Computational Methods in Many-Body Physics
Prof. M. Knap, Prof. F. Pollmann
A. Romen, M. Lastres, R. Morral Yepes, P. Zechmann

Exam projects
Summer Term 2024

Projects marked with a star are a bit more elaborate. You can pick your favorite project starting from Friday, 21 June, 1 p.m. on our Moodle page. Projects will be assigned on a first come, first serve basis, and each project can be chosen five times. Contact us (= the tutors) if you have questions. Submission of the projects will be possible via our Moodle page until the last Friday before the exam.

Project 1: N-state Potts model (Monte Carlo)

Implement the Metropolis algorithm for the N-state Potts model in 2D and discuss the behavior of the phase transition for different N. The N-state Potts model is a generalization of the Ising model, where each s_i takes N possible values, and the Hamiltonian is given by $H = -J \sum_{\langle i,j \rangle} \delta_{s_i,s_j}$, where $\langle i,j \rangle$ denotes nearest neighbours and δ_{s_i,s_j} the Kronecker-delta. For several values of $N = 2, 4, 6, (8), ...$ perform a finite size analysis $L = 8, 16, 32, (64), ...$ to see the critical behavior in the magnetization and specific heat. Compare your results to the ones for the Ising model ($N = 2$) from the lecture and tutorials.

Project 2: Criticality of the (classical) 3D Ising model (Monte Carlo)

Generalize the Swendsen-Wang algorithm to the (classical) Ising model on a 3D simple cubic lattice and a 3D body-centered cubic lattice (make sure you choose the nearest neighbors correctly) and perform a finite size ($L = 4, 8, 16, 32, ...$) scaling analysis to obtain the critical behavior at the phase transition. Similar to the 2D case discussed in the tutorials, compute the specific heat, magnetization, susceptibility, and Binder cumulant to extract the critical exponents.

Project 3*: The worm algorithm for the 6-vertex model (Monte Carlo)

Implement the worm algorithm discussed in the lecture for the 6-vertex model (`https://en.wikipedia.org/wiki/Ice-type_model`) at infinite temperature, where all allowed configurations are equally likely. Calculate the correlation function $\langle s_0 s_r \rangle \propto \frac{1}{r^a}$ and extract the decay the exponent a.

Project 4*: Entanglement Cooling algorithm (Monte Carlo + Exact Diagonalization)

In this project, we want to study a Metropolis-like algorithm aimed to disentangle a given state, i.e. to reduce its entanglement entropy to 0. We consider a one-dimensional chain of L qubits. The cooling algorithm is described in the paper `https://arxiv.org/pdf/`

1407.4419.pdf (see Algorithm 1 for a summary). In our case, we start from a product state $|0...0\rangle$ and a set of gates $\mathcal{I}$. We then evolve the state applying random gates in $\mathcal{I}$ for L^2 updates, leading to an entangled state (this is the entanglement *heating*). To disentangle, one applies random unitaries among the set $\mathcal{I}$. If the operation reduces the entanglement or keeps it constant, then it is accepted. Otherwise, if it increases it, the operation is accepted with probability $\exp[\beta(S_{\text{new}} - S_{\text{old}})]$, where β plays the role of an inverse temperature. Try the algorithm, for example, for the following set of gates:

- $\mathcal{I} = \{CNOT, NOT, H\}$
- $\mathcal{I} = \{CNOT, H, T\}$

In which cases is the disentangling algorithm successful? How fast is the disentangling as a function of system size (remember to average over many realizations)? How does the temperature affect the algorithm?

Project 5*: Ground state properties of the 2D (transverse field) Ising model with Conservation laws (Exact Diagonalization)

Generalize your ED implementation of exercise 6.1 to the transverse field Ising model in two spatial dimensions, using both k_x and k_y as quantum numbers. Find the ground state (and possibly a few excited states) to locate the quantum phase transition.

