Reinforcement Learning

—

Lecture 8: Quantum Reinforcement Learning

Nico Meyer (Guest Lecture)

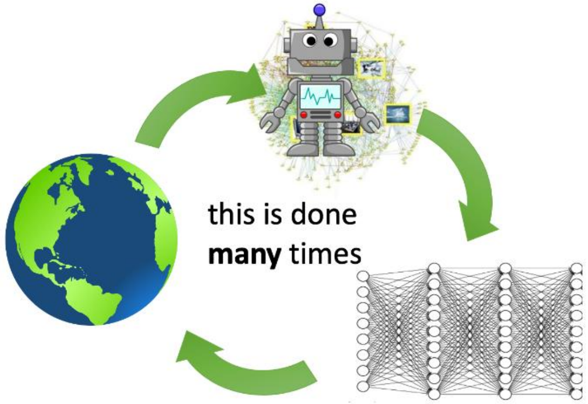
Fraunhofer IIS

Fraunhofer-Institut für Integrierte Schaltungen IIS

What today will be about

Combination of two of the latest research fields: RL and Quantum Computing

(Deep) Reinforcement Learning



https://www.wevolver.com/article/datadriven.deep.reinforcement.learning

Quantum Computing

https://www.ibm.com/blogs/digitale-perspektive/2021/06/quantum-opening-in-ehningen/

Quantum Reinforcement Learning

Enhance RL algorithms with concepts from quantum computing – a novel (and potentially more powerful) computing paradigm based on quantum mechanics

Quantum Computing – Sounds like Science Fiction?!

It is not!

- Quantum computers are reality since a few years
  - You can experiment with some of them yourself: https://www.ibm.com/quantum
  - A lot of research is going on to develop soft- and hardware
- Promise to revolutionize computing in different fields
  - Cryptography
  - Simulation
  - Optimization
  - Machine Learning
  - …
- Still in very early stages of development
  - but just take a look at classical computers in e.g. the 70's

IBM Quantum System One in Ehningen, Germany

https://de.newsroom.ibm.com/ibm-dach-special-coverage-Fraunhofer-IBM

Outline

General Concept
- Quantum Computing
- Quantum Reinforcement Learning

Recap
- Policy-based Reinforcement Learning
- Policy Gradients

Concrete Algorithm
- Quantum Policy Gradients
- Extension #1: Classical Post-Processing
- Extension #2: Quantum Natural Gradients

Quantum Computing

Let's start

https://www.youtube.com/watch?v=JhHMJCUmq28

Quantum Computing

A bit more formal: Qubits and Superpositions

A qubit is the pendant to the classical bit. The two basis states are:

$$|0\rangle \mapsto \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |1\rangle \mapsto \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

It can be in a superposition of those states:

$$\alpha\,|0\rangle + \beta\,|1\rangle \mapsto \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \text{ with } \alpha, \beta \in \mathbb{C} \text{ and } |\alpha|^2 + |\beta|^2 = 1$$

A n-qubits system gives us access to an Hilbert space of dimension 2^n:

$$|\psi\rangle = c_0\,|0\ldots00\rangle + c_0\,|0\ldots01\rangle + \cdots + c_{2^n-1}\,|1\cdots11\rangle,\ c_i \in \mathbb{C},\ \sum_{i=0}^{2^n-1} |c_i|^2 = 1$$

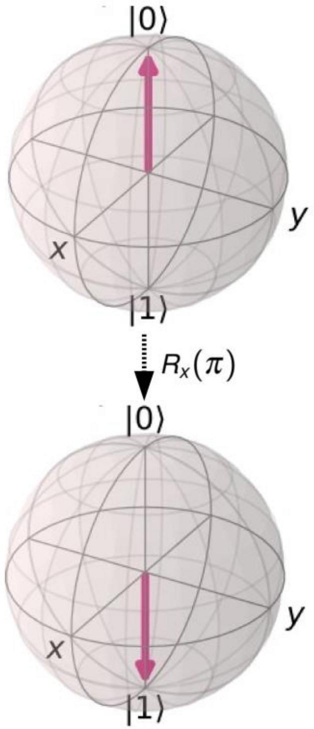
Simplified!

$\hookrightarrow$ A quantum computer allows us to process 2^n complex numbers at once!

Measurement on system $|\psi\rangle$

- destroys the superposition, $|\psi\rangle$ collapses to basis state
- measurement result is inherently random
- $|\psi\rangle$ collapses with probability $|c_i|^2$ to basis state $|i\rangle$

$\hookrightarrow$ it requires a 'clever' combination of superposition, entanglement, and interference to make use of quantum computing



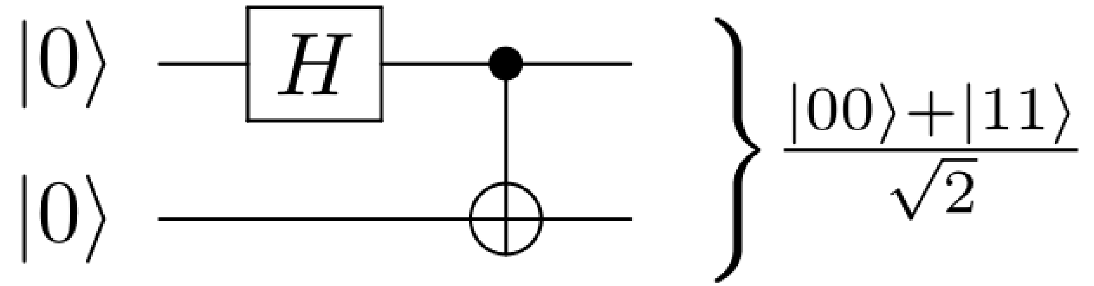
State manipulation:

Reversible (unitary operators), e.g. flip amplitudes with $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

Fraunhofer IIS

Quantum Computing

Making it more visual: Quantum Circuits



Creating maximally entangeled Bell state

Step 1: Hadamard gate on first qubit

$$(H \otimes I)\,|00\rangle \rightarrow \left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \otimes \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right)\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

Step 2: Controlled-NOT gate in both qubits

$$\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\1\\0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\0\\1\end{bmatrix} \rightarrow \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right)$$

Measurement:

$$\frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right)$$

$$\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2} \qquad \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$$

$$|00\rangle \qquad |11\rangle$$

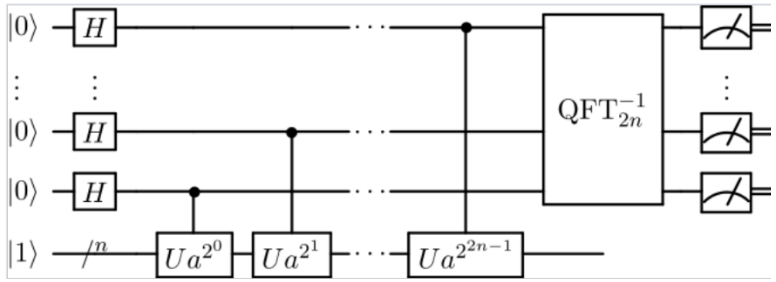
Fraunhofer IIS

Quantum Computing

Promising prospects, but currently still quite limited

Provable **Quantum advantage** (potentially) exists in several areas:

- Material Science
- Optimization
- <u>Machine Learning</u>
- Cryptography
- Simulation
- $\cdots$



Subroutine of Shor's algorithm

Currently only **"Noisy Intermediate-Scale Quantum"** (NISQ) devices

- Quantum systems are very instable ($\hookrightarrow$ error correction)
- Largest quantum device only has 433 qubits ($\hookrightarrow$ enigneering overhead)

Quantum Reinforcement Learning

Variational Quantum Circuits (VQCs)

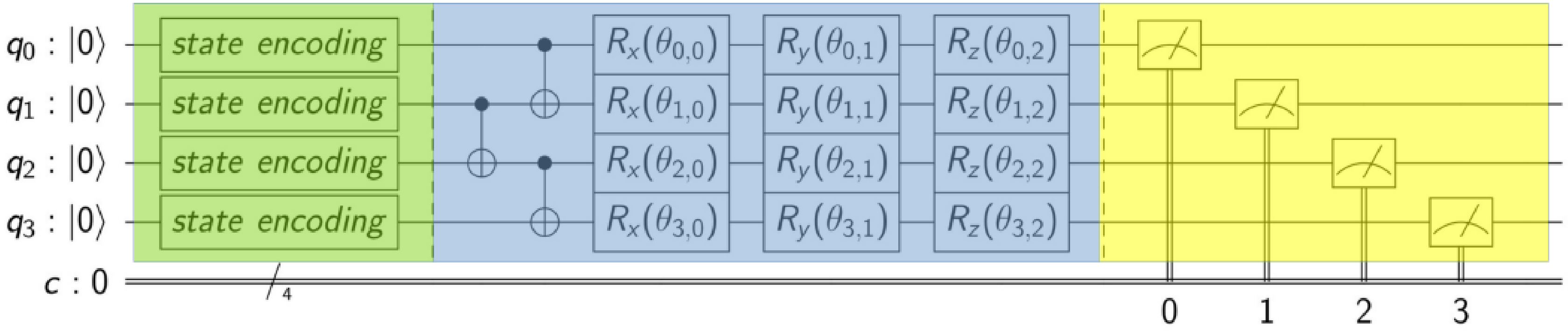


Figure: VQ-DQL with four qubits and one learnable block

Encode state into VQC

- depends on specific task
- (in general) not learnable

Measurement

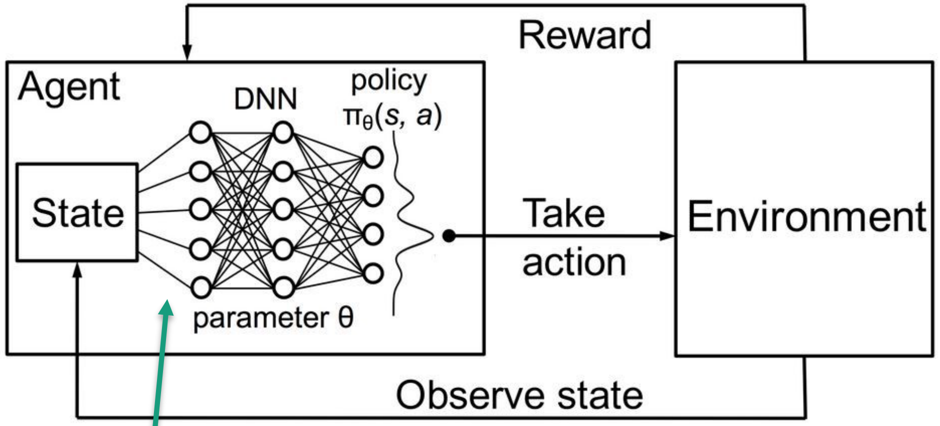
- used for action selection

Learnable buliding block

- target of iterative optimization
- CNOT gates for entanglement
- learnable single-qubit rotations
- could be repeated several times

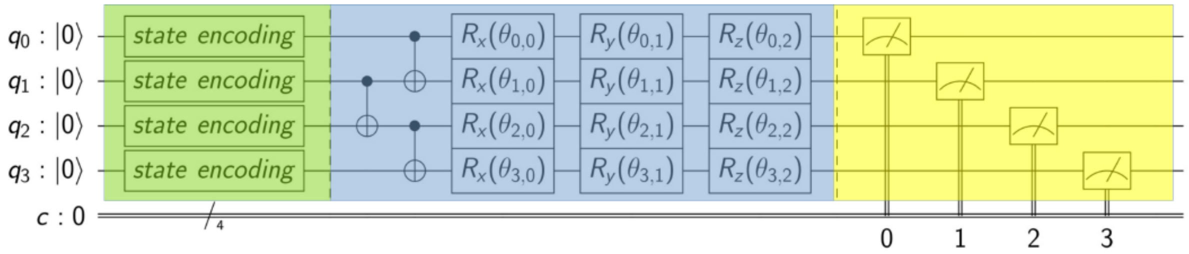
Quantum Reinforcement Learning

Variational Quantum Circuits (VQCs) – How and Why?



https://towardsdatascience.com/self-learning-ai-agents-iv-stochastic-policy-gradients-b53f088fce20

Replace Function Approximator!



- VQCs are **potentially more 'powerful'** function approximators then DNNs:
  - Access to exponentially large Hilbert space
  - Reduction in parameter complexity
  - Reduction in sampling complexity (less interaction with environment)
  - Allow to work with 'quantum data'
- Allow to **make use of NISQ hardware**
  - Believed to somewhat 'intrinsically' learn and adapt to noise

➢ More on concrete working principles later!

Fraunhofer IIS

Quantum Reinforcement Learning

Outlook: A broad field with many different ideas

today

	Classical Compute Resources	NISQ Resources	Universal, Fault-Tolerant and Error-Corrected Quantum Processing Unit	
	classical	Degree of quantum-classical hybridization		quantum
Algorithm Class	Quantum-inspired RL Algorithms	VQC-based Function Approximation	RL Algorithms with Quantum Subroutines	Full Quantum RL
Subtype	Amplitude-Amplification-based Action Selection	VQC-based Critic	Quantum Policy Iteration (Cherrat et al.)	Quantum Policy Iteration (Wiedemann et al.)
		VQC-based Actor	Quantum Value Iteration	Oracularized Environments
		VQC-based Actor-Critic	Projective Simulation	Quantum Gradient Estimation
		VQC-based Multi-Agent RL	Boltzmann-Machines for Function Approximation	
		VQC-based Distributional RL		

N. Meyer et al., ***A Survey on Quantum Reinforcement Learning****,*
arXiv:2211.03464 (2022).

Fraunhofer IIS

Outline

General Concept

- Quantum Computing
- Quantum Reinforcement Learning

Recap

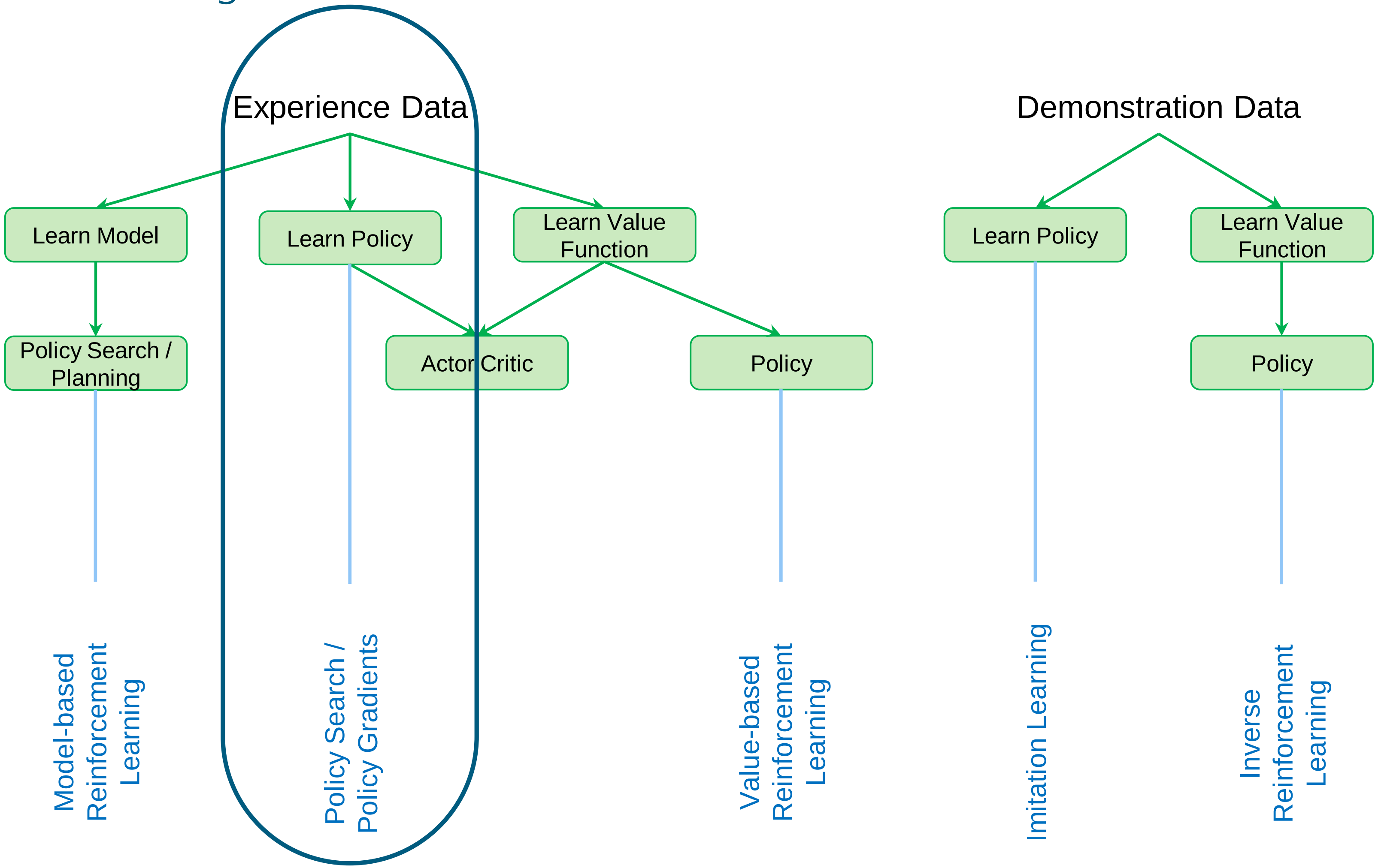
- Policy-based Reinforcement Learning
- Policy Gradients

