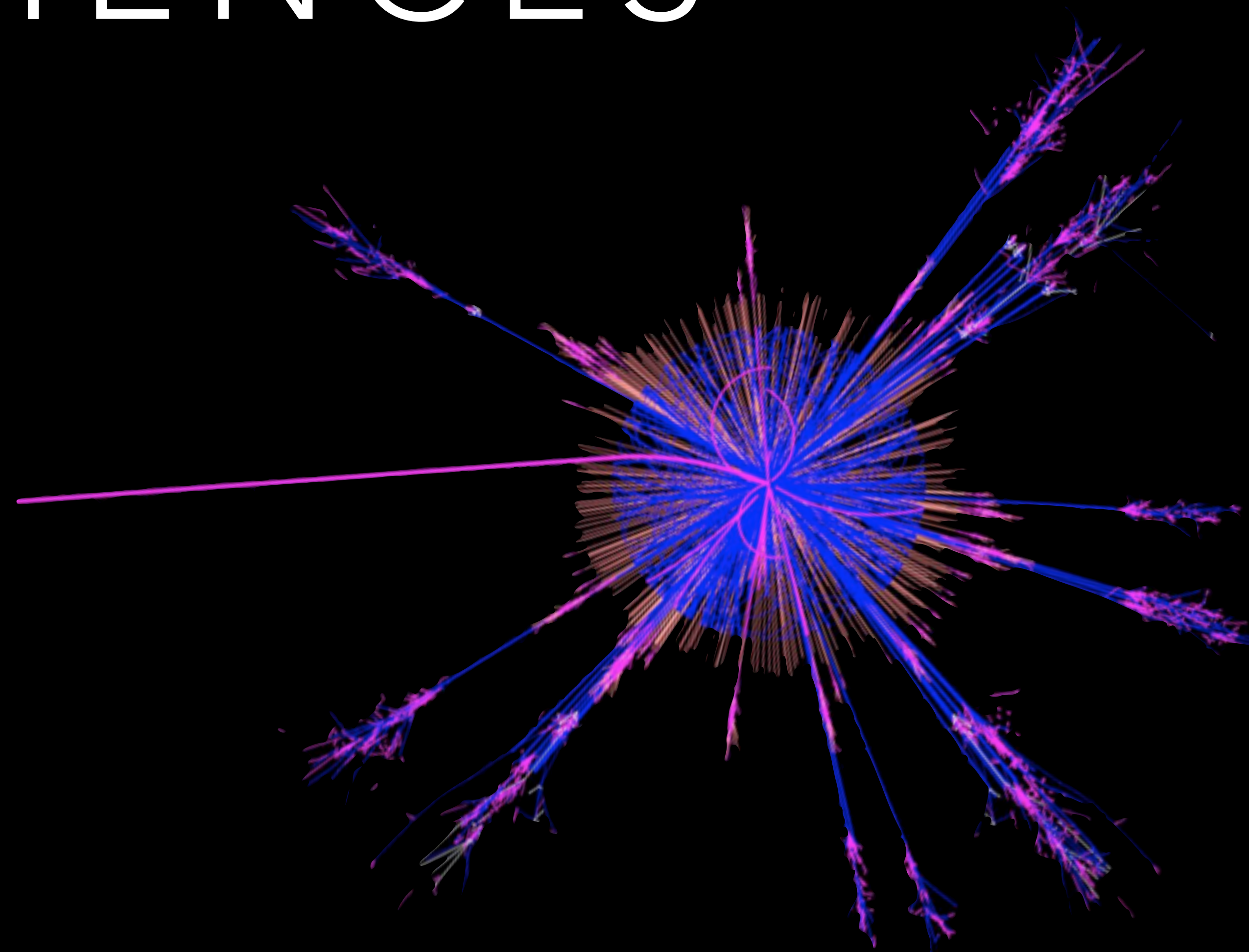
NYU Center for Data Science

Center for Cosmology and Particle Physics

DEEP LEARNING IN THE

PHYSICAL SCIENCES



@KyleCranmer
New York University
Department of Physics
Center for Data Science
CILVR Lab

Reductionist

mechanistic models
clear causal structure

descriptive models
unclear causal structure

ecology

nuclear & particle physics

astrophysics

health

documents

climate

language

cosmology

connectome

perception

lattice simulations

protein folding

psychology

quantum chemistry

systems biology

Reductionist

mechanistic models
clear causal structure

Maybe AI should start with
problems where causal structure is
clear and mechanistic models are available?

descriptive models
unclear causal structure

ecology

nuclear & particle physics

astrophysics

health

documents

climate

language

cosmology

connectome

perception

lattice simulations

protein folding

psychology

quantum chemistry

systems biology

Deep Learning for Physical Sciences

Workshop at the 31st Conference on Neural Information Processing Systems (NIPS)

December 8, 2017

PANEL DISCUSSION

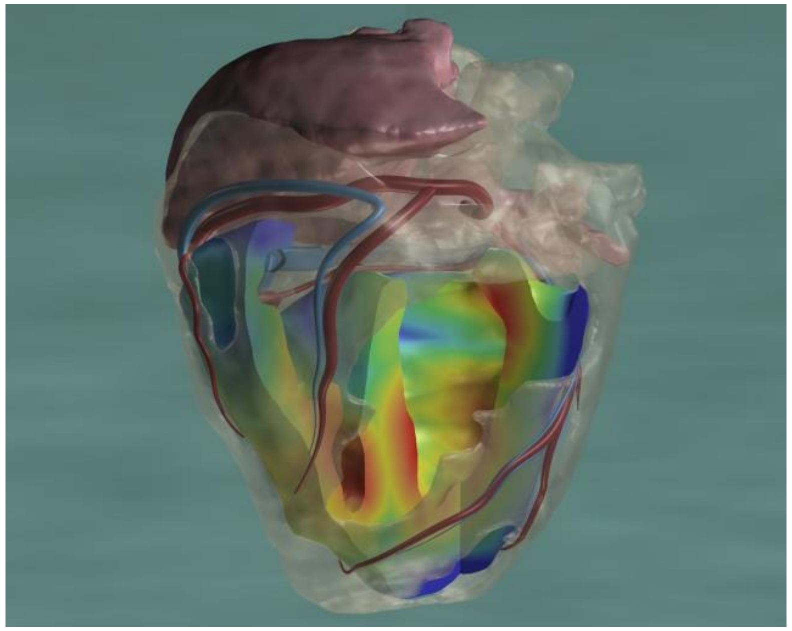
Moderator: **Kyle Cranmer** (New York University)
Iain Murray (University of Edinburgh)
Max Welling (University of Amsterdam)
Juan Carrasquilla (D-Wave Systems / Vector Institute for Artificial Intelligence)
Gilles Louppe (University of Liège)
George Dahl (Google Brain)
Anatole von Lilienfeld (University of Basel)

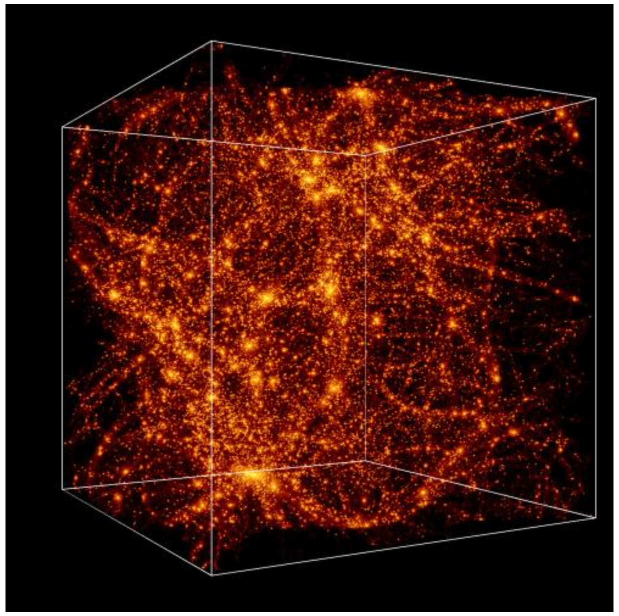
1. There is a lot of low hanging fruit, we can use M.L. to
   - improve what we normally do
   - speed up accelerate what we normally do

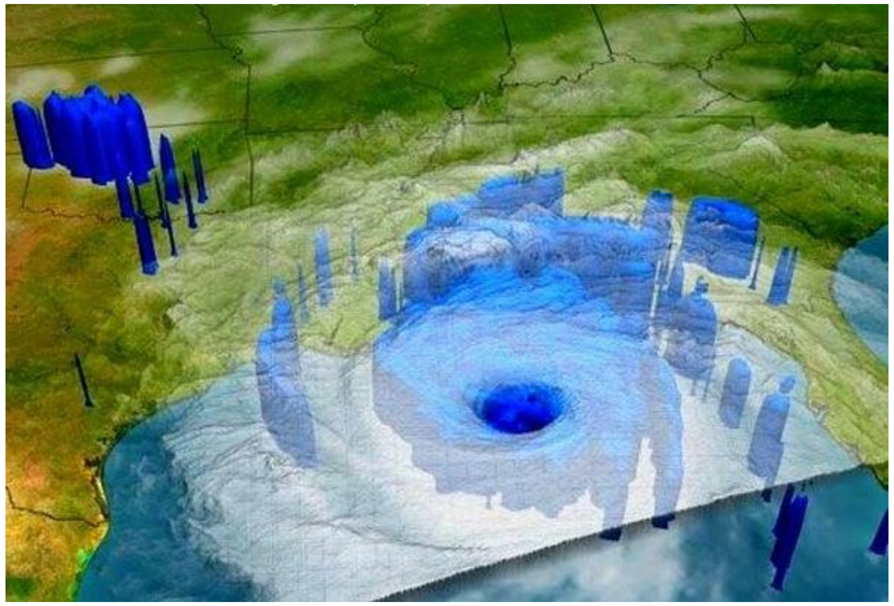
2. More profound changes to how we approach physics
   - new capabilities to be exploited
   - attack previously intractable problems

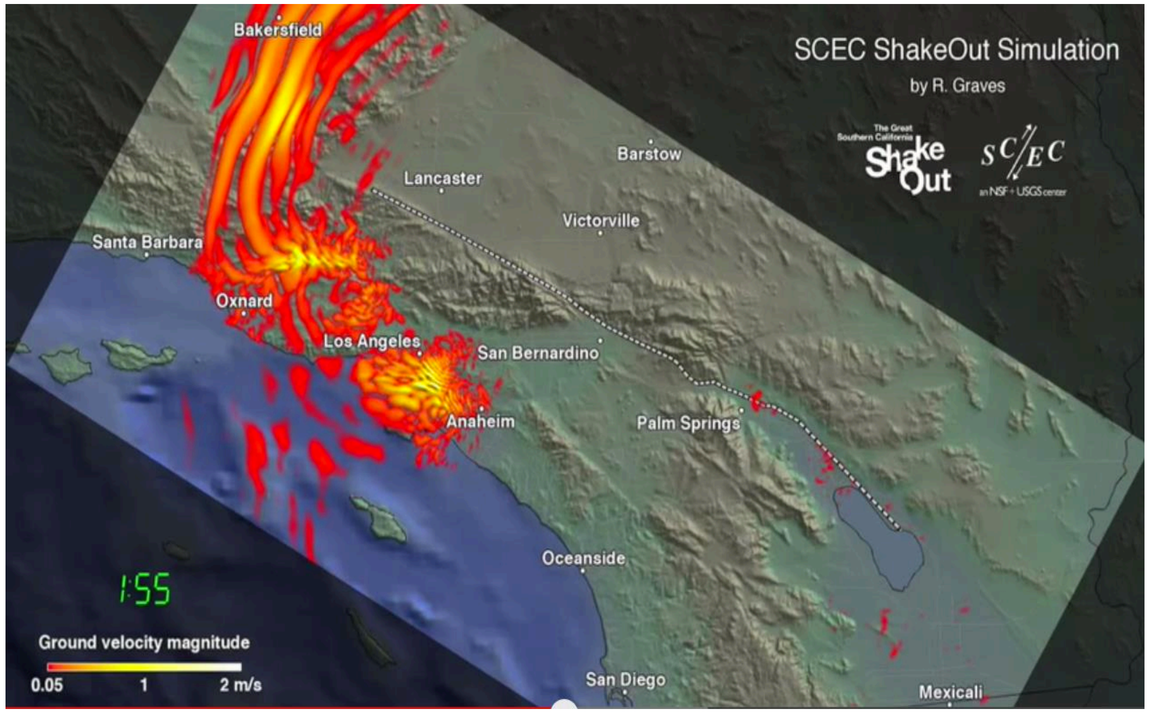
Max Welling

Generative Models: Simulators





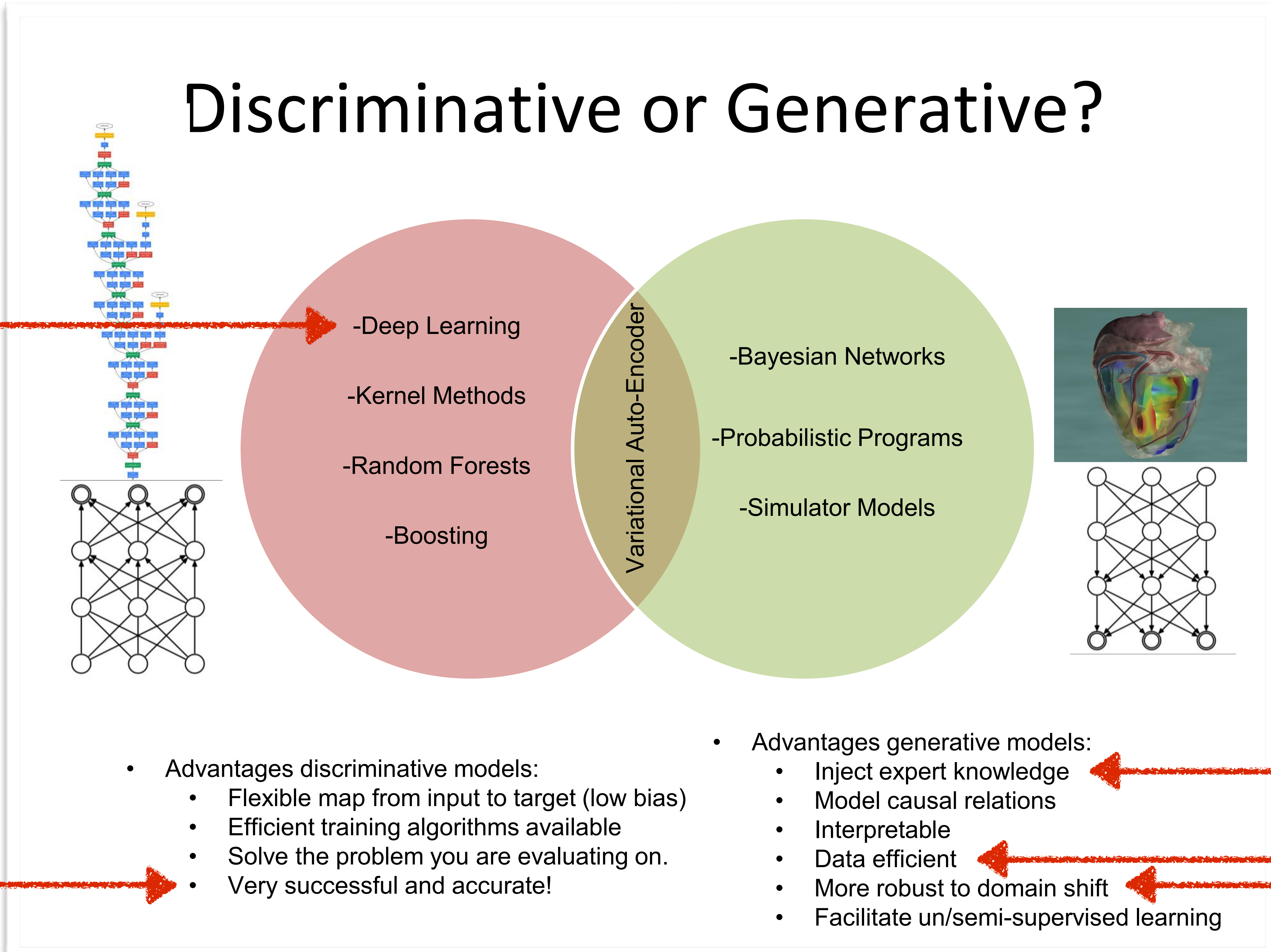




Simulators can produce labeled training data for supervised learning

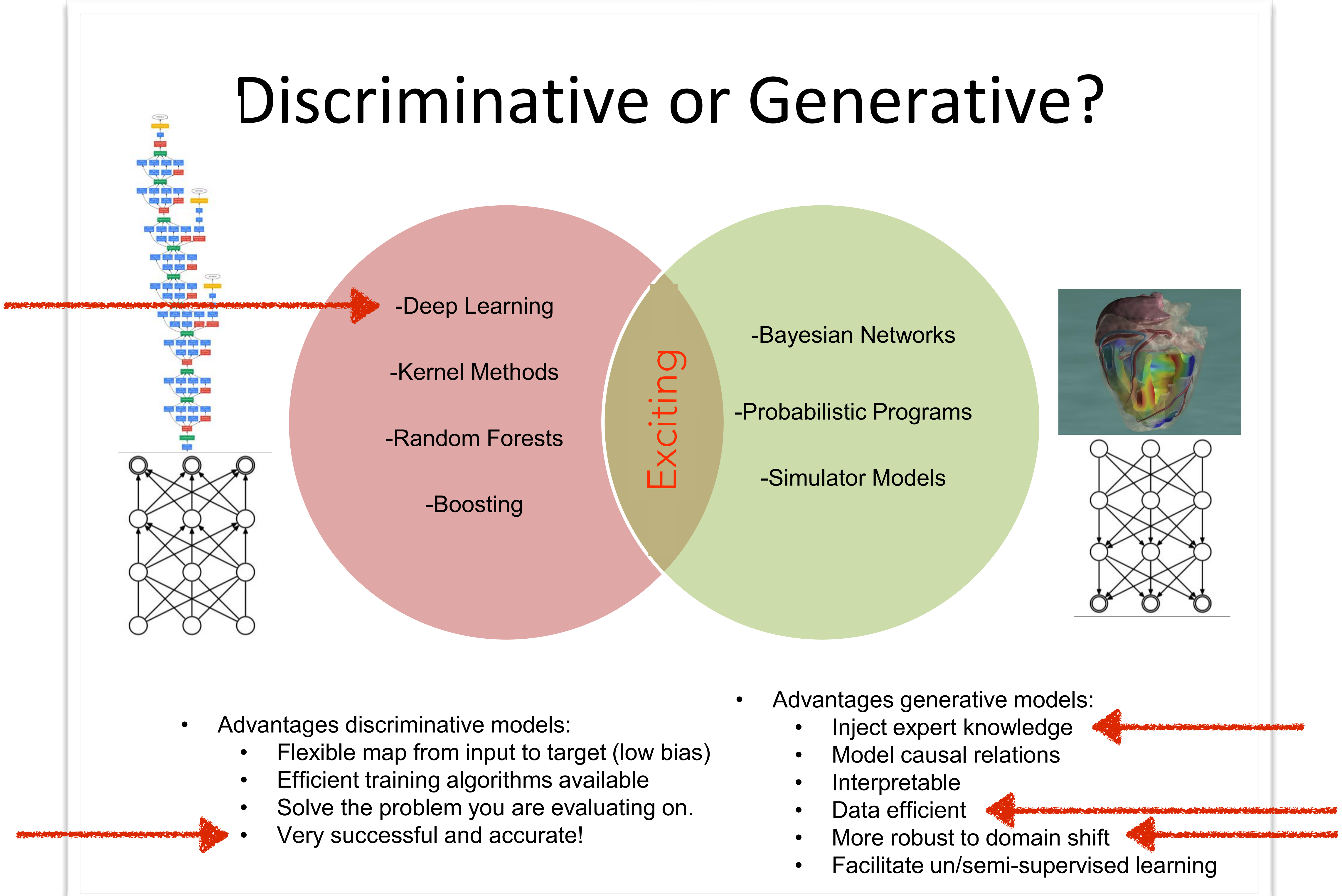
(note: some simulation are very computationally expensive)

We can leverage both the power of deep learning and inject our expert / domain knowledge



Max Welling

We can leverage both the power of deep learning and inject our expert / domain knowledge



NOTATION / TERMINOLOGY

forward modeling
generation
simulation

PREDICTION

θ
parameters of interest

$p(x, z \mid \theta, \nu)$

x
observed data
simulated data

z
latent variables

ν
nuisance parameters

INFERENCE

inverse problem
measurement
parameter estimation

Quiz:

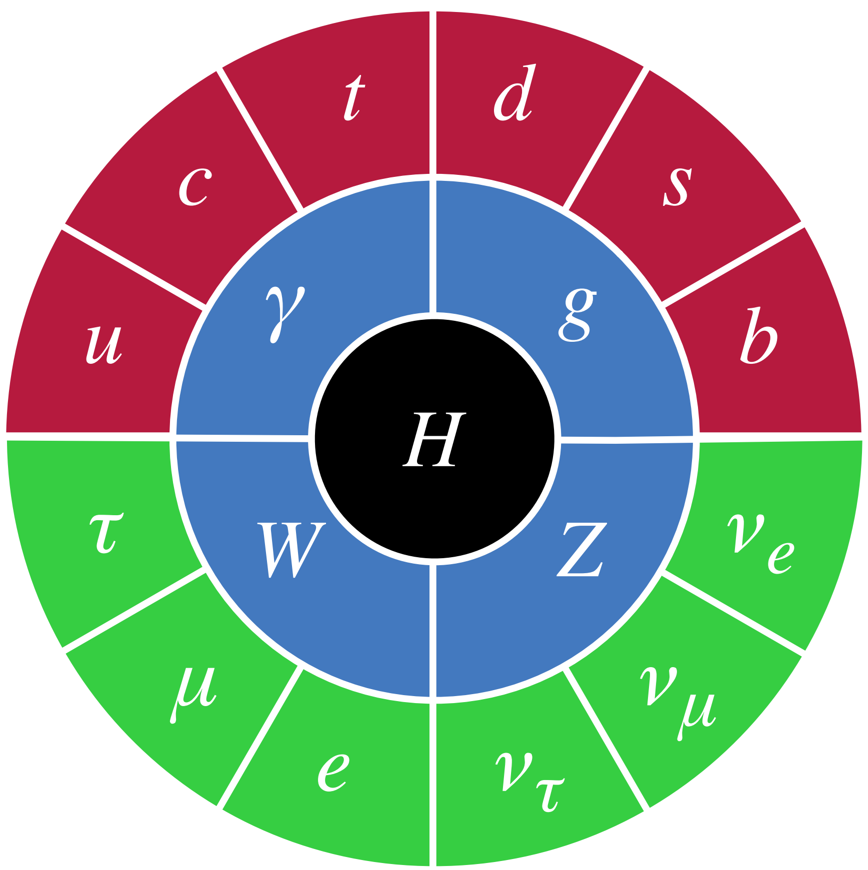
The Standard Model has 19 parameters.

The LHC has collected 10^{15} collisions.

Is this a parametric or non-parametric problem?

PARTICLE PHYSICS: 19 PARAMETERS

$$\begin{aligned}\mathcal{L}_{SM} = {} & \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^a_{\mu\nu}G^{\mu\nu}_a}_{\text{kinetic energies and self-interactions of the gauge bosons}} \\ & + \underbrace{\bar{L}\gamma^\mu(i\partial_\mu - \frac{1}{2}g\tau\cdot\mathbf{W}_\mu - \frac{1}{2}g'YB_\mu)L + \bar{R}\gamma^\mu(i\partial_\mu - \frac{1}{2}g'YB_\mu)R}_{\text{kinetic energies and electroweak interactions of fermions}} \\ & + \underbrace{\frac{1}{2}\left|(i\partial_\mu - \frac{1}{2}g\tau\cdot\mathbf{W}_\mu - \frac{1}{2}g'YB_\mu)\,\phi\right|^2 - V(\phi)}_{W^\pm, Z, \gamma, \text{and Higgs masses and couplings}} \\ & + \underbrace{g''(\bar{q}\gamma^\mu T_a q)\,G^a_\mu}_{\text{interactions between quarks and gluons}} + \underbrace{(G_1\bar{L}\phi R + G_2\bar{L}\phi_c R + h.c.)}_{\text{fermion masses and couplings to Higgs}}\end{aligned}$$



Symbol	Description	Value
m_e	Electron mass	511 keV
m_μ	Muon mass	105.7 MeV
m_τ	Tau mass	1.78 GeV
m_u	Up quark mass	1.9 MeV
m_d	Down quark mass	4.4 MeV
m_s	Strange quark mass	87 MeV
m_c	Charm quark mass	1.32 GeV
m_b	Bottom quark mass	4.24 GeV
m_t	Top quark mass	172.7 GeV
θ_{12}	CKM 12-mixing angle	13.1°
θ_{23}	CKM 23-mixing angle	2.4°
θ_{13}	CKM 13-mixing angle	0.2°
δ	CKM CP-violating Phase	0.995
g_1	U(1) gauge coupling	0.357
g_2	SU(2) gauge coupling	0.652
g_3	SU(3) gauge coupling	1.221
θ_{QCD}	QCD vacuum angle	~0
v	Higgs vacuum expectation value	246 GeV
m_H	Higgs mass	125 GeV

$$
\begin{aligned}
\mathcal{L}_{SM} = \; & \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}} \\
& + \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L \; + \; \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}} \\
& + \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\,\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma,\text{and Higgs masses and couplings}} \\
& + \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)\,G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} \; + \; \underbrace{(G_{1}\bar{L}\phi R + G_{2}\bar{L}\phi_{c}R + h.c.)}_{\text{fermion masses and couplings to Higgs}}
\end{aligned}
$$

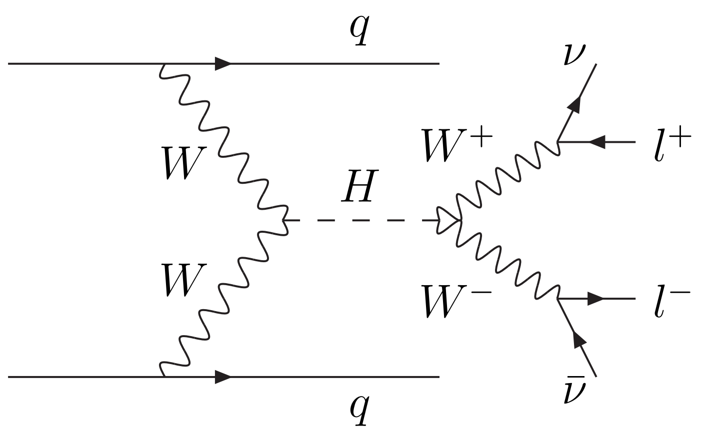
1) We begin with Quantum Field Theory

PREDICTION IN PARTICLE PHYSICS

$$
\begin{aligned}
\mathcal{L}_{SM} = \quad & \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}} \\
+ \quad & \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L \;+\; \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}} \\
+ \quad & \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\,\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma,\text{and Higgs masses and couplings}} \\
+ \quad & \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)\,G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} \;+\; \underbrace{(G_{1}\bar{L}\phi R + G_{2}\bar{L}\phi_{c}R + h.c.)}_{\text{fermion masses and couplings to Higgs}}
\end{aligned}
$$

1) We begin with Quantum Field Theory

2) Theory gives detailed prediction for high-energy collisions



hierarchical: 2 → O(10) → O(100) particles

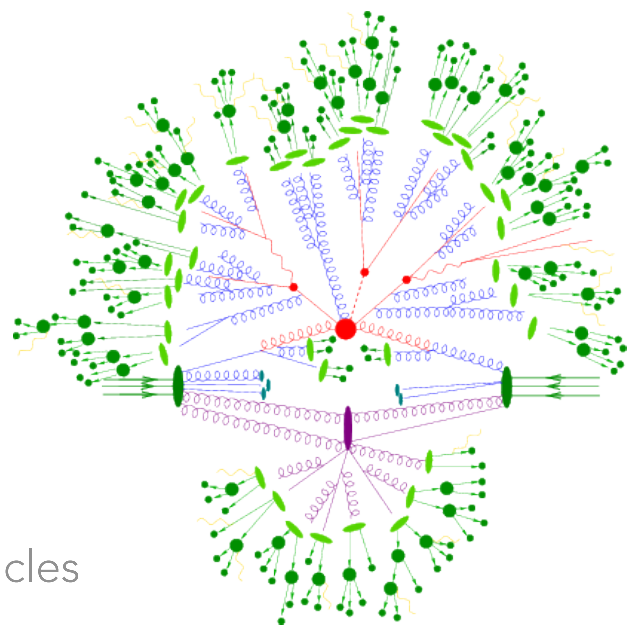
PREDICTION IN PARTICLE PHYSICS

$$
\begin{aligned}
\mathcal{L}_{SM} = \; & \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^a_{\mu\nu}G^{\mu\nu}_a}_{\text{kinetic energies and self-interactions of the gauge bosons}} \\
& + \underbrace{\bar{L}\gamma^\mu(i\partial_\mu - \frac{1}{2}g\tau\cdot\mathbf{W}_\mu - \frac{1}{2}g'YB_\mu)L + \bar{R}\gamma^\mu(i\partial_\mu - \frac{1}{2}g'YB_\mu)R}_{\text{kinetic energies and electroweak interactions of fermions}} \\
& + \underbrace{\frac{1}{2}\left|(i\partial_\mu - \frac{1}{2}g\tau\cdot\mathbf{W}_\mu - \frac{1}{2}g'YB_\mu)\,\phi\right|^2 - V(\phi)}_{W^\pm, Z, \gamma, \text{and Higgs masses and couplings}} \\
& + \underbrace{g''(\bar{q}\gamma^\mu T_a q)\,G^a_\mu}_{\text{interactions between quarks and gluons}} + \underbrace{(G_1\bar{L}\phi R + G_2\bar{L}\phi_c R + h.c.)}_{\text{fermion masses and couplings to Higgs}}
\end{aligned}
$$

