Secrets revealed in this session:

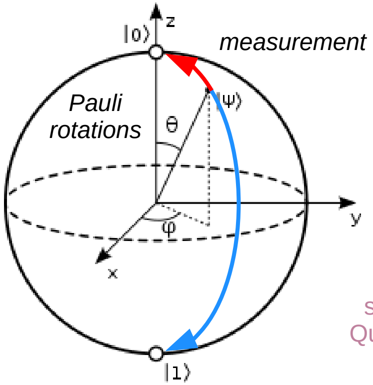
To explore the principles of quantum machine learning models, their parameterisation and optimisation

Quantum Machine Learning

Introduction

Jacob L. Cybulski

Enquanted, Melbourne, Australia



We will assume
some knowledge of
Quantum Computing
ML and Python

Our QML Team

Sebastian Zając

Assistant Professor
SGH Warsaw School of Economics
LinkedIn

Dr Paweł Gora

Coordinator of QPoland *and*
CEO of Fundacja Quantum AI
LinkedIn

Jacob Cybulski
Founder, Researcher, Consultant at Enquanted

and
Honorary A/Prof
In Quantum Computing
Deakin University
LinkedIn

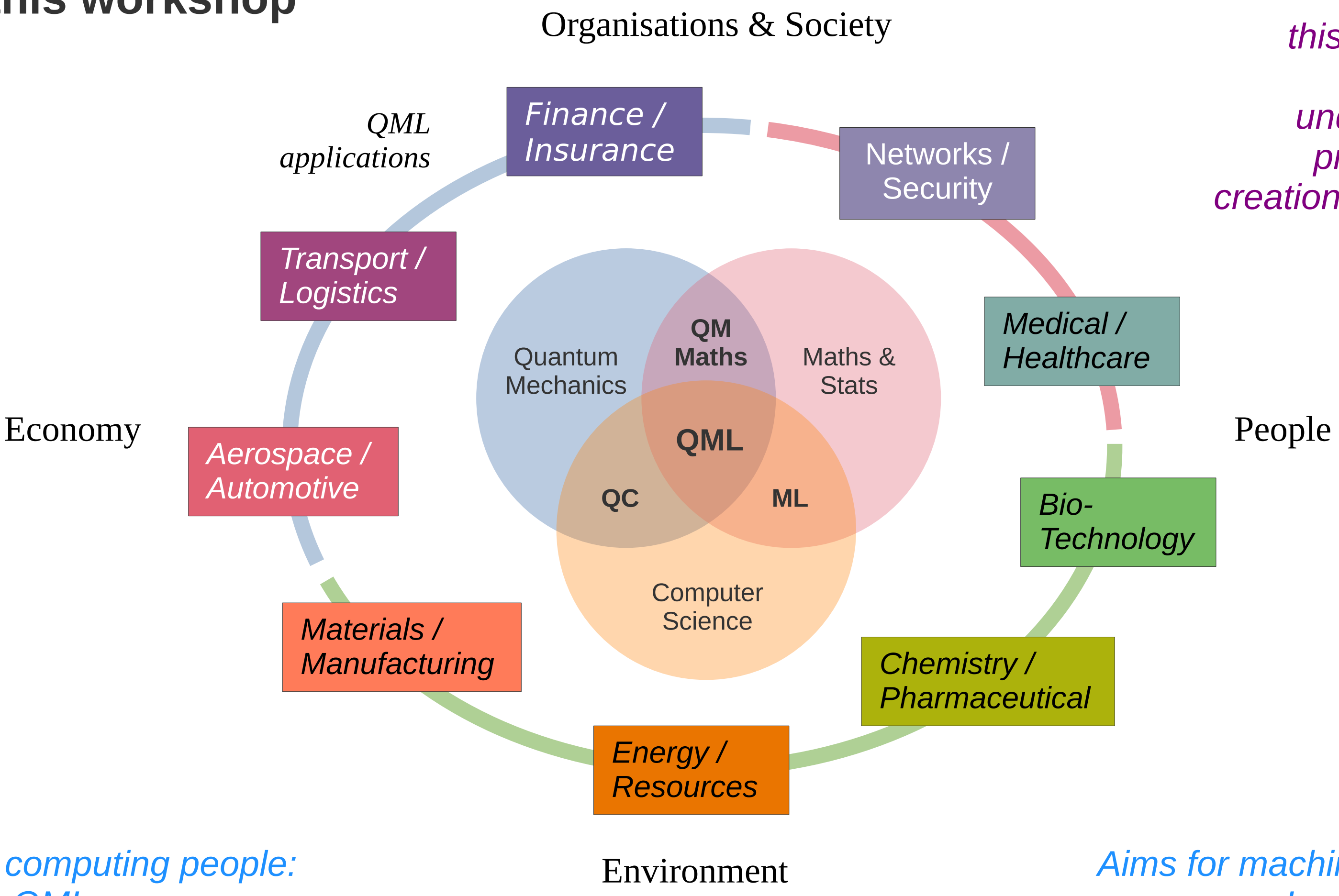
Tomek Rybotycki

QML Researcher
SRI PAS, NCAC PAS, CEAI AGH, KPLabs
LinkedIn

Jacob L. Cybulski, Quantum Business Series (Deakin, RMIT, ACS, Warsaw School of Economics)
Jacob L. Cybulski, Quantum Computing Intro Series (SheQuantum, Assoc of Polish Profs in Australia)
2021-2025

Quantum ML

aims of this workshop



this workshop aims at developing the understanding of and practical skills in the creation and application of QML models

