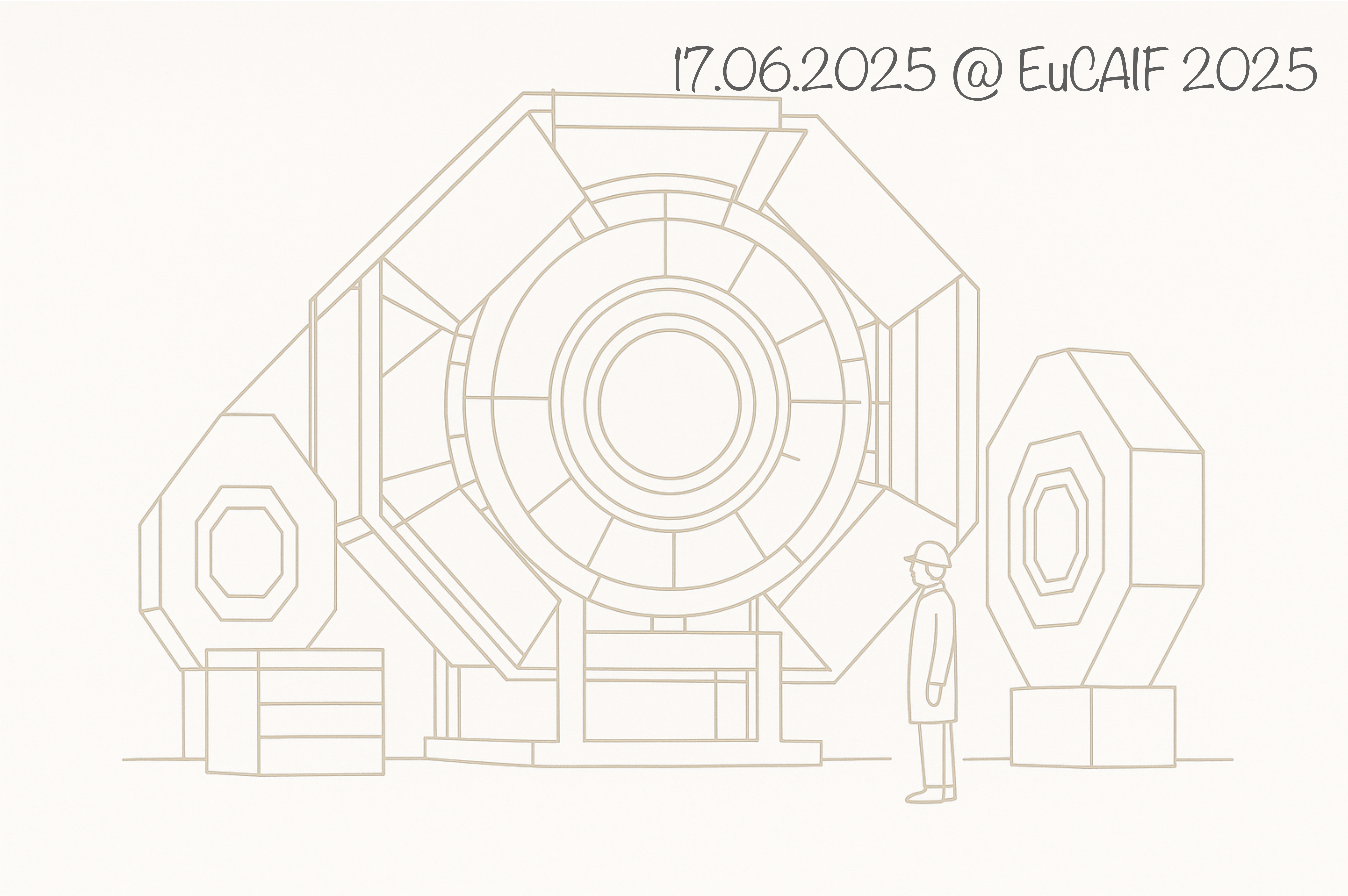
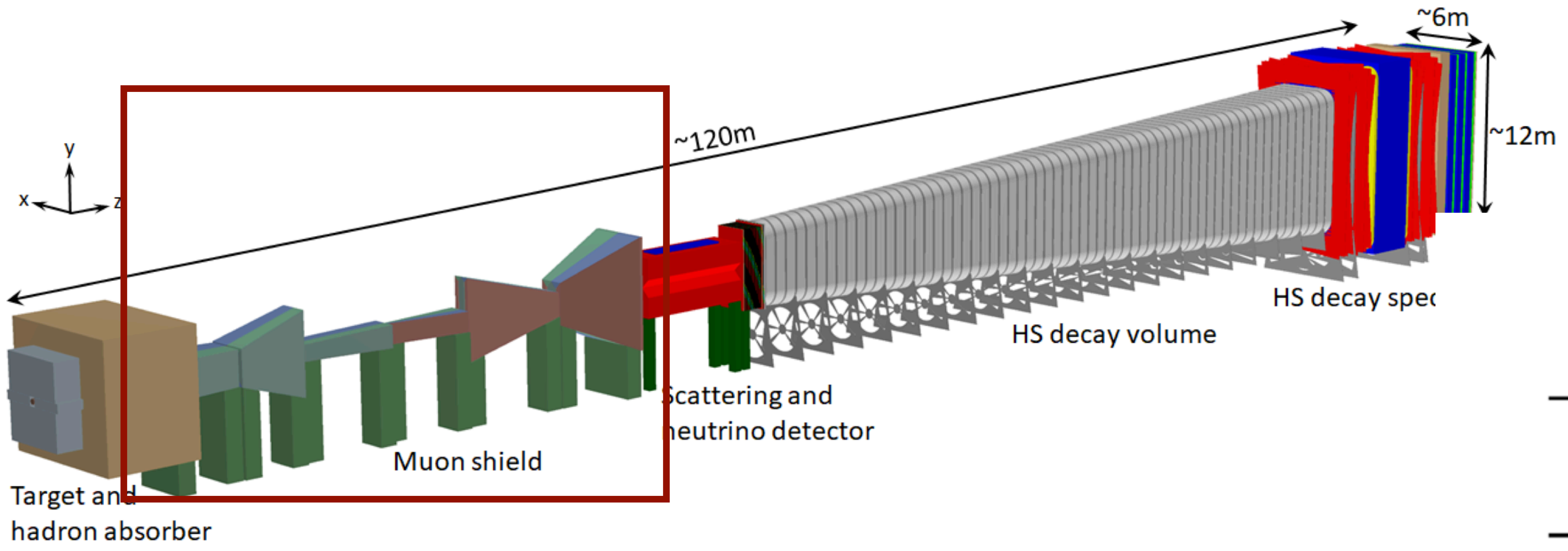


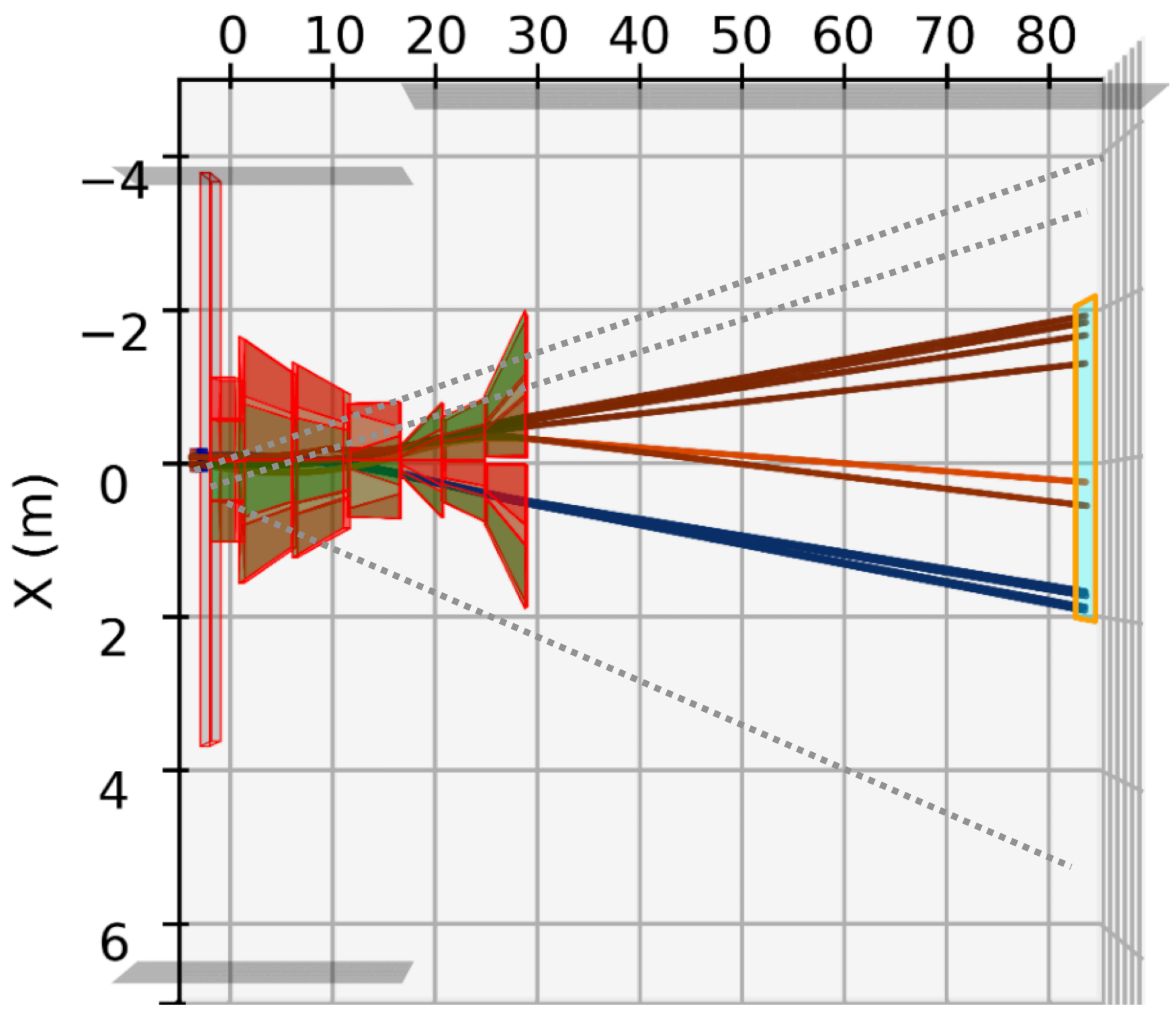
Physics Instrument Design with Reinforcement Learning

Shah Rukh Qasim, Patrick Owen, and Nicola Serra (2024/2025) [arXiv:2412.10237]
Cites work from Luis Felipe Cattelan, Sara Zoccheddu, Melvin Liebsch

Instrument Design is challenging

- The ideas presented here aren't limited to physics instrument





- Here we have the SHiP experiment and the Muon Shield
  - We have been involved in these studies for about a year now
- Lesson: Design of every single component is a lot of work
  - Months of effort, sometimes years
  - I believe we can do much better

