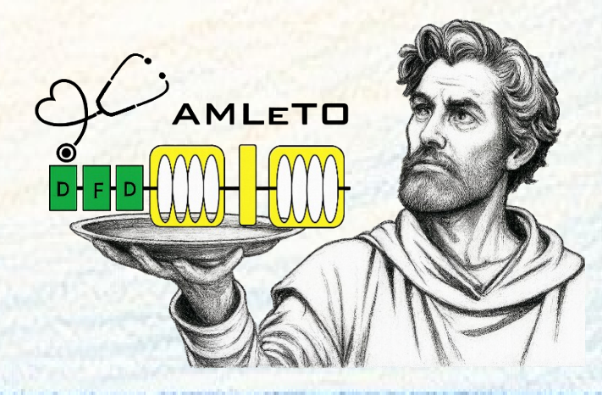
Extremely Fast Modular Neural Surrogates for Reinforcement Learning–Based Accelerator Tuning

Author: D. Zebele, M. Montis, L. Bellan, Department of Information Engineering, UniPd, Italy, also at INFN-LNL Legnaro, Italy

INTRODUCTION

In this work, is presented a **modular surrogate model** that emulates the **TraceWin** simulator with good accuracy (~0.1 $\sigma_{dataset}$ mean error) and extremely fast (~1ms vs 20s per simulation). Leveraging this surrogate, a **reinforcement learning agent (PPO)** is trained to optimize beam quality by adjusting lattice parameters. The agent is trained on the surrogate and tested on TraceWin, providing **instantaneous decisions** unlike Bayesian Optimization approach. Small incremental steps ensure robustness, prevent exploitation of surrogate biases, and allow the agent to adapt continuously to the actual system.
This approach enables **autonomous beam optimization**, virtual diagnostics for each module, and opens the door to future **gradient-based optimization and real-time accelerator control**.

Advance Machine LEarning Transport Optimization Group (AMLETO)

