Reinforcement Learning applied to RF manipulation optimization in the Proton Synchrotron

Lightning talk @ the thematic Cern School of Computing, Split, 2024

Presenter: Joel Wulff

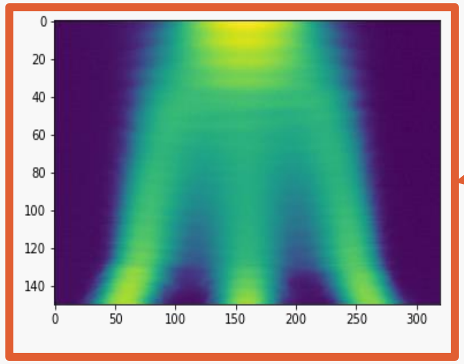
Date of presentation:14/10/2024

Acknowledgements: A. Lasheen,
H. Damerau, PS operation crew, SY-RF-BR section

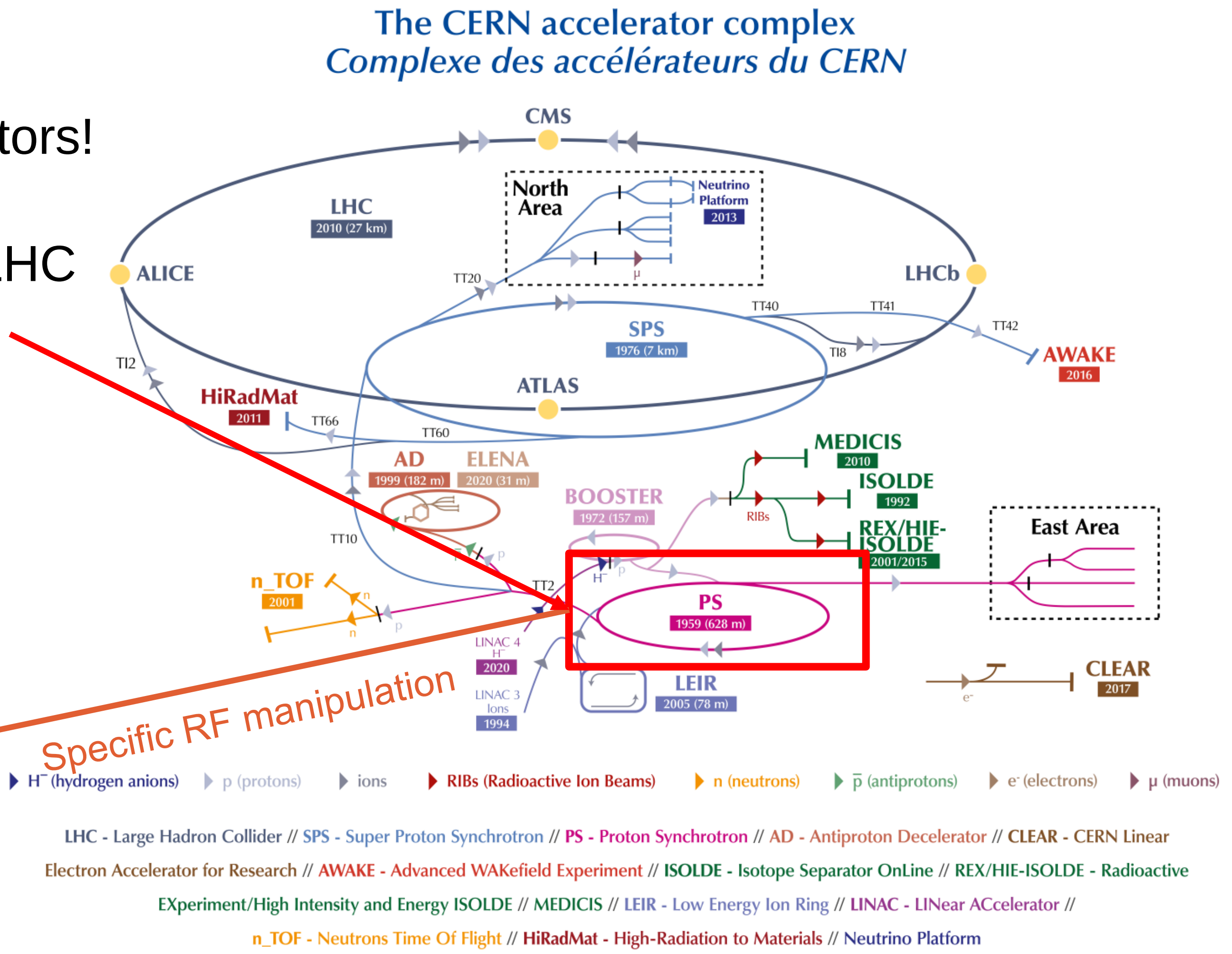
The CERN accelerator complex

- Complicated network of multiple accelerators!
- Longitudinal structure of the beam in the LHC is created in the Proton Synchrotron (PS) through a series of RF manipulations

Longitudinal triple splitting
→ requires frequent optimization

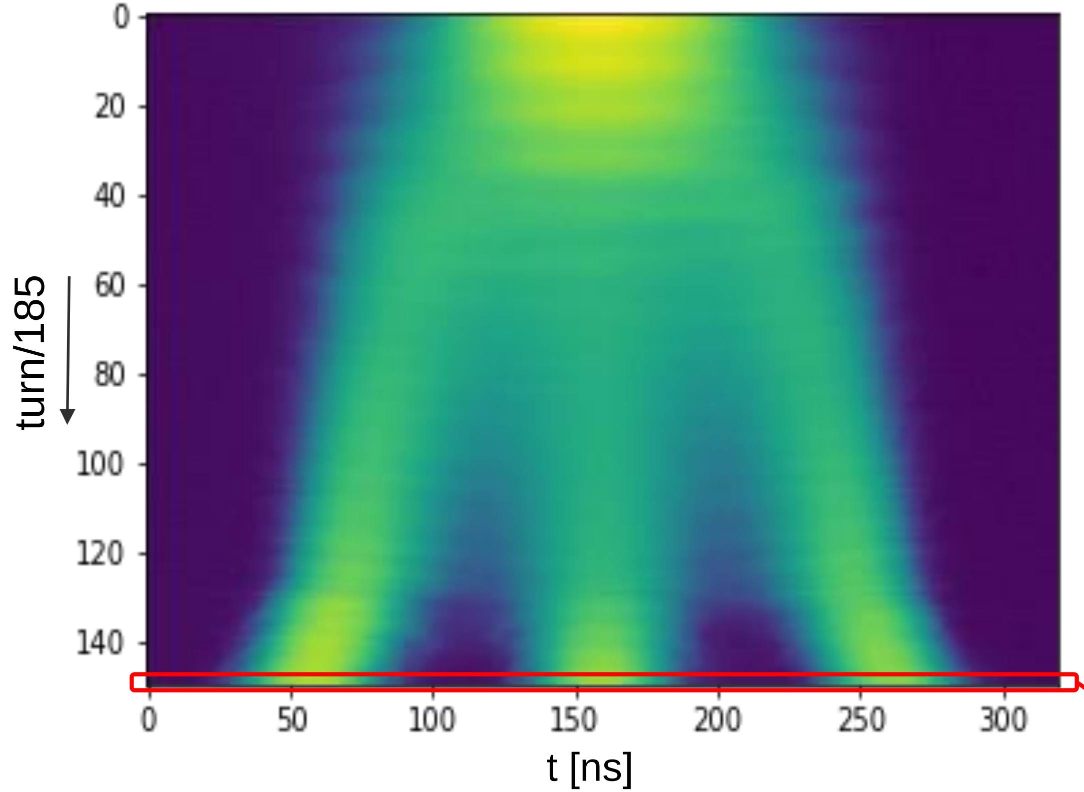


Specific RF manipulation



The longitudinal triple splitting

Particle *bunch* evolution over time



Each row is one measurement of the longitudinal distribution of particles in our bunch → a *profile*

Triple split → from 1 bunch to 3 longitudinally

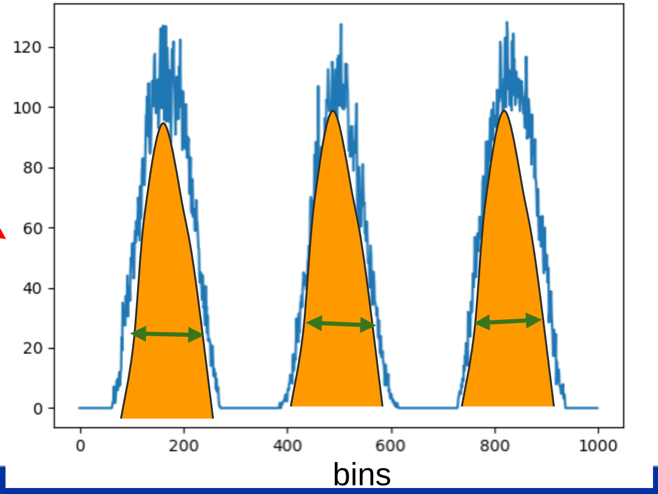
- 3 RF cavities on 3 different harmonics pulsed at the same time to accomplish.

Three parameters to optimize:

- Phases and voltage: ϕ_{14}, ϕ_{21}, V_{14}

Goal

- Observables of all final bunches equal, e.g.



Bunch profile

Bunch length

Bunch intensity

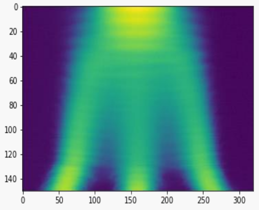
tCSC on Machine Learning 2024

How to optimize?

