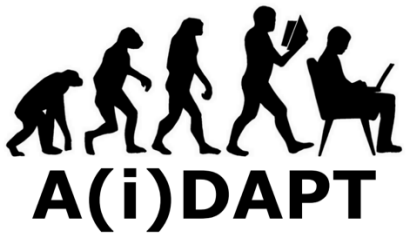
GANs towards data smearing and acceptance corrections

Tommaso Vittorini – Unige, INFN Genova

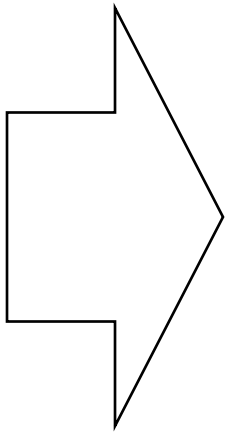
on behalf of A(i)DAPT Working Group

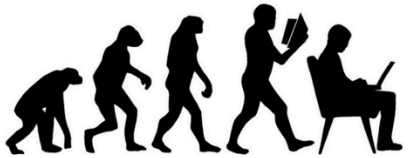


Digital Twins for Nuclear and Particle Physics – NPTwins
16-18 December 2024

- Data collected by NP/HEP experiments are (always) affected by the detector's effects
- Before starting physics analysis the detector's effect unfolding is required
- Traditional observables may not be adequate to extract physics in multidimensional space (multi-particles in the final state)
- At High-Intensity frontiers, data sets are large and difficult to manipulate/preserve

Should AI support NP/HEP experiments to extract physics from data in more efficient way?





A(i)DAPT

AI for Data Analysis and PreservaTion

Develop AI – supported procedures to:

- Accurately fit data in multiD space
- Unfold detector effects
- Compare synthetic (AI-generated) to experimental data
- Quantify the uncertainty (UQ)

Collaborative effort (regular meeting)

- ML experts (ODU, Jlab)
- Experimentalists (Jlab Hall-B)
- Theorists (JPAC, JAM)

Overview

Goal: Implement a single GAN which is able to take care all the detector effects:

- Present GANs as tools to accurately fit multi-dimensional cross-sections in the different regions of the phase space available to different experiments

- Provide a tool to take into account for all the detector effects (smearing, acceptance, detector inefficiencies) in the most accurate way and unfold them to recover the vertex level distributions

- Focus on the study of the acceptance problem: Quantify how the increasing coverage of the phase space improves the reproduction of the data in the measured regions

Detector unfolding

- Detector effects make measured observables (detector-level) different from the ‘true’ observables (vertex level)

  Acceptance: Any measurement can access only a limited portion of the phase space. What can we say about these unmeasured regions?

  - Interpolation: deal with the holes in the phase space
  - Extrapolation: extend our coverage from the borders of measured regions
  - Inefficiencies: Regions with mixed topologies

  Resolution: Any measurement has an experimental resolution that may modify cover up effects that we’re looking for

  - Spikes may be concealed behind the detector resolution
  - Measurements could be extended to unphysical regions

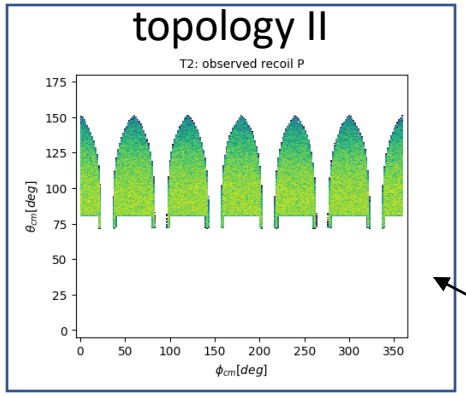
- Mitigation strategy:
  - Acceptance: ‘Fiducial volumes’ to exclude unmeasured regions and extend the covered measured of the phase space
  - Resolution: build and validate ML-models to unfold resolution effects

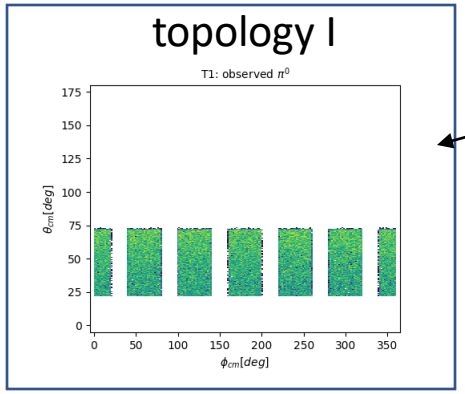
Acceptance regions definition

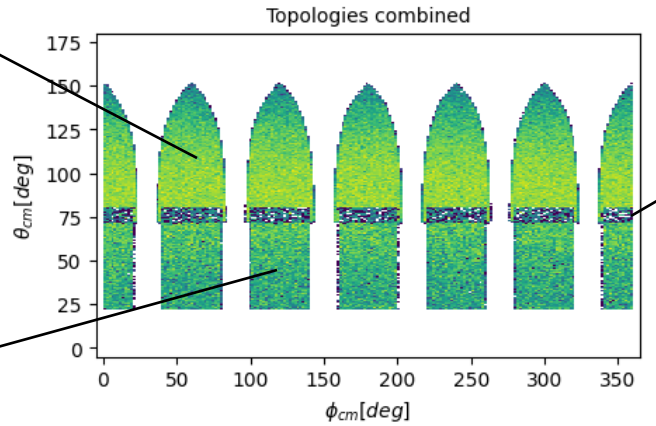
- Simple 2-body process: $\gamma p \rightarrow \Delta^{+}(1232) \rightarrow \pi^{0} p$
- Two independent variables (at fixed energy): θ_{cm} and ϕ_{cm}
- Monte Carlo event generator

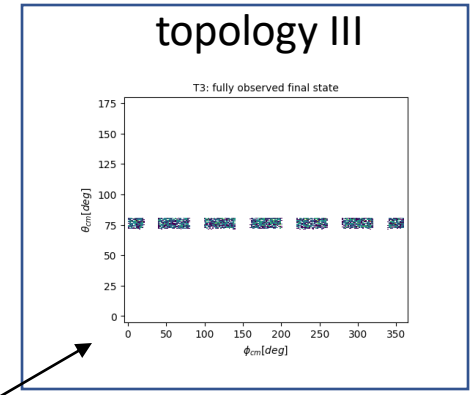
- Detector acceptance (CLAS) implemented via fiducial cuts (coils, minimum proton momentum and angle in the lab frame)
  - topology 1: $\gamma p \rightarrow (p)\pi^{0}$(proton missing)
  - topology 2: $\gamma p \rightarrow p(\pi^{0})$ (π^{0} missing)
  - topology 3: $\gamma p \rightarrow p\pi^{0}$ (all detected)
  - [topology 0: unmeasured]

Build a single Network able to generate in the full phase space according to the correct distributions







