

Measurement-based Quantum Machine Learning

Luis Mantilla



**Unitary
Foundation**

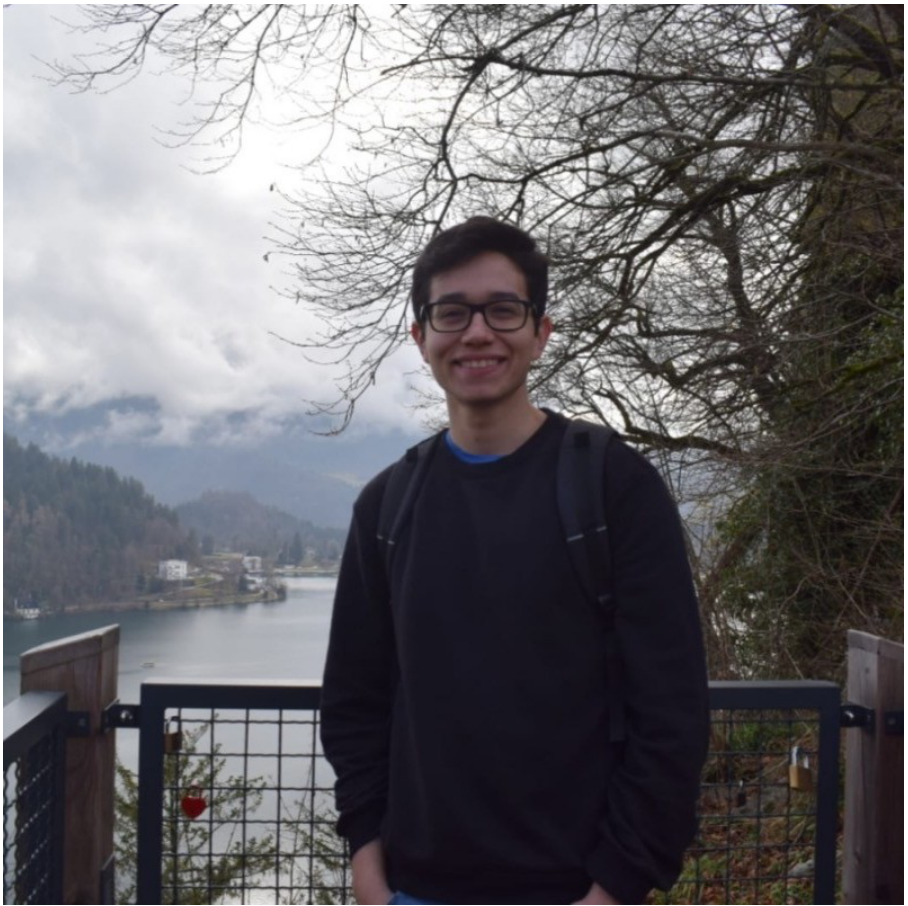
Graphix Workshop, 2025

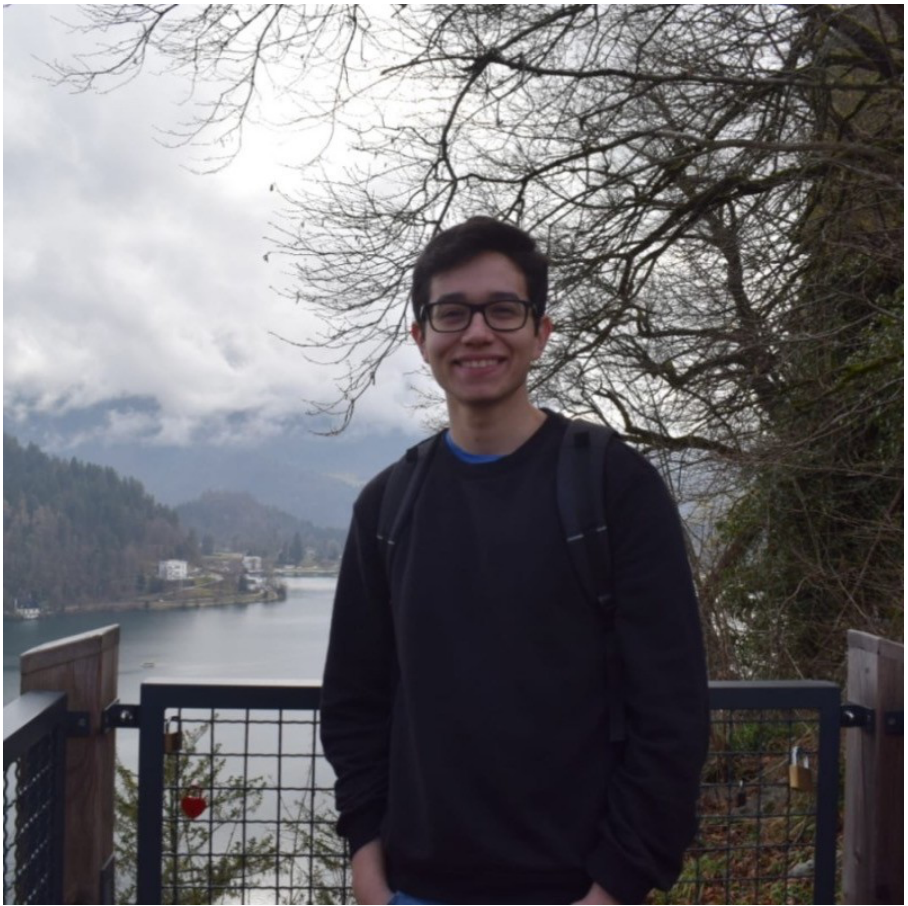
Prev:



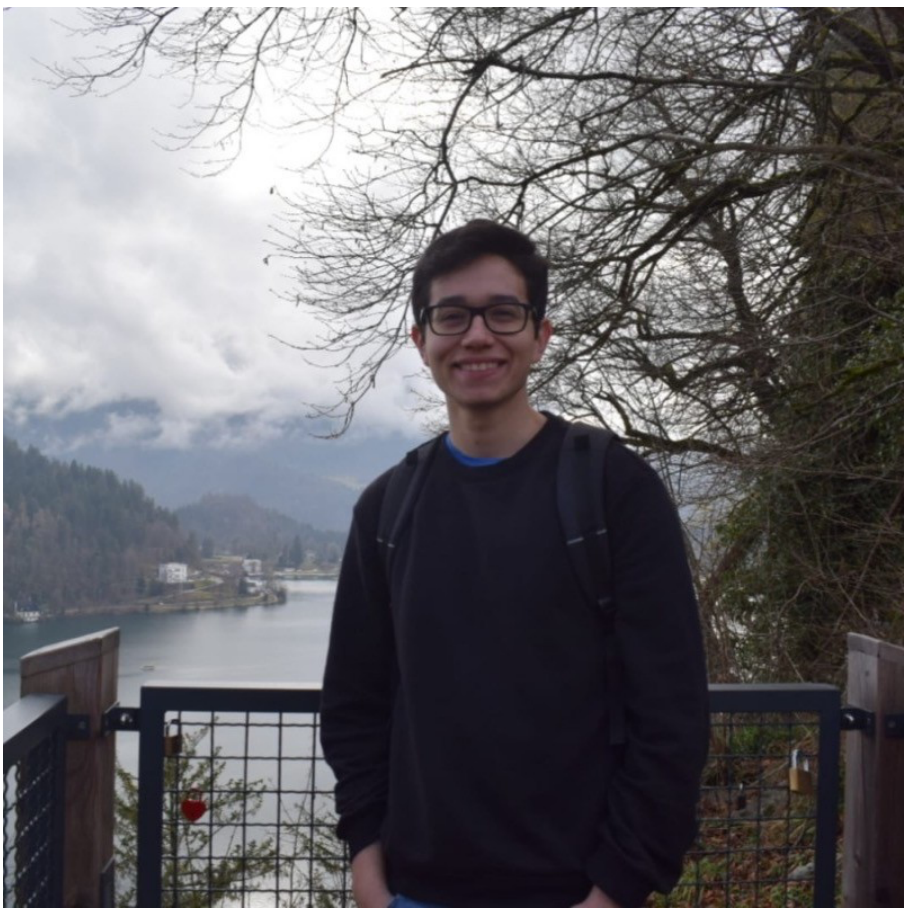
Now:

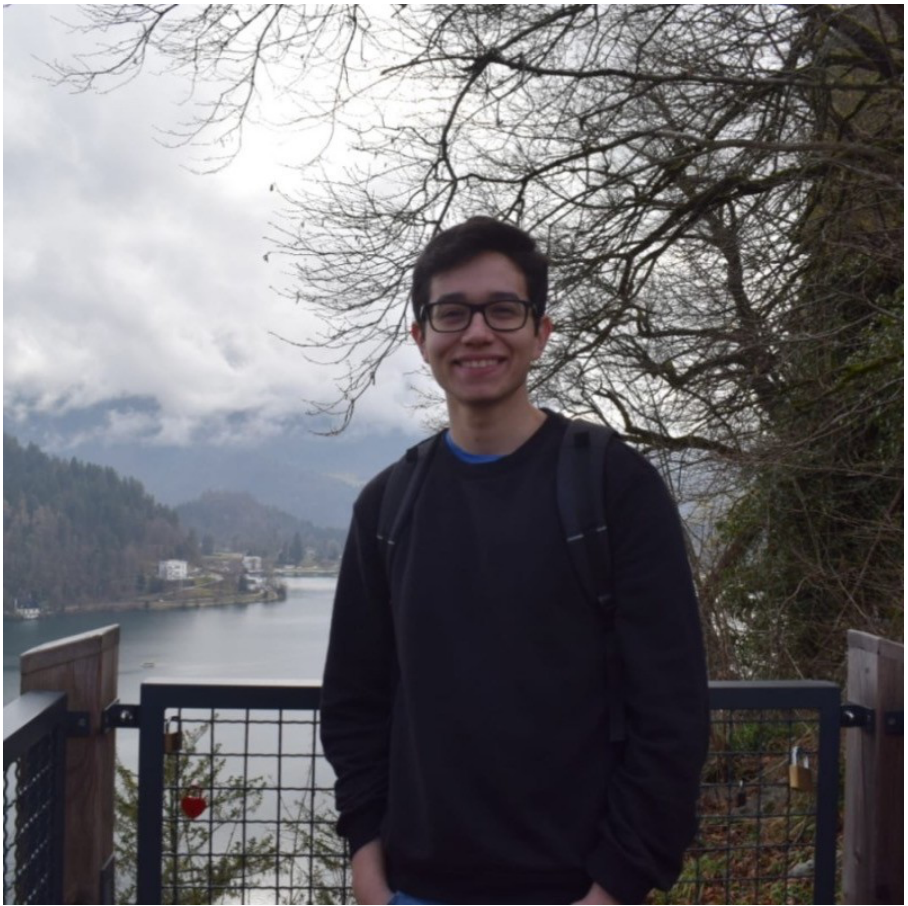


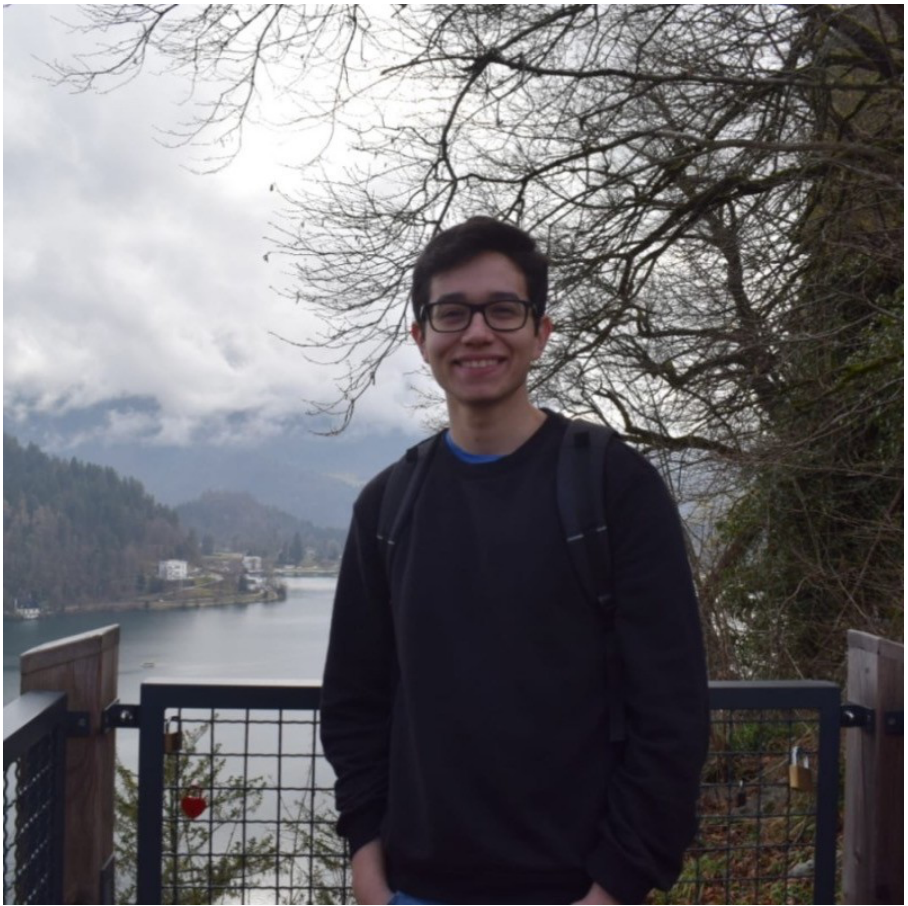


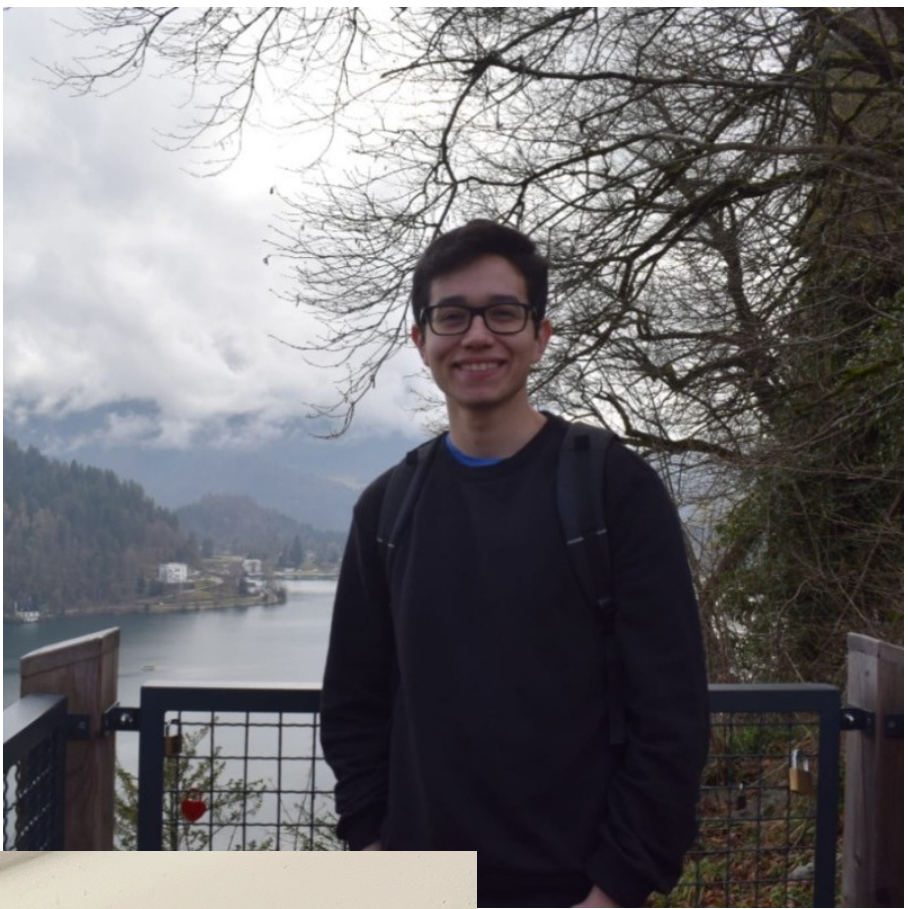
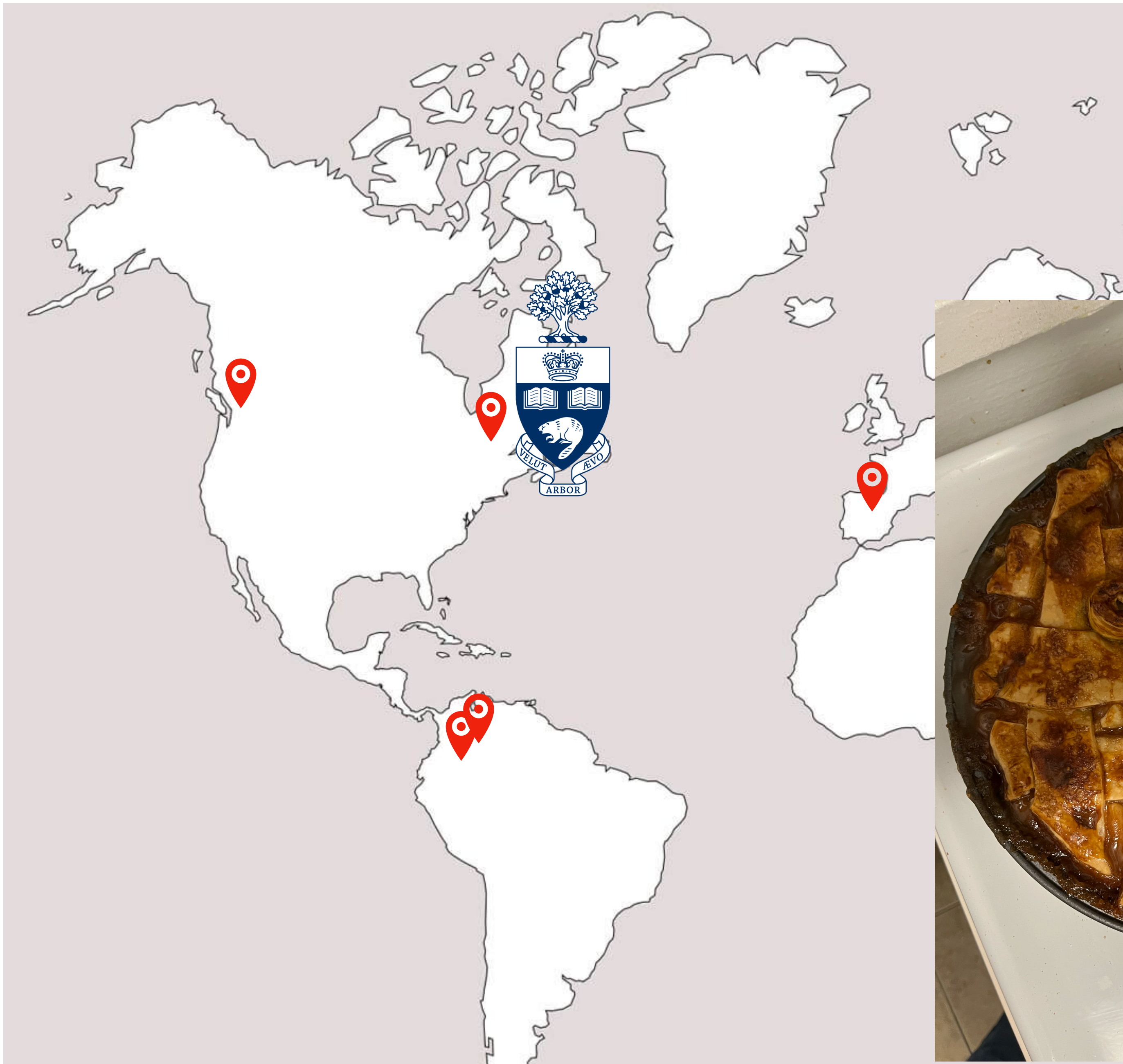


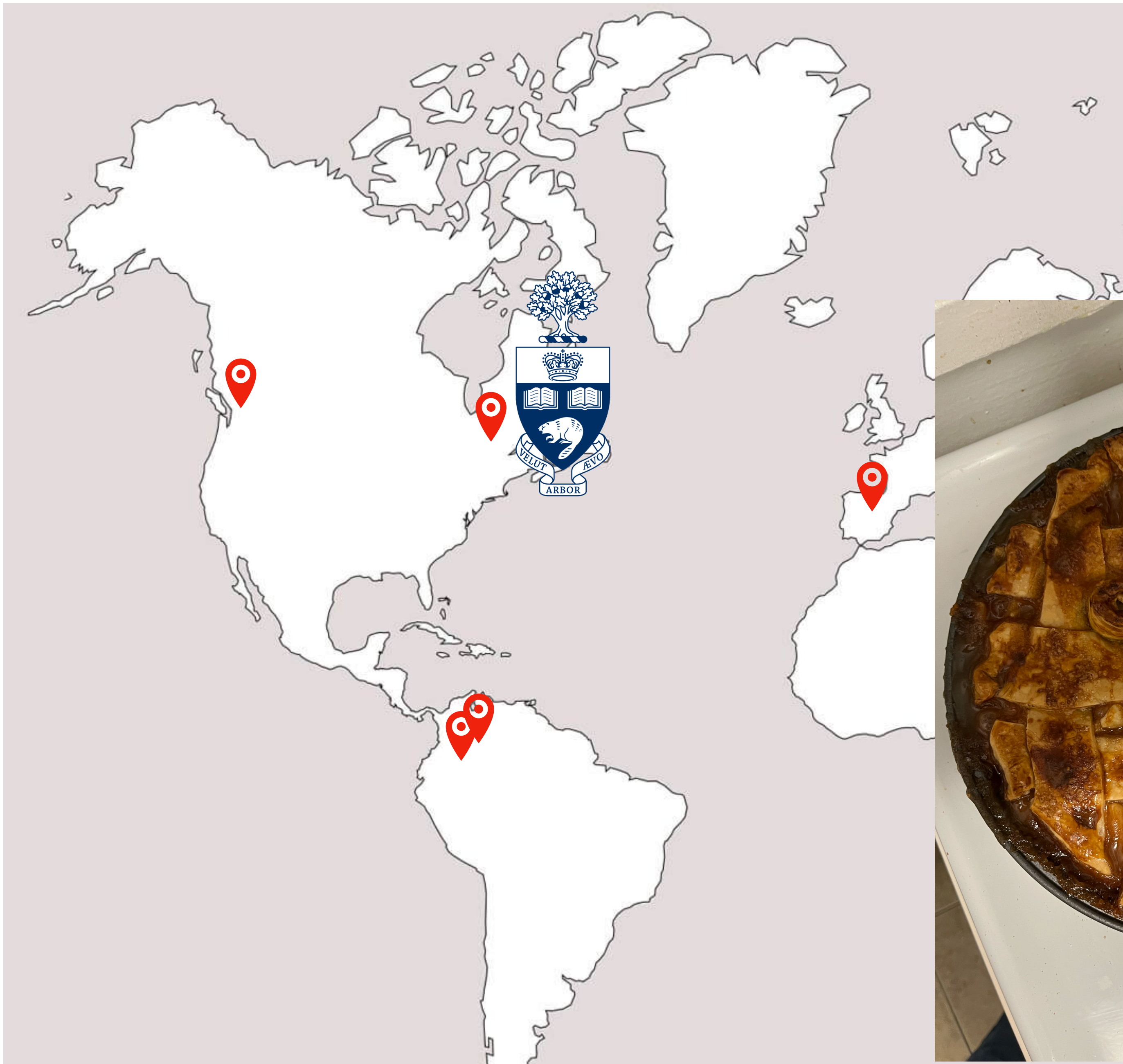


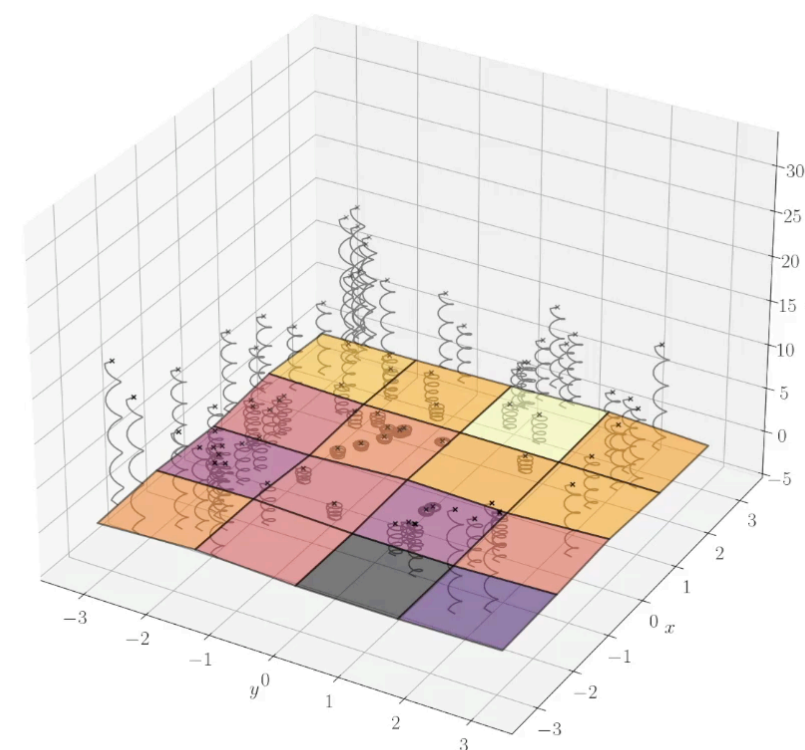








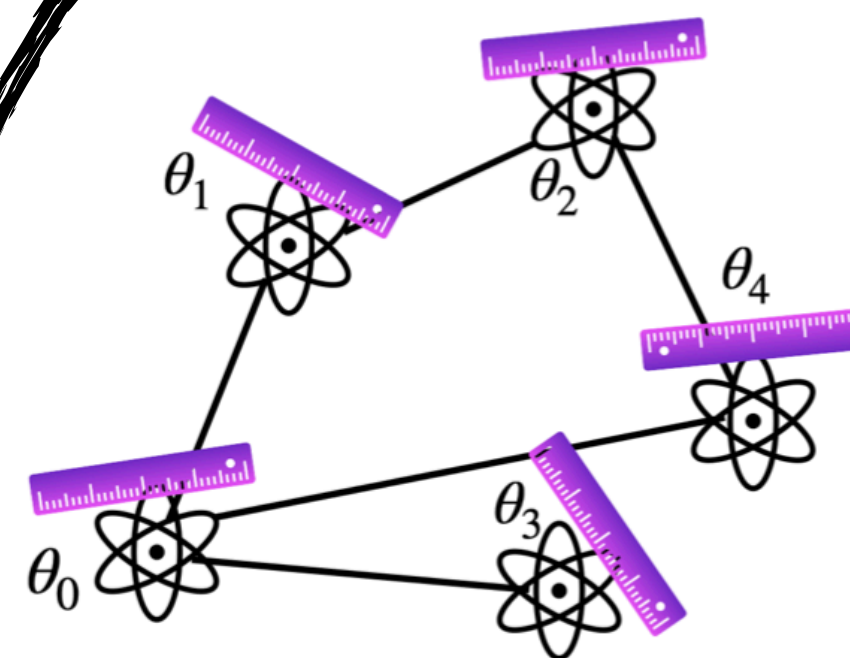
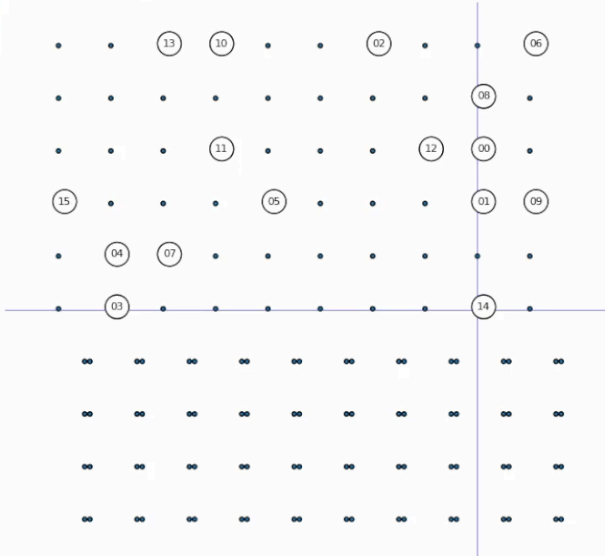




IQEra
Computing Inc.



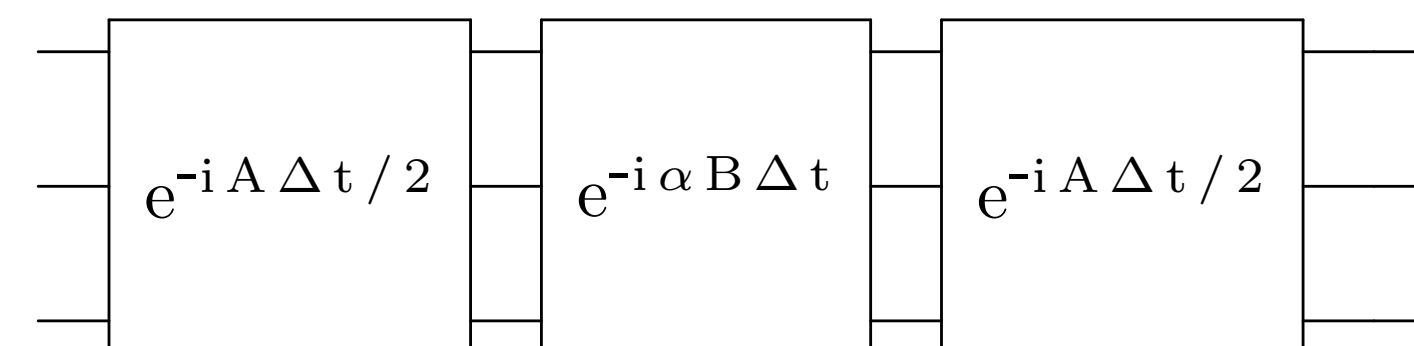
QC-Daemon



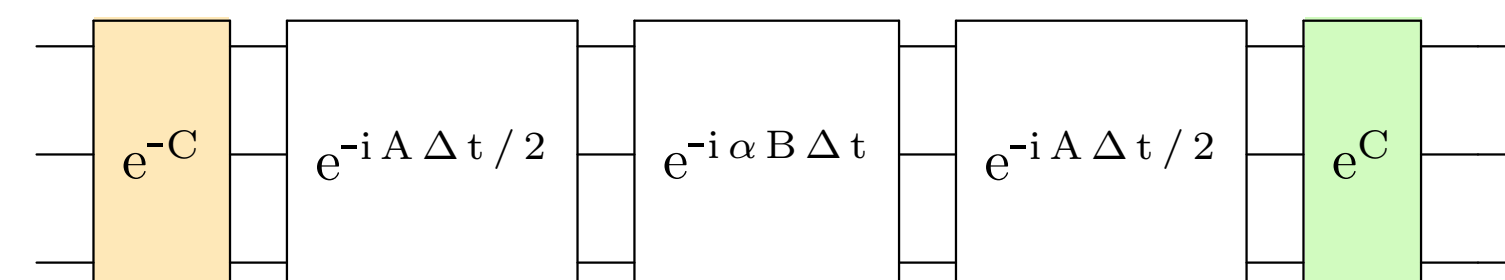
MB-QML

Quantum computing

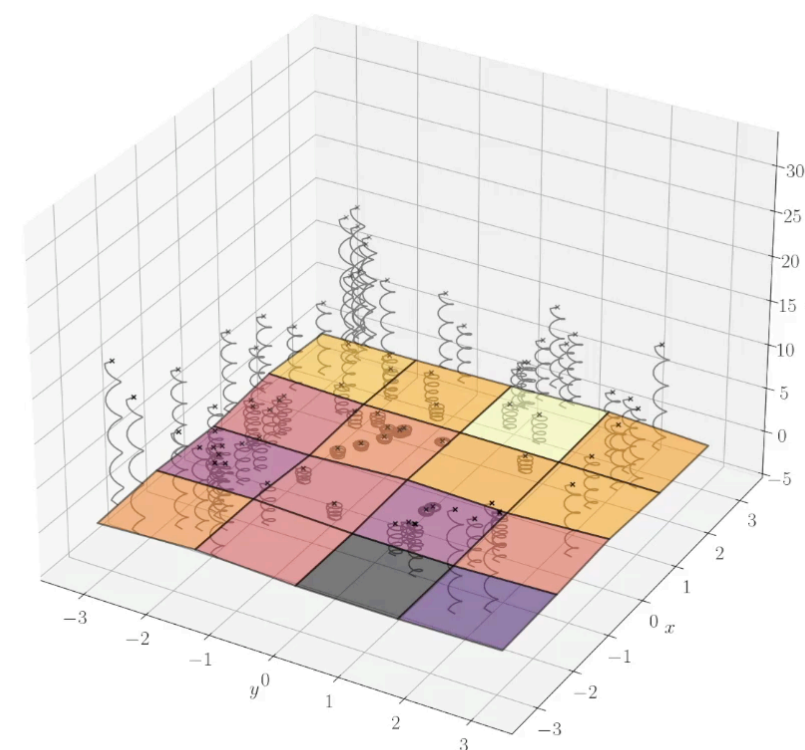
$$\text{PF2} : e^{iH\Delta t} \approx$$



$$\text{CPF2} : e^{iH\Delta t} \approx$$



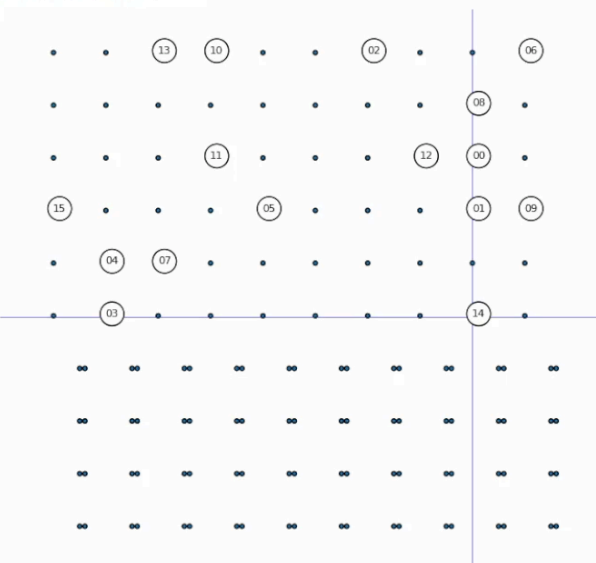
Machine learning



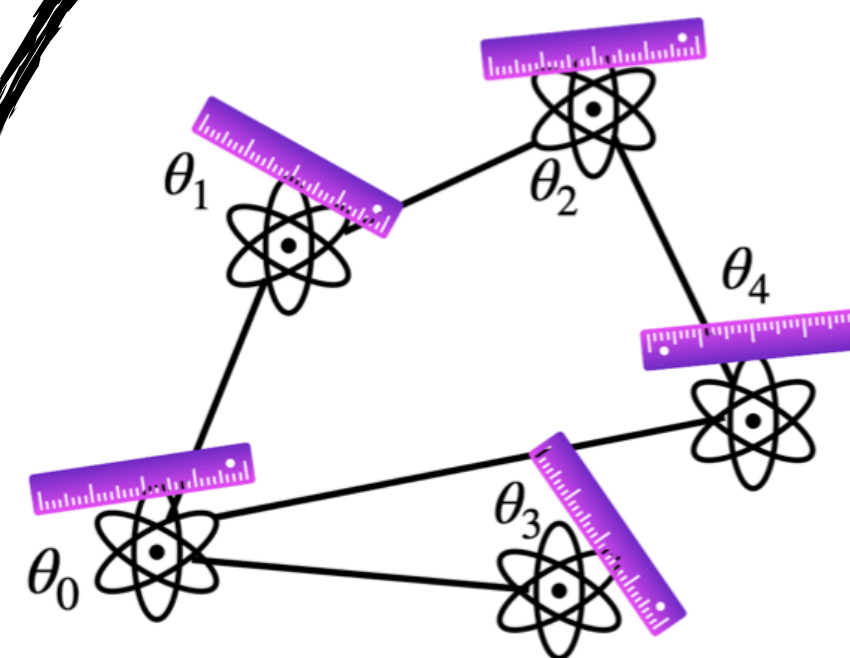
IQEra
Computing Inc.



QC-Daemon



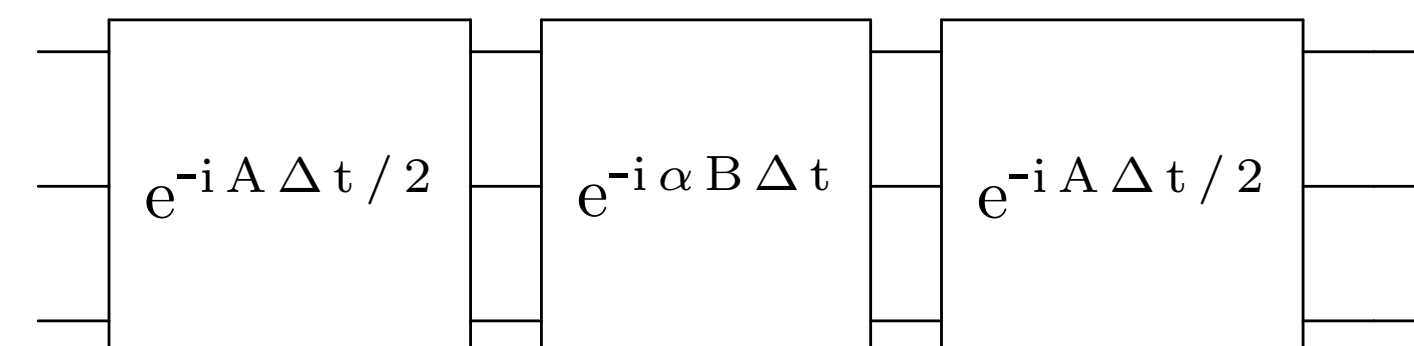
Machine learning



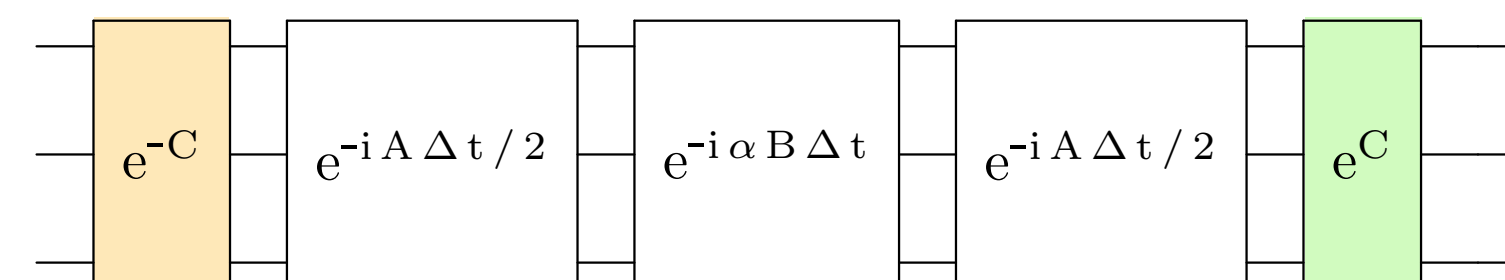
MB-QML

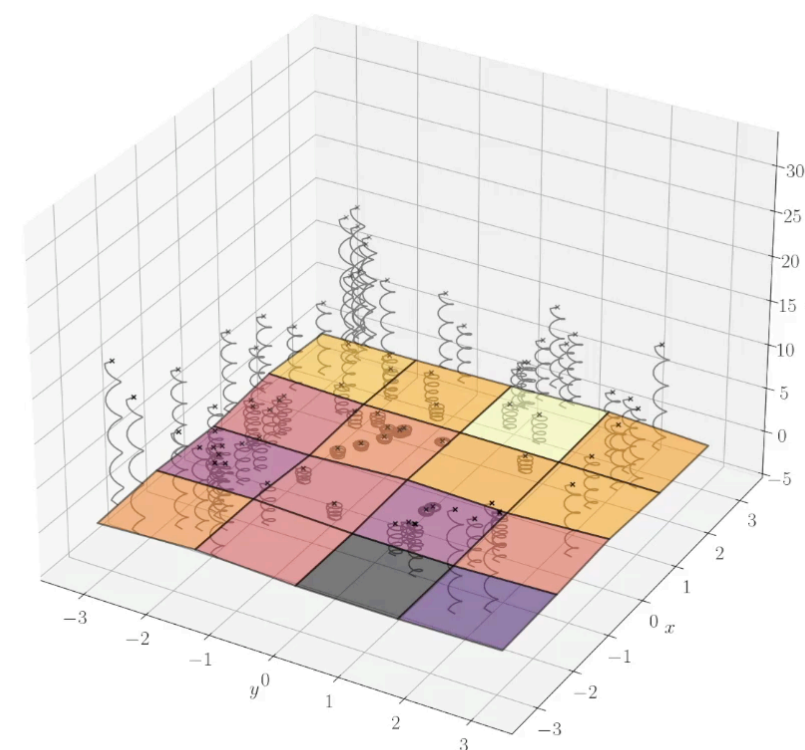
Quantum computing

$$\text{PF2} : e^{iH\Delta t} \approx$$



$$\text{CPF2} : e^{iH\Delta t} \approx$$

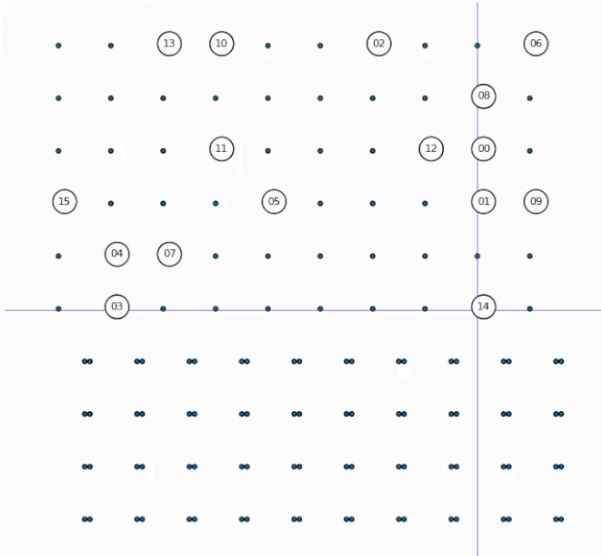




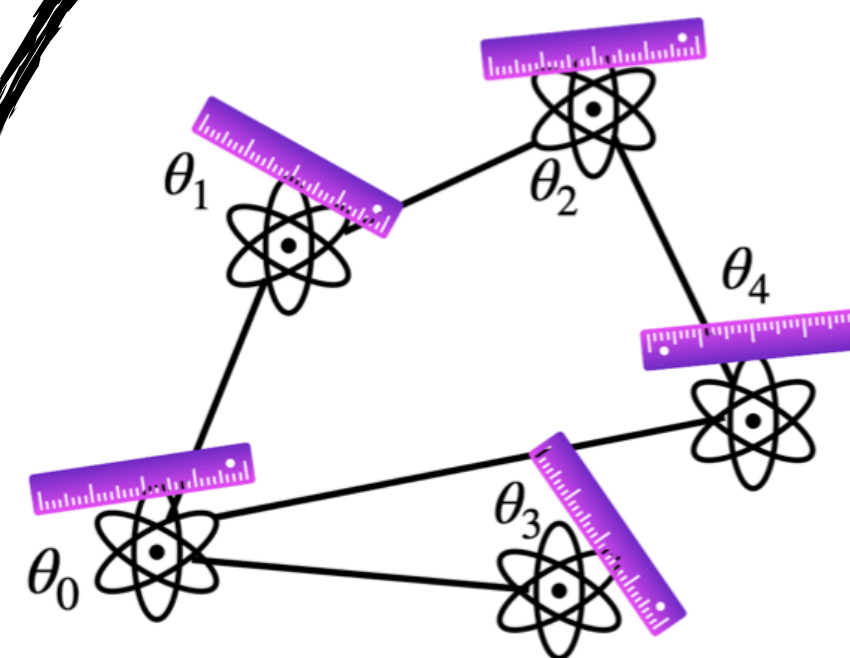
IQEra
Computing Inc.