Project 6: Many-body localization in a random Heisenberg chain (Exact Diagonalization)

The Lanczos algorithm is an efficient tool to approximately compute the ground state of a Hamiltonian H by first projecting H into the Krylov subspace, finding the ground state there, and then transforming it back to the full Hilbert space. Similarly, it is possible to compute the time evolution in the Krylov subspace to approximate the time evolution in the full Hilbert space. This is equivalent to finding a good approximation for the multiplication of an exponential of an operator to a state; see https://www.jstor.org/stable/2158085. Implement the Lanczos time evolution, building upon the provided implementation of the Lanzcos algorithm in lanczos.py. To perform the evolution to large times $t/J \approx 1000$, divide the time into moderate time steps $dt/J \approx 1$ and rebuild the Krylov space in each time step. Starting from the Néel state $|\uparrow\downarrow\uparrow\downarrow\ldots\rangle$, compute the dynamics of the Heisenberg model with a random field, $H = J\sum_i \vec{S}_i \cdot \vec{S}_{i+1} - \sum_i h_i S_i^z$, where the values of the field $h_i \in [-W, W]$ are chosen from a uniform distribution with a disorder strength W. Make sure to average your results over a sufficient number of realizations of the disorder. Plot the evolution of the imbalance $\mathcal{I} = \langle S^z_{\text{odd}}\rangle - \langle S^z_{\text{even}}\rangle$ for different values of $W/J = 0.1, \ldots 10$. Compute the half-chain entanglement entropy and analyze how it grows over time (see https://arxiv.org/pdf/1202.5532.pdf).

Project 7: Quantum many-body scars in the PXP model (Exact Diagonalization)

In this project you study the evolution of the $|\mathbb{Z}_1\rangle = |0\rangle\,|0\rangle\,...$ as well as the $|\mathbb{Z}_n\rangle = |1\rangle\,\underbrace{|0,...,0\rangle}_{n-1 \text{ times}}\,|1\rangle\,|0,...,0\rangle\,...$, $n = 2, 3, 4$ state under the PXP model given by

$$H_{\text{PXP}} = \sum_i P_{i-1}\sigma_i^x P_{i+1}. \tag{1}$$

Here, $\sigma^x = |1\rangle\langle 0| + |0\rangle\langle 1|$ is the usual Pauli-x matrix and $P = |0\rangle\langle 0|$ is the projector onto the ground state (defined by $|0\rangle$ in our case) and we consider periodic boundary conditions. Your task is to compute the fidelity

$$\mathcal{L}(t) \equiv |\,\langle\psi(0)|\psi(t)\rangle\,|^2 \tag{2}$$

as function of time for all states above, and to observe the Loschmidt echo.
To do so, construct H_{PXP} as sparse matrix. Follow a similar approach as in exercise 6 (you don't need to resolve any symmetries though). You can use the function `scipy.sparse.linalg.expm_multiply` to sequentially evolve your initial states by small time steps (e.g. $\delta t = 0.1$). Look up the documentation to see how it works. Compute $\mathcal{L}(t)$ for all four states up to times $t = 30$ for $L = 12$, and for the $|\mathbb{Z}_1\rangle$, $|\mathbb{Z}_2\rangle$ and $|\mathbb{Z}_4\rangle$ states up to $t = 10$ for $L = 16$. What do you notice? Can you explain? How do your results relate to `https://arxiv.org/pdf/1806.10933`?

Project 8: Eigenstate thermalization of hard-core bosons (Exact Diagonalization)

Analyze how the energy eigenvalues of the hardcore boson model

$$\begin{aligned} H =& J\sum_j [c_j^\dagger c_{j+1} + \text{h.c.}] + J_2 \sum_j [c_j^\dagger c_{j+2} + \text{h.c.}] + \\ & J\sum_j (1+\delta\cdot j)(\hat{n}_j - \frac{1}{2})(\hat{n}_{j+1} - \frac{1}{2}) + J_2 \sum_j (1+\delta\cdot j)(\hat{n}_j - \frac{1}{2})(\hat{n}_{j+2} - \frac{1}{2}). \end{aligned} \tag{3}$$