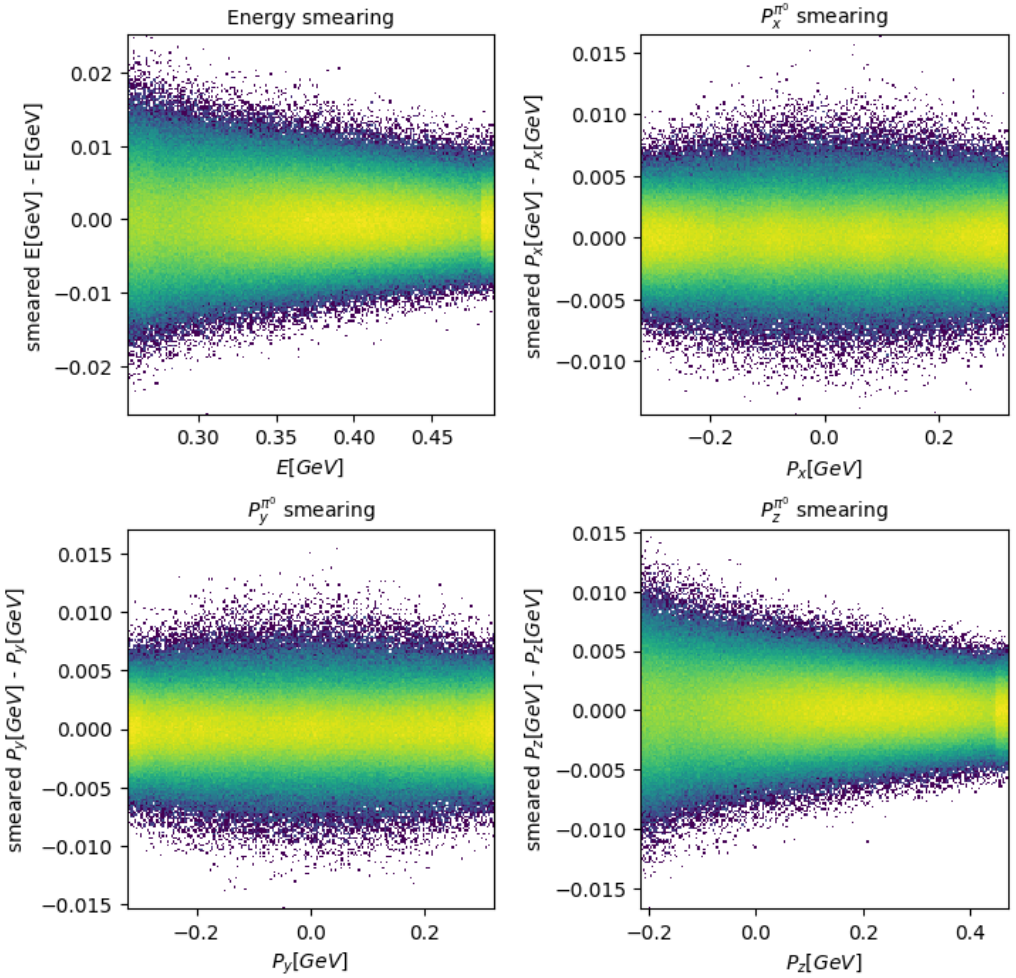


Credit: T.Vittorini , T.Alghamdi,Y. Li

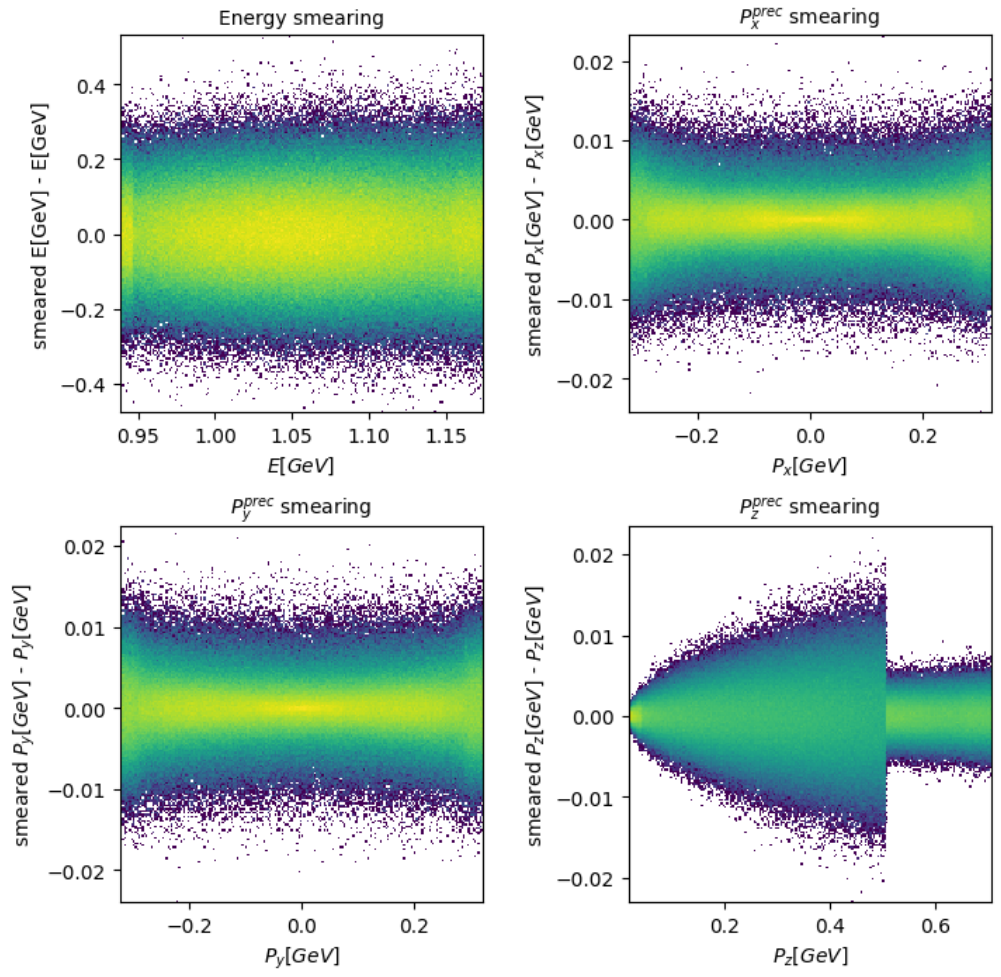
Simple smearing function

- Reasonable smearing applied to the training variables:

π^0 4-momenta



Recoil proton 4-momenta

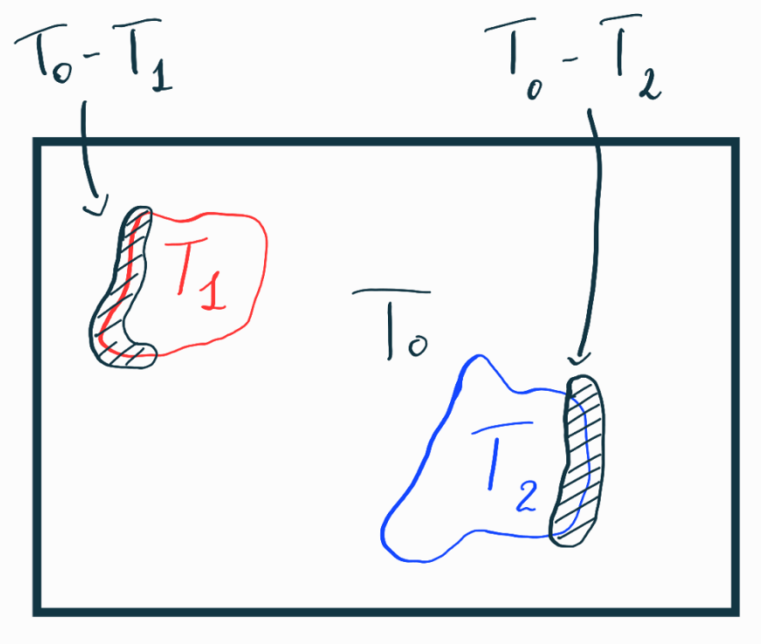


Detector inefficiencies

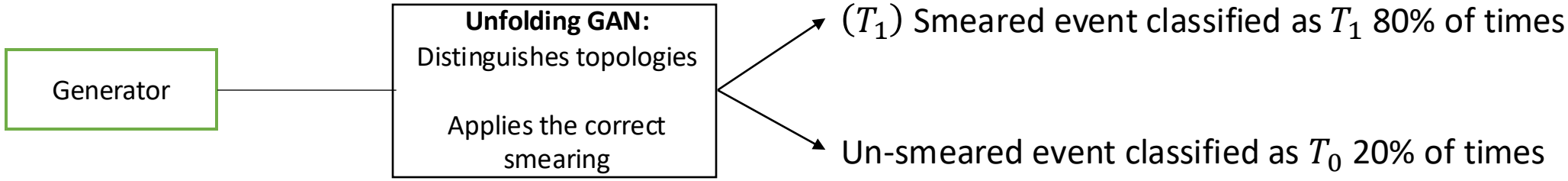
- Some evets may sometimes be classified in the wrong category:

  - Defective paddles can lead to missing events (T_1 which become T_0)

  - Reconstructed events can get missclassified (T_1 which become T_2 and viceversa)



- Existence of regions where a given generated event can go into different topologies with a given probability: Mixed events

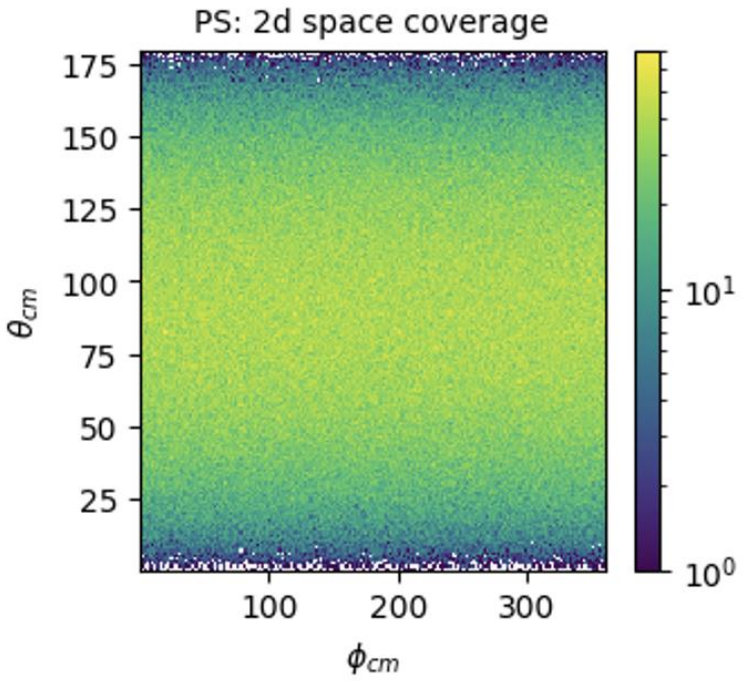


Dataset definition

- To test and validate our procedure we consider three different models:

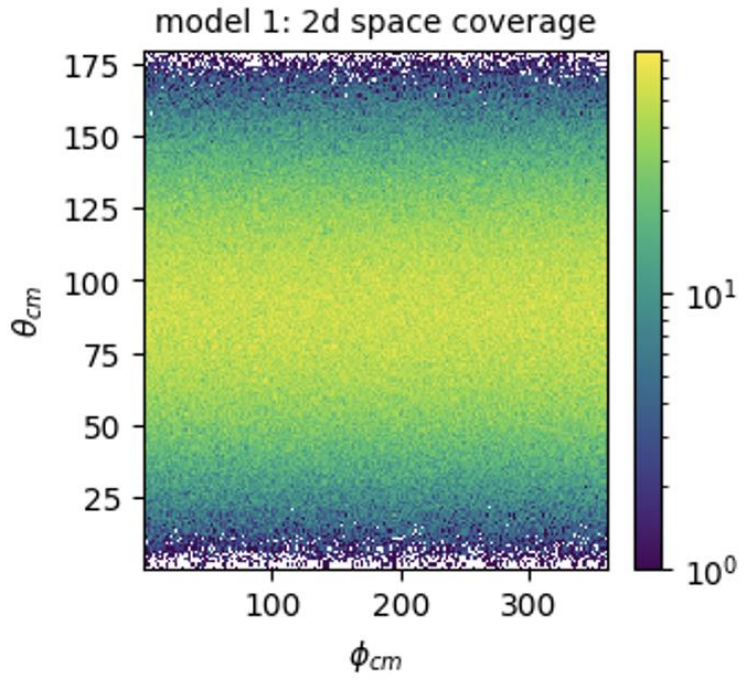
Phase space model

$$\frac{d\sigma}{d\Omega} = \frac{1}{64\pi^2}\frac{1}{4}\frac{p_f}{p_i\, s} * 1$$



