Generative Surrogate Models for Differentiable Optimization of a Parallel-Plate Avalanche Counter with Optical Readout

María Pereira Martínez, Pietro Vischia, Xabier Cid Vidal

XVII CPAN Days, Valencia

November 20th 2025

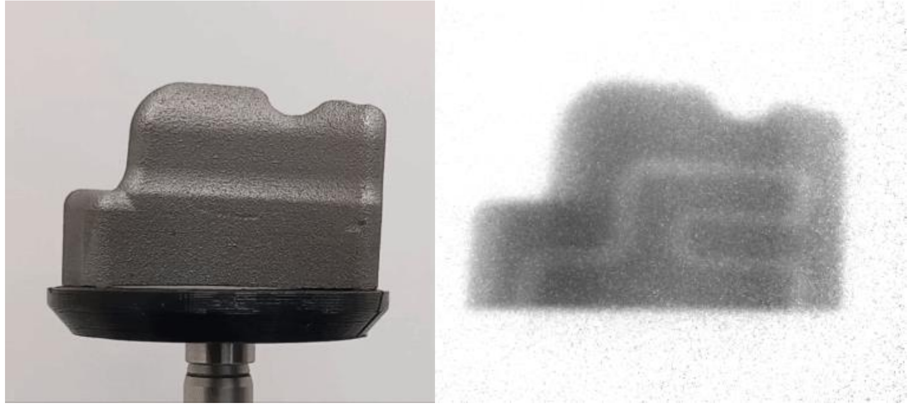
I. Introduction

Neutron tomography

Tomography by **emission and detection of neutrons for non-destructive tests**.

High penetration, effective for dense materials like metals and alloys.

Metal industry, additive manufacturing, border security…



Frances Yassid Ayyad Limonge

Pablo Cabanelas Eiras

Images from Neutron Insights website

What do we want to do?

Optimize the neutron tomography system but… **where do we start?**

Optical Parallel-Plate Avalanche Counter

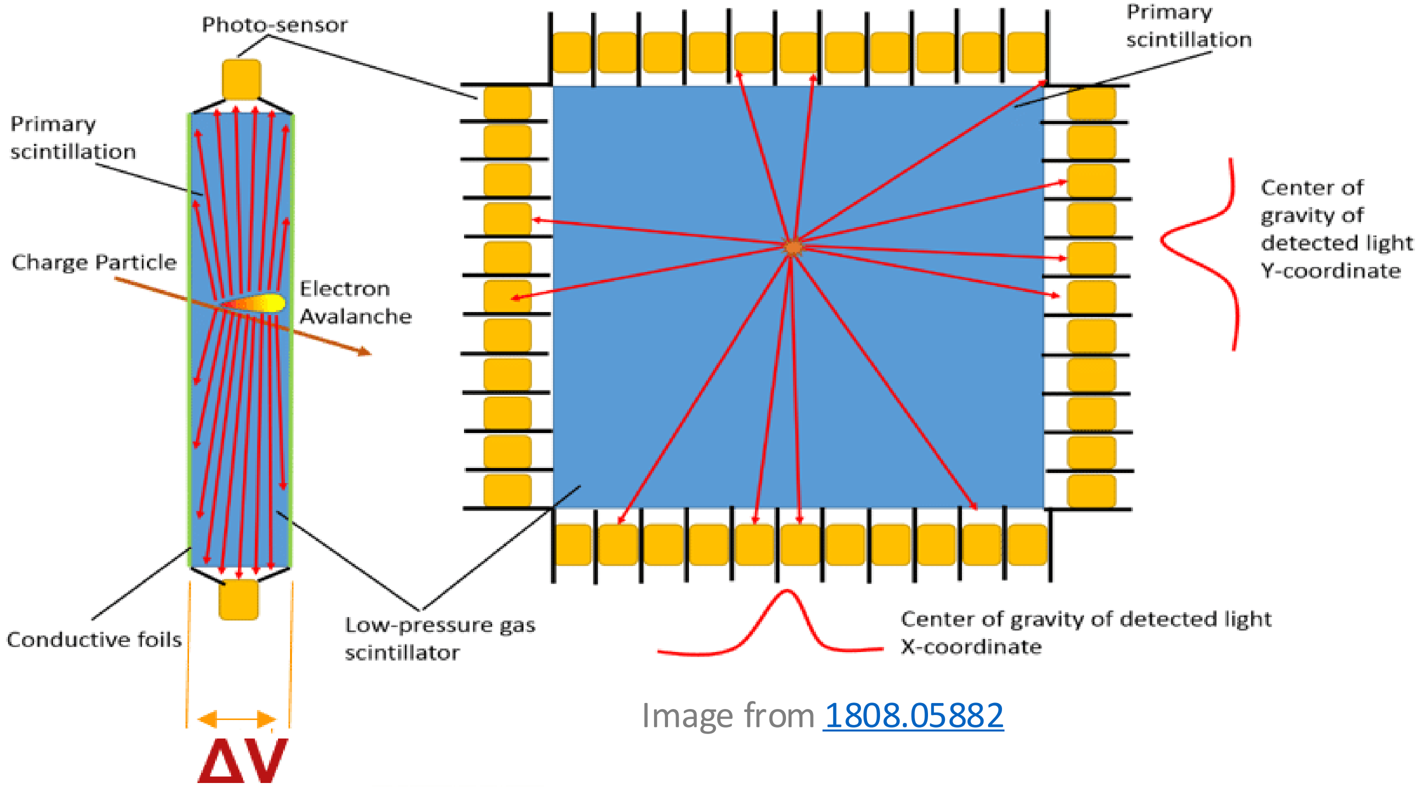


Image from [1808.05882](1808.05882)

- Parallel-plates filled with a **high electroluminiscense yield gas** (CF4).
- Charged particles crossing active volume ionize medium and produce an avalanche.
- Electroluminiscense light detected by 4 arrays of small, collimated silicon photomultipliers (SiPMs).

Optical Parallel Plate Avalanche Counter

Geant4 model of a 10×10 cm^2 O-PPAC, 33 SiPMs per array

Example of an event triggered by an impining alpha particle:

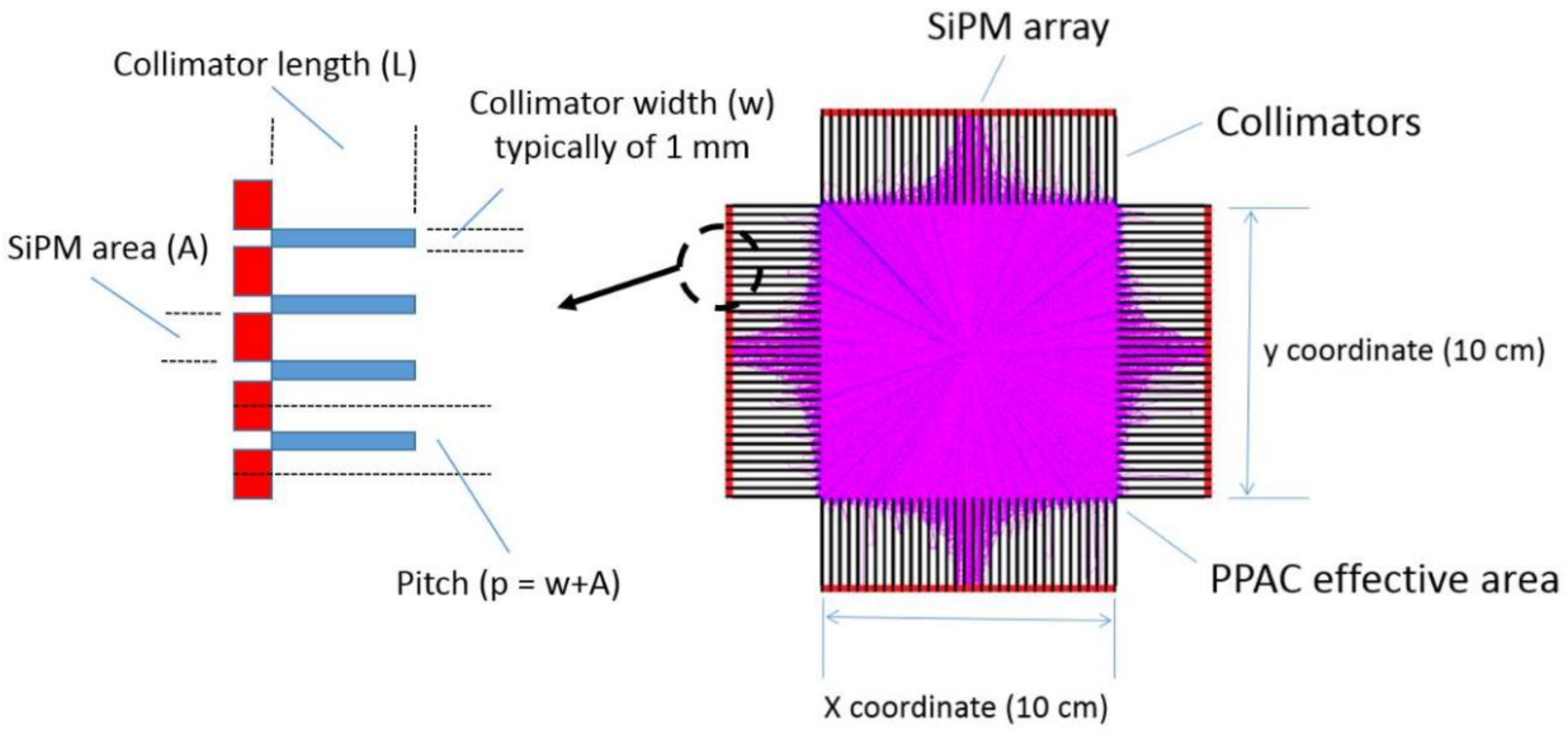
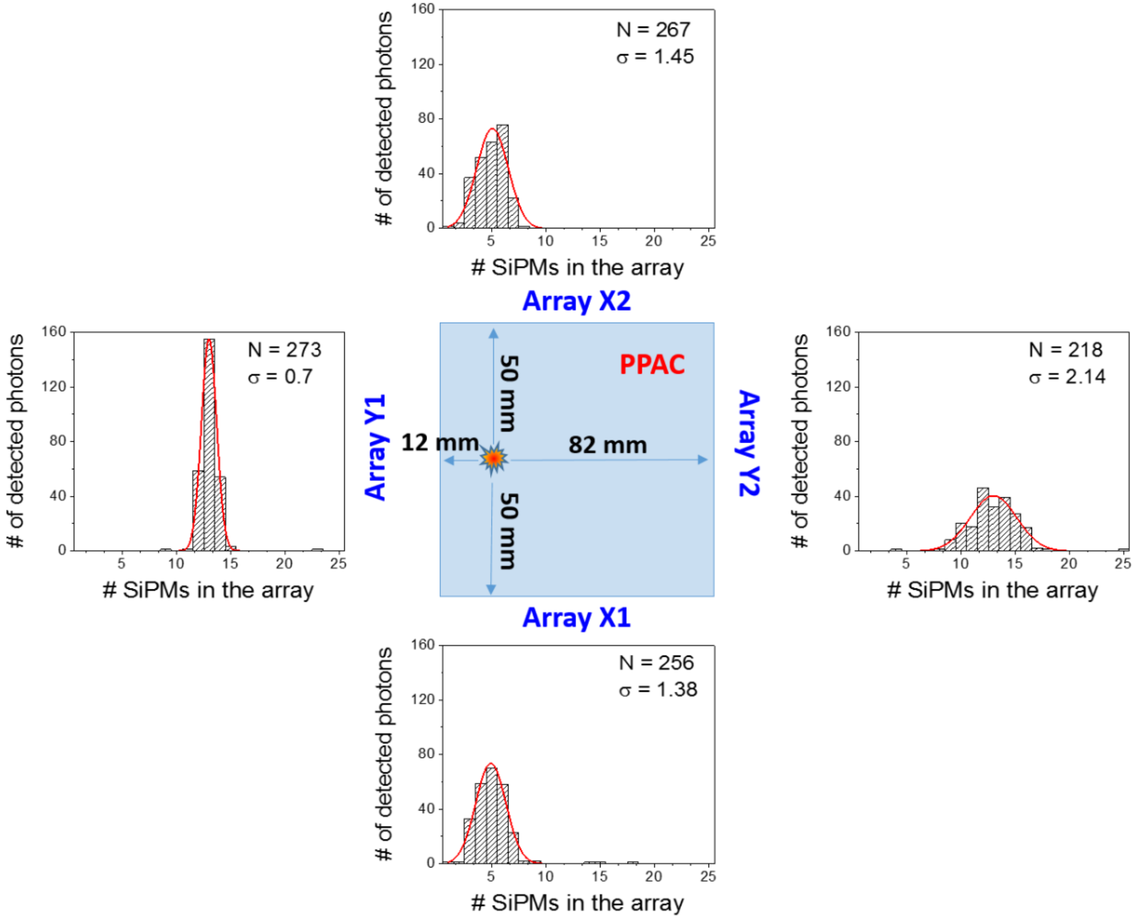


Image from 1808.05882

Reconstruction of the position

Reconstructed position $(\hat{x},\ \hat{y})$ obtained from the number of photons detected in each SiPM



Weighted average

$$\hat{x} = \frac{\left(\frac{P_{x1}\cdot N_{x1}}{\sigma_{x1}} + \frac{P_{x2}\cdot N_{x2}}{\sigma_{x2}}\right)}{\left(\frac{N_{x1}}{\sigma_{x1}} + \frac{N_{x2}}{\sigma_{x2}}\right)}$$

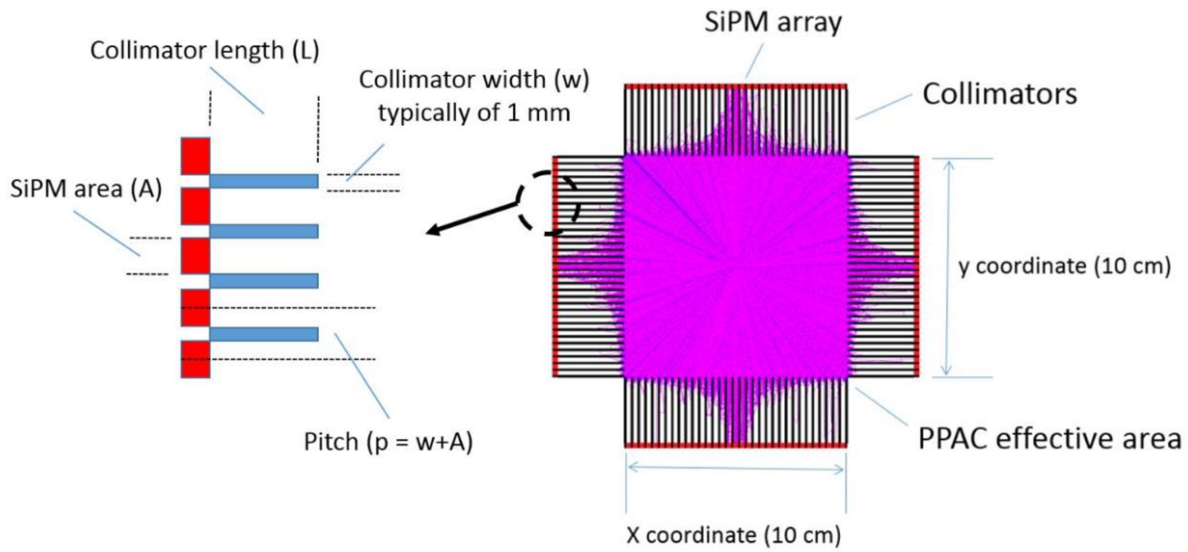
$$\hat{y} = \frac{\left(\frac{P_{y1}\cdot N_{y1}}{\sigma_{y1}} + \frac{P_{y2}\cdot N_{y2}}{\sigma_{y2}}\right)}{\left(\frac{N_{y1}}{\sigma_{y1}} + \frac{N_{y2}}{\sigma_{y2}}\right)}$$

Image from 1808.05882

Parameters of interest

- **Collimator Length (L):**

  - **Large L** → better resolution, poor statistics
  - **Small L** → worse resolution, better statistics



- **Pressure (p):**

  - High pressure → higher photon statistics

What is the optimal combination of these parameters?