QC-Daemon



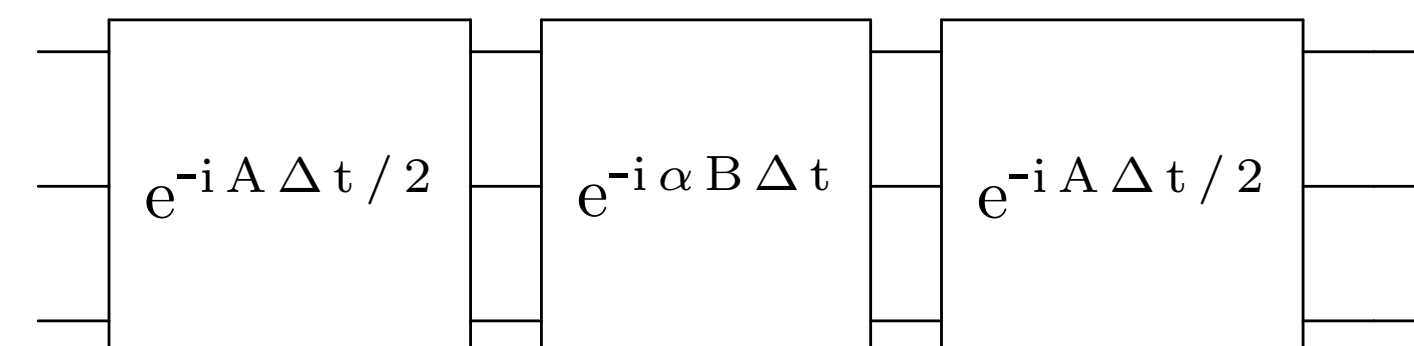
Machine learning



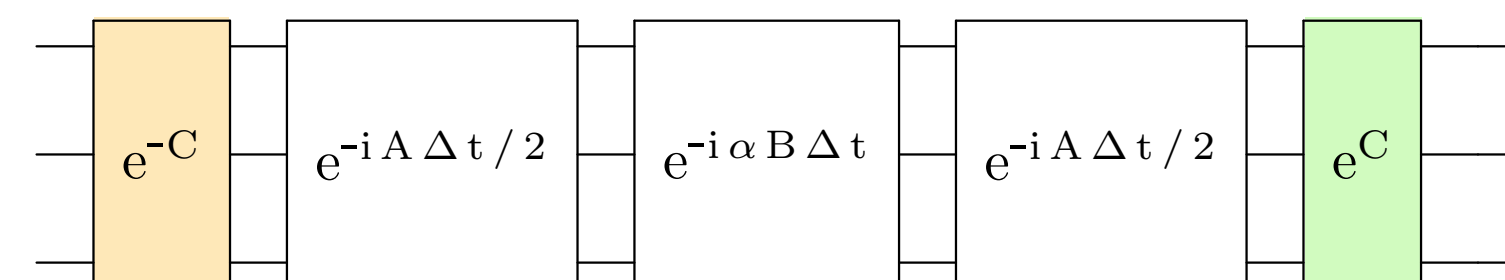
MB-QML

Quantum computing

$$\text{PF2} : e^{iH\Delta t} \approx$$



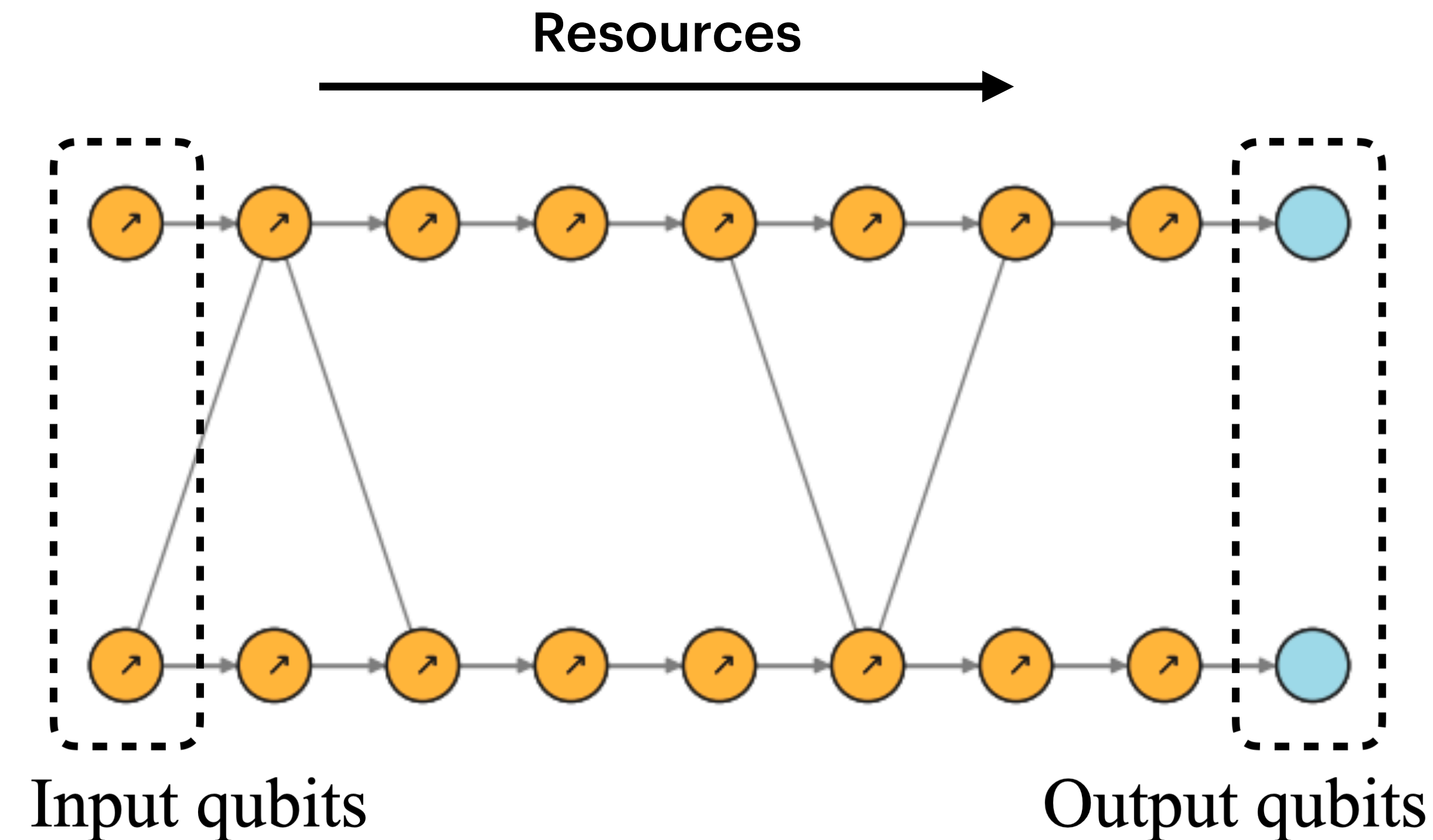
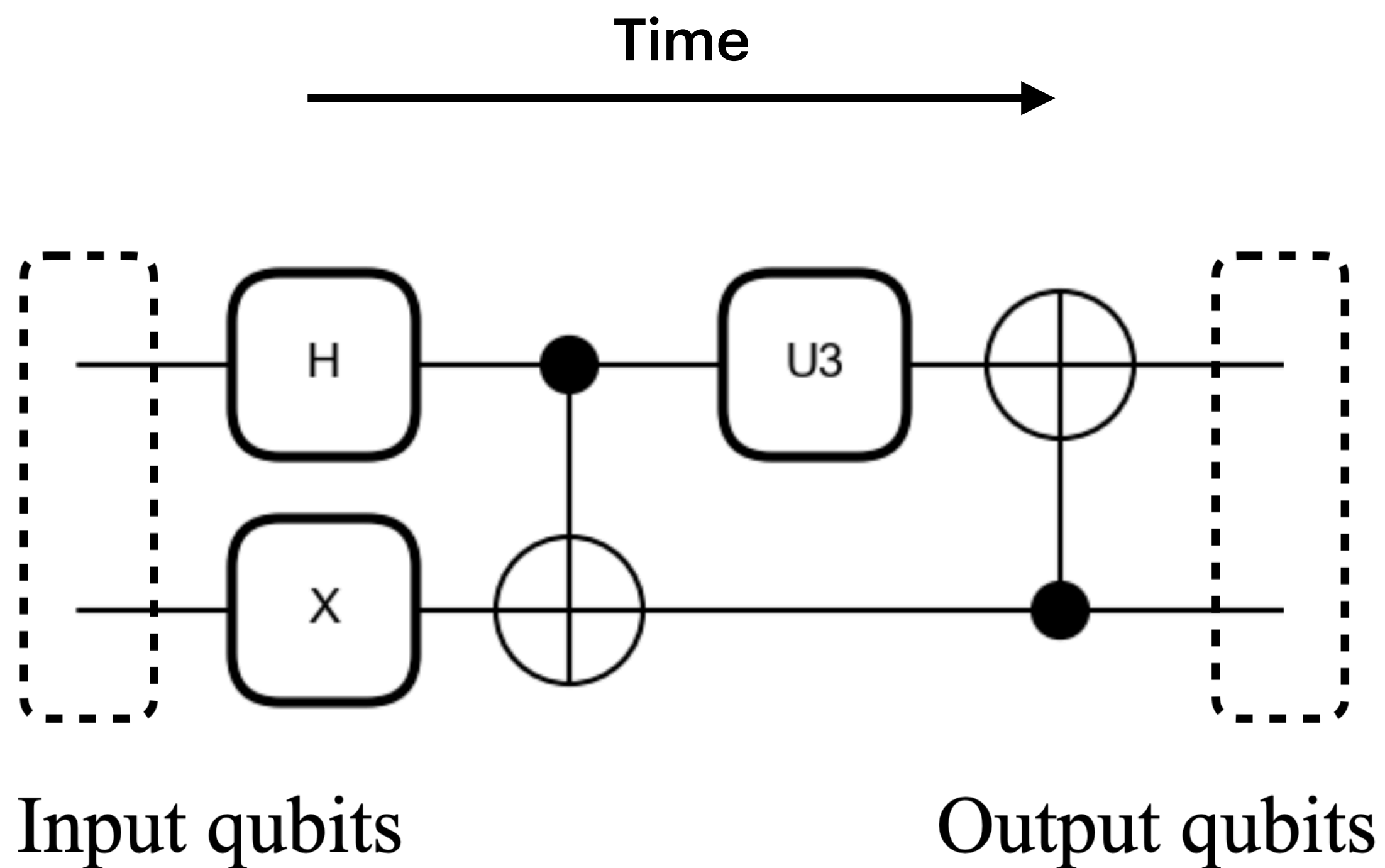
$$\text{CPF2} : e^{iH\Delta t} \approx$$



Measurement-based quantum computation

Circuit model vs MBQC

Simulate the time dimension

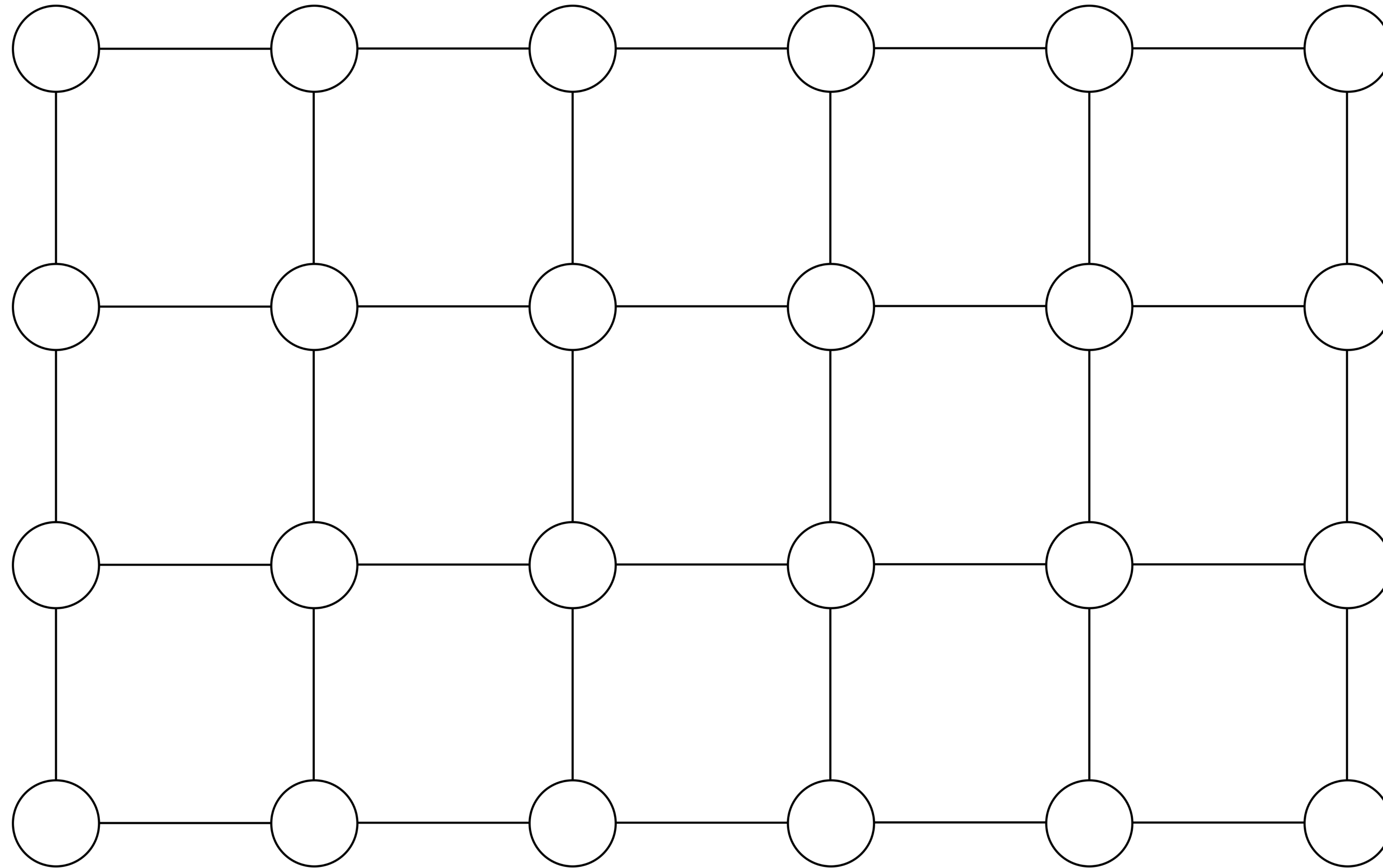


How does MBQC work?

Resource state + single-qubit measurements

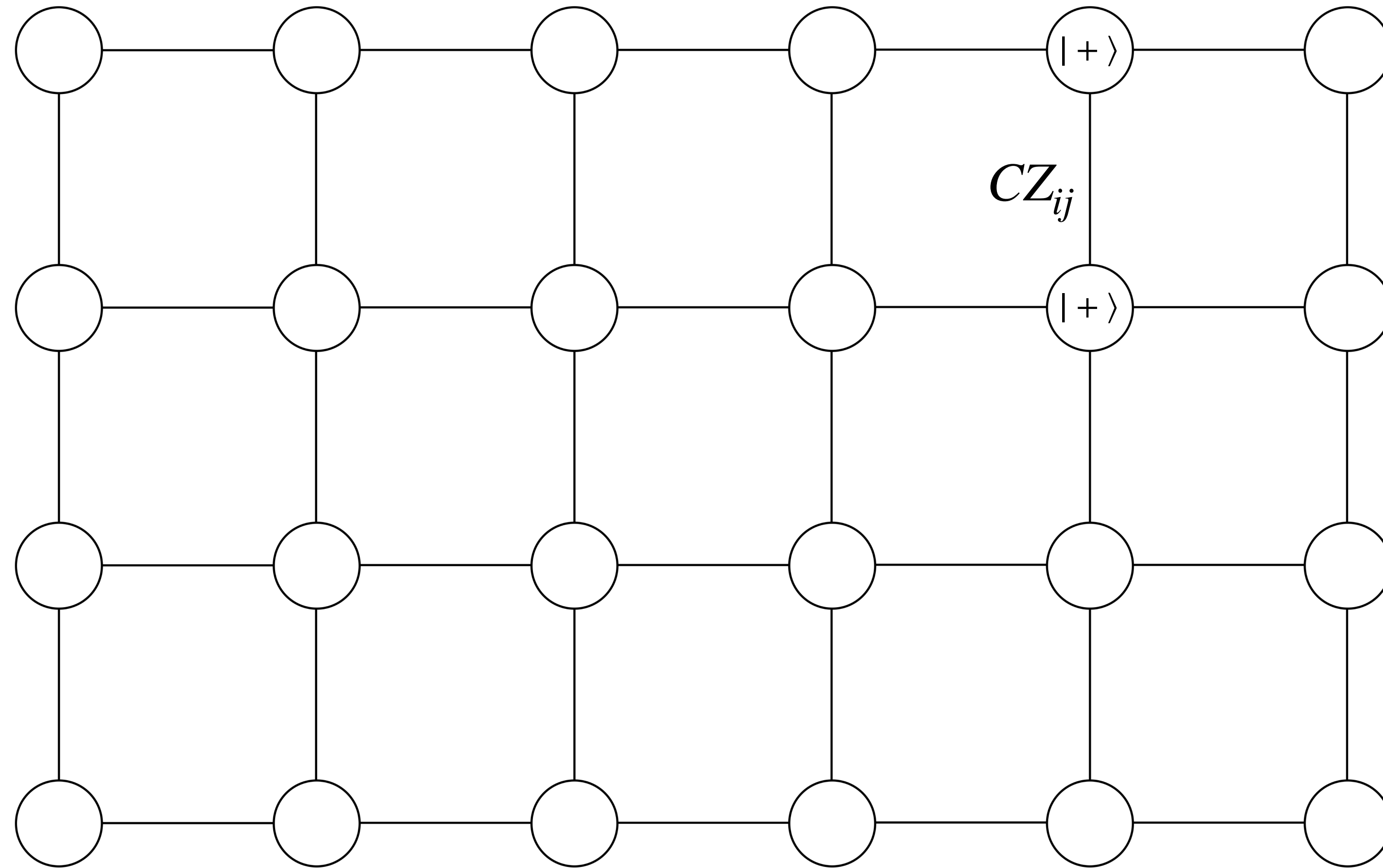
How does MBQC work?

Resource state + single-qubit measurements



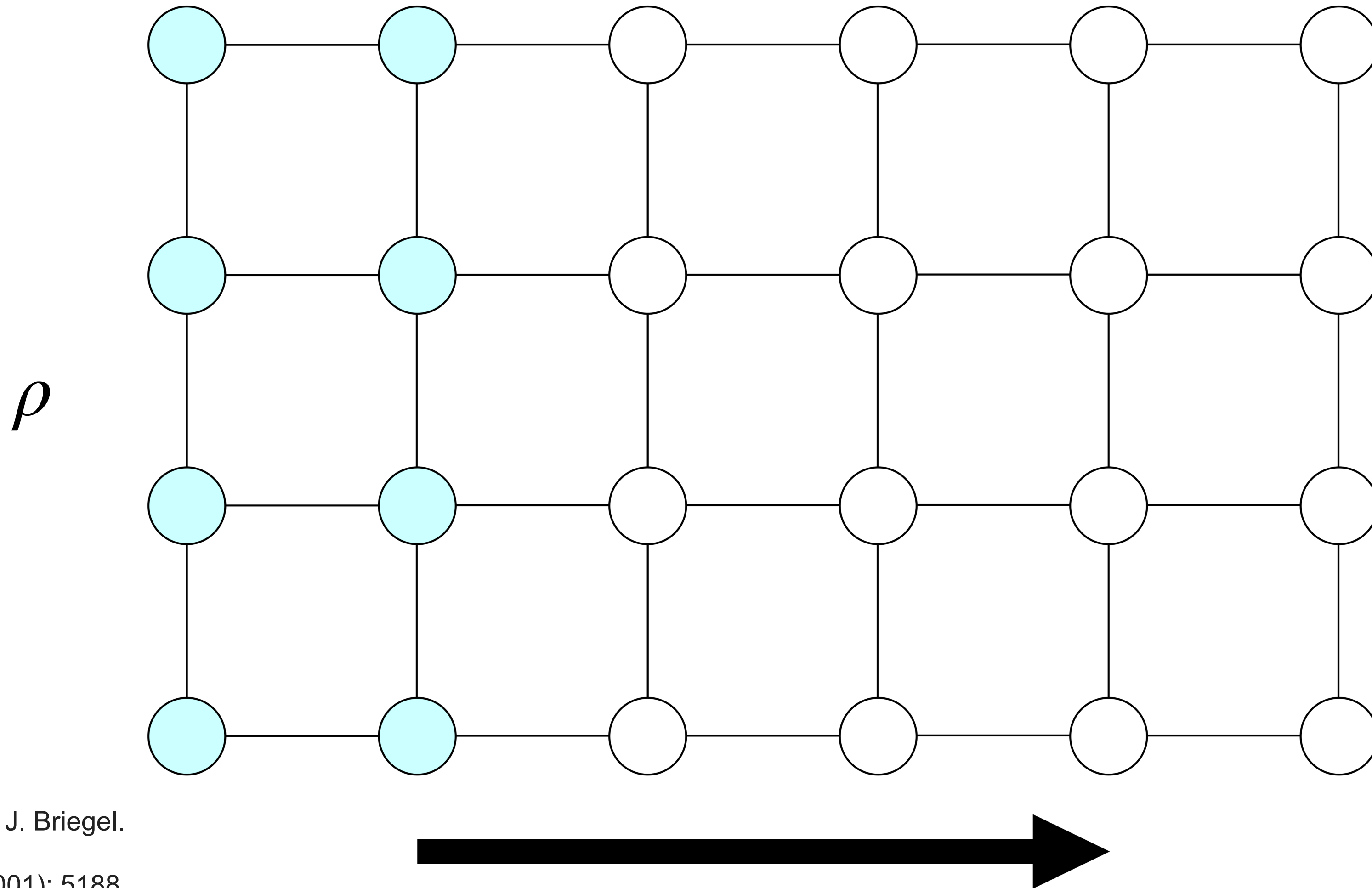
How does MBQC work?

Resource state + single-qubit measurements



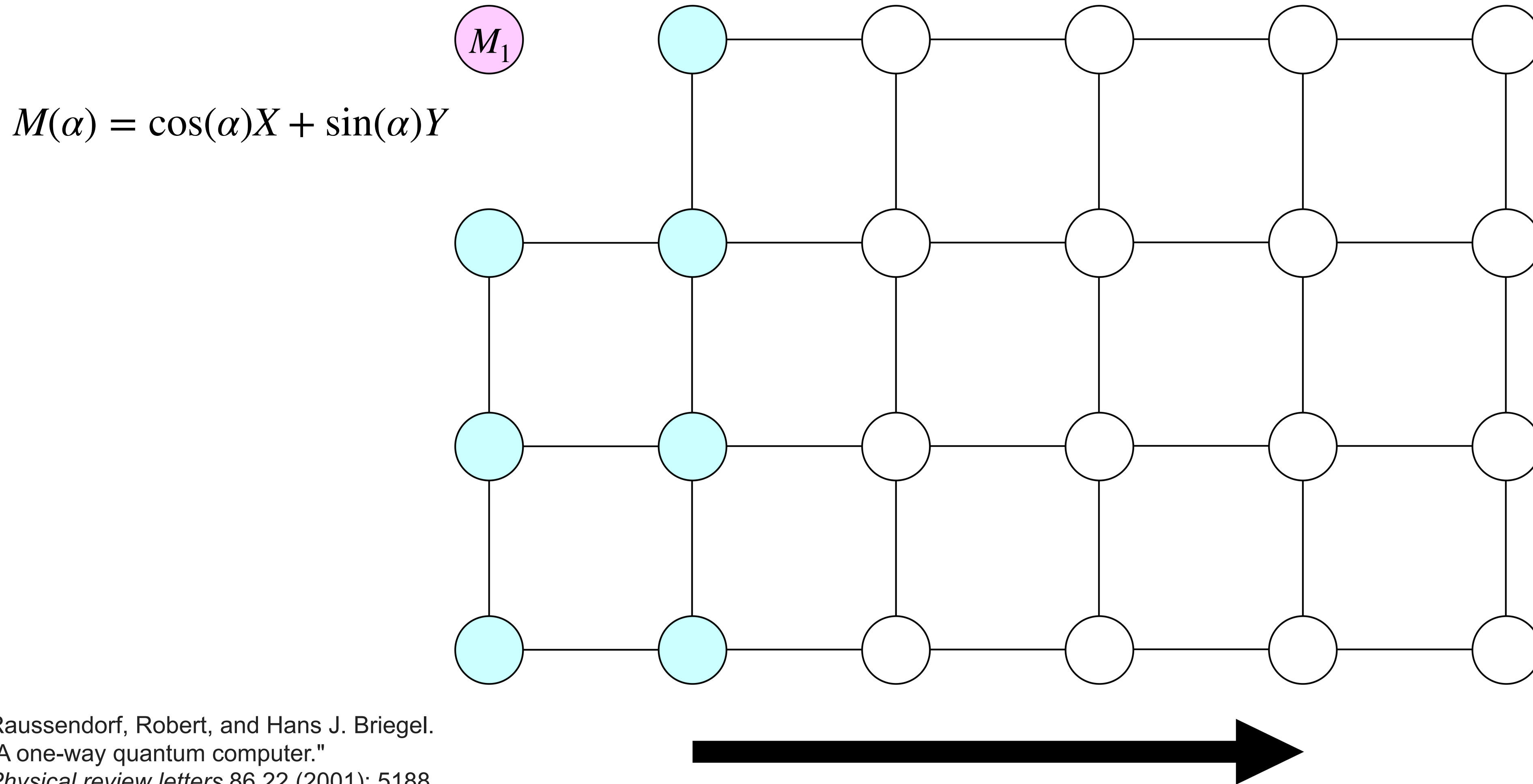
How does MBQC work?

Resource state + single-qubit measurements



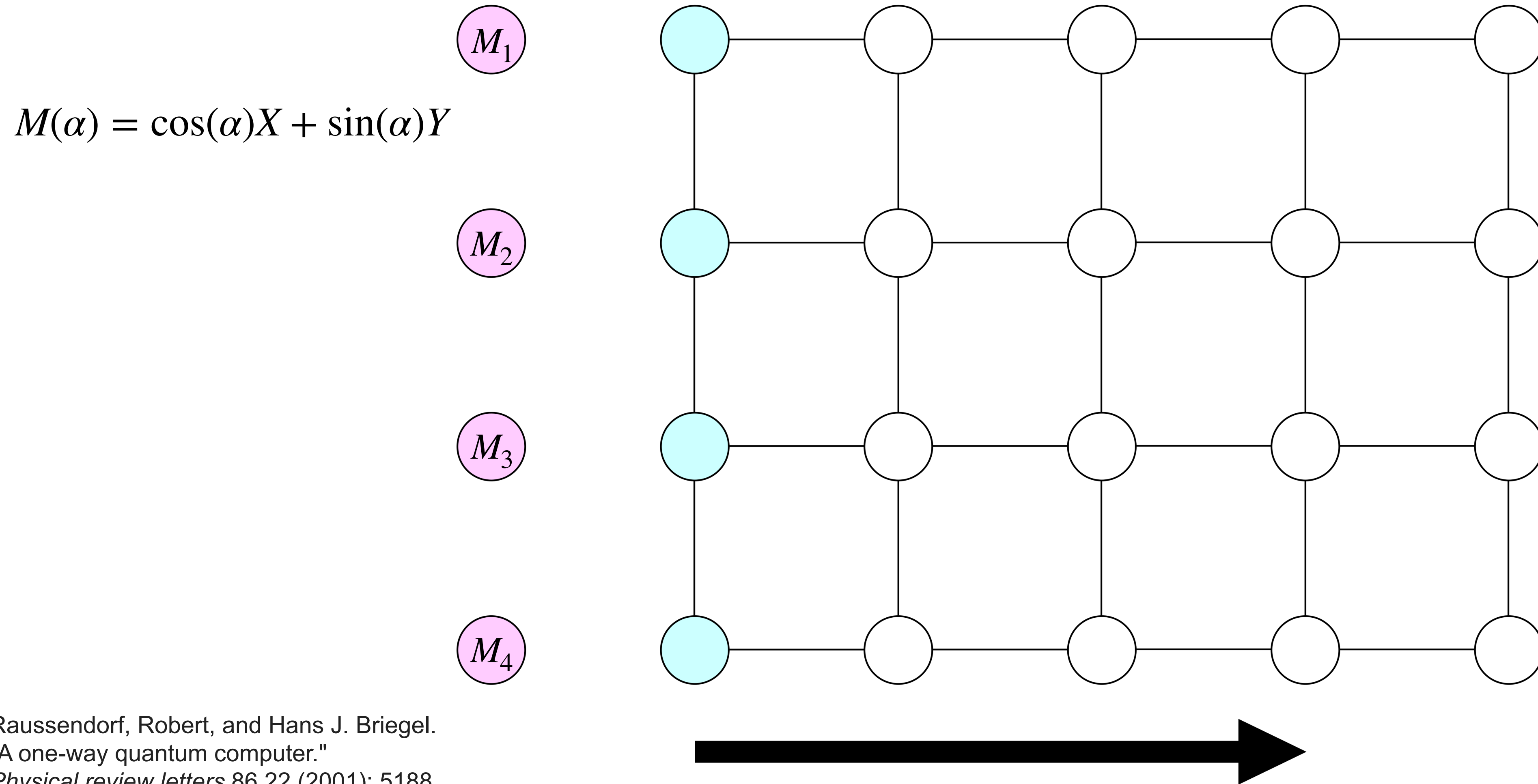
How does MBQC work?

Resource state + single-qubit measurements



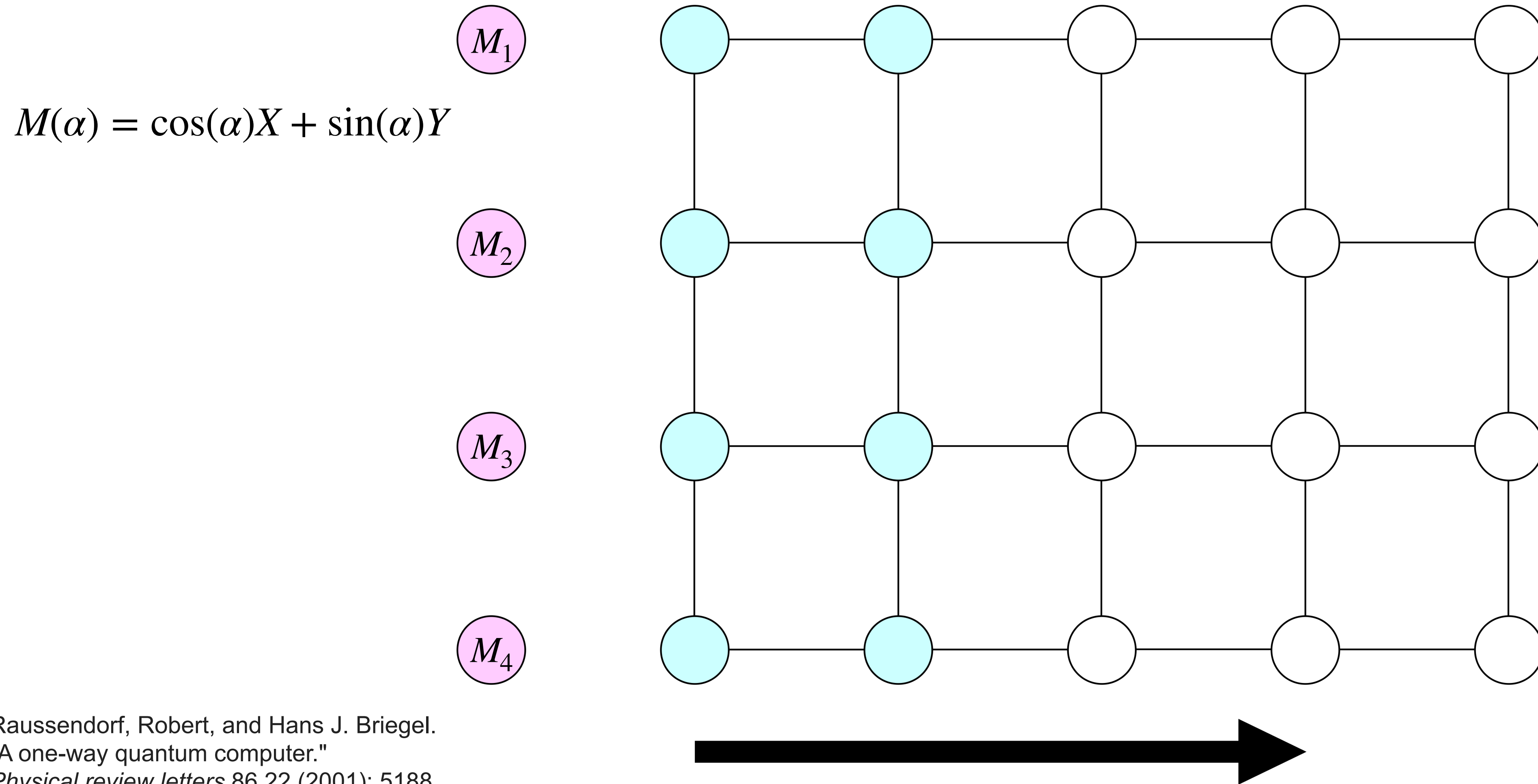
How does MBQC work?

Resource state + single-qubit measurements



How does MBQC work?

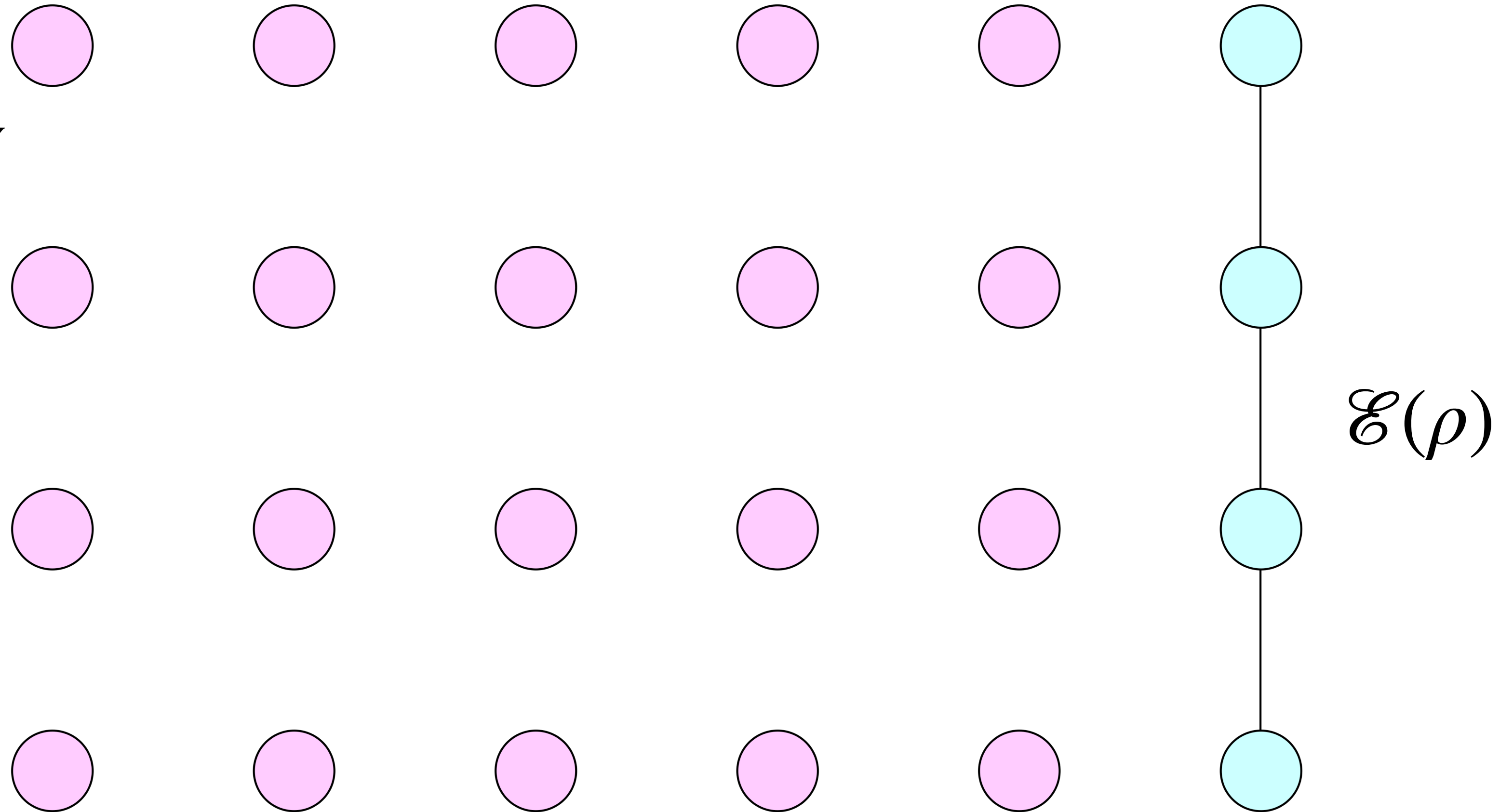
Resource state + single-qubit measurements



How does MBQC work?

