

Adaptive Model-Based Reinforcement Learning for Orbit Feedback Control in NSLS-II Storage Ring

Zeyu Dong¹, Yuke Tian² and Yu Sun³

Abstract—The National Synchrotron Light Source II (NSLS-II) uses highly stable electron beam to produce high-quality X-ray beams with high brightness and low-emittance synchrotron radiation. The traditional algorithm to stabilize the beam applies singular value decomposition (SVD) on the orbit response matrix to remove noise and extract actions. Supervised learning has been studied on NSLS-II storage ring stabilization and other accelerator facilities recently. Several problems, for example, machine status drifting, environment noise, and non-linear accelerator dynamics, remain unresolved in the SVD-based and supervised learning algorithms. To address these problems, we propose an adaptive training framework based on model-based reinforcement learning. This framework consists of two types of optimizations: trajectory optimization attempts to minimize the expected total reward in a differentiable environment, and online model optimization learns non-linear machine dynamics through the agent-environment interaction. Through online training, this framework tracks the internal status drifting in the electron beam ring. Simulation and real in-facility experiments on NSLS-II reveal that our method stabilizes the beam position and minimizes the alignment error, defined as the root mean square (RMS) error between adjusted beam positions and the reference position, down to $\sim 1 \mu\text{m}$.

I. INTRODUCTION

NSLS-II is a third-generation storage ring producing synchrotron radiation through laser-electron interactions. Electrons are accelerated through a synchrotron and injected into the storage ring. Low emittance in a light source facility requires stable electron beam orbit [1]. Figure 1 shows a simplified electron orbit that remains in the beam position and emits X-ray radiation at multiple X-ray experiment locations.

The beam operators rely on beam monitoring and control systems to interact with orbits. In each unit of the storage ring, beam position monitors (BPMs) measure the relative position of the beam. Each storage ring also includes corrector control to adjust beam dynamically. Ideally, the corrector currents in orbit controls are initialized at the beginning of experiments, and their initialization depends only on the design of the light source facility. In reality, noise and environmental change cause the beam to gradually drift away from the reference position. Orbit feedback control systems

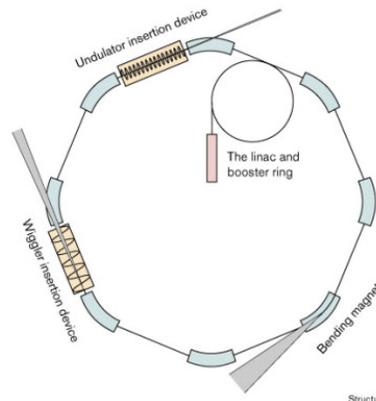


Fig. 1. An illustration of a storage ring [2]

[3] apply corrections to corrector controls and regains the beam reference position.

The orbit response matrix (ORM) reflects how BPMs respond to the correction. The matrix is static and belongs to the original machine design. In practice, beam operators periodically measure the ORM by tuning the beam close to the golden beam position and changing the current setting on one corrector one by one. However, we cannot obtain precise measurement on the ORM because of the following reasons:

- 1) the machine dynamics drifts slightly over time, due to external environment influence, for example, hysteresis of the correctors [4], room temperature;
- 2) Nonlinear beam; the linear approximation of the ORM measured at the previous time incurs a large bias in modeling the beam system that is non-linear and evolves continuously during facility operation;
- 3) Environment factors, including vibration and electronic noise, introduce measurement error (noises) in ORM.

Traditionally, SVD-based algorithms produce feedback signals by filtering out high-frequency components in ORMs [5]. Modern NSLS-II design for fast orbit feedback control combines SVD-based feedback control with other controllers, such as PID (proportional–integral–derivative), to ensure robust and stable beam orbits [6]. However, this method has several limitations in practice. Empirically, a large λ value leads to robust control to the ORM errors while generating biased RMS values. On the other hand, because of the machine’s internal drifting, the controller still has a degenerated performance, even with sufficiently large λ . Figure 2 shows that after a long time, some dimensions will lose control even with very large λ .

