

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

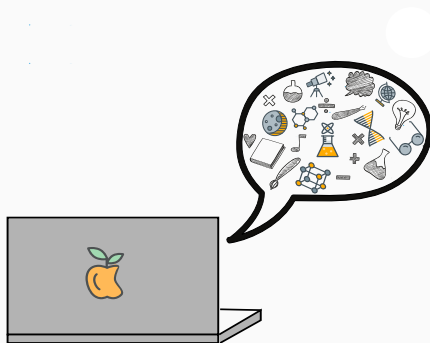


Table of contents

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

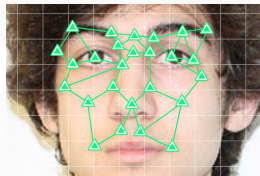
Introduction

Enhancing machine learning with quantum mechanics



Machine learning

Enable computers
to learn from data



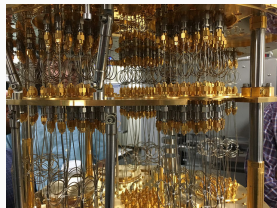
Source: IEEE Spectrum

- ML algorithms often involve¹
 - solving large systems of linear equations
 - inverting large matrices
 - distance computations
- Performing these computations on large data sets gets increasingly difficult²

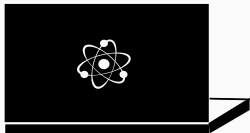
¹Bishop, C. M. (2006). Pattern recognition. Machine Learning, 128 .

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

Enhancing machine learning with quantum mechanics



Source: IBM

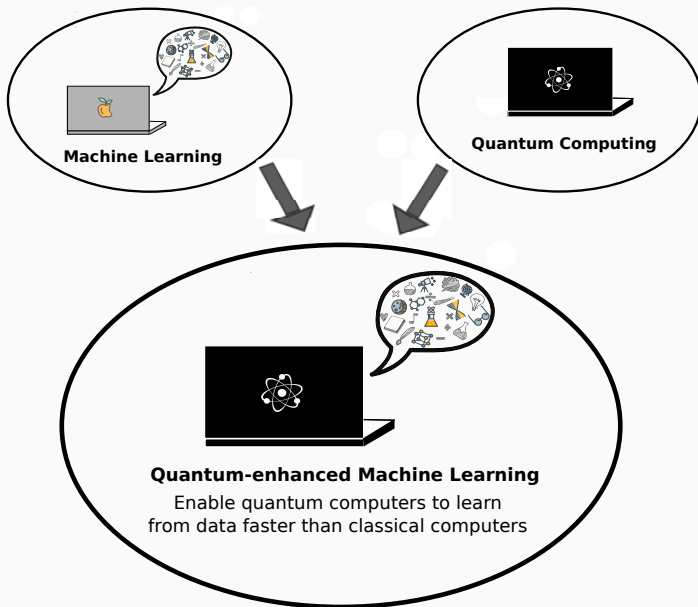


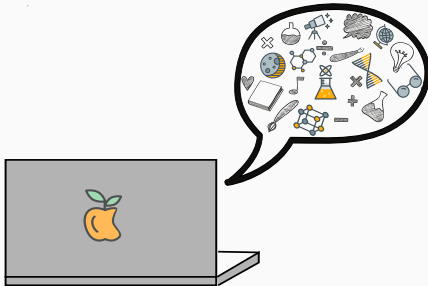
Quantum computing

Build computer hardware
based on quantum physics

- 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

Enhancing machine learning with quantum mechanics





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 (happy or sad)
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:

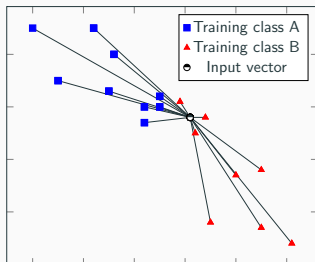
$$D_T = v_0, v_1, \dots, v_{16}$$

$$v_i \in \{\text{red}, \text{blue}\}$$

Given a new vector \tilde{x} (black halfcircle):

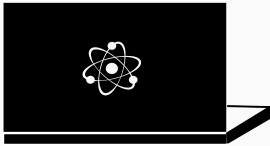
- consider k nearest neighbours
- classify \tilde{x} , based on majority vote, as *red* or *blue*

$k = \text{all}$



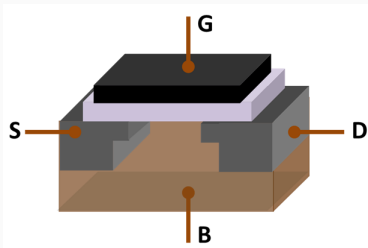
Assign distance-dependent weights
e.g. $\frac{1}{\text{distance}}$ to increase the influence of close vectors over more distant ones!

