Quantum-enhanced machine learning: Implementing a quantum *k*-nearest neighbour algorithm

Bachelor thesis defense

19. January 2017

Mark Fingerhuth

Supervisors: Dr. Fabrice Birembaut, Prof. Francesco Petruccione

Maastricht Science Programme, Maastricht University, The Netherlands
Thesis work at the Centre for Quantum Technology, University of KwaZulu-Natal
Durban, South Africa

Maastricht University

UNIVERSITY OF KWAZULU-NATAL

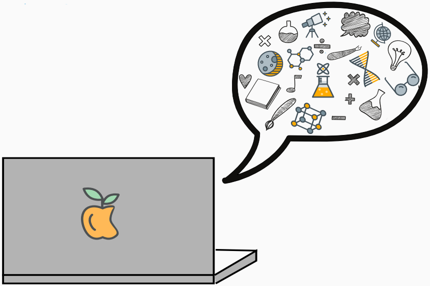
CENTRE FOR QUANTUM TECHNOLOGY

Table of contents

1. Introduction
2. Machine Learning
3. Quantum Computing
4. Methods
5. Quantum-enhanced Machine Learning
6. Results: Amplitude-based quantum k-nearest neighbour algorithm
7. Conclusion
8. Outlook

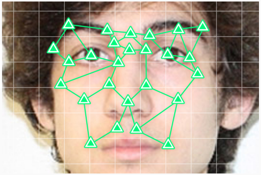
Introduction

Enhancing machine learning with quantum mechanics



Machine learning

Enable computers to learn from data



Source: IEEE Spectrum

- ML algorithms often involve[1]
  - solving large systems of linear equations
  - inverting large matrices
  - distance computations
- Performing these computations on large data sets gets increasingly difficult[2]

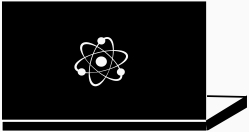
[1] Bishop, C. M. (2006). Pattern recognition. Machine Learning, 128 .

[2] Bekkerman, R., Bilenko, M., & Langford, J. (2011). Scaling up machine learning: Parallel and distributed approaches. Cambridge University Press.

Enhancing machine learning with quantum mechanics

Sources: IBM

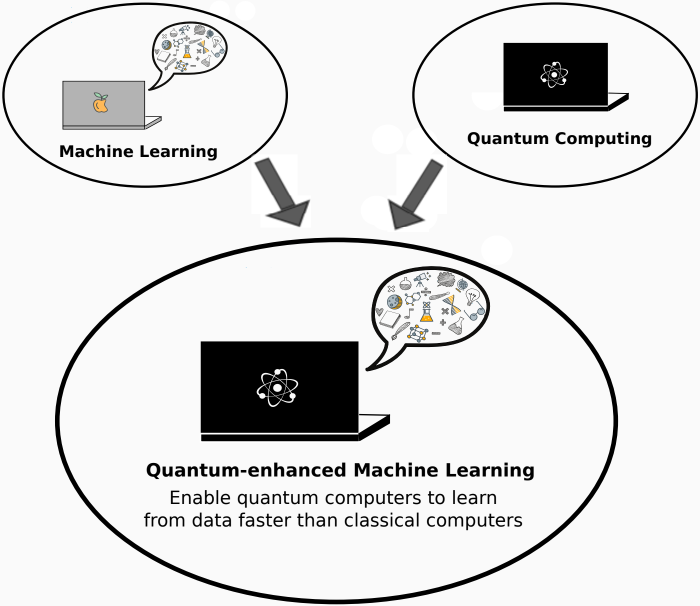
- Quantum mechanics is about vectors in complex Hilbert spaces
- Quantum computers are performing linear operations on qubits
- Many-qubit systems are described by large vectors that can be manipulated in parallel on quantum computers
- Machine learning involves manipulation of large vectors and matrices



Quantum computing

Build computer hardware based on quantum physics

Enhancing machine learning with quantum mechanics

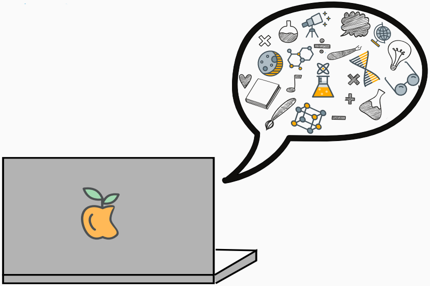


Machine Learning

Quantum Computing

Quantum-enhanced Machine Learning

Enable quantum computers to learn
from data faster than classical computers



Machine Learning

Supervised machine learning

The problem statement

Given a dataset of inputs and their corresponding outputs,
predict the output of a new unknown input.

Input	Output
faces	emotions
heartbeat	healthy or sick
last year's daily weather	tomorrow's weather
message of a users	intention of text content
search history of a user	chance of clicking on a particular ad

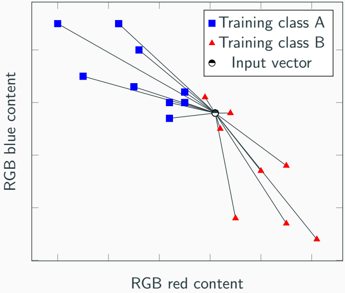
Classical k-nearest neighbour with distance-weighting

- k is a positive integer

Given training dataset:
$D_T = v_0, v_1, .., v_{16}$
$v_i \in \{red, blue\}$

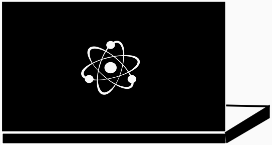
Given a new vector $\tilde{x}$ (black halfcircle):
- consider k nearest neighbours
- classify $\tilde{x}$, based on majority vote, as *red* or *blue*

$k = all$



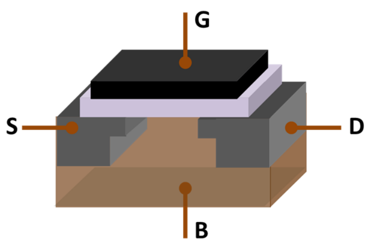
Assign distance-dependent weights e.g. $\frac{1}{\text{distance}}$ to increase the influence of close vectors over more distant ones!

Classification → **RED**



Quantum Computing

Classical vs. quantum bits (qubits)



Source: Wikipedia

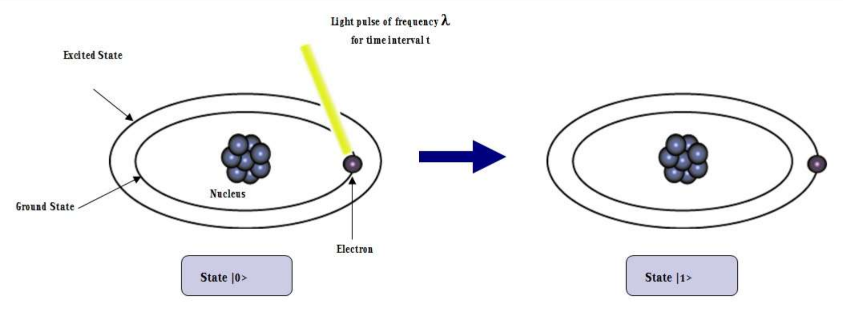
Classical bit:

- Usually implemented through MOSFETs
- 2 definite states (0,1)
- Can be either 0 OR 1

Classical vs. quantum bits (qubits)

Quantum bit (qubit):

- Can be $|0\rangle$ OR $|1\rangle$
- But it can also be $|0\rangle$ AND $|1\rangle$ → quantum superposition



Source: RF Wireless World

A qubit

Mathematically, the superposition of a qubit is expressed as:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \doteq \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \tag{1}$$

where $\alpha, \beta \in \mathbb{C}$ and they are called amplitudes.

The last expression is called the **amplitude vector**.

The Bloch sphere

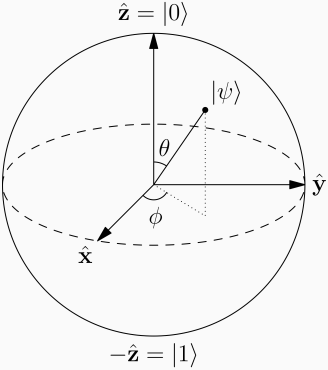


Figure 1: Arbitrary two-dimensional qubit $|\psi\rangle$ visualized on the Bloch sphere[1]

Most general form of a 2-D qubit:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \;, \qquad (2)$$

where $\alpha, \beta \in \mathbb{C}$.

