The two quantum chemical approaches to the quantum N-bod problem - Hartree-Fock and density functional theory - aim at breaking down the complexity of the problem by making certain assumptions about the ground-state wavefunction of the many-body system, while keeping a rather general form of the Hamiltonian (typically in Born-Oppenheimer approximation).

Probably the crudest approximation lies in the assumption that electron correlation effects are weak. While this is acceptable for normal metals, band insulators and semi-conductors, this approach fails for all materials in which electron correlations are strong and play an essential role. This includes in particular, almost all transition metal compounds (with partially filled inner shells) which over the past 25 years have been shown to exhibit a zoo of remarkable phenomena, most strikingly high-temperature superconductivity.

To understand the properties of these materials, the Hamilton operator of the full quantum chemical problem is usually simplified to generic models, which still contain the same important features, but which are at the same time easier to investigate. While these generic models can be used to understand the physics of these materials, they should not be used to directly make quantitative fits to experiments.

The simplest model along these lines is the so-called tight-binding model, which concentrates on the valence bands. It assumes that valence electrons can "hop" between nearest neighbor atoms only arriving at a Hamiltonian of the form

$$\mathcal{H} = \sum_{\langle ij \rangle \beta} \left(t_{ij} \, c^{+}_{i\beta} \, c_{j\beta} + h.c. \right)$$

This model is easily solvable by Fourier transformation, as there are no interactions between the electrons.

The Hubbard model

To include effects of electron correlations, the Hubbard model includes only the often dominant intra-orbital repulsion

$$\mathcal{H} = \sum_{\langle ij \rangle \beta} \left(t_{ij} \, c^{+}_{i\beta} \, c_{j\beta} + h.c. \right) + \sum_{i} U_i \, n_{i\uparrow} n_{i\downarrow}$$

4N

The Hubbard model can also be viewed as a minimal model encoding the competition of kinetic and potential energies. *) It has been studied quite extensively (in the context of high-temperature superconductivity or the metal-insulator transition), but except for its 1D version remains a far from understood model for correlated electron systems.

* In contrast to band insulators, which are insulators because all bands are either completely filled or empty, the Hubbard model at large U is insulating at half-filling where there is one electron per orbital. The reason is the strong Coulomb repulsion U between the electrons, which prohibit any electron movement in the half-filled case at low temperatures.

In this insulating state of the Hubbard model (the so-called Mott insulator) the Hubbard model can be simplified to a quantum Heisenberg model, containing exactly one spin per site.

$$\mathcal{H} = \sum_{\langle ij \rangle} J_{ij} \, \vec{s}_i \, \vec{s}_j$$

$(2S+1)^N$

For large U/t a perturbation expansion in t/U readily gives (in leading order)

$$J_{ij} = 2 t_{ij}^2 \left(\frac{1}{U_i} + \frac{1}{U_j} \right).$$

The Heisenberg model is the relevant effective model at temperatures $T \ll t_{ij}, U$ ($\approx 10^4 K$ in copper oxides).

The t-J model

The t-J model is the effective model for large U at low temperatures away from half-filling. Its Hamiltonian is given by

$$\mathcal{H} = \sum_{\langle ij \rangle \sigma} \left\{ (1 - n_{i,-\sigma}) \, t_{ij} \, c_{i\sigma}^{+} \, c_{j\sigma} \, (1 - n_{j,-\sigma}) + \text{h.c.} \right\}$$

$$+ \sum_{\langle ij \rangle} J_{ij} \left(\vec{s}_i \vec{s}_j - n_i n_j / 4 \right).$$

3^N

As double-occupancy is prohibited in the t-J model there are only three instead of four states per orbital — thus greatly reducing the Hilbert space size.

The most accurate method to solve these quantum lattice models is exact diagonalization of the Hamiltonian matrix

$$H|\psi\rangle = E|\psi\rangle$$

Typically, this is done via a sparse diagonalization technique that will calculate only the few lowest eigenvalues and eigenvectors and thus allows to discuss the zero-temperature limit of these models. A commonly used algorithm for sparse diagonalization is the Lanczos algorithm, which we will discuss in the following. Before doing so, let's take a look at what can currently be done with these techniques:

- Hubbard models

  20 sites square lattice at half filling, 21 sites triangular lattice
  22-25 sites in ultracold atoms setting (w/o spatial symmetries)
  up to 160 billion basis states

- Spin $S=1/2$ models

  40 spins square lattice, 39 sites triangular, 42 sites honeycomb
  64 spins or more in elevated magnetization sectors
  up to 1.5 billion basis states with symmetries

- t-J models

  32 sites checkerboard with 2 holes
  32 sites square with 4 holes
  up to 2.8 billion basis states

- fractional quantum Hall effect

  up to 16-20 electrons for different filling fractions ν
  up to 3.5 billion basis states

Instead of sparse diagonalization one could also perform a full diagonalization of the Hamiltonian matix. While this can be done for even smaller system sizes only, the benefit comes in form of a full set of eigenvalues and eigenvectors which readily allows to access all thermodynamic properties of the model system.

