⇝ Diagonalize T to find lowest eigenvalue E_0 and eigenvector $\vec{y}_0$.

not fully stored!

⇝ In order to find $|E_0\rangle = Q\vec{y}_0$, repeat Lanczos-iteration with one additional D-vector $|n\rangle$, initialized as $|n\rangle = y_0[0]\,|\psi_i\rangle$ and update in step j as $|n\rangle \to |n\rangle + y_0[j]\,|q_j\rangle$

at the end of the while-loop ($|q_j\rangle = \frac{q}{\beta[j]}$ at step j).

At the final value of j: $|n\rangle = Q\vec{y}_0 = |E_0\rangle$.

- Computational cost of Lanczos algorithm

↳ Runtime: $O(\dim(\mathcal{H}) \cdot \log(\dim\mathcal{H}) \cdot n)$

dim of Krylov space: $O(1)$

↳ Memory consumption:

2 real 32 bit floats

3 complex 64-bit doubles ($3 \times 2 \times 8$)

$2-3$ $\dim \mathcal{H}$-vectors $\simeq (8-48) \cdot \dim \mathcal{H}$ Bytes

$\rightsquigarrow \dim \mathcal{H} = 2^N \approx 10^9$ for $N = 30$ sites doable on modern desktop computer/laptop with 8–64 GB RAM.

$\rightsquigarrow$ Runtime for one Lanczos iteration $\mathcal{O}(1-10$ minutes) at $\dim \mathcal{H} = 10^9$.

- Numerical issues with Lanczos algorithm:

↳ $\beta[j]$ gets very small for $j \geq 50-100 \rightarrow$ loss of orthogonality between Krylov basis vectors $|q_j\rangle$. This problem is intrinsic to the algorithm as it comes hand in hand with the invariance of $K_n(H, |\psi_i\rangle)$.

↳ Precision of eigenvalues typically higher than of eigenstates.

↳ Possible remedies: (i) Re-orthogonalize during iteration (costly as more vectors need to be stored)

(ii) Quick and dirty: Ignore loss of orthogonality and keep iterating to get better estimate for $|E_0\rangle$. Works in many cases (and can be verified by checking $\frac{\langle E_0|H|E_0\rangle}{\langle E_0|E_0\rangle} \overset{?}{\simeq} E_0$)

However, a <u>caveat</u> is that spurious additional ground states (ghost states) may occur as an artefact.

↳ See, e.g. R. M. Noack and S. R. Manmana, arXiv: cond-mat 0510321

for a detailed discussion (and references) on the limitations of Lanczos algorithm and extensions, such as Arnoldi algorithm for non-Hermitian matrices.

B.3: Including Symmetries

- Basic idea: Symmetries manifest in operators that commute with the Hamiltonian.

  ⇝ Diagonalizing these operators (decomposing their representation on the Hilbert space into a direct sum of irreducible representations) leads to a block-diagonal form of the Hamiltonian, where each block corresponds to a fixed set of quantum numbers associated with the symmetry

$$H = \begin{pmatrix} \left(H^{(1)}\right) & 0 & \cdots & 0 \\ 0 & \left(H^{(2)}\right) & & \\ \vdots & & \ddots & \\ 0 & & & \end{pmatrix}$$

$\rightsquigarrow$ ED problem can be decomposed and only the blocks of interest need to be considered one by one.

- Key issues:- Basis states $|\rangle$ respecting the symmetries need to be found and grouped into the aforementioned blocks.

- Since the action of operators is implemented more efficiently on unrestricted states I, an efficient mapping $I \hookrightarrow J$ needs to be constructed (lookup tables).

$\rightsquigarrow$ Schematic procedure when applying block $H^{(i)}$ to state:

(a) Loop on states J in block $H^{(i)}$ {

(b) Retrieve representative $\hat{I}(J)$ of this state in occupation number basis

(c) Implement action of H with bitwise operations

(d) Translate resulting I' back to J-basis ($\hat{J}(I')$)

[Optional:
(e) Store matrix-element $\langle \hat{J}[I'] | H^{(i)} | J \rangle$ to J sparse matrix representation of $H^{(i)}$]

}

- Primary example : Total S_z conservation in spin systems (equivalent to partical number conservation in fermionic language)

↳ $S_z^{tot} = \sum_{j=1}^{N} S_z^{(j)}$ conservation for spin $\frac{1}{2}$ systems simply amounts to a fixed number of 1-bits in basis integers $I_6 = \sum_{j=1}^{N} 6_j \, 2^{N-j}$

⇝ Size of block with $S_z^{tot} = (2n-N)\frac{\hbar}{2}$, $n = 0, \dots, N$ is $\dim(\mathcal{H}^{(n)}) = \binom{N}{n} < \dim(\mathcal{H}) = 2^N$.

Concrete example: For $N = 30$ sites ($\dim(\mathcal{H}) = 2^{30} = 1{,}07\dots 10^9$), the bigger block (at $n = 15$) has dimension $\dim(\mathcal{H}^{(15)}) = \binom{30}{15} = 7.55\dots 10^8$, and at $n = 10$, $\dim(\mathcal{H}^{(10)}) = 3.0\dots 10^7$ ⇝ block dimension drastically reduced compared to $\dim(\mathcal{H})$

↳ To find all basis state in block n, start with $\sigma = (\underbrace{0,\dots,0}_{N-n},\underbrace{1,\dots,1}_{n})$ and generate all distinct permutations $P(\sigma)$. Order the associated integers $I_{P(\sigma)}$ from smallest (I_σ) to biggest $(I_{\tilde{\sigma}})$ with $\tilde{\sigma} = (\underbrace{1,1,\dots,1}_{n},\underbrace{0,\dots,0}_{N-n})$.

↳ Simplest lookup table $\hat{I}[j]$, with $j = 0, 1, 2, \dots, \binom{N}{n}-1$ is the simply the ordered array of the basis integers $I_{P(\sigma)}$, i.e. $I[0] = I_\sigma, \dots, I\left[\binom{N}{n}-1\right] = I_{\tilde{\sigma}}$.

- Lookup table $\hat{j}[I]$ requires more thought: The naive array of length 2^N would not only contain many "null" entries, but may also exceed memory resources (going to bigger N was the motivation for implementing symmetries).