Machine Learning Tools at BESSY II

Luis Vera Ramírez, Gregor Hartmann

Helmholtz-Zentrum Berlin

AMALEA Meeting, DESY, Hamburg, 02/12/2019

Motivation

At the **synchrotron radiation source BESSY II** (Helmholtz-Zentrum Berlin, HZB) both the **beamline** and the **machine groups** have started working towards setting up the infrastructure to introduce **modern analysis, optimization and automation** in order to improve the performance and the experimental setups.

Our first studies with Machine Learning tools include **analysis and prediction models** with real accelerator data as well as first prototypes and use cases for **parameter tuning** with (Deep) Reinforcement Learning agents.

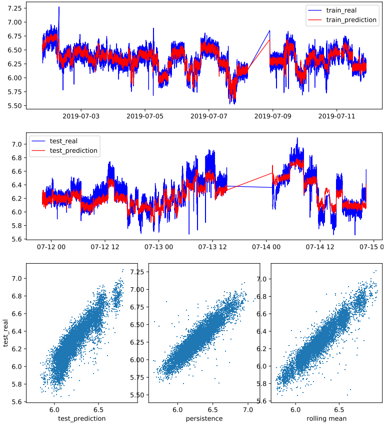
Table of Contents

Table of Contents

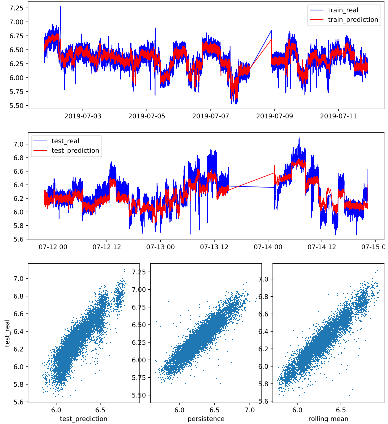
Overview: Current state at BESSY II

Device	Injector Ring accelerator	Undulator	Beamline	Experiment
Data	• Archive, Diagnostic • Simulations • Online optimization	• Diagnostic • Raytracing • Scans/online data		• Demands • Simulations • Beamtimes
Methods	• SVR-RFF • DNN • Deep-RL-Control • RNN, LSTM	• Autoencoder • CNN, MLP, GBoost • Dataloader • Tensor product • kNN, auto-diff.		• Reasonable random generator
Agent	Operator	Beamline scientist		(Random-) User

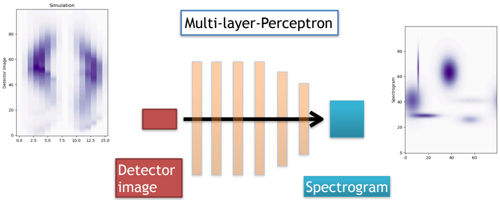
At the **accelerator side**: prediction of beam loss, injection efficiency, **beam lifetime...**



At the **accelerator side**: prediction of beam loss, injection efficiency, **beam lifetime**...



At the **beamline side**: mapping between detector images and spectrograms...



... and also **beamline raytracing** (will be discussed later)

Table of Contents

Overview

RLControl - Experiments

Optimization of booster current

Optimization of injection efficiency

Beamline raytracing

Conclusion

References

- **Deep Deterministic Policy Gradient** [LHP+16]: Actor-critic Reinforcement Learning algorithm for continuous environments.
- Off-policy data and the Bellman equation used to learn the Q-function.
- Q-function used to learn the policy.
- *Approximated* with NNs.

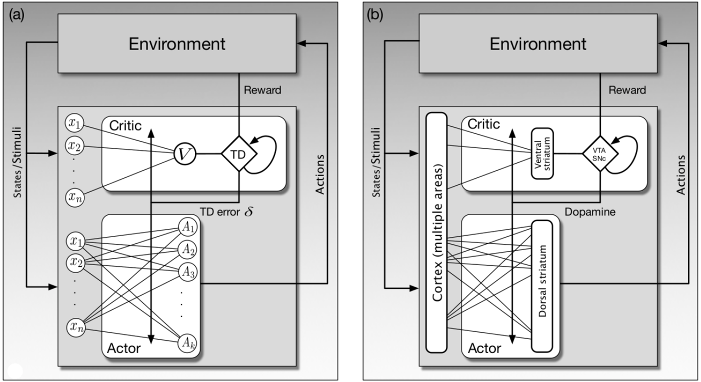


Figure: From [SB18]

- **Deep Deterministic Policy Gradient**: Actor-critic Reinforcement Learning algorithm for continuous environments.
- Introduced in [LHP+16] with several implementation tricks: delayed target networks ($Q_{\tilde{\omega}}$, $\mu_{\tilde{\theta}}$), replay buffer...
- **Critic update** with stochastic behavior policy β and loss[1]:

$$L(\omega) = \mathbb{E}_{s_t \sim \rho^\beta, a_t \sim \beta}\left[\left(Q_\omega(s_t, a_t) - (r(s_t, a_t) + \gamma Q_{\tilde{\omega}}(s_{t+1}, \mu_{\tilde{\theta}}(s_t))\right)^2\right]$$

- **Actor update** - (off-policy) Deterministic Policy Gradient Theorem ([SLH+14]): for the performance objective $J_\beta(\mu_\theta) = \mathbb{E}_{s \sim \rho^\beta}\left[Q^{\mu_\theta}(s, \mu_\theta(s))\right]$,

$$\nabla_\theta J_\beta(\mu_\theta) \approx \mathbb{E}_{s \sim \rho^\beta}\left[\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu_\theta}(s, a)\big|_{a=\mu_\theta(s)}\right]$$

[1] In [SLH+14], $\rho^\beta(s') := \int_S \sum_{t=1}^{\infty} \gamma^{t-1} p_0(s) p(s \to s'|t, \beta) ds$

