REPRESENTATION WORKSHOP SUMMARY

Savannah Thais

Learning to Discover

04/28/2022

Thanks to our wonderful speakers!

Machine learning for charged particle tracking

Jan Stark
Laboratoire des 2 Infinis - Toulouse (L2IT)

Learning to discover, Institut Pascal, Université Paris-Saclay
Workshop on representation learning, April 19th 2022

Université Toulouse III Paul Sabatier — cnrs

Overview of Machine Learning for Calorimeter and Particle Flow

Joosep Pata (NICPB, Estonia)

April 19, 2022
Learning To Discover
Institut Pascal, Université-Paris-Saclay

Learning to Discover – April 2022

GEOMETRIC DEEP LEARNING
THE ERLANGEN PROGRAMME OF ML

Michael Bronstein

University of Oxford

DeepMind

Learning general purpose physical simulators

Presenter: Alvaro Sanchez-Gonzalez

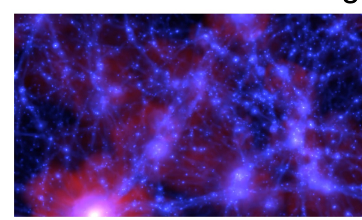
April 19, 2022

Workshop on Representation Learning from Heterogeneous/Graph-Structured Data

Learning to Discover - Institut Pascal Paris-Saclay

University of Sussex — UCL

Rediscovering orbital mechanics with Machine Learning



Pablo Lemos
Learning to Discover - Representation Learning Workshop
https://arxiv.org/abs/2202.02306

DeepMind

Enabling Empirically & Theoretically Sound Algorithmic Alignment

Petar Veličković

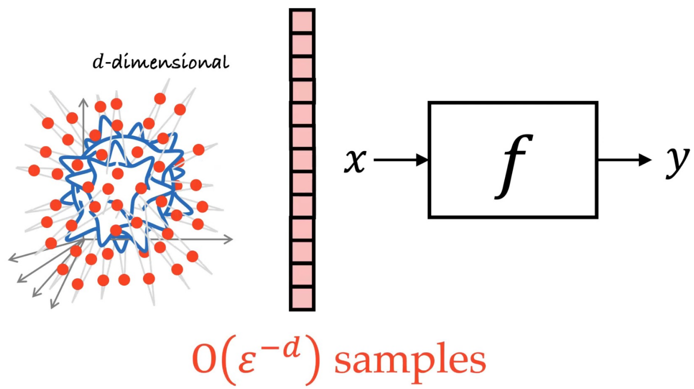
Learning to Discover
20 April 2022

Physics and ML are concerned with characterizing the true probability distributions of nature, how do we represent truth, data, and models to best enable learning these distributions?

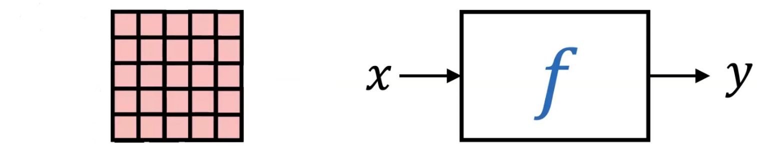
Geometric Deep Learning

Representation Priors

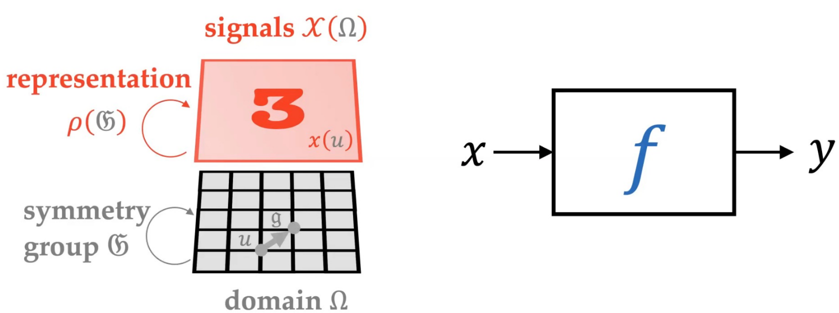
The Curse of Dimensionality



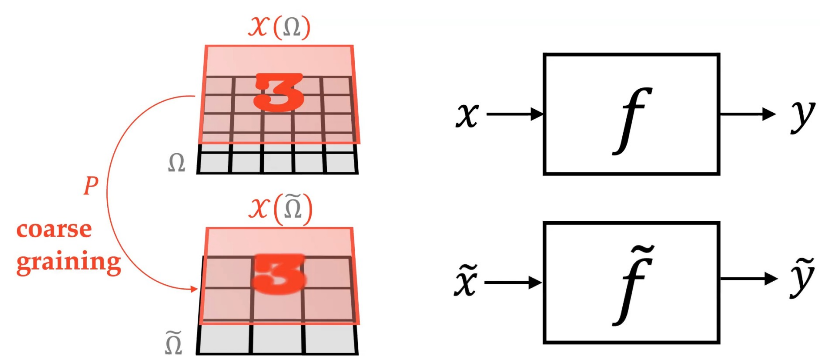
Geometric Priors



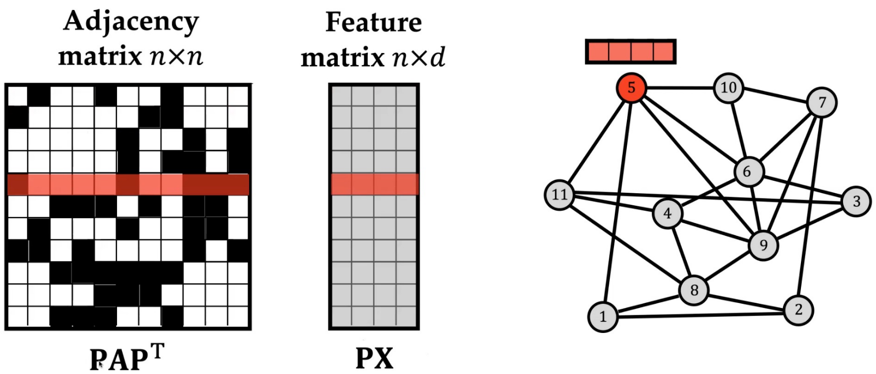
Symmetry Prior



Scale Separation Prior



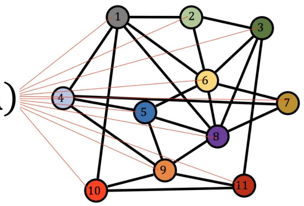
Graph Neural Networks



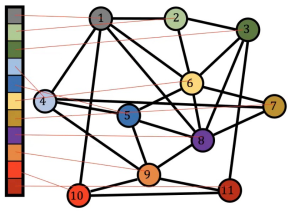
arbitrary ordering of nodes

n! permutations

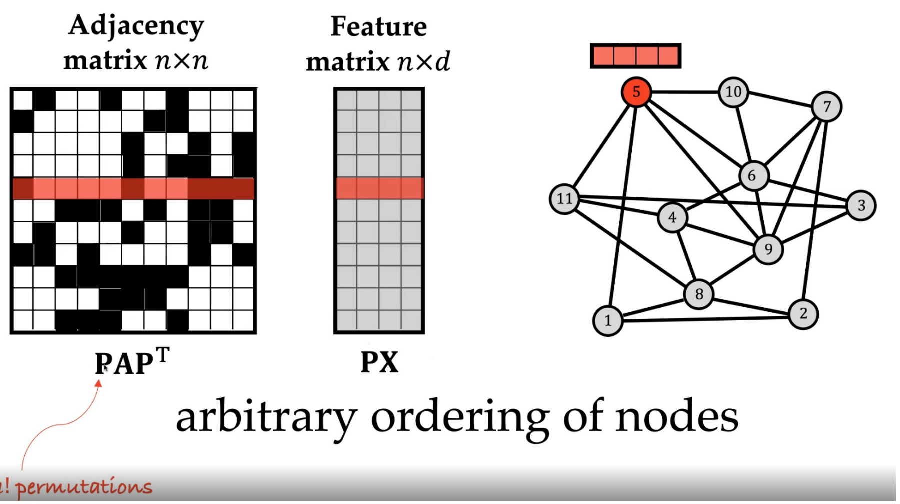
graph function $f(\mathbf{X}, \mathbf{A})$

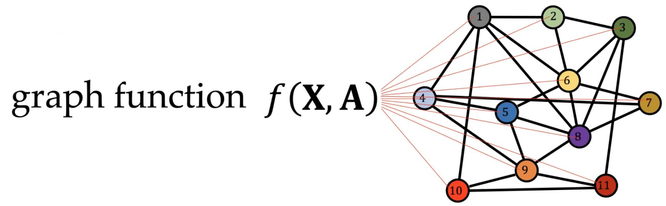


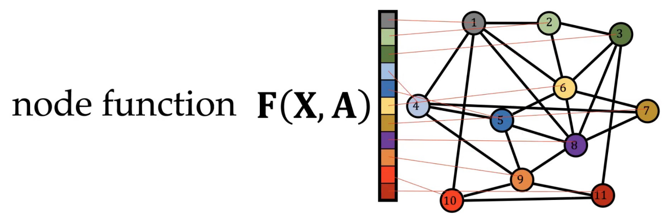
node function $\mathbf{F}(\mathbf{X}, \mathbf{A})$

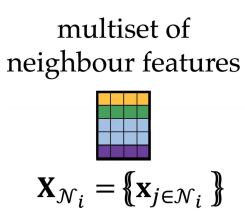


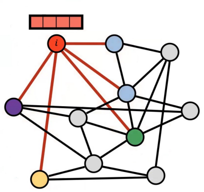
Graph Neural Networks

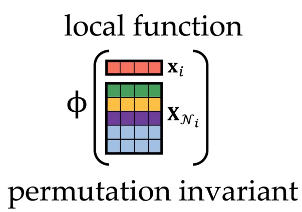


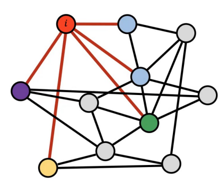


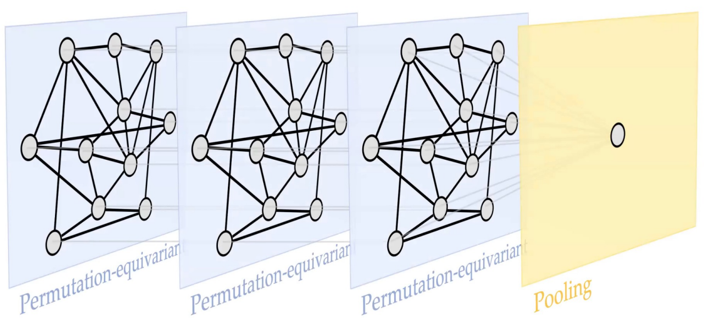




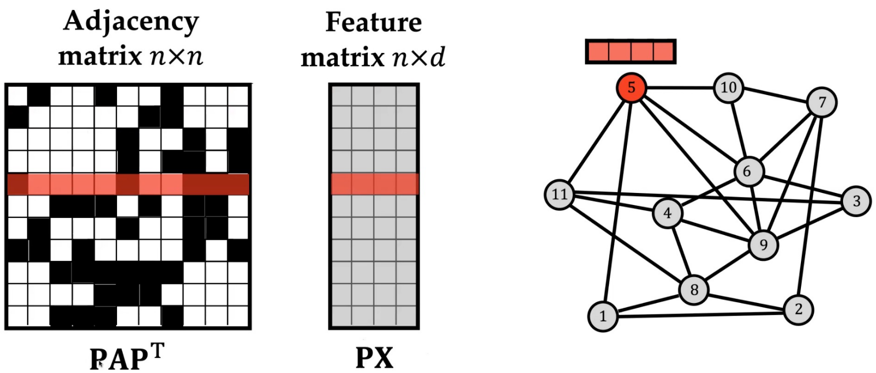








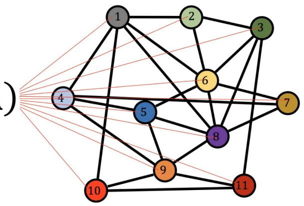
Graph Neural Networks



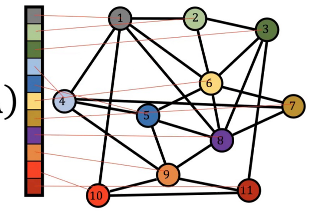
arbitrary ordering of nodes

$n!$ permutations

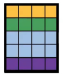
graph function $f(\mathbf{X}, \mathbf{A})$