Resource state + single-qubit measurements

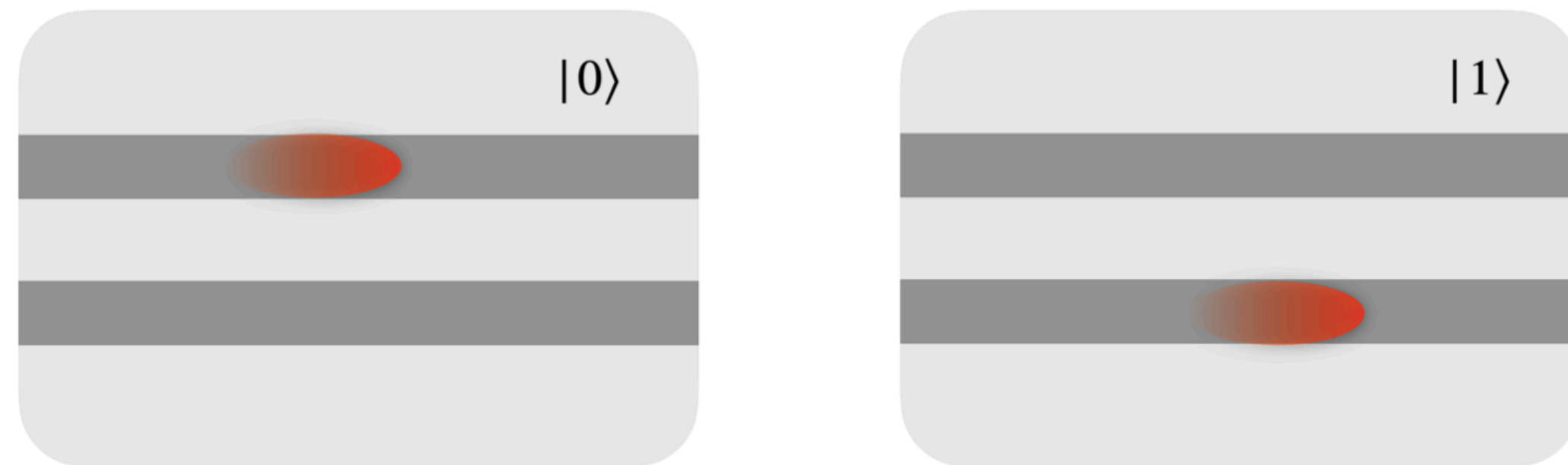
$$M(\alpha) = \cos(\alpha)X + \sin(\alpha)Y$$



Qubit encodings with photons

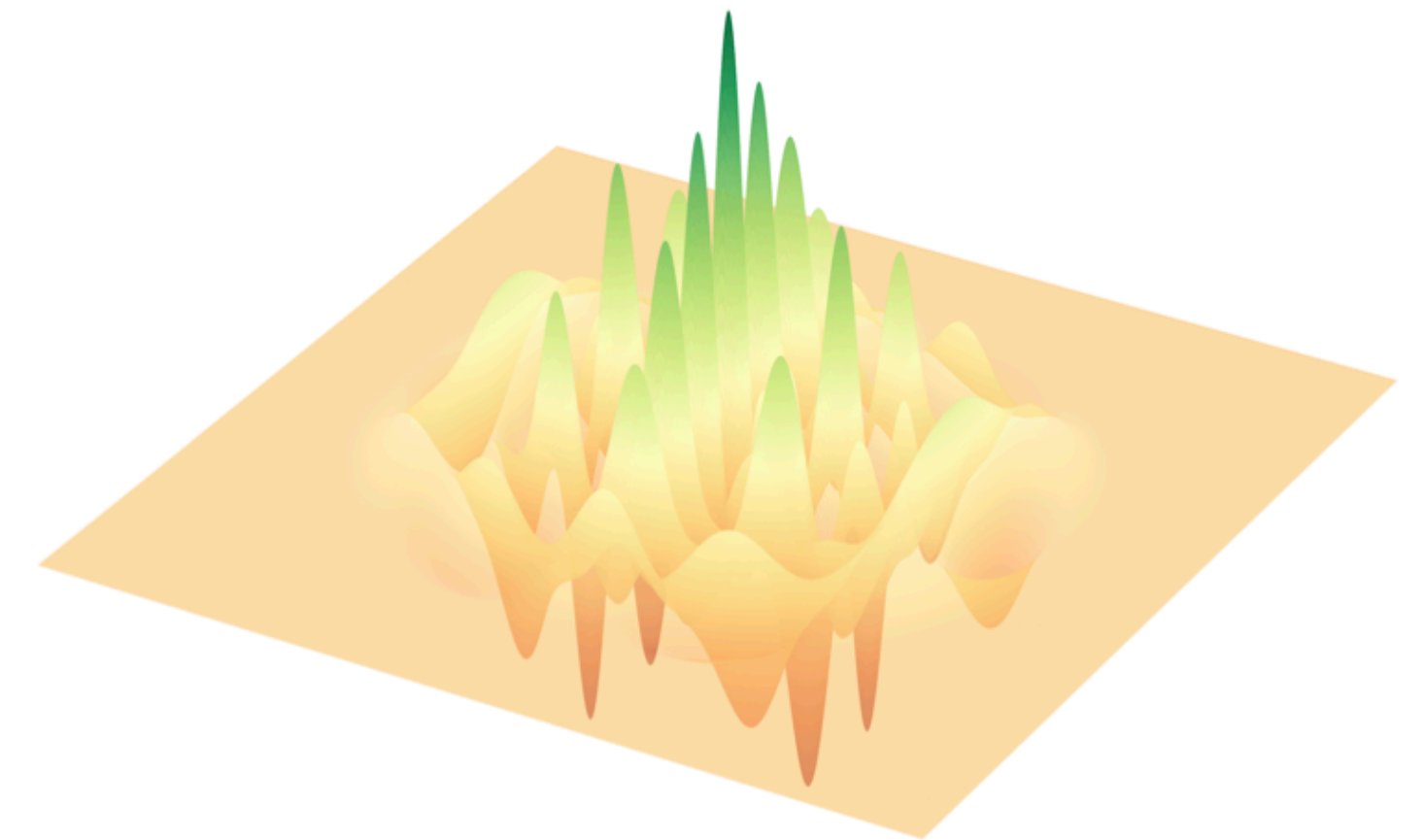
(Very simplified)

Dual-rail encoding



- Can prepare states easily
- Probabilistic entanglement via KLM protocol

GKP encoding



- Non-linearity needed to prepare state
- Deterministic entanglement via Beam splitters

MBQC summary

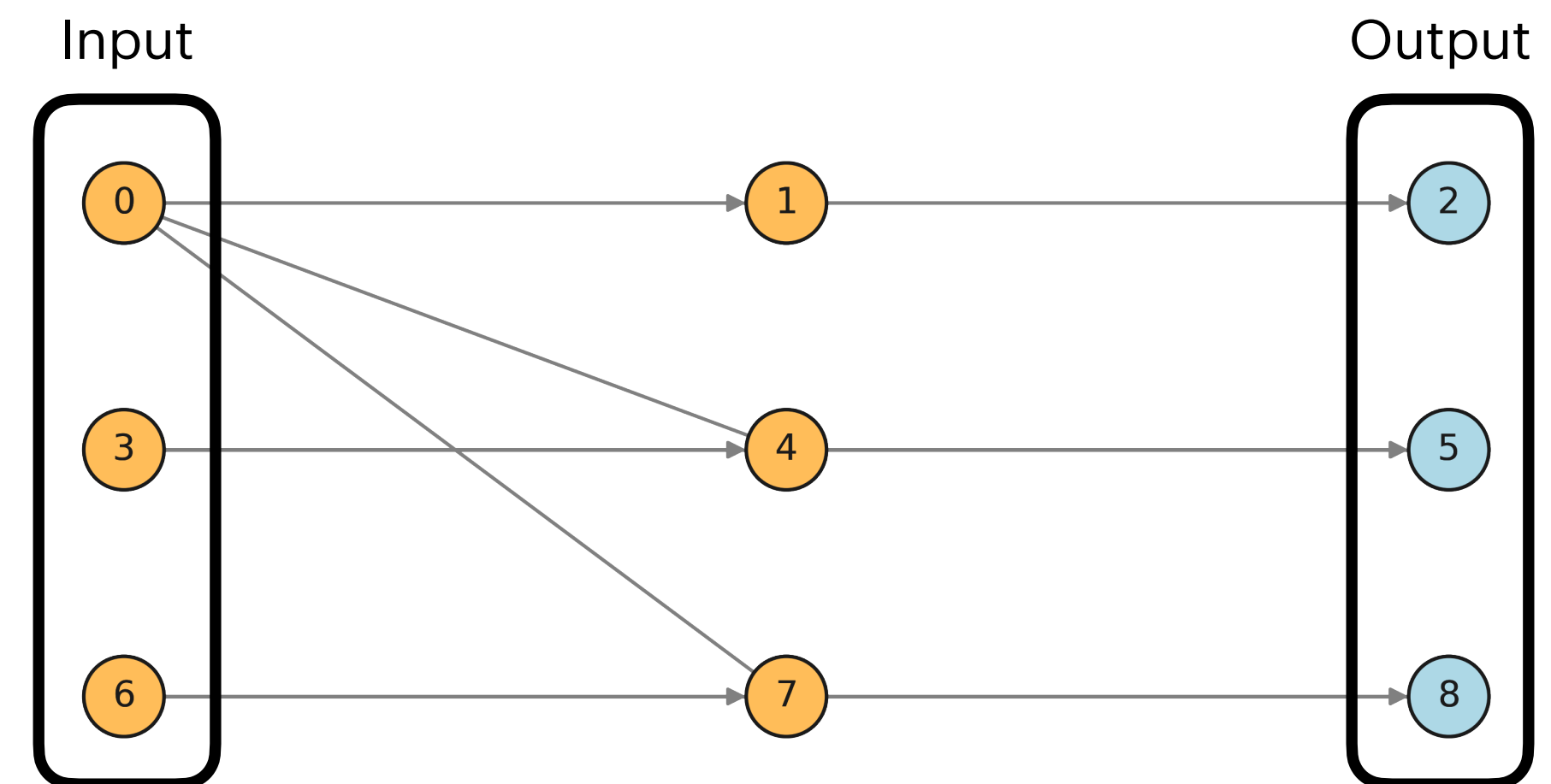
Not all graph states are useful!

An MBQC circuit (G, I, O) has **flow** if $\exists f: O^c \rightarrow I^c$ and a partial order $<$ over $G = (V, E)$ such that $\forall i \in V$:

1. $(i, f(i)) \in E$

2. $i < f(i)$

3. $\forall j \in N_G(f(i)) \setminus \{i\}, \quad i < j$

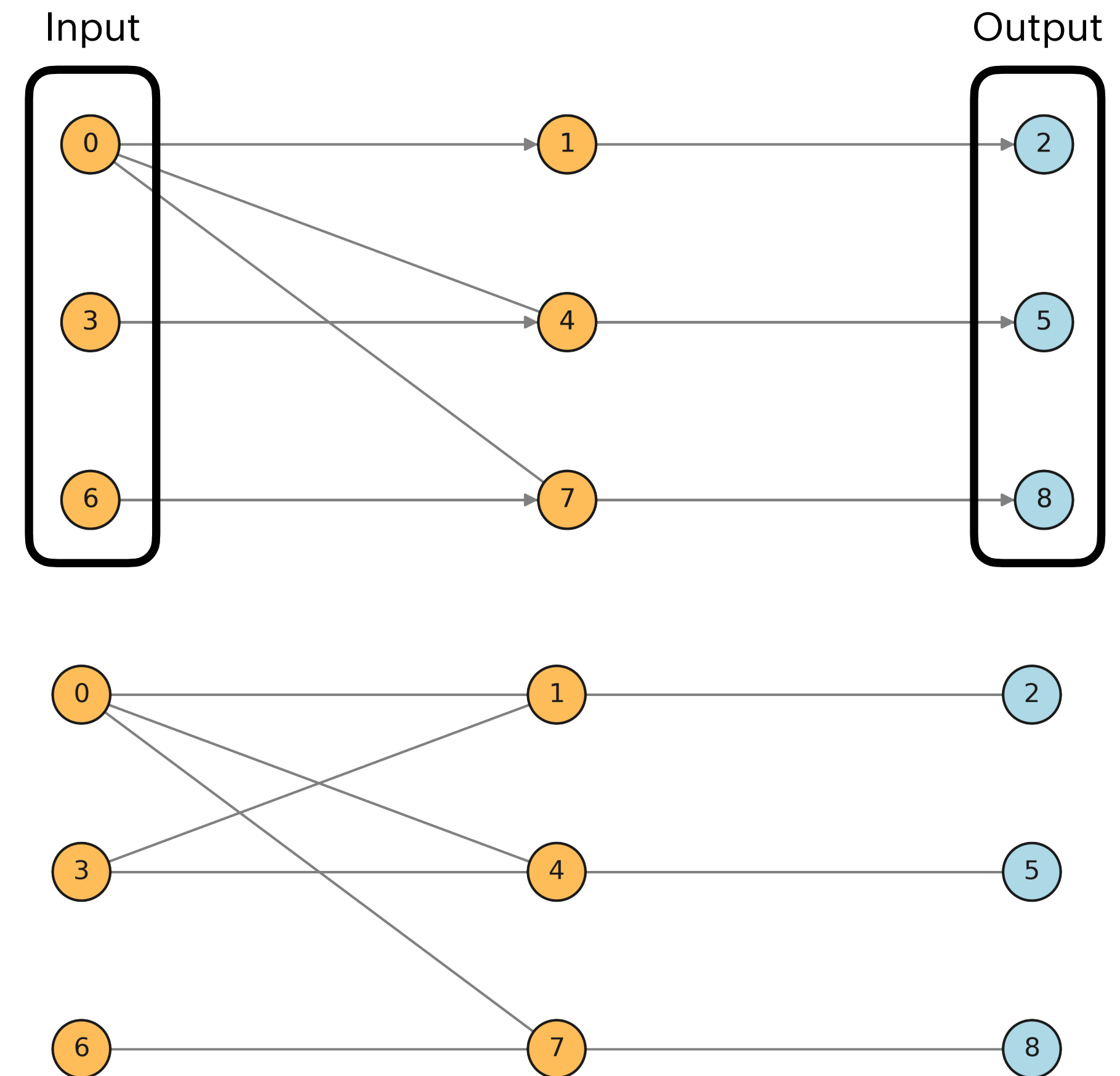


MBQC summary

Not all graph states are useful!

An MBQC circuit (G, I, O) has **flow** if $\exists f: O^c \rightarrow I^c$ and a partial order $<$ over $G = (V, E)$ such that $\forall i \in V$:

1. $(i, f(i)) \in E$
2. $i < f(i)$
3. $\forall j \in N_G(f(i)) \setminus \{i\}, \quad i < j$

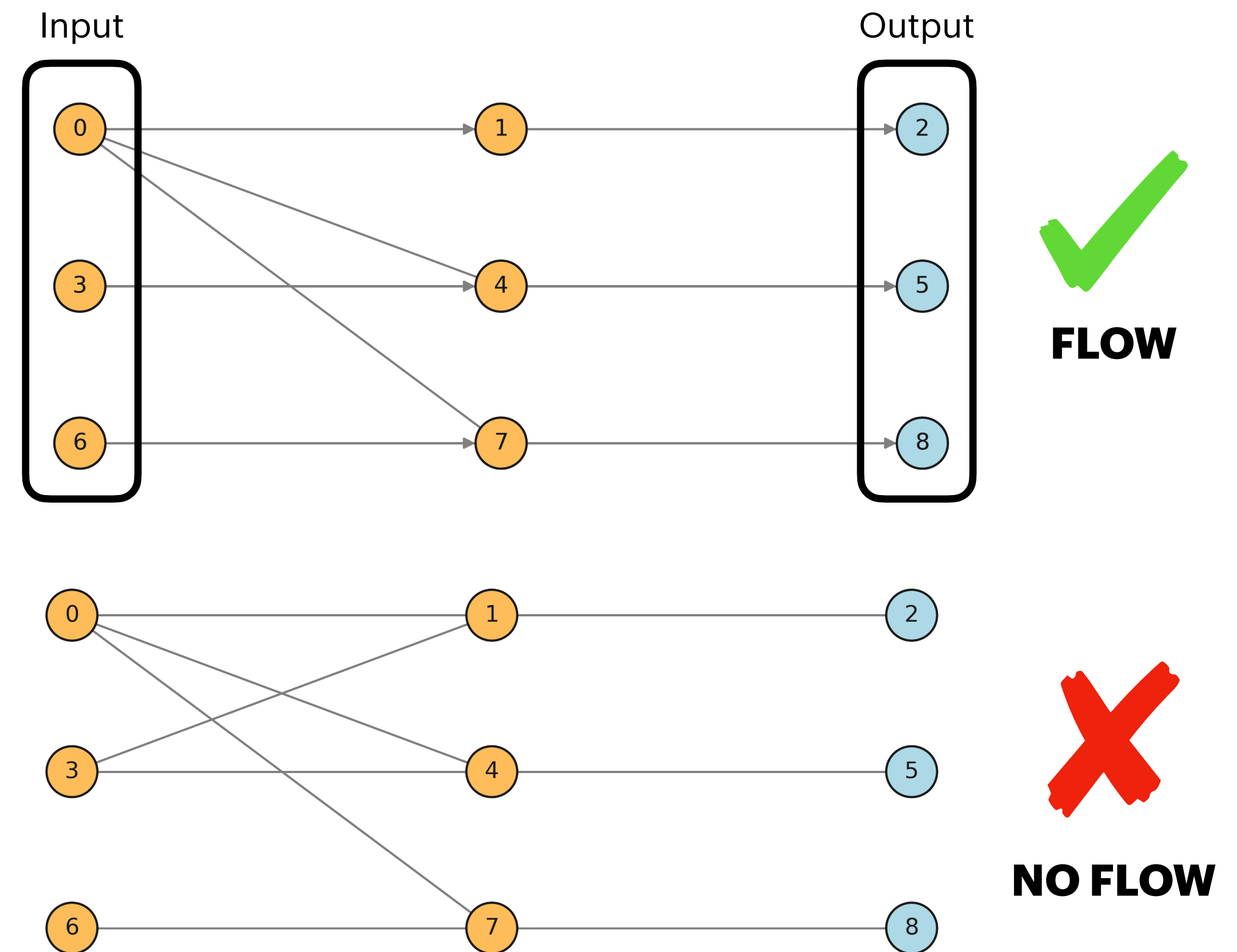


MBQC summary

Not all graph states are useful!

An MBQC circuit (G, I, O) has **flow** if $\exists f: O^c \rightarrow I^c$ and a partial order $<$ over $G = (V, E)$ such that $\forall i \in V$:

1. $(i, f(i)) \in E$
2. $i < f(i)$
3. $\forall j \in N_G(f(i)) \setminus \{i\}, \quad i < j$



Measurement-based quantum machine learning



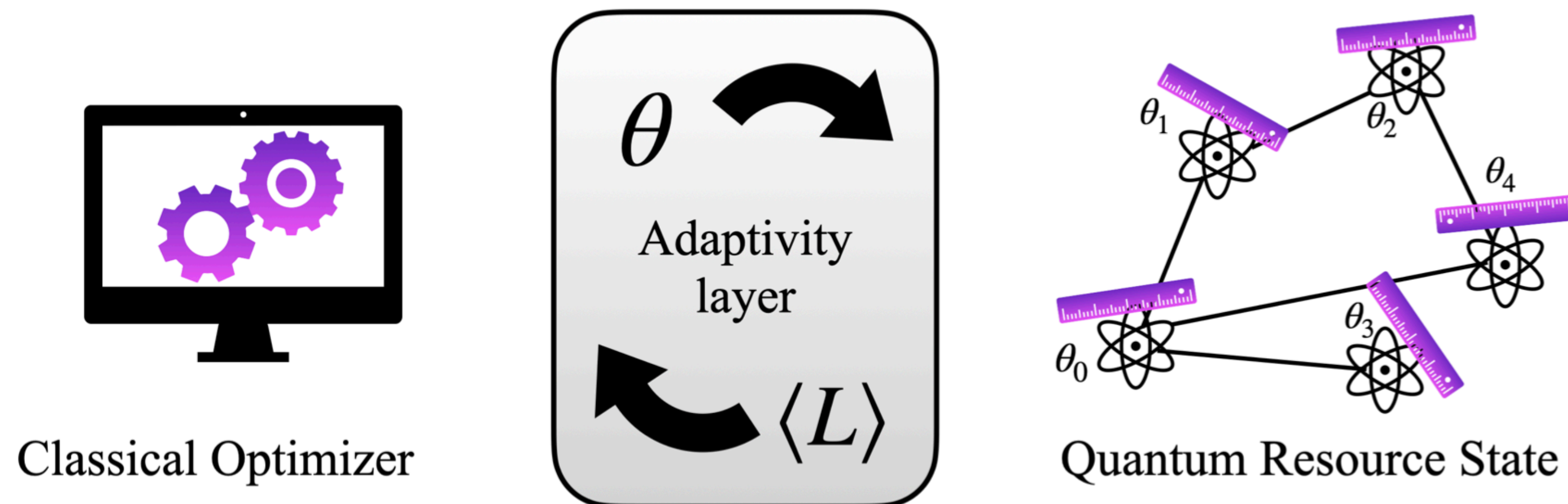
Mantilla Calderón, Luis, et al. "Measurement-based quantum machine learning." *arXiv preprint arXiv:2405.08319* (2024).

What is MB-QML?

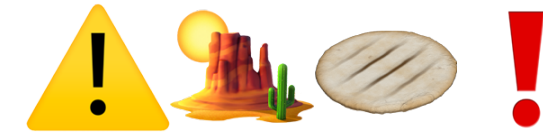
Learn measurement angles

A parametrized MBQC circuit is a triplet (G, I, O) with flow $(f, <)$ together with a parametrized measurement pattern $\mathcal{P}_\theta$.

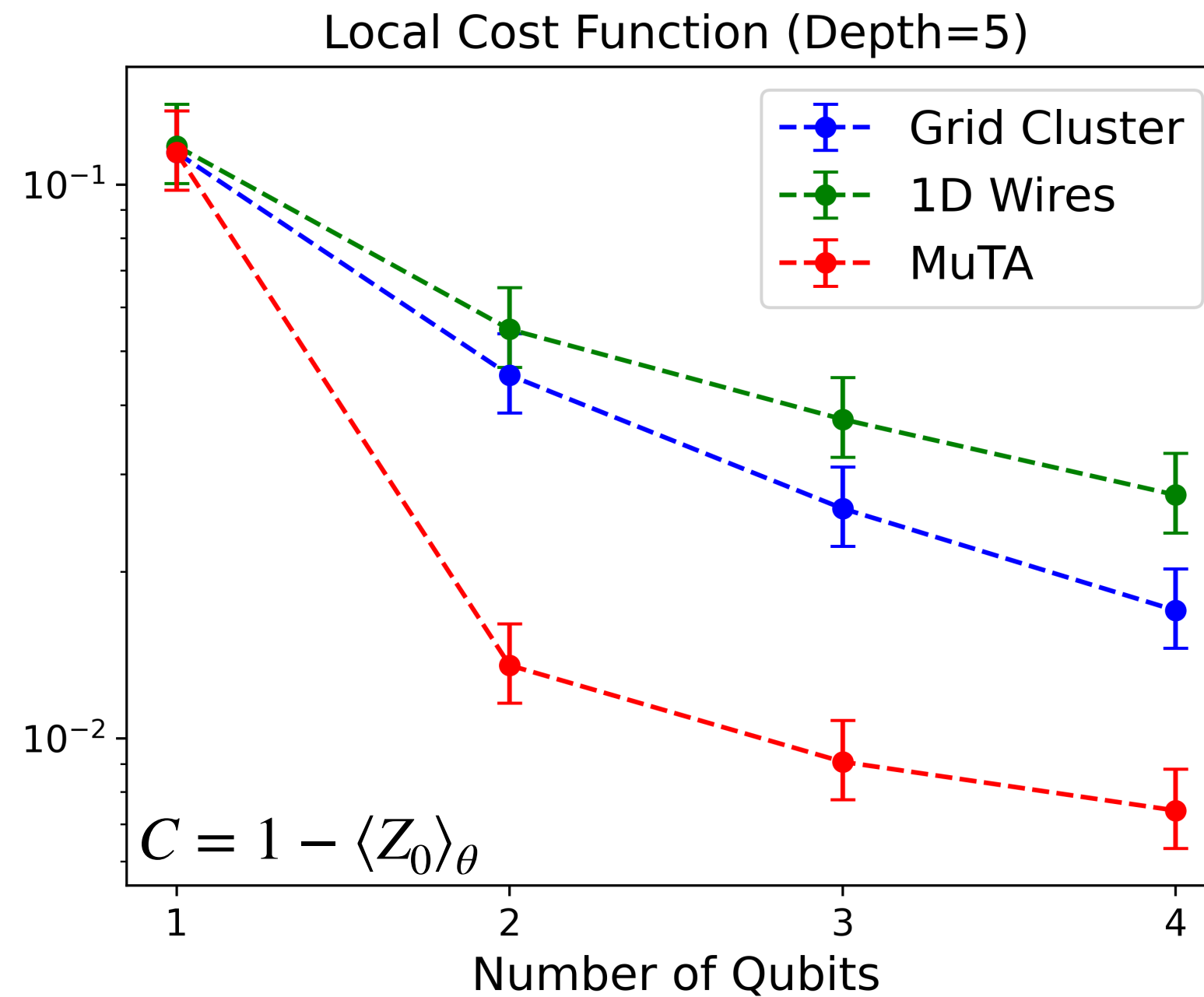
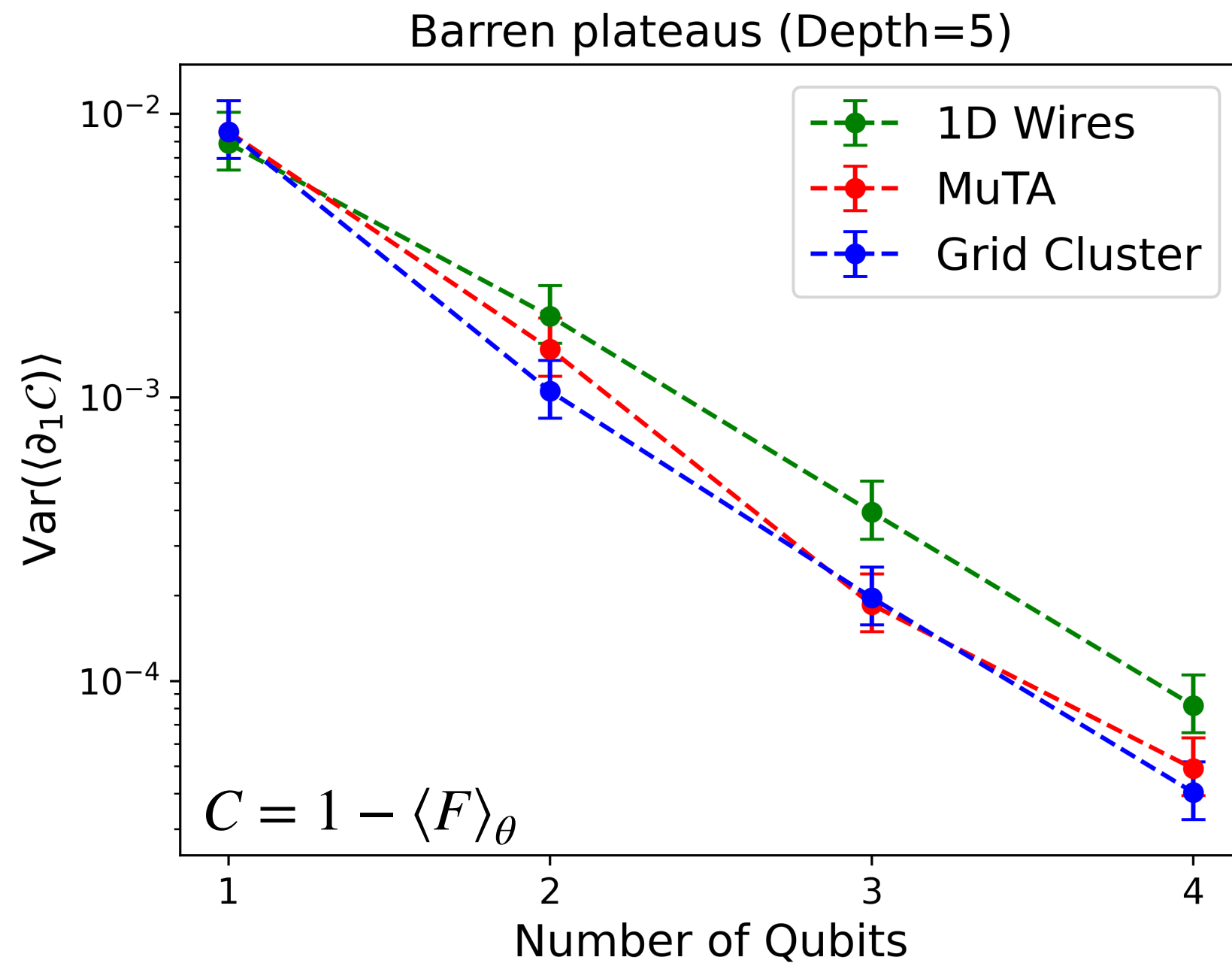
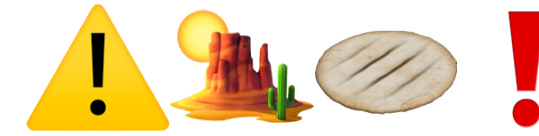
$$U_\theta |\psi\rangle_{in} = \left(\prod_{v \in O^c} \langle \theta |_v \right) |G\rangle_{in} = |\psi\rangle_{out}$$



Beware of barren plateaus



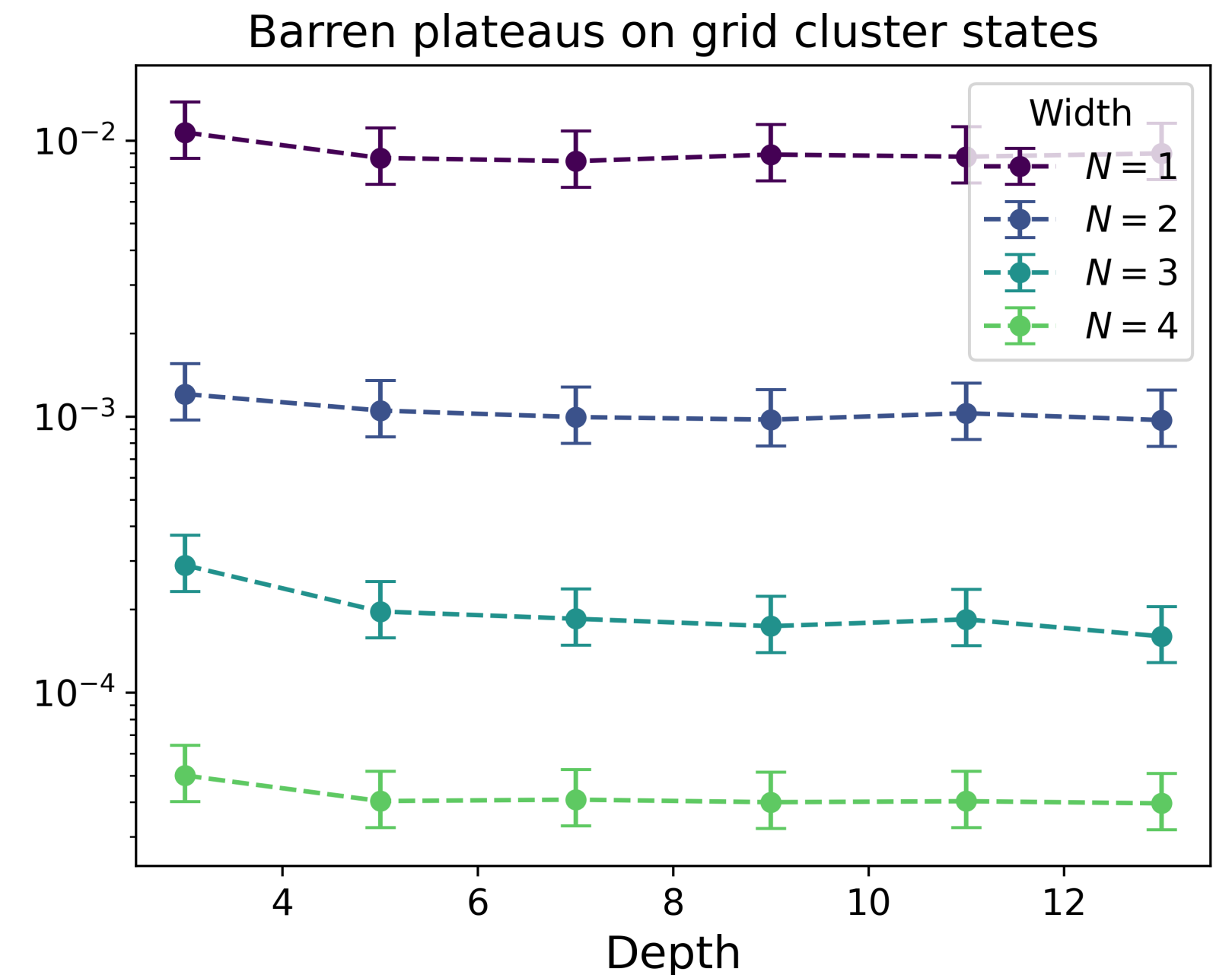
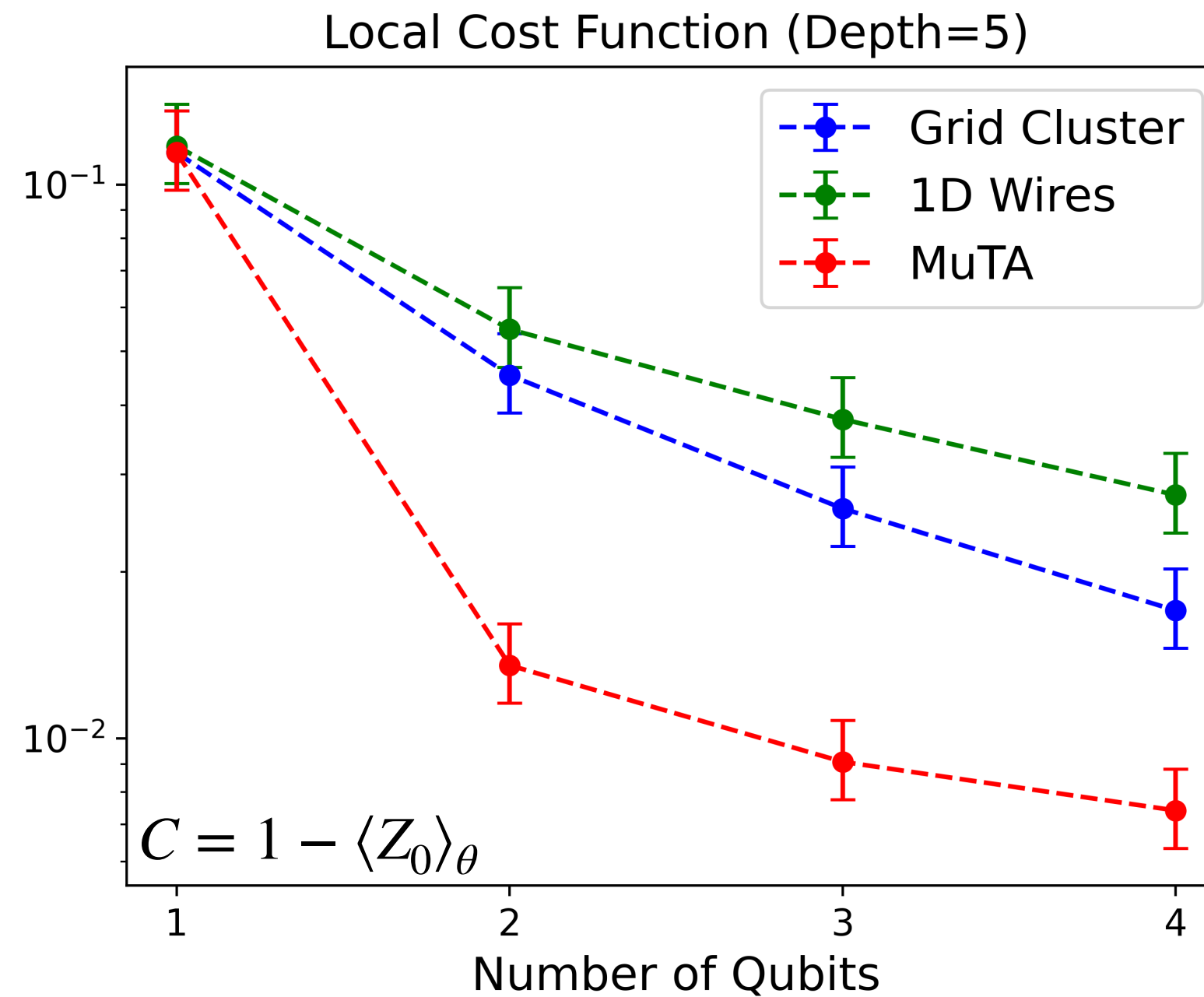
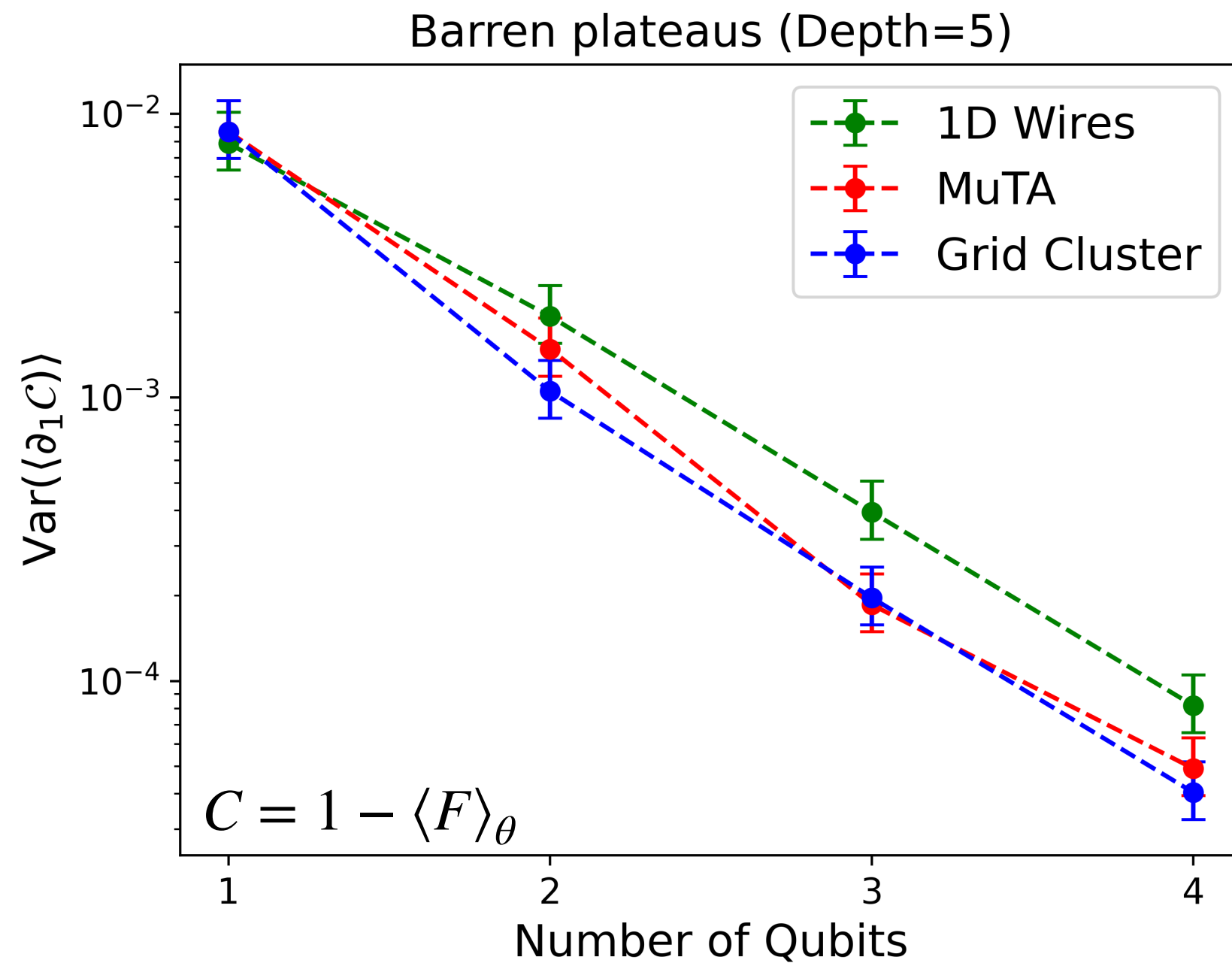
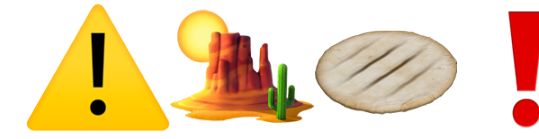
Beware of barren plateaus



Global cost functions, like the fidelity, cause barren plateaus

Cerezo, M., Sone, A., Volkoff, T., Cincio, L., & Coles, P. J. (2021).

