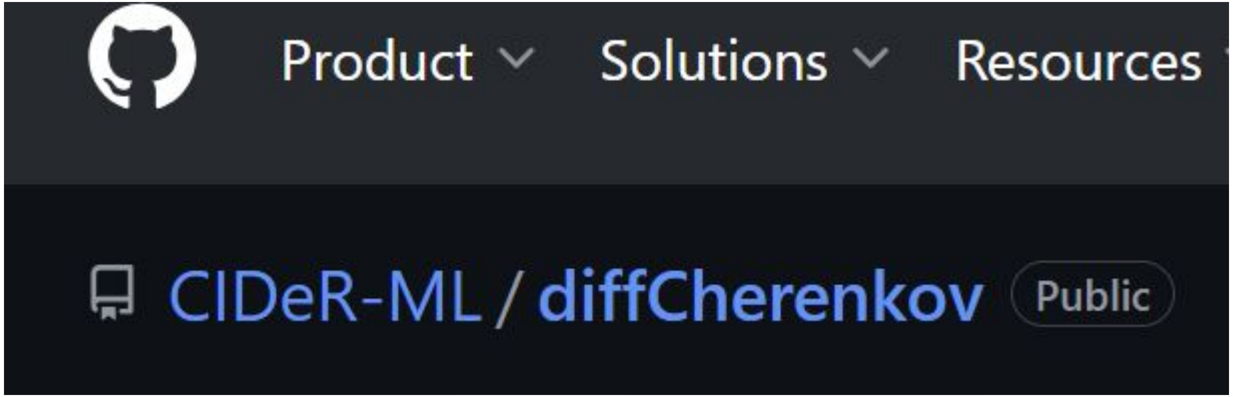
11/15

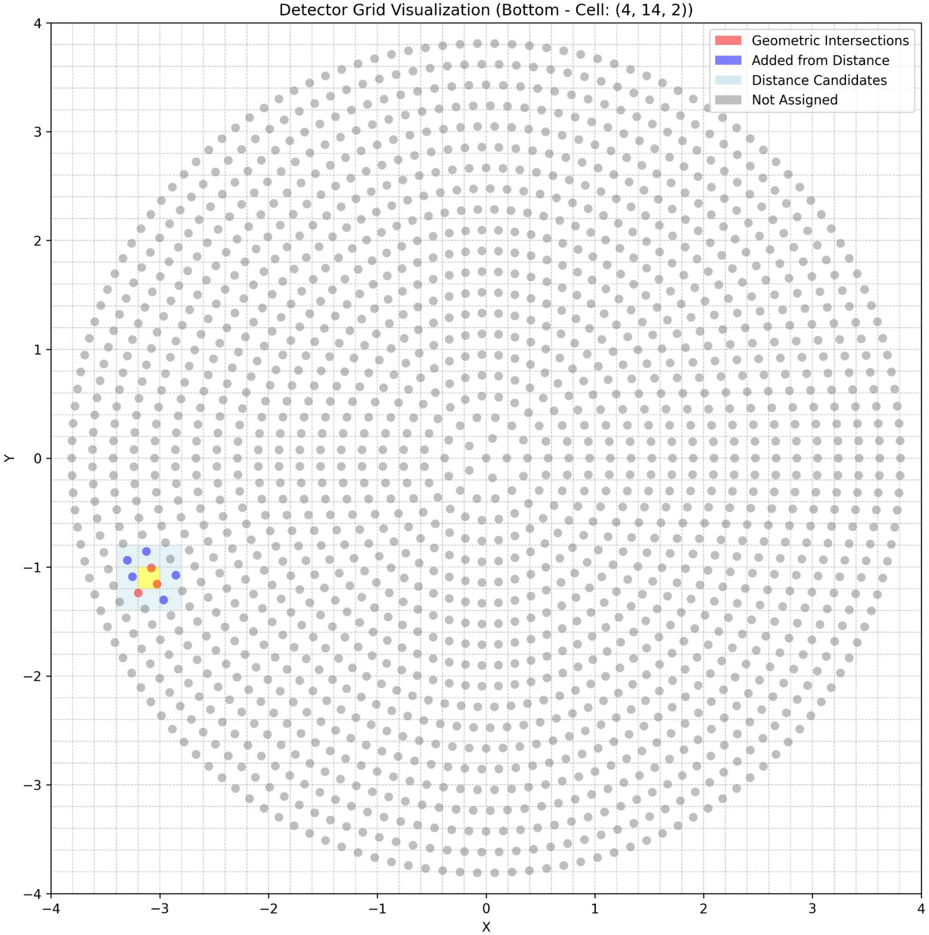
Currently in the process of cleaning everything up to push updated code. I expect to push the code today with all the updates I have worked on.

I will mention in the end what are the next big steps for short term and long term for the project



11/15

- After the smoothing update mentioned last week, the code is a bit more robust to the loss and hyperparameters (sigma_position and sigma_time)).
- Instead, The code is now a bit sensitive to the smoothing effect we have. But this is because grids have a hard cutoff for which detectors to check.
- I currently have (an almost working) version that checks closeby detectors as well.