Classification \rightarrow **RED**



Quantum Computing

Classical vs. quantum bits (qubits)



Source: Wikipedia

Classical bit:

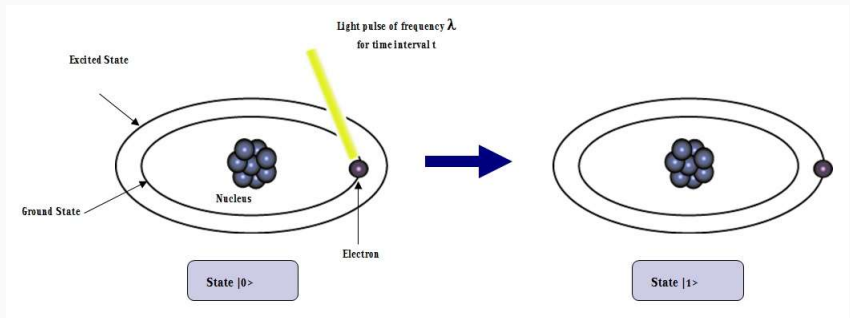
- Usually implemented through MOSFETs¹
- 2 definite states (0,1)
- Can be either 0 OR 1

¹ MOSFET - Metal Oxide Semiconductor Field Effect Transistor

Classical vs. quantum bits (qubits)

Quantum bit (qubit):

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



Source: RF Wireless World

Mathematically, the superposition of a qubit is expressed as:

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

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

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

The Bloch sphere

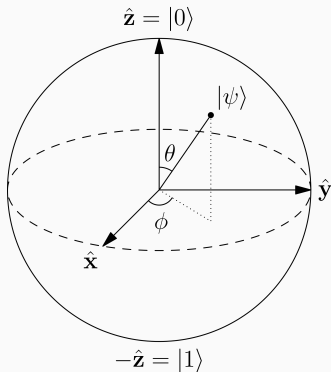


Figure 1: Arbitrary two-dimensional qubit $|\psi\rangle$ visualized on the Bloch sphere¹

Most general form of a 2-D qubit:

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

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 $0 \leq \phi \leq 2\pi$

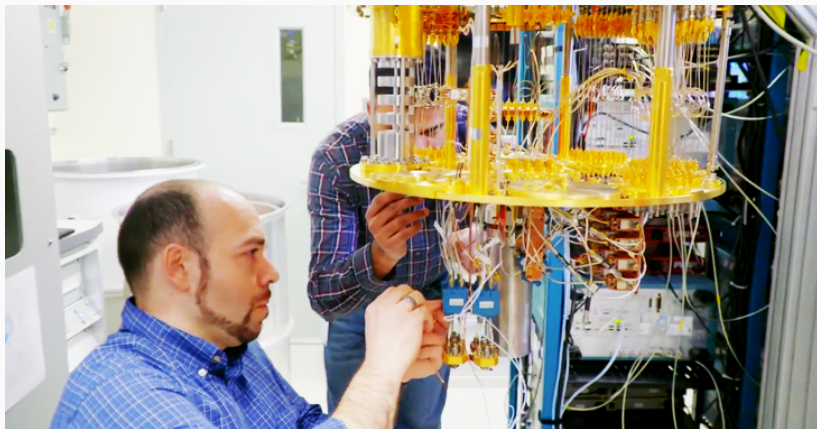
¹Source: Wikipedia

State-of-the-art quantum computation

Quantum computing is cutting edge technology at the frontier of supercomputing!

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

State-of-the-art quantum computation



Source: IBM

State-of-the-art quantum computation

Quantum computing is cutting edge technology at the frontier of supercomputing!

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

Current state-of-the-art prototypes:

- IBM → **5 superconducting qubits**¹
- Google → **9 superconducting qubits**²
- Weizmann Research Group in Israel → **8 trapped ions**³
- DWave → **1,152 qubits**⁴

BUT it solves only a narrow class of problems

¹ IBM Quantum Experience , ² Google Quantum A.I. Lab Team, ³ Roee Ozeri, Weizmann Institute of Science, ⁴ DWave Systems

Multi-qubit systems: the power of quantum computing

A quantum computer with n qubits has 2^n quantum amplitudes.

→ these amplitudes can be used to store huge amounts of information!