Beware of barren plateaus



Global cost functions, like the fidelity, cause barren plateaus

Cerezo, M., Sone, A., Volkoff, T., Cincio, L., & Coles, P. J. (2021).

For low width circuits, depth is not a source of barren plateaus

How useful is a model?

Calculate the Lie algebra

$\dim(\mathfrak{g}) = 18$ $\mathfrak{g} = \mathfrak{su}(4) \oplus \mathfrak{su}(2)$	$\dim(\mathfrak{g}) = 63$ $\mathfrak{g} = \mathfrak{su}(8)$	$\dim(\mathfrak{g}) = 36$ $\mathfrak{g} = \mathfrak{sp}(8)$	$\dim(\mathfrak{g}) = 63$ $\mathfrak{g} = \mathfrak{su}(8)$

Expressivity

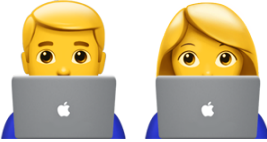
$$\mathbb{G} = \{e^{i\theta\mathfrak{g}} \mid \theta \in \mathbb{R}\}$$



Trainability

$$\text{Var}_{\theta} [\ell_{\theta}(\rho, O)] = \frac{1}{\dim(\mathfrak{g})} \text{Tr} [O_{\mathfrak{g}}^2] \text{Tr} [\rho_{\mathfrak{g}}^2]$$

Coding with MentPy

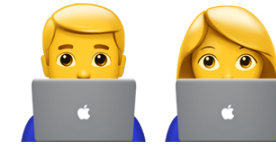


Find more at
docs.mentpy.com





Coding with MentPy



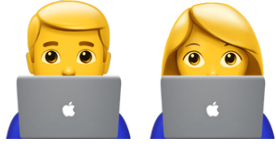
1. Define a resource state

```
import mentpy as mp

gs = mp.templates.muta(2,1, one_column=True)
gs[3] = mp.Ment('X')
gs[8] = mp.Ment('X')
```




Coding with MentPy

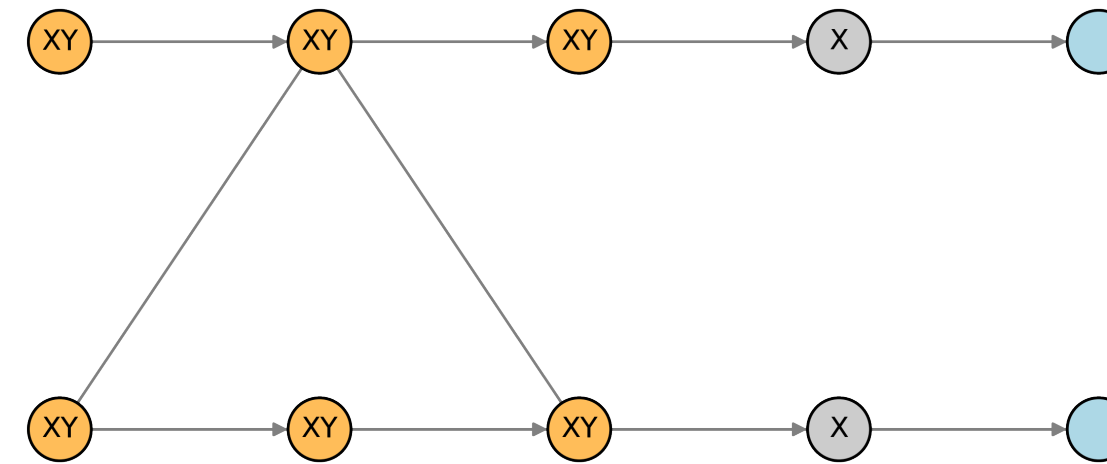


1. Define a resource state



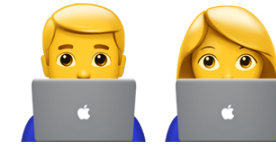
```
import mentpy as mp

gs = mp.templates.muta(2,1, one_column=True)
gs[3] = mp.Ment('X')
gs[8] = mp.Ment('X')
```





Coding with MentPy



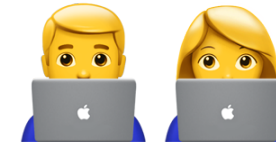
1. Define a resource state

```
import mentpy as mp

gs = mp.templates.muta(2,1, one_column=True)
gs[3] = mp.Ment('X')
gs[8] = mp.Ment('X')
```



Coding with MentPy



1. Define a resource state

```
import mentpy as mp

gs = mp.templates.muta(2,1, one_column=True)
gs[3] = mp.Ment('X')
gs[8] = mp.Ment('X')
```

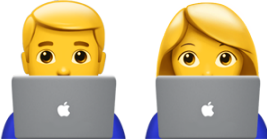
2. Pick a backend for simulations

```
import numpy as np

model = mp.PatternSimulator(gs, backend='numpy-dm')
params = np.random.rand(6)
output = model(params)
```




Coding with MentPy



1. Define a resource state

```
import mentpy as mp

gs = mp.templates.muta(2,1, one_column=True)
gs[3] = mp.Ment('X')
gs[8] = mp.Ment('X')
```

2. Pick a backend for simulations

```
import numpy as np

model = mp.PatternSimulator(gs, backend='numpy-dm')
params = np.random.rand(6)
output = model(params)
```

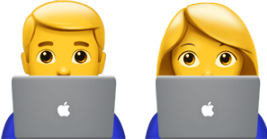
3. Construct a loss function

```
def loss(params, inputs, targets):
    outputs = []
    for psi in inputs:
        model.reset(input_state=psi)
        rho = model(params)
        outputs.append(rho)

    afidelity = 0
    for rho, phi in zip(outputs, targets):
        afidelity += mp.calculator.fidelity(rho, phi)
    return 1-(afidelity/len(targets))
```



Coding with MentPy



1. Define a resource state

```
import mentpy as mp

gs = mp.templates.muta(2,1, one_column=True)
gs[3] = mp.Ment('X')
gs[8] = mp.Ment('X')
```

2. Pick a backend for simulations

```
import numpy as np

model = mp.PatternSimulator(gs, backend='numpy-dm')
params = np.random.rand(6)
output = model(params)
```

3. Construct a loss function

```
def loss(params, inputs, targets):
    outputs = []
    for psi in inputs:
        model.reset(input_state=psi)
        rho = model(params)
        outputs.append(rho)

    afidelity = 0
    for rho, phi in zip(outputs, targets):
        afidelity += mp.calculator.fidelity(rho, phi)
    return 1-(afidelity/len(targets))
```

4. Use an optimizer

```
cost = lambda params: loss(params, X_train, y_train)
opt = mp.optimizers.AdamOptimizer(step_size=0.01)

params = np.random.rand(6)
for i in range(n_steps):
    params = opt.step(cost, params, i)

optimal_cost = cost(params)
```



Welcome to MentPy's documentation

Welcome to MentPy's documentation

- Features
- Roadmap
- Contributing
- Acknowledgements
- Citation



Note

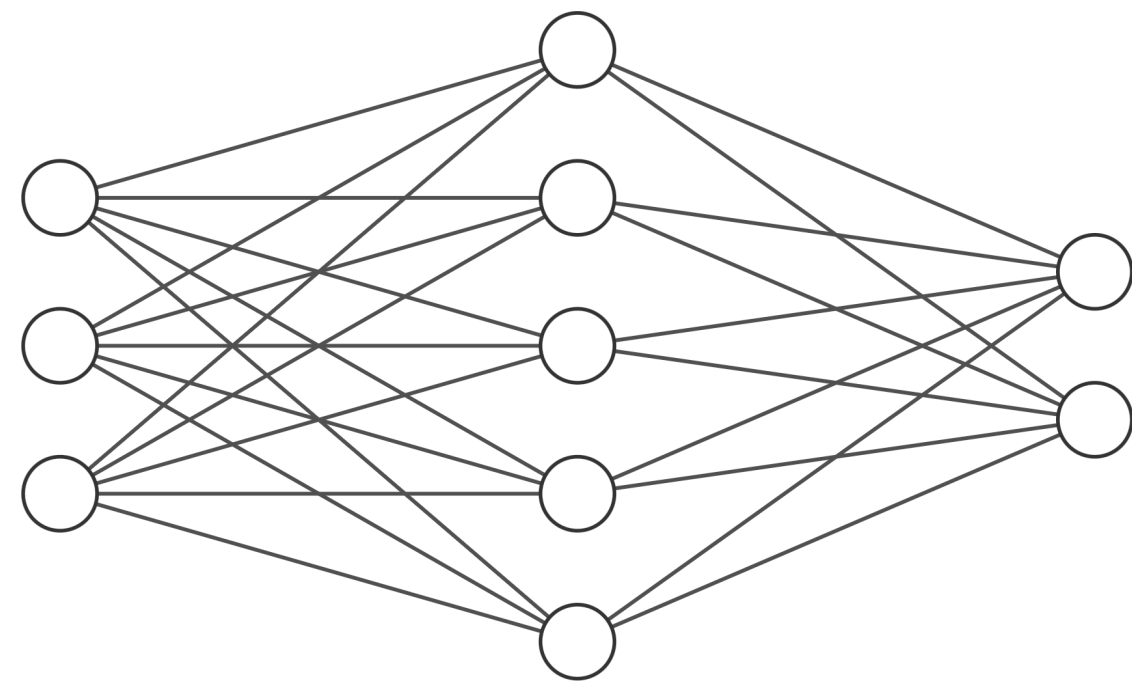
MentPy is currently in alpha and is under active development.

The `mentpy` library is an open-source Python package for creating and training quantum machine learning (QML) models in the measurement-based quantum computing (MBQC) framework. This library contains functions to automatically calculate the causal flow or generalized flow of a graph and tools to analyze the expressivity of the MBQC ansatzes.

Solving problems with MB-QML

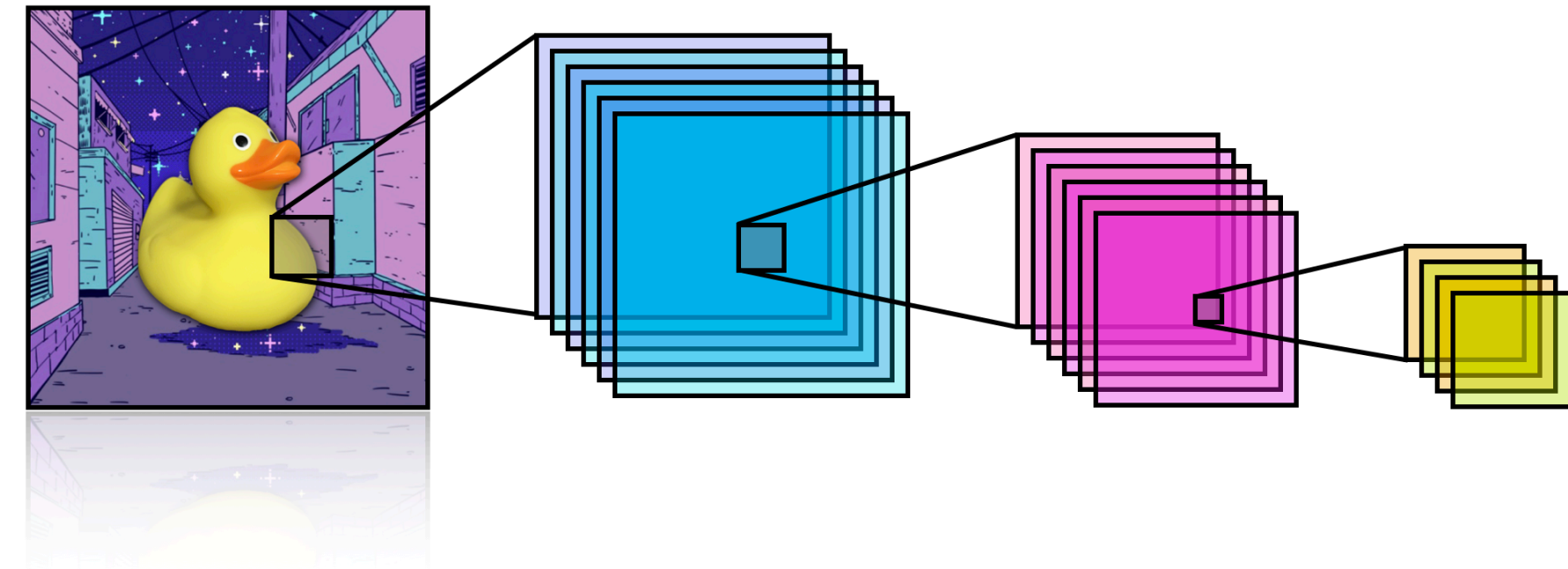
Two approaches

Universality

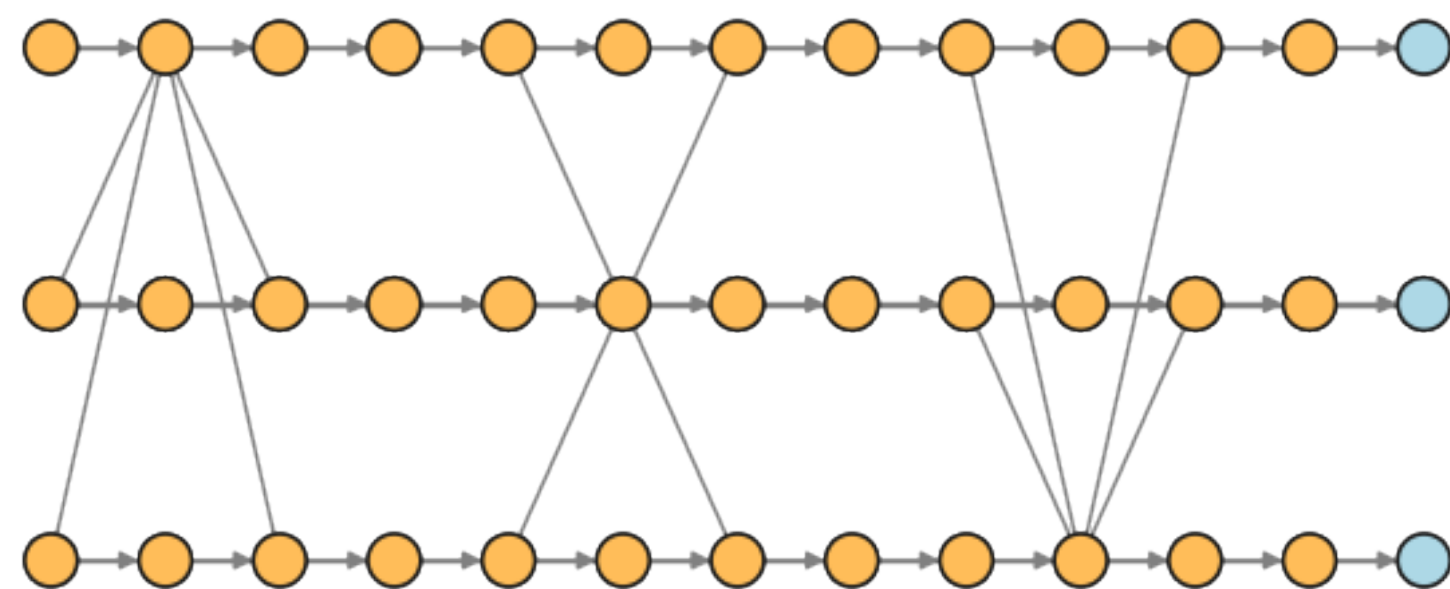


Classical

Custom Architectures

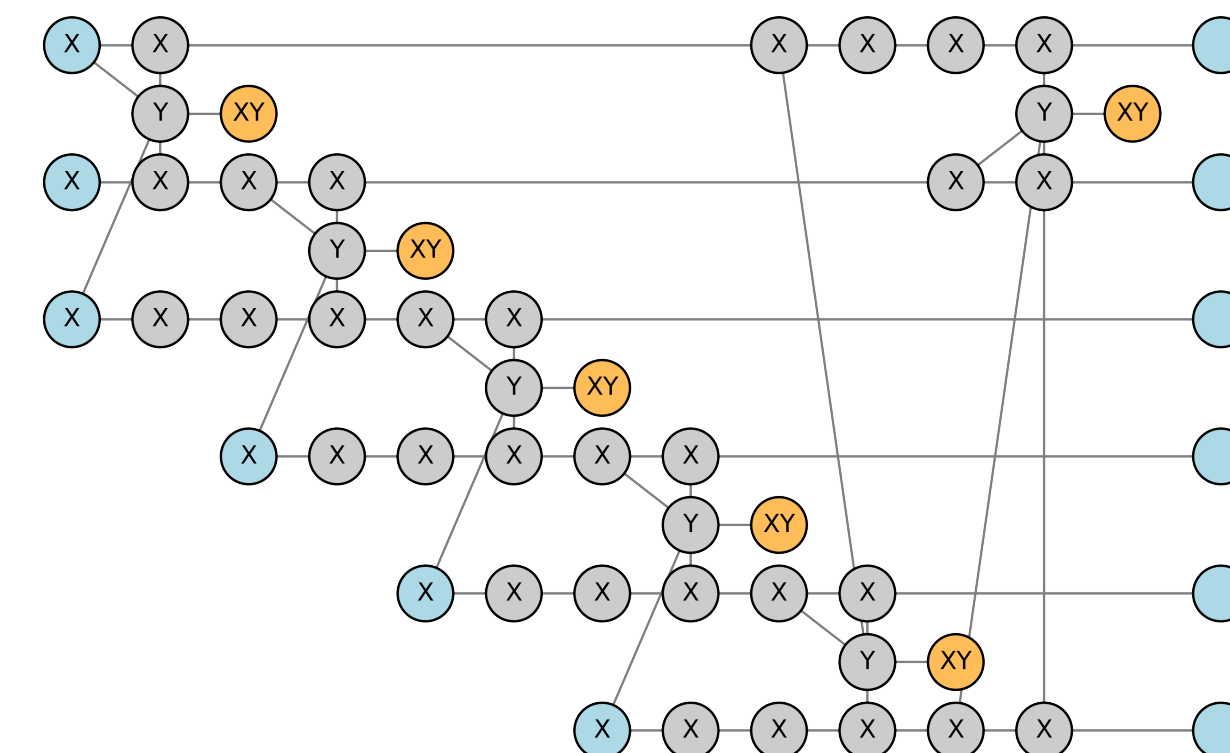


The 'to-go' tool



Quantum

Translation invariant

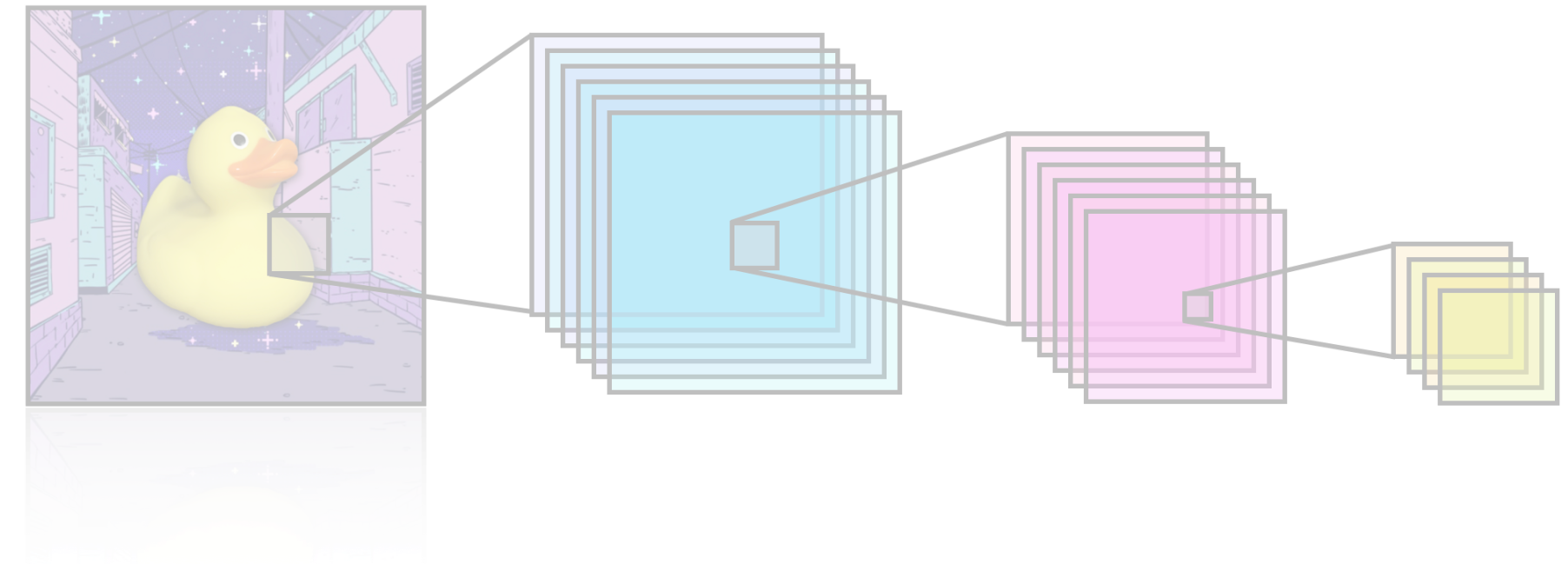
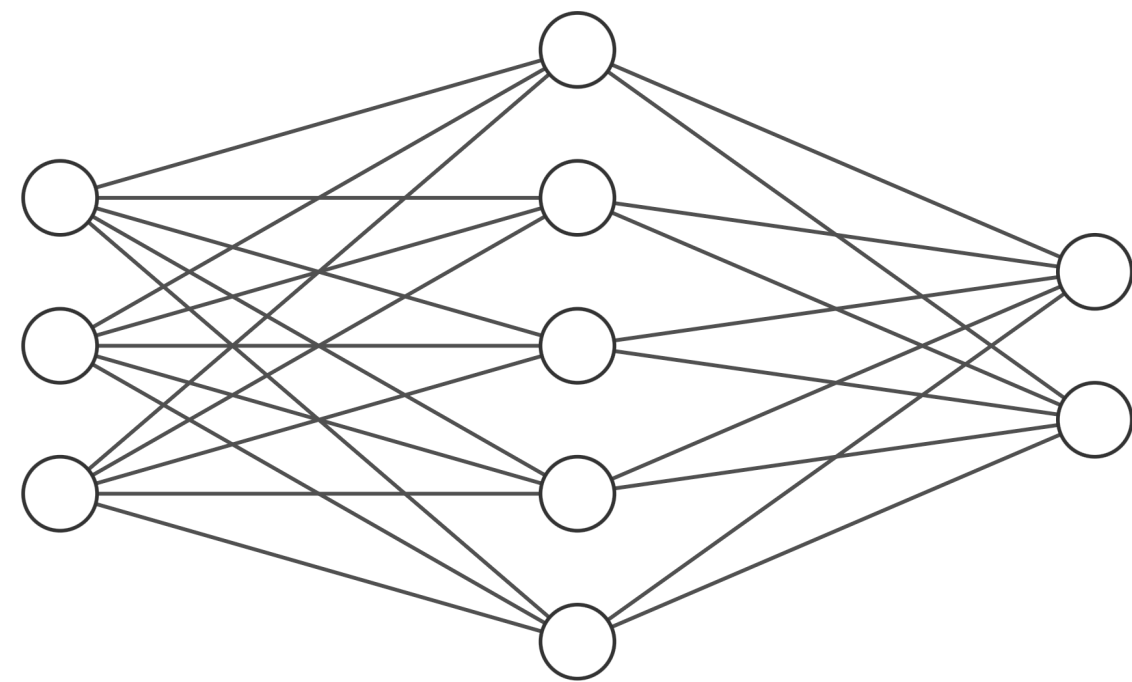


Solving problems with MB-QNNs

Two main messages

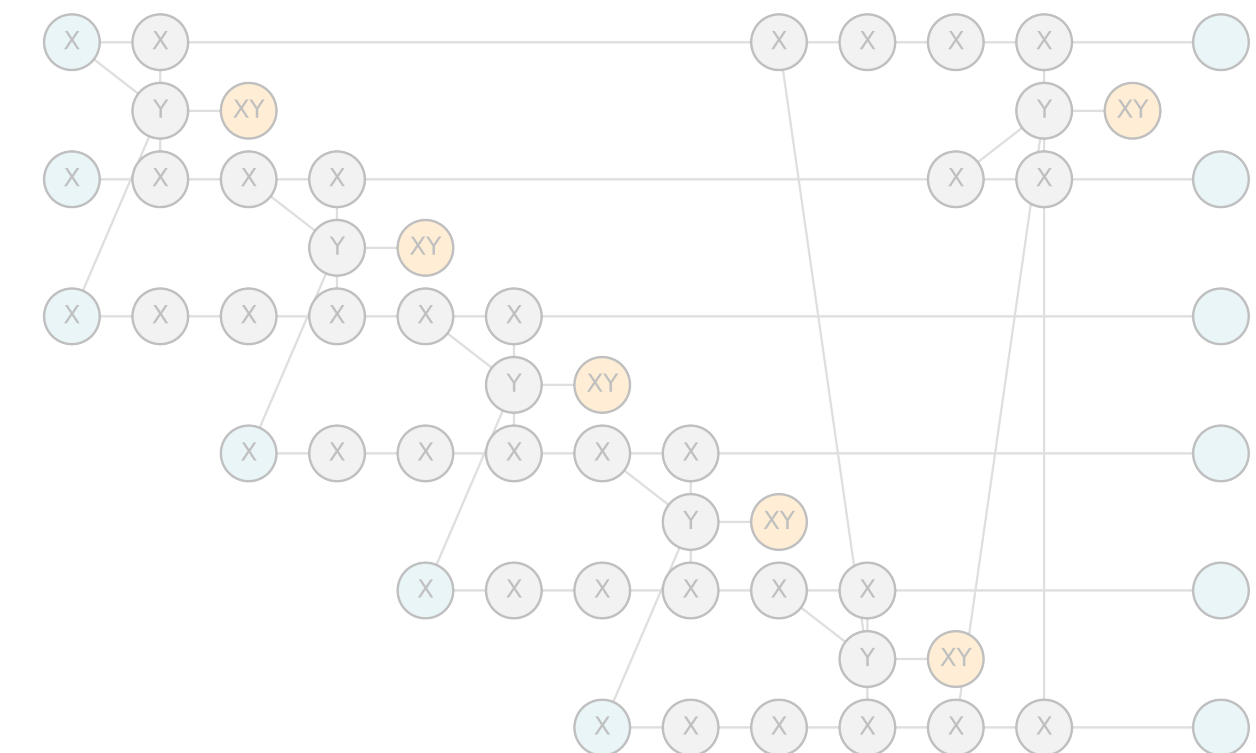
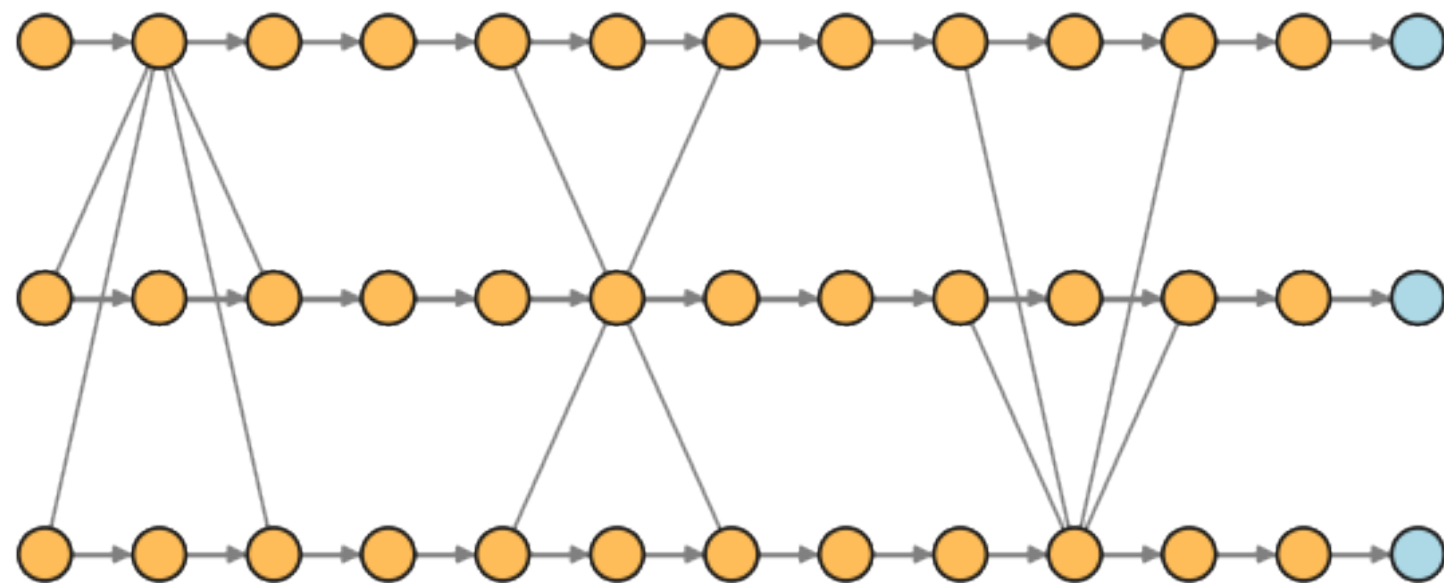
Universality

Classical



The 'to-go' tool

Quantum

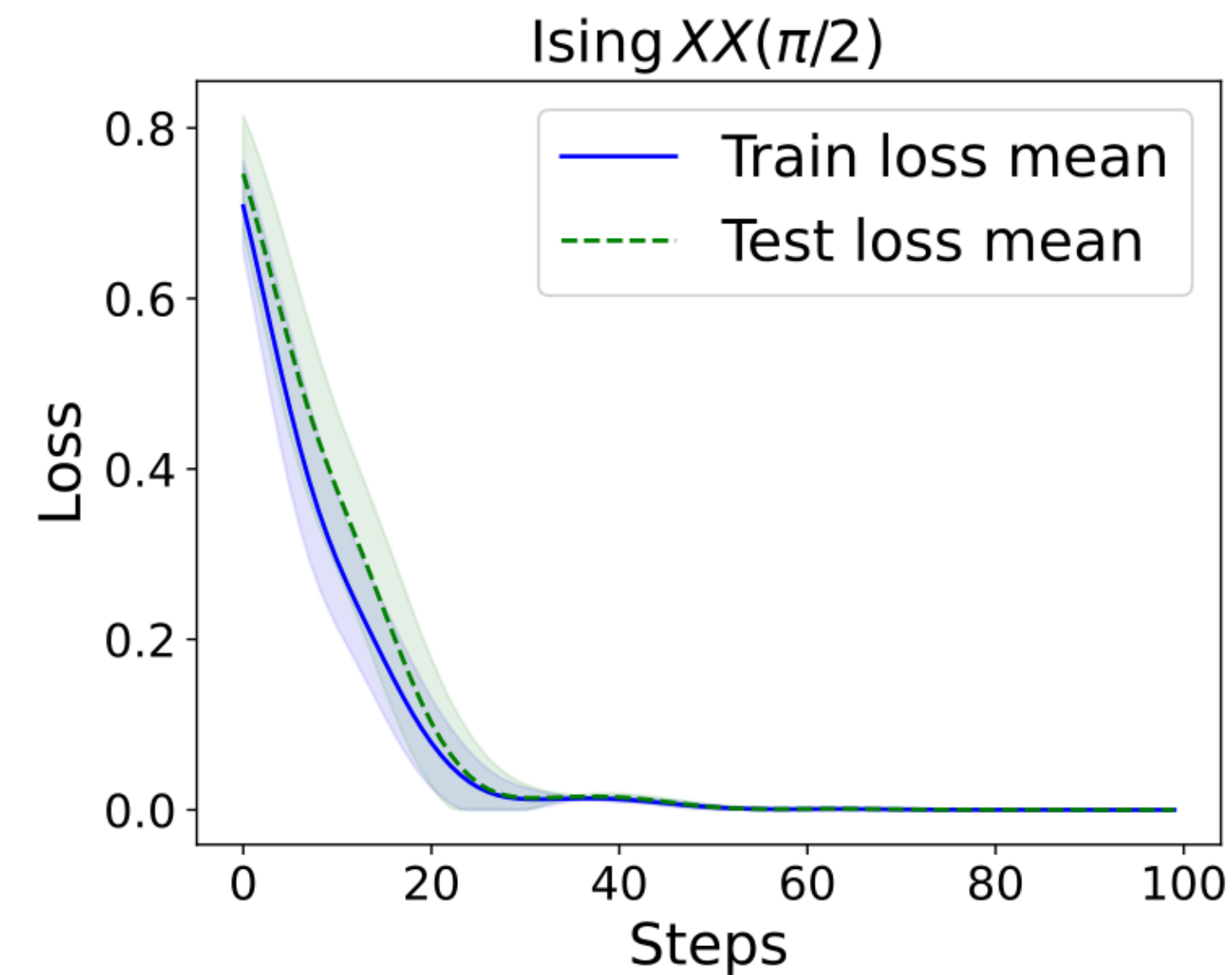
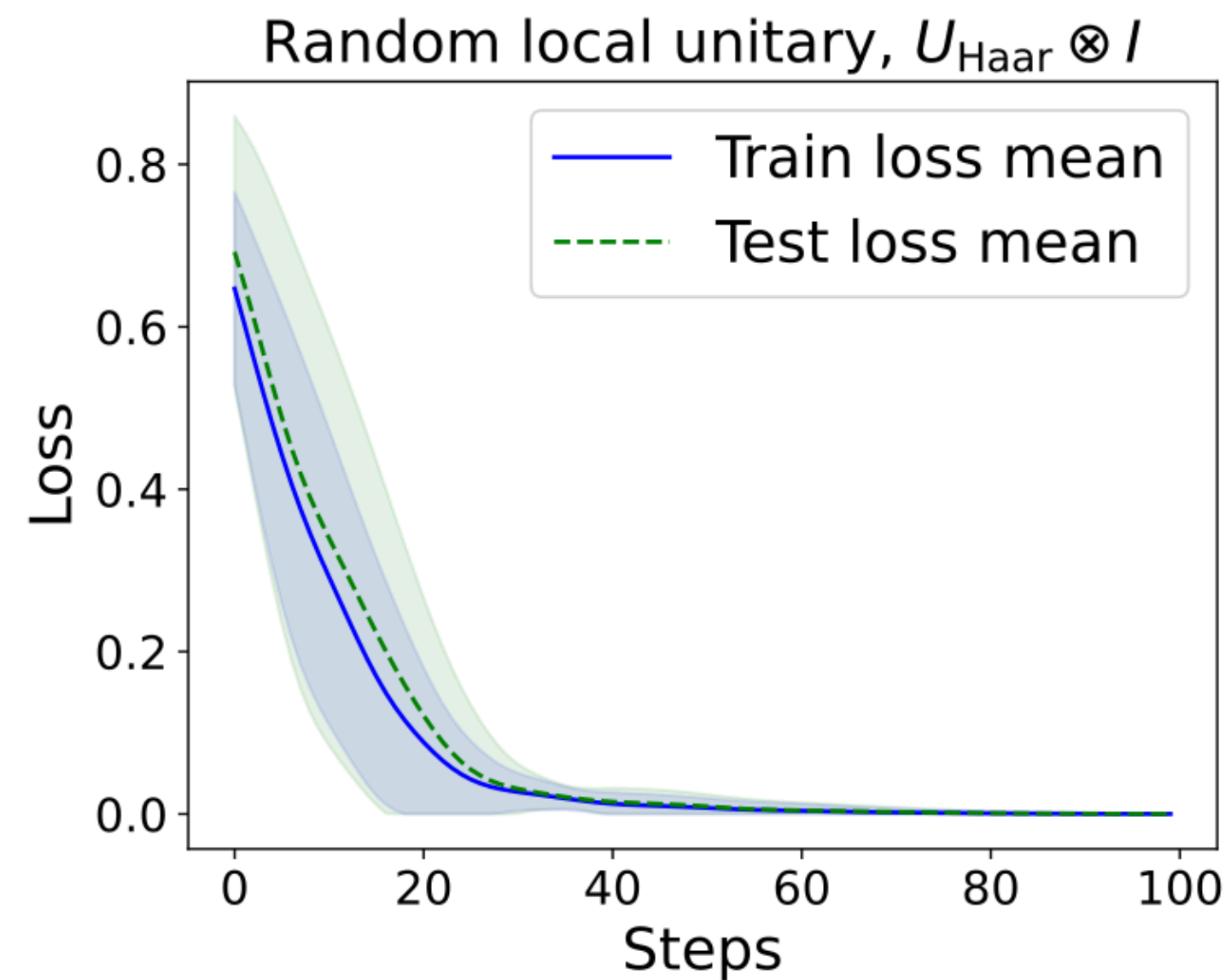
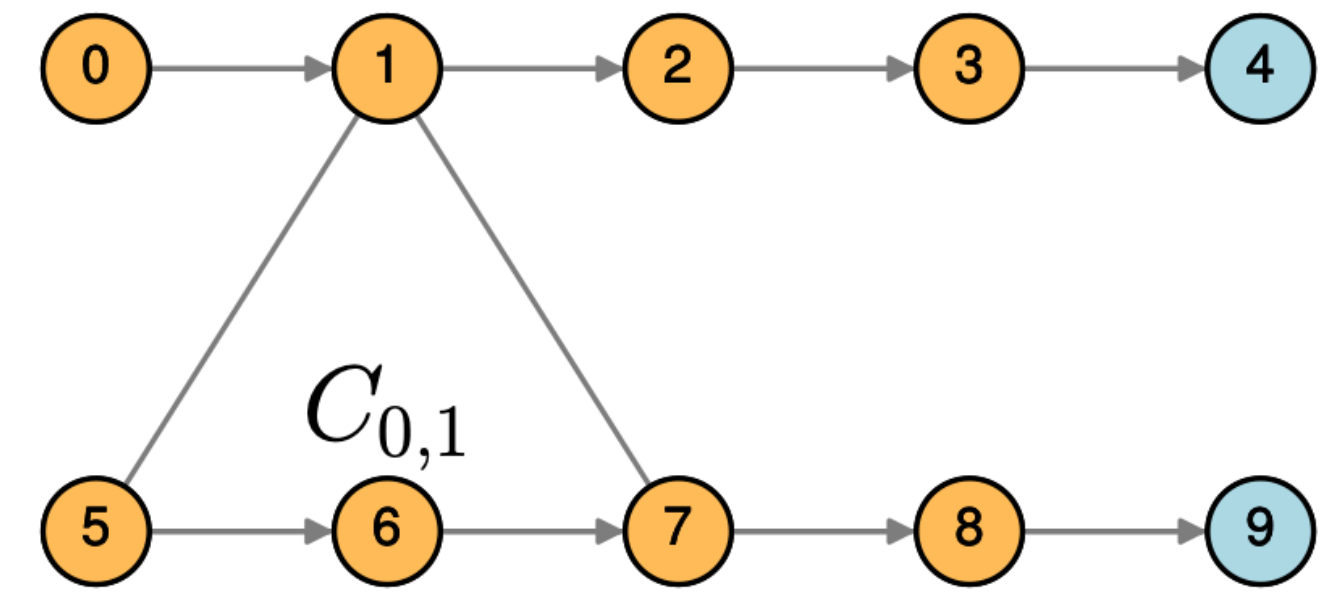


