225
```

```
I.E % identity
```

```
     Q:  1x [1 1]  having 'SU2',   { , * }
  data:  2-D double (112 B)      1 x 1 => 2 x 2

     1.  1x1        | 2x2        [ 1 ; 1 ]           1.  {1.414}
```

```
J = 1; % Heisenberg coupling strength
N = 15; % chain length
Nkeep = 20; % number of multiplets to keep
```

Note that, when non-Abelian symmetries are exploited, we treat **symmetry multiplets**, not bare states. Each multiplet may represent multiple states. For example, $S = 1$ multiplet is a triplet, i.e., represents three states $S_z = -1, 0, +1$.

Define the isometry which applies to the first chain site. As a convention, we index the first site as 00. (The next will be 01, then 02, and so on.)

```
A0 = getIdentity(getvac(I.E),2,I.E,2,[1 3 2]);
A0.info.itags = {'L00','R00*','s00'};
A0
```

```
A0 =
     Q:  1x [1 1 1]  having 'SU2',   A-matrix,   { L00, R00*, s00 }
```

```
  data:  3-D double (112 B)      1 x 1 x 1 => 1 x 2 x 2

     1.  1x1        | 1x2x2      [ 0 ; 1 ; 1 ]       1.414
```

We assign the itags, to specify the Hilbert spaces for which the isometry is defined. The third itag means the local space for the **s**ite **00**. The first and second itags mean that the first and second legs are for the **L**eft and **R**ight bond spaces of the site **00**, respectively.

When one converts a pure MATLAB code to a QSpace code, a key step to do is to assign the itags. The itags are useful in distinguishing the tensors, since the itags unambiguously indicate which spaces are associated with the legs of the tensors. Also, by using the itags, the contraction of multiple tensors can be done in a one-liner.

We define the local Hamiltonian term for the site 00.

```
H0 = I.E*1e-30;
H0.info.itags = {'s00','s00*'};
H0
```

```
H0 =
     Q:  1x [1 1]  having 'SU2',   { s00, s00* }
  data:  2-D double (112 B)      1 x 1 => 2 x 2

     1.  1x1        | 2x2        [ 1 ; 1 ]       1e-30  {1.414}
```

Though the term has only zero elements by definition, we have assigned very small but finite numbers, to span the Hilbert space. It is because that QSpace automatically drops off the sectors with all zeros, to have better computational performance; see the "Introduction to QSpace" tutorial material for detail.

Now we implement the iterative diagonalization.

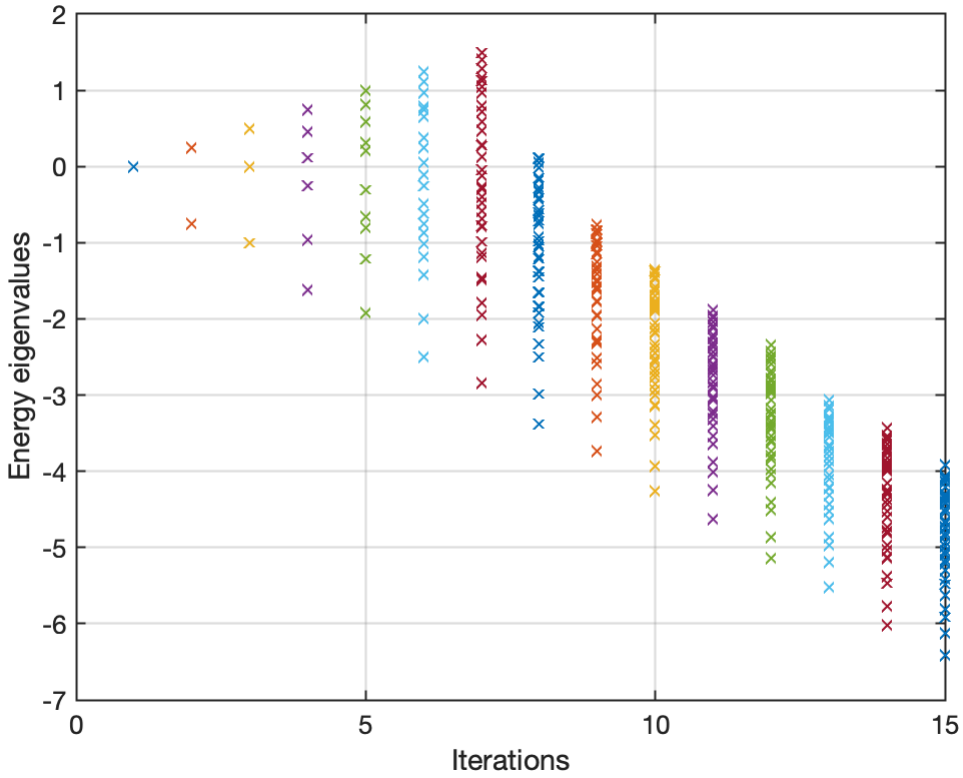
```
Es = cell(N,1); % energy spectrum at each iteration