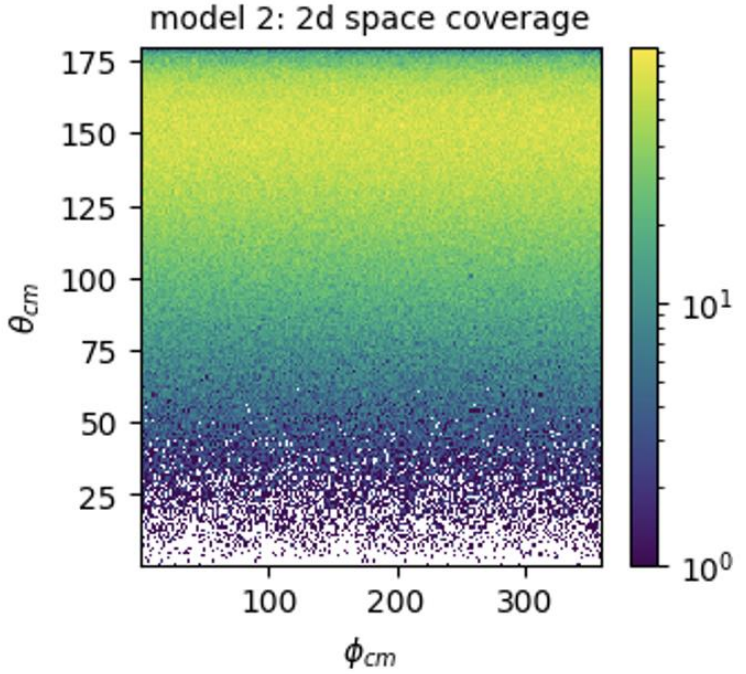
Model 1

$$\frac{d\sigma}{d\Omega} = \frac{1}{64\pi^2}\frac{1}{4}\frac{p_f}{p_i\, s}\frac{3\left|H_{3/2}\right|^2 + 5\left|H_{1/2}\right|^2}{(m_\Delta^2 - s)^2 + \Gamma_\Delta^2 m_\Delta^2} - \frac{3\cos 2\theta\left(\left|H_{3/2}\right|^2 - \left|H_{1/2}\right|^2\right)}{(m_\Delta^2 - s)^2 + \Gamma_\Delta^2 m_\Delta^2}$$



Model 2

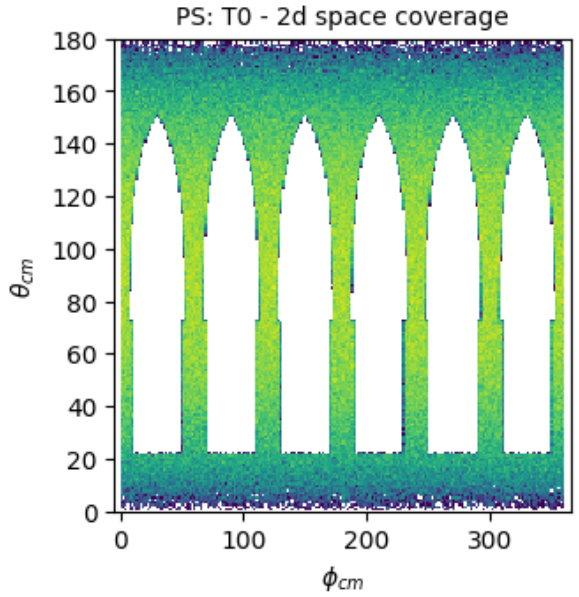
$$\frac{d\sigma}{d\Omega} = \frac{1}{64\pi^2}\frac{1}{4}\frac{p_f}{p_i\, s}\frac{3\left|H_{3/2}\right|^2 + 5\left|H_{1/2}\right|^2}{(m_\Delta^2 - s)^2 + \Gamma_\Delta^2 m_\Delta^2} - \frac{\left(\left|H_{3/2}\right|^2 - \left|H_{1/2}\right|^2\right)3e^{2\theta}}{(m_\Delta^2 - s)^2 + \Gamma_\Delta^2 m_\Delta^2},$$

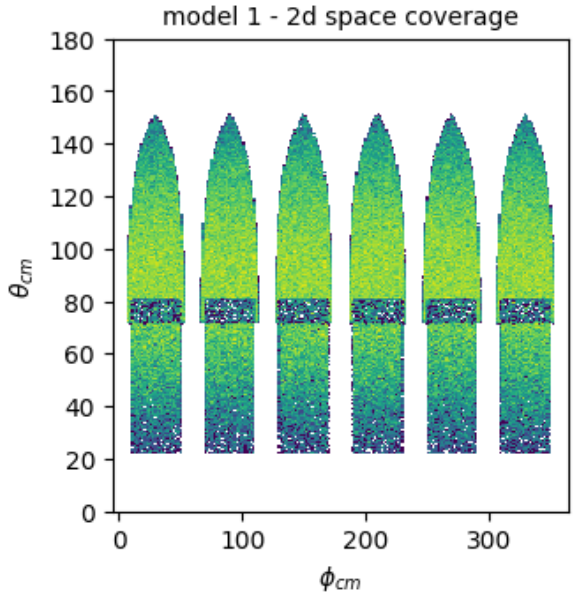


Dataset definition

Dataset 1:

$$T^{D1} = (T^{0^{D1}}, T^{meas^{M1}})$$



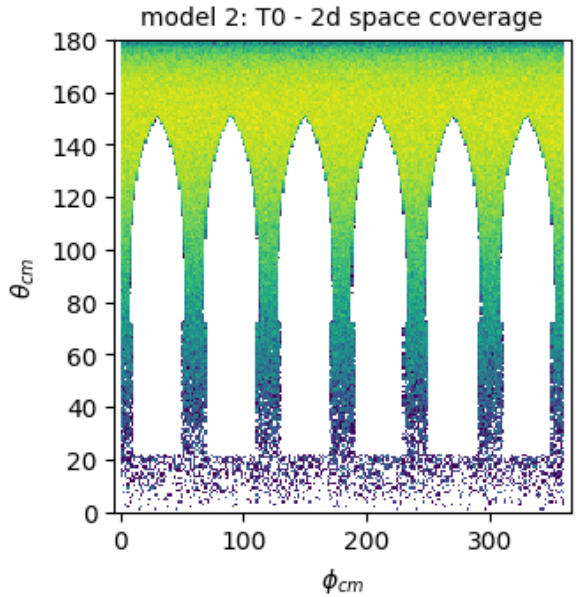


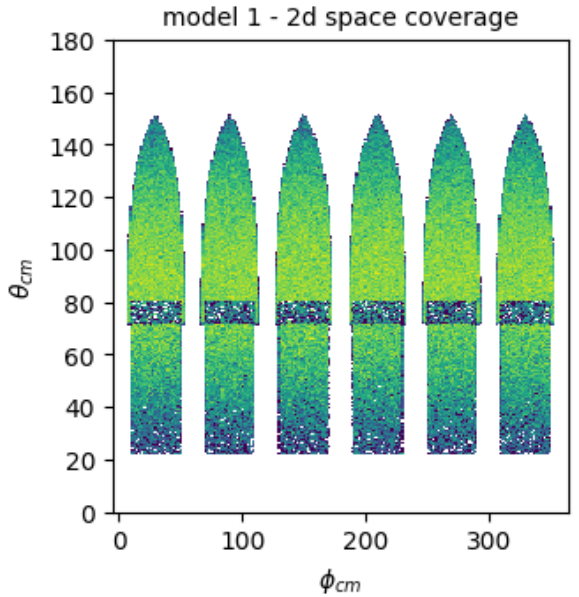
Weighting the number of events in $T^{0^{D1}}$ as:

$$\#T^{0^{D1}} = \frac{\#T^{meas^{M1}}}{\#T^{meas^{PS}}} * \#T^{0^{PS}}$$

Dataset 2:

$$T^{D2} = (T^{0^{D2}}, T^{meas^{M1}})$$





Weighting the number of events in $T^{0^{D2}}$ as:

$$\#T^{0^{D2}} = \frac{\#T^{meas^{M1}}}{\#T^{meas^{M2}}} * \#T^{0^{M2}}$$

Two different approaches to take care of the different topologies

Strategy 1: Train the GAN on mixed events

- Define a probability for each event to belong to a given topology
- Each event is now defined by $(\theta_{cm}, \phi_{cm}, P_{T0}, P_{T1}, P_{T2}, P_{T3})$
- Train a single GAN to take care of all the detector effects at once

- The architecture is harder to train
- The training dataset is computationally expensive to build
- The mixed events are already included in the training dataset

