# Optimal transport solutions for pileup mitigation at hadron colliders

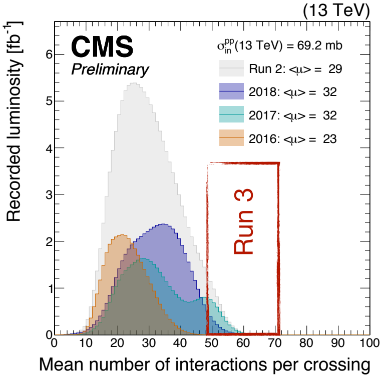
L. Gouskos [1] **F. Iemmi** [2] S. Liechti [4] B. Maier [1]
V. Mikuni [3] H. Qu [1]

[1]European Organization for Nuclear Research (CERN), Geneva
[2]Institute of High Energy Physics (IHEP), Beijing
[3]National Energy Research Scientific Computing Center (NERSC), Berkeley
[4]University of Zurich (UZH), Zurich

IHEP Deep learning seminar CN

based on arXiv:2211.02029

# PU mitigation at hadron colliders




- **Pileup**: additional pp collisions superimposing to main collision
- **PU** has **increased** in Run3 ($\langle nPU \rangle = 50$) and will increase in HL-LHC ($\langle nPU \rangle = 140$)
- Will severely **degrade quality of observables** (jet multiplicity, jet substructure, ...) if not properly treated
- PU mitigation is **crucial at hadron colliders**
- **Easy task for charged** particles: use tracking information to disentangle particles
- **Very challenging for neutral** particles

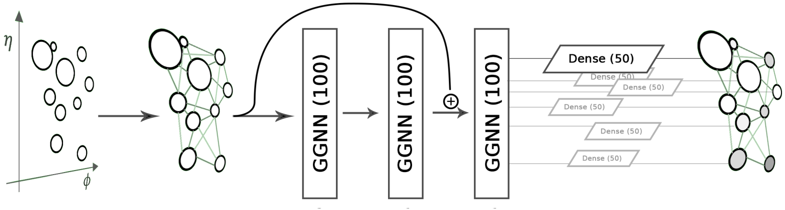
# State-of-the-art at CMS: PUPPI [1407.6013]

- Starting from Run3, **default PU mitigation technique** in CMS **is PUPPI**
- **Rule-based** algorithm
- Calculates a weight $w \in [0, 1]$ for each particle in the event
  - Encodes the probability for a particle to be LV or not
  - Weight used to **reweight the particle 4-momentum before jet clustering**
- For charged: use tracking information and assign 0 or 1
- For neutrals: build $\alpha$ variable

$$\alpha_i = \log \sum_{j \neq i, \Delta R_{ij} < R_0} \left( \frac{p_{T,j}}{\Delta R_{ij}} \right)^2 \begin{cases} |\eta_i| < 2.5 & j \text{ are all charged particles from LV} \\ |\eta_i| > 2.5 & j \text{ are all kinds of particles} \end{cases}$$

- QCD is harder and more collimated than PU $\implies$ higher $\alpha$ than PU
- After some math and assumptions (details in backup) translate $\alpha_i$ into $w_i$

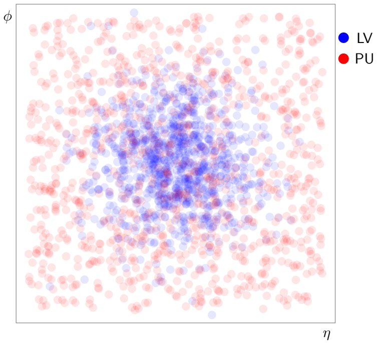
# ML for pileup mitigation




- Published literature demonstrates that **ML can drastically improve over current state-of-the-art** [1, 2, 3]
- In particular, **GNNs** proved to be **very effective**
  - Collect info about neighboring particles in a much more expressive way
- **General strategy**: train a supervised model in Delphes fast-simulation **using per-particle truth labels**

