Outline

This project consists of two parts. Part 1 contains 8 mandatory (and one extra credit) questions - covering theory of both quantum machine learning and strongly correlated systems **(you have to answer the questions to both topics)** - and it comprises 40% of the final mark for the project. For part 2 you choose **one** out of 8 topics **(i.e., unlike the theory part you can choose between either a project in quantum machine learning or strongly correlated systems)** and it comprises 60% of the final mark for the project.

Each project in part 2 has a list of "Tasks and Objectives" together with the required content of the report. Correctly carrying out the listed tasks and objectives together with an appropriate report will result in a passing grade that can go up to an 8,5. In order to obtain an even higher grade, you have to carry out the extra objective/task listed at the end of each of the projects.

At the end of this project **you will have to turn in a report**. The report should contain your solutions to the mandatory questions in part 1 and a write-up on the topic of your choice in part 2. This write up should be between 5-10 pages long and be concise and to the point. Also, please do not forget to submit your code for part 2, as this will be taken into account when grading.

The deadline for both parts of the project is 19th of May 2025.

Part 1

Strongly correlated systems

1. (a) Write down the commutation relations for Fermionic creation and annihilation operators.
   (b) Write down a transformation of fermionic creation and annihilation operators into Majorana operators.
   (c) Write down the commutation relations of the resulting Majorana operators.

2. Derive the average locality of a single fermionic creation or annihilation operator after being transformed to a qubit operator by the Jordan-Wigner transform.

3. Write down the Hamiltonian for the Fermionic Hubbard model on a 1-dimensional chain.

4. Transform the Fermionic Hubbard Hamiltonian onto a qubit basis via the Jordan-Wigner transformation. Do these two Hamiltonians have the same eigenspectrum (please explain in words rather than calculating)?

5. What is the scaling of the number of terms in the Hamiltonian with the system size for
   (a) the Hubbard model
   (b) the Heisenberg model
   (c) the Electronic structure problem

Quantum machine learning

1. Describe the implicit ("the quantum kernel estimator") and explicit ("the quantum variational classifier") of quantum variational SVMs. (Hint: `https://arxiv.org/abs/1804.11326`). Feel free to describe either the generic circuits, or the particular ones used in the provided reference.
2. (a) What are feature maps and kernels in SVMs?
   (b) What is the relationship between the two?
   (c) When does a kernel have a corresponding feature map?
3. Do there exist learning problems which quantum leaners can learn but classical learners can not? If yes, provide an example.
4. (Hard question, extra credit) Given an n-qubit circuit of depth d (not dependent on n). Show that you can estimate the probability of the first qubit being in the state $|0\rangle$ in time independent of n using only a classical computer.

Topic 2.F. (*Towards learning separation*). In this project you will use a quantum algorithm to solve a ML problem related to quantum many body physics. Specifically, the task is to learn to predict from data expectation values of unknown observables on time evolved quantum states under a known Hamiltonian. Alternatively, you can consider the task of learning to predict expectation values on ground states of a class of local Hamiltonians. As it will be discussed in the lectures, for a special class of Hamiltonians one could show that those tasks exhibit a provable learning separation with respect to classical algorithms. In this project, we will focus on a class of physically motivated Hamiltonians.

Literature:

1. https://arxiv.org/pdf/2306.16028.pdf
2. https://www.science.org/doi/full/10.1126/science.abk3333
3. https://arxiv.org/pdf/2301.13169.pdf
4. https://arxiv.org/abs/1912.13146
5. https://arxiv.org/abs/2205.06278

Questions:

- Understand and explain what we mean by separation in learning. Why is it different that a separation in computing? Can you give an example of a function which is hard to compute but not hard to learn?
- What are the assumptions on the class of Hamiltonians in the references 2-3? Explain, from an high level point of view, the algorithm in reference 3 and the main idea of why it is guaranteed to work.
- What are the main differences between the algorithms in references 2 and 3? (Resources, sample complexity, time complexity)

Implementation: There are two possible variants of the learning problem we can consider: in **case A** the task is to predict observables on time evolved quantum states, in **case B** we will be dealing with ground states instead. Notice that **case A** will be easier to implement as the quantum algorithm will just have to perform a fixed time evolution on input states. The implementation of **case B** will be instead more involved, as the quantum algorithm will have to find the ground state for the Hamiltonians associated at each data point. However, **case B** allows for a direct comparison with relevant recent results in the literature (Ref. 2-3) and may lead to interesting unexplored scenarios.
Anyway, you can choose either **case A** or **case B**. Notice that the final grade of the

miniproject will not be affected by the choice of **case A** or **case B**.

