Quantum methods for Optimization and Machine Learning

Iordanis Kerenidis
Paris Centre for Quantum Computing PCQC, CNRS Paris
The **HHL algorithm** [Harrow, Hassidim, Lloyd 2009]

Quantum computers provide a quantum solution to a system of linear equations in certain cases exponentially faster than classical algorithms, given quantum access to the data.
Quantum Algorithms for Optimisation / ML

The HHL algorithm [Harrow, Hassidim, Lloyd 2009]

Quantum computers provide a quantum solution to a system of linear equations in certain cases exponentially faster than classical algorithms, given quantum access to the data.
The HHL algorithm [Harrow, Hassidim, Lloyd 2009]
Quantum computers provide a quantum solution to a system of linear equations in certain cases exponentially faster than classical algorithms, given quantum access to the data.

“It opens the possibility of dramatic speedups for machine learning tasks, richer models for data sets and more natural settings for learning and inference”

Quantum Machine Learning Workshop during NIPS 2015

Remark: “Solving” systems of linear equations is BQP-complete
Three remarks on Quantum Machine Learning

QML needs a full-scale computer with quantum access to data for “exponential savings”
Three remarks on Quantum Machine Learning

QML needs a **full-scale computer with quantum access to data** for “exponential savings”

Most **overhyped and underestimated** field at the same time
Three remarks on Quantum Machine Learning

QML needs a full-scale computer with quantum access to data for "exponential savings"

Most overhyped and underestimated field at the same time

(One of) the most convincing reasons to build quantum computers
Quantum Machine Learning: the model

Data storage and quantum access
1. Data can be accessed quantumly directly
2. Quantum RAM: Efficient storage of classical data allowing quantum access to it
   • It takes polylogarithmic time to store/update/delete an element \((i, j, a_{ij})\)
   • Query in polylogarithmic time: \(\sum c_{ij} |i, j, 0\rangle \rightarrow \sum c_{ij} |i, j, a_{ij}\rangle\)
3. Other access models...
Quantum Machine Learning: the model

Data storage and quantum access
1. Data can be accessed quantumly directly
2. Quantum RAM: Efficient storage of classical data allowing quantum access to it
   - It takes polylogarithmic time to store/update/delete an element \((i, j, a_{ij})\)
   - Query in polylogarithmic time \(\sum c_{ij} |i, j, 0\rangle \rightarrow \sum c_{ij} |i, j, a_{ij}\rangle\)
3. Other access models...

Computation on the data
- Given quantum access to data, learn some property of the data
- Running time of quantum algorithm can be more efficient than classical
Use-case: Recommendation Systems
Use-case: Recommendation Systems

Step 1

Matrix Reconstruction Sampling

Products

<table>
<thead>
<tr>
<th>?</th>
<th>1</th>
<th>...</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>0</td>
<td>...</td>
<td>?</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>?</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

Users
Use-case: Recommendation Systems

**Matrix Reconstruction**

<table>
<thead>
<tr>
<th>Products</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>1</td>
<td>...</td>
<td>?</td>
</tr>
<tr>
<td>?</td>
<td>0</td>
<td>...</td>
<td>?</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>?</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

**Step 1**

**Step 2**

**Sampling**

**Singular Value Estimation**

\[ W = U S V^T \]
Use-case: Recommendation Systems

Matrix Reconstruction Sampling

Step 1: Matrix Reconstruction

Step 2: Sampling

Step 3: Singular Value Estimation

Exponentially faster quantum algorithm for Singular Value Estimation
**Use-case: Recommendation Systems**

**Step 1:** Matrix Reconstruction

**Step 2:** Sampling

**Step 3:** Singular Value Estimation

Quantum computers provide competitive recommendations exponentially faster than known classical algorithms

[Kerenidis, Prakash '17]
Use-case: Recommendation Systems

Matrix Reconstruction

Sampling

Products

Improvised Linear Algebra for low-rank matrices

Quantum computers provide competitive recommendations exponentially faster than known classical algorithms [Kerenidis, Prakash '17]

Step 1

Users

Step 2

Step 3

Singular Value Estimation

Exponentially faster quantum algorithm for Singular Value Estimation

Improved Linear Algebra for low-rank matrices

Quantum computers provide competitive recommendations exponentially faster than known classical algorithms [Kerenidis, Prakash '17]
Use-case: Recommendation Systems