# ML for pileup mitigation

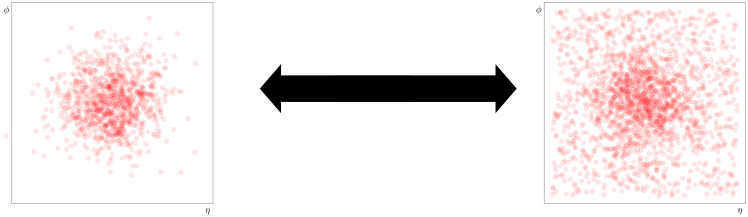
- **Critical issue**: per-particle lables are not available in Geant4-based full simulations
  - Previous approaches can't be ported to experiments such as ATLAS and CMS
- Recently proposed to train on charged and infer on neutrals [1]
  - Can be done in ATLAS/CMS using tracker
  - Relies on extrapolations
  - Charged → neutrals; central → forward
- We **developed a PU mitigation strategy that does not rely on per-particle truth labels or extrapolations**




**Not available in full-sim!**

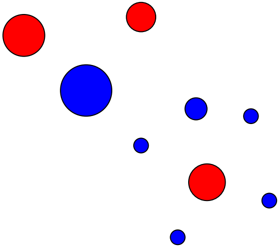
# A novel approach to PU mitigation

- Per-particle truth labels are not available in simulations at hadron colliders
- **Our approach**: simulate **identical** proton-proton **collisions in two scenarios**
  1. Only the hard interaction is simulated: **no-PU sample** ($\mathbf{X}_{\text{no-PU}}$)
  2. Pileup is superimposed to the hard interaction: **PU sample** ($\mathbf{X}_{\text{PU}}$)
- Train network to **learn differences between the two samples**
- Network choice: Attention-Based Cloud Network: **ABCNet** [1]



# How to learn: OT concepts for a loss function

- **Optimal transport (OT)** can measure the "distance" between probability distributions



- **Network output**: per-particle weights $\omega$, à-la-PUPPI
- Output weights aim at removing PU (give $\approx 0$ to PU and $\approx 1$ to LV)
- During training, weight $\mathbf{X}_{\text{PU}}$ by the weights $\boldsymbol{\omega}$
- Tweak weights to minimize the distance between $\mathbf{X}_{\text{no-PU}}$ and $\boldsymbol{\omega} \cdot \mathbf{X}_{\text{PU}}$
- Use Sliced Wasserstein Distance (SWD) as an OT-inspired loss function for the network
- **No need for per-particle labels in this setup**

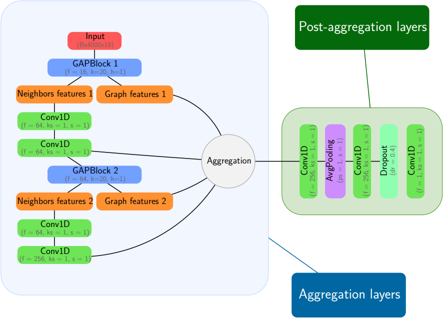
# Loss function

- SWD focuses on the optimal matching between individual particles in no-PU and PU samples
  - No guarantee that energy is conserved between the two
- Add an **event-level MET constraint** term to the loss
  - Enforce energies in no-PU and PU events to be similar
- Final loss function:

$$\mathcal{OT} = \mathrm{SWD}(\boldsymbol{\omega} \cdot \mathbf{X}_{\mathrm{PU}}, \mathbf{X}_{\text{no-PU}}) + \lambda \times \mathrm{MSE}\left(\mathrm{MET}(\boldsymbol{\omega} \cdot \mathbf{X}_{\mathrm{PU}}), \mathrm{MET}(\mathbf{X}_{\text{no-PU}})\right)$$

  where $\mathbf{X}_{\mathrm{PU}}$ = PU sample; $\mathbf{X}_{\text{no-PU}}$ = no-PU sample; MSE = mean squared error
- $\lambda$ gives the strength of the energy regularization; tested both $\lambda = 0$ and $\lambda = 10^{-3}$
- Call this **Training Optimal Transport with Attention Learning: TOTAL**