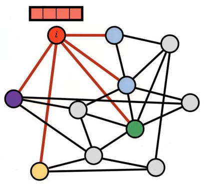
node function $\mathbf{F}(\mathbf{X}, \mathbf{A})$



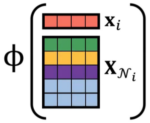
multiset of neighbour features



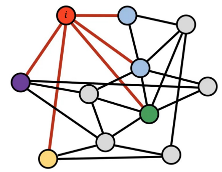
$\mathbf{X}_{\mathcal{N}_i} = \{\mathbf{x}_{j \in \mathcal{N}_i}\}$

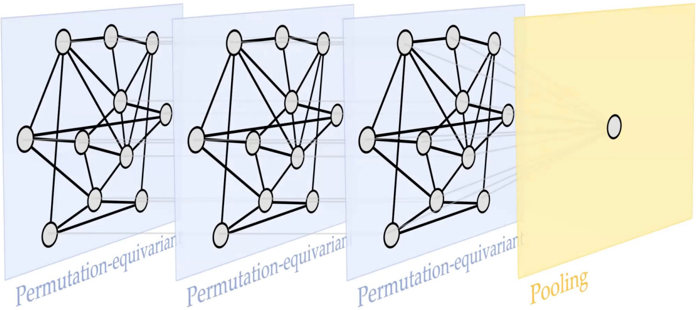


local function



permutation invariant





permutation-invariant aggregation operator, e.g. sum

$$f(\mathbf{x}_i) = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} \psi(\mathbf{x}_j)\right)$$

learnable functions

$$f(\mathbf{x}_i) = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} c_{ij}\psi(\mathbf{x}_j)\right)$$

"convolutional"

$$f(\mathbf{x}_i) = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right)$$

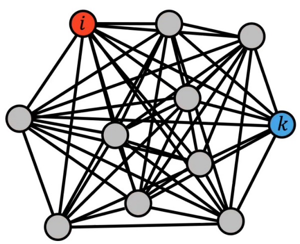
"message passing"

$$f(\mathbf{x}_i) = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} a(\mathbf{x}_i, \mathbf{x}_j)\psi(\mathbf{x}_j)\right)$$

"attentional"

Extensions of GNNs

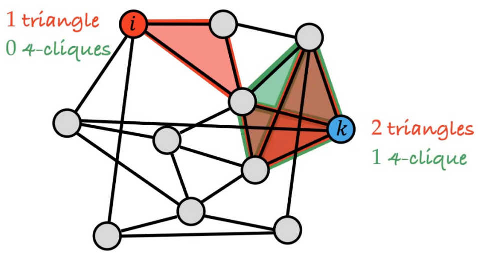
Transformers



$$\phi\left(\mathbf{x}_i, \underset{j=1}{\overset{n}{\square}} a(\mathbf{x}_i, \mathbf{x}_j, \mathbf{p}_i, \mathbf{p}_j)\psi(\mathbf{x}_j)\right)$$

positional encoding

Graph Substructure Network

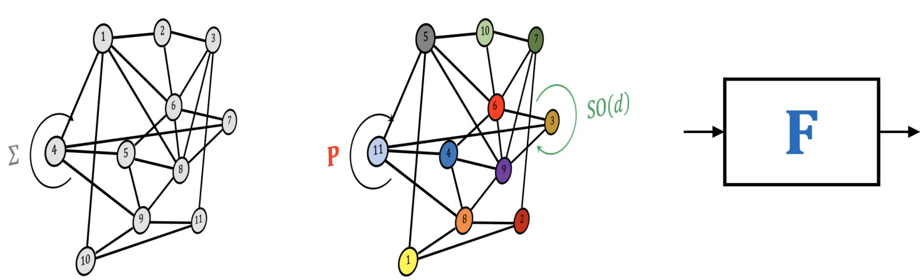


$$\phi\left(\mathbf{x}_i, \underset{j \in \mathcal{N}_i}{\square} \psi(\mathbf{x}_i, \mathbf{x}_j, \mathbf{p}_i)\right)$$

structural encoding

Equivariant Graph Neural Networks

Graph $G = (V, E)$ | **Node features $\mathcal{X}(G)$** | **functions $\mathcal{F}(\mathcal{X}(\Omega))$**



Permutation group Σ_n | **Permutation matrix P** | **Equivariant message passing**

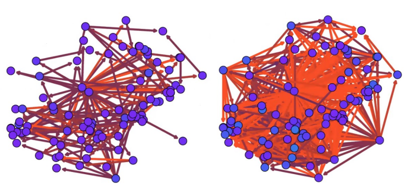
Rotation R

$$\mathbf{F}(\mathbf{PXR}, \mathbf{PAP}^\top) = \mathbf{PF}(\mathbf{X}, \mathbf{A})\mathbf{R}$$

Graph Rewiring

Decouple **input graph** from **information propagation graph** (at the expense of link to WL)

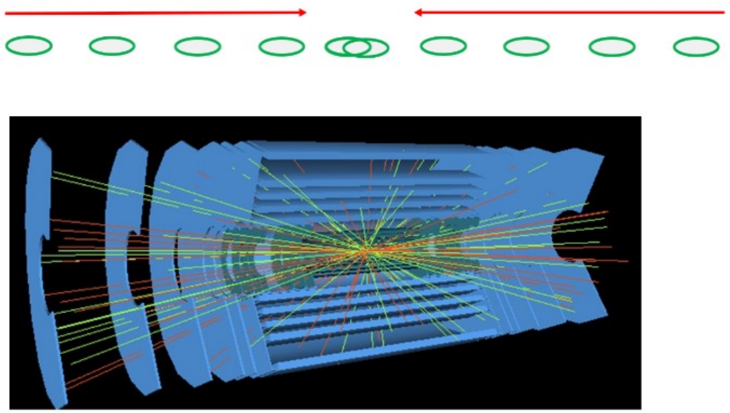
- Neighbourhood sampling (GraphSAGE)[1]
- Multi-hop filters (SIGN)[2]
- Complete graph[3]
- Topology diffusion (DIGL)[4]
- Learnable graph (Dynamic Graph CNN)[5]

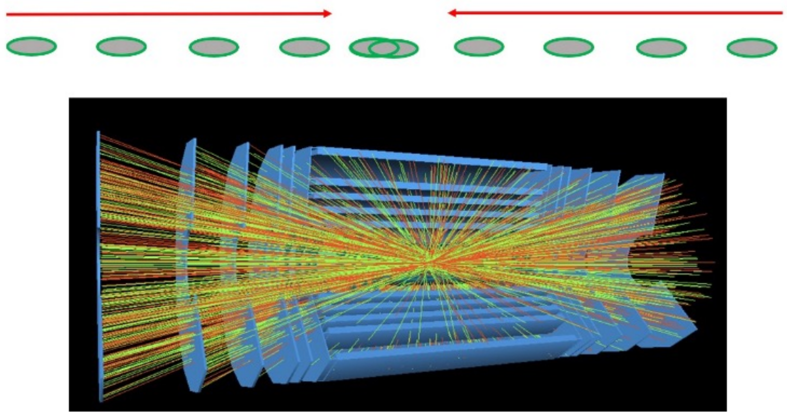


GDL for Physics Tasks

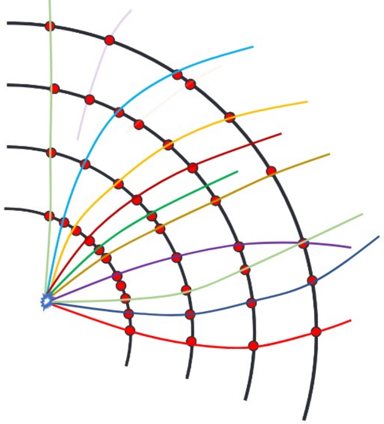
Tracking

High luminosity: how ? Cannot reduce distance between bunches any further. More protons/bunch !

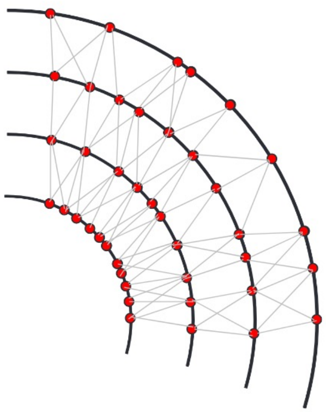




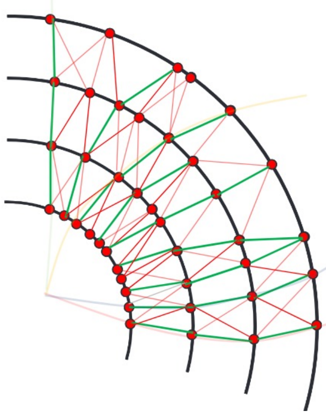
Charged particles leave hits in the detector



Represent the data using a graph



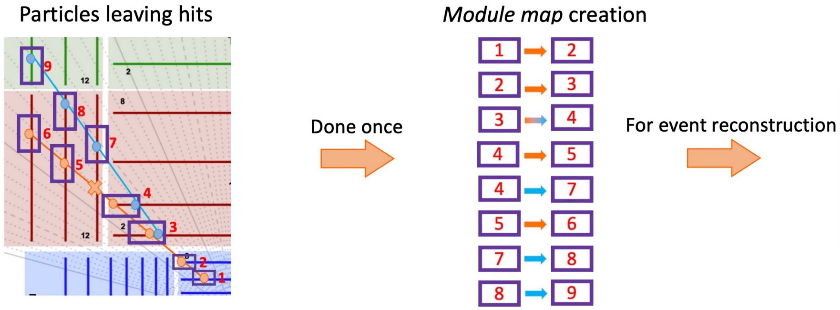
Goal:
classify the edges of the graph

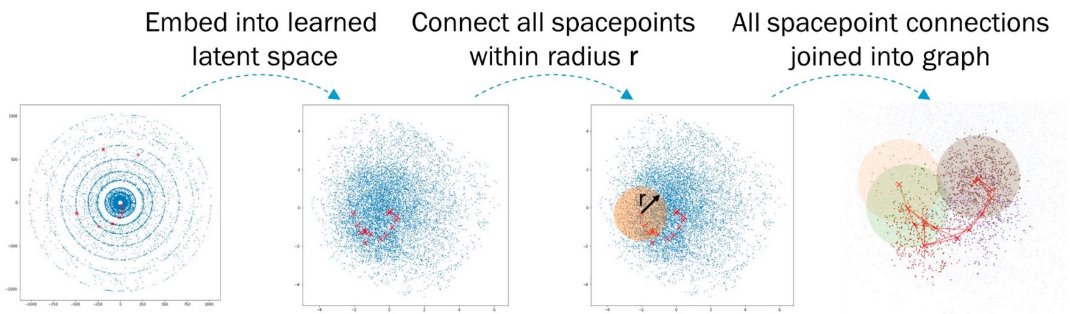


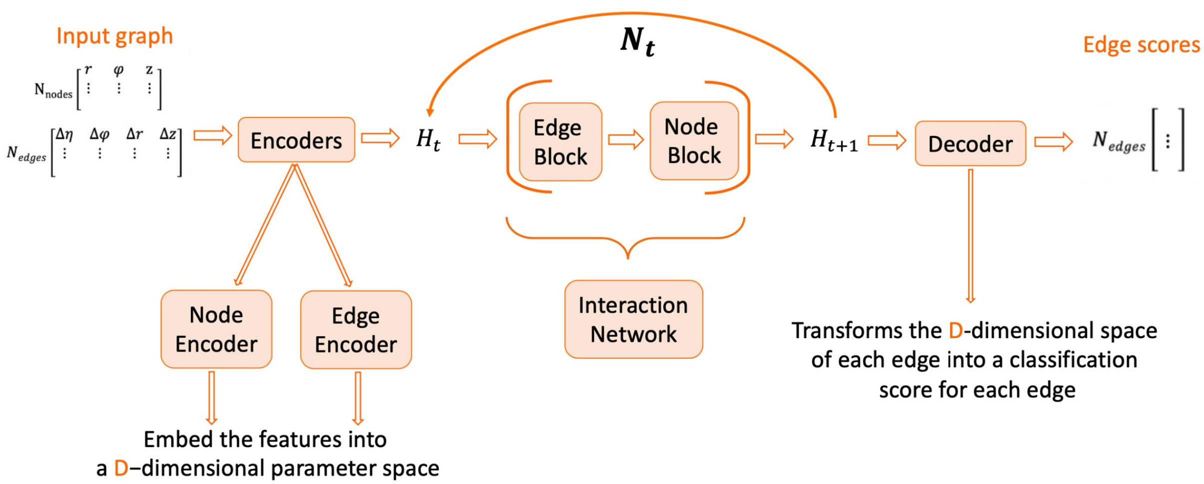
High classification score
=> **high probability** that the edge is part of a track

