Outline

This project consists of two parts. Part 1 contains 12 mandatory 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 7 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 Objective" 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 (except for project 5, which is generally more difficult). 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 you code for part 2, as this will be taken into account when grading. **The tentative deadline for this project is 23-05-2021**.

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. What is the average size (locality) of a single fermionic creation or annihilation operator after being tranformed to a qubit operator by
   (a) the Jordan-Wigner transform, and
   (b) the Bravyi-Kitaev 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 (you can explain rather than explicitely calculating it)?

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, and
   (c) the Electronic structure problem

Quantum machine learning

1. Describe the implicit (i.e., "the quantum kernel classifier") and explicit (i.e, "the quantum variational classifier") versions of quantum variational SVMs.

   Hints. *https://arxiv.org/abs/1804.11326*

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?

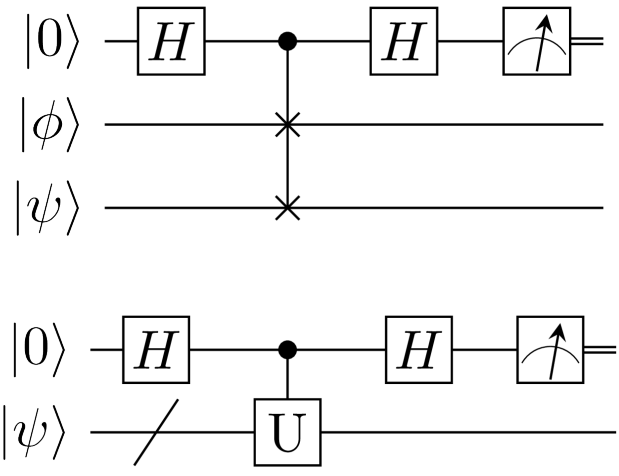
3. Consider two unitaries U_1 and U_2 that prepare the n-qubit quantum states $|\psi\rangle$ and $|\phi\rangle$ from the state $|0^n\rangle$, respectively, i.e.,

$$U_1 |0^n\rangle = |\psi\rangle ,$$
$$U_2 |0^n\rangle = |\phi\rangle .$$

   One can see that estimating the probability of observing (i.e., 'measuring') the state $|0^n\rangle$, from the state $U_2^\dagger U_1 |0\rangle$ can be used to estimate the overlap (i.e., inner product) $|\langle\psi|\phi\rangle|$.

   Remark. *Since global phase does not matter, to discuss the phase/sign as well, extra assumptions and tricks are needed.*

   There are other ways of doing this, each requiring different resources. Namely, to estimate the overlap (i.e., inner product) between two quantum states $|\psi\rangle$ and $|\phi\rangle$, consider the following circuits



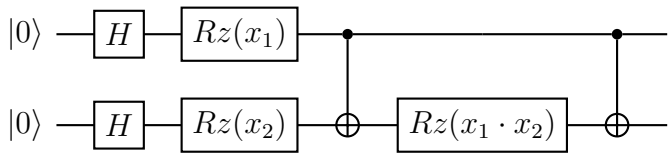
   where the unitary U satisfies $U|\psi\rangle = |\phi\rangle$. These are called the SWAP and the Hadamard test, respectively. In the SWAP test, we use the controlled-SWAP gate.

   (a) Give the measurement outcome probabilities of both circuits when measuring the first qubit in the computational basis.
   (b) Explain how the outcome probabilities can be used to estimate (some aspect) of the overlap (i.e., inner product) of $|\psi\rangle$ and $|\phi\rangle$.

(c) How does the precision of the esimation depend on the number of uses of U, U_1, U_2?

Hints.
- *https://www.cs.umd.edu/~amchilds/talks/qalg.pdf, pg. 31.*
- *https://arxiv.org/pdf/1804.03719.pdf, pg. 36.*

4. Consider the feature map circuit



measured in the computational basis. The outputs are governed by a probability distribution over bit strings.

(a) Do angles exist for which this probability distribution is a product distribution (i.e. the output probabilities of the two bits are uncorrelated)?

(b) Can the outcome probabilities be correlated?

5. Describe two classes of classical reinforcement learning algorithms and explain their basic ideas. No more than 5 sentences per model are needed.

6. Do there exist learning problems which quantum learners can learn but classical learners can not?

7. (Hard question, extra credit) Given an n-qubit circuit of depth d (not depending 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.