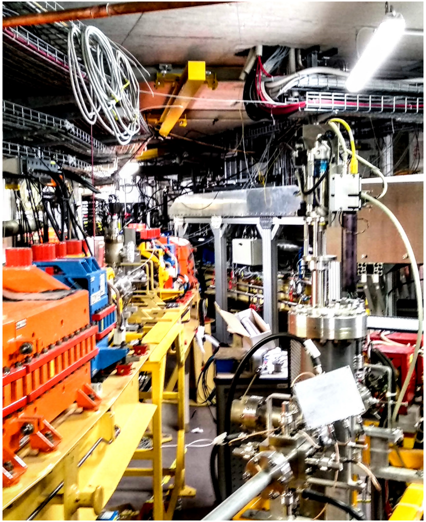
Table of Contents

Optimization of booster current: Case description

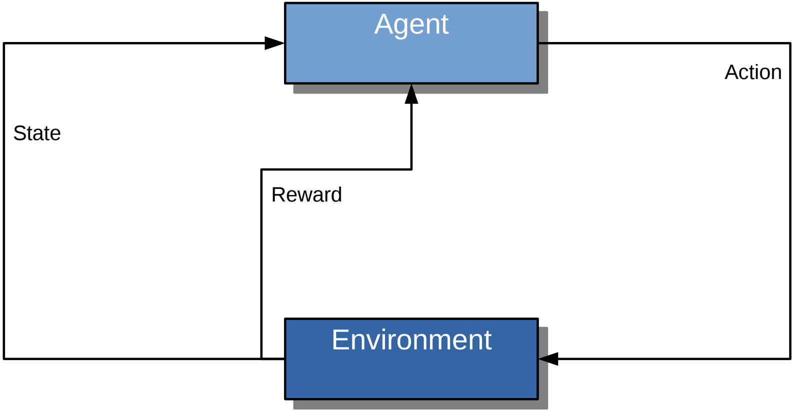
After long interruptions of the machine operation, the booster current tends to be low - as for today, **manual parameter tuning** is required.

We seek an automatized, RL-based solution.

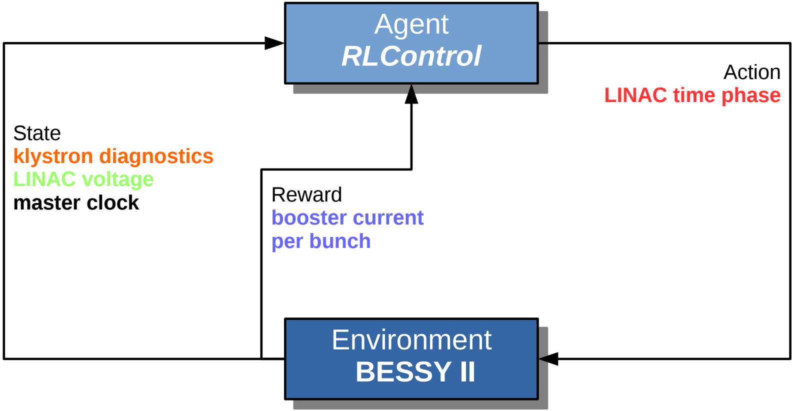
- State variables:
  - High (radio) frequency - master clock.
  - Voltage in LINAC.
  - Klystron current diagnostic measurements.
- Action variable: **time phase in LINAC.** Observations show that this parameter does not affect the injection efficiency.
- Reward: (normalized) **booster current per bunch.**



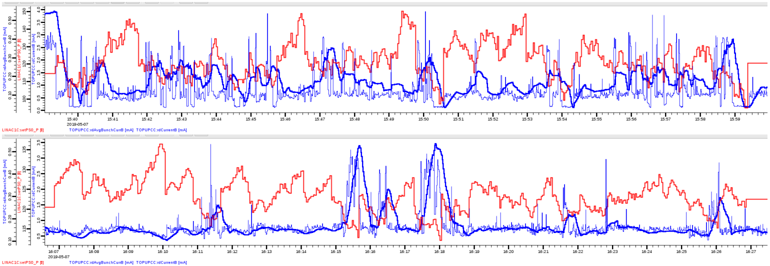
RLControl - Experiments: Case description



RLControl - Experiments: Case description

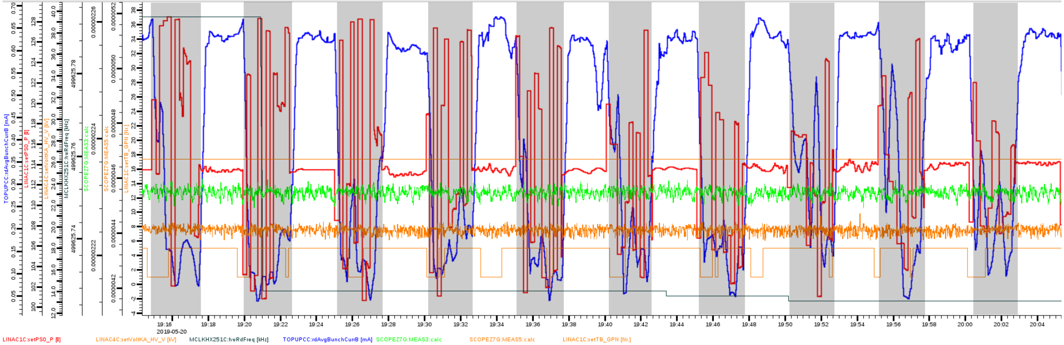


- Long training time and normalization problems $\rightarrow$ improved through **demonstration with historical data** (inspired by [ZM18]).
- Slow *reaction time* to reward modifications $\rightarrow$ solved by using low γ and giving up average current as reward and using the **instantaneous current per bunch**.
- Non-optimal exploration $\rightarrow$ solved by implementing **Parameter Space Noise** ([PHD$^+$17]).