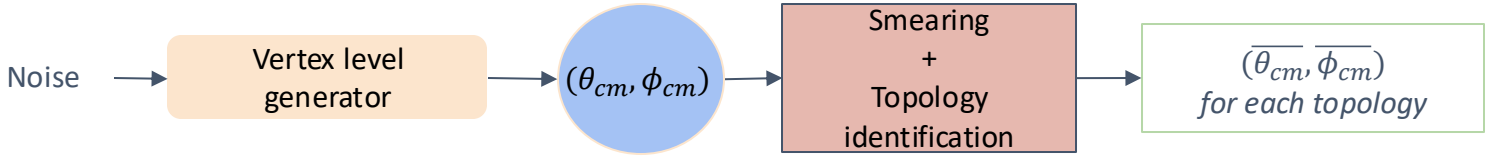
Strategy 2: Train the GAN architecture on distinct topologies

- Train the architecture on an independent topologies
- Each events is still defined by (θ_{cm}, ϕ_{cm})
- Implement the efficiency on each topology (mixed events) as a separate tool (through density ratio estimate, as Derek Glazier showed yesterday)

- The architecture is easier to train
- The efficiency calculations are built upon an already working model
- The efficiency is implemented as an external tool

Procedure overview

Regardless of the strategy chosen, once everything is trained, we should have the following workflow:



- **Data**
  - a) M1: Realistic π^0 photoproduction model to be used as a **proxy for the real data** in the measured region. We will have $T^{\text{meas}^{\text{M1}}} = (T^{1^{M1}}, T^{2^{M1}}, T^{3^{M1}})$
  - b) PS: The simplest we can build, with no dynamics, will serve as one of the models to fill out the unmeasured region of the phase space. $(T^{0^{PS}}, T^{1^{PS}}, T^{2^{PS}}, T^{3^{PS}})$
  - c) M2: Unphysical model with modified amplitude. This will serve as a second model to fill out the unmeasured space. $(T^{0^{M2}}, T^{1^{M2}}, T^{2^{M2}}, T^{3^{M2}})$

Test the procedure on different combinations of these models:

- Check that the training procedure is robust enough using different datasets
- Cover the unmeasured region with different models

Strategy 1: Dataset preparation

- Define the dataset as the 4-momenta of the final state: $(\overrightarrow{P_{\pi}^{0}}, \overrightarrow{P_{prec}})$
- Apply the smearing function to each component
- Convert each event into its independent variables pair: $(\overline{\theta_{cm}}, \overline{\phi_{cm}})$
- Split the training dataset as 90% (1-topology events) and 10% (Mixed topology events)
- Concatenate to each event, the probability too belong to a given topology
- The mixed events will be assigned with a random probability P_i to belong to any topology

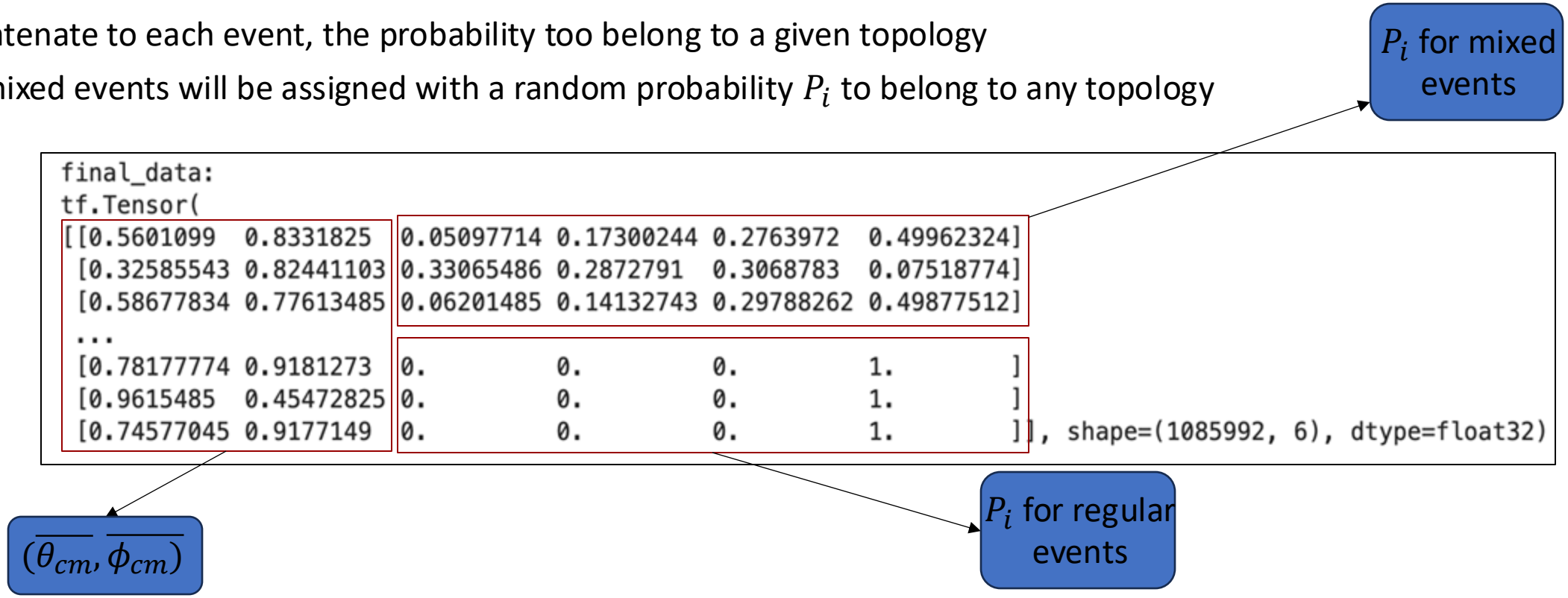
Strategy 1: Dataset preparation

- Define the dataset as the 4-momenta of the final state: $(\overrightarrow{P_{\pi}^{0}}, \overrightarrow{P_{prec}})$
- Apply the smearing function to each component
- Convert each event into its independent variables pair: $(\overline{\theta_{cm}}, \overline{\phi_{cm}})$
- Split the training dataset as 90% (1-topology events) and 10% (Mixed topology events)
- Concatenate to each event, the probability too belong to a given topology
- The mixed events will be assigned with a random probability P_i to belong to any topology

```
final_data:
tf.Tensor(
[[0.5601099  0.8331825  0.05097714 0.17300244 0.2763972  0.49962324]
 [0.32585543 0.82441103 0.33065486 0.2872791  0.3068783  0.07518774]
 [0.58677834 0.77613485 0.06201485 0.14132743 0.29788262 0.49877512]
 ...
 [0.78177774 0.9181273  0.         0.         0.         1.        ]
 [0.9615485  0.45472825 0.         0.         0.         1.        ]
 [0.74577045 0.9177149  0.         0.         0.         1.        ]], shape=(1085992, 6), dtype=float32)
```

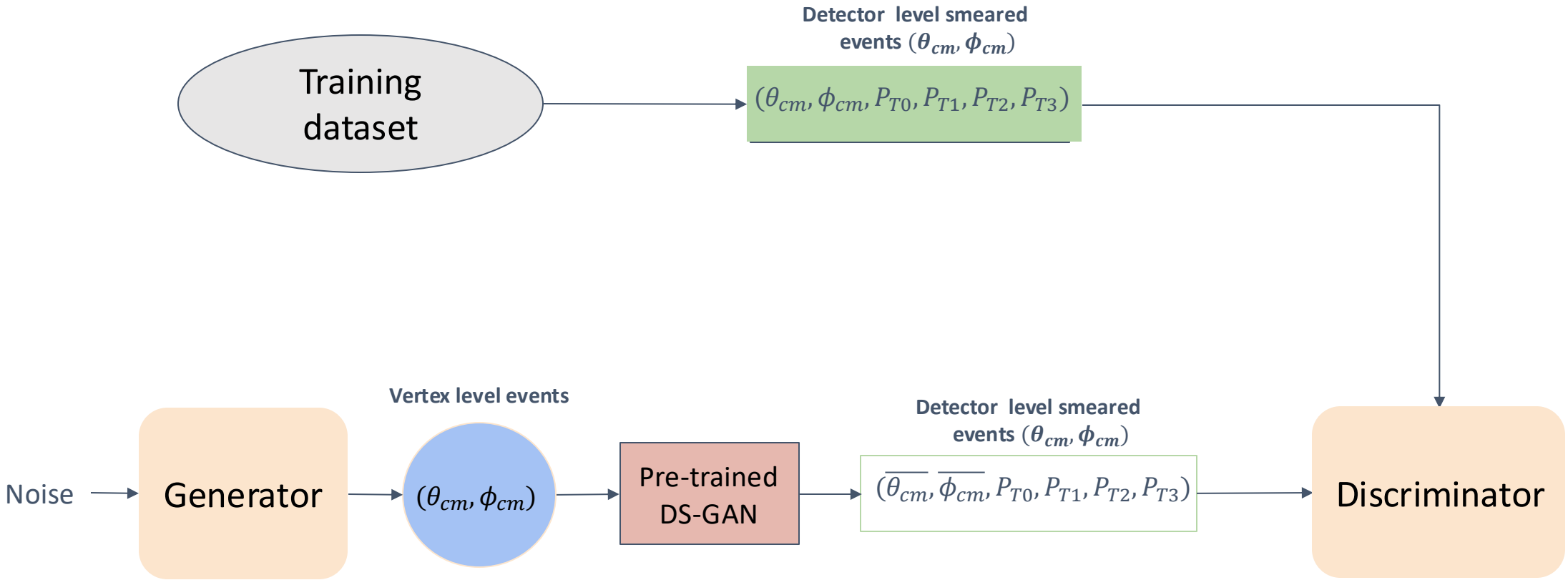
Strategy 1: Dataset preparation

- Define the dataset as the 4-momenta of the final state: $(\overrightarrow{P_{\pi}^{0}}, \overrightarrow{P_{prec}})$
- Apply the smearing function to each component
- Convert each event into its independent variables pair: $(\overline{\theta_{cm}}, \overline{\phi_{cm}})$
- Split the training dataset as 90% (1-topology events) and 10% (Mixed topology events)
- Concatenate to each event, the probability too belong to a given topology
- The mixed events will be assigned with a random probability P_i to belong to any topology