Next Steps (PMT)

So I spent some time during this project trying to robustly have the contribution of photons per PMT. This makes it possible to compare PMTs to PMTs

This allows for multiple things, but mainly:

- It is easier to include a smoothing of stochastic effects. Because a photon can now contribute to both the PMTs it nearly hit and the ones after scattering
- One can add PMT surface effects such as angular acceptance and so on. This could also possibly be learned/finetuned in the future on data

Next Step (Stochasticity)

So I spent some time during this project trying to robustly have the contribution of photons per PMT. This makes it possible to compare PMTs to PMTs

This allows for multiple things, but mainly:

- So the main next step is to include stochasticity through Gumbel-Softmax for reflections and then for scattering. Then to run the same gradient tests and see how much bias we get

Next Step (Stochasticity)

So I spent some time during this project trying to robustly have the contribution of photons per PMT. This makes it possible to compare PMTs to PMTs

This allows for multiple things, but mainly:

- So the main next step is to include stochasticity through Gumbel-Softmax for reflections and then for scattering. Then to run the same gradient tests and see how much bias we get

Next Step (Photon Generation)

- Currently, we very simply sample photons along the line trajectory and have cones along the line. Need more realistic photon generation. This is where the cherenkov profile siren work comes in:
    - One question is given the gaps we see in cherenkov profiles during last workshop, is there anything we can do?
    - The first question is, how accurate is the approximation of having a straight line. I got the particle data watchmal uses to look at this, but still haven't gotten around to it. But that should us a more quantitative answer to this question!

Next Step (Photon Generation)

- I was discussing with Patrick about a possible project idea to solve this problem. To solve the gaps in the cherenkov profile we see during the last workshop (stochasticity):
    - What if we try to develop a diffusion model version of the cherenkov siren.

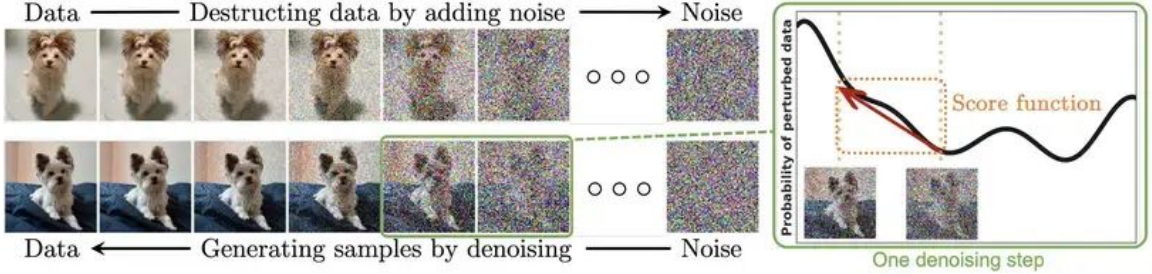

Data
Destructing data by adding noise
Noise
Probability of perturbed data
Score function
Data
Generating samples by denoising
Noise
One denoising step

Fig. 2. Diffusion models smoothly perturb data by adding noise, then reverse this process to generate new data from noise. Each denoising step in the reverse process typically requires estimating the score function (see the illustrative figure on the right), which is a gradient pointing to the directions of data with higher likelihood and less noise.

Next Step (Photon Generation)

Denoising Diffusion Models work by training a model to learn the process of slightly denoising an image. This makes it simple to sample from the random noise and then go to our true distribution through the model.

This could be a new project for someone on the Japan side to work on

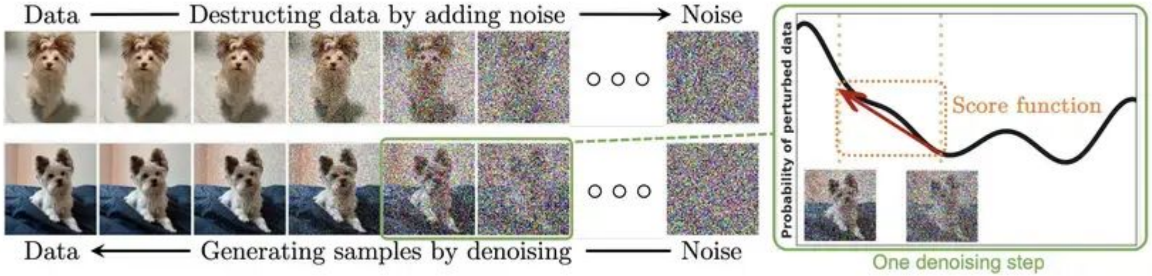

Data
Destructing data by adding noise
Noise
Data
Generating samples by denoising
Noise
Probability of perturbed data
Score function
One denoising step

Fig. 2. Diffusion models smoothly perturb data by adding noise, then reverse this process to generate new data from noise. Each denoising step in the reverse process typically requires estimating the score function (see the illustrative figure on the right), which is a gradient pointing to the directions of data with higher likelihood and less noise.

Next Step (More realistic simulation)

Need to add the correct positions for Super K PMTs. Benda was interested in joining this project. So that is the first step of possible help regarding this.