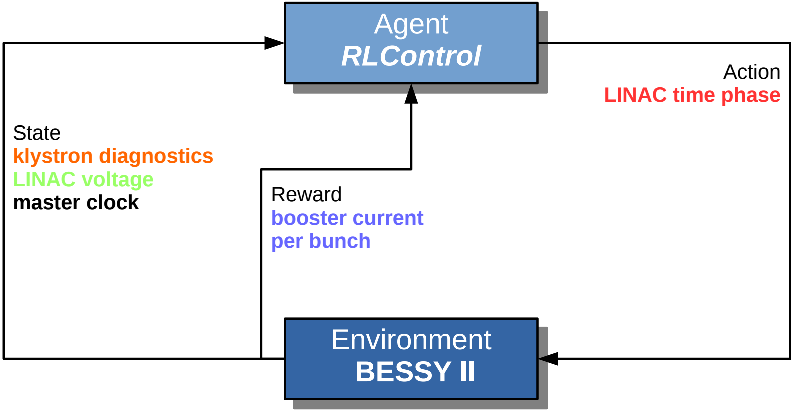


Optimization of booster current: Preliminary tests

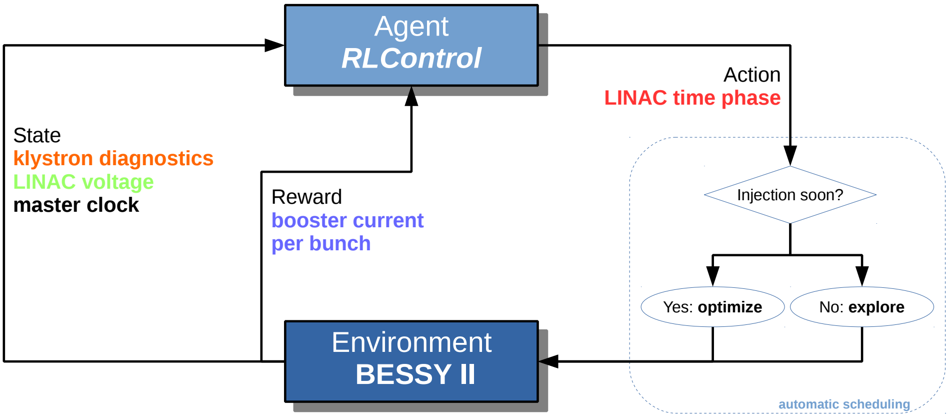
Short test (20/05/19). **Reward (booster current per bunch)** in **blue**, **action (LINAC time phase)** in **red** - remaining lines correspond to state variables. Pretraining with 30 days of historical data. **Exploration** with Parameter Space Noise ([PHD+17]) appears **shaded** - period length of ca. 2 min chosen as experiment for automatic scheduling...



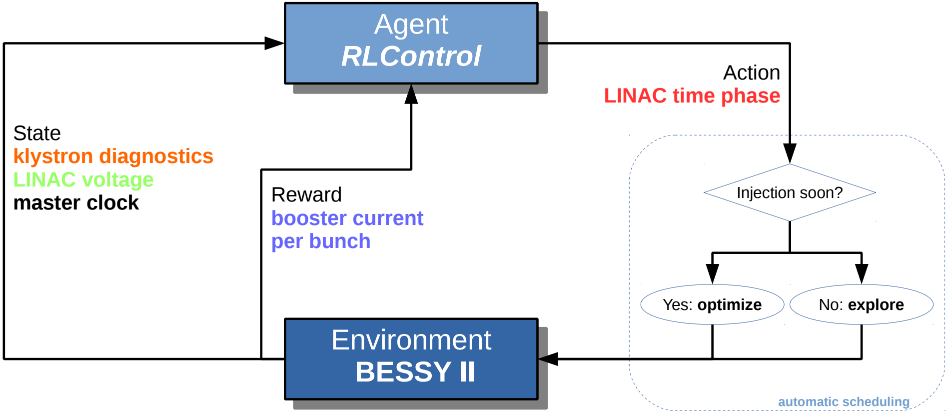
Optimization of booster current: Automatic scheduling



Optimization of booster current: Automatic scheduling



Optimization of booster current: Automatic scheduling



Exploration is scheduled in the meantime between injections to avoid disturbing user activity - optimization activated shortly before each injection.

Optimization of booster current: Test during user operation

*Long test during user time with **automatic exploration schedule** (09/07/19).* Reward (booster current per bunch) in blue, action (LINAC time phase) in red - remaining lines correspond to state variables. Pretraining with 30 days of historical data. Exploration with automatic schedule shaded - **first hour.** The agent optimizes (and learns) successfully during the next **8.5 hours of user operation.**

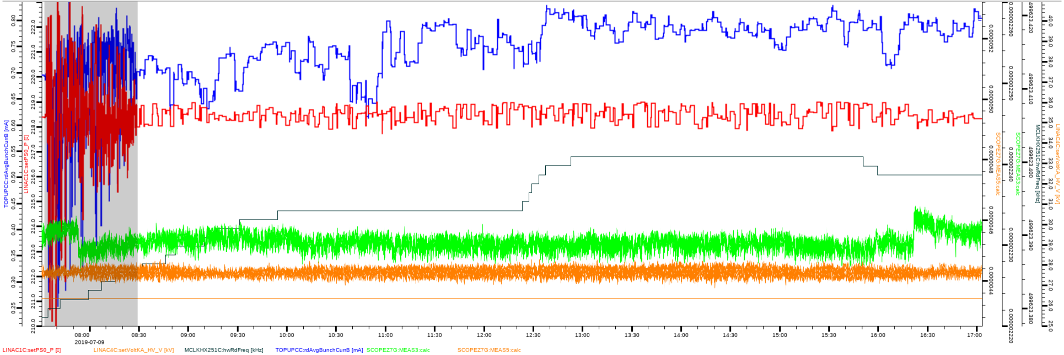
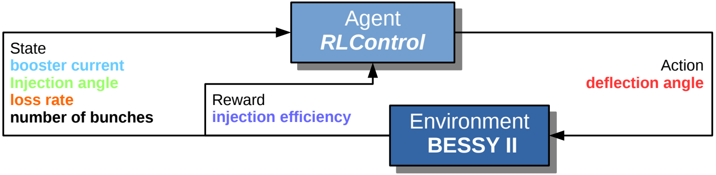