For the Muon Shield

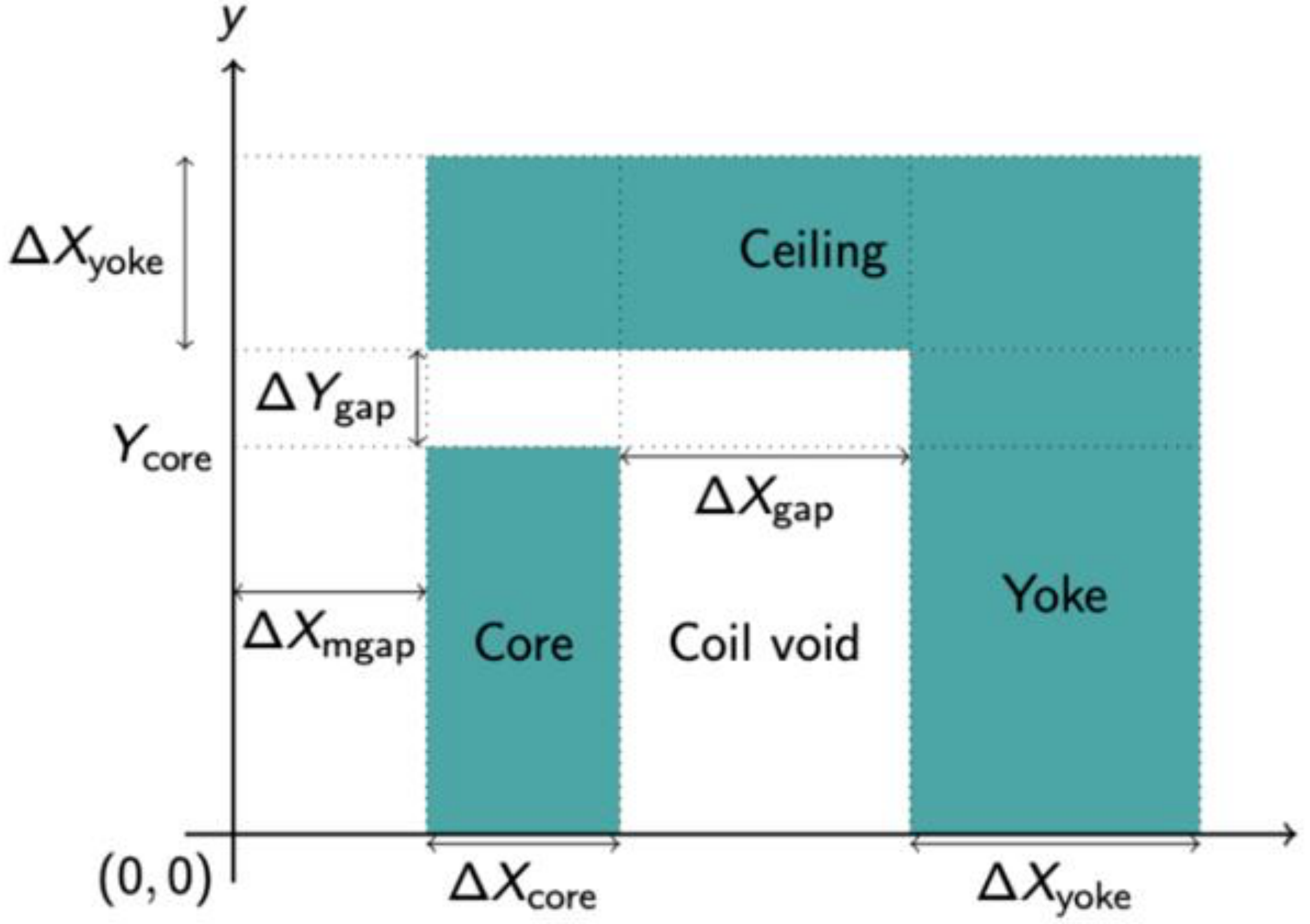
- You got a bunch of parameters
  - It's a strong assumption that you can define an instrument by a set of parameters
  - Let me get back to this in a bit
- And then simulate and test designs

Magnets

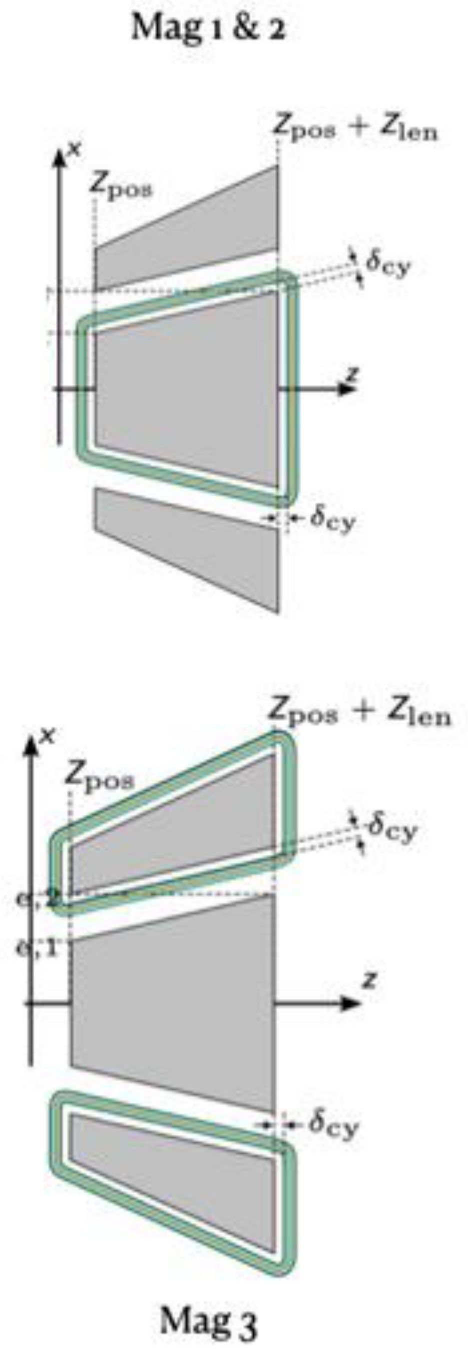
Warm configuration

Total: 60 parameters

- HA + 6 magnets separated by 10cm
- $100 \leq Z_{len} \leq 600$
- $5 \leq \Delta X_{core} \leq 250$
- $5 \leq \Delta Y_{core} \leq 300$
- $0.3 \leq \frac{\Delta X_{yoke}}{\Delta X_{core}} \leq 3$
- $0 \leq X_{mgap} \leq 200$
- $2 \leq \Delta X_{gap} \leq 150$
- $0 < NI < 500\ kA$
- $\Delta Y_{gap} = \begin{cases} 5 & \text{if SC,} \\ 0 & \text{if warm.} \end{cases}$

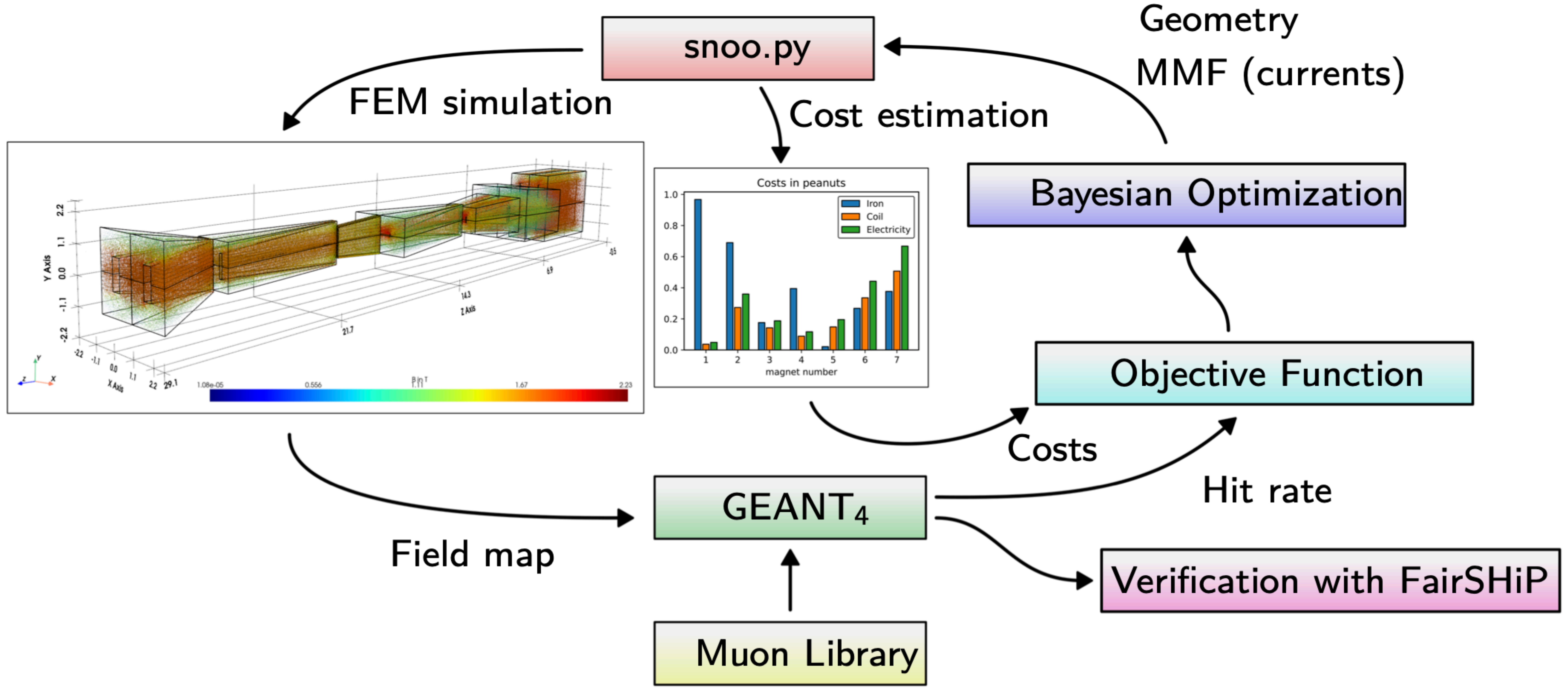


*all spacial dimensions are in cm

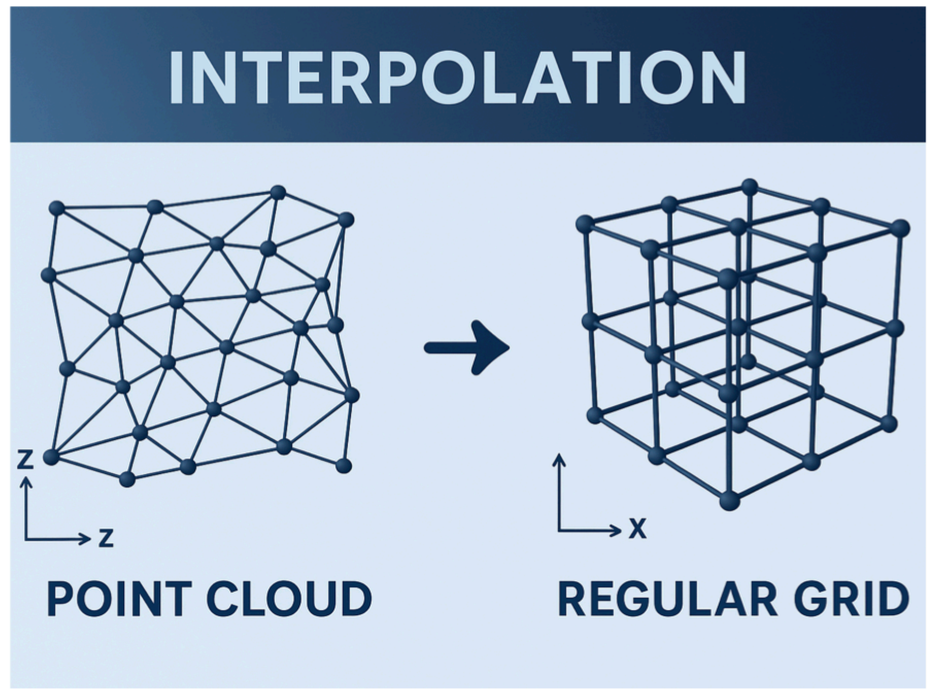


Simulation is hard

- Simulation is expensive and hard and will remain a limitation for a while
- In some cases, the simulation does connect to simulation surrogates (like GANs, VAEs)
    - You are lucky
- In our case, it takes $\mathcal{O}(10)$ seconds using $\mathcal{O}(100)$ CPU cores
    - We developed a complex pipeline where we are using Google Cloud to allocate workloads on remote clusters efficiently
    - Right now we are using 1500 CPU cores continuously