¹Zeyu Dong is with the Department of Applied Math and Statistics, Stony Brook University, 100 Nicolls Road, Stony Brook, NY 11794, USA Zeyu.Dong@stonybrook.edu

²Yuke Tian is with National Synchrotron Light Source II, Brookhaven National Laboratory, Bldg. 741, Upton, NY 11973, USA ytian@bnl.gov

³Yu Sun is with Sunrise Technology Inc., 1500 Stony Brook Rd, Stony Brook, NY 11790, NY 11790, USA yu.sun@sunriseaitech.com

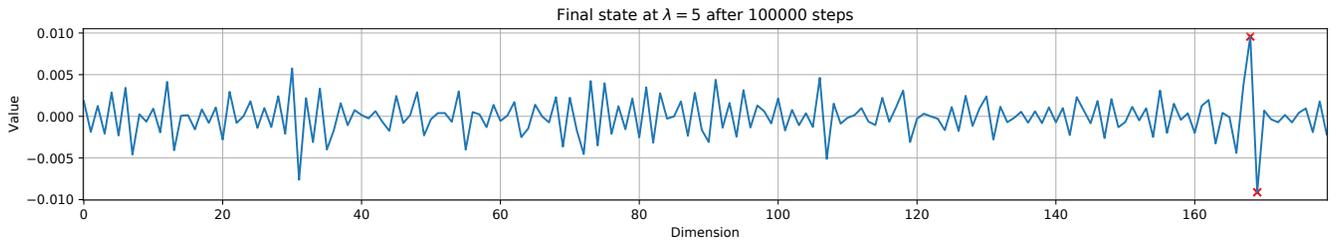


Fig. 2. A degenerated final state after a long run. The dimensions that lose control are marked in red.

We seek an adaptive feedback control system to address this problem. Reinforcement learning (RL) is a good solution for robots, self-driving, optimization and scheduling, and control systems. With agent-environment interaction, the RL agent learns the behaviors of the orbit system and captures any machine drifting. Multi-layer Neural networks (NN) in RL can be trained to model the non-linear dynamics of the orbit system. However, high dimensional control with reinforcement learning is a challenging task. Our feedback system in NSLS-II is high-dimensional and consists of 180 inputs for BPM measurements and 180 outputs of control systems. Therefore, we must use prior knowledge to model the target machine and use the model to regularize RL training and overcome the curse of dimensionality. In this paper, we design and implement an orbit feedback system based on deep reinforcement learning and address the following issues: (i) *System drifting*; (ii) *Degenerated performance with traditional SVD-based linear method*; (iii) *High dimensional control with reinforcement learning*. Our contributions are summarized as follows:

- 1) In trajectory optimization, a model-based RL algorithm optimizes a policy neural network. Trajectory optimization targets the entire control process instead of a single step. Consequently, the trained policy chooses actions to ensure the stability of the future episode. The trajectory sampling process simulates the control process to better fit the actual operation data of the machine. On the other hand, the optimization runs on a differentiable surrogate model with the ideal environment setting (i.e., no noise). This improves policy accuracy and accelerates the training process.
- 2) In online model optimization, the policy network is applied to the environment. Real-time data is collected to train the system model adaptively. Online model optimization targets adaptive control by interacting with the orbit feedback system. This addresses the problem of system drifting. The forward propagation neural network captures the non-linear behavior of the system with high accuracy. Moreover, the training data for model optimization can be efficiently collected during beam daily operations. We do not need extra facility maintenance time for dataset collection.
- 3) We use the existing SVD-based method and the supervised learning model as the baseline and evaluate the model-based reinforcement learning system for the

NSLS-II feedback control. We compare their performance with the simulation environment. Then we conduct real-world experiments in the NSLS-II feedback system for additional evaluations. A neural network with three hidden layers of size 512 is trained to run on the NSLS-II feedback system, having 180 input and output dimensions. Our method control stabilizes the RMS of beam position to $\sim 1 \mu\text{m}$, about 80% improvement compared to the current SVD-based method. We plan to add our RL model to the production beam system and provide it to the operators of the NSLS-II storage ring.