Image from 1808.05882

2. Differentiable programming for experiment design

Differentiable programming for experiment design

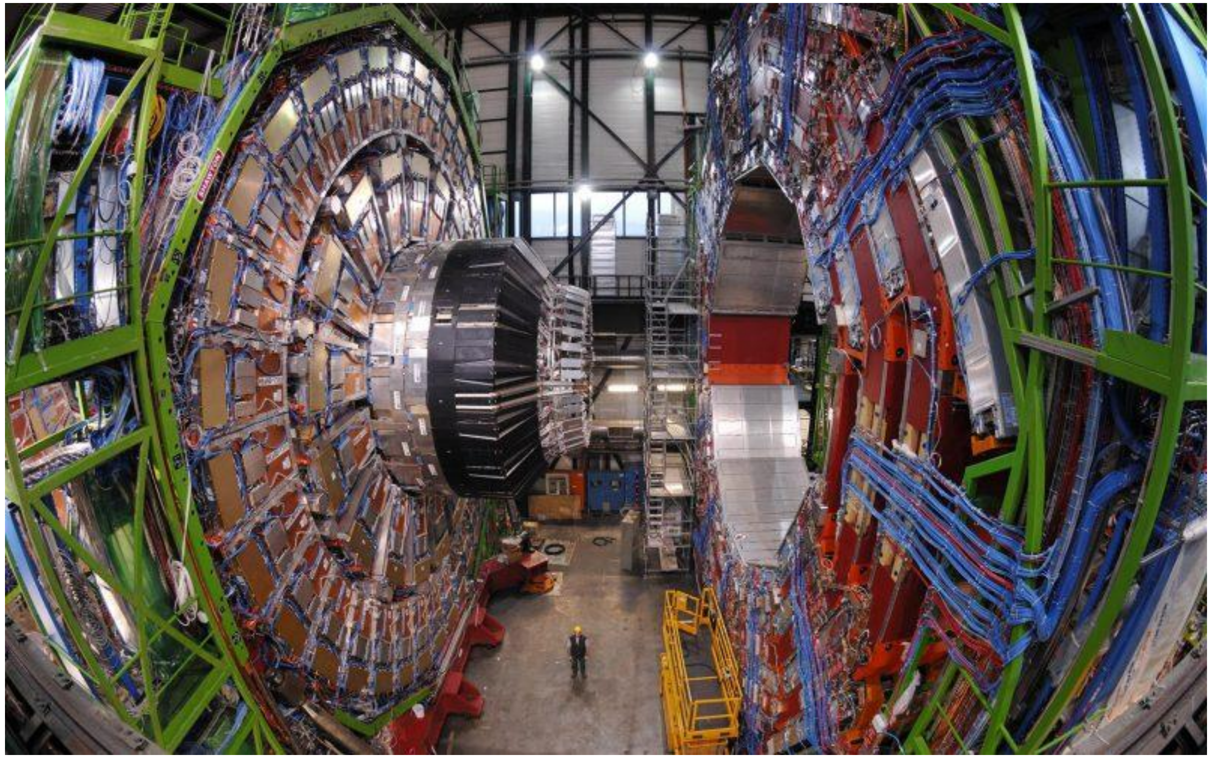
Designing experiments is a challenging task.