Qubit number	classical RAM needed	Simulation time
5	256 bytes	seconds on smartwatch
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

Motivation for this thesis research

- Classical machine learning is a very applied field
- State-of-the-art quantum computers only have few qubits
- Quantum-enhanced machine learning is almost purely theoretical
- There have been only a handful of proof-of-principle studies in quantum-enhanced machine learning¹
- Need for more proof-of-principle implementations and simulations to demonstrate the benefits of quantum computation for machine learning
- Small-scale machine learning problems need to be identified & implemented.

¹Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., & Lloyd, S. (2016). Quantum machine learning. arXiv preprint arXiv:1611.09347 . 15

Methods

Methods: IBM Quantum Experience

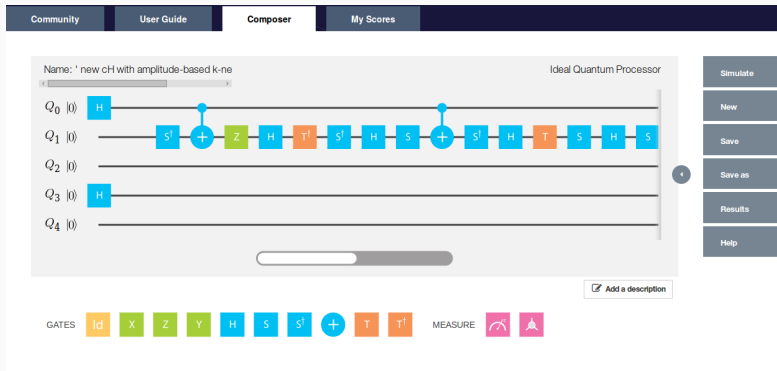


Figure 2: IBM's quantum composer¹

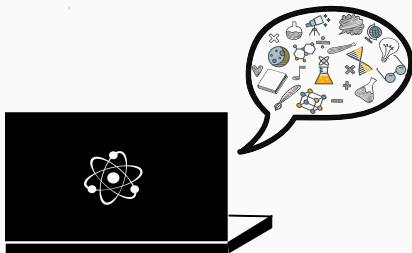
- Accessible to the public
- Allows for ideal + real simulations
- 5 superconducting qubits
- 40 gates (39 gates + 1 measurement)

¹ Screenshot taken from <https://quantumexperience.ng.bluemix.net/qstage/#/editor>

Small quantum computers can be simulated on conventional 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.

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

Schuld, Fingerhuth, and Petruccione (Manuscript in preparation) developed a new **amplitude-based** kNN algorithm.

→ requires only a few qubits and provides great speed-up

The amplitude-based kNN algorithm

1. $|\psi_0\rangle = \frac{1}{\sqrt{2M}} \sum_{m=1}^M (|0\rangle |\Psi_{\tilde{x}(\ast)}\rangle + |1\rangle |\Psi_{x^m}\rangle) |y^m\rangle |m\rangle$
[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. $\text{Prob}(|y^m\rangle = |1\rangle) = \sum_{m|y^m=1} 1 - \frac{1}{4M} |\tilde{x} - x^m|^2$
[Probability to measure a certain class]
5. $y = \begin{cases} 0, & \text{if } \text{Prob}(|y^0\rangle) > \text{Prob}(|y^1\rangle) \\ 1, & \text{if } \text{Prob}(|y^1\rangle) > \text{Prob}(|y^0\rangle) \\ -, & \text{otherwise} \end{cases}$ [Classification]

Calculating distances with interference



Source: TutorVista

Results: Amplitude-based k-nearest neighbour algorithm

Simple binary classification case

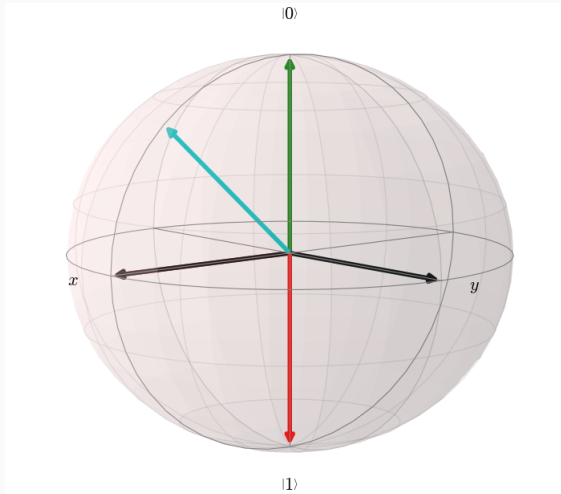


Figure 3: Simple binary classification problem of a quantum state

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 of about $150\text{ }\mu\text{s}$ 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!
- Need for proof-of-principle simulations within the Qiskit toolsuite 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 works as expected, is scalable and has the potential to outperform the classical counterpart.