Concrete Algorithm

- Quantum Policy Gradients
- Extension #1: Classical Post-Processing
- Extension #2: Quantum Natural Gradients

Policy-based Reinforcement Learning

Recap: Where it fits in the grand scheme



Policy-based Reinforcement Learning

Recap: Where it differs from value-function approximation

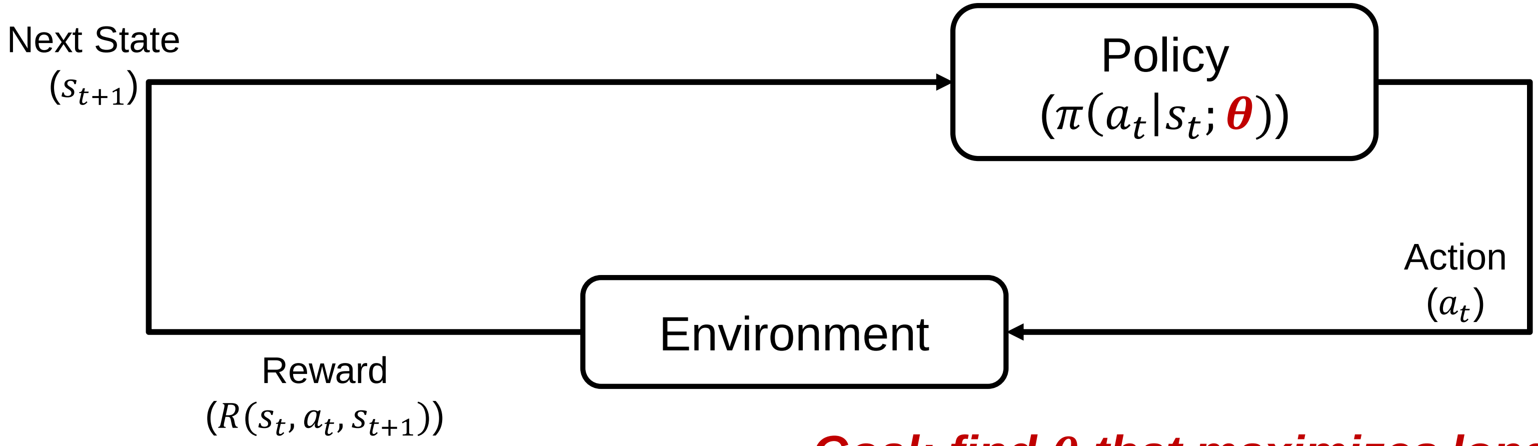
- Previously we approximated parametric value functions:

$$v_w(s) \approx v_\pi(s)$$
$$q_w(s,a) \approx q_\pi(s,a)$$

- A policy can be generated from these values
- **Instead: Directly parameterize the policy**

$$\pi_\theta\,(a|s) = p(a|s;\theta)$$

- **We still focus on model-free reinforcement learning**



Goal: find θ that maximizes long term reward

Policy-based Reinforcement Learning

Recap: Advantages and Disadvantages

Advantages:

- **Good convergence properties**
- Easily extended to high-dimensional or continuous state and action spaces
- Can learn ***stochastic* policies**
- Sometimes policies are simple while values and models are complex
  - e.g., rich domain, but optimal is always to go left

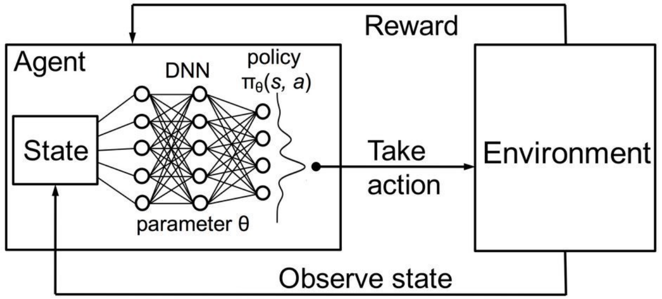
Disadvantages:

- Susceptible to local optima (especially with non-linear FA)
- Obtained knowledge is specific, does not always generalize well
- Ignores a lot of information in the data (when used in isolation)

Policy-based Reinforcement Learning

Recap: Stochastic policies

We have seen deterministic policies like this:



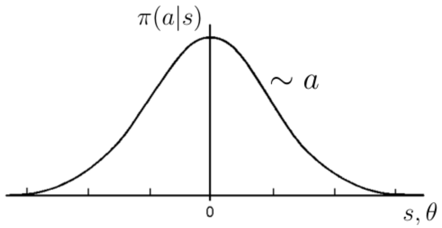
State gives $Q(s, a; w)$ and we selected $\pi(a|s)$ by $\text{argmax}_a\, Q(s, a; w)$

https://towardsdatascience.com/self-learning-ai-agents-iv-stochastic-policy-gradients-b53f088fce20

Instead, stochastic policies do something like this:

$$\pi(a|s) = \mathbb{P}[a|s; \theta]$$

(policy is represented as a probability distribution)



- ➢ optimal policy might be stochastic

Policy-based Reinforcement Learning

Recap: Smooth policy improvement

- Learn directly a policy without calculating value functions in between
- Why?
  - **Greedy updates**

$$\theta_{n+1} = \underset{\theta}{\operatorname{argmax}} E_{\pi_\theta}\{Q^\pi(s,a)\}$$

$$Q^\pi \xrightarrow[\text{Change}]{\text{Small}} \pi \xrightarrow[\text{Change}]{\text{Large}} Q^\pi \xrightarrow[\text{Change}]{\text{Large}} \pi \xrightarrow[\text{Change}]{\text{Large}}$$

Potentially unstable learning process with large policy "jumps"

  - **Smooth updates**

$$\theta_{n+1} = \theta_n + \boxed{\alpha_n \nabla G_{\theta_n}}$$

$$Q^\pi \xrightarrow[\text{Change}]{\text{Small}} \pi \xrightarrow[\text{Change}]{\text{Small}} Q^\pi \xrightarrow[\text{Change}]{\text{Small}} \pi \xrightarrow[\text{Change}]{\text{Small}}$$

Reminder:

$$\mathrm{G} = r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 r_3 + \cdots = \sum_{t=0}^{\infty} \gamma^t r_t$$

Stable learning process with smooth policy improvement

Policy Gradients

Recap: Maximize expected reward…

- Assume a state-action sequence in a complete trajectory with T steps (with s_T being a terminal state):

$$\tau = (s_0, a_0, \dots, s_{T-1}, a_{T-1}, s_T)$$

- As usual:
  - $R(s_t, a_t)$ is the reward received after observing s_t and performing action a_t
  - $G(\tau) \coloneqq \sum_{t=0}^{T-1} \gamma^t R(s_t, a_t)$ is the (discounted) sum of rewards (return)

- Our goal is to maximize the expected reward:

$$\max_{\theta} \mathbb{E}_{\pi_\theta} G(\tau)$$

(where π_θ is a parameterized policy, e.g., a neural network)

Policy Gradients

Recap: … via gradient ascent

- But how do we maximize this?
  → Gradient Ascent! *Suppose* we know how to calculate the gradient w.r.t. the parameters:

$$\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta} G(\tau)$$

- Then we can update our parameters θ in the direction of the gradient:

$$\theta \leftarrow \theta + \alpha \underline{\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta} G(\tau)}$$

Policy Gradient

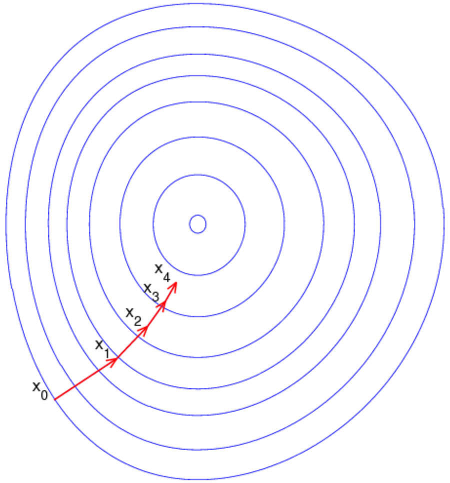
often in literature referred to as $\nabla_\theta J(\pi_\theta)$

- Search for a local maximum in $J(\theta)$ by ascending the gradient of the policy w.r.t. the parameters θ:

$$\Delta\theta = \underline{\alpha}\,\underline{\nabla_\theta J(\theta)}$$

step-size

$$\begin{pmatrix} \frac{\partial J(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_n} \end{pmatrix}$$



Fraunhofer IIS

Policy Gradients

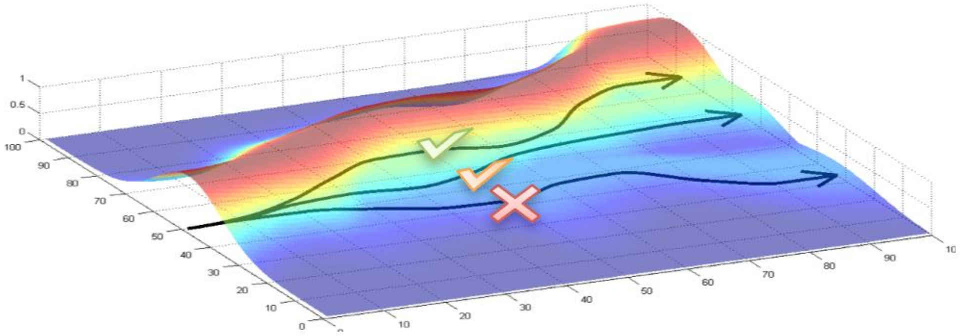
Recap: Reformulating the gradient

Let $P(\tau|\theta)$ be the probability of a trajectory τ under policy π_θ, then

$$
\begin{aligned}
\nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta} G(\tau) &= \nabla_\theta \sum_\tau P(\tau|\theta) G(\tau) \\
&= \cdots \\
&= \mathbb{E}_{\tau\sim\pi_\theta}(\nabla_\theta \log P(\tau|\theta) G(\tau)) \\
&= \mathbb{E}_{\tau\sim\pi_\theta}\left(\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) G(\tau)\right) \\
&\approx \frac{1}{L}\sum_\tau \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) G(\tau) \\
&\approx \frac{1}{L}\sum_\tau \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t}^{T} \gamma^{t'-t} R(s_{t'}, a_{t'})
\end{aligned}
$$

Intuition: The gradient tries to

- Increase probability of paths with positive G
- Decrease probability of paths with negative G



Pieter Abbeel. DeepRL Bootcamp 4A Policy Gradients.

Fraunhofer IIS

Policy Gradients

Recap: REINFORCE

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

Policy Gradients

Recap: Natural Policy Gradients

- First-order derivatives approximate the (parameter) surface to be flat
- But if the surface exhibits high curvature it gets dangerous
- ➢ Small changes in parameter space might lead to large changes in policy space!

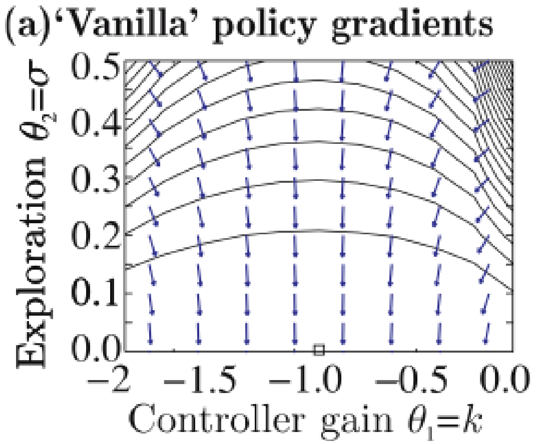
What we essentially do
(optimization perspective on 1st order gradient descent)

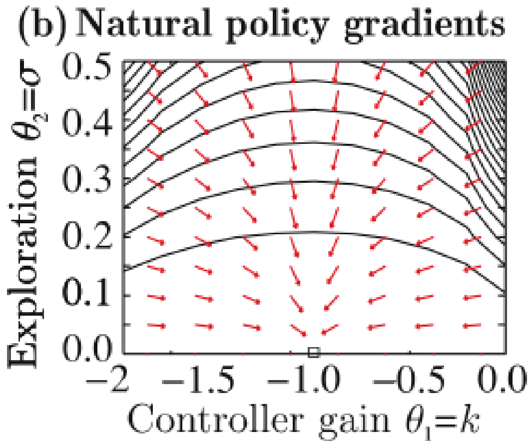
$$\theta' \leftarrow \arg\max_{\theta'} (\theta' - \theta)^T \nabla_\theta J(\theta), \text{ subject to } \|\theta' - \theta\|^2 \leq \epsilon$$

What we want to do
(incorporate 2nd order information)

$$\theta' \leftarrow \arg\max_{\theta'} (\theta' - \theta)^T \nabla_\theta J(\theta), \text{ subject to } \|\theta' - \theta\|_{\boldsymbol{F}}^2 \leq \epsilon$$

→ $\theta \leftarrow \theta + \alpha \mathbf{F}^{-1} \nabla_\theta J(\theta)$ with e.g. KL-divergence





Peters et al.: Natural Actor-Critic. 2018

Outline

General Concept

- Quantum Computing
- Quantum Reinforcement Learning

Recap

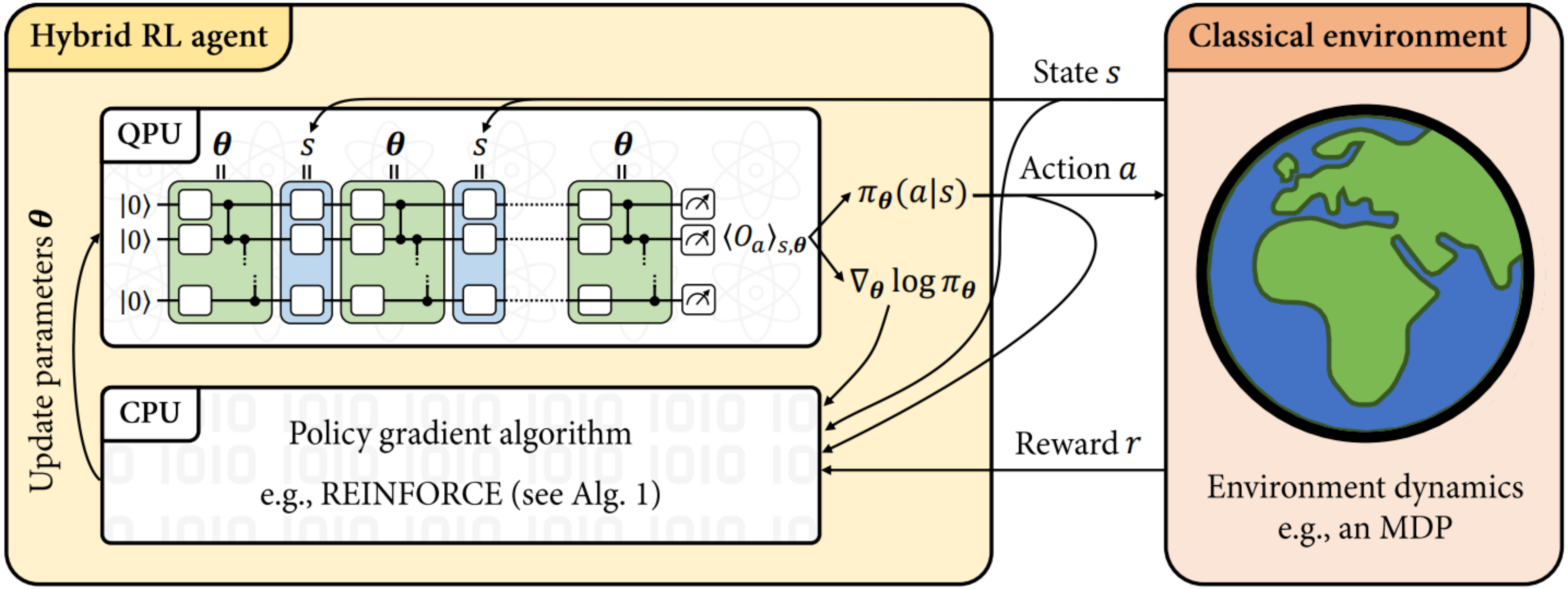
- Policy-based Reinforcement Learning
- Policy Gradients