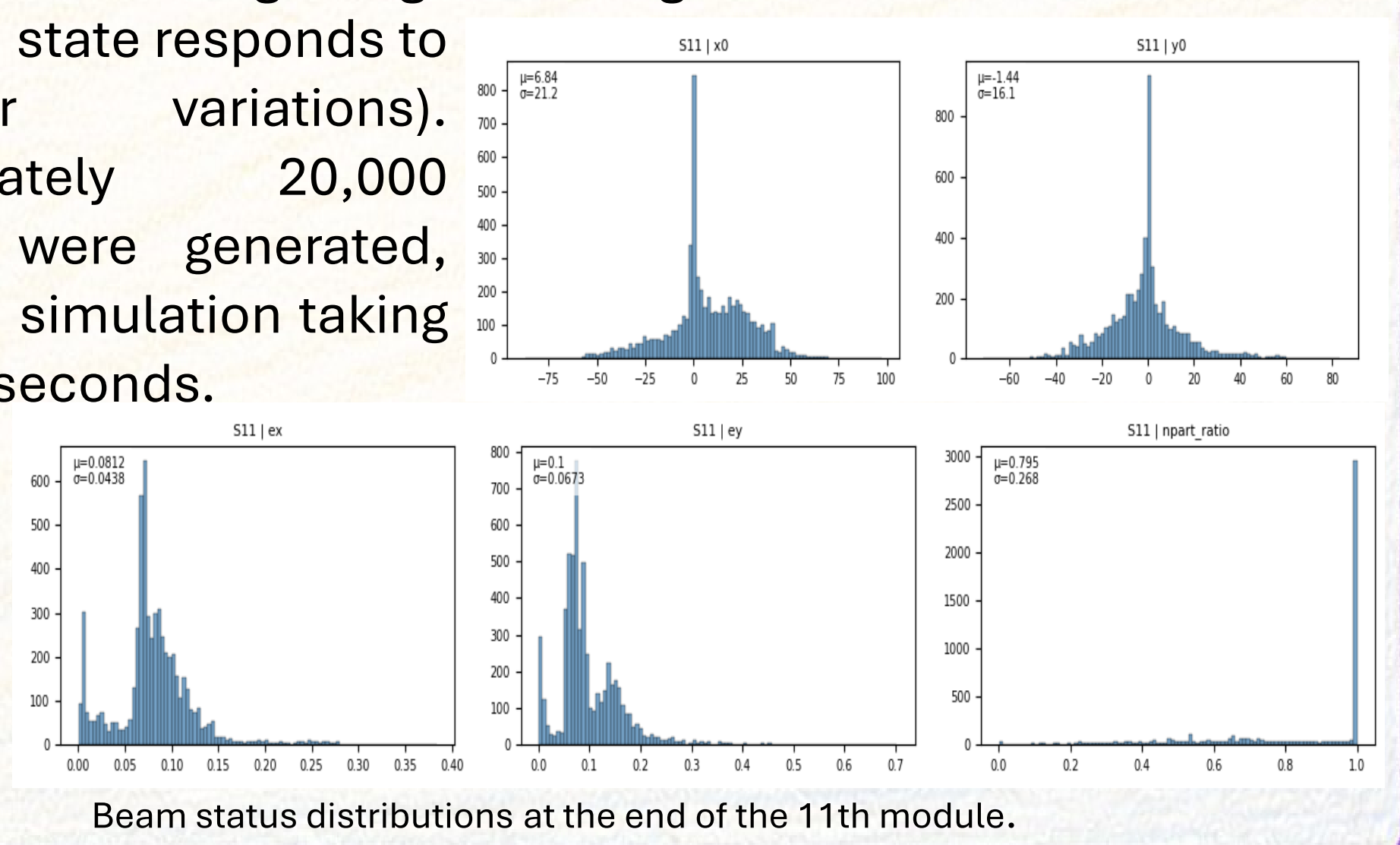


ARCHITECTURE

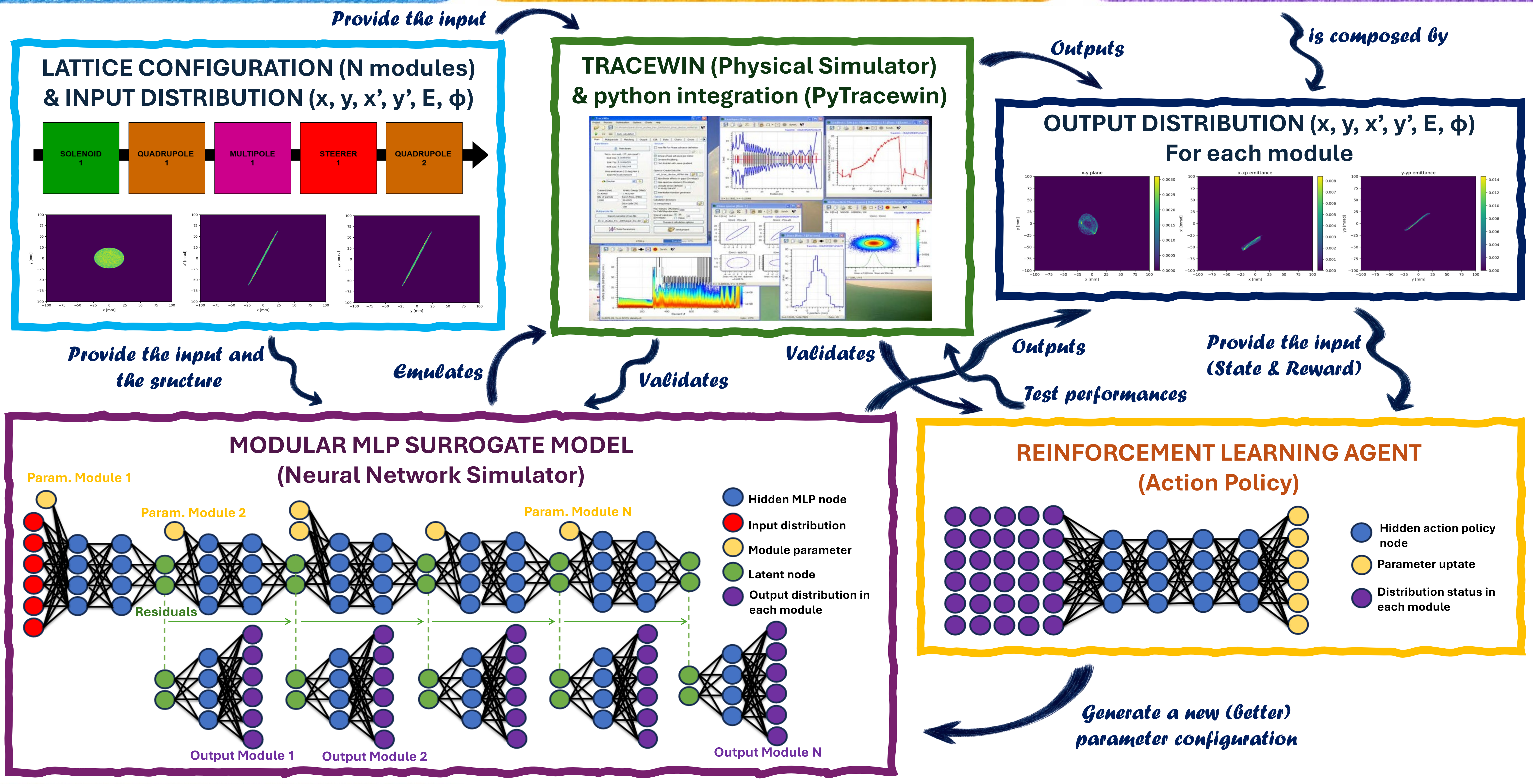
Surrogate model take as input the initial beam distribution and consists of multiple stages, each corresponding to a lattice module with a variable number of configurable parameters. Each stage consists in a MLP that takes as input the latent vector from the previous stage combined with the module parameters and outputs an updated latent vector. A **residual architecture** is used: each stage predicts the difference relative to the previous latent vector, facilitating incremental beam transformations. For each stage, a different MLP maps the latent vector to the predicted output distribution at the end of each module. In the present setup, the lattice has **11 modules** and **16 total parameters**. Each stage is composed by **two hidden layers of 256 neurons**, producing a **latent space of size 64**, while the **output network** has **two hidden layers of 128 neurons**. Inputs and outputs correspond to the **beam distribution state** (centroid, emittances, transmission), though the framework can be extended to the full particle distribution using the latent vector as a **bottleneck knowledge distillator.**