**Matrix Reconstruction**

<table>
<thead>
<tr>
<th>?</th>
<th>1</th>
<th>…</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>0</td>
<td>…</td>
<td>?</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>1</td>
<td>?</td>
<td>…</td>
<td>0</td>
</tr>
</tbody>
</table>

Users

- Step 1
- Step 2

**Sampling**

**Singular Value Estimation**

$\mathbf{W} = \mathbf{U} \mathbf{S} \mathbf{V}^T$

- Step 3

**Improved Linear Algebra for low-rank matrices**

- Quantum computers provide competitive recommendations exponentially faster than known classical algorithms [Kerenidis, Prakash '17]

- Running time: $O(r \times \text{polylog}(mn))$

Exponentially faster quantum algorithm for Singular Value Estimation
General quantum methods for Optimization

Iterative methods (ubiquitous in practice)
1. Start with an initial solution.
2. Update the solution according to an Update Rule
3. Repeat until the solution is satisfactory

Types of Iterative Methods
First order – Gradient Descent
Second order – Interior point methods
General quantum methods for Optimization

**Iterative methods** (ubiquitous in practice)
1. Start with an initial solution.
2. Update the solution according to an Update Rule.
3. Repeat until the solution is satisfactory.

**Types of Iterative Methods**
- First order – Gradient Descent
- Second order – Interior point methods

Efficient Quantum Gradient Descent algorithm for Linear Systems and Stochastic Least Squares.
[Kerenidis, Prakash 2017]
General quantum methods for Optimization

**Iterative methods** (ubiquitous in practice)
1. Start with an initial solution.
2. Update the solution according to an Update Rule
3. Repeat until the solution is satisfactory

**Types of Iterative Methods**
- First order – Gradient Descent
- Second order – Interior point methods

Efficient Quantum Gradient Descent algorithm for Linear Systems and Stochastic Least Squares. [Kerenidis, Prakash 2017]

**Remark 1:** Improved Linear Algebra
**Remark 2:** Great savings in QRAM
Problem:
Given matrix $A$ and vector $x$, output $Ax$, $A^{-1}x$, ...
Quantum Linear Algebra

Problem:
Given matrix $A$ and vector $x$, output $Ax, A^{-1}x, ...$

Step 1
Map $A$ to some unitary $U$ s.t.
1. The spectra of $A$ and $U$ are related
2. $U$ is efficient to implement

$A/\mu(A) = P \circ Q$, $U = (2PP^t - I)(2QQ^t - I)$

Efficiency via QRAM data structures
Quantum Linear Algebra

**Problem:**
Given matrix $A$ and vector $x$, output $Ax$, $A^{-1}x$, ...

**Step 1**
Map $A$ to some unitary $U$ s.t.
1. The spectra of $A$ and $U$ are related
2. $U$ is efficient to implement

$$A/\mu(A) = P \circ Q, \quad U = (2PP^t - I)(2QQ^t - I)$$

Efficiency via QRAM data structures

**Step 2**
Phase Estimation on $U$
LCU, Qubitization on $U$

Apply a circuit with $O(\log 1/\epsilon)$ U’s
Problem:
Given matrix $A$ and vector $x$, output $Ax$, $A^{-1}x$, ...

Step 1
Map $A$ to some unitary $U$ s.t.
1. The spectra of $A$ and $U$ are related
2. $U$ is efficient to implement

Step 2
Phase Estimation on $U$
LCU, Qubitization on $U$

Step 3
Amplitude Amplification (VT)

$A/\mu(A) = P \circ Q$, $U = (2PP^t - I)(2QQ^t - I)$

Efficiency via QRAM data structures

Apply a circuit with $O(\log 1/\varepsilon)$ $U$’s

$O(\kappa(A))$ iterations
Quantum Linear Algebra

**Problem:**
Given matrix $A$ and vector $x$, output $Ax$, $A^{-1}x$, ...

**Running time:** $O(\kappa(A)\mu(A)\log 1/\varepsilon)$
Quantum Linear Algebra

**Problem:**
Given matrix $A$ and vector $x$, output $Ax$, $A^{-1}x$, ...

**Running time:** $O(\kappa(A)\mu(A)\log 1/\varepsilon)$

**Open Question:**
What is the optimal $\mu(A)$?
Could QML work on real data?
Could QML work on real data?

**Frobenius Distance Classification**

**QFE 4 Frobenius Distance Estimator**

Require:
QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

Ensure:
An estimate $\bar{F}_k(x(0))$ such that $|F_k(x(0)) - \bar{F}_k(x(0))| < \eta$.

$$F_k(x(0)) = \frac{\|X_k - X(0)\|_F^2}{2(\|X_k\|_F^2 + \|X(0)\|_F^2)}$$
Could QML work on real data?