Concrete Algorithm

- Quantum Policy Gradients
- Extension #1: Classical Post-Processing
- Extension #2: Quantum Natural Gradients

Quantum Policy Gradients

“Quantified” version of REINFORCE



S. Jerbi et al., ***Parameterized Quantum Policies for Reinforcement Learning***, *NeurIPS 34, 28362-28375 (2021).*

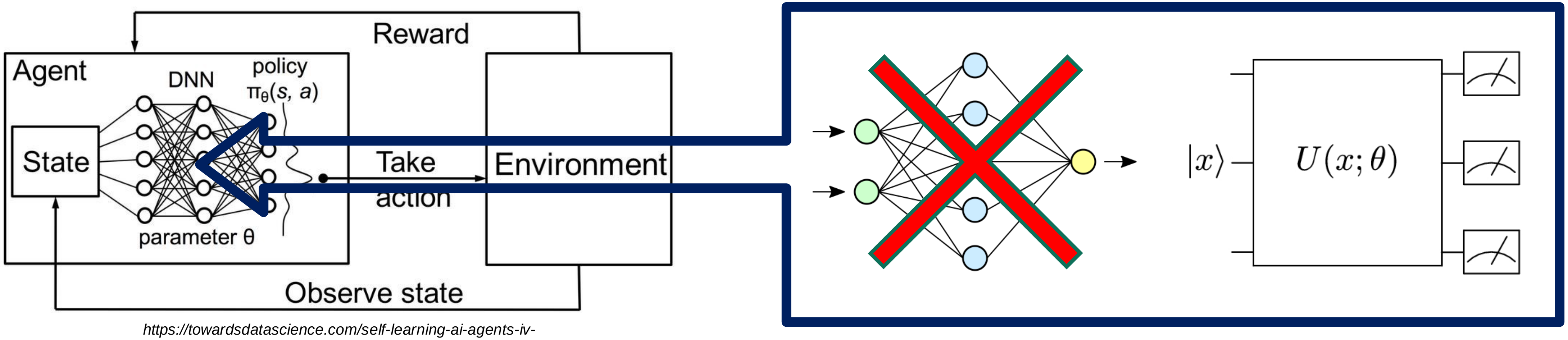
Quantum Policy Gradients

Replace function approximator

Parameterized policy: $\pi_{\boldsymbol{\theta}} : \mathcal{S} \times \mathcal{A} \mapsto [0,1]$

Update parameters via gradient ascent: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

Policy gradient theorem: $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{H-1} \nabla_{\boldsymbol{\theta}} \ln \pi_{\boldsymbol{\theta}}(a_t \mid \boldsymbol{s}_t) \cdot (G_t - b(\boldsymbol{s}_t)) \right]$



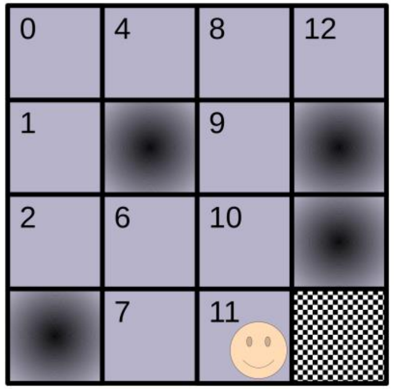
https://towardsdatascience.com/self-learning-ai-agents-iv-stochastic-policy-gradients-b53f088fce20

Quantum Policy Gradients

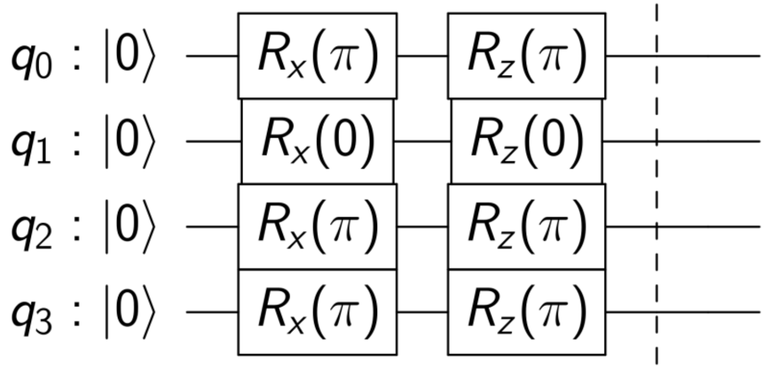
How to encode the environment state into the quantum circuit? [1/2]

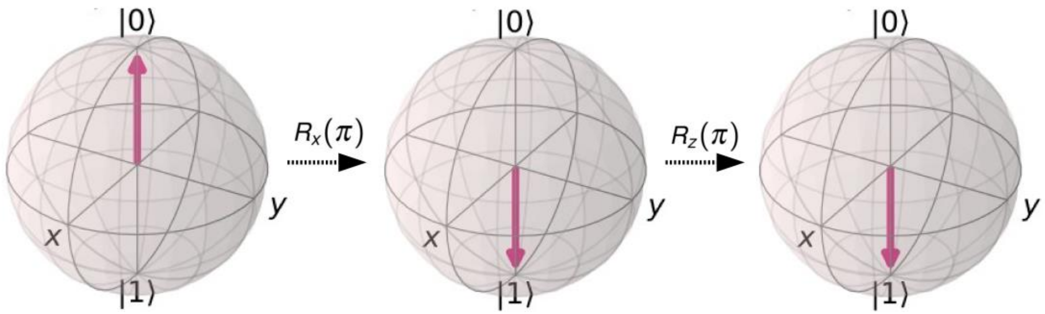
(optionally) removes global phase

computational encoding: encode binary representation of state



state 11 $\overset{binarize}{\rightsquigarrow}$ $(1,0,1,1)$ $\overset{times\ \pi}{\rightsquigarrow}$ $(\pi,0,\pi,\pi)$ $\overset{encode}{\rightsquigarrow}$ $|1011\rangle$





$$R_z(\pi)R_x(\pi)\,|0\rangle = (-i\sigma_z)(-i\sigma_x)\,|0\rangle$$
$$= (-i\sigma_z)(-i)\,|1\rangle = |1\rangle$$

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

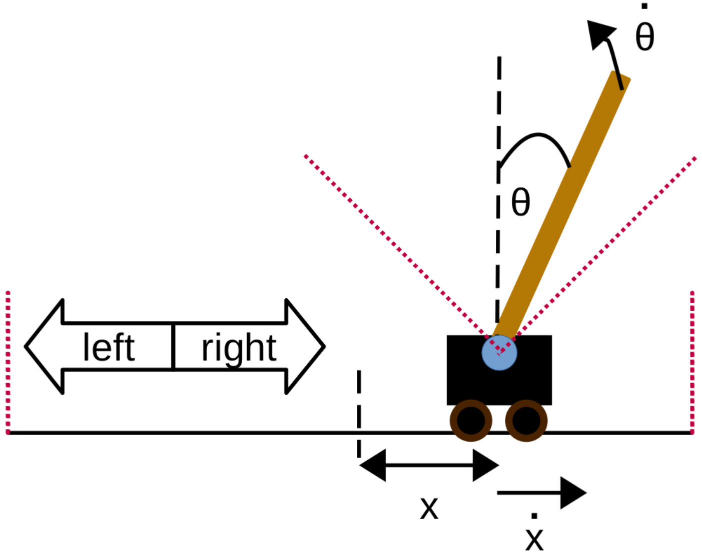
Chen et al., Variational Quantum Circuits for Deep Reinforcement Learning, IEEE Access 8, 141007-141024 (2020).

Quantum Policy Gradients

How to encode the environment state into the quantum circuit? [2/2]

(optional) scaling parameters

angle encoding: encode each (rescaled) state element as rotation on a qubit



4-dim state $(x, \dot{x}, \theta, \dot{\theta})$

rescale $\rightsquigarrow$ to range $[0, 2\pi)$

encode $\rightsquigarrow$ via rotation

$$U_{s,\boldsymbol{\lambda}_i} = \begin{array}{cc} R_y(\lambda_{i,0} \cdot s_0) & R_z(\lambda_{i,1} \cdot s_0) \\ R_y(\lambda_{i,2} \cdot s_1) & R_z(\lambda_{i,3} \cdot s_1) \\ \cdots & \cdots \\ R_y(\lambda_{i,2n-2} \cdot s_{n-1}) & R_z(\lambda_{i,2n-1} \cdot s_{n-1}) \end{array}$$

Lockwood and Si, Reinforcement Learning with Quantum Variational Circuits, AAAI Conference on AI and Interactive Digital Entertainment 16.1 (2020).

amplitude encoding: encode into amplitudes of $|\psi\rangle = \sum c_i |i\rangle$, $\sum |c_i|^2 = 1$

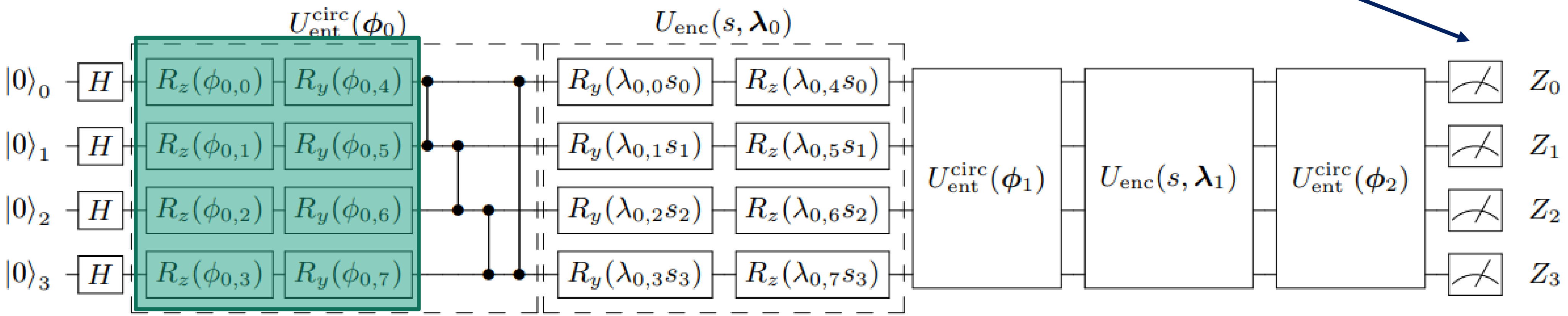
learned encoding: allows to incorporate more concrete objectives

Fraunhofer IIS

Quantum Policy Gradients

Learnable parameters and gradients

we measure the expectation value $\langle O \rangle$ of some observable O at the end



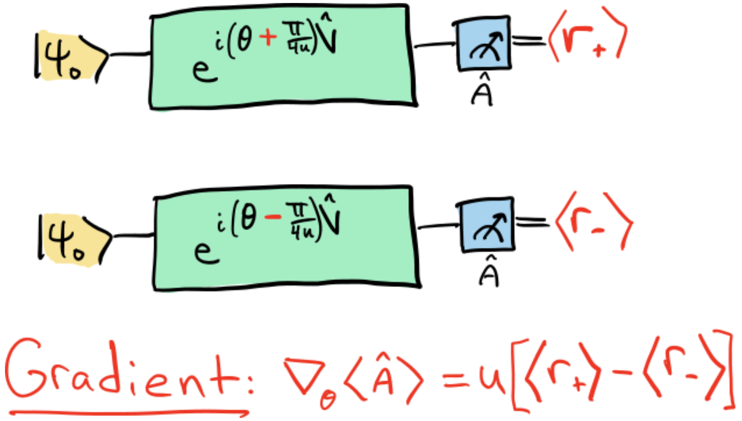
S. Jerbi et al., Parameterized Quantum Policies for Reinforcement Learning, NeurIPS 34, 28362-28375 (2021).

Computing gradients is not possible directly

$\hookrightarrow$ wave function collaps

but can use parameter-shift rule:

(special case of finite differences)



https://pennylane.ai/qml/demos/tutorial_stochastic_parameter_shift.html

Fraunhofer IIS

Quantum Policy Gradients

Measurements and policies [1/2]

quantum state prepared by encoding and variational layers

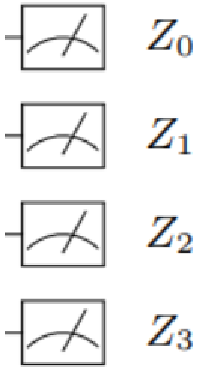
How do we obtain information from the quantum state $|\psi_{s,\theta}\rangle$?

Measure some observable (Hermitian operator) O

$\hookrightarrow$ observe one of the eigenvalues

$\hookrightarrow$ quantum system has collapsed to associated eigenstate

$\hookrightarrow$ multiple measurements retreive expectation value $\langle O \rangle$



Typical procedure: Measure in computational basis with $O = \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

$\hookrightarrow$ collapses to one of the basis states $|0 \cdots 00\rangle$, $|0 \cdots 01\rangle$, $\cdots$, $|1 \cdots 11\rangle$

Quantum Policy Gradients

Measurements and policies [2/2]

This can be used to define the policy:

action-associated projectors onto computational basis states

$$\pi_\theta(a|s) = \langle P_a \rangle_{s,\theta}$$

raw-VQC policy

$$\boxed{\nabla_\theta \log \pi_\theta(a|s) = \frac{\nabla_\theta \langle P_a \rangle_{s,\theta}}{\langle P_a \rangle_{s,\theta}}}$$

An alternative approach with arbitrary observables:

$$\pi_\theta(a|s) = \frac{e^{\beta \langle O_a \rangle_{s,\theta}}}{\sum_{a'} e^{\beta \langle O_a \rangle_{s,\theta}}}$$

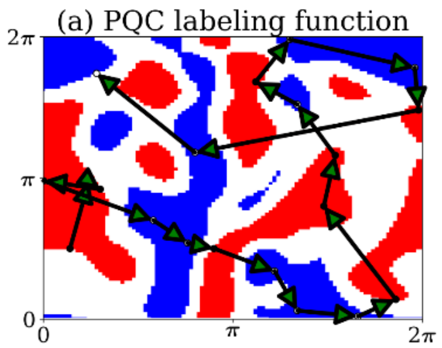
softmax-VQC policy

inverse-temperature parameter

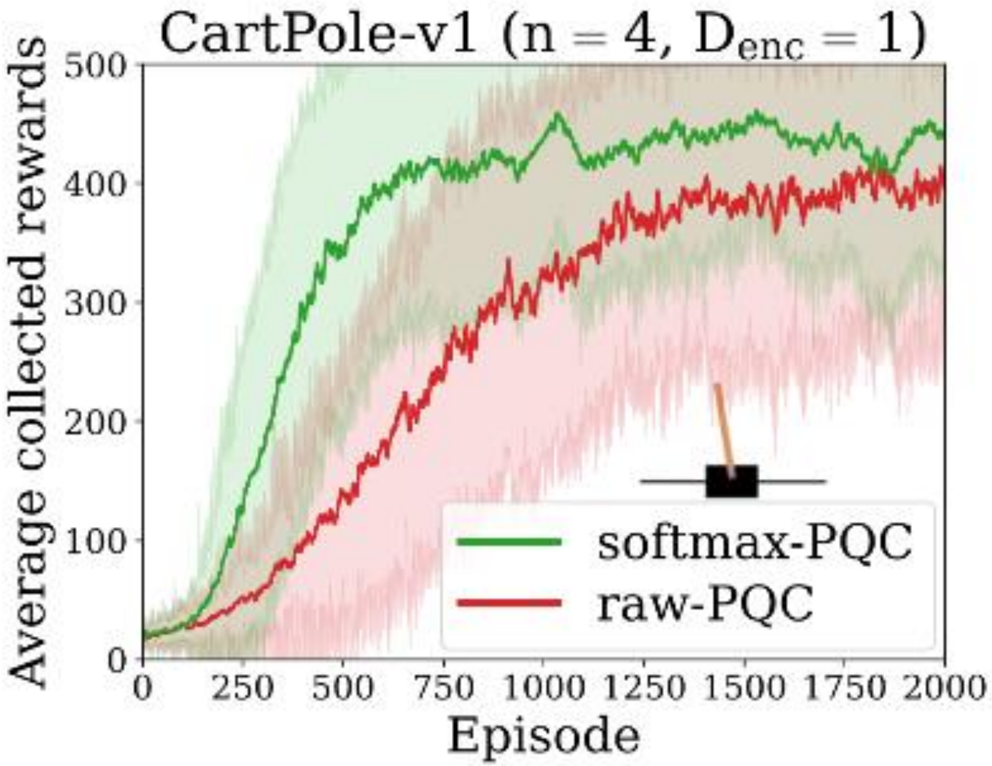
S. Jerbi et al., Parameterized Quantum Policies for Reinforcement Learning, NeurIPS 34, 28362-28375 (2021).