- Number of parameters can be **too high.**

- Correlations between parameters can be **non-trivial.**

- Traditional approaches are **computationally costly.**

Development of deep learning techniques allows us to explore new approaches



Differentiable programming for experiment design

1. Model detector modules as differentiable functions of n parameters ϕ_n.

2. Set loss function:

$$\mathcal{L} = \mathcal{L}(\phi_1, \phi_2, \dots \phi_n)$$

3. Minimise $\mathcal{L}$ w.r.t. ϕ_n with automatic differentiation.

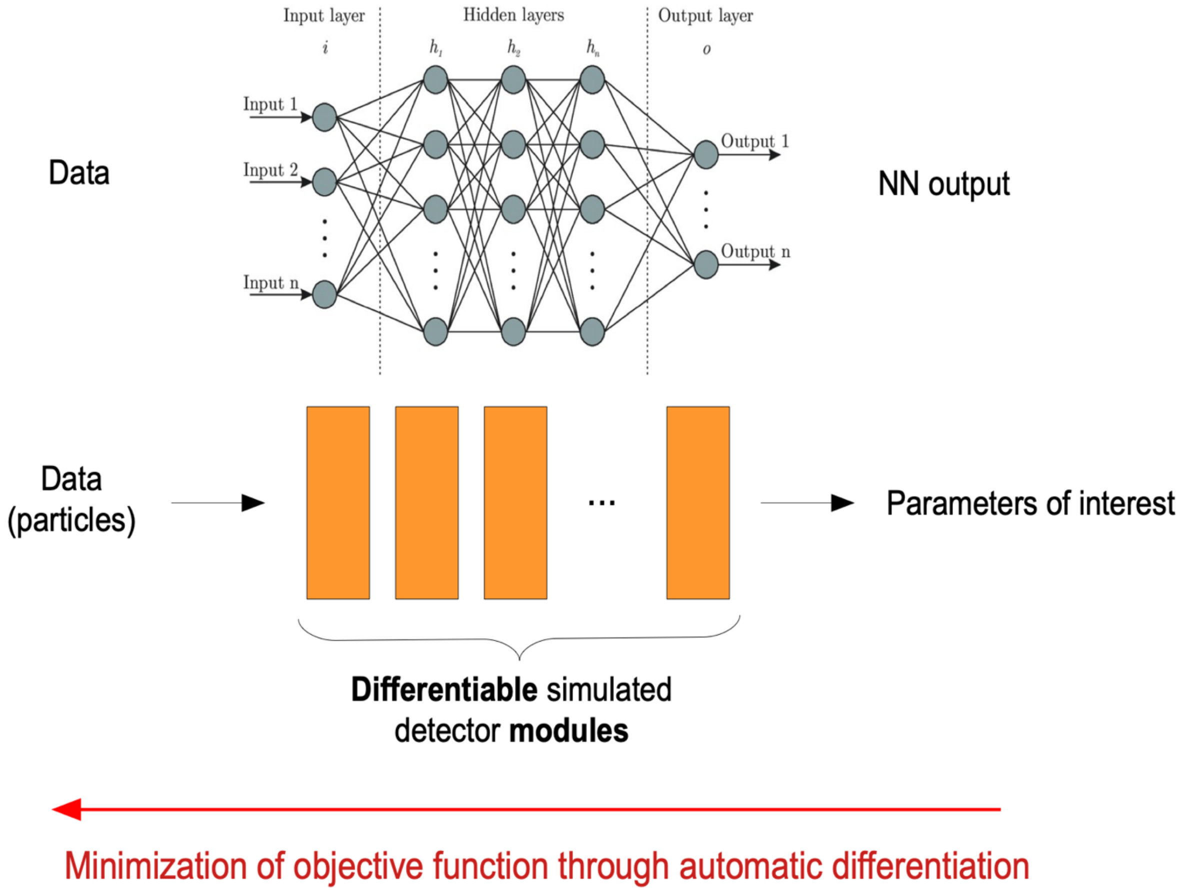


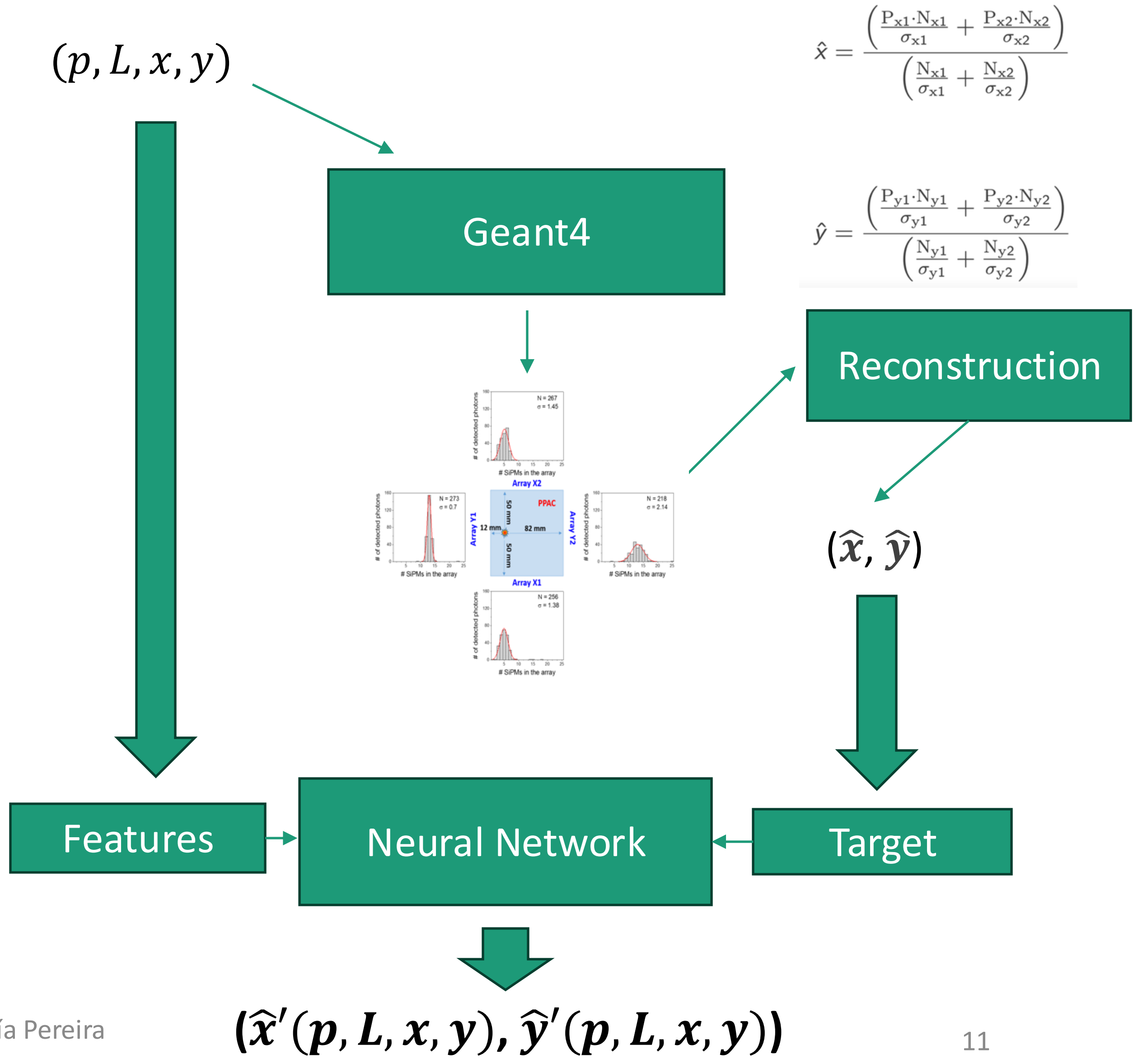
Image from [Julien Donini - Seminaire LPNHE - 14/02/2022]

3. Automatic optimization of the O-PPAC detector

Automatic optimization of O-PPAC

- **Geant4 is not differentiable.**
- Ongoing efforts on making it differentiable [(Differentable EM showers in Geant4 - 2405.07944)]
- **Surrogate modelling** is a practical workaround.
- **Approaches:**

1. Train a NN to predict the reconstructed position as a function of (p, L, x, y).

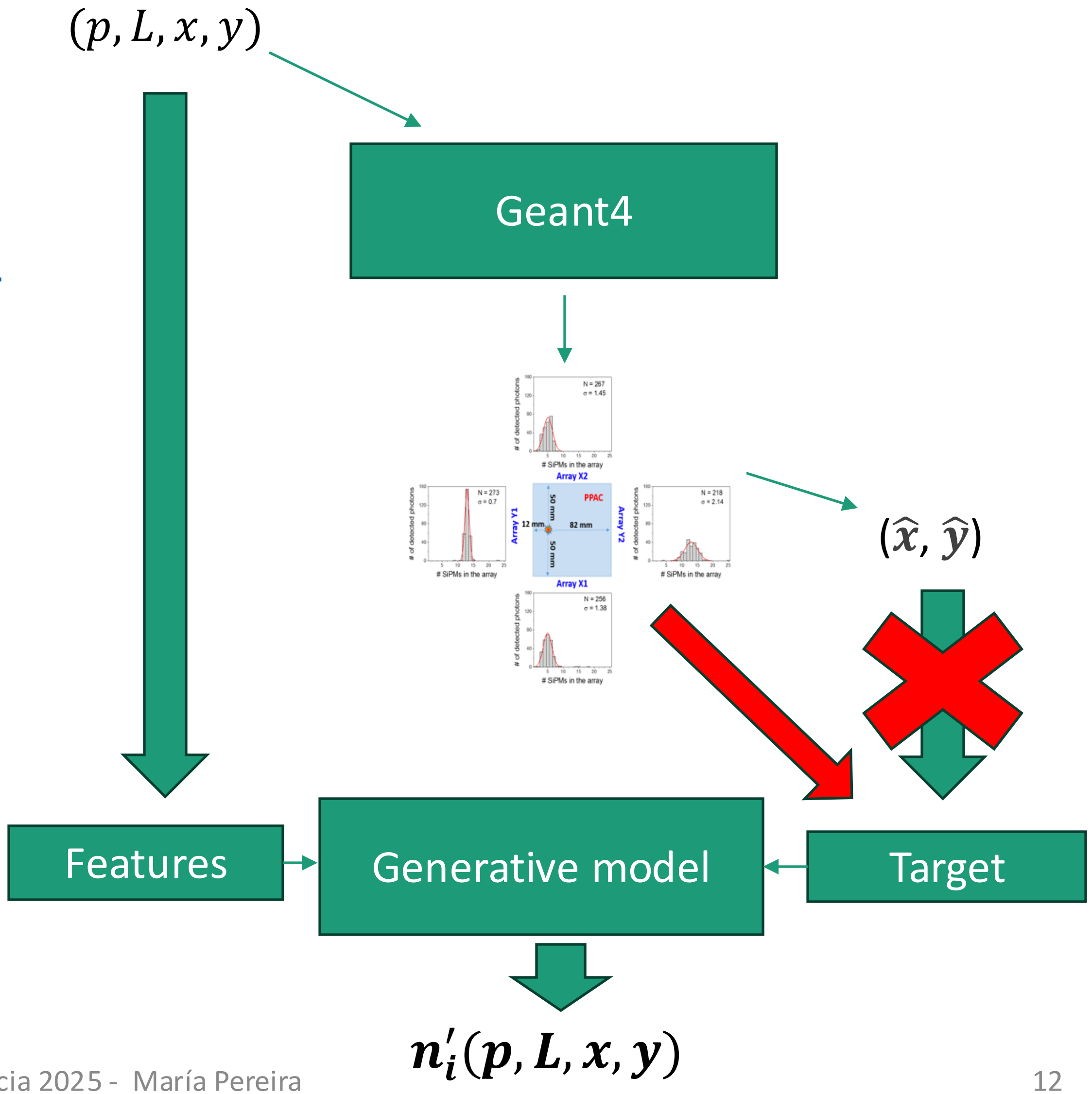


Automatic optimization of O-PPAC

- **Geant4 is not differentiable.**
- Ongoing efforts on making it differentiable [(Differentable EM showers in Geant4 - 2405.07944)]
- Surrogate modelling is a practical workaround.
- **Approaches:**