DATASET

The dataset was initially generated using a fixed input distribution and lattice, sampling from a multivariate Gaussian centered on a default parameter configuration, with standard deviations chosen based on parameter sensitivity (i.e., how strongly each parameter influences the system). It was later expanded by incorporating data collected during RL agent testing.
The beam state responds to parameter variations). Approximately 20,000 samples were generated, with each simulation taking about 20 seconds.

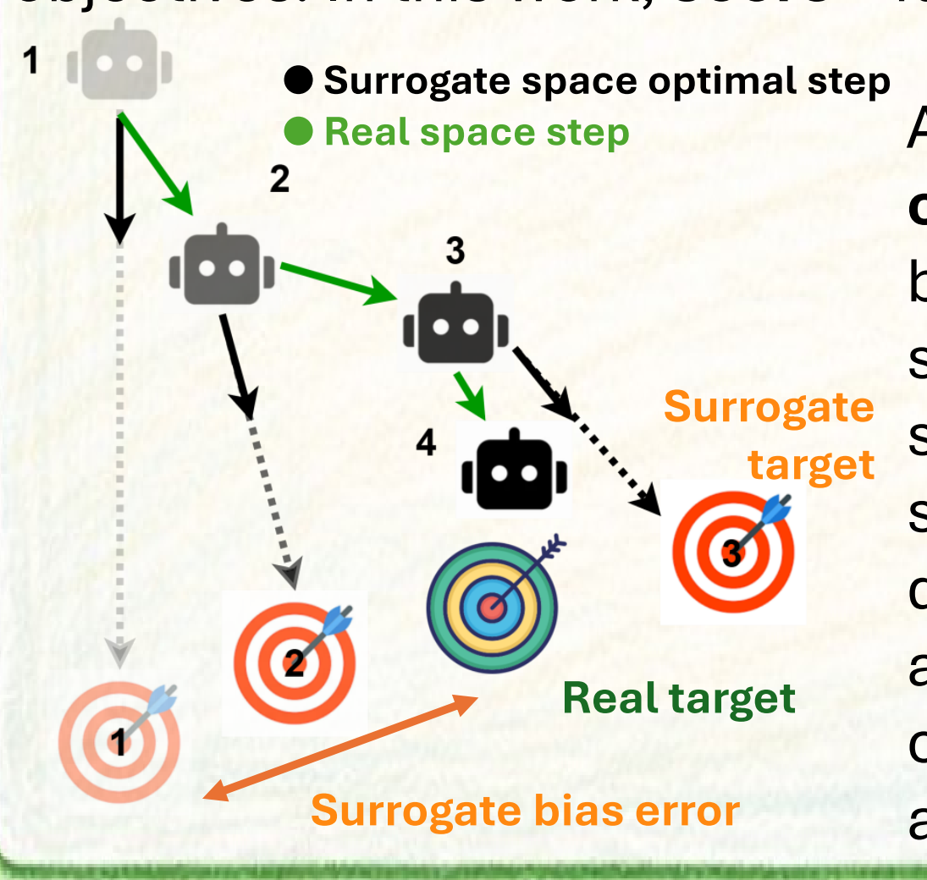


Beam status distributions at the end of the 11th module.


Provide the input
LATTICE CONFIGURATION (N modules) & INPUT DISTRIBUTION (x, y, x', y', E, ϕ)
SOLENOID 1
QUADRUPOLE 1
MULTIPOLE 1
STEERER 1
QUADRUPOLE 2
TRACEWIN (Physical Simulator) & python integration (PyTracewin)
Outputs
is composed by
OUTPUT DISTRIBUTION (x, y, x', y', E, ϕ) For each module
x-y plane
x-xp emittance
y-yp emittance
Provide the input and the sgructure
Emulates
Validates
Validates
Outputs
Test performances
Provide the input (State & Reward)
MODULAR MLP SURROGATE MODEL (Neural Network Simulator)
Param. Module 1
Param. Module 2
Param. Module N
Residuals
Output Module 1
Output Module 2
Output Module N
Hidden MLP node
Input distribution
Module parameter
Latent node
Output distribution in each module
REINFORCEMENT LEARNING AGENT (Action Policy)
Hidden action policy node
Parameter uptate
Distribution status in each module
Generate a new (better) parameter configuration