### Frobenius Distance Classification

**QFL 4 Frobenius Distance Estimator**

**Require:**
- QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

**Ensure:**
- An estimate $\hat{F}_k(x(0))$ such that $|F_k(x(0)) - \hat{F}_k(x(0))| < \eta$.

1. Create the state

$$
\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} \|x(0)\| |i\rangle |x(0)\rangle + |1\rangle \sum_{i \in T_k} \|x(i)\| |i\rangle |x(i)\rangle \right)
$$
Could QML work on real data?

Frobenius Distance Classification

**Frobenius Distance Estimator**

**Require:**
- QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

**Ensure:**
An estimate $\tilde{F}_k(x(0))$ such that $|F_k(x(0)) - \tilde{F}_k(x(0))| < \eta$.

1. Create the state

$$\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} \|x(0)\| |i\rangle |x(0)\rangle + |1\rangle \sum_{i \in T_k} \|x(i)\| |i\rangle |x(i)\rangle \right)$$

2. Apply a Hadamard

$$\frac{1}{\sqrt{2N_k}} \left[ |0\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle + \|x(i)\| |i\rangle |x(i)\rangle \right) + \frac{1}{\sqrt{2N_k}} \left| 1 \right\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle - \|x(i)\| |i\rangle |x(i)\rangle \right) \right]$$

**Classification**

[KL 18]
Could QML work on real data?

**Frobenius Distance Classification**

**FQE 4 Frobenius Distance Estimator**

Require: QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

Ensure: An estimate $\hat{F}_k(x(0))$ such that $|F_k(x(0)) - \hat{F}_k(x(0))| < \eta$.

$$F_k(x(0)) = \frac{\|X_k - X(0)\|_F^2}{2(\|X_k\|_F^2 + \|X(0)\|_F^2)}$$

1. Create the state

$$\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} \|x(0)\| |i\rangle |x(0)\rangle + |1\rangle \sum_{i \in T_k} \|x(i)\| |i\rangle |x(i)\rangle \right)$$

2. Apply a Hadamard

$$\frac{1}{\sqrt{2N_k}} \left( |0\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle + \|x(i)\| |i\rangle |x(i)\rangle \right) + \frac{1}{\sqrt{2N_k}} \left( |1\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle - \|x(i)\| |i\rangle |x(i)\rangle \right) \right)$$

3. Repeat and Estimate Prob[ outcome 1] = $F_k(x(0))$
Could QML work on real data?

**Frobenius Distance Classification**

**Frobenius Distance Estimator**

**Require:**
QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

**Ensure:**
An estimate $\bar{F}_k(x(0))$ such that $|F_k(x(0)) - \bar{F}_k(x(0))| < \eta$.

1. Create the state

   $$\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} \|x(0)\| |i\rangle |i\rangle |x(0)\rangle + \frac{1}{\sqrt{2N_k}} \left( |1\rangle \sum_{i \in T_k} \|x(i)\| |i\rangle |x(i)\rangle + \|x(i)\| |i\rangle |x(0)\rangle \right) \right)$$

2. Apply a Hadamard

   $$\frac{1}{\sqrt{2N_k}} \left( |0\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle + \|x(i)\| |i\rangle |x(i)\rangle \right) \right) + \frac{1}{\sqrt{2N_k}} |1\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle - \|x(i)\| |i\rangle |x(i)\rangle \right)$$

3. Repeat and Estimate $\text{Prob}[\text{outcome } 1] = F_k(x(0))$

4. Assign $x(0)$ to the closest cluster
Could QML work on real data?

**Frobenius Distance Classification**

**QFE 4 Frobenius Distance Estimator**

**Require:**
QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

**Ensure:**
An estimate $\hat{F}_k(x(0))$ such that $|F_k(x(0)) - \hat{F}_k(x(0))| < \eta$.

1. **Create the state**

$$
\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} \|x(0)\| \ |i\rangle \ |x(0)\rangle + |1\rangle \sum_{i \in T_k} \|x(i)\| \ |i\rangle \ |x(i)\rangle \right)
$$