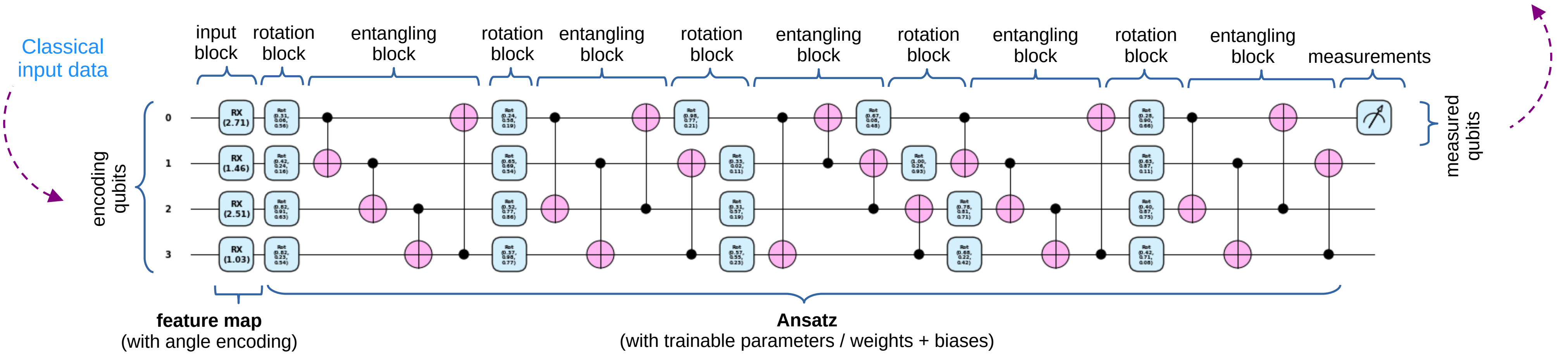
Aims for quantum computing people:
Learn about ML in QML

Aims for machine learning people:
Learn about Q in QML

Variational Quantum Models

= *Parameterised Quantum Circuits*

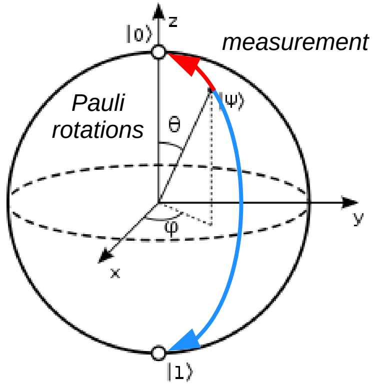
Ansatz parameters are trainable.
Each parameter defines a dimension in the model's parameter space.



We can create a "variational" model = a circuit template with parameterised gates, e.g. P(a), Ry(a) or Rz(a), each allowing rotation of a qubit state in x, y or z axis (as per Bloch sphere).

Typically, but now always, the circuit consists of three blocks:

- a feature map (input)
- an ansatz (processing)
- measurements (output)



Classical input data is encoded into the feature map's parameters, setting the model's initial quantum state.

The quantum state is then altered by an ansatz, which consists of parameterised gates (operations), which alter the initial quantum state.

The quantum state of the circuit is then measured and interpreted as the model's output in classical data form, e.g. as binary values, integer or real value, a single event's probability or the probability distribution.

Data encoding strategies

Data encoding

There are many methods of data embedding, such as: the *basis*, *angle*, *amplitude*, *QRAM*, ... encoding,

In this workshop we will rely on *angle encoding* realised as qubit state rotation by the angle defined by the data.

The rotation operators are always available in a quantum platform API (e.g. *Rx*, *Ry*, *Rz* or *Rxyz*).

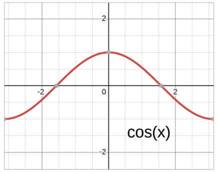
Typically, the encoding rotation is performed around x or y axis, or both (allowing two values per qubit).

Rotations are *relative to a specific qubit state*, commonly starting at |0⟩ state, or (|0⟩+|1⟩)/sqrt(2), which require qubits to be initialised in these states.

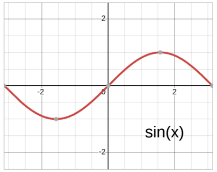
The encoded value could be represented either by the *angular rotation*, or the *amplitude* of the qubit projective measurement (Z).

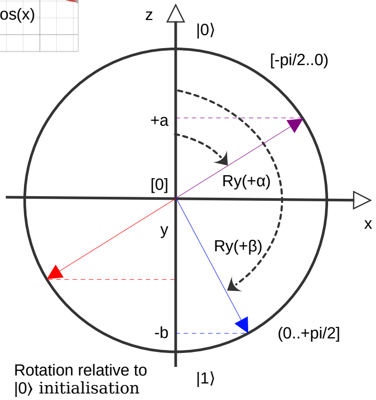
In some cases, input data is repeatedly encoded and interspersed with ansatz layers, called *data reuploading*, which improves the model performance.

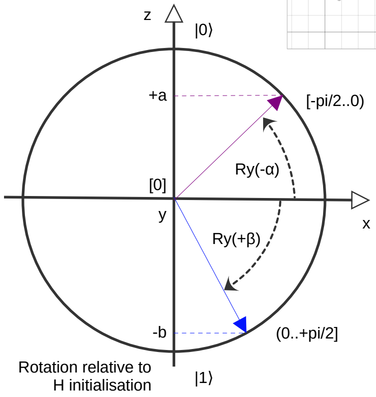
Maria Schuld and Francesco Petruccione.
Machine Learning with Quantum Computers. 2nd ed. Springer, 2021.