are distributed. Here, the operator $c_j^\dagger$ creates a hard core boson at site j and c_j destroys it, and $\hat{n}_j$ is the number operator at site j, i.e. measures how many hardcore bosons are present at site j (you may want to define $|0\rangle$ as the unoccupied state and $|1\rangle$ as the occupied one, since at most one hardcore boson can be present at a site). Note that the Hamiltonian conserves the total number of hardcore bosons N.
Diagonalize the Hamiltonian with open b.c. to find the energy eigenvalues for moderate system sizes and given particle number N (you can choose $J = 1.0$, $J_2 = 0.64$, $\delta = 0.05$ and $L = 12, 16$, $N = 3/4 \cdot L$, make sure that $N \neq L/2$). You should use a similar approach as in exercise 6.
Once you found the energy eigenvalues E_i, plot the distribution of the normalized level spacings $s_i = (E_{i+1} - E_i)/\langle s\rangle$ (drop around 10% of the eigenvalues at the lower and upper

end of the spectrum before computing s_i). How are they distributed? *Hint:* Check out `https://en.wikipedia.org/wiki/Random_matrix` or `https://arxiv.org/abs/1509.06411`

Project 9: Measurement-Induced Phase Transitions (Exact Diagonalization)

In this project, we want to study the evolution of entanglement under a random unitary evolution with measurements. The unitary evolution without measurements generically leads to a maximally entangled state while introducing measurements of σ_z during the time evolution destroys the entanglement. We consider a set-up with the structure described in Figure 1 of `https://arxiv.org/pdf/1901.08092.pdf`. The competition between unitary dynamics and measurements leads to a measurement-induced phase transition between a highly entangled (volume law) phase and an area law phase. Study the behavior of this system and the phase transition it features. You can start the evolution of the wavefunction in the state $|0...0\rangle$. As an order parameter for the transition, you can use the quantity $\langle S_{L/2}\rangle/L$, which is expected to be 0 in the area law phase and finite in the volume law one. You can sample random unitaries using scipy, see `scipy.stats.unitary_group`.

Project 10: Construction of the reduced density matrix (MPS)

Use DMRG to calculate the ground state $|\psi\rangle$ of the transverse field Ising model for a large system ($L \approx 100$) as an MPS. Find a method to construct the reduced density matrix of n sites ($n \lesssim 10$) in the center. Measure the entanglement entropy of this subsystem of n sites for the different n.

Project 11*: Purification (MPS)

Purification is the concept that any mixed state can be represented by a pure state on an enlarged Hilbert space. This can be used to represent the thermal density matrix $\rho = e^{-\beta H}$ as an MPS and calculate the thermodynamic properties of the system. The idea is to construct the MPS purification of an infinite temperature state and cool it down to inverse temperature β by imaginary time evolution. Calculate the energy, specific heat, and magnetization as a function of temperature for the transverse field Ising model. You can study large systems ($L \approx 100$), but the reachable inverse temperature β is limited by the growth of operator space entanglement (controlled by the MPS bond dimension).

Project 12: Dynamical correlation functions (MPS)

Use the TEBD and DMRG to calculate correlation functions of the form $\langle\psi_0| e^{\mathrm{i}Ht} S_i^+ e^{-\mathrm{i}Ht} S_j^- |\psi_0\rangle$, where $|\psi_0\rangle$ is the ground state of the transverse field Ising model. Perform a Fourier transformation in space and time to obtain the spectral function and compare your results to your results obtained with ED on exercise sheets 5/6.

Project 13: Phase diagram of the 1D Bose-Hubbard model (MPS)

Use DMRG to find the ground states of the interacting 1D Bose-Hubbard model $H = -t\sum_i (a_i^\dagger a_{i+1} + h.c.) + \frac{U}{2}\sum_i n_i(n_i-1) - \mu\sum_i n_i$ for different choices of the parameters $t/U, \mu/U$. Note that you need to cut off the maximum number of bosons on each site to $n_c = 1, 2, 3, \ldots$. This model features two phases: a Mott insulator and a superfluid. Calculate the correlation function of superfluid order $\langle a_i^\dagger a_{i+r}\rangle$ and the density $\langle n_i\rangle$ and use them to determine the ground state phase diagram. Compare the influence of different cutoffs n_c on the result. To estimate the range of relevant parameters, have a look into literature: `https://arxiv.org/abs/cond-mat/9712307`.

