# ICFM tutorial

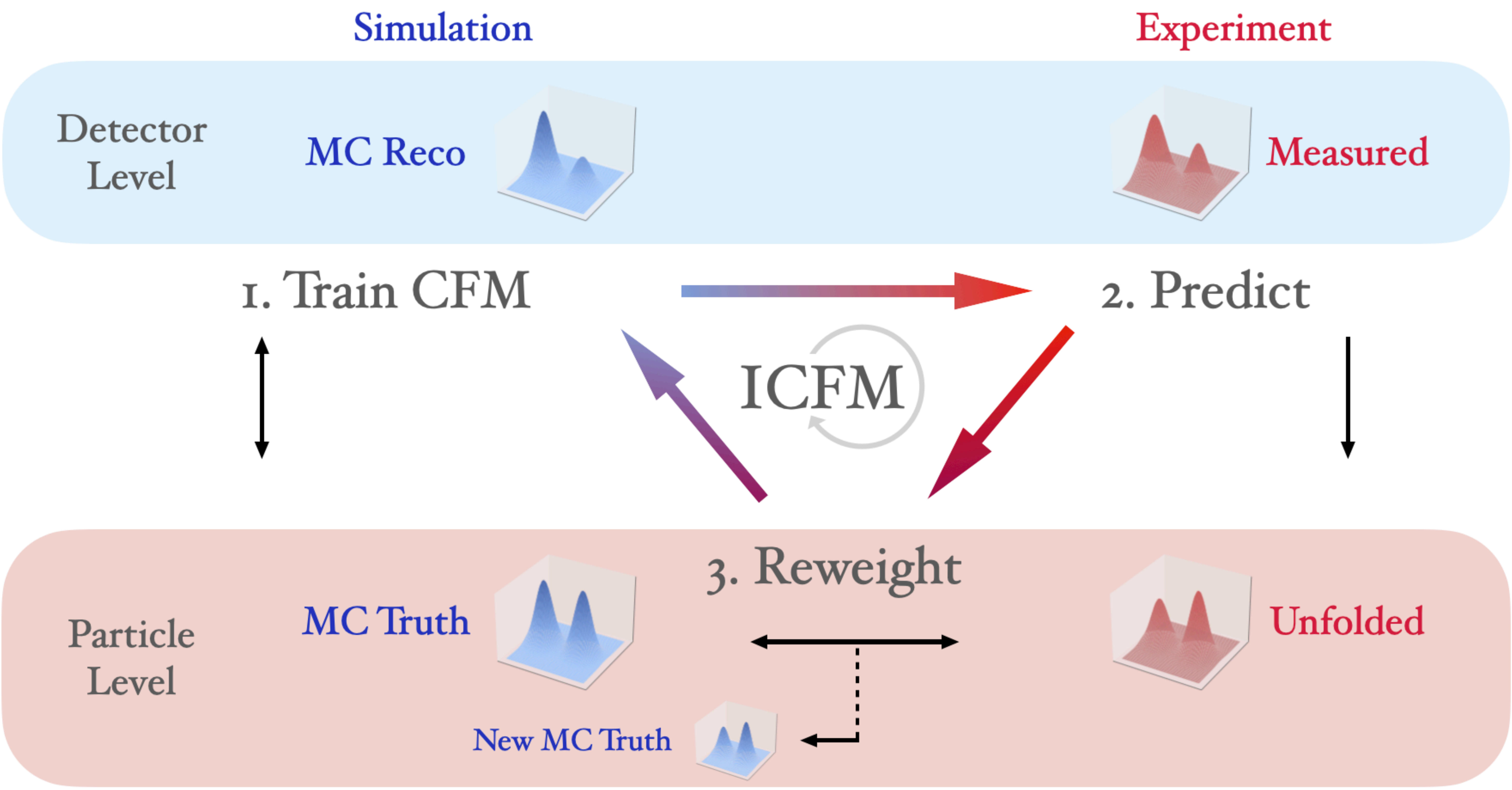
Javier Mariño Villadamigo

PHYSTAT Conference on Unfolding
10-13 June 2024

Institut für Theoretische Physik - University of Heidelberg

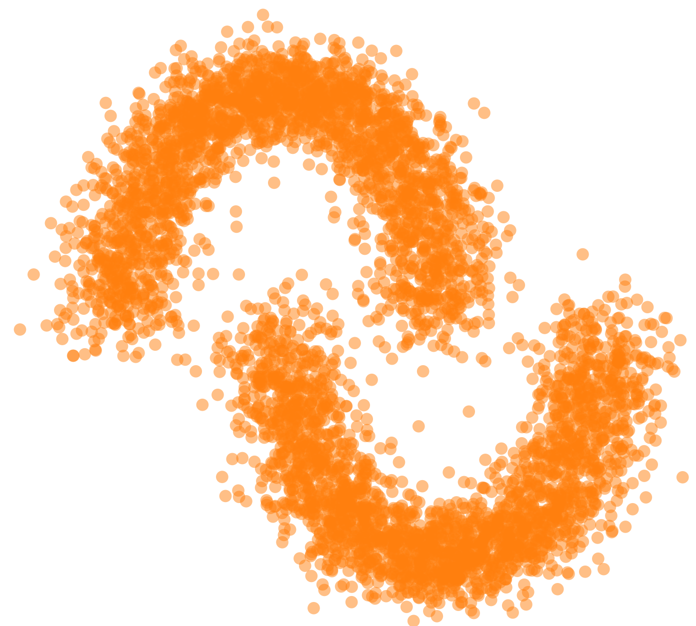
SPONSORED BY THE
Federal Ministry of Education and Research

# Iterative algorithm

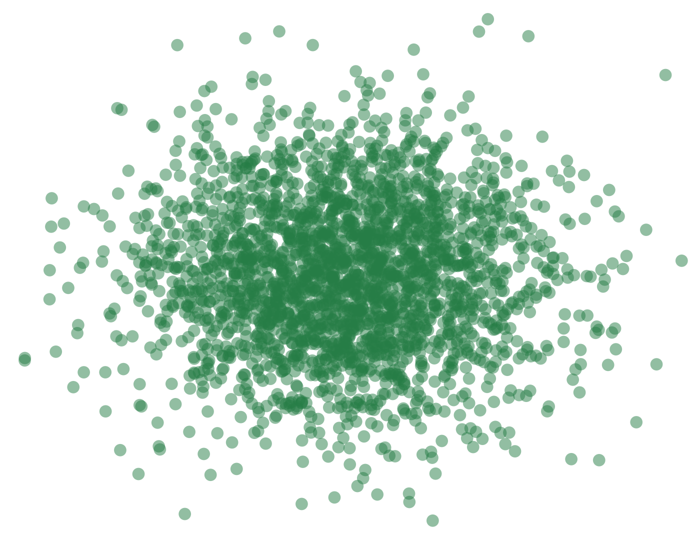



- **Step 1**: Train CFM to unfold MC.
- **Step 2**: Apply to Data
- **Step 3**: Train classifier to reweight Data Unfolded to match MC Truth
- **Step 4**: Use the reweighted Data Unfolded as the new MC Truth
- Repeat!

# Conditional Flow Matching (CFM)



$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = v_\theta(x(t), t \,|\, x_{\mathrm{reco}})$$



$x_0 \sim p_{\mathrm{model}}(x_{\mathrm{hard}} \,|\, x_{\mathrm{reco}})$

$\epsilon = z \sim p_{\mathrm{latent}}(z)$

- Connect $x_0$ and $\epsilon$ with a linear trajectory: $x(t) = (1-t)x_0 + t\epsilon$
- The NN is regressed to predict the velocity field: $v_\theta(x(t), t \,|\, x_{\mathrm{reco}}) \approx \dfrac{\mathrm{dx(t)}}{\mathrm{dt}} = \epsilon - x_0$
- For sampling, solve ODE starting from $\epsilon$: $x_0 = \epsilon + \int_1^0 v_\theta(x(t), t \,|\, x_{\mathrm{reco}})\mathrm{dt}$
- Loss: $\mathscr{L}_{\mathrm{CFM}} = \left\langle [v_\theta((1-t)x_0 + t\epsilon,\ t,\ x_{\mathrm{reco}}) - (\epsilon - x_0)]^2 \right\rangle_{t\sim\mathscr{U}([0,1]),\ (x_0, x_{\mathrm{reco}})\sim p(x_{\mathrm{hard}}, x_{\mathrm{reco}}),\ \epsilon\sim\mathscr{N}(0,1)}$