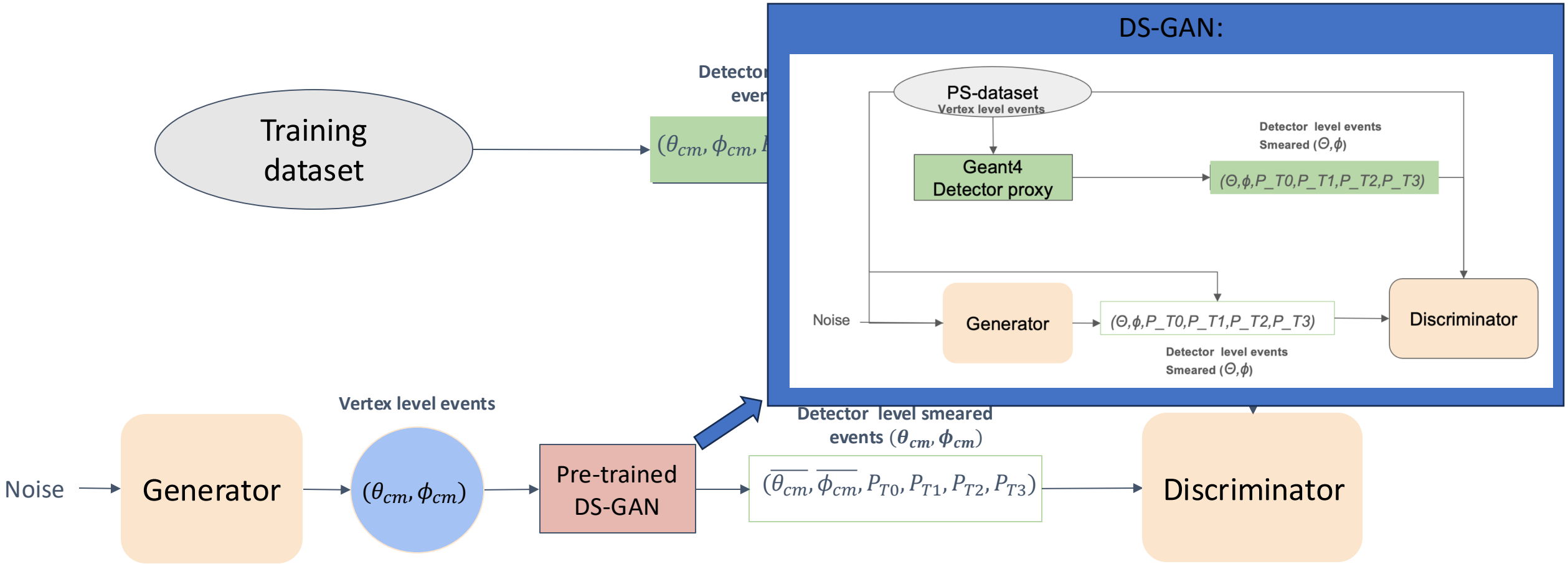
Strategy 1: architecture workflow

Outer GAN:

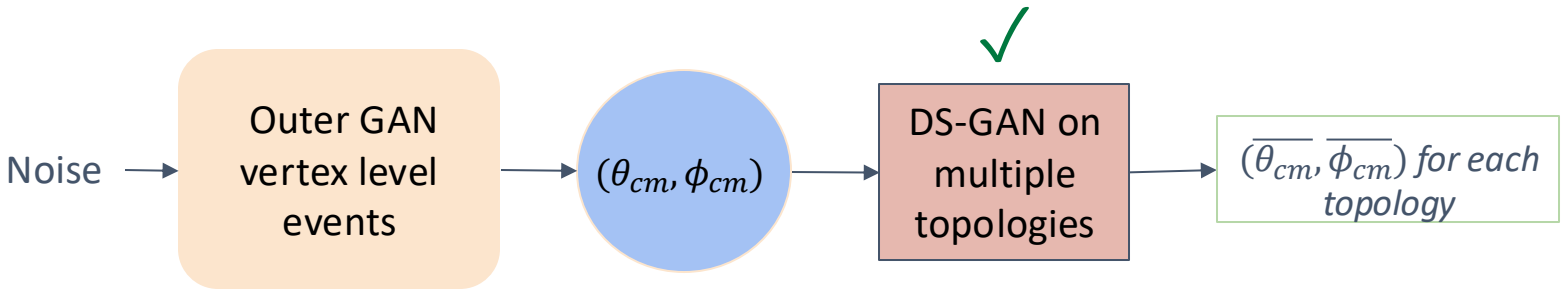


Strategy 1: architecture workflow

Outer GAN:



Strategy 1: Where are we?

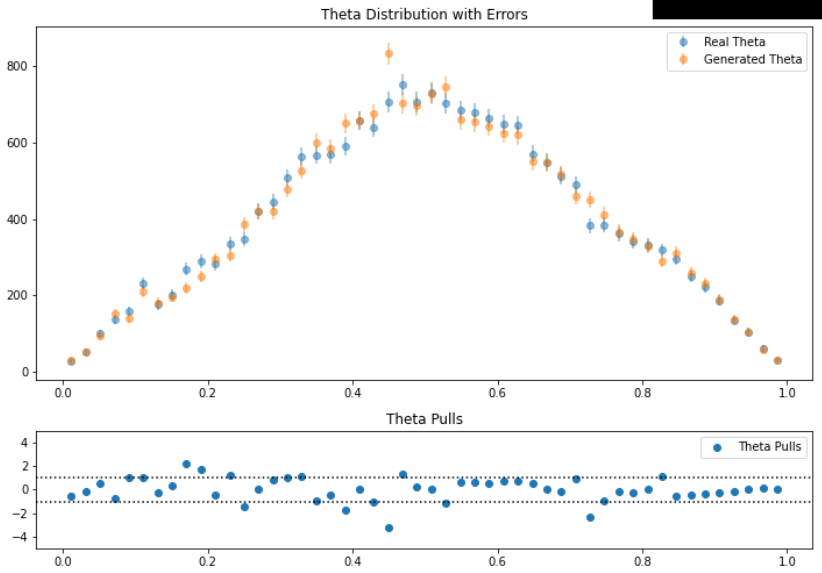


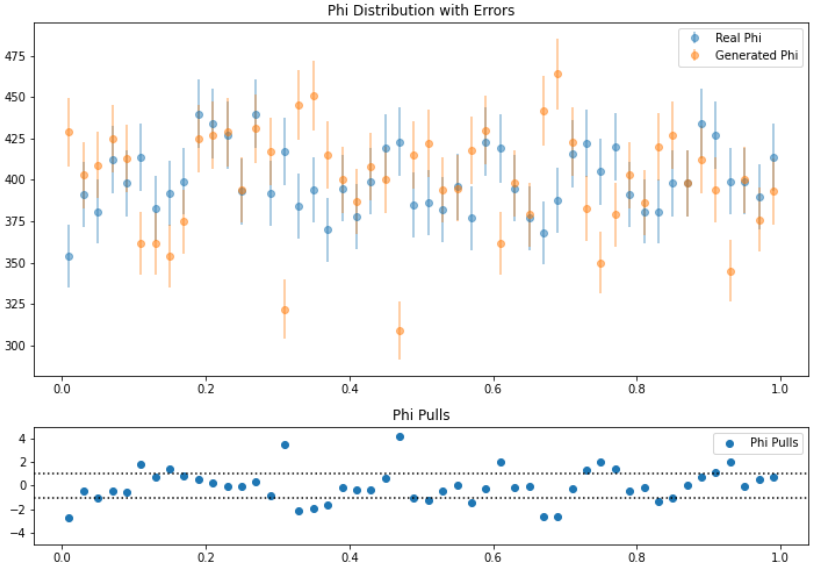
- The DS-GAN has been trained on Dataset 1 where 10% of the available dataset has been randomly chosen as “mixed events”. Random probabilities of these events belonging to a given topology have been assigned

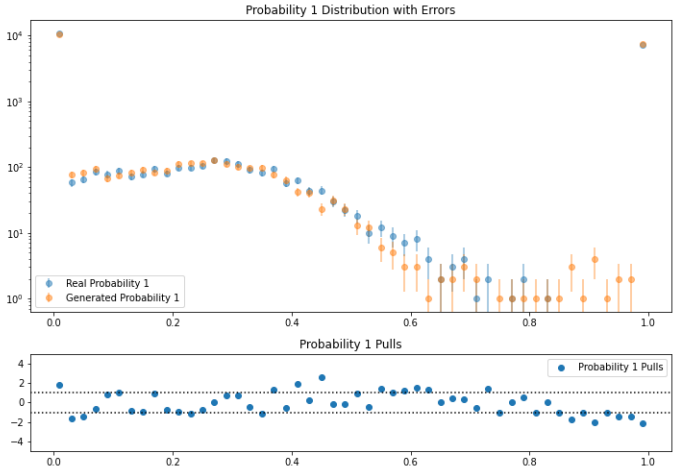
Work to do:

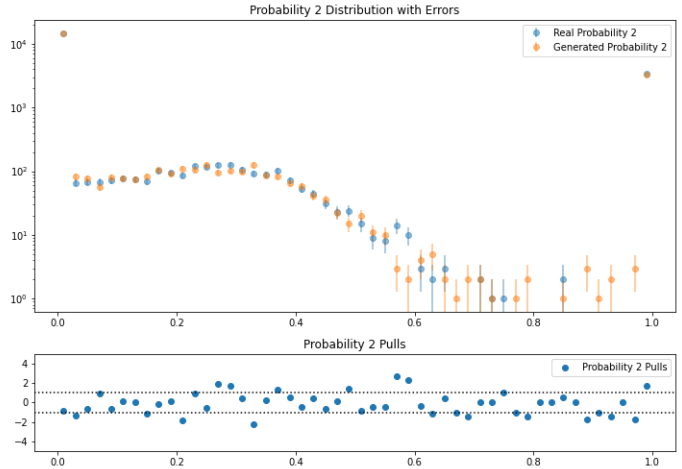
- Define realistic probabilities for the Inner GAN mixed events
- Train multiple DS-GANs to reduce the systematic error
- Train the Outer GAN to obtain vertex level events
- Refine both DS-GAN and outer GAN prediction through bootstrapping technique

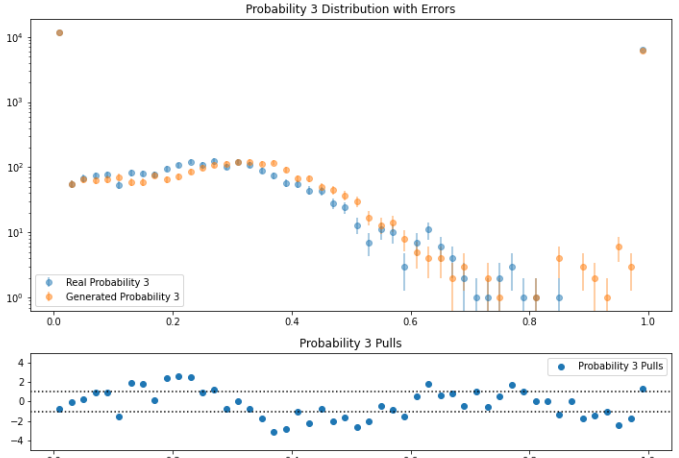
Strategy 1: DS-GAN preliminary results

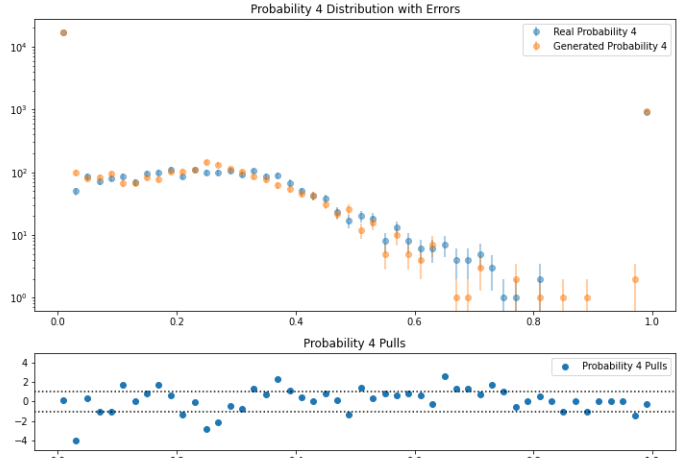












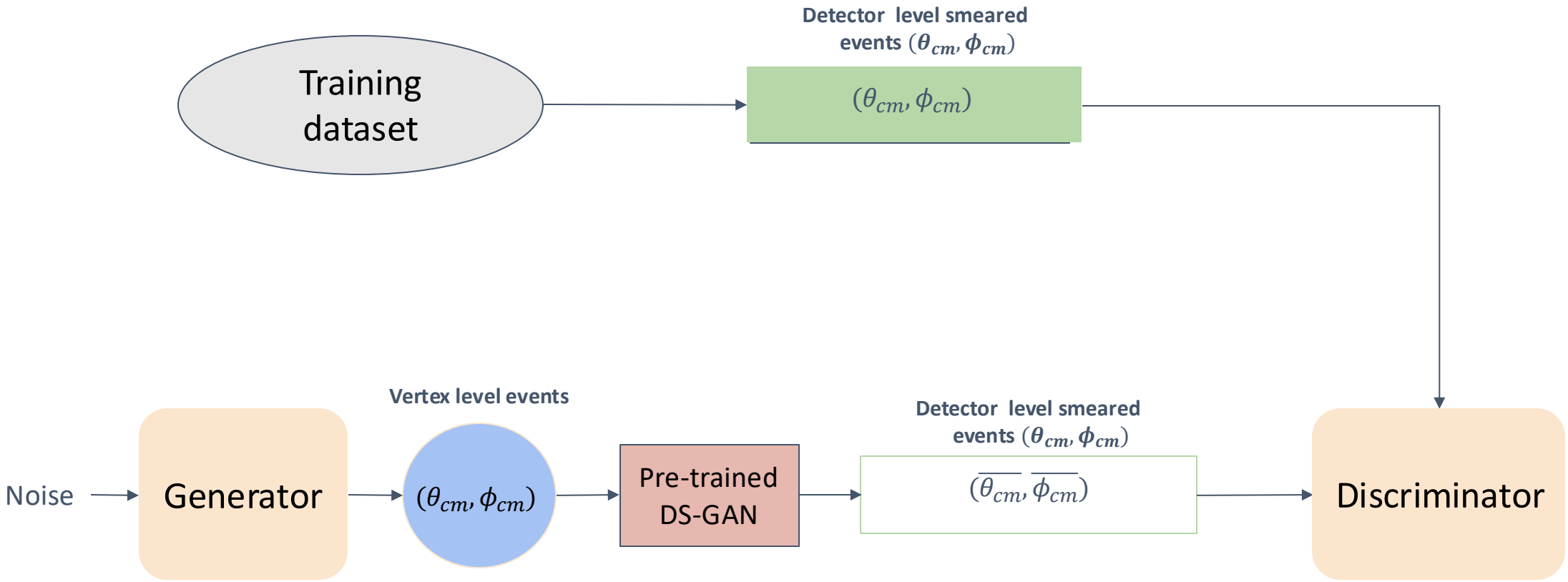
Strategy 2: Dataset preparation

- Define the dataset as the 4-momenta of the final state: $(\overrightarrow{P_{\pi}^{0}}, \overrightarrow{P_{prec}})$
- Apply the smearing function to each component
- Convert each event into its independent variables pair: $(\overline{\theta_{cm}}, \overline{\phi_{cm}})$
- The training will be performed on the full dataset and the detection efficiency of each particle (hence the different topologies) will be defined through the density ratio technique on the trained outer GAN:

  - ❑ Define a binary classifier that mimics the acceptance for each particle.
  - ❑ The output of the classifier $c(x) \in (0,1)$ can be related to the efficiency through $\frac{\mathrm{P_a(x)}}{\mathrm{P_{g(x)}}} \sim \frac{c(x)}{1-c(x)}$
  - ❑ The efficiency for each topology will then be: $\vec{\epsilon}(\theta, \phi) = \frac{\overrightarrow{P_a}}{P_g} = \frac{1}{P_g}(P_a^0, P_a^1, P_a^2, P_a^3)$
  - ❑ The number of accepted events for each topology will be $\overrightarrow{N_a}(\theta,\ \phi) = N_g(\theta,\ \phi) \times \vec{\epsilon}(\theta, \phi)$

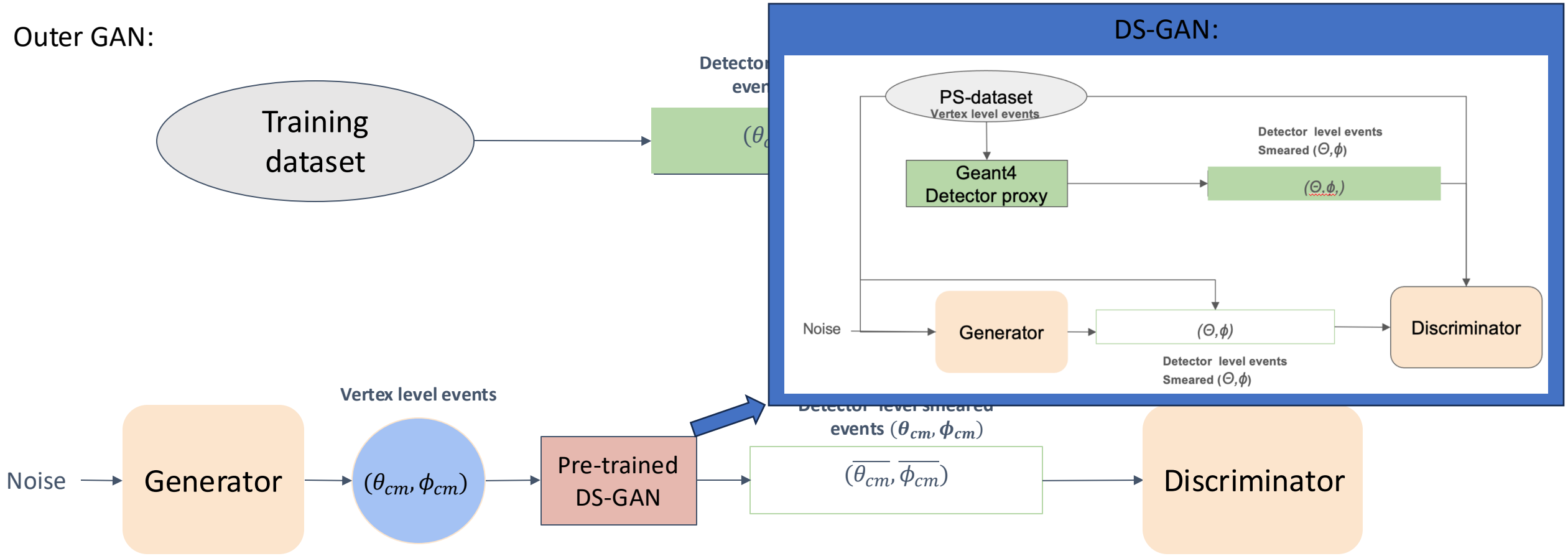
Strategy 2: Outer GAN

Outer GAN:



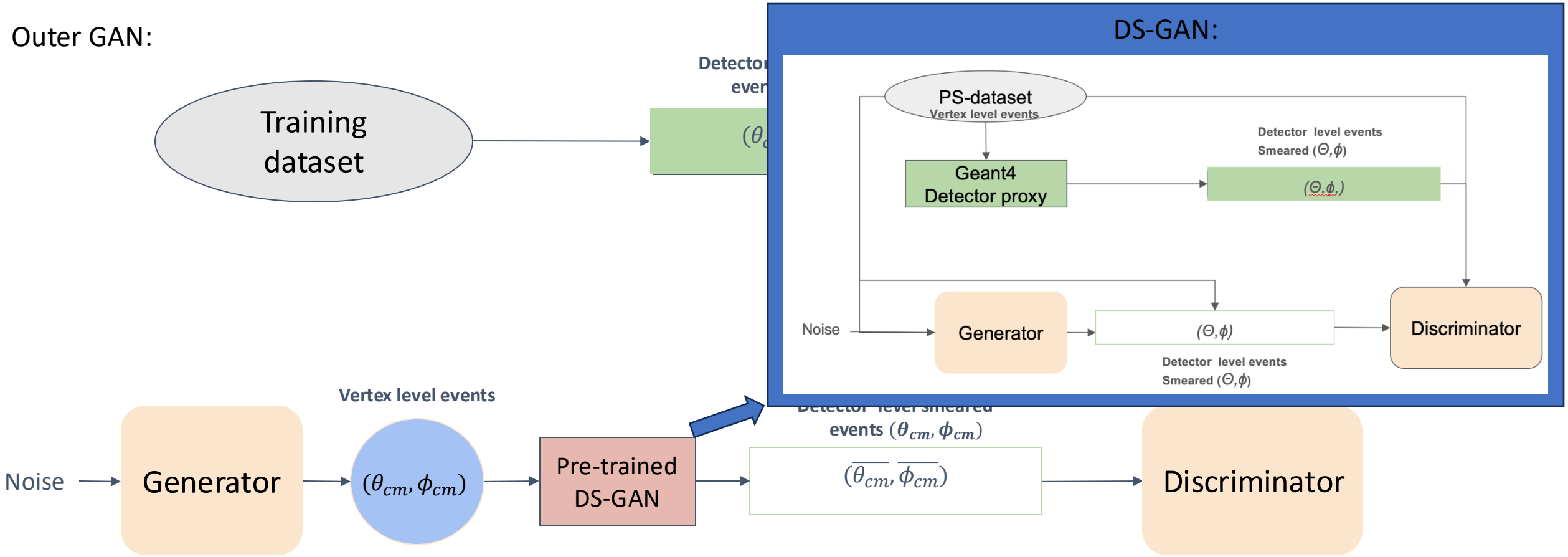
Strategy 2: Outer GAN

Outer GAN:



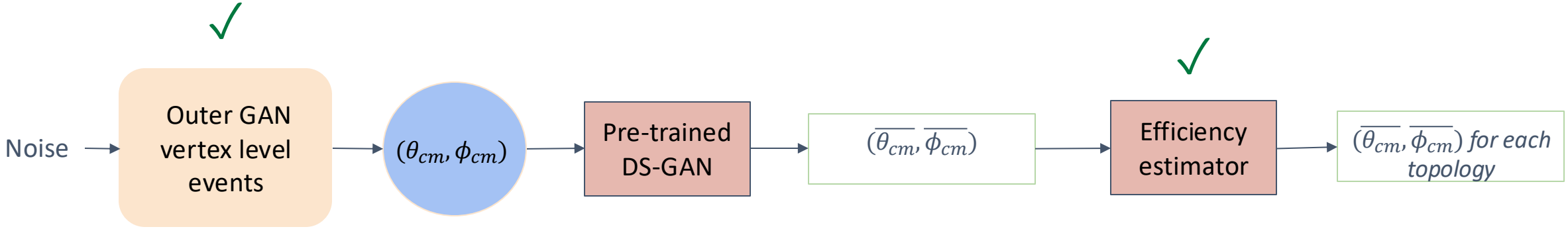
Strategy 2: Outer GAN

Outer GAN:



- Efficiency implementation as a density ratio on the trained outer GAN

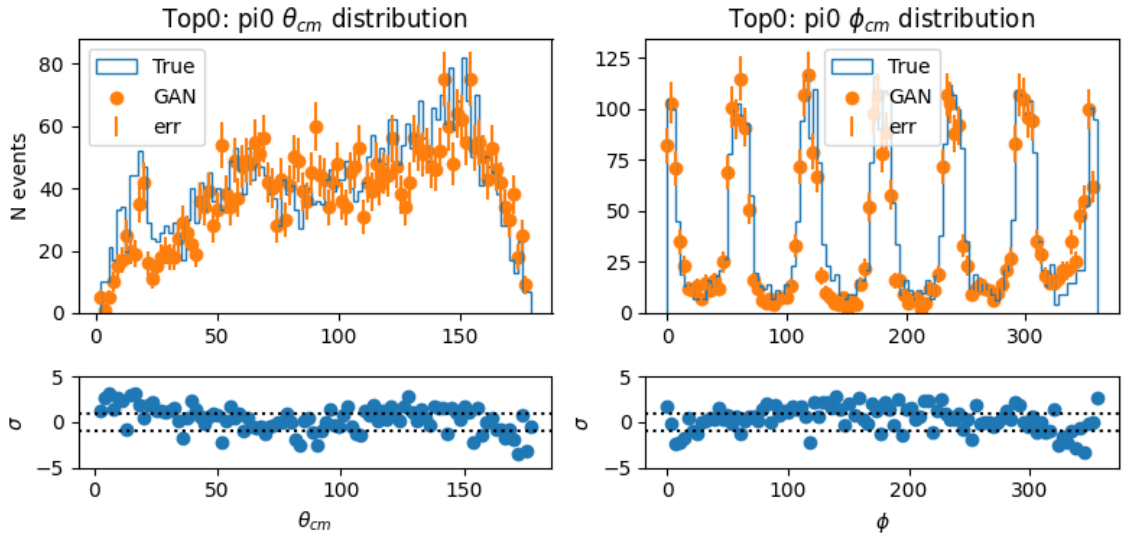
Strategy 2: Where are we?

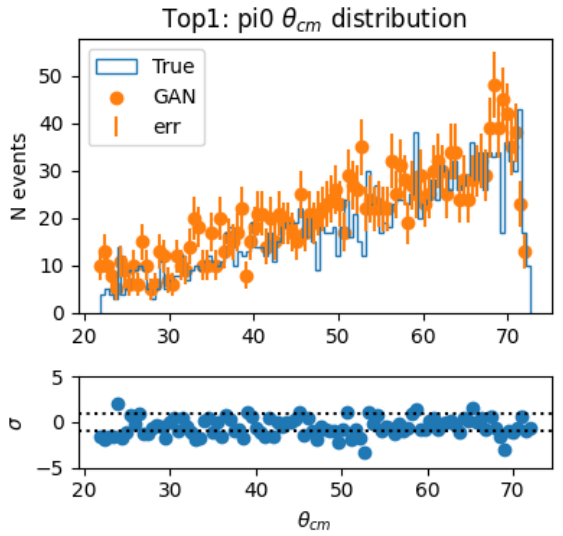


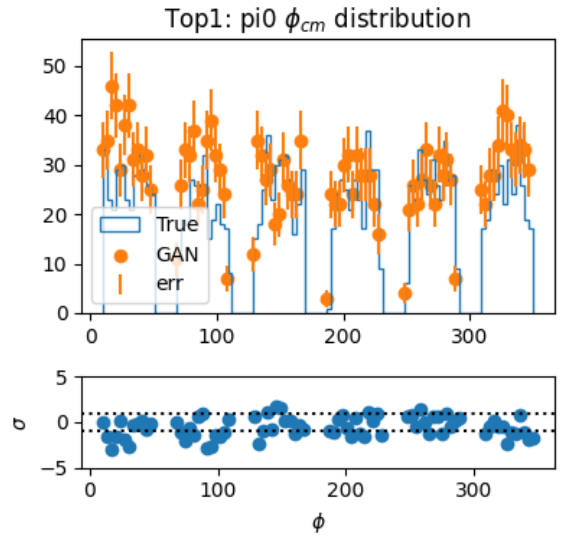
Work to do:

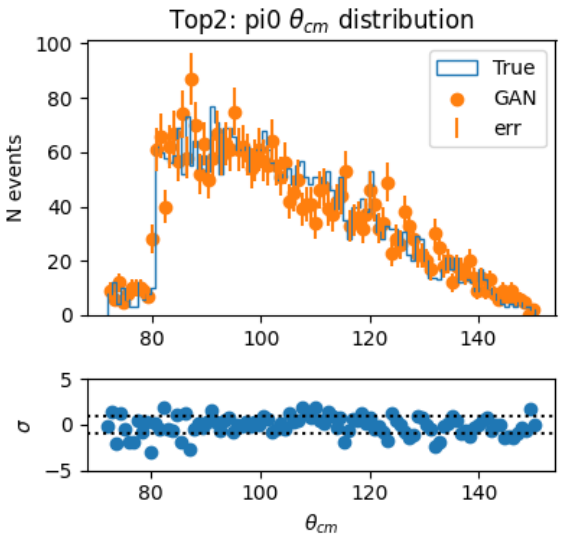
- Train DS-GAN to obtain the correct smearing for each topology
- Train Outer GAN on after the correct DS-GAN implementation
- Refine both DS-GAN and outer GAN prediction through bootstrapping technique

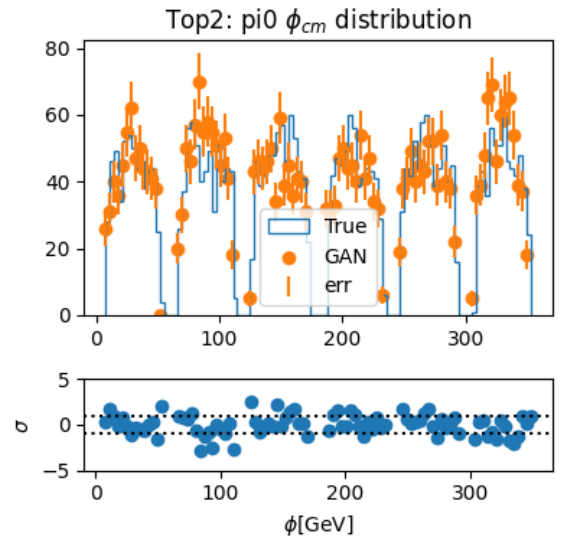
Strategy 2: Outer GAN preliminary results

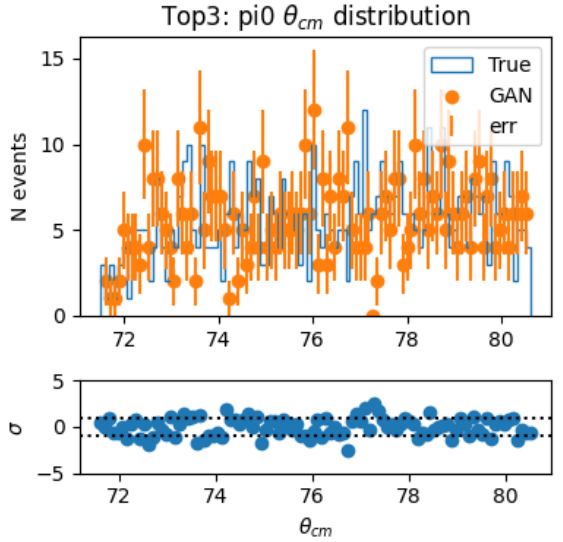


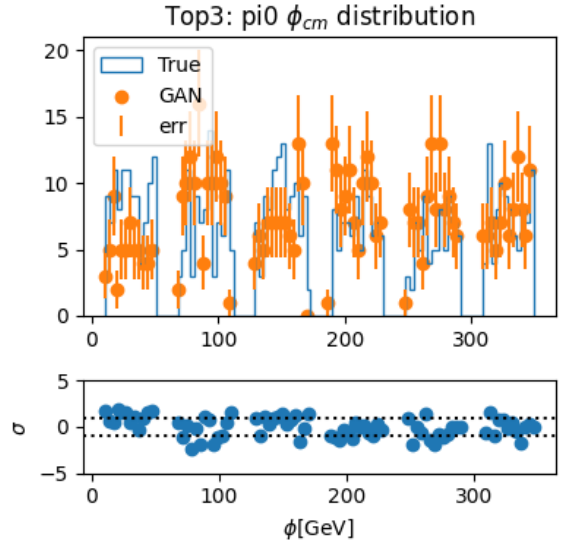






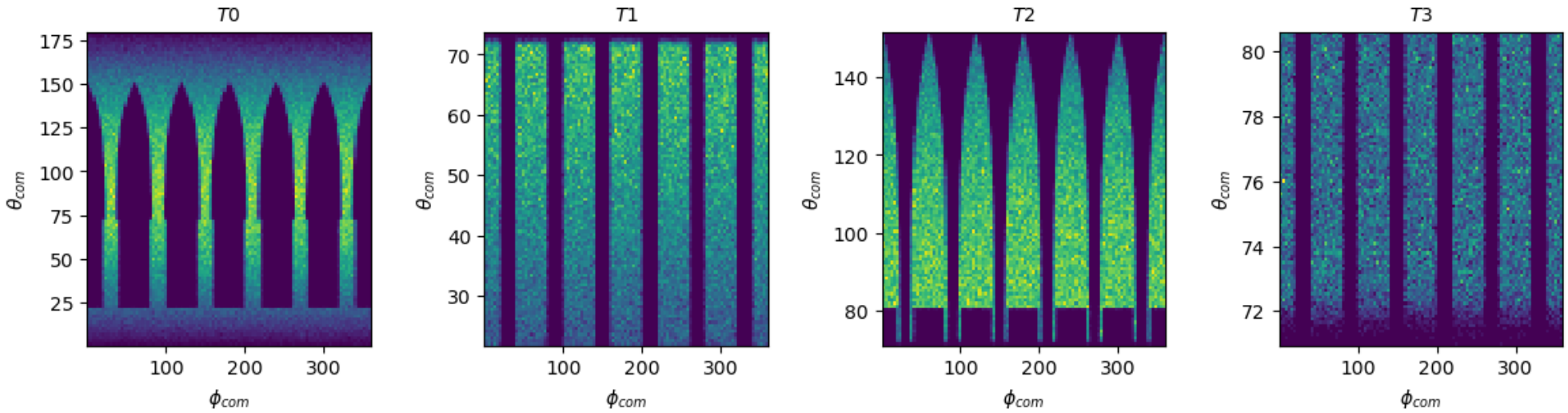




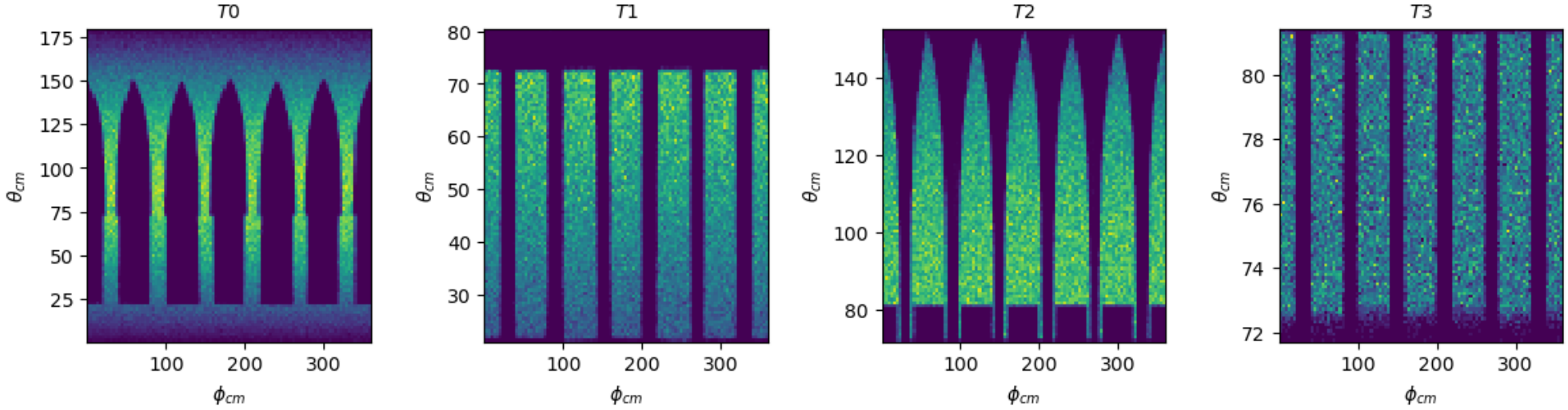


Strategy 2: Efficiency estimation

Truth:



Generated:



Conclusions and outlook

Conclusions:

- We developed two different methodologies to retrieve the full vertex-level phase space, extending the measured region as far as we can

- Provided a tool to take into account for all the detector effects (smearing, acceptance, detector inefficiencies) in the most accurate way and unfold them to recover the vertex level distributions

Work to do:

- Complete the missing pieces in each strategy
- Confront the two metodologies
- Repeat the training with the second model to guarantee robustness and model independence

Thank you for your attention!