The remainder of this work is organized as follows. Section II offers a short review of current machine-learning methods for storage rings. Section III analyzes the orbit control challenges and explains the SVD-based algorithm and supervised learning model. Section IV details our feedback system based on reinforcement learning. Section V presents simulation results and experiment outcomes on the NSLS-II beam. Section VI presents the conclusion and future plan.

II. RELATED WORK

Deep learning and big-data-driven methods have drawn much attention recently. The orbit feedback system is a multiple-input-multiple-output (MIMO) feedback system. Treating the MIMO system as a black box, the neural network can model the inverse relationship between the machine status (inputs) and the corrective actions (output) with supervised learning algorithms. In [7], [8], [9], [10], [11], [12], [13], neural networks were trained with supervised learning algorithm based on the input and output data. The input and output dimensions were usually less than 100. The network was trained with simulated data and validated with actual operating data for adapting to the real operation environment [9]. In [10], the surrogate model was regressed from collected operating data, and a network was additionally trained with the surrogate model. In [11], [12], [13], real-time-control experiments were conducted on the storage ring to achieve low RMS errors or fast controls.

Reinforcement learning (RL) agents learn to make decisions by interacting with the environment. Meier [14] trained an actor-critic algorithm with input states and output actions of a small dimensionality (< 10) in a storage ring simulator to achieve real-time control. Yang [15] proposed a multi-agent DDPG design for orbit calibration in MEBT. Apart from storage ring stabilization, studies [16], [17], [18], [19],

[20], [21] also explored applying RL algorithm into other accelerator facilities, for example, linear accelerator, free-electron laser, etc.

Throughout the study, data for supervised learning is either generated from the simulation software or collected from historical operations. This does not fit our situation for adaptive control. Current studies on model-free RL only handle lower-dimension systems. However, our system has high dimensionality. Thus, we design a model-based RL algorithm to achieve adaptive control with high dimensions.

III. BACKGROUND

A. Problem Definition

The orbit feedback system runs in a closed control loop, shown in Figure 3. The goal of the feedback controller is to

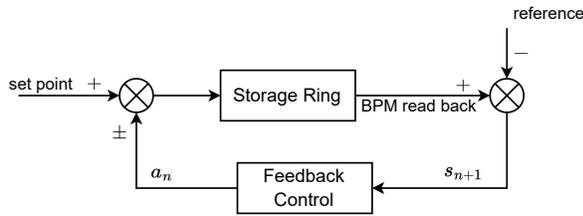


Fig. 3. An illustration of the closed loop feedback system

produce a_n , such that s_{n+1} maintains below the threshold.

We use the first-order approximation to model the feedback control system as follows:

$$s_{n+1} = s_n + Ra_n, \quad (1)$$

where a_n indicates the corrections applied, s_n and s_{n+1} are BPMs observed before and after we apply the correction, and R is the orbit response matrix.

B. SVD-based Feedback Control

We aim to control the next beam position s_{n+1} to 0. A straightforward way is to solve for $a_n = -R^{-1}s_n$. However, measurements of the ORM indicate the system has highly ill-posed dynamics. With measurement errors, the inverse of the response matrix could be extremely unstable. Singular value decomposition (SVD) is used to inverse the problem [5]. In NSLS-II fast orbit feedback (FOFB) control, the SVD method combines with PID controller [6]. Specifically, the controller is applied on each component of the spectrum space by doing SVD on the ORM: $R = U\Sigma V^T$. Figure 4 illustrates this process.

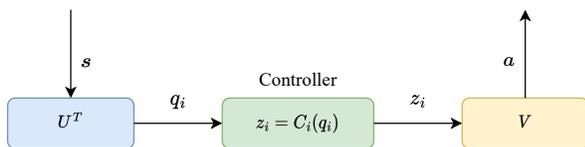


Fig. 4. The SVD-based PID control: q_i and z_i stand for each component of the spectrum space after the transformation of the current input.

The parameter set for the PID controller is given by (with proportional component only)

$$z_i = C(q_i) = -\frac{\sigma_i}{\sigma_i^2 + \lambda} q_i, \quad (2)$$

where σ_i is the i th singular value.