Part 2

Strongly correlated systems

Project 1. (*Hamiltonian variational ansatze*).
In this project you will implement and optimize the Hamiltonian variational ansatz of Wecker, Hastings and Troyer for the Hubbard model on a 2×2 lattice

Literature:

- `https://arxiv.org/abs/1507.08969`.

Objectives and tasks:

- Learn and understand the Hamiltonian variational ansatz.
  - What is the problem that is being solved?
  - What is the motivation for this choice of variational ansatz?
  - For which systems do the authors suggest this ansatz might be preferable?
- Understand the Hubbard model on a square lattice.
  - How many nearest neighbours does each site connect to on an infinite lattice?
  - How many nearest neighbours does each site connect to on a 2×2 lattice?
  - Write h_h, h_v, and h_U (defined below Eq. 2) in terms of creation and annihilation operators for a 2×2 lattice.
  - How many qubits are required to represent the Hubbard model on a single site?
  - What is the non-interacting Hamiltonian for this system?
- Provide an implementation of the Hamiltonian variational ansatz for the Hubbard model
  - Write a function to generate hermitian terms in h_h, h_v and h_U.
  - Write a function to generate the non-interacting and interacting Hamiltonians.
  - Write a function to generate circuits implementing $e^{i\theta h}$ terms given the above. Remember to leave θ a free parameter, and transform from fermions into qubits.
  - Write a function to generate the Hamiltonian variational ansatz using the above sub-functions.
  - Write a function to generate the non-interacting starting state using the above (the `openfermion.circuits.prepare_gaussian_state` function may be relevant here).
  - Implement the VQE using SciPy and the above, measure the energy at $t = 1$ and $U = 2$, and compare performance to the result found in ArXiv:1507.08969 as a function of the step size S. Be careful with optimization!

The report should contain:

1. A description of the Hamiltonian and system being studied.

2. A description and explanation of the algorithm and quantum-classical hybrid scheme.
3. An implementation of the ansatz and evaluation of performance (preferably in a jupyter notebook)
4. Optimize the VQE without noise. Then test its robustness with respect to either circuit or sampling noise.

For 8+: Analyze the robustness of the VQE with respect to circuit or sampling noise during the optimization step.
For 10: Make a thorough, realistic, analysis of the effects of various sources of imperfection in the performance of the algorithm.

Project 2. (*Obtaining excited state energies via eigenstate witnessing*).
In this project you will implement the eigenstate witnessing algorithm of Santagati *et al* for the H_2 molecule.

Literature:

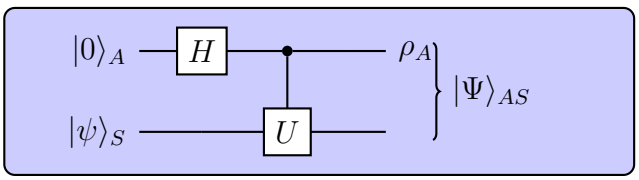
- `https://arxiv.org/abs/1611.03511`.

Objectives and tasks:

- Learn and understand the WAVES protocol
  - What is the main purpose of the WAVES protocol?
  - How does the WAVES protocol achieve this?
  - How is the cost function of the WAVES protocol measured?
  - What are the excitation operators (eq. 10) used for in the WAVES protocol?
- Provide an implementation of the WAVES protocol to find ground and excited states of the H2 molecule
  - Write a function to generate the Hamiltonian of the H_2 molecule (choose your favourite bond distance).
  - Write a function to generate the Trotterized parametrized Hamiltonian ansatz (Eq. 7).
  - Write a function to calculate an appropriate value of t based on the non-interacting part of the H_2 Hamiltonian.
  - Write a function to generate a circuit for time evolution under the full H_2 Hamiltonian for time t, conditional on an ancilla qubit.
  - Combine the above functions to implement step 1 of the WAVES protocol.
  - Write a function to generate excitation operators for the H_2 molecule.
  - Combine this with the above to implement step 2 of the WAVES protocol.

The report should contain:

1. A brief description of the WAVES protocol and what it achieves.
2. A detailed explanation on how to obtain the reduced density matrix ρ_A describing the ancillary 1 qubit state of the following circuit:



where A denotes *Ancilla* which is a $1-$qubit register and S denotes *System*, being an n-qubit register. U is an arbitrary unitary and $|\psi\rangle_S$ is an arbitrary state. Hint: use the eigendecomposition of U to express $|\psi\rangle_S$ conveniently. The state after the circuit is $|\Psi\rangle_{AS}$ from which we trace out the system register, thus obtaining the 1-qubit reduced density matrix ρ_A.