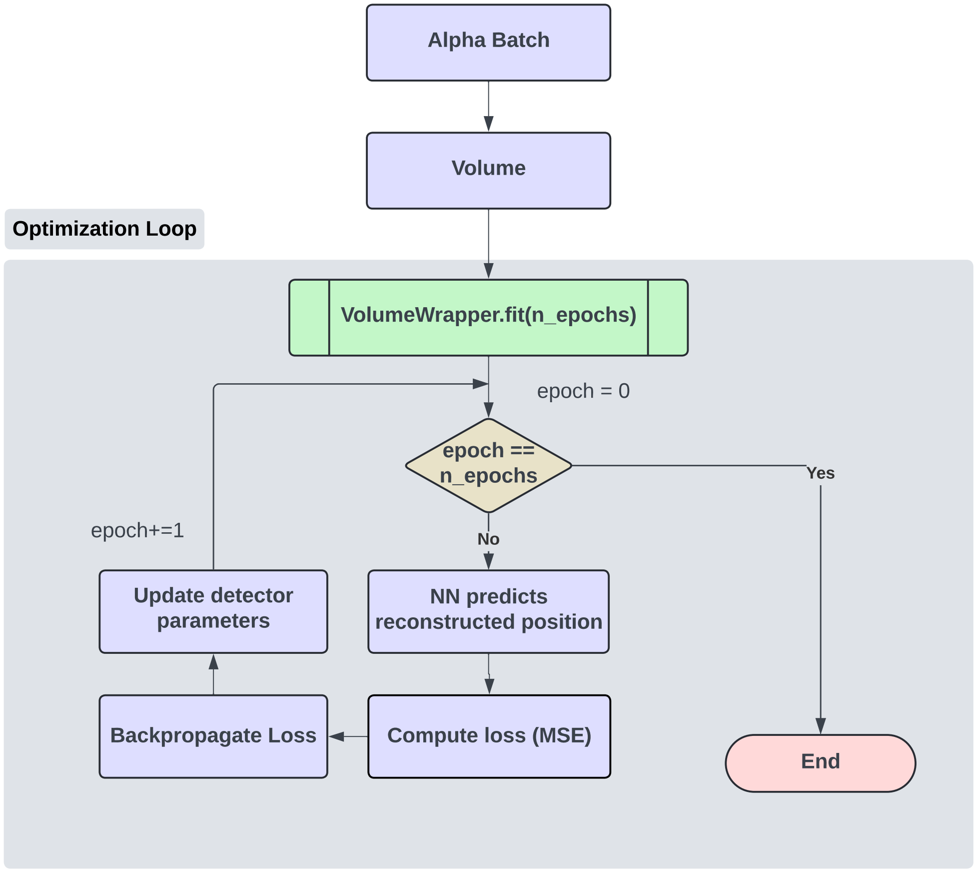
  1. Train a NN to predict the reconstructed position as a function of (p, L, x, y).

  2. Train a generative model to predict photon distributions from $(\boldsymbol{p}, \boldsymbol{L}, \boldsymbol{x}, \boldsymbol{y})$.

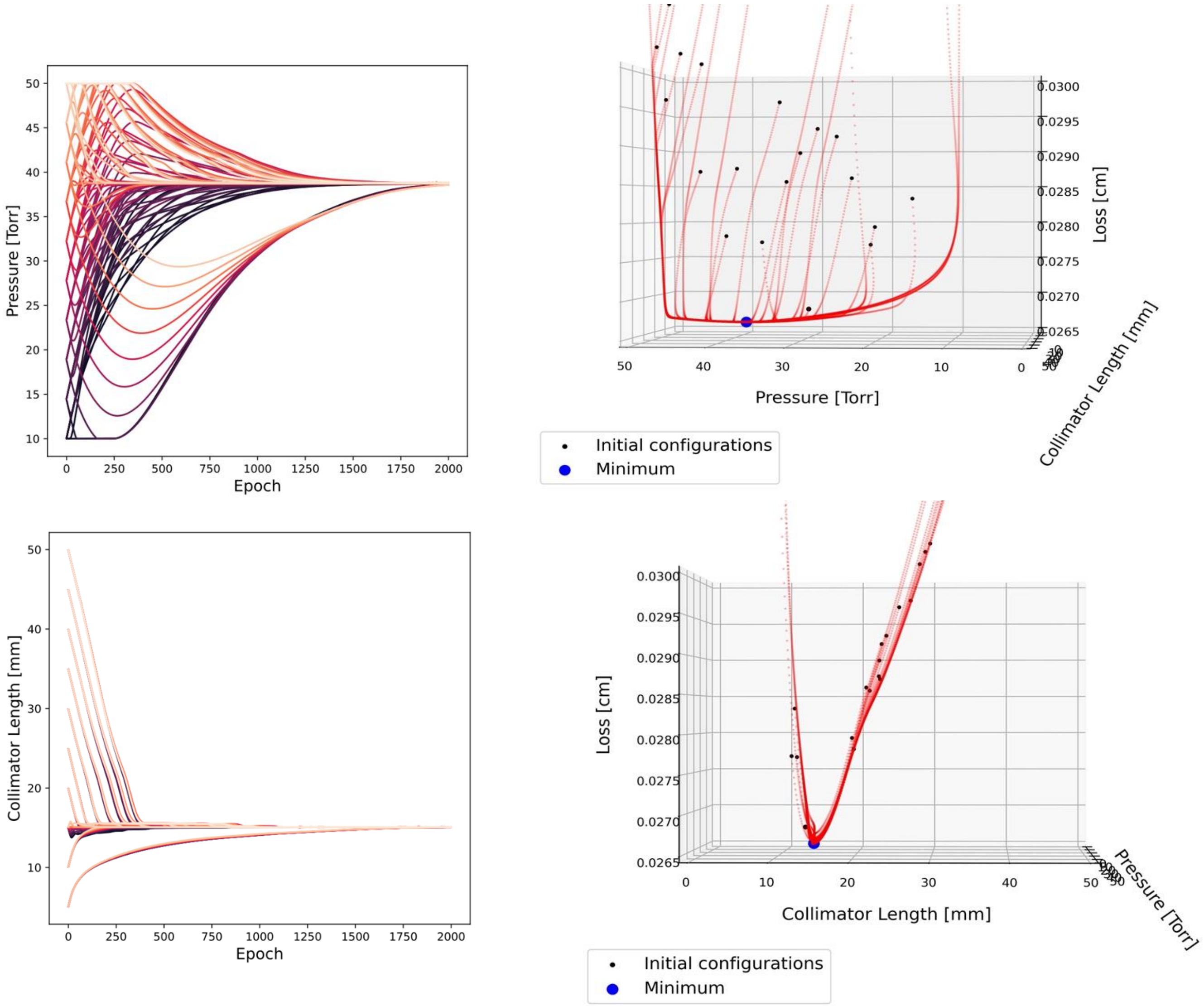


3.1. Neural Network Surrogate

1. NN surrogate



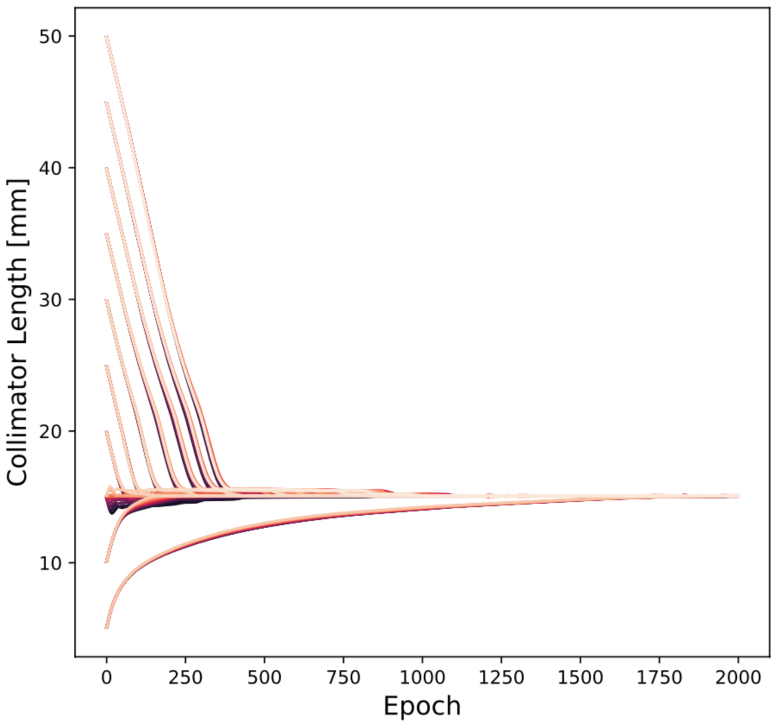
Solution remarkably stable regardless of initial configuration