2. **Apply a Hadamard**

$$
\frac{1}{\sqrt{2N_k}} \left( |0\rangle \sum_{i \in T_k} \left( \|x(0)\| \ |i\rangle \ |x(0)\rangle + \|x(i)\| \ |i\rangle \ |x(i)\rangle \right) + \frac{1}{\sqrt{2N_k}} |1\rangle \sum_{i \in T_k} \left( \|x(0)\| \ |i\rangle \ |x(0)\rangle - \|x(i)\| \ |i\rangle \ |x(i)\rangle \right) \right)
$$

3. **Repeat and Estimate**

$\text{Prob}[\text{outcome 1}] = F_k(x(0))$

4. **Assign** $x(0)$ to the closest cluster

**Remark 1:** Classification as easy as creating the states
Could QML work on real data?

Frobenius Distance Classification

**FQE 4 Frobenius Distance Estimator**

Require:
QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

Ensure:
An estimate $F_k(x(0))$ such that $|F_k(x(0)) - \bar{F}_k(x(0))| < \eta$.

1. Create the state
   $$\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} \|x(0)\| |i\rangle |x(0)\rangle + |1\rangle \sum_{i \in T_k} \|x(i)\| |i\rangle |x(i)\rangle \right)$$

2. Apply a Hadamard
   $$\frac{1}{\sqrt{2N_k}} |0\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle + \|x(i)\| |i\rangle |x(i)\rangle \right) + \frac{1}{\sqrt{2N_k}} |1\rangle \sum_{i \in T_k} \left( \|x(0)\| |i\rangle |x(0)\rangle - \|x(i)\| |i\rangle |x(i)\rangle \right)$$

3. Repeat and Estimate $\text{Prob}[\text{outcome } 1] = F_k(x(0))$

4. Assign $x(0)$ to the closest cluster

Remark 2:
Comparable accuracy to classical classifiers

**Classification**

[KL 18]
Could QML work on real data?

Frobenius Distance Classification

**Frobenius Distance Estimator**

**Require:**
QRAM access to the matrix $X_k$ of cluster $k$ and to a test vector $x(0)$. Error parameter $\eta > 0$.

**Ensure:**
An estimate $\tilde{F}_k(x(0))$ such that $|F_k(x(0)) - \tilde{F}_k(x(0))| < \eta$.

1. **Create the state**

$$\frac{1}{\sqrt{N_k}} \left( |0\rangle \sum_{i \in T_k} ||x(0)|| |i\rangle |x(0)\rangle + |1\rangle \sum_{i \in T_k} ||x(i)|| |i\rangle |x(i)\rangle \right)$$

2. **Apply a Hadamard**

$$\frac{1}{\sqrt{2N_k}} |0\rangle \sum_{i \in T_k} \left( ||x(0)|| |i\rangle |x(0)\rangle + ||x(i)|| |i\rangle |x(i)\rangle \right) + \frac{1}{\sqrt{2N_k}} |1\rangle \sum_{i \in T_k} \left( ||x(0)|| |i\rangle |x(0)\rangle - ||x(i)|| |i\rangle |x(i)\rangle \right)$$

3. **Repeat and Estimate**

$\text{Prob[ outcome 1]} = F_k(x(0))$

4. **Assign $x(0)$ to the closest cluster**

Remark 2:
Comparable accuracy to classical classifiers

**Accuracy**
67.5%

Classification

[KL 18]
Could QML work on real data?

Dimensionality Reduction: Slow Feature Analysis

Classification

[KL 18]
Could QML work on real data?

**Dimensionality Reduction: Slow Feature Analysis**

### SFA - Algorithm 1 (Classical) Slow Feature Analysis

**Require:**
- Input $X \in \mathbb{R}^{n \times d}$ (normalized and polynomially expanded), and $K < d \in \mathbb{N}$

**Ensure:**
- $Y = ZW$, where $Z = XB^{-1/2}$ is the whitened input signal, and $W \in \mathbb{R}^{d \times (K-1)}$ are the $K-1$ eigenvectors of the matrix $A = \hat{Z}^T\hat{Z}$ corresponding to the smallest eigenvalues

1. Whiten the signal: $Z := XB^{-1/2}$, and create $\hat{Z}$ from $Z$.
2. Perform PCA on the derivative covariance matrix $A = \hat{Z}^T\hat{Z}$ of the whitened data.
3. Return $Y = ZW$, the projection of whitened data onto $W$, the $K-1$ slowest eigenvectors of $A$
Could QML work on real data?

