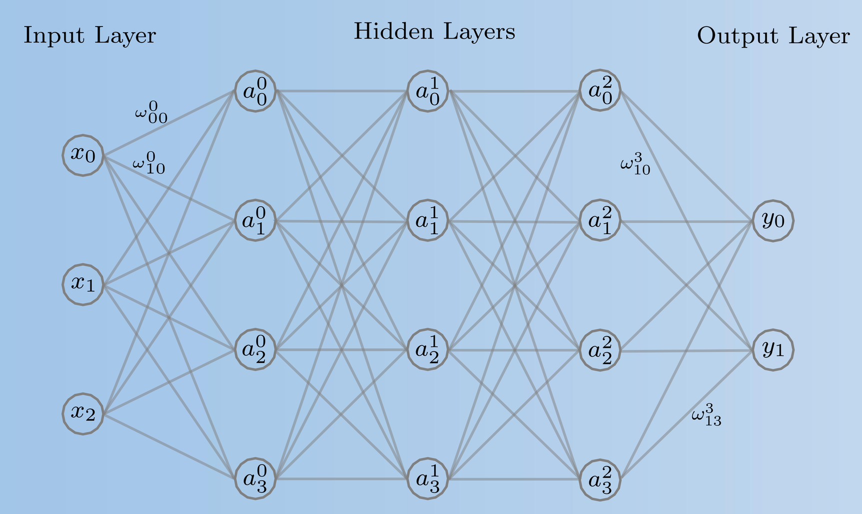
Machine Learning on Quantum Systems

Nils-Erik Schütte*[1,2], Niclas Götting[2], Hauke Müntinga[1], Meike List[1,3] and Christopher Gies[2]

[1]German Aerospace Center (DLR), Institute for Satellite Geodesy and Inertial Sensing, Bremen
[2]Institute for Theoretical Physics, University of Bremen, Bremen
[3]University of Bremen, Bremen
*nils-erik.schuette@dlr.de

Why do we want to do quantum machine learning?

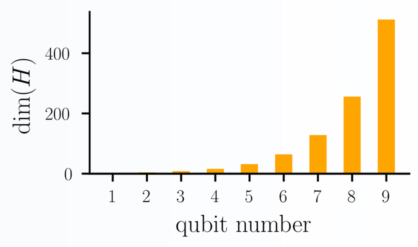
Machine learning (ML) enables us to use computer systems for human-like tasks. Neural networks are inspired by the human brain and consist of neurons that are connected by edges modelling the synapses of a brain.

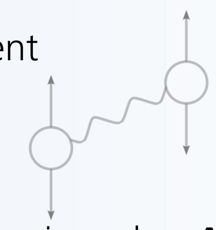


Quantum computing describes information processing with a device, whose working principles are governed by the laws of quantum mechanics.

Quantum systems possess unique properties that scientists try to exploit to gain advantages and functionalities beyond classical ML:

- feature maps into exponentially large phase spaces
- native processing of quantum input
- non-classical correlations via entanglement



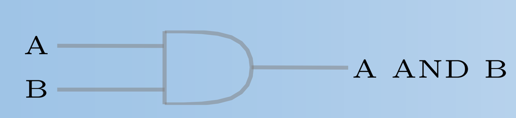


Quantum machine learning (QML) can be viewed as **A)** implementing algorithms through quantum gates on quantum computers, or **B)** using the inherent temporal dynamics of quantum systems as input-output map.