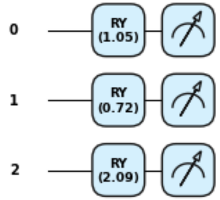


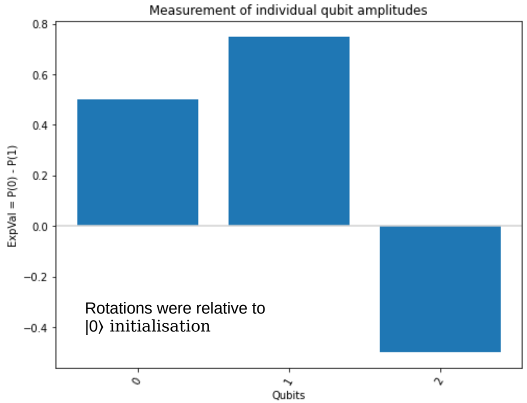
Note that training will place qubit states in areas x < 0 and arbitrarily around the z axis. Measurements of such states cannot distinguish them from "pure" x > 0 and z = 0.











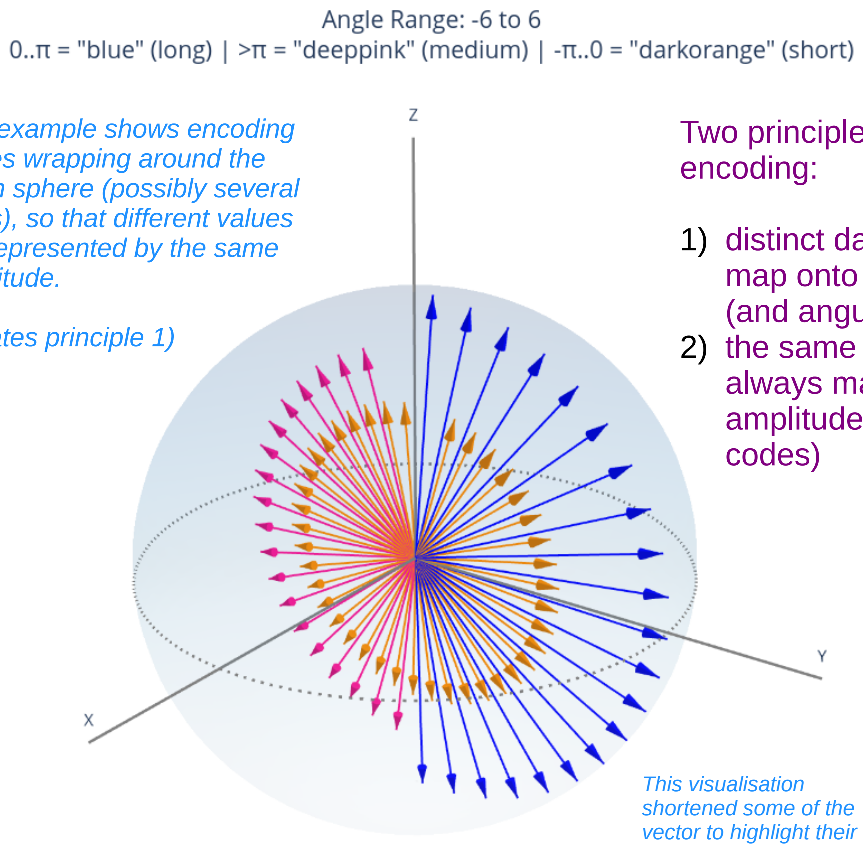
Input

Values entered:	[np.arccos(0.5), np.arccos(0.75), np.pi-np.arccos(0.5)]
Ry angles used:	[1.047, 0.723, 2.094]

Measurements

Probabilities:	[[0.25, 0.75], [0.562, 0.438], [0.25, 0.75]]
Amplitudes:	[0.5, 0.75, -0.5]

Angle encoding
The Good, the Bad and the Ugly



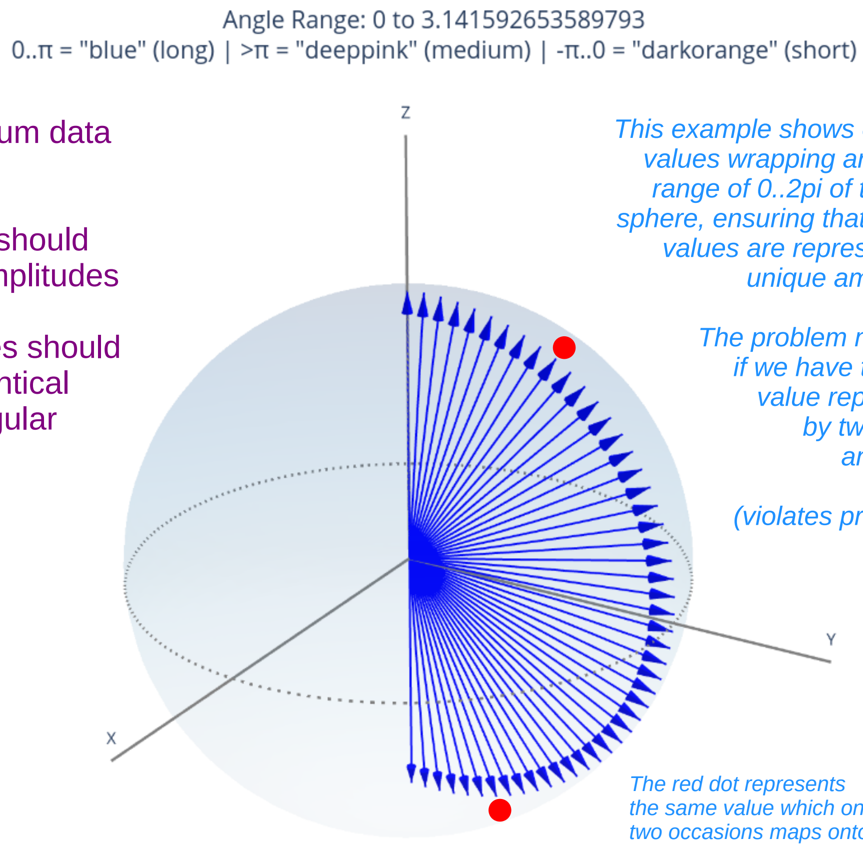
This example shows encoding values wrapping around the Bloch sphere (possibly several times), so that different values are represented by the same amplitude.