Requirements	Decisions
Must be efficient and accurate →	**Trained ML models**: leverage experience from training for fast convergence (if well trained)
Requires labeled data (real data expensive, time-consuming) →	**Trained using simulated data, applied directly to machine**

Two types of architectures used in conjunction:

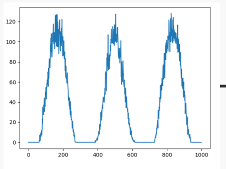
Convolutional Neural Network (CNN)



$\rightarrow \{\phi_{14}, \phi_{21}, V_{14}\}$

Reinforcement Learning (RL) Agents

Final profile



→ *Extract final bunch lengths/intensities*

Input: BLs, Int. $\rightarrow \{\phi_{14}, \phi_{21}, V_{14}\}$

How to optimize?

Requirer ... **isions**

Must be efficient

els: leverage
raining

Requires labeled data (real data
expensive, time-consuming)

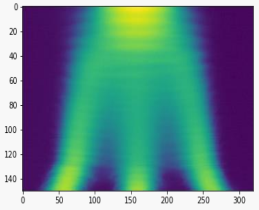
simulated data, applied directly to machine
(BLonD)

Both approaches work great in simulation!

Does it work in the real machine?

Two t ... **nction:**

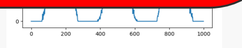
Convolutional Neural Netw



→ $\{\phi_{14}, \phi_{21}, V_{14}\}$

NO :(
(not initially)

ent Learning (RL) Agents



Extract final bunch lengths/intensities

Input: BLs, Int. → $\{\phi_{14}, \phi_{21}, V_{14}\}$

Why did first models fail in the real machine?

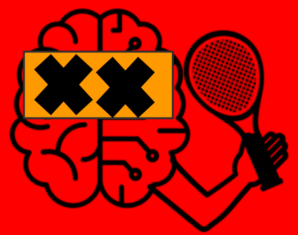
- *In a sentence*: they fail to generalize from simulation domain to real domain.
  - An analogy: training our agent to win a tennis match

In simulation:

Perfect eyesight,

Facing child

In operation:



Broken glasses,

Facing Federer

How do we beat Federer? (Make our model work in the real accelerator)

Improve training environment:
make it more similar to actually facing Federer

We do both!

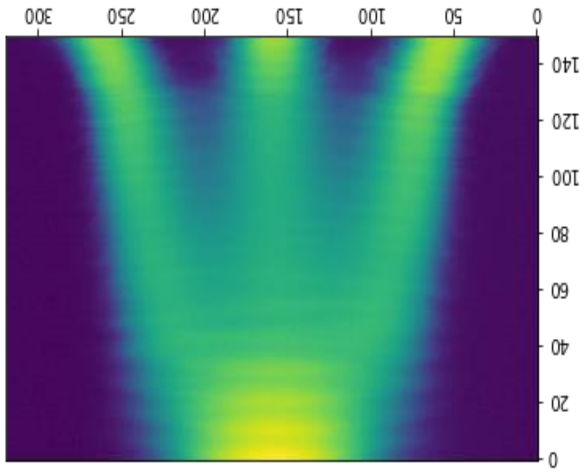
Simplify the problem:
Somehow make Federer an easier opponent…

Improving the training environment

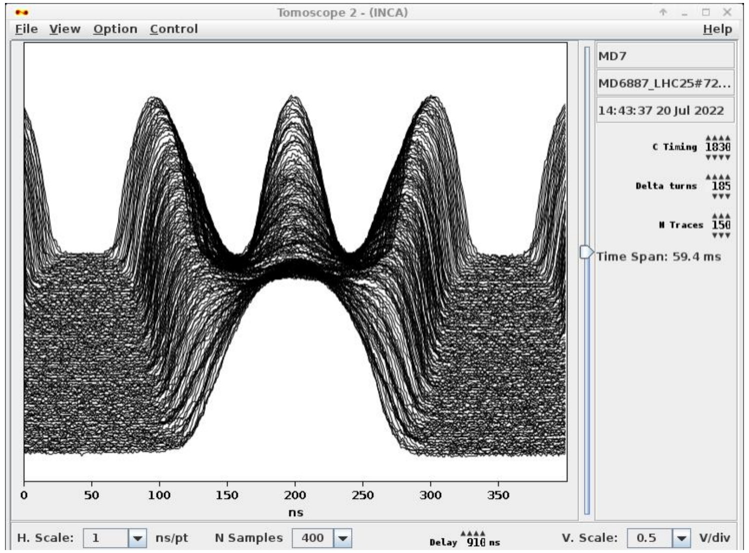
Minimize domain gap (sim2real) through data augmentation:

- Adapt simulation data to look more like real data
  - Add noise
  - transverse shifts
  - etc.

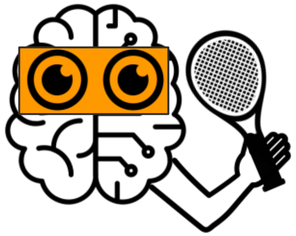
Sim. data



Real data (in control room application



Simplify the problem: Instead of beating Federer…



→

First beat Federers armless head,

→

Then Federers headless arms?

Can we simplify our task, by breaking it down to smaller, less demanding ones?

Simplify through segmentation

~~RL Agent, $\{\phi_{14}, \phi_{21}, V_{14}\}$~~ → *Optimize sequentially, phase → voltage*: RL Phase Agent, $\{\phi_{14}, \phi_{21}\}$ → RL Volt Agent, $\{V_{14}\}$

Required decoupling of phases and voltages in loss function,
→ **Possible if phase is optimized first!**

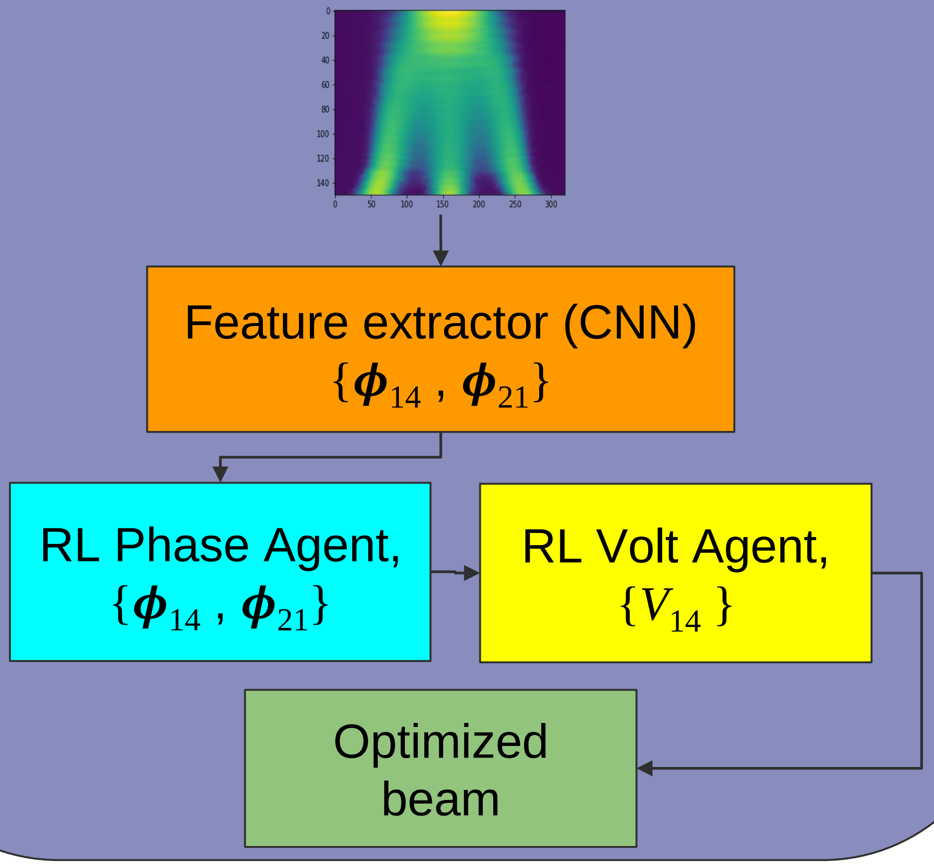
Final setup: high level view

- Final setup result of extensive testing
- Three separate ML models used in sequence
  - **CNN feature extractor**: predicts phase errors and provides good initial condition for RL agents
  - **Two RL agents** trained using Soft Actor Critic (SAC)
    - Optimizing both phases
    - Optimizing voltage

Episodic optimization with consistent success, operationally used!

For more information, see: [Reinforcement Learning applied to RF manipulation optimization in the PS. J. Wulff]()

Triple splitting setup:
One Convolutional Neural Network (CNN), two RL agents in *sequence*

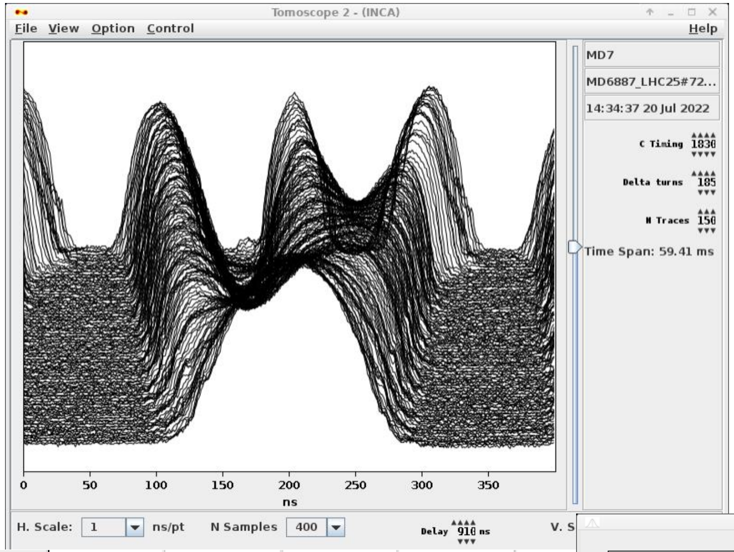


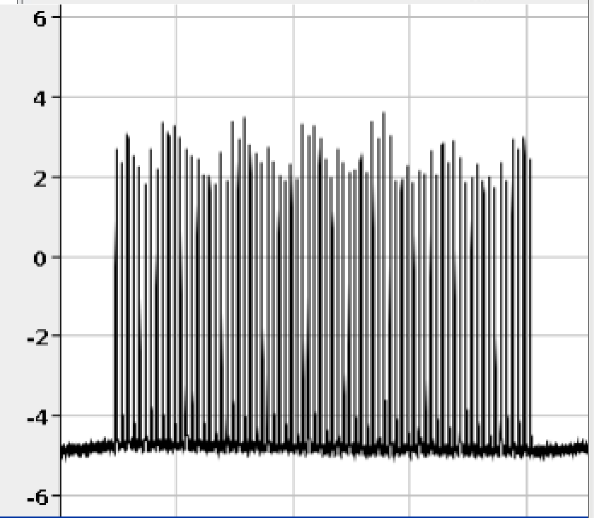
Thank you for listening! Questions?