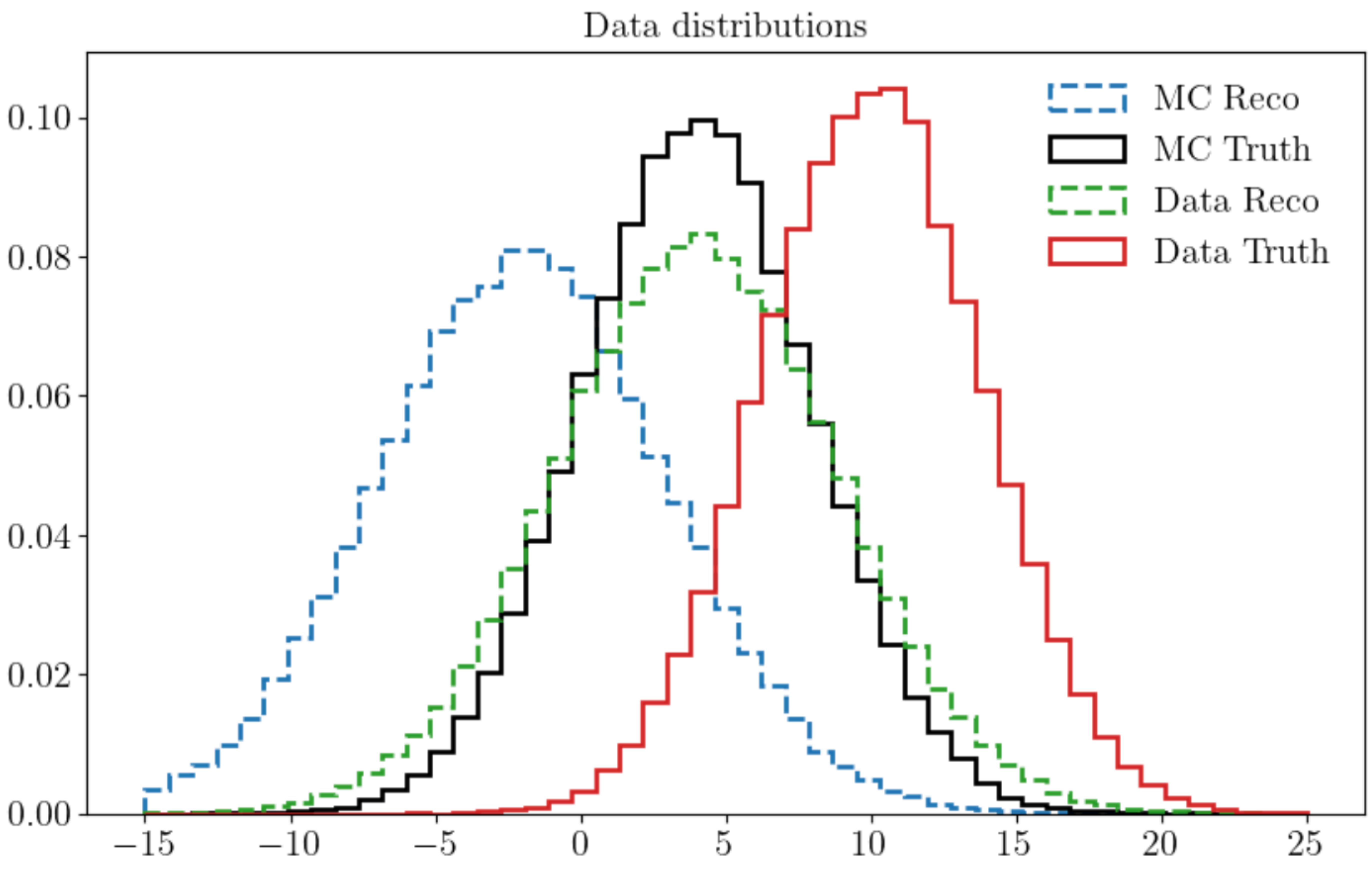
# ICFM: inputs

```python
class Iterative_CFM(nn.Module):
    def __init__(self,
                 dims_truth,            # number of truth features
                 dims_reco,             # number of reco features
                 cfm_params,            # parameters dict for the CFM
                 classifier_params,     # parameters dict for the classifier
                 MC_reco,               # MC reco events
                 MC_truth,              # MC truth events
                 data_reco,             # Data reco events
                 data_truth,            # Data truth events
                 bins,                  # bins for each observable for plotting
                 mu_unf=None,           # initial mu of the prior (for toy data)
                 sigma_unf=None,        # initial sigma of the prior (for toy data)
                 data_truth_mu=None,    # mean of the data truth (for toy data)
                 mu_smear=None,         # mean of the smearing (for toy data)
                 sigma_smear=None       # sigma of the smearing (for toy data)
                 ):
```