Low classification score
=> **low probability** that the edge is part of a track

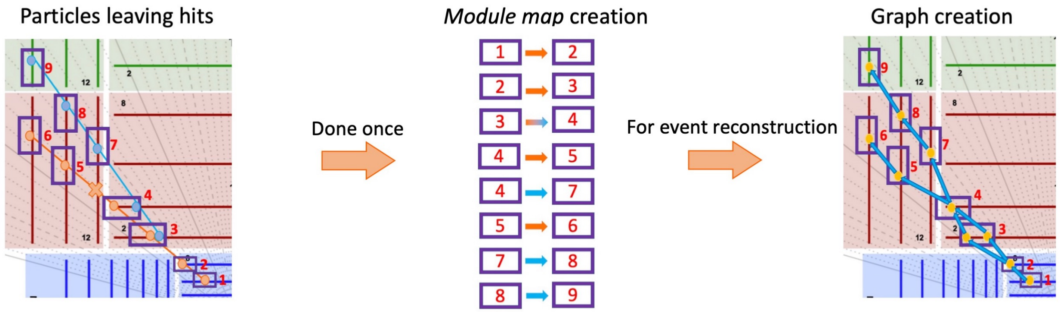
GNNs for Tracking

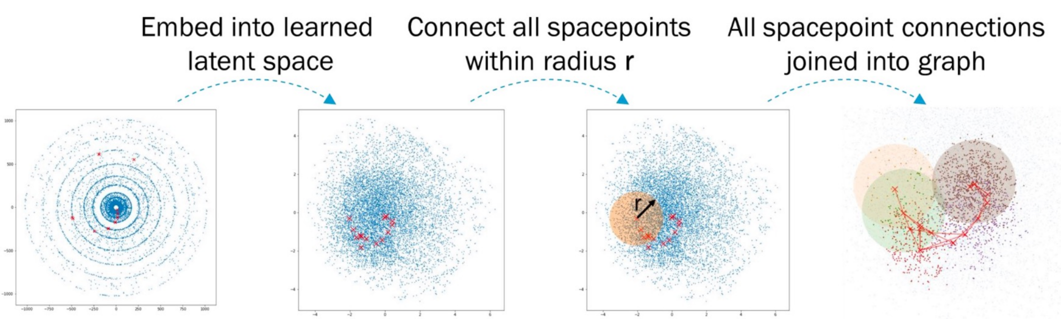


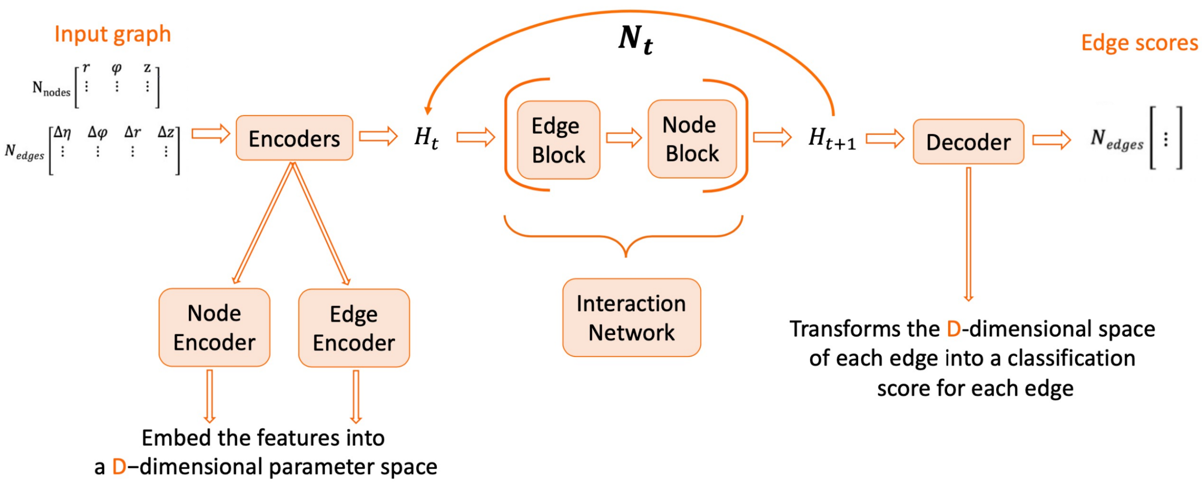


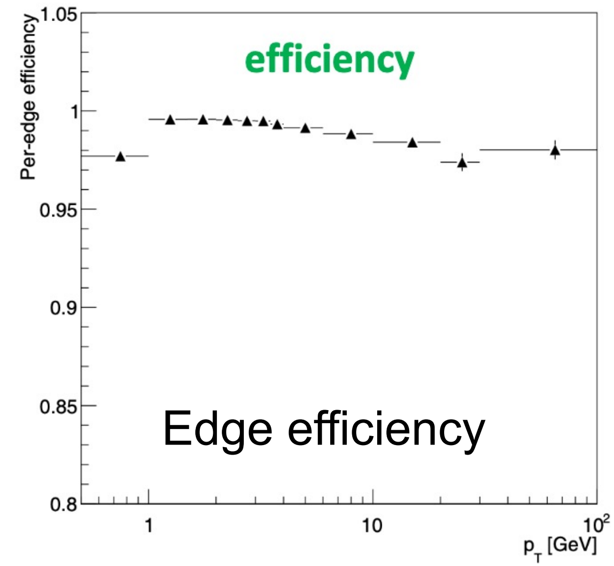


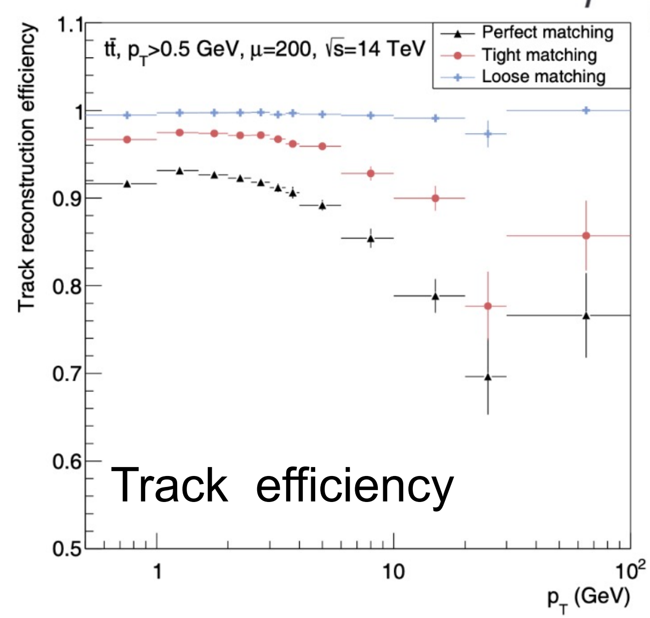
GNNs for Tracking





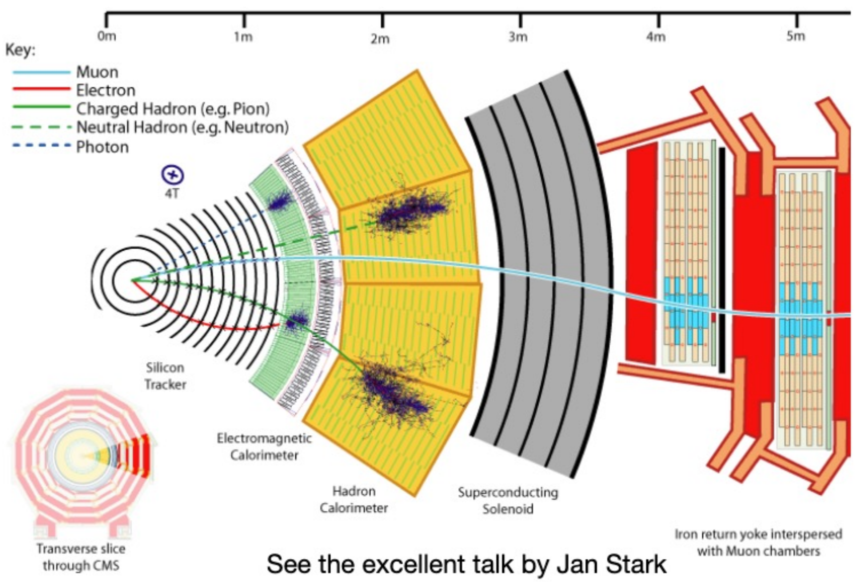






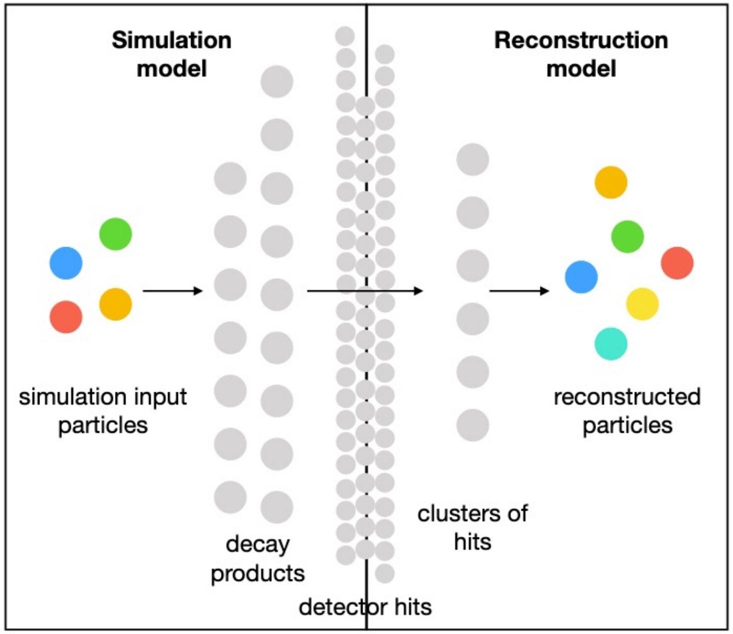
Reconstruction

Multilayered detectors



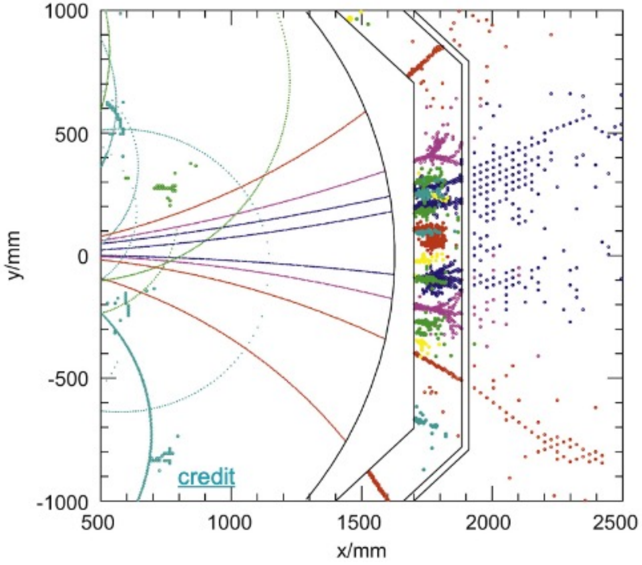
See the excellent talk by Jan Stark earlier today.