Can also be visualized in spherical polar coords on the unit or Bloch sphere as follows:

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle \;, \quad (3)$$

where $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$

[1] Reprinted from Wikipedia, n.d., Retrieved September 7, 2016, from https://en.wikipedia.org/wiki/Bloch_Sphere. Copyright 2012 by Glosser.ca. Reprinted with permission.

Multi-qubit systems: the power of quantum computing

A quantum computer with n qubits has 2^n quantum amplitudes.
$\rightarrow$ these amplitudes can be used to store huge amounts of information!

Qubit number	classical RAM needed	Simulation time
5	256 bytes	microseconds on smartphone
25	2 gigabytes	seconds on a laptop
50	8000 terabytes	seconds on next year's supercomputer
275	number of atoms in the visible universe	age of the universe

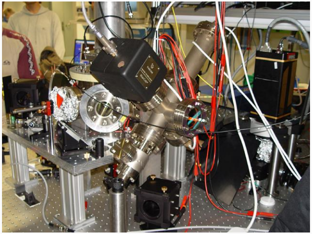
State-of-the-art quantum computation

However, quantum computation is cutting edge research at the frontier of supercomputing technology!

→ State-of-the art quantum computers are complicated laboratory experiments.

Current state-of-the-art:

- IBM → **5 superconducting qubits**
- Google → **9 superconducting qubits**
- Weizmann Research Group in Israel → **8 trapped ions**
- DWave → **1,152 qubits** BUT solves only a narrow class of problems



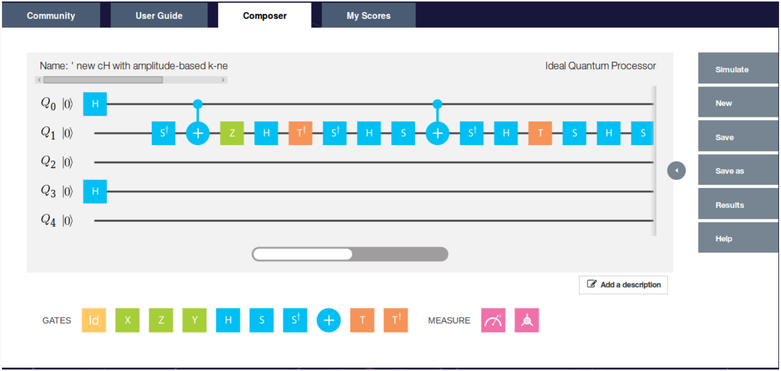
Source: University of Innsbruck

Motivation for this thesis research

- Classical machine learning is a very applied field
- State-of-the-art quantum computer have very small numbers of qubits
- Thus, quantum-enhanced machine learning is almost purely theoretical
- There have been only a handful of proof-of-principle studies in quantum-enhanced machine learning
- Need for more proof-of-principle implementations and simulations to demonstrate the benefits of quantum computation for machine learning
- Small machine learning problems need to be identified and implemented.

Methods

Methods: IBM Quantum Experience



Name: ' new cH with amplitude-based k-ne

Ideal Quantum Processor

Figure 2: IBM's quantum composer[1]

- Accessible to the public
- Allows for ideal + real simulations
- 5 superconducting qubits
- 40 gates (39 gates + 1 measurement)

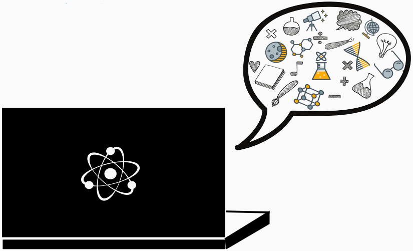
[1] Screenshot taken from `https://quantumexperience.ng.bluemix.net/qstage/#/editor`

Methods: Liqui$|\rangle$

Small quantum computers can be simulated on conventional classical computers!

Liqui$|\rangle$...

- stands for Language-Integrated Quantum Operations.
- is a quantum simulation toolsuite written in F# and developed by Microsoft Research.
- allows for simulations of up to 30 qubits with 16GB RAM.
- was used in this thesis to provide proof-of-principle simulations of quantum-enhanced machine learning algorithms.



Quantum-enhanced Machine Learning

Encoding classical data into amplitudes

Data encoded into amplitudes

k-dimensional probability vector is encoded into $log_2(k)$ qubits, e.g.

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} \quad \rightarrow \quad |n\rangle = \sqrt{0.6}\,|0\rangle + \sqrt{0.4}\,|1\rangle$$

Schuld, Fingerhuth, and Petruccione (Manuscript in preparation) developed a new **amplitude-based** kNN algorithm.
→ requires only a few qubits and provides great speed-up

[1]

The amplitude-based kNN algorithm

1. $|\psi_0\rangle = \frac{1}{\sqrt{2M}} \sum_{m=1}^{M} (|0\rangle |\Psi_{\tilde{x}(\star)}\rangle + |1\rangle |\Psi_{x^m}\rangle) |y^m\rangle |m\rangle$
   [Initial quantum state]

2. $|\psi_1\rangle = \frac{1}{2\sqrt{M}} \sum_{m=1}^{M} (|0\rangle [|\Psi_{\tilde{x}}\rangle + |\Psi_{x^m}\rangle] + |1\rangle [|\Psi_{\tilde{x}}\rangle - |\Psi_{x^m}\rangle]) |y^m\rangle |m\rangle$
   [Distance computations with quantum interference]

3. $|\psi_2\rangle = \frac{1}{2\sqrt{M}} \sum_{m=1}^{M} \sum_{i=1}^{N} (\tilde{x}_i + x_i^m) |0\rangle |i\rangle |y^m\rangle |m\rangle$
   [Conditional measurement]

4. $\mathrm{Prob}(|y^m\rangle = |1\rangle) = \sum_{m|y^m=1} 1 - \frac{1}{4M} \mid \tilde{x} - x^m \mid^2$
   [Probability to measure a certain class]

5. $y = \begin{cases} 0, & \text{if } \mathrm{Prob}(|y^0\rangle) > \mathrm{Prob}(|y^1\rangle) \\ 1, & \text{if } \mathrm{Prob}(|y^1\rangle) > \mathrm{Prob}(|y^0\rangle) \\ -, & \text{otherwise} \end{cases}$ [Classification]

Schuld, Fingerhuth, and Petruccione (Manuscript in preparation) Amplitude-based quantum k-nearest neighbour algorithm.

Calculating distances with interference

Source: TutorVista

Results: Amplitude-based kNN algorithm

Simple binary classification case

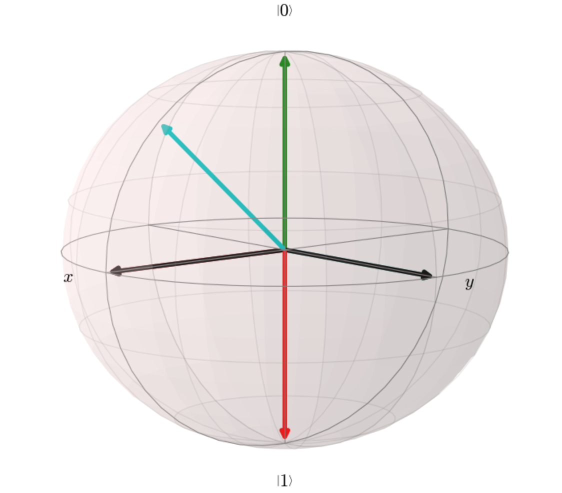


Figure 3: Simple binary classification problem of a quantum state

IBM's universal gate set

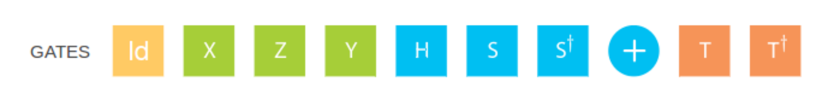


Figure 4: IBM's universal gate set

How can we implement the amplitude-based quantum kNN algorithm with this small gate set?

Results: IBM quantum computer implementation

- IBM's short qubit lifetimes only allow for 40 quantum gates.
- IBM Quantum Experience has a very small universal gate set making it difficult to run complicated algorithms.
- For this particular classification problem the quantum kNN algorithm requires at least 55 quantum gates.
- Until now, it is **impossible** to solve this particular machine learning problem on IBM's actual quantum hardware!