A) Gate-Based Approach: Parameterized Quantum Circuits (PQCs)

Classical Computer

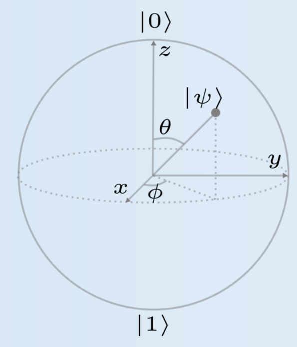
- basic unit of information: bit (0 or 1)
- NOT-gate is the only non-trivial single-bit gate
- AND-gate as example for a two-bit gate



Quantum Computer

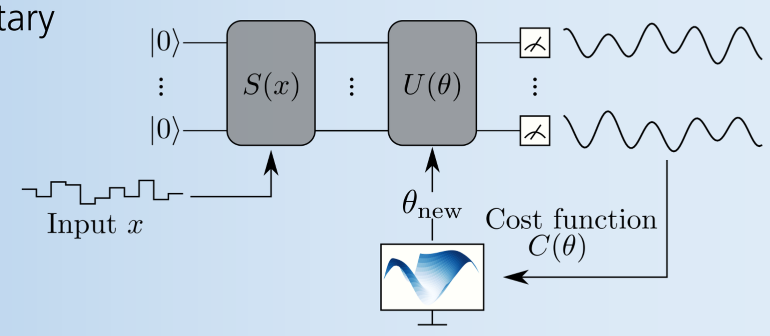
- information is stored in quantum bits (qubits)
- infinite number of single-qubit gates, which rotate the state on the Bloch sphere
- arbitrary superpositions are possible

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle$$



Parameterized Quantum Circuits (PQCs)

- gates are represented by unitary matrices
- PQCs are realized by making the unitary matrices dependent on real parameters



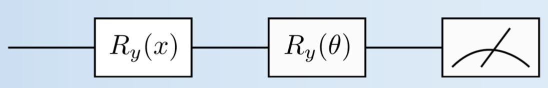
Workflow:

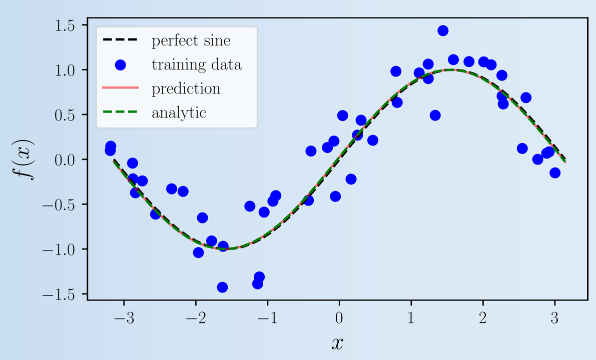
1. use unitary gates $S(x)$ to embed data into the quantum system
2. use a block of parameterized gates $U(\theta)$ (called ansatz) as trainable part of the circuit
3. perform measurement on the system
4. use an optimizer on a classical computer to update the parameters

Why does it work - an example:

- use a single qubit PQC to fit a noisy sine curve
- inputs: $x \in [-\pi, \pi]$
- parameter: $\theta \in [-\pi, \pi]$
- the measured expectation value is given analytically by

$$f_\theta(x) = \langle\psi(x,\theta)\,|\,Z\,|\,\psi(x,\theta)\rangle = \cos(\theta + x)$$

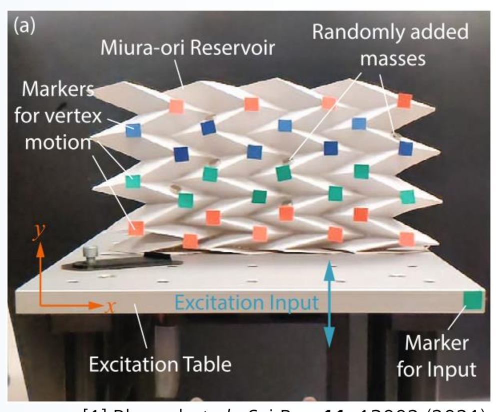




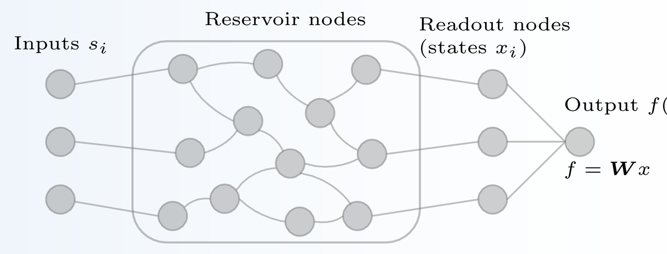
B) Complex Dynamics Approach: Quantum Reservoir Computing (QRC)

