Potential problem:
The normalization constants $N_m$ can become very large (think of $E_0^m$)

Solution:
generate the normalized basis directly
• start with $\ket{\phi_0}$ arbitrary, normalized, and then

$$
\ket{\phi_1} = \frac{1}{N_1} \left( H \ket{\phi_0} - a_0 \ket{\phi_0} \right).
$$

$$
\ket{\phi_{m+1}} = \frac{1}{N_{m+1}} \left( H \ket{\phi_m} - a_m \ket{\phi_m} - N_m \ket{\phi_{m-1}} \right) = \frac{\ket{\gamma_{m+1}}}{N_{m+1}}
$$

The definition of $N_m$ is different, and no $b_m$:

$$
a_m = \langle \phi_m | H | \phi_m \rangle
$$

$$
N_m = \langle \gamma_m | \gamma_m \rangle^{-1/2}
$$

Generate $\ket{\gamma_m}$ first, normalize to get $N_{m+1}$

The H-matrix is

$$
\langle \phi_{m-1} | H | \phi_m \rangle = N_m
$$

$$
\langle \phi_m | H | \phi_m \rangle = a_m
$$

$$
\langle \phi_{m+1} | H | \phi_m \rangle = N_{m+1}
$$
Lanczos basis generation in practice

Here: generate the orthogonal basis \{Φ_m\} directly

\[ |φ_m⟩ = \sum_{a=1}^{M} φ_m(a)|a⟩, \quad m = 0, \ldots, Λ \]

in a given symmetry block of size M

The coefficients \( φ_m(a) \) are stored as Λ+1 vectors of size M

• may store only the vectors \( φ_{m-1} \) and \( φ_m \) to generate \( φ_{m+1} \)
  • but basis has to be re-generated when computing expectation values
  • stabilization by “re-orthogonalization” (later) requires storage of all \( φ_m \)

The main computational effort is in acting with the hamiltonian; \( H|φ_m⟩ \)

• implement as a subroutine hoperation(φ,γ), where |γ⟩=H|φ⟩
• state normalization implemented as normalize(φ,n)
  • \( φ \) = vector to normalize, \( n = <φ|φ> \) before normalization
Pseudocode; Lanczos basis generation

Initial random state

\[
do \ i = 1, M \\
\phi_0(i) = \text{random}[0 - 1] \\
\text{enddo} \\
\text{call normalize}(\phi_0, n_0)
\]

second state

\[
\text{call hoperation}(\phi_0, \phi_1) \\
a_0 = \langle \phi_0 | \phi_1 \rangle; \ \phi_1 = \phi_0 - a_0 | \phi_1 \rangle \\
\text{call normalize}(\phi_1, n_1)
\]

Generate the rest of the states

\[
do \ m = 1, \Lambda - 1 \\
\text{call hoperation}(\phi_m, \phi_{m+1}) \\
a_m = \langle \phi_m | \phi_{m+1} \rangle \\
\phi_{m+1} = \phi_{m+1} - a_m \phi_m - n_m \phi_{m-1} \\
\text{call normalize}(\phi_{m+1}, n_{m+1}) \\
\text{enddo}
\]

Note: the H-matrix can be constructed and diagonalized after each step
  • follow evolution of energy versus \Lambda
  • stop based on some convergence criterion on \(E_0\) (or higher energy)
  • expectation values converge slower than energies
The subroutine \texttt{hoperation}(\phi, \gamma)\ implements

\[ H |\phi \rangle = |\gamma \rangle = \sum_{a=1}^{M} \sum_{b=1}^{M} \phi(a) \langle b | H | a \rangle | b \rangle \]

\[ |\phi \rangle = \sum_{a=1}^{M} \phi(a) | a \rangle \]

in a given symmetry block \((M = \text{block size})\)

We do not want to store \(H\) as an \(M \times M\) matrix (too big). Two options:
- carry out the operations on the fly; only the vectors are stored
- store \(H\) in a compact form; only non-0 elements (sparse matrix)

**Storing \(H\) speeds up the Lanczos iterations**
- but may require a lot of memory

**Compact storage of \(H\):** For each \(a=1,M\)
- \(e_a\) is the number of non-0 elements \(\langle b | H | a \rangle\)
- labels \(i=s_a+1, s_a+e_a\) will refer to these matrix elements;
- \(H(i)\) contains the values of the matrix elements \(\langle b | H | a \rangle\)
- \(B(i)\) contains the corresponding “target” state index \(b\)
- The hamiltonian is symmetric
  - store only elements with \(b \leq a\) (divide diagonal elements by 2)
Pseudocode; hamiltonian operation with compact storage

```
subroutine hoperation(φ, γ)
    γ = 0; i = 0
    do a = 1, M
        do j = 1, e_a
            i = i + 1
            γ(B(i)) = γ(B(i)) + H(i)φ(a)
            γ(a) = γ(a) + H(i)φ(B(i))
        enddo
    enddo
enddo
```

\[
H |ϕ⟩ = |γ⟩ = \sum_{a=1}^{M} \sum_{b=1}^{M} φ(a)⟨b|H|a⟩|b⟩
\]

**Further storage compactification possible**

- small number of different elements
- use mapping \(⟨b|H|a⟩ \rightarrow \text{integer}\)
- many operations on \(|a⟩\) give same \(|b⟩\)
  - add up all contributions before storing
**Operator expectation values**

Diagonalizing the tri-diagonal matrix → eigenstates in the Lanczos basis
- eigenvectors $v_n$, energies $E_n$
- only some number of low-energy states ($<< \Lambda$) are correct eigenstates of $H$

To compute expectation values we normally go back to the original basis

$$
\psi_n(a) = \sum_{m=0}^{\Lambda} v_n(m) \phi_m(a), \quad a = 1, \ldots, M
$$

To compute $\langle \psi_n | O | \psi_n \rangle$ first construct

$$
O | \psi_n \rangle = | \psi_n^O \rangle = \sum_{a=1}^{M} \psi_n(a) O | a \rangle
$$

$$
= \sum_{a=1}^{M} \sum_{b=1}^{M} \psi_n(a) | b \rangle \langle b | O | a \rangle
$$

$$
= \sum_{b=1}^{M} \psi_n^O(b) | b \rangle
$$

$\psi_n^O(b) = \sum_{a=1}^{M} \psi_n(a) \langle b | O | a \rangle$

Then evaluate the scalar product

$$
\langle \psi_n | O | \psi_n \rangle = \langle \psi_n | \psi_n^O \rangle = \sum_{a=1}^{M} \psi_n(a) \psi_n^O(a)
$$
Convergence properties of the Lanczos method

Example; 24-site chain
\( m_z = 0, k = 0, p = 1, z = 1 \)
block size \( M = 28416 \)

Ground state converges first, then successively excited states

**Loss of orthogonality:** accumulation of numerical error → basis becomes non-orthogonal
- higher states collapse down onto lower ones
- can be cured with re-orthogonalization

Example; 16-site chain
\( m_z = 0, k = 0, p = 1, z = 1 \)
block size \( M = 212 \)
- (a) non-orthogonality
- (b) re-orthogonalized
Re-orthogonalization procedure

For each state generated, remove all components of prior states, \( i=1,...,m \)
- easy if we work with the normalized basis and all states are stored

\[
|\phi_m\rangle \rightarrow \frac{|\phi_m\rangle - q|\phi_i\rangle}{1 - q^2}, \quad q = \langle \phi_i | \phi_m \rangle
\]

Pseudocode: modify state generation

\[
\text{do } m = 1, \Lambda - 1 \\
\quad \text{call hoperation}(\phi_m, \phi_{m+1}) \\
\quad a_m = \langle \phi_m | \phi_{m+1} \rangle; \ \phi_{m+1} = \phi_{m+1} - a_m \phi_m - n_m \phi_{m-1} \\
\quad \text{call normalize}(\phi_{m+1}, n_{m+1}) \\
\quad \text{do } i = 1, m \\
\quad \quad q = \langle \phi_{m+1} | \phi_i \rangle; \ \phi_{m+1} = (\phi_{m+1} - q \phi_i)/(1 - q^2) \\
\quad \text{enddo} \\
\text{enddo}
\]

Note: the Lanczos method can only generate a single state of a multiplet
- some random linear combination of degenerate states

Example: 2 degenerate states \( i, j \):

\[
H^\Lambda |\Psi\rangle = \sum_{m \neq i, j} c_m E_m^\Lambda |\psi_m\rangle + E_{i,j}^m (c_i |\psi_i\rangle + c_j |\psi_j\rangle)
\]

The mixing of the duplet is determined by \( c_i, c_j \) of the initial state