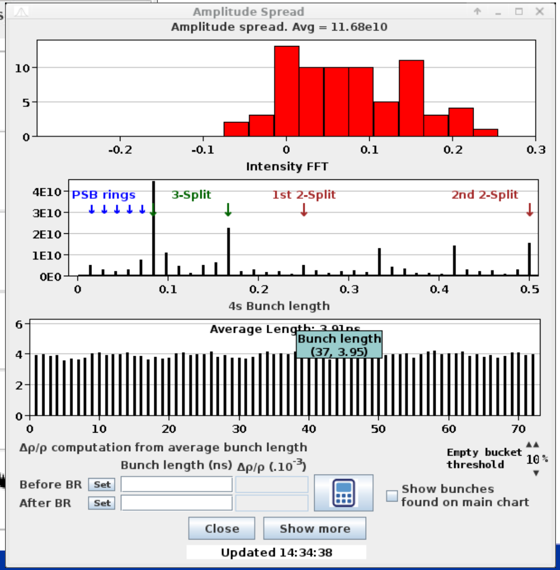
Example episode:

Approx. initial offset: ϕ_{14} = 10, ϕ_{21} = -20, V_f = 1.08

Init





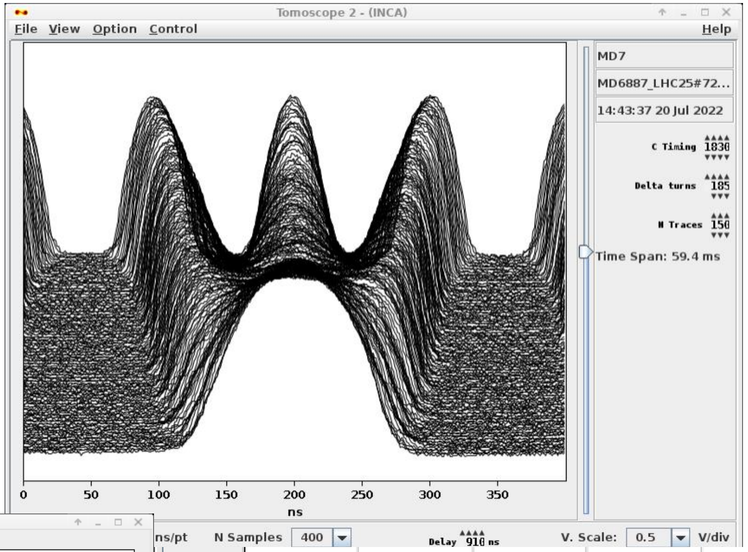


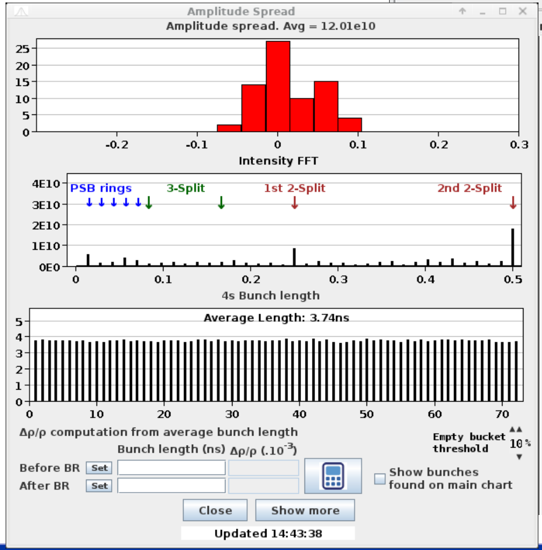
Phase opt. steps: 3

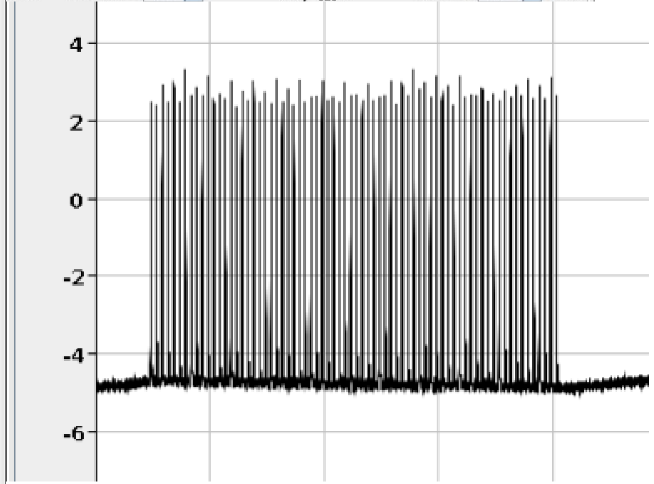
Volt opt. steps: 4

Total iterations required: 7

Final







Links and contact information

Additional information available in:

1. Reinforcement Learning applied to RF manipulation optimization in the PS. J. Wulff (March 21, 2023) · Indico (cern.ch)
2. Implementing and deploying trained neural networks through the Machine Learning Platform (MLP), J. Wulff, 2023 ML community forum
3. Reinforcement Learning Applied to Optimization of LHC Beams in the CERN Proton Synchrotron, J. Wulff, 3rd ICFA Beam Dynamics Mini-Workshop on Machine Learning Applications for Particle Accelerators
4. Progress with RL for controlling RF manipulations in the PS, J. Wulff, 2022 ML community forum
5. Reinforcement learning applied for RF manipulations in the PS, J. Wulff, 2021 ML Coffee
6. Summer student technical note - J. Wulff, 2021

Contact information

Authors: Joel Wulff, joel.wulff@cern.ch

Extra slides

Segmented RL-Agents: Setup and sim. results

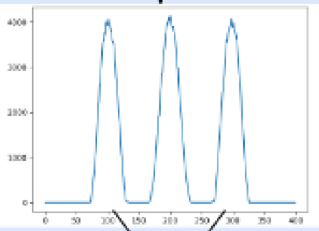
Phase Agent

Training: ~20k steps
(60k to best performance)

Performance: ~3 steps
in simulation

Environment

State



Observation: $\{bl_1, bl_3, bi_1, bi_3\}$

Reward: $r(\phi_{loss}(b_1, b_3))$

SAC-Phase-Sim2Real

Actions Limited: $\phi_{14}, \phi_{21} \in [-10, 10]$

Action: $\{\phi_{14}, \phi_{21}\}$

Work great in simulation, what about operation?

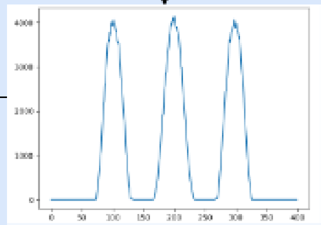
Voltage Agent

Training: ~2k steps
(20k to best performance)