- The methods inside this class are varied and lengthy: feel free to ask me about specific doubts!

- <u>dims_truth/reco</u>: dimensions you want to simultaneously unfold.
- <u>cfm/classifier_params</u>: dictionary with epochs, number and width of MLP layers, etc
- <u>MC/data_reco/truth</u>: data inputs. The algorithm expects them to be [N_events, dims]
- The last 5 inputs are used for the computation of the <u>analytical posterior</u> at each iteration, only possible for the toy example

# ICFM: toy example




- MC Truth:
  $G(x;\ \mu_{\text{MC,t}} = 4,\ \sigma_{\text{MC,t}} = 4)$
- Data Truth:
  $G(x;\ \mu_{\text{Data,t}} = 10,\ \sigma_{\text{Data,t}} = 3.8)$
- Detector effects:
  $G(x;\ \mu_{\text{smear}} = -6,\ \sigma_{\text{smear}} = 3)$

# ICFM: analytical posterior

At iteration 0, one can compare the unfolded **toy example** to the analytical posterior, which is a gaussian whose mean and variance can be calculated as:

$$\mu_{u,0} = \frac{(\mu_{\mathrm{Data,r}} - \mu_{\mathrm{smear}})\,\sigma^2_{\mathrm{MC,t}} + \mu_{\mathrm{MC,t}}\,\sigma^2_{\mathrm{smear}}}{\sigma^2_{\mathrm{MC,t}} + \sigma^2_{\mathrm{smear}}},$$

$$\sigma_{u,0} = \frac{\sigma_{\mathrm{MC,t}}\sqrt{\sigma^2_{\mathrm{MC,t}}\,\sigma^2_{\mathrm{Data,r}} + \sigma^2_{\mathrm{MC,t}}\,\sigma^2_{\mathrm{smear}} + \sigma^4_{\mathrm{smear}}}}{\sigma^2_{\mathrm{MC,t}} + \sigma^2_{\mathrm{smear}}}.$$

[2212.08674]

For further iterations, one can use the same formula by substituting the (MC, t)- mean and variance by the previously obtained unfolded mean and variance (which ideally represent the new MC prior)