1) We begin with Quantum Field Theory

2) Theory gives detailed prediction for high-energy collisions

hierarchical: 2 → O(10) → O(100) particles



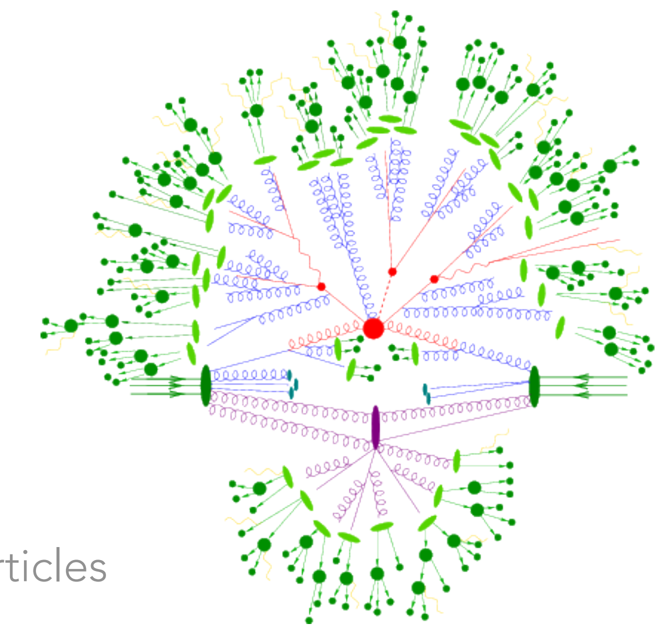
PREDICTION IN PARTICLE PHYSICS

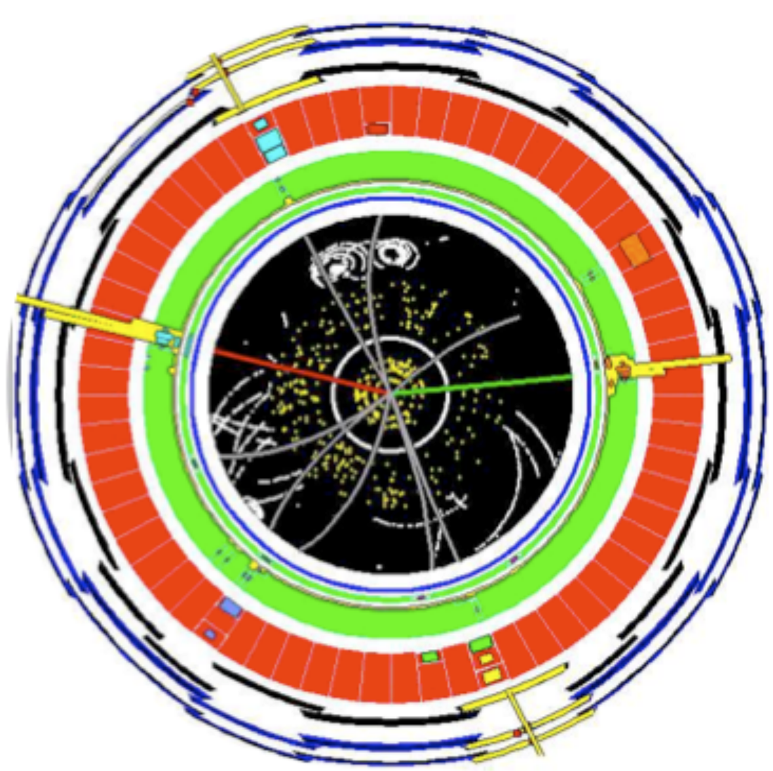
$$
\begin{aligned}
\mathcal{L}_{SM} = &\underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}} \\
&+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}} \\
&+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\,\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma,\text{and Higgs masses and couplings}} \\
&+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)\,G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_{1}\bar{L}\phi R + G_{2}\bar{L}\phi_{c}R + h.c.)}_{\text{fermion masses and couplings to Higgs}}
\end{aligned}
$$

1) We begin with Quantum Field Theory

2) Theory gives detailed prediction for high-energy collisions

hierarchical: 2 → O(10) → O(100) particles





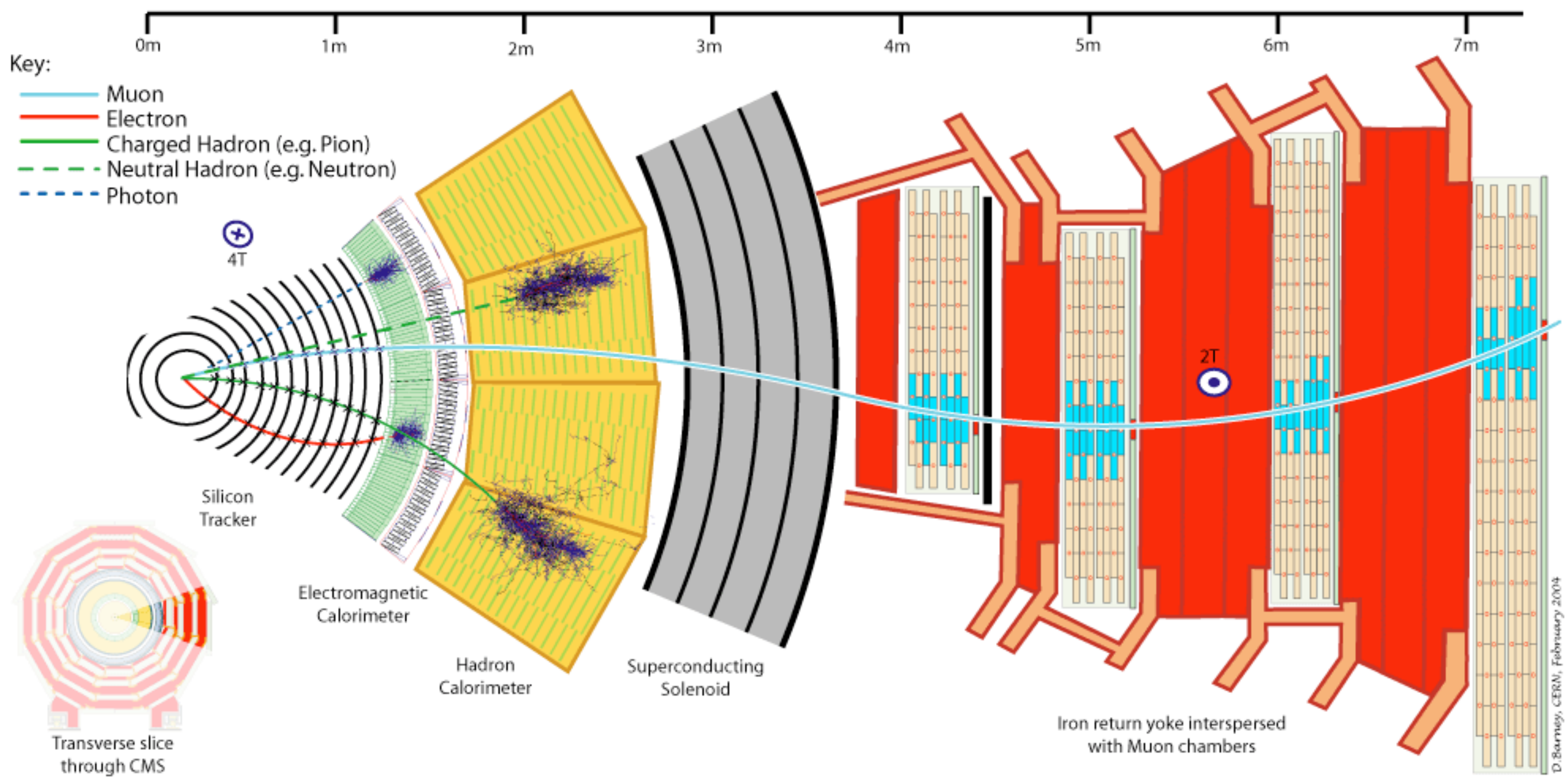
3) The interaction of outgoing particles with the detector is simulated.

>100 million sensors

Conceptually: Prob(detector response | particles)

Implementation: Monte Carlo integration over micro-physics

Consequence: evaluation of the likelihood is intractable



Conceptually: Prob(detector response | particles)

Implementation: Monte Carlo integration over micro-physics

Consequence: evaluation of the likelihood is intractable

The Crux:

observed

Monte Carlo Sampling

what happened in simulation

$$p(x|\theta) = \boxed{\int dz}\, p(x, z|\theta)$$

PARAMETRIC VS. NON-PARAMETRIC

PARAMETRIC VS. NON-PARAMETRIC

Parametric:

- num parameters < num data points
- model is highly constrained & tractable

PARAMETRIC VS. NON-PARAMETRIC

Parametric:

- num parameters < num data points
- model is highly constrained & tractable

Non-Parametric

- num parameters > num data points
- model is very flexible, but tractable

PARAMETRIC VS. NON-PARAMETRIC

Parametric:

- num parameters < num data points
- model is highly constrained & tractable

Non-Parametric

- num parameters > num data points
- model is very flexible, but tractable

Implicit Models / Simulation-based inference / Likelihood-free inference

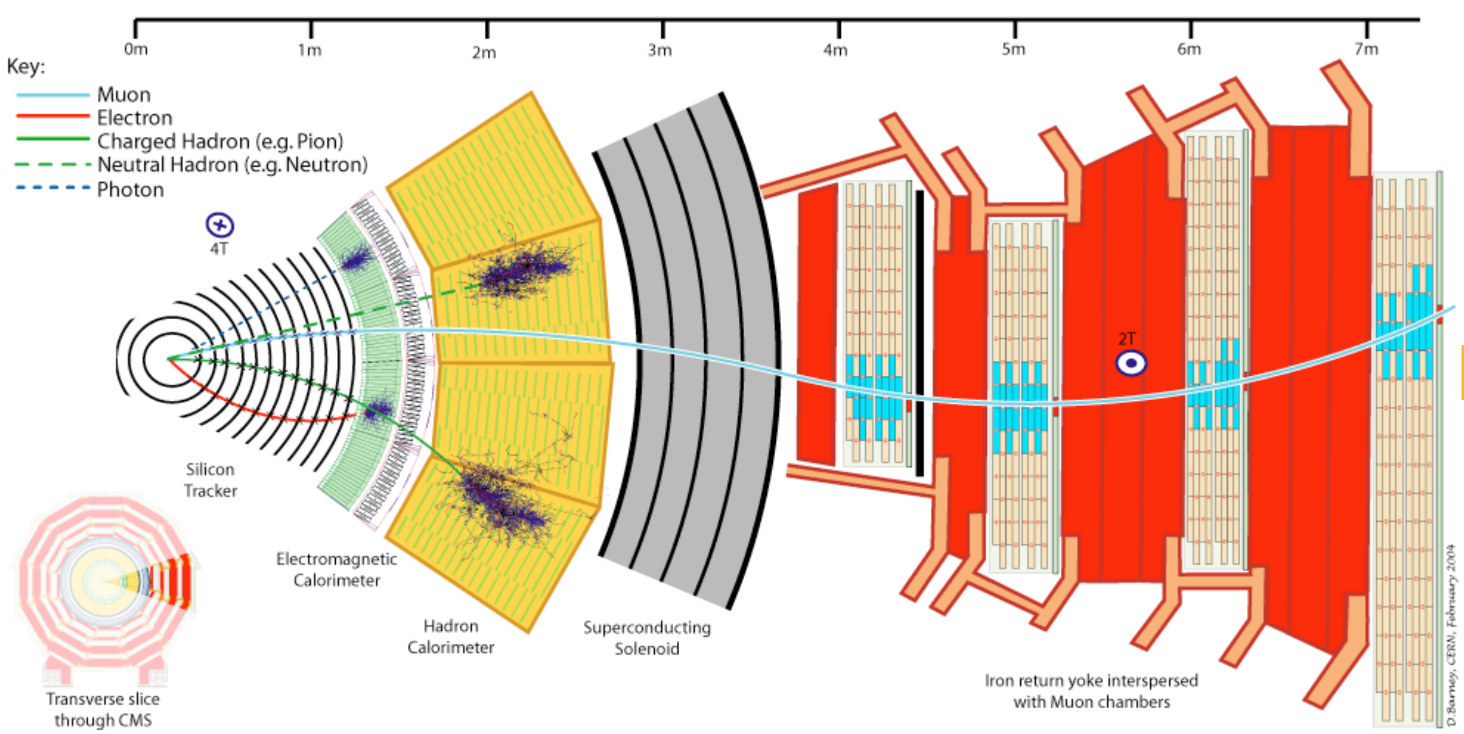
- num parameters of simulator < num data points
- **but** data distribution is very complicated and density is intractable
  - hard to identify the relevant “degrees of freedom” in the data (sufficient statistics)
- model is highly constrained, **but** hard to leverage that structure
  - **deep learning can help! learn a surrogate that captures relevant aspects of $p(x|\theta)$**

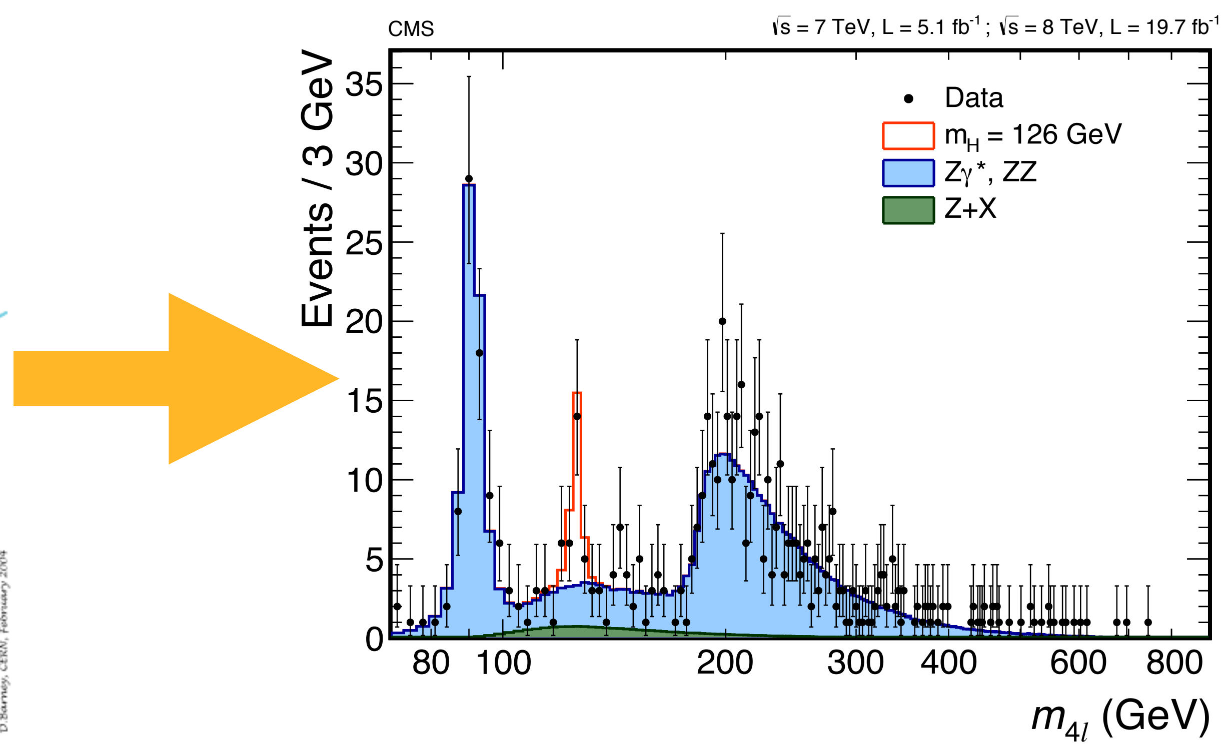
The Traditional Approach

10^8 SENSORS → 1 REAL-VALUED QUANTITY

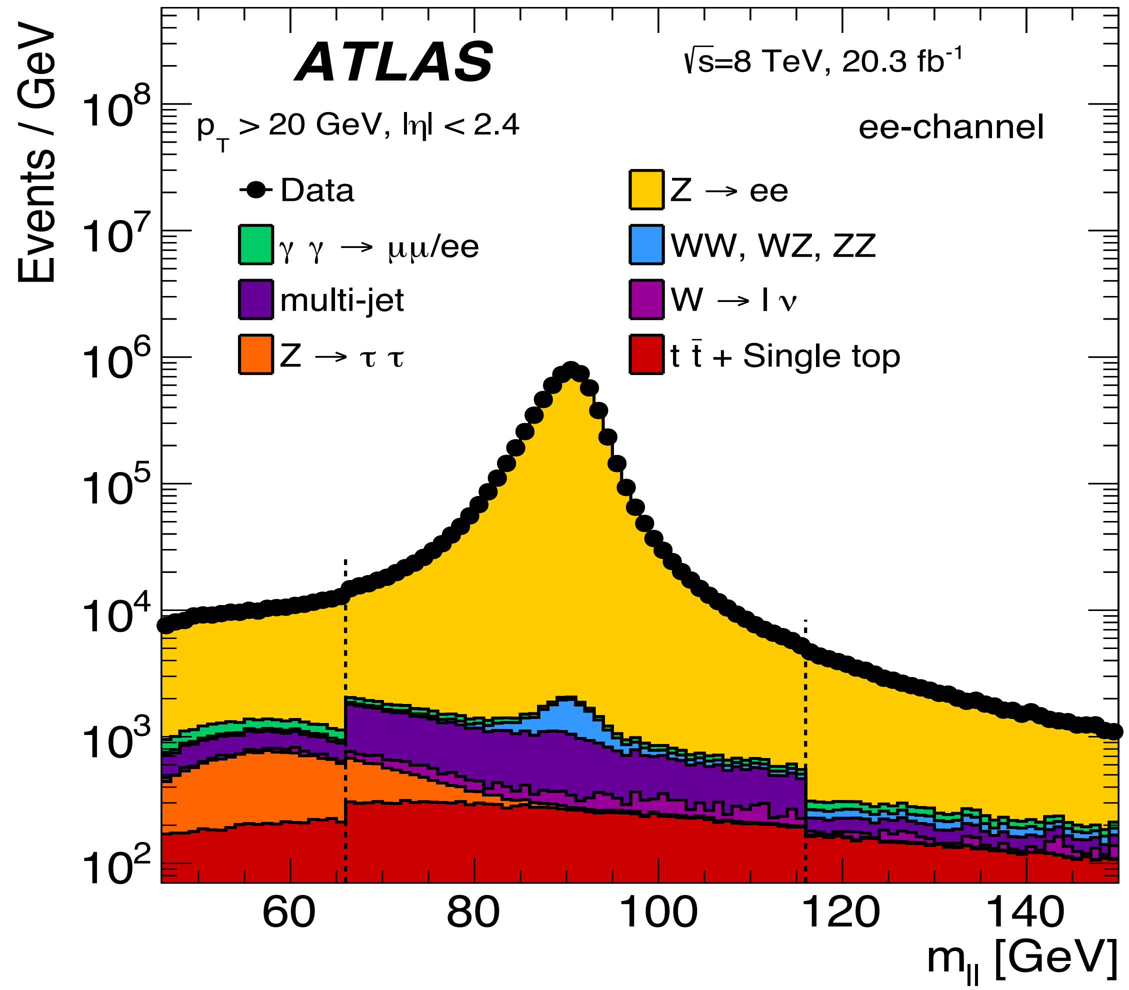
Most measurements and searches for new particles at the LHC are based on the distribution of a single variable / feature / summary statistic

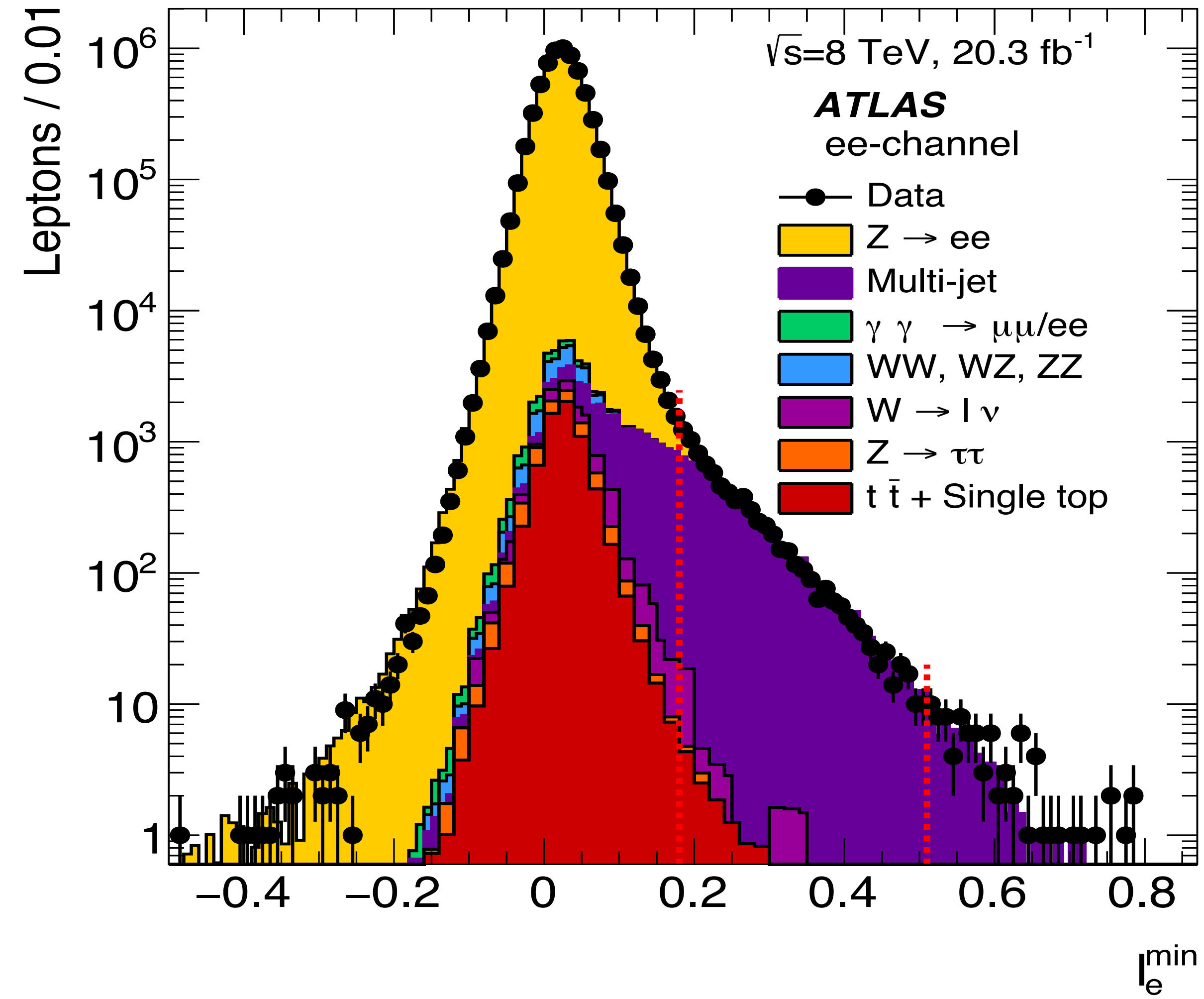
- choosing a good variable (feature engineering) is a task for a skilled physicist and tailored to the goal of measurement or new particle search
- likelihood p(x|θ) **approximated** using histograms (univariate density estimation)