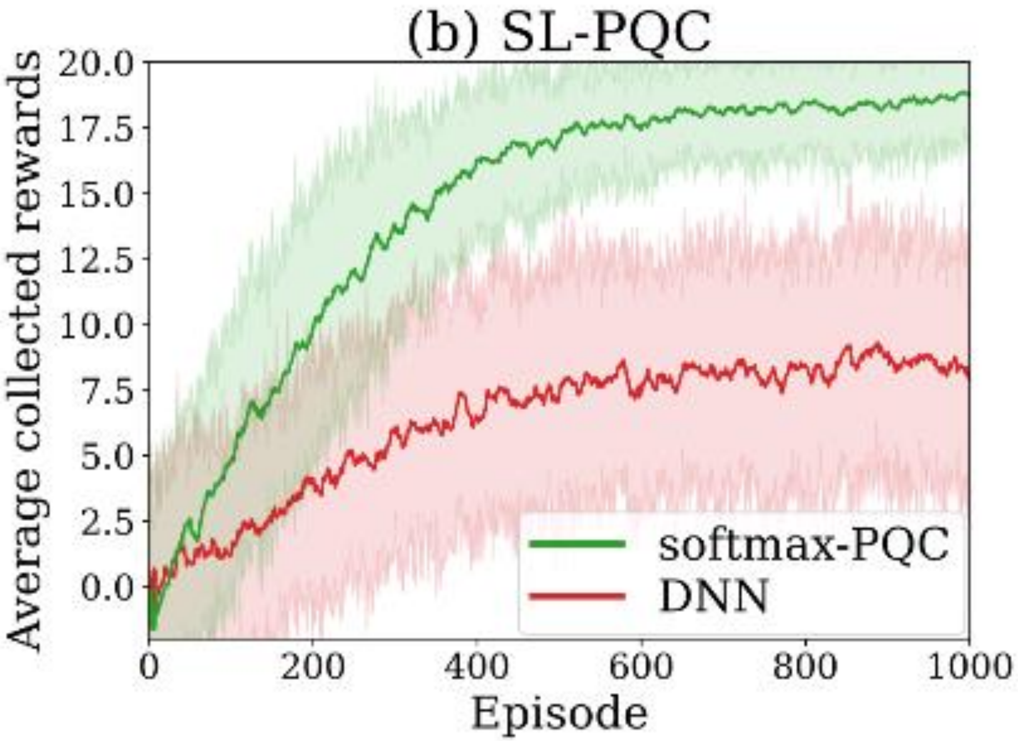
Quantum Policy Gradients

Some experimental result



In simulation, no quantum hardware!





S. Jerbi et al., Parameterized Quantum Policies for Reinforcement Learning, NeurIPS 34, 28362-28375 (2021).

Quantum Policy Gradients

(Empirical) parameter complexity

Not supported by theory or larger-scale experiments, take with grain of salt!

Algorithm	Encoding	Complexity (n = size of input, N = largest value)
Tabular Q-Learning		$\mathcal{O}(n^3)$
DQL		$\mathcal{O}(n^2)$
VQ-DQL	computational	$\mathcal{O}(n)$ / $\mathcal{O}(n \cdot log(N))$
	scaled	$\mathcal{O}(n)$
	directional	$\mathcal{O}(n)$
	amplitude	$\mathcal{O}(log(n))$

Chen et al., Variational Quantum Circuits for Deep Reinforcement Learning, IEEE Access 8, 141007-141024 (2020).

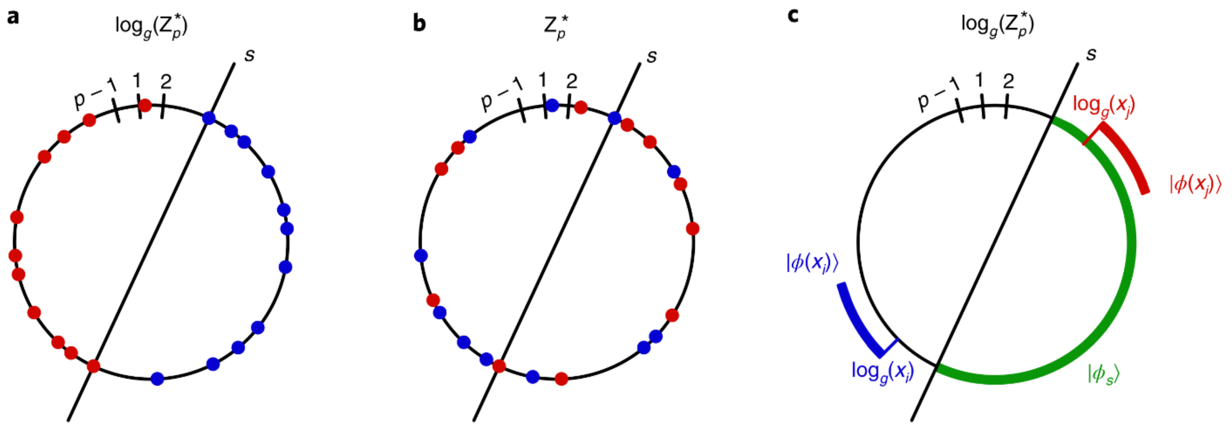
Lockwood and Si, Reinforcement Learning with Quantum Variational Circuits, AAAI Conference on AI and Interactive Digital Entertainment 16.1 (2020).

Quantum Policy Gradients

Provable quantum advantage

required interactions with environment

Exponential advantage in **sampling complexity** for (artificial) problem!



Y. Liu et al., A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. 17, 1013-1017 (2021).

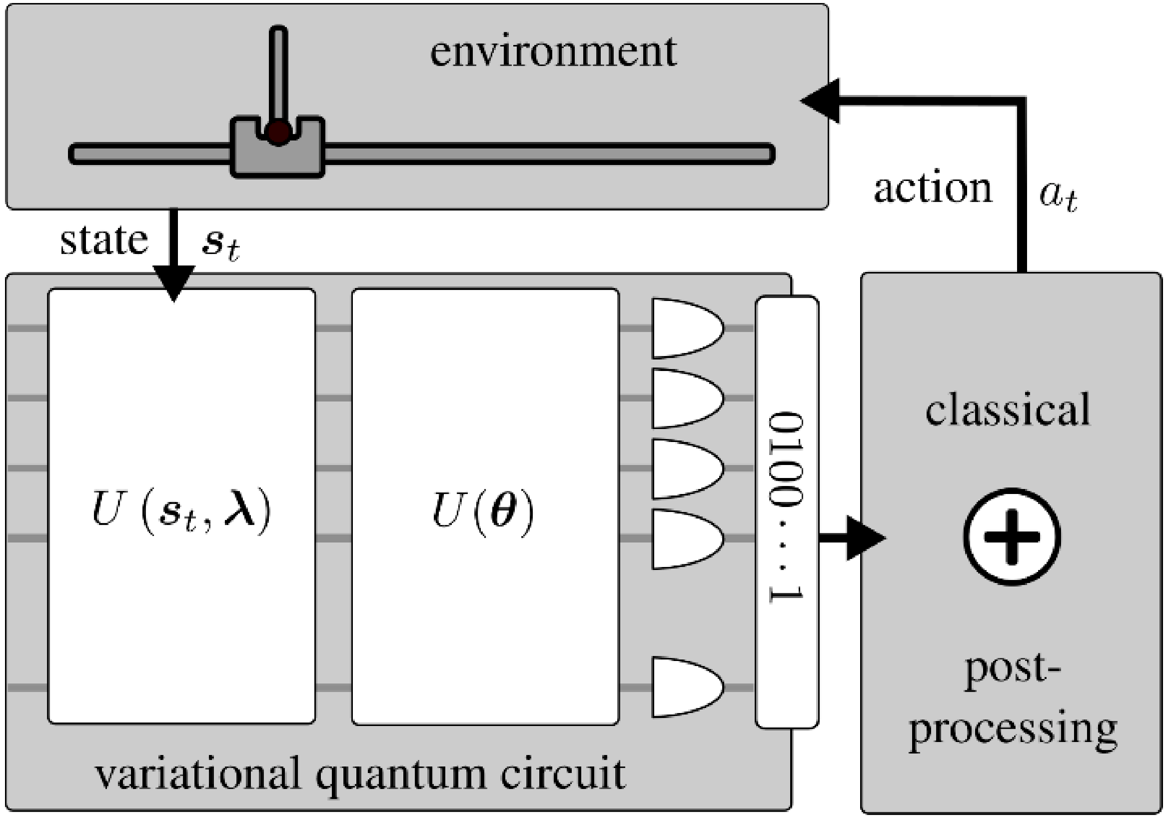
QPG agent is able to identify structure in data, that has been scrabled with discrete logarithm

$\hookrightarrow$ unfortunately far beyond capability of NISQ devices

Fraunhofer IIS

Extension #1: Classical Post-Processing

Improving convergence by “good” mapping from basis states to actions

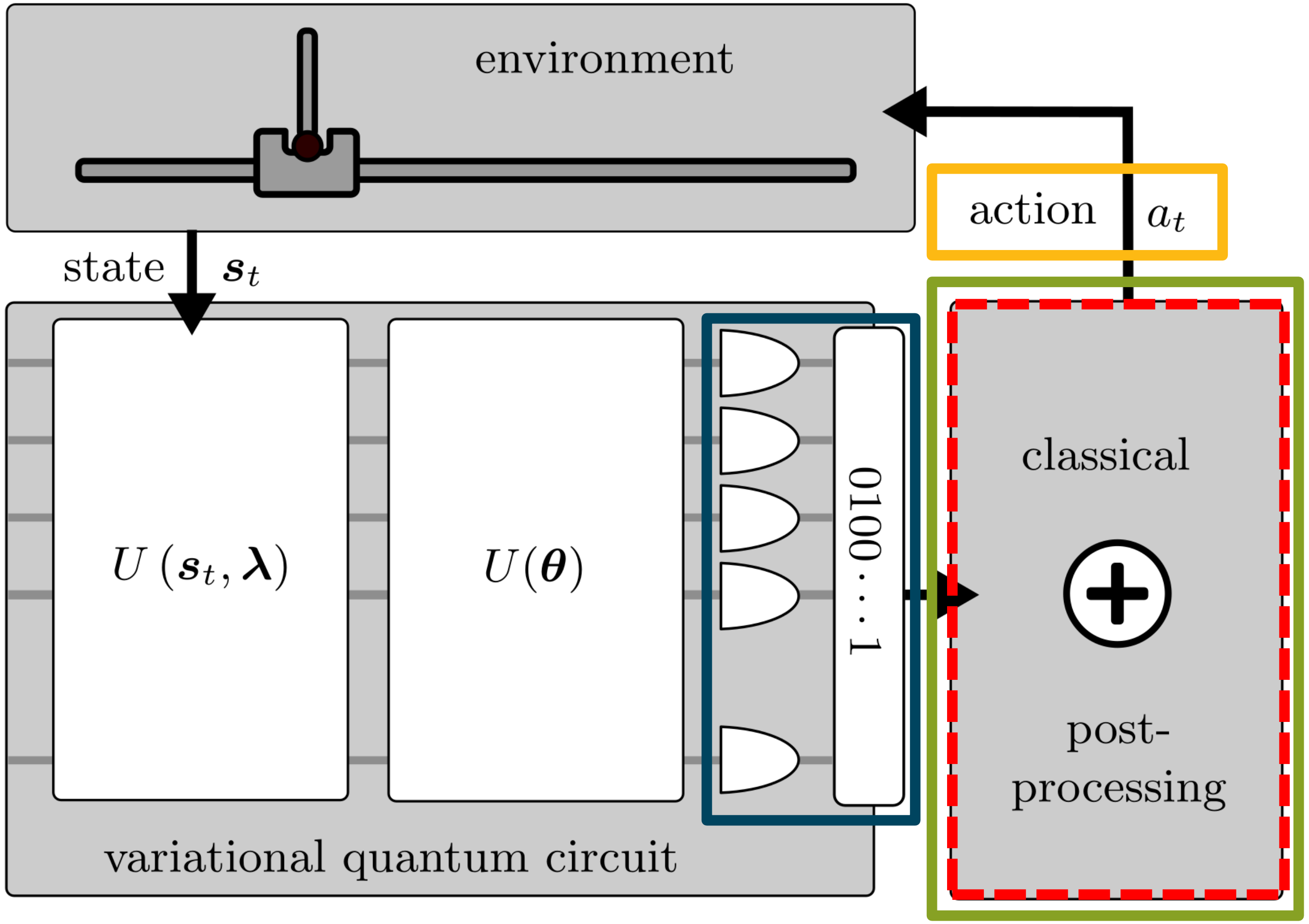


N. Meyer et al., ***Quantum Policy Gradient Algorithm with Optimized Action Decoding****,*
40th International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA (2023).

Fraunhofer IIS

Extension #1: Classical Post-Processing

Overview QPG algorithm



1. Quantum state is measured in the computational basis
2. Results are post-processed with a function maximizing our proposed globality measure
3. An action is selected
4. Update of the parameters utilizes the same post-processing scheme

Fraunhofer IIS

Extension #1: Classical Post-Processing

Different possibilities to partition basis states

$$|\psi_{s,\theta}\rangle = |q_0 q_1 q_2\rangle = \sum_{i=0}^{7} c_i |i\rangle$$

prepared quantum state

$$\pi_\theta(a \mid s)$$
$$= \langle \psi_{s,\theta} | \sum_{v \in \mathcal{V}_a} |v\rangle \langle v| | \psi_{s,\theta} \rangle$$
$$= \sum_{v \in \mathcal{V}_a} \langle \psi_{s,\theta} \mid v\rangle \langle v \mid \psi_{s,\theta}\rangle$$
$$= \sum_i |c_{a_i}|^2$$

remember that $\sum_i |c_i|^2 = 1!$

	Action 0		Action 1	
$\mathcal{V}_0^{1-loc} =$	$\|000\rangle$, $\|001\rangle$, $\|010\rangle$, $\|011\rangle$	$\mathcal{V}_1^{1-loc} =$	$\|100\rangle$, $\|101\rangle$, $\|110\rangle$, $\|111\rangle$	$a = q_0$
	Action 0		**Action 1**	
$\mathcal{V}_0^{glob} =$	$\|000\rangle$, $\|011\rangle$, $\|101\rangle$, $\|110\rangle$	$\mathcal{V}_1^{glob} =$	$\|001\rangle$, $\|010\rangle$, $\|100\rangle$, $\|111\rangle$	$a = \bigoplus_{i=0}^{n-1} q_i$

Extension #1: Classical Post-Processing

Switch from basis states to bitstrings

$$\boldsymbol{b} = b_{n-1} b_{n-2} \cdots b_0$$

Measured bitstring

Post-processing function: $f_{\mathcal{C}} : \{0,1\}^n \rightarrow \{0, 1, \cdots, |\mathcal{A}| - 1\}$

such that $f_{\mathcal{C}}(\boldsymbol{b}) = a$, iff $\boldsymbol{b} \in \mathcal{C}_a$ partition $\mathcal{C}_a \subset \mathcal{C}$ associated with action a

$$\pi_{\boldsymbol{\theta}}(a \mid \boldsymbol{s}) = \sum_{\boldsymbol{b} \in \{0,1\}^n}^{f_{\mathcal{C}}(\boldsymbol{b}) = a} \langle \psi_{\boldsymbol{s},\boldsymbol{\theta}} \mid \boldsymbol{b} \rangle \langle \boldsymbol{b} \mid \psi_{\boldsymbol{s},\boldsymbol{\theta}} \rangle$$