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
- $\text{Liqui}| \rangle$ simulations demonstrated 100% classification accuracy on Bloch vector classification task
- Open problem: How to encode arbitrary classical data into quantum amplitude distributions?

- 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 with three additional quantum logic gates and 80 quantum gate slots. (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.
- Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., & Lloyd, S. (2016). Quantum machine learning. arXiv preprint arXiv:1611.09347.
- Bishop, C. M. (2006). Pattern recognition. Machine Learning, 128 .
- Denchev, V. S., Boixo, S., Isakov, S. V., Ding, N., Babbush, R., Smelyanskiy, V., ... & Neven, H. (2015). What is the Computational Value of Finite Range Tunneling?. arXiv preprint arXiv:1512.02206.
- Nielsen, M. A., & Chuang, I. L. (2010). Quantum computation and quantum information. Cambridge University Press.
- Schuld, M., Fingerhuth, M., & Petruccione, F. (Manuscript in preparation). Amplitude-based quantum k-nearest neighbour algorithm. Manuscript in preparation.

Einstein said that if quantum mechanics were correct then the world would be crazy. Einstein was right - the world is crazy.

- Daniel Greenberger -

Questions?

Results: Qubit-based kNN algorithm

Quantum Computing & Qubits

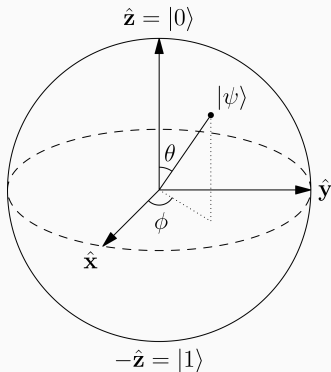


Figure 5: Arbitrary two-dimensional qubit $|\psi\rangle$ visualized on the Bloch sphere¹

Most general form of a 2-D qubit:

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

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

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

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

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

¹Reprinted 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 day¹
- 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 difficult²

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

²Bekkerman, 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
- 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.

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

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.

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

Classical k-nearest neighbour

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

Given training data set: Given a new vector \tilde{x} (red star):

$$D_T = v_0, v_1, \dots, v_{10}$$

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

- consider k nearest neighbours

- classify \tilde{x} , based on majority vote, as A or B

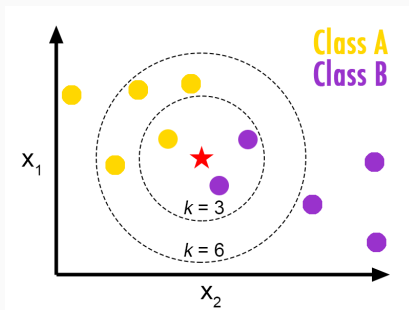


Figure 6: Visualization of a kNN classifier¹

¹Reprinted 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

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

where

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

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

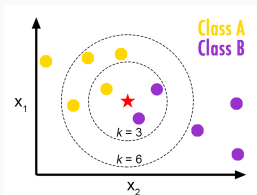


Figure 7: Visualization of a kNN classifier¹

The algorithm

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

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

→ Perform **conditional measurement** on ancilla qubit.

Successful if $|0\rangle$ state is measured.

The algorithm

After successful conditional measurement, the state is proportional to

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

Probability to measure class B:

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

Overall algorithmic complexity

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

Simple binary classification case

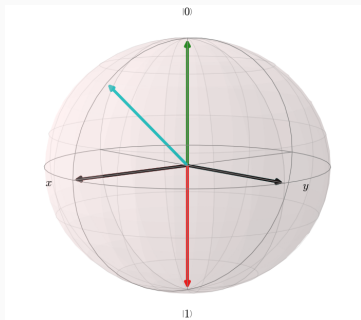


Figure 8: Simple binary classification problem of a quantum state

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

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

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

Apply controlled rotation ${}_0^1CR_y(\frac{\pi}{4})$ s.t.

$${}_0^1CR_y(\frac{\pi}{4}) |\Psi_0\rangle = |\Psi_1\rangle = \frac{1}{2} \sum_{m=1}^2 (|0\rangle |0\rangle + |1\rangle |\Psi_{\tilde{x}}\rangle) |y^m\rangle |m\rangle \quad (14)$$