# The model



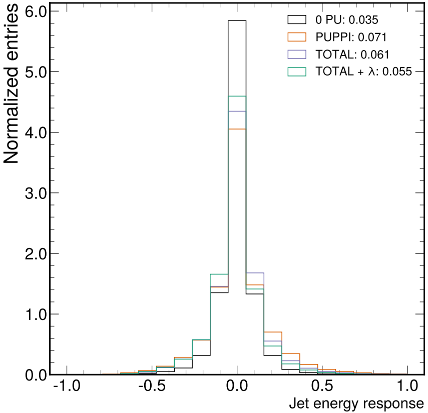

- **Compare** TOTAL **with PUPPI and no-PU** scenario
- **Reweight** each particle's 4-momentum by the network weight
- **Cluster** TOTAL jets and TOTAL MET

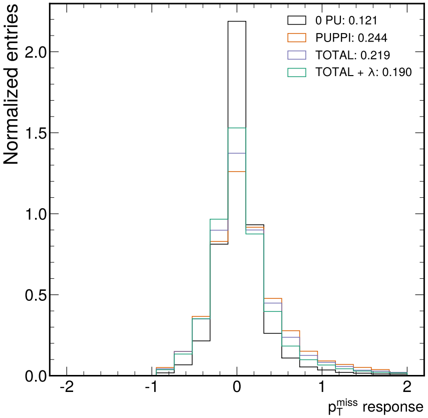
- We define the resolution as:

$$\delta = \frac{q_{75\%} - q_{25\%}}{2}$$

where $q_{X\%}$ is the X-th quantile of the considered response distribution

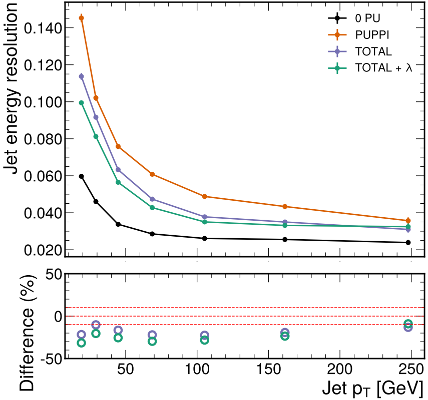
# Inclusive responses

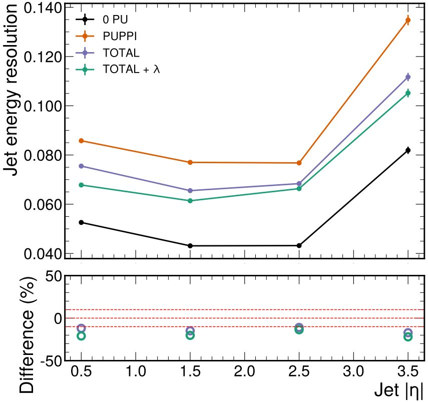



- Jet energy response in QCD (left) and MET response in $t\bar{t}$ (right)
- Improvement up to 23% and 22% respectively

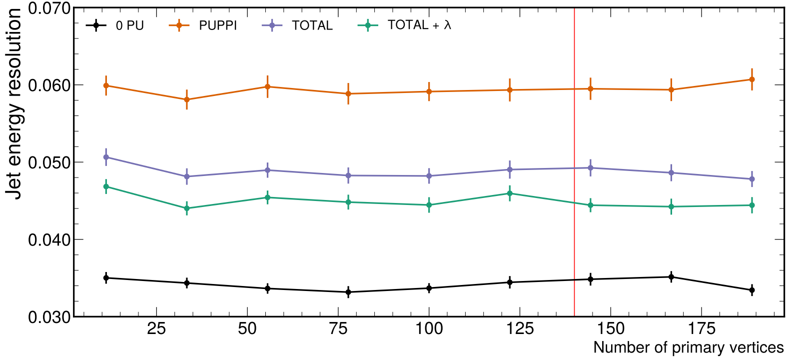
# Differential resolutions




- Jet energy resolution vs jet $p_T$ in $t\bar{t}$ (left) and vs jet $\eta$ in QCD (right)
- Improvement up to 30% in JER, up to 20% in $\eta$ resolution