Performance: ~1-2 steps
in simulation

Environment

State



Observation: $\{bl_1, bl_2, bl_3, bi_1, bi_2, bi_3\}$

Reward: $r(volt_{loss}(b_1, b_2, b_3))$

SAC-Volt-Sim2Real

Actions Limited: $v_{14} \in [-0.10, 0.10]$

Action: v_{14}

Extra: Plots of phase/voltage optimization in example episode

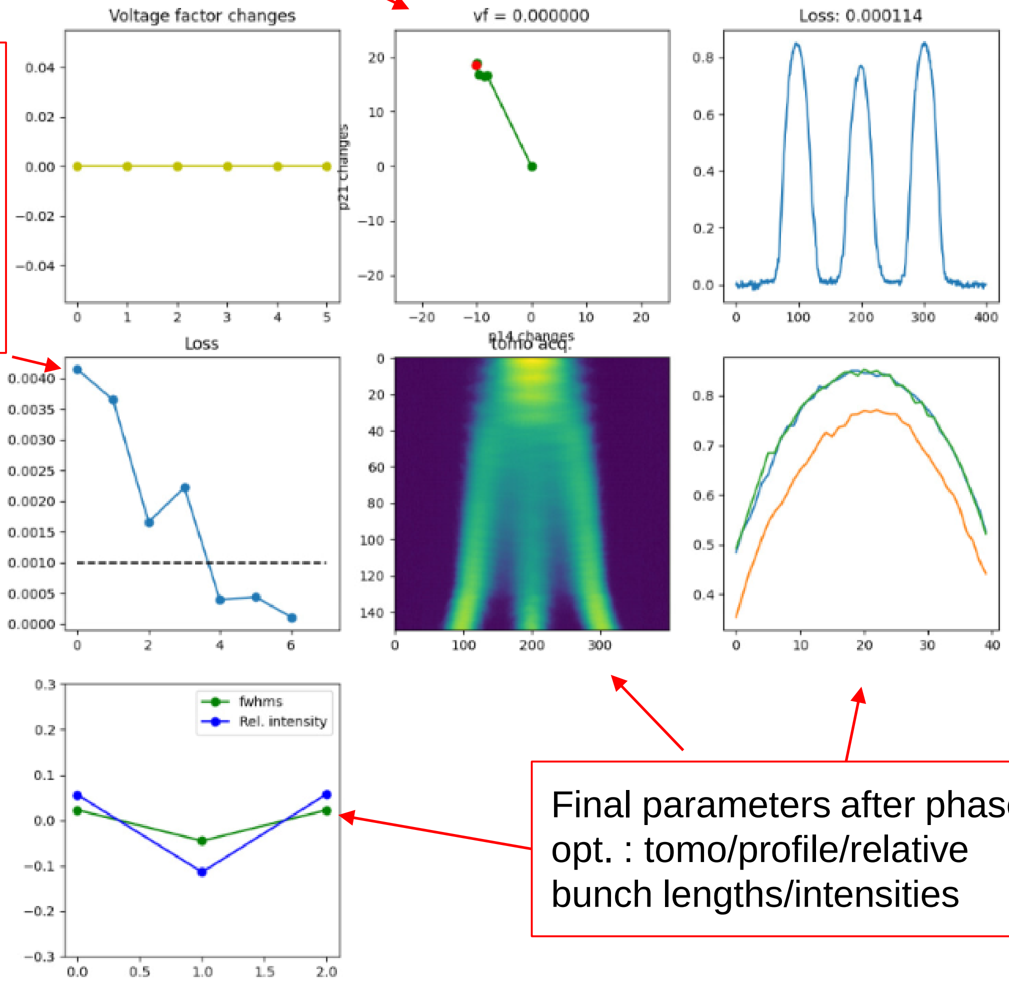
Example episode:
Approx. initial offset: p14 = 10, p21 = -20, vf = 1.08

Phase path: actions taken

Phase optimisation

NOTE: First step no action taken.

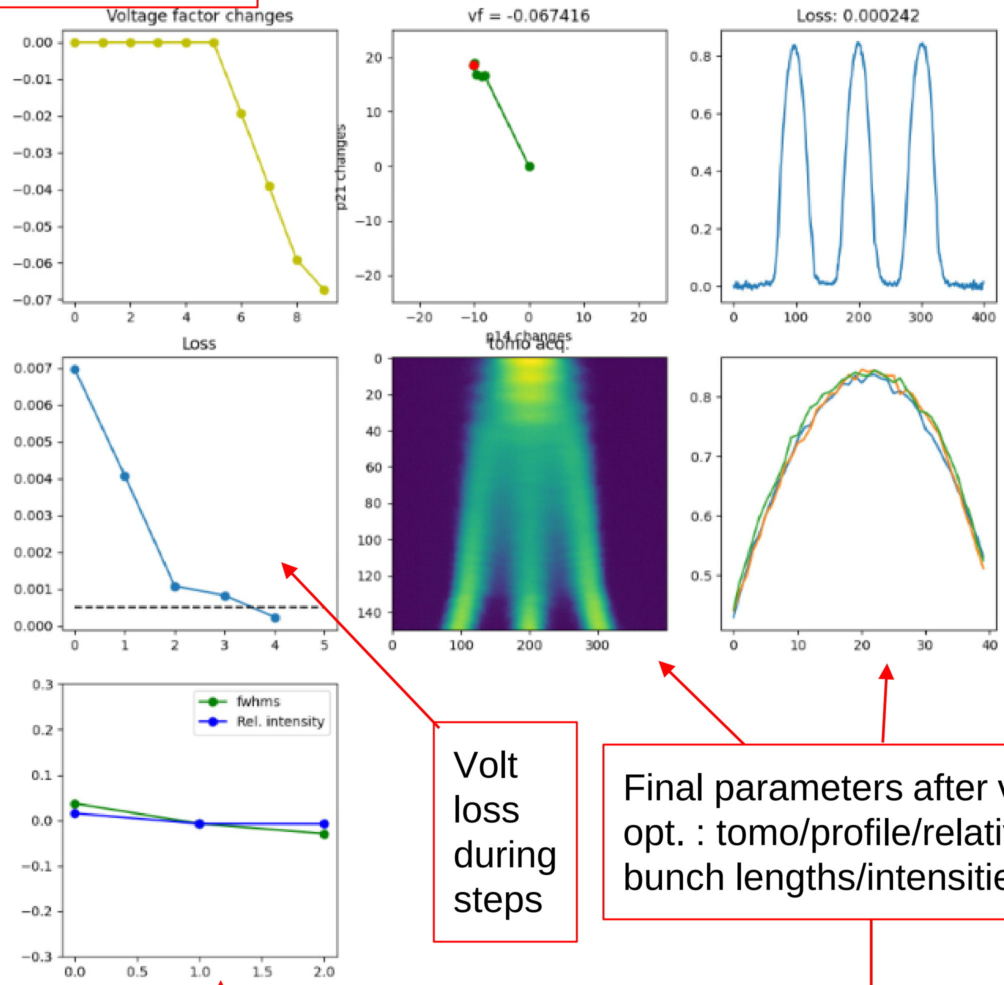
Phase loss during steps



Final parameters after phase opt. : tomo/profile/relative bunch lengths/intensities

Voltage path: actions taken

Voltage optimisation



Volt loss during steps

Final parameters after volt opt. : tomo/profile/relative bunch lengths/intensities

Results

- **<u>Triple splitting solution</u>**
  - **100% successful optimization in 60+ test episodes (on nominal 72 bunch beam)!**

- **Crucial steps for success**
  - **Enabling zero+shot transfer from simulation to real world**
    - **Great simulation**
    - **Data augmentation to simulate measurement noise, injection delays**
    - **Simplified inputs: extracted bunch lengths / intensities**

  - **Creative problem solving:**
    - **Combining different models in final optimization loop.**

The feature extractor

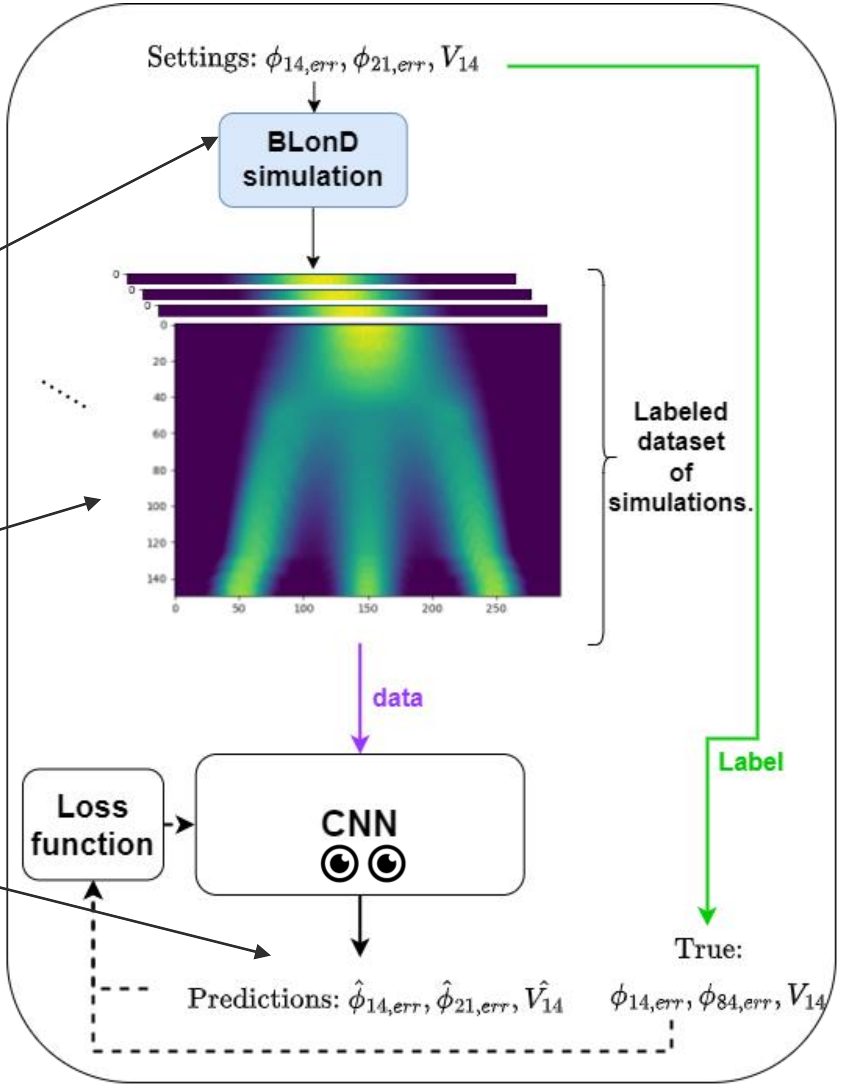
- A *supervised* **C**onvolutional **N**eural **N**etwork (CNN)

Simulated dataset using the BLonD tracking code
→ Necessary to acquire enough labeled data

Data: series of bunch profiles over time. Entire bunch evolution during splitting.

