Quantum Reinforcement Learning for Particle Beam Steering

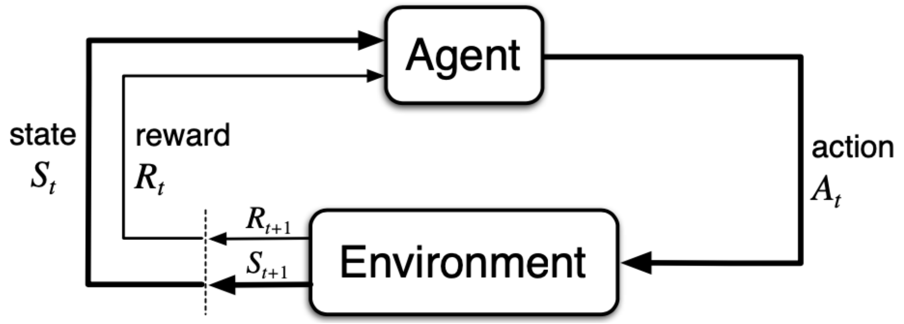
M. Schenk, M. Grossi, V. Kain, K. Li, S. Vallecorsa
CERN, Switzerland

E. F. Combarro
University of Oviedo, Spain

M. Popa
Politehnica University of Bucharest, Romania

Introduction

Reinforcement learning (RL) in a nutshell



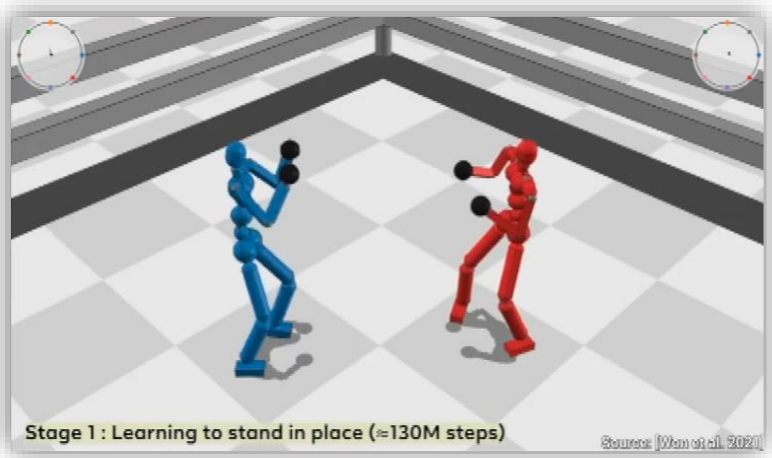
RL book: Sutton & Barto

Agent interacts with environment

- **Receives reward after every action**
- Learns through **trial-and-error**
- **Training sample:** $(s_t, a_t, r_t, s_{t+1}, d_t)$

Decision making

- Agent follows **policy π**: $S \to A$
- **Goal:** find optimal policy π^*
- **Optimal ⇔ maximizing return:** $G_t = \sum_k \gamma^k R_{t+k}$



https://www.youtube.com/watch?v=SsJ_AusntiU
https://www.youtube.com/watch?v=Lu56xVlZ40M
https://www.youtube.com/watch?v=imOt8ST4Ej

Expected return can be estimated through ***value function $Q(s, a)$***

- Helps answering: **"Best action to take in given state?"**
- Not a priori known, but **can be learned iteratively**
- **Q-learning:** learn $Q(s, a)$ using **function approximator**
  - **DQN:** Deep Q-learning *(feed-forward neural network)*
  - **FERL:** Free energy based RL *(quantum Boltzmann machine)*

Introduction

FERL motivation

- **Free energy based RL**
  - Efficient for **high-dimensional spaces**
  - Q-function estimate: **free energy of coupled spin system**
  - **Spin system ⇔ quantum Boltzmann machine** (QBM)
- **Higher sample efficiency compared to classical deep Q-learning**
- Limiting here: **discrete state and action spaces**

Free energy-based reinforcement learning using a quantum processor

Anna Levit,[1] Daniel Crawford,[1] Navid Ghadermarzy,[1,2]
Jaspreet S. Oberoi,[1,3] Ehsan Zahedinejad,[1] and Pooya Ronagh[1,2,*]
[1] 1QBit, 458-550 Burrard Street, Vancouver (BC), Canada V6C 2B5
[2] Department of Mathematics, The University of British Columbia, 121-1984 Mathematics Road, Vancouver (BC), Canada V6T 1Z2
[3] School of Engineering Science, Simon Fraser University, 8888 University Drive, Burnaby (BC), Canada V5A 1S6

Recent theoretical and experimental results suggest the possibility of using current and near-future quantum hardware in challenging sampling tasks. In this paper, we introduce free energy-based reinforcement learning (FERL) as an application of quantum hardware. We propose a method for processing a quantum annealer's measured qubit spin configurations in approximating the free energy of a quantum Boltzmann machine (QBM). We then apply this method to perform reinforcement learning on the grid-world problem using the D-Wave 2000Q quantum annealer. The experimental results show that our technique is a promising method for harnessing the power of quantum sampling in reinforcement learning tasks.

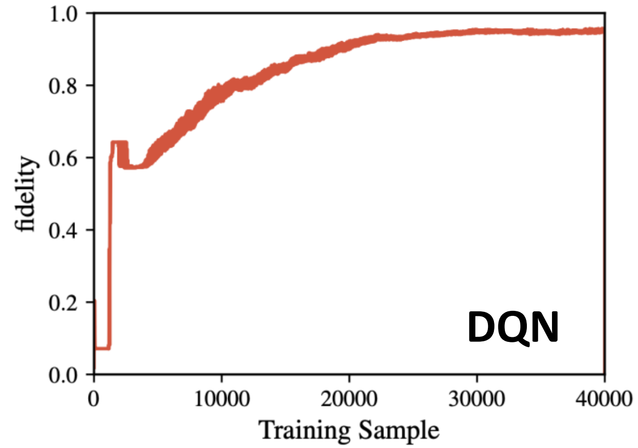
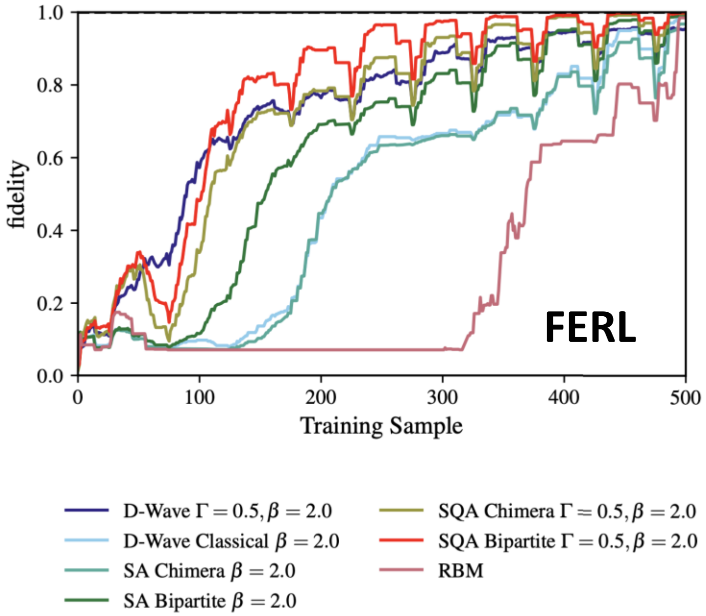


FIG. 4: The learning curve of a deep Q-network (DQN) with two hidden layers, each with eight hidden nodes, for the grid-world problem instance as shown in Fig. IV.



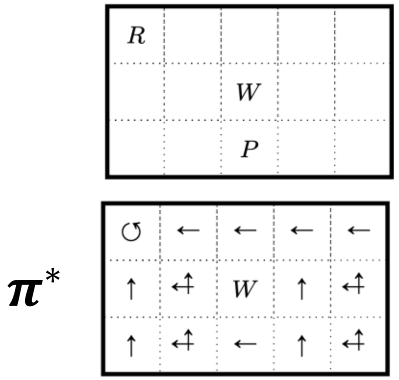


FIG. 3: (top) A 3×5 grid-world problem instance with one reward, one wall, and one penalty. (bottom) An optimal policy for this problem instance is a selection of directional arrows indicating movement directions.

https://arxiv.org/pdf/1706.00074.pdf

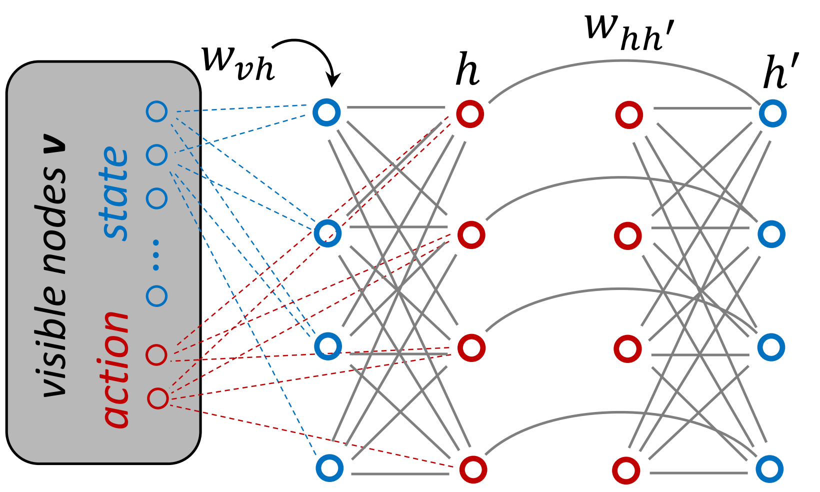
Introduction

QBM vs. DQN

FERL: QBM

- **Network of coupled, stochastic, binary units** *(e.g. qubits in spin up / down states)*
- $\boldsymbol{Q(s,a)}$ ≈ **negative free energy** of coupled spin system
- **Sampling ground-state** spin configuration using (simulated) **quantum annealing**
- **Implicit**

QBM

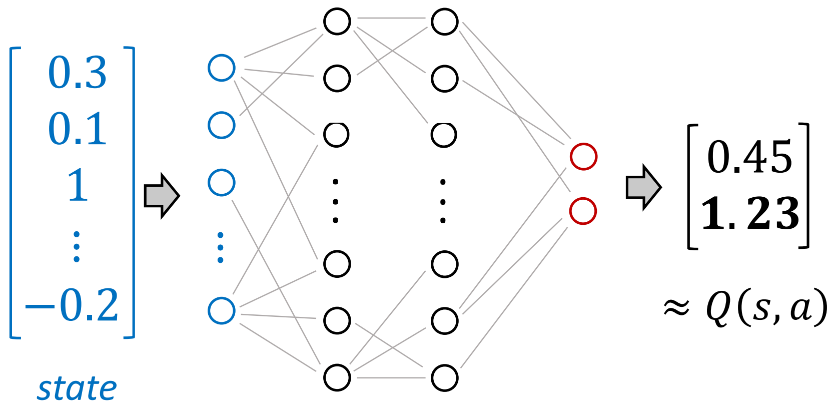


$$Q(s,a) \approx -F(\boldsymbol{v}) = -\langle H_{\boldsymbol{v}}^{\text{eff}} \rangle - \frac{1}{\beta}\sum_c \mathbb{P}(c|\boldsymbol{v}) \log \mathbb{P}(c|\boldsymbol{v})$$

Classical Q-learning: DQN

- **Feed-forward, dense neural network**
- **Explicit**

DQN



Project overview

Objectives

- **Implement FERL** using **simulated quantum annealing** and am **actual quantum annealer** (D-Wave)
- **Extend to continous state-action spaces** for real-world applications: **quantum actor-critic**
- **Compare quantum approach to classical RL** in terms of
  1) **Training efficiency** – *"# steps required to train agent"*
  2) **Descriptive power of QBM** – *"# weights needed"*

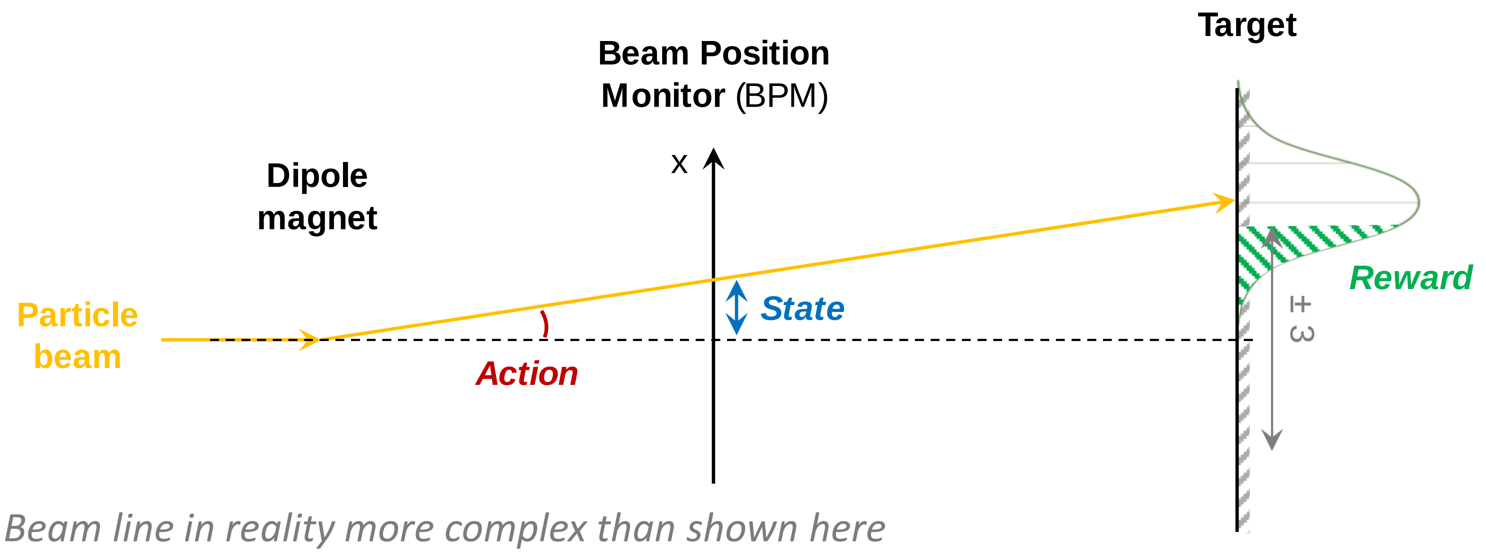
Use case I: Q-learning on 1D beam steering model *(simulated environment)*

Use case II: quantum actor-critic on 10D AWAKE beam line *(simulated and real environment)*

Use case I: Q-learning on 1D beam steering

Environment

- **OpenAI gym template**
- **Action:** deflection angle *(Discrete)*
- **State:** beam position *(continous)*
- **Reward:** integrated beam intensity on target

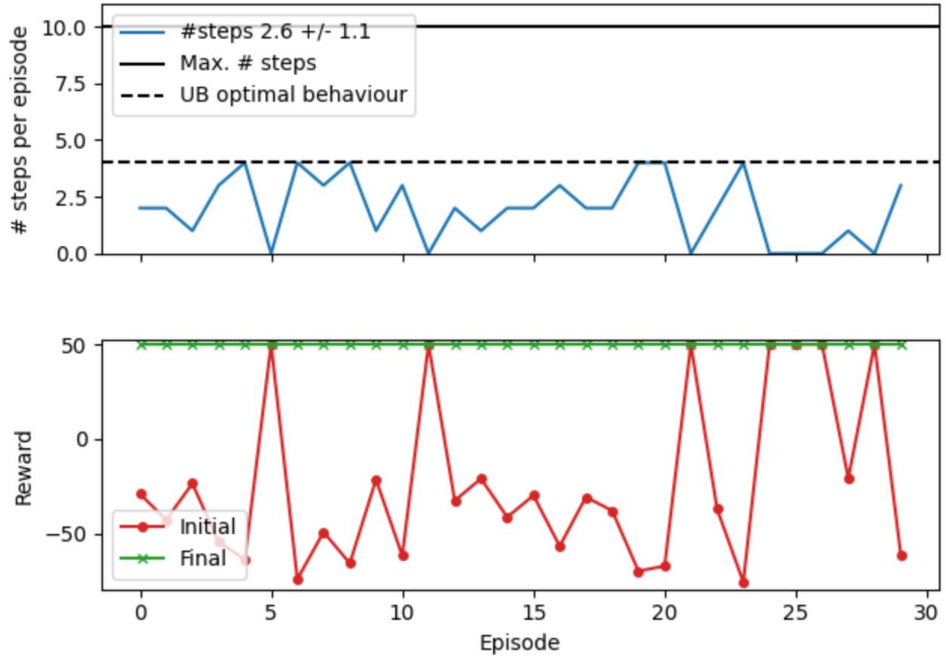


Beam line in reality more complex than shown here

Use case I: Q-learning on 1D beam steering

First successes with simulator and D-Wave quantum annealer

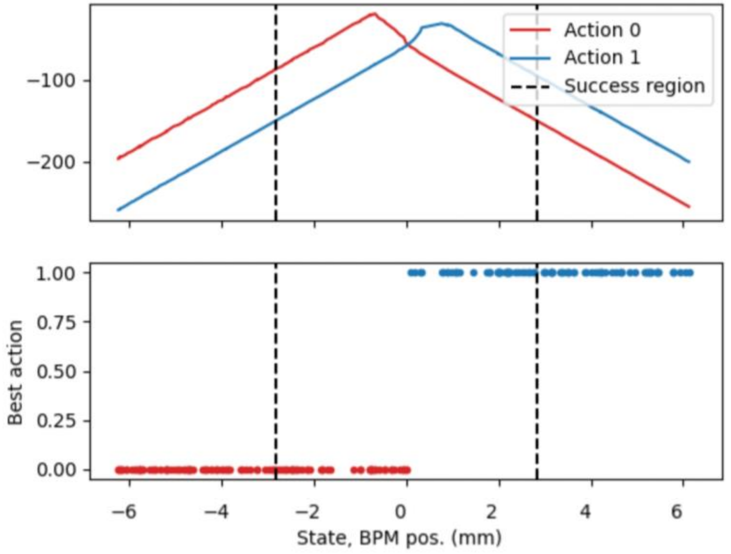
D-Wave training *and* evaluation



- **First success on D-Wave 2000Q: FERL works!**
- **Training** on hardware and with simulator **equally efficient**
- **Using same hyperparameters:** very helpful to optimize with simulator and then run on real hardware

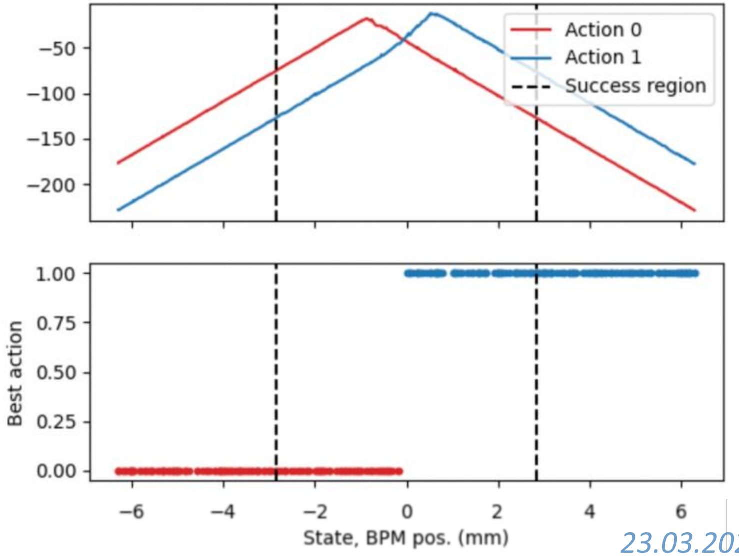
Trained with simulator

120 steps, batch size: 10



Trained on D-Wave quantum annealer

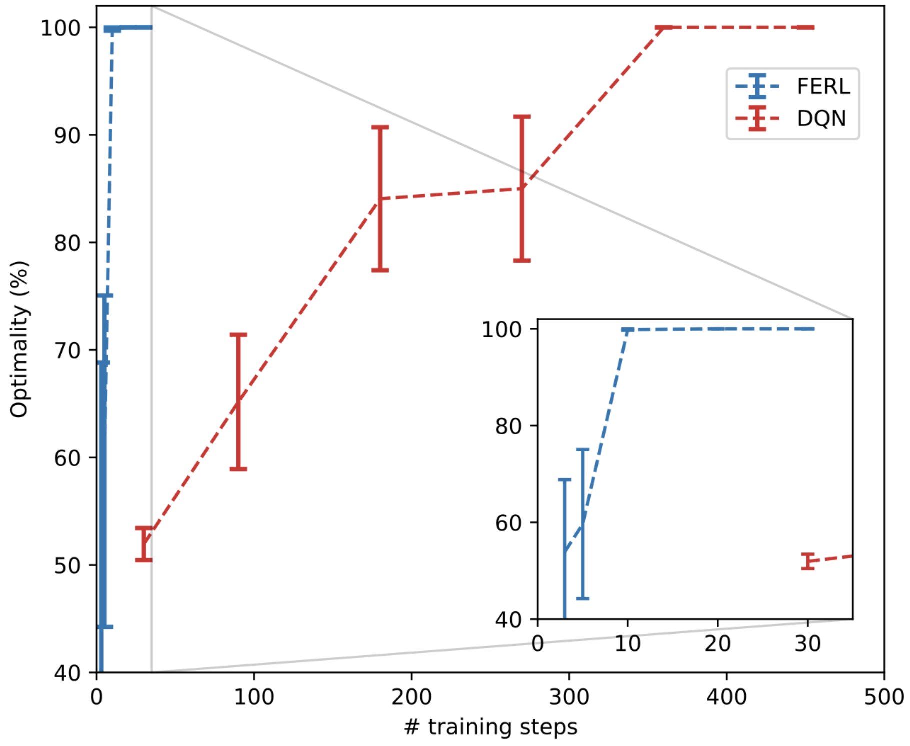
~120 steps, batch size: 7



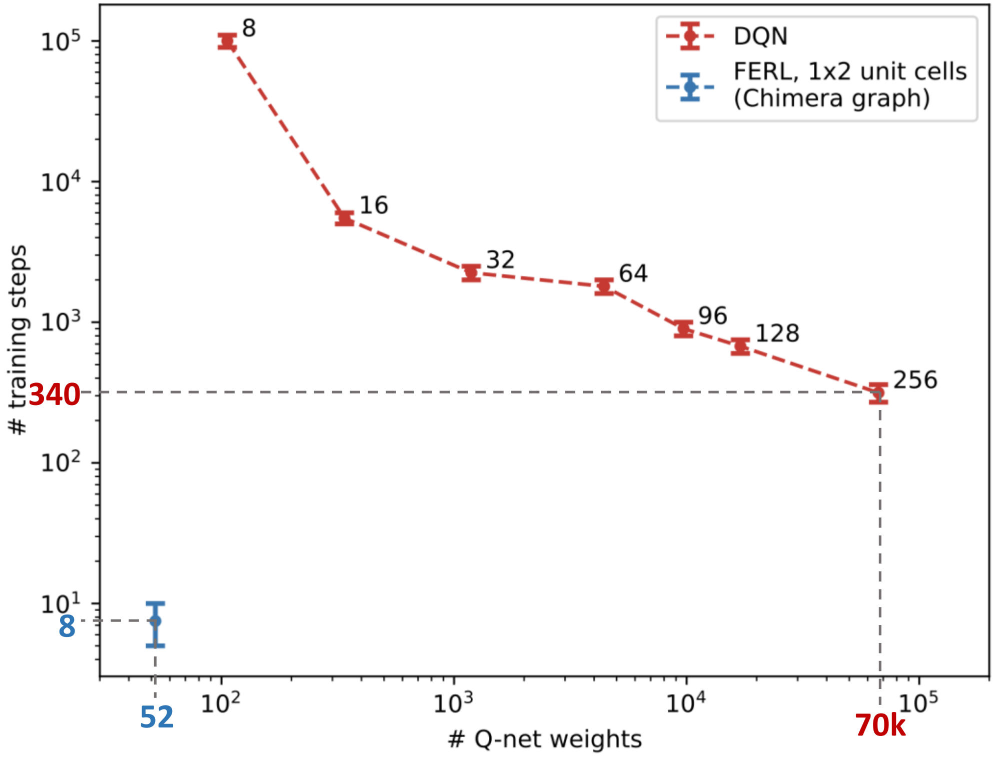
Use case I: Q-learning on 1D beam steering

Training efficiency & descriptive power

Training efficiency



Training efficiency vs. # Q-net / QBM weights



- **Optimality metric:** "in what fraction of possible states does agent take the right decision"
- **Training efficiency:** FERL massively outperforms classical Q-learning *(8±2 vs. 320±40 steps)*
- **Descriptive power:** QBM can reach high performance with **much fewer weights** than DQN (*52 vs. ~70k*)

Project overview

Objectives

- **Implement FERL** using **simulated quantum annealing** and am **actual quantum annealer** (D-Wave)
- **Extend to continous state-action spaces** for real-world applications: **quantum actor-critic**
- **Compare quantum approach to classical RL** in terms of
    1) **Training efficiency** – *"# steps required to train agent"*
    2) **Descriptive power of QBM** – *"# weights needed"*

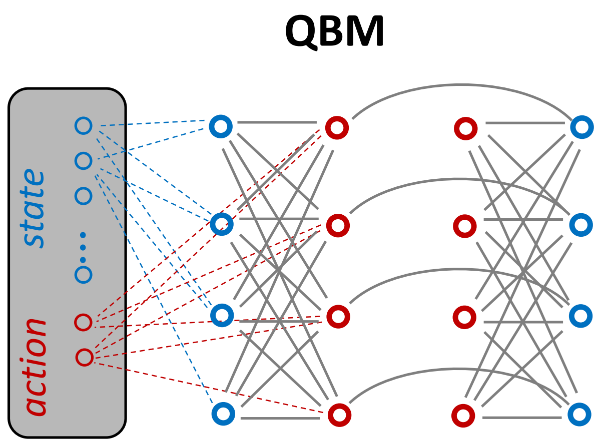
Use case I: Q-learning on 1D beam steering model *(simulated environment)*

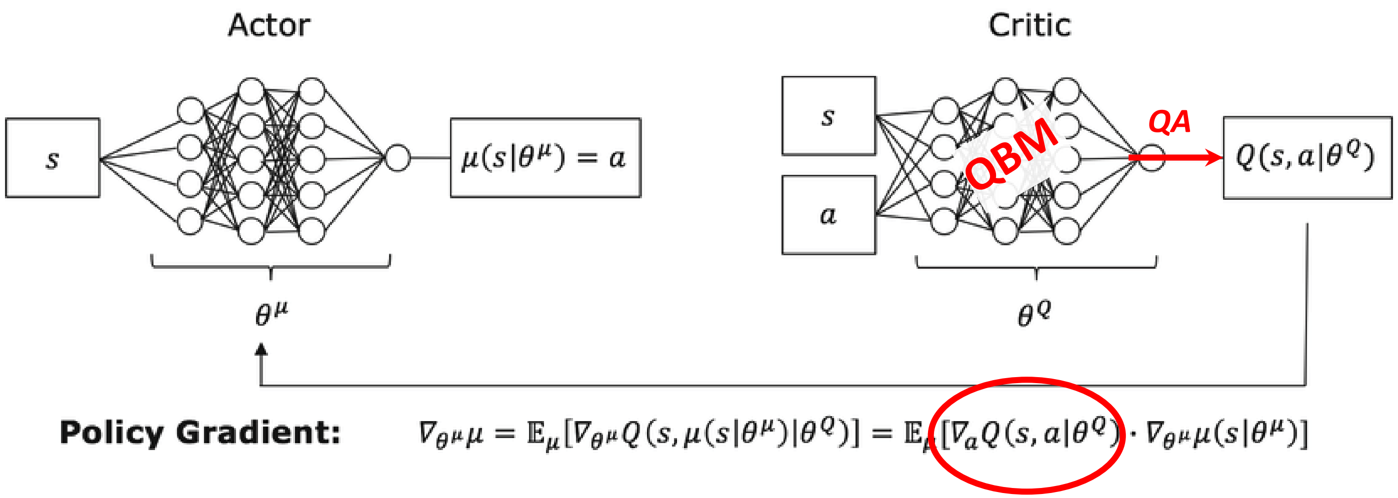
Use case II: quantum actor-critic on 10D AWAKE beam line *(simulated and real environment)*

Developing the quantum actor-critic

Quantum DDPG

- **FERL for continous state-action spaces to tackle real-world problems:** inspired by classical actor-critic methods
- **Why use FERL in combination with classical policy network?**
  - **QBM has ideal structure** to replace classical critic
  - Can we benefit from **high training efficiency of QBM** (?!)
    Intuitively: if critic learns faster, should be beneficial for actor training



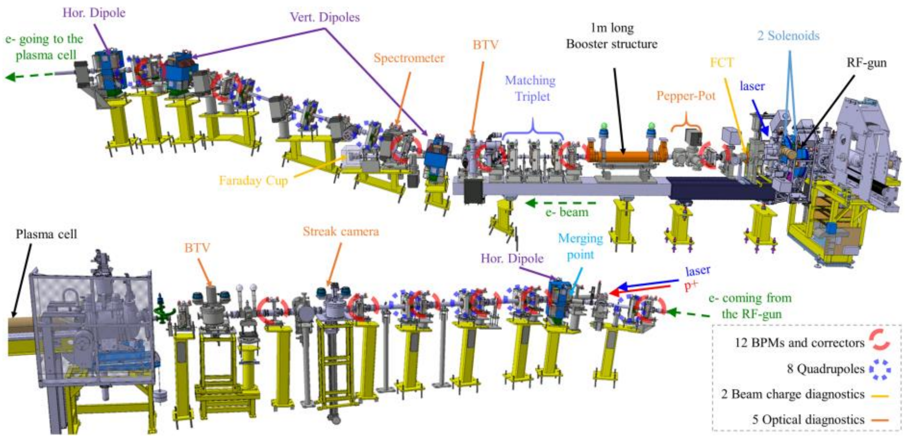


Main challenge

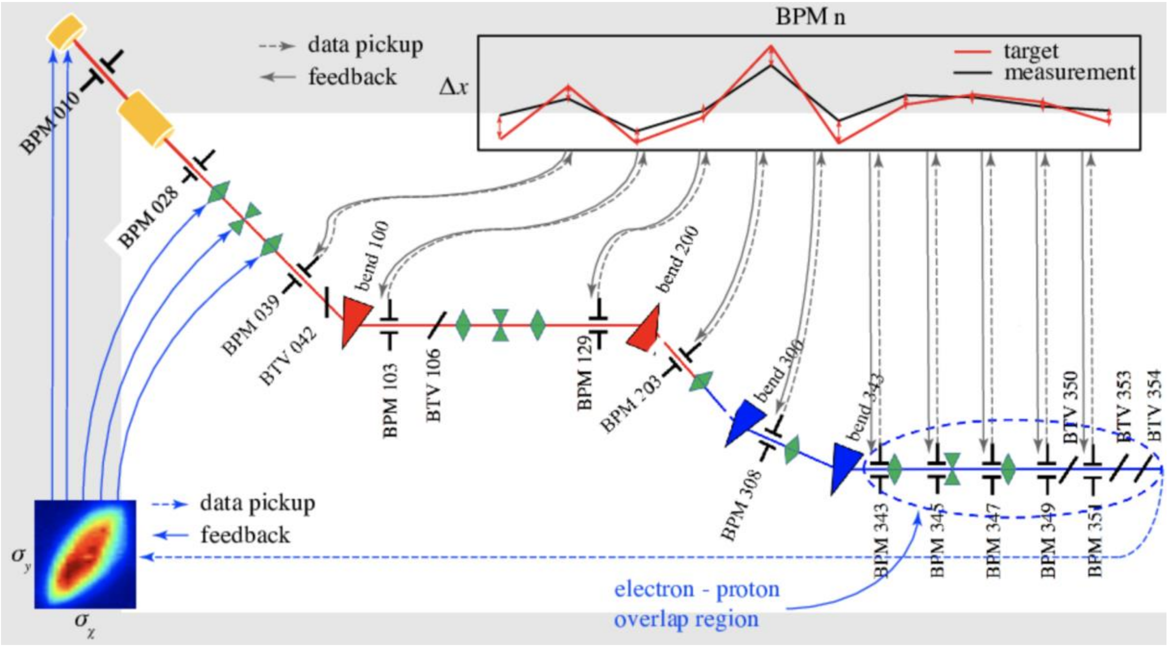
- Calculating derivative of critic wrt. action $\nabla_a Q(s,a|\theta^Q)$
- Numerical (finite difference) or semi-analytical derivative options

Use case II: Q-learning on 10D AWAKE beam line

Environment



- **AWAKE electron beam line**
  https://gitlab.cern.ch/be-op-ml-optimization/envs/awake
- **OpenAI gym template**
- **Action:** deflection angles at 10 correctors *(continous)*
- **State:** beam positions at 10 BPMs *(continuous)*
- **Reward:** negative rms from 10 BPMs

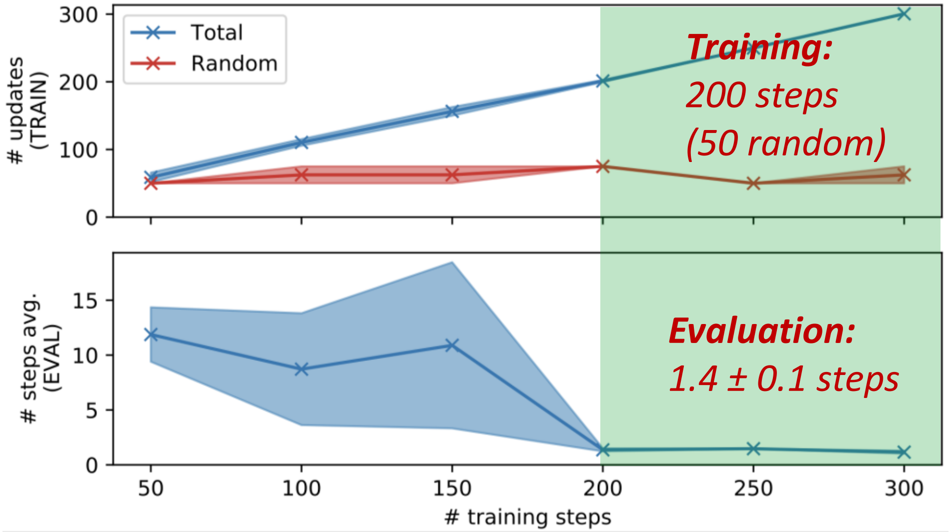


Credits: A. Scheinker

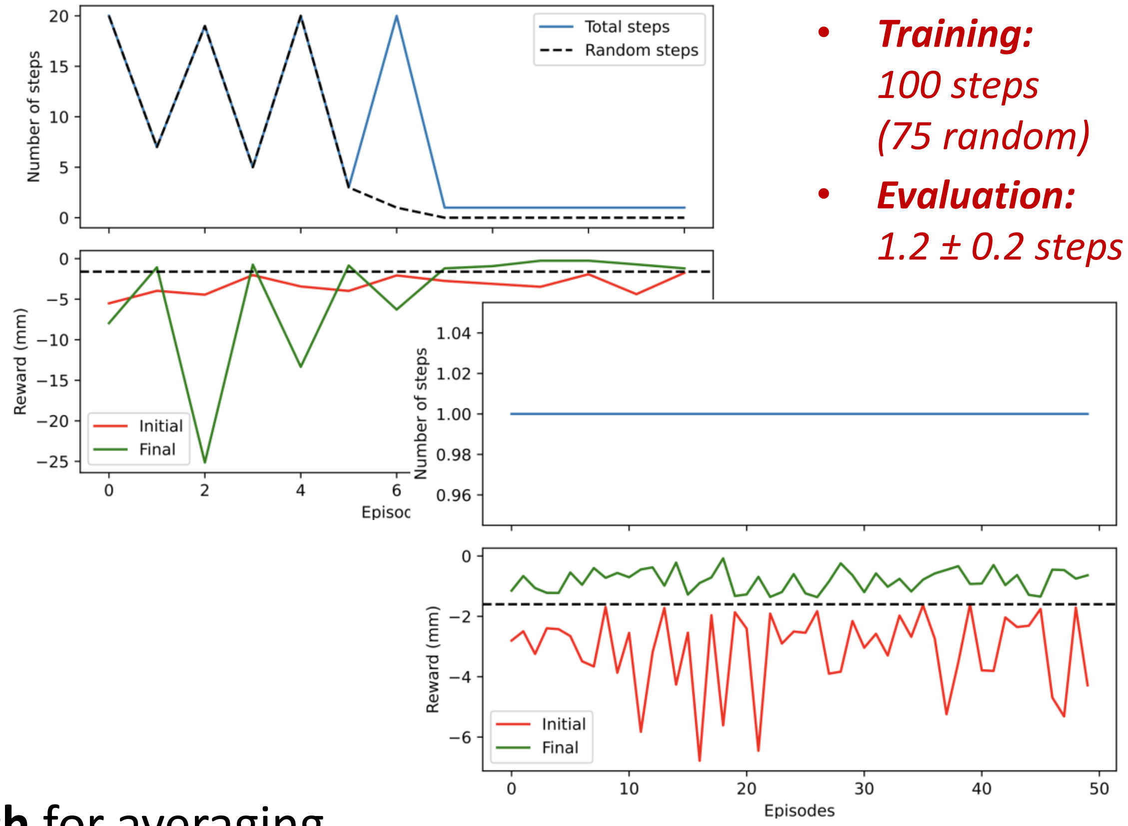
Use case II: Q-learning on 10D AWAKE beam line

Classical vs. quantum actor-critic: training efficiency

Classical actor-critic



Quantum actor-critic

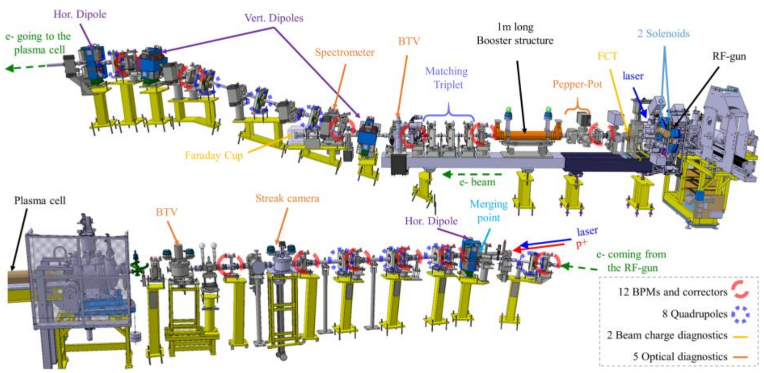


- ***Training:***
  100 steps
  (75 random)
- ***Evaluation:***
  1.2 ± 0.2 steps

- Running **5 trainings and evaluations from scratch** for averaging
- Showing current best performance, **yet to finish hyperparameter optimization** for both
- **Quantum actor-critic is ahead, but the race is still on …**

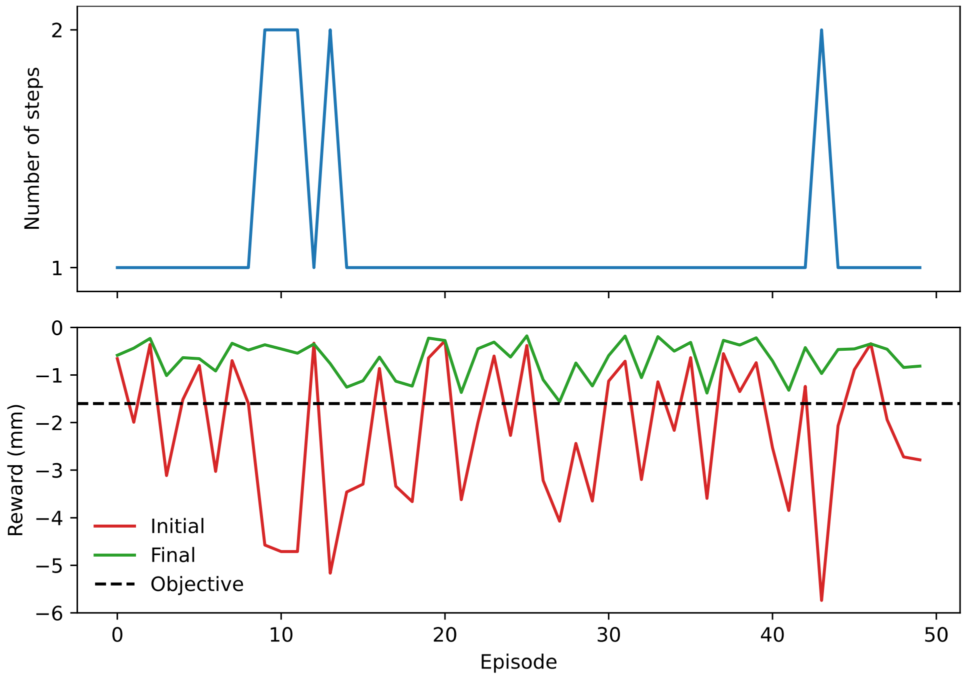
Use case II: Q-learning on 10D AWAKE beam line

Test on actual AWAKE beam line

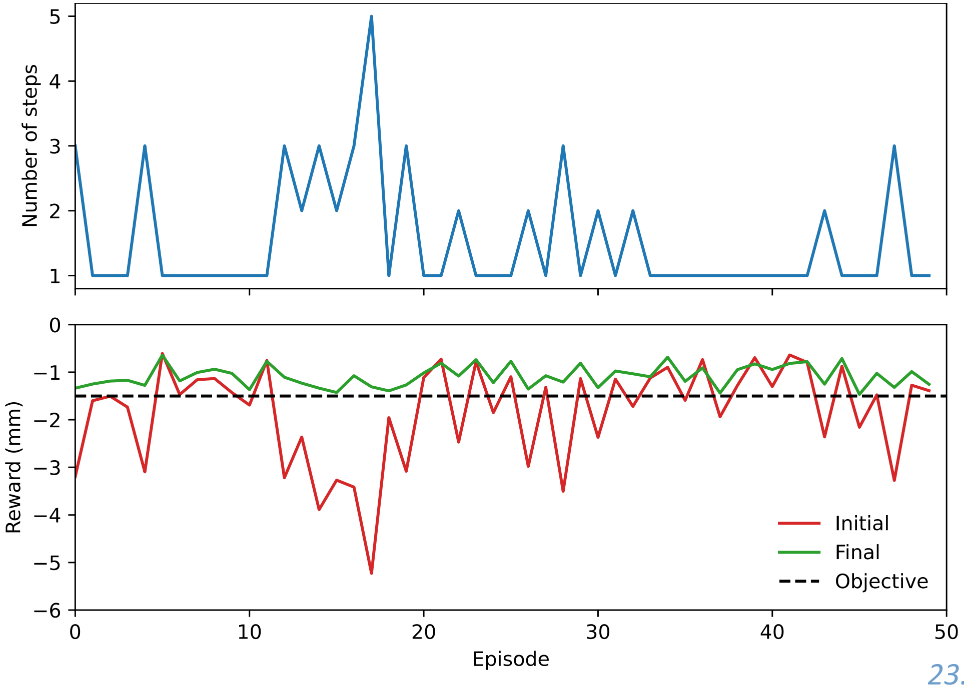


- **Trained and tested our quantum actor-critic agent** on *simulated* 10D AWAKE beam line
- **Deployment on *real* beam line => agent works successfully ☺ !**
  Even with 1 broken beam position monitor (BPM) …
- **Will redo with optimized agent and fixed BPM**

Evaluation on simulated beam line



Evaluation on real beam line



Summary

- **FERL works both with simulator and on quantum annealing hardware**
- **Developed new quantum actor-critic algorithm** that performs well and solves 10x10D **continuous state-action** problem both in **simulated and real environments**
- See **advantage** in terms of **sample efficiency *and* descriptive power** for all cases studied
- **More studies on D-Wave annealer planned**
- Attempt training in **more complex environment**

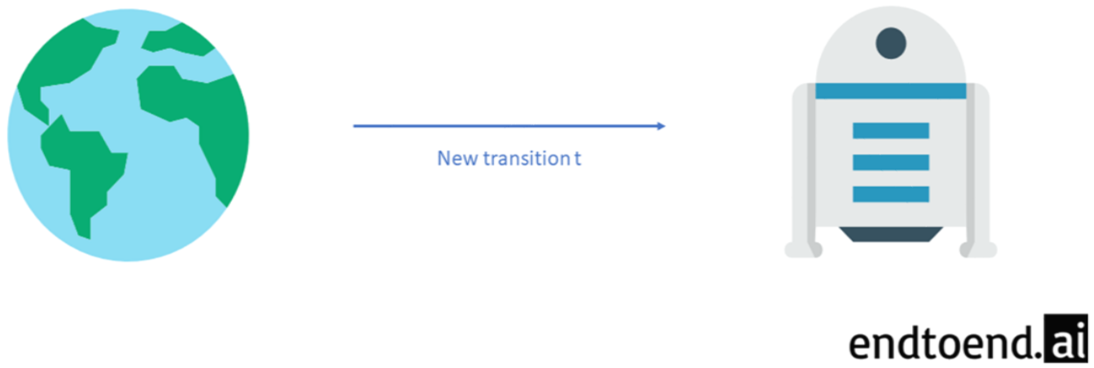
Thank you !

Backup

Introduction
How to learn from training samples

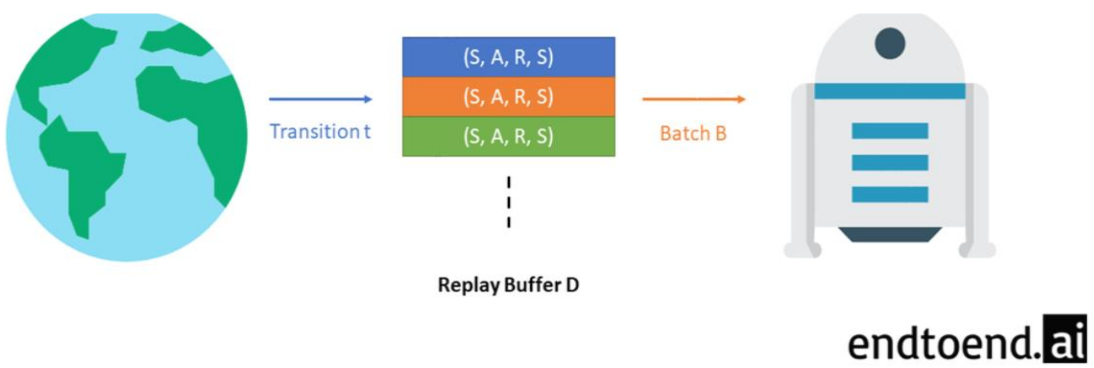
Online Learning

- Learn directly and only from **latest experience**
- Highly correlated data
- Agent learns from each interaction **once and discards** it immediately



Experience Replay

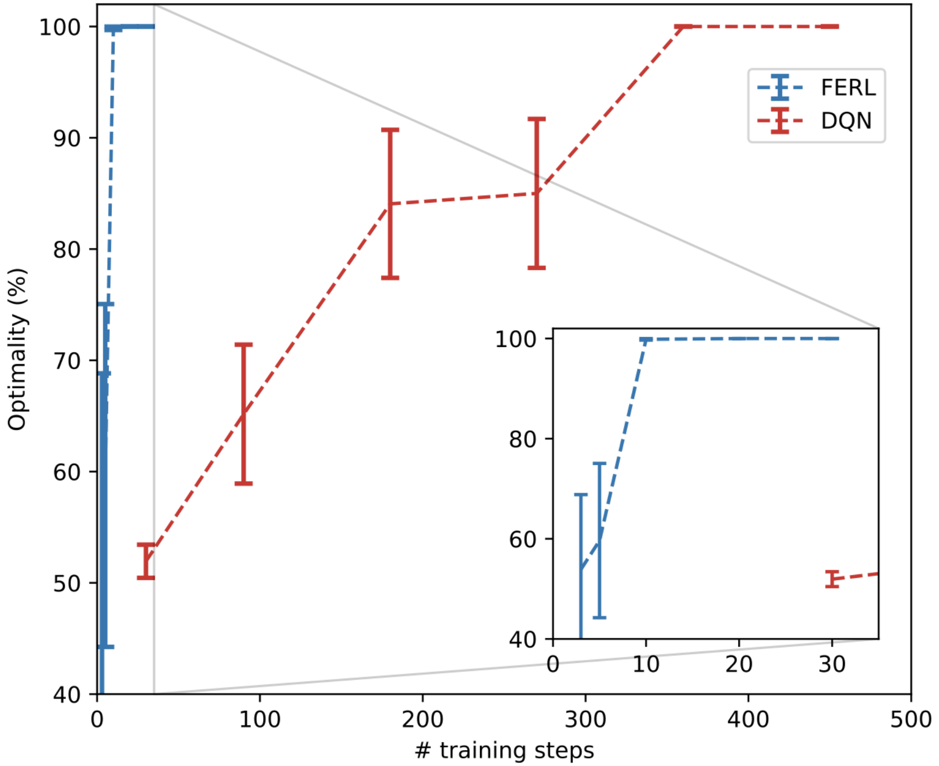
- Save transitions into **memory buffer**
- **Sample batch** from buffer to train agent on **multiple past training samples** at every step



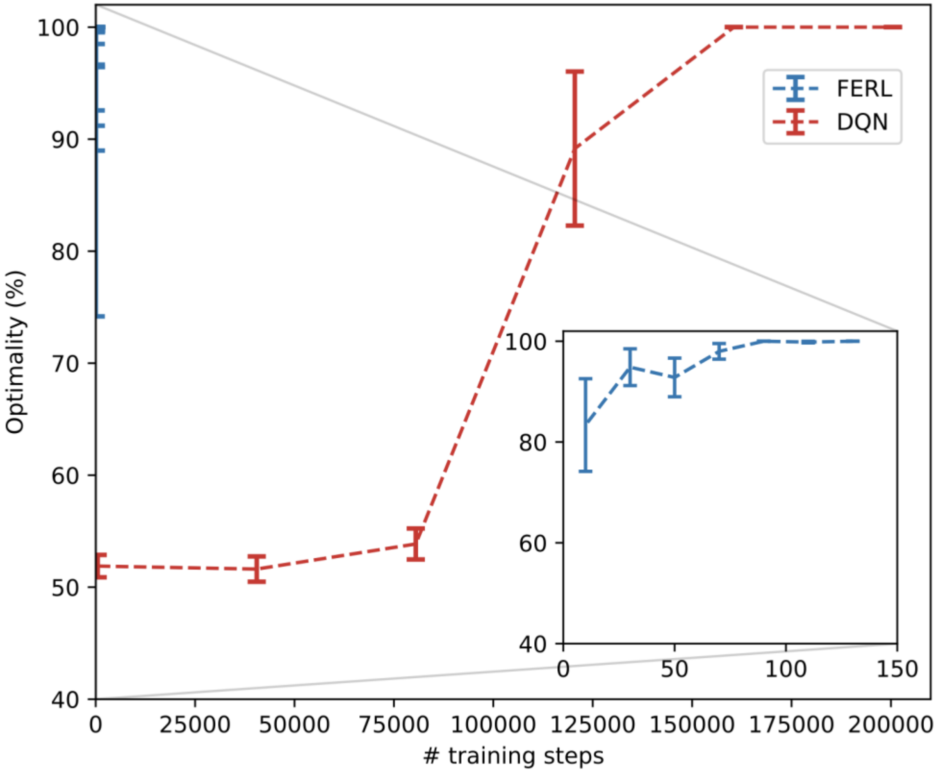
https://www.endtoend.ai/paper-unraveled/cer/

Part I: Q-learning on 1D beam steering
Sampling efficiency

Experience replay ON



Experience replay OFF



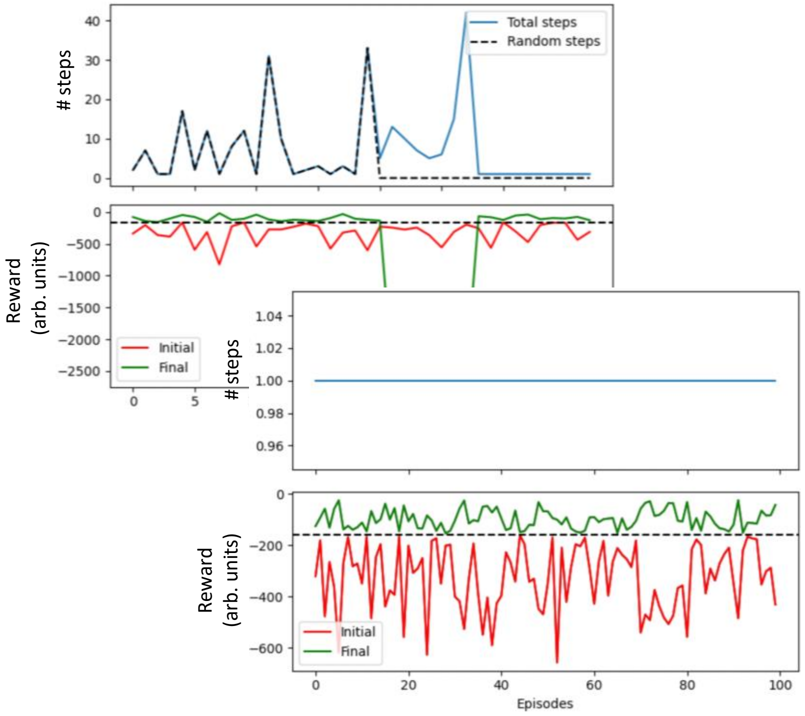
- **Optimality:** “in what fraction of possible states does agent take the right decision”
- **FERL massively outperforms classical DQN:** 10 vs. 360 steps (ER), 90 vs. 160’000 steps (no ER)
- **Required # weights** QBM vs. Q-net is also **completely different!**

Part II: Q-learning on 10D AWAKE beam line

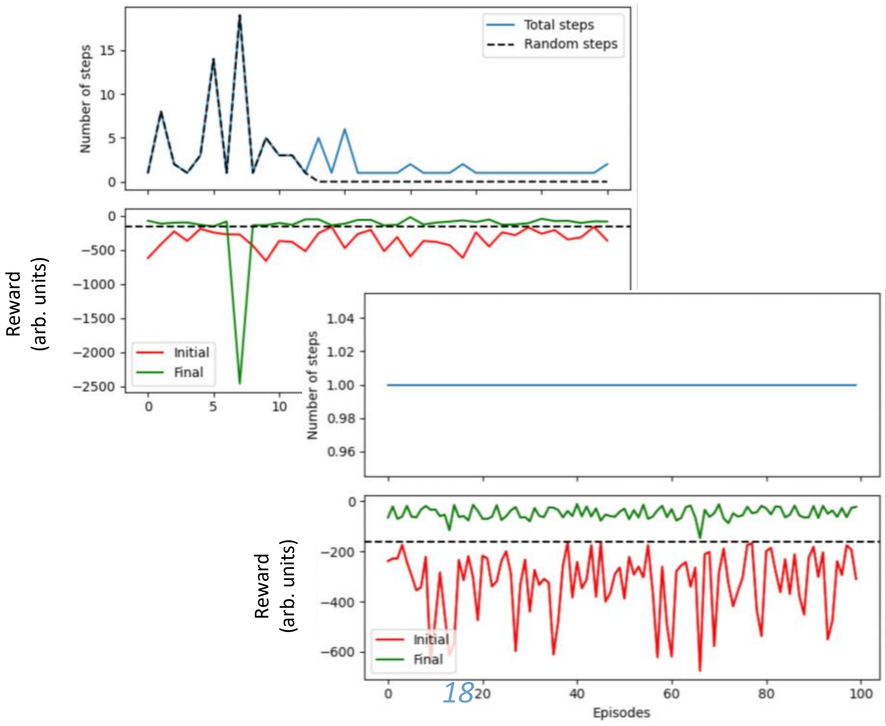
Quantum DDPG

- **Once issue fixed worked immediately really well ☺ !!!**
- **Every training is a success**, sometimes with a few more or less evaluation steps
- **QBM critic can be very small** and still produce good performance
- **Here: unoptimized. Hyperparameter optimization will bring performance well up …**

1x2 QBM: 300 + 110 steps

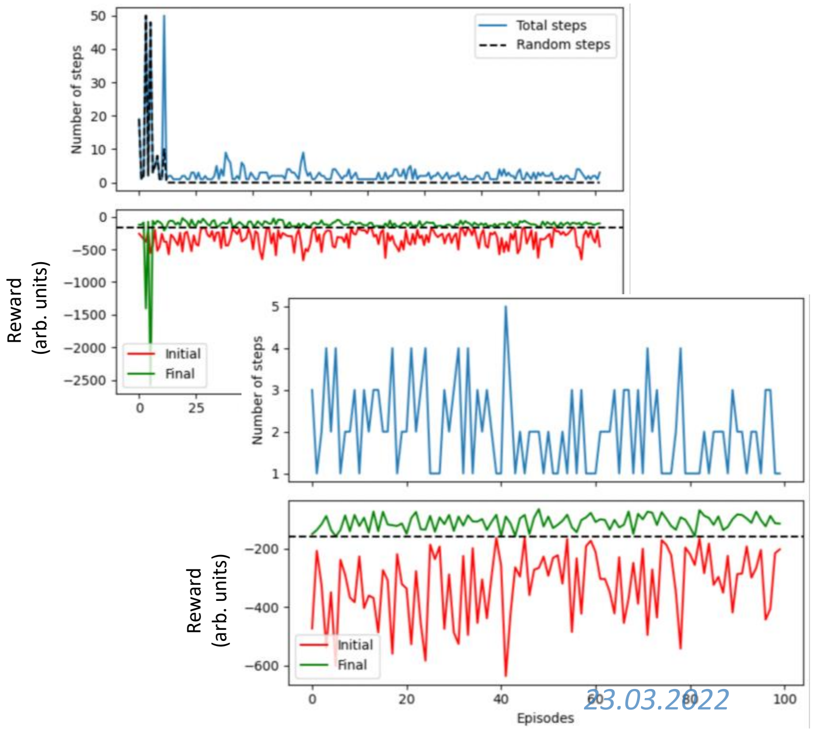


3x3 QBM: 150 + 35 steps



"Worst case"

3x3 QBM: 150 + 451 steps



Part II: Q-learning on 10D AWAKE beam line

Classical vs. quantum DDPG: # critic weights

- **Following numbers are valid for 6D env** *(yet to rerun for 10D env)*
- **Classical DDPG**
  - Best compromise between # training updates vs. # evaluation steps
  - Critic with: **400 x 300 x 1 nodes**, i.e. **123k+ weights** *(see backup)*
- **QBM**
  - Best performance to date with **4 x 4 unit cells, 8 qubits** each
  - **Not fully connected:** following D-Wave 2000Q Chimera topology
  - **Total number** of hidden-hidden (352) + visible-hidden (768) **weights: 1’120**

Factor 100 difference in # critic weights needed

actor networks are identical