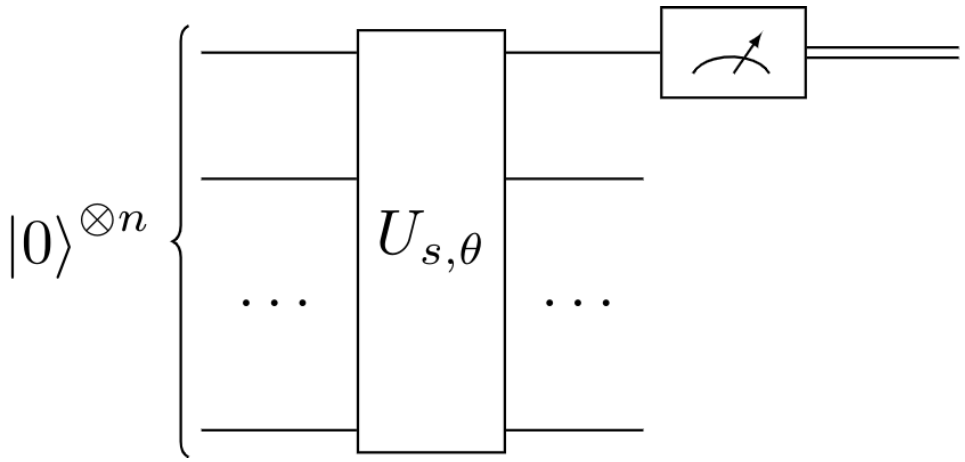
$$\approx \frac{1}{K} \cdot \sum_{k=0}^{K-1} \delta_{f_{\mathcal{C}}\left(\boldsymbol{b}^{(k)}\right) = a}$$

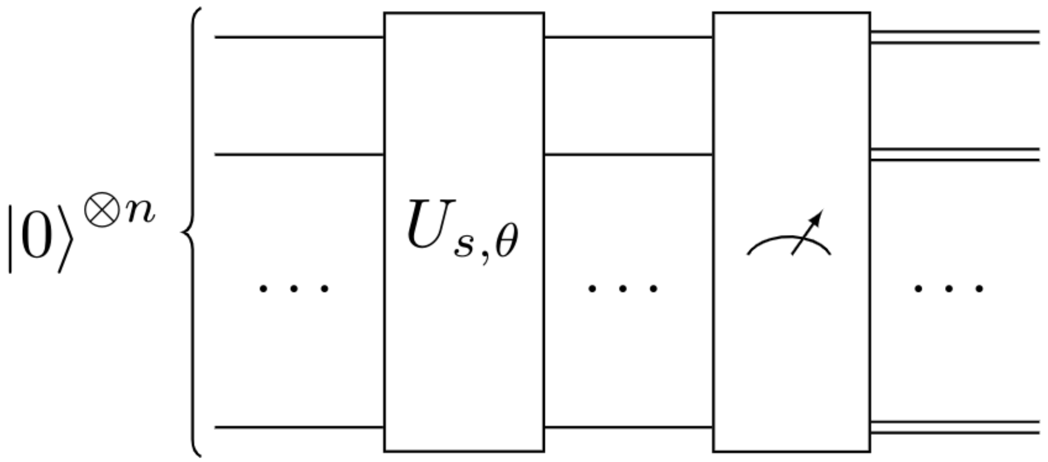
Fraunhofer IIS

Extension #1: Classical Post-Processing

Introducing globality measure [1/2]

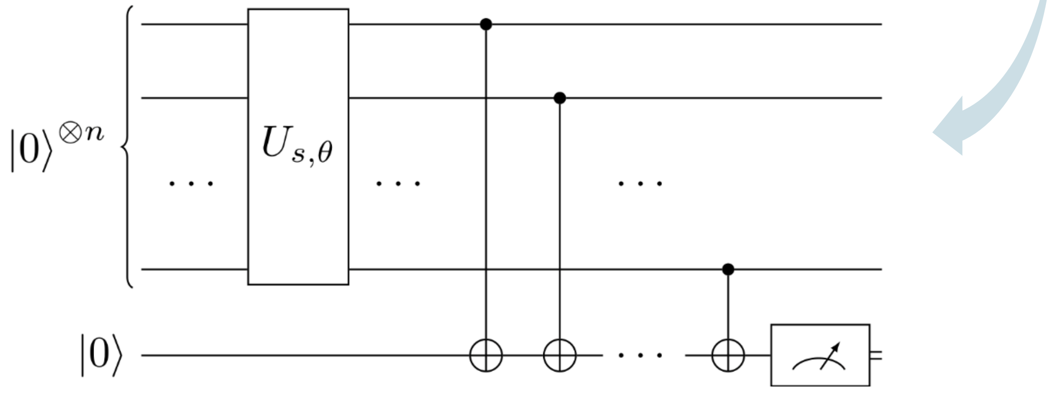
$$\pi_\theta^{q-loc} = \frac{(-1)^a \left\langle \psi_{s,\theta} \middle| Z^{\otimes q} \otimes I^{\otimes n-q} \middle| \psi_{s,\theta} \right\rangle + 1}{2}$$





This consideration has some flaws…

- only describes border cases for $|A| = 2$
- global measurement can be expressed as local measurement on ancilla qubit:



Extension #1: Classical Post-Proces

Introducing globality measure [2/2]

Example

$$\mathcal{C}_{a=0} = \{0000, 0010, 0100, 0110\}$$
$$\mathcal{C}_{a=1} = \{0001, 0011, 0101, 0111\}$$
$$\mathcal{C}_{a=2} = \{1000, 1010, 1101, 1111\}$$
$$\mathcal{C}_{a=3} = \{1001, 1011, 1100, 1110\}$$

$$EI_{f_\mathcal{C}}(0111) = 2 \qquad (0111 \in \mathcal{C}_{a=1})$$
$$EI_{f_\mathcal{C}}(1010) = 3 \qquad (1010 \in \mathcal{C}_{a=2})$$

$\hookrightarrow$ at least $\log_2(|\mathcal{A}|)$ bits

Definition 4.1 (extracted information)**.** Let $f_\mathcal{C}$ be a classical post-processing function, with a partitioning of the set of n-bit strings $\mathcal{C} = \bigcup_a \mathcal{C}_a$, for which $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ for all $i \neq j$. Furthermore, $\boldsymbol{b} = b_{n-1}b_{n-2}\cdots b_1 b_0$ denotes an arbitrary bitstring. The extracted information $EI_{f_\mathcal{C}}(\boldsymbol{b}) \in \mathbb{N}$ is the minimum number of bits b_i necessary, to compute $f_\mathcal{C}(\boldsymbol{b})$, i.e. assign $\boldsymbol{b}$ unambiguously to a set $\mathcal{C}_a$.

"Average amount of information necessary to have an unambigious distinction between actions."

$$G_{f_\mathcal{C}} := \frac{1}{2^n} \sum_{\boldsymbol{b}\in\{0,1\}^n} EI_{f_\mathcal{C}}(\boldsymbol{b})$$

upper bound $G_{f_\mathcal{C}} \leq n$ (Holevo's theorem)

Fraunhofer IIS

Extension #1: Classical Post-Processing

Post-processing that optimizes globality measure

$$m = \log_2(|\mathcal{A}| - 1) \in \mathbb{N}_0$$

$$f_{\mathcal{C}}(\boldsymbol{b}) = \left[b_0 \cdots b_{m-1} \left(\bigoplus_{i=m}^{n-1} b_i \right) \right]_{10}$$

decimal representation

$\hookrightarrow$ Lemma 1 guarantees $G_{f_{\mathcal{C}}} = n$

$$\mathcal{C}^{(m)}_{[a]_2} = \left\{ \boldsymbol{b} \mid \bigoplus_{i=m}^{n-1} b_i = a_0 \wedge \boldsymbol{b} \in \mathcal{C}^{(m-1)}_{a_m \cdots a_2 (a_1 \oplus a_0)} \right\}$$

$$\mathcal{C}^{(0)}_{[0]_2} = \left\{ \boldsymbol{b} \mid \bigoplus_{i=0}^{n-1} b_i = 0 \wedge \boldsymbol{b} \in \{0,1\}^n \right\}$$

$$\mathcal{C}^{(0)}_{[1]_2} = \left\{ \boldsymbol{b} \mid \bigoplus_{i=0}^{n-1} b_i = 1 \wedge \boldsymbol{b} \in \{0,1\}^n \right\}$$

Lemma 1. *Let an arbitrary VQC act on an n-qubit state. The* `RAW`*-VQC policy needs to distinguish between $M := |\mathcal{A}|$ actions, where M is a power of 2, i.e., $m = \log_2(M) - 1 \in \mathbb{N}_0$. Using Eqs. (13) to (15) we define*

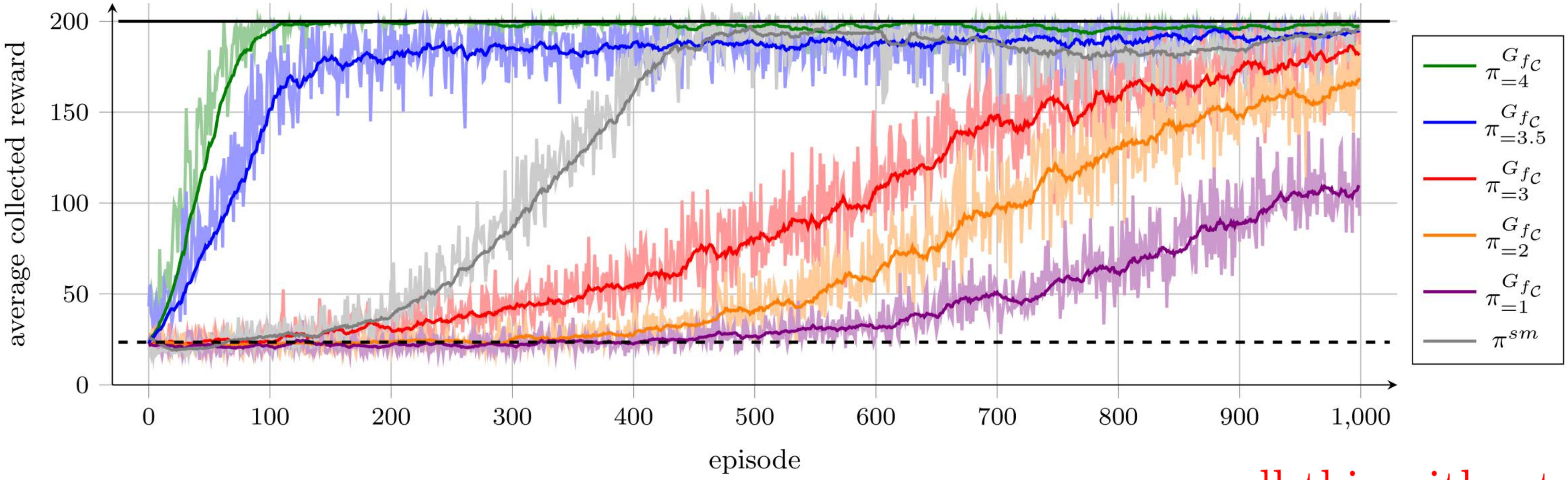
$$\pi^{glob}_{\Theta}(a \mid \boldsymbol{s}) = \sum_{v \in \mathcal{C}^{(m)}_{[a]_2}} \langle \psi_{\boldsymbol{s},\Theta} \mid v \rangle \langle v \mid \psi_{\boldsymbol{s},\Theta} \rangle \tag{16}$$

$$\approx \frac{1}{K} \sum_{k=0}^{K-1} \delta_{f_{\mathcal{C}^{(m)}}(\boldsymbol{b}^{(k)})=a} \tag{17}$$

where K is the number of shots for estimating the expectation value, $\boldsymbol{b}^{(k)}$ is the bitstring observed in the k-th shot, and δ is an indicator function. The post-processing function is guaranteed to have the globality value $G_{f_{\mathcal{C}}} = n$.

Extension #1: Classical Post-Processing

Why we did all of this? Improved RL performance (exemplary on CartPole-v0)!



$\hookrightarrow$ Similar results also for other environments (`CartPole-v1`, `FrozenLake`, `ContextualBandits`)

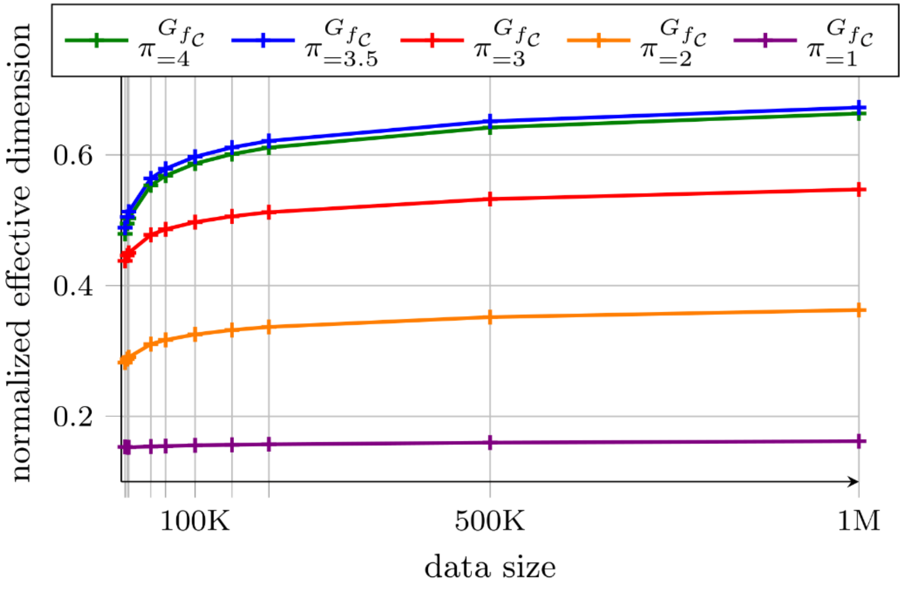
all this without basically any (quantum) overhead!

Extension #1: Classical Post-Processing

Additional performance measures

Expressibility

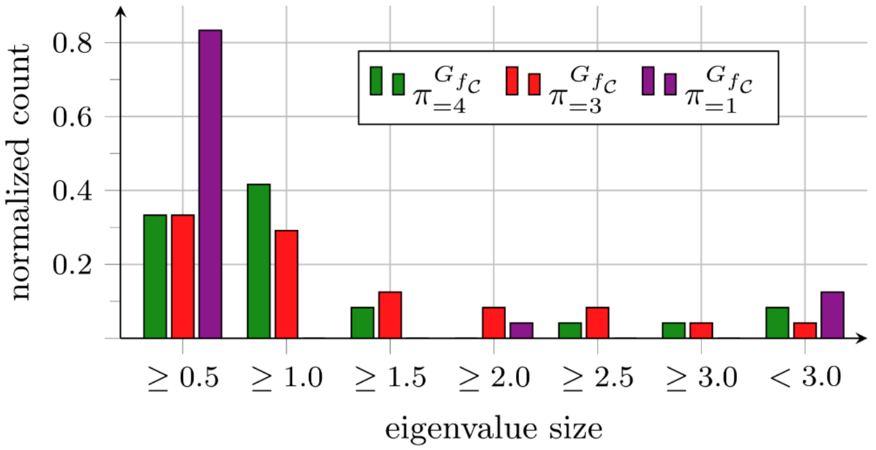
"Variety of functions than the model can approximate"



(a) Normalized effective dimension

Trainability

"Related to the geometry (*flatness*) of the parameter space"