Flip the ancilla qubit in the first register

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

Implementation with IBM's quantum computer

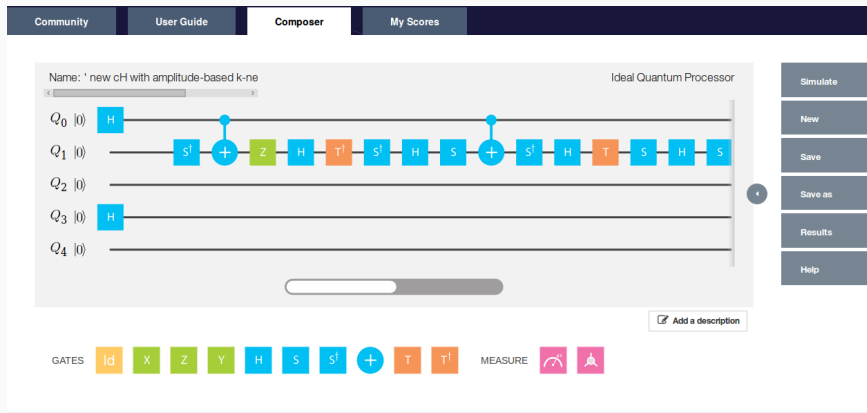


Figure 9: IBM's quantum composer

- 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 $\frac{1}{0}CR_y(\frac{\pi}{4})$ 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:

$$e^{t/T_i} \quad (16)$$

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 \rightarrow qubit lost quantum behaviour

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 bench¹
- 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 mechanically²
- 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 systems³

¹Li, 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

²Cai, 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| \rangle simulations: Taking it further

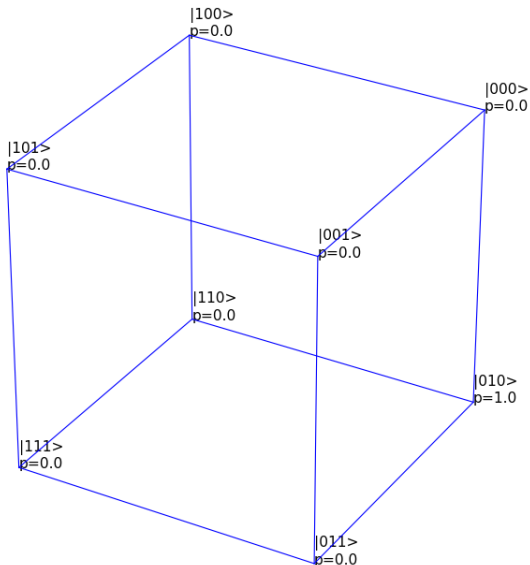


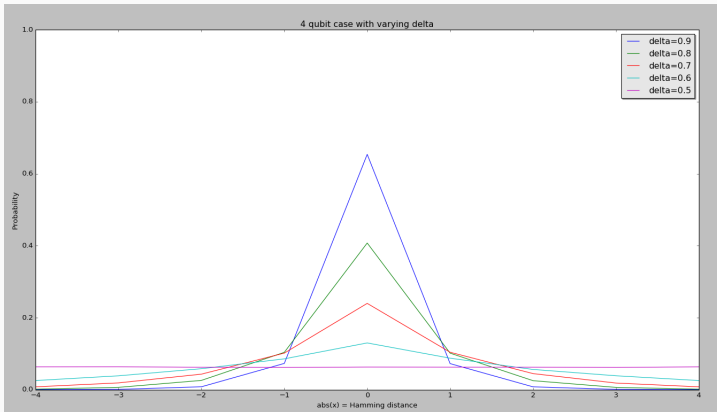
Figure 11: Representation of hamming distance on 3D cube

Liqui|⟩ simulations: Taking it further

Applying the following matrix

$$\begin{pmatrix} \sqrt{\delta} & 1 - \sqrt{\delta} \\ 1 - \sqrt{\delta} & -\sqrt{\delta} \end{pmatrix} \quad (17)$$