(violates principle 1)

Two principles of quantum data encoding:

1) distinct data values should map onto distinct amplitudes (and angular codes)
2) the same data values should always map into identical amplitudes (and angular codes)

This visualisation shortened some of the vector to highlight their difference, however, they are all of length 1.



This example shows encoding values wrapping around the range of 0..2pi of the Bloch sphere, ensuring that different values are represented by unique amplitudes.

The problem may arise if we have the same value represented by two distinct amplitudes

(violates principle 2)

The red dot represents the same value which on two occasions maps onto two different angles and thus two amplitudes.

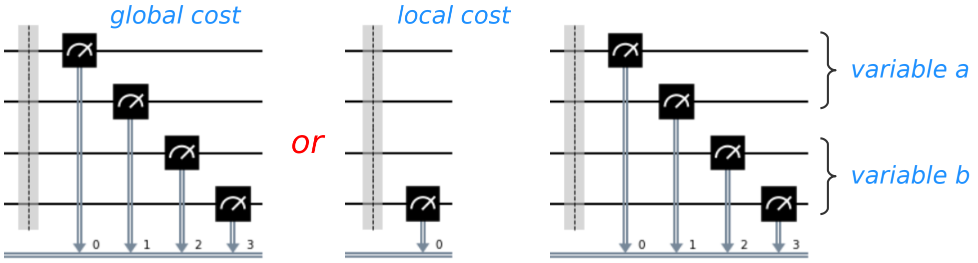
Commonly used
measurements and interpretation

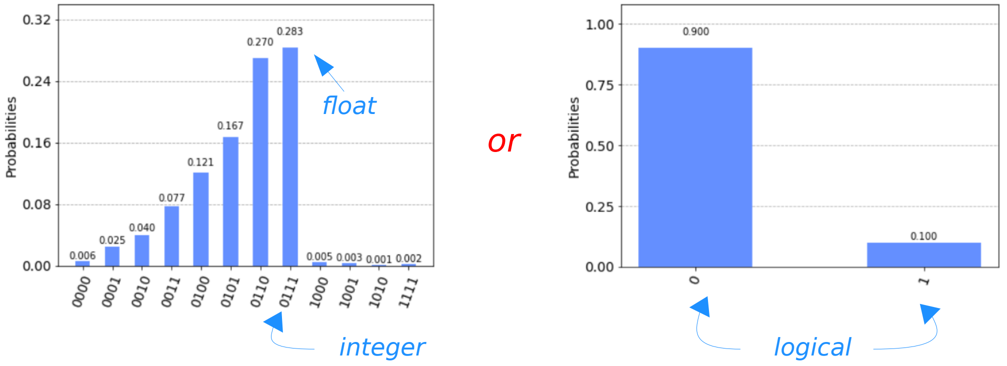
There are many ways of obtaining the outcome of a circuit execution, e.g. we can measure:

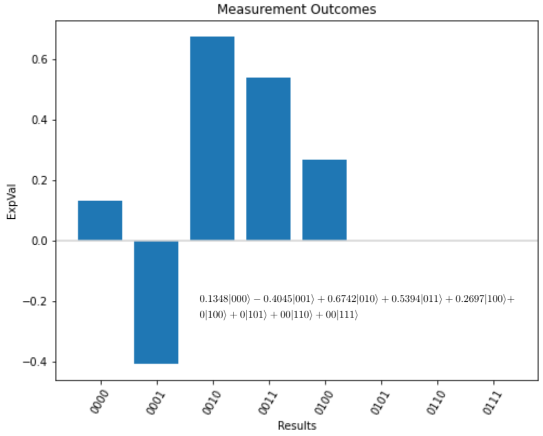
- all qubits (global cost / measurement)
- a few selected qubits (local cost / measurement)
- groups of qubits (each as a variable value)
- as counts of outcomes (repeated measurements)
- as probabilities of outcomes (e.g. P(|0111⟩))
- as Pauli expectation values (i.e. of eigenvalues)
- as expectation of interpreted values (e.g. 0 to 15)
- as variance, etc.

Repeated circuit measurement can be interpreted as outcomes of different types, e.g.

- as a probability distribution (as is)
- as a series of values (via expvals)
- as a binary outcome:
  single qubit measurement or parity of kets
- as an integer:
  most probable ket in multi-qubit measurement
- as a continuous variable:
  probability of the selected ket (e.g. $|0^n\rangle$)







Or we can measure expectation values of the circuit state and interpret them as a series of values

Beware that
adding 1 measurement →
doubles the number of outcomes!