Universal set of gates

Minimum requirement

We train MuTA with $\left\{ (\rho_i, \sigma_i) \right\}_{i=1}^N$ where $\sigma_i = U\rho_i U^\dagger$

and the cost function is $C(\theta) = 1 - \frac{1}{N} \sum_{i=1}^N F(\mathcal{E}_\theta(\rho_i), \sigma_i)$

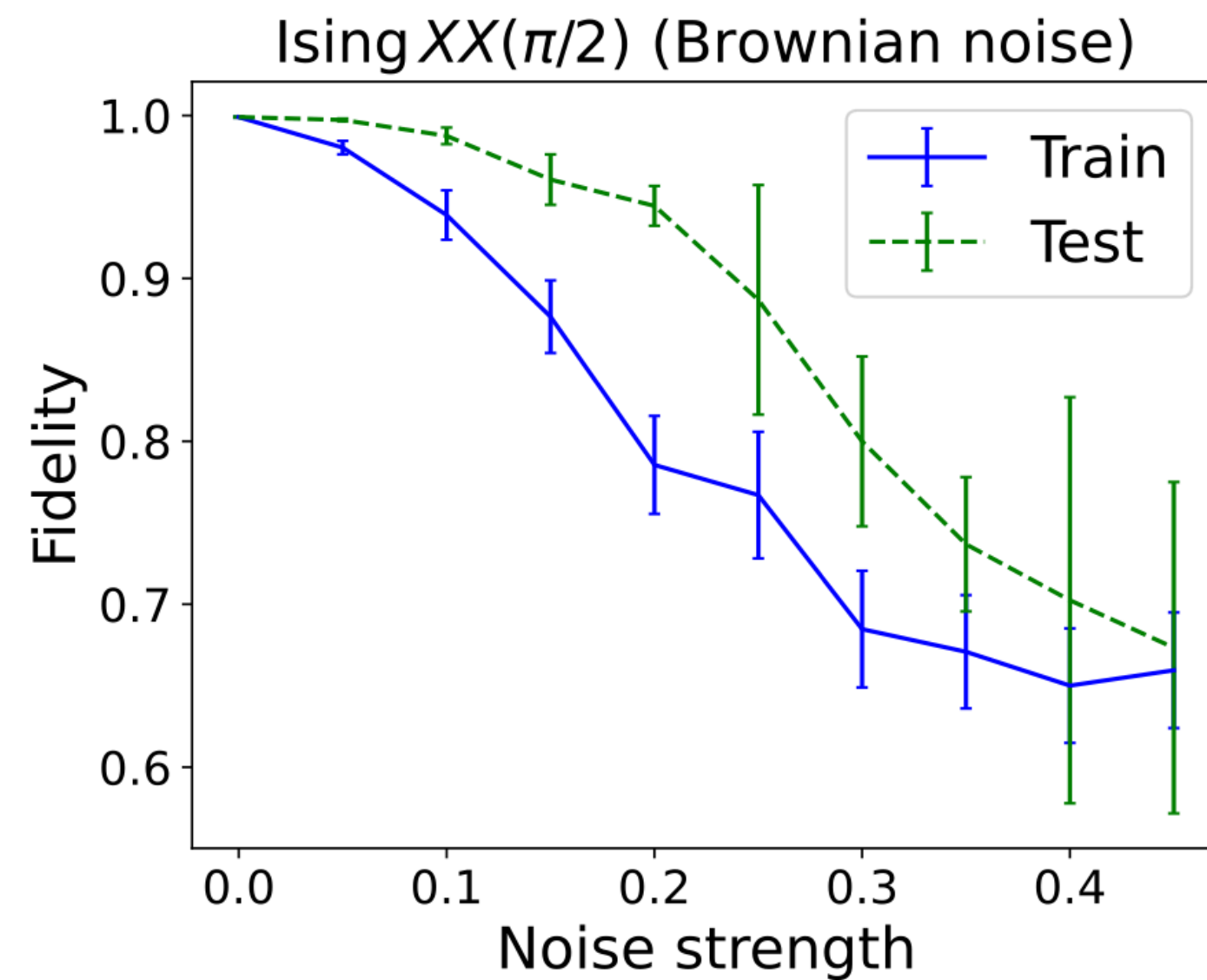


Universal set of gates

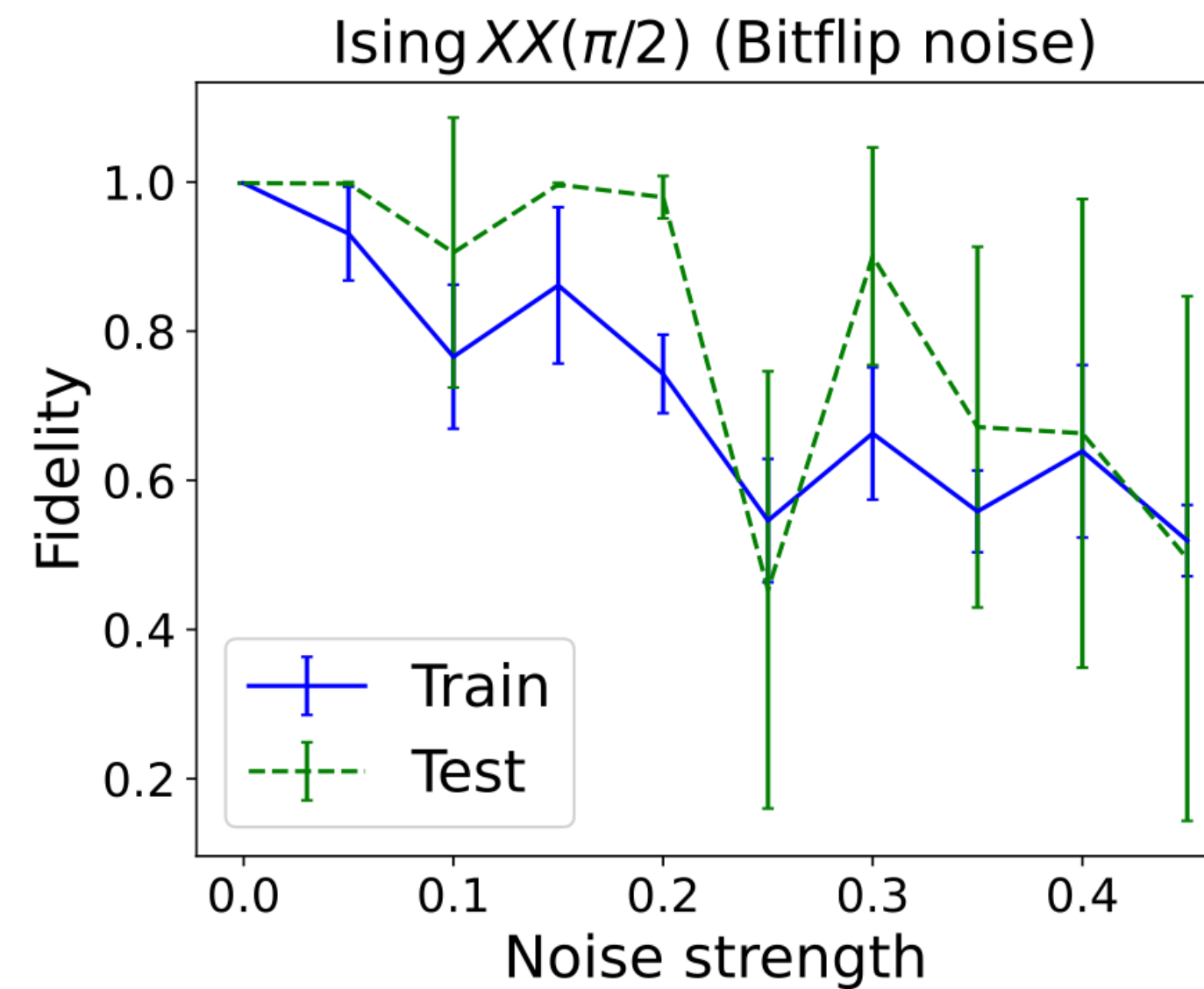
+ Noise

Noisy data $\left\{ \left(\rho_i, \mathcal{N}(\sigma_i) \right) \right\}_{i=1}^N$

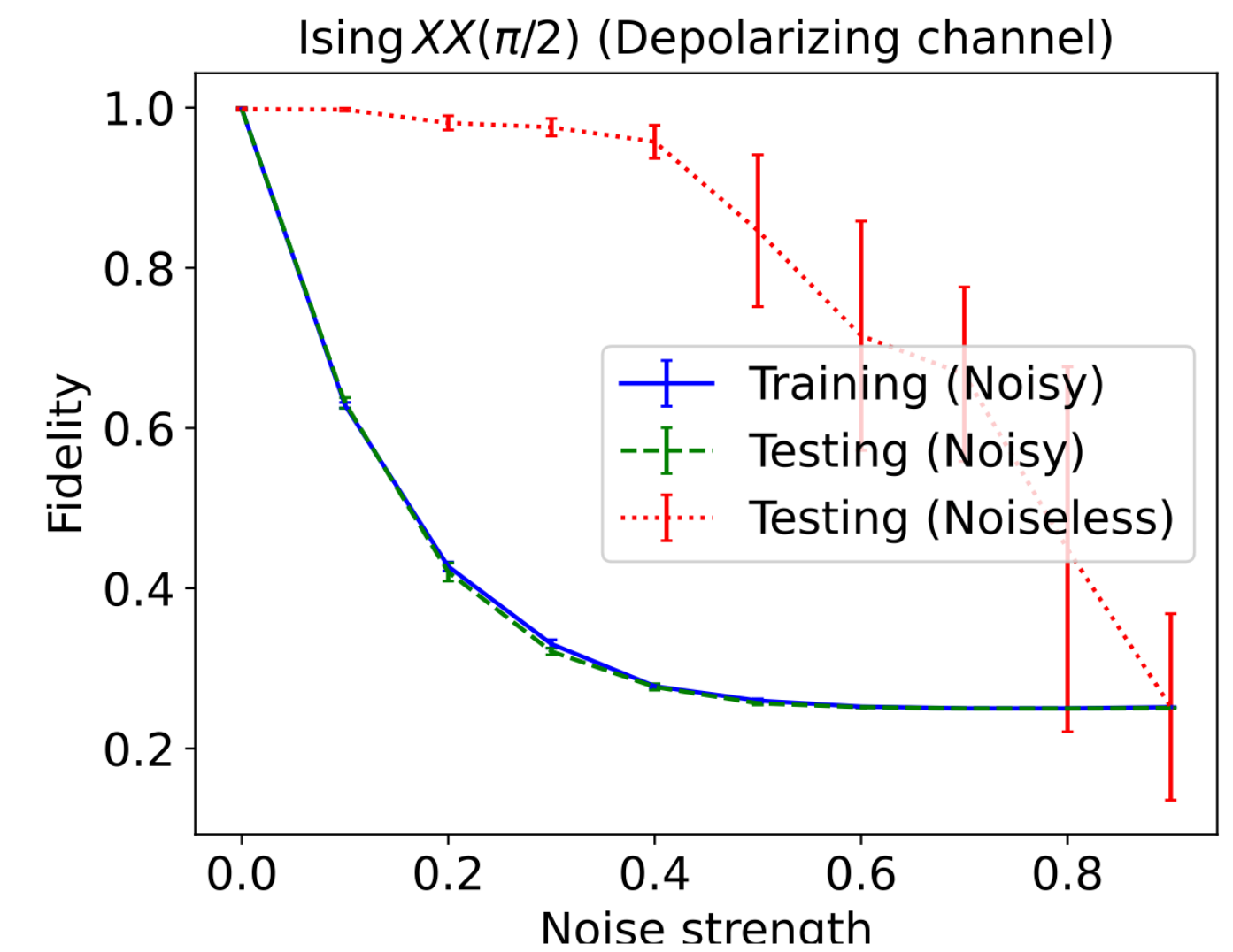
Noisy resource state $\mathcal{N}(|G\rangle)$



$$\mathcal{N}_{rand}(\rho) = (\mathbf{1} + \epsilon U)^\dagger \rho (\mathbf{1} + \epsilon U)$$



$$\mathcal{N}_{bit}(\rho) = (1 - p)\rho + pX\rho X$$



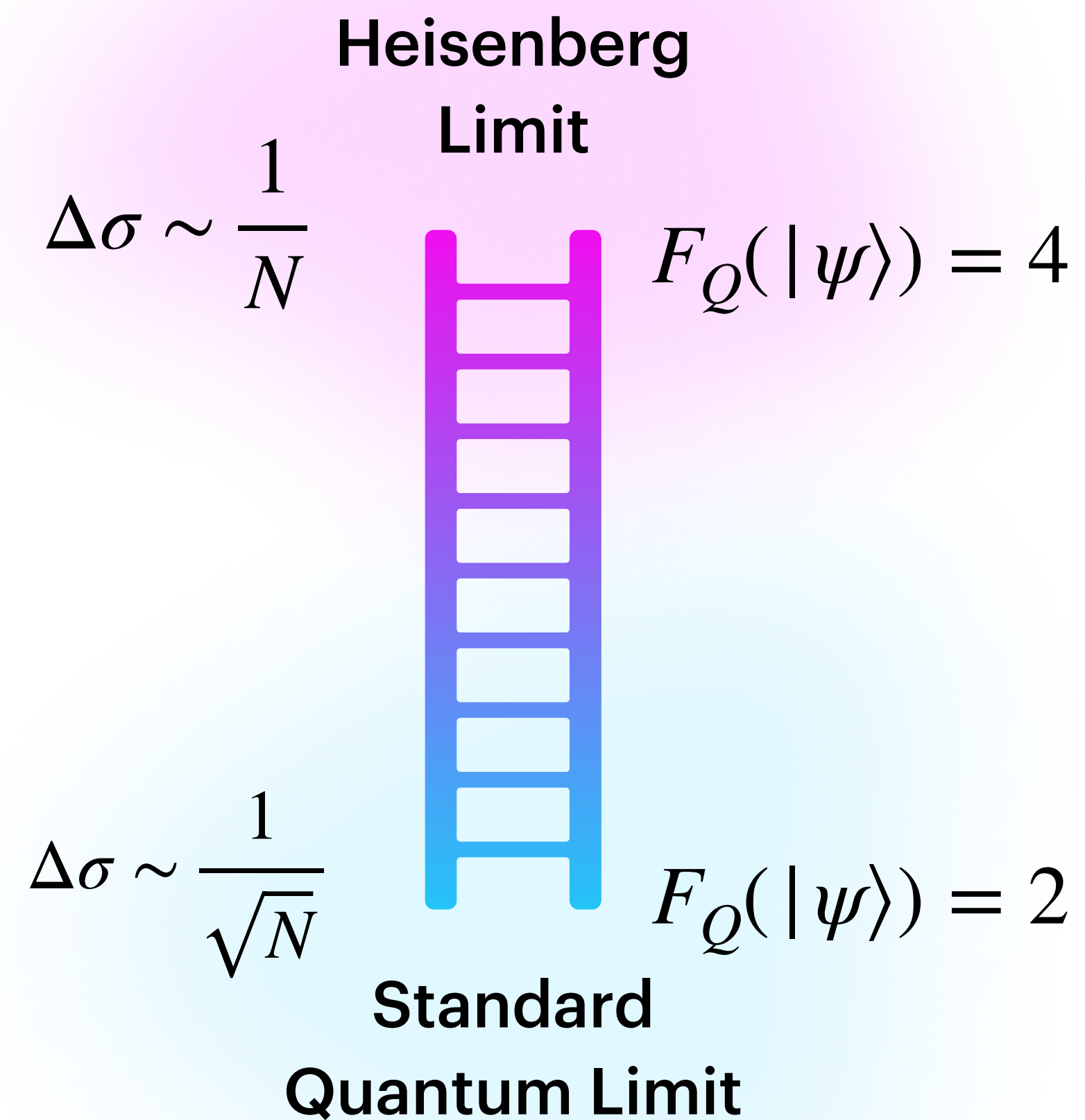
$$\mathcal{N}(\rho) = (1 - p)\rho + \frac{p}{3}X\rho X + \frac{p}{3}Y\rho Y + \frac{p}{3}Z\rho Z$$

Learning POVMs

Classifying states with high Fisher information

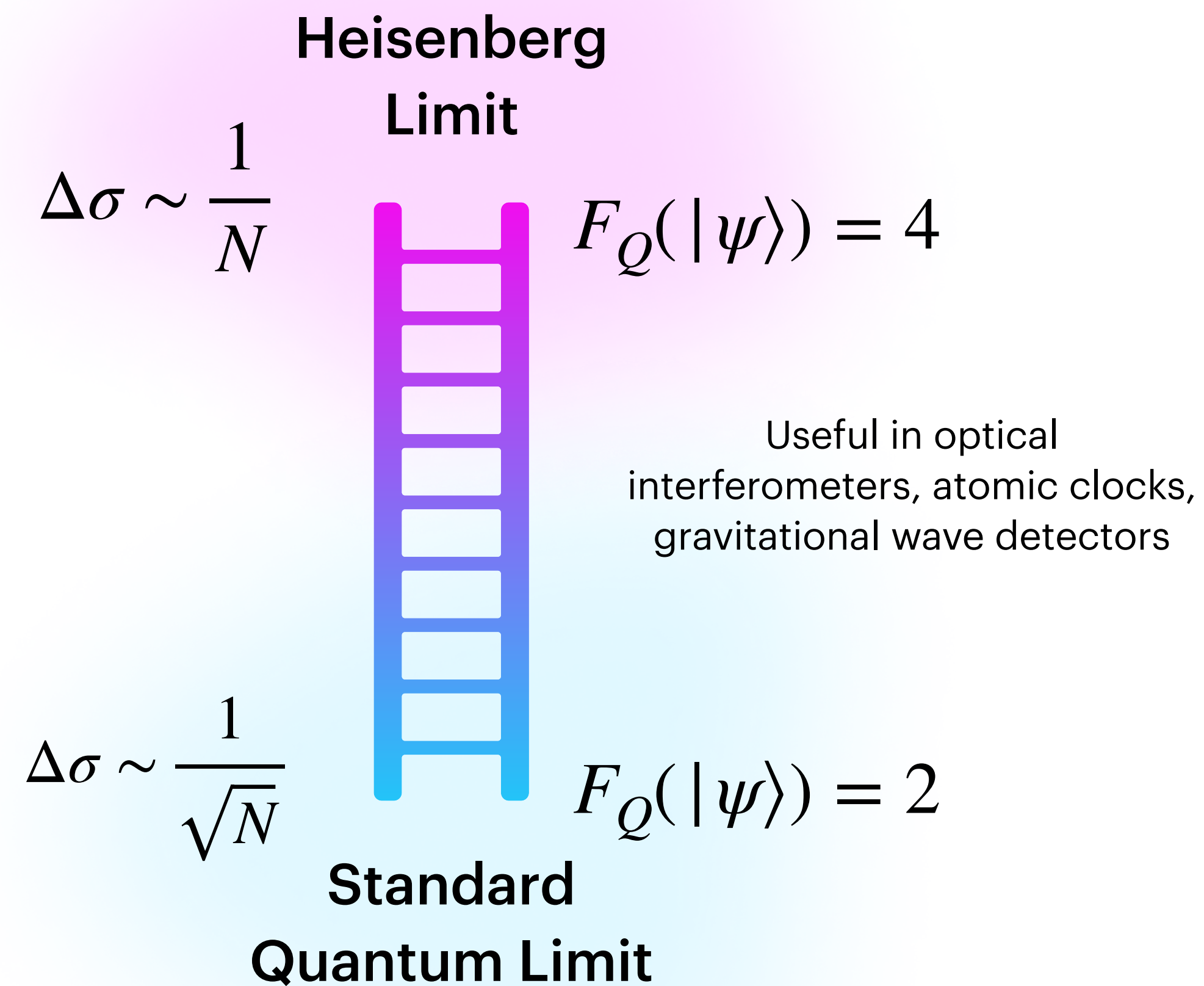
Learning POVMs

Classifying states with high Fisher information



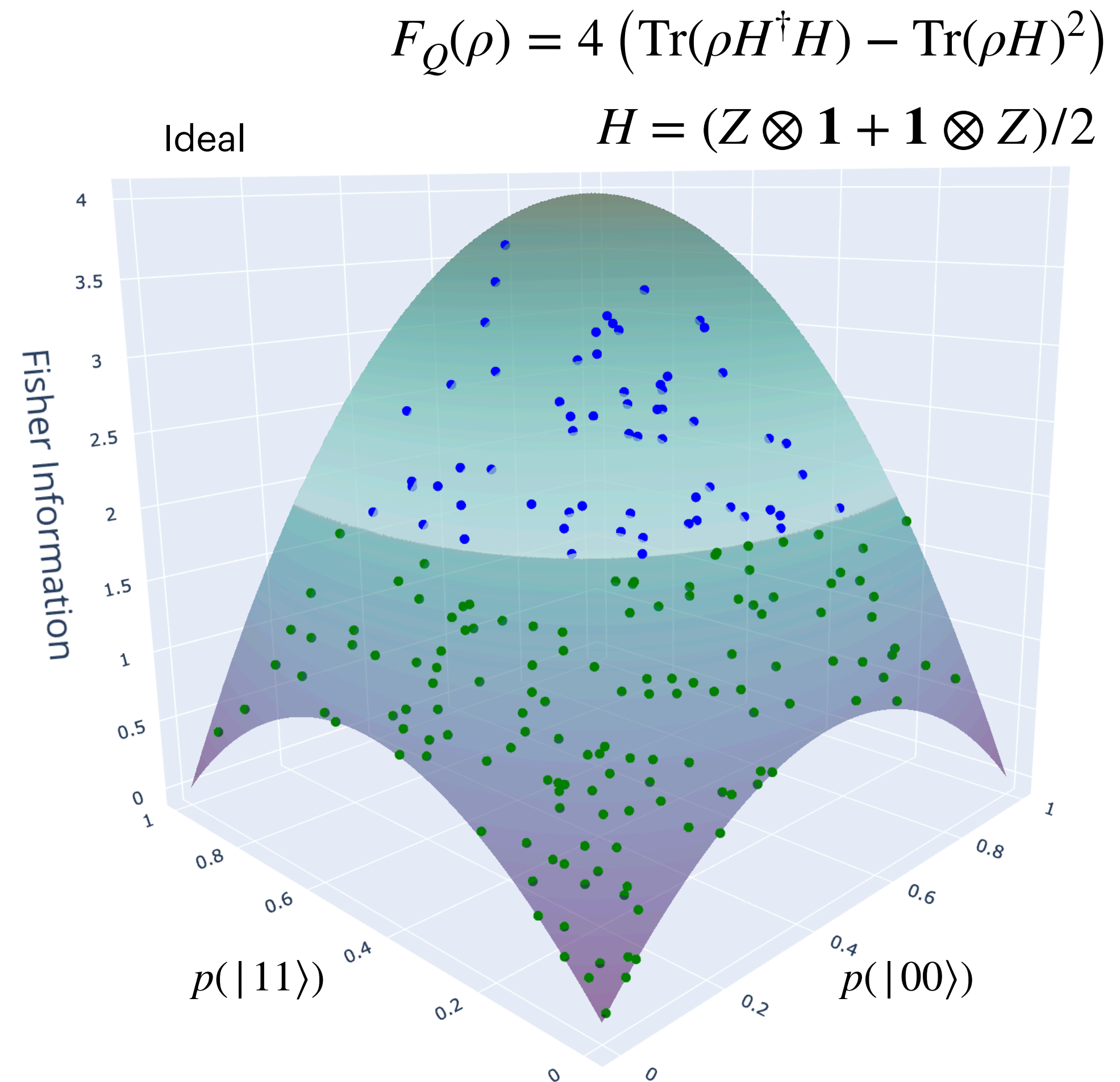
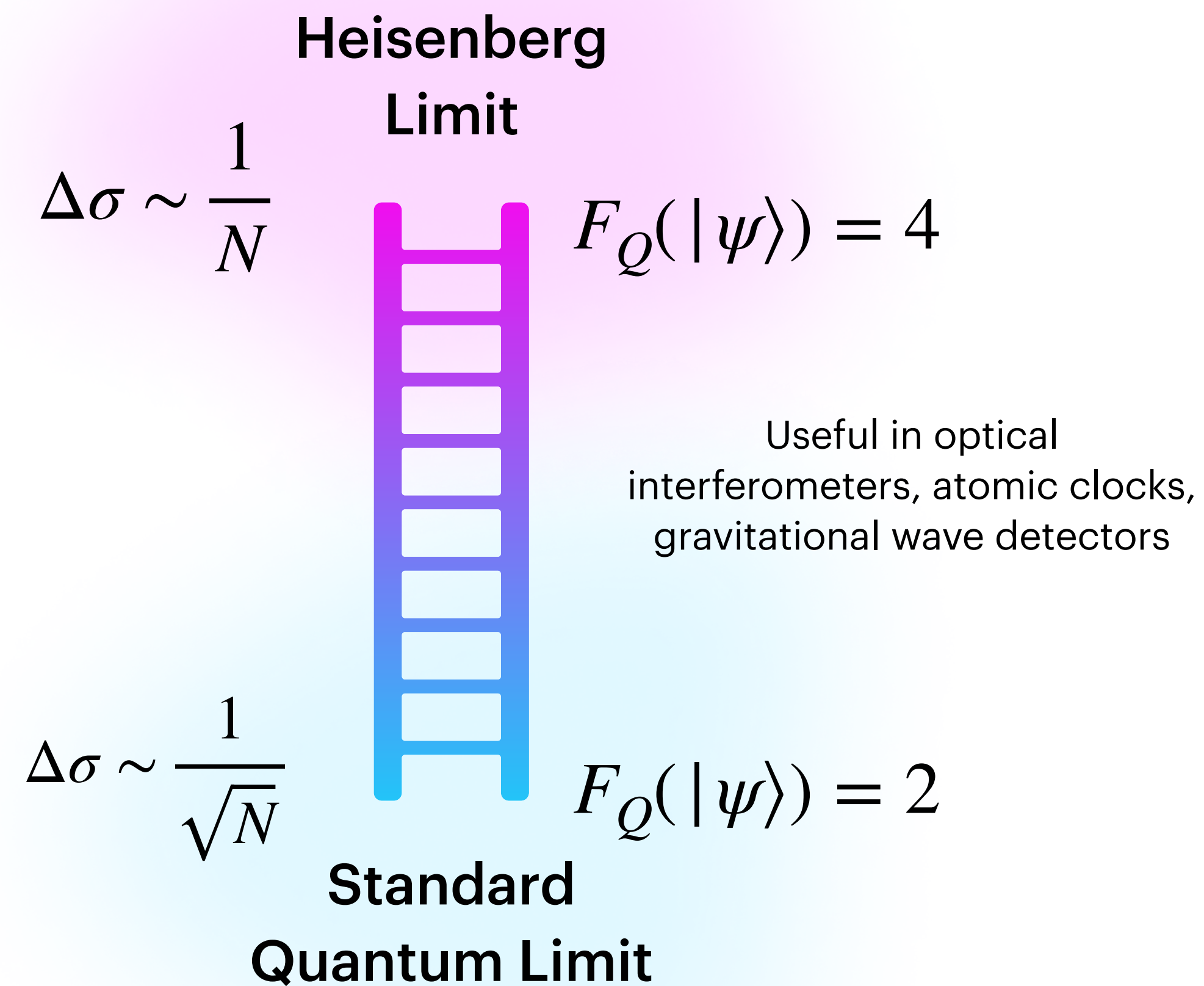
Learning POVMs

Classifying states with high Fisher information



Learning POVMs

Classifying states with high Fisher information



Learning POVMs

Classifying states with high Fisher information

Cost function

$$C(\vec{\theta}) = \frac{1}{N} \sum_{i=1}^N \left[y_i \max \left\{ 0, \hat{F}_Q(\rho_i) + 2 + \epsilon \right\} \right. \\ \left. + (1 - y_i) \max \left\{ 0, \hat{F}_Q(\rho_i) - 2 + \epsilon \right\} \right]$$

Dataset

$$S_1 = \left\{ \cos \theta |00\rangle + e^{i\phi} \sin \theta |11\rangle \mid \phi, \theta \in \mathcal{U}_{[0,2\pi]} \right\}$$

$$S_2 = \left\{ \cos \theta |++\rangle + e^{i\phi} \sin \theta |--\rangle \mid \phi, \theta \in \mathcal{U}_{[0,2\pi]} \right\}$$

We use these two datasets to avoid a basis that make them 1D families rather than 2D

Learning POVMs

Classifying states with high Fisher information

$$\hat{F}_Q(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_1 x_2 + \beta_5 x_2^2$$

Cost function

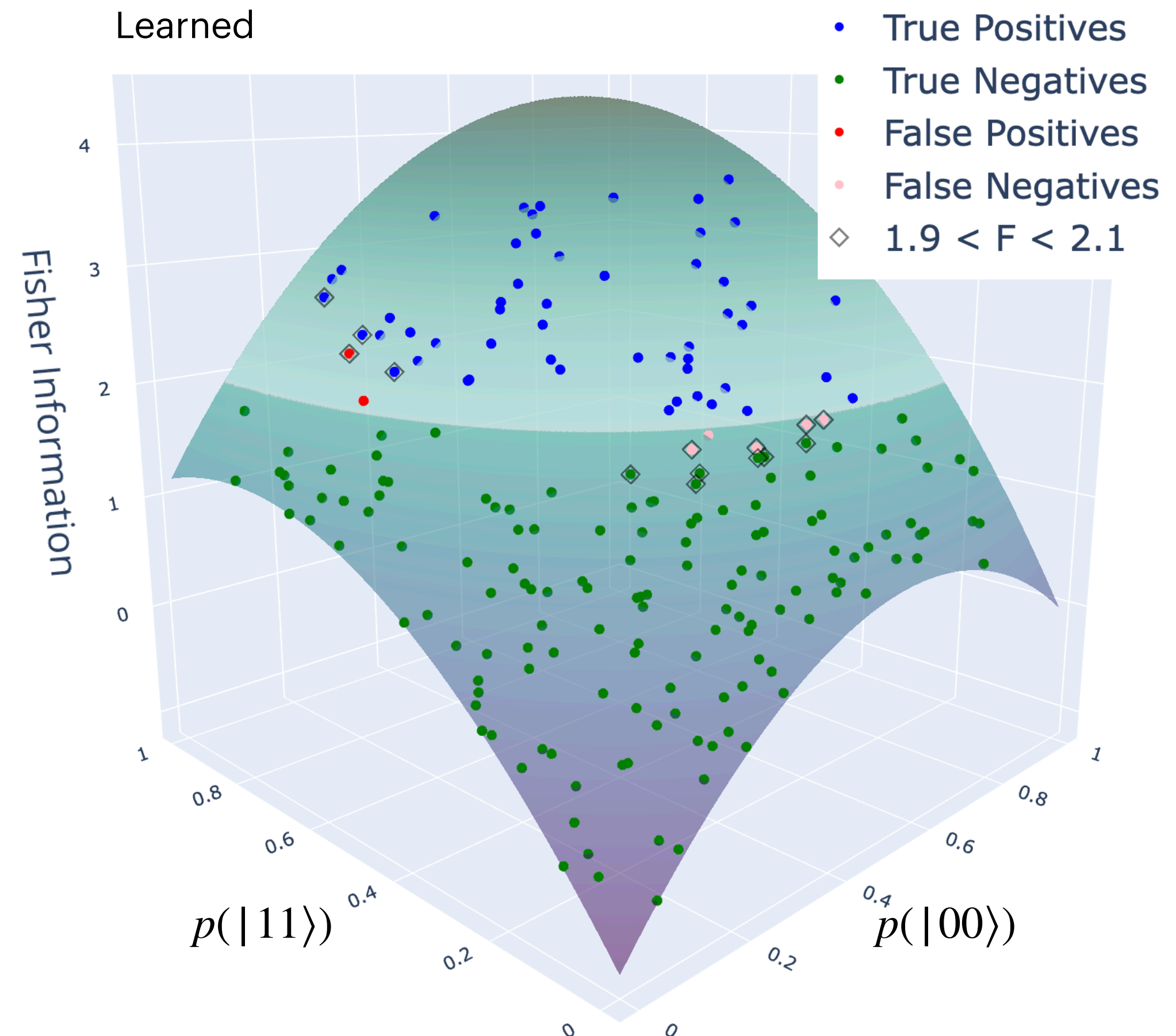
$$C(\vec{\theta}) = \frac{1}{N} \sum_{i=1}^N \left[y_i \max \left\{ 0, \hat{F}_Q(\rho_i) + 2 + \epsilon \right\} \right. \\ \left. + (1 - y_i) \max \left\{ 0, \hat{F}_Q(\rho_i) - 2 + \epsilon \right\} \right]$$

Dataset

$$S_1 = \left\{ \cos \theta |00\rangle + e^{i\phi} \sin \theta |11\rangle \mid \phi, \theta \in \mathcal{U}_{[0,2\pi]} \right\}$$

$$S_2 = \left\{ \cos \theta |++\rangle + e^{i\phi} \sin \theta |--\rangle \mid \phi, \theta \in \mathcal{U}_{[0,2\pi]} \right\}$$

We use these two datasets to avoid a basis that make them 1D families rather than 2D



Learning POVMs

Classifying states with high Fisher information

$$\hat{F}_Q(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_1 x_2 + \beta_5 x_2^2$$

Cost function

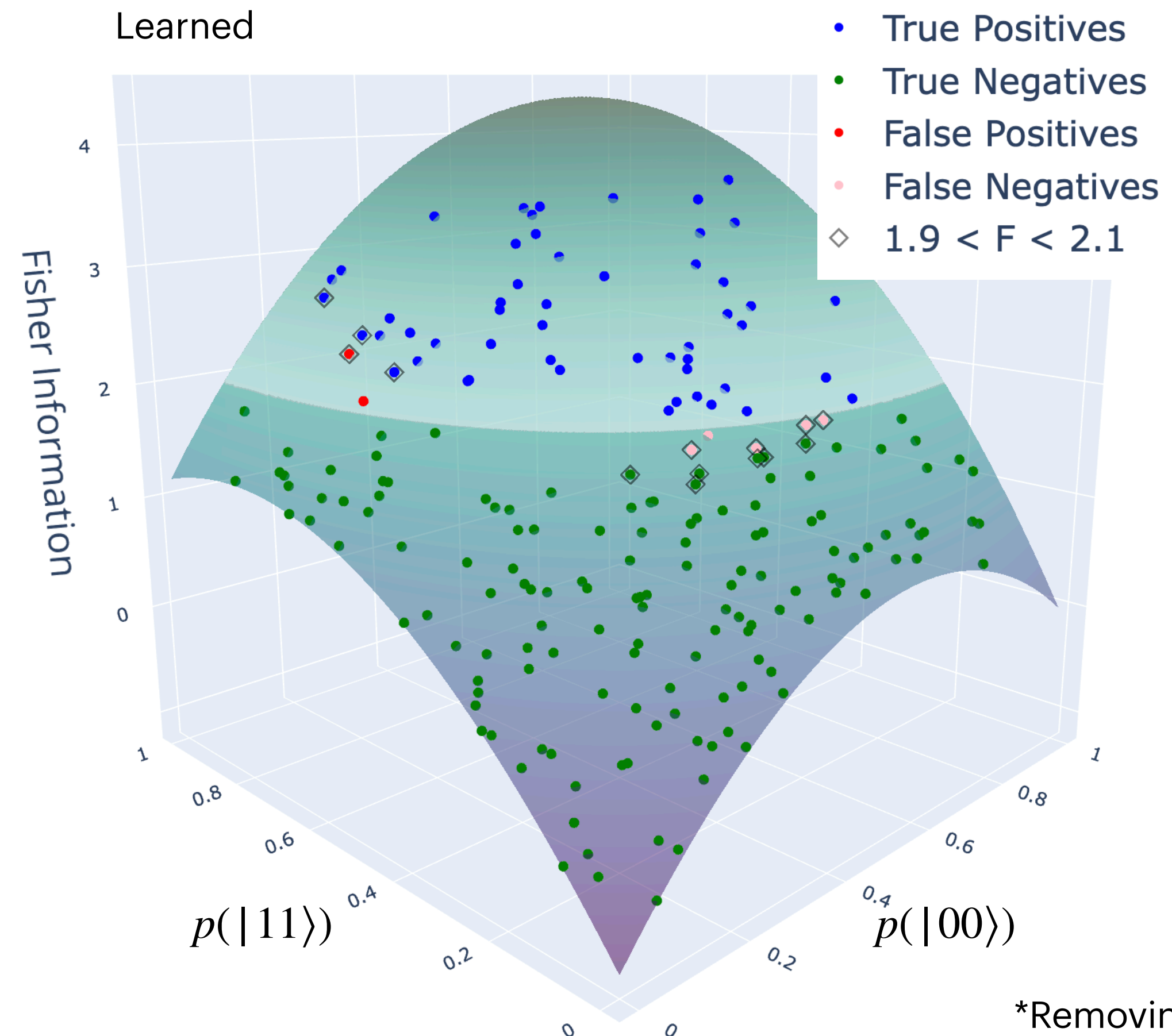
$$C(\vec{\theta}) = \frac{1}{N} \sum_{i=1}^N \left[y_i \max \left\{ 0, \hat{F}_Q(\rho_i) + 2 + \epsilon \right\} \right. \\ \left. + (1 - y_i) \max \left\{ 0, \hat{F}_Q(\rho_i) - 2 + \epsilon \right\} \right]$$

Dataset

$$S_1 = \left\{ \cos \theta |00\rangle + e^{i\phi} \sin \theta |11\rangle \mid \phi, \theta \in \mathcal{U}_{[0,2\pi]} \right\}$$

$$S_2 = \left\{ \cos \theta |++\rangle + e^{i\phi} \sin \theta |--\rangle \mid \phi, \theta \in \mathcal{U}_{[0,2\pi]} \right\}$$

We use these two datasets to avoid a basis that make them 1D families rather than 2D