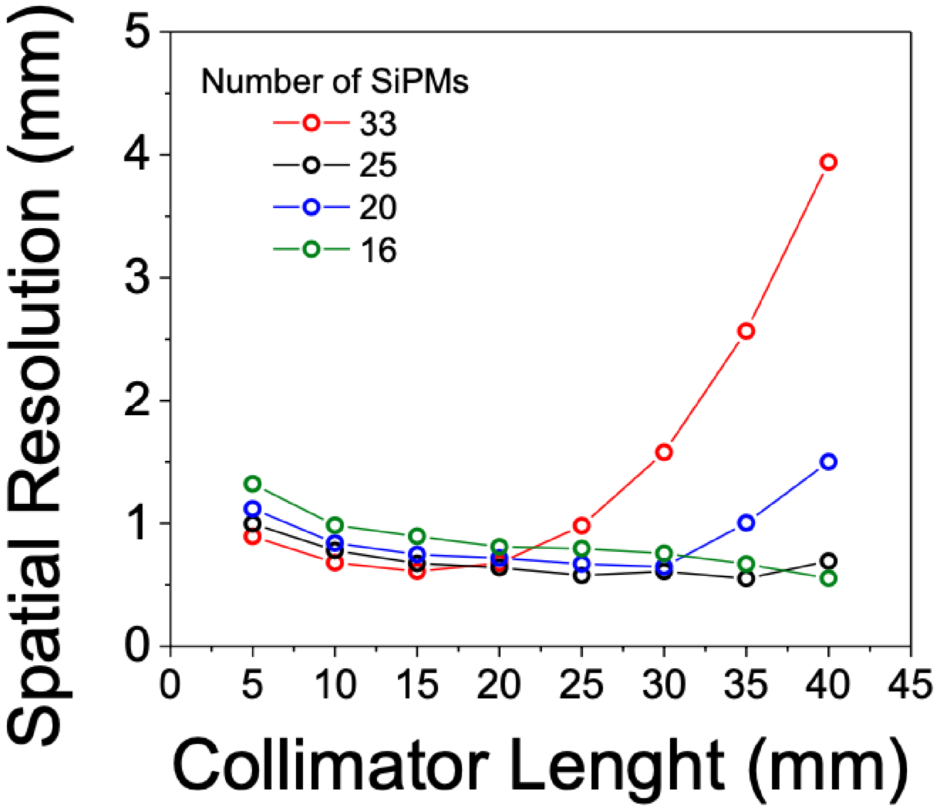


Results published in *Particles* **2025**, *8*(1), 26; https://doi.org/10.3390/particles8010026

1. NN surrogate

Collimator length result matches the traditional approach: 10.1088/1748-0221/13/10/P10006





Results published in *Particles* **2025**, *8*(1), 26; https://doi.org/10.3390/particles8010026

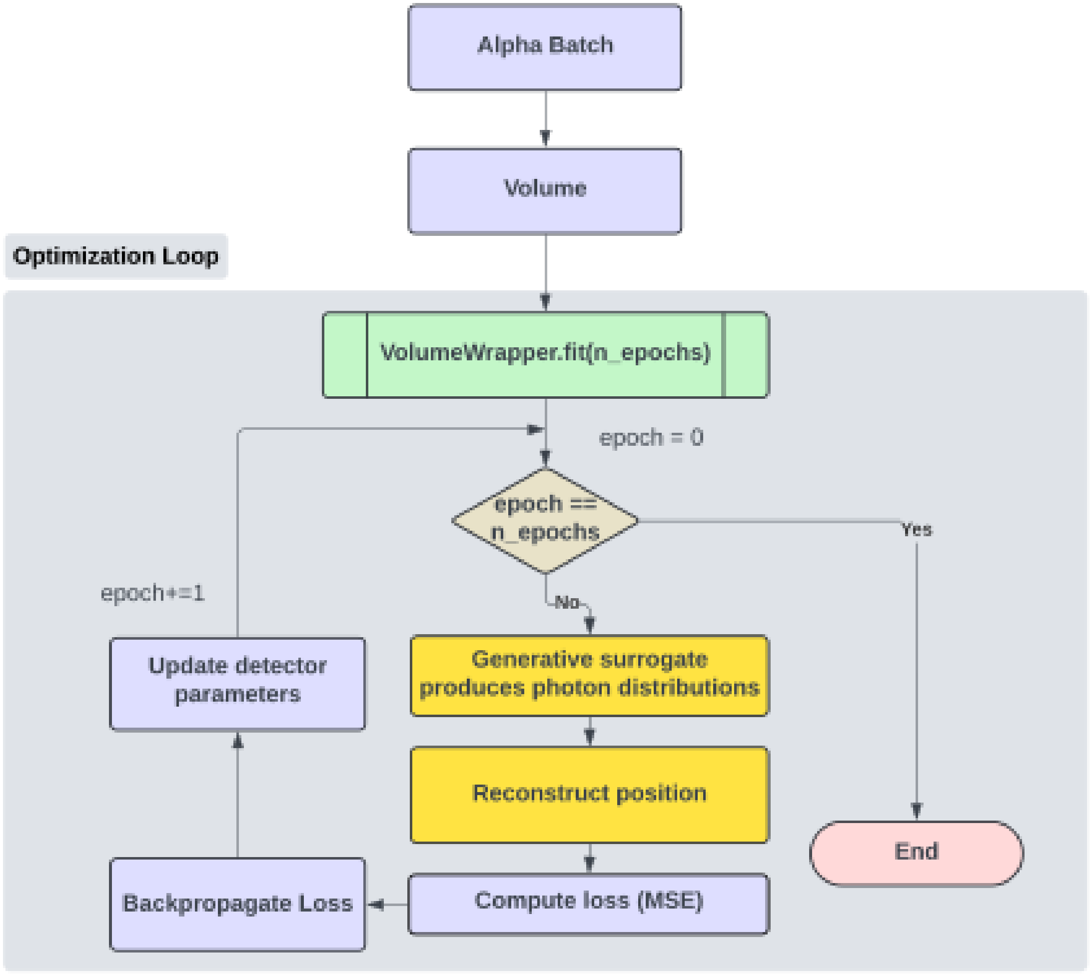
3.2. Generative surrogate

2. Generative Surrogate

- **Why generative surrogates**?
  - Can give a more detailed description of detector behaviour.
  - Allow us to **skip 'manual' reconstruction step.**
  - Freedom to change reconstruction step in the future.
  - Trained model will be **used later in a bigger pipeline.**