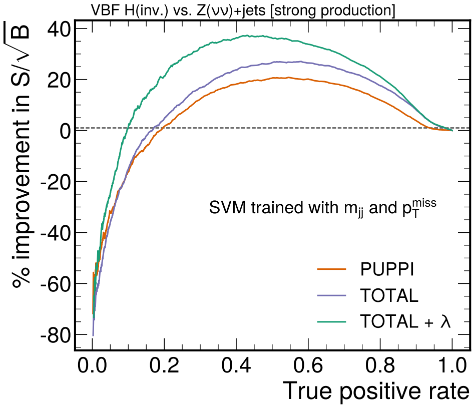
# Robustness




- Evaluate resolution on **processes and PU scenarios unseen during training**
- Network is trained on QCD+$t\bar{t}$+VBF with $\langle \text{NPV} \rangle = 140$
- Evaluate on W+jets production, flat NPV between 0 and 200

# Physics impact

- Study **impact of TOTAL on LHC searches**
  - Search for BSM VBF H(inv.)
- **Signal signature**: pair of forward jets and MET
- **Main background**: strongly produced $Z(\nu\nu)$
- **Perform toy analysis** by training a linear classifier (SVM) using dijet mass and MET
- Improvement in $S/\sqrt{B}$ of the order of 15% for TOTAL

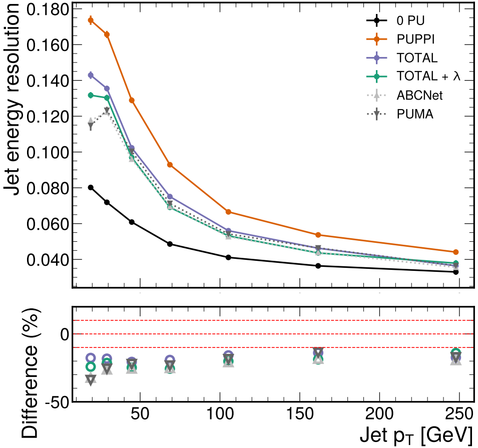



TOTAL PU mitigation

F. Iemmi

# Self-supervised vs fully-supervised trainings

- **Compare performance of TOTAL with fully-supervised algorithms**
- Compare with backbone architecture of TOTAL (ABCNet) and PUMA
- **Performance of TOTAL is comparable** with fully-supervised approaches
- But, contrary to previous approaches, **TOTAL can be ported to full simulation**



# Conclusions

- We presented **novel algorithm to reject PU particles** at high-intensity hadron colliders
  - Trained and tested on Delphes simulation of Phase2 CMS detector
- We are Training Optimal Transport with Attention Learning: **TOTAL**
- We **solved the longstanding problem of neutral labels** in PU mitigation
- **We do not rely on explicit, per-particle labeling**
- **Learning** happens **through OT in a self-supervised fashion**
- Such an algorithm will be **crucial at the High-Luminosity LHC**, where much harsher data-taking conditions are expected
- Our **approach can be generalized** to a wide range of denoising problems
  - Only needed input is a reliable simulation of signal and noise

# Backup slides

# State-of-the-art at CMS: PUPPI [1407.6013]



- Starting from Run3, **default PU mitigation technique** in CMS **is PUPPI**
- **Rule-based** algorithm
- Calculates a weight $w \in [0, 1]$ for each particle in the event
  - Encodes the probability for a particle to be LV or not
  - Weight used to **reweight the particle 4-momentum before jet clustering**
- For charged: use tracking information and assign 0 or 1
- For neutrals: build $\alpha$ variable

$$\alpha_i = \log \sum_{j \neq i, \Delta R_{ij} < R_0} \left( \frac{p_{T,j}}{\Delta R_{ij}} \right)^2 \begin{cases} |\eta_i| < 2.5 & j \text{ are all charged particles from LV} \\ |\eta_i| > 2.5 & j \text{ are all kinds of particles} \end{cases}$$