$\rightarrow$ Need for proof-of-principle simulations with Liqui$|\rangle$ to show that the quantum algorithms work!

Results: Liqui$|\rangle$ simulations of amplitude-based kNN algorithm

- In Liqui$|\rangle$, one can simply define any quantum logic gate.
- Large universal gate set possible!
- No limit on the number of quantum gate slots.
- No need to worry about qubit lifetimes.

Simulations demonstrated 100% accuracy on the small-scale simple Bloch vector classification problem.

→ The amplitude-based kNN algorithm does work as expected and is scalable!

Conclusion

Summary

- Quantum computing is at the frontier of supercomputing and bears the potential to vastly speed up classical machine learning algorithms
- A small-scale machine learning problem was selected for implementation and simulation of an amplitude-based quantum kNN algorithm
- The Bloch vector classification task could not be implemented with the IBM Quantum Experience
- Liqui$|\rangle$ simulations demonstrated 100% classification accuracy on Bloch vector classification task
- Open problem: How to encode arbitrary classical data into quantum amplitude distributions?

Outlook

- Finding more small-scale machine learning problems that can already be solved with quantum-enhanced machine learning algorithms
- Last week, IBM has released the IBM Quantum Experience 2.0 which makes an actual proof-of-principle implementation feasible. (Publication in preparation)
- Possible collaboration with the Weizmann Research Group in Israel to implement the amplitude-based kNN algorithm in their ion trap quantum computer.

References

Bekkerman, R., Bilenko, M., & Langford, J. (2011). Scaling up machine learning: Parallel and distributed approaches. Cambridge University Press.

Booth Jr, J. (2012). Quantum compiler optimizations. arXiv preprint arXiv:1206.3348.

Dawson, C. M., & Nielsen, M. A. (2005). The Solovay-Kitaev algorithm. arXiv preprint quant-ph/0505030.

IBM. (2016). What is big data? https://www-01.ibm.com/software/data/bigdata/what-is-big -data.html. (Accessed: 2016-09-08)

Jat, R. N., & Ruhela, D. S. (2011). Comparative study of complexity of algorithms for iterative solution of non-linear equations. Journal of International Academy Of Physical Sciences, 15(4).

Nielsen, M. A., & Chuang, I. L. (2010). Quantum computation and quantum information. Cambridge University Press.

Questions?

Results: Qubit-based kNN algorithm

[1]

Quantum Computing & Qubits

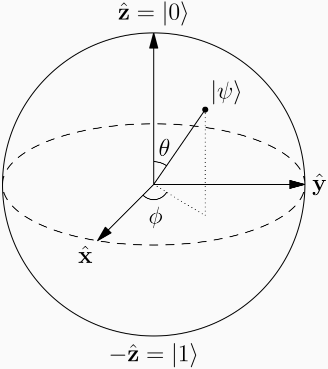


Figure 5: Arbitrary two-dimensional qubit $|\psi\rangle$ visualized on the Bloch sphere[1]

Most general form of a 2-D qubit:

$$|q\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad (4)$$

where $\alpha, \beta \in \mathbb{C}$.

Can also be visualized in spherical polar coords on the unit or Bloch sphere as follows:

$$|q\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle \qquad (5)$$

where $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$

[1] Reprinted from Wikipedia, n.d., Retrieved September 7, 2016, from https://en.wikipedia.org/wiki/Bloch_Sphere. Copyright 2012 by Glosser.ca. Reprinted with permission.

Machine Learning

- Approximately 2.5 quintillion (10^{18}) bytes of digital data are created every day[1]
- Need for advanced algorithms that can make sense of data content, retrieve patterns and reveal correlations → Machine learning (ML)
- ML algorithms often involve
  - solving large systems of linear equations
  - inverting large matrices
  - distance computations
- Performing these computations on large data sets gets increasingly difficult[2]

[1] IBM. (2016). What is big data? https://www-01.ibm.com/software/data/bigdata/what-is-big -data.html. (Accessed: 2016-09-08)

[2] Bekkerman, R., Bilenko, M., & Langford, J. (2011). Scaling up machine learning: Parallel and distributed approaches. Cambridge University Press.

Quantum Machine Learning

1. ML involves manipulation of large vectors and matrices
2. Quantum mechanics is about vectors $\in$ complex Hilbert spaces
3. Quantum computers are performing linear operations on qubits

$\rightarrow$ Hence, we can manipulate large vectors in parallel on quantum computers

So can we use QC to improve classical ML algorithms??

- Classical ML is a very practical topic
- BUT, QML has been of almost entirely theoretical nature

Quantum data encoding

There are two fundamentally different ways for state preparation:

Data encoded into qubits

k-dimensional probability vector requires $4k$ classical bits which are encoded one-to-one into $4k$ qubits, e.g.

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} * 10 \rightarrow \begin{pmatrix} 6 \\ 4 \end{pmatrix} \rightarrow \begin{pmatrix} 0110 \\ 0100 \end{pmatrix} \rightarrow n = 01100100 \rightarrow |n\rangle = |01100100\rangle$$

Data encoded into amplitudes

k-dimensional probability vector is encoded into $log_2(k)$ qubits, e.g.

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} \quad \rightarrow \quad |n\rangle = \sqrt{0.6}\,|0\rangle + \sqrt{0.4}\,|1\rangle$$

Quantum data encoding

There are two fundamentally different ways for state preparation:

Data encoded into qubits

k-dimensional probability vector requires $4k$ classical bits which are encoded one-to-one into $4k$ qubits, e.g.

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} * 10 \rightarrow \begin{pmatrix} 6 \\ 4 \end{pmatrix} \rightarrow \begin{pmatrix} 0110 \\ 0100 \end{pmatrix} \rightarrow n = 01100100 \rightarrow |n\rangle = |01100100\rangle$$

Data encoded into amplitudes

k-dimensional probability vector is encoded into $log_2(k)$ qubits, e.g.

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} \quad \rightarrow \quad |n\rangle = \sqrt{0.6}\,|0\rangle + \sqrt{0.4}\,|1\rangle$$

Classical k-nearest neighbour

- kNN is a non-parametric classifier
- k is a positive integer, usually chosen small

Given training data set:

$D_T = v_0, v_1, .., v_{10}$

$v_i \in \{A, B\}$

Given a new vector $\tilde{x}$ (red star):

- consider k nearest neighbours
- classify $\tilde{x}$, based on majority vote, as A or B

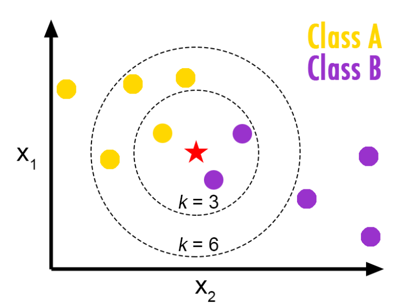


Figure 6: Visualization of a kNN classifier[1]

[1] Reprinted from GitHub, Burton de Wilde, Retrieved September 13, 2016, from http://bdewilde.github.io/blog/blogger/2012/10/26/classification-of-hand-written-digits-3/. Copyright 2012 by Burton de Wilde. Reprinted with permission.

The algorithm

$$\frac{1}{\sqrt{2M}}\sum_{m=1}^{M}\left(|0\rangle\,|\Psi_{\tilde{x}}(\star)\rangle + |1\rangle\,|\Psi_{x^m}\rangle\right)|y^m(A \text{ or } B)\rangle\,|m\rangle \tag{6}$$

where

$$|\Psi_{\tilde{x}}(\star)\rangle = \sum_{i=1}^{N}\tilde{x}_i\,|i\rangle \qquad |\Psi_{x^m}\rangle = \sum_{i=1}^{N}x_i^m\,|i\rangle \tag{7}$$

$$e.g. \quad \begin{pmatrix}0.6\\0.4\end{pmatrix} \quad \rightarrow \quad |n\rangle = \sqrt{0.6}\,|0\rangle + \sqrt{0.4}\,|1\rangle \tag{8}$$

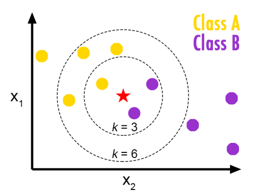


Figure 7: Visualization of a kNN classifier[1]

[1] Maria Schuld (2016), unpublished

The algorithm

Applying the **Hadamard gate** interferes the input and the training vectors:

$$\frac{1}{2\sqrt{M}}\sum_{m=1}^{M}(|0\rangle[|\Psi_{\tilde{x}}\rangle + |\Psi_{x^m}\rangle] + |1\rangle[|\Psi_{\tilde{x}}\rangle - |\Psi_{x^m}\rangle])\,|y^m(A \text{ or } B)\rangle\,|m\rangle \quad (9)$$

→ Perform **conditional measurement** on ancilla qubit.
Successful if $|0\rangle$ state is measured.