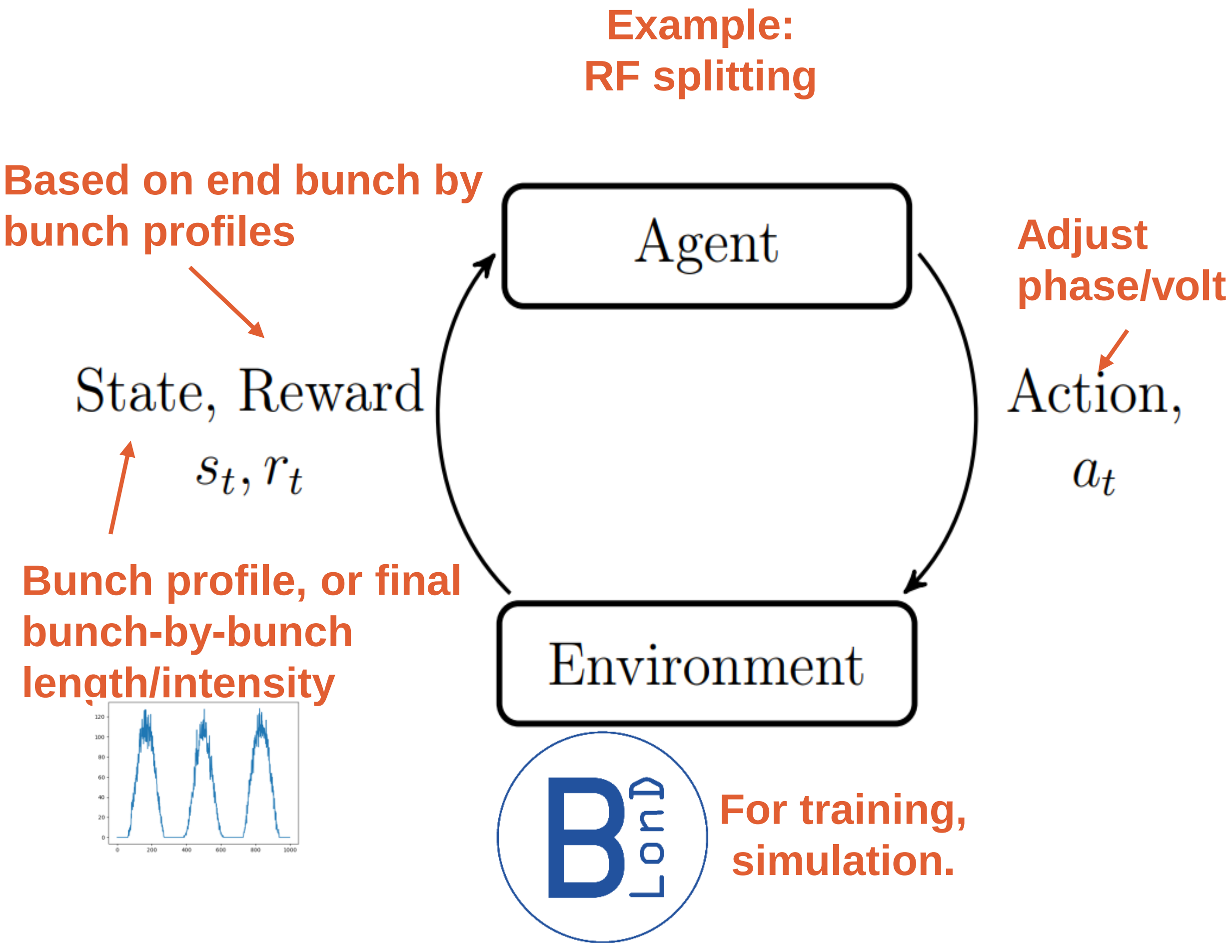
Predicts $\phi_{14,err}, \phi_{21,err}, V_{14}$.

Works in simulation, with small prediction errors!



The RL-agent

- Based on **R**einforcement **L**earning methods
    - Trained in the trial-and-error manner described by the *Agent-environment interaction loop*.
    - Agent is optimized to achieve maximum cumulative reward
    - Model-free algorithm used:
      Soft **A**ctor **C**ritic (**SAC**)
- Several versions tested.
    - In this presentation only the **final triple splitting setup** is presented.



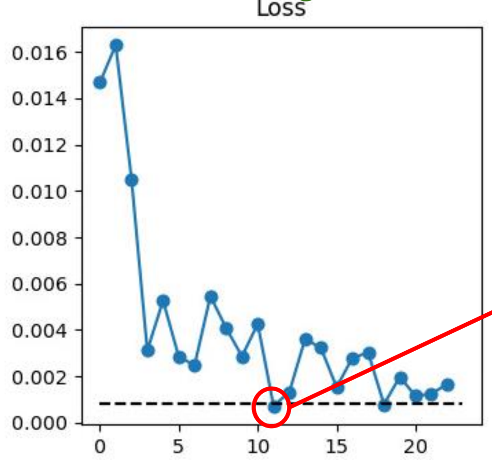
Agent-environment interaction loop

Segmented RL-Agents: direct application

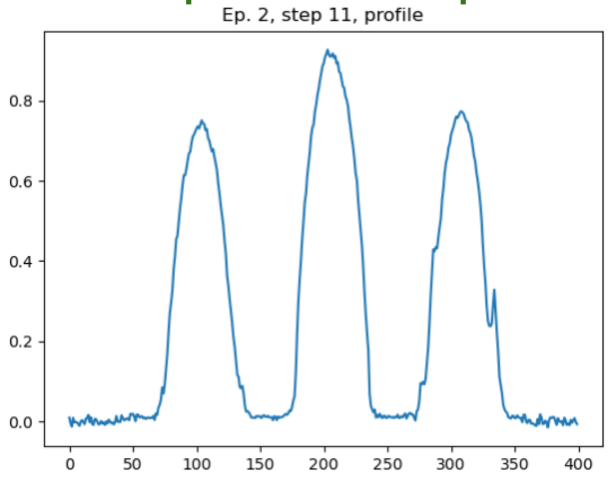
Initial test: Apply the pre-trained RL-Agents **directly** to the output from the PS, optimizing **<u>Phase</u>** → **Voltage**. No CNN used.

Unreliable → Succeeded most of the time, but not always. Why? An example…

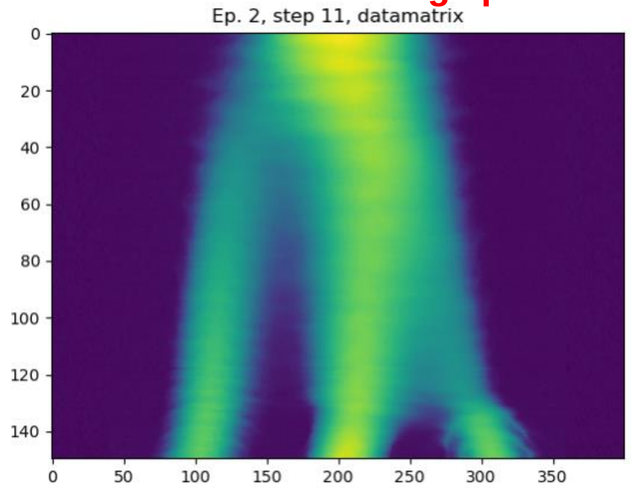
Phase loss looks good on step 11,



Final profile looks in phase



Bunch evolution shows large phase error!

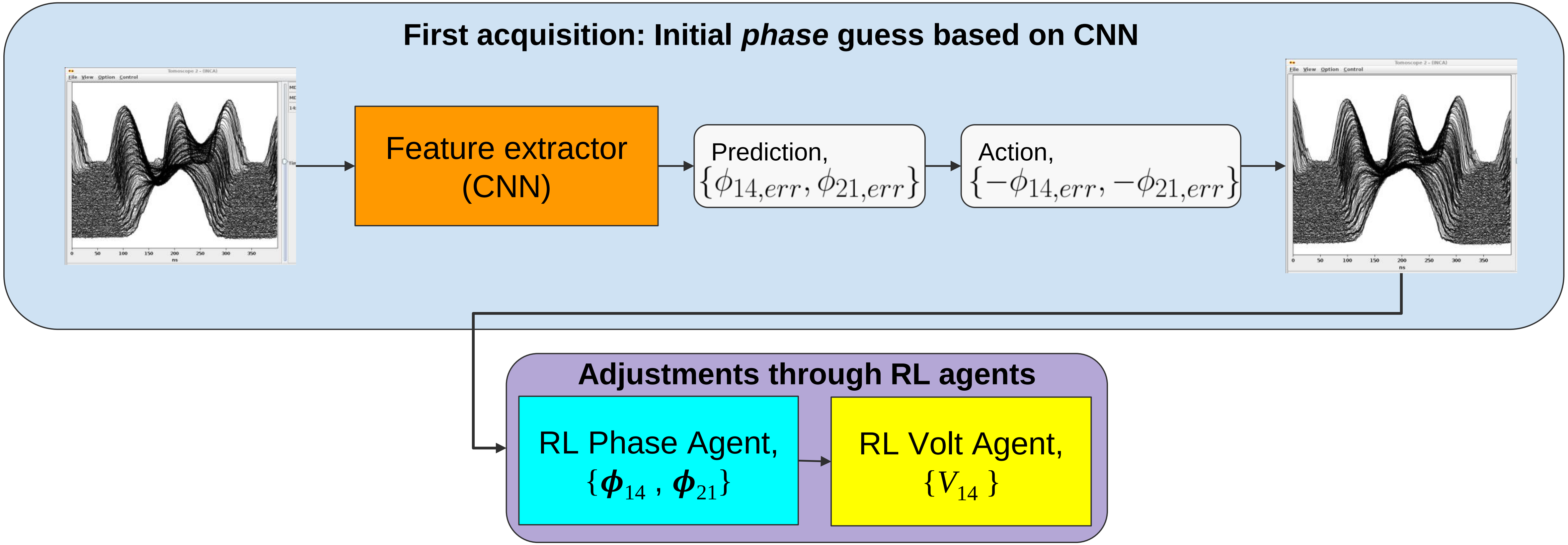


In some special cases, the information contained in final profile sometimes **not enough** to solve the problem. Could **more** information be leveraged to find a better initial condition?

→ **Yes**, by using the pre-trained feature extractor!