Simulation to reconstruction



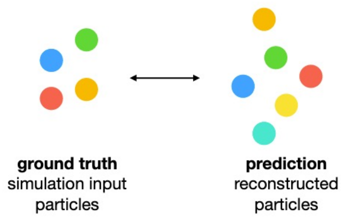
Clustering

- Segment the energy deposits (hits) according to the originator particles
- The hits are embedded in a complicated feature space (Cartesian position, energy, signal significance, timing, layer information, ...)
- Showers from different particles may overlap spatially
- **Standard heuristic approaches** based on seeding & collecting neighbors, typically iterative



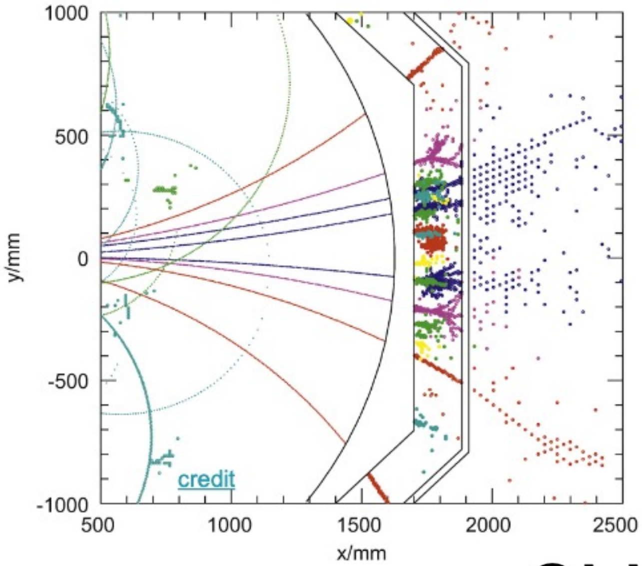
Set-to-set problem

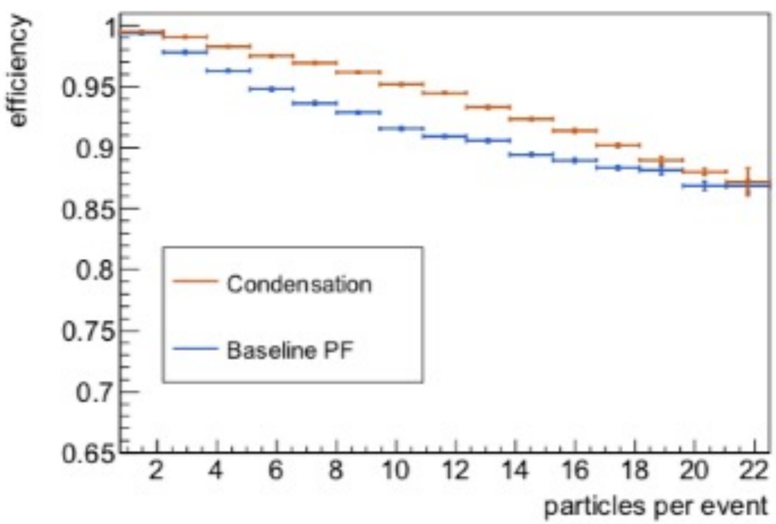
Each particle is described by a multi-class label, and is embedded in a complex, problem-dependent feature space.



Clustering

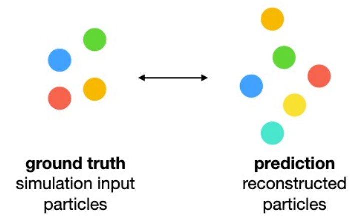
- Segment the energy deposits (hits) according to the originator particles
- The hits are embedded in a complicated feature space (Cartesian position, energy, signal significance, timing, layer information, ...)
- Showers from different particles may overlap spatially
- **Standard heuristic approaches** based on seeding & collecting neighbors, typically iterative





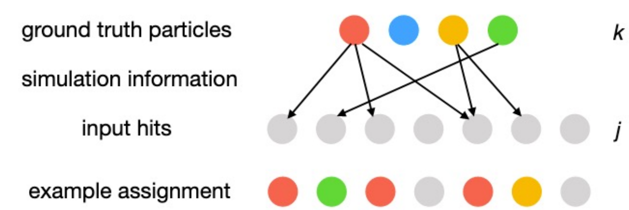
Set-to-set problem

Each particle is described by a multi-class label, and is embedded in a complex, problem-dependent feature space.



Object condensation

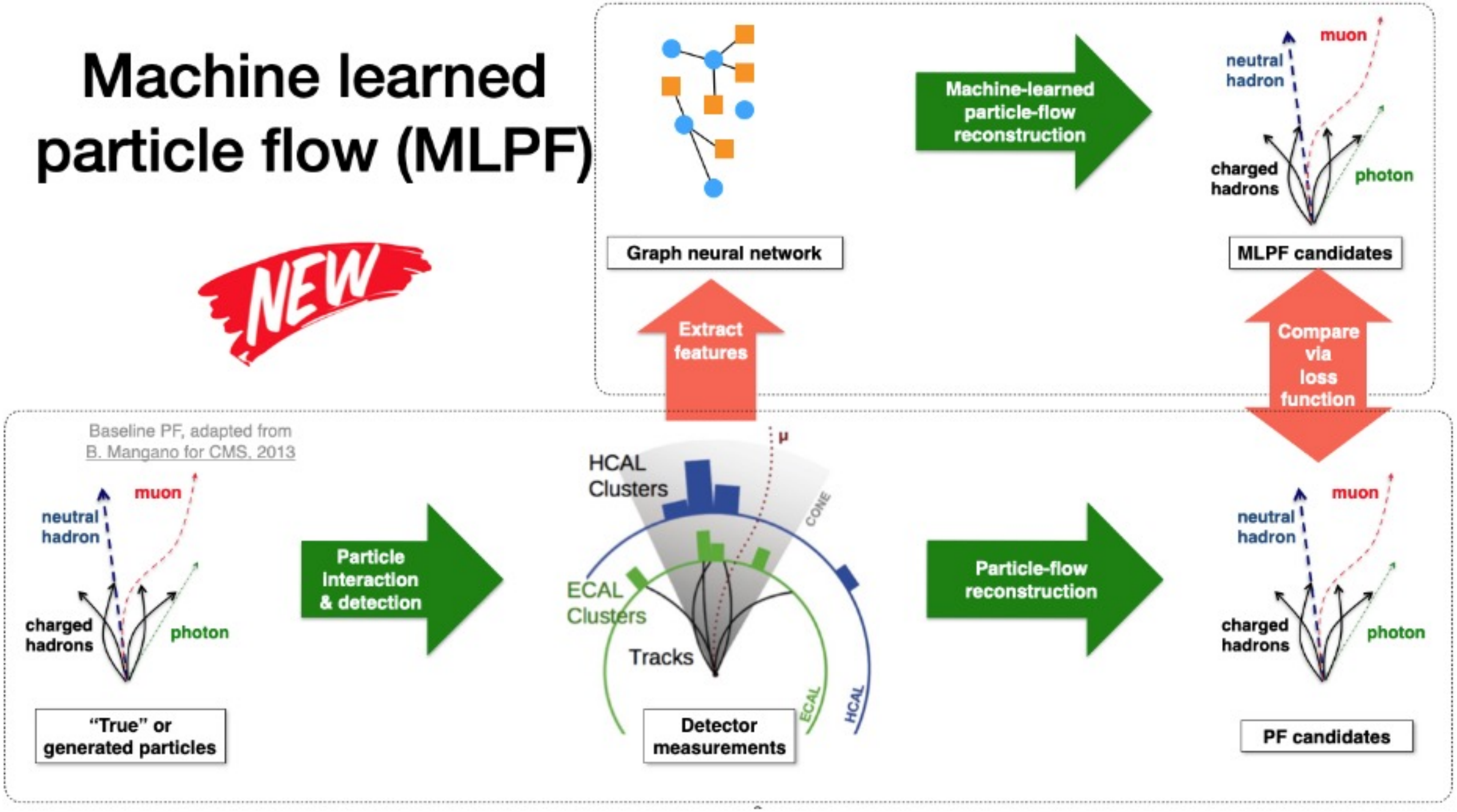
Boundedness: the number of truth particles usually cannot be larger than the number of inputs (typically it's much smaller).



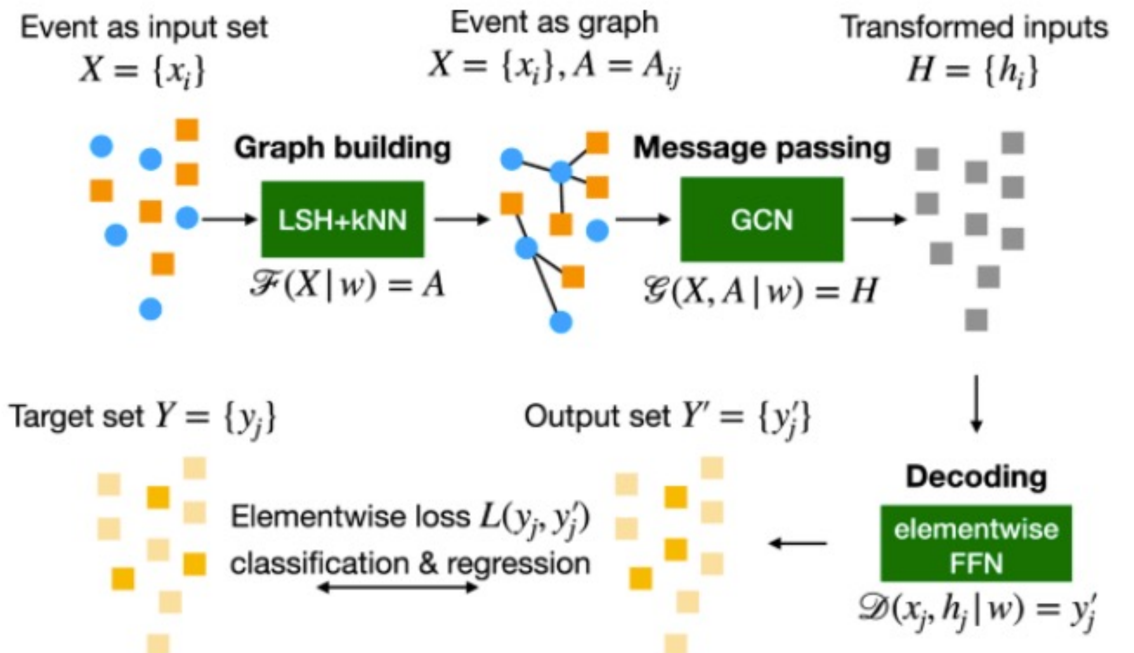
Each input represents exactly one truth particle, with attractive/repulsive potentials in a learned space x_j between correct/incorrect assignments.

$$L_V = \frac{1}{N}\sum_{j=1}^{N} q_j \sum_{k=1}^{K}\left(\underbrace{M_{jk}\check{V}_k(x_j)}_{\text{attractive}} + \underbrace{(1-M_{jk})\hat{V}_k(x_j)}_{\text{repulsive}}\right).$$

Particle Flow



Particle Flow



$$x_i = [\text{type}, p_T, E_{ECAL}, E_{HCAL}, \eta, \phi, \eta_{outer}, \phi_{outer}, q, \ldots], \quad \text{type} \in \{\text{track}, \text{cluster}\}$$

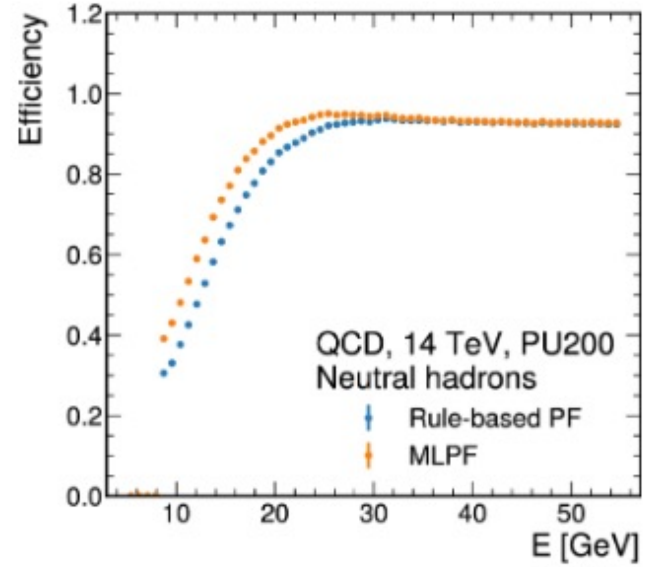
$$y_j = [\text{PID}, p_T, E, \eta, \phi, q, \ldots], \quad \text{PID} \in \{\text{none}, \text{charged hadron}, \text{neutral hadron}, \gamma, e^{\pm},$$