Project 14: Heisenberg model with short and exponential decaying interactions (MPS)

There is a natural way to write down matrix product operators for interactions that decay exponentially in distance (`https://doi.org/10.1088/1367-2630/12/2/025012`). Here we consider $H = \sum_i \sum_{j>i} J e^{-\frac{|i-j|}{\xi}} \vec{S}_i \cdot \vec{S}_j$. Use DMRG to find the ground state. Find out which influence ξ has on the correlations $\langle S_i^z S_j^z\rangle$.

Project 15: Single Site DMRG for the transverse field Ising model (MPS)

In exercise 10 you discussed the two-site version of DMRG. In this project, you implement a simpler approach where you only update a single site at a time. To do so, first adjust the code to compute the single-site effective Hamiltonian, which is then diagonalized to update the MPS. Figure out how to perform the final SVD on the updated tensor for an efficient sweep schedule. Can the bond dimension increase when using this particular update schedule?
Find the ground state of the transverse field Ising model using your single-site DMRG version (you might want to start with a random state of bond dimension χ). Take $J = 1$, and values of g close to the critical point and vary chain length and bond dimension. How do the runtime (per sweep) and the number of sweeps for convergence compare to the two-site version? Compare the energy of this state to the one obtained from the two-site version. How large is the overlap of the two ground states?

Project 16: Finite entanglement scaling (MPS)

Use infinite-DMRG (iDMRG) to study the scaling of the entanglement entropy with maximal bond dimension $\chi_{\max}$ for the critical TFI, i.e. at g=1, and for the easy-plane XXZ model, i.e. $H = \sum_j S_j^x S_{j+1}^x + S_j^y S_{j+1}^y + \Delta S_j^z S_{j+1}^z$ for $\Delta \in [0, 1]$. See `https://arxiv.org/abs/0812.2903`.
While TEBD can be used directly in the thermodynamic limit as seen in the exercises, this is not the case for DMRG. Conceptually, the idea is to start from a finite MPS with a unit cell of length L and continuously update the left and right environment by inserting unit cells to the left and right after every sweeps, see `https://arxiv.org/abs/1805.00055` for further information. This can be efficiently achieved by modifying the finite DMRG

sweep schedule to reuse previously computed environments without explicitly storing additional unit cells.

Project 17*: Heisenberg time-evolution with TEBD (MPS)

Compute the Out of Time Order Correlator (OTOC) from operator TEBD: for Pauli matrices, the OTOC is defined as

$$1 - \frac{1}{Z}\mathrm{Tr}(V_i(t)W_jV_i(t)W_j) \tag{4}$$

with $Z = 2^L$ (the dimension of the Hilbert space). Choose $V_i(t=0) = \sigma_i^z$ and $W_j = \sigma_j^z$. Choose $i = L/2$. Write V and W as MPOs. Reshape $V_i(t=0)$ into an MPS. Write the correct gates such that one can time evolve an MPO (viewed as an MPS) using TEBD. Time evolve V_i and regularly measure the OTOC according to Eq. (4) for all j. Plot the OTOC as a function of time and space (y axis time, x axis space). What structure do you observe?

Project 18*: Time evolution with the MPO method (MPS)

An alternative method for TEBD to perform a real-time evolution was introduced in https://arxiv.org/abs/1407.1832, where the time evolution operator is approximated by MPOs $W^{I,II}$. Construct the MPO W^I for the transverse field Ising model and apply it to an initial state $|\uparrow\uparrow \ldots \uparrow\uparrow\rangle$. Use second order time-steps $\delta t_1 = \frac{1+i}{2}\delta t$ and $\delta t_2 = \frac{1-i}{2}\delta t$ as described in the reference.
To apply an MPO to an MPS there are several possible methods, which are described in section 2.8 of https://arxiv.org/abs/1901.05824. For the *variational application* you can use the DMRG code from the tutorials and modify it accordingly.
Compare your results to what you get by using the TEBD implementation from the tutorials.