Classical Reservoir Computing

- inspired by the human brain
- a reservoir computer is a recurrent neural network with fixed weights; only the weights of the linear readout layer are trained with a simple linear regression
- computation may be viewed as any transformation of an input signal to an output signal arising from the intrinsic system dynamics
- physical RC: the reservoir is realized by a real physical system, e.g. an origami-structure [1]



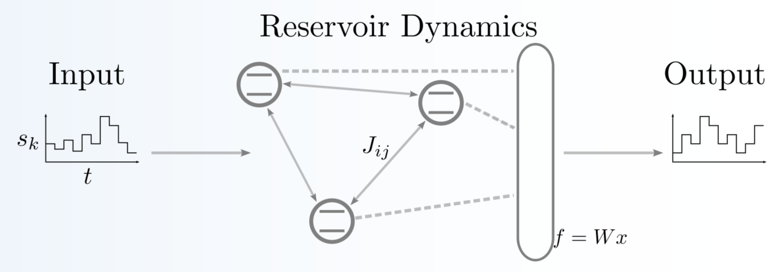
[1] Bhovad *et al.*, Sci Rep **11**, 13002 (2021)



Quantum Reservoir Computing (QRC)

- quantum system is used as reservoir
- example is the transverse-field Ising model (TFIM):

$$H = h\sum_{i=1}^{N}\sigma_z^{(i)} + \sum_{i<j} J_{ij}\sigma_x^{(i)}\sigma_x^{(j)}$$



- dynamics of the system are described by the unitary time-evolution operator:

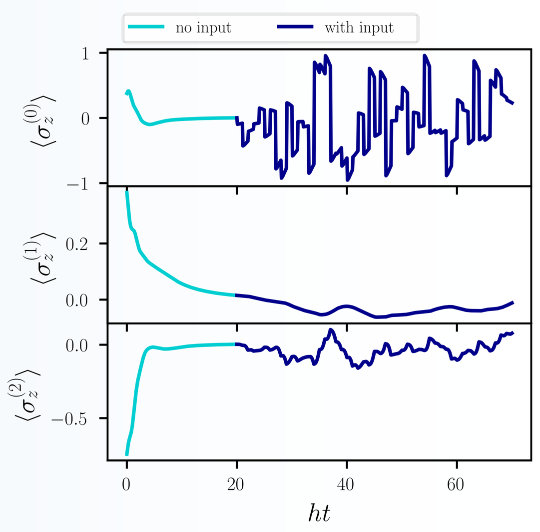
$$U_{\Delta_t} = e^{-iH\Delta_t}$$

- as readout, the expectation values of the σ_z observables are chosen:

$$\langle\sigma_z^{(i)}\rangle = \mathrm{Tr}\{\sigma_z^{(i)}\rho\}$$

- inputs are injected via state initialization:

$$\rho \mapsto |\psi_{s_k}\rangle\langle\psi_{s_k}| \otimes \mathrm{Tr}_1[\rho]$$
$$|\psi_{s_k}\rangle = \sqrt{1-s_k}\,|0\rangle + \sqrt{s_k}\,|1\rangle$$