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



Liqui|> simulations: Taking it further

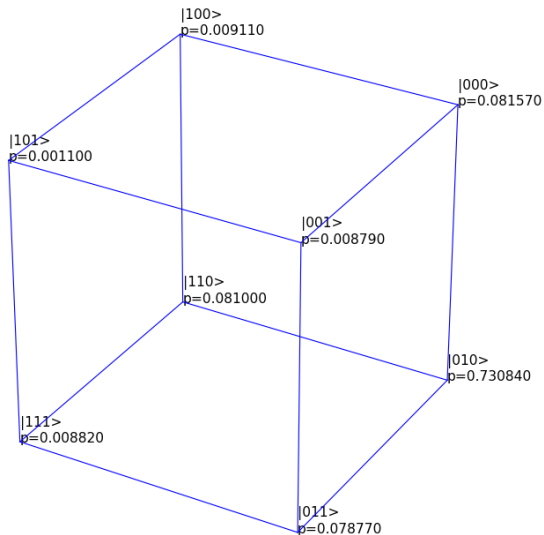


Figure 13: Representation of gaussian diffusion on 3D cube

Liqui|> simulations: Taking it further

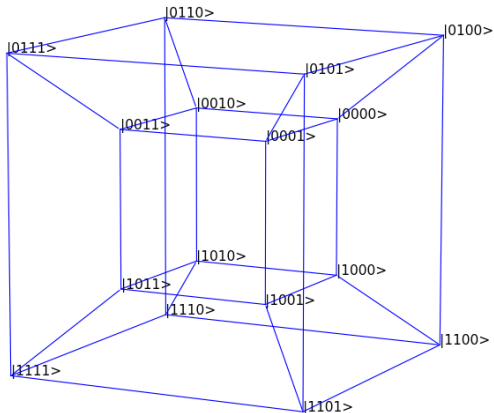


Figure 14: Representation of gaussian diffusion on 3D cube



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*

Lists

Items

- Milk
- Eggs
- Potatos

Enumerations

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

Descriptions

PowerPoint Meeh.
Beamer Yeeeha.

- 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

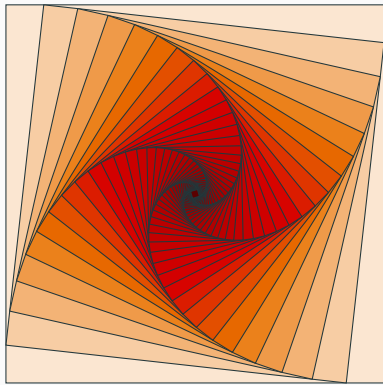


Figure 15: Rotated square from texample.net.

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

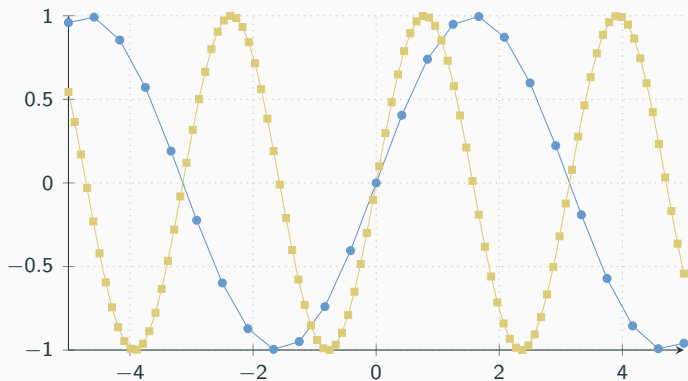
Block content.

Example

Block content.

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

Line plots



Bar charts



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

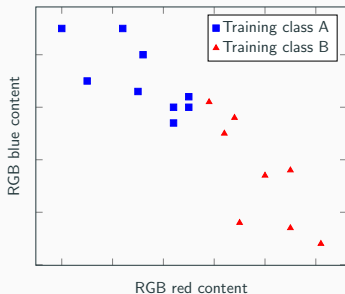
Transferring the colours into vectors

In the case of 9-bit RGB colours:

000 000 000

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

$$\begin{pmatrix} \text{red content} \\ \text{green content} \\ \text{blue content} \end{pmatrix} = \begin{pmatrix} 7 \\ \emptyset \\ 0 \end{pmatrix} \equiv \begin{pmatrix} 7 \\ 0 \end{pmatrix} \quad (18)$$



Classical k -nearest neighbour

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

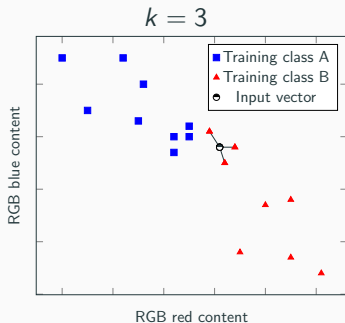
Given training dataset:

$$D_T = v_0, v_1, \dots, v_{16}$$

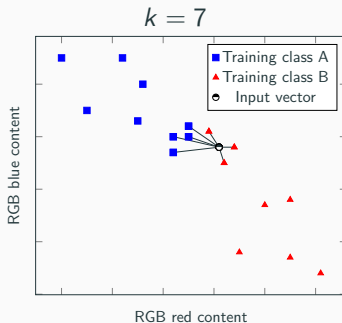
$$v_i \in \{\text{red}, \text{blue}\}$$

Given a new vector \tilde{x} (black halfcircle):

- consider k nearest neighbours
- classify \tilde{x} , based on majority vote, as *red* or *blue*

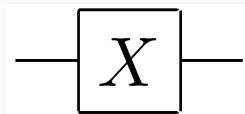
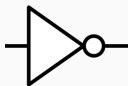