Dimensionality Reduction: Slow Feature Analysis

**SFA - Algorithm 1**

**Require:**
- Input $X \in \mathbb{R}^{n \times d}$

**Ensure:**
- $Y = ZW$, where $Z$ is the whitened input signal, and $W$ are the eigenvectors of $A^{-1}$

1. Whiten the signal $X$.
2. Perform PCA on the derivative covariance matrix.
3. Return $Y = ZW$.

Classification

[KL 18]
Could QML work on real data?

**Dimensionality Reduction: Slow Feature Analysis**

**SFA - Algorithm 1 (Classical) Slow Feature Analysis**

**Require:**
Input $X \in \mathbb{R}^{n \times d}$ (normalized and polynomially expanded), and $K < d \in \mathbb{N}$

**Ensure:**
$Y = ZW$, where $Z = XB^{-1/2}$ is the whitened input signal, and $W \in \mathbb{R}^{d \times (K-1)}$ are the $K - 1$ eigenvectors of the matrix $A = \hat{Z}^T \hat{Z}$ corresponding to the smallest eigenvalues

1: Whiten the signal: $Z := XB^{-1/2}$, and create $\hat{Z}$ from $Z$.
2: Perform PCA on the derivative covariance matrix $A = \hat{Z}^T \hat{Z}$ of the whitened data.
3: Return $Y = ZW$, the projection of whitened data onto $W$, the $K - 1$ slowest eigenvectors of $A$
Could QML work on real data?

**SFA - Algorithm 1** (Classical) Slow Feature Analysis

**Require:**
Input $X \in \mathbb{R}^{n \times d}$ (normalized and polynomially expanded), and $K < d \in \mathbb{N}$

**Ensure:**
$Y = ZW$, where $Z = XB^{-1/2}$ is the whitened input signal, and $W \in \mathbb{R}^{d \times (K-1)}$ are the $K-1$ eigenvectors of the matrix $A = \hat{Z}^T \hat{Z}$ corresponding to the smallest eigenvalues

1. Whiten the signal: $Z := XB^{-1/2}$, and create $\hat{Z}$ from $Z$.
2. Perform PCA on the derivative covariance matrix $A = \hat{Z}^T \hat{Z}$ of the whitened data.
3. Return $Y = ZW$, the projection of whitened data onto $W$, the $K-1$ slowest eigenvectors of $A$

**Quantum Slow Feature Analysis**

Efficient Quantum Linear Algebra  
(Matrix Multiplication, Inversion, Projection)
Could QML work on real data?

**Dimensionality Reduction: Slow Feature Analysis**

**SFA - Algorithm 1** (Classical) Slow Feature Analysis

Require:
Input $X \in \mathbb{R}^{n \times d}$ (normalized and polynomially expanded), and $K < d \in \mathbb{N}$

Ensure:
$Y = ZW$, where $Z = XB^{-1/2}$ is the whitened input signal, and $W \in \mathbb{R}^{d \times (K-1)}$ are the $K - 1$ eigenvectors of the matrix $A = \hat{Z}^T \hat{Z}$ corresponding to the smallest eigenvalues

1: Whiten the signal: $Z := XB^{-1/2}$, and create $\hat{Z}$ from $Z$.
2: Perform PCA on the derivative covariance matrix $A = \hat{Z}^T \hat{Z}$ of the whitened data.
3: Return $Y = ZW$, the projection of whitened data onto $W$, the $K - 1$ slowest eigenvectors of $A$

**Quantum Slow Feature Analysis**

**Efficient Quantum Linear Algebra**
(Matix Multiplication, Inversion, Projection)

**Remark:**
Classification only needs quantum states
Could QML work on real data?

Quantum Classifier

**Input:** X, a new vector x(0)

1. Do QSFA to quantumly project X and x(0) to Y and y(0)
2. Use Frobenius Distance Classification on Y, y(0)
Could QML work on real data?

Quantum Classifier

Input: X, a new vector x(0)
1. Do QSFA to quantumly project X and x(0) to Y and y(0)
2. Use Frobenius Distance Classification on Y, y(0)

Accuracy

We simulate the quantum procedures including errors in an HPC machine and test it on the 10000 test digits of MNIST for different parameters

Accuracy 98.5%
Could QML work on real data?