Maria Schuld (2016), unpublished

The algorithm

After successful conditional measurement, the state is proportional to

$$\frac{1}{2\sqrt{M}} \sum_{m=1}^{M} \sum_{i=1}^{N} (\tilde{x}_i + x_i^m) \left|0\right\rangle \left|i\right\rangle \left|y^m(A \text{ } or \text{ } B)\right\rangle \left|m\right\rangle \tag{10}$$

Probability to measure class B:

$$p(\left|y^m\right\rangle = \left|1(B)\right\rangle) = \sum_{m|y^m=1(B)} 1 - \frac{1}{4M} \mid \tilde{x} - x^m \mid^2 \tag{11}$$

Overall algorithmic complexity

$\mathcal{O}(\frac{1}{p_{acc}})$ where p_{acc} is the probability of measuring ancilla in the $\left|0\right\rangle$ state

Maria Schuld (2016), unpublished

Simple binary classification case

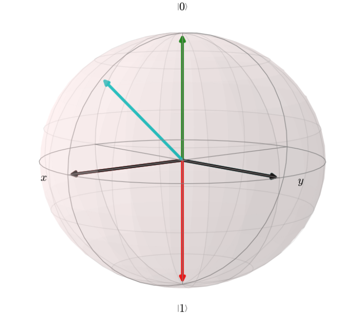


Figure 8: Simple binary classification problem of a quantum state

$$\frac{1}{\sqrt{2M}}\sum_{m=1}^{M}(|0\rangle\,|\Psi_{\tilde{x}}(\star)\rangle + |1\rangle\,|\Psi_{x^m}\rangle)\,|y^m(A \text{ or } B)\rangle\,|m\rangle \tag{12}$$

Procedure to load the input vector $\tilde{x}$:

$$|\Psi_0\rangle = \frac{1}{2}\sum_{m=1}^{2}(|0\rangle\,|0\rangle + |1\rangle\,|0\rangle)\,|y^m\rangle\,|m\rangle \tag{13}$$

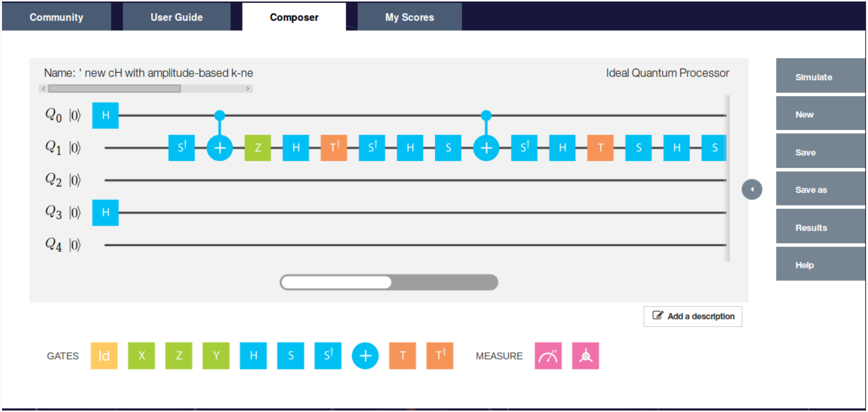
Apply controlled rotation ${}^{1}_{0}CR_y(\frac{\pi}{4})$ s.t.

$${}^{1}_{0}CR_y(\frac{\pi}{4})\,|\Psi_0\rangle = |\Psi_1\rangle = \frac{1}{2}\sum_{m=1}^{2}(|0\rangle\,|0\rangle + |1\rangle\,|\Psi_{\tilde{x}}\rangle)\,|y^m\rangle\,|m\rangle \tag{14}$$

Flip the ancilla qubit in the first register

$$(X\otimes\mathbb{1}\otimes\mathbb{1}\otimes\mathbb{1})\,|\Psi_1\rangle = |\Psi_2\rangle = \frac{1}{2}\sum_{m=1}^{2}(|0\rangle\,|\Psi_{\tilde{x}}\rangle + |1\rangle\,|0\rangle)\,|y^m\rangle\,|m\rangle \tag{15}$$

Implementation with IBM’s quantum computer



Name: ' new cH with amplitude-based k-ne

Ideal Quantum Processor

Figure 9: IBM’s quantum composer

- Accessible to the public
- Allows for ideal + real simulations

- 5 superconducting qubits
- 40 gates (39 gates + 1 measurement)

IBM’s universal gate set

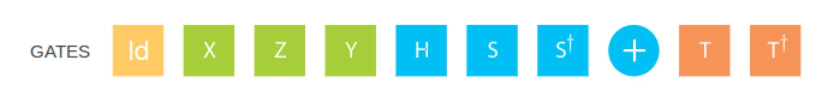


Figure 10: IBM’s universal gate set

How can we implement the ${}^{1}_{0}CR_y(\frac{\pi}{4})$ gate?

Liqui$|\rangle$ simulations

Currently impossible to implement the quantum algorithm on IBM's quantum computer! $\rightarrow$ can only simulate it with i.e. Liqui$|\rangle$

In Liqui$|\rangle$ we can directly implement the controlled R_y rotation!

Conclusion

Summary

- Initial state preparation is non-trivial! (even for very simple examples)
- In this case, state preparation dominates the overall algorithmic complexity
- Solovay-Kitaev yields long gate sequences for good approximations
- Some universal gate sets are only useful when combined with long qubit lifetimes
- **Need for better quantum compiling and more general state preparation algorithms!**

Taking it further

- Complexity analysis of the qubit-based kNN algorithm
- Classification of gaussian probability distributions
- Implementing more general state preparation algorithms
- Waiting for IBM QASM 2.0 ...

References

Bekkerman, R., Bilenko, M., & Langford, J. (2011). Scaling up machine learning: Parallel and distributed approaches. Cambridge University Press.

Booth Jr, J. (2012). Quantum compiler optimizations. arXiv preprint arXiv:1206.3348.

Dawson, C. M., & Nielsen, M. A. (2005). The Solovay-Kitaev algorithm. arXiv preprint quant-ph/0505030.

IBM. (2016). What is big data? https://www-01.ibm.com/software/data/bigdata/what-is-big -data.html. (Accessed: 2016-09-08)

Jat, R. N., & Ruhela, D. S. (2011). Comparative study of complexity of algorithms for iterative solution of non-linear equations. Journal of International Academy Of Physical Sciences, 15(4).

Nielsen, M. A., & Chuang, I. L. (2010). Quantum computation and quantum information. Cambridge University Press.

Questions?

Backup slide: Qubit decoherence times

We expect exponential decay:

$$e^{t/T_i} \tag{16}$$

T_1: Longitudinal coherence time (amplitude damping)

- Prepare $|0\rangle$ state
- Apply the X (NOT) gate s.t. qubit is in $|1\rangle$ state
- Wait for time t
- Measure the probability of being in $|1\rangle$ state

T_2: Transversal coherence time (phase damping)

- Prepare $|0\rangle$ state
- Apply Hadamard $\rightarrow$ $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$
- Wait for time t
- Apply Hadamard again
- Measure the probability of being in $|0\rangle$ state

We expect this probability to go to 0.5 → qubit lost quantum behaviour

Chuang I. (2003) MIT Lecture 19: How to Build Your Own Quantum Computer

Backup Slide II: Experimental realizations

Only few experimental verifications of QML algorithms:

- Li, Liu, Xu, and Du (2015) successfully distinguished a handwritten six from a nine using a quantum support vector machine on a four-qubit nuclear magnetic resonance test bench[1]

- Cai et al. (2015) were first to experimentally demonstrate quantum machine learning on a photonic QC and showed that the distance between two vectors and their inner product can indeed be computed quantum mechanically[2]

- Rist et al. (2015) solved a learning parity problem with five superconducting qubits and found that a quantum advantage can already be observed in non error-corrected systems[3]

[1] Li, Z., Liu, X., Xu, N., & Du, J. (2015). Experimental realization of a quantum support vector machine. Physical Review Letters, 114 (14), 15. doi: 10.1103/PhysRevLett.114.140504

[2] Cai, X. D., Wu, D., Su, Z. E., Chen, M. C., Wang, X. L., Li, L., . . . Pan, J. W. (2015). Entanglement- based machine learning on a quantum computer.