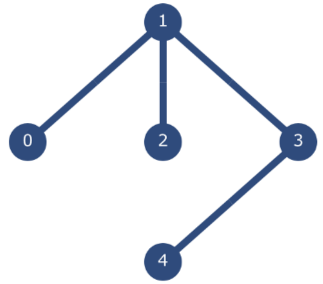
(b) Fisher information spectrum

A. Abbas et al., **The power of quantum neural networks**, *Nat. Comput. Sci. 1, 403-409 (2021)*

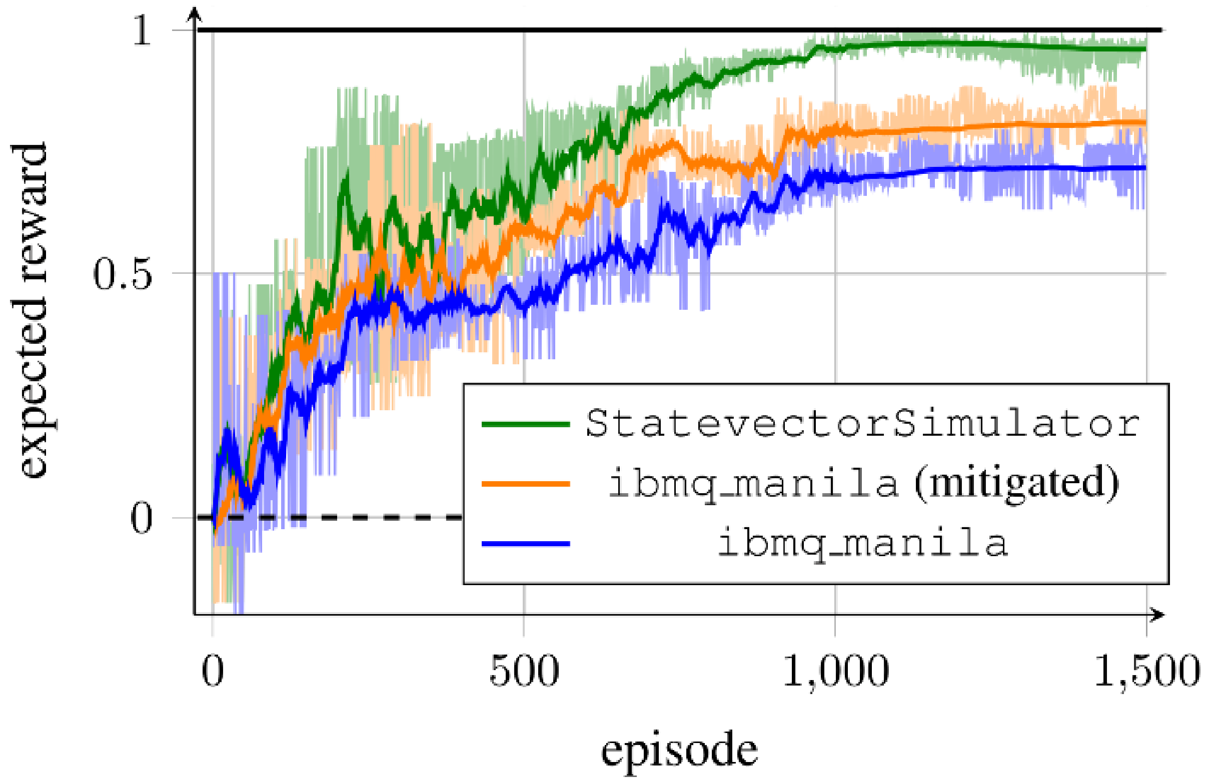
Extension #1: Classical Post-Processing

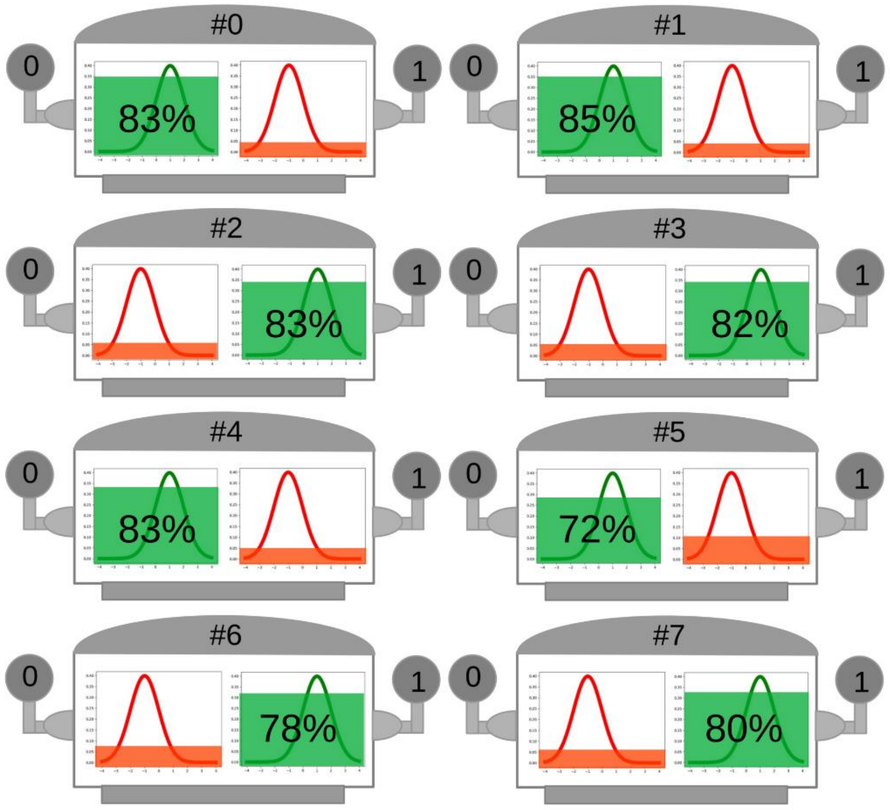
Training on actual quantum device!

Hardware architecture (ibmq_manila)



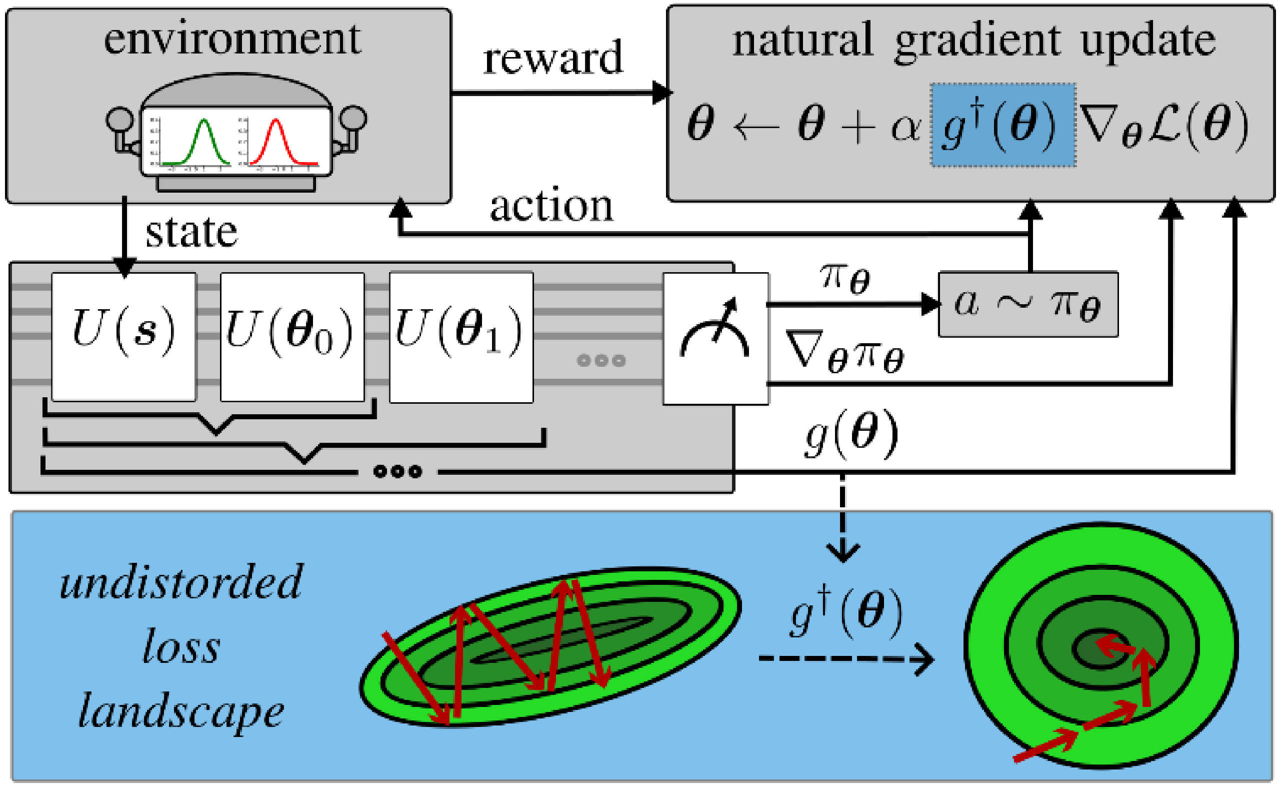
Policy learned on quantum device





Extension #2: Quantum Natural Gradients

Extending with second-order updates



*N. Meyer et al., **Quantum Natural Policy Gradients: Towards Sample-Efficient Reinforcement Learning**, arXiv:2304.13571 (2023).*

Extension #2: Quantum Natural Gradients

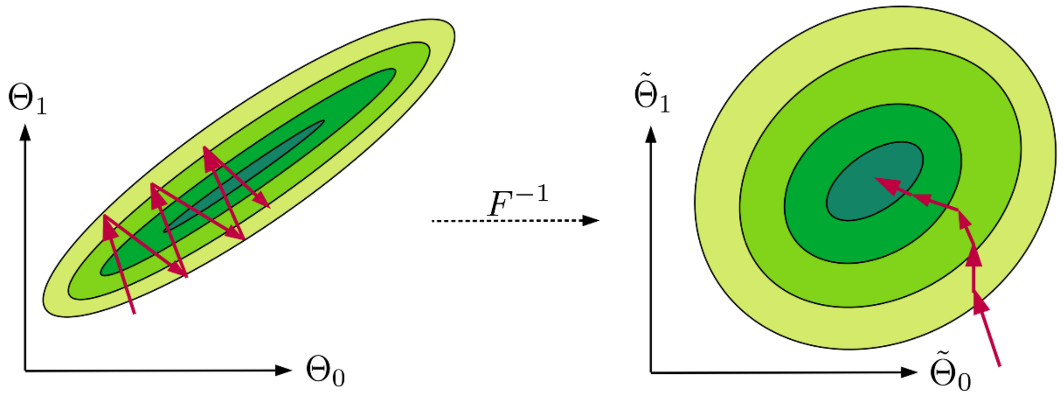
Reminder: Underlying idea of natural gradients

Usual gradient ascent update: $\theta \leftarrow \theta + \alpha \cdot \nabla_\theta \mathcal{L}(\theta)$

Problem: Does not pay attention to geometry of parameter space

$\hookrightarrow$ inverse of **Fisher information matrix** $F(\theta)$ to undistords parameter space:

$$\tilde{\nabla}_\theta = F(\theta)^{-1} \nabla_\theta \mathcal{L}(\theta)$$



Schematic effect of natural gradient update on parameter space

Extension #2: Quantum Natural Gradients

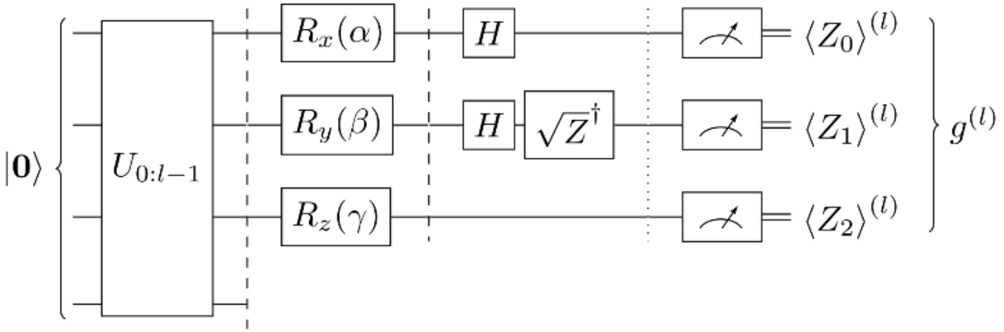
(Approximated) Quantum Fisher Information

For quantum state $|\psi_{s,\theta}\rangle$, the Fubini-Study metric tensor (aka QFIM) is:

$$G_{ij}(\theta) = Re\left[\left\langle \frac{\partial\psi_{s,\theta}}{\partial\theta_i} \,\middle|\, \frac{\partial\psi_{s,\theta}}{\partial\theta_j}\right\rangle - \left\langle \frac{\partial\psi_{s,\theta}}{\partial\theta_i} \,\middle|\, \psi_{s,\theta}\right\rangle \left\langle \psi_{s,\theta} \,\middle|\, \frac{\partial\psi_{s,\theta}}{\partial\theta_j}\right\rangle\right]$$

Problem: Exact evalaution not possible on quantum hardware

Solution: (Block-) Diagonal representation can be sampled



$$g^{(l)}(\boldsymbol{\theta}) = \begin{bmatrix} \langle Z_0^2\rangle - \langle Z_0\rangle^2 & \cdots & \langle Z_0 Z_q\rangle - \langle Z_0\rangle\langle Z_q\rangle \\ \vdots & \ddots & \vdots \\ \langle Z_q Z_0\rangle - \langle Z_q\rangle\langle Z_0\rangle & \cdots & \langle Z_q^2\rangle - \langle Z_q\rangle^2 \end{bmatrix}$$

J. Stokes et al., **Quantum Natural Gradient**, *Quantum 4, 269 (2020).*

Fraunhofer IIS

Extension #2: Quantum Natural Gradients

QNPG Algorithm

Pseudoinverse undistords parameter space:

$$\tilde{\nabla}_\theta = g(\theta)^+ \nabla_\theta \mathcal{L}(\theta)$$

Increase numerical stability:

$$\hat{\eta} = \arg\min_{\eta} \; \|g(\theta)\eta - \nabla_\theta \mathcal{L}(\theta)\|_2^2$$

(Optional) Regularization:

$$\hat{\eta}_\xi = \arg\min_{\eta} \; \|g(\theta)\eta - \nabla_\theta \mathcal{L}(\theta)\|_2^2 + \xi \, \|\eta\|_2^2$$

Algorithm 1 Quantum Natural Policy Gradients

Input: initial policy $\pi_{\boldsymbol{\theta}}$, batch size B, discount factor γ, learning rate α, termination condition

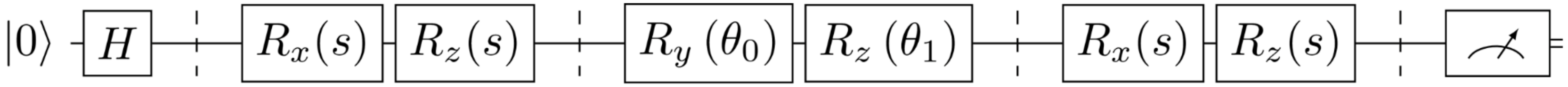
Output: policy $\pi_{\boldsymbol{\theta}_*}$ trained to maximize long term reward

```
 1: while termination condition not satisfied do
 2:     generate B trajectories [s_0, a_0, r_0, s_1, a_1 ···] from π_θ
 3:     for all trajectories τ in batch do
 4:         for timestep t in 0, ··· , H − 1 do
 5:             compute discounted returns G_t ← Σ_{t'=t}^{H−1} γ^{t'−t} r_{t'}
 6:             sample first-order gradients ∇_θ ln π_θ(a_t|s_t)
 7:             estimate Fubini-Study metric tensor g(θ)
 8:             solve for η_t in g(θ)η_t = ∇_θ ln π_θ(a_t|s_t)
 9:         end for
10:     end for
11:     compute batch average Δθ ← 1/(B·H) Σ_τ Σ_{t=0}^{H−1} η_t G_t
12:     perform gradient ascent update θ ← θ + αΔθ
13: end while
```

Extension #2: Quantum Natural Gradients

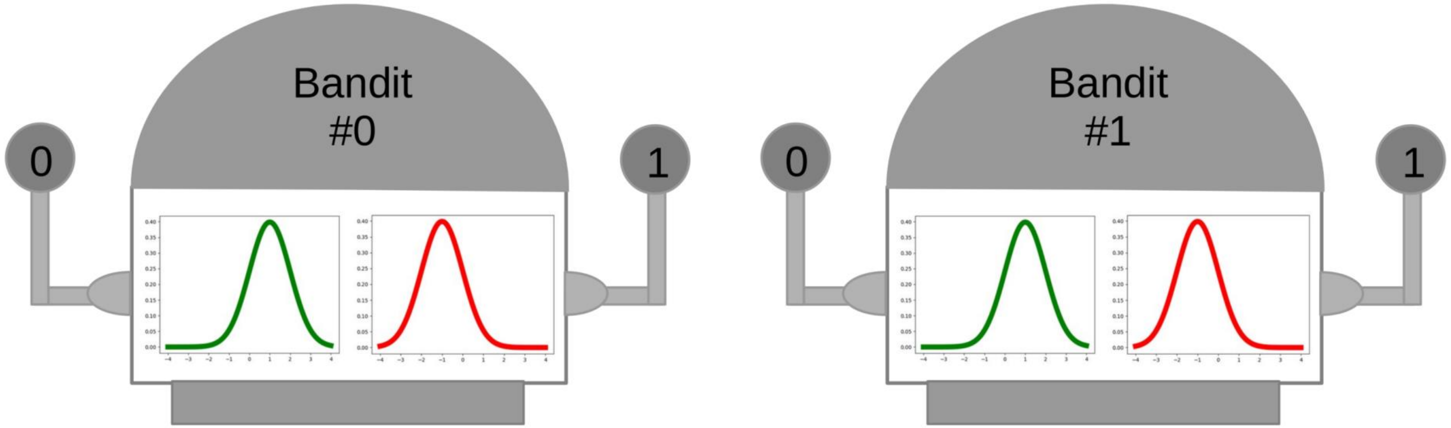
Simple experimental setup [1/2]

Simple one-qubit circuit with two parameters:



(a) Variational quantum circuit with two learnable parameters θ_0 and θ_1. The two states are encoded with first-order angle encoding, where state 0 is encoded as $s = 0$ and state 1 as $s = 1$.

Contextual bandit environment with two states and actions:



Extension #2: Quantum Natural Gradients

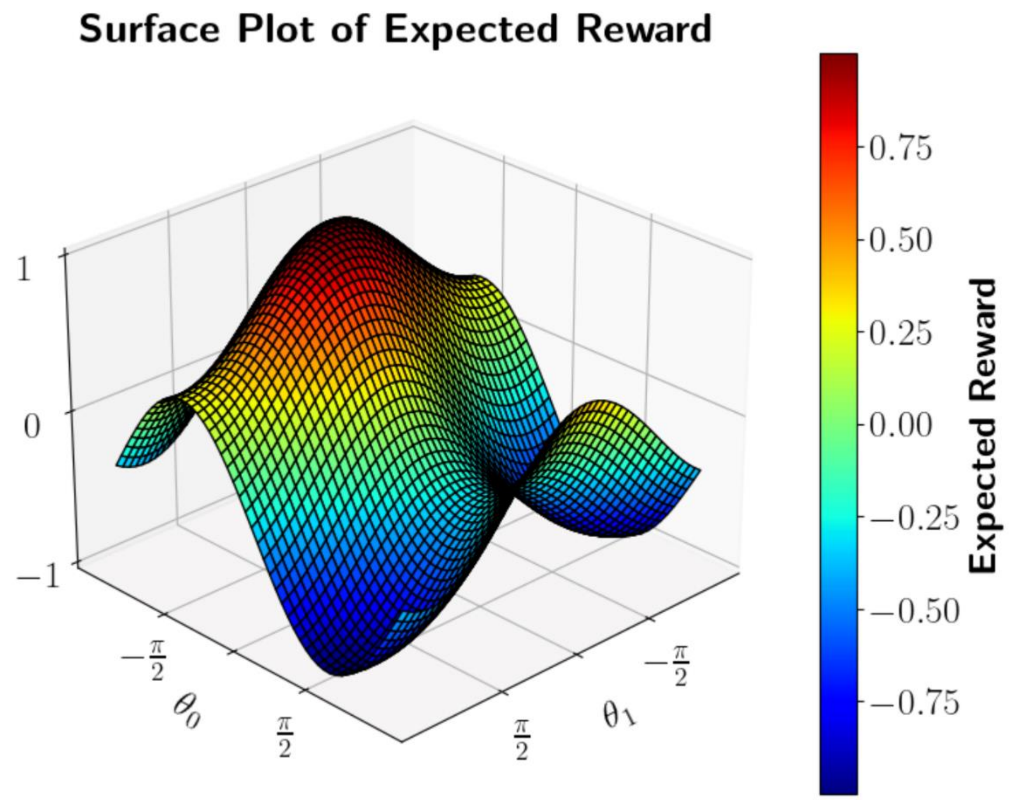
Simple experimental setup [2/2]

As the parameter space is only two-dimensional, the expected reward in this space can be visualized:

Expected reward:

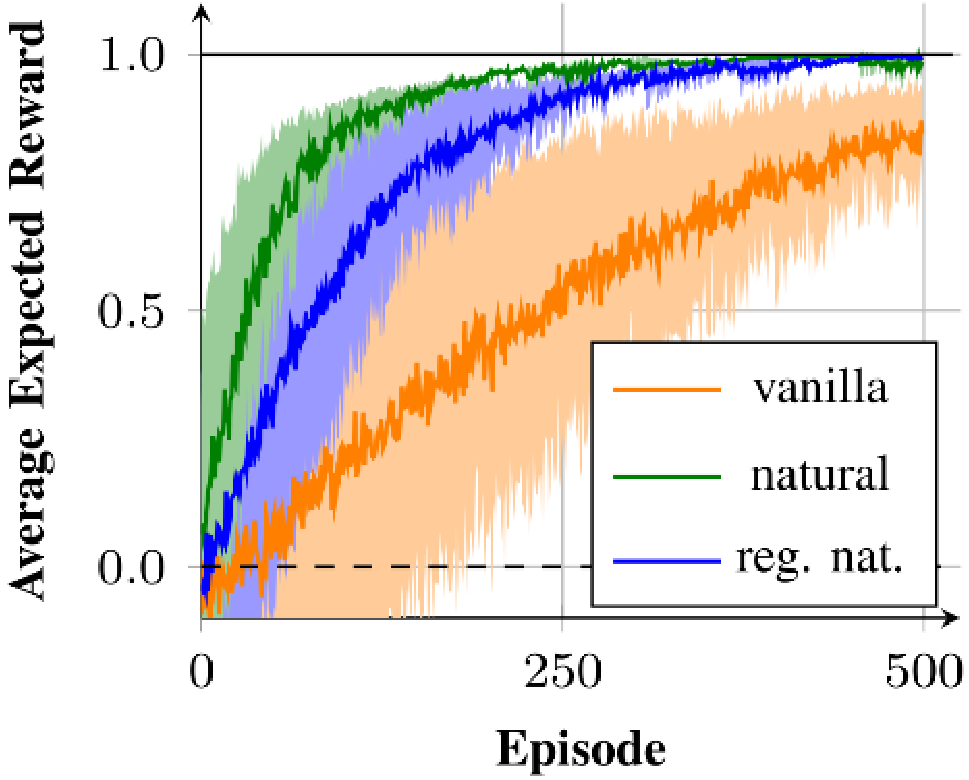
$$\langle r \rangle = \frac{1}{2^n} \sum_{s\in\{0,1\}^n} \pi_\theta(a_{opt}|s) - \pi_\theta(\tilde{a}_{opt}|s)$$

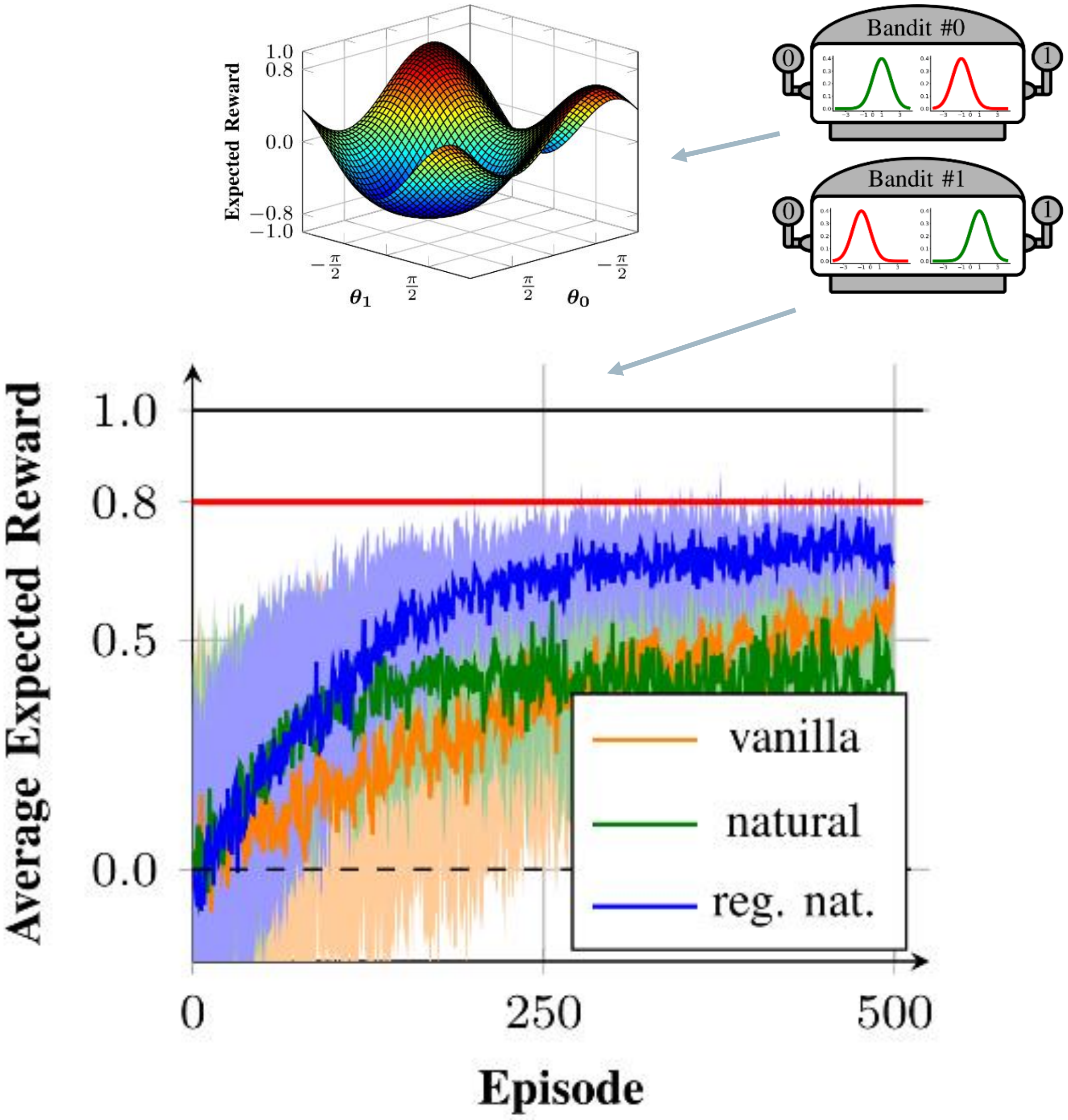
optimal action　　suboptimal action



Extension #2: Quantum Natural Gradients

Results for random initialization



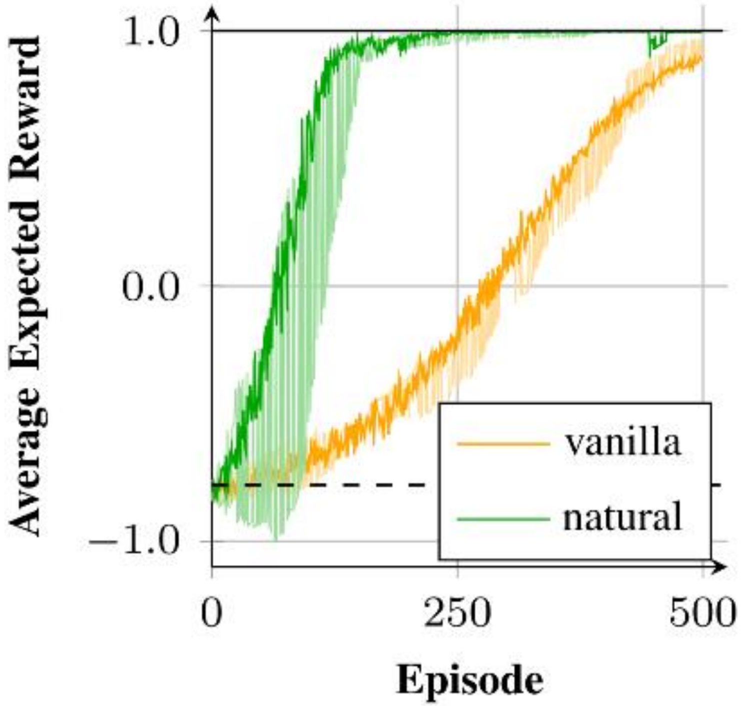


QNPG clearly outperforms QPG algorithm

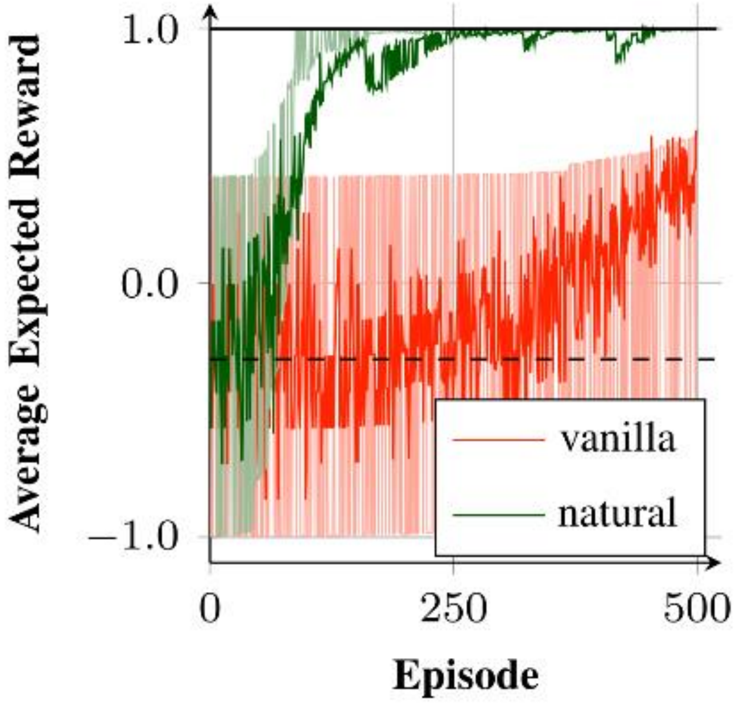
Fraunhofer IIS

Extension #2: Quantum Natural Gradients

A closer look at specific initializations [1/2]



(c) Initialized near minimum.



(a) Initialized in distorted region.

Natural gradients improve convergence especially in distorted regions

Extension #2: Quantum Natural Gradients

A closer look at specific initializations [2/2]

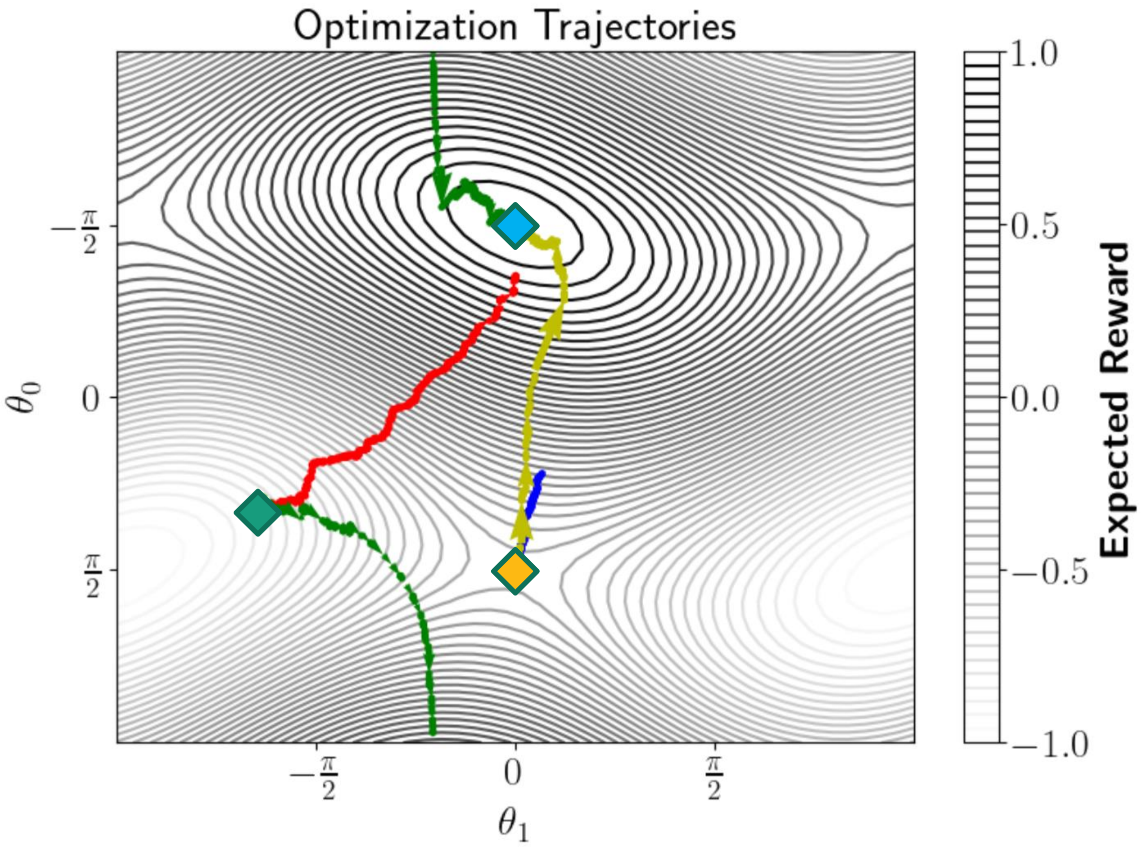
Near Minimum:

- vanilla gradient
- natural gradient

Distorted Region:

- vanilla gradient
- natural gradient

Maximum



Natural gradient helps to traverse distorted regions in parameter space

$\hookrightarrow$ might be suited to defer barren plateau problem

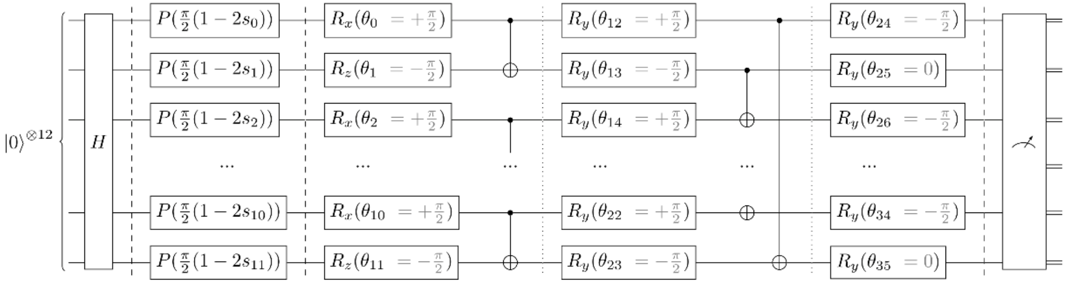
similar to classical vanishing gradients, but much worse

Fraunhofer IIS

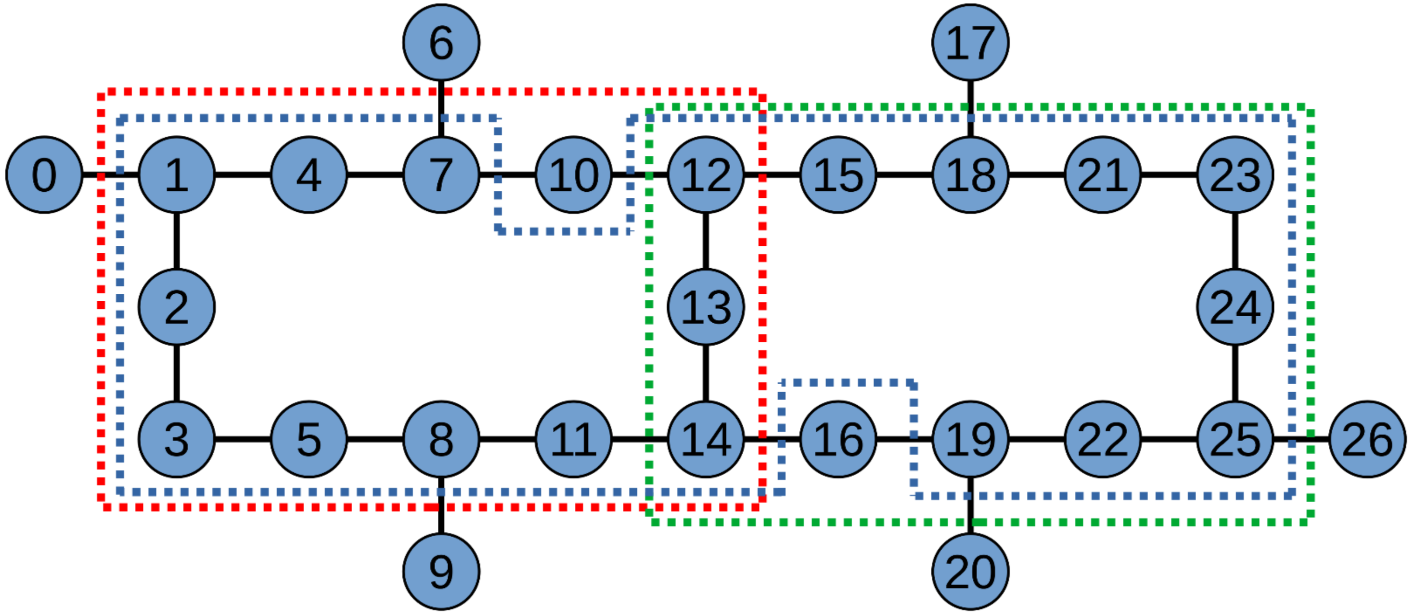
Extension #2: Quantum Natural Gradients

Larger-scale hardware experiment on 12 qubits [1/2]

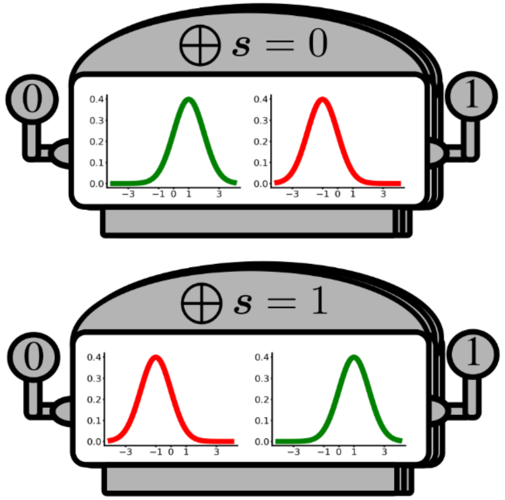
About 30 qubits can be simulated classically



sparsely entangeled VQC



subtopology of ibmq_ehningen device



(b) *Parity* environment.

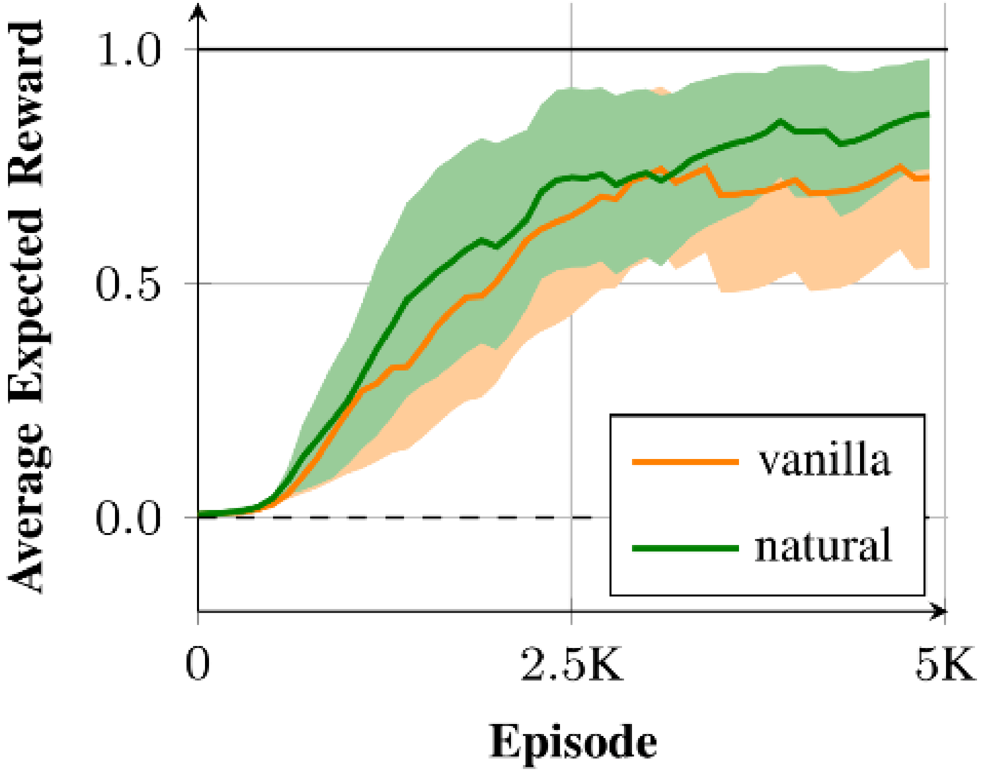
https://quantum-computing.ibm.com/https://quantum-computing.ibm.com/

Fraunhofer IIS

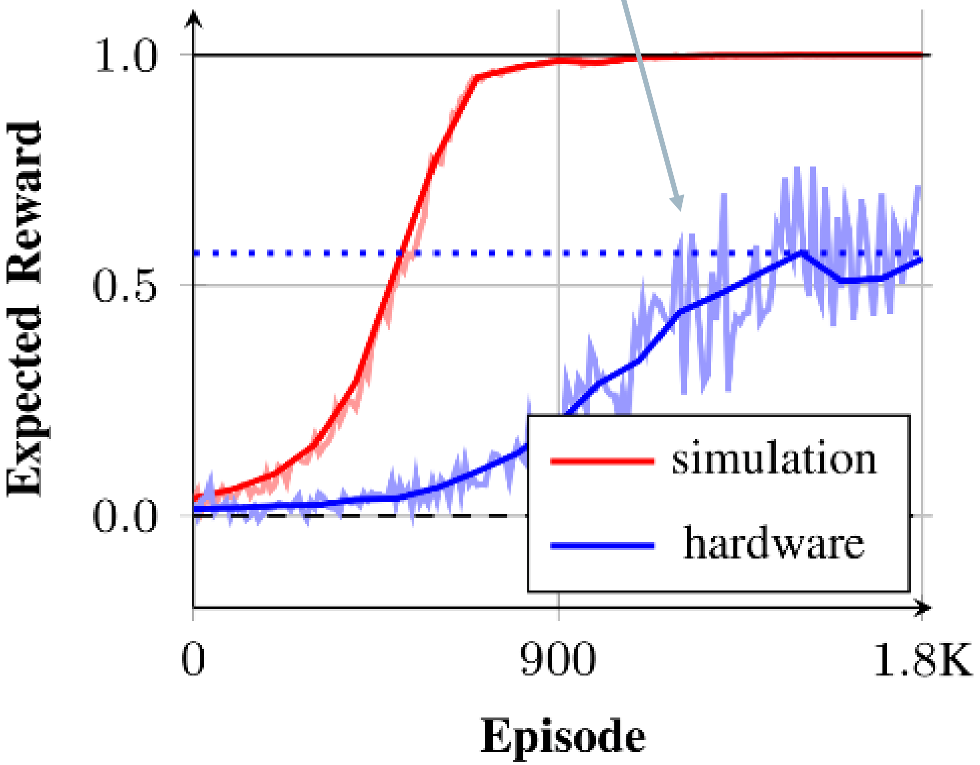
Extension #2: Quantum Natural Gradients

Larger-scale hardware experiment on 12 qubits [2/2]

deteriation is caused by currently inevitable hardware noise



comparison in simulation



quantum natural gradients only introduce insignificant overhead!

Summary

General Concept

- Quantum Computing
- Quantum Reinforcement Learning

Recap

- Policy-based Reinforcement Learning
- Policy Gradients

Concrete Algorithm

- Quantum Policy Gradients
- Extension #1: Classical Post-Processing
- Extension #2: Quantum Natural Gradients

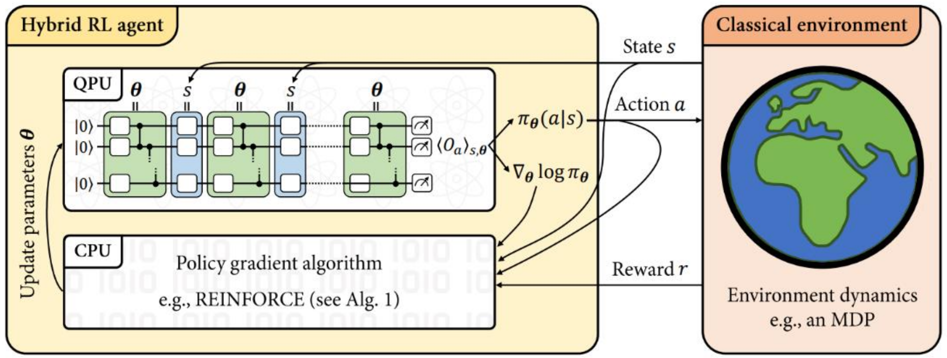
Quantum Reinforcement Learning

A field with great potential but also a lot of unsolved problems

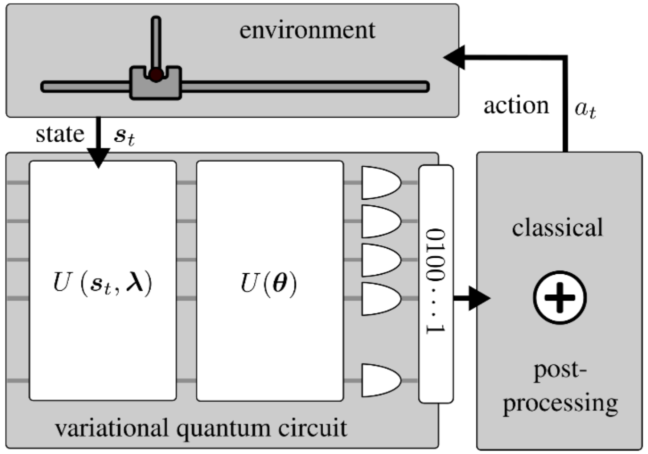
(Deep) Reinforcement Learning + **Quantum Computing**

↓

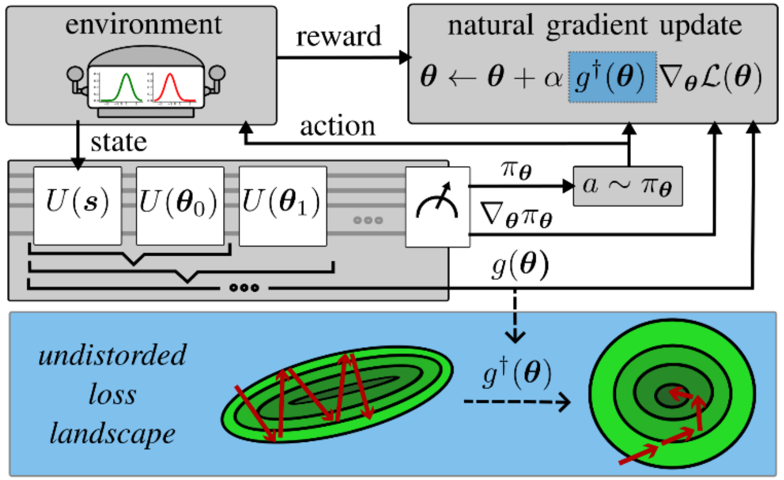
Quantum Reinforcement Learning



Vanilla Quantum Policy Gradients (QPG)



QPG with optimized (classical) action decoding



Quantum Natural Policy Gradients (QNPG)

Additional references for delving deeper

- The 'bible' of QC: M. Nielsen and I. Chuang, **Quantum Computation and Quantum Information**, http://mmrc.amss.cas.cn/tlb/201702/W020170224608149940643.pdf (2000).
- A bit easier digestible: N. D. Mermin, **Quantum Computer Science: An Introduction**, https://library.uoh.edu.iq/admin/ebooks/22831-quantum_computer_science.pdf (2007).
- Somewhere in-between the two others: P. Kaye et al., **An Introduction to Quantum Computing**, http://mmrc.amss.cas.cn/tlb/201702/W020170224608149125645.pdf (2007).
- **Video lecture by John Preskill** (one of the entities in the field): https://www.youtube.com/watch?v=w08pSFsAZvE&list=PL0ojjrEqIyPy-1RRD8cTD_lF1hflo89Iu
- **Lecture notes by Scott Aaronson** (probably THE quantum computing blogger): https://scottaaronson.blog/?p=3943
- **Video lecture by Michael Hartmann** (more physics-focused than the others): https://www.fau.tv/course/id/846
- Overview of several concepts from QML (but not up to date anymore): M. Schuld and F. Pettrocione, **Supervised Learning with Quantum Computers**, http://ndl.ethernet.edu.et/bitstream/123456789/73371/1/320.pdf (2018).
- A closer look what might be possible with current-day hardware: J. Preskill, **Quantum Computing in the NISQ era and beyond**, https://quantum-journal.org/papers/q-2018-08-06-79/ (2018).
- Same as previous, but bit more up to date: K. Bharti et al., **Noisy intermediate-scale quantum algorithms**, https://journals.aps.org/rmp/abstract/10.1103/RevModPhys.94.015004 (2022).
- Our survey ☺: N. Meyer et al., **A Survey on Quantum Reinforcement Learning**, https://arxiv.org/abs/2211.03464 (2022).