Table of Contents

HZB Helmholtz Zentrum Berlin

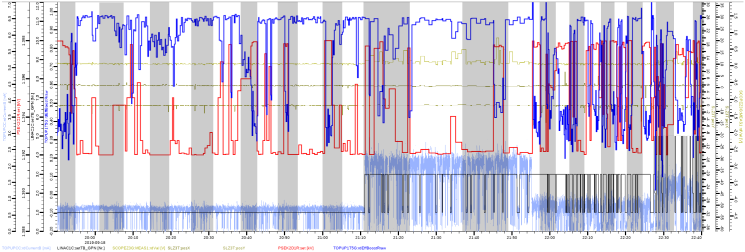
Injection efficiency is known to be affected by temperature - nowadays it also needs **manual tuning**. RL-based optimization is a work in progress.



- State variables:
  - Number of **bunches** generated by the LINAC (1, 3 or 5).
  - Injection angle **mismatch**, measured by the beam position in the transfer line.
  - **Current** measured during the booster acceleration phase.
  - Measured **loss rate after extraction** from the booster.
- Action: **deflection angle into the storage ring**, generated by the 2^{nd} septum.
- Reward: **last injection efficiency** - fraction of current increase generated in the storage ring by the charge accelerated in the booster.

RLControl - Experiments: Issues

- Initial problems regarding **pretraining with historical data**: septum external conditions vary along time, inducing variations in the optimal action intervals that were not reflected in the chosen state variables.
- First tests with agents trained from scratch: good performance with few steps in stable conditions, they failed when certain unknown states (induced by external modifications in the booster current) had to be faced.



- Solved with **shorter pretraining period** but including also **non-top-up data**.

HZB Helmholtz Zentrum Berlin

Short test (23/09/19). Pretraining with 23 days of historical data. **Reward (injection efficiency)** is plotted in **blue**, **actions (septum deflection angle)** is plotted in **red**. Exploration periods appear shaded. *Ad-hoc* modifications in the number of pulses (in black) and booster current (in purple) are carried out during the test - the agent manages to find and improve the optimal action regions.

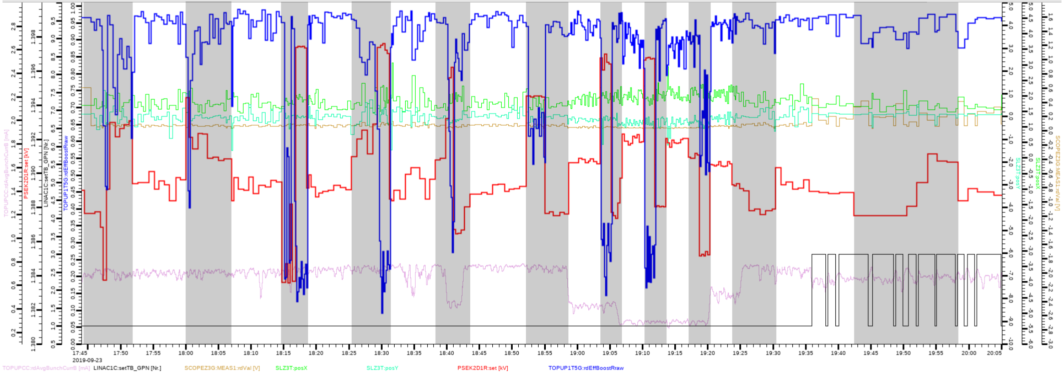
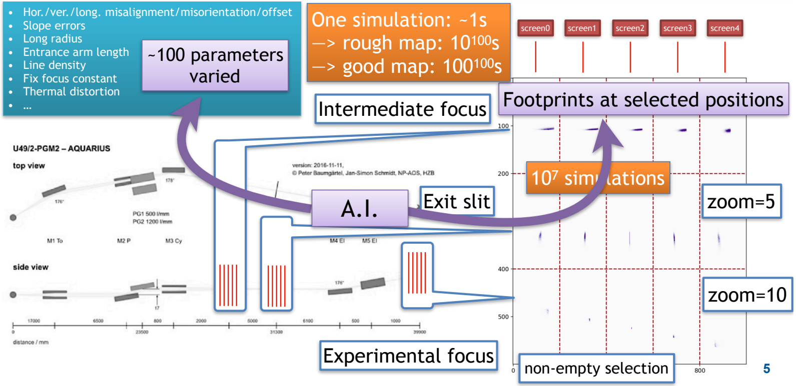


Table of Contents

- **Beamline raytracing** is a powerful tool to understand X-ray-beam propagation and for optimizing beam properties for the experimental requirements.
- Nevertheless, the amount of parameters is **impossible to map** with traditional simulation tools:



Beamline raytracing tools: Simulation via Ray-UI

HZB Helmholtz Zentrum Berlin

User Interface for the ray tracing software **RAY** (developed at BESSY in the 80s)