Case A: Predicting unknown observables on time evolved states The ML problem you will consider is of the same kind of the ones in Ref 2-3, however we consider the simpler case where we predict observables on time evolved states. Let H be a Hamiltonian and consider the associated concept class

$$\mathcal{F}_{H,O} = \{f^\alpha(x) \ \mid \alpha \in [-1,1]^m\} \tag{1}$$

$$\text{with: } f^\alpha(x): \ \ x \in \{0,1\}^n \to f^\alpha(x) = \text{Tr}[\rho_H(x)O(\alpha)] \ \ \& \ \ O(\alpha) = \sum_{i=1}^{m} \alpha_i P_i \tag{2}$$

Where x specifies the initial state $\rho_0(x) = |x\rangle\langle x|$ and $\rho_H(x)$ is the evolved state under the Hamiltonian H, i.e. $\rho_H(x) = U\rho_0(x)U^\dagger$ with $U = e^{iHt}$. The ML algorithm has to learn a model $h(x)$ which approximates the function f^{α^*} for an unknown α^* using $\mathcal{T}^{\alpha^*} = \{(x_\ell, y_\ell)\}_{\ell=1}^N$ as training data. In the ML problem you will consider, the input x will come from an underlying distribution. You can choose the uniform distribution over $x \in \{0,1\}^n$ for example.

1. We will consider the same class of Hamiltonian considered in Ref. 3 given by the 2D anti-ferromagnetic random Heisenberg model $H(J) = \sum_{i,j} J_{ij}(X_iX_j + Y_iY_j + Z_iZ_j)$, you can set the coupling J_{ij} randomly. The Hamiltonian will act on a system of $N = 10$ qubits.

2. As unknown observable you can choose an arbitrary combination of k local Pauli string $O(\alpha_i) = \sum_i \alpha_i P_i$. You can set $k = 3$ for example. This means that the observable $O(\alpha)$ will be the linear combination of all the 3-local Pauli matrices with an arbitrary (of your choice) but fixed α.

3. At this point you can generate data classically by explicitly constructing the initial state $\rho_0(x)$ and computing $\rho_H(x)$ by matrix multiplication.

4. Construct the quantum model and train it using the LASSO regression algorithm:

   (a) For every training point in $T^\alpha = \{(x_\ell, y_\ell)\}_{i=1}^N$ the quantum algorithm prepares multiple copies of the state $\rho_H(x_\ell)$ and computes the estimates of the expectation values $\langle P_j\rangle_\ell = \text{Tr}[\rho_H(x_\ell)P_j] \ \ \forall j = 1, ..., m$ up to a certain precision ϵ_1. The number of copies needed will depend on the desired error ϵ_1 on the expectation values of each $\langle P_j\rangle_\ell$. As you saw in the lectures, you will have to take in consideration a sampling error which scales polynomially with the number of copies of the state.

Note, also, that m scales at most polynomially in n as $\{P_j\}_{j=1}^m$ are local observables.

(b) To prepare the states $\rho_H(x_\ell)$ you can use Trotterization methods that you studied in the course. (You will have to choose an adequate number of Trotter steps!)

(c) Define the model $h(x) = w \cdot \phi(x)$, where $\phi(x)$ is the vector of the Pauli string expectation values $\phi(x) = [\mathrm{Tr}[\rho_H(x)P_1], ..., \mathrm{Tr}[\rho_H(x)P_m]]$ computed at Step 1. Set the right hyperparameter $B \geq 0$ (motivate the choice!) and run the LASSO regression finds an optimal w^* from the following optimization process:

$$\min_{\substack{w\in\mathbb{R}^m \\ ||w||_1\leq B}} \quad \frac{1}{N}\sum_{l=1}^{N} |w \cdot \phi(x_l) - y_l|^2 \tag{3}$$

with $\{(x_l, y_l = \mathrm{Tr}[\rho(x_l)O(\alpha)])\}_{l=1}^N$ being the training data.

Importantly, to meet the learning condition the optimization does not need to be solved exactly, i.e. $w^* = \alpha$. As reported in Ref. 3 it is sufficient to obtain a w^* whose training error is ϵ_2 larger than the optimal one.

5. Compare the results using a classical model (you can use your favourite Neural Network).

The report should contain:

1. Answer to the questions above
2. Report of the ML experiment and comparison of the results between the classical and quantum method.
3. **8+** Try using different Hamiltonians and compare the results, specifically you should look at how the performance of the quantum model vs the classical one varies with respect to the Hamiltonian considered.
4. **9+** Discussion with formal arguments on why there should exist a Hamiltonian which gives rise to a learning separation for this kind of task. Identify an example of such Hamiltonian providing proofs of your claims.