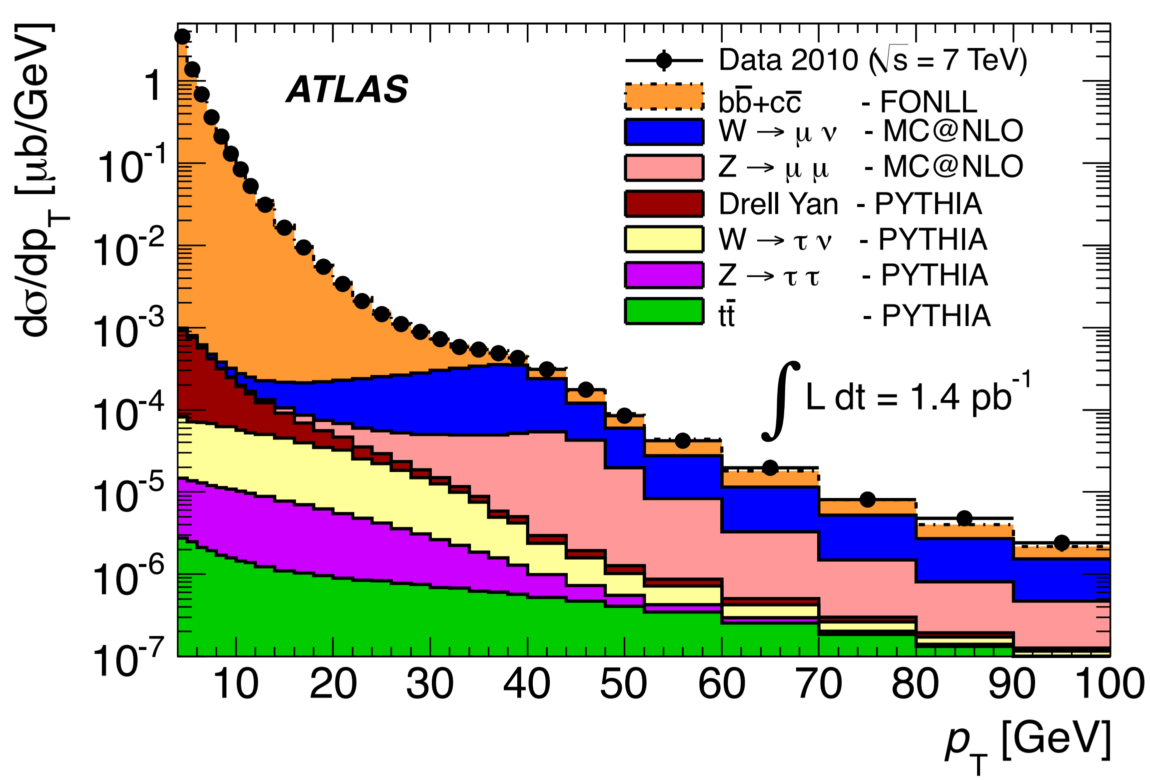


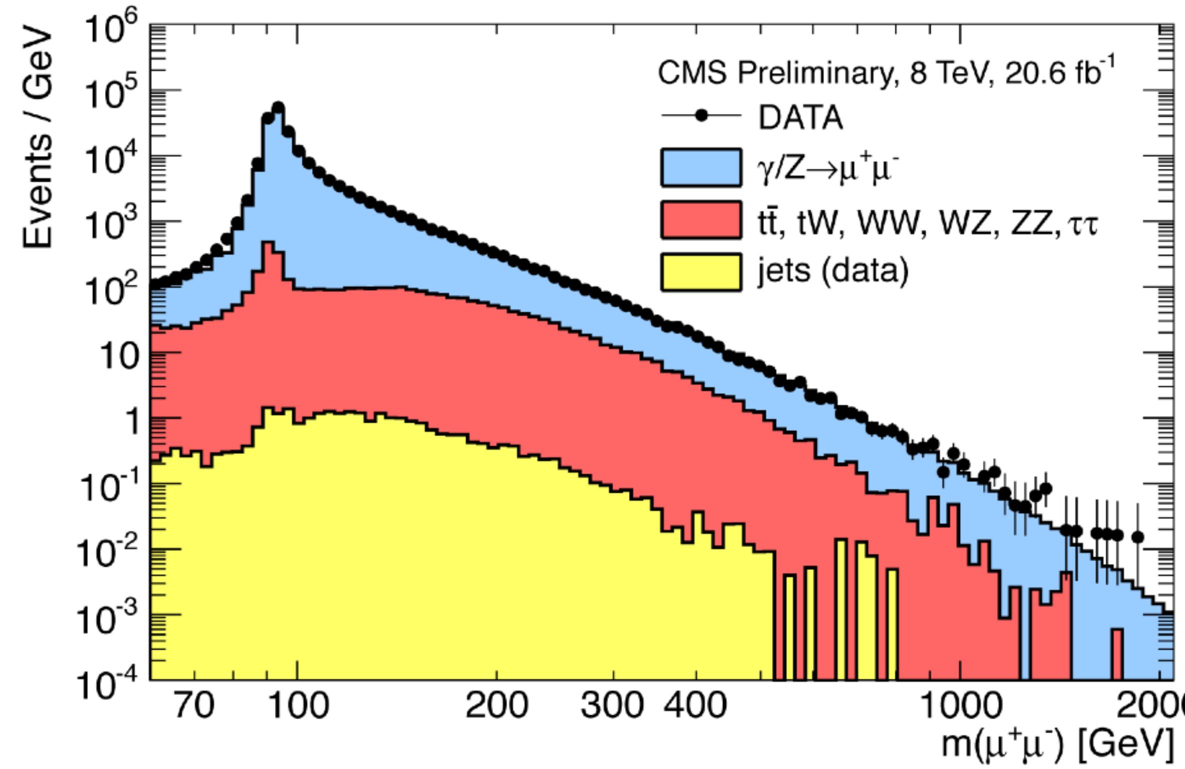


HIGH FIDELITY SIMULATION

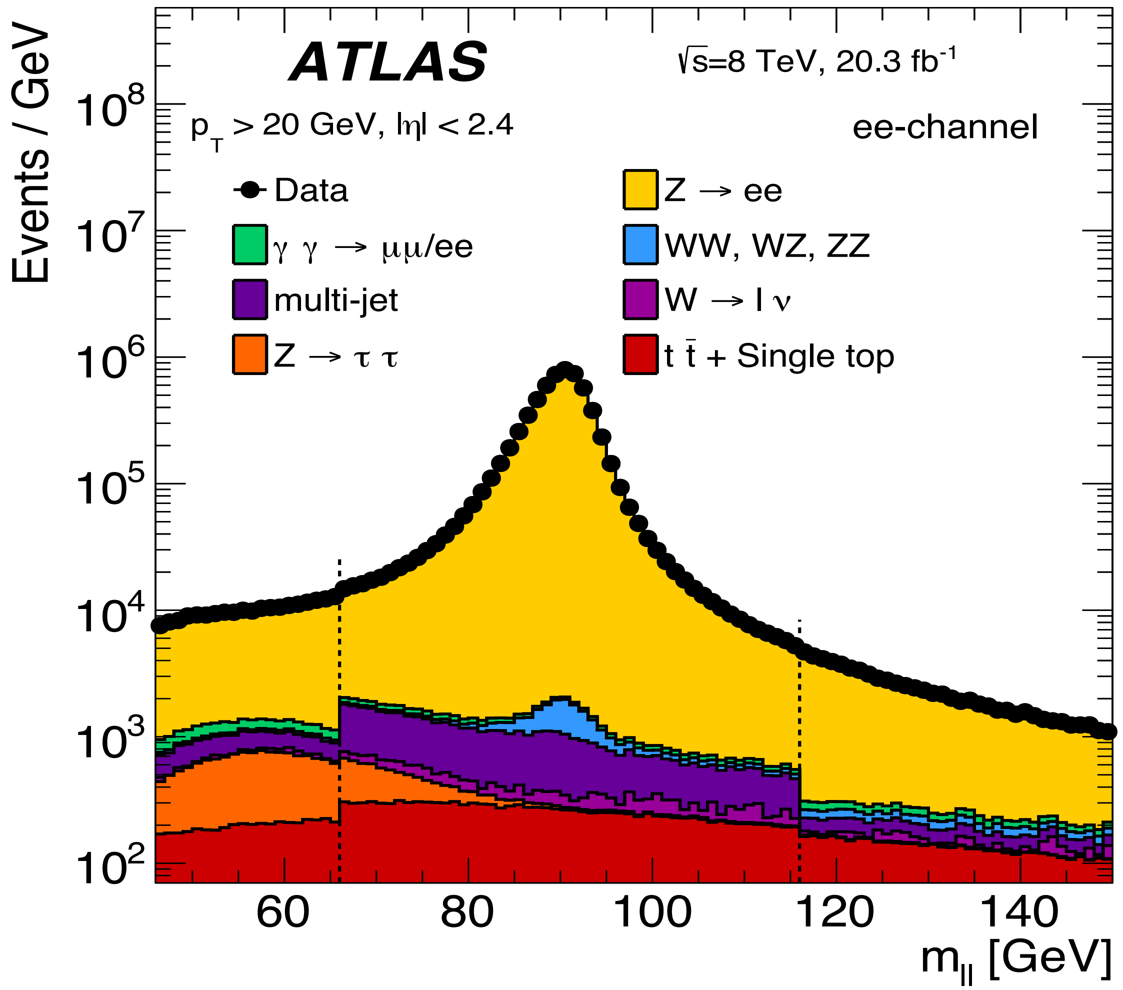


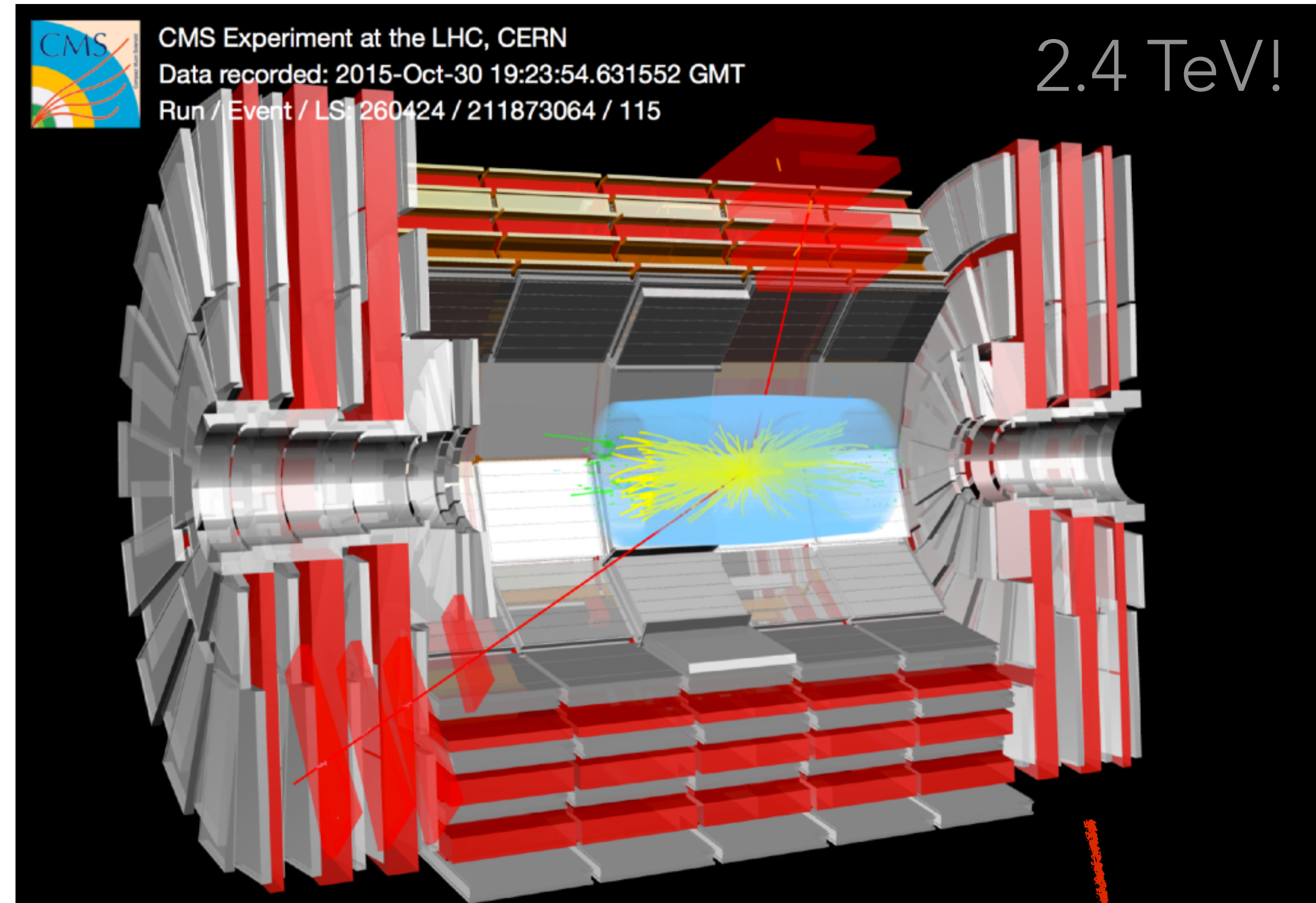


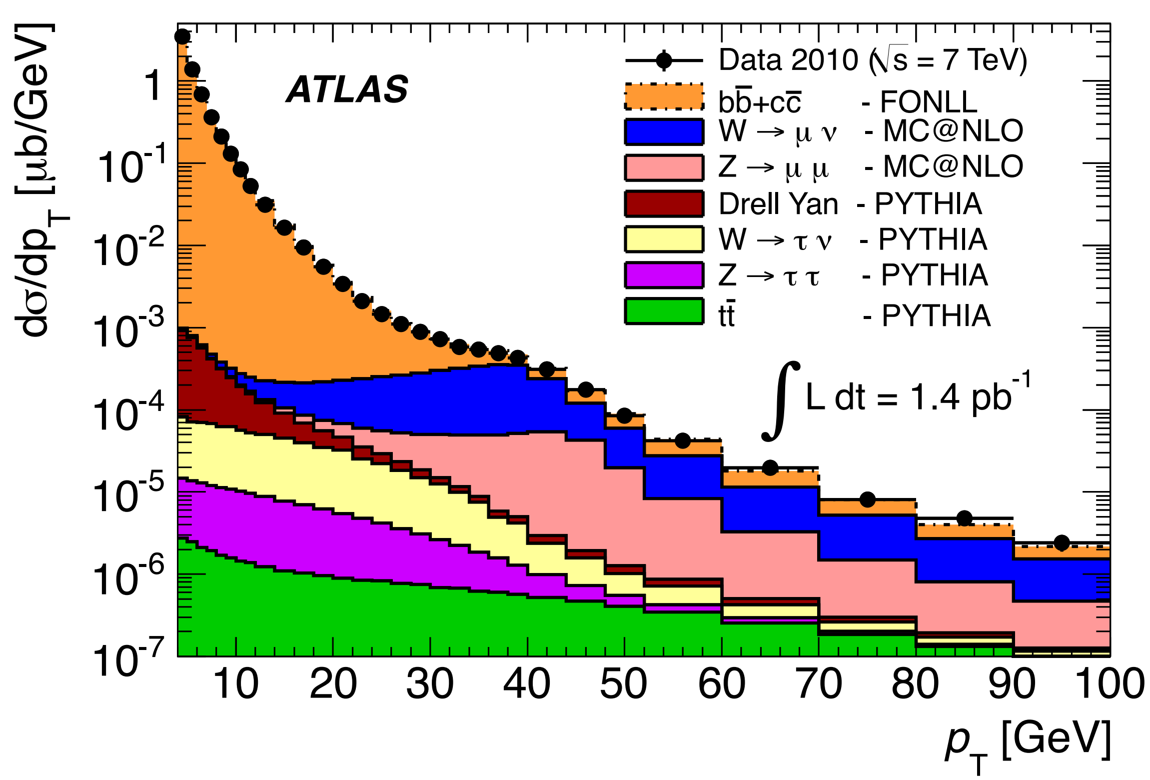


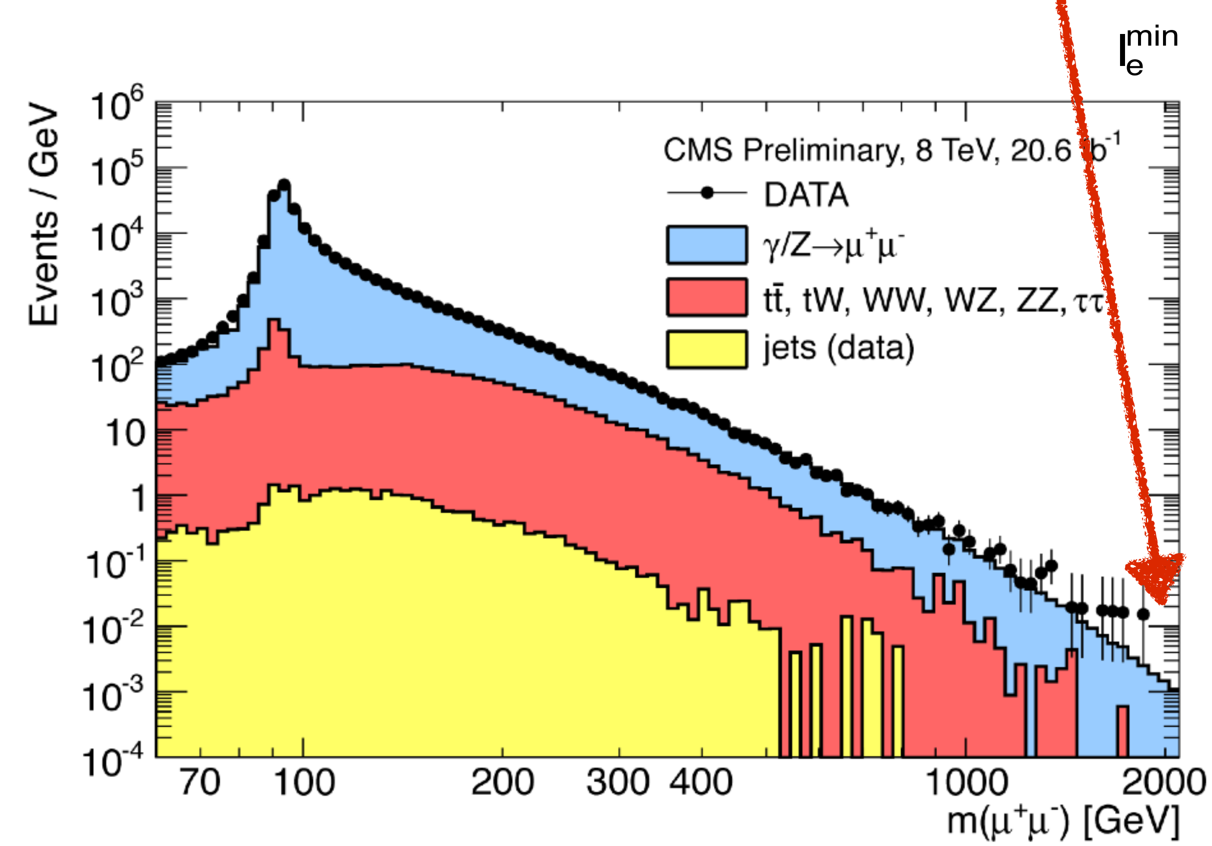


HIGH FIDELITY SIMULATION



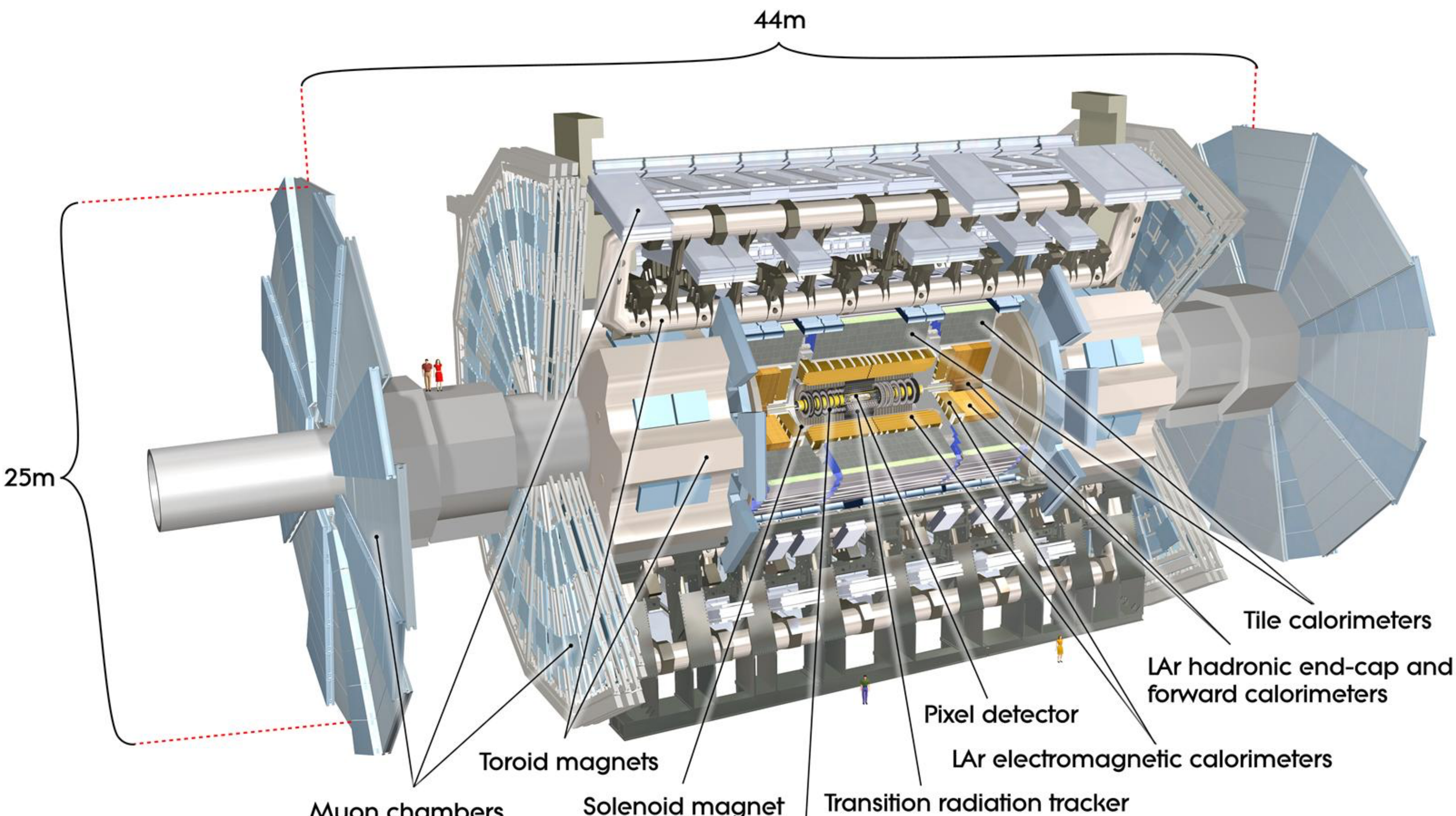






Detector is 44m long

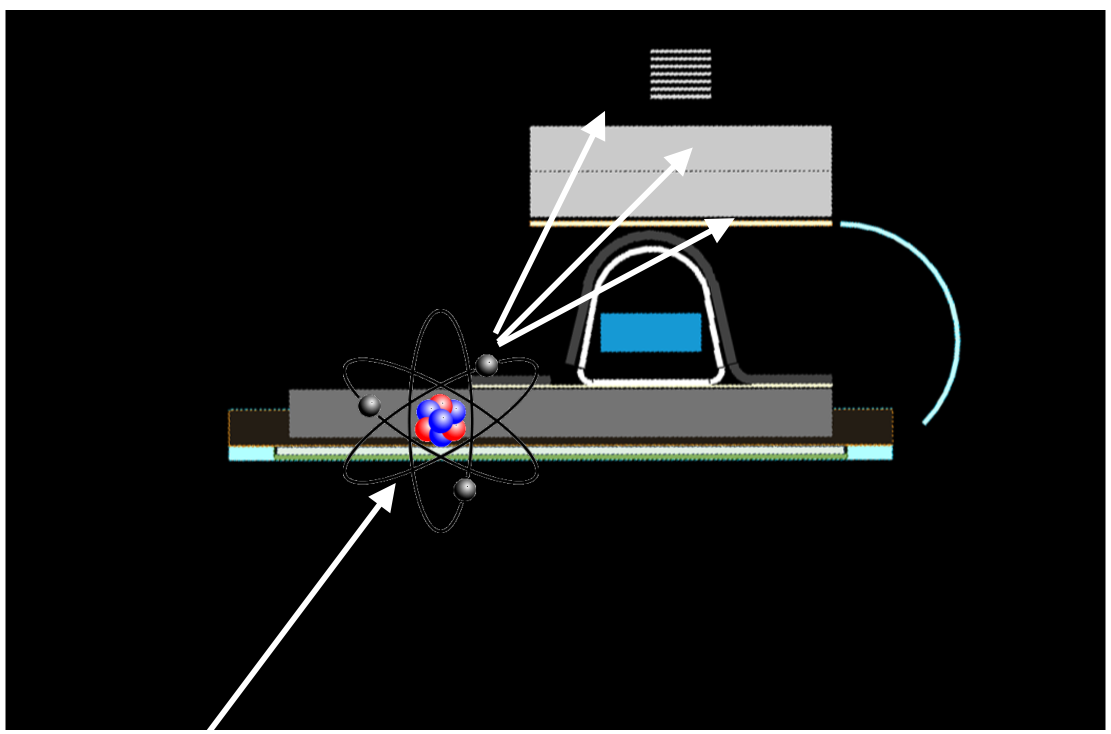
- Detector resolves details at <mm scale; Simulation accurate!



HIGH FIDELITY SIMULATION

Detector is 44m long

- Detector resolves details at <mm scale; Simulation accurate!



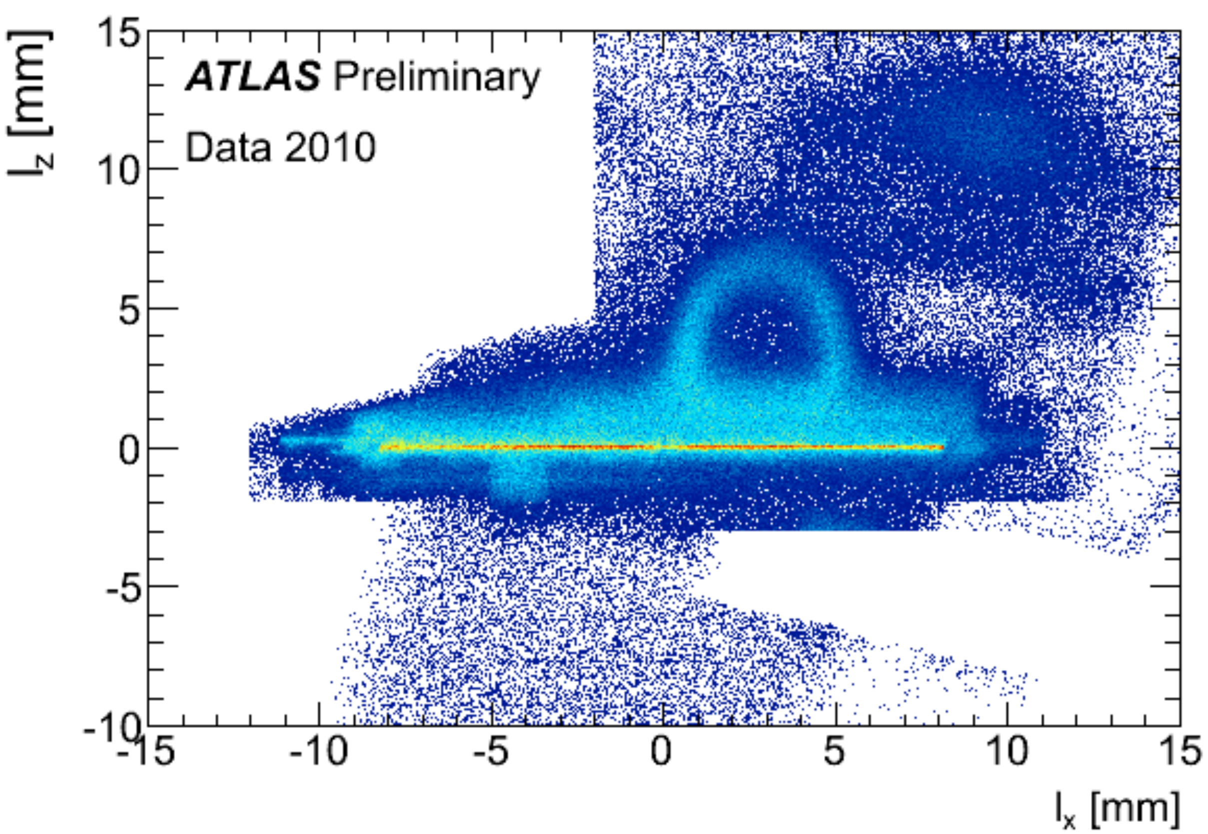
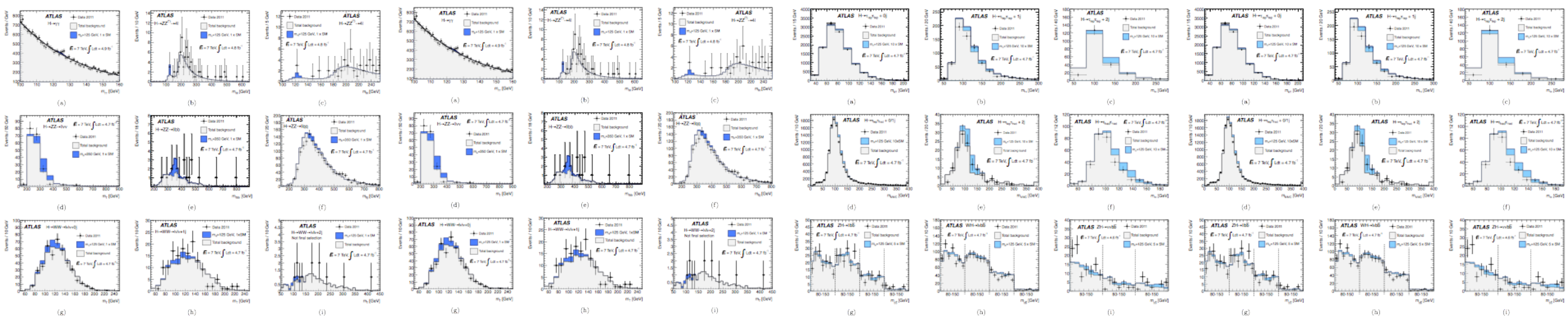
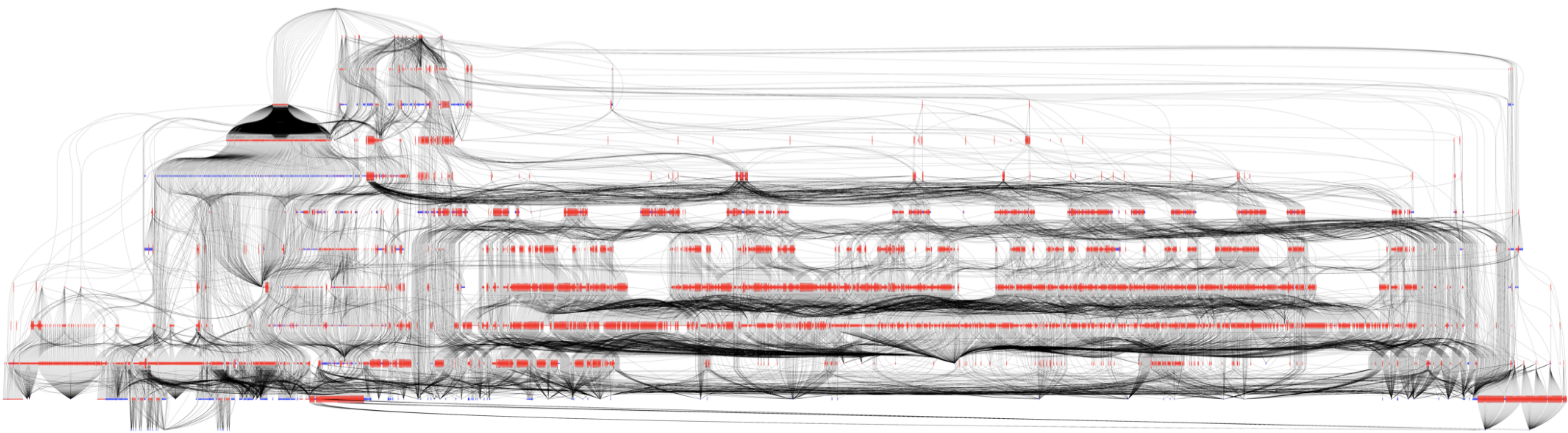


Figure:
ATLAS pixel model as described in simulation (left), tomography from vertices built from tracks for hadronic interactions (right)

Slide Credit: A. Salzburger (CERN)

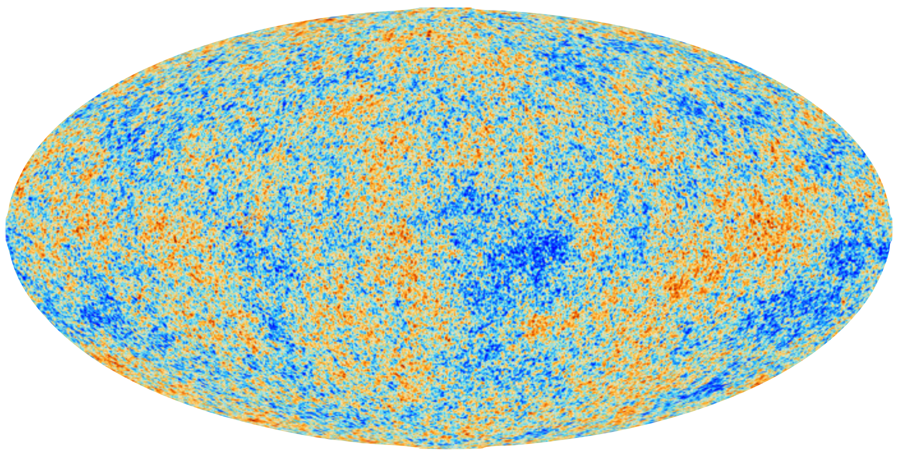
THE DISCOVERY OF THE HIGGS BOSON

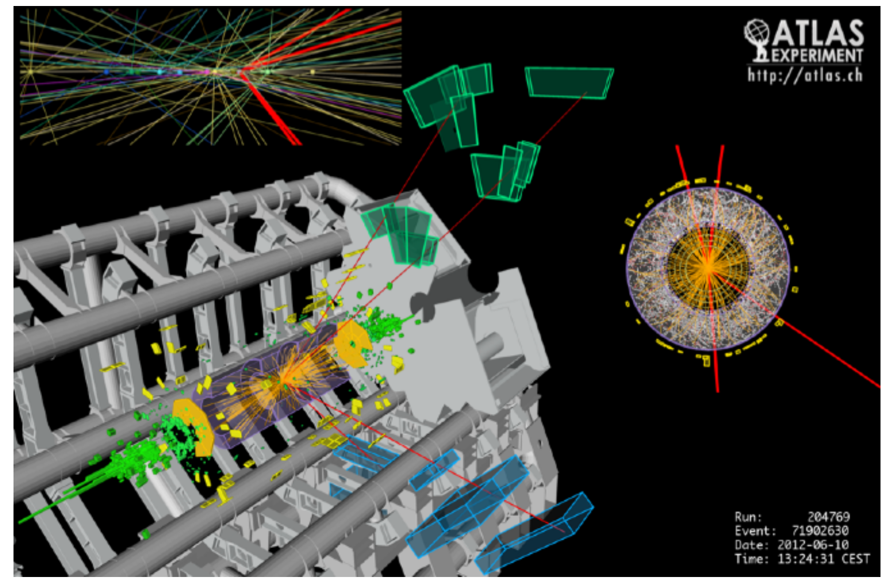


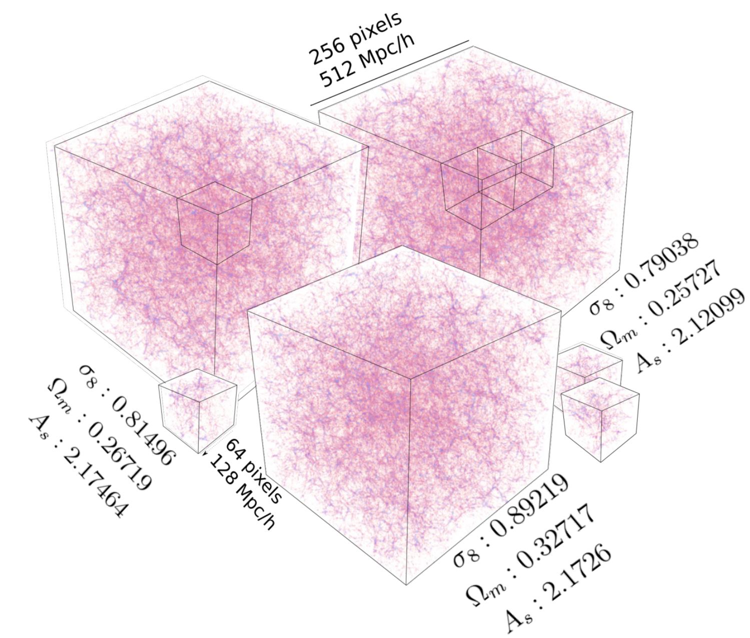


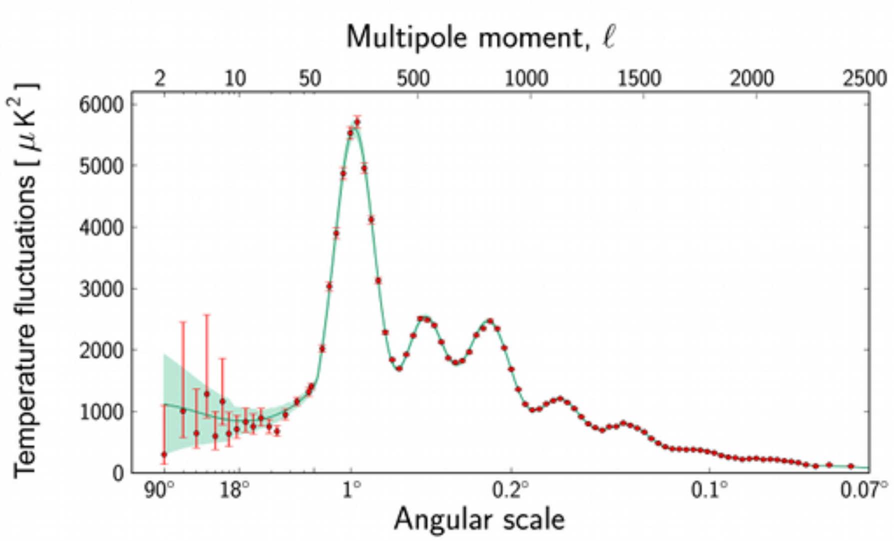
$$\mathbf{f}_{\rm tot}(\mathcal{D}_{\rm sim}, \mathcal{G}|\boldsymbol{\alpha}) = \prod_{c\in\text{channels}} \left[\text{Pois}(n_c|\nu_c(\boldsymbol{\alpha})) \prod_{e=1}^{n_c} f_c(x_{ce}|\boldsymbol{\alpha})\right] \cdot \prod_{p\in\mathbb{S}} f_p(a_p|\alpha_p)$$