Contact

Dr. Peter Baumgärtel

(030) 8062 - 15154

Email

Prof. Dr. Hartmut Schirmacher

Email
Business card

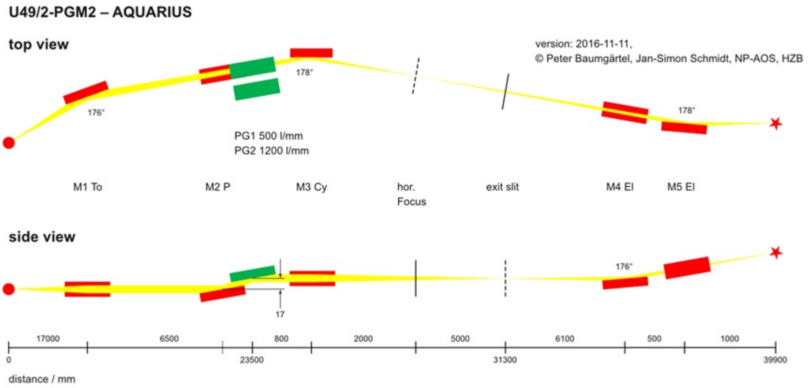
W-i-p:
- CUDA
- Coherence

- P. Baumgärtel, M. Witt, J. Baensch, M. Fabarius, A. Erko, F. Schäfers and H. Schirmacher, RAY-UI: A Powerful and Extensible User Interface for RAY, AIP Conf. Proc. 1741, 040016 (2016)
- P. Baumgärtel, P. Grundmann, T. Zeschke, A. Erko, J. Viefhaus, F. Schäfers, and H. Schirmacher, RAY-UI: New Features and Extensions, AIP Con. Proc. 2054, 060034 (2019)

http://hz-b.de/ray

15

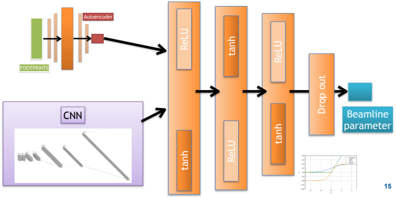
- The Aquarius project provides BESSY II a soft x-ray beamline and user infrastructure at the segment H15 (high-β section), including chemistry laboratory and laser hutch.

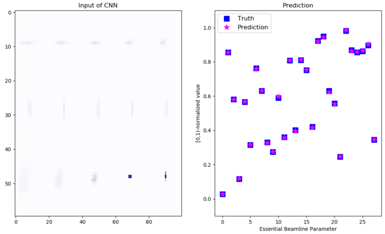


Beamline raytracing: ML approaches

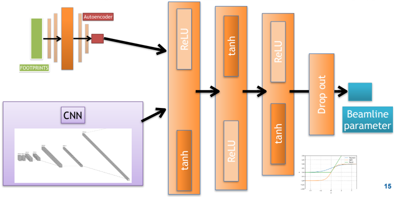
- We aim to use ML methods to learn raytracing as well as beamline parameter prediction from photon diagnostics.
- Diagnostic: **photon footprint screens** at three different positions in the Aquarius beamline (intermediate focus, exit slit and experimental focus).
- Several approaches to the problem:

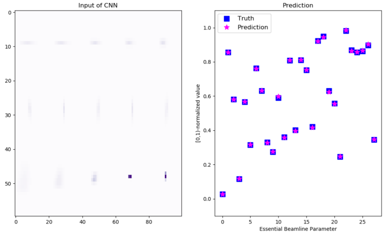
- We aim to use ML methods to learn raytracing as well as beamline parameter prediction from photon diagnostics.
- Diagnostic: **photon footprint screens** at three different positions in the Aquarius beamline (intermediate focus, exit slit and experimental focus).
- Several approaches to the problem:
  - **Supervised learning**: inversion of beamline raytracing with neural networks - **CNNs** and **autoencoders** $\rightarrow$ work by Dr. Gregor Hartmann





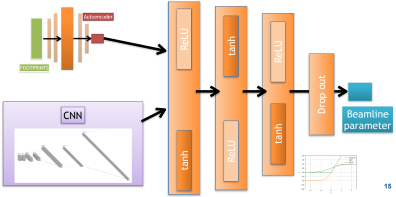
- We aim to use ML methods to learn raytracing as well as beamline parameter prediction from photon diagnostics.
- Diagnostic: **photon footprint screens** at three different positions in the Aquarius beamline (intermediate focus, exit slit and experimental focus).
- Several approaches to the problem:
  - **Supervised learning**: inversion of beamline raytracing with neural networks - **CNNs** and **autoencoders** → work by Dr. Gregor Hartmann
  - **Automatic differentiation** → work by Florentin David Hildebrandt

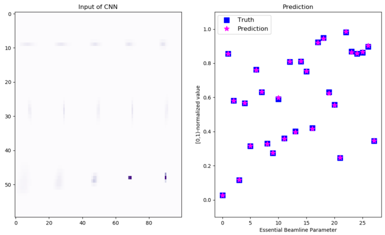




Beamline raytracing: ML approaches

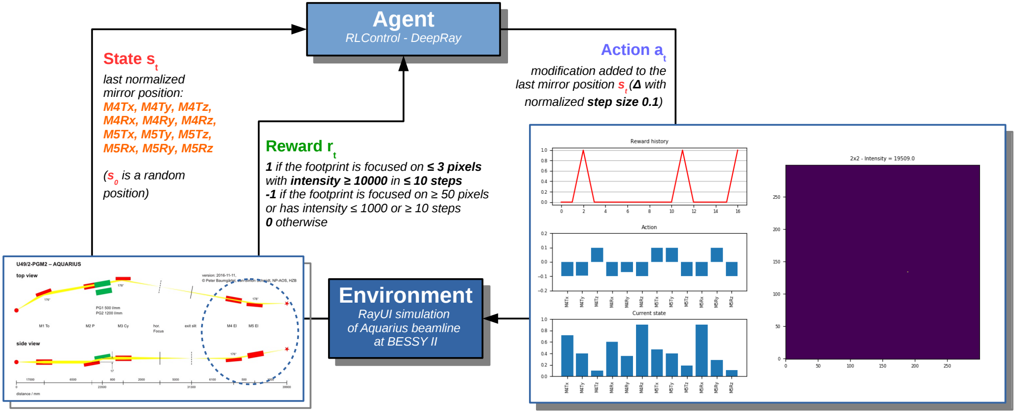
- We aim to use ML methods to learn raytracing as well as beamline parameter prediction from photon diagnostics.
- Diagnostic: **photon footprint screens** at three different positions in the Aquarius beamline (intermediate focus, exit slit and experimental focus).
- Several approaches to the problem:
  - **Supervised learning**: inversion of beamline raytracing with neural networks - **CNNs** and **autoencoders** $\rightarrow$ work by Dr. Gregor Hartmann
  - **Automatic differentiation** $\rightarrow$ work by Florentin David Hildebrandt
  - **Deep RL**





Beamline raytracing: Deep RL approach - Aquarius beamline

HZB Helmholtz Zentrum Berlin



- **Step**: one single raytracing (one screen at experimental focus, res. 300x300).
- **Episode**: end after 10 modifications of the initial mirror position or reward ± 1.