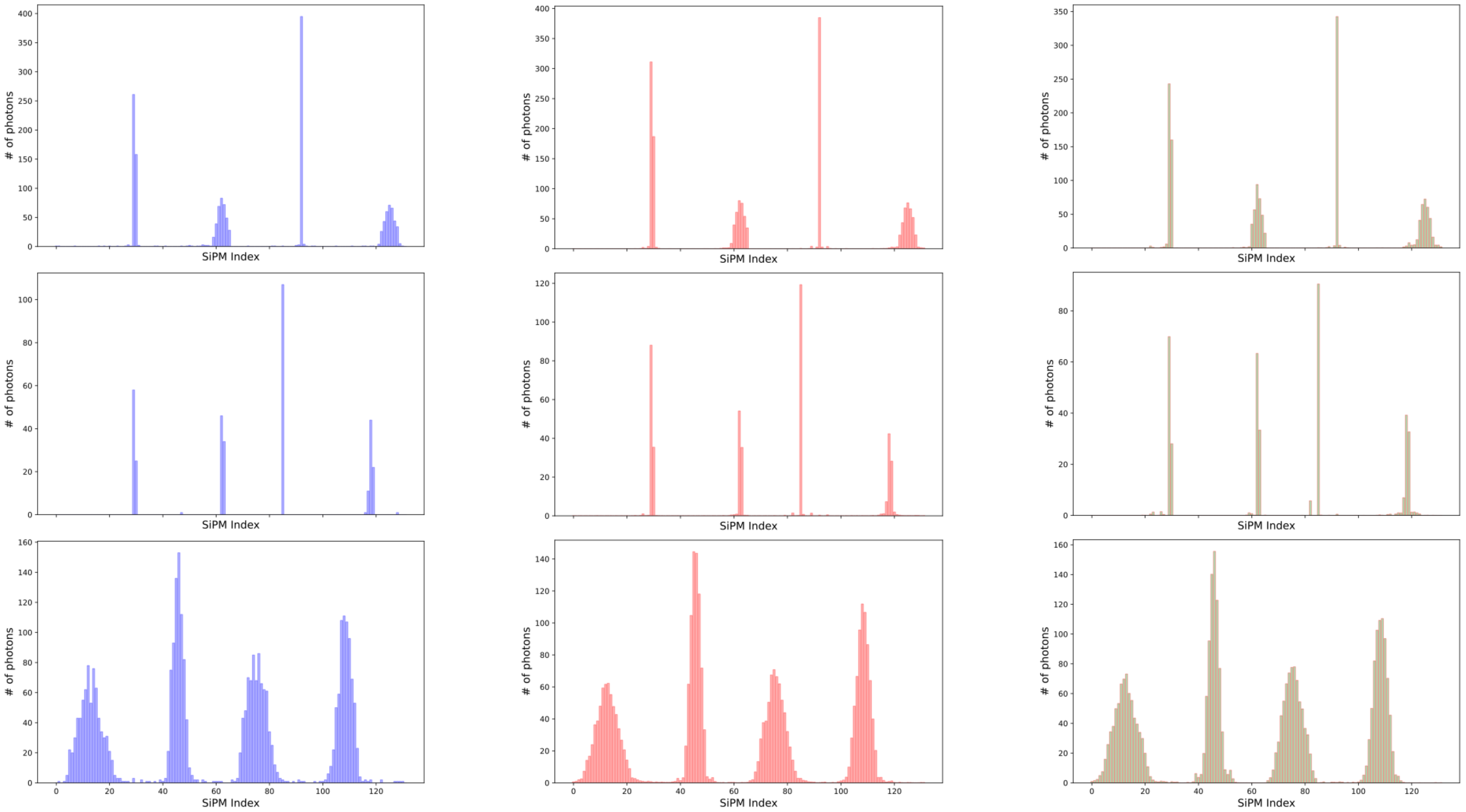
- **Models we are testing:**
  - GP-WGAN
  - β-VAE

- **Hyperparameter tuning:**
  - TPE algorithm parallelised in gpus.



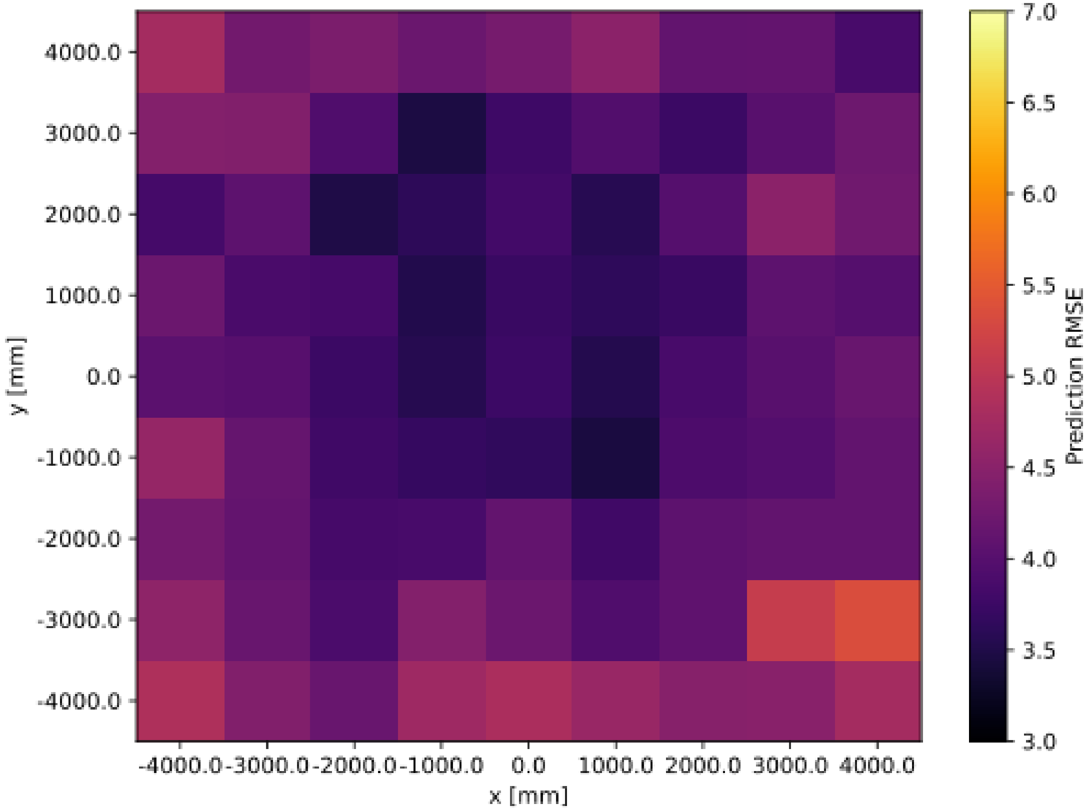
2. Generative Surrogate: Current status and challenges

Real events | GP-WGAN (RMSE = 5.2829) | β-VAE (RMSE = 5.2356)