Accuracy:*
 0.9725 ± 0.0042

False positives:
 0.0149 ± 0.0006

False negatives:
 0.0125 ± 0.0006

*Removing 0.2 band around $F=2$
Without it the accuracy is 0.9479

Quantum Instruments

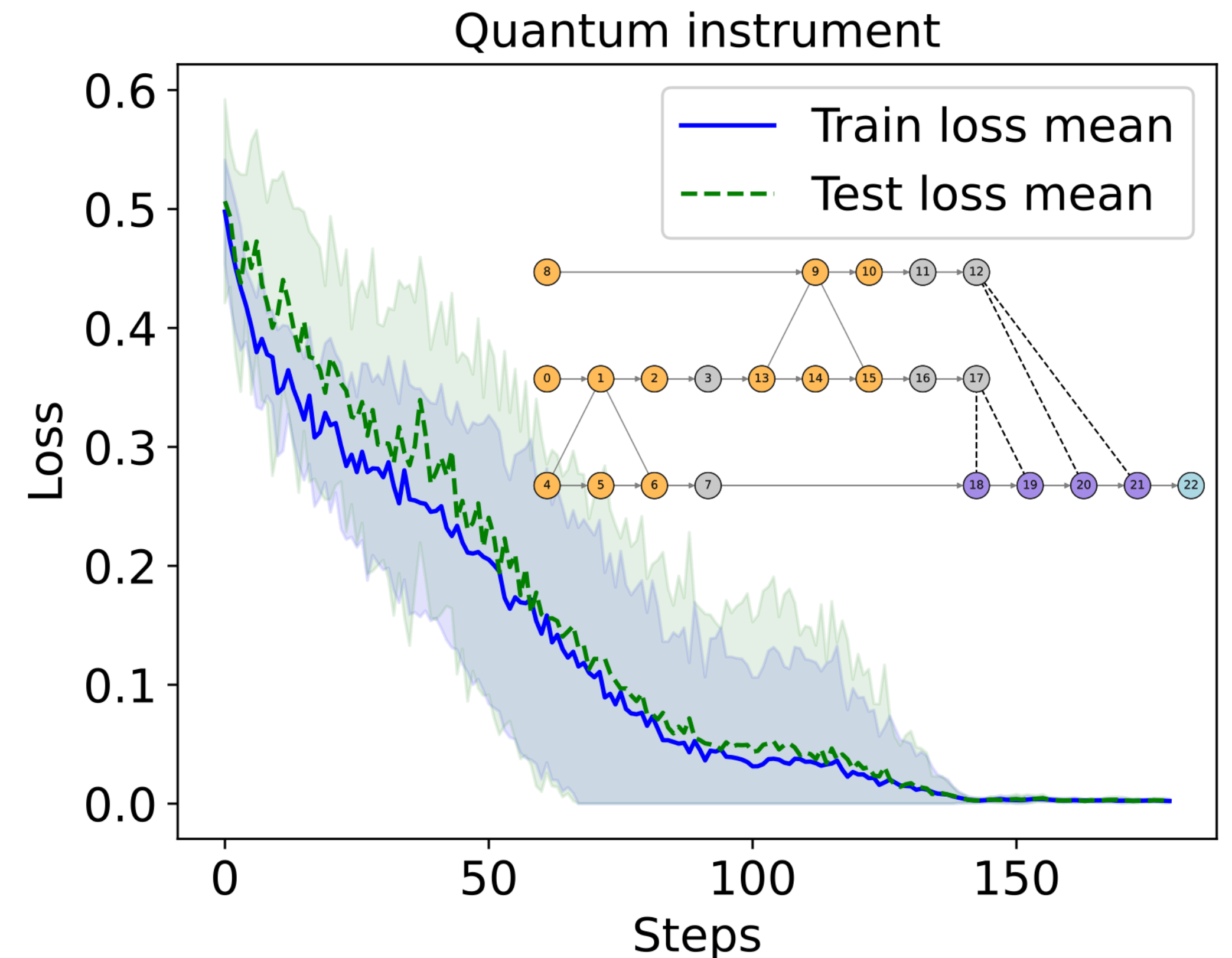
A teleportation device 🥁🎹

Since each measurement outcome has a different loss

$$C_{avg} = \langle C_{00\dots 0} + C_{10\dots 0} + \dots C_{11\dots 1} \rangle$$

Controlled measurements introduce more noise to the generalization error

$$N_{samples} = \mathcal{O}\left(\frac{1}{\epsilon^2}\right) \longrightarrow \mathcal{O}\left(\frac{2^m}{\epsilon^2}\right)$$

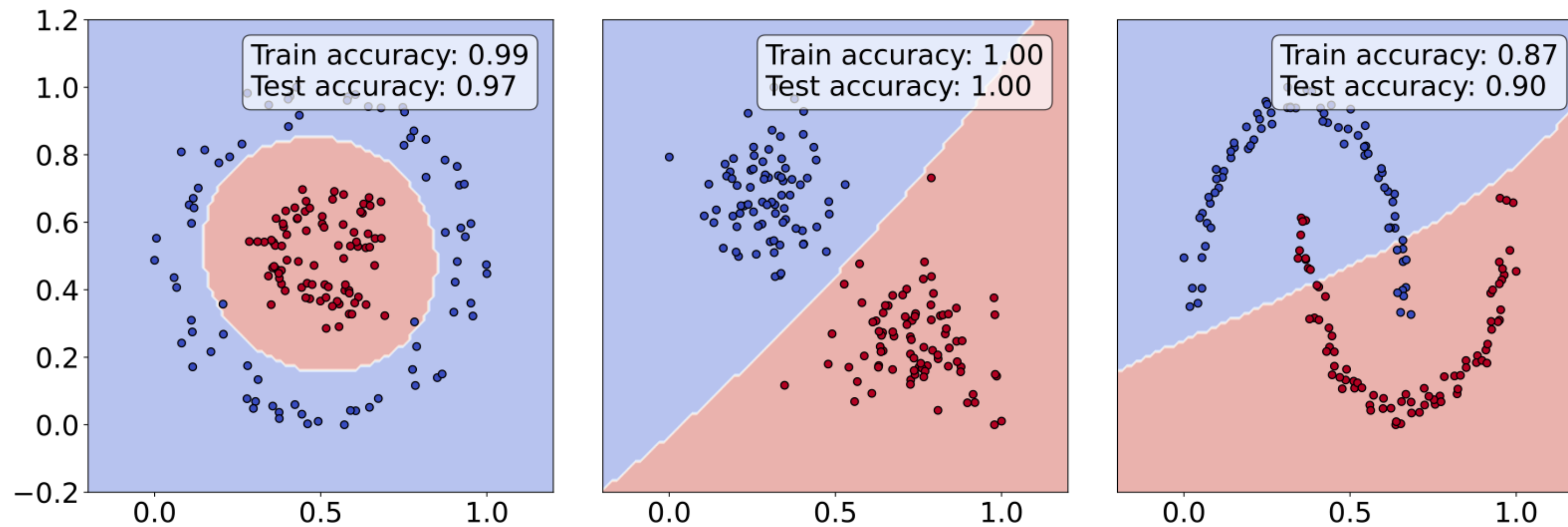


Classical data

SVM with a quantum kernel 🌽

$$K(x_i, x_j) = \langle \phi(x_i) | \phi(x_j) \rangle$$

$$\phi(x) = U_{z1}(x_1)U_{z0}(x_0)U_{xx}(\cos(x_0)\cos(x_1))U_{z1}(x_1)U_{z0}(x_0)$$

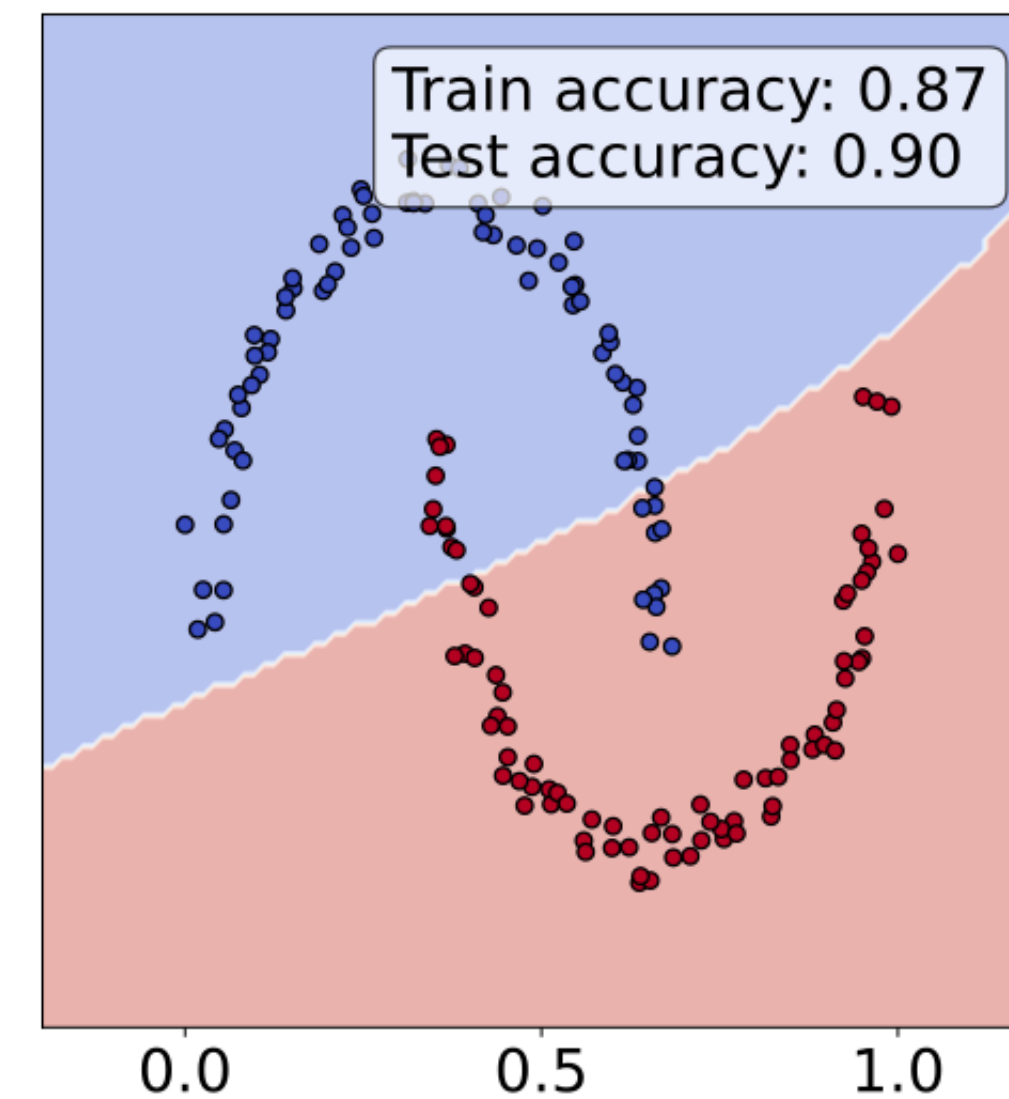
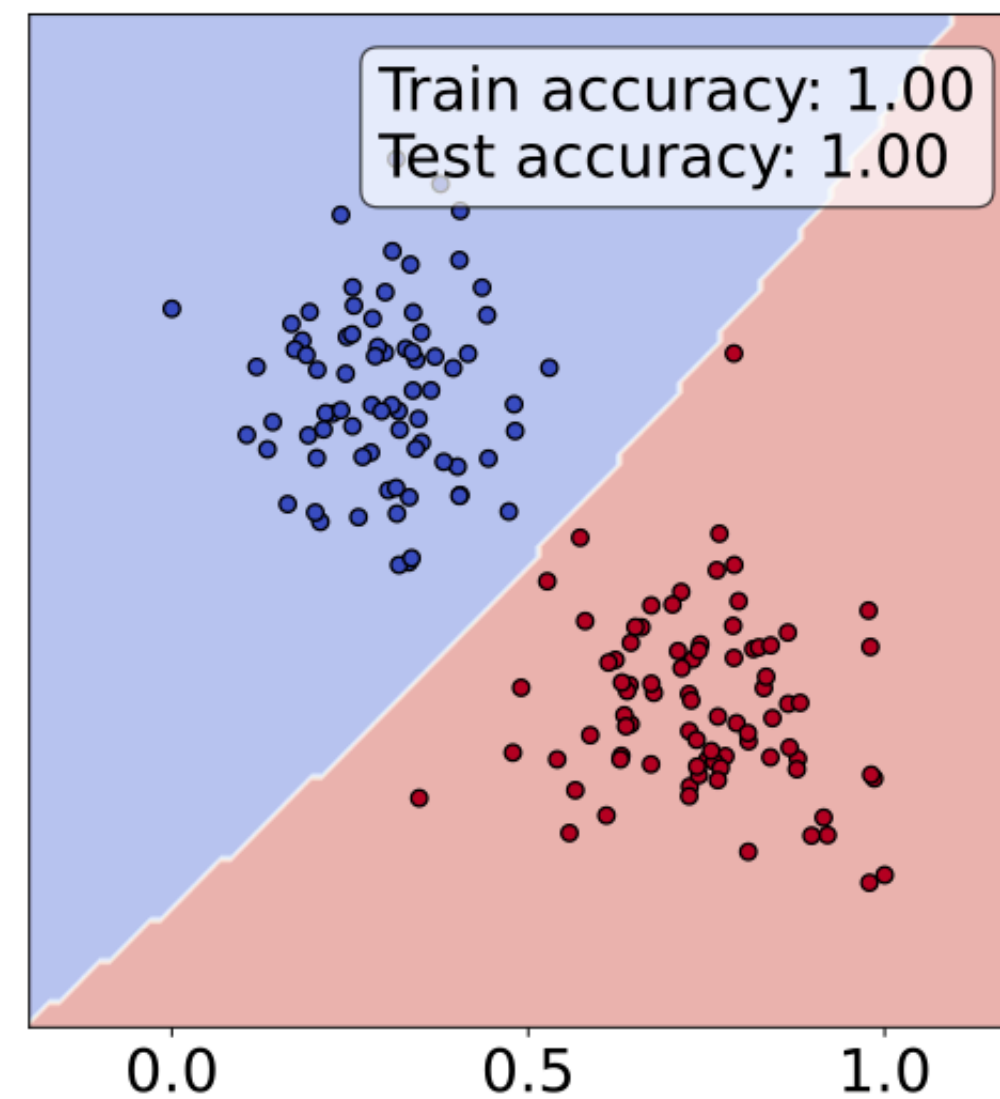
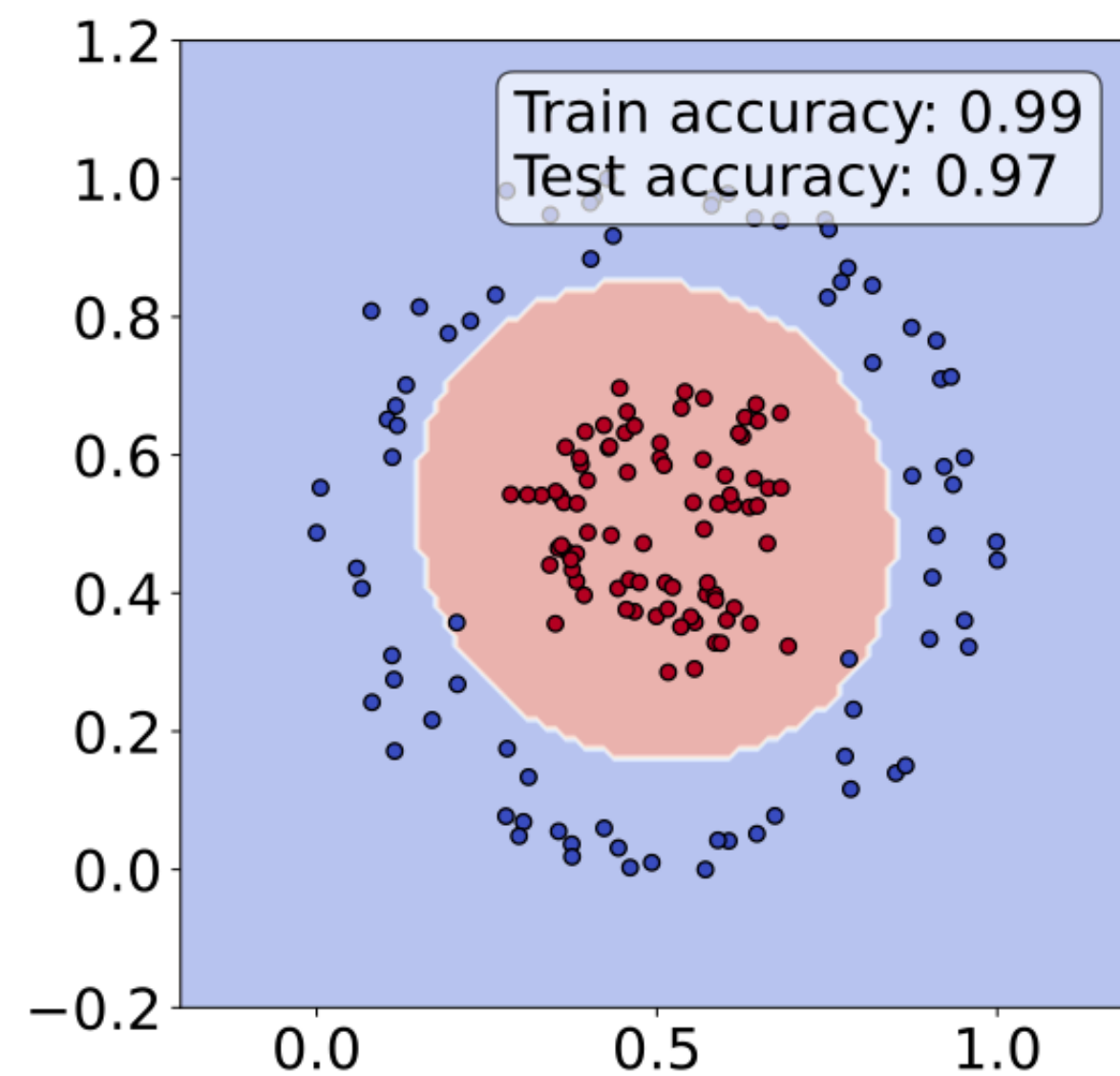
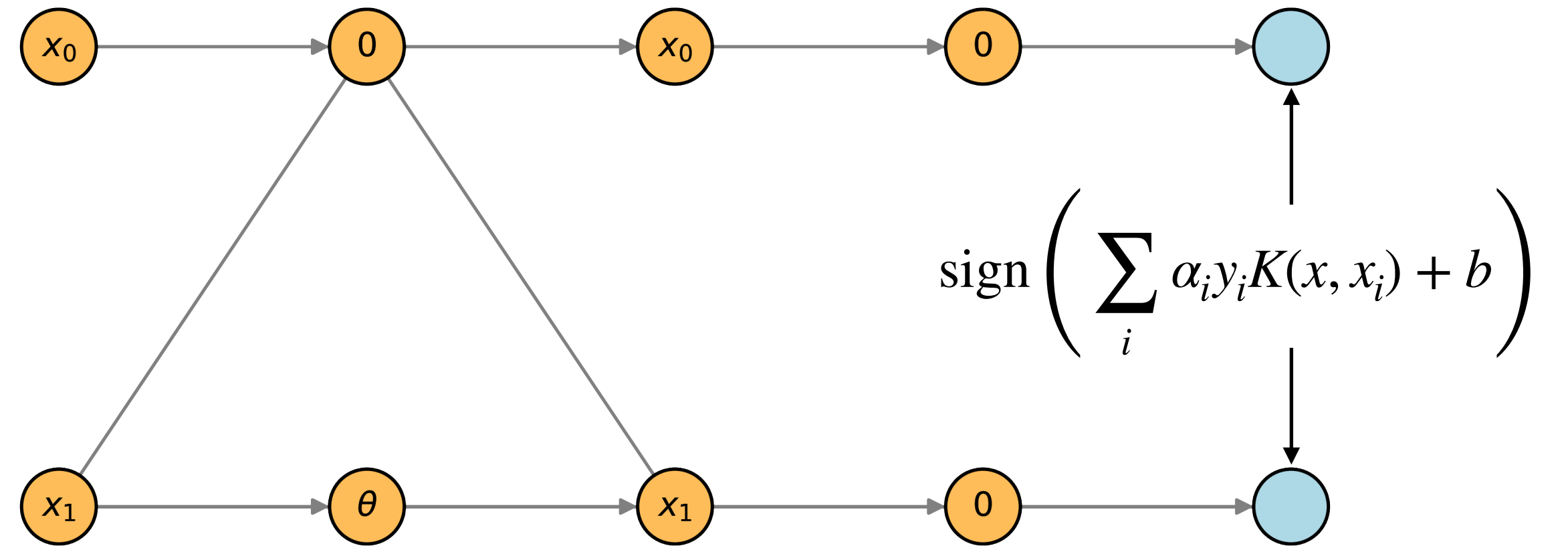


Classical data

SVM with a quantum kernel 🌽

$$K(x_i, x_j) = \langle \phi(x_i) | \phi(x_j) \rangle$$

$$\phi(x) = U_{z1}(x_1)U_{z0}(x_0)U_{xx}(\cos(x_0)\cos(x_1))U_{z1}(x_1)U_{z0}(x_0)$$

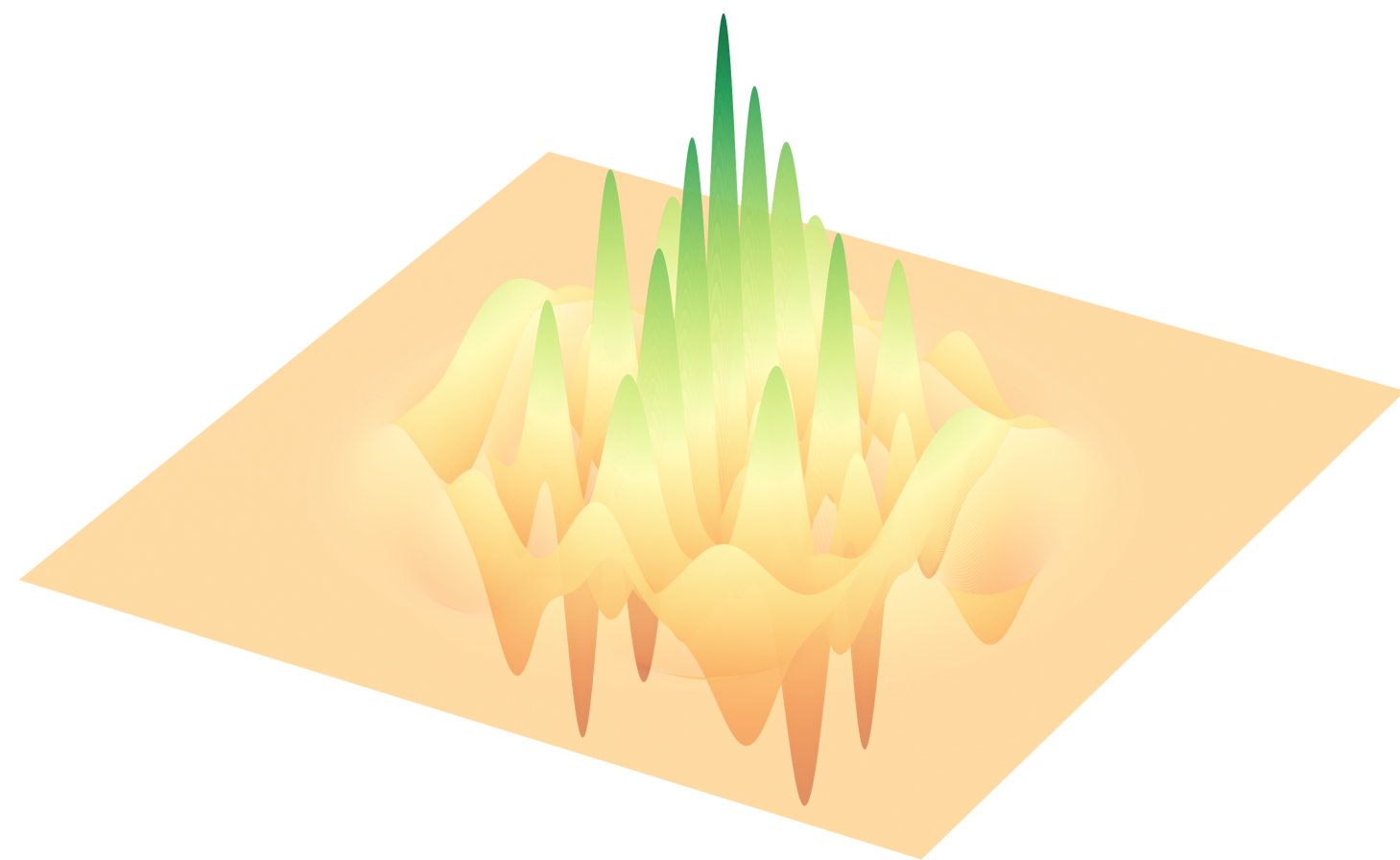


Hardware efficient Ansatz

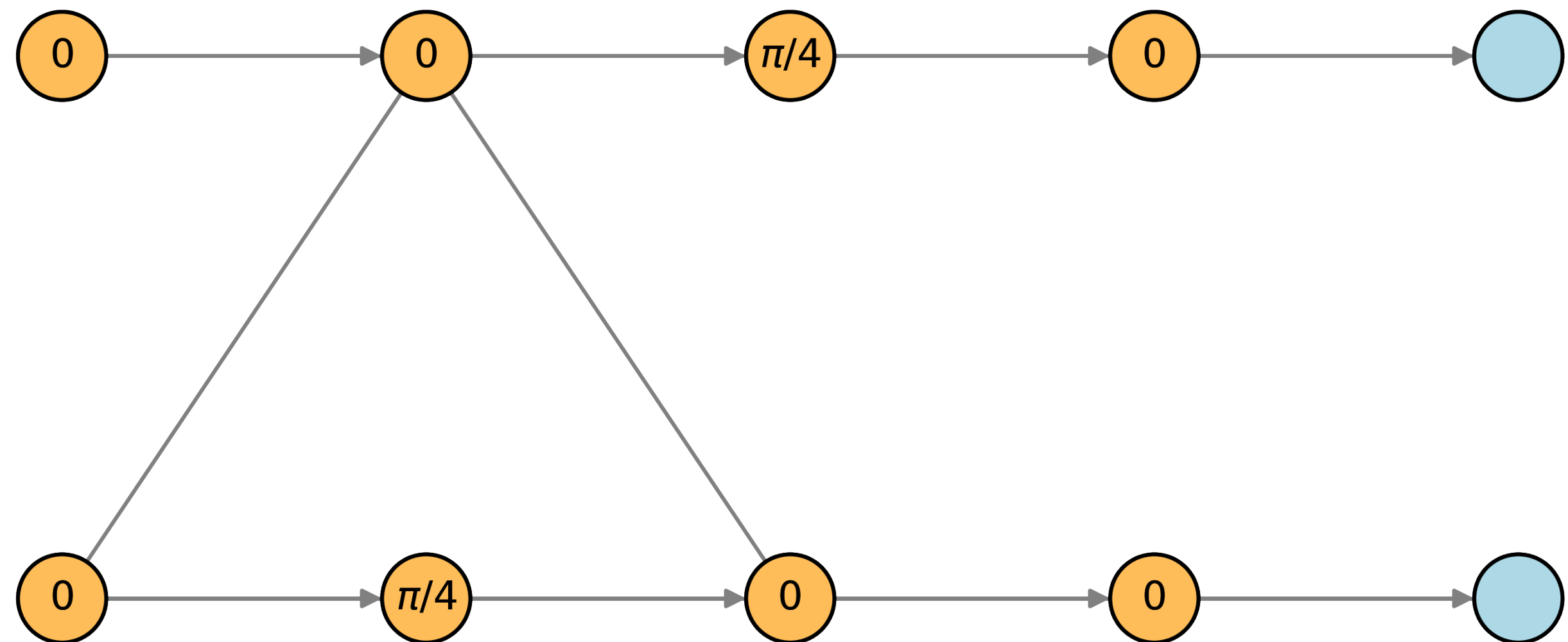
Pauli measurement + T magic states ✨

Restrictions:

GKP encoding + Homodyne
detection of p and q quadratures



$$U = (T \otimes \mathbf{1}) e^{-i\frac{\pi}{8}X \otimes X}$$

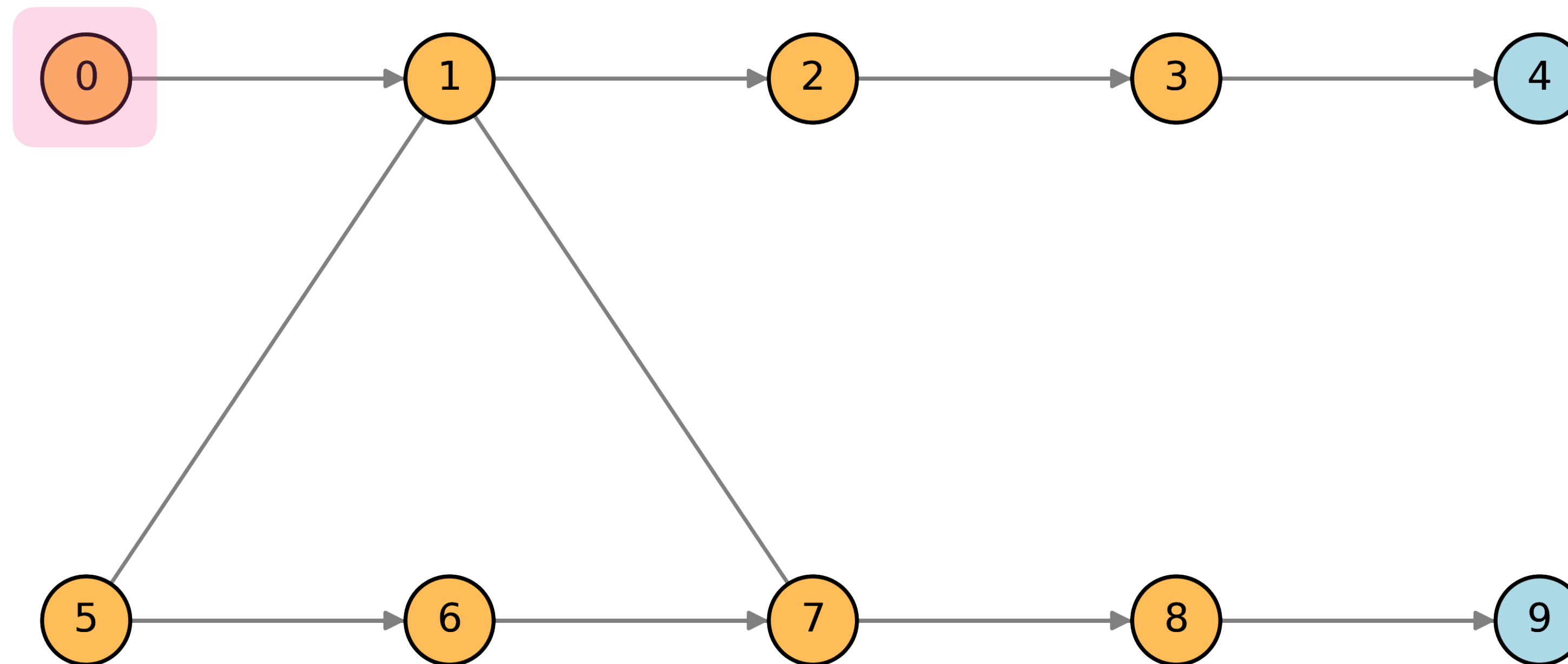


Discrete optimization is **NP-hard** 😊

Hardware efficient Ansatz

ϵ -Greedy layer optimization

Idea: Do a greedy random search in local neighbourhood

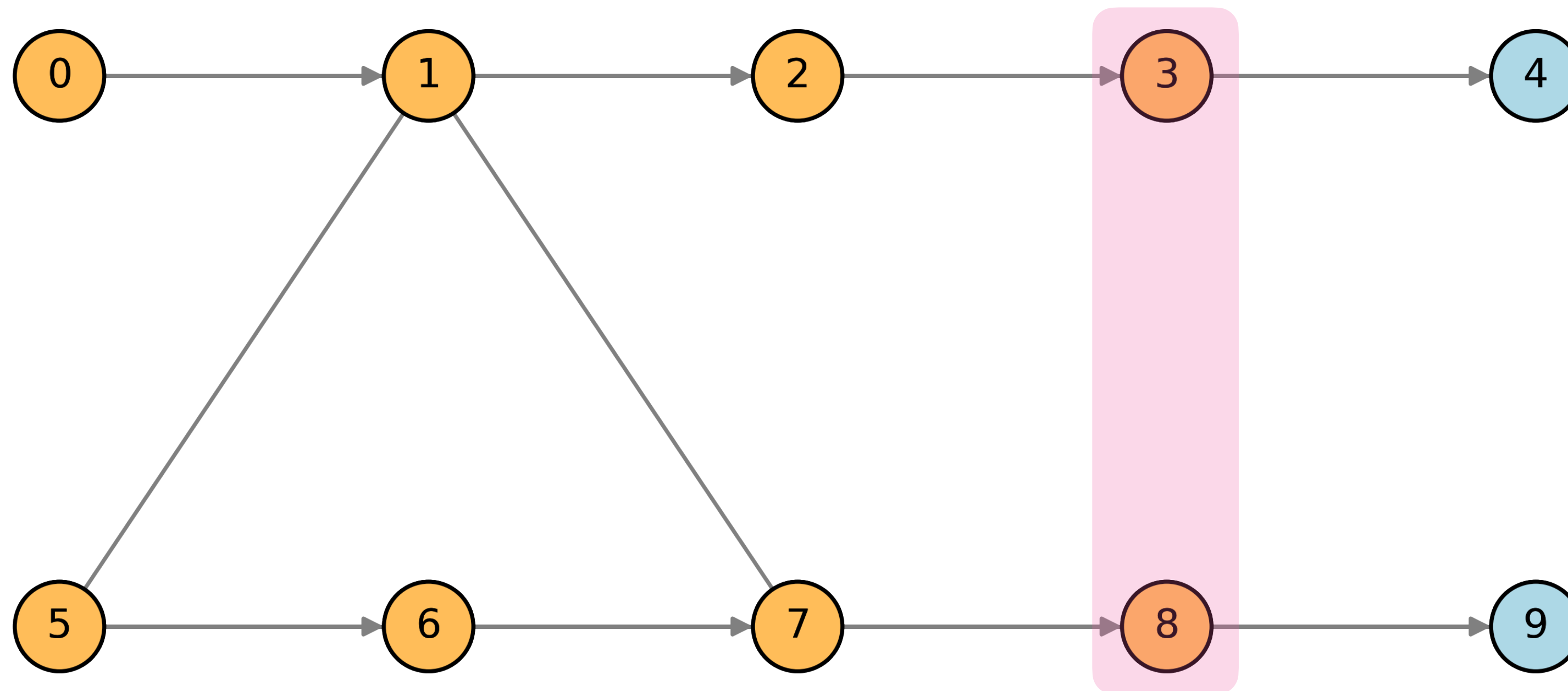


Ordered Layers = `[[0], [5], [6], [1], [2, 7], [3, 8], [4, 9]]`

Hardware efficient Ansatz

ϵ -Greedy layer optimization

Idea: Do a greedy random search in local neighbourhood

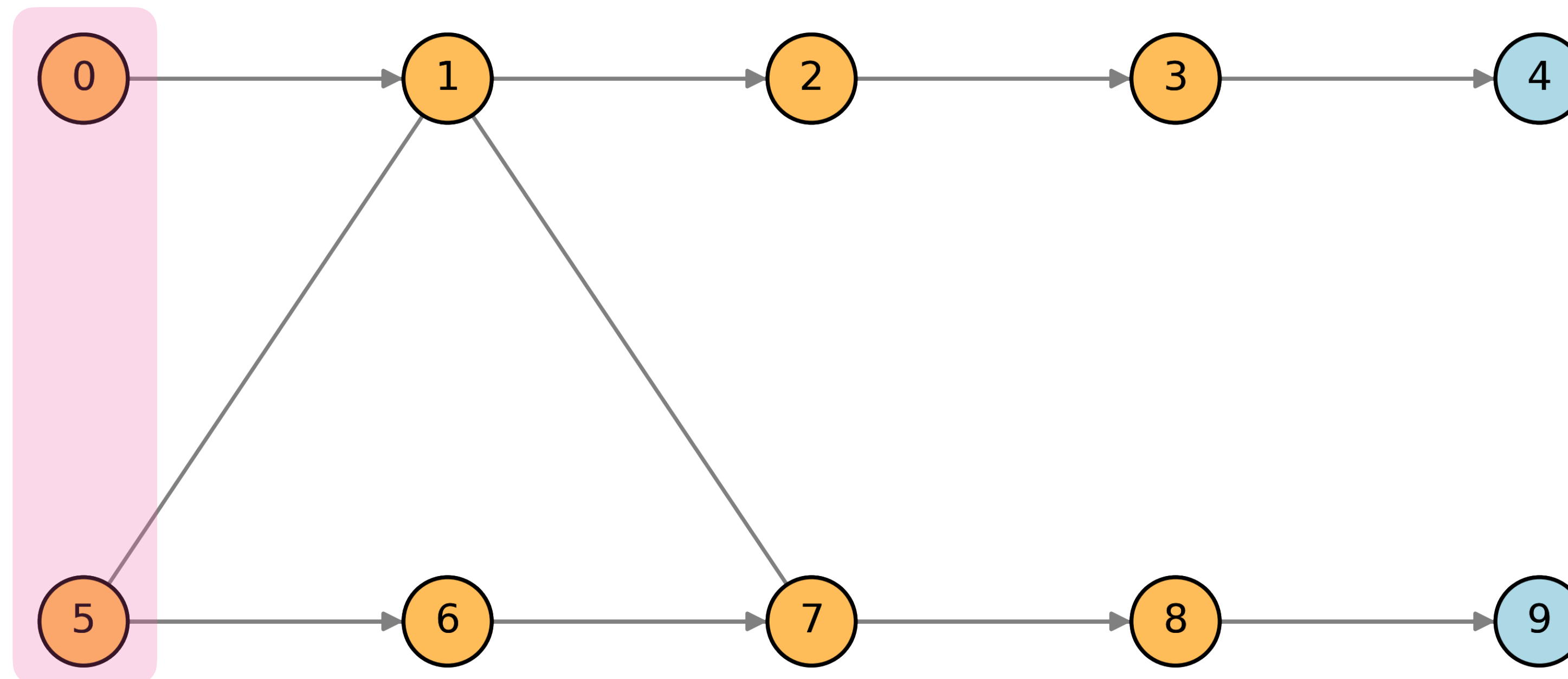


Ordered Layers = `[[0], [5], [6], [1], [2, 7], [3, 8], [4, 9]]`

Hardware efficient Ansatz

ϵ -Greedy layer optimization

Idea: Do a greedy random search in local neighbourhood

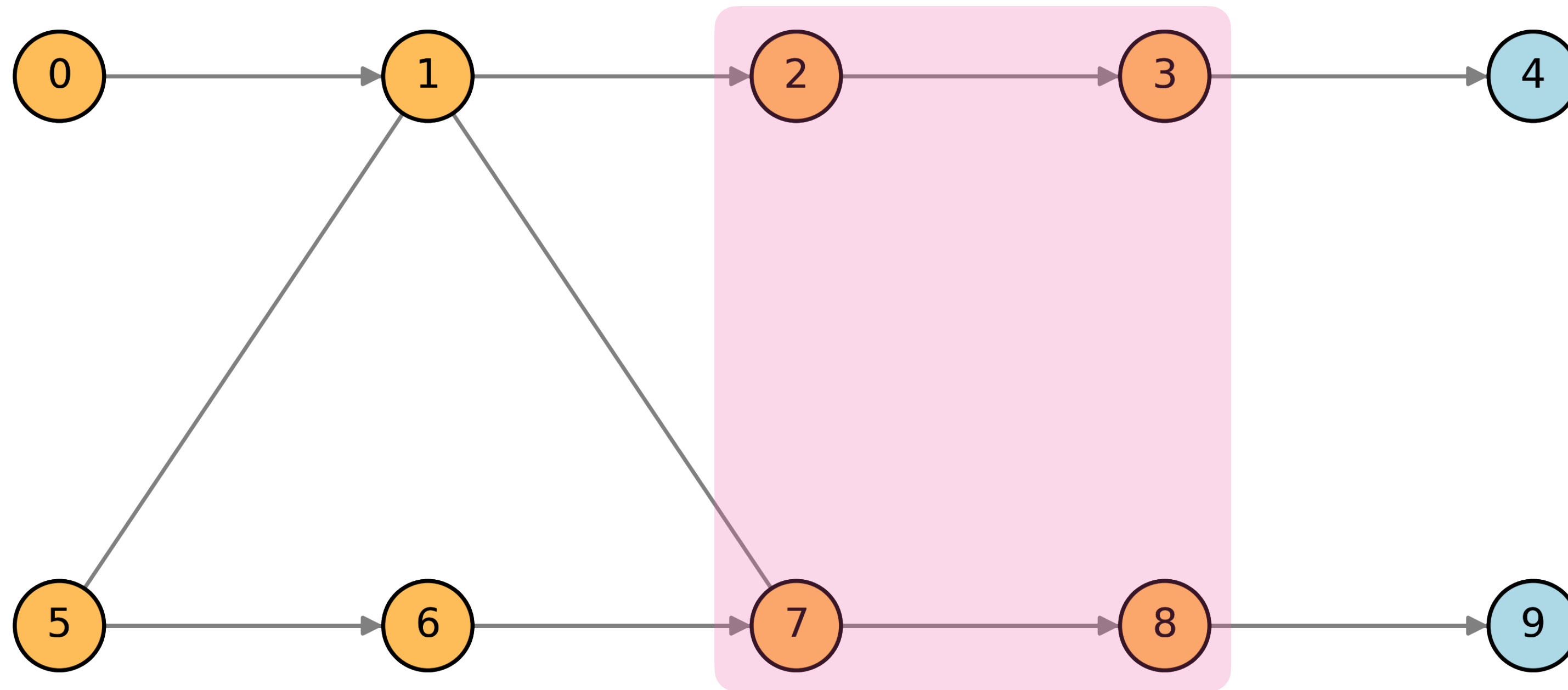


Ordered Layers = `[[0], [5], [6], [1], [2, 7], [3, 8], [4, 9]]`

Hardware efficient Ansatz

ϵ -Greedy layer optimization

Idea: Do a greedy random search in local neighbourhood

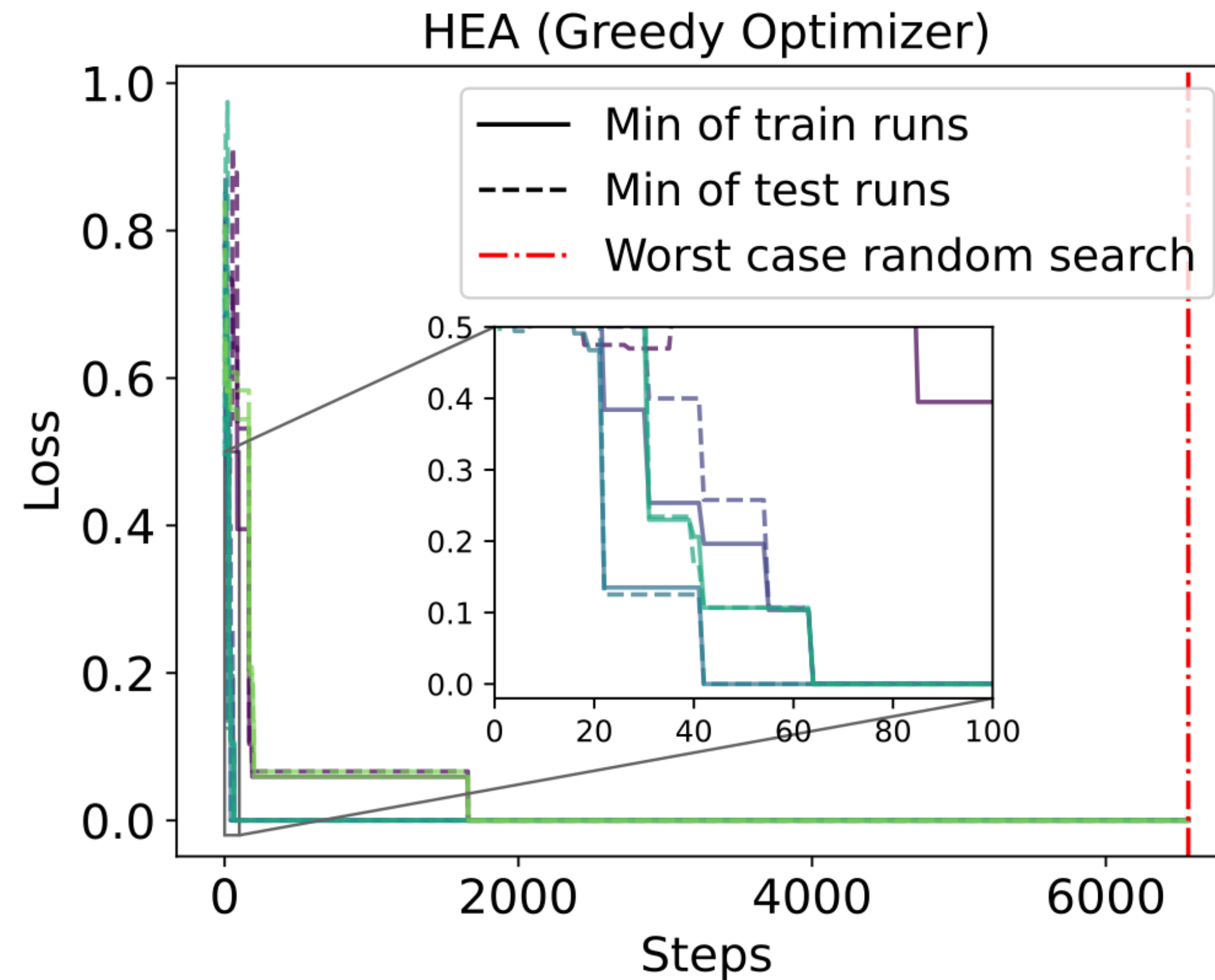


Ordered Layers = `[[0], [5], [6], [1], [2, 7], [3, 8], [4, 9]]`

Hardware efficient Ansatz

ϵ -Greedy layer optimization

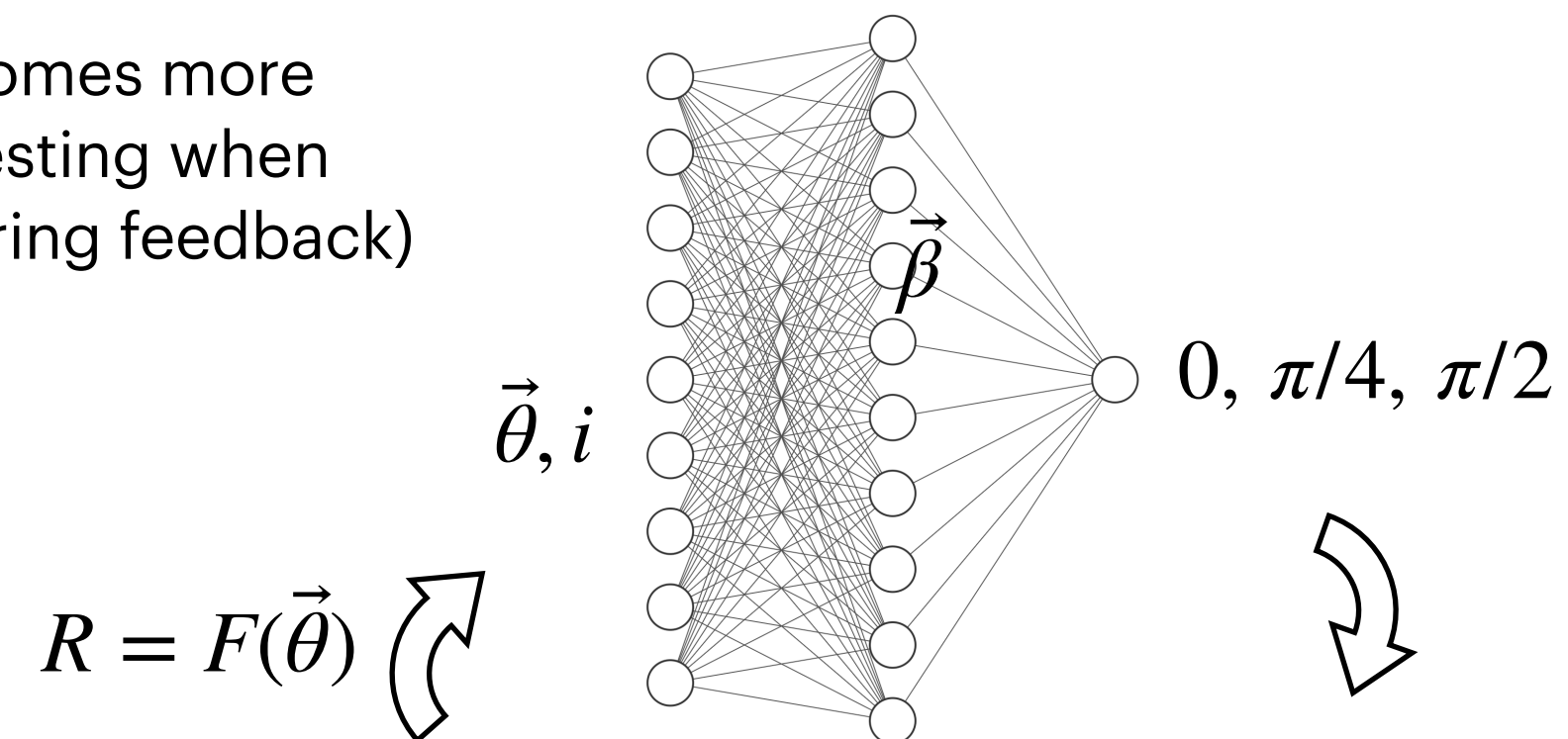
Idea: Do a greedy random search in local neighbourhood



Hardware efficient Ansatz

Deep Q-learning optimization

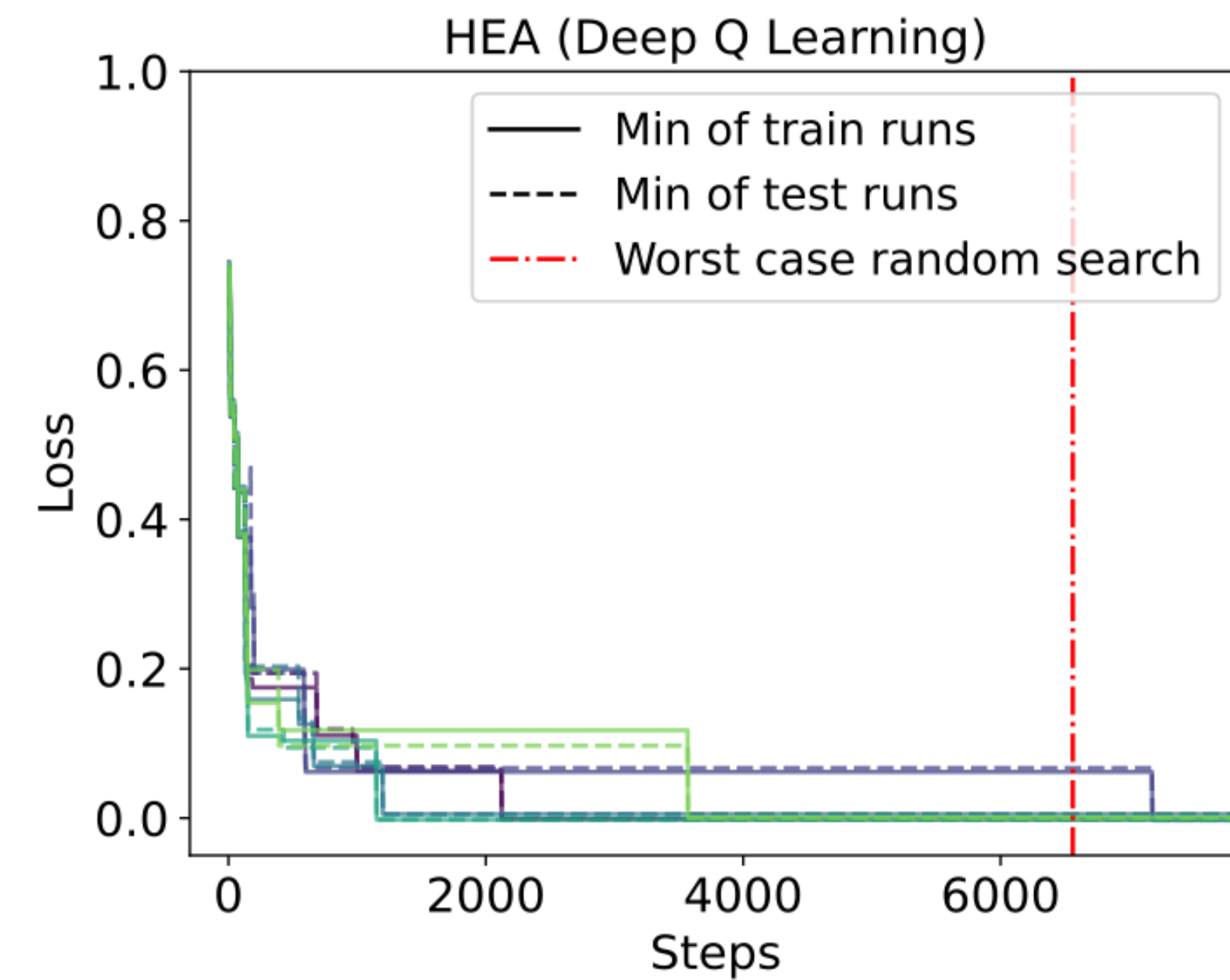
(Becomes more interesting when considering feedback)



Ordered Layers = $[[0], [5], [6], [1], [2, 7], [3, 8], [4, 9]]$

$$\text{DQN tries to find } \operatorname{argmax}_{\vec{\beta}} \langle R \rangle = \sum_{t=0}^n \gamma^t F_t$$

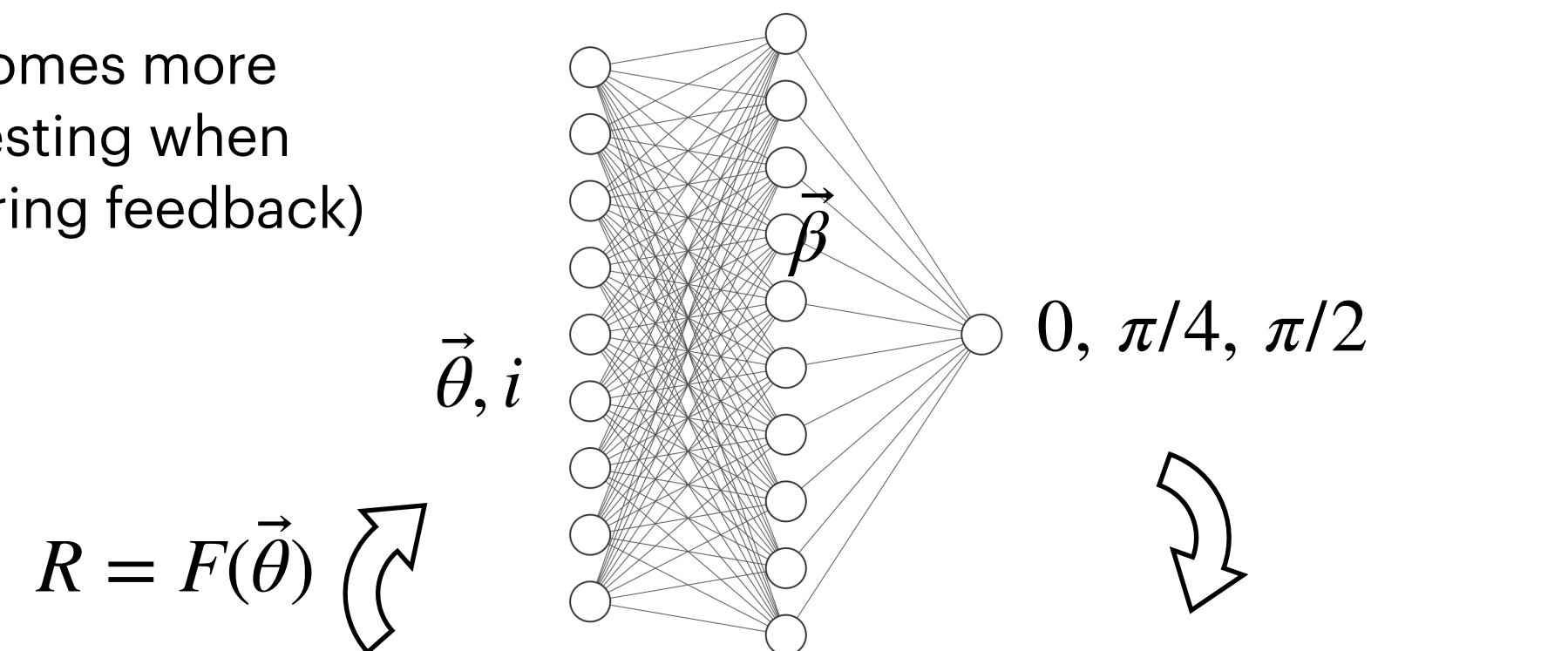
Main problem: It is too expensive to train (requires many samples)



Hardware efficient Ansatz

Deep Q-learning optimization

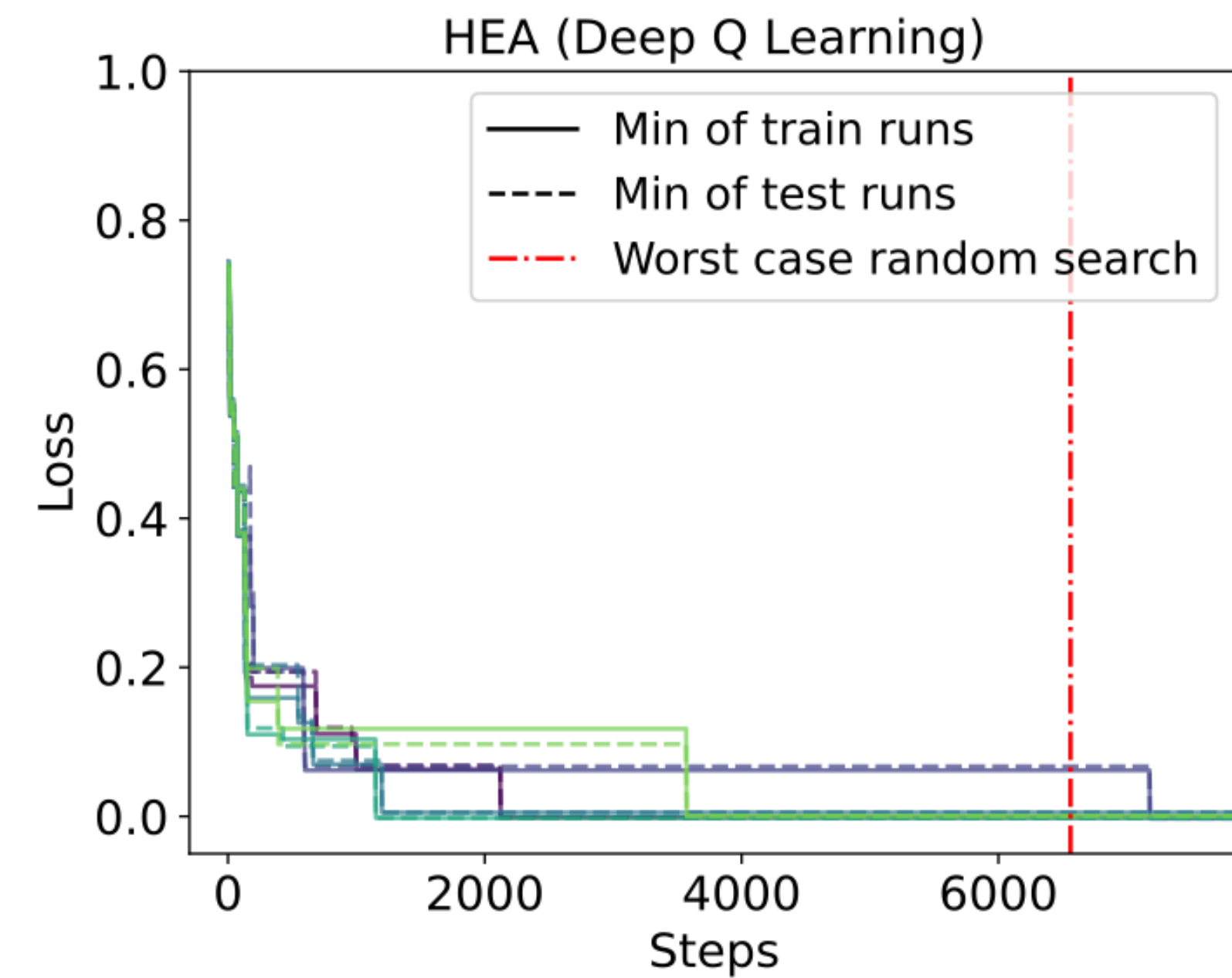
(Becomes more interesting when considering feedback)



Ordered Layers = $[[0], [5], [6], [1], [2, 7], [3, 8], [4, 9]]$

$$\text{DQN tries to find } \operatorname{argmax}_{\vec{\beta}} \langle R \rangle = \sum_{t=0}^n \gamma^t F_t$$

Main problem: It is too expensive to train (requires many samples)

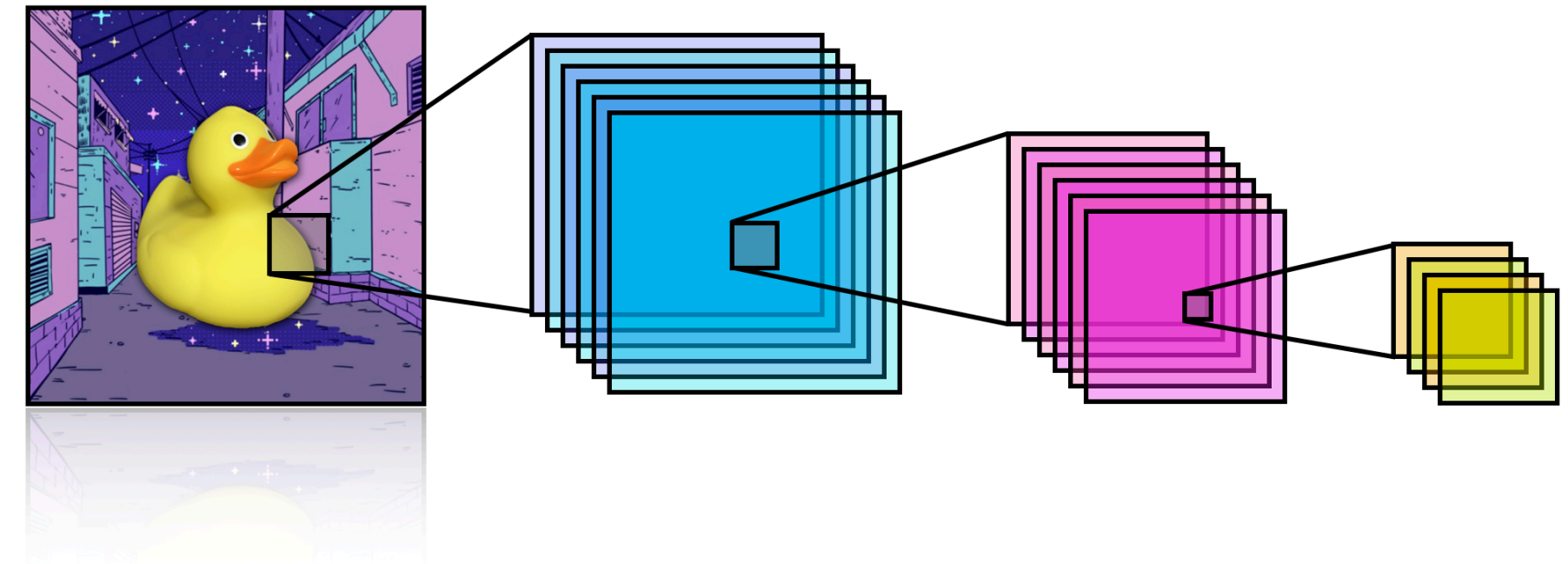
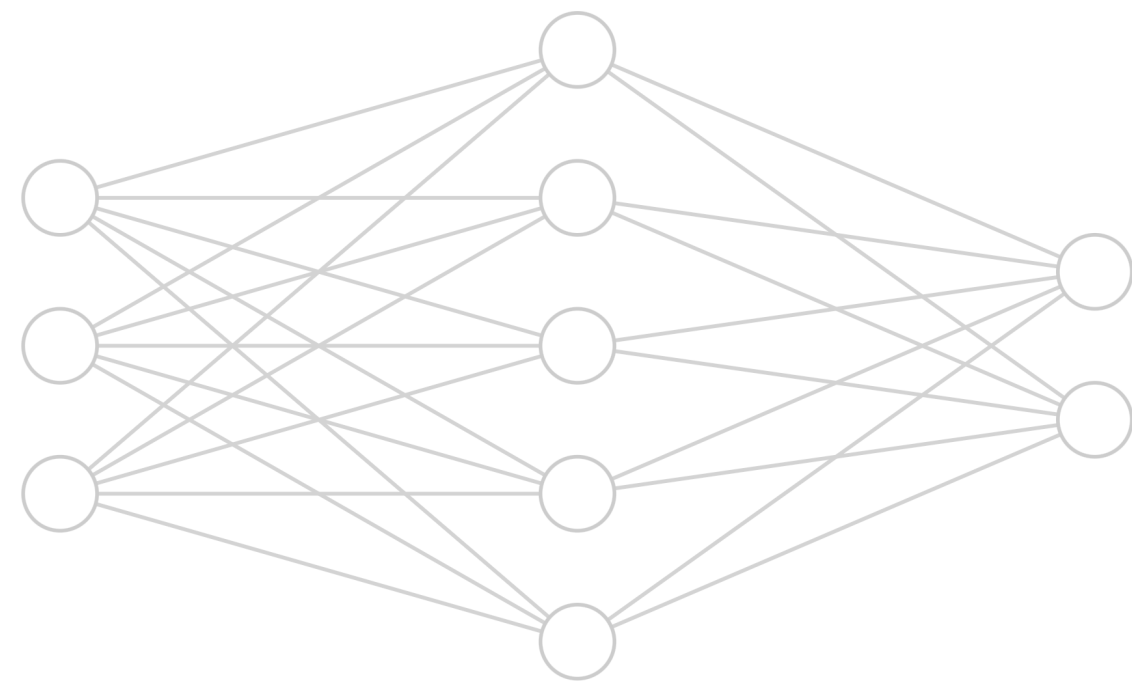


Solving problems with MB-QNNs

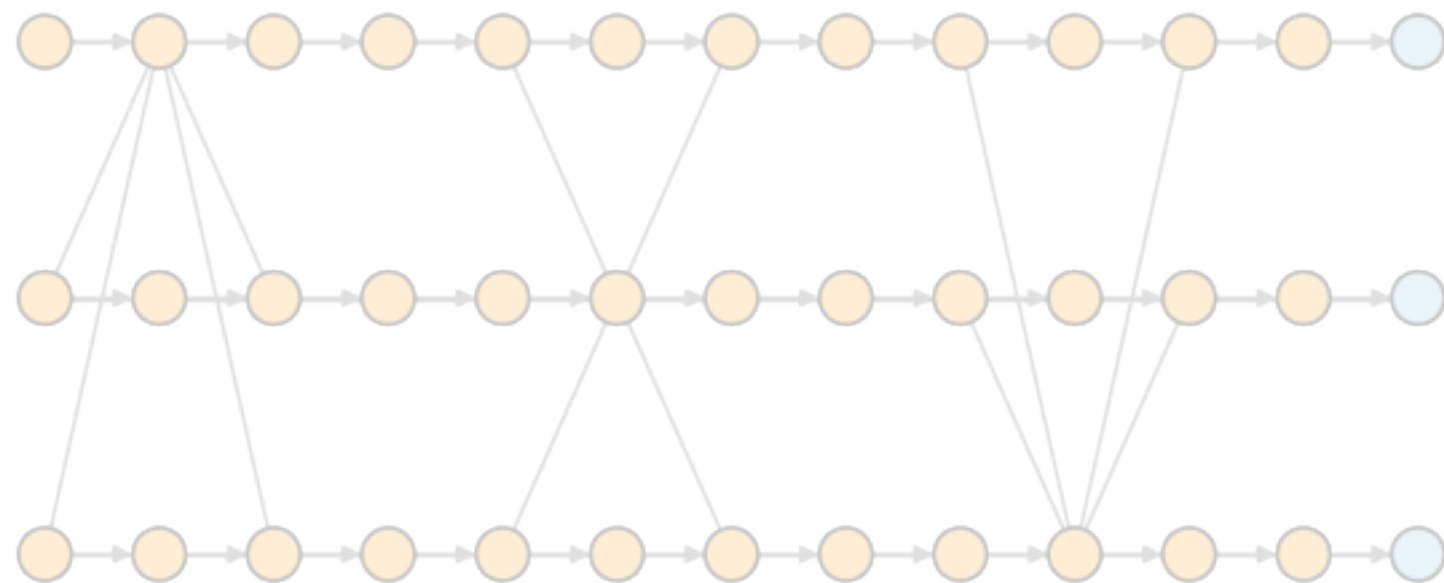
Two main messages

Custom Architectures

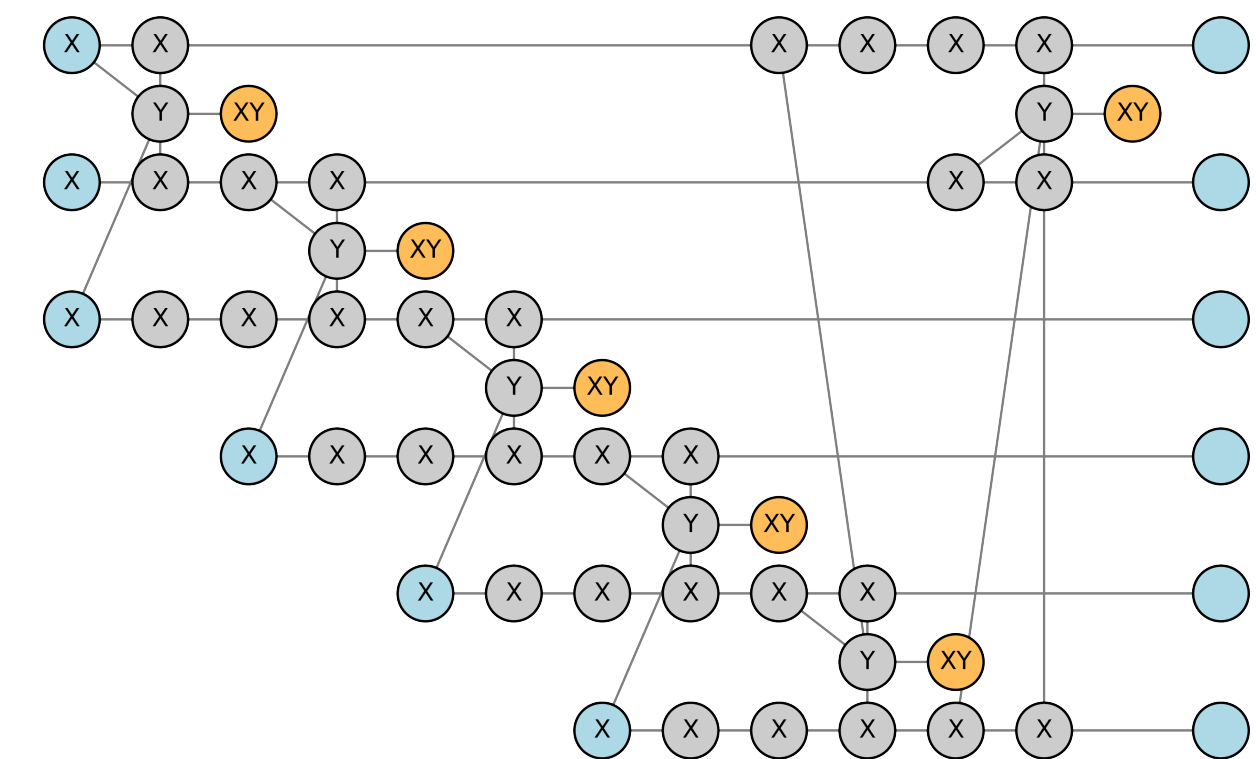
Classical



Quantum



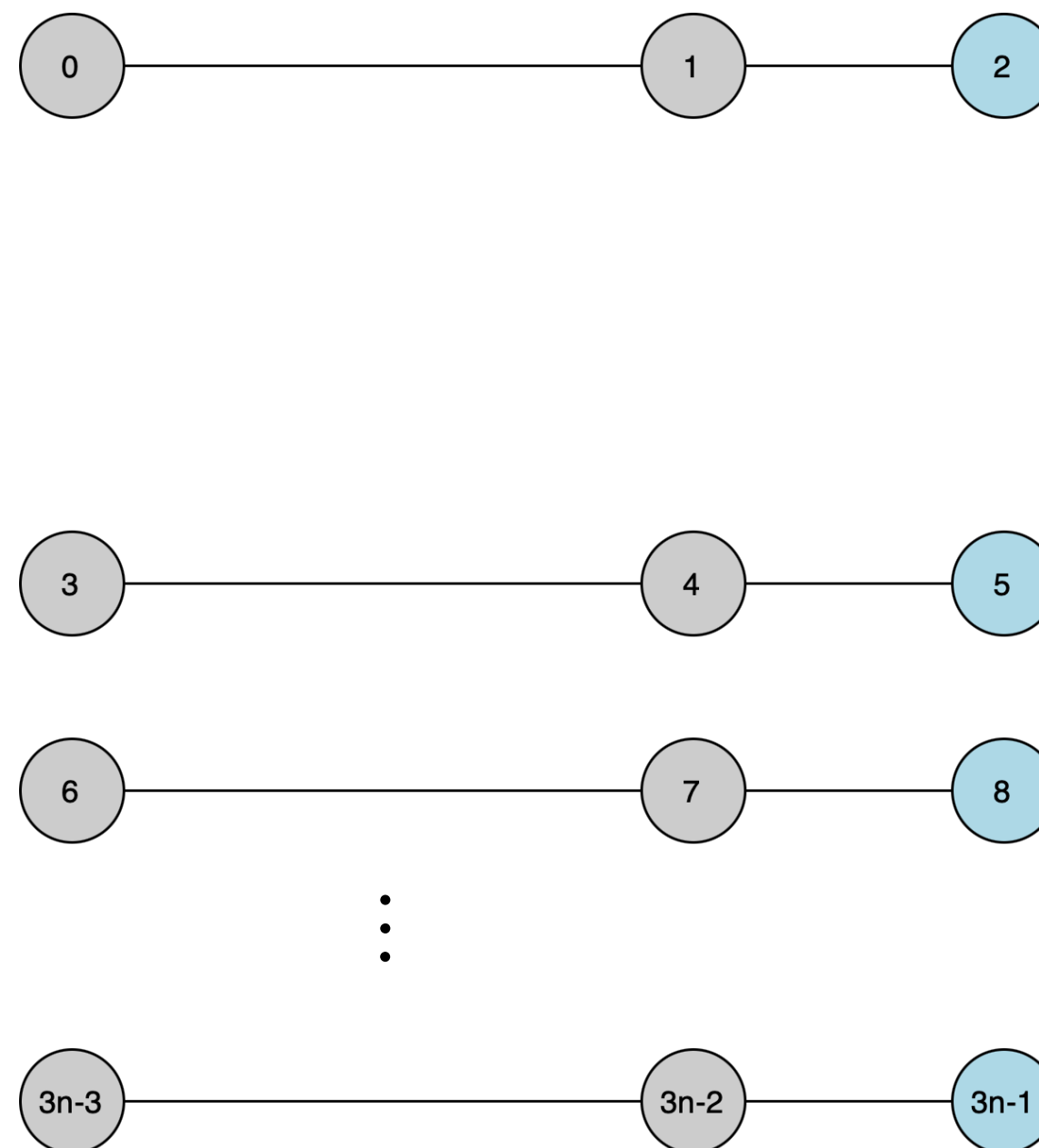
Translation invariant



Designing a custom ansatz

Ansatz  Gates

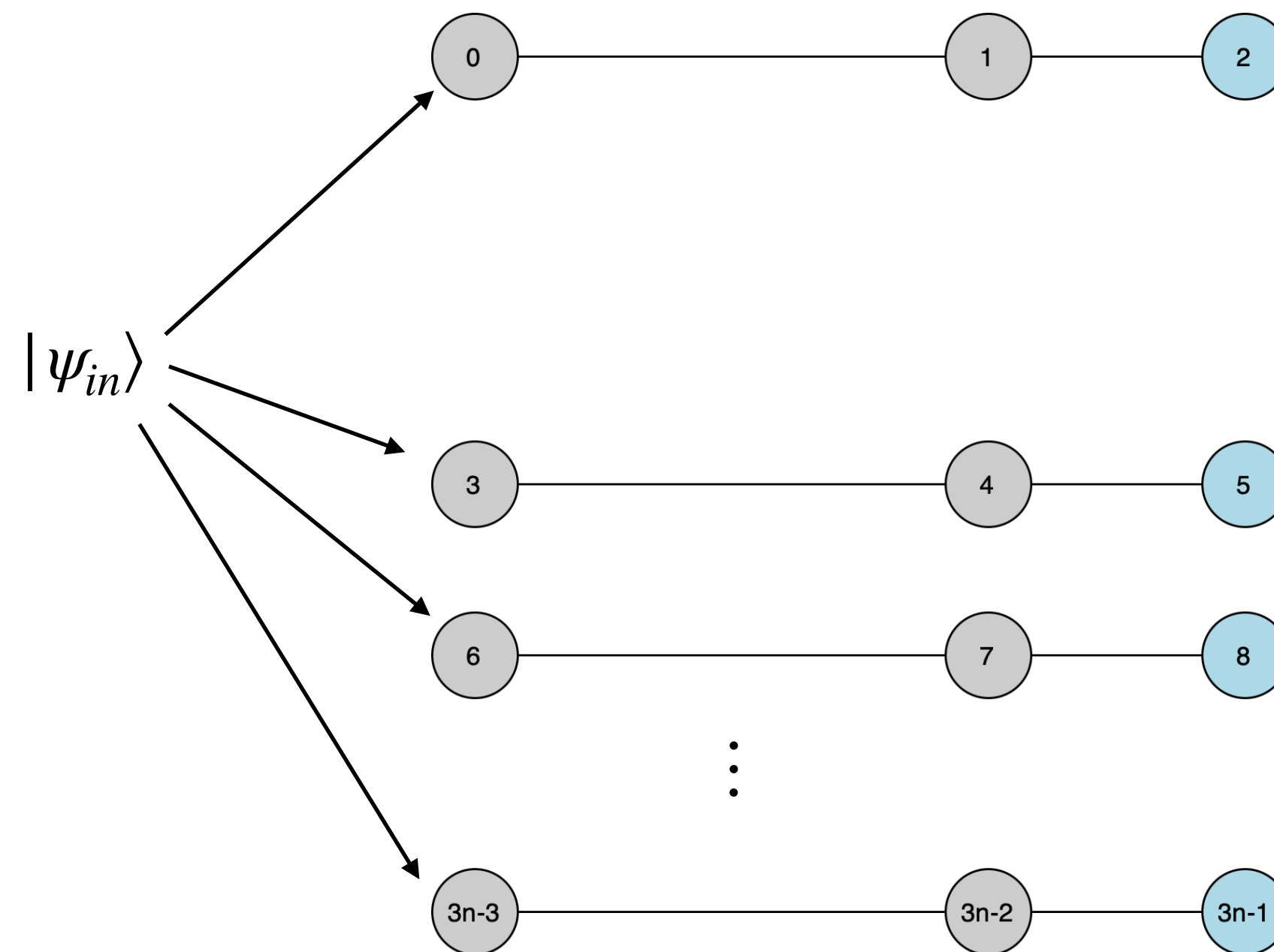
Given a gate $U(\theta) = e^{i\theta P}$ where $P = X \otimes Y \otimes I \cdots \otimes I \otimes Z$