- QCD is harder and more collimated than PU $\implies$ higher $\alpha$ than PU

# State-of-the-art at CMS: PUPPI [1407.6013]



- To translate into a weight, compare each particle's $\alpha$ with the mean and RMS of PU particles

$$\text{signed}\chi^2_i = \frac{(\alpha_i - \bar{\alpha}_{\text{PU}})|\alpha_i - \bar{\alpha}_{\text{PU}}|}{(\alpha_{\text{PU}}^{\text{RMS}})^2}$$

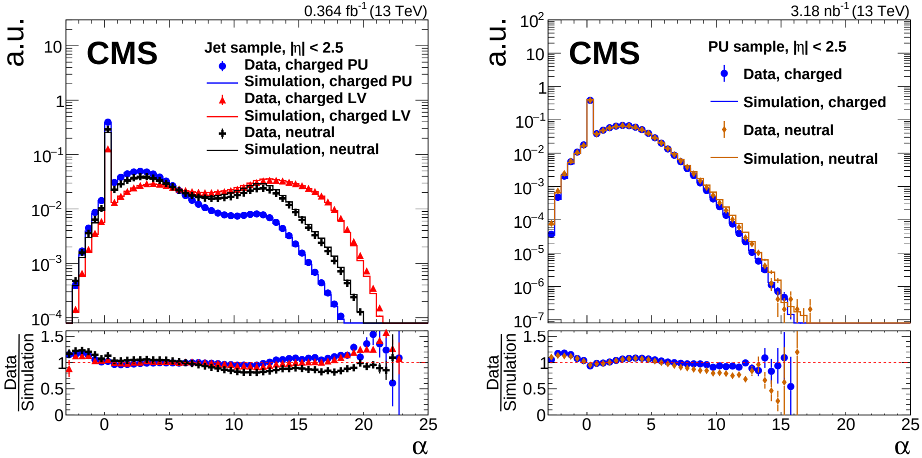
- Use **charged particles** for $\bar{\alpha}_{\text{PU}}$ and $(\alpha_{\text{PU}}^{\text{RMS}})^2$ computation
- Finally, assume signed$\chi^2$ follows a $\chi^2$ distribution and assign weight based on CDF

$$w_i = F_{\chi^2,\text{NDF}=1}(\text{signed}\chi^2)$$

- **LV particle** $\Longrightarrow$ large signed$\chi^2$ $\Longrightarrow$ large CDF $\Longrightarrow$ **large weight**
- **PU particle** $\Longrightarrow$ small signed$\chi^2$ $\Longrightarrow$ small CDF $\Longrightarrow$ **small weight**

# State-of-the-art at CMS: PUPPI [1407.6013]

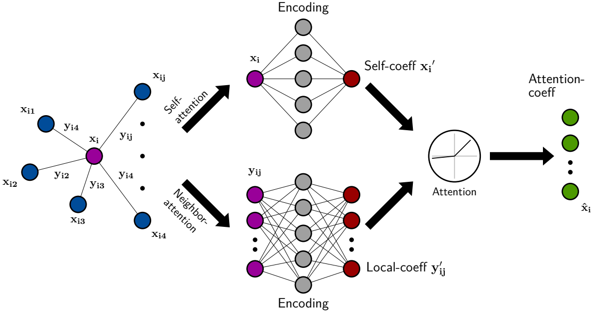




# Attention-Based Cloud network

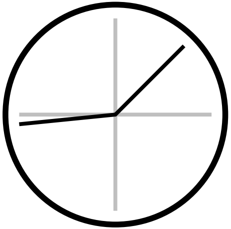


- ABCNet is an **graph neural network** enhanced **with attention mechanisms**
  - Treat particle collision data as a set of permutation-invariant objects
  - Attention mechanisms filter out the particles that are not relevant for the learning process
- Implemented inside custom **graph attention pooling layers** (GAPLayers)