So... having ***n***
measurements leads to ***2ⁿ***
outcomes

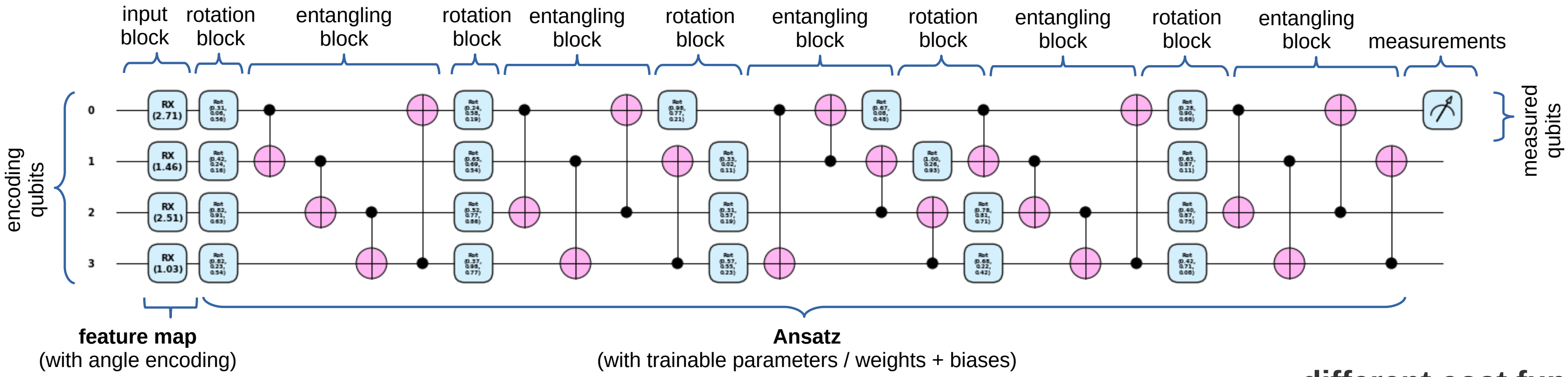
Maria Schuld and Francesco Petruccione.
Machine Learning with Quantum Computers. 2nd ed. Springer, 2021.

Ansatz design and training

A simple quantum classifier ...

Beware that
adding qubits adds
parameters and entanglements!

The number of states represented by the
circuit grows exponentially with the
number of qubits!



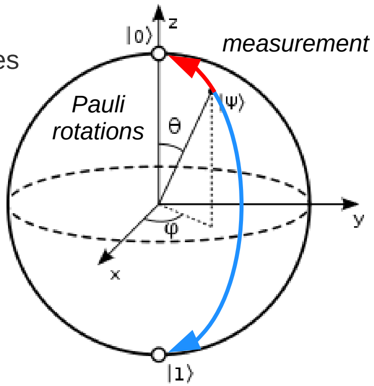
feature maps vary in:
structure and function

ansatze vary in:

- width (qubits #)
- depth (layers #)
- dimensions (param #)
- structure (e.g. funnelling)
- entangling (circular, linear, sca)

ansatz layers consist of:
rotation blocks and entangling blocks
of R(x, y, z) and CNOT gates
(rotation) (entanglement)

rotation gates
alter qubit states
around x, y, z
axes



To execute a circuit we just apply it to input data
and the optimum parameters

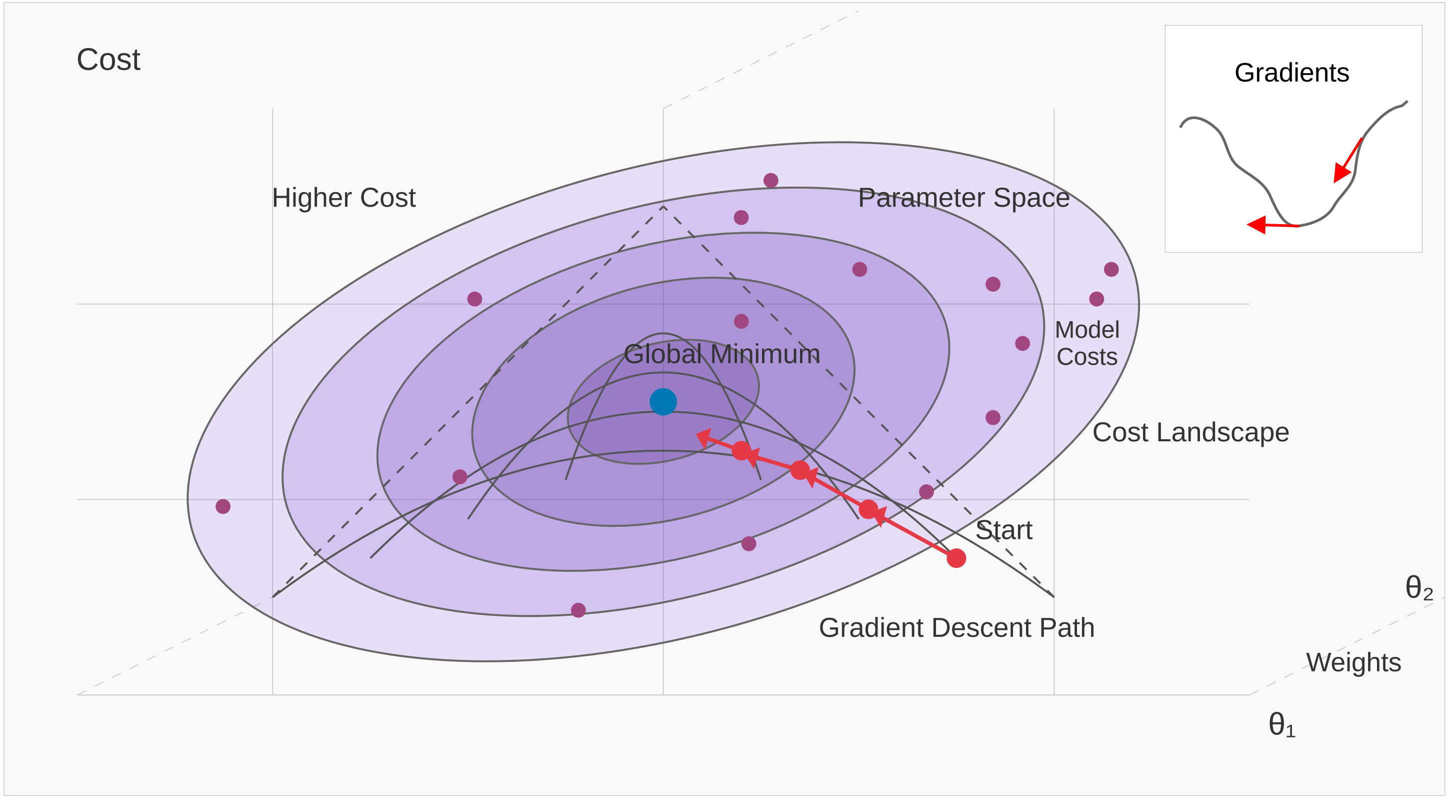
different cost functions:
R2, MAE, MSE, Huber, Poisson, cross-entropy,
hinge-embedding, Kullback-Leibner divergence