Here are the key ingredients of an exact diagonalization code:

- Hilbert space
  - basis representation, lookup techniques
  - symmetries
- Hamiltonian matix
  - sparse matix representation (memory / disk)
  - matix recalculation on the fly (matix-free)
- linear algebra: Eigensolver / time propagation
  - LAPACK full diagonalization
  - Lanczos type diagonalization
  - more exotic eigensolver techniques (real or imaginary time propagation.
- Observables
  - static quantities (multipoint correlation functions, ...)
  - dynamic observables (spectral functions, density of states, ...)
  - real-time evolution

We need to represent all states of a given Hilbert space in the computer. In particular, we want to choose a representation which makes it simple to act with the Hamiltonian (or other operators) on the states, and at the same time minimize the effort to localize a given state in the basis.

Often, a binary encoding is used. Here is an example for an ensemble of $S=1/2$ sites

$$|\uparrow\uparrow\downarrow\uparrow\rangle \longrightarrow [1101]_2 = 13$$

This binary representation allows for an efficient detection of up or down spins which can be done with bit-tests.
The transverse exchange $S^+S^- + S^-S^+$ of two neighboring sites can be performed by an XOR operation:

$$\underset{\text{initial configuration}}{[1101]_2} \text{ XOR } [0110]_2 = \underset{\text{final configuration}}{[1011]_2}$$

the two coupled sites are indicated by bit 1

A similar binary representation can also be used for spinless fermions (or bosons). For higher spins, e.g. $S=1$, or spinfull fermions one bit representations are obviously not sufficient.
Use ternary representation (for $S=1$) or use a 2-bit representation.

If we constrain the Hilbert space, e.g. the total $S^z = 0$ sector for a set of spin-$\frac{1}{2}$'s or the total number of spinless fermions, we will only use a fraction of the total state space (which grows like 2^N). Of course, we want to avoid any overhead (in particular with regard to the allocated memory), but more importantly we need a fast mechanism to find the index of a newly generated configuration (e.g. after acting with the Hamiltonian on a given state) in this constrained Hilbert space. We thus need a clever way to map the original 2^N possible states to the actual set of basis states.

One way to solve this **lookup problem** is to sort the (allowed) basis states via their binary (or decimal) representation and then perform a binary search, which will give a logarithmic access time. A faster (but potentially more memory intensive) alternative is to employ a hash table, which allows for constant access time.

→ Lin. Algebra

Basis representation - symmetries

Consider an XXZ spin model on a lattice

$$H = \sum_{\langle ij \rangle} J_{ij}^{xy} \left(S_i^x S_j^x + S_i^y S_j^y \right) + J_{ij}^z S_i^z S_j^z .$$

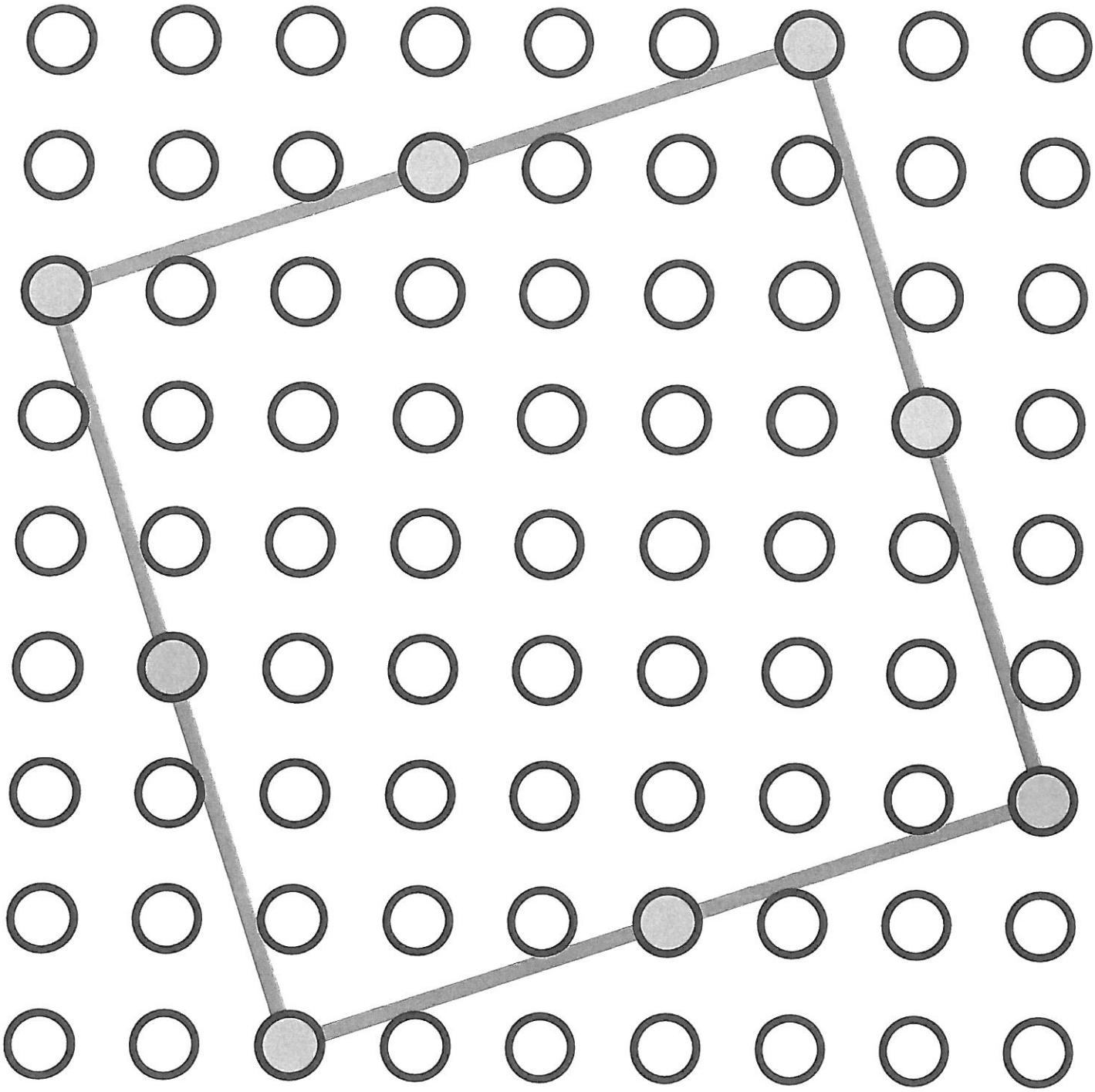
We would like to understand what the symmetries of this problem are ?