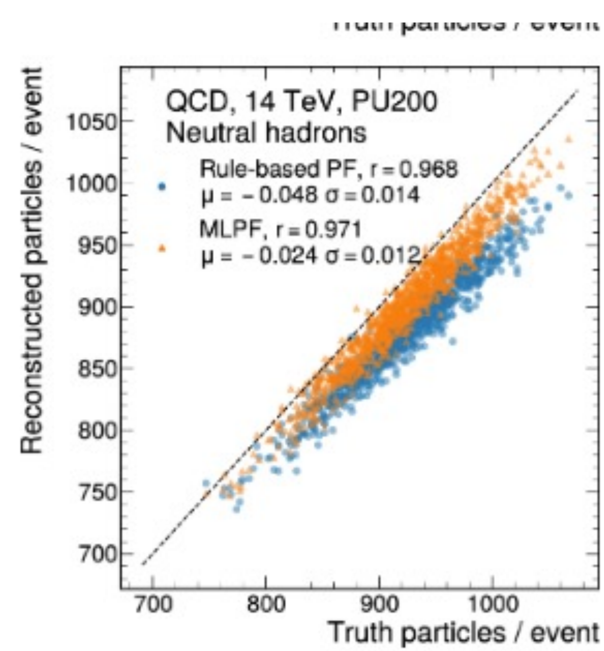
$$h_i \in \mathbb{R}^{256}$$

Trainable neural networks: $\mathscr{F}, \mathscr{G}, \mathscr{D}$

● - track, ■ - calorimeter cluster, ■ - encoded element

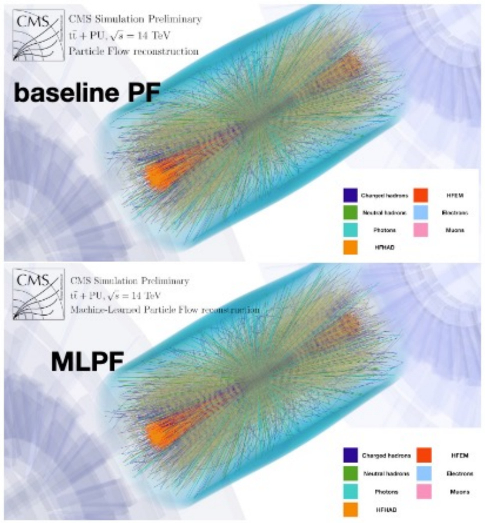
■ - target (predicted) particle, ■ - no target (predicted) particle

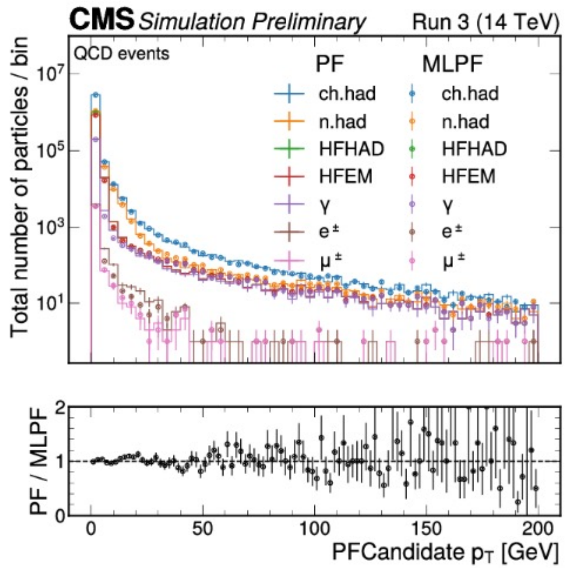




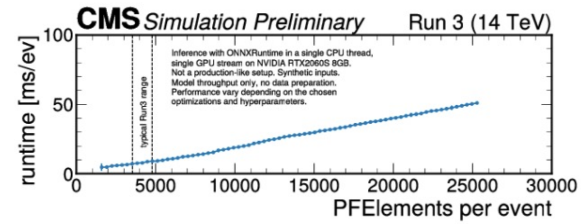
Applicability

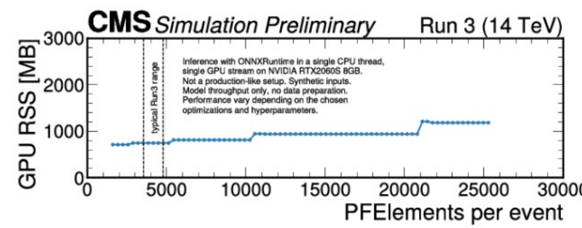
In a realistic environment





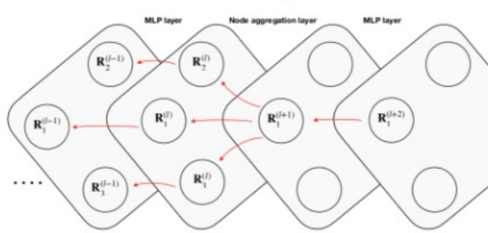
Computational scalability

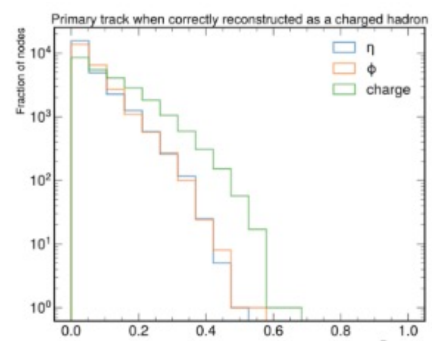


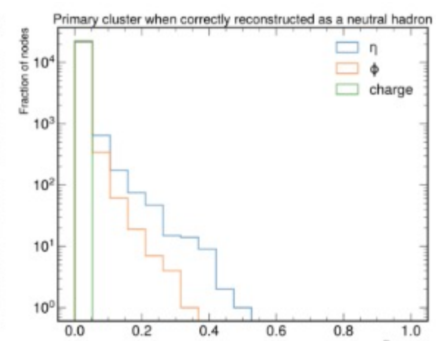


Interpretability

- What inputs are relevant for a particular model output?
- Compute layerwise relevance scores **R**
- Aggregate along the graph







Adding Physics In

Modeling Physical Systems

AI, ML & DATA ENGINEERING

QCon Plus (May 17-28): Stay Ahead of Emerging Software Trends

Deep Learning Accelerates Scientific Simulations up to Two Billion Times

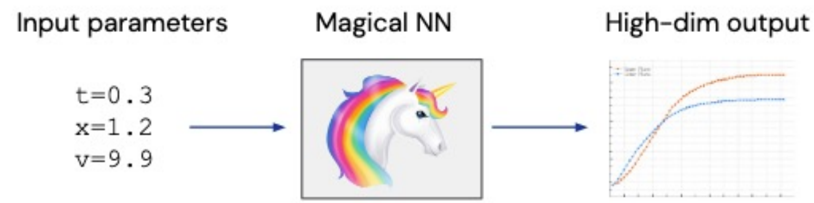
LIKE DISCUSS

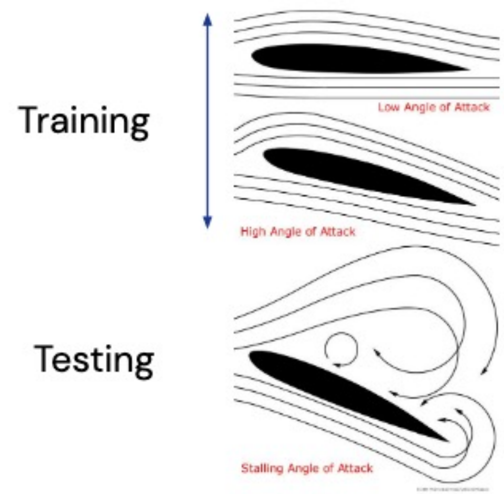
MAR 10, 2020 • 3 MIN READ

by

Anthony Alford

FOLLOW





1. Neural networks are good at interpolation, bad at extrapolation
2. Learned physics models often don't learn anything close to the underlying physical equations
3. There's no way we can build a dataset that covers the input space of a general-purpose simulator

Physics Inspired Priors/Inductive Biases

A simple inductive bias: Inertial dynamics

$x^{t+1} = \mathbf{NN}(x^t, v^t)$

Has to learn to predict static motion

Position: x(t)

Velocity: v(t)

$$\sum \mathbf{F} = m\mathbf{a} = m\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2}$$

Static prior

$x^{t+1} = x^t + \mathbf{NN}(x^t, v^t)$

Trivial to predict static motion

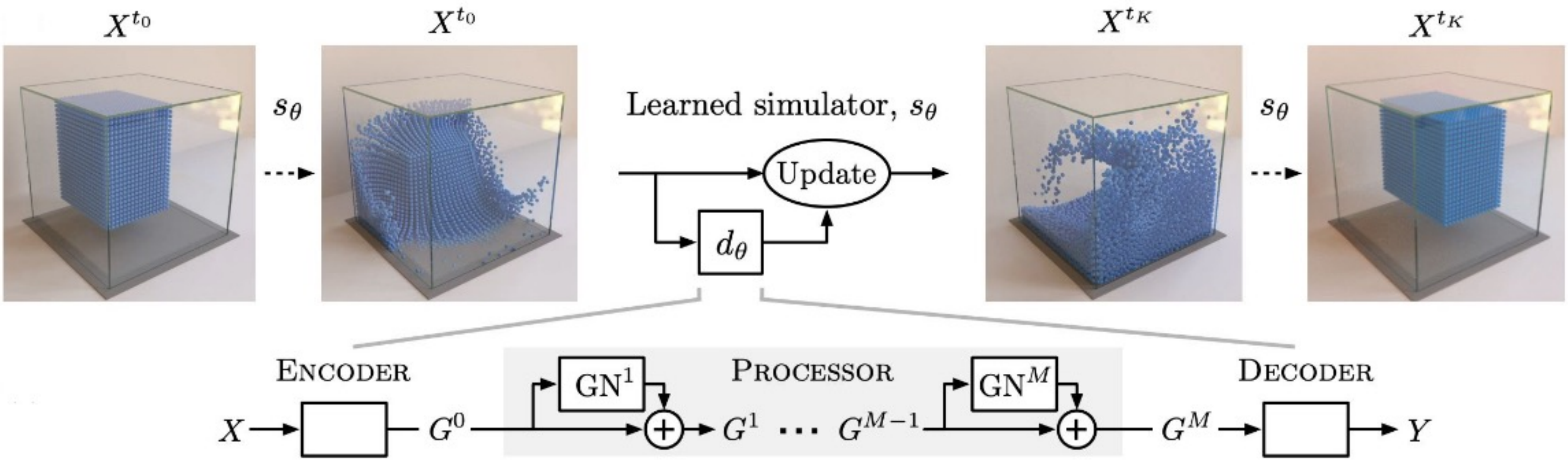
Has to learn to predict inertial motion

Inertial prior

$x^{t+1} = x^t + \Delta t \cdot v^t + \mathbf{NN}(x^t, v^t)$

Trivial to predict inertial motion!