Segmented RL-Agents: Add initial guess from CNN

Feature extractor predicts phases given bunch profiles over the entire splitting (more info.)
→ can identify errors earlier in the bunch splitting otherwise not visible in the final profile,
→ is usually within 3-10 degrees of the true offset when predicting phase,
→ **can provide an initial guess leading to a better initial condition for the RL agents!**

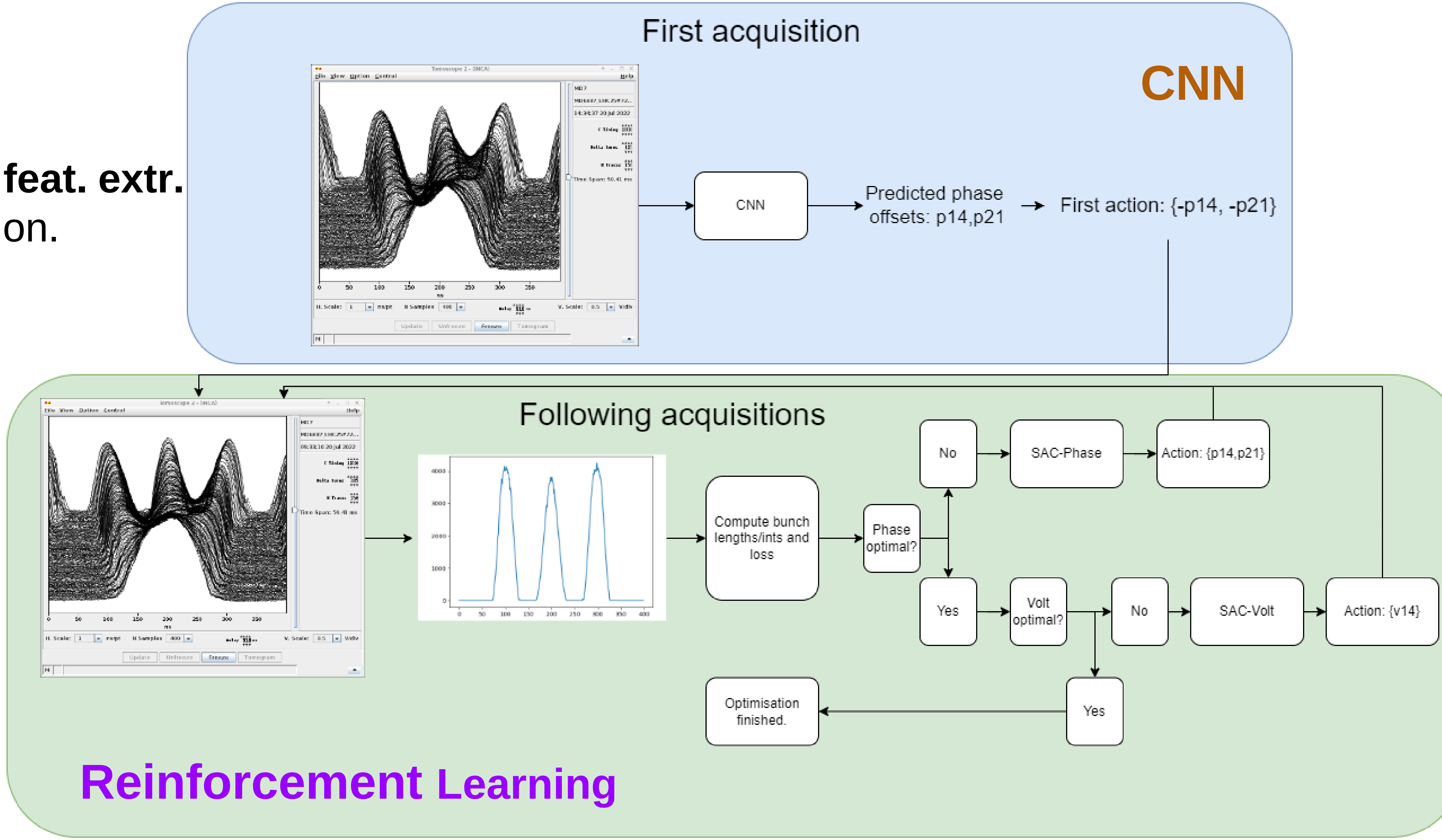


Segmented RL-Agents: Final setup

First:
→ **Initial *phase* step from feat. extr.**
Provides good initial condition.

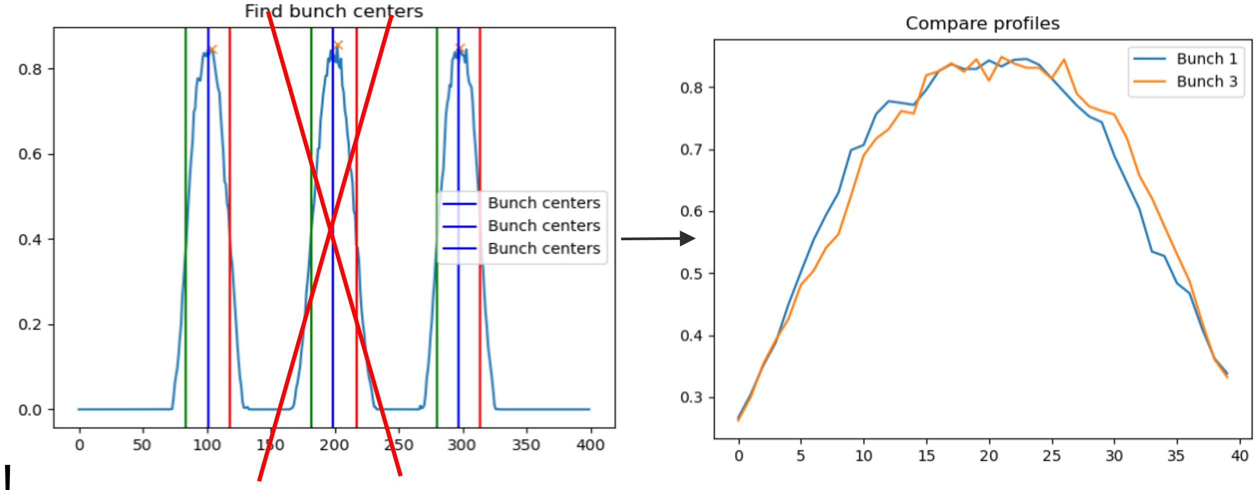
Followed by:
→ **SAC-phase optimizes phase**
→ **SAC-Volt optimizes voltage.**

How well does it work?



Extra: The phase and voltage losses

Figure: Illustration of phase loss. Isolated outer bunches are compared through MSE.



1. Phase Loss:
Compare only the outer two bunches. From beam dynamics, we know that for almost all combinations of phase offset and voltage factor we will observe a difference in their shapes.

With optimal phase, they should **always** be identical!
Gives a semi-voltage agnostic loss.

2. Voltage Loss: Assume phase is already optimized,
→ Optimization reduced to a univariate problem.

Reuse original three-bunch comparison,
→ Provides a nice, approximately parabolic loss!

Note: See the extra slides for a scan of phase losses for phase errors at different fixed voltages.

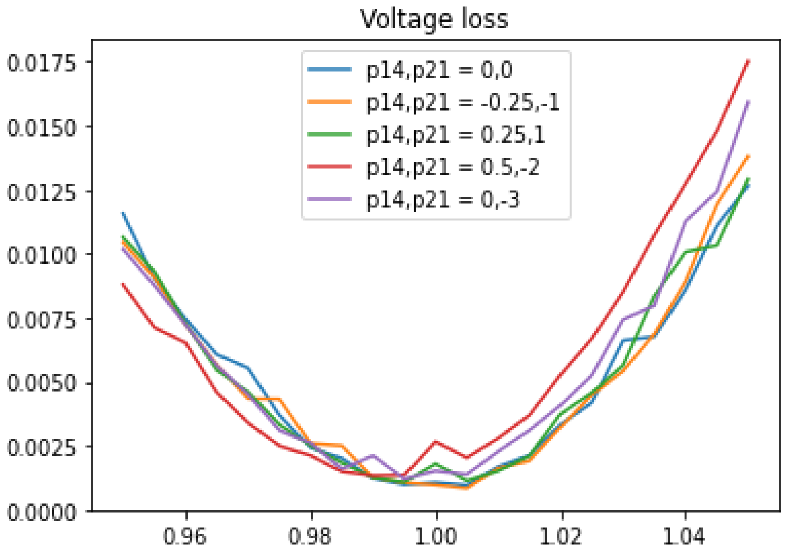


Figure: Scan of profile loss as a function of voltage factor for small residual phase errors.

Extra: Judging splitting quality, the loss function

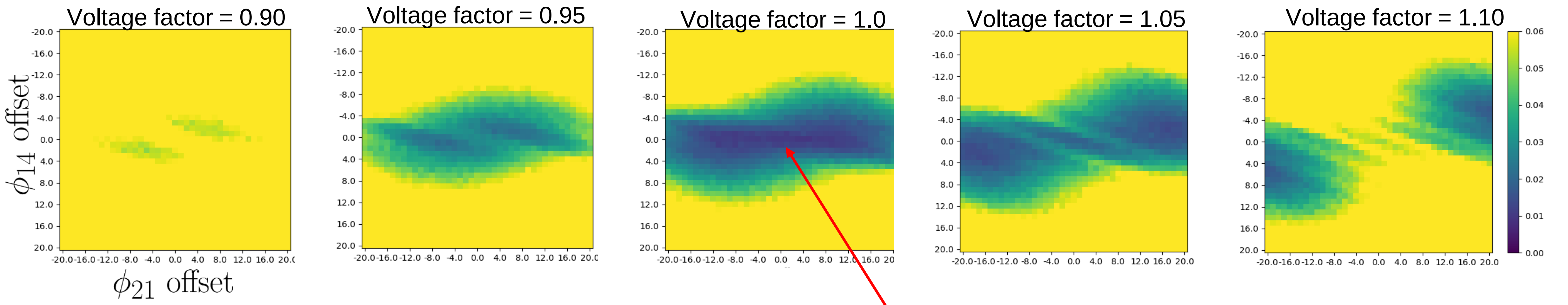


Figure: Clipped losses for different fixed voltages: when voltage is changed, optimal phase also changes.