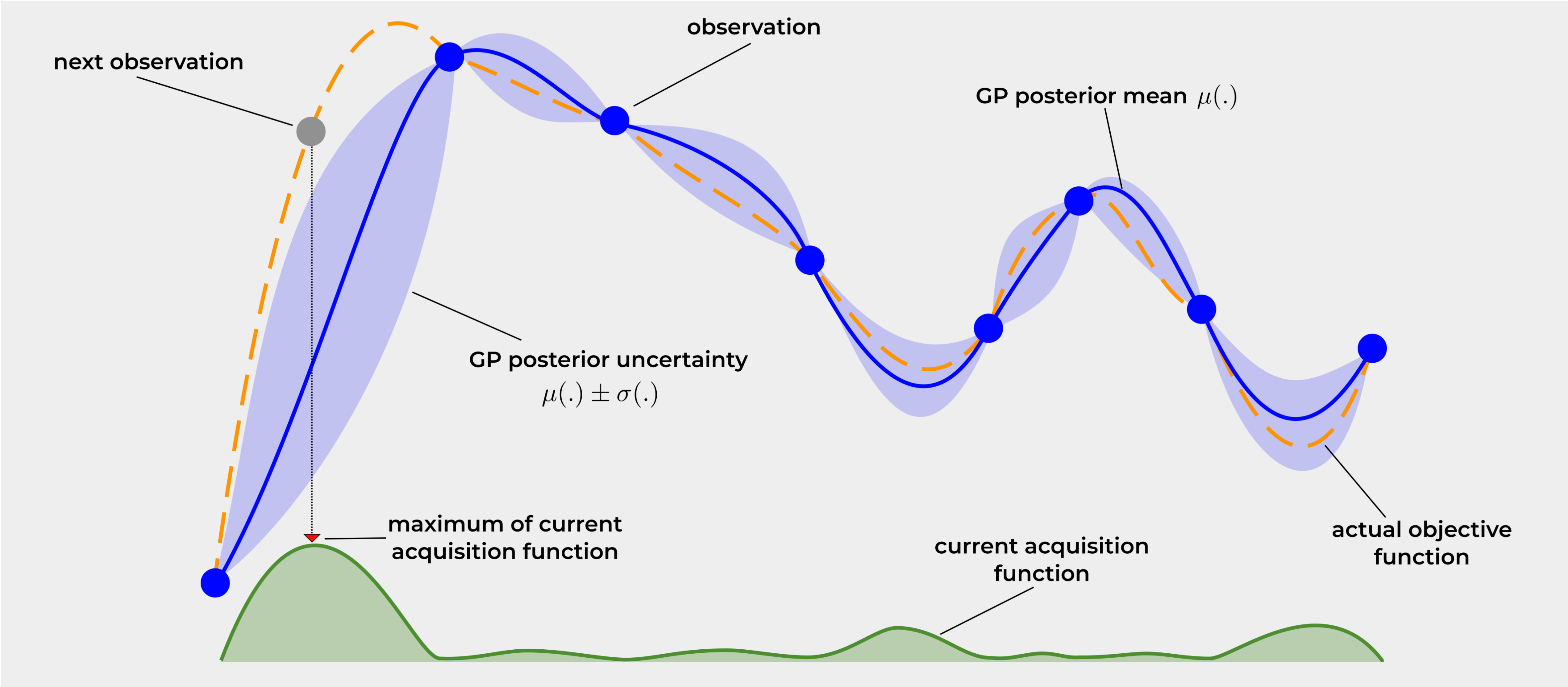


- Developed magnetic field simulation in-house because other software (COSMOL etc) were simply too cumbersome to use
- Used custom CUDA kernels to make that point cloud to regular grid interpolation faster

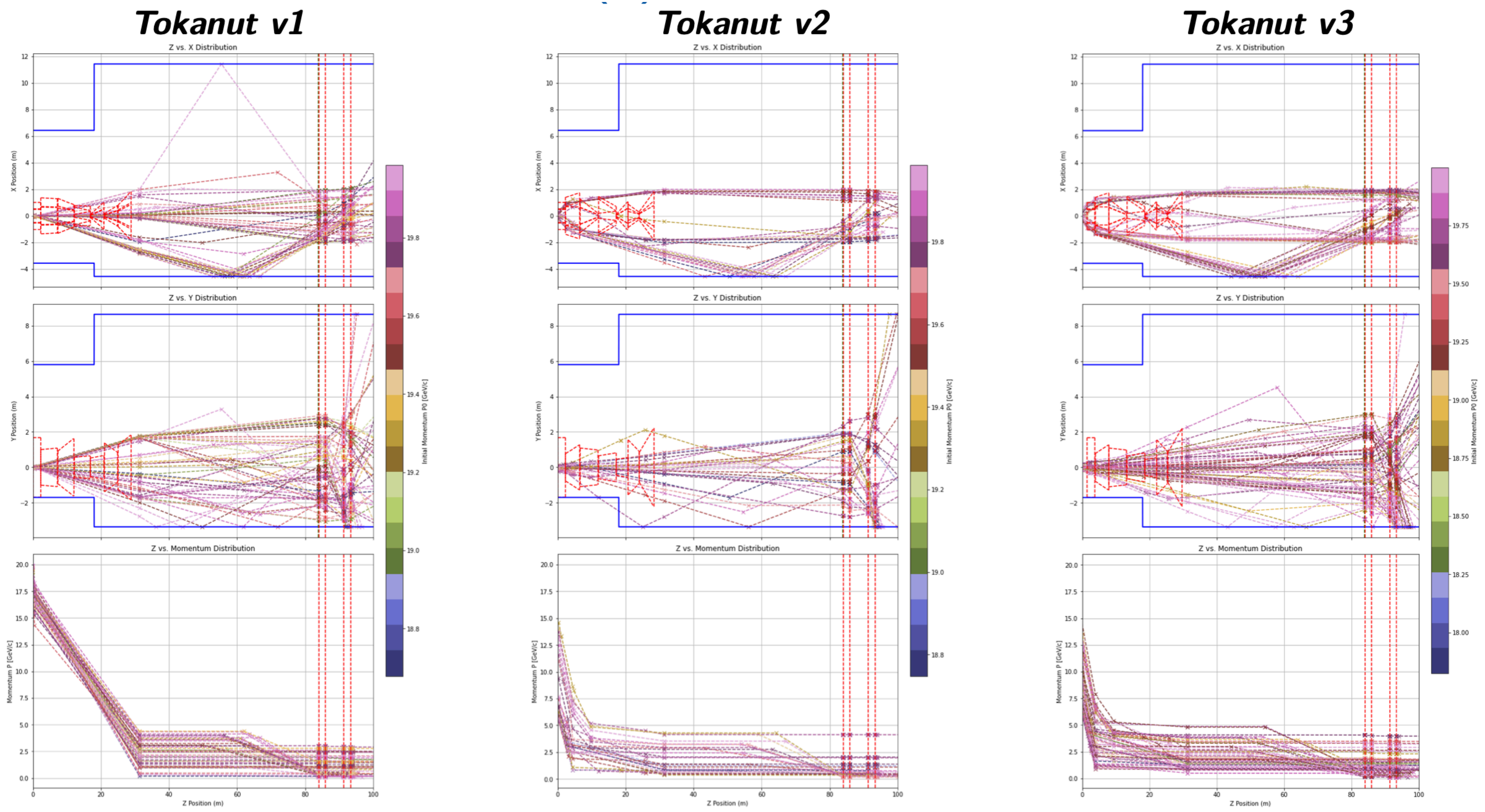


Bayesian Optimization

- You have your blackbox function (simulation)
- Test at N points
  - Choose $(N+1)^{\text{th}}$point based on where you will gain the most information
- Continue till you are done
- The problem is it doesn't scale well because one needs to do matrix inversions

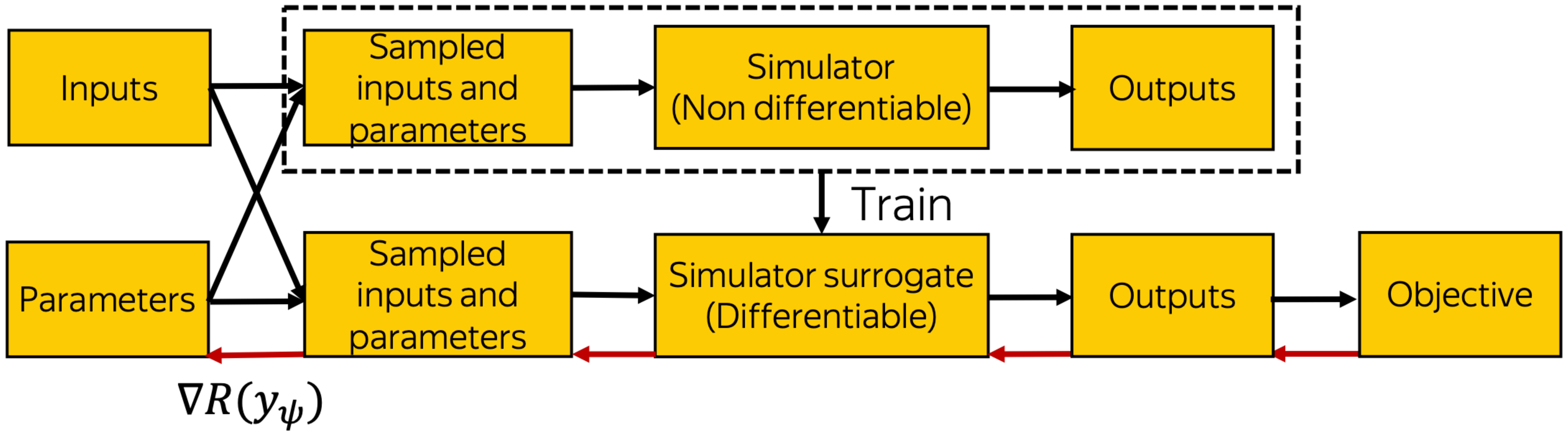



min_loss
tokanut_v2
120
100
80
60
40
0
500
1k
1.5k
Step



- **Learn more at our poster: Design of the SHiP's Muon Shield with Machine Learning (Luís Felipe Cattelan)**