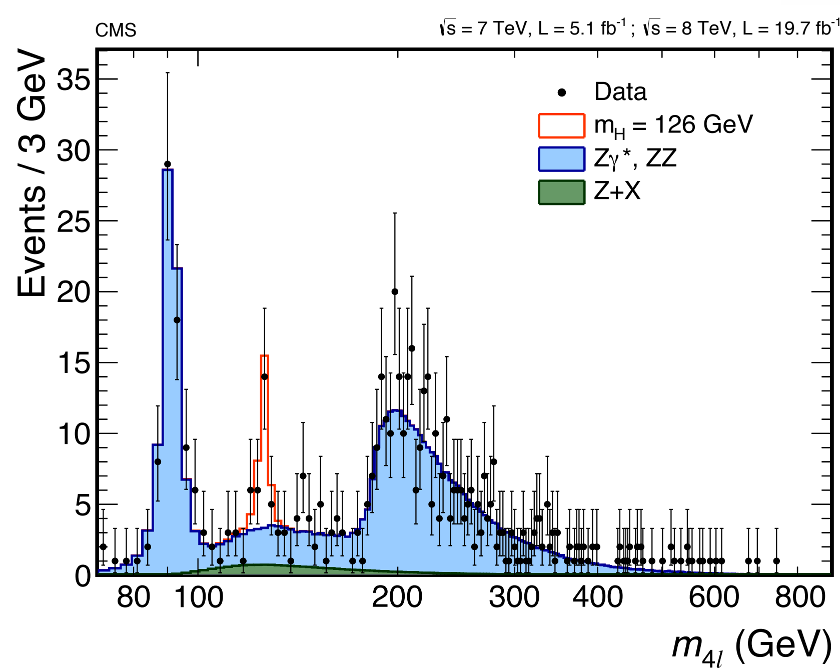
SIMULATION → TEMPLATE

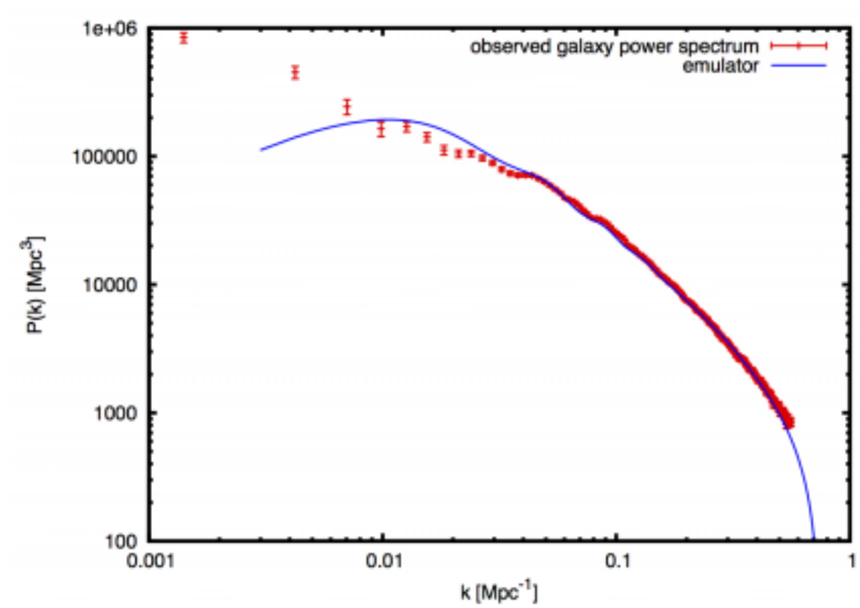












The Problem:

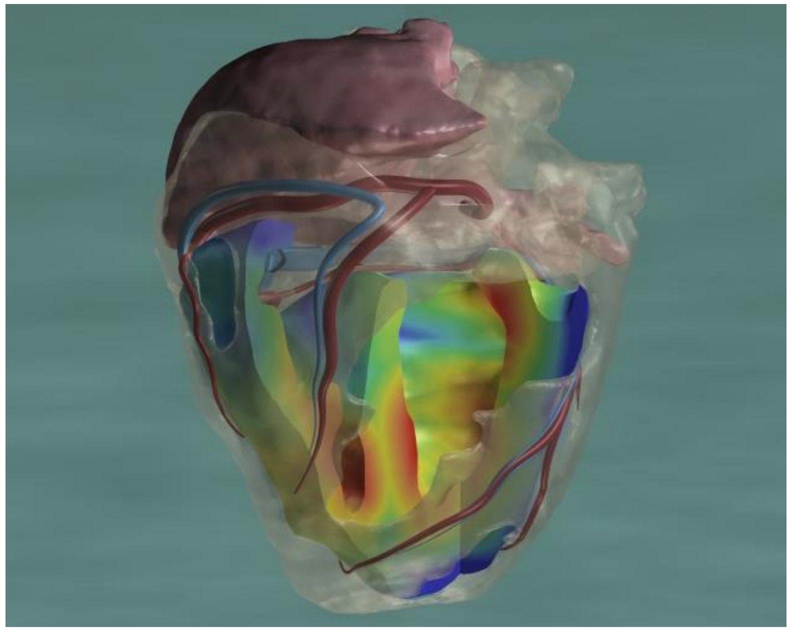
This doesn't scale if **x** is high dimensional!

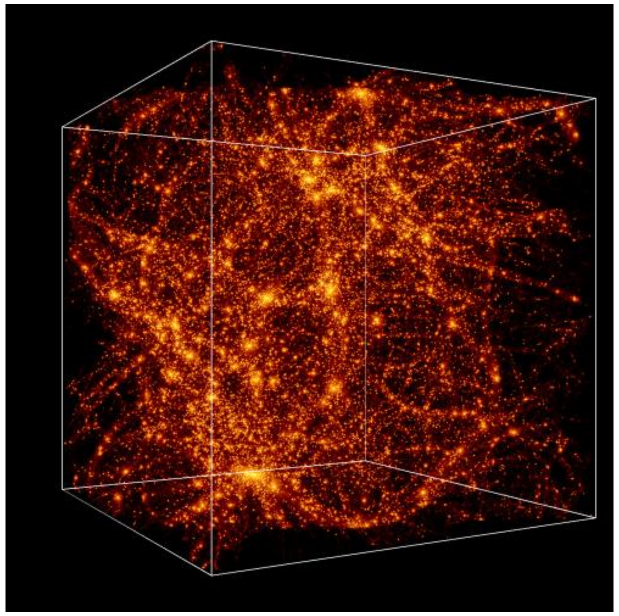
How much are we loosing in feature engineering?

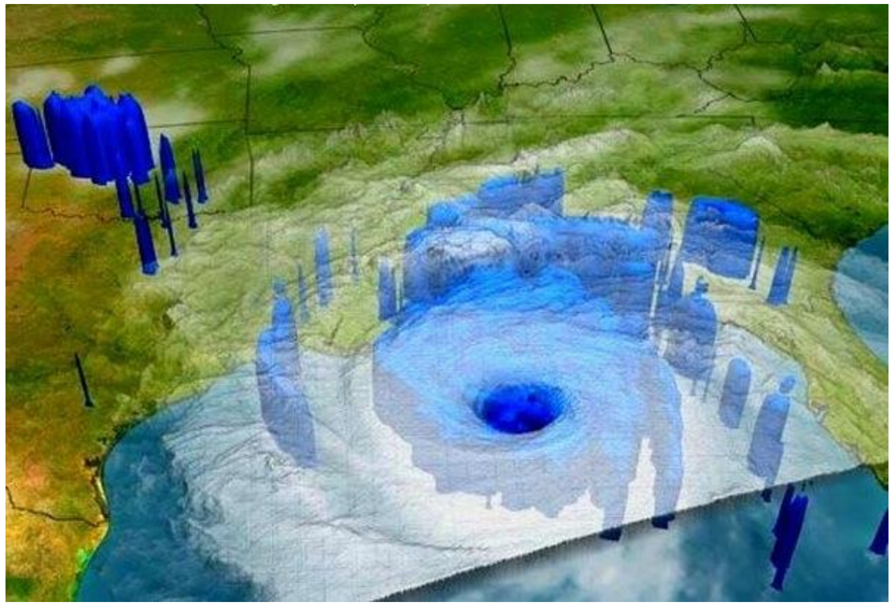
What if we don't know how to design a good feature?

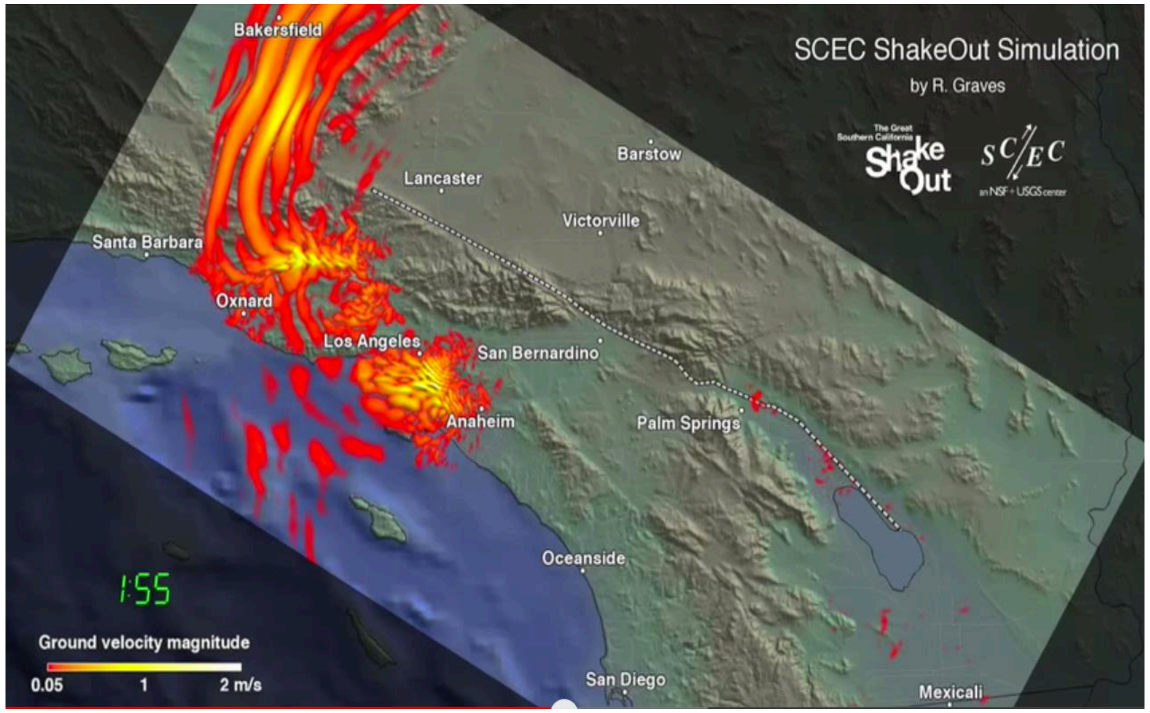

Max Welling

Generative Models: Simulators

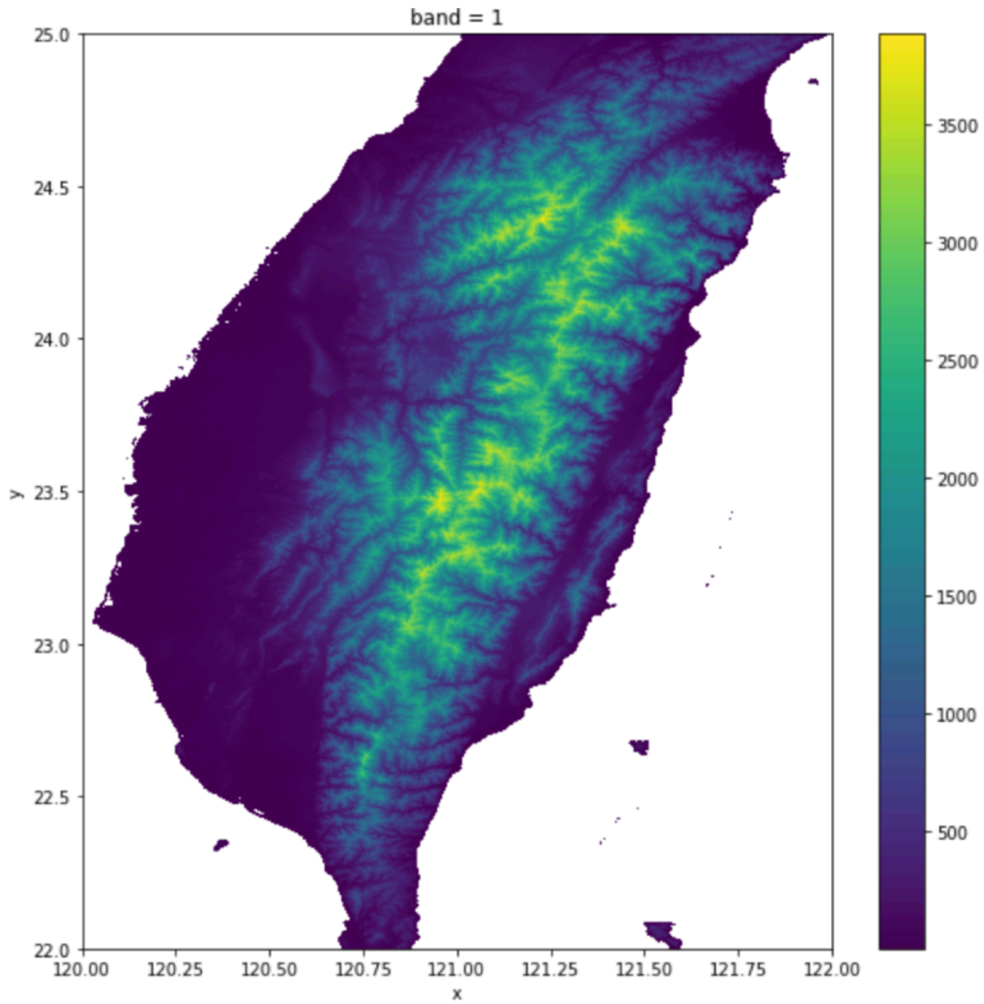


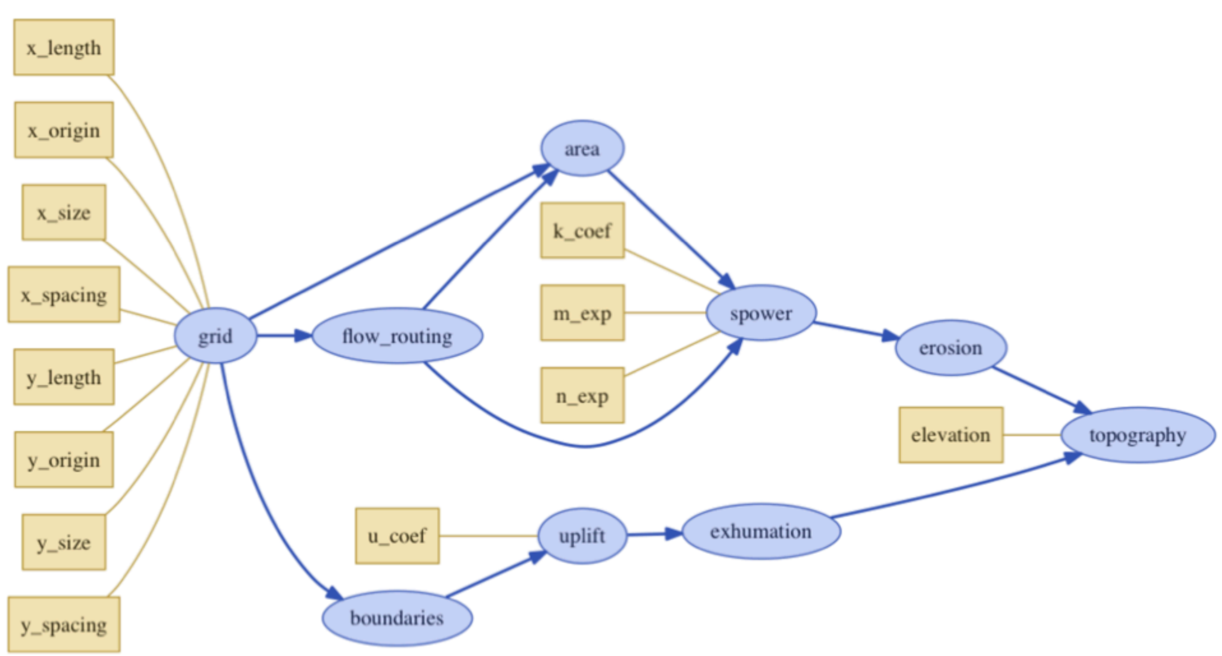




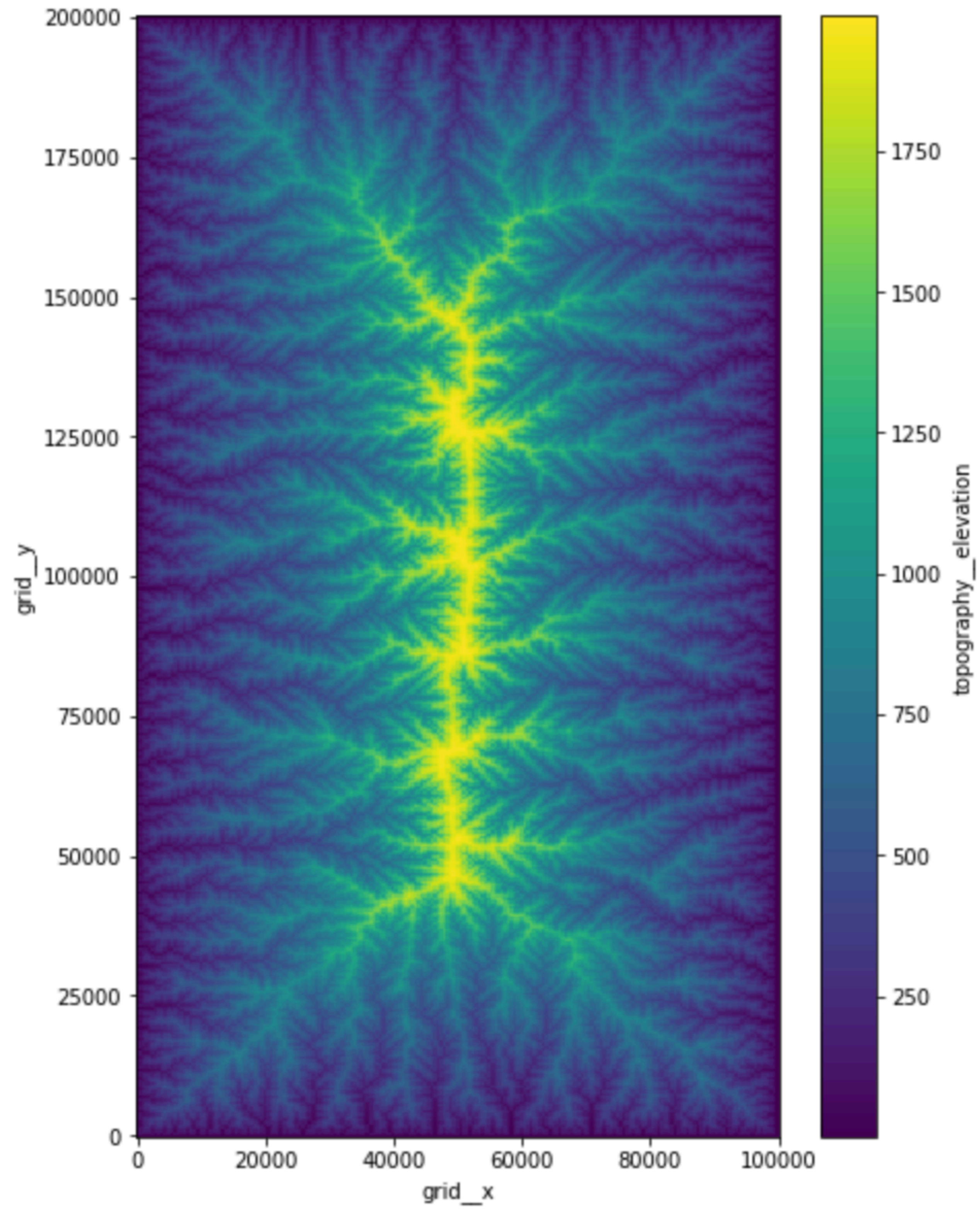


COMPUTATIONAL TOPOGRAPHY

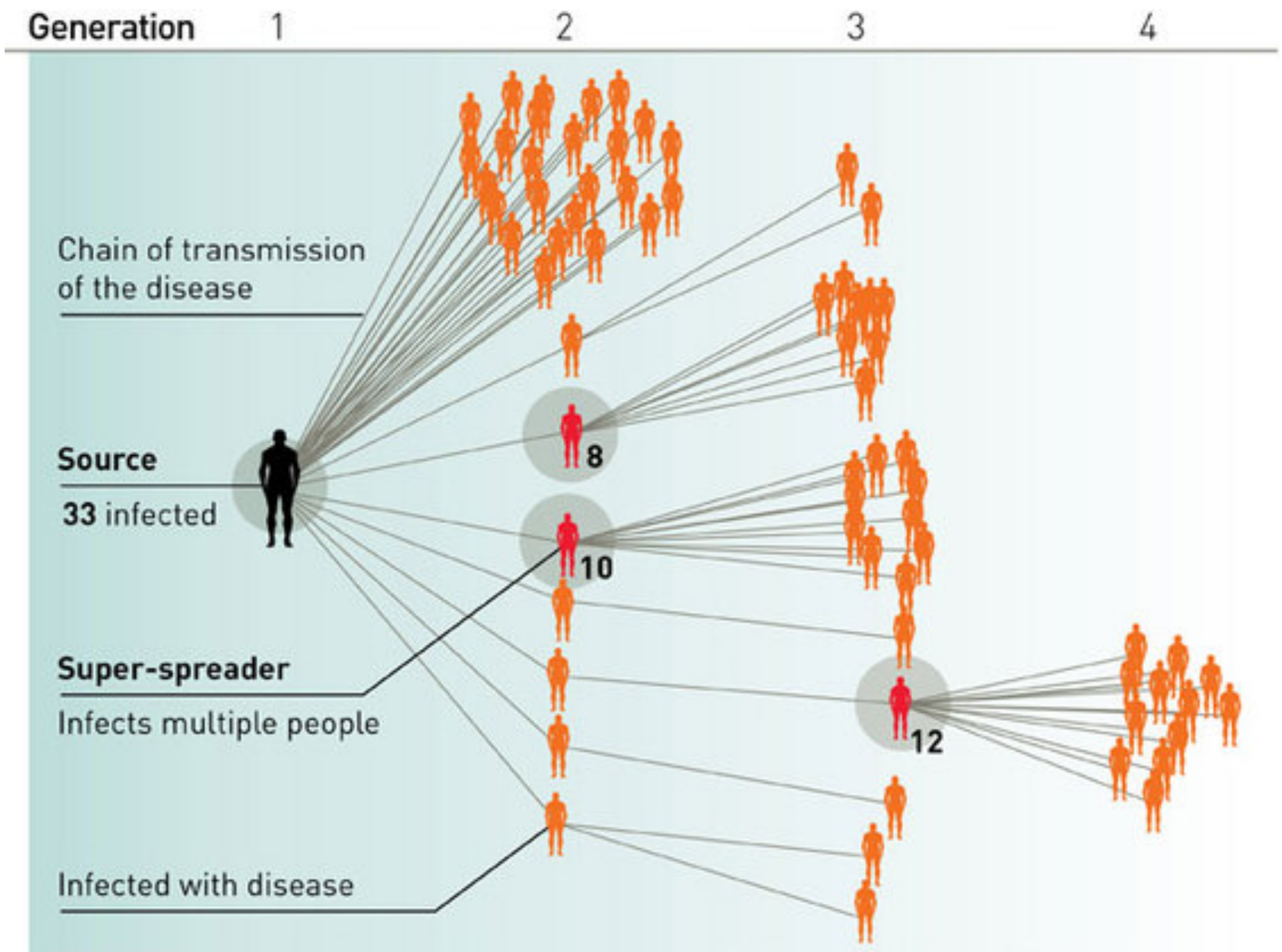


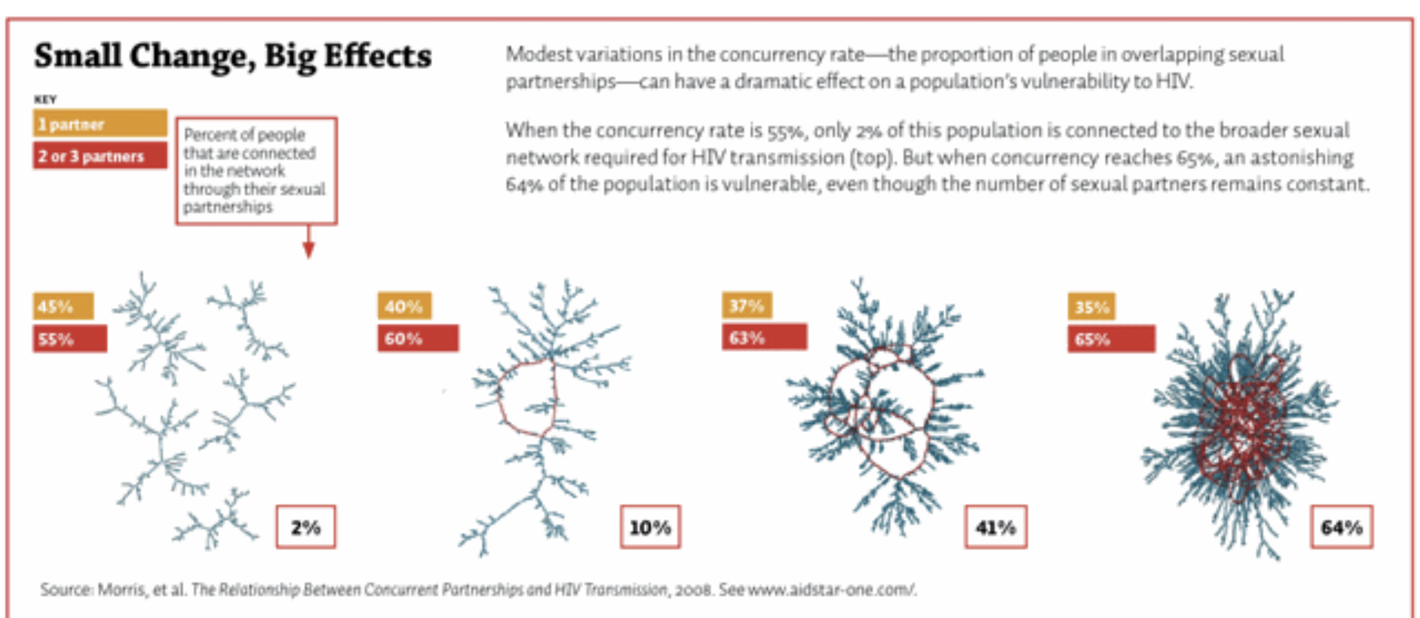


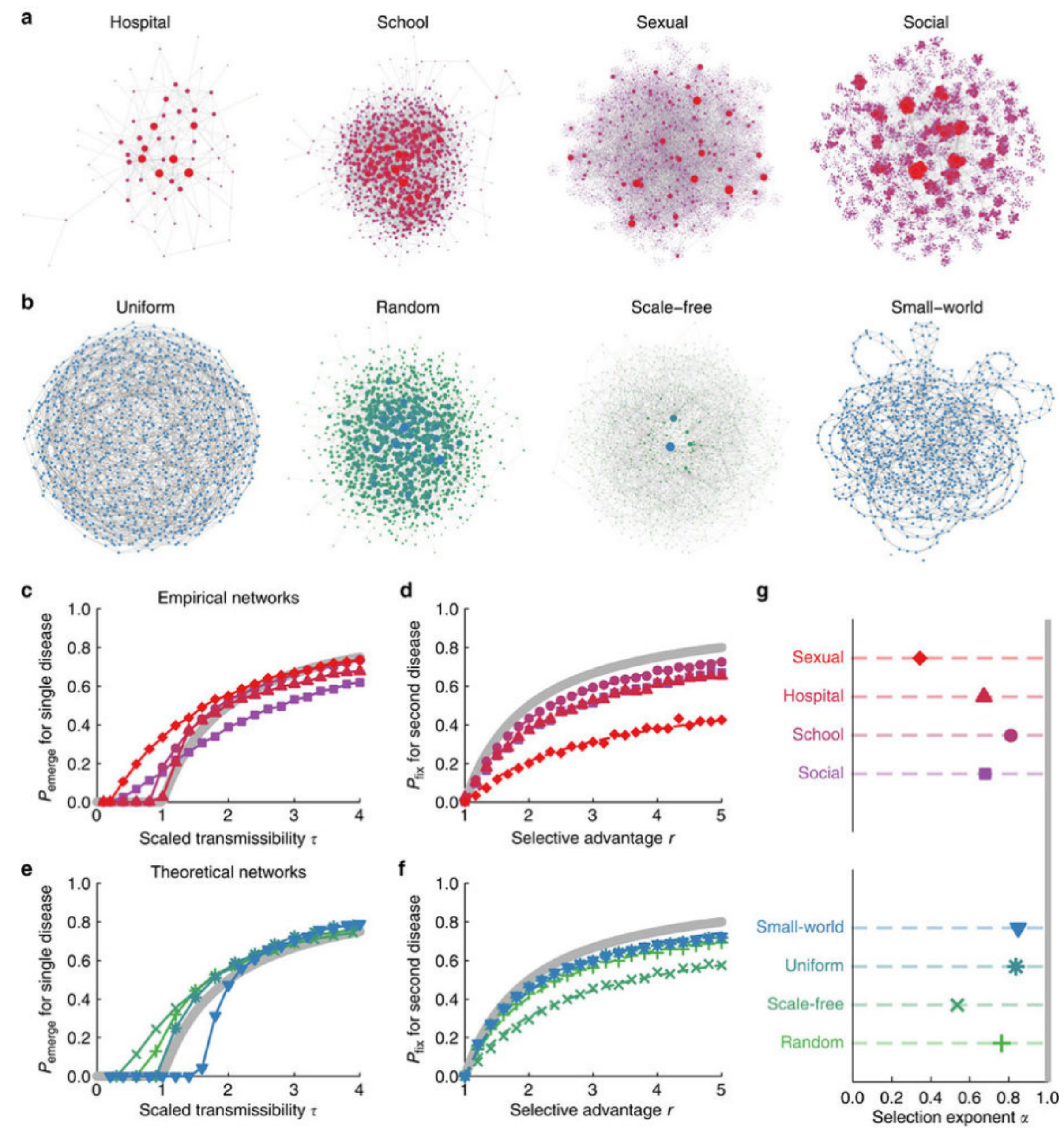
We create a simulation setup for this model, run it, and then plot the final topography (after 1 million years of simulation).