Classification → **RED**



Classification → **BLUE**

Single-qubit quantum logic gates

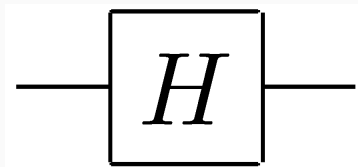


Any single-qubit quantum logic gates can be represented by a unitary 2×2 matrix whose action on a qubit is defined as:

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

- 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:

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

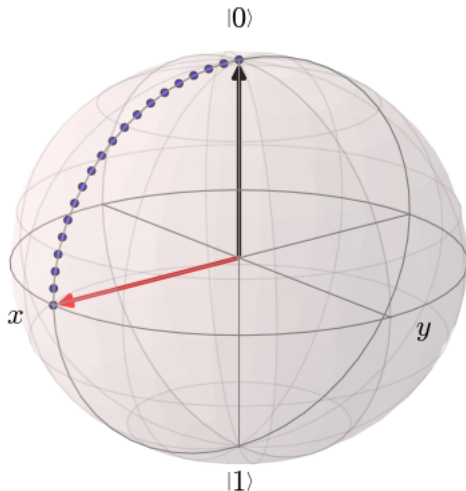
Consider acting the H gate on the $|0\rangle$ state:

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

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

Single-qubit quantum logic gates: Hadamard gate

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



Multi-qubit systems

Tensor products are required when combining several qubits.

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





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

And for three $|0\rangle$ kets:





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

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
\cdot	\cdot
\cdot	\cdot
\cdot	\cdot
p_8	111



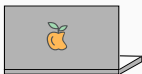
Probability	Quantum state
p_1	$ 000\rangle$
p_2	$ 010\rangle$
p_3	$ 001\rangle$
\cdot	\cdot
\cdot	\cdot
\cdot	\cdot
p_8	$ 111\rangle$

Three random bits vs. three qubits



Prob.	Combination	Prob.	Amplitude	Quantum state
$p_1 = \frac{1}{8}$	000	$p_1 = a_1 ^2$	a_1	$ 000\rangle$
$p_2 = \frac{1}{8}$	010	$p_2 = a_2 ^2$	a_2	$ 010\rangle$
$p_3 = \frac{1}{8}$	001	$p_3 = a_3 ^2$	a_3	$ 001\rangle$
.
.
.
$p_8 = \frac{1}{8}$	111	$p_8 = a_8 ^2$	a_8	$ 111\rangle$

Three random bits vs. three qubits



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

→ In quantum mechanics amplitudes can be interfered 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
$p_1 = a_1 + a_5 ^2$	$a_1 + a_5$	$ 000\rangle$
$p_2 = a_2 + a_6 ^2$	$a_2 + a_6$	$ 010\rangle$
$p_3 = a_3 + a_7 ^2$	$a_3 + a_7$	$ 001\rangle$
$p_4 = a_4 + a_8 ^2$	$a_4 + a_8$	$ 011\rangle$
$p_5 = a_1 - a_5 ^2$	$a_1 - a_5$	$ 100\rangle$
$p_6 = a_2 - a_6 ^2$	$a_2 - a_6$	$ 110\rangle$
$p_7 = a_3 - a_7 ^2$	$a_3 - a_7$	$ 101\rangle$
$p_8 = a_4 - a_8 ^2$	$a_4 - a_8$	$ 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
$p_1 = a_1 + a_5 ^2$	$\frac{1}{\sqrt{2}}\left(\frac{1}{2\sqrt{2}} + \frac{1}{2\sqrt{2}}\right)$	$ 000\rangle$
$p_2 = a_2 + a_6 ^2$	$a_2 + a_6$	$ 010\rangle$
$p_3 = a_3 + a_7 ^2$	$a_3 + a_7$	$ 001\rangle$
$p_4 = a_4 + a_8 ^2$	$a_4 + a_8$	$ 011\rangle$
$p_5 = a_1 - a_5 ^2$	$\frac{1}{\sqrt{2}}\left(\frac{1}{2\sqrt{2}} - \frac{1}{2\sqrt{2}}\right)$	$ 100\rangle$
$p_6 = a_2 - a_6 ^2$	$a_2 - a_6$	$ 110\rangle$
$p_7 = a_3 - a_7 ^2$	$a_3 - a_7$	$ 101\rangle$
$p_8 = a_4 - a_8 ^2$	$a_4 - a_8$	$ 111\rangle$