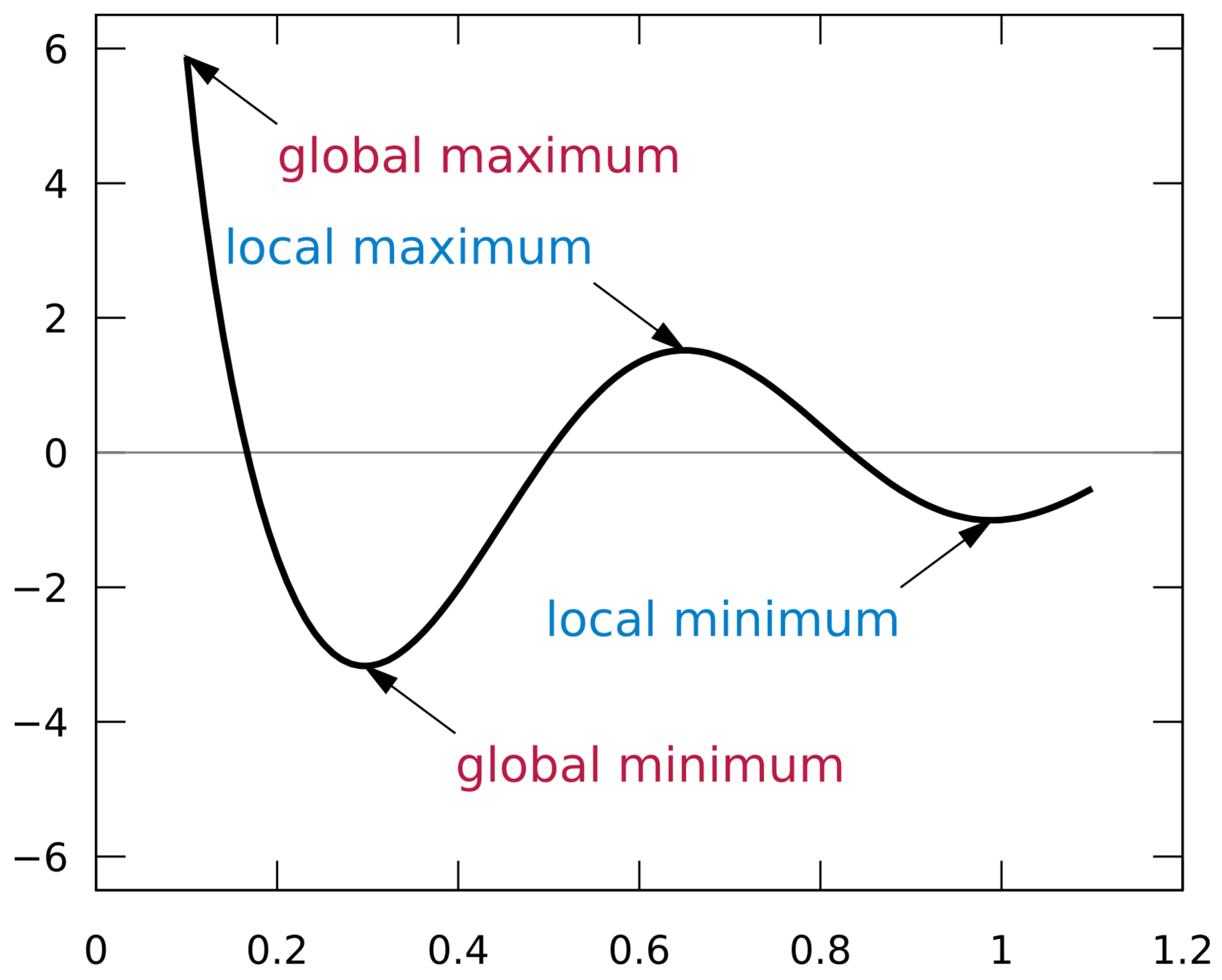
Gradient Based Optimization



- Originally proposed in arXiv:2002.04632
- Was in fact demonstrated on the Muon Shield
    - We think for the Muon Shield Optimization as defined, Bayesian Opt. Works better
    - Differentiable method is of course more scalable
        - **Although this is not necessarily a good thing**
- But now is the principal approach being studied for physics experiment optimization https://mode-collaboration.github.io/#about
- There are efforts to make everything differentiable, including Geant4

We take a different view

- Local optimizations
  - It is dangerous to compare this to Neural Network optimization with gradients
  - NNs have A LOT of randomly defined: locally unstable
  - No such guarantee with design parameters which map to a physical quantities even in a very non-linear fashion



We take a different view

- In general, it is hard to define an instrument as a set of parameters
  - The number of parameters will vary, depending on how many sub-components are placed within a design
  - Let's say each magnet has 10 parameters, a design with 5 magnets will have 50 and a design with 6, 60.
  - In some cases, the number of combinations will scale exponentially
- Can also make discrete choices
  - Choosing the material in a detector for example
- And finally, reward function being differentiable or not

Solution: Reinforcement Learning

- Take actions
    - These actions are contextually dependent
    - We end up in very different states as we take these actions
- And then idea is to “reinforce” the contextually dependent actions that led to the best outcome
    - Temporal credit assignment (TD learning / Monte Carlo etc)
    - Increase the probability of these actions being taken (learning the policy)

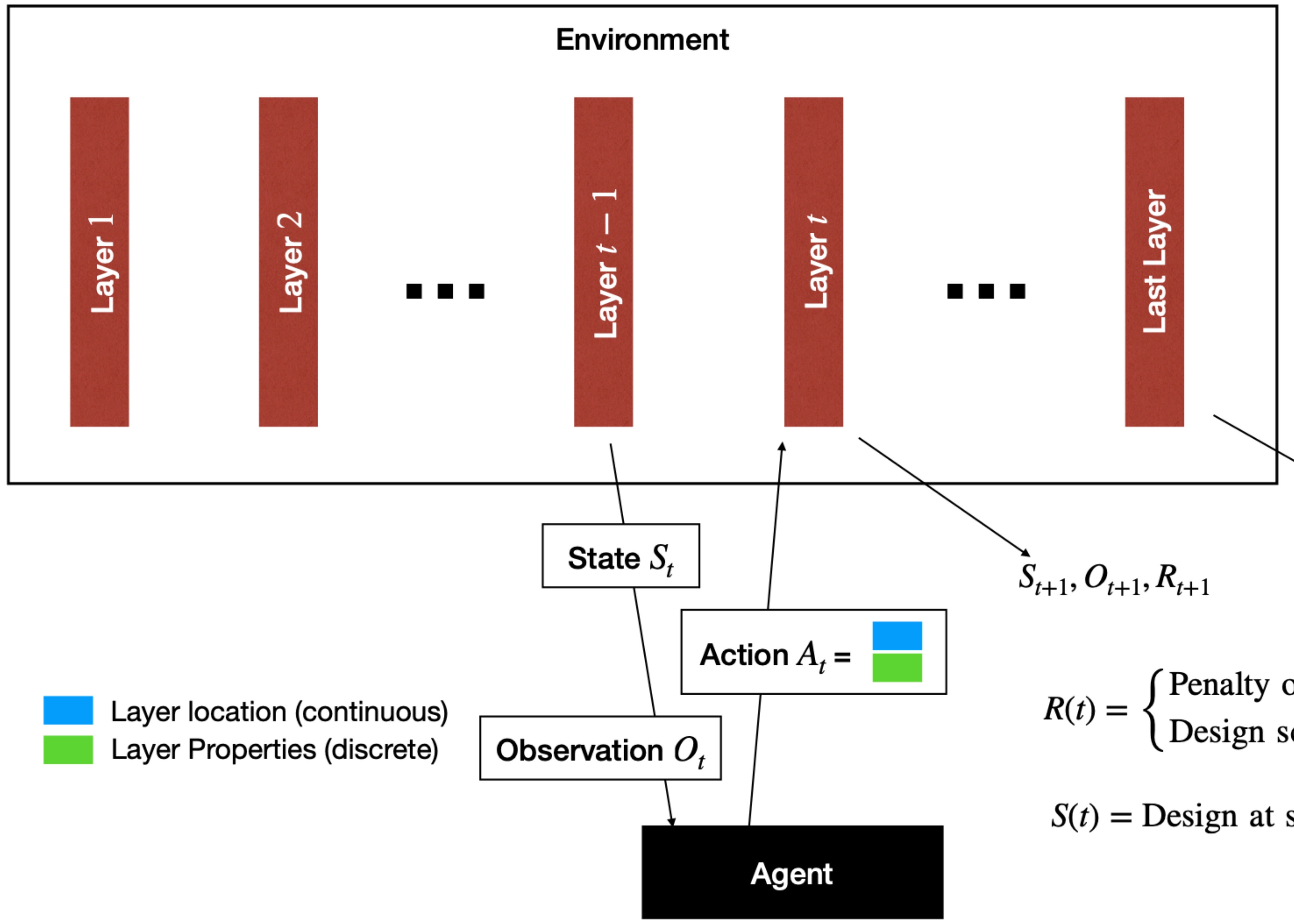
RL allows exploration

- And balance it with exploration

$$\pi(a|s) = \begin{cases} 1 - \varepsilon + \frac{\varepsilon}{|\mathcal{A}|} & \text{if } a = \arg\max_{a'} Q(s, a') \\ \frac{\varepsilon}{|\mathcal{A}|} & \text{otherwise} \end{cases}$$

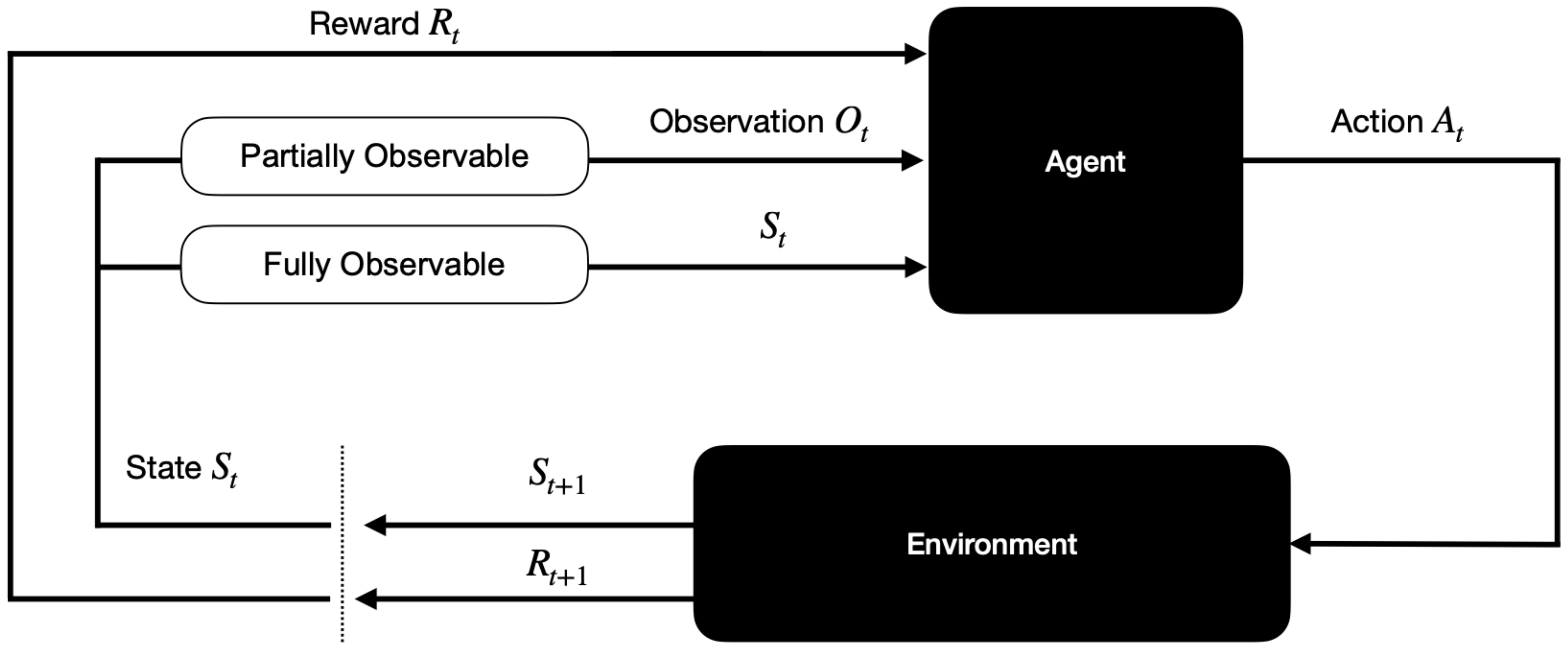
- These random actions brings you to states that are very different from each other
- One needs to be able to reach varying rewards to allow exploration and optimization and cold start is a problem
    - If you are training an agent to play chess while it learns only from the a super good bot. It will always get a reward of 0 => no training possible
- Finding globally optimal solution is not a magic in RL
    - Framework allows it
    - A lot of work has been performed