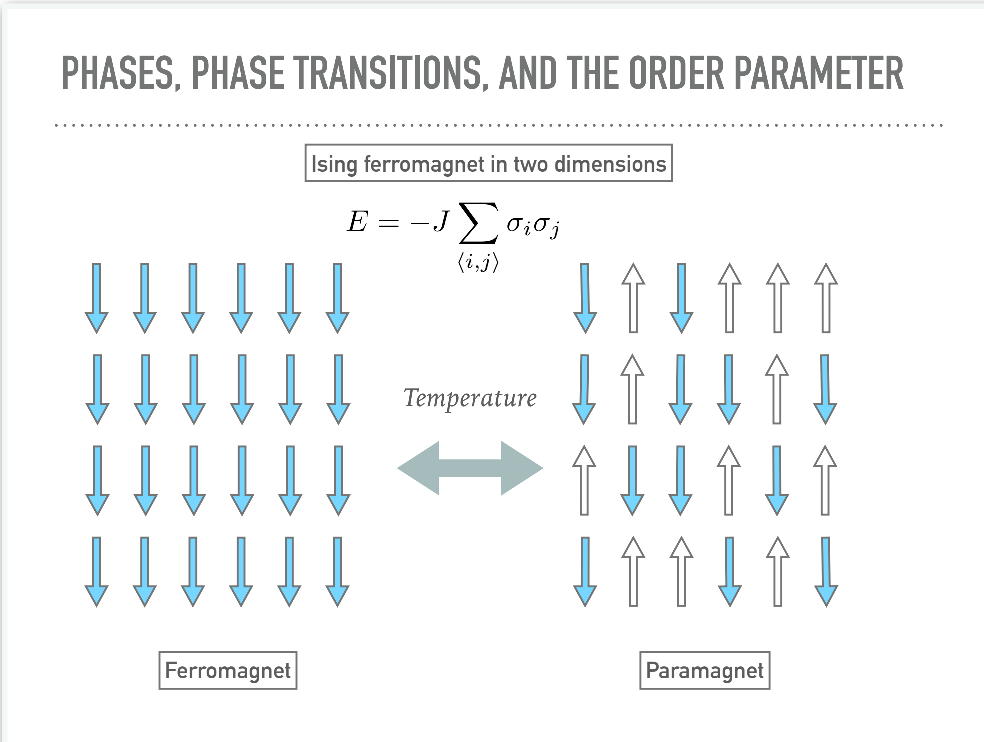
EPIDEMIOLOGY & POPULATION GENETICS

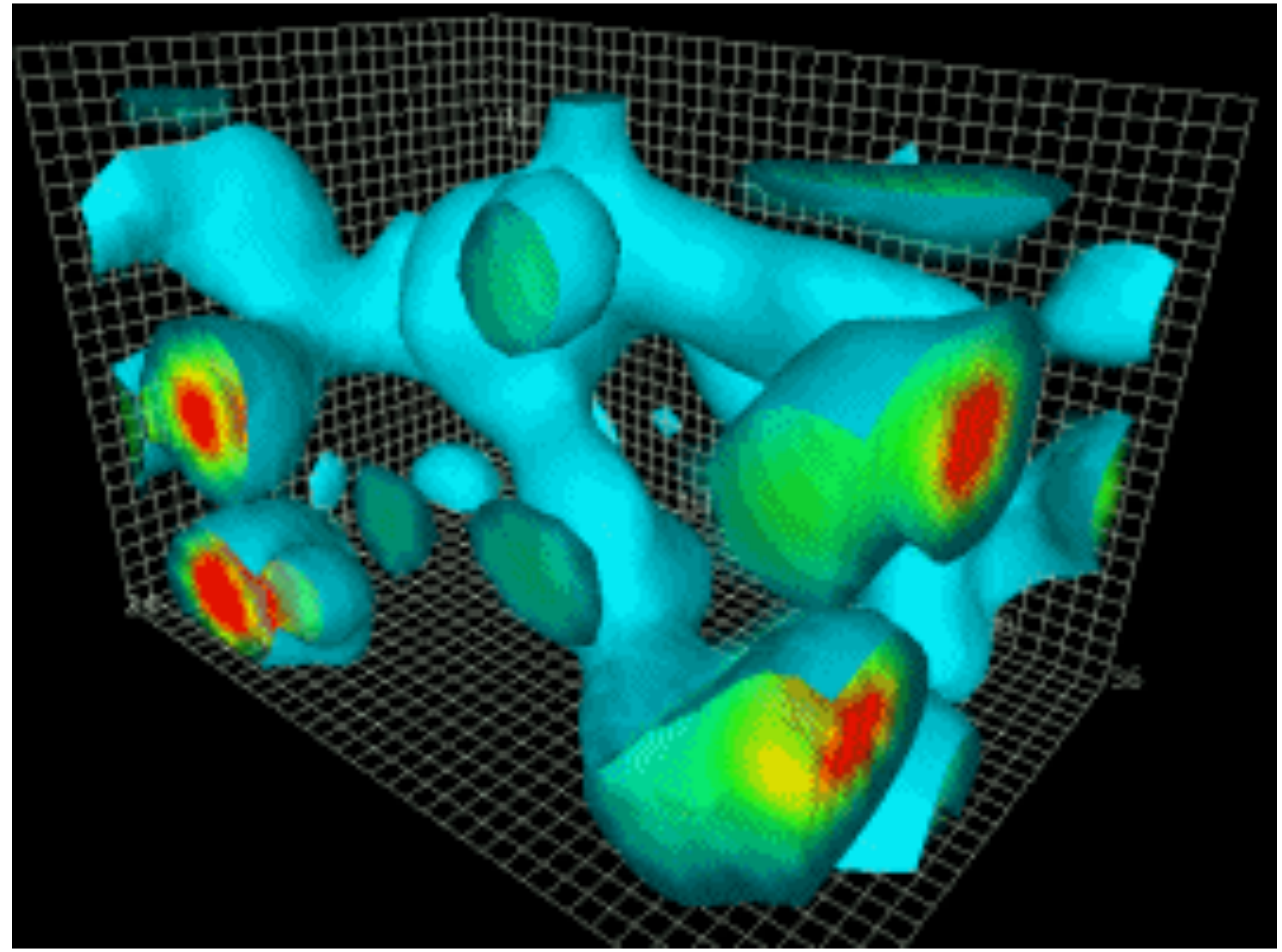






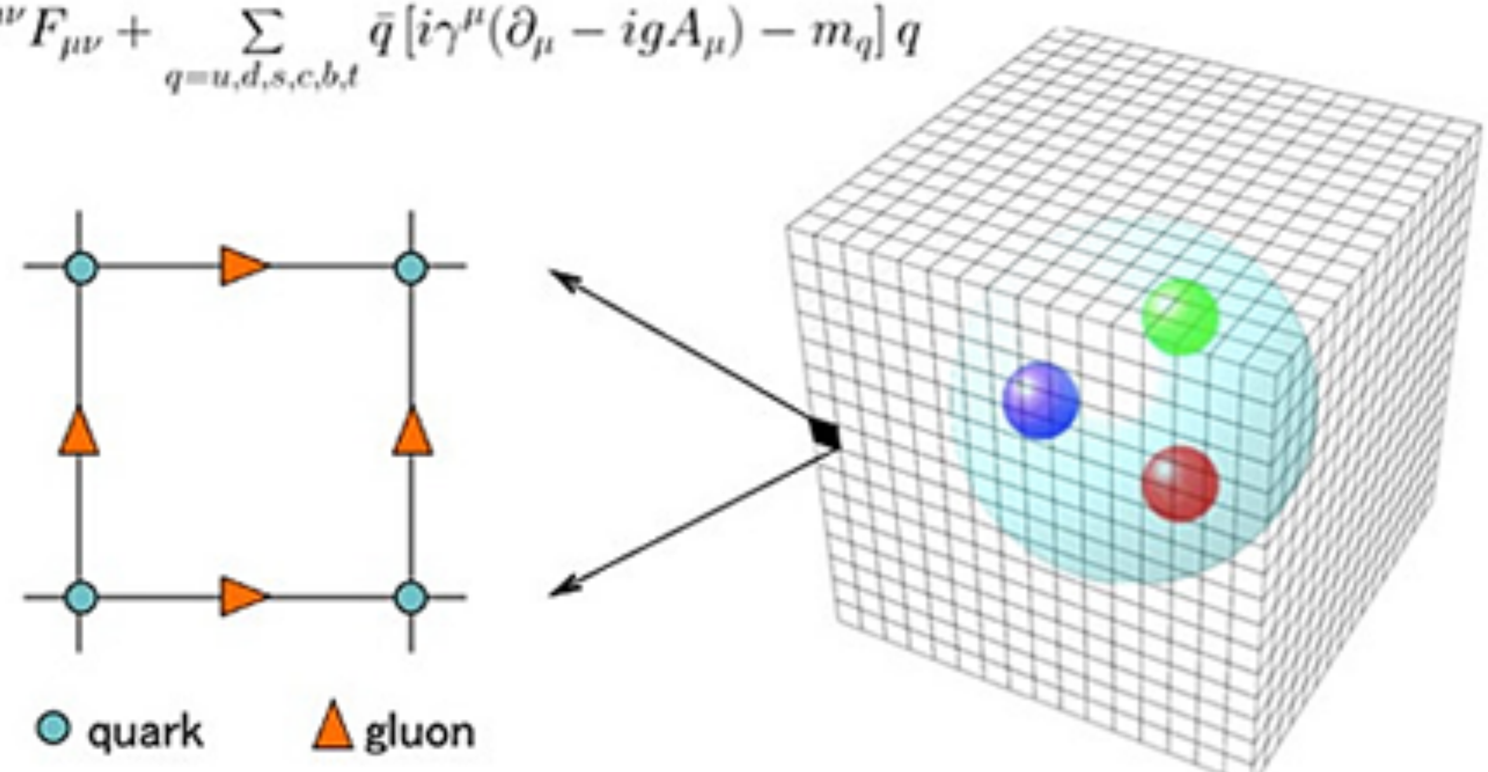
LATTICE FIELD THEORY



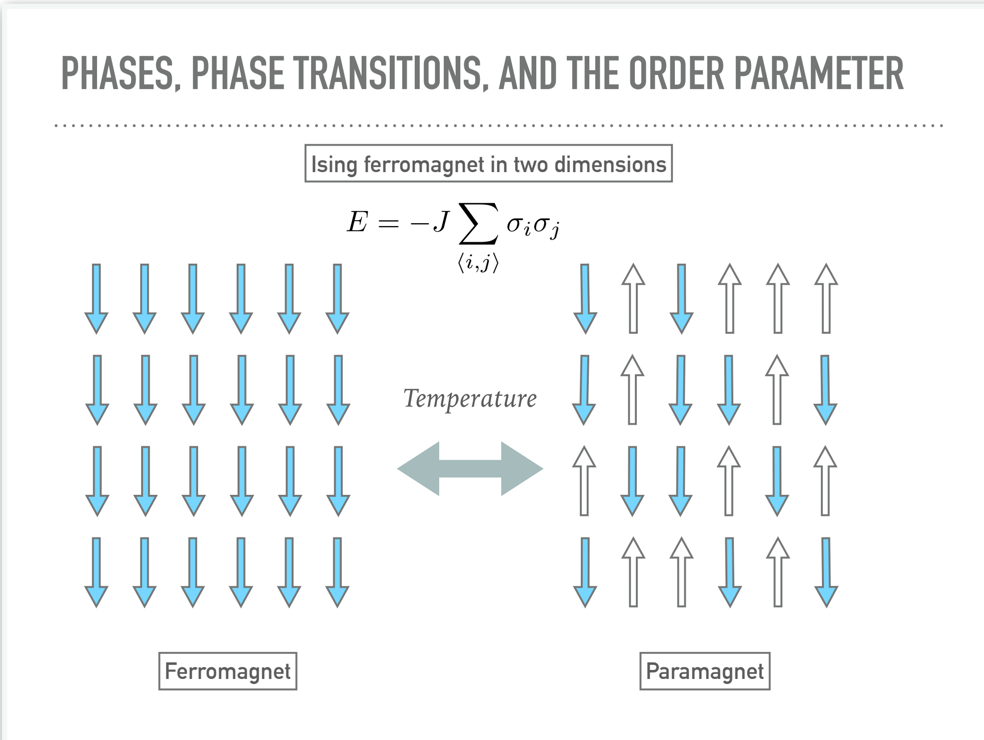


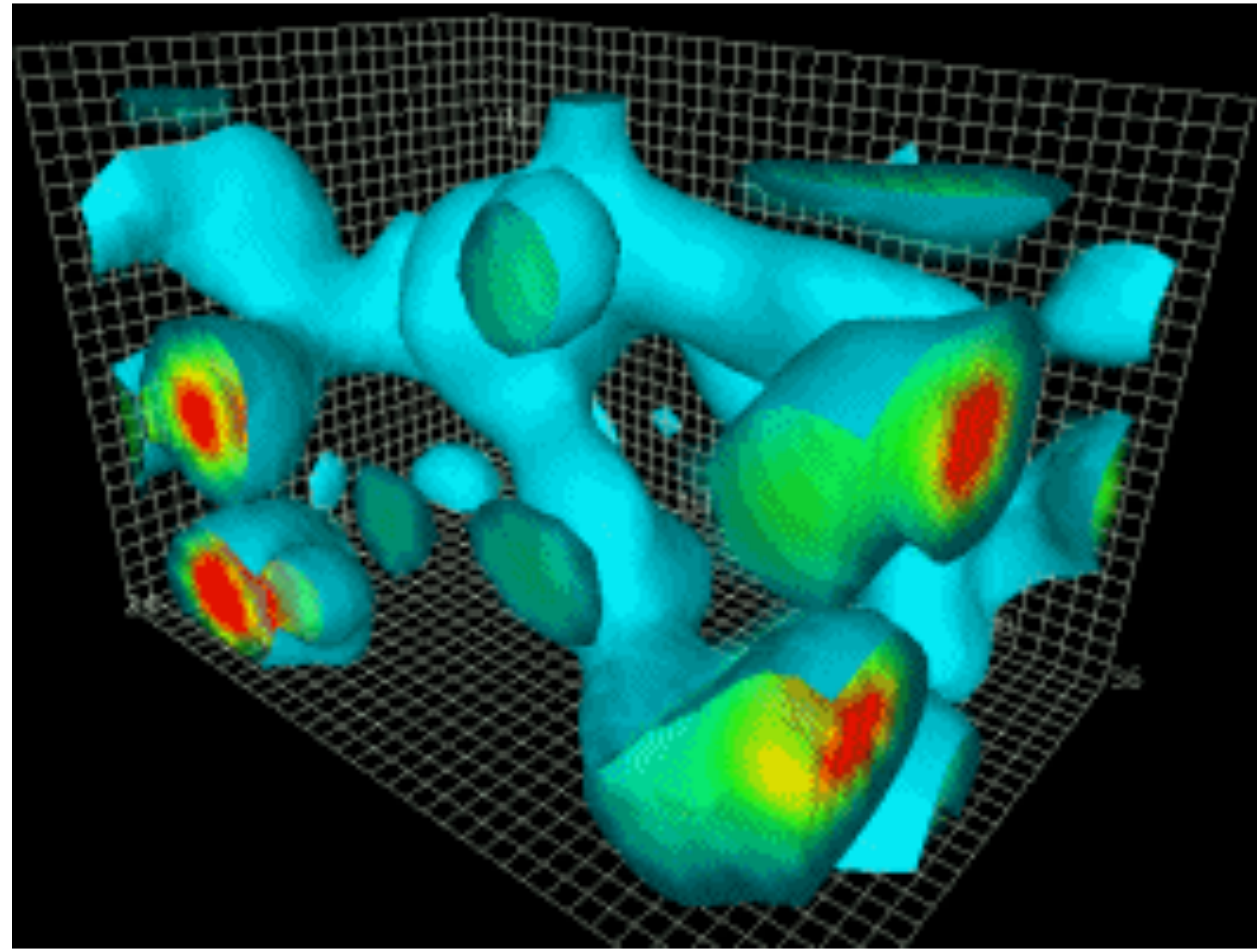
QCD Lagrangian

$$\mathcal{L} = -\frac{1}{4} F^{\mu\nu} F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q} \left[i\gamma^{\mu} (\partial_{\mu} - igA_{\mu}) - m_q \right] q$$



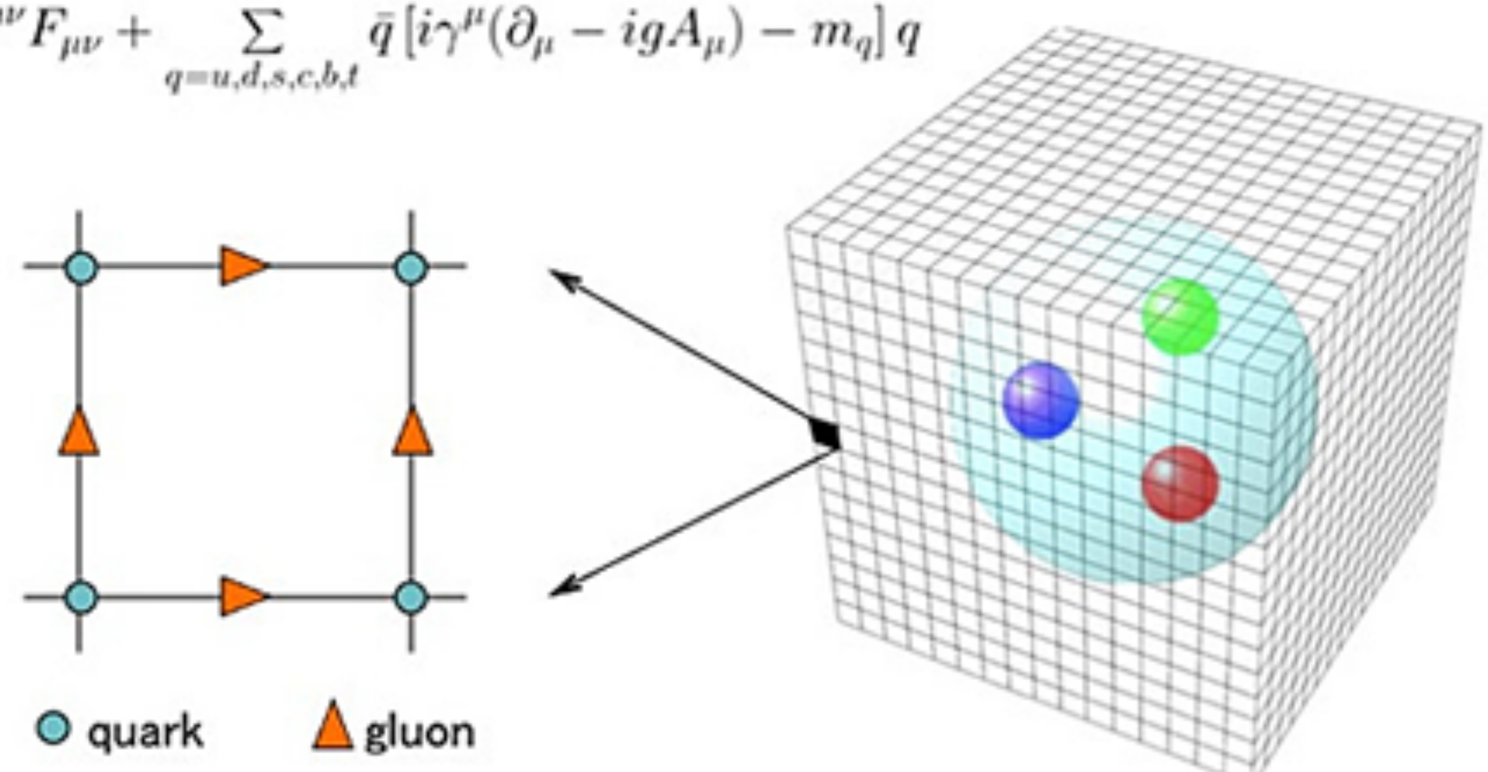
LATTICE FIELD THEORY



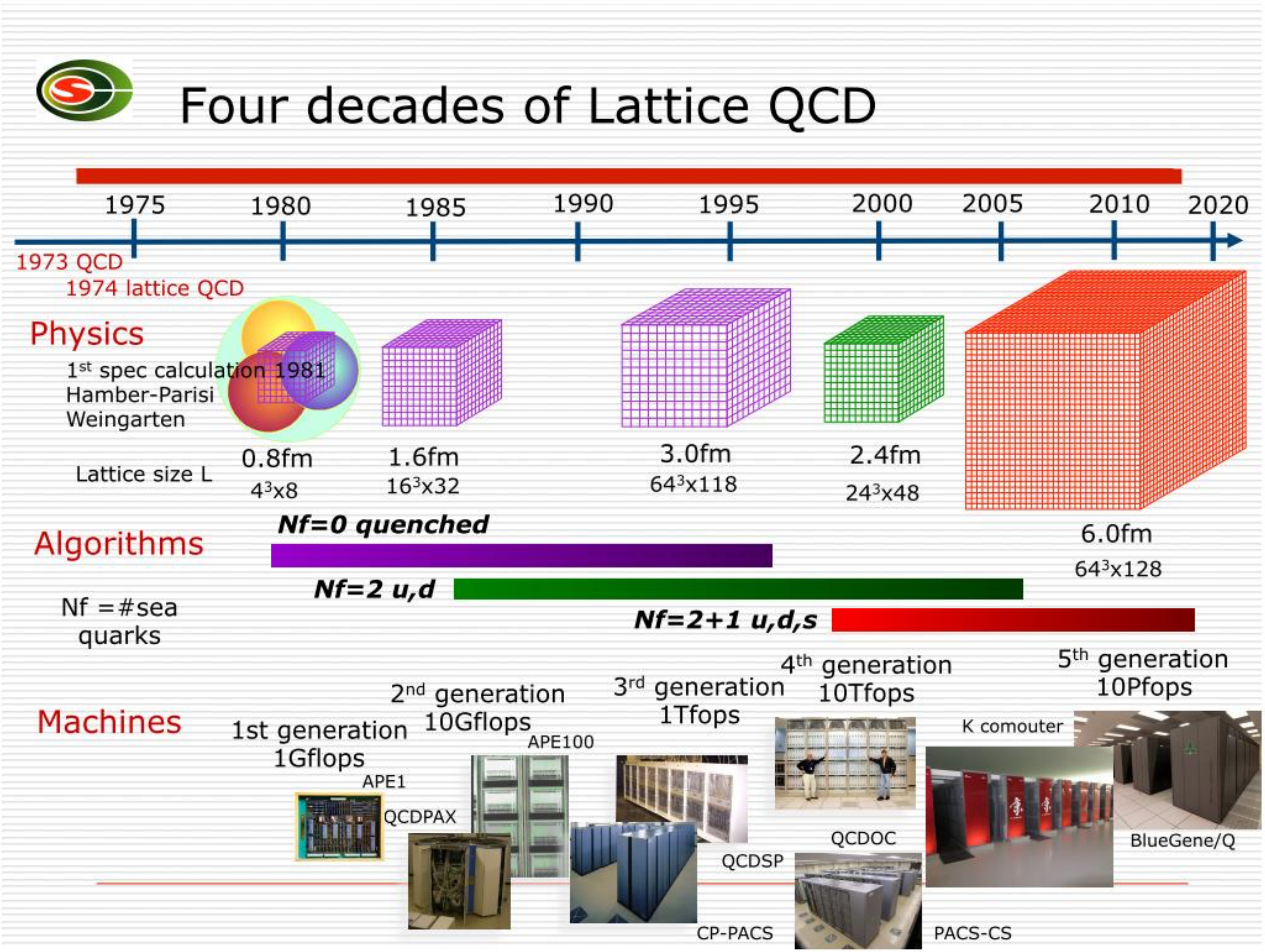


QCD Lagrangian

$$\mathcal{L} = -\frac{1}{4} F^{\mu\nu} F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q} \left[i\gamma^{\mu}(\partial_{\mu} - igA_{\mu}) - m_q \right] q$$



Very expensive simulations, ~1000 examples with x∈ $\mathbb{R}^{10^9}$



ICML 2017 Workshop on Implicit Models

Workshop Aims

Probabilistic models are an important tool in machine learning. They form the basis for models that generate realistic data, uncover hidden structure, and make predictions. Traditionally, probabilistic models in machine learning have focused on prescribed models. Prescribed models specify a joint density over observed and hidden variables that can be easily evaluated. The requirement of a tractable density simplifies their learning but limits their flexibility --- several real world phenomena are better described by simulators that do not admit a tractable density. Probabilistic models defined only via the simulations they produce are called implicit models.

Arguably starting with generative adversarial networks, research on implicit models in machine learning has exploded in recent years. This workshop's aim is to foster a discussion around the recent developments and future directions of implicit models.

Implicit models have many applications. They are used in ecology where models simulate animal populations over time; they are used in phylogeny, where simulations produce hypothetical ancestry trees; they are used in physics to generate particle simulations for high energy processes. Recently, implicit models have been used to improve the state-of-the-art in image and content generation. Part of the workshop's focus is to discuss the commonalities among applications of implicit models.

Of particular interest at this workshop is to unite fields that work on implicit models. For example:

- **Generative adversarial networks** (a NIPS 2016 workshop) are implicit models with an adversarial training scheme.
- Recent advances in **variational inference** (a NIPS 2015 and 2016 workshop) have leveraged implicit models for more accurate approximations.
- **Approximate Bayesian computation** (a NIPS 2015 workshop) focuses on posterior inference for models with implicit likelihoods.
- Learning implicit models is deeply connected to **two sample testing, density ratio and density difference** estimation.

We hope to bring together these different views on implicit models, identifying their core challenges and combining their innovations.

WHY SCIENTISTS SHOULD CARE

Many areas of science have simulations based on some well-motivated mechanistic model.

However, the aggregate effect of many interactions between these low-level components leads to an intractable inverse problem.

The developments in machine learning and AI have the potential to effectively bridge the microscopic - macroscopic divide & aid in the inverse problem.