- Scan of the three-bunch loss values while varying phase errors at **fixed** voltages
  - Shows how the "optimal" phase varies with the voltage setting.
  - Compare with phase loss on next slide!

Note: the "true" minimum over these different settings is still located in voltage factor 1.0 and phases 0, 0, as expected.

Extra: The phase loss, scanning phases for set voltages

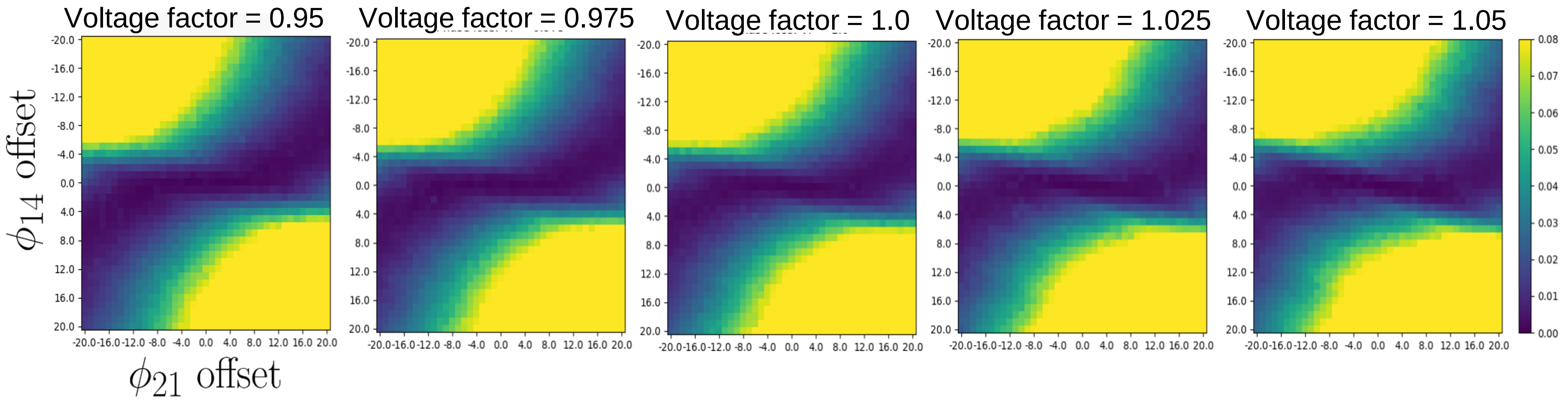


Figure: Clipped phase losses for different fixed voltages: when voltage is changed, optimal phase also changes.

- With the phase loss function, we no longer see the same variation in the loss landscape when varying voltage: as expected, the loss is (semi-) voltage agnostic.

Note: The quality of the triple splitting is much more dependent on the p14 phase setting than the p21.

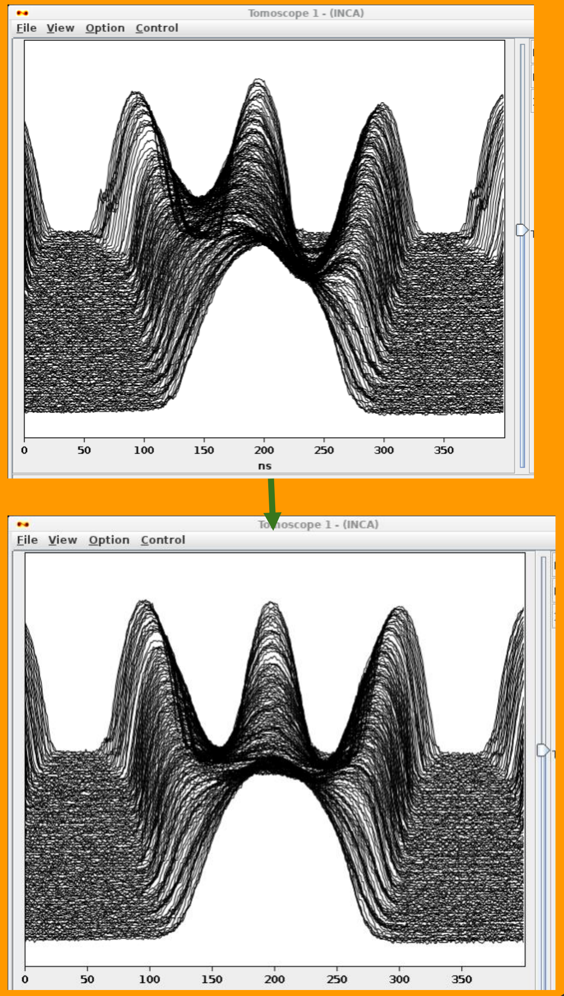
Example: Segmented RL-Agents only (no init. guess from CNN)

- Three initial episodes ran with the setup described in slide __
  - Two successes, one failure.
  - Generally slower than desired (>10 steps).

Episode	Init settings [p14,p21, v14_offset]	Phase opt.	Voltage opt.	Total steps	Comment	Success
1	-15,5, -0.07	12	3	15		Yes!
2	20,-20,-0.10	22+	-	n/a	Did not finish. Failed to optimise phase to a good degree.	No.
3	10,-10,-0.10	10	12	22		Yes!

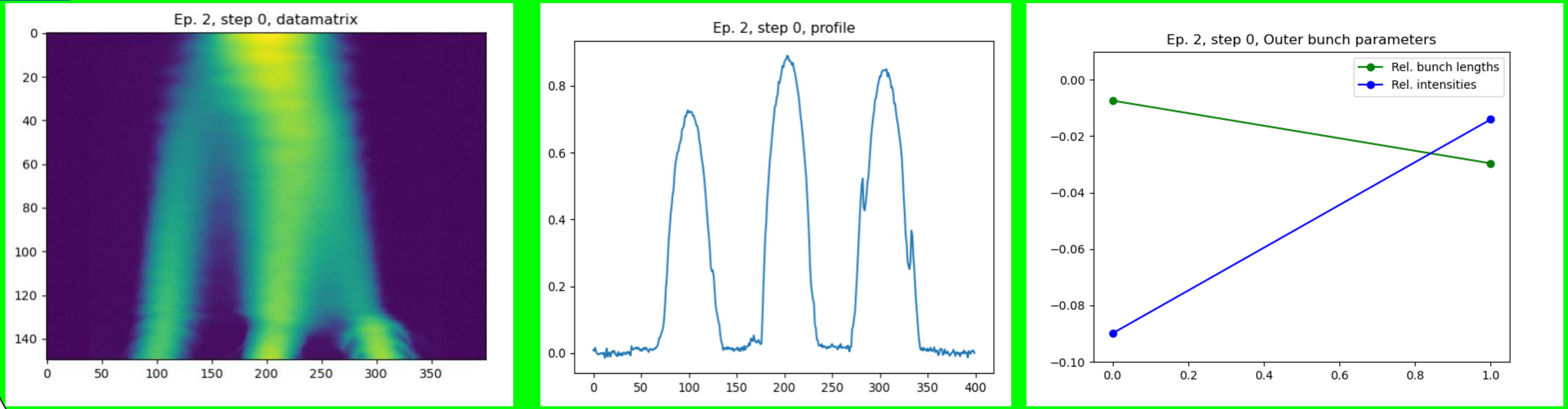
Why did the agents fail in this episode?
→ Explored in next slide

Figure: Init and end tomoscope acq. of ep. 1.

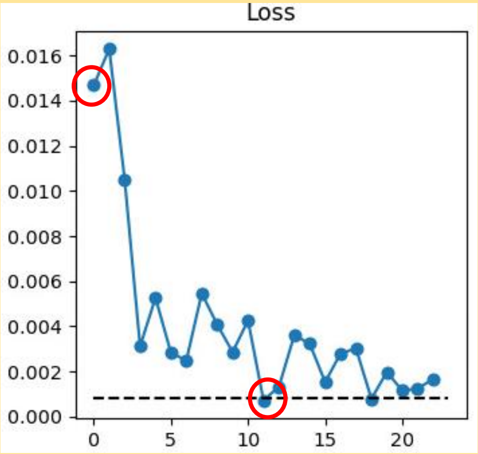


Example: Segmented RL-Agents only (no init. guess from CNN)

Initial acquisition: Start offset 20, -20, -0.10. Initial tomo looks very poor, final profile looks less poor.

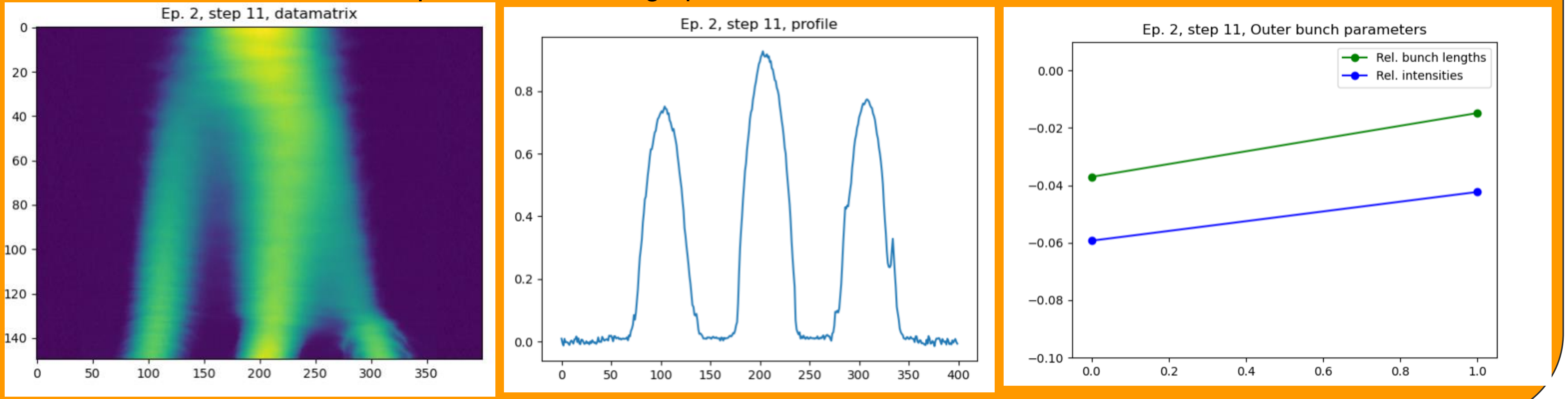


Phase loss during optimisation: We will look closer on steps 0 and 11.