**Quantum Classifier**

**Input:** X, a new vector x(0)
1. Do QSFA to quantumly project X and x(0) to Y and y(0)
2. Use Frobenius Distance Classification on Y, y(0)

**Accuracy**

We simulate the quantum procedures including errors in an HPC machine and test it on the 10000 test digits of MNIST for different parameters

**Running time**

**Classical:** $O(n d^2) \approx 10^{13}$ (1 hour on 6Tb RAM HPC)

**Classification**

Accuracy 98.5%

[KL 18]
Could QML work on real data?

**Quantum Classifier**

Input: X, a new vector x(0)
1. Do QSFA to quantumly project X and x(0) to Y and y(0)
2. Use Frobenius Distance Classification on Y, y(0)

**Accuracy**

We simulate the quantum procedures including errors in an HPC machine and test it on the 10000 test digits of MNIST for different parameters

**Running time**

Classical: O(n d^2) \(\approx\) 10^{13} (1 hour on 6Tb RAM HPC)
Quantum: O(\kappa, \mu, 1/\theta, 1/\delta, 1/\eta, K, \text{polylog}(n, d, 1/\varepsilon),...)

Accuracy 98.5%
Could QML work on real data?

**Quantum Classifier**

**Input:** X, a new vector x(0)
1. Do QSFA to quantumly project X and x(0) to Y and y(0)
2. Use Frobenius Distance Classification on Y, y(0)

**Accuracy**

We simulate the quantum procedures including errors in an HPC machine and test it on the 10000 test digits of MNIST for different parameters.

**Running time**

**Classical:** $O(n \, d^2) \sim 10^{13}$ (1 hour on 6Tb RAM HPC)
**Quantum:** $O(\kappa, \mu, 1/\theta, 1/\delta, 1/\eta, K, \text{polylog}(n, d, 1/\epsilon),...) \sim 10^7$

**Classification**

Accuracy 98.5%
Could QML work on real data?

**Main question:**
Better accuracy by increasing the dimension, keeping efficient time?
Quantum time: $O(\kappa, \mu, 1/\theta, 1/\delta, 1/\eta, K, \text{polylog}(n, d, 1/\epsilon),...$)
Could QML work on real data?

Main question:
Better accuracy by increasing the dimension, keeping efficient time?
Quantum time: $O(\kappa, \mu, 1/\theta, 1/\delta, 1/\eta, K, \text{polylog}(n, d, 1/\epsilon), ...)$
Could QML work on real data?

Main question:
Better accuracy by increasing the dimension, keeping efficient time?
Quantum time: $O(\kappa, \mu, 1/\theta, 1/\delta, 1/\eta, K, \text{polylog}(n, d, 1/\varepsilon),...)$
Could QML work on real data?

Main question:
Better accuracy by increasing the dimension, keeping efficient time?
Quantum time: $O(\kappa, \mu, 1/\theta, 1/\delta, 1/\eta, K, \text{polylog}(n, d, 1/\epsilon),...)$

Hope (and some evidence):
Quantum classification algorithms can handle bigger dimensions (hence be more accurate), since their running time scales much more favourably with the dimension.

Classification
[KL 18]
Unsupervised Classification: **Q-means** [KLLP 18]

**K-means**

**Input:** M N-dimensional points, K clusters

1. Start with some random points as centroids
   Repeat until convergence
   2. For each point compute distances to the centroids and assign to closest cluster \(O(KMN)\)
   3. Recompute the centroids \(O(MN)\)
Unsupervised Classification: Q-means [KLLP 18]

K-means
Input: M N-dimensional points, K clusters

1. Start with some random points as centroids
   Repeat until convergence
   2. For each point compute distances to the centroids and assign to closest cluster $O(KMN)$
   3. Recompute the centroids $O(MN)$

Q-means
Input: M N-dimensional points with quantum access, K clusters

1. Start with some random points as centroids
   Repeat until convergence
   2. For all points in superposition compute distances to centroids and assign to closest cluster $O(K \log(MN))$
   2. Use Matrix Multiplication and tomography to recompute the centroids $O(KN \log(MN))$
Summary and open questions

Summary
- QML is (one of) the best reason to build quantum computers
  - Use case: Quantum recommendation systems
  - General Methods: Quantum gradient descent for linear gradients
  - Benchmarking: Classification of MNIST dataset
  - ML data has some hidden structure (e.g. low rank approximations)
  - ML is very robust to errors

Open Questions
- Build quantum computers and QRAMs
- Find new quantum methods (Interior point methods, fully quantum methods,...)
- Find more real-world applications
- Benchmark hardware via applications