This process is proven identical to ridge regression.

$$\min_a \|Ra + s\|_2^2 + \lambda \|a\|_2^2. \quad (3)$$

C. Supervised Learning Model

Supervised learning learns from labeled data. Several research efforts applied supervised learning to the orbit feedback control problems [7], [8], [9], [10], [11], [12], [13]. For our problem, we train a neural network to generate action a_n given current state s_n as input. In the following context, we denote this network $\pi(s_n)$.

Given the dataset $\mathcal{D} \in R^m \times R^m$, the loss function for supervised learning is

$$L(w_\pi) = \mathbb{E}_{(s_n, a_n) \in \mathcal{D}} \|a_n - \pi(s_n)\|. \quad (4)$$

1) *Dataset preparation*: The training dataset \mathcal{D} can be obtained by: (i) extracting state-action pairs directly from the data archive of the running machine; (ii) running simulation software to generate states randomly and using an SVD-based algorithm to produce the corresponding action; (iii) running forward simulations to generate random actions as inputs and produce the subsequent states.

D. Reinforcement Learning Framework

RL aims to obtain a strategy to maximize the expected cumulative returns by interacting with the system. A typical RL framework comprises the tuple (S, A, r, P, γ) . State space S describes all possible running statuses of the storage ring, and action space A specifies the action to alter the system. Then the system can be abstracted as the probability mapping of the next state given the current state and action, say $P(s_{n+1}|s_n, a_n)$. Given a reward function $r(s, a)$, we aim to find the optimal control $a_n = \pi(s_n)$, called policy function, which maximizes the expected total reward

$$R(\pi) = \mathbb{E}_{s_0} \left[\sum_n \gamma^n r(s_n, a_n) \right] \quad (5)$$

over the whole trajectory. Here γ is the decay parameter to ensure the convergence of the expectation.

For the beam control problem in NSLS-II, we take the current BPMs as S , and the control signal as action A . If we do not involve noises and machine drifting, the system model is deterministic and given by Equation (1).

IV. METHOD

Model-based RL algorithms optimize the expected total reward based on the system model information. This section explores a model-based way to optimize the policy network.

Figure 5 shows the entire process for our framework when we run the algorithm online. The upper part of Figure 5 presents the trajectory optimization while the lower is for online model optimization.

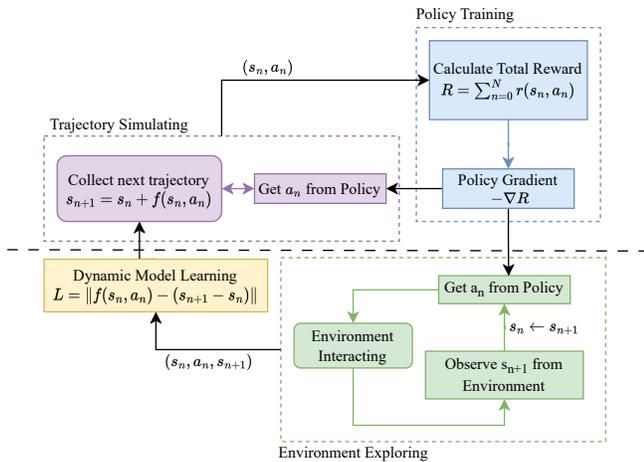


Fig. 5. The data flow for model-based RL with online model optimization

A. Trajectory Optimization

The system model is given by Equation (1), which is a differentiable model. Therefore, we can always generate a differentiable reward function when running the policy on this model. Leveraging the *autograd* engine in PyTorch, we do not have to bother calculating the complex gradient by hand, but collect the gradient information directly from the trajectory sampling process.

We utilize policy gradient directly to train the policy network. In the trajectory sampling process, we first sample a random state s_0 , then iteratively calculate the new state for time horizon N .

$$s_{n+1} = s_n + R\pi(s_n), \quad n \text{ from } 0 \text{ to } N-1. \quad (6)$$

Then the loss function will be the negative of the total reward

$$L(\pi) = - \sum_{i=0}^{N-1} r(s_i, \pi(s_i)). \quad (7)$$

The policy network will be updated based on $\nabla_{\pi} L$. Details are shown in Algorithm 1.