Reinforcement Learning for Design

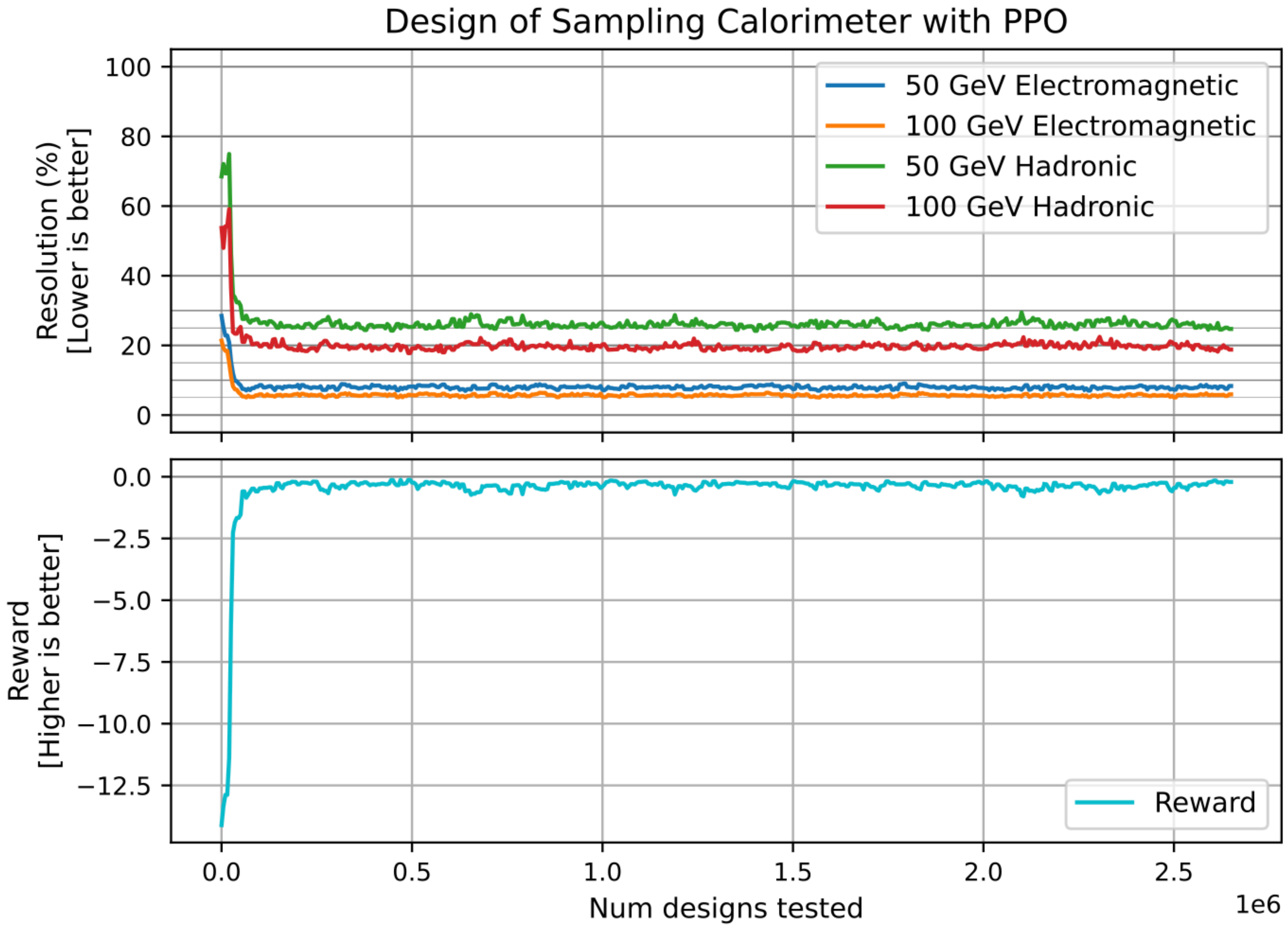


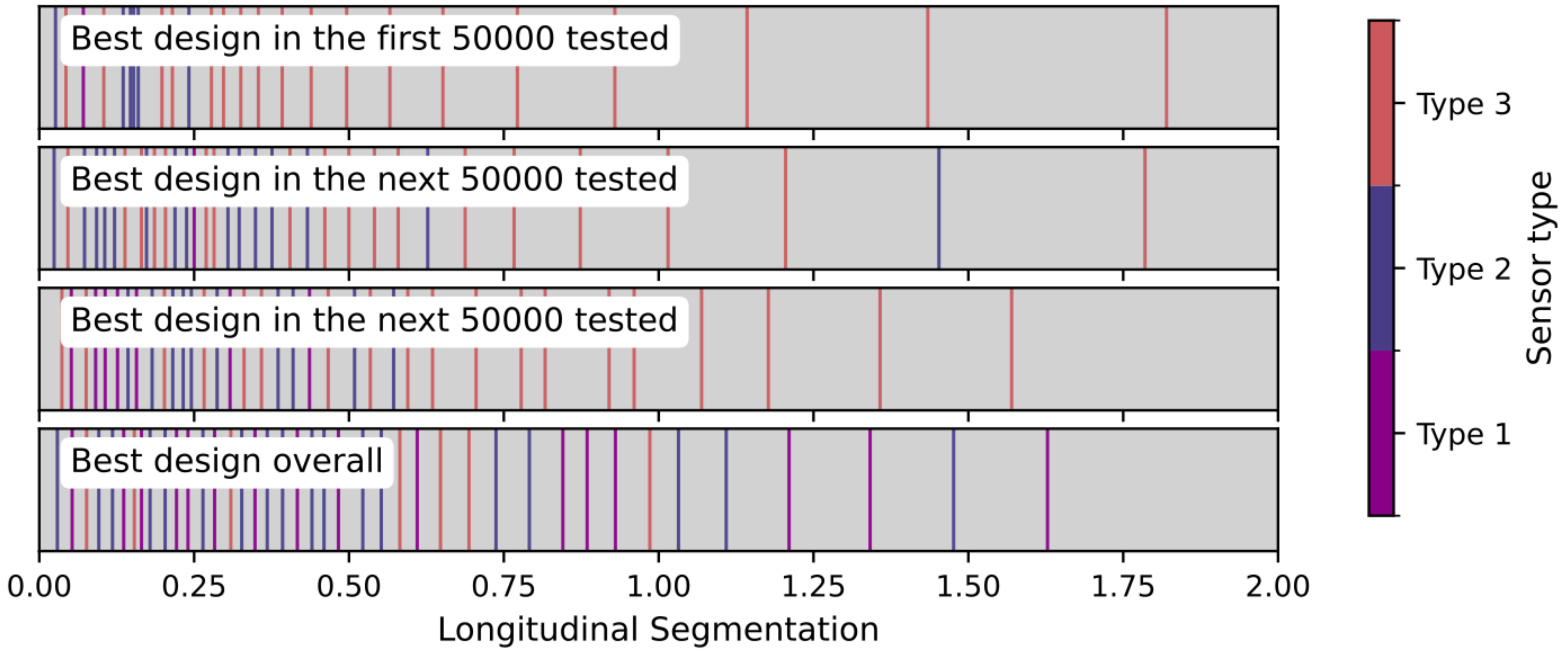
$$R(t) = \begin{cases} \text{Penalty or } 0.0 & \text{if } t \neq T, \\ \text{Design score} & \text{otherwise}. \end{cases}$$