different optimisers:
gradient based (Adam, NAdam and SPSA)
linear approximation methods (COBYLA)
non-linear approximation methods (BFGS)
quantum natural gradient optimiser (QNG)

circuit execution on:
simulators (CPUs), accelerators (GPUs) and
real quantum machines (QPUs)

Problem-solving with DL models

Classical model optimisation



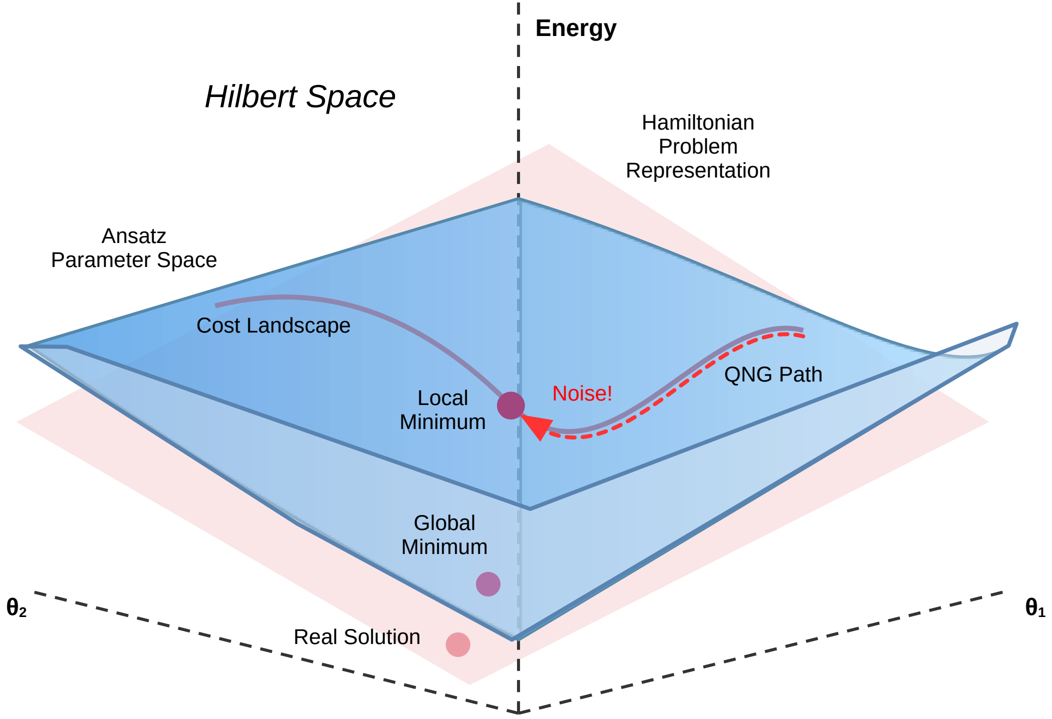
Gradients are local, i.e. their changes influence only their immediate neighbourhood

- A Deep Learning model aims to *represent the problem*.
- It is parameterised with *weights* and *biases*.
- The model quality is linked to the *cost function*, where the lower the cost, the better is the model.
- The costs of all possible model parameterisations, form a multi-dim surface, or the *cost landscape*.
- The *optimisation process* relies on the shape of the landscape, which in turn is reflected in the *gradient* of points on the cost surface.
- *Gradient descent* algorithms can assist in the identification of the model with the *minimum cost*.
- *Backpropagation* can also be used to very efficiently re-calculate NN weights.

Problem-solving with QML

Quantum model optimisation

- The abstract mathematical model represents the problem to be solved, e.g. in the form of a *Hamiltonian*
- The Hamiltonian defines some geometry in *Hilbert space* with the optimum solution associated with the optimum energy
- The *ansatz* of a quantum circuit approximates the Hamiltonian and therefore the problem
- The parameters of the ansatz define a *parameter space* that overlaps and intersects the Hamiltonian problem space
- Our search for the problem solution is hence restricted to the ansatz parameter space
- Ideally, the selected *optimiser* and the *cost function* should understand the principles and processes of quantum models, e.g. Quantum Natural Gradient (QNG) optimiser
- The QNG method defines gradients, which are then calculated for the *cost landscape* (or the manifold), which spans the ansatz geometry
- The QNG optimiser can then identify a *local optimum* solution for this ansatz
- Noise can prevent finding the local optimum