B. Online Model Optimization

Trajectory optimization pre-trains the policy network using the given system model (1) with R . However, this R might not be accurate and not represent the actual system behavior. Thus, after interacting with the environment, we can update the system model based on the collected data.

To fit the latest system model, data collection should not happen in Algorithm 1. Instead, the policy should run in parallel on the physical machine to collect the data point $(s_n, a_n, s_{n+1}) \in \mathcal{D}$. Then we can fit a new response matrix through least square.

$$\hat{R} = \arg \min_R \mathbb{E}_{\mathcal{D}} \|(s_{n+1} - s_n) - Ra_n\|_2^2. \quad (8)$$

The resulting \hat{R} is then used to replace the original R matrix in Algorithm 1.

Algorithm 1 Policy Gradient with Trajectory Sampling

Require: The system model R , Reward function $r(s, a)$

Ensure: Policy $\pi(s)$

Initialize neural network π .

while total episodes less than limit **do** Initialize s_0 .

while steps less than limit N **do**

$a_n \leftarrow \pi(s_n)$.

$s_{n+1} = s_n + R\pi(s_n)$.

Save s_n, s_{n+1}, a_n for training.

$s_n \leftarrow s_{n+1}$.

end while

Calculate policy loss based on expected reward.

$$L(w_{\pi}) = - \sum_{i=0}^{N-1} r(s_i, a_i).$$

Update weight w_{π} based on the $\nabla L_{w_{\pi}}$.

end while

Furthermore, the system model for online training does not have to be a linear function. In fact, the real machine does not have linear dynamics. Thus, we train another neural network $f(s_n, a_n)$ for forward system model learning. This network is trained in a supervised learning way.

$$L(w_f) = \|(s_{n+1} - s_n) - f(s_n, a_n)\|. \quad (9)$$

The system model then becomes

$$s_{n+1} = s_n + f(s_n, a_n). \quad (10)$$

$f(s_n, a_n)$ replaces the original system model in Algorithm 1 for further training.

V. EXPERIMENTS

We experiment on the simulation environment with SVD-based, supervised learning, and our model-based RL methods. The trained models are tested in the NSLS-II storage ring for further validation.

A. Experiment Setups

1) *Environment Setup:* In preliminary experiments, we employ a simulated environment, which runs the system model (1) to produce the next state. The input dimension and output dimension are both 180. However, some features are added to address two key properties of the actual machine: (i) two ORMs are measured at different machine states. The ORM used for system model will drift from one to another over iterations; (ii) observation noises are added to the BPM readback to simulate electronic noise.

2) *Evaluation Metrics:* For each algorithm, we run N long trajectories with length m in the simulated environment. We calculate the root mean square for each trajectory and plot the average of the RMS with its variance across N trajectories. The following performance metrics are considered: (i) best state RMS; (ii) worst state RMS; (iii) final state RMS; (iv) steps needed to reduce the RMS to a certain threshold.

3) *Neural network details*: For this problem, we will use a deep neural network with a 180-dimension input and output layer, and 3 hidden layers of 512 dimensions each. This gives us approximately 0.7M parameters in total.

This particular problem has a special property of the action taken. That is to take zero action if the state is already zero. To fit this property, we will use unbiased linear layers (set b_i to zero), and use the hyperbolic tangent function (\tanh) as the activation function.

We use Adam [22] as the training algorithm with learning rate 10^{-4} . This algorithm is considered to achieve superior performance in machine learning research. The same policy network design, training algorithm, and initial parameters will be used consistently throughout the experiment.

B. Simulation Experiment

We evaluate three methods with identical environment settings: the SVD-based, supervised learning, and our method. For our method, the policy network is pre-trained with trajectory optimization before interacting with the environment. The reward function used is the negative of the RMS. We sample 1,000 trajectories with a length of 10,000 for simulation, and plot the mean and std across different runs.