$$S(t) = \text{Design at step } t$$



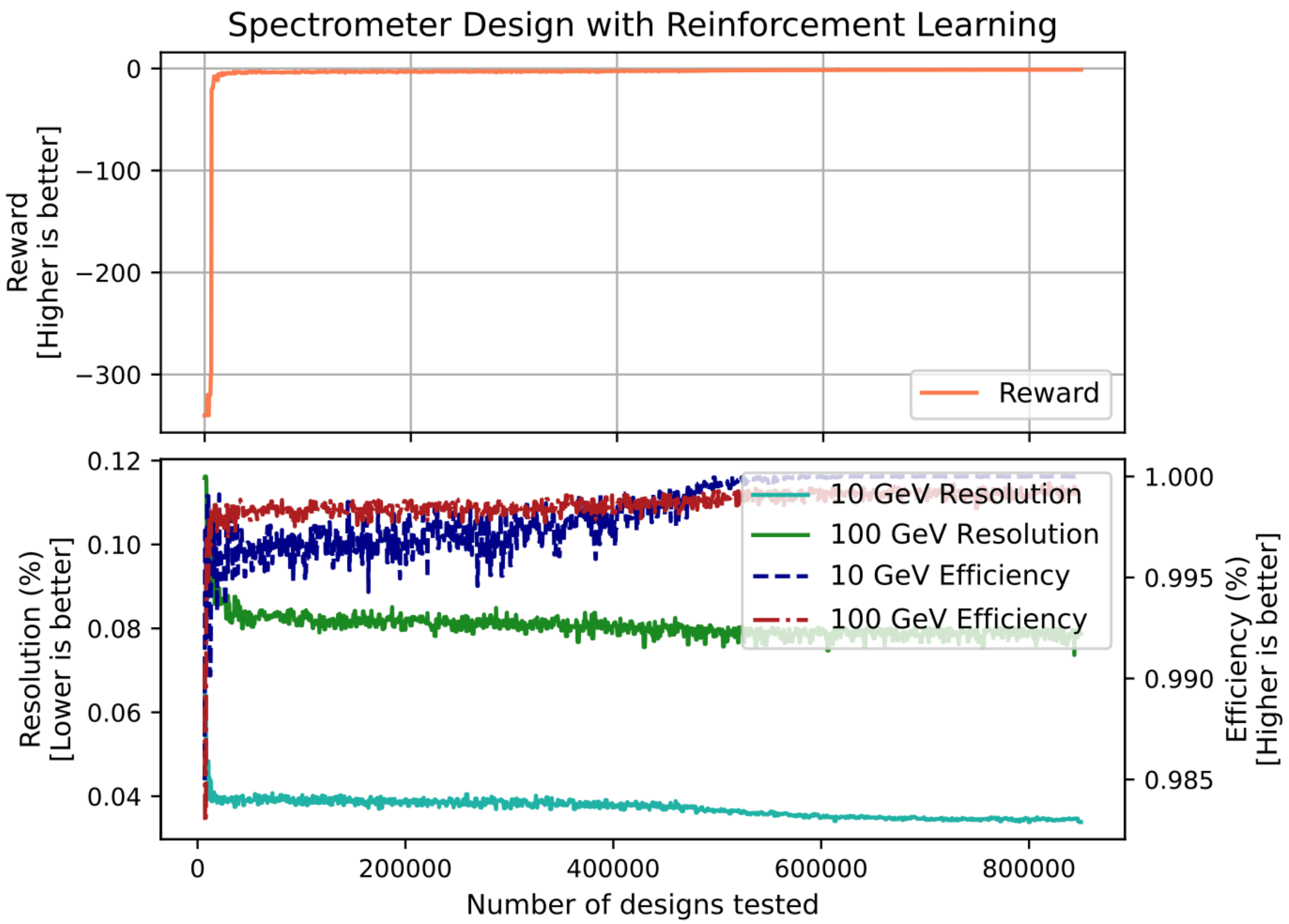
We did this with a toy calorimeter

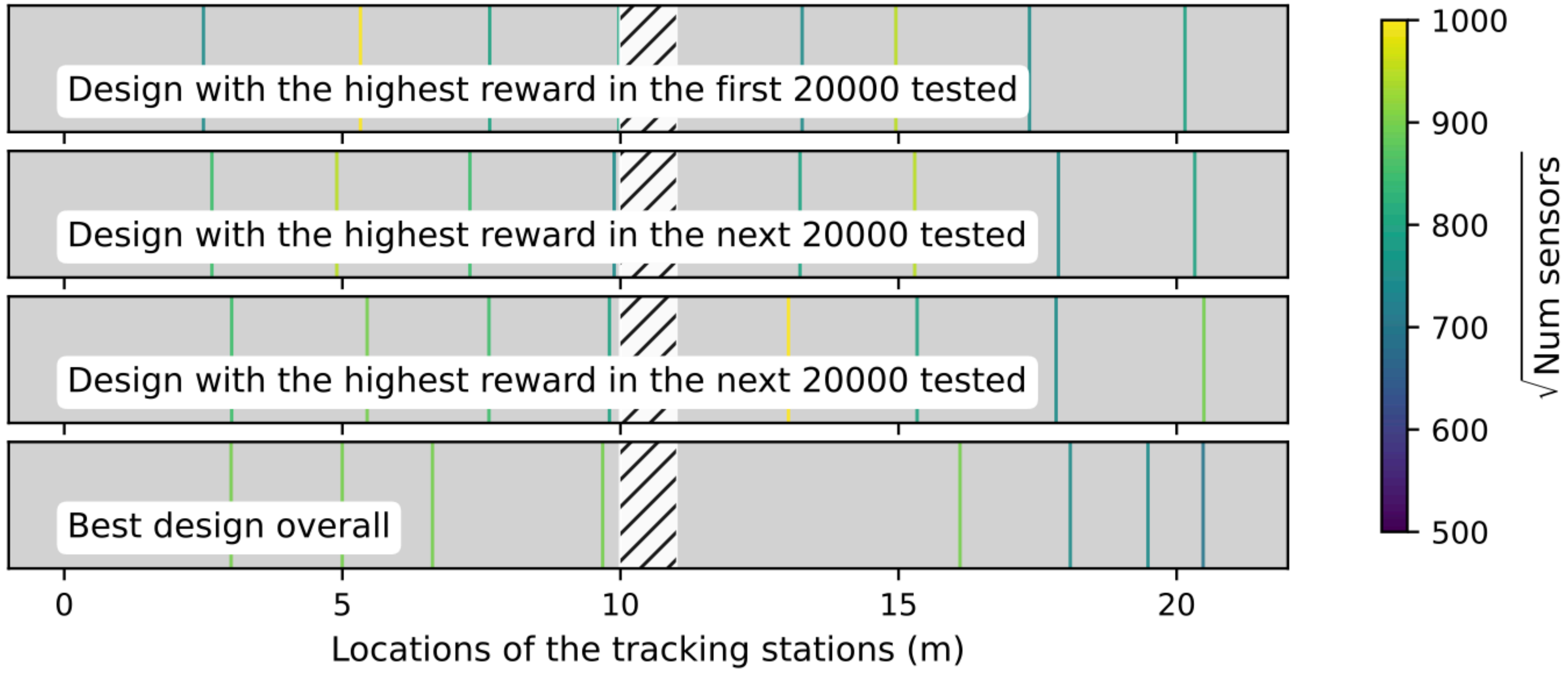




$$S \cdot 10 = -\max(0, \Sigma_{\mathrm{em50}} - 8) - \max(0, \Sigma_{\mathrm{em100}} - 5) - \max(0, \Sigma_{\mathrm{had50}} - 25) - \max(0, \Sigma_{\mathrm{had100}} - 18)\ , \quad (2)$$

And with a spectrometer design





$$S_{10} = -3 \cdot \left(95 - \min(\mathrm{eff}_{10}, 95) + \max(\mathrm{res}_{10}, 3) - 3\right) \ ,$$

$$S_{100} = -3 \cdot \left(95 - \min(\mathrm{eff}_{100}, 95) + \max(\mathrm{res}_{100}, 8) - 8\right) \ ,$$

$$S = S_{10} + S_{100} \ .$$

Results

- Can work better than baseline designs
  - Don’t read too much into the numbers but just to illustrate a point

Spectro

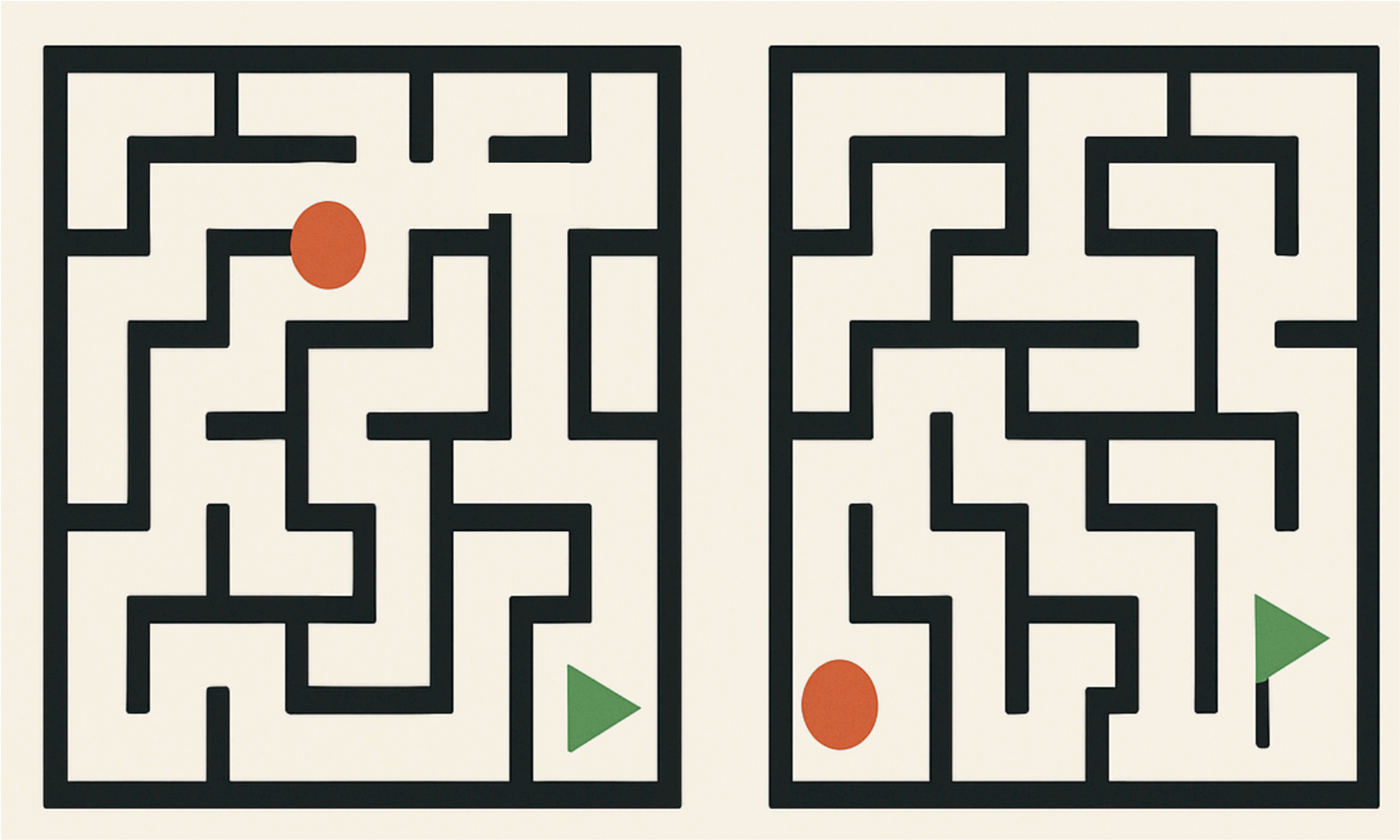
	10 GeV Eff	10 GeV Res	100 GeV Eff	100 GeV Res
Baseline design 1	89.0695 ± 0.1275	6.48 ± 0.03	98.0982 ± 0.0558	13.83 ± 0.06
Baseline design 2	93.7342 ± 0.0989	5.97 ± 0.03	99.1317 ± 0.0379	13.15 ± 0.05
RL design	100.0000 ± 0.0000	3.74 ± 0.02	99.9417 ± 0.0099	7.87 ± 0.03