Designing a custom ansatz

Ansatz  Gates

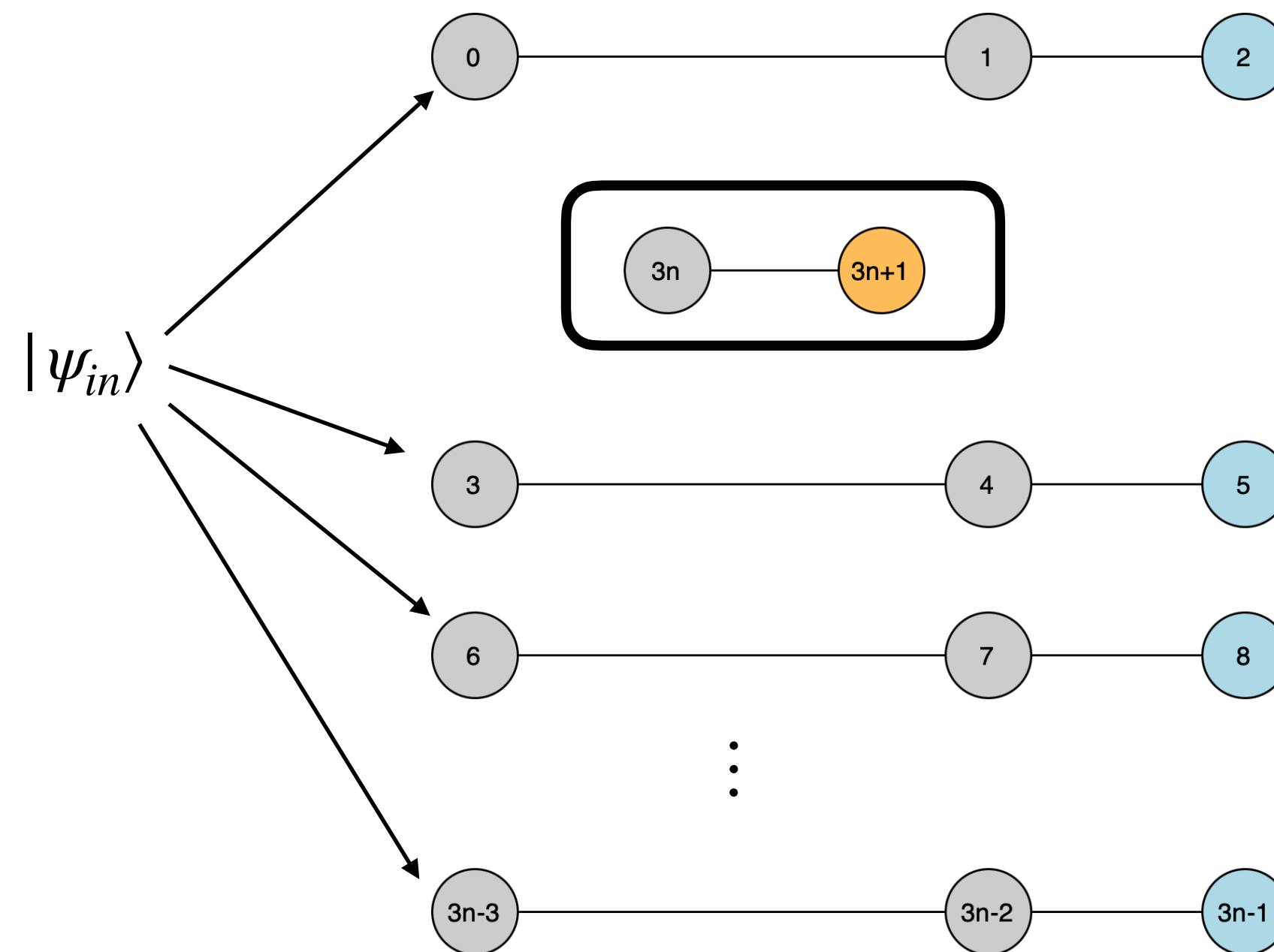
Given a gate $U(\theta) = e^{i\theta P}$ where $P = X \otimes Y \otimes I \cdots \otimes I \otimes Z$



Designing a custom ansatz

Ansatz  Gates

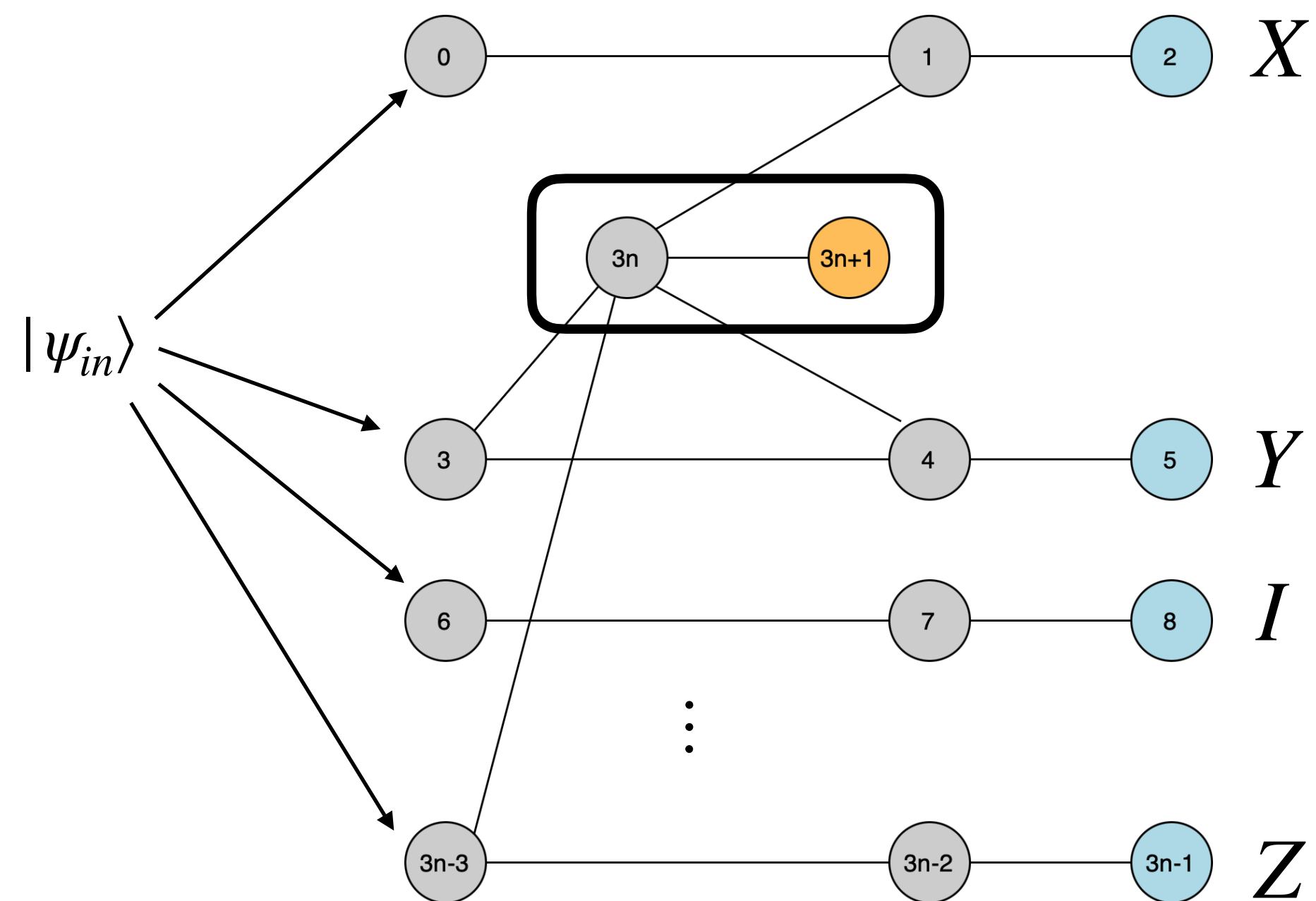
Given a gate $U(\theta) = e^{i\theta P}$ where $P = X \otimes Y \otimes I \cdots \otimes I \otimes Z$



Designing a custom ansatz

Ansatz  Gates

Given a gate $U(\theta) = e^{i\theta P}$ where $P = X \otimes Y \otimes I \cdots \otimes I \otimes Z$

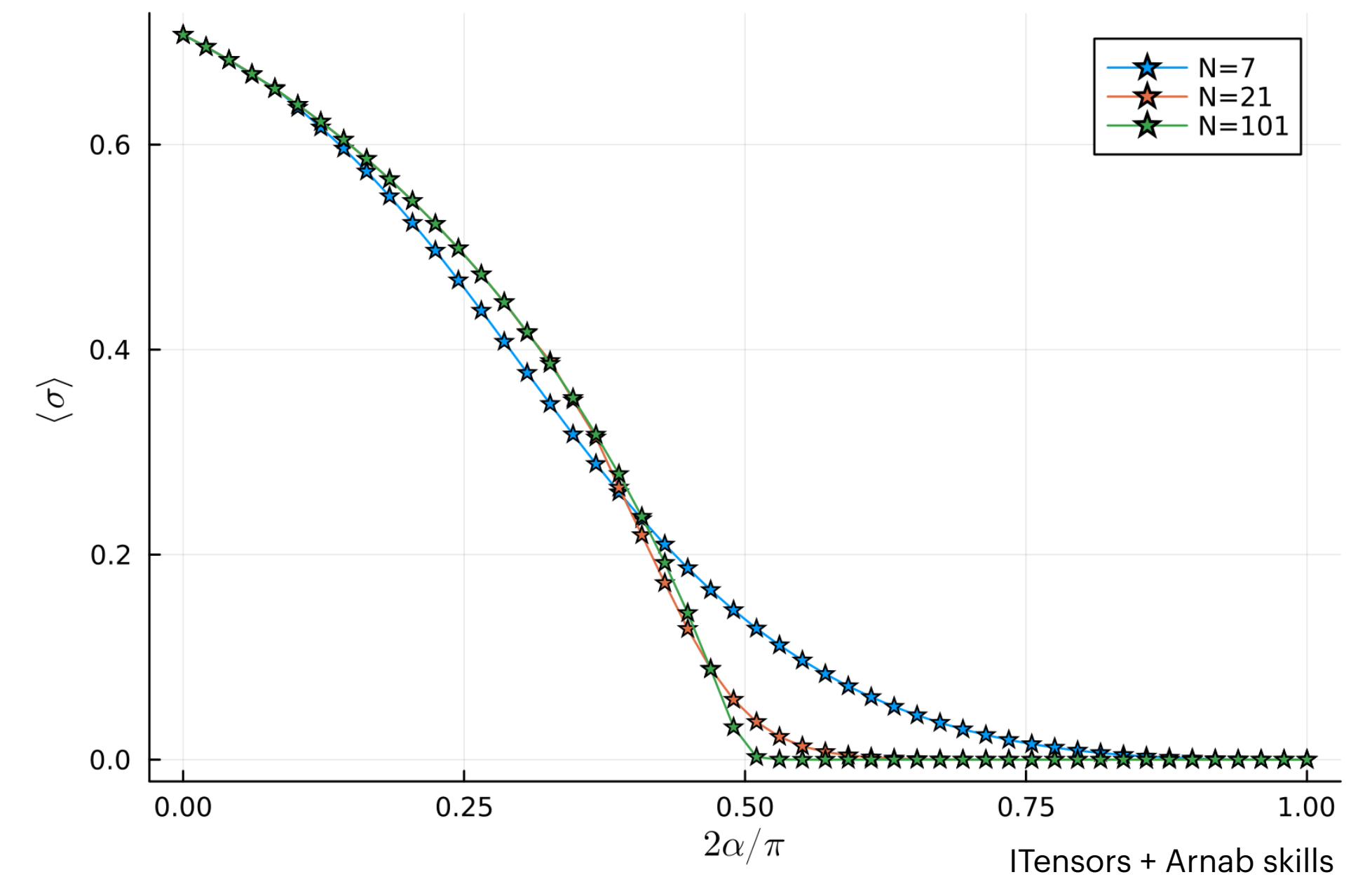


Computational phases of matter

In a ring 

The transverse field cluster Hamiltonian has two phases

$$H(\alpha) = -\cos \alpha \sum_{i=1}^N Z_{i-1} X_i Z_{i+1} - \sin \alpha \sum_{i=1}^N X_i$$



Computational phases of matter

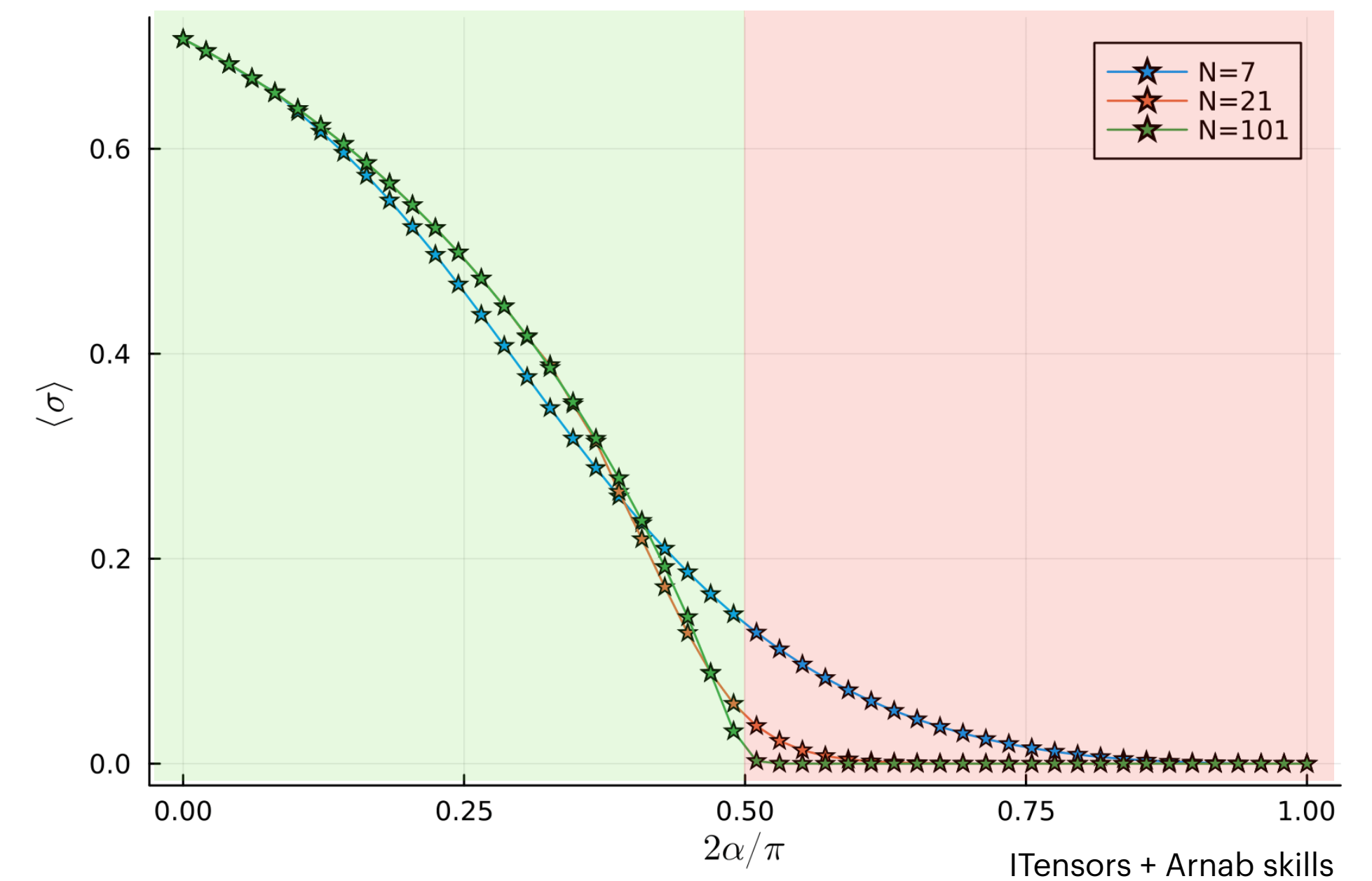
In a ring 

The transverse field cluster Hamiltonian has two phases

$$H(\alpha) = -\cos \alpha \sum_{i=1}^N Z_{i-1} X_i Z_{i+1} - \sin \alpha \sum_{i=1}^N X_i$$

The string order parameter measures computational power

$$\sigma = Z_0 \otimes X_1 \otimes I \otimes \cdots \otimes X_{m-1} \otimes Z_m$$



First-order approximation

Construct a custom Ansatz

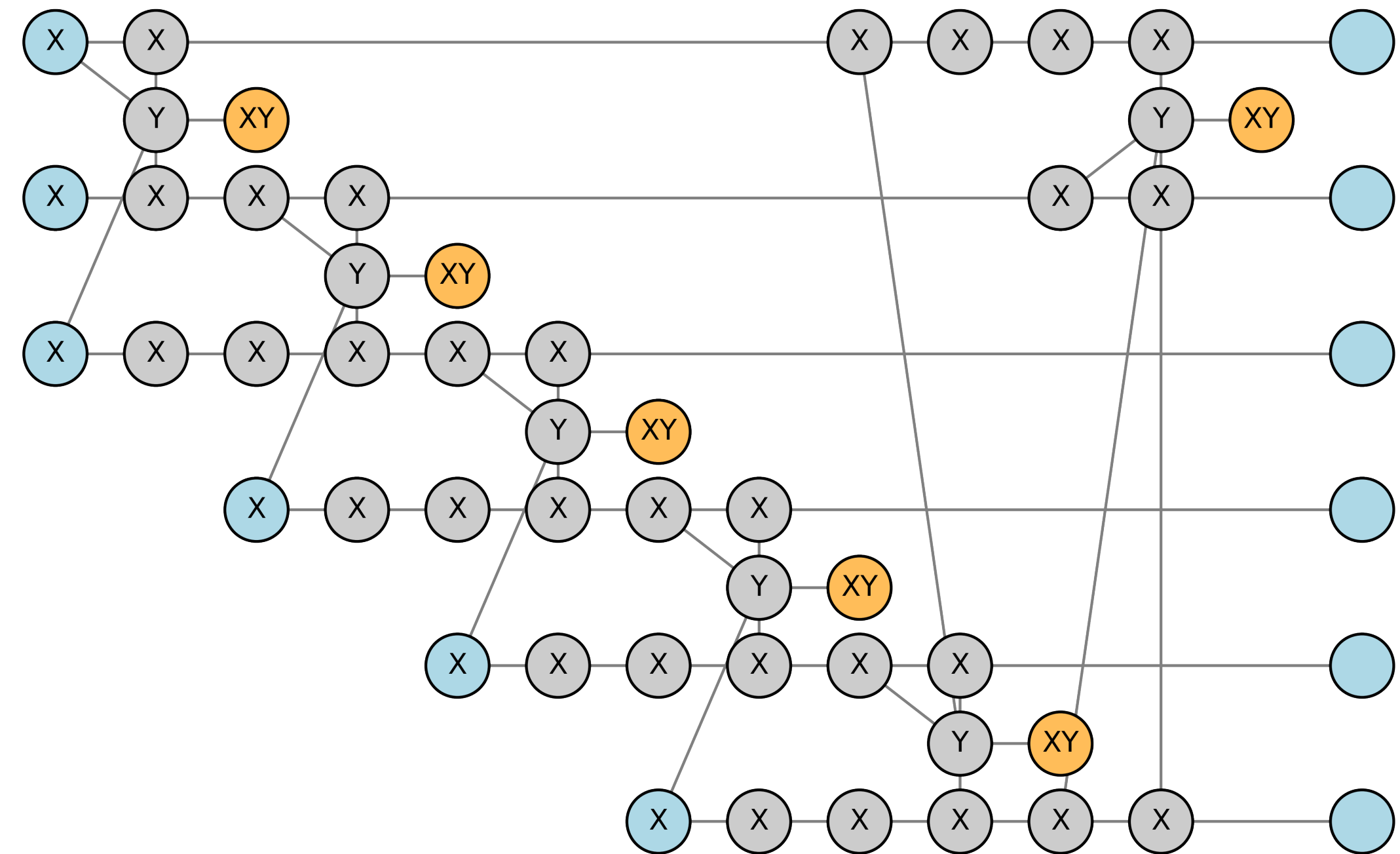
With first-order perturbation theory

$$|\psi(\phi)\rangle = \bigotimes_j \left[\cos(\phi) \mathbf{1}_j + \sin(\phi) X_j \right] |\mathcal{C}\rangle$$

we can implement the non-unitary gate as

$$\begin{aligned} e^{\theta X_j} |\mathcal{C}\rangle &= e^{\theta X_j} Z_j X_{j+1} Z_{j+2} |\mathcal{C}\rangle \\ &= e^{i\theta Y_j X_{j+1} Z_{j+2}} |\mathcal{C}\rangle \end{aligned}$$

and repeat over all j .



First-order approximation

Construct a custom Ansatz

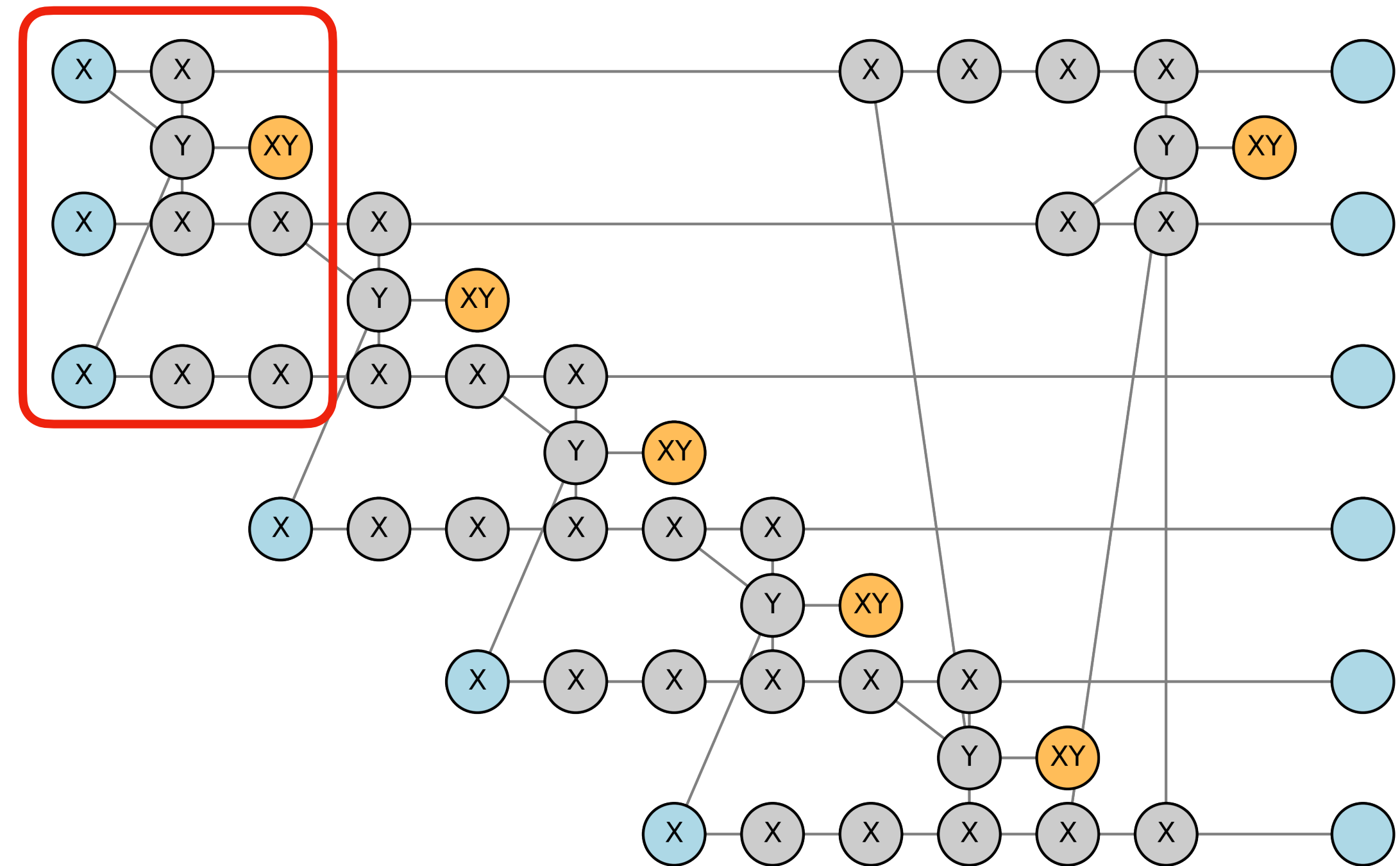
With first-order perturbation theory

$$|\psi(\phi)\rangle = \bigotimes_j \left[\cos(\phi) \mathbf{1}_j + \sin(\phi) X_j \right] |\mathcal{C}\rangle$$

we can implement the non-unitary gate as

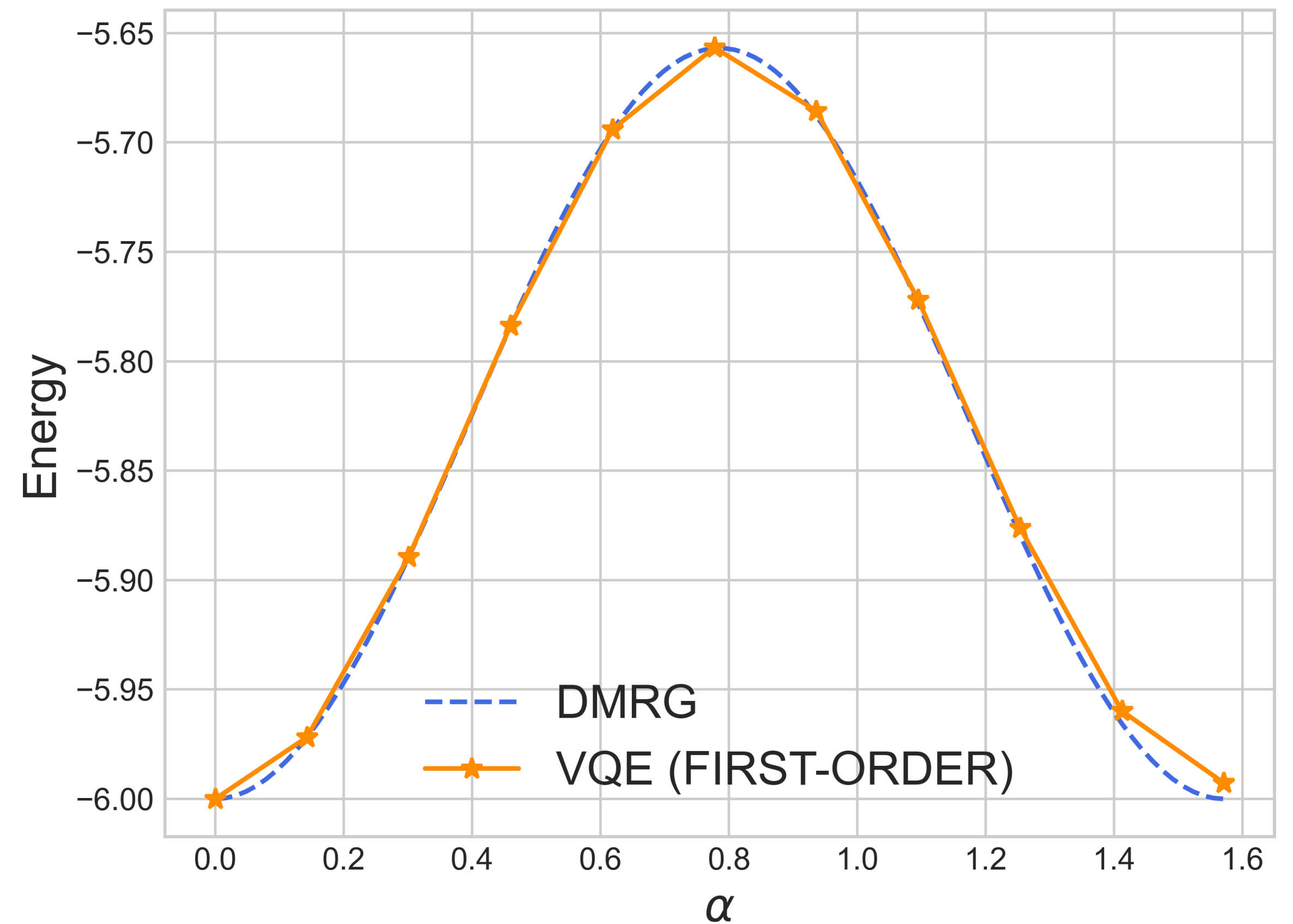
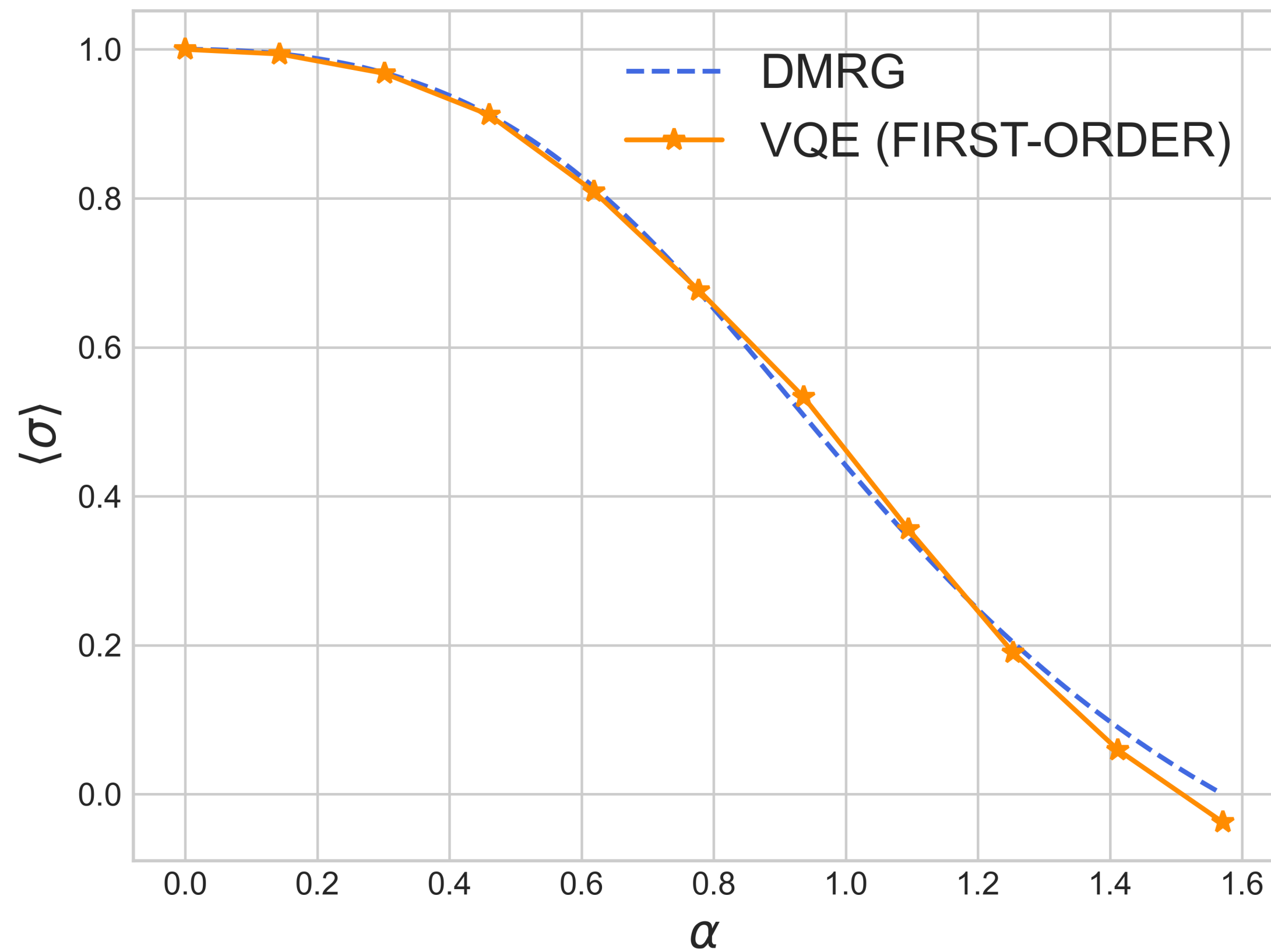
$$\begin{aligned} e^{\theta X_j} |\mathcal{C}\rangle &= e^{\theta X_j} Z_j X_{j+1} Z_{j+2} |\mathcal{C}\rangle \\ &= e^{i\theta Y_j X_{j+1} Z_{j+2}} |\mathcal{C}\rangle \end{aligned}$$

and repeat over all j .



Minimize the energy

Comparing DMRG and MB-VQE



**How do we make sure these
models are useful?**

Have rigorous guarantees for speedups

Not “well understood” yet

Negative!

Perspective | [Open access](#) | Published: 25 August 2025

Does provable absence of barren plateaus imply classical simulability?

[M. Cerezo](#) , [Martin Larocca](#), [Diego García-Martín](#), [N. L. Diaz](#), [Paolo Braccia](#), [Enrico Fontana](#), [Manuel S. Rudolph](#), [Pablo Bermejo](#), [Aroosa Ijaz](#), [Supanut Thanasilp](#), [Eric R. Anschuetz](#) & [Zoë Holmes](#)

[Nature Communications](#) **16**, 1013–1025 (2025)

6925 Accesses | **10** Citations

Abstract

A large amount of effort has been devoted to understanding the barren plateau phenomenon. In this paper, we consider a different perspective and ask a question that has not been asked before: *Does the absence of barren plateaus imply that a quantum circuit can be simulated classically?* We collect evidence that suggests that the answer is negative. We consider models whose loss landscapes are barren plateaus, and we provide that one can construct a quantum circuit that achieves high accuracy and from finite sampling statistics.

PRX QUANTUM **4**, 010309 (2023)

Limitations of Variational Quantum Algorithms: A Quantum Optimal Transport Approach

Giacomo De Palma,^{1,*} Milad Marvian,^{2,†} Cambyse Rouzé,^{3,‡} and Daniel Stilck França^{4,5,§}

¹*Department of Mathematics, University of Bologna, Bologna 40126, Italy*

²*Department of Electrical & Computer Engineering and Center for Quantum Information and Control, University of New Mexico, Albuquerque, New Mexico 87131, USA*

³*Zentrum Mathematik, Technische Universität München, Garching 85748, Germany*

⁴*QMATH, Department of Mathematical Sciences University of Copenhagen, Universitetsparken 5, Copenhagen 2100, Denmark*

⁵*ENS Lyon, Inria, F-69342, Lyon Cedex 07, France*

 (Received 3 May 2022; revised 26 July 2022; accepted 5 December 2022; published 23 January 2023)

The impressive progress in quantum hardware of the last years has raised the interest of the quantum computing community in harvesting the computational power of such devices. However, in the absence of error correction, these devices can only reliably implement very shallow circuits or comparatively deeper circuits at the expense of a nontrivial density of errors. In this work, we obtain extremely tight limitation bounds for standard noisy intermediate-scale quantum proposals in both the noisy and noiseless regimes, with or without error-mitigation tools. The bounds limit the performance of both circuit model algorithms, such as the quantum approximate optimization algorithm, and also continuous-time algorithms, such as quantum annealing. In the noisy regime with local depolarizing noise p , we prove that at depths $L = \mathcal{O}(p^{-1})$ it is exponentially unlikely that the outcome of a noisy quantum circuit outperforms efficient classical algorithms for combinatorial optimization problems like max-cut. Although

Positive!

Article | Published: 12 July 2021

A rigorous and robust quantum speed-up in supervised machine learning

[Yunchao Liu](#), [Srinivasan Arunachalam](#) & [Kristan Temme](#) 

[Nature Physics](#) **17**, 1013–1025 (2021)

18k Accesses | **449** Citations

Abstract

Recently, several quantum speed-ups over classical algorithms have been demonstrated. However, these results are heuristic or assume that the quantum circuit can be simulated classically. We resolve this challenge by introducing *generative quantum advantage*: the ability of quantum computers to learn and generate desired outputs substantially better than classical computers. We resolve this challenge by introducing families of generative quantum models that are hard to simulate classically, are efficiently trainable, exhibit no barren plateaus or proliferating local minima, and can learn to generate distributions beyond the reach of classical computers. Using a 68-qubit superconducting quantum processor, we demonstrate these capabilities in two scenarios: learning classically intractable probability distributions and learning quantum circuits for accelerated physical simulation. Our results establish that both learning and sampling can be performed efficiently in the beyond-classical regime, opening new possibilities for quantum-enhanced generative models with provable advantage.

Generative quantum advantage for classical and quantum problems

Hsin-Yuan Huang,^{1,2,*} Michael Broughton,¹ Norhan Eassa,^{1,3} Hartmut Neven,¹ Ryan Babbush,¹ and Jarrod R. McClean^{1,†}

¹*Google Quantum AI, 340 Main Street, Venice, CA 90291, USA*

²*Department of Physics, California Institute of Technology, Pasadena 91125, CA, USA*

³*Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47906, USA*
(Dated: September 12, 2025)

Recent breakthroughs in generative machine learning, powered by massive computational resources, have demonstrated unprecedented human-like capabilities. While beyond-classical quantum experiments can generate samples from classically intractable distributions, their complexity has thwarted all efforts toward efficient learning. This challenge has hindered demonstrations of *generative quantum advantage*: the ability of quantum computers to learn and generate desired outputs substantially better than classical computers. We resolve this challenge by introducing families of generative quantum models that are hard to simulate classically, are efficiently trainable, exhibit no barren plateaus or proliferating local minima, and can learn to generate distributions beyond the reach of classical computers. Using a 68-qubit superconducting quantum processor, we demonstrate these capabilities in two scenarios: learning classically intractable probability distributions and learning quantum circuits for accelerated physical simulation. Our results establish that both learning and sampling can be performed efficiently in the beyond-classical regime, opening new possibilities for quantum-enhanced generative models with provable advantage.

I. Introduction

Fueled by the overwhelming growth of computational power, recent progress in machine learning has continued to redefine the boundaries of what was once deemed impossible. Much of this progress has focused on large language models (LLMs) (1) and diffusion models (2), which represent specific types of generative machine learning models (3). The promise of quantum computers to dramatically expand our computational capabilities for certain problems has naturally led to speculation about how quantum technology will reshape the landscape of machine learning.

When is MB-QML better than QML?

Exploit long-range correlations!

- MBQC and regular circuits can be transformed with **Poly(n)** overhead
- MBQC can allow **shallower circuits** than the circuit model in some cases e.g., Toric code $O(1)$ vs $O(\log n)$
- Going beyond variational algorithms e.g., using **Gibbs** states
- Applying it in quantum chemistry?

Thermal States as Universal Resources for Quantum Computation with Always-on Interactions

Ying Li¹, Daniel E. Browne^{1,2}, Leong Chuan Kwek^{1,3}, Robert Raussendorf⁴, and Tzu-Chieh Wei⁴

¹Centre for Quantum Technologies, National University of Singapore, 2 Science Drive 3, Singapore

²Department of Physics and Astronomy, University College London, Gower Street, London WC1E 6BT, United Kingdom

³National Institute of Education and Institute of Advanced Studies, Nanyang Technological University, 1 Nanyang Walk, Singapore and

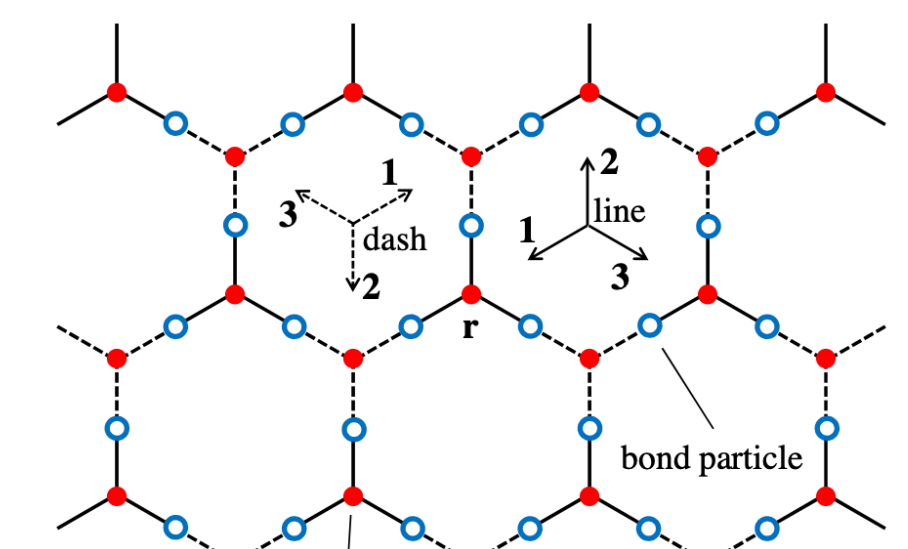
⁴Department of Physics and Astronomy, University of British Columbia, Vancouver, British Columbia V6T 1Z1, Canada

(Dated: August 6, 2018)

Measurement-based quantum computation utilizes an initial entangled resource state and proceeds with subsequent single-qubit measurements. It is implicitly assumed that the interactions between qubits can be switched off so that the dynamics of the measured qubits do not affect the computation. By proposing a model spin Hamiltonian, we demonstrate that measurement-based quantum computation can be achieved on a thermal state with always-on interactions. Moreover, computational errors induced by thermal fluctuations can be corrected and thus the computation can be executed fault-tolerantly if the temperature is below a threshold value.

PACS numbers: 03.67.Lx, 03.67.Pp, 75.10.Jm

Introduction.- A quantum computer can solve certain problems considered hard for a classical computer with an exponential speedup [1]. Standard quantum computing uses unitary evolution as a basic mechanism for information processing. Another paradigm is measurement-based quantum computing (MBQC), in which one processes quantum information by single-particle operations and measurements only, on a nontrivial entangled state [2]. Such entangled states serve as universal resources of MBQC [3]. The first identified universal resource was the cluster state. It can be obtained as the



Thank you!