Goal of the Quantum Fellowship Project

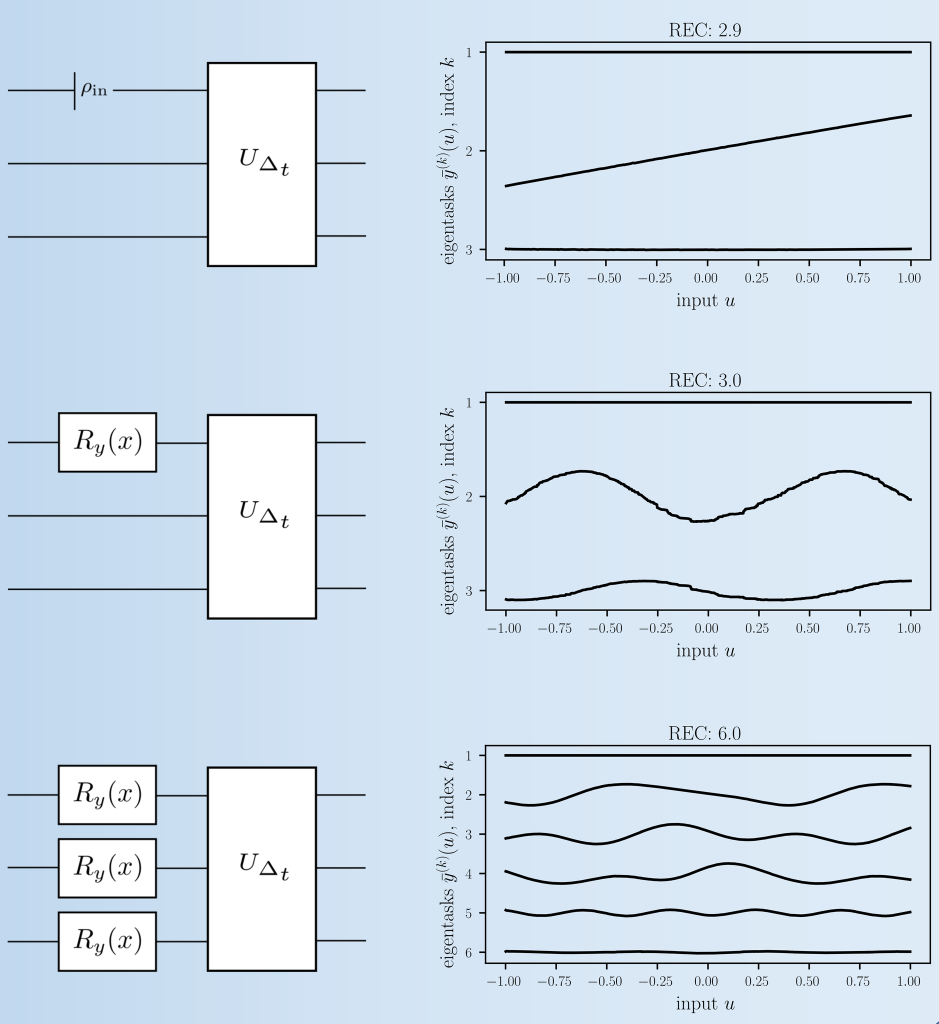
Understand the working mechanisms of QML algorithms that rely on either of the two approaches and compare their capabilities and limitations.

First connection: Embed the TFIM on a gate-based quantum computing architecture and compare it with a PQC.

One goal is to quantify and compare the expressivities of both approaches [2, 3], via calculation of

- eigentasks: set of orthogonal functions which can be optimally approximated by the system
- resolvable expressive capacity (REC): quantification of how many linearly independent functions can be expressed by the system

[2] Schuld *et al.*, Phys. Rev. A **103**, 032430 (2021)
[3] Hu *et al.*, Phys. Rev. X **13**, 041020 (2023)



QML Strategies Side by Side

A) Gate-based approach:

- potential to be realized on currently available noisy intermediate scale quantum (NISQ) hardware
- adaptability to specific problems via quantum circuit design
- hard and tedious to train
- susceptible to device noise

B) Complex dynamics approach:

- any dynamical quantum system can be used as reservoir
- internal dynamics are fixed and cannot be optimized
- very easy to train
- noise is (to some extent) good for the performance

Conclusion

Implementing the unitary time evolution of a dynamical system on a gate-based quantum computer allows the direct comparison with a PQC enabling the study of the expressive power of both approaches in dependence on the input encoding.

University of Bremen

QUANTUM TEAM

DLR