TRAINING

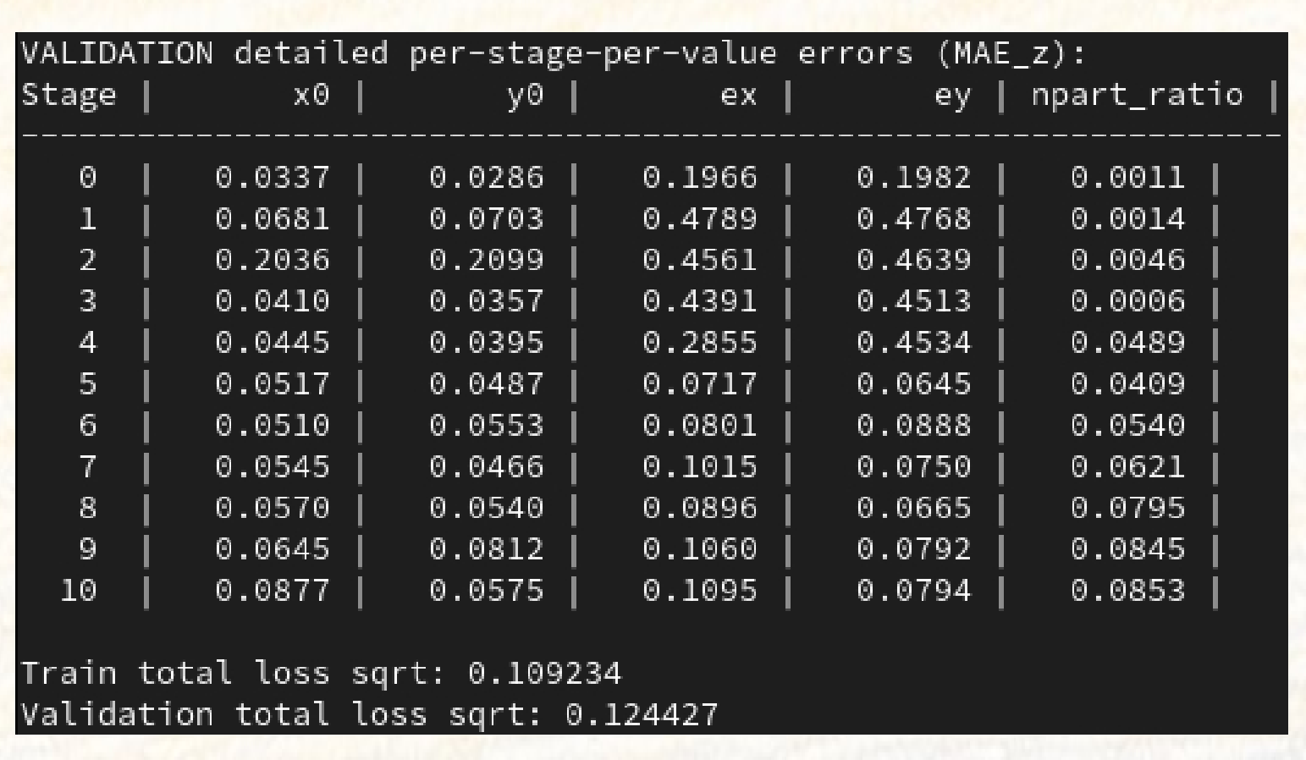
The **SURROGATE MODEL** was trained with **Adam (lr=3e⁻⁴), ReLU, layer and in/out normalization**, and **dropout (p=0.2)**. The dataset was split **80/20** for training and testing, and a **weighted MSE loss** was used to prioritize specific modules or beam properties.
RL AGENT was trained using **PPO** from Stable Baselines3 in a **custom environment** interfacing with the surrogate.
Initial states were generated by adding Gaussian noise on a randomly selected dataset element, with σ=$\sigma_{dataset}$/2.
The agent is trained to **maximize a user-defined score function** based on beam properties, allowing flexible operational objectives. In this work, **score** = ratio * 100 - ε_x * 100 - |μ_x| - |μ_y|



Actions consist of **small updates of the lattice parameters**, bounded by a fraction of their sensitivities. These incremental steps limit exploitation of surrogate biases and, when deployed on a simulator or real accelerator, enable near-continuous feedback and adaptive correction.

RESULTS SURROGATE

The surrogate model reproduces the TraceWin simulator with a **mean error of ~$0.1\sigma_{dataset}$** after 200 epochs, with slight variations across lattice stages due to the weighted loss. It has also been observed that, as the size of the dataset increases, the loss consistently decreases.
Its **inference time is ~1 ms**, compared to ~20 s for TraceWin (**20,000× faster**), enabling RL training. Once fine-tuned with real accelerator measurements, it could provide **near–real-time virtual diagnostics at each stage**. The model is also fully differentiable, enabling gradient-based optimization strategies.