3. An implementation (preferably in a jupyter notebook) of the WAVES protocol on the H_2 molecule to find the ground state and excited state manifold.

For 8+: Implement IPEA on the obtained ground and excited states and determine the accuracy of the resulting energies.
For 10: Compare the WAVES protocol to other protocols for finding excited states in terms of cost to implement and accuracy obtained.

Quantum machine learning

Project 1. (*High-dimensional data*).

In this project you will compare different techniques for dealing with high-dimensional real-world datasets. Examples of such datasets are the Wine[1], BreastCancer[2] or MNIST[3] dataset. The main challenge with high-dimensional data is that you have to find a way to fit it on a limited number of qubits (i.e., whatever size your simulations can handle). Ultimately, you will benchmark the performance of your techniques using cross-validation and compare them.

Literature:

- `https://arxiv.org/pdf/1804.11326.pdf`
- `https://arxiv.org/pdf/1802.06002.pdf`
- `https://arxiv.org/pdf/1804.00633.pdf`
- `https://www.tensorflow.org/quantum/tutorials/mnist`
- `https://pennylane.ai/qml/demos/tutorial_variational_classifier.html`

Objectives and tasks:

- Devise and implement a technique to classify high-dimensional data using a parameterized quantum circuit (PQC). Examples of techniques you could use are:
  - Principal Component Analysis (PCA)[4].
  - Datareuploading to sequentially upload parts of the data[6].
    - `pennylane.ai/qml/demos/tutorial_data_reuploading_classifier.html`
- Benchmark and analyze the resulting models using cross-validation.

The report should contain:

1. An explanation of the different techniques you use to allow a parameterized quantum circuit to classify high-dimensional data.
2. An explanation of the corresponding implementation (i.e., the code).
3. Discussion of the benchmarking results of your models.

For an 8,5+: Design, implement and analyze your own technique to classify high-dimensional data on a parameterized quantum circuit with a limited number of qubits. Note that you still have to implement at least one of the methods mentioned in the "Objectives and Tasks" list.

[4] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`
[1] `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_wine.html`
[2] `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html`
[3] `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html#sklearn.datasets.load_digits`
[5] `https://www.tensorflow.org/quantum/api_docs/python/tfq/layers/ControlledPQC`
[6] `https://arxiv.org/pdf/1907.02085.pdf`

Project 2. (*Expressivity and rank of the observable*).

In this project you will experiment with the influence of the rank of the observable you measure at the end of your parameterized quantum circuit when doing classification (as discussed in the lectures on the explicit realization of quantum SVMs). In particular, you will generate artificial data using a parameterized quantum circuit and a *high-rank* observable and try to see whether a parameterized quantum circuit with a *low-rank* observable is able to correctly classify the data.

Literature:

- `https://arxiv.org/pdf/1804.11326.pdf`
- `https://arxiv.org/pdf/1802.06002.pdf`
- `https://arxiv.org/pdf/1804.00633.pdf`
- `https://www.tensorflow.org/quantum/tutorials/mnist`
- `https://pennylane.ai/qml/demos/tutorial_variational_classifier.html`

Objectives and tasks:

- Generate artificial data by fixing a parameterized quantum circuit model, feeding it random input parameters, and measuring a *high-rank* observable to obtain the labels.

  Remark. – *Analogous to the artificial data in* ***`arxiv.org/pdf/1804.11326.pdf`****.*
  – *Do not use data reuploading for your parameterized quantum circuit.*

- Train the same parameterized quantum circuit but with a *low-rank* observable to classify the above generated artificial dataset.
- Benchmark the performance of the resulting low-rank model using cross-validation.

The report should contain:

1. An explanation of how the artificial data is generated.
2. An explanation of the code that generates the artificial data and trains a low-rank parameterized quantum circuit model to classify this data.
3. Discussion of the benchmarking resuts of the low-rank model.

For an 8,5+: Experiment with using the rank of the observable as a *regularization technique*. More specifically, instead of looking at the effect of increasing the rank of the observable on the expressivity of the classifier, investigate whether restricting the rank of the observable can help prevent *overfitting* (i.e., near perfect training accuracy but poor generalization performance).

Project 3. (*Sparse and deep versus shallow QCBMs*).