Case B (Hard): Predicting unknown observables on ground states

The ML problem you will consider is of the same as the ones in Ref 2-3. Let

$\mathcal{H} = \{H(x) \mid x \in \{0,1\}^n\}$ be a family of local Hamiltonians and consider the associated concept class

$$\mathcal{F}_{\mathcal{H},O} = \{f^\alpha(x) \mid \alpha \in [-1,1]^m\} \tag{4}$$

$$\text{with: } f^\alpha(x): \ x \in \{-1,1\}^n \to f^\alpha(x) = \text{Tr}[\rho_H(x)O(\alpha)] \ \ \& \ \ O(\alpha) = \sum_{i=1}^m \alpha_i P_i \tag{5}$$

Where x specifies the hamiltonian $H(x)$ in the family $\mathcal{H}$ and $\rho_H(x)$ is the ground state of the Hamiltonian $H(x)$. The ML algorithm has to learn a model $h(x)$ which approximates the function f^{α^*} for an unknown α^* using $\mathcal{T}^{\alpha^*} = \{(x_\ell, y_\ell)\}_{\ell=1}^N$ as training data.

1. We will consider the same class of Hamiltonian considered in Ref. 2 given by the 2D anti-ferromagnetic random Heisenberg model $H(J) = \sum_{i,j} J_{ij}(X_iX_j + Y_iY_j + Z_iZ_j)$, for $N = 10$ qubits. Notice that in this case the couplings J_{ij} will exactly be your input vectors x for $x \in \{-1,1\}^n$. You may assume they come from a underlying distributions (e.g. uniform distribution).

2. As unknown observable you can choose an arbitrary combination of k local Pauli string $O(\alpha_i) = \sum_i \alpha_i P_i$. You can set $k = 3$ for example. This means that the observable $O(\alpha)$ will be the linear combination of all the 3-local Pauli matrices with an arbitrary (of your choice) but fixed α.

3. At this point you can generate data classically by diagonalizing $H(J)$ for different vectors J coming (as said before) from an underlying distribution. Notice that the theoretical guarantees for the classical ML algorithm of Ref.2 only works for data coming from the uniform distribution.

4. Construct the quantum model and train it using the LASSO regression algorithm:

   (a) For every training point in $T^\alpha = \{(x_\ell, y_\ell)\}_{i=1}^N$ the quantum algorithm prepares poly(n) copies of the state $\rho_{\mathcal{H}}(x_\ell)$ and computes the estimates of the expectation values $\langle P_j \rangle_\ell = \text{Tr}[\rho_{\mathcal{H}}(x_\ell)P_j] \ \ \forall j = 1, ..., m$ up to a certain precision ϵ_1.
   Note that m scales at most polynomially in n as $\{P_j\}_{j=1}^m$ are local observables.

   (b) To prepare the states $\rho_{\mathcal{H}}(x_\ell)$ you can use the variational algorithm in Ref. 4-5

   (c) Define the model $h(x) = w \cdot \phi(x)$, where $\phi(x)$ is the vector of the Pauli string expectation values $\phi(x) = [\text{Tr}[\rho_{\mathcal{H}}(x)P_1], ..., \text{Tr}[\rho_{\mathcal{H}}(x)P_m]]$ computed at Step 1. Set the right hyperparameter $B \geq 0$ (motivate the choice!)

and run the LASSO regression finds an optimal w^* from the following optimization process:

$$\min_{\substack{w\in\mathbb{R}^m \\ ||w||_1\leq B}} \frac{1}{N}\sum_{l=1}^{N}|w\cdot\phi(x_l)-y_l|^2 \tag{6}$$

with $\{(x_l, y_l = \mathrm{Tr}[\rho(x_l)O(\alpha)])\}_{l=1}^N$ being the training data.
Importantly, to meet the learning condition the optimization does not need to be solved exactly, i.e. $w^* = \alpha$. As reported in Ref. 3 it is sufficient to obtain a w^* whose training error is ϵ_2 larger than the optimal one.

5. Construct the classical model as described in Ref.3

The report should contain:

1. Answer to the questions above
2. Report of the ML experiment and comparison of the results between the classical and quantum method.
3. **8+** Try using different families of local Hamiltonians and compare the results, specifically you should look at how the performance of the quantum model vs the classical one varies with respect to the Hamiltonian family considered.
4. **9+** Discussion with formal arguments on why there should exist a family of Hamiltonians which give rise to a learning separation for this kind of task. Identify an example of such family of Hamiltonians providing proofs of your claim.