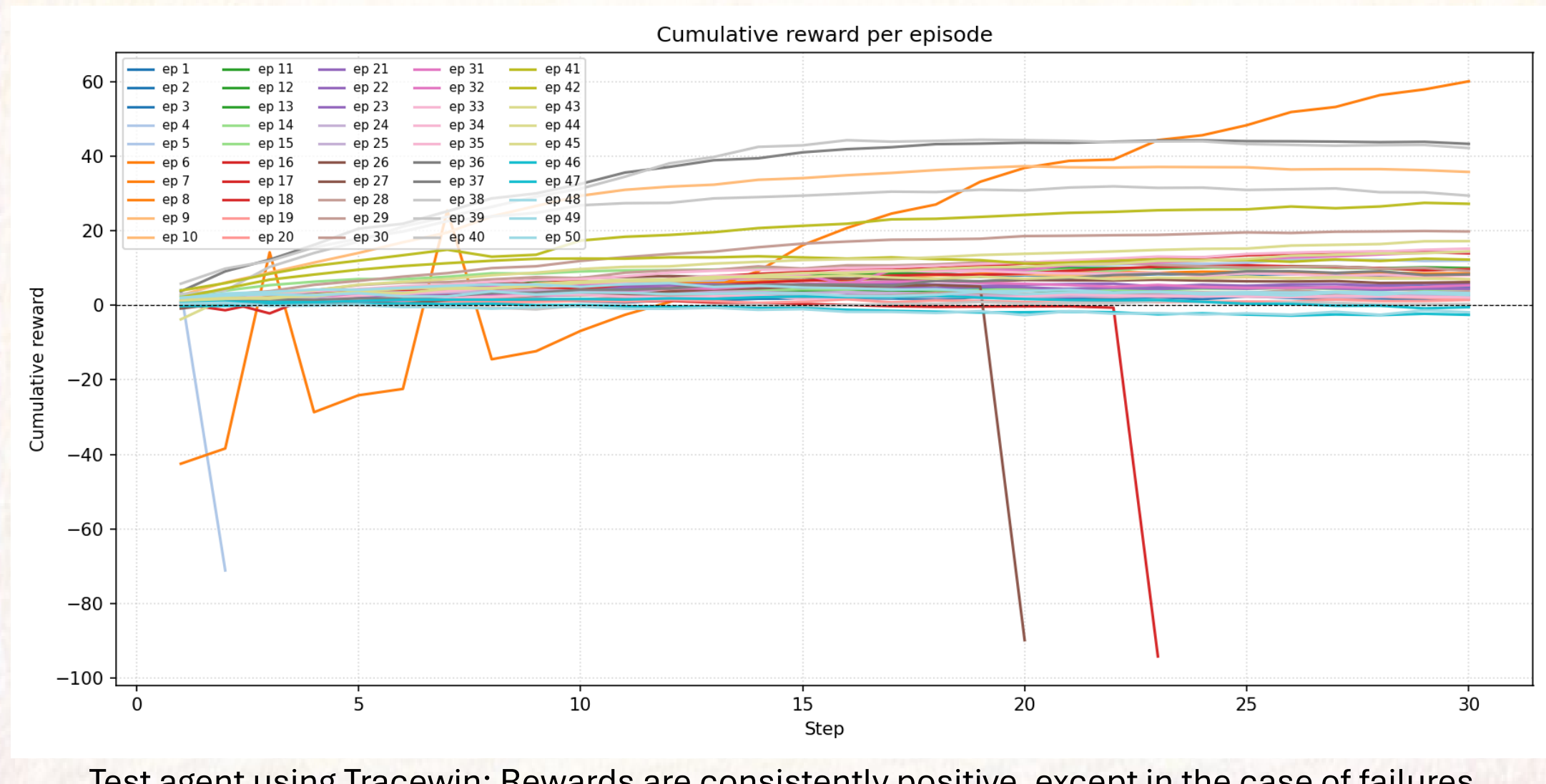
```
VALIDATION detailed per-stage-per-value errors (MAE_z):
Stage |      x0 |      y0 |      ex |      ey | npart_ratio |
-----------------------------------------------------------
  0   |  0.0337 |  0.0286 |  0.1966 |  0.1982 |   0.0011 |
  1   |  0.0681 |  0.0703 |  0.4789 |  0.4768 |   0.0014 |
  2   |  0.2036 |  0.2099 |  0.4561 |  0.4639 |   0.0046 |
  3   |  0.0410 |  0.0357 |  0.4391 |  0.4513 |   0.0006 |
  4   |  0.0445 |  0.0395 |  0.2855 |  0.4534 |   0.0489 |
  5   |  0.0517 |  0.0487 |  0.0717 |  0.0645 |   0.0409 |
  6   |  0.0510 |  0.0553 |  0.0801 |  0.0888 |   0.0540 |
  7   |  0.0545 |  0.0466 |  0.1015 |  0.0750 |   0.0621 |
  8   |  0.0570 |  0.0540 |  0.0896 |  0.0665 |   0.0795 |
  9   |  0.0645 |  0.0812 |  0.1060 |  0.0792 |   0.0845 |
 10   |  0.0877 |  0.0575 |  0.1095 |  0.0794 |   0.0853 |

Train total loss sqrt: 0.109234
Validation total loss sqrt: 0.124427
```