Project 19: Gradient descent optimization of MPS for Ground state (MPS)

Consider the transverse field Ising model in 1D. Find a way to compute the gradient of the energy expectation value of an MPS, either by analytic formula or automatic differentiation library (Pytorch, Tensorflow, Jax, etc). Perform gradient descent to update the MPS tensors to obtain the ground state. Compare the performance with the TEBD, DMRG algorithms. Bonus: Try to improve the convergence rate by considering second-order gradient methods, e.g. scipy.optimize.minimize.

Project 20: PEPS with simplified update on a square lattice (PEPS)

Generalize the code from exercise 11.2 (PEPS with simplified update) to (approximately) get the ground state of the transverse field Ising model on the square lattice. In addition

to the on-site expectation values, evaluate (the decay of) correlation functions along the direction of the boundary MPS. Use the decay of correlation functions to locate the transition to the symmetry-broken state.

Project 21: Solving 2-dimensional classical models using boundary MPS

First, express the classical partition function of the classical Ising model using the tensor network discussed in class. Compute the energy and magnetization of the model using the boundary MPS approach and observe/discuss the behavior of the phase transition. How does the behavior change as you increase D_{cut}? Second, extend the result to the N-state Potts model.
The N-state Potts model is a generalization of the Ising model, where each s_i takes N possible values, and the Hamiltonian is given by $H = -J \sum_{\langle i,j \rangle} \delta_{s_i,s_j}$, where $\langle i,j \rangle$ denotes nearest neighbours and δ_{s_i,s_j} the Kronecker-delta. The phase transition is second order below $N \leq 4$ and first order for $N > 4$.

Project 22: Machine learning with Monte-Carlo data for the 2D classical Ising model (Monte-Carlo + Machine Learning)

Generate some Monte-Carlo spin configurations of the 2D square lattice Ising model at two different temperatures (above and below the critical temperature), and use them to train a neural network. Then use this network and study the output for different temperatures, how good are the predicted temperatures? How do your results depend on the system size?

Project 23*: Classification MNIST dataset (digits) with MPS (Tensor Network + Machine Learning)

In this project, we would like to use MPS for the classification of images of digits.

- Take the MNIST dataset and course-graining each image down to a size of 14 by 14 pixels. You can do this by averaging over each 2 by 2 pixels.
- Rescale the pixel value $x_j \in [0,1]$, and map it to a quantum state of one spin,

$$\phi^{s_j}(x_j) = [\cos(\frac{\pi}{2}x_j), \sin(\frac{\pi}{2}x_j)]. \tag{5}$$

  With this mapping, each image corresponds to a product state

$$\Phi^{s_1 s_2 \cdots s_{196}}(\mathbf{x}) = \phi^{s_1}(x_1) \otimes \phi^{s_2}(x_2) \otimes \cdots \otimes \phi^{s_{196}}(x_{196}) \tag{6}$$

- Suppose we can describe the distribution of a given digit by a quantum many-body wavefunction over these 14 by 14 spins, e.g.

$$|\psi_{\text{digit}=2}\rangle = \sum_{s_1 s_2 \cdots s_{196}} \psi^{s_1,s_2,\cdots,s_{196}}_{\text{digit}=2} |s_1 s_2 \cdots s_{196}\rangle, \tag{7}$$

such that any measurement over the spin configuration would give an image of digit 2. One way to obtain an approximation of such wavefunction is simply summing up all the product states of a specific digit in the dataset from step (ii) and then normalizing the wavefunction. Hint: The algorithm of MPS addition can be found in chapter 4.3 in https://arxiv.org/abs/1008.3477.

- Show that by doing so and obtaining the wavefunction of each digit, you can then classify a given digit based on the magnitudes of the overlap with the wavefunction of each digit.