- they can provide effective statistical models that describe macroscopic phenomena that are tied back to the low-level microscopic (reductionist) model
- generative models and likelihood-free inference are two particularly exciting areas

HAVE A PLAN TO TELL YOU WHERE YOU ARE GOING

GOALS

DEFINE THE MILESTONES THAT WILL ASSIST WITH REACHING YOUR GOALS

OBJECTIVES

DECIDE THE PLAN OF ACTION TO ACHIEVE YOUR OBJECTIVES

STRATEGIES

IDENTIFY THE TOOLS YOU WILL USE TO IMPLEMENT YOUR STRATEGIES

TACTICS

GOALS & STRATEGIES

Goals:

- Use machine learning to do better science

Strategies:

- Import domain knowledge into models (inductive bias)
- Export knowledge from learned models
- Leverage machine learning for intractable inverse problems
- Incorporate traditional scientific concerns into the learning paradigm
  - include impact of domain shift / systematics uncertainties into objective
  - maintain an actionable, scientifically-useful notion of “interpretability”
  - use real-world data for training when possible
  - be data efficient
- Modify codebase to facilitate use of these techniques

STRATEGIES & TACTICS

Strategy: Import domain knowledge into models

- **Tactic**: exploit symmetries in the data
- **Tactic**: exploit geometric structure of the data
- **Tactic**: exploit causal structure of the generative process
- **Tactic**: exploit hierarchical / compositional structure
- **Tactic**: exploit Markov property of the generative process
- **Tactic**: exploit tangent space of statistical manifold

Strategy: Export knowledge from learned models

- **Tactic**: learnable components that can be interpreted

Strategy: maintain an actionable, scientifically-useful notion of "interpretability"

- **Tactic:** compose model from reusable components that perform a specific task and can be individually characterized & validated

STRATEGIES & TACTICS

Strategy: Leverage machine learning for intractable inverse problems

- **Tactic**: Use the likelihood ratio trick to convert a discriminative classifier into a density ratio
- **Tactic:** Use autoregressive models & normalizing flows for conditional density estimation
- **Tactic:** Use universal probabilistic programming
- **Tactic:** Approximate gradients of non-differentiable, black-box models (AVO, RELAX, …)

Strategy: include impact of systematics uncertainties into objective

- **Tactic**: Design loss functions more relevant to scientific goals
- **Tactic**: Adversarial training for continuous domain adaptation & fairness ("learning to pivot")

Strategy: use real-world data for training when possible

- **Tactic**: Weakly supervised learning

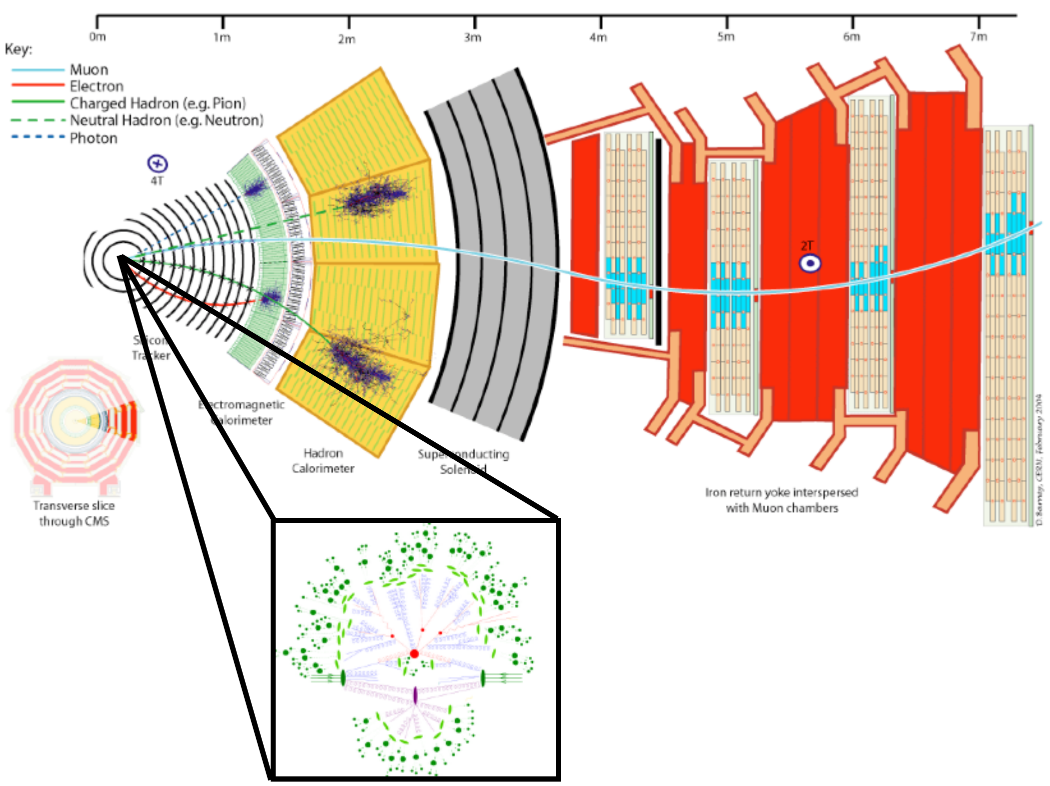
Strategy: be data efficient

- **Tactic**: exploiting domain knowledge can dramatically reduce number of parameters

TACTICS FOR INTRACTABLE INVERSE PROBLEMS

Use simulator

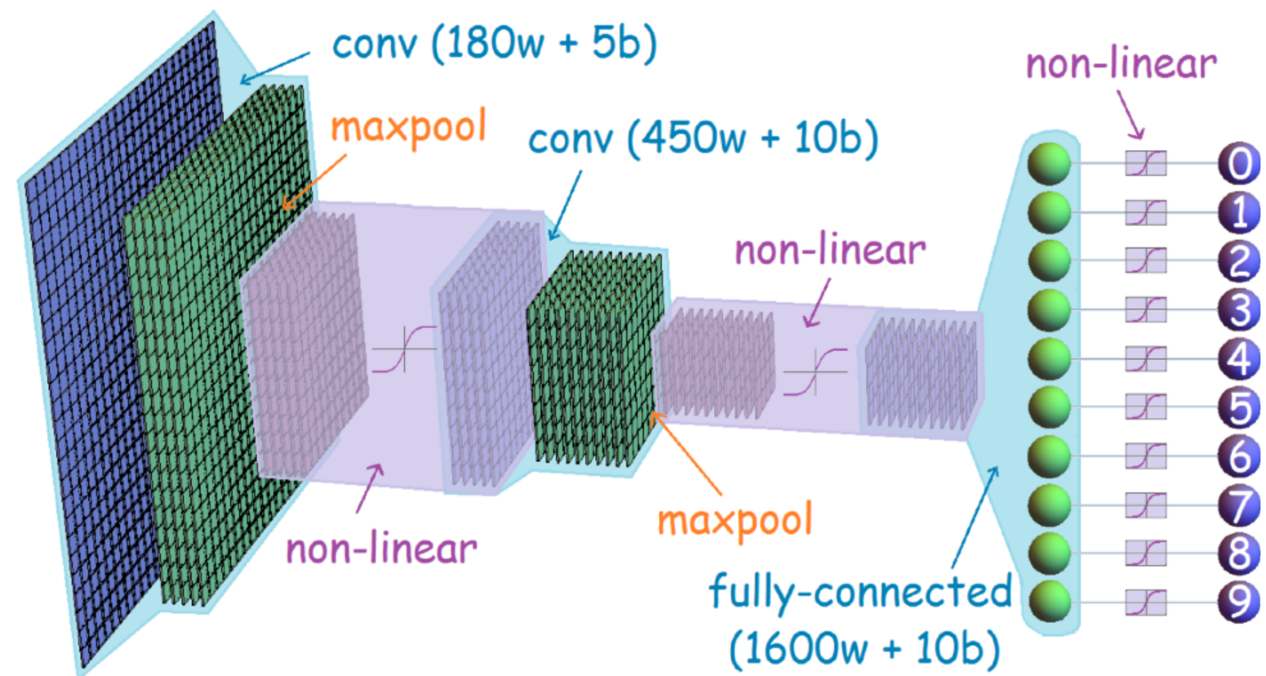
(much more efficiently)



- Approximate Bayesian Computation (ABC)
- Probabilistic Programming
- Adversarial Variational Optimization (AVO)

Learn simulator

(with deep learning)



- Generative Adversarial Networks (GANs), Variational Auto-Encoders (VAE)
- Likelihood ratio from classifiers (CARL)
- Autoregressive models, Normalizing Flows

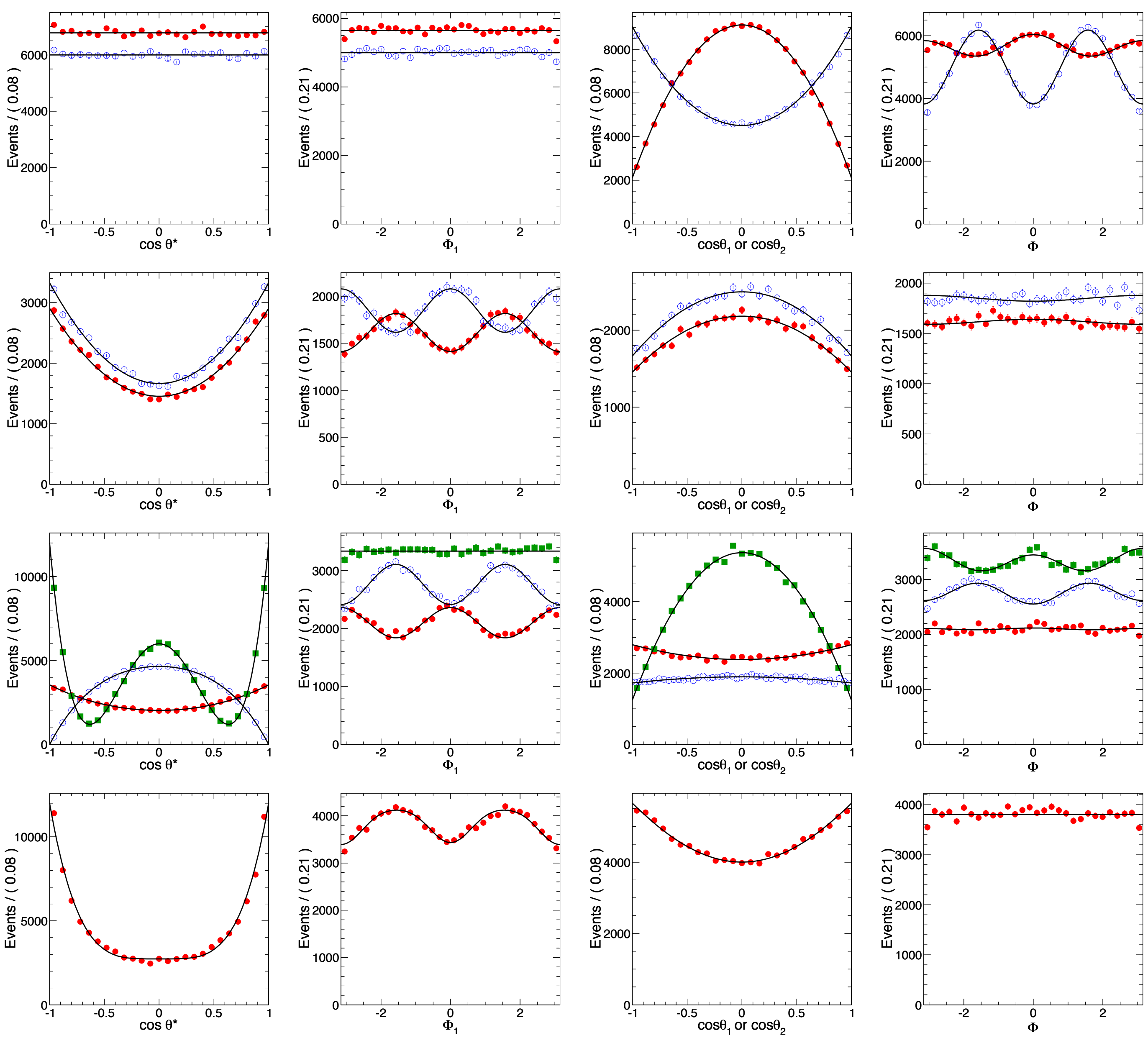
[image credit: A.P. Goucher]

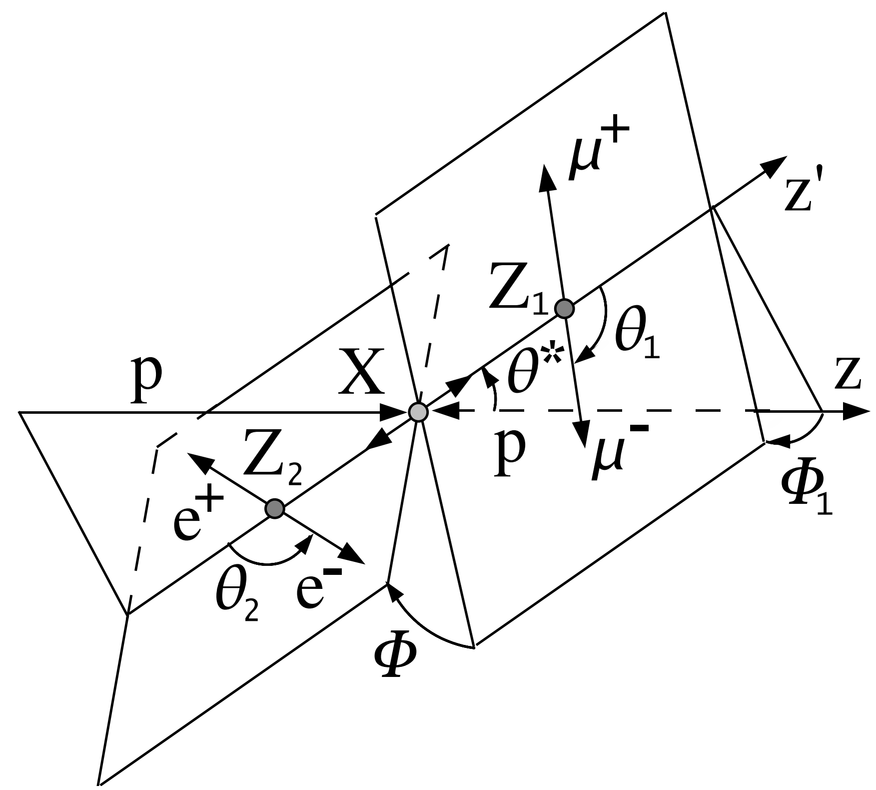
Example:

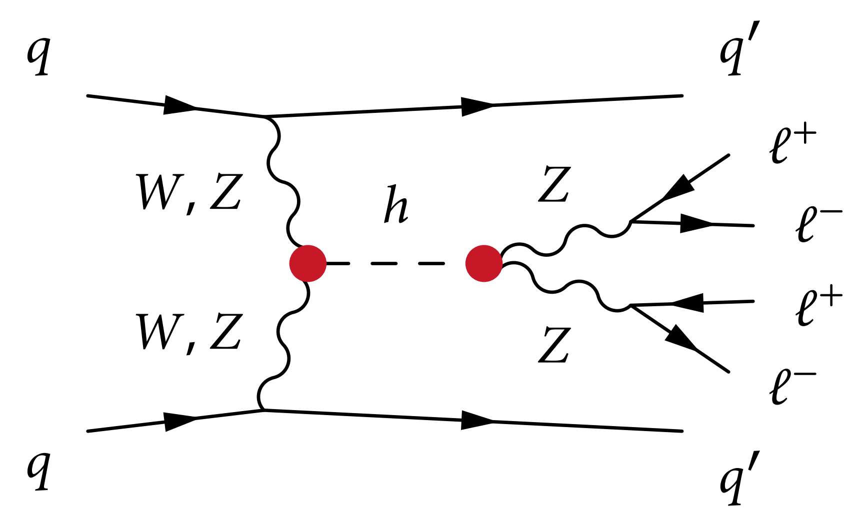
More Powerful Higgs Measurements

When looking for deviations from the standard model Higgs, we would like to look at all sorts of kinematic correlations

- thus each observation **x** is high-dimensional







A binary classifier approximates

$$s(x) = \frac{p(x|H_1)}{p(x|H_0) + p(x|H_1)}$$

Which is one-to-one with the likelihood ratio

$$\frac{p(x|H_1)}{p(x|H_0)} = 1 - \frac{1}{s(x)}$$

Can do the same thing for any two points θ_0 & θ_1 in parameter space $\mathbf{\Theta}$. I call this a **parametrized classifier**

$$s(x; \theta_0, \theta_1) = \frac{p(x|\theta_1)}{p(x|\theta_0) + p(x|\theta_1)}$$

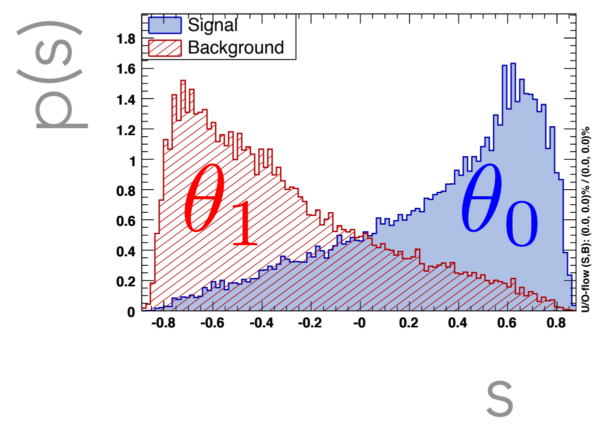
K.C., G. Louppe, J. Pavez: Approximating Likelihood Ratios with Calibrated Discriminative Classifiers [arXiv:1506.02169]

The intractable likelihood ratio based on high-dimensional features x is:

$$\frac{p(x|\theta_0)}{p(x|\theta_1)}$$

We can show that an **equivalent test** can be made from 1-D projection

$$\frac{p(x|\theta_0)}{p(x|\theta_1)} = \frac{p(s(x;\theta_0,\theta_1)|\theta_0)}{p(s(x;\theta_0,\theta_1)|\theta_1)}$$



if the scalar map s: X → $\mathbb{R}$ has the same level sets as the likelihood ratio

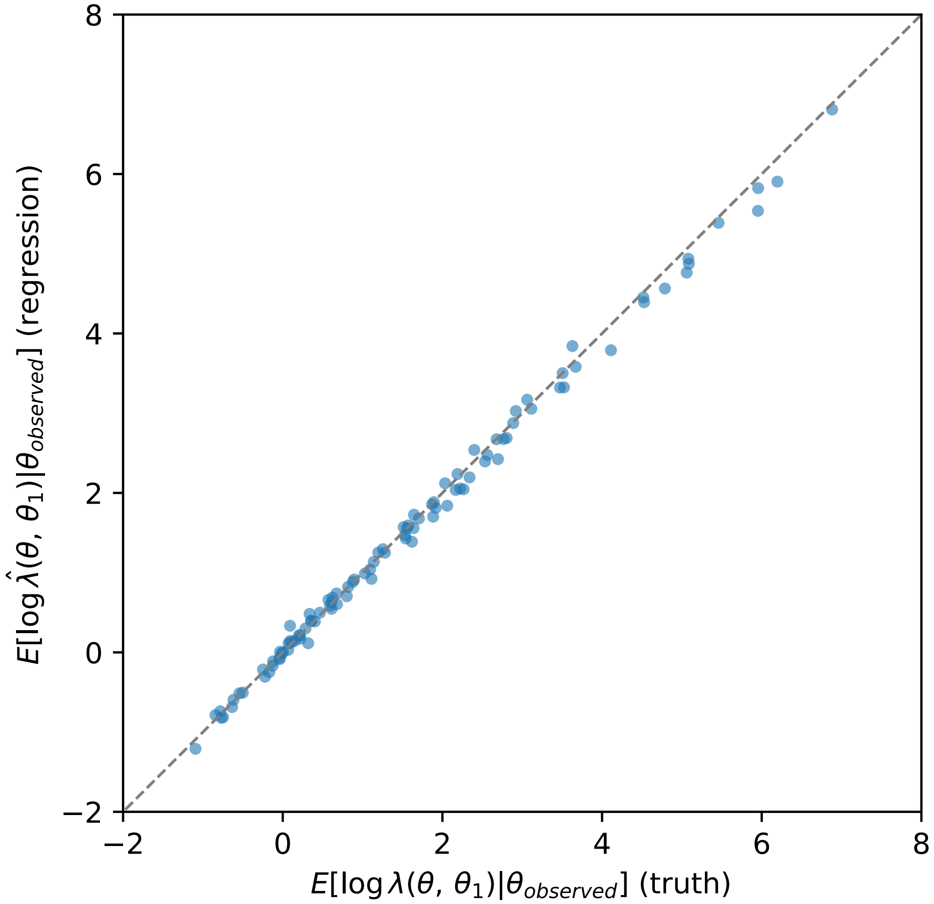
$$s(x;\theta_0;\theta_1) = \text{monotonic}[\ p(x|\theta_0)/p(x|\theta_1)\ ]$$

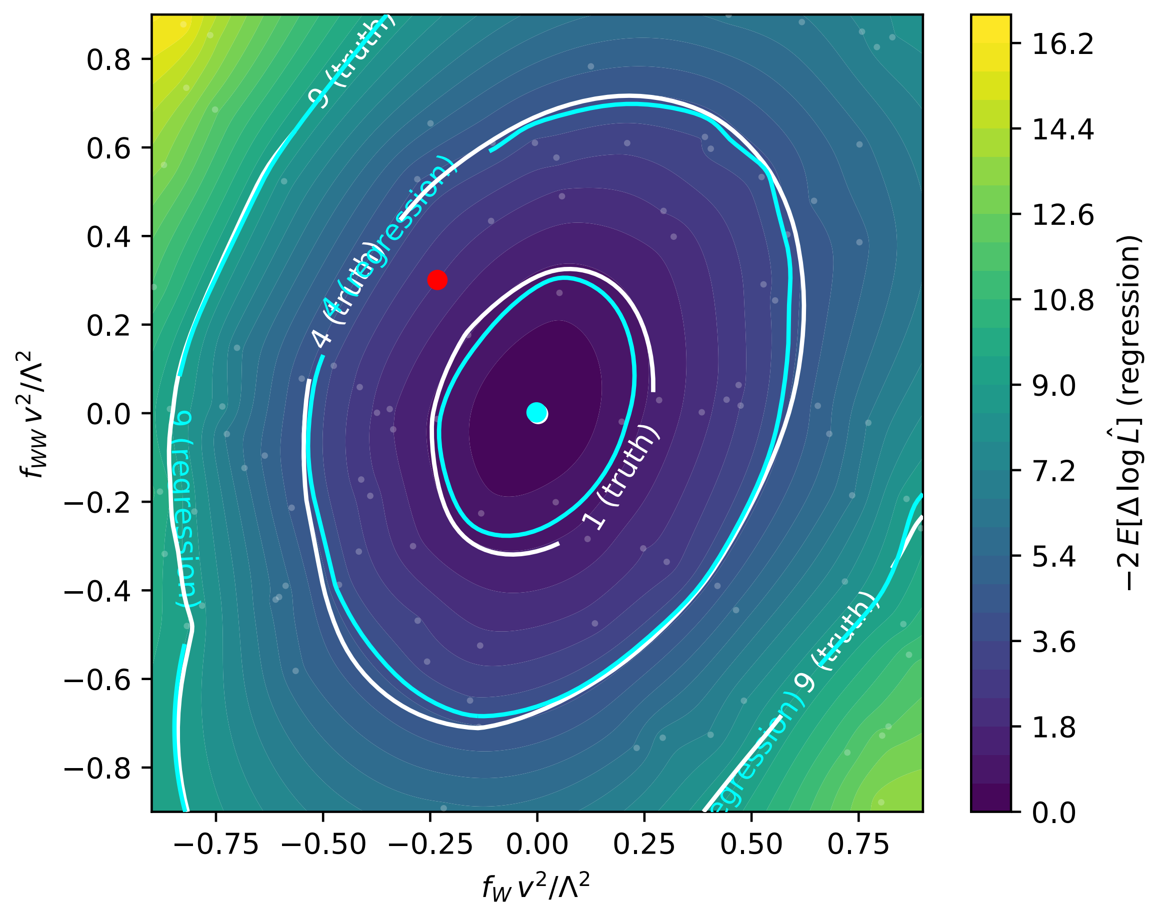
Estimating the density of $s(x;\theta_0,\theta_1)$ via the simulator calibrates the ratio.

K.C., G. Louppe, J. Pavez: Approximating Likelihood Ratios with Calibrated Discriminative Classifiers [arXiv:1506.02169]

(based on a 16-D observation **x**)

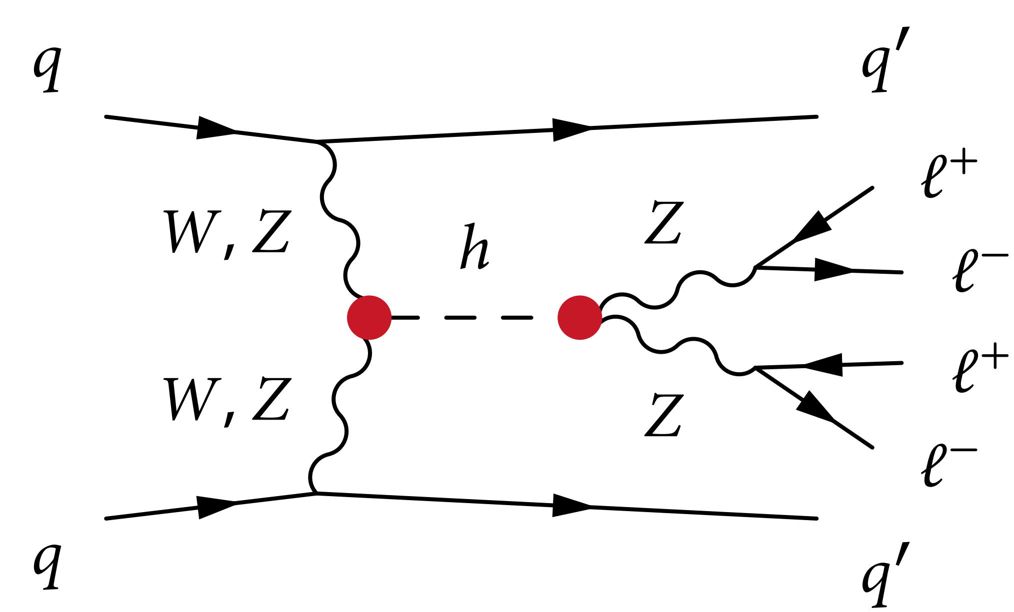
Estimated likelihood





True likelihood

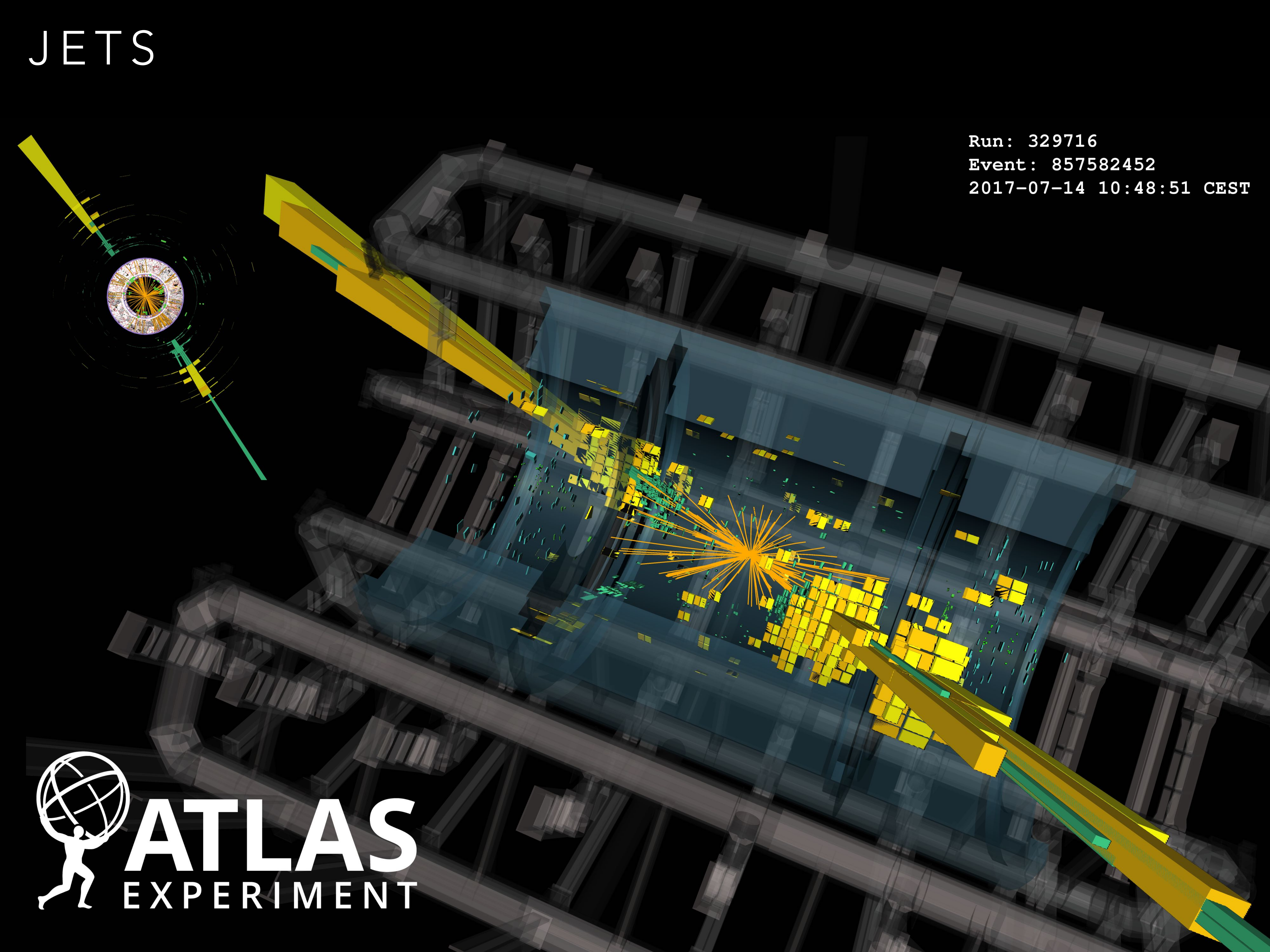
Equivalent to 3x more data!



work with Juan Pavez, Gilles Louppe, Cyril Becot, and Lukas Heinrich; Johann Brehmer, Felix Kling, and Tilman Plehn
“Better Higgs Measurements Through Information Geometry” [arXiv:1612.05261] & CARL [arxiv:1506.02169]