Machine Learning

Machine learning can be subdivided into three major fields.

Supervised ML

- Based on *input* and *output* data

"I know how to classify this data but I need the algorithm to do the computations for me."

Unsupervised ML

- Based on *input* data only

"I have no clue how to classify this data, can the algorithm create a classifier for me?"

Reinforcement learning

- Based on *input* data only

"I have no clue how to classify this data, can the algorithm classify this data and I'll give it a reward if it's correct or I'll punish it if it's not."

Machine Learning

Machine learning can be subdivided into three major fields.

Supervised ML

- Based on *input* and *output* data

"I know how to classify this data but I need the algorithm to do the computations for me."

Unsupervised ML

- Based on *input* data only

"I have no clue how to classify this data, can the algorithm create a classifier for me?"

Reinforcement learning

- Based on *input* data only

"I have no clue how to classify this data, can the algorithm classify this data and I'll give it a reward if it's correct or I'll punish it if it's not."

Liqui|⟩ simulations: Taking it further

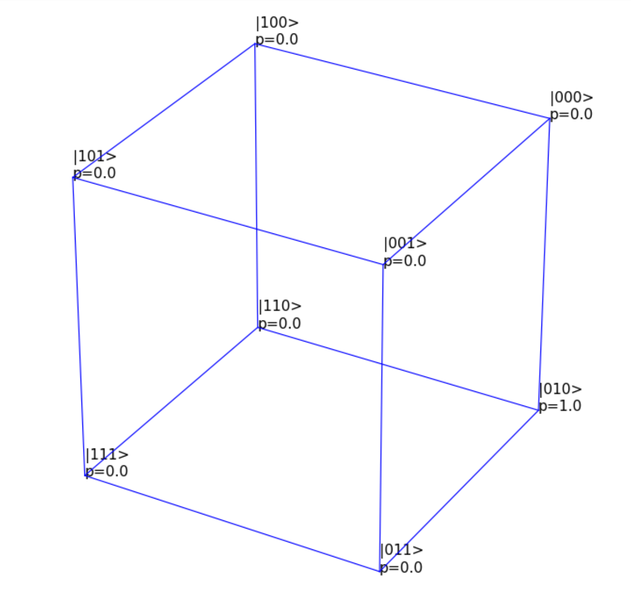


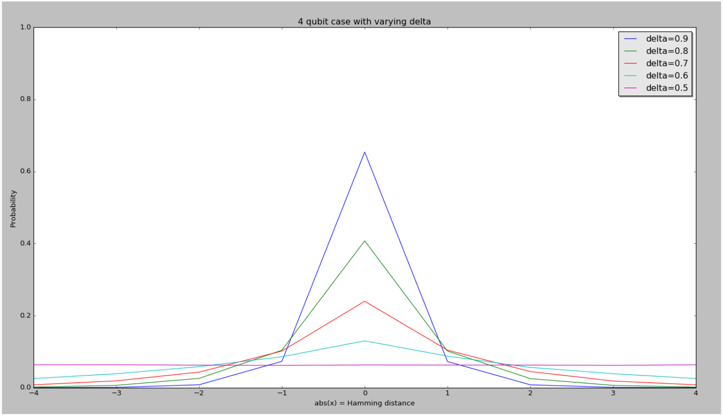
Figure 11: Representation of hamming distance on 3D cube

Liqui|⟩ simulations: Taking it further

Applying the following matrix

$$\begin{pmatrix} \sqrt{\delta} & 1-\sqrt{\delta} \\ 1-\sqrt{\delta} & -\sqrt{\delta} \end{pmatrix} \tag{17}$$

to all qubits in the data register leads to a gaussian distribution over the "Hamming distance" cube:



Liqui|⟩ simulations: Taking it further

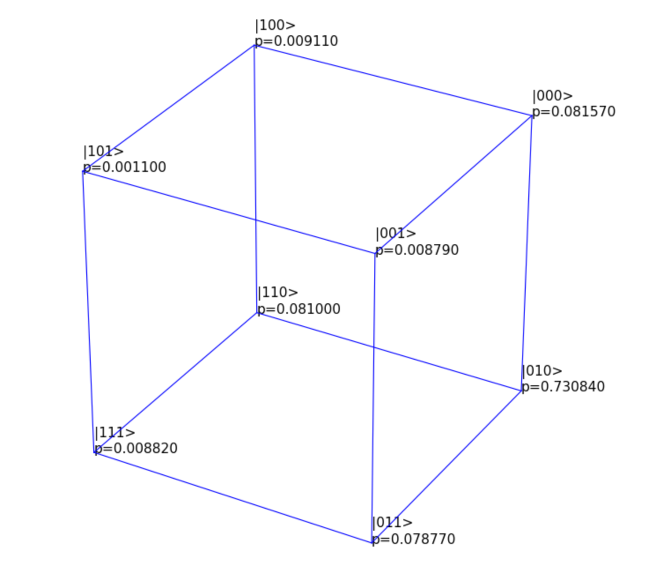


Figure 13: Representation of gaussian diffusion on 3D cube

Liqui|⟩ simulations: Taking it further

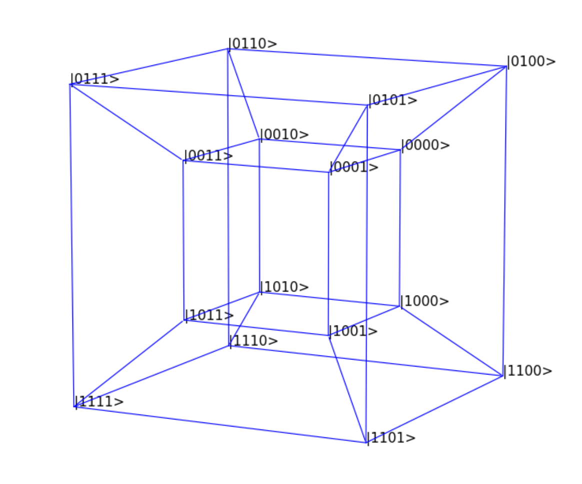


Figure 14: Representation of gaussian diffusion on 3D cube

Backup slide II I

M. Schuld, M. Fingerhuth, and F. Petruccione.

Amplitude-based quantum k-nearest neighbour algorithm. Manuscript in preparation.

2016.

Qubit-based kNN quantum algorithm

Typography

```
The theme provides sensible defaults to
\emph{emphasize} text, \alert{accent} parts
or show \textbf{bold} results.
```

becomes

The theme provides sensible defaults to *emphasize* text, accent parts or show **bold** results.

Font feature test

- Regular
- *Italic*
- SmallCaps
- **Bold**
- **Bold Italic**
- **Bold SmallCaps**
- `Monospace`
- *`Monospace Italic`*
- `Monospace Bold`
- *`Monospace Bold Italic`*

References go here

Lists

Items

- Milk
- Eggs
- Potatos

Enumerations

1. First,
2. Second and
3. Last.

Descriptions

PowerPoint Meeh.

Beamer Yeeeha.

Animation

- This is important

Animation

- This is important
- Now this

Animation

- This is important
- Now this
- And now this

Animation

- This is really important
- Now this
- And now this

Figures

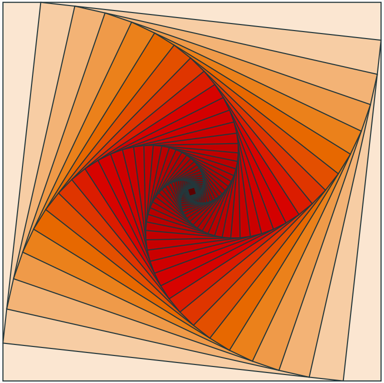


Figure 15: Rotated square from texample.net.