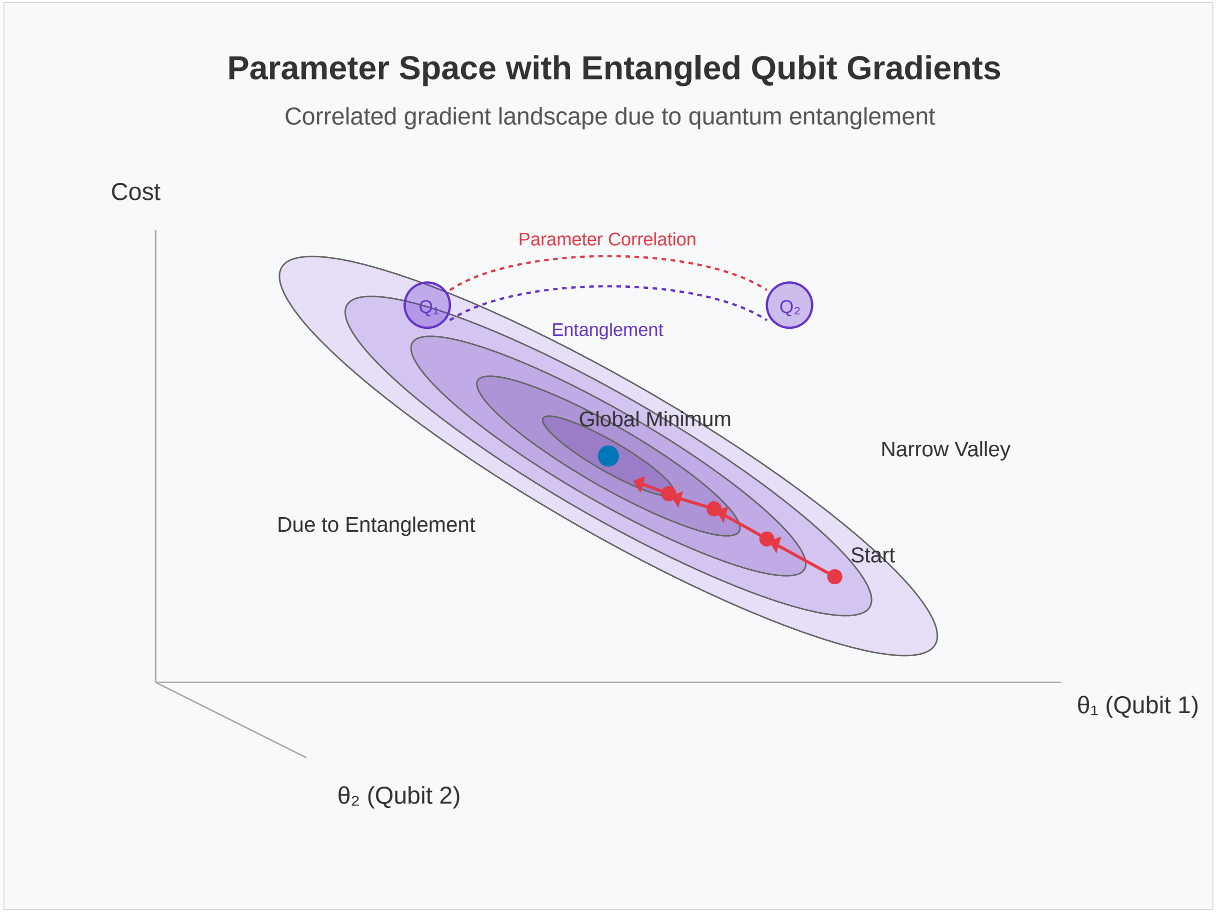


It is more complex than problem-solving with classical models

Quantum model optimisation

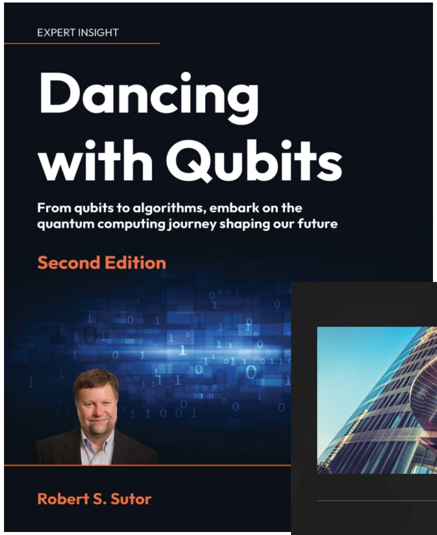
- Optimisation of quantum model needs unique approaches due to the emergence of *non-local gradients*
- *Entangled qubits* result in *correlated parameters and gradients*, so the changes to one are reflected in the distant others

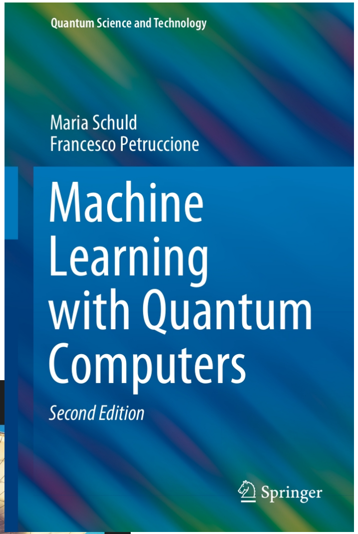
- The cost landscape of highly entangled circuits commonly features *narrow valleys*
- Also, *backpropagation* cannot be used directly in training quantum circuits, as their state is not directly accessible and the measurement collapses the state
- *Gradient descend* can still be used with *global gradients*, i.e. those derived from the geometry of the cost landscape
- *Stochastic optimisation techniques* are highly effective when the cost landscape is smooth (no quantum moise)
- Other techniques are also available, such as *particle swarm optimisers*, these however are applicable to smaller models