Physics Inspired Priors/Inductive Biases



Spatial equivariance

Local interactions

Universal rules

Pairwise interactions

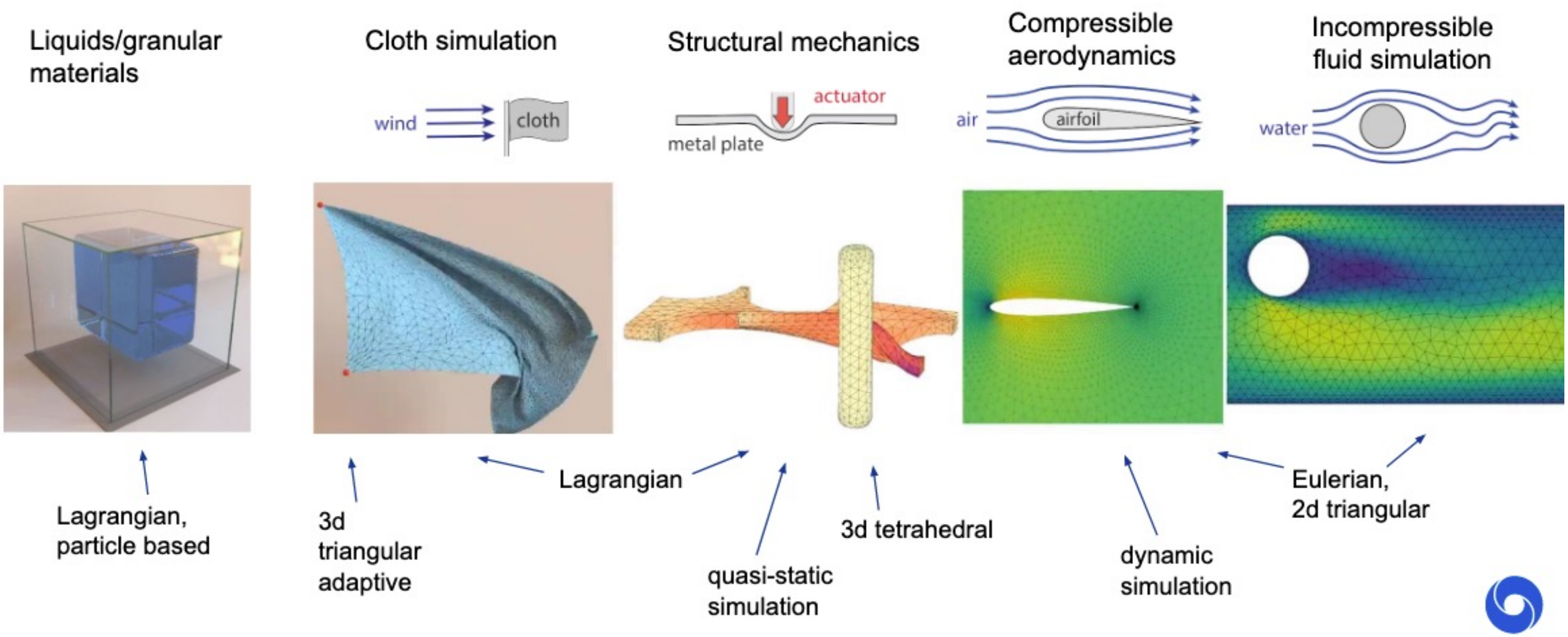
Superposition principle

Differential equations

Permutation equivariance

Learning Physical Simulators

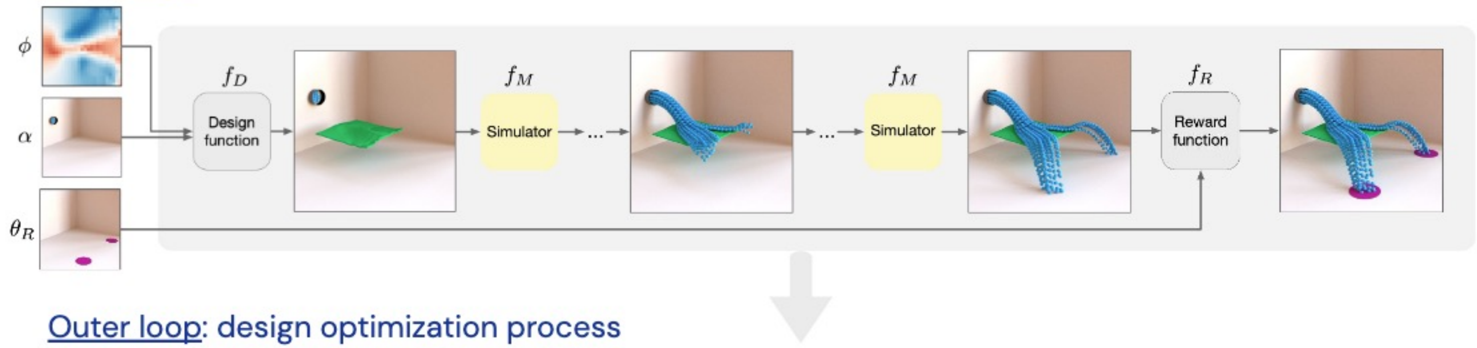
same model, same hyperparameters can simulate many systems



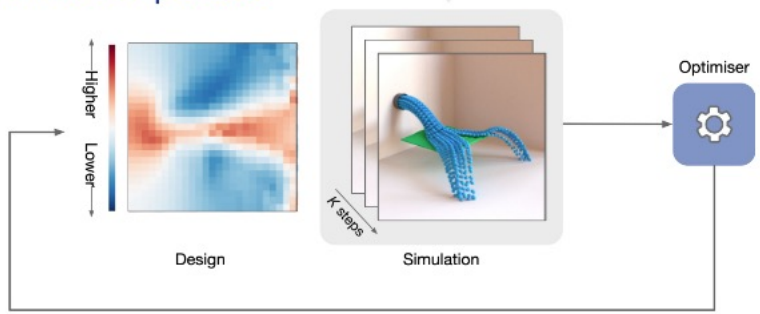
- And many ways to improve:
  - Adding noise, ask update function to remove, improves stability
  - Remeshing (scale prior) improves precision/speed
  - Adaptive remeshing improves precision/compute utilization

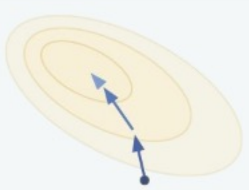
Learning System Design

Inner loop: forward model rollout

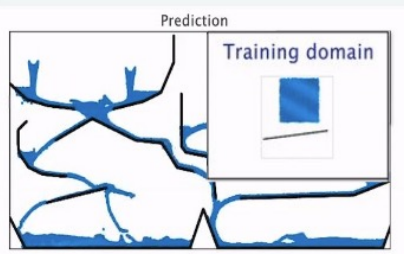


Outer loop: design optimization process





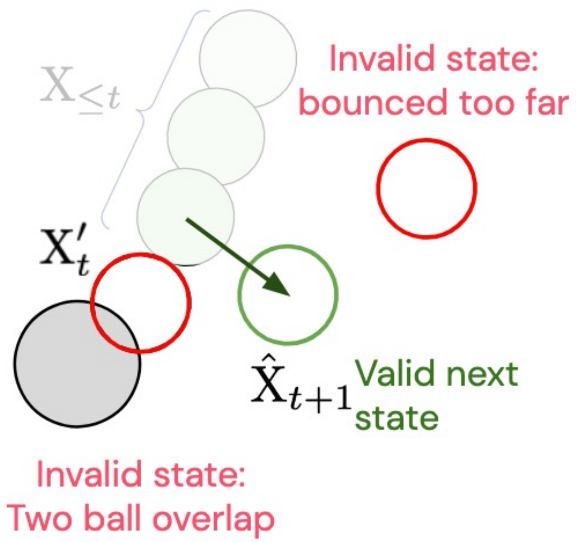
Gradient-based with learned models



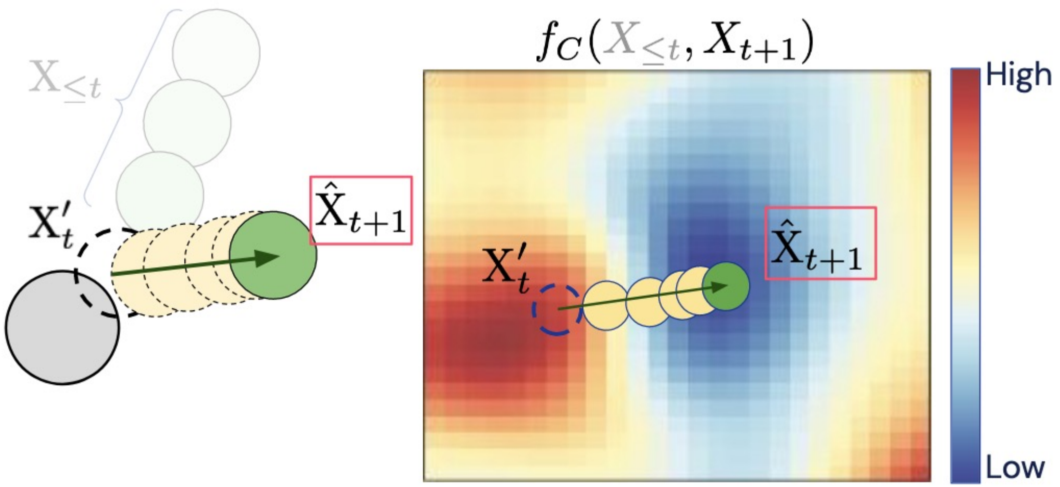
*Can we use a **GNN** based model **pre-trained** on physical dynamics for **inverse design**?*

Sanchez-Gonzalez*, Godwin*, Pfaff*, Ying*, *et al*, ICML 2020 / Pfaff*, Fortunato*, *et al*, ICLR 2021;

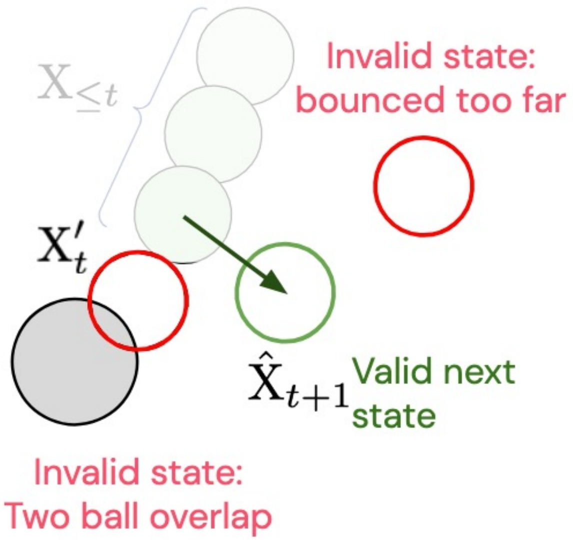
Constraint-Based GNNs



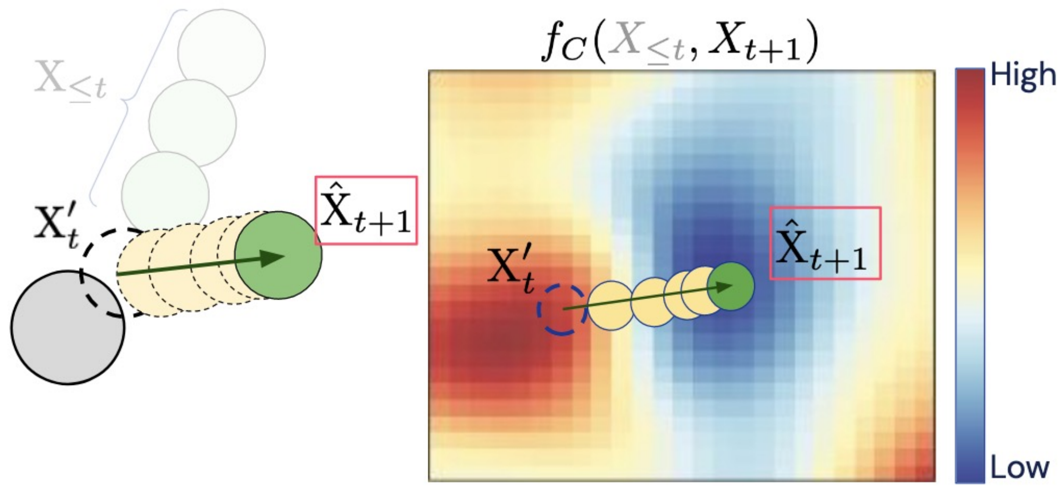
Run gradient descent on f_C to find $\boxed{\hat{X}_{t+1}}$



Constraint-Based GNNs



Run gradient descent on f_C to find $\boxed{\hat{X}_{t+1}}$

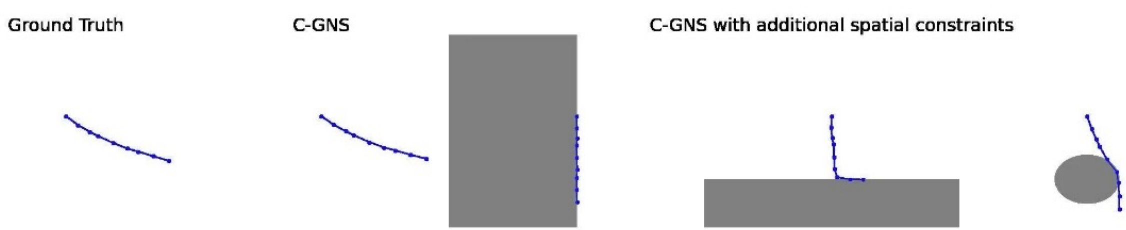


At test time optimize $f_C(X_{\leq t}, X_{t+1}) + f_{\text{obstacle}}(X_{t+1})$

a learned constraint

a user-defined constraint (e.g. new obstacle)

No collisions were ever observed at training time!