Tables

Table 1: Largest cities in the world (source: Wikipedia)

City	Population
Mexico City	20,116,842
Shanghai	19,210,000
Peking	15,796,450
Istanbul	14,160,467

Blocks

Three different block environments are pre-defined and may be styled with an optional background color.

Default

Block content.

Alert

Block content.

Example

Block content.

Default

Block content.

Alert

Block content.

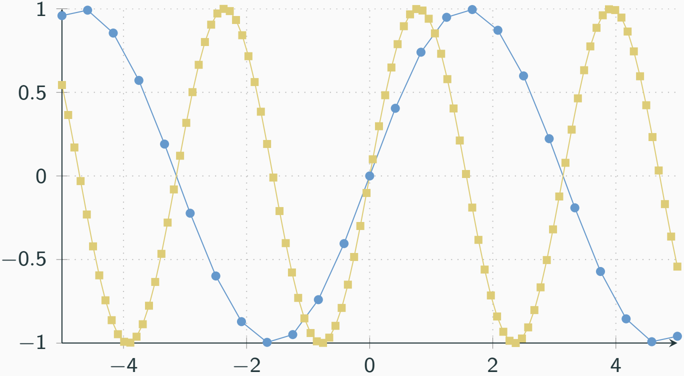
Example

Block content.

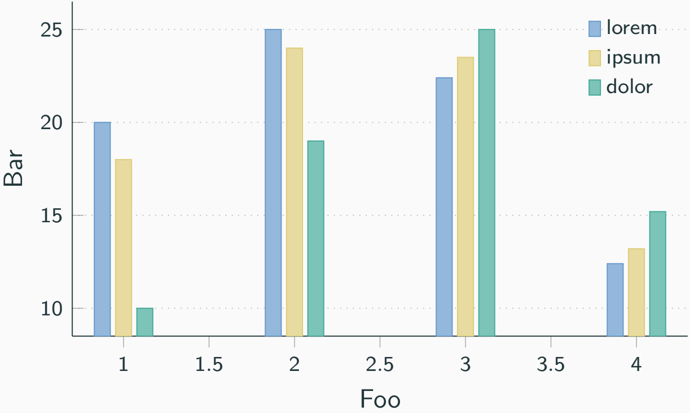
Math

$$e = \lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n$$

Line plots



Bar charts



Quotes

> *Veni, Vidi, Vici*

Supervised machine learning: concrete example

ID	Colour	Class label
1	🟥	red
2	🟥	red
3	🟥	red
4	🟦	blue
5	🟦	blue
6	🟦	blue

Table 2: Example training dataset.

ID	Colour	Class label
1	🟥	?
2	🟦	?

Table 3: Example input dataset.

Classical *k*-nearest neighbour

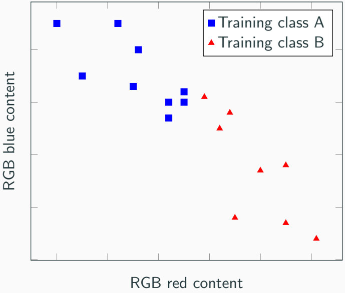
Transferring the colours into vectors

In the case of 9-bit RGB colours:

$$\color{red}{000}\ \color{lime}{000}\ \color{blue}{000}$$

3 bits for red, 3 bits for green and 3 bits for blue.

$$\begin{pmatrix} \color{red}{red \quad content} \\ \color{lime}{\cancel{green \quad content}} \\ \color{blue}{blue \quad content} \end{pmatrix} = \begin{pmatrix} 7 \\ \cancel{0} \\ 0 \end{pmatrix} \equiv \begin{pmatrix} 7 \\ 0 \end{pmatrix} \tag{18}$$



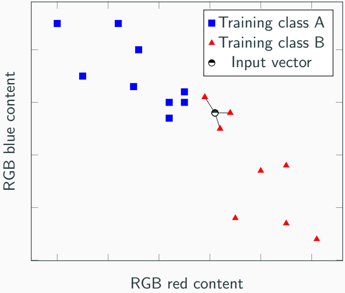
Classical k-nearest neighbour

- kNN is a non-parametric classifier
- k is a positive integer, usually chosen small

Given training dataset:
$D_T = v_0, v_1, .., v_{16}$
$v_i \in \{red, \ blue\}$

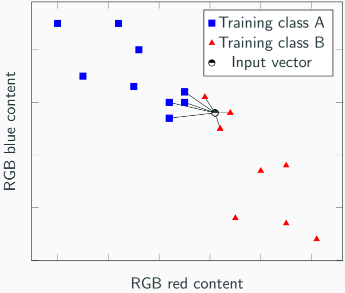
Given a new vector $\tilde{x}$ (black halfcircle):
- consider k nearest neighbours
- classify $\tilde{x}$, based on majority vote,
as *red* or *blue*

$k = 3$



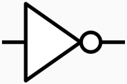
Classification → **RED**

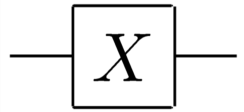
$k = 7$



Classification → **BLUE**

Single-qubit quantum logic gates



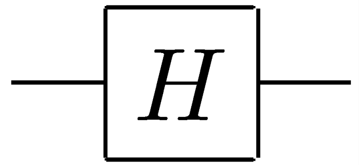


Any single-qubit quantum logic gates can be represented by a unitary 2×2 matrix whose action on a qubit is defined as:

$$U|\psi\rangle \doteq \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a\alpha + b\beta \\ c\alpha + d\beta \end{pmatrix} . \tag{19}$$

- Quantum computers perform linear (unitary) operations on qubits
- A quantum computation is the manipulation of an amplitude vector with a matrix representing a quantum logic gate

Single-qubit quantum logic gates: Hadamard gate



A very important single-qubit quantum logic gate is the **Hadamard** gate. It is represented by the matrix:

$$H \doteq \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} . \tag{20}$$

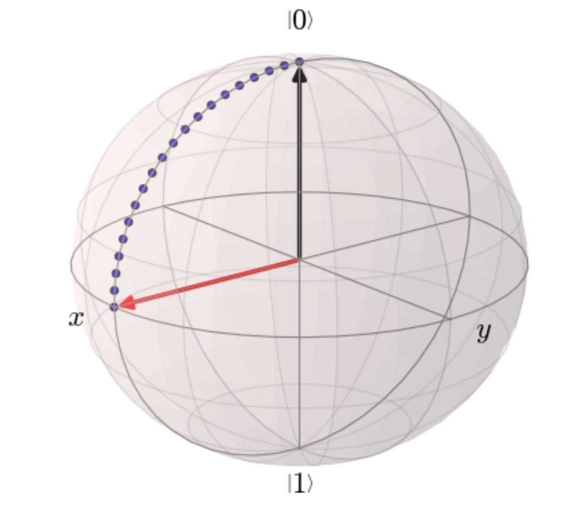
Consider acting the H gate on the $|0\rangle$ state:

$$H|0\rangle \doteq \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \doteq \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle . \tag{21}$$

→ **creates an equal superposition of $|0\rangle$ and $|1\rangle$!**

Single-qubit quantum logic gates: Hadamard gate

$$H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \ . \tag{22}$$



Multi-qubit systems

Tensor products are required when combining several qubits.