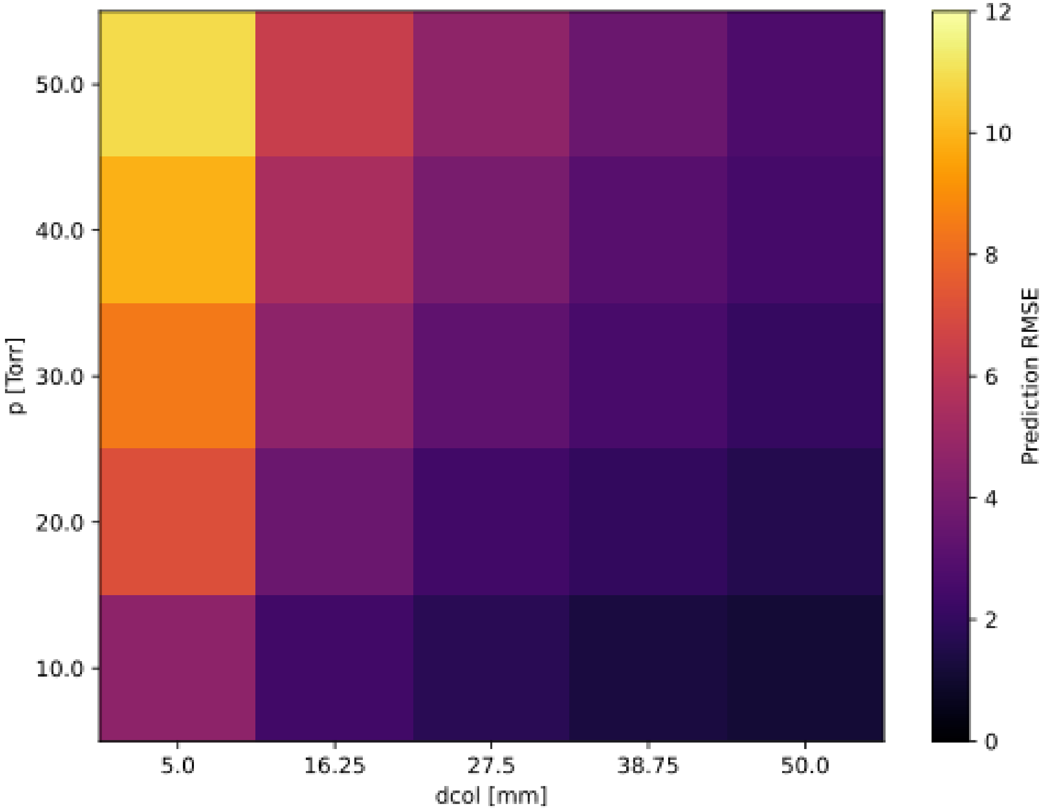


2. Generative Surrogate: Current status and challenges

Behaviour is pretty uniform in x, y



but not so uniform in p, L



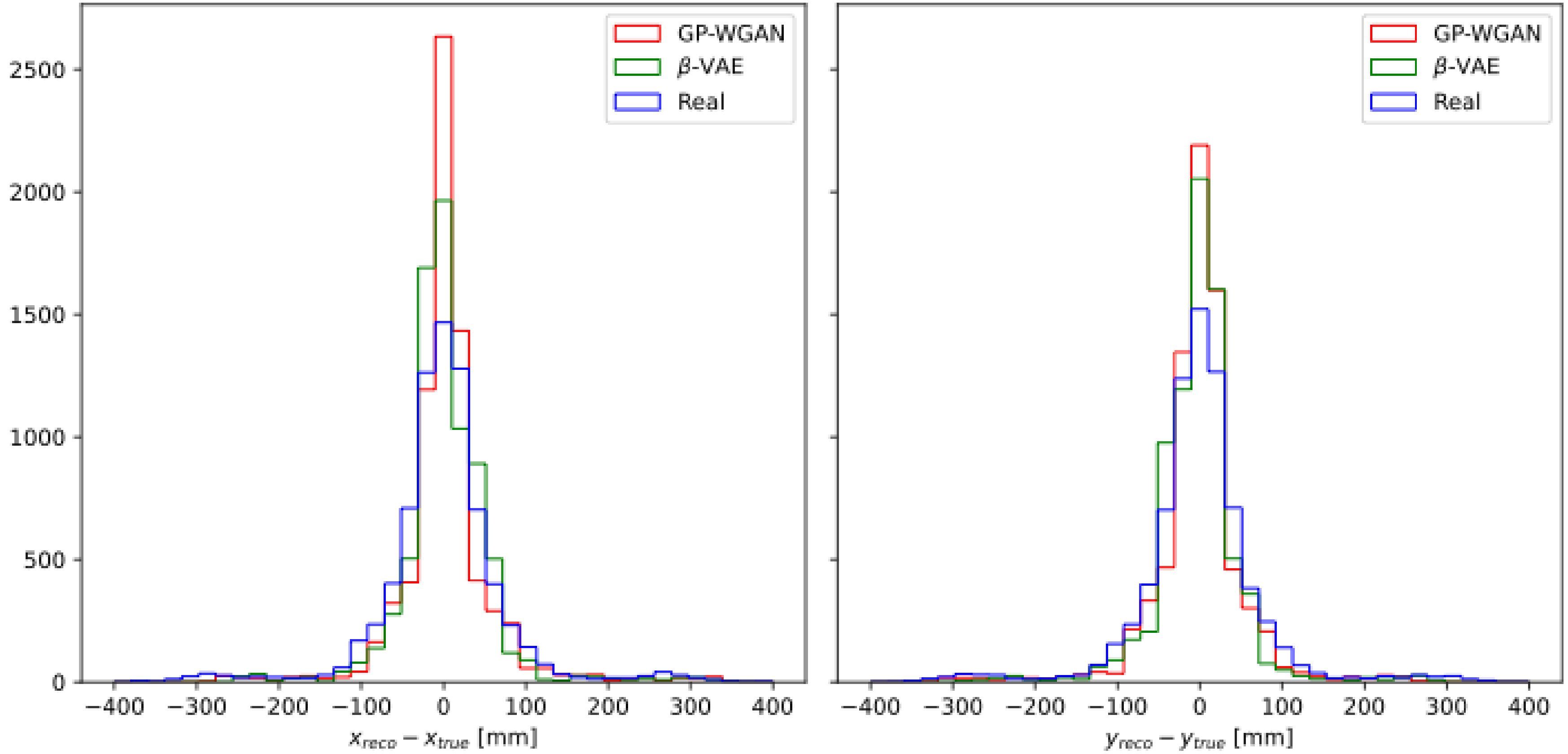
Proposed solutions:

- Train separate models for different regions.
- Penalize bad behaviour in this region through weights in loss function

2. Generative Surrogate: Current status and challenges

- **Interesting parameter:** error in reconstructed position

- β-VAE is better than GP-WGAN but there's room for improvement.



Conclusions

Conclusions

- We are applying **differentiable programming for** the optimization of the O-PPAC detector.
- **With NN surrogate approach**:
    - Solution independent **of the initial configuration**.
    - Collimator length result **aligns with traditional methods**.
- **With generative surrogate approach**:
    - GP-WGANs and β-VAEs show comparable performance.
    - Behaviour is dependent on parameter region.

Next steps

- Improve generative surrogate response throughout the parameter space.
- Try state-of-the-art models:
    - Diffusion models
    - Normalizing flows
- Build pipeline for the full tomography system.

Thank you for your attention!

Questions?

Backup slides

Vanilla GANs (2014) (https://arxiv.org/abs/1406.2661)

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}\big[\log D(\mathbf{x})\big] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}\big[\log\big(1 - D(G(\mathbf{z}))\big)\big]$$

- Discriminator tries to classify events as real or fake
- Generator tries to fool the discriminator into thinking that generated events are real

Know problems:

- Training can converge without good results. Loss doesn’t give us information of the quality of the samples

-Inestability: D and G have to be very coordinated through architecture and learning rate. If not, convergence is not guaranteed.

- **Mode Collapse:** G learns to fool D producing the same event or very few events.

- **Overtrained discriminator:** if D is very good at classifying events, G cannot learn

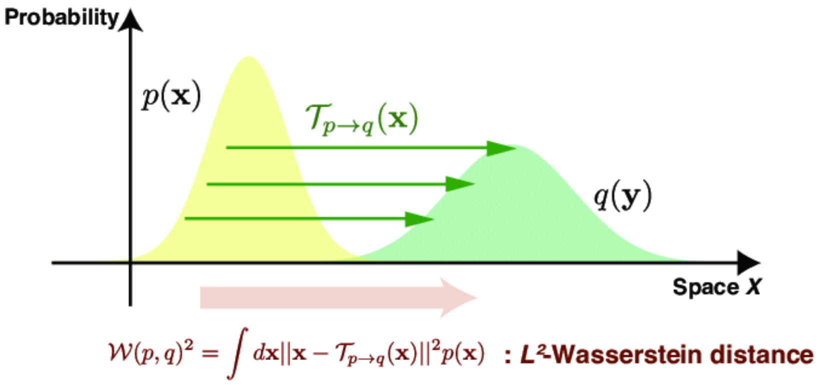
Wasserstein GANs (2017) (https://arxiv.org/pdf/1701.07875)

The *Earth-Mover* (EM) distance or Wasserstein-1

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \big[\, \|x - y\| \, \big] ,$$

On the other hand, the Kantorovich-Rubinstein duality [22] tells us that

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x)]$$



We can use the critic (old discriminator) function as an estimator of the W distance, while the generator tries to minimize it

$$\min_G \max_{D \in \mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r} \big[D(\boldsymbol{x})\big] - \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g} \big[D(\tilde{\boldsymbol{x}}))\big]$$

K-Lipschitz constraint:

$$|f(x_1) - f(x_2)| \leq K|x_1 - x_2|$$

Wasserstein GANs: keys

- Train the critic more often than the generator to have a good estimation of WD before updating G
- Don’t use batch norm on the critic
- Enforce the Lipschitz constraint. Two ways:
    - Gradient clipping (not ideal)
    - Gradient penalty (https://arxiv.org/pdf/1704.00028)

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.
Require: initial critic parameters w_0, initial generator parameters θ_0.

```
while θ has not converged do
    for t = 1, ..., n_critic do
        for i = 1, ..., m do
            Sample real data x ~ P_r, latent variable z ~ p(z), a random number ε ~ U[0, 1].
            x̃ ← G_θ(z)
            x̂ ← εx + (1 − ε)x̃
            L^(i) ← D_w(x̃) − D_w(x) + λ(‖∇_x̂ D_w(x̂)‖_2 − 1)^2
        end for
        w ← Adam(∇_w (1/m) Σ_{i=1}^m L^(i), w, α, β1, β2)
    end for
    Sample a batch of latent variables {z^(i)}_{i=1}^m ~ p(z).
    θ ← Adam(∇_θ (1/m) Σ_{i=1}^m −D_w(G_θ(z)), θ, α, β1, β2)
end while
```

β-VAE

L = RMSE(input, output) + **β** KLD(p(z|x), gaussian)