# Attention mechanism





Attention

- Add together self- ($x'_i$) and local- ($y'_{ij}$) coefficients and apply non-linearity

$$c_{ij} = \mathrm{LeakyRelu}(x'_i + y'_{ij})$$

- Align coefficients $c_{ij}$ by applying SoftMax

$$c'_{ij} = \frac{\exp(c_{ij})}{\sum_k \exp(c_{ik})}$$

- Get attention coefficients by multiplying $y'_{ij}$ by $c'_{ij}$

$$\hat{x}_i = \mathrm{Relu}\left(\sum_j c'_{ij} y'_{ij}\right)$$

# Efficient OT: sliced Wasserstein distance (SWD)



- The optimal transport problem has a **closed form for 1D problems**:

$$W_c(p_X, p_Y) = \int_0^1 c\left(P_X^{-1}(\tau), P_Y^{-1}(\tau)\right) \mathrm{d}\tau$$

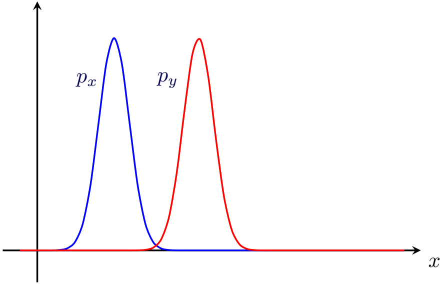
where $p_X, p_Y$ are 1D PDFs, $P_X^{-1}(\tau), P_Y^{-1}(\tau)$ are the respective CDFs and $c(\cdot,\cdot)$ is the transportation cost function

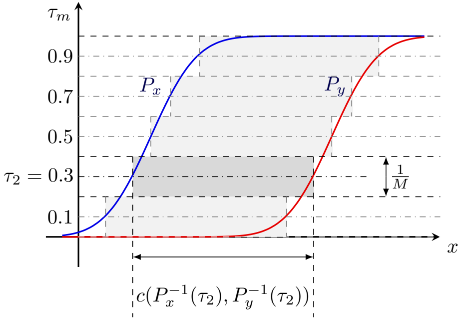
- No guarantee that the integral is solvable (it depends on the form of $c(\cdot,\cdot)$)
- The **integral can always be approximated** by the finite sum

$$\frac{1}{M}\sum_{m=1}^{M} c\left(P_X^{-1}(\tau_m), P_Y^{-1}(\tau_m)\right), \qquad \tau_m = \frac{2m-1}{2M}$$

# Example: M = 5






- $m \in \{1, 2, 3, 4, 5\} \implies \tau_m = \frac{2m-1}{2M} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$

# Efficient OT: sliced Wasserstein distance (SWD)



- In the **special case of discrete distributions** (discrete in nature, or resulting from a sampling), PDFs are sums of Dirac's deltas

$$p_x = \frac{1}{M}\sum_{m=1}^{M}\delta(x - x_m); \qquad p_y = \frac{1}{M}\sum_{m=1}^{M}\delta(y - y_m);$$

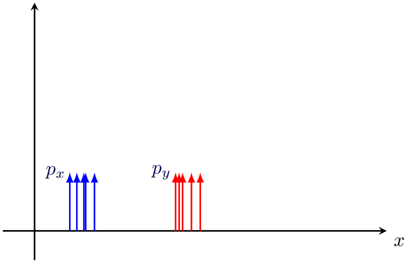
- The integral of a Dirac's delta is the Heaviside's step function $\Theta \implies$
  $\implies$ CDFs are Heaviside functions

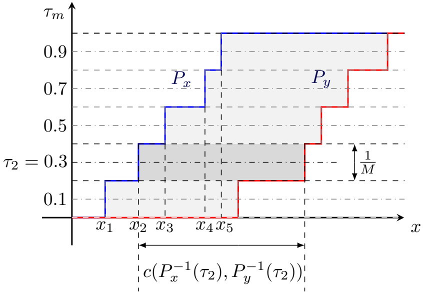
$$P_x(t) = \int_{-\infty}^{t} p_x(z)\mathrm{d}z = \frac{1}{M}\int_{-\infty}^{t}\sum_{m=1}^{M}\delta(z - x_m)\mathrm{d}z = \frac{1}{M}\sum_{m=1}^{M}\Theta(t - x_m)$$