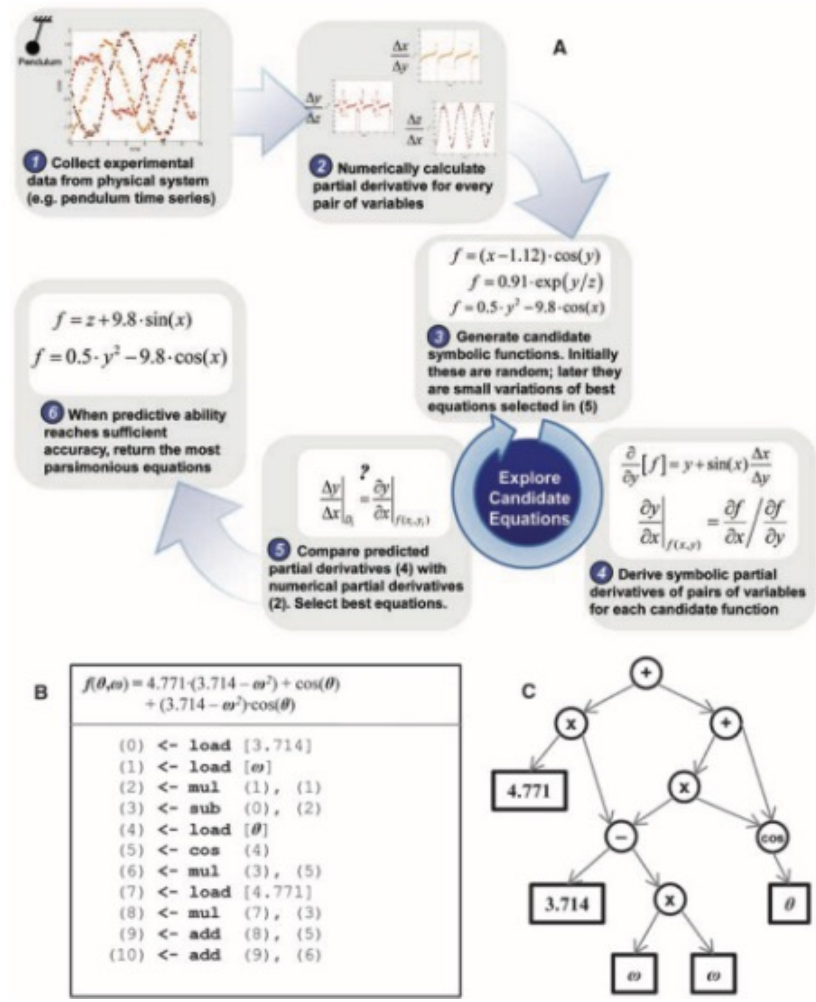
Getting Physics Back Out

Symbolic Regression

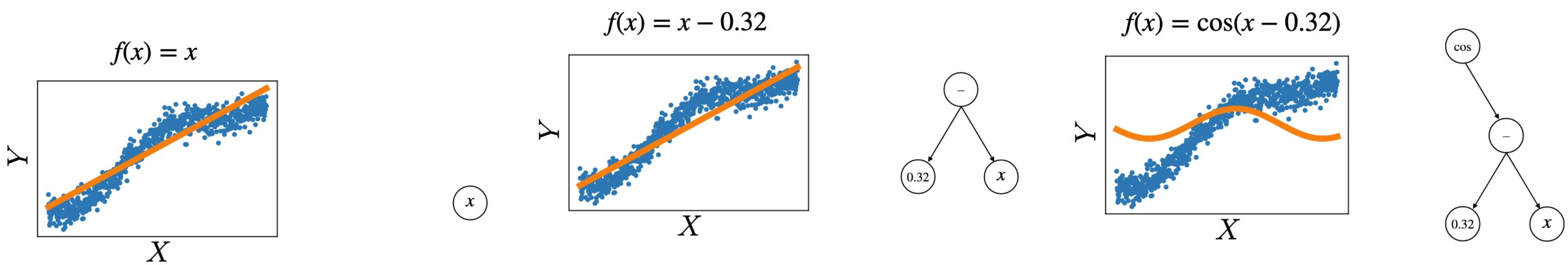
Distilling Free-Form Natural Laws from Experimental Data

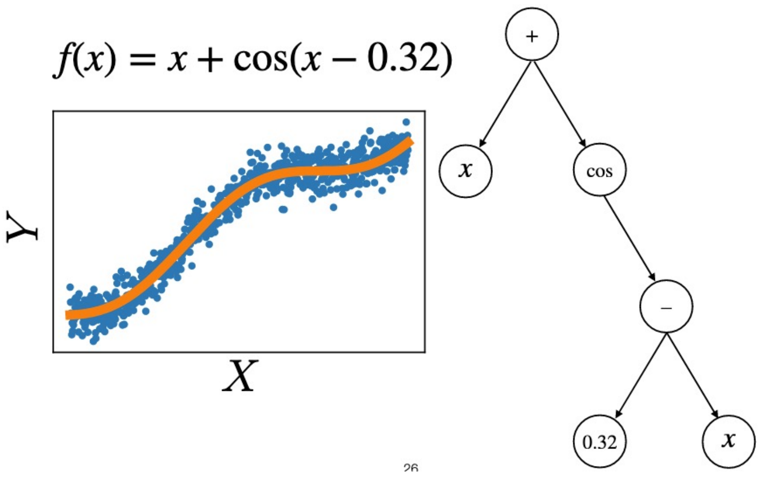
Michael Schmidt[1] and Hod Lipson[2,3]*

For centuries, scientists have attempted to identify and document analytical laws that underlie physical phenomena in nature. Despite the prevalence of computing power, the process of finding natural laws and their corresponding equations has resisted automation. A key challenge to finding analytic relations automatically is defining algorithmically what makes a correlation in observed data important and insightful. We propose a principle for the identification of nontriviality. We demonstrated this approach by automatically searching motion-tracking data captured from various physical systems, ranging from simple harmonic oscillators to chaotic double-pendula. Without any prior knowledge about physics, kinematics, or geometry, the algorithm discovered Hamiltonians, Lagrangians, and other laws of geometric and momentum conservation. The discovery rate accelerated as laws found for simpler systems were used to bootstrap explanations for more complex systems, gradually uncovering the "alphabet" used to describe those systems.



Symbolic Regression





https://github.com/MilesCranmer/PySR

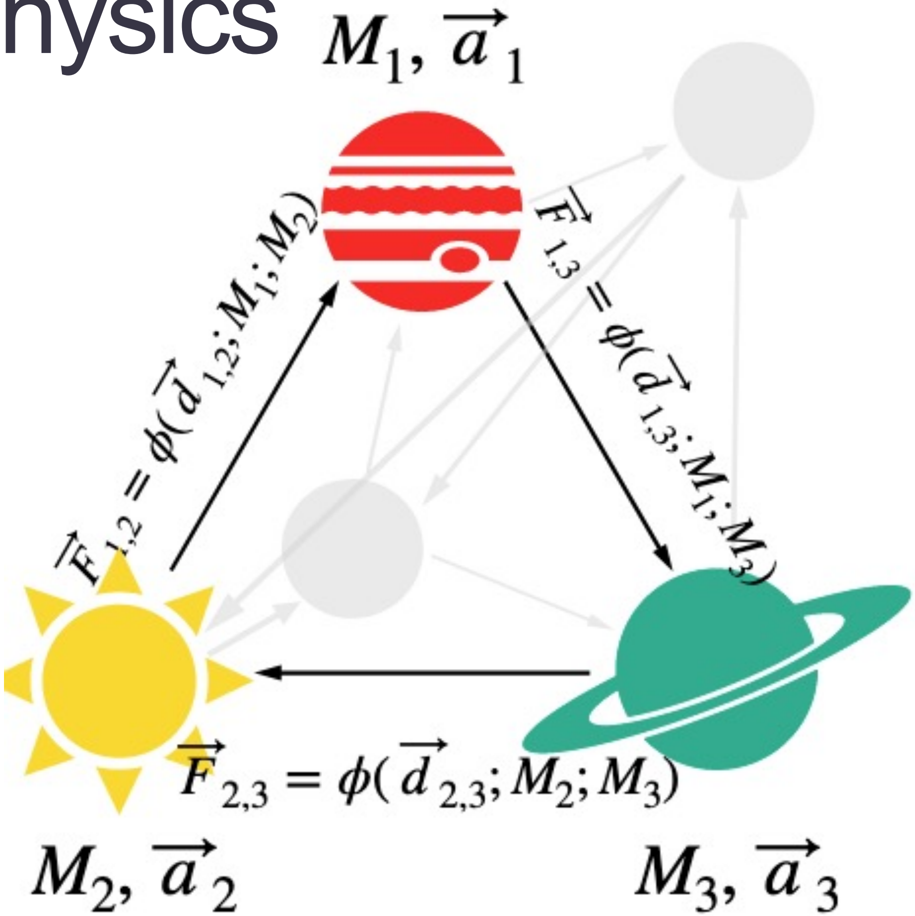
PySR: High-Performance Symbolic Regression in Python

PySR is built on an extremely optimized pure-Julia backend, and uses regularized evolution, simulated annealing, and gradient-free optimization to search for equations that fit your data.

Docs	pip	conda	Stats
docs passing	pypi package 0.7.9	conda-forge v0.7.9	downloads 110k

Repeats process iteratively to yield set of candidate equations

Learning Astrophysics



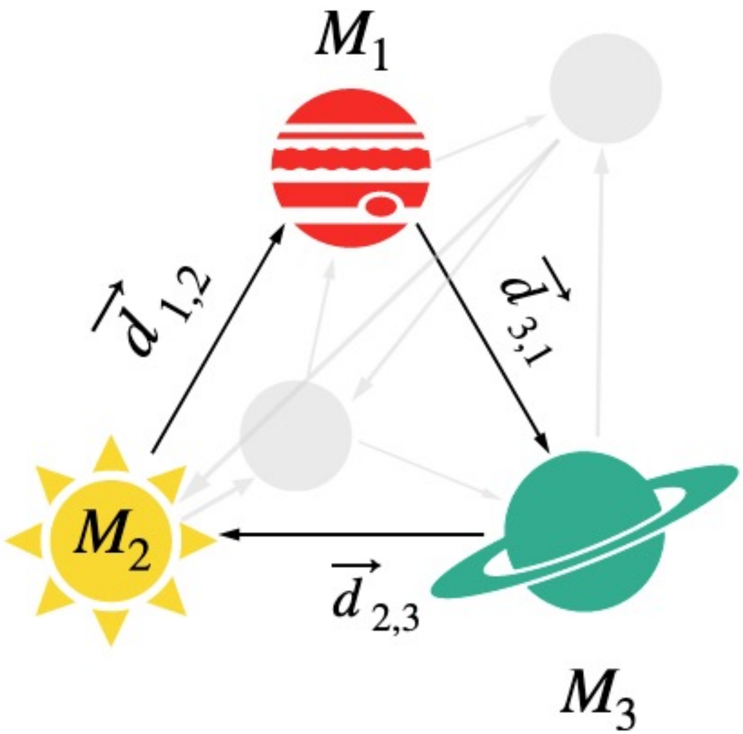
1. Our inputs are the positions of the bodies
2. They are converted into pairwise distances
3. **Our model tries to guess a mass for each body**
4. **It then also guesses a force, that is a function of distance and masses**
5. **Using Newton's laws of motion ($\sum \vec{F} = M\vec{a}$) it converts the forces into accelerations**
6. **Finally, it compares this predicted acceleration, with the true acceleration from the data**