Calo

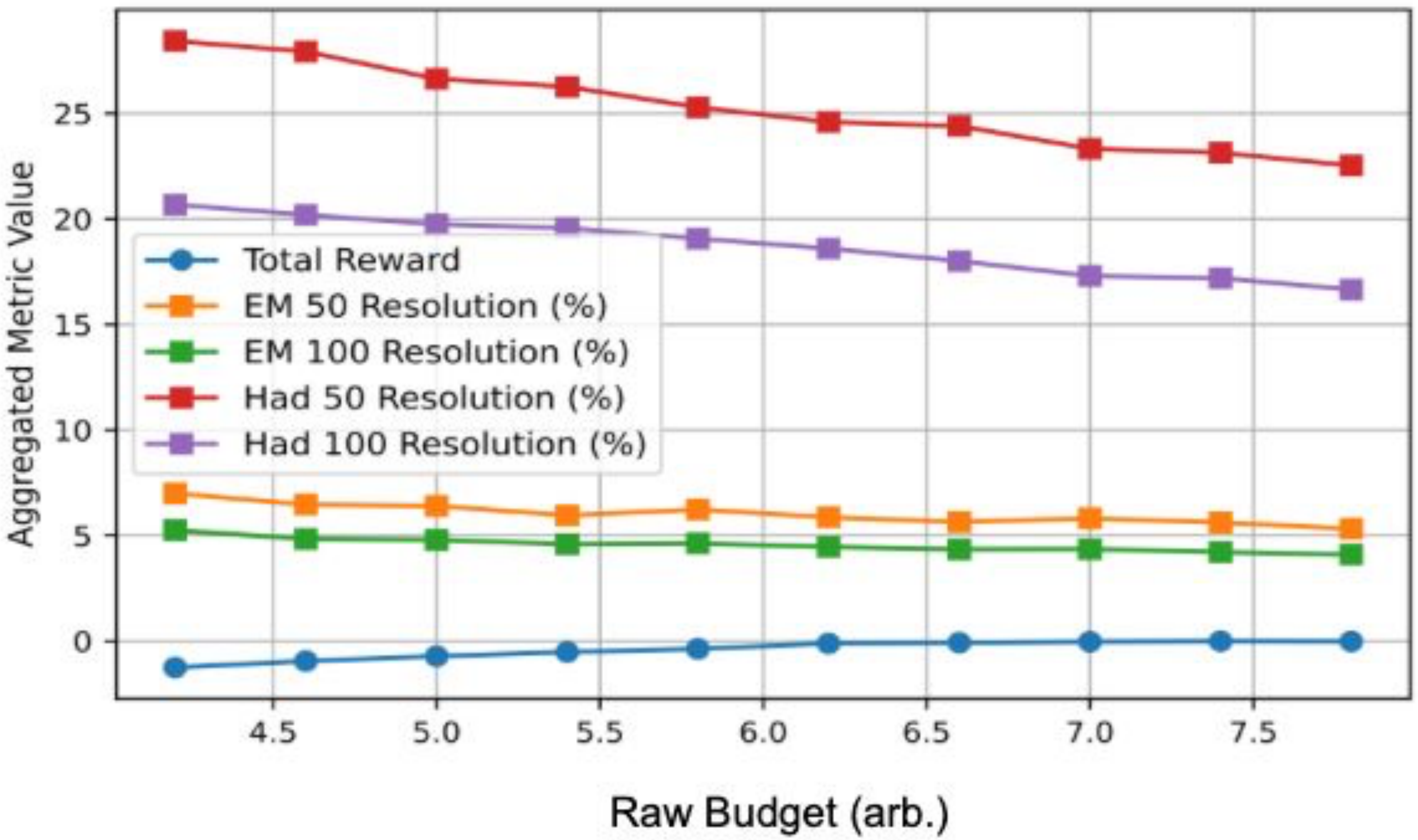
	50 GeV EM	100 GeV EM	50 GeV Had	100 GeV Had
Baseline design	8.24 ± 0.16	5.94 ± 0.12	34.13 ± 0.68	24.48 ± 0.49
RL design	8.15 ± 0.16	5.83 ± 0.12	25.27 ± 0.51	17.79 ± 0.36

Conditional

- Learn conditionally!
- Whenever the agent is designing an instrument at the start of the episode, it knows what the budget
    - And we sample the budgets randomly
        - The agent will learn to design the instrument according to different budgets



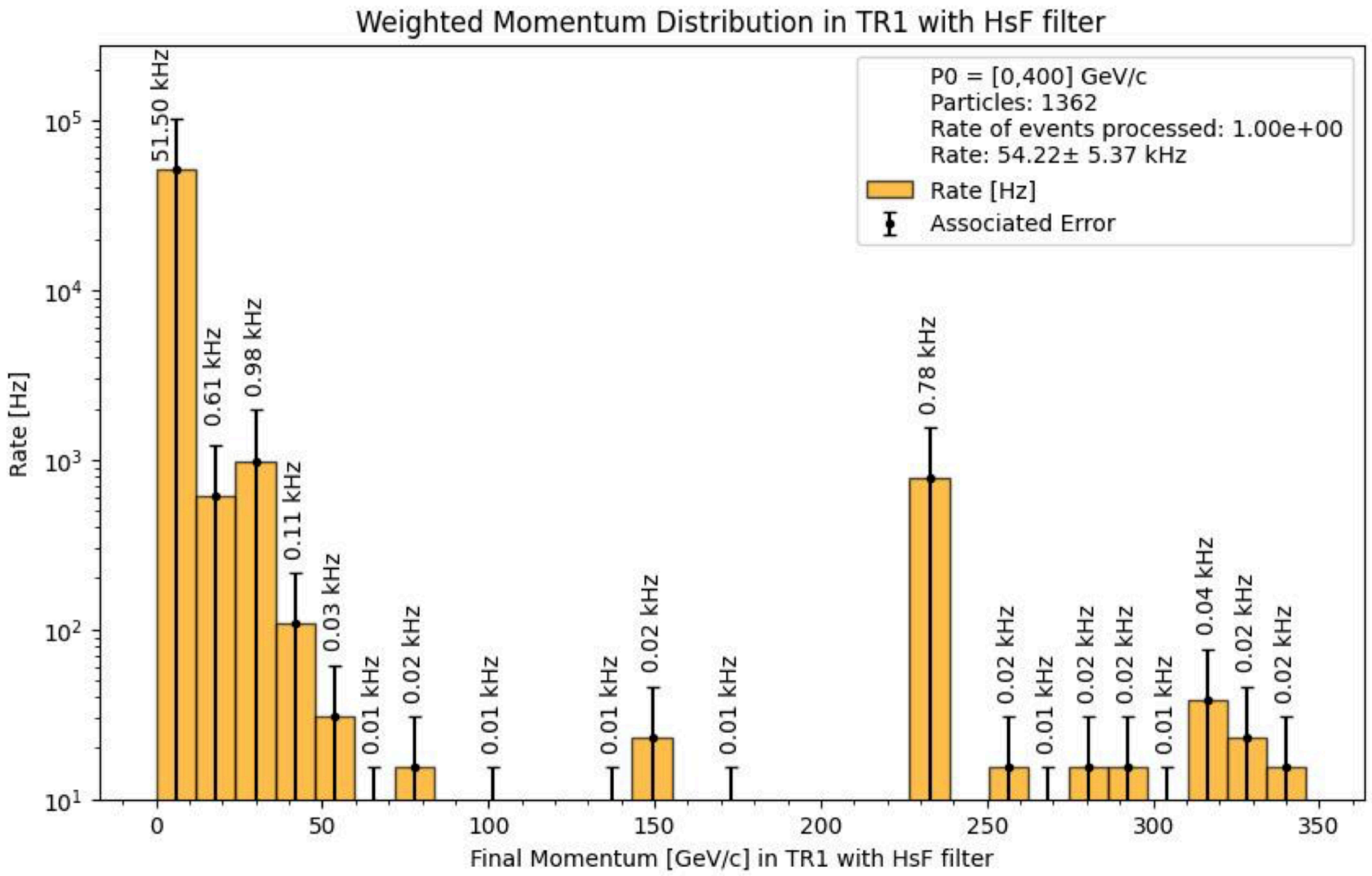
Some preliminary results



More concretely

- This will be useful for physics experiments
- You have many different teams designing different subcomponents
- The best design depends on how the other components are designed
  - We have meetings where we discuss how the other components are going
  - We propose: train a robust agent which spits out different designs within the expected vicinity
- RL will allow design of different instrument separately before putting it all together
- This will also allow us to re-use R&D from other experiments further
  - True even if you are not expecting the agent do discover a magic solution