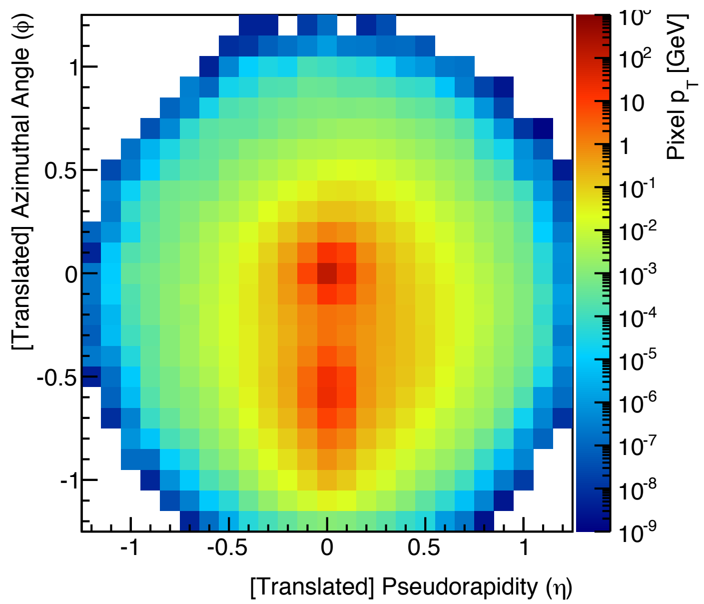
Example:

Jet Classification

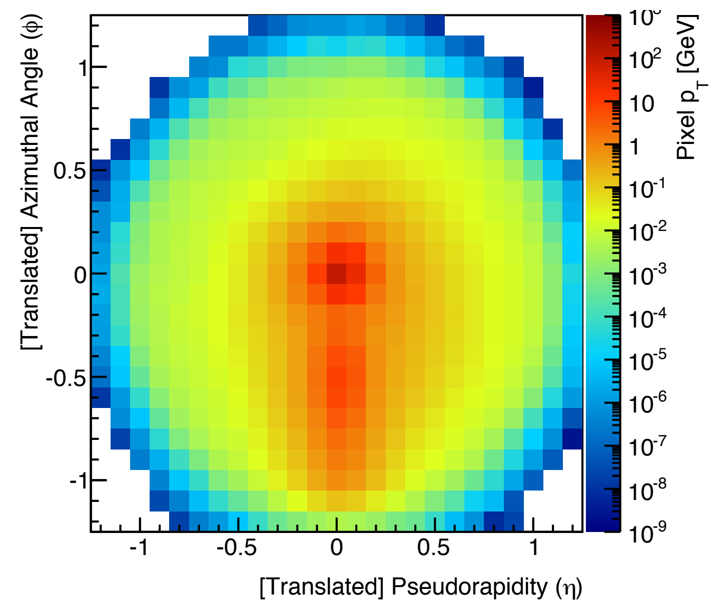

JETS
Run: 329716
Event: 857582452
2017-07-14 10:48:51 CEST
ATLAS
EXPERIMENT

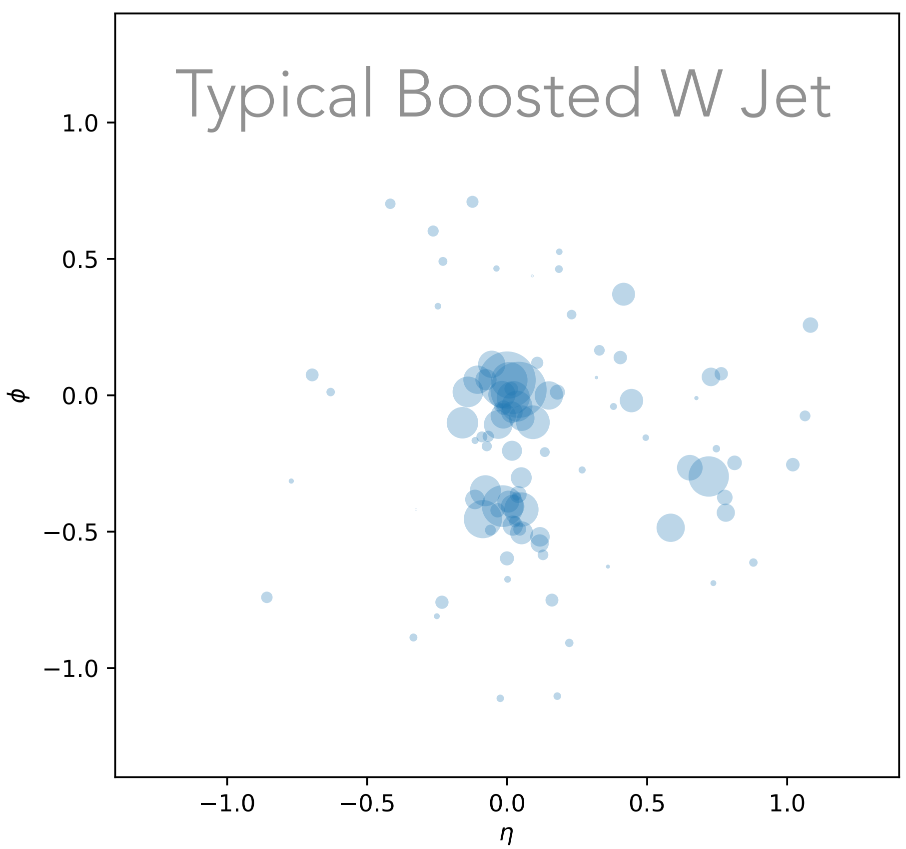
CLASSIFICATION

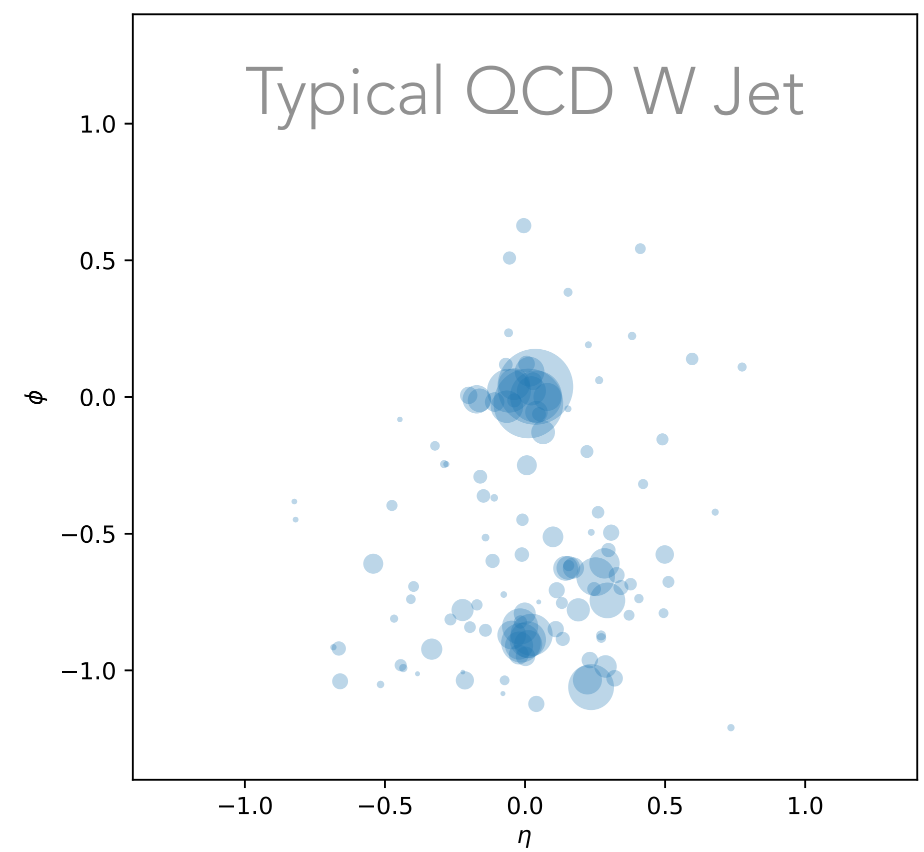
Average Boosted W Jet



Average QCD Jet

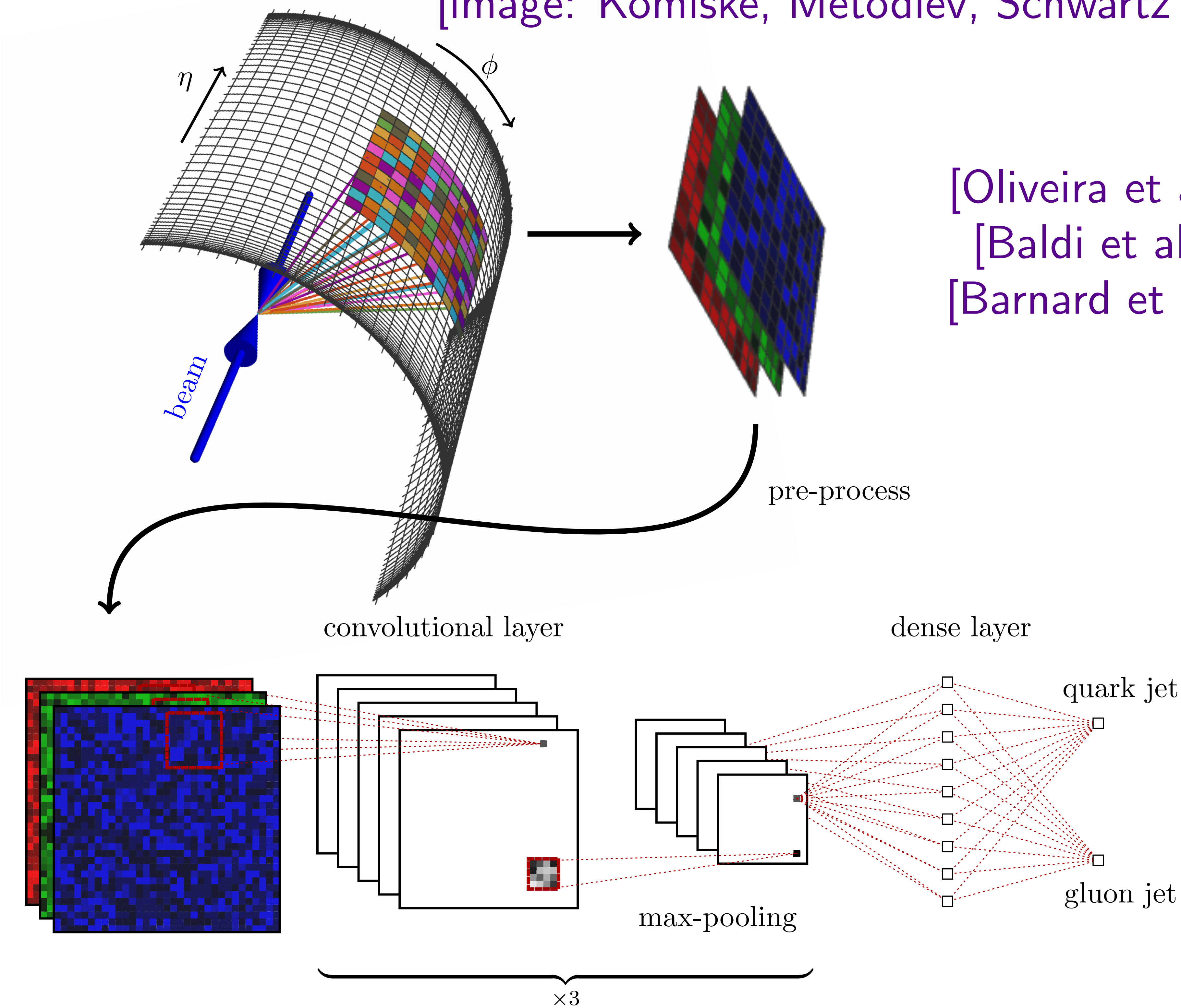






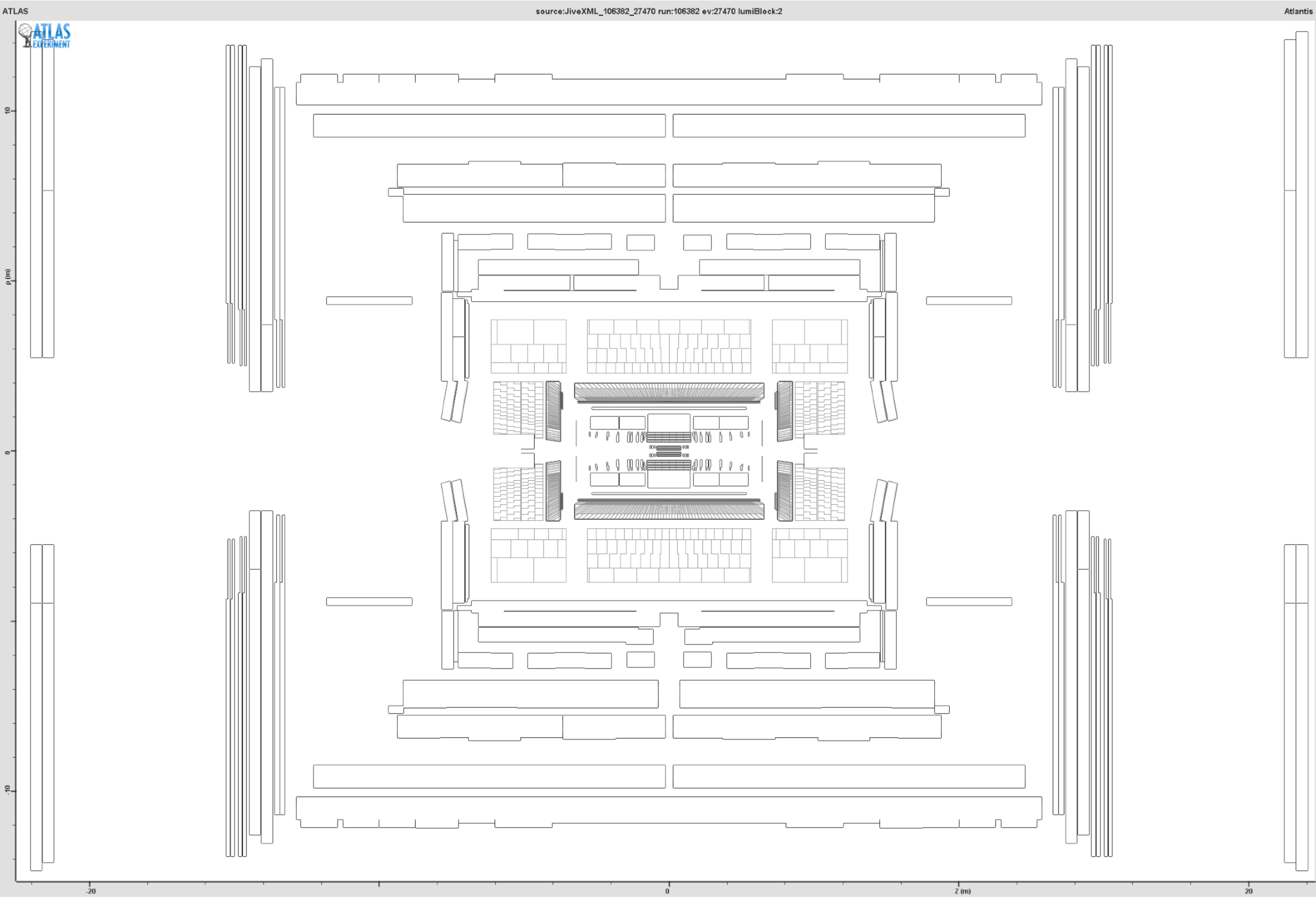
JET IMAGES

[image: Komiske, Metodiev, Schwartz arxiv:1612.01551]

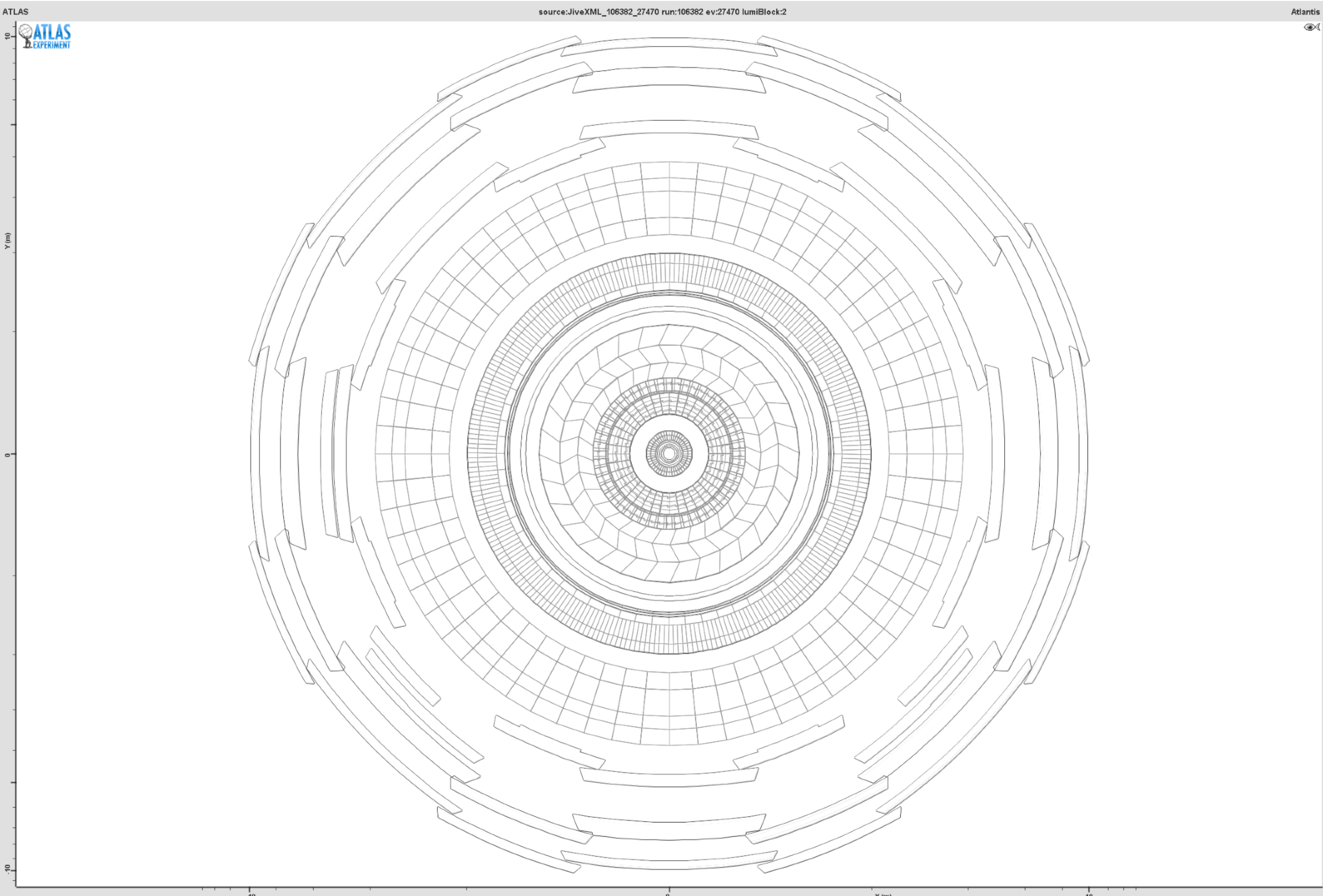


[Oliveira et al arXiv:1511.05190]
[Baldi et al arXiv:1603.09349]
[Barnard et al arXiv:1609.00607]

NON-UNIFORM GEOMETRY

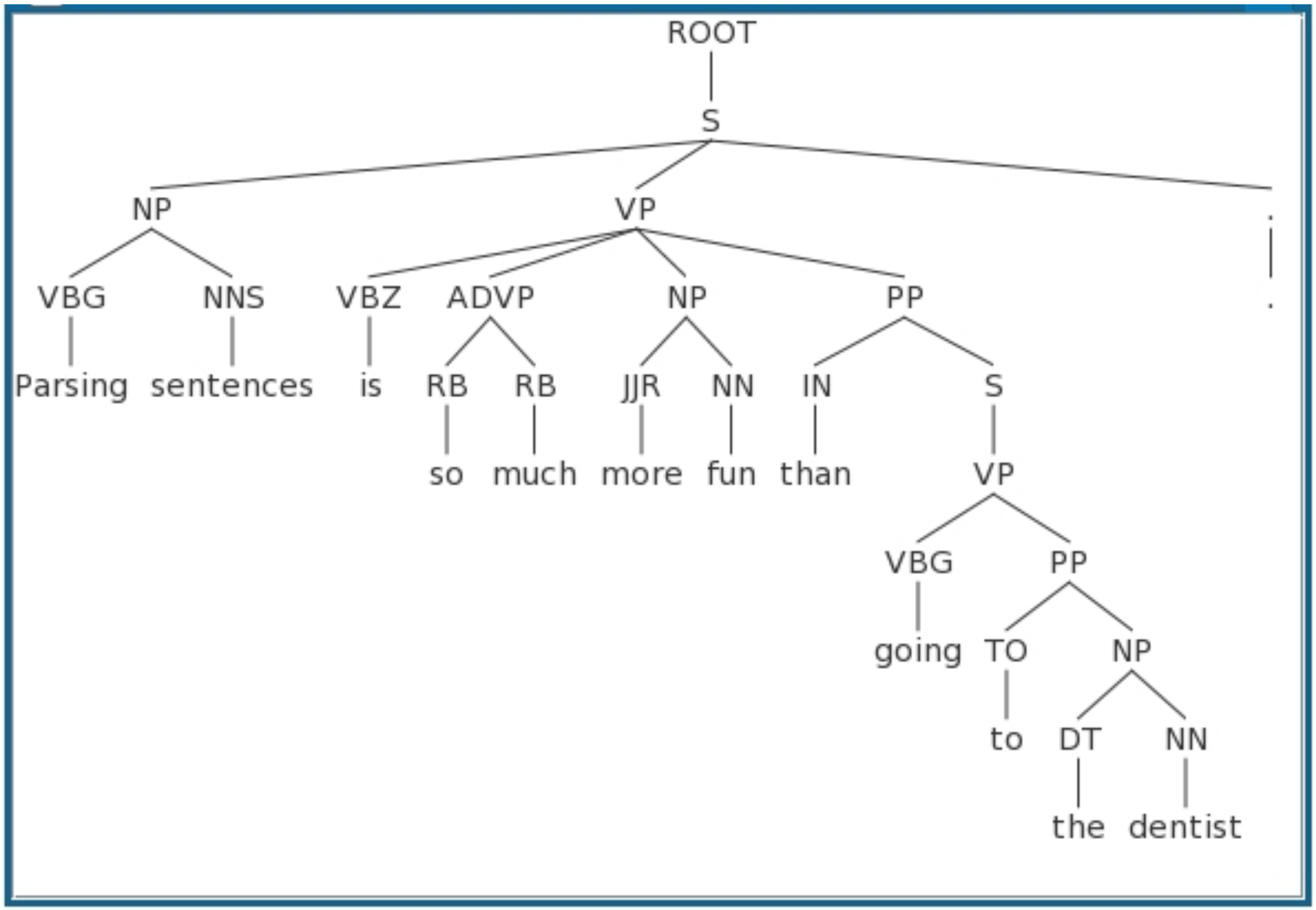


NON-UNIFORM GEOMETRY



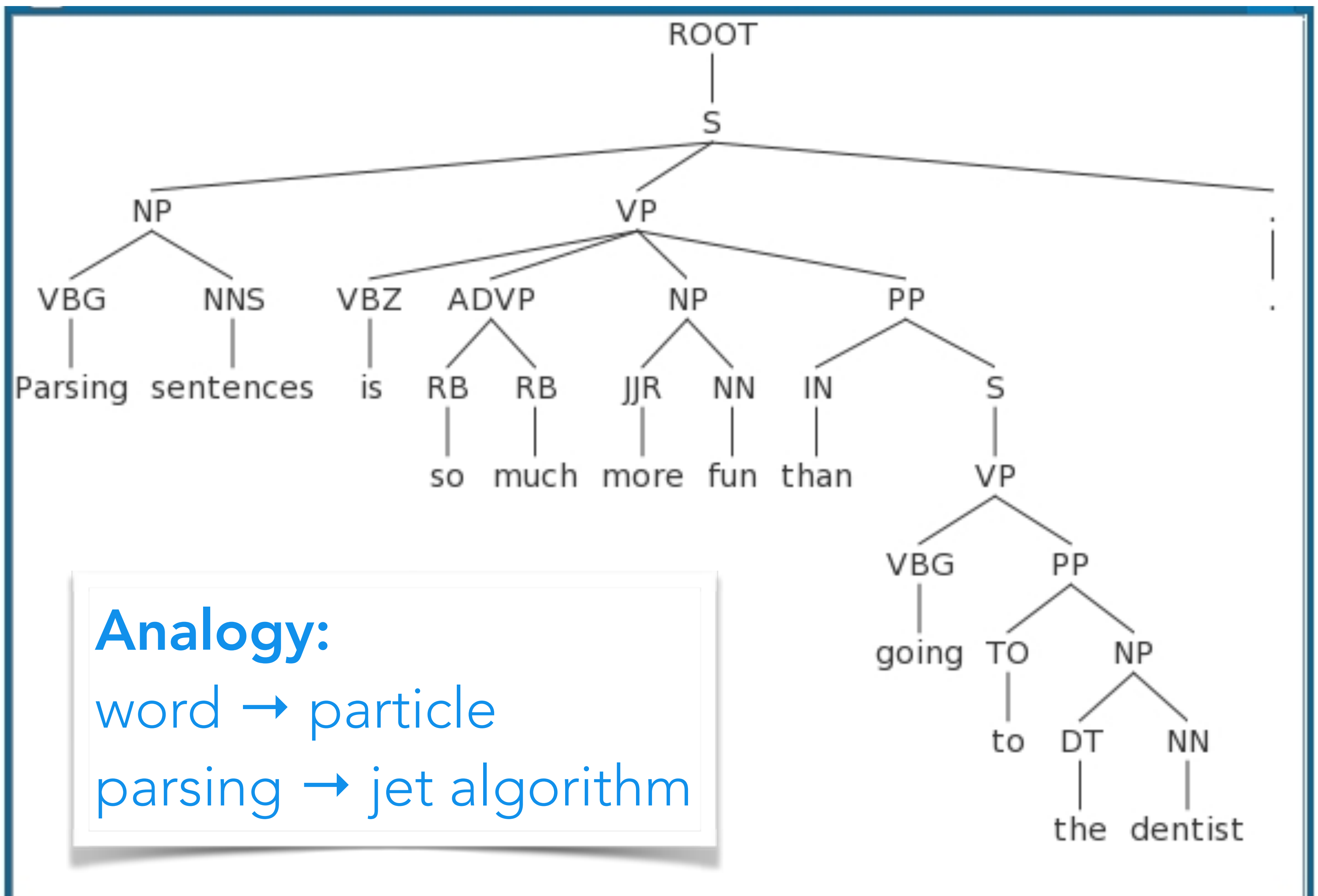
Recursive Neural Networks showing great performance for Natural Language Processing tasks

- neural network's topology given by parsing of sentence!



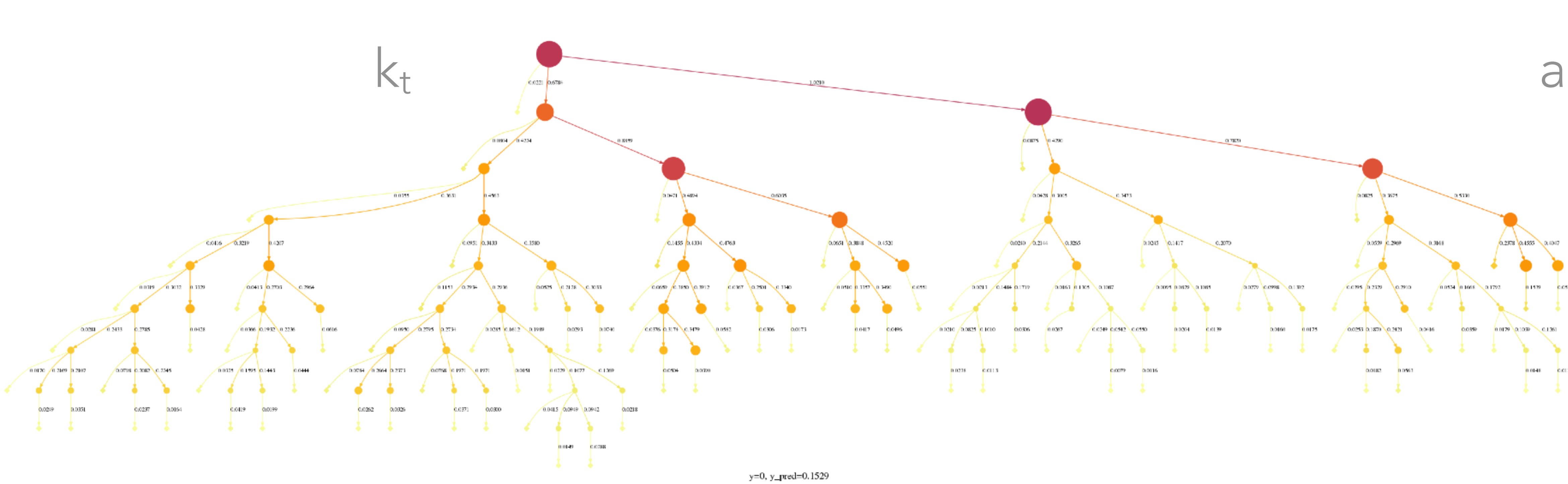
Recursive Neural Networks showing great performance for Natural Language Processing tasks

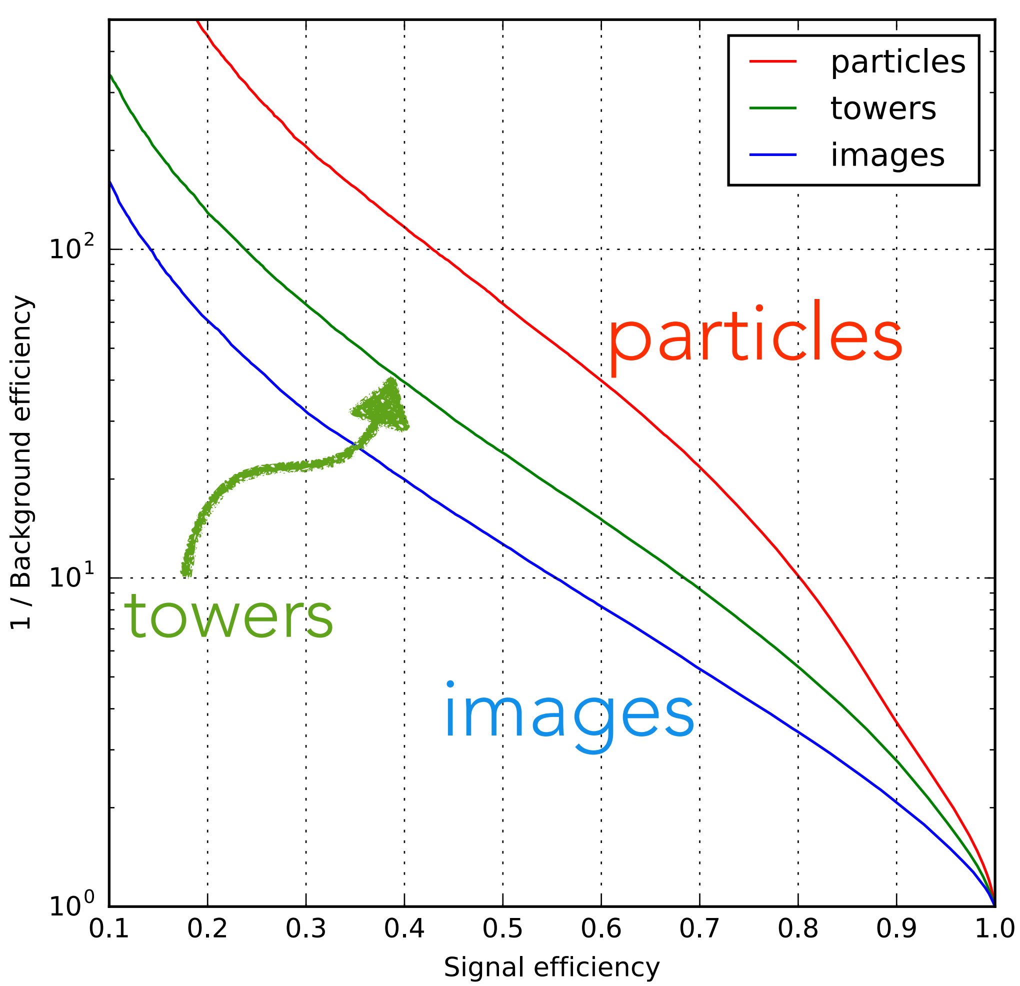
- neural network's topology given by parsing of sentence!