- **If we sort the samples by feature**, the CDFs become a **sum of steps**

# Example: M = 5






- $m \in \{1, 2, 3, 4, 5\} \implies \tau_m = \frac{2m-1}{2M} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$
- Note that

$$P_x^{-1}(\tau_m) = x_m; \qquad P_y^{-1}(\tau_m) = y_m$$

# Efficient OT: sliced Wasserstein distance (SWD)



- Note that

$$P_x^{-1}(\tau_m) = x_m; \qquad P_y^{-1}(\tau_m) = y_m$$

- Therefore

$$W_c(p_X, p_Y) = \frac{1}{M}\sum_{m=1}^{M} c\left(P_X^{-1}(\tau_m), P_Y^{-1}(\tau_m)\right) = \frac{1}{M}\sum_{m=1}^{M} c\left(x_m, y_m\right)$$

- The **1D OT problem is reduced to a sorting** of the 1D feature
  - **Fast and easy to solve**

# CHECKPOINT

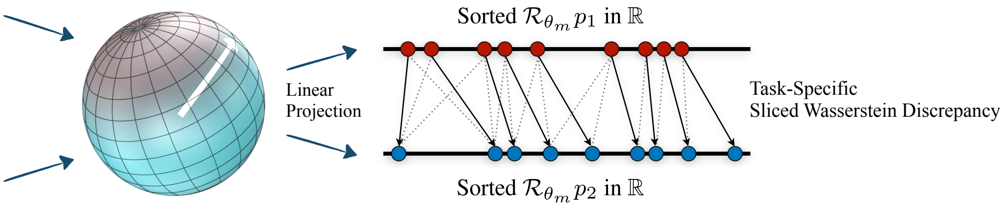
1. Optimal transport problem has a closed form in 1D
2. For sampled distributions, the problem is reduced to a sorting of the 1D feature
3. Particles have multi-dimensional distributions though. How to apply this?

# Efficient OT: sliced Wasserstein distance (SWD)



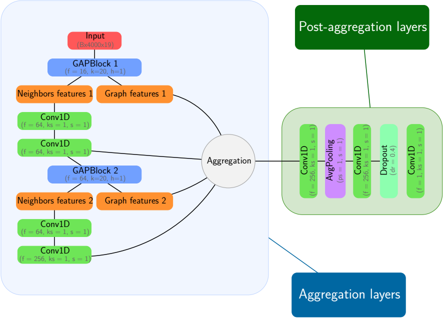
- Each particle is a sample from a *n*-D feature space
- **SWD**: take *n*-D feature space and **project (slice)** it **to 1D**
- Project on a vector belonging to $S^{n-1}$
- For robustness, take **multiple random slices**

- Now can **solve the 1D OT problem for each slice**
- **Sort particles by slice**
- The **average on all slices and particles** becomes the **loss function**



# The model






- **9 input features**:
  - ($p_T$, $\eta$, $\phi$, $E$)
  - Charge
  - PDG ID
  - dXY & dZ impact parameters
  - Vertex association (for charged)
- **Loss**: $\mathrm{SWD}(\vec{x}_p \cdot \vec{\omega}, \vec{x}_{np}) + \mathrm{MET}$ constraint
- **Cost function**: squared distance
- **Sliced features**: ($p_T$, $\eta$, $\phi$, $E$)
- **Output**: per-particle weight $\vec{\omega}$

- Train on **300k events**, equally split between QCD multijet, $t\bar{t}$ dileptonic and VBF Higgs($4\nu$) processes
- Consider **9000 particles per event** (zero-padding included)
- Gather the **20 *k*-nearest neighbors** for each particle when building graph