# ICFM: single event unfolding

```
def plot_single_event_unfolding(event_idx, n_unfoldings = 128):
```

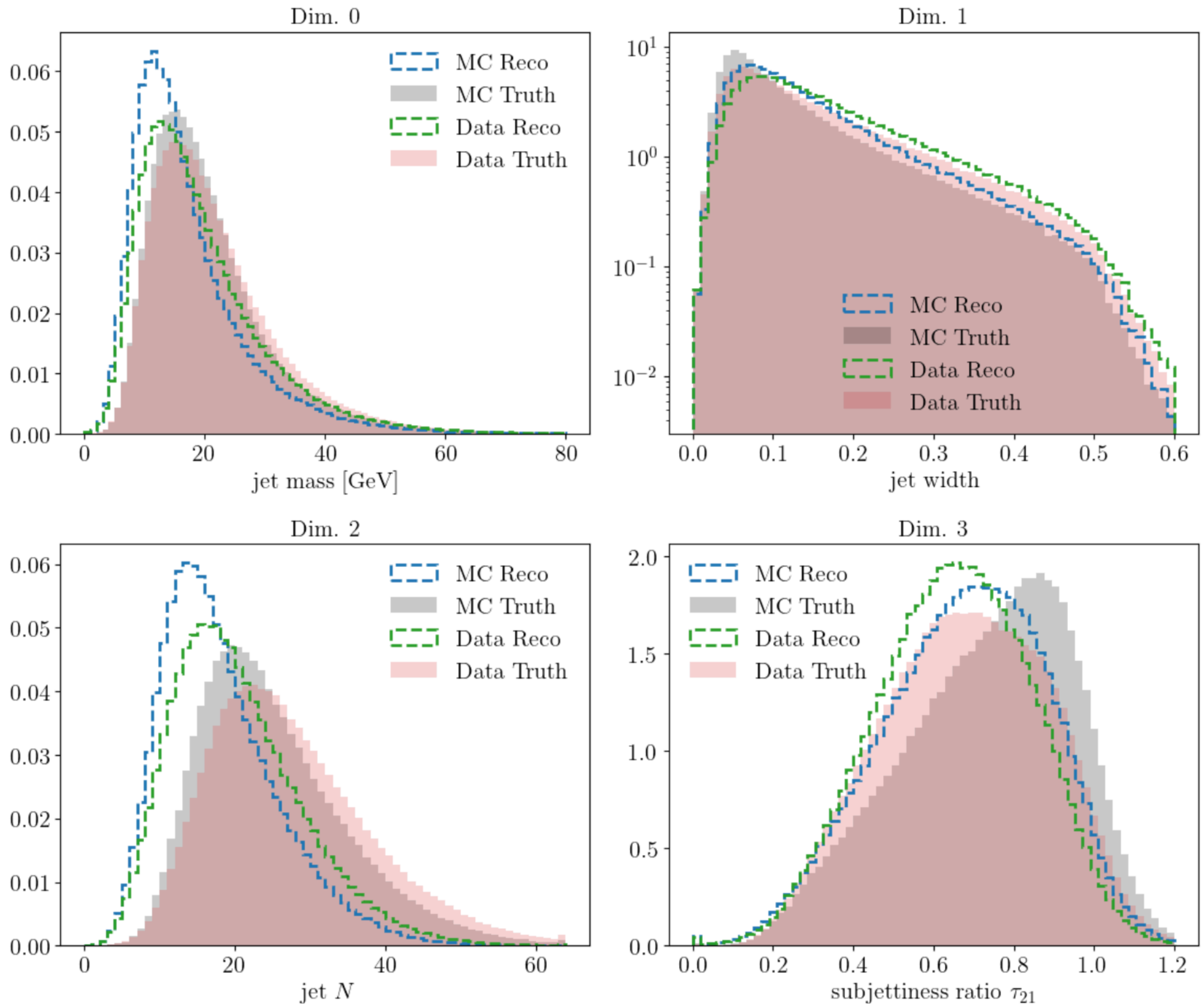
Something that one can do with generative models is unfolding the same event multiple times, which effectively samples the learned posterior. For a gaussian toy model, this can be compared to a gaussian analytical posterior with mean and variance

$$\mu_{\text{single}} = \frac{\sigma^2_{\text{smear}}\,\mu_{\text{MC,t}} - \sigma^2_{\text{MC,t}}(\mu_{\text{smear}} - y_m)}{\sigma^2_{\text{smear}} + \sigma^2_{\text{MC,t}}}, \qquad \sigma^2_{\text{single}} = \frac{\sigma^2_{\text{smear}}\sigma^2_{\text{MC,t}}}{\sigma^2_{\text{smear}} + \sigma^2_{\text{MC,t}}}.$$

[2212.08674]

where $y_m$ is the measured data event at reco level, $\mu_{\text{smear}}$, $\sigma_{\text{smear}}$ are the gaussian convolution parameters that describe the detector effects; and $\mu_{\text{MC, t}}$, $\sigma_{\text{MC, t}}$ represent the prior.

# ICFM: Pythia/Herwig example




$Z\,(\,p_T > 200$ GeV ) + jets. We use a subset of leading jet observables:

- Jet mass $m$
- Jet width $w$
- Jet constituents multiplicity $N$
- N-subjettiness ratio $\tau_{21} = \tau_2^{(\beta=1)}/\tau_1^{(\beta=1)}$