for itN = (1:N)
    % spin operator at the current site
    Snow = S;
    Snow.info.itags = {['s',sprintf('%02i',itN-1)], ...
        ['s',sprintf('%02i',itN-1),'*'],'op*'};

    if itN == 1
        % isometry for the current iteration
        Anow = A0;
        % Hamiltonian for the current iteration
        Hnow = contract(A0,'!2*',{H0,'!1',A0});
    else
        % add new site
        Anow = getIdentity(Aprev,2,I.E,2,[1 3 2]);
        Anow.info.itags(2:3) = ...
            {['R',sprintf('%02i',itN-1),'*'], ...
            ['s',sprintf('%02i',itN-1)]};

        % update the Hamiltonian up to the last sites
        % to the enlarged Hilbert space
```

```
        Hnow = contract(Anow,'!2*',{Hprev,'!1',Anow});

        % add the coupling term
        Hcoup = contract(Anow,'!2*',{Sprev,'!1',{Snow,'!2*',Anow}});
        Hnow = Hnow + J*Hcoup;
    end

    % diagonalize Hamiltonian; Hermitianization is done within eigQS
    [Eeig,Ieig] = eigQS(Hnow,'Nkeep',Nkeep,'deps',1e-10);
    % convert struct -> QSpace object
    Ieig.EK = QSpace(Ieig.EK);
    Ieig.AK = QSpace(Ieig.AK);

    Ieig.EK.info.itags = {['R',sprintf('%02i',itN-1)], ...
        ['R',sprintf('%02i',itN-1),'*']};
    Ieig.AK.info.itags = Ieig.EK.info.itags;

    Es{itN} = Eeig(:,1);

    % to be used for the next iteration
    Hprev = diag(Ieig.EK); % Hamiltonian
    Aprev = contract(Anow,'!1',Ieig.AK,[1 3 2]); % isometry
    Sprev = contract(Aprev,'!2*',{Snow,'!1',Aprev},[1 3 2]); % spin
end
```

Here we have used `'deps'` option for `eigQS`, to keep or truncate the multiplets, whose energies are lying within a tolerance range, altogether. Note that there is a typo in the documentation of `eigQS`; it should be `'deps'`, **not** `'eps'`.

We visualize how the energy eigenvalues distribute, for different iterations.

```
figure;
hold on;
for itN = (1:N)
    plot(itN+zeros(numel(Es{itN}),1),Es{itN}(:), ...
        'LineStyle','none','Marker','x','LineWidth',1);
end
hold off;
set(gca,'LineWidth',1,'Box','on','FontSize',13);
grid on;
xlabel('Iterations');
ylabel('Energy eigenvalues');
```



We observe that the spectra are denser at higher energies.

Here comes a teaser for a later topic. When one truncates the Hilbert space along the iterative diagonalization, it is better to put the truncation threshold at a larger separation between the energy eigenvalues. That is, the energy of the highest kept state and that of the lowest discarded state should be separated enough. One can think of its justification in a sense of the second-order perturbation theory. This way of choosing the truncation threshold leads to more accurate result, and will be used for the numerical renormalization group (NRG). For now, however, we don't use it for brevity.

Exercise (a): Non-interacting spin-1/2 fermions on a chain

Here we repeat the analysis done in the previous tutorial on iterative diagonalization of fermionic chains.

Consider a tight-binding chain of spin-1/2 fermions of length N. The particles are not interacting. Its Hamiltonian is given by

$$\hat{H} = -t \sum_{s=\uparrow,\downarrow} \sum_{\ell=1}^{N-1} \left(\hat{c}^\dagger_{\ell+1,s} \hat{c}_{\ell,s} + \hat{c}^\dagger_{\ell,s} \hat{c}_{\ell+1,s} \right),$$

where $\hat{c}_{\ell,s}$ annihilates a particle of spin $s = \uparrow, \downarrow$ at site ℓ. We set the hopping amplitude $t = 1$.

Compute the ground-state energy of a chain of length $N = 30$ by using the iterative diagonalization. Use two different symmetry settings: (i) `Acharge,Aspin`, i.e., U(1) charge and U(1) spin, (ii) `Acharge,SU2spin`, i.e., U(1) charge and SU(2) spin. **Perform the error analysis** for the iterative diagonalization results, in

comparison with the semi-analytic result. Anaylze how the error depends on the **bond dimension** `Nkeep` and the **symmetries** used.