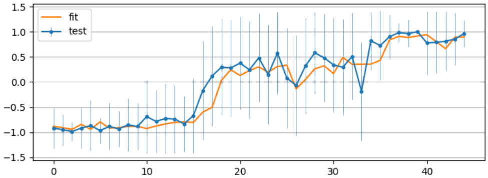
Beamline raytracing: Deep RL approach - Aquarius beamline

- **Extreme model-free approach**: the agent *only* learns how to optimize the output in few steps given any initial mirror position, not the effect of the different mirror parameters.

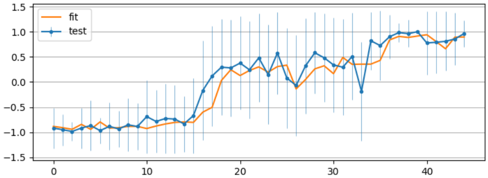
Beamline raytracing: Deep RL approach - Aquarius beamline

- **Extreme model-free approach**: the agent *only* learns how to optimize the output in few steps given any initial mirror position, not the effect of the different mirror parameters.
- Heavy **parallelization** required because of the slow Ray-UI simulations, was a challenge from the mathematical point of view.

- **Extreme model-free approach**: the agent *only* learns how to optimize the output in few steps given any initial mirror position, not the effect of the different mirror parameters.
- Heavy **parallelization** required because of the slow Ray-UI simulations, was a challenge from the mathematical point of view.
- The convergence was surprisingly fast - usually achieved **within** $\sim$ **100000 steps** - i.e. $\sqrt[12]{100000} \simeq 2.6102$ points per dimension.

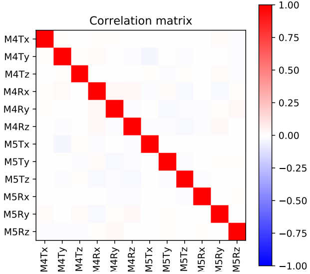


- **Extreme model-free approach**: the agent *only* learns how to optimize the output in few steps given any initial mirror position, not the effect of the different mirror parameters.
- Heavy **parallelization** required because of the slow Ray-UI simulations, was a challenge from the mathematical point of view.
- The convergence was surprisingly fast - usually achieved **within** $\sim$ **100000 steps** - i.e. $\sqrt[12]{100000} \simeq 2.6102$ points per dimension.



- The proximity condition (trying to find optimal points in few steps, i.e. close to the initial random point) leads to find **many new optimal mirror positions.**

- Test: **5000 episodes** with random initial positions



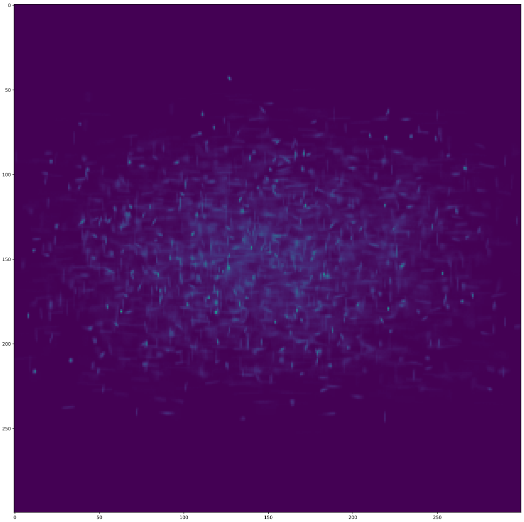
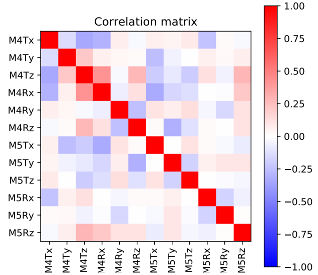


Figure: 5000 initial random positions

- Test: **5000 episodes** with random initial positions
- Results: **95.66% success rate** (after 4.62 ± 2.3 steps per episode)