Agent believes it is close to the minimum, but is actually far from it. A special case where the agent can get stuck!

Step 11: Phase loss below criteria, final profile has similar outer bunches. Rel. bunch lengths/intensities also similar. But, tomo acquisition shows large phase error remains → Error in observable!



Extra: The datasets

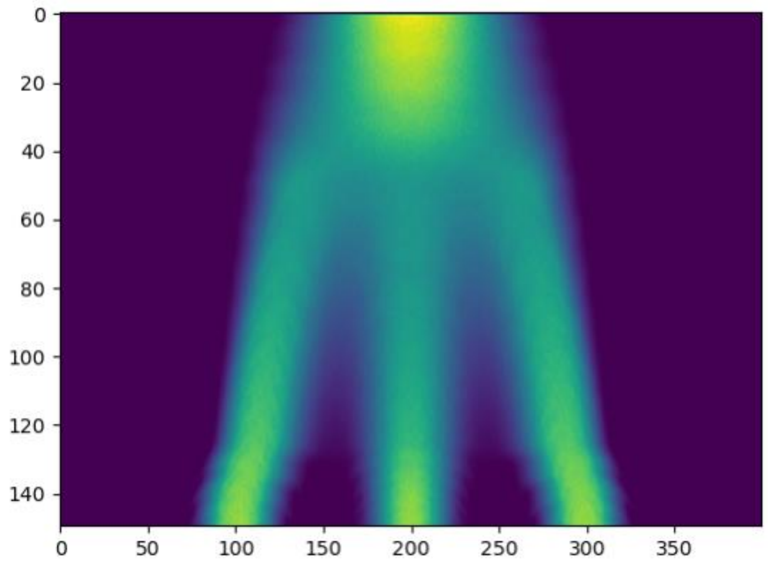
- **The Triple splitting dataset:**
  - Scan of absolute phase errors in range **$\phi_{h=14}$, $\phi_{h=21}$, = [-20,20]**, and voltage factors for *h=14* in range **$v_{h=14}$ = [0.95, 1.05]**.
  - A total of 59541 samples in dataset.

- Each sample stores **the entire datamatrix** of traces along with **the label** of the offset used to simulate it.

- A 9:1 training/validation split was used.

- Note: These same datasets are used for training of RL agents, but only extracted features such as end bunch-by-bunch length/intensities are given to the agents.

Datamatrix (150x400)

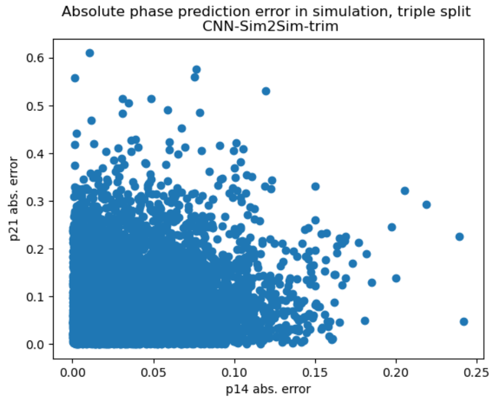


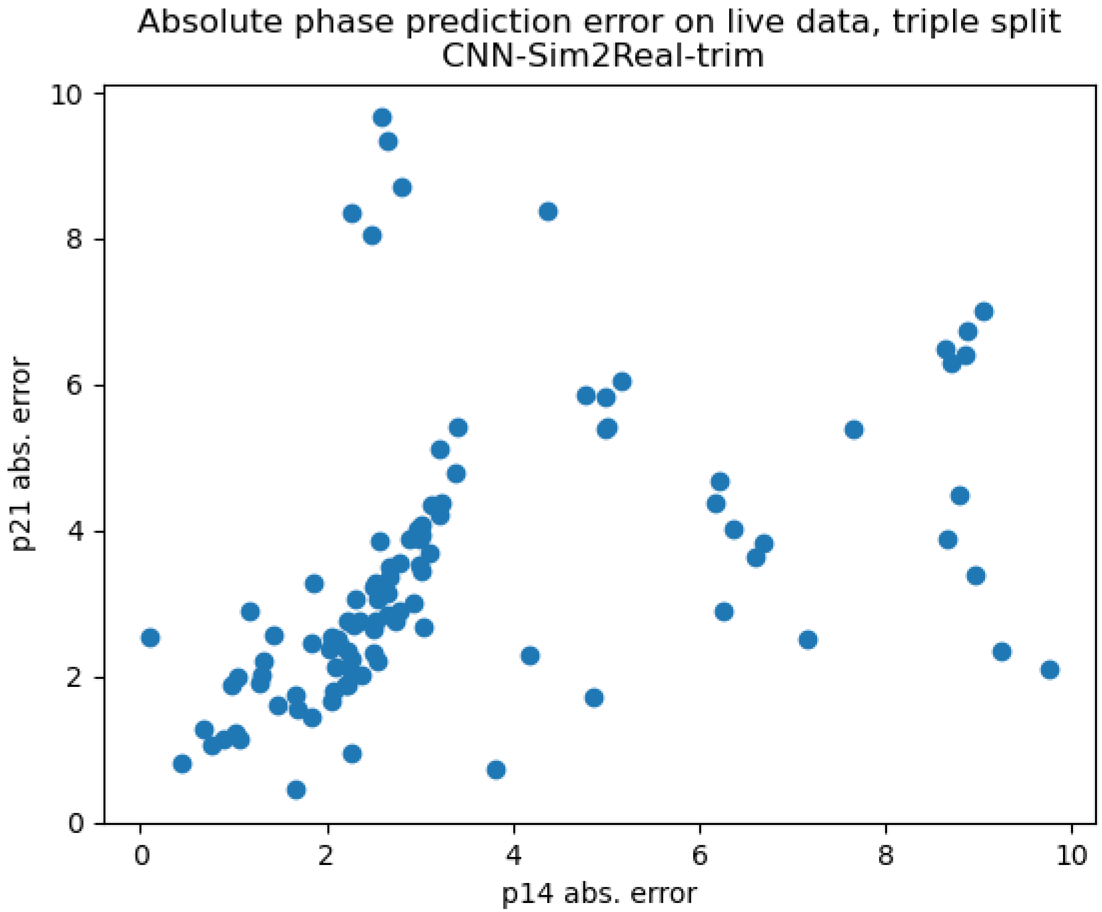
Label = [0, 0, 1.0]

Feature extractor performance on real trisplit data

- Problem: CNN fails to generalise and is not accurate on real data.
  - Error is however most often only ~3 degrees, which means it can improve on large phase errors.
  - However, finetuning of phase becomes difficult. The agent is pre-trained with an almost perfect CNN, and trusts it too much

Compare with simulation accuracy <1 degree.

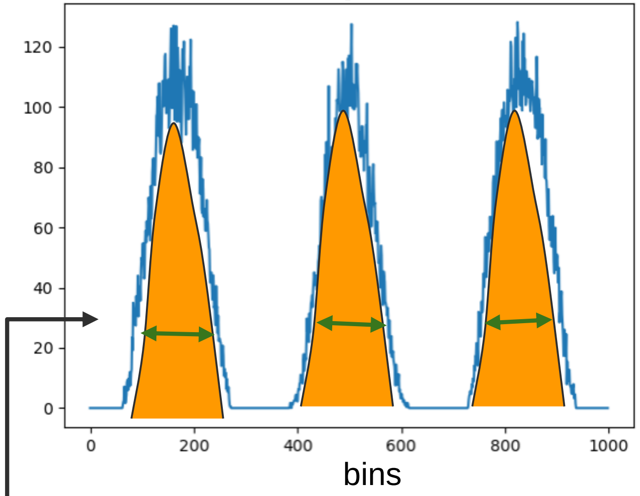




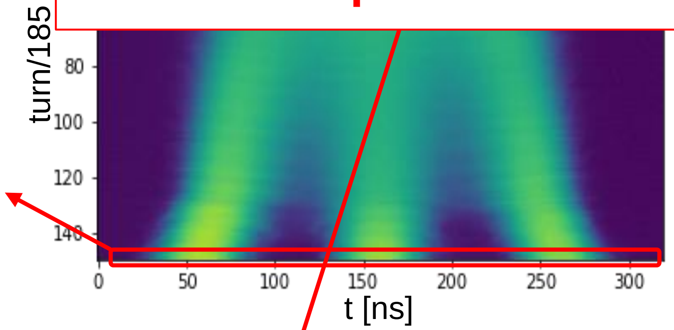
Longitudinal triple splitting: Parameters, Observables and Goal

- **Triple split → from 1 bunch to 3 longitudinally**
  - RF cavities on 3 different harmonics pulsed at the same time to accomplish.
- **Parameters (to optimize):**
  - Phases and voltage of 2/3 RF cavities, ϕ_{14}, ϕ_{21} **, and** V_{14} **.**
- **Observables:**
  - Final bunch **profiles**, final bunch-by-bunch **length** + **intensity**.
- **Goal:**
  - All bunch-by-bunch observables equal after splitting.

Final profile



Three **simultaneously** active cavities with different voltages → **Non-linear** interactions, **difficult to optimize**...



RF voltage program