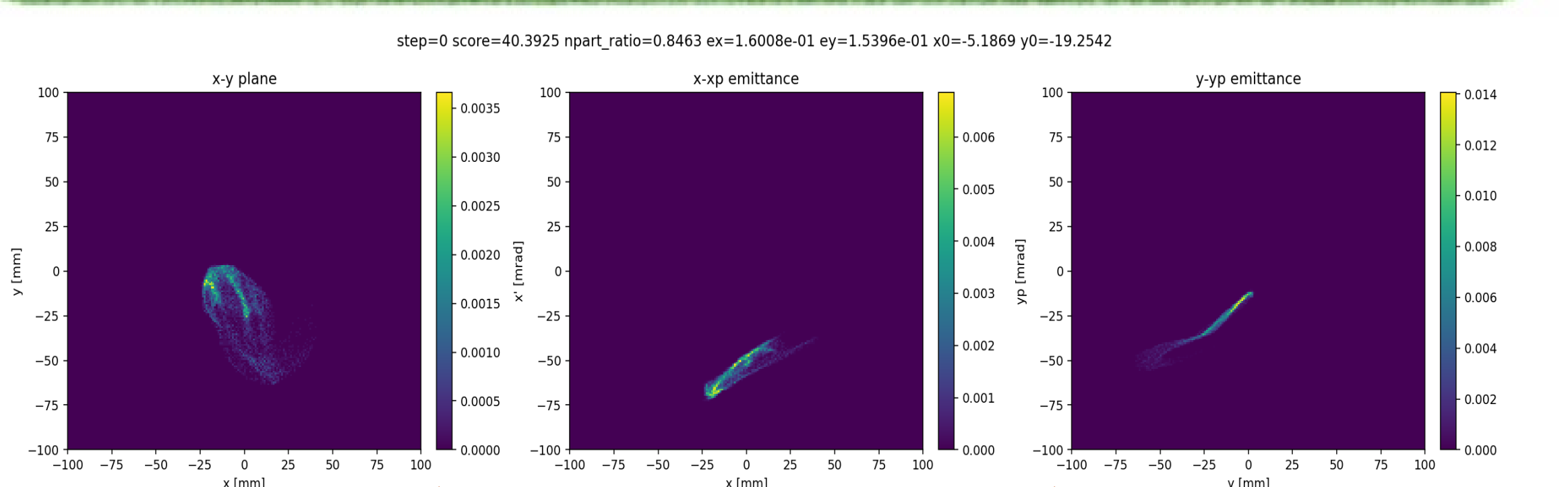
Mean errors for each output of the surrogate model (w.r.t $\sigma_{dataset}$)

RESULTS RL AGENT

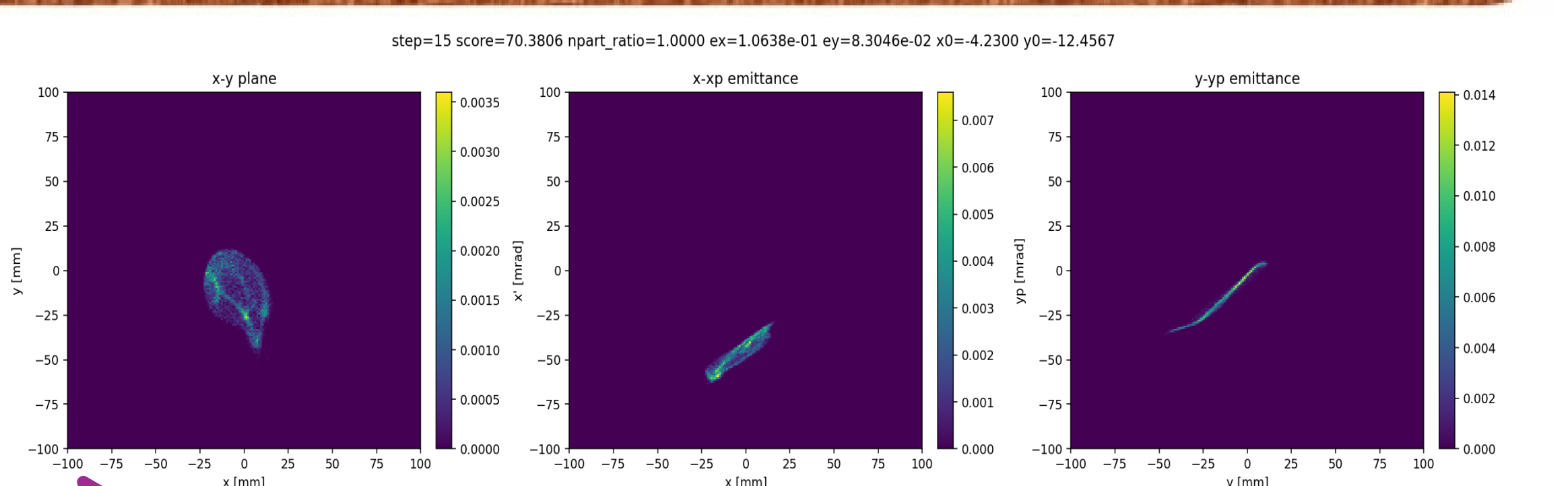
The RL agent was **trained on the surrogate model** and **tested on TraceWin**, providing **instantaneous parameter adjustments** unlike iterative methods such as Bayesian optimization or PSO. Performance depends on initial conditions and action constraints, but average **rewards remain positive**, indicating the agent generally **improves beam conditions** except when starting near saturated states. Rewards = $score_n - score_{n-1}$. In our case, **score** = ratio * 100 - ε_x * 100 - |μ_x| - |μ_y|