Figure 6 displays a short trajectory period, illustrating how our method is capable of recovering from poor machine BPMs. Our findings indicate that supervised learning approaches are unable to reduce the state to a smaller RMS value. This is likely due to overfitting so that the network cannot adapt to drifting environments. Our method initially converges slower than the SVD method. Interactions with the environment allow the agent to acquire new information about changes in environment dynamics, and refine its policies accordingly. Eventually, our method continues to reduce the RMS and surpasses the SVD method.

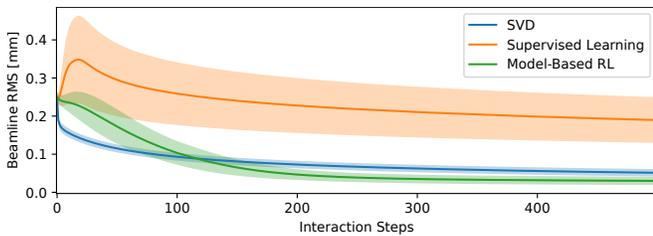


Fig. 6. Plot of trajectory at first 500 interactions. the solid line and shadows show the mean and standard deviation across different simulations

Figure 7 simulates the long-run experiments. As the system’s dynamics change over time, other methods cannot capture this and result in degenerated performance. For our method, the longer it interacts with the environment, the more robust it will be. At the final iteration, our agent controls the beam RMS down to the machine measurement accuracy ($\sim 0.2 \mu\text{m}$). Table I summarizes the key metric in the experiment.

C. Experiments on NSLS-II System

Based on the above work, we tested our machine-learning method directly on the storage ring beam. Due to the study

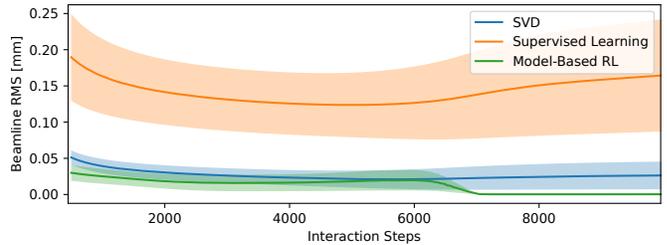


Fig. 7. Plot of trajectory for long term run (10000 steps)

time limit, we could not do a long-run test for our method. The online model optimization was performed once to fit into the current machine status. The results are shown in Figure 8.

In the experiment, we randomly kicked the beam off its original position and applied our models to control the orbit back. In Figure 8, the SVD-based method only stabilizes the beam RMS to $\sim 5 \mu\text{m}$, while our method reaches $\sim 1 \mu\text{m}$, showing a 80% improvement of RMS values.

VI. CONCLUSION

This study investigates the machine-learning methods for controlling the beam in the NSLS-II storage ring. The feedback system of NSLS-II is modeled by the orbit response matrix. The ORM cannot be obtained precisely due to machine internal status drift, environmental noise, and non-linear behavior of the system. Thus, the SVD controller leads to beam drift over time. Supervised learning is unsuitable for our control system because it tends to overfit and is not adaptive to machine drift. The model-based RL algorithm runs interactively with the environment to achieve adaptive control. Trajectory optimization optimizes the expected total reward using policy gradient. This approach involves using a neural network to learn the non-linear dynamics of the beam orbit system and extracting the optimal control signal over the trajectory. Online model optimization adaptively fits the current environment behavior by collecting real-time running data of the policy. Through both simulation and real-world experiments, our proposed method outperforms many existing algorithms and achieves 80% improvement compared to the current SVD-based method deployed at NSLS-II.

The adaptive control for our method runs in an overfitting way. That means we tried to consume as much training time as to keep track of the system drift. It could lead to a biased dataset for the algorithm to train on, ultimately impeding the algorithm’s ability to learn and generalize effectively. To mitigate this issue, we recommend collecting a significant amount of data before starting model optimization. We propose exploring new algorithms that allow the system to detect performance degeneration in real-time and perform online model optimization on demand.

ACKNOWLEDGEMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced

TABLE I
SUMMARY OF THE EXPERIMENT RESULT

	Min RMS	Max RMS	Final RMS	Steps to reach 0.05
SVD-Based Method	0.021	0.25*	0.026	532
Supervised Learning	0.12	0.35	0.16	N/A
Model-Based RL	0.00028	0.25*	0.00033	187

* Initial state

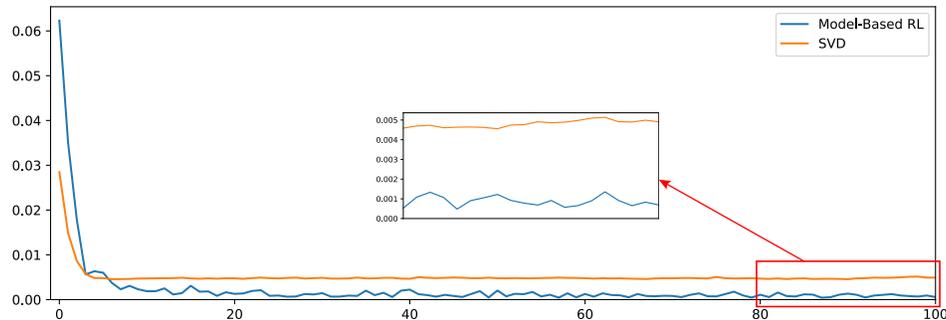


Fig. 8. Performance of Model-Based RL Algorithm on NSLS-II Storage Ring

Scientific Computing Research, Office of Nuclear Physics,
under Award Number DE-SC0019518.