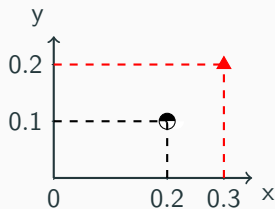
Three random bits vs. three qubits

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



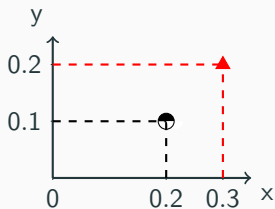
Prob.	Amplitude	Quantum state	
$p_1 = \left \frac{1}{2} \right ^2 = \frac{1}{4}$	$\frac{1}{2}$	$ 000\rangle$	→ constructive interference
$p_2 = \left a_2 + a_6 \right ^2$	$a_2 + a_6$	$ 010\rangle$	"
$p_3 = \left a_3 + a_7 \right ^2$	$a_3 + a_7$	$ 001\rangle$	"
$p_4 = \left a_4 + a_8 \right ^2$	$a_4 +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

Calculating distances with interference



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

Calculating distances with interference



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

Controlled U gate

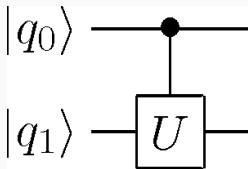


Figure 16: Controlled U-gate

Choose A,B,C and α s.t.

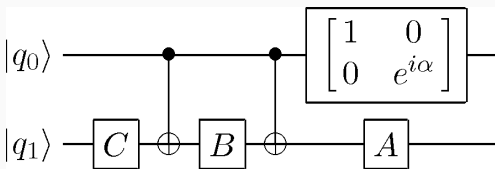


Figure 17: Decomposition of a controlled U-gate¹

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

Need to solve the following equation¹

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

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

²Jat, 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 $\frac{1}{0}CR_Y(\frac{\pi}{4})$:

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

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

Then,

$$A = R_z(\beta)R_y(\frac{\gamma}{2}) = R_z(2\pi) = \mathbb{1} \quad (28)$$

$$B = R_y(-\frac{\gamma}{2})R_z(-\frac{\delta + \beta}{2}) = R_z(-\frac{23}{16}\pi) = ??? \quad (29)$$

$$C = R_z(\frac{\delta - \beta}{2}) = R_z(-\frac{9}{16}\pi) = ??? \quad (30)$$

$$\begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{pmatrix} = Z \quad (31)$$

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

The Solovay-Kitaev theorem

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

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

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.**

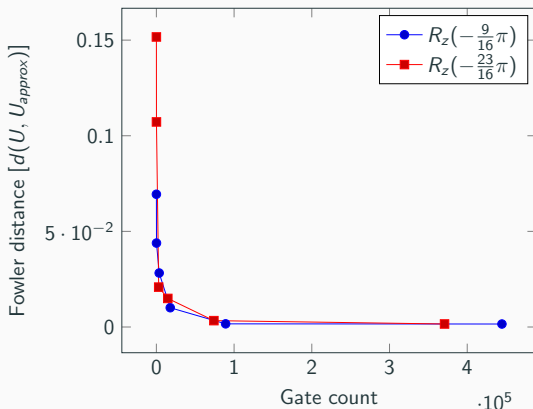
→ **But needs to be computed classically!**

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

The Solovay-Kitaev algorithm

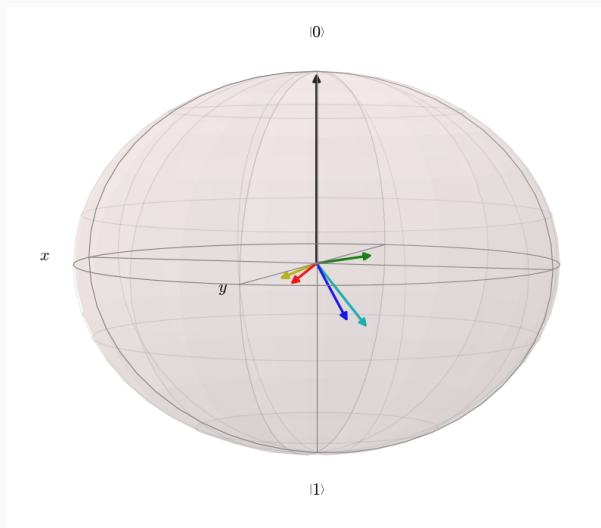
Fowler distance¹:

$$\text{dist}(U, U_{\text{approx}}) = \sqrt{\frac{2 - |\text{tr}(U \cdot U_{\text{approx}}^\dagger)|}{2}} \quad (34)$$



¹Booth 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:

$49.5 \mu\text{s} \leq T_1 \leq 85.3 \mu\text{s}$ "amplitude damping"

$56.0 \mu\text{s} \leq T_2 \leq 139.7 \mu\text{s}$ "phase damping"

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

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. \rightarrow requires a lot of qubits

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