In this project you will compare the performances of Quantum Circuit Born Machines (QCBMs) that are deep but sparse (i.e., only a few of its parameters are nonzero) with ones that are shallow and potentially dense.

Literature:

- `https://arxiv.org/pdf/1804.04168.pdf`

Objectives and tasks:

- Write code that implements and trains QCBMs.
- Generate a shallow QCBM (i.e., one which uses a relatively small parameterized quantum circuit) and initialize it with randomly chosen parameters (i.e., do **not** train it).
- Generate and train a second deeper QCBM (i.e., one which uses a deeper parameterized quantum circuit as above) on samples generated by the previous QCBM.
  - Do not forget to benchmark its performance beyond training examples.
- Enforce *sparsity* for the deeper QCBM. That is, train the deeper QCBM while making sure that only few of its parameters are nonzero. For example, this can be done by
  - Setting the $x\%$ of smallest parameters to zero after a set period of time.
  - Change the loss function by adding a term that penalizes the number/magnitude of the parameters, i.e., $||\vec{\theta}||_0$, $||\vec{\theta}||_1$ or $||\vec{\theta}||_2$ where $\vec{\theta} \in \mathbb{R}^d$ are the QCBM parameters.
- Benchmark and test whether the deep but sparse QCBM still performs well.

The report should contain:

1. An explanation and implementation of QCBMs.
2. A discussion of why regularization (e.g., enforcing sparsity) is important for ML.
3. Discussion of the experimental results on training a deep QCBM to learn the distribution generated by a shallow QCBM with and without enforcing sparsity of the deep QCBM.

For an 8,5+: Train your QCBM on a classical dataset that you find online and use the method of enforcing sparsity to increase the performance.

Project 4. (*Quantum generative adversarial network*).

In this project you will use a Quantum Circuit Born Machine (QCBM) as a generator in a Quantum Generative Adverserial Network (QGAN). Beware that training a QGAN is challenging.

Literature:

- `https://en.wikipedia.org/wiki/Generative_adversarial_network`
- `https://arxiv.org/pdf/1804.09139.pdf`
- `https://arxiv.org/pdf/1804.08641.pdf`
- `https://pennylane.ai/qml/demos/tutorial_QGAN.html`

Objectives and tasks:

- Learn what a generative adverserial network is, i.e., how it works and how to train it.
- Implement a QCBM to take the role of the "generator" in the QGAN.
- Extend the QCBM to a QGAN by using a "discriminator" to train it.

  Remark. *You are free to choose how you implement the discriminator (i.e., quantum/classical).*

- Perform benchmarking experiments of your QGAN.

The report should contain:

1. An explanation of GANs and QGANs.
2. An implementation of a QGAN that uses a QCBM as the "generator".
3. Experimental results of the performance of the QGAN.

For an 8,5+: Compare classical and quantum discriminators or generators. That is, compare multiple configurations of quantum/classical for the discriminator and generator (e.g., use a quantum generator with both a classical and quantum discriminator and compare the results).

Project 5. (*Reinforcement learning with a parameterized quantum circuit*).

In this project you will use parameterized quantum circuit learning models in a reinforcement learning setting. Beware that we only recommend you pick this project if you are familiar and experienced with implementing a reinforcement learning agent. The main research question behind this project is finding ways to fit larger reinforcement learning environments on quantum circuits that use a (relatively) small amount of qubits.

Literature:

- `https://arxiv.org/pdf/2103.05577.pdf`

Objectives and tasks:

- Study the paper linked in the "Literature" section.
- Reproduce the implementation and results of the paper for one of the RL environments.

  Remark. *We will assist you by supplying the relevant hyperparameters.*

- Reduce the number of qubits of the parameterized quantum circuit by either,
  - using a neural network to preprocess the state into parameters of the quantum circuit,
  - or sequentially uploading parts the data (i.e., trading-off depth for #qubits).

The report should contain:

1. A discussion of the paper linked in the "Literature" section.
2. A discussion of your implementation of the RL agent that follows the same paper.
3. Experimental results on your attempts to reduce the number of qubits of the parameterized quantum circuit that is used by your RL agent.

For an 9+: Use your technique for fitting a large reinforcement learning environment on a quantum circuit with (relatively) few qubits to get a reinforcement learning agent to learn an environment with very large states (e.g., an Atari game[1]).

[1] `https://gym.openai.com/envs/#atari`