Analogy:
word → particle
parsing → jet algorithm

QCD-INSPIRED RECURSIVE NEURAL NETWORKS



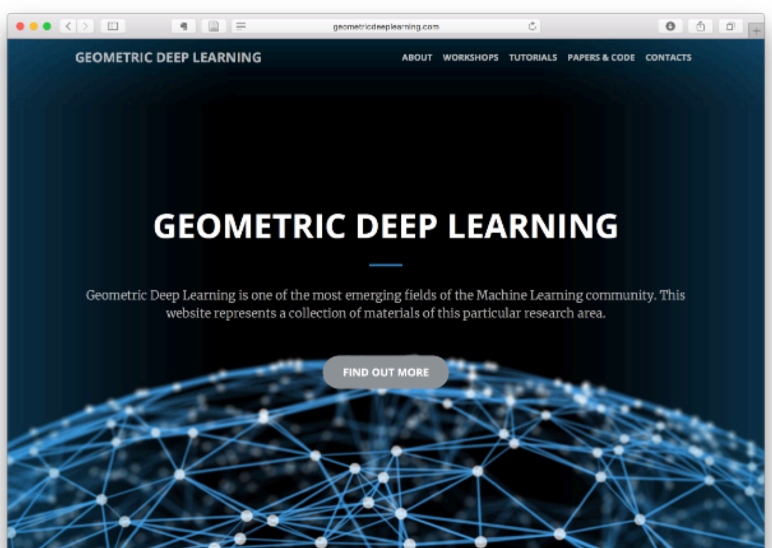


- Generative process is a tree-like, ~stationary Markov Process
- Physics algorithms exist to estimate the tree
- Tree-RNN needs much less data to train!

Neural Message Passing for Jet Physics

Isaac Henrion, Johann Brehmer, Joan Bruna, Kyunghyun Cho, Kyle Cranmer, Gilles Louppe, Gaspar Rochette

Courant Institute & Center for Data Science



NYU

Isaac Henrion

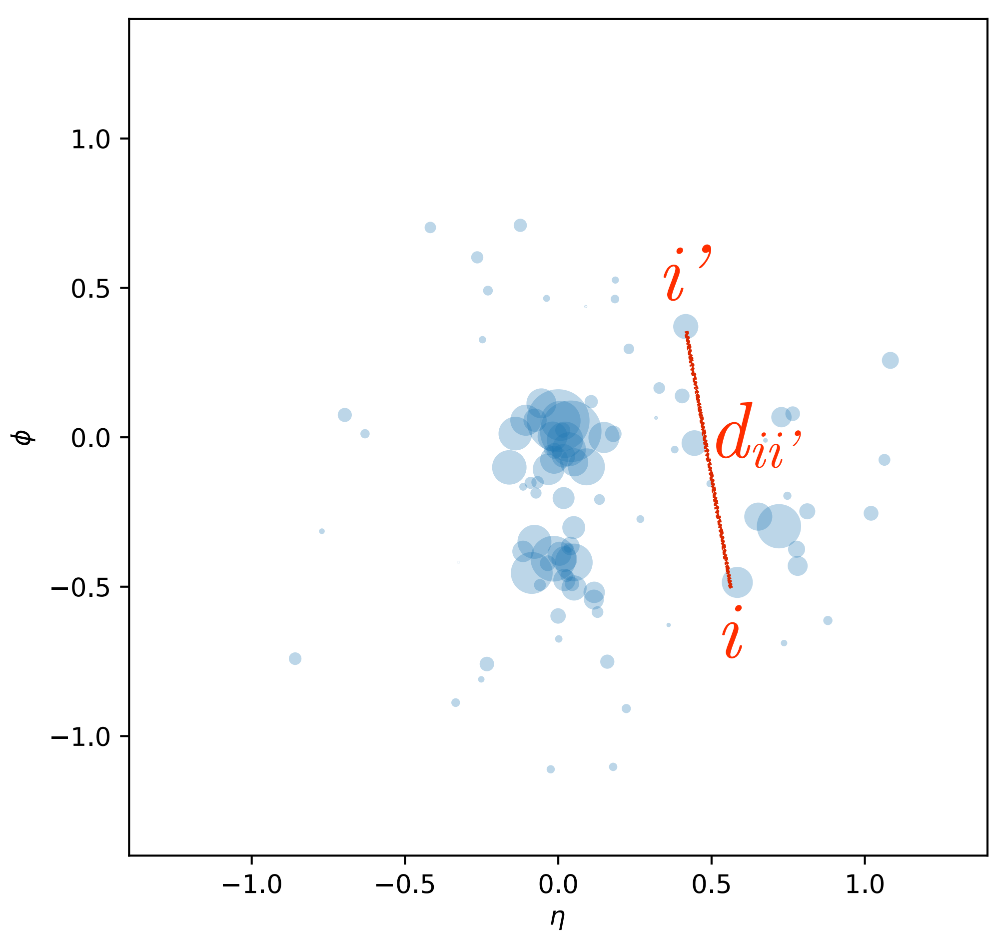
Paper: https://dl4physicalsciences.github.io/files/nips_dlps_2017_29.pdf

Talk: https://dl4physicalsciences.github.io/files/nips_dlps_2017_slides_henrion.pdf

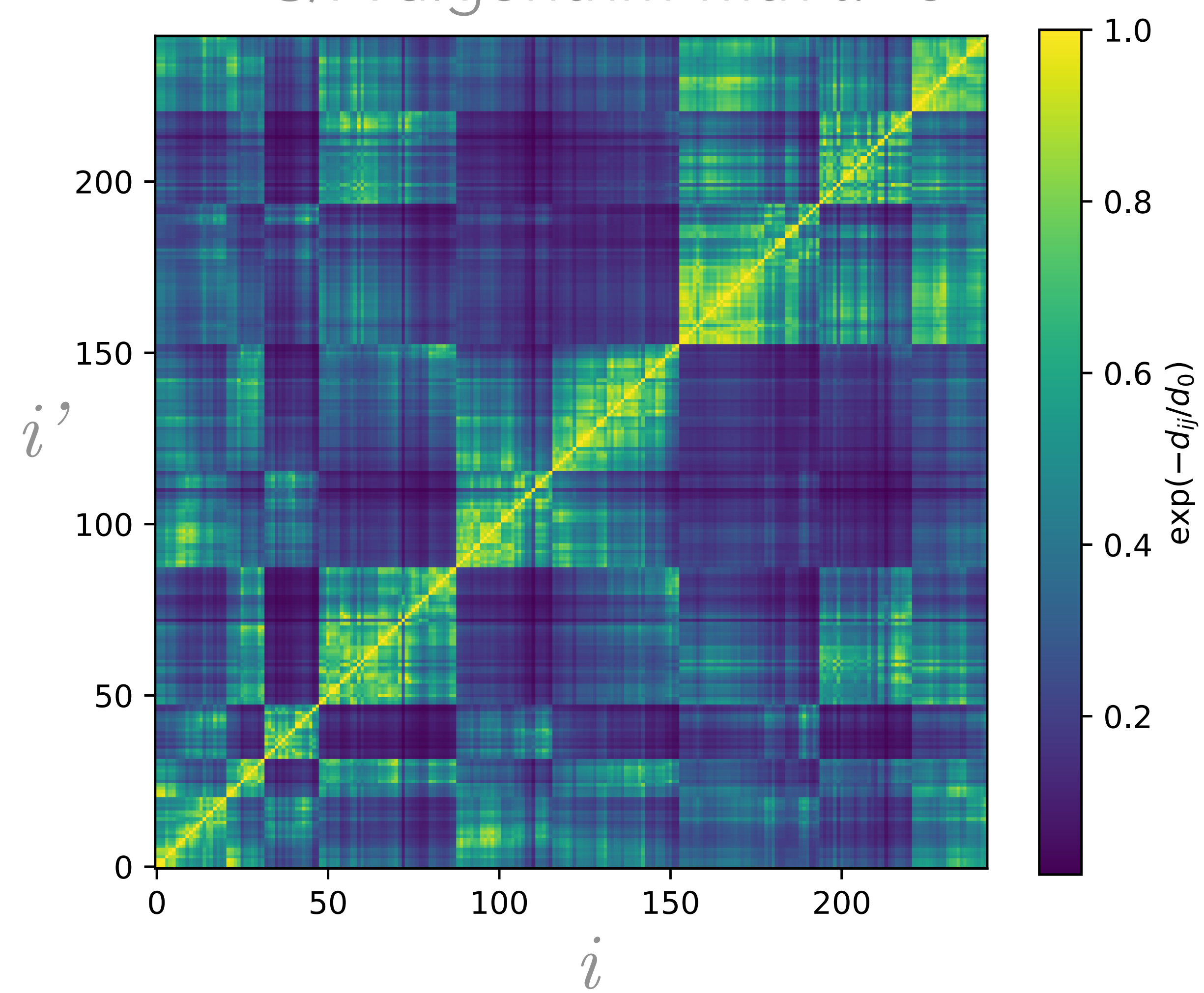
JETS AS A GRAPH

Using message passing neural networks over a fully connected graph on the particles

- Two approaches for adjacency matrix for edges
  - **import** physics knowledge by using metric of jet algorithms $d^{\alpha}_{ii'} = \min(p_{ti}^{2\alpha}, p_{ti'}^{2\alpha})\frac{\Delta R^2_{ii'}}{R^2}$
  - learn adjacency matrix and **export** new jet algorithm



C/A algorithm with α=0



1/FPR @ TPR = 50%

Model	Iterations	$R_{\epsilon=50\%}$
Rec-NN (no gating)	1	70.4 ± 3.6
Rec-NN (gating)	1	$\mathbf{83.3 \pm 3.1}$
MPNN (directed)	1	89.4 ± 3.5
MPNN (directed)	2	$\mathbf{98.3 \pm 4.3}$
MPNN (directed)	3	85.9 ± 8.5
MPNN (identity)	3	74.5 ± 5.2
Relation Network	1	67.7 ± 6.8

Significant improvement on W vs. QCD jet classification!

This is with a learned adjacency matrix

- what did it learn? Is that adjacency matrix useful?
- we are working MPNN with QCD-motivated adjacency matrix

Example:

Optimizing Non-Differentiable Simulators

NEW! AVO

G. Louppe & K.C. arXiv:1707.07113

Adversarial Variational Optimization of Non-Differentiable Simulators

Gilles Louppe[1] and Kyle Cranmer[1]

[1] *New York University*

Complex computer simulators are increasingly used across fields of science as generative models tying parameters of an underlying theory to experimental observations. Inference in this setup is often difficult, as simulators rarely admit a tractable density or likelihood function. We introduce Adversarial Variational Optimization (AVO), a likelihood-free inference algorithm for fitting a non-differentiable generative model incorporating ideas from empirical Bayes and variational inference. We adapt the training procedure of generative adversarial networks by replacing the differentiable generative network with a domain-specific simulator. We solve the resulting non-differentiable mini-max problem by minimizing variational upper bounds of the two adversarial objectives. Effectively, the procedure results in learning a proposal distribution over simulator parameters, such that the corresponding marginal distribution of the generated data matches the observations. We present results of the method with simulators producing both discrete and continuous data.

Leo is G Tom is D

Similar to W-GAN setup, but instead of using a neural network as the generator, use the actual simulation (eg. Pythia, GEANT)

Continue to use a neural network discriminator / critic.

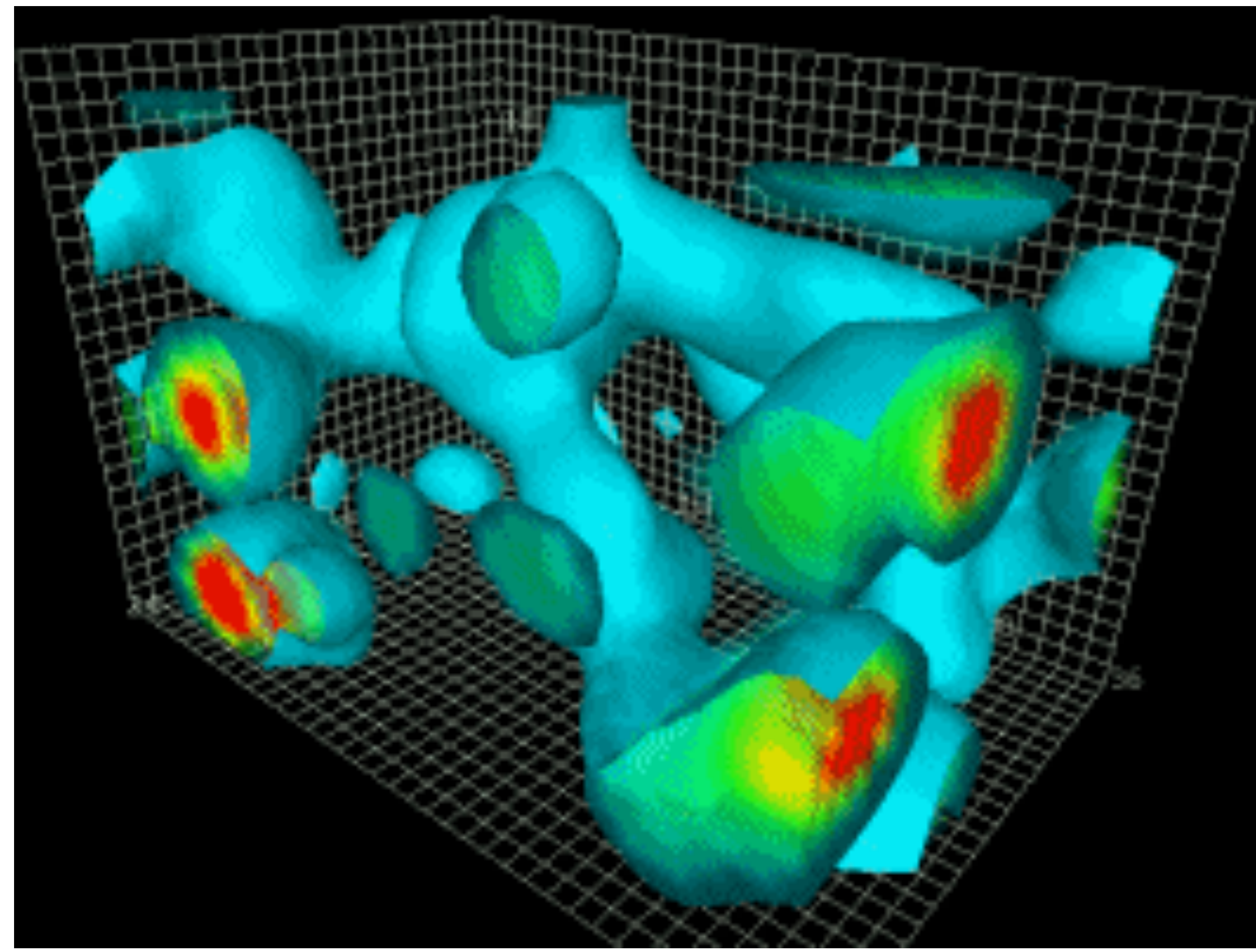
Difficulty: the simulator isn't differentiable, but there's a **trick**!

Allows us to efficiently fit / **tune simulation** with stochastic gradient techniques!

Example:

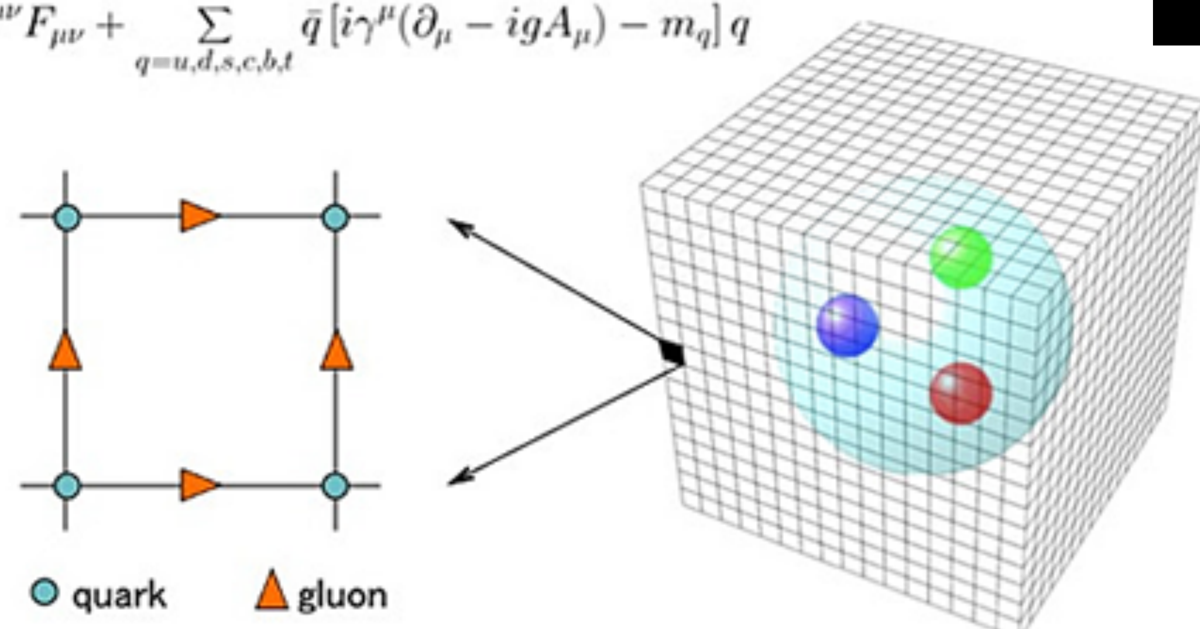
Lattice Field Theory

Very expensive simulations, ~1000 examples with $x \in \mathbb{R}^{10^9}$

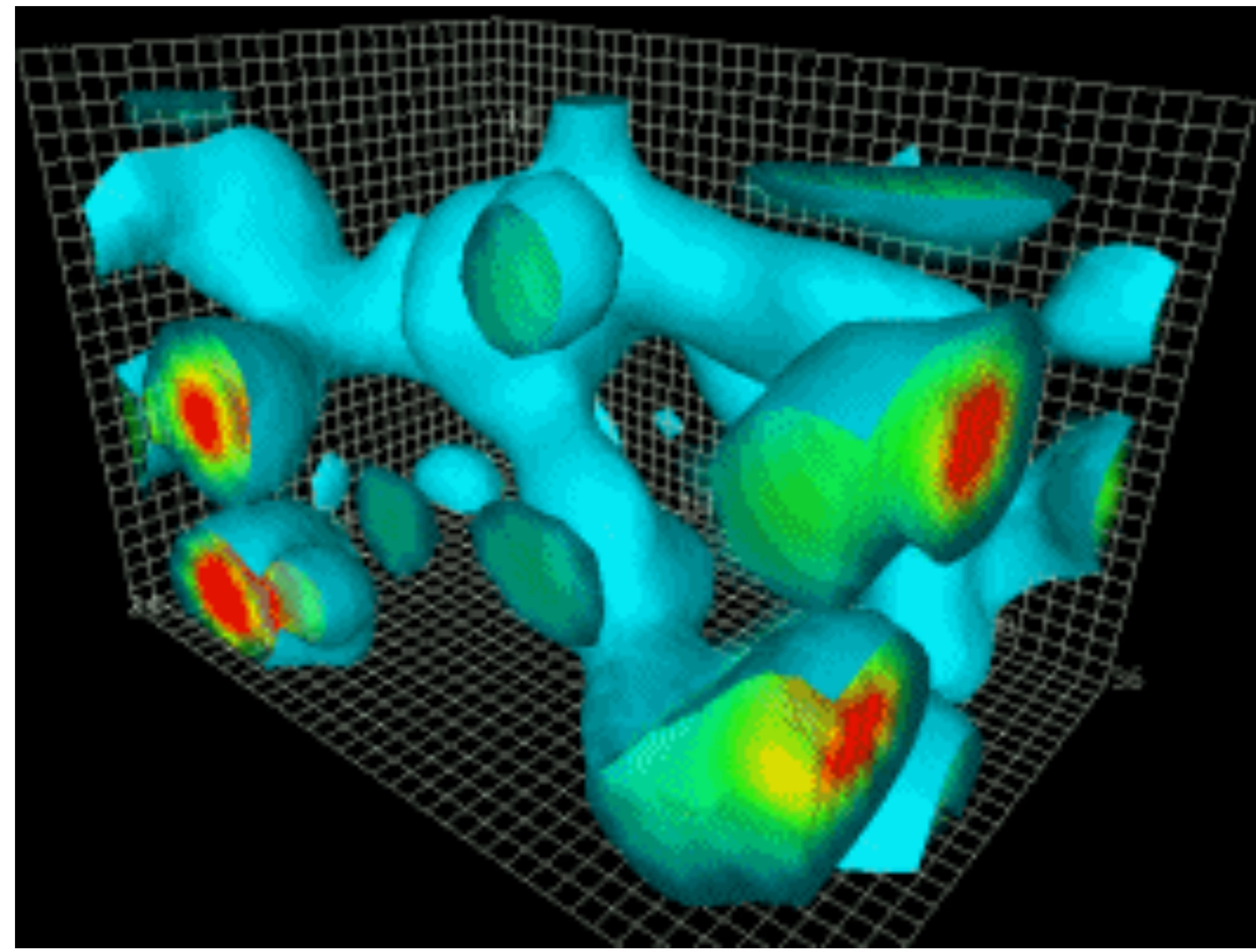


QCD Lagrangian

$$\mathcal{L} = -\frac{1}{4}F^{\mu\nu}F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q}\left[i\gamma^{\mu}(\partial_{\mu} - igA_{\mu}) - m_q\right]q$$

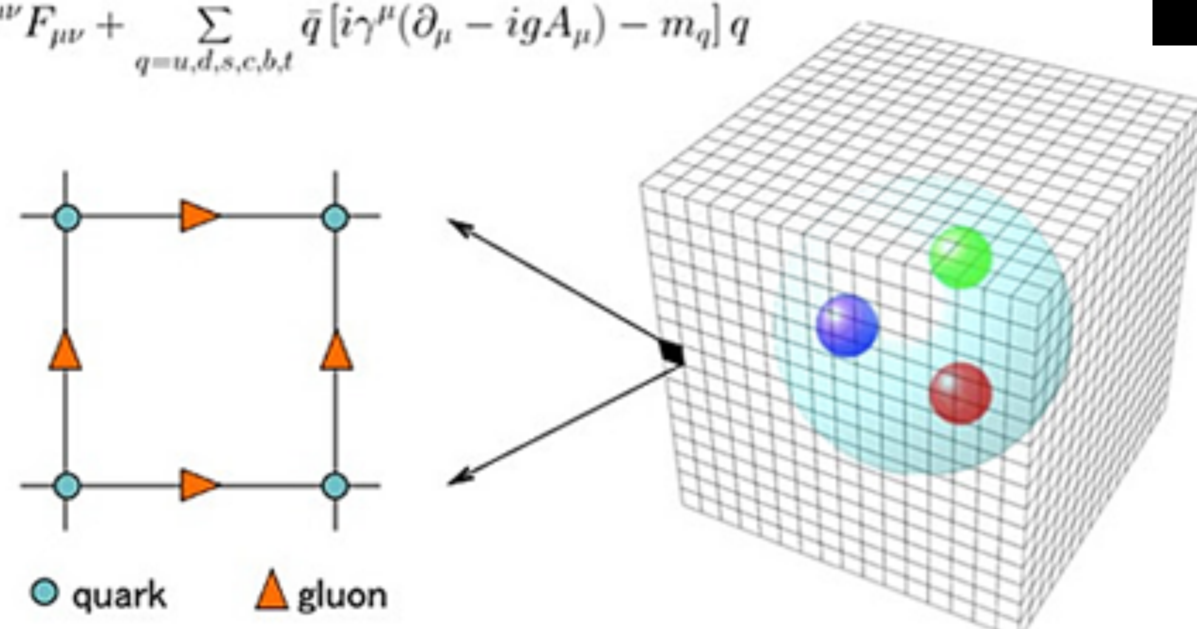


Very expensive simulations, ~1000 examples with $x \in \mathbb{R}^{10^9}$



QCD Lagrangian

$$\mathcal{L} = -\frac{1}{4}F^{\mu\nu}F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q}\left[i\gamma^{\mu}(\partial_{\mu} - igA_{\mu}) - m_q\right]q$$

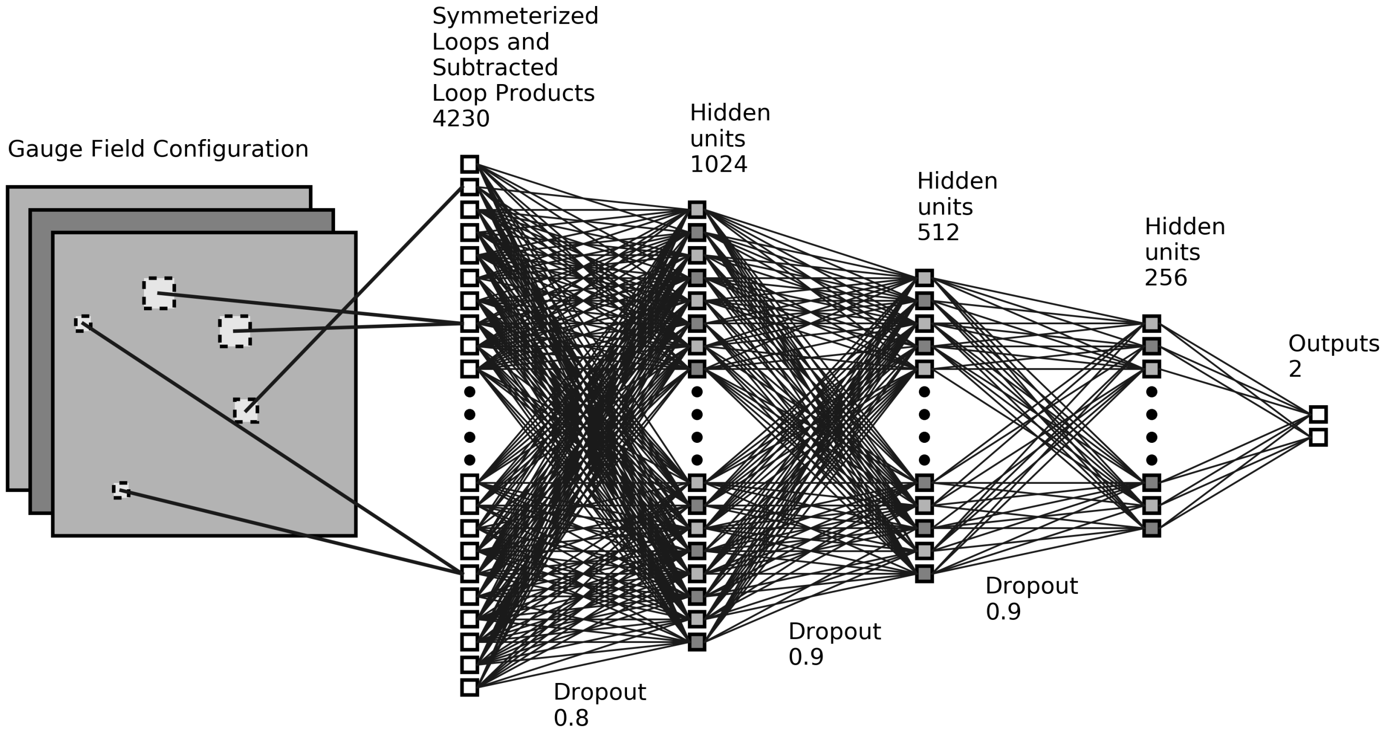


LATTICE QCD

Each of the 10^7 lattice locations has data $x_i \in \mathbb{R}^{32}$ with non-trivial data with continuous local symmