Model based reinforcement learning at FERMI FEL

Simon Hirlaender
Univertsity of Salzburg, Salzburg, Austria

Niky Bruchon
University of Trieste, Trieste, IT
(Dated: November 17, 2020)

In this paper we discuss a model based reinforcement learning approach in comparison to a model free reinforcement learning approach applied at the FERMI FEL system. Both algorithms and approaches are new in this context and the main purpose of this paper is to show that reinforcement learning can be applied on an operational level in a feasible training time on real accelerator physics problems. In terms of sample-complexity the model-based approach is faster, while the finale performance of the model free method is superior - the so called asymptotic performance. The model-based algorithm is done in a Dyna-style using an uncertainty aware model and the model-free algorithm is based on tailored deep Q-learning using some tricks to increase the sample efficiency

- Noise benchmarks with pendulum
- Benchmark NAF2 with pendulum
- What is our improvement concerning the algorithm
- The first time model based on accelerator problem compared to a novel state of the art MFRL: NAF2
- Tailored to accelerators - short horizons are standard?
- Improvements of the algorithms:
  - Adding the noise feature
  - Jumping feature using SAC
  - Noise in the policy

I. INTRODUCTION AND MOTIVATION

In particle accelerators one main goal is to provide stable and reproducible performance. In order to achieve this, a number of control problems have to be considered simultaneously. Especially if there is no way to model the physics, one might use optimisation techniques as e.g. derivative free optimisers (DFOs) or model-based optimisations as Gaussian processes. Another promising way is to apply reinforcement learning which has several advantages over optimisation methods:

- Covers a larger class of problems.
- Solves the problem not always completely from the beginning as an DFO.
- Ability to use existing data.
- The underlying structure of the problem might be deduced.

RL optimises a sequence of decisions. On critical aspect using RL is the number of iteration to train a controller. In this paper, we present the study carried out to solve the maximisation-problem of the radiation intensity generated by a seeded Free-Electron Laser (FEL) on the Free Electron laser Radiation for Multidisciplinary Investigations (FERMI) at Elettra Sincrotrone Trieste. Three algorithms were applied successfully showing different advantages, which will e discussed in the following.

In a seeded free-electron laser one of the most critical parameters is the temporal and transverse overlap of the electron and laser beam in the magnetic section called modulator undulator.

II. THE PHYSICAL SET-UP

At FERMI several beam-based feedback systems are deployed to control the beams trajectories shot to shot with the main purpose of guaranteeing a steady intensity of the light radiation. Nevertheless, in the last years various approaches have been studied to investigate their applicability in tuning operations. The paper is organised as follows.

- Description of the problem set-up at FERMI
- Overview of RL
- Details of the implementations used in these studies and theoretical concerns
- Results
- Summary and outlook

III. DEEP REINFORCEMENT LEARNING

Assume states s the set of all states $\mathbf{s}$ and actions $\mathbf{a}$ all actions $\mathcal{A}$ and rewards r and the set of all rewards

$\mathcal{R}$ an initial state distribution d_0. Goal of reinforcement learning is to find a $\pi \mapsto a$, which is the solution of:

$$\max J(\pi) = \mathbb{E}_{\tau\sim p_\pi}\left[\sum_{t=0}^{H}\gamma^t r(\mathbf{s}_t,\mathbf{a}_t)\right], \tag{1}$$

where $p_\pi = d_0(\mathbf{s}_0)\prod_{t_0}^{H}\pi(\mathbf{a}_t,\mathbf{s}_t)T(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)$ is the distribution of all trajectories $\tau := (\mathbf{s}_0,\mathbf{a}_0,\mathbf{s}_1,a_1,\dots s_H,a_H)$ drawn by π with a horizon H. In the modern field of RL one distinguishes if the policy $\pi(\mathbf{a}) \approx \pi_\phi(\mathbf{s})$ or the state-action value function $Q(\mathbf{s},\mathbf{a})$ is approximated using a high capacity function approximator, as e.g. a deep neural network. In the first case one speaks about policy gradient methods in the latter about approximate dynamic programming, which we now discuss.

A. Approximate dynamic programming

The state-value function $Q(\mathbf{s},\mathbf{a})$ is expressed as $Q_\theta(\mathbf{s},\mathbf{a})$, where θ denotes the parameters of the function approximator. By satisfying the Bellmann-optimality operator the optimal $Q^*(\mathbf{s},\mathbf{a})$ can be found:

$$\min_\theta\left(\vec{Q}_\theta - \mathcal{B}^*\vec{Q}_\theta\right)^2. \tag{2}$$

This operator has a unique fixed point but is non-linear, due to the max - operator:

$$\mathcal{B}^*\vec{Q}_\theta(\mathbf{s}_t,\mathbf{a}_t) := r(\mathbf{s}_t,\mathbf{a}_t) + \gamma\max_{\mathbf{a}}\left(Q_\theta(\mathbf{s}_{t+1},\mathbf{a}) - Q_\theta(\mathbf{s}_t,\mathbf{a}_t)\right). \tag{3}$$

The form of this equation can cause overestimation and other complications, when using a function approximator. Several methods exist which try to mitigate the problems []. One way to reduce the effect is to take a simple analytical form of the Q-function.

B. NAF

If a specific quadratic form of the Q function is assumed:

$$Q_\theta(\mathbf{s},\mathbf{a}) = -\frac{1}{2}(\mathbf{a}-\mu_\theta(\mathbf{s}))P_\theta(\mathbf{a}-\mu_\theta(\mathbf{s}))^T + V_\theta(s). \tag{4}$$

The policy is given by:

$$\pi_\theta(\mathbf{s}) = (\delta(\mathbf{a}) - \operatorname{argmax}_{\mathbf{a}} Q_\theta(\mathbf{s},\mathbf{a})). \tag{5}$$

One modification, which is used in these tests is a twin network (weights denoted by $\tilde{\theta}$), where only one is used to obtain the policy, while the other is used for the update rule to avoid over-estimation. The maximum is given analytically as $\max_{\mathbf{a}} Q(\mathbf{s},\mathbf{a}) = V(\mathbf{s})$, hence from eq. (2):

$$\min_\theta\left((r(\mathbf{s}_t,\mathbf{a}_t) + \gamma\min_{1,2} V_{\theta^i_{\text{targ}}}(\mathbf{s}_{t+1}) - (1+\gamma)Q_\theta(\mathbf{s}_t,\mathbf{a}_t)\right)^2 \tag{6}$$

θ_{targ} are the weights of a target network, which is softly updated. To stabilize the network training a small artificial noise is added to the actions in the update. This algorithm has an extremely good sample-efficiency for suitable problems as can be found in accelerators and is used as the baseline for the considered control problem.

Algorithm 1: On-policy policy gradient with Monte Carlo estimator

1: initialise θ_0 **for** *iteration* $k \in [0,\dots,K]$ **do**
2: sample trajectories $\{\tau_i\}$ by running $\pi_{\theta_k}(\mathbf{a}_t|\mathbf{s}_t)$
▷ each τ_i consists of $\mathbf{s}_{i,0},\mathbf{a}_{i,0},\dots,\mathbf{s}_{i,H},\mathbf{a}_{i,H}$

IV. UNCERTAINTY AWARE DYNA-STYLE REINFORCEMENT LEARNING

The original Dyna algorithm [11] is modified here in several aspects. Generally, Dyna style algorithms denote algorithms, where a MFRL algorithm is trained on purely synthetic data from an approximate dynamics model or on a mixture of synthetic and real data. We use only synthetic data to reduce the interaction with the real environment to a minimum. An overview of the used method is shown in fig. 1. At the beginning the data is collected or read in from a pre-collection. An uncertainty aware model is trained, using anchored ensembles on the data, which allows to take the allegorical (measurement errors) as well as the epistemic uncertainty (lack of data) into account. Subsequently a MFRL algorithm is deployed to learn the controller on the learned model by only using the synthetic data, by taking the uncertainty into account. After a defined number of training steps the controller is testes on each model individually. If there is no improvement in a specific ratio<1 of models, the controller is tested on the real environment for a number of episodes. The training is stopped if the controller solves the problem on the real environment.

A. Critical design decisions

In the following the most important aspects of a successful application of a Dyna-AE algorithm are listed:

- The ANN - including the prior - number of models - noise level - early stopping
- The number of data points and the policy.
- The uncertainty - how is this included?
- The episodic design - avoid long trajectories.
- The MFRL agent as e.g. the TRPO and PPO or the SAC and its training
- Tuning the algorithm on a model.

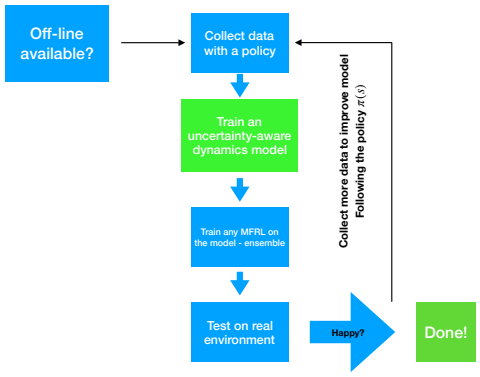

FIG. 1: A schematic overview of the AE-Dyna approach used in this paper.

We try to discuss all items in some detail in the following. The anchoring ensemble methods usually yields good results already with a small number of different networks. The idea behind it, is to mimic the posterior probability of the dynamics model to capture first of all its own uncertainty due to the lack of data. Empirical results show that three to five models were sufficient to see clear advantages over a single network approach and the main goal is not to calculate the exact posterior of the model. A small two-layer network with around 15-25 nodes and the *tanh* activation were used. The last layer was linear and the inputs were normalized to the interval [-1,1]. The prior is controlled by the initialization of the weights, which were normalized by the number of nodes in each layer. The noise level is also respected during the initialization. For the moment only homeoscetastic errors are used. Several methods to set the training method were implemented. Early stopping, a standard techniques showed good results, mainly to the fact, that the number of training steps is increasing with more data. The advantage is that the uncertainty at regions without data shrink in this way leading to a better training performance. A fixed loss value threshold was also tested, as also used in the original anchore-ensemble technique. In the experiments a combination of both was taken.

One of the most crucial points is how this uncertainties are taken into account by the RL algorithms. Several different approaches were tried. It is possible to take the average of the models and add Gaussian noise leveled by the standard deviation of the models. Another straight forward way is to randomly select a model to provide at each training step, which is the original implementation of the ME-TRPO algorithm and was used in the experiments labeled *ME-TRPO*. A pessimistic setting only would select the model resulting in the lowest predicted reward. Good results were obtained by following a randomly selected single model each full episode and were used in the experiments labeled *AE-Dyna*.

The number of data-points taken every time the model as improved was firstly determined by the number of initial data points, collect using a random policy. The initialization phase has to be chosen not to small to minimize the chance of getting trapped in a local minimum for too long. Afterwards at least on full episode should be taken. We decided to use a short total episode length to reduce the impact of the compound error, the accumulation error following a wrong model, as well known for Dyna-style approaches. The maximal number of steps was ten. During the data collected on the real system, the latest policy was taken with some small noise added to improve the exploration.

To decide which MFRL algorithm to use, two main algorithm classes have to be considered: the on and the off-policy algorithms. Online algorithm show a more stable convergence in general, while off-policy algorithms have the advantage that the data buffer can be filled with real data as well. A very attractive off-policy algorithm is the Soft-actor-critic. This algorithm tries to maximize the entropy of the actions to find a good trade-off between exploration and exploitation. On the other hand the on-policy algorithm TRPO provides some theoretical improvement guarantees. In the latter case the policy was improved over the whole training, while for the SAC the policy was reset, to take advantage of the exploration features.

B. Operational deployment

Niky

C. short FERMI introduction

The Free Electron laser Radiation for Multidisciplinary Investigation better known as the FERMI, is the seeded free-electron laser facility next to the third-generation synchrotron facility ELETTRA at the Elettra Sinchrotrone Trieste laboratories. A free-electron laser is a fourth-generation light source where the lasing medium consists of very-high-speed electron moving freely through a magnetic structure. The FERMI peculiarity is given by the usage of an external seeding source that provides several advantages, as the increased stability in pulse and photon energy, reduced size of the device, improved longitudinal coherence, more control on the pulse structure, and improved temporal synchronization with external timing systems.

The external optical laser signal provide is contribution to the FEL process in the modulator where it interacts with the relativistic electron beam modulating it in energy. The modulation in energy is the converted into a charge modulation in the dispersive section, and finally the density modulated beams radiation is amplified in the radiators section. The importance of ensuring the

best possible overlapping between the seed laser and the electron beam in the modulator is therefore evident.

For this reason the proposed study focuses on the control of the seed laser trajectory in the modulator, looking at FERMI performance as a reference.

D. our system: the modulator and the seed laser

A more detailed description of the alignment process in the modulator is here provided. The most critical parameters in a seeded free-electron laser are the temporal and transverse overlap of the electron and laser beams in the modulator magnetic section. The problem is simplified by keeping constant the trajectory of the electron beam and the mechanical delay line that controls the temporal alignment. The final problem faced consists in optimising the seed laser trajectory to match the electron beam and consequently increase the intensity of the FEL radiation.

V. PROOF-OF-PRINCIPLE APPLICATION OF RL AGENT AT FERMI TRAJECTORY CORRECTION

Niky

A. Experiment results from FERMI RL tests

Niky and Simon

VI. DISCUSSION AND OUTLOOK

VII. CONCLUSIONS

Appendix A: A non-linear standard control problem

To provide some transparency of these studies we provide results on a famous classical standard control problem, the inverted pendulum. It is a non-linear low dimensional unsolved continuous control problem, which means there is no threshold for the reward to terminate an episode. The episode length is set to 200 steps.

1. The impact of noise

Appendix B: NAF2 details

Appendix C: AE training details

Appendix D: Theoretical aspects on the AE-DYNA on FERMI FEL

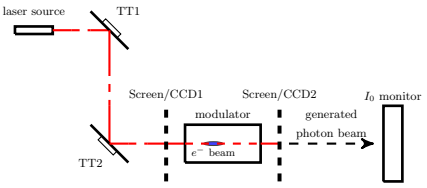

FIG. 2: The training .

[1] D. Jacquet, R. Gorbonosov, G. Kruk, "LSA - The High Level Application Software of the LHC and its Performance during the first 3 years of Operation", ICALEPS, San Francisco, CA, USA, 6 - 11 Oct 2013, pp.thppc058.

[2] A. Edelen, N. Neveu, M. Frey, Y. Huber, C. Mayes, and A. Adelmann, "Machine learning for orders of magnitude speedup in multiobjective optimization of particle accelerator systems", Phys. Rev. Accel. Beams 23, 044601, 2020.

[3] G. Azzopardi, A. Muscat, G. Valentino, S. Redaelli, B. Salvachua, "Operational results of LHC collimator alignment using machine learning", Proc. IPAC'19, Melbourne, Australia, pp. 1208-1211, 2019.

[4] S. Hirlaender, M. Fraser, B. Goddard , V. Kain, J. Prieto, L. Stoel, M. Szakaly, F. Velotti, "Automatisation of the SPS ElectroStatic Septa Alignment", in 10th Int. Partile Accelerator Conf.(IPAC'19), 2019, p. 4001-4004.

[5] E. Shaposhnikova et al., "LHC Injectors Upgrade (LIU) Project at CERN", 7^{th} International Particle Accelerator Conference, Busan, Korea, 8 - 13 May 2016, pp.MOPOY059, https://doi.org/10.18429/JACoW-IPAC2016-MOPOY059.

[6] E. Adli, A. Ahuja, O. Apsimon, R. Apsimon, A.-M. Bachmann, D. Barrientos, F. Batsch, J. Bauche, V. B. Olsen, M. Bernardini *et al.*, "Acceleration of electrons in the plasma wakefield of a proton bunch," Nature 561, 363–367 (2018), https://doi.org/10.1038/s41586-018-0485-4).

[7] A. Scheinker *et al.*, "Online multi-objective particle accelerator optimization of the AWAKE electron beam line for simultaneous emittance and orbit control", AIP Advances 10, 055320 (2020), https://doi.org/10.1063/5.0003423.

[8] R. Sutton, A. Barto, "Introduction to Reinforcement Learning", MIT Press, Cambridge, MA, USA, 2018.

[9] H. Kwakernaak, R. Sivan, "Linear Optimal Control Systems", Wiley-Interscience, 1972.

[10] A. Nagabandi, G. Kahn, R. S. Fearing, S. Levine, "Neural Network Dynamics for Model-Based Deep Reinforcement Learning with Model-Free Fine-Tuning",

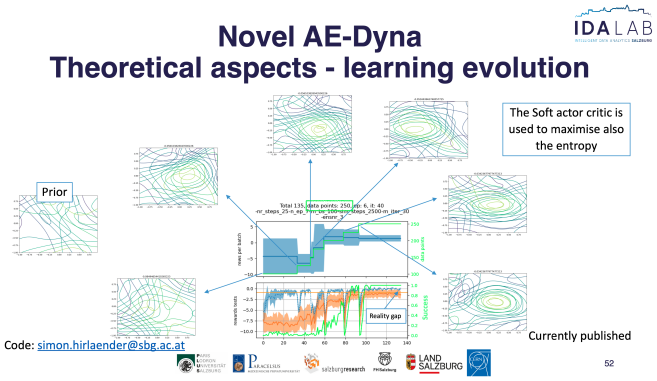

FIG. 3: The training .

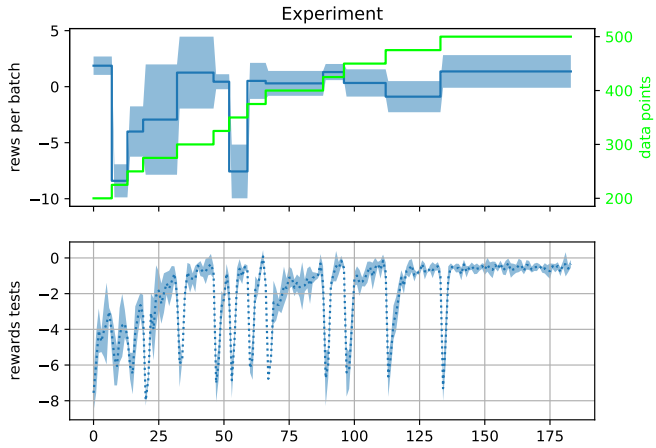

FIG. 4: The training .

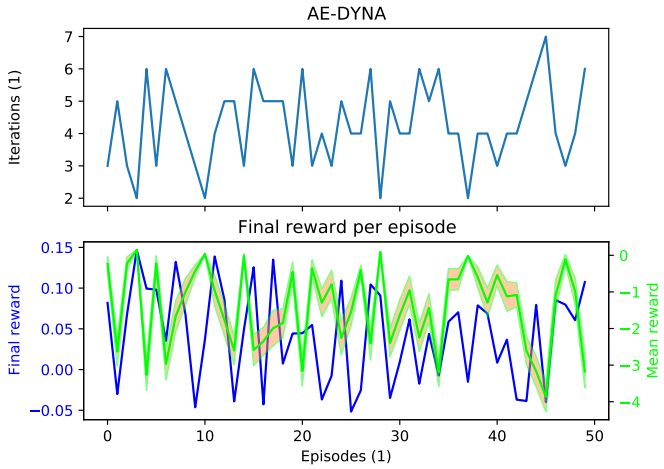

FIG. 5: The training .

arXiv:1708.02596, 2017.

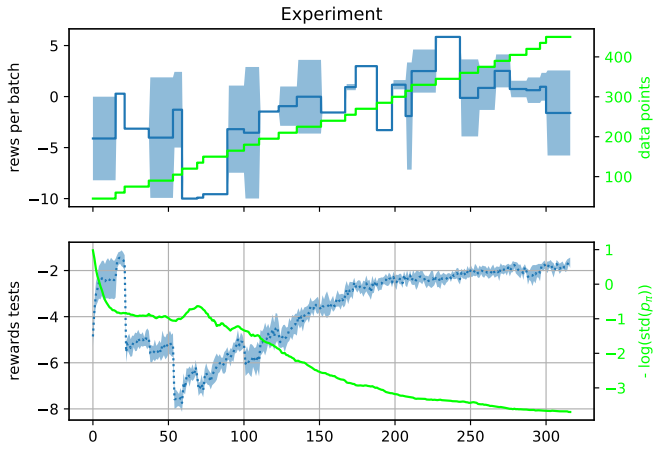

FIG. 6: The training .

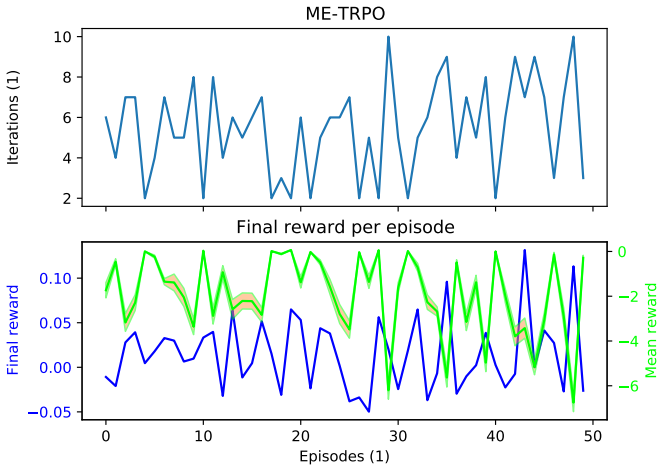

FIG. 7: The training .

[11] R. Sutton, "Dyna, an integrated architecture for learning, planning, and reacting", AAAI Spring Symposium, pp. 151-155, 1991, https://doi.org/10.1145/122344.122377.

[12] T. Wang et al., "Benchmarking Model-Based Reinforcement Learning", https://arxiv.org/abs/1907.02057v1, 2019-07-03.

[13] V. Mnih *et al.*, "Human-level control through deep reinforcement learning", Nature 518, 529–533 (2015).

[14] T. Lillicrap *et al.*, "Continuous control with deep reinforcement learning", Proc. ICLR 2016 https://arxiv.org/abs/1509.02971.

[15] S. Gu, T. Lillicrap, I. Sutskever, S. Levine, "Continuous Deep Q-Learning with Model-based Acceleration", in Proc. 33rd International Conference on Machine Learning, New York, NY, USA, 2016.

[16] S. Fujimoto, H. van Hoof, D. Meger, "Addressing Function Approximation Error in Actor-Critic Methods", arXiv:1802.09477, 2018.

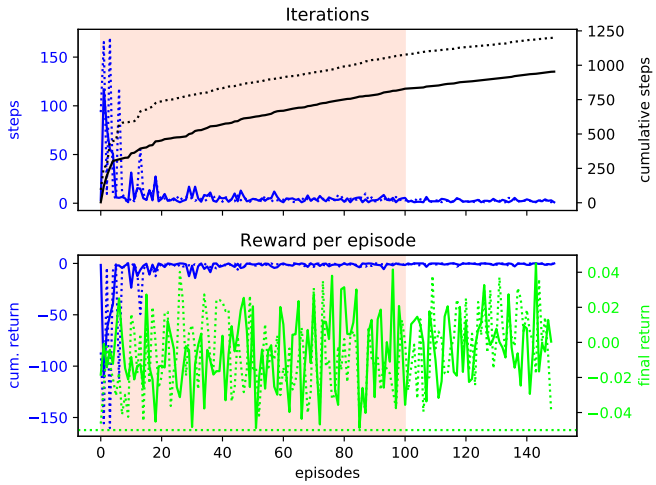

FIG. 8: The training .

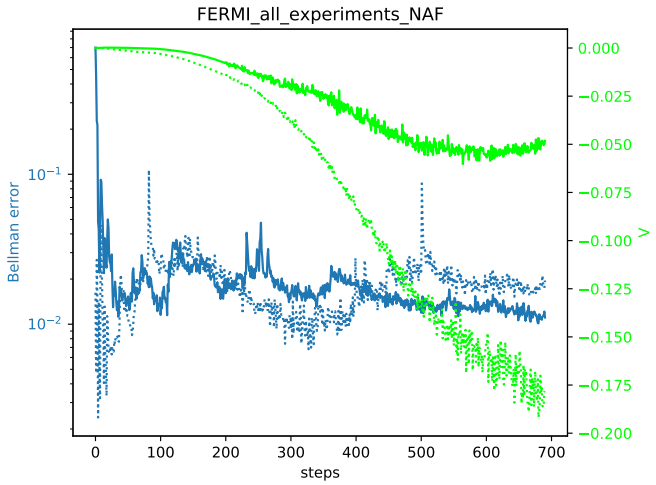

FIG. 9: The training .

[17] Y. Gal, R. McAllister, and C. E. Rasmussen, "Improving PILCO with Bayesian neural network dynamics models", in Data-Efficient Machine Learning workshop, ICML, 2016, vol. 4, p. 34.

[18] T. Schaul, J. Quan, I. Antonoglou, D. Silver, "Prioritized Experience Replay", arXiv:1511.05952, 2015-11-18.

[19] S. Hirlaender, *PER-NAF* available at `https://github.com/MathPhysSim/PER-NAF/` and `https://pypi.org/project/pernaf/`.

[20] "pyjapc" available at `https://pypi.org/project/pyjapc/`.

[21] `http://gym.openai.com`

[22] MAD-X documentation and source code available at `https://mad.web.cern.ch/mad/`.

[23] G. Bellodi, "Linac4 Commissioning Status and Challenges to Nominal Operation", $61^s t$ ICFA Advanced Beam Dynamics Workshop on High-Intensity and High-Brightness Hadron Beams, Daejeon, Korea, 17 - 22 Jun 2018, pp.MOA1PL03, `https://doi.org/10.18429/JACoW-HB2018-MOA1PL03`.

[24] F. Velotti, B. Goddard et al., "Automatic AWAKE electron beamline setup using unsupervised machine learning", to be submitted to Phys. Rev. Accel. Beams.

[25] A. Kumar, A. Gupta, S. Levine, "DisCor: Corrective Feedback in Reinforcement Learning via Distribution Correction," arXiv:2003.07305, 2020-03-16.

[26] A. Kumar, J. Fu, G. Tucker, S. Levine, "Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction," arXiv:1906.00949, 2019-06-03.

[bruchon2020basic]