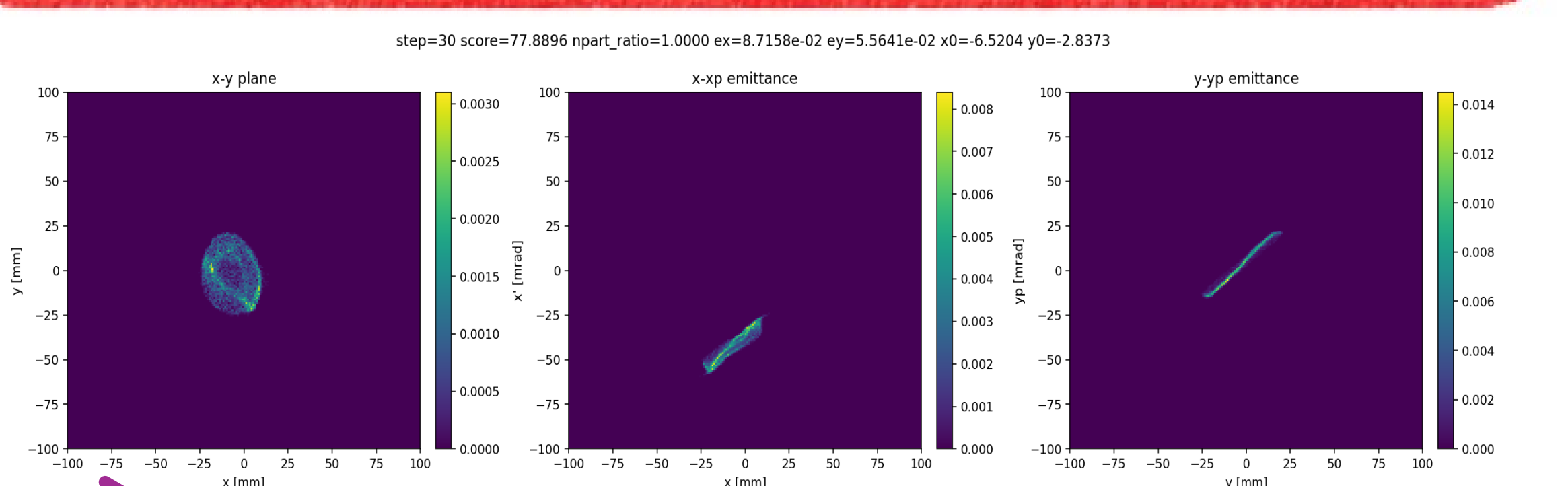
Test agent using Tracewin: Rewards are consistently positive, except in the case of failures corresponding to simulation errors.



Initial state



15 steps: optimal transmission and lower emittance



30 steps: misalignment correction

INFN-LNL: http://www.lnl.infn.it

Contacts: zebele@lnl.infn.it