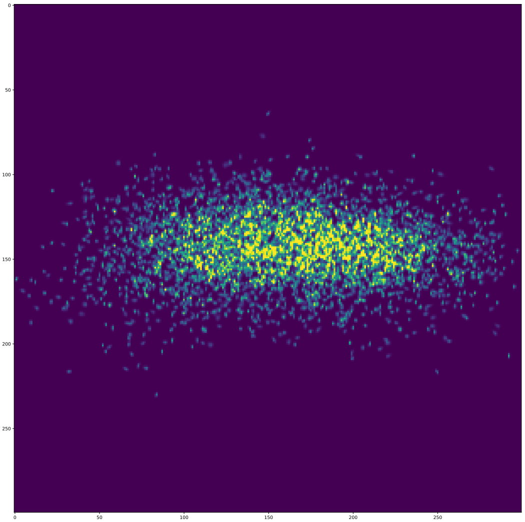
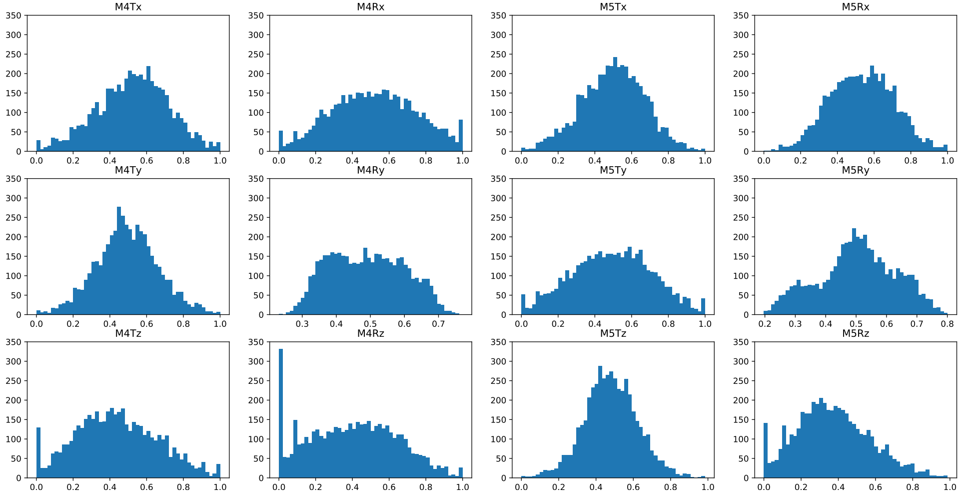
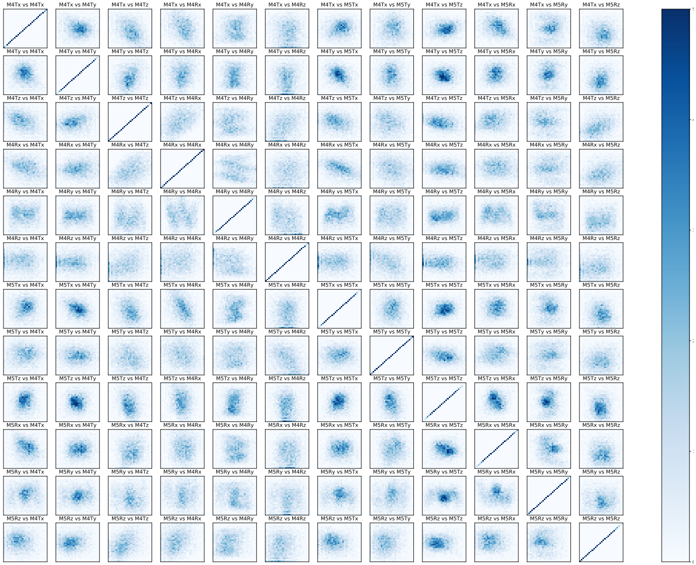


Figure: 4783 optimal positions

Beamline raytracing: Aquarius' 4783 optimal positions - Histograms



Beamline raytracing: Aquarius’ 4783 optimal positions - Histograms



Finally, we can also play with the pretrained RL-agent in an interactive way...

Play movie...

Table of Contents

Conclusion: Next steps for RL @ BESSY II

- Further tests at the machine: **injection efficiency with additional state and action variables**
- Further development of the raytracing case and application at the real beamlines.
- Additional use cases: orbit correction with OCELOT pretraining?
- Bluesky integration
- User interfaces and standalone applications

Table of Contents

[BKH16] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016.

[Bre01] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[GEW06] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, Apr 2006.

[LHP+16] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[PHD+17] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y. Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. *CoRR*, abs/1706.01905, 2017.

[Roj96] Raul Rojas. *Neural networks : a systematic introduction*. Springer-Verlag, Berlin ;;New York, 1996.

[RR08] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184. Curran Associates, Inc., 2008.

Bibliography II

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[SLH+14] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 387–395, Bejing, China, 22–24 Jun 2014. PMLR.

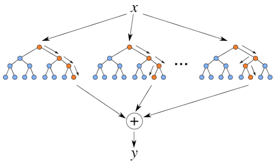
[Smo98] Alexander Johannes Smola. Learning with kernels, 1998.

[SS03] Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. Technical report, STATISTICS AND COMPUTING, 2003.

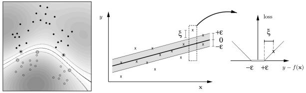
[Vap95] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995.

[ZM18] Xiaoqin Zhang and Huimin Ma. Pretraining deep actor-critic reinforcement learning algorithms with expert demonstrations. *CoRR*, abs/1801.10459, 2018.

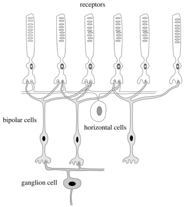
- **Ensemble methods**: Random Forests, Extremely Randomized Trees... ([Bre01], [GEW06]). For regression, **MSE** as loss $\rightarrow$ variance as impurity measure. **Self-explaining**: allows individual analysis of each variable's behavior.



- **Support Vector Regression** [Smo98] with **Random Fourier Features** ([RR08]). SVR extends traditional SVM (for classification) via Vapnik's ϵ-insensitive loss function ([Vap95]).

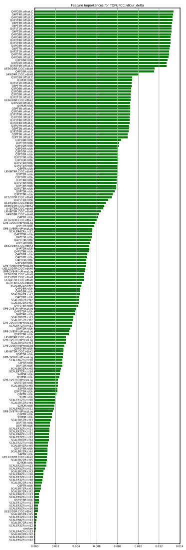


- **Neural Networks** (e.g. see [Roj96]). Feed-forward NNs for regression (i.e. MSE as loss function).



Figs. from https://dsc-spidal.github.io/harp/docs/examples/rf/, [SS03], [Roj96].