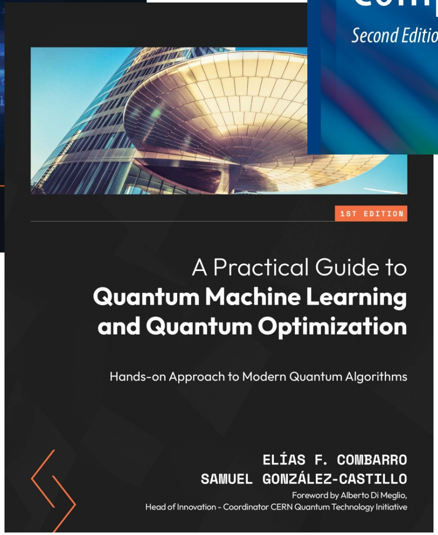


Recommended reading

on QML







14 Feb 2020

PennyLane: Automatic differentiation of hybrid quantum-classical computations

Ville Bergholm,[1] Josh Izaac,[1] Maria Schuld,[1] Christian Gogolin,[1] M. Sohaib Alam,[2] Shahnawaz Ahmed,[3] Juan Miguel Arrazola,[1] Carsten Blank,[4] Alain Delgado,[1] Soran Jahangiri,[1] Keri McKiernan,[2] Johannes Jakob Meyer,[5] Zeyue Niu,[1] Antal Száva,[1] and Nathan Killoran[1]

[1]Xanadu, 777 Bay Street, Toronto, Canada
[2]Rigetti Computing, 2919 Seventh Street, Berkeley, CA 94710
[3]Wallenberg Centre for Quantum Technology, Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden
[4]data cybernetics, Martin-Kolmsperger-Str 26, 86899 Landsberg, Germany
[5]Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany

tum computing with applications in quantum chemistry [1], quantum optimization [2], factoring [3], state diagonalization [4], and quantum machine learning [5–18]. In a reversal from the usual practices in quantum computing research, a lot of research for these mostly heuristic algorithms necessarily focuses on numerical experiments rather than rigorous mathematical analysis. Luckily, there are various publicly accessible platforms to simulate quantum algorithms [19–26] or even run them on real quantum devices through a cloud service [27, 28]. However, even though some frameworks are designed with variational circuits in mind [25, 29, 30], there is at this stage no unified tool for the hybrid optimization of quantum circuits across quantum platforms, treating all simulators and devices on the same footing.

PennyLane is an open-source Python 3 framework that facilitates the optimization of quantum and hybrid quantum-classical algorithms. It extends several seminal ma-

Modern applications of machine learning in quantum sciences

Anna Dawid[1,2*], Julian Arnold[3†], Borja Requena[2†], Alexander Gresch[4†], Marcin Płodzień[2], Kaelan Donatella[5], Kim A. Nicoli[6,7], Paolo Stornati[2], Rouven Koch[8], Miriam Büttner[9], Robert Okuła[10,11], Gorka Muñoz–Gil[12], Rodrigo A. Vargas–Hernández[13,14], Alba Cervera-Lierta[15], Juan Carrasquilla[14], Vedran Dunjko[16], Marylou Gabrié[17], Patrick Huembeli[18,19], Evert van Nieuwenburg[16,20], Filippo Vicentini[18], Lei Wang[21,22], Sebastian J. Wetzel[23], Giuseppe Carleo[18], Eliška Greplová[24], Roman Krems[25], Florian Marquardt[26,27], Michał Tomza[1], Maciej Lewenstein[2,28] and Alexandre Dauphin[2*]

1 Faculty of Physics, University of Warsaw, Poland
2 ICFO - Institut de Ciències Fotòniques, The Barcelona Institute of Science and Technology, 08860 Castelldefels (Barcelona), Spain
3 Department of Physics, University of Basel, Switzerland
4 Quantum Technology Research Group, Heinrich-Heine-Universität Düsseldorf, Germany
5 Université de Paris, CNRS, Laboratoire Matériaux et Phénomènes Quantiques, France
6 Machine Learning Group, Technische Universität Berlin, Germany
7 BIFOLD, Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany
8 Department of Applied Physics, Aalto University, Espoo, Finland
9 Institute of Physics, Albert-Ludwig University of Freiburg, Germany
10 International Centre for Theory of Quantum Technologies, University of Gdańsk, Poland
11 Department of Algorithms and System Modeling, Faculty of Electronics, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Poland
12 Institute for Theoretical Physics, University of Innsbruck, Austria
13 Department of Chemistry, University of Toronto, Canada
14 Vector Institute for Artificial Intelligence, MaRS Centre, Toronto, Canada
15 Barcelona Supercomputing Center, Spain
16 LIACS, Leiden University, The Netherlands
17 CMAP, École Polytechnique, France
18 Institute of Physics, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
19 Menten AI, Inc., Palo Alto, California, United States of America
20 Niels Bohr Institute, Copenhagen, Denmark
21 Beijing National Lab for Condensed Matter Physics and Institute of Physics, Chinese Academy of Sciences, Beijing, China
22 Songshan Lake Materials Laboratory, Dongguan, China
23 Perimeter Institute for Theoretical Physics, Waterloo, Canada
24 Kavli Institute of Nanoscience, Delft University of Technology, NL-2600 GA Delft, The Netherlands
25 Department of Chemistry, University of British Columbia, Vancouver, Canada
26 Max Planck Institute for the Science of Light, Erlangen, Germany
27 Department of Physics, Friedrich-Alexander Universität Erlangen-Nürnberg, Germany
28 ICREA, Pg. Lluís Companys 23, 08010 Barcelona, Spain
† These authors contributed equally.

* Anna.Dawid@fuw.edu.pl, Alexandre.Dauphin@icfo.eu

June 23, 2022

Abstract

In these Lecture Notes, we provide a comprehensive introduction to the most recent advances in the application of machine learning methods in quantum sciences. We cover the use of deep learning and kernel methods in supervised, unsupervised, and reinforcement learning algorithms for phase classification, representation of many-body quantum states, quantum feedback control, and quantum circuits optimization. Moreover, we introduce and discuss more specialized topics such as differentiable programming, generative models, statistical approach to machine learning, and quantum machine learning.

1

Thank you!

Any questions?