REFERENCES

- [1] O. Singh, B. Bacha, A. Blednykh, W. Cheng, L. R. Dalesio, C. Danneil, J. De Long, A. Joseph, D. Penna, K. Ha, Y. Hu, B. N. Kosciuk, M. A. Maggipinto, J. Mead, D. Padrazo, I. Pinayev, Y. Tian, K. Vetter, and L.-H. Yu, "NSLS-II BPM and fast orbit feedback system," *accelconf.web.cern.ch*, 2013, iSBN: 9783954501274.
- [2] E. Mitchell, P. Kuhn, and E. Garman, "Demystifying the synchrotron trip: a first time user's guide," *Structure*, vol. 7, no. 5, pp. 111–121, May 1999, publisher: See full text options at Elsevier Science.
- [3] K. Haga, Y. Kamiya, M. Katoh, T. Katsura, T. Mitsuhashi, and N. Nakamura, "Global orbit feedback system at the Photon Factory storage ring," *Part. Accel.*, vol. 33, pp. 105–109, 1990.
- [4] J. Choi and T. V. Shafan, "Reproducibility of Orbit and Lattice at NSLS-II," in *Proceedings of IPAC2016*, 2016.
- [5] J. Corbett, D. Keeley, R. Hettel, I. Linscott, and J. Sebek, "Algorithms for orbit control on SPEAR," Stanford Linear Accelerator Center, Menlo Park, CA (United States); Stanford Univ., CA (United States). Stanford Synchrotron Radiation Lab., Tech. Rep. SLAC-PUB-6517; SLAC/SSRL-0077; CONF-940618-90, June 1994.
- [6] O. Singh, B. Bacha, A. Blednykh, W. Cheng, L. R. Dalesio, C. Danneil, J. De Long, A. Joseph, D. Penna, K. Ha, Y. Hu, B. N. Kosciuk, M. A. Maggipinto, J. Mead, D. Padrazo, I. Pinayev, Y. Tian, K. Vetter, and L.-H. Yu, "NSLS-II fast orbit feedback system," *accelconf.web.cern.ch*, 2015, iSBN: 9783954501274.
- [7] E. Bozoki and A. Friedman, "Neural networks and orbit control in accelerators," Brookhaven National Lab., Tech. Rep., 1994.
- [8] E. Fol, J. M. Coello De Portugal, G. Franchetti, and R. Tomás, "Optics Corrections Using Machine Learning in the LHC," *Proceedings of the 10th Int. Particle Accelerator Conf.*, vol. IPAC2019, pp. 4 pages, 0.414 MB, 2019, artwork Size: 4 pages, 0.414 MB ISBN: 9783954502080 Medium: PDF Publisher: JACoW Publishing, Geneva, Switzerland.
- [9] D. Xiao, C. Chu, and Y. Qiao, "Orbit Correction With Machine Learning," *Proceedings of the 10th Int. Particle Accelerator Conf.*, vol. IPAC2019, pp. 3 pages, 0.464 MB, 2019, artwork Size: 3 pages, 0.464 MB ISBN: 9783954502080 Medium: PDF Publisher: JACoW Publishing, Geneva, Switzerland.
- [10] X. Dengjie, Q. Yusi, and C. Zhongming, "Orbit correction based on machine learning," *High Power Laser and Particle Beams*, vol. 33, no. 5, pp. 054004–7, May 2021, publisher: High Power Laser and Particle Beams.
- [11] D. Schirmer, "Orbit Correction With Machine Learning Techniques at the Synchrotron Light Source DELTA," *Proceedings of the 17th International Conference on Accelerator and Large Experimental Physics Control Systems*, vol. ICALEPCS2019, pp. 5 pages, 0.814 MB, 2020, artwork Size: 5 pages, 0.814 MB ISBN: 9783954502097 Medium: PDF Publisher: JACoW Publishing, Geneva, Switzerland.
- [12] L. Ruichun, Z. Qinglei, M. Qingru, J. Bocheng, W. Kun, L. Changliang, and Z. Zhentang, "Application of machine learning in orbital correction of storage ring," *High Power Laser and Particle Beams*, vol. 33, no. 3, pp. 034007–9, Mar. 2021, publisher: High Power Laser and Particle Beams.
- [13] K. Chen, F. Wang, G. Wang, Z. Wang, M. Hosaka, L. Guo, and W. Xu, "Beam orbit shift due to BPM thermal deformation using machine learning," *Journal of Physics: Conference Series*, vol. 2420, no. 1, p. 012014, Jan. 2023, publisher: IOP Publishing.
- [14] E. Meier, G. LeBlanc, and Y. E. Tan, "Orbit Correction Studies using Neural Networks," *Proc. 3rd Int. Particle Accelerator Conf.(IPAC'12)*, 2012.
- [15] X. Yang, Y. Chen, J. Wang, H. Zheng, H. Liu, D. Zhou, Y. He, Z. Wang, and Q. Zhou, "Online beam orbit correction of MEBT in CiADS based on multi-agent reinforcement learning algorithm," *Annals of Nuclear Energy*, vol. 179, p. 109346, Dec. 2022.
- [16] V. Kain, S. Hirlander, B. Goddard, F. M. Velotti, G. Z. Della Porta, N. Bruchon, and G. Valentino, "Sample-efficient reinforcement learning for CERN accelerator control," *Physical Review Accelerators and Beams*, vol. 23, no. 12, p. 124801, Dec. 2020, publisher: American Physical Society.
- [17] X. Pang, S. Thulasidasan, and L. Rybarczyk, "Autonomous Control of a Particle Accelerator using Deep Reinforcement Learning," Dec. 2020, arXiv:2010.08141 [physics].
- [18] D. Y. Wang, H. Bagri, C. Macdonald, S. Kiy, P. Jung, O. Shelbaya, T. Planche, W. Fedorko, R. Baartman, and O. Kester, "Accelerator Tuning with Deep Reinforcement Learning," in *Proceedings of the 35th Annual Conference on Neural Information Processing Systems*, 2021, p. 11.
- [19] S. Hirlander and N. Bruchon, "Model-free and Bayesian Ensembling Model-based Deep Reinforcement Learning for Particle Accelerator Control Demonstrated on the FERMI FEL," Jan. 2022, arXiv:2012.09737 [physics].
- [20] A. Scheinker, F. Cropp, S. Paiagua, and D. Filippetto, "An adaptive approach to machine learning for compact particle accelerators," *Scientific Reports*, vol. 11, no. 1, p. 19187, Sept. 2021, number: 1 Publisher: Nature Publishing Group.
- [21] F. M. Velotti, B. Goddard, V. Kain, R. Ramjiawan, G. Z. Della Porta, and S. Hirlander, "Automatic setup of 18 MeV electron beamline using machine learning," Sept. 2022, arXiv:2209.03183 [physics].
- [22] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017, arXiv:1412.6980 [cs].