SuperNut v2

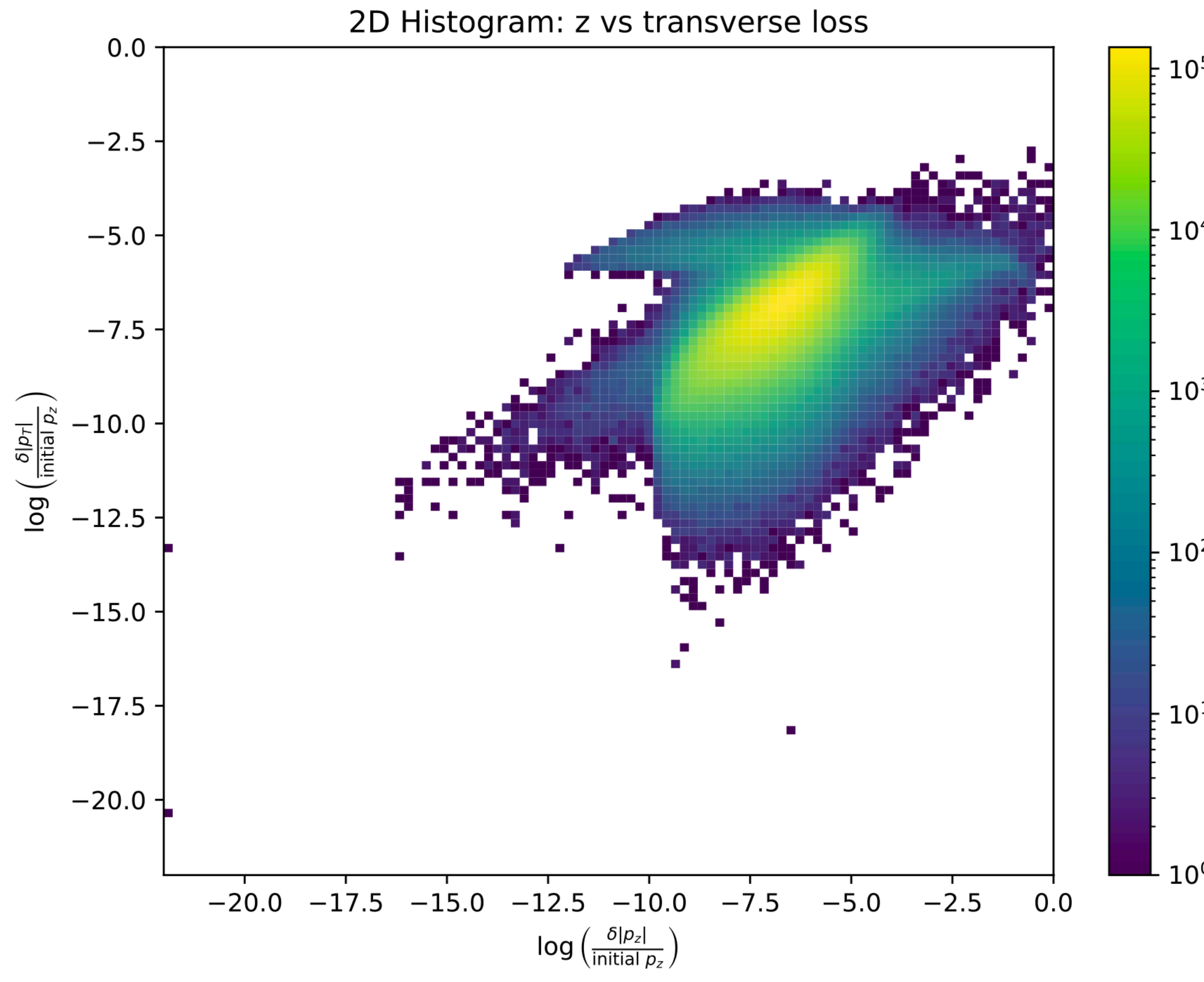


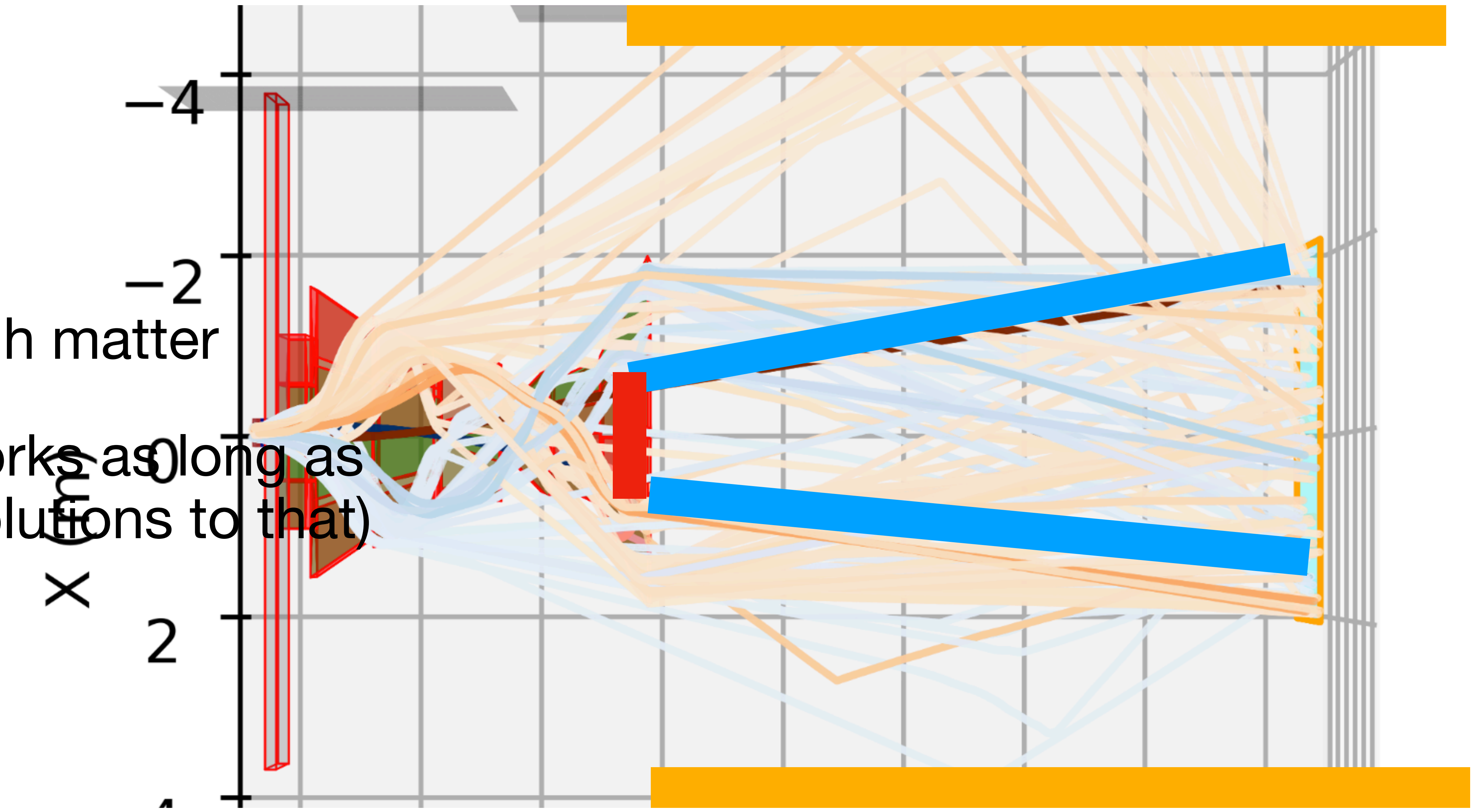
Rate bouncing $P_0 \in [0, 400]$ ***GeV/c 3 kHz***

Rate = 54 kHz

SHiP Experiment Further studies

- Easy-to-do tasks
  - If we re-write the simulation: slow rollouts/querying remains a limitation at the moment (but will improve in the future)
    - Placement of sensor layers for monitoring system of the Muon Shield
    - Design of the VETO system
- Harder
  - Faster Muon Simulation
  - We managed to collect step data from Geant4
  - And use alias sampling CUDA code to transport Muons through matter
    - Includes inelastic scattering and multiple scattering and works as long as particles don't decay into other particles (there are other solutions to that)
  - Incredibly fast, orders of magnitude faster than Geant4
- Contextual / Conditional RL will find workarounds





Challenges and opportunities

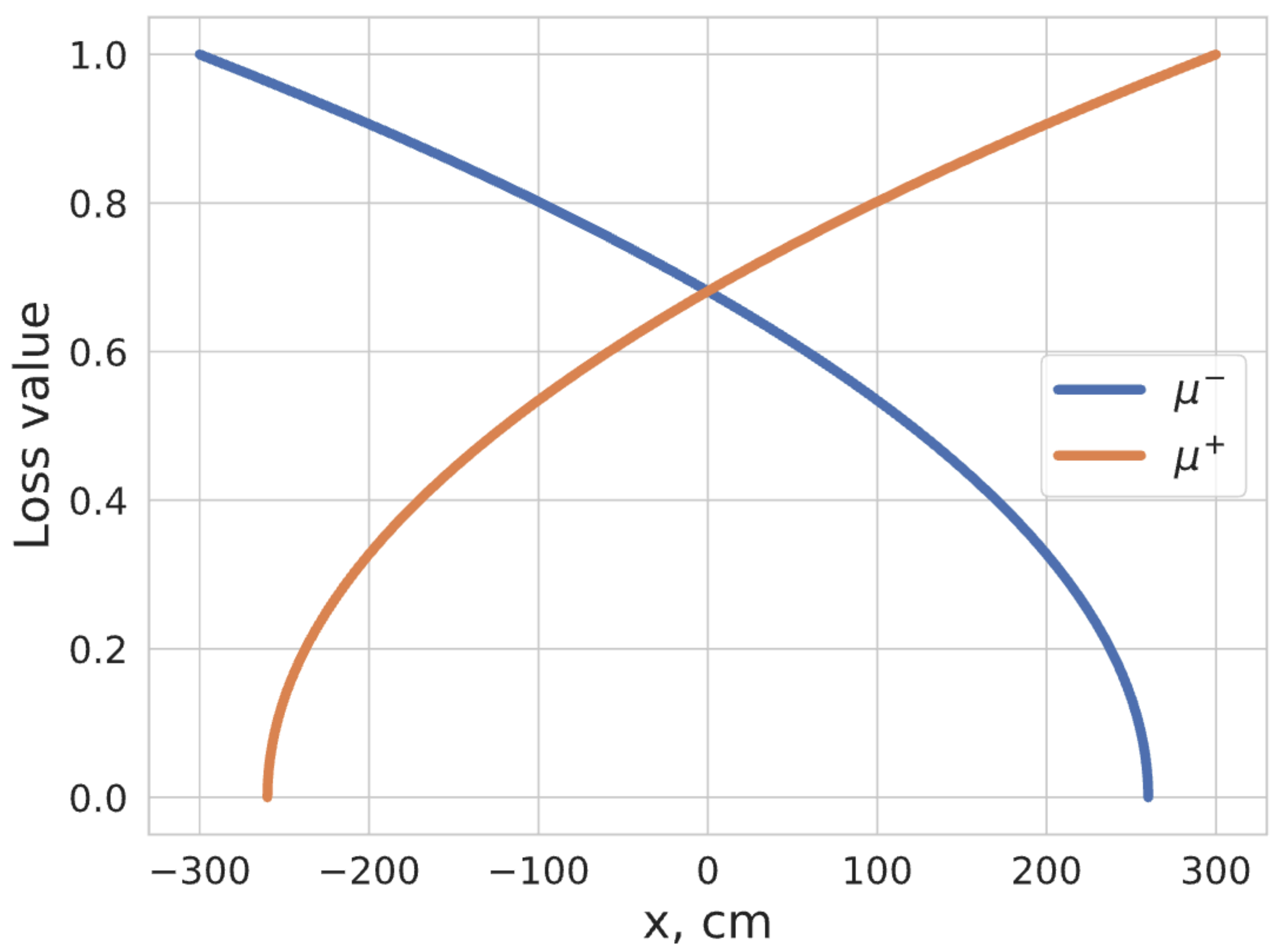
- Biggest limitation:
  - Querying is slow
  - Future: surrogate simulation
- If you are a computing expert, you can find ways around and already start using these methods
- Researched can be performed along many lines:
  - For example, for the Muon Shield, find a strategy for intelligent sampling. A bad design can be discarded right from the start and one doesn't need to test it on a billion samples (which is computationally expensive)
  - Better exploration algorithms specifically designed for "Design"

Rest

Muons Rate

- Sensitive plane:
  - $\Delta x = 4.2m$
  - $\Delta y = 6m$
  - $z = 82m$ *

$$\mathcal{L}_M = \sum_{i=1}^{N} \sqrt{\left(1 + \frac{Q \cdot x(\psi) - \Delta x/2}{\Delta x}\right)}$$



*Origin is the beginning of Hadron Absorber

Cost function

- Total cost:
- $$\mathscr{C} = \frac{\sum_i C_i - \mathscr{C}_0}{\mathscr{C}_0}$$

The cost C_i for magnet i is

$$C_i = \underbrace{W_i \, \kappa_{\mathrm{yoke}}}_{\text{Yoke}} + \underbrace{M_i \, \kappa_{\mathrm{coil}}}_{\text{Coil}} + \underbrace{Q_i \, T \, \kappa_{\mathrm{EDF}}}_{\text{Operation}}$$

Variable	Meaning	Depends on
W_i	Iron yoke mass	geometry, density
M_i	Coil mass	geometry, density
Q_i	Power consumption	geometry, MMF, J_{tar}, conductivity
T	Operation time	always 72'000 h

Table: The variables for the cost estimation.

We will represent costs in the toy currency Peanut (∞)

Loss function

$$\mathscr{L} = \left(1 + e^{10\mathscr{C}}\right)\left(1 + \sum_{i=1}^{N}\sqrt{\left(1 + \frac{Q \cdot x(\psi) - \Delta x}{2\Delta x}\right)}\right)$$

Loss with constraints

- $\mathscr{L} = \left(1 + e^{10\mathscr{C}}\right)\left(1 + \sum_{i=1}^{N} \sqrt{\left(1 + \frac{Q \cdot x(\psi) - \Delta x}{2\Delta x}\right)}\right)$

- $\mathscr{L}(\Psi) = \mathscr{L} + \lambda \left[\sum_{i}^{6} \sum_{j}^{2} \max(0, C_{i,j}(\Psi))^2 + \max(0, L_{max} - L)^2 \right]$

- $C(x) = (\Delta X - x_{wall}) + (\Delta Y - y_{floor})$
- $x_{wall}(z) = \begin{cases} 3.54m, & \text{if } z \in \text{TCC8} \\ 4.54m, & \text{if} z \in ECN3 \end{cases}$
- $y_{floor}(z) = \begin{cases} 1.68m, & \text{if } z \in TCC8 \\ 3.34m, & \text{if} z \in ECN3 \end{cases}$
- $\boxed{L_{max} = 29.65m}$

$$\Delta X = \Delta X_{mgap} + \Delta X_{core} + \Delta X_{gap} + \Delta X_{yoke}$$

$$\Delta Y = \Delta Y_{core} + \Delta Y_{gap} + \Delta X_{yoke}$$

*2cm of gap between magnet and cavern