Minimize

$$\left| \vec{a}(\text{pred}) - \vec{a}(\text{true}) \right|^2$$

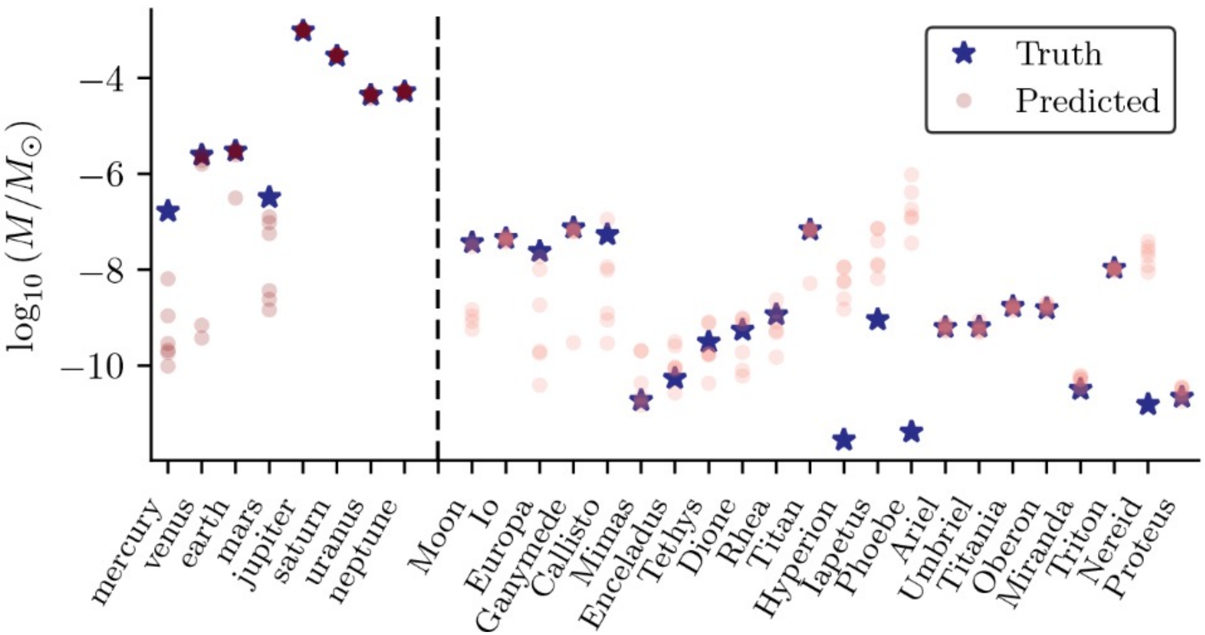
Inductive Biases

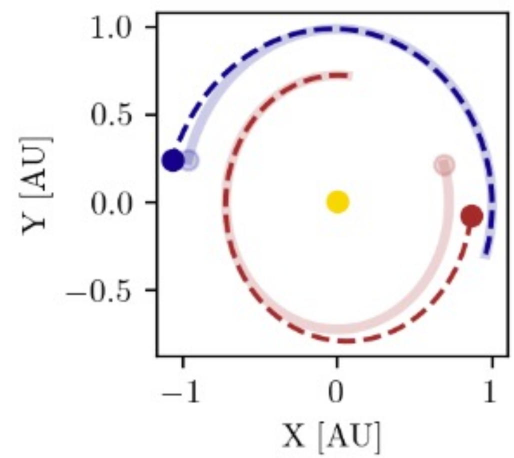
- Translational symmetry
- Rotational symmetry
- Newton's second law
  $$\sum \overrightarrow{F} = M\overrightarrow{a}$$
- Newton's third law
  $$\overrightarrow{F}_{ij} = -\overrightarrow{F}_{ji}$$
- Choice of reference frame, units, etc.

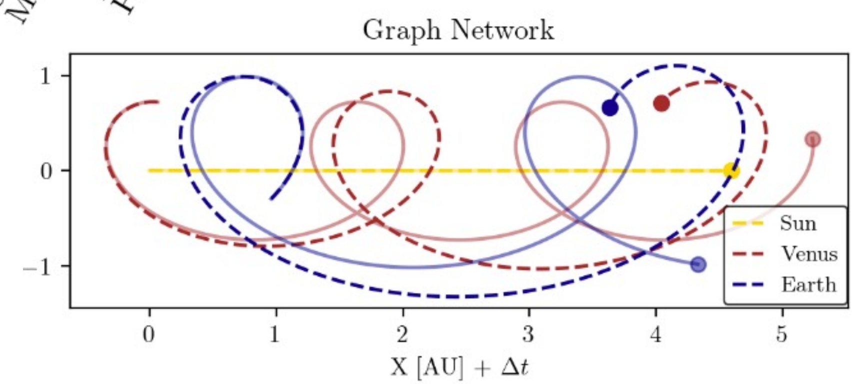


Learning Astrophysics

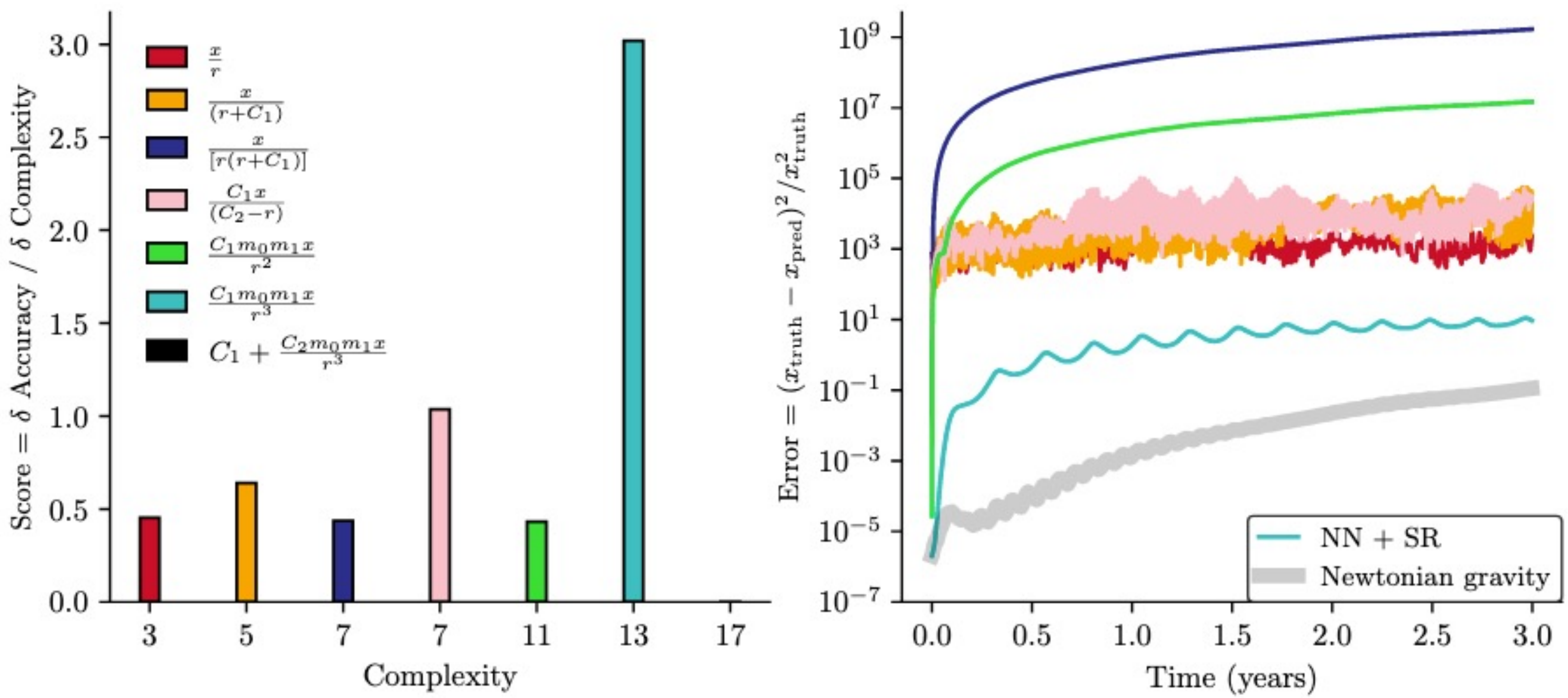
Predicted masses







Extracting the Physics



- Apply symbolic regression with a constraint to balance accuracy and equation complexity
- Can substitute learned equation for the force guess to improve the simulator

Discussion Highlights

How Can We Make This Usable?

- Graph construction is critical for effective learning and meeting computing constraints
  - Are there ways to do effective segmentation or hierarchical graphs
  - How do we balance information sharing with size
- Incorporating inductive biases can improve stability, generalizability, and model efficiency
  - Equivariant GNNs could reduce training resources, generalize
  - Attention mechanisms weight physically important information
  - Are there other types of (intermediate) functions we could model
  - Constrained problems may be harder to solve in some cases
- We need to ensure the problem is truly physical
  - In high pileup overlapping tracks can share hits and even segments
  - How do we handle noise, missing information, detector effects
- Hardware-based acceleration is likely necessary

Does This Help Us Do Physics?

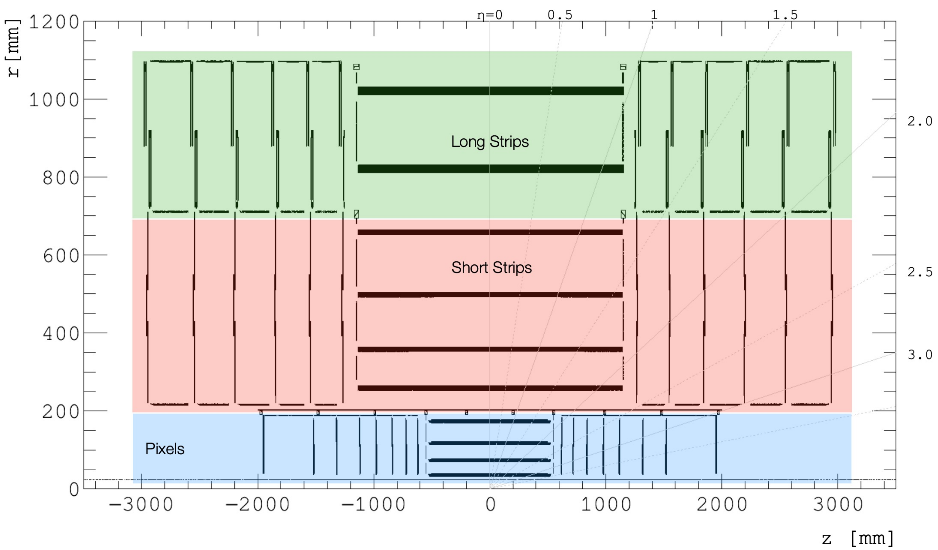
- Graphs seem to be the most effective representation of particle physics experiment data
  - Reduced information loss, allows hierarchical representations
  - But are we fully exploiting this
- Symbolic regression can help understand if a model is learning the true physics of the universe
  - Potentially help us refine physical laws
- Interpretability of GNNs is extremely under-studied in physics
  - Attention mechanisms and relevance propagation are proxies but are not precise
  - Other methods like black box methods, disentangle representation learning have not been studied
  - Central debate in ML for physics: do we care about getting the physics back (data-driven science)

Are There New Directions to Explore?

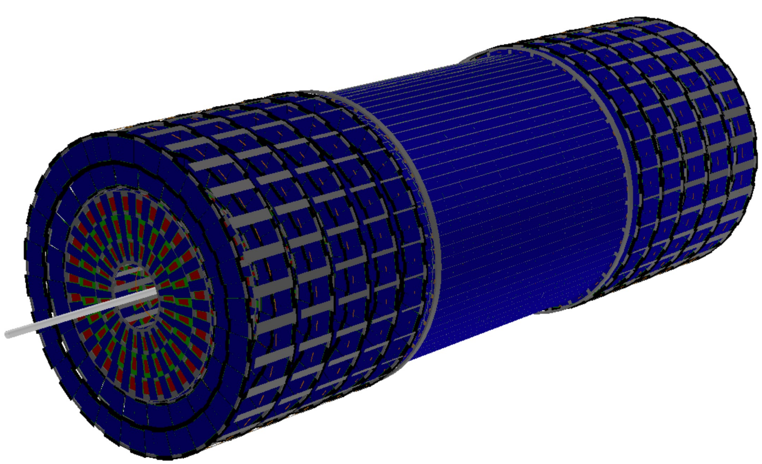
- Transformers are effective on many problem types
  - positional encoding/graph substructure models
- Study graph rewiring/nonphysical graphs/message passing only edges/information aggregating nodes
- Incorporate additional priors/inductive biases
  - Loss function constraints (number of decay products, consistency with true tracks)
  - Constraint-based GNNs
  - Graph level conservation laws
- Apply these methods to more physics tasks
  - Underexplored for simulation
  - Full hierarchical reconstruction
  - Experimental design optimization (trigger operations, detector/accelerator design)
- Represent existing problems in new ways
  - Tracking as denoising VAE or mesh generation

A Note on Datasets

- The TrackML dataset is not realistic for several reasons
- A new open data detector is nearly ready
- Can we create other benchmark/open datasets
  - Particularly that are designed for GDL
  - Even benchmark GNN models
- Always the concern of mismatch between data and simulation
  - Are there ways we can train directly on data



Full tracker view



Thank you to all participants!