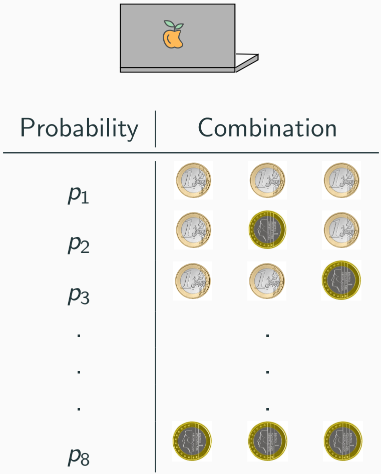
For example, the tensor product of two $|0\rangle$ kets is defined as:

$$|0\rangle \otimes |0\rangle = |00\rangle \doteq \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{23}$$

And for three $|0\rangle$ kets:

$$|00\rangle \otimes |0\rangle = |000\rangle \doteq \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{24}$$

Three random bits vs. three qubits



Probability | Combination

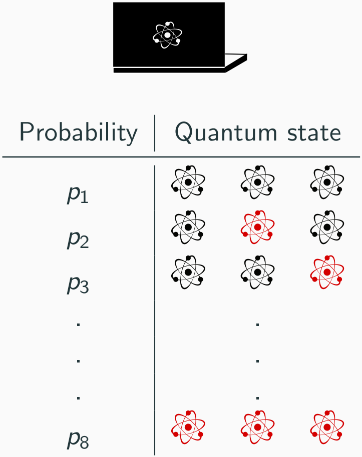
p_1

p_2

p_3

.
.
.

p_8



Probability | Quantum state

p_1

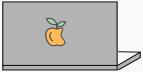
p_2

p_3

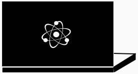
.
.
.

p_8

Three random bits vs. three qubits

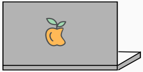


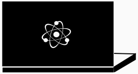
Probability	Combination
p_1	000
p_2	010
p_3	001
.	.
.	.
.	.
p_8	111



Probability	Quantum state
p_1	$\|000\rangle$
p_2	$\|010\rangle$
p_3	$\|001\rangle$
.	.
.	.
.	.
p_8	$\|111\rangle$

Three random bits vs. three qubits

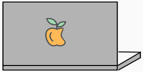


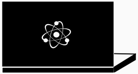


Prob.	Combination
$p_1 = \frac{1}{8}$	000
$p_2 = \frac{1}{8}$	010
$p_3 = \frac{1}{8}$	001
.	.
.	.
.	.
$p_8 = \frac{1}{8}$	111

Prob.	Amplitude	Quantum state
$p_1 = \mid a_1 \mid^2$	a_1	$\lvert 000\rangle$
$p_2 = \mid a_2 \mid^2$	a_2	$\lvert 010\rangle$
$p_3 = \mid a_3 \mid^2$	a_3	$\lvert 001\rangle$
.	.	.
.	.	.
.	.	.
$p_8 = \mid a_8 \mid^2$	a_8	$\lvert 111\rangle$

Three random bits vs. three qubits





Prob.	Combination
$p_1 = \frac{1}{8}$	000
$p_2 = \frac{1}{8}$	010
$p_3 = \frac{1}{8}$	001
.	.
.	.
.	.
$p_8 = \frac{1}{8}$	111

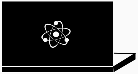
Prob.	Amplitude	Quantum state
$p_1 =\mid \frac{1}{2\sqrt{2}} \mid^2 = \frac{1}{8}$	$\frac{1}{2\sqrt{2}}$	$\|000\rangle$
$p_2 = \frac{1}{8}$	$-\frac{1}{2\sqrt{2}}$	$\|010\rangle$
$p_3 = \frac{1}{8}$	$-\frac{i}{2\sqrt{2}}$	$\|001\rangle$
.	.	.
.	.	.
.	.	.
$p_8 = \frac{1}{8}$	$\frac{i}{2\sqrt{2}}$	$\|111\rangle$

→ In quantum mechanics amplitudes can be interferred with each other!

→ This is impossible to do on a classical computer!

Three random bits vs. three qubits

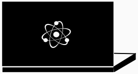
Applying an H gate to the first qubit leads to quantum interference such that:



Prob.	Amplitude	Quantum state
$p_1 = \mid a_1 + a_5 \mid^2$	$a_1 + a_5$	$\lvert 000\rangle$
$p_2 = \mid a_2 + a_6 \mid^2$	$a_2 + a_6$	$\lvert 010\rangle$
$p_3 = \mid a_3 + a_7 \mid^2$	$a_3 + a_7$	$\lvert 001\rangle$
$p_4 = \mid a_4 + a_8 \mid^2$	$a_4 + a_8$	$\lvert 011\rangle$
$p_5 = \mid a_1 - a_5 \mid^2$	$a_1 - a_5$	$\lvert 100\rangle$
$p_6 = \mid a_2 - a_6 \mid^2$	$a_2 - a_6$	$\lvert 110\rangle$
$p_7 = \mid a_3 - a_7 \mid^2$	$a_3 - a_7$	$\lvert 101\rangle$
$p_8 = \mid a_4 - a_8 \mid^2$	$a_4 - a_8$	$\lvert 111\rangle$

Three random bits vs. three qubits

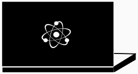
For example, substituting the values for $a_1 = \frac{1}{2\sqrt{2}}$ and $a_5 = \frac{1}{2\sqrt{2}}$ yields:



Prob.	Amplitude	Quantum state
$p_1 = \mid a_1 + a_5 \mid^2$	$\frac{1}{\sqrt{2}}(\frac{1}{2\sqrt{2}} + \frac{1}{2\sqrt{2}})$	$\|000\rangle$
$p_2 = \mid a_2 + a_6 \mid^2$	$a_2 + a_6$	$\|010\rangle$
$p_3 = \mid a_3 + a_7 \mid^2$	$a_3 + a_7$	$\|001\rangle$
$p_4 = \mid a_4 + a_8 \mid^2$	$a_4 + a_8$	$\|011\rangle$
$p_5 = \mid a_1 - a_5 \mid^2$	$\frac{1}{\sqrt{2}}(\frac{1}{2\sqrt{2}} - \frac{1}{2\sqrt{2}})$	$\|100\rangle$
$p_6 = \mid a_2 - a_6 \mid^2$	$a_2 - a_6$	$\|110\rangle$
$p_7 = \mid a_3 - a_7 \mid^2$	$a_3 - a_7$	$\|101\rangle$
$p_8 = \mid a_4 - a_8 \mid^2$	$a_4 - a_8$	$\|111\rangle$

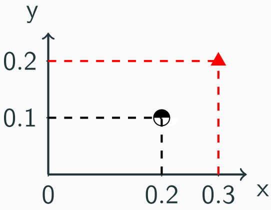
Three random bits vs. three qubits

For example, substituting the values for $a_1 = \frac{1}{2\sqrt{2}}$ and $a_5 = \frac{1}{2\sqrt{2}}$ yields:



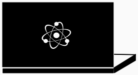
Prob.	Amplitude	Quantum state	
$p_1 = \mid \frac{1}{2} \mid^2 = \frac{1}{4}$	$\frac{1}{2}$	$\lvert 000\rangle$	$\rightarrow$ constructive interference
$p_2 = \mid a_2 + a_6 \mid^2$	$a_2 + a_6$	$\lvert 010\rangle$	"
$p_3 = \mid a_3 + a_7 \mid^2$	$a_3 + a_7$	$\lvert 001\rangle$	"
$p_4 = \mid a_4 + a_8 \mid^2$	$a_4 + a_8$	$\lvert 011\rangle$	"
$p_5 = \mid 0 \mid^2 = 0$	0	$\lvert 100\rangle$	$\rightarrow$ destructive interference
$p_6 = \mid a_2 - a_6 \mid^2$	$a_2 - a_6$	$\lvert 110\rangle$	"
$p_7 = \mid a_3 - a_7 \mid^2$	$a_3 - a_7$	$\lvert 101\rangle$	"
$p_8 = \mid a_4 - a_8 \mid^2$	$a_4 - a_8$	$\lvert 111\rangle$	"

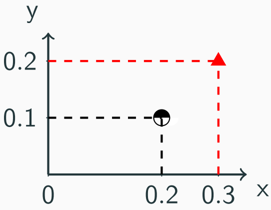
Calculating distances with interference



Prob.	Amplitude	Quantum state
$p_1 = 0.22$	$\frac{0.2}{\sqrt{0.18}}$	$\lvert 000\rangle$
$p_2 = 0.055$	$\frac{0.1}{\sqrt{0.18}}$	$\lvert 010\rangle$
$p_3 = 0$	0	$\lvert 001\rangle$
$p_4 = 0$	0	$\lvert 011\rangle$
$p_5 = 0.5$	$\frac{0.3}{\sqrt{0.18}}$	$\lvert 100\rangle$
$p_6 = 0.22$	$\frac{0.2}{\sqrt{0.18}}$	$\lvert 110\rangle$
$p_7 = 0$	0	$\lvert 101\rangle$
$p_8 = 0$	0	$\lvert 111\rangle$