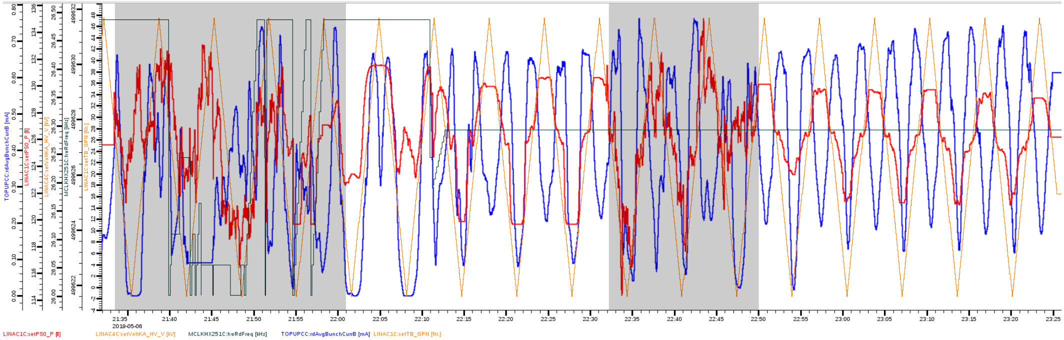
Optimization of injection efficiency: Experiments



- Target: instant approximation to **beam lifetime** defined via **current decay rate** ($k = 20$): $\frac{1}{\tau} = -\frac{\mathring{I}_t}{I_t} \approx -\frac{1}{I_t}\frac{\sum_{i=0}^{k}(I_{t-i}-I_{t_0})(t-i-t_0)}{\sum_{i=0}^{k}(t-i-t_0)^2}$
- Input variables (185 after preprocessing):
  - Gap and shift of **insertion devices** undulators (21 vars)
  - **Power supply currents** and **offsets** into **quadrupoles** (58 + 38 vars) and **sextupoles** (7 vars)
  - **Collisions** with rest gas particles, **vacuum pressures** (12 vars)
  - **Local beam loss fractions** (49 vars)
- Feature importance analysis with `RandomTrees`: even distribution, but **quadrupoles** (offsets) and **insertion devices** stand out

Test set	Algorithm	RMSE				R^2		
		Avg.	Pers.	Mov. Pers.	Model	Pers.	Mov. Pers.	Model
Random 20%	`ExtraTrees`	0.201	0.099	0.091	0.068	0.757	0.794	0.885
	`SVR-RFF`				0.077			0.852 ± 0.001
	`DNN`				0.069			0.881 ± 0.001
Last 20%	`ExtraTrees`	0.231	0.096	0.079	0.195 ± 0.001	0.829	0.884	0.292 ± 0.006
	`SVR-RFF`				$\mathbf{0.121 \pm 0.003}$			$\mathbf{0.725 \pm 0.015}$
	`DNN`				$\mathbf{0.125 \pm 0.006}$			$\mathbf{0.707 \pm 0.027}$

Short test (06/05/19): **LINAC voltage** set to **oscillate with a triangle wave** in order to observe agent reactions. Averaged booster current is used as reward. 7 days of historical data as demonstration and two exploration periods (with output noise) are used for training - shaded in the plot.



Backup: RLControl @ BESSY II - Settings

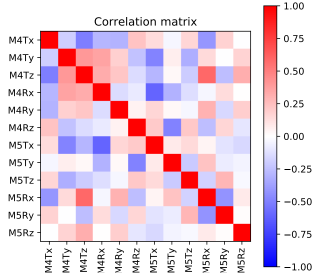
- Neural networks: in both cases, `relu` used as inner activation function and `adam` as optimizer (lr = 0.001).
  - Critic network: five hidden layers (25+50+25+10+5 neurons) and concatenates actions at the first hidden layer. Linear activation at the output layer.
  - Actor network: three hidden layers (25+10+5 neurons), all of them with layer normalization ([BKH16]). `tanh` used as activation for the output layer.
  - In the injection efficiency case, the number of neurons is doubled.
- Data preprocessing: $[-1, 1]$ linear normalization, historical data downsampled to 60 seconds.
- Parameter Space Noise: $\delta = 0.01$.
- Training parameters: $\gamma = 0.2$, pretraining with 10000 steps (2000 before actor training), warm-up with 32 steps, target model update rate = 0.1.
- *Brute-force* synchronization: update every **2 seconds** through EPICS.

Backup: RLControl - Aquarius raytracing - Settings

- Neural networks: `tanh` used as inner activation function and `adam` as optimizer (lr = 0.001).
  - Critic network: five hidden layers (250+500+250+100+50 neurons) and concatenates actions at the first hidden layer. Linear activation at the output layer.
  - Actor network: three hidden layers (250+100+50 neurons).
- Data preprocessing: $[-1, 1]$ linear normalization.
- Parameter Space Noise: $\delta = 0.2$.
- Training parameters: $\gamma = 0.99$, warm-up with 32 steps, target model update rate = 0.01

Backup: Deep RL approach - Aquarius beamline

- Test: **5000 episodes** with random initial positions
- Results: **95.66% success rate** (after 4.62 ± 2.3 steps per episode)



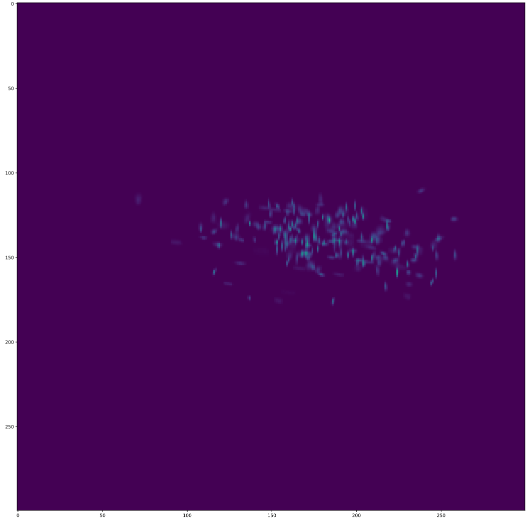


Figure: 217 fail positions