- The Hamiltonian conserves the total S^z (which we can see by noting that $S_i^x S_j^x + S_i^y S_j^y = \frac{1}{2}(S_i^+ S_j^- + S_i^- S_j^+)$. We can therefore work within a given S^z-sector. This is easily implemented while constructing the basis (as we have discussed before).

- The Hamiltonian is invariant under the space group, typically a few hundred elements. On top we have often translational invariance When we consider the Hamiltonian on a finite cluster, we want to carefully implement all these symmetry relations (present on the finite cluster).

  Spatial symmetries are important for a reduction of the Hilbert space, symmetry resolved eigenstates teach us a lot about the physics at work, e.g. the dispersion of excitations, symmetry breaking tendencies, (topological) degeneracies and much more.

- At the Heisenberg point, the total spin is also conserved. It is however difficult to combine the SU(2) symmetries with the lattice symmetries in a computationally useful way.



The Lanczos algorithm (Cornelius Lanczos 1951)

The Lanczos algorithm is an iterative algorithm that is an adaptation of the power method to find eigenvalues and eigenvectors of a square Hermitian matrix. It is particularly useful for finding decompositions of very large sparse matrices.

• the power method

Let's first take a look at the power method, which for a given matrix A will find the largest eigenvalue and corresponding eigenvector.

We start from a random vector u_1 and iteratively calculate

$$u_{n+1} = A \cdot u_n$$

which will converge in the large-n limit to the largest eigenvalue $\|u_{n+1}\| / \|u_n\|$ with eigenvector $u_n / \|u_n\|$.

• the Lanczos method

The Lanczos algorithm builds a basis $\{v_1, v_2, \dots, v_M\}$ for the Krylov-subspace $K_M = \text{span}\{u_1, u_2, \dots, u_M\}$, which is constructed via the power method above.

In each step m the algorithm transforms the matrix A into a trigonal matrix T_{mm}: $T_{mm} = V_m^* A V_m$ which has the form

$$T_{mm} = \begin{pmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \beta_2 & \alpha_2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & & \beta_m \\ 0 & \cdots & 0 & \beta_m & \alpha_m \end{pmatrix}$$

The basis $\{ v_1, v_2, \dots, v_M \}$ and the tridiagonal matrix T_{MM} are obtained via the following Lanczos iterations

- v_1 = random vector with norm 1 (of size N)
- $v_0 = 0$
- $\beta_1 = 0$

for $j = 1, 2, \dots, M$ do {

- $\underline{w_j} = A \cdot \underline{v_j} - \beta_j \underline{v_{j-1}}$
- $\alpha_j = v_j^\dagger \cdot w_j$
- $w_j = w_j - \alpha_j v_j$
- $\beta_{j+1} = \| w_j \|$
- $v_{j+1} = w_j / \beta_{j+1}$ }

We only need to keep three vectors of size N in memory.

Dense matrix solvers, in comparison, will need $O(N^2)$ memory.

The so-calculated trigonal matrix T_{MM} can be easily diagonalized (in $O(M^2)$ when also calculating its eigenvectors), which provides eigenvalues $\{ \tau_1, \tau_2, \dots, \tau_M \}$. These turn out to be extremely good approximations of the eigenvalues of A. Convergence is very fast and it takes $M \ll N$ iterations to obtain accurate results.

The eigenvectors of the tridiagonal basis T_{MM} are given in the Krylov basis $\{ v_1, v_2, \dots, v_M \}$ and to obtain the eigenvectors in the original basis (of A) we need to perform a basis rotation. Since we usually do not store all v_i, but only the last three vectors (due to memory constraints) we will have to run the Lanczos iterations a second time — starting from the same initial random vector v_1.