Calculating distances with interference





Prob.	Amplitude	Quantum state
$p_1 = \frac{(0.2+0.3)^2}{0.18}$	$\frac{0.2+0.3}{\sqrt{0.18}}$	$\lvert 000\rangle$
$p_2 = \frac{(0.1+0.2)^2}{0.18}$	$\frac{0.1+0.2}{\sqrt{0.18}}$	$\lvert 010\rangle$
$p_3 = 0$	0	$\lvert 001\rangle$
$p_4 = 0$	0	$\lvert 011\rangle$
$p_5 = \frac{(0.2-0.3)^2}{0.18}$	$\frac{0.2-0.3}{\sqrt{0.18}}$	$\lvert 100\rangle$
$p_6 = \frac{(0.1-0.2)^2}{0.18}$	$\frac{0.1-0.2}{\sqrt{0.18}}$	$\lvert 110\rangle$
$p_7 = 0$	0	$\lvert 101\rangle$
$p_8 = 0$	0	$\lvert 111\rangle$

Controlled U gate

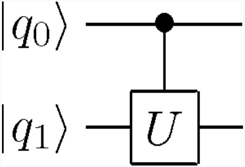


Figure 16: Controlled U-gate

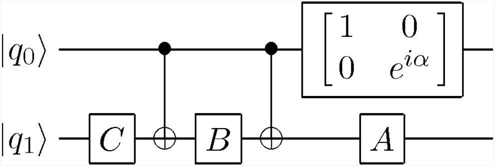


Figure 17: Decomposition of a controlled U-gate[1]

Choose A,B,C and α s.t.

$$e^{i\alpha} * A * X * B * X * C = U \quad and \quad A * B * C = \mathbb{1} \tag{25}$$

Need to solve the following equation[1]

$$U = \begin{pmatrix} e^{i(\alpha - \frac{\beta}{2} - \frac{\delta}{2})} \cos\frac{\gamma}{2} & -e^{i(\alpha - \frac{\beta}{2} + \frac{\delta}{2})} \sin\frac{\gamma}{2} \\ e^{i(\alpha + \frac{\beta}{2} - \frac{\delta}{2})} \sin\frac{\gamma}{2} & e^{i(\alpha + \frac{\beta}{2} + \frac{\delta}{2})} \cos\frac{\gamma}{2} \end{pmatrix} \tag{26}$$

[1] Nielsen, M. A., & Chuang, I. L. (2010). Quantum computation and quantum information. Cambridge University Press.

[2] Jat, R. N., & Ruhela, D. S. (2011). Comparative study of complexity of algorithms for iterative solution of non-linear equations. Journal of International Academy Of Physical Sciences, 15(4).

Problems with universal gate sets

In our case we need to find A, B, C and α for ${}^1_0 CR_y(\frac{\pi}{4})$:

Using a root finding algorithm for non-linear equations we find:

$$\alpha = \pi; \quad \beta = 2\pi; \quad \delta = \frac{7}{8}\pi; \quad \gamma = 0 \tag{27}$$

Then,

$$\begin{aligned} A &= R_z(\beta)R_y(\frac{\gamma}{2}) &&= R_z(2\pi) &&= \mathbb{1} && (28)\\ B &= R_y(-\frac{\gamma}{2})R_z(-\frac{\delta+\beta}{2}) &&= R_z(-\frac{23}{16}\pi) &&= ??? && (29)\\ C &= R_z(\frac{\delta-\beta}{2}) &&= R_z(-\frac{9}{16}\pi) &&= ??? && (30)\\ & \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} &&= \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{pmatrix} &&= Z && (31) \end{aligned}$$

[1] Dawson, C. M., & Nielsen, M. A. (2005). The Solovay-Kitaev algorithm. arXiv preprint quant-ph/0505030.

The Solovay-Kitaev theorem

$$B = R_z(-\frac{23}{16}\pi) = ??? \tag{32}$$

$$C = R_z(-\frac{9}{16}\pi) = ??? \tag{33}$$

The Solovay-Kitaev theorem guarantees that given a set of single-qubit quantum gates which generates a dense subset of $SU(2)$, then that set is guaranteed to fill $SU(2)$ quickly.[1]

→ **Hence, given any universal gate set it is possible to obtain good approximations to any desired gate.**

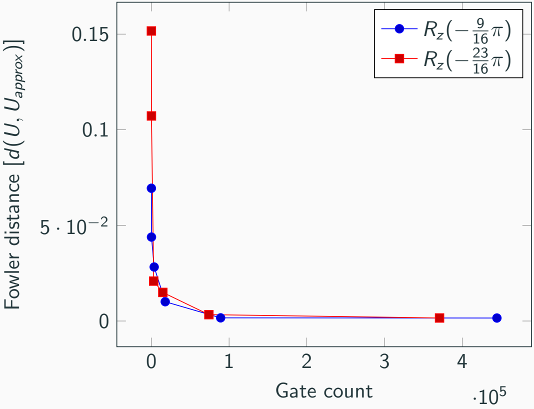
→ **But needs to be computed classically!**

[1] Dawson, C. M., & Nielsen, M. A. (2005). The Solovay-Kitaev algorithm. arXiv preprint quant-ph/0505030.

The Solovay-Kitaev algorithm

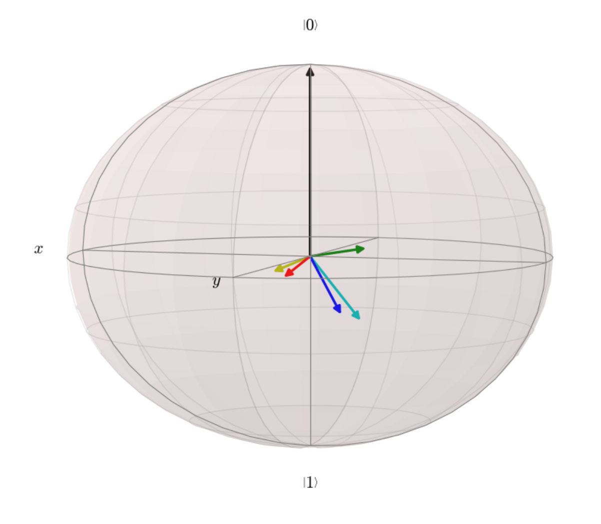
Fowler distance[1]:

$$dist(U, U_{approx}) = \sqrt{\frac{2-\mid tr(U \cdot U^{\dagger}_{approx}) \mid}{2}} \tag{34}$$



[1] Booth Jr, J. (2012). Quantum compiler optimizations. arXiv preprint arXiv:1206.3348.

The Solovay-Kitaev algorithm



$$d = 0.22739 \quad (35)$$

$$d = 0.15165 \quad (36)$$

$$d = 0.10722 \quad (37)$$

$$d = 0.02086 \quad (38)$$

$$d = 0.00156 \quad (39)$$

Figure 18: Various Fowler distances visualized on Bloch sphere

The Solovay-Kitaev algorithm

IBM's quantum computer needs **130ns for single-qubit gates** and **500ns for CNOT gates.**

IBM qubit decoherence times:

$49.5\,\mu s \leq T_1 \leq 85.3\,\mu s$ "amplitude damping"
$56.0\,\mu s \leq T_2 \leq 139.7\,\mu s$ "phase damping"

Approx. Gate	Distance	Gate count	Execution time
$R_z(-\frac{23}{16}\pi)$	0.15165	25	$\sim$3 μs
	0.10722	109	$\sim$14 μs
	0.02086	2997	$\sim$390 μs
	0.01494	14721	$\sim$1914 μs
	0.003327	74009	$\sim$9621 μs
	0.001578	370813	$\sim$48 206 μs

Table 4: SK algorithm results

Encoding classical data into qubits

1. Data encoded into qubits

k-dimensional probability vector requires $4k$ classical bits which are encoded one-to-one into $4k$ qubits, e.g.

$$\begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} * 10 \rightarrow \begin{pmatrix} 6 \\ 4 \end{pmatrix} \rightarrow \begin{pmatrix} 0110 \\ 0100 \end{pmatrix} \rightarrow n = 01100100 \rightarrow |n\rangle = |01100100\rangle$$

Schuld, Sinayskiy, and Petruccione (2014) developed a **qubit-based** quantum kNN algorithm. → requires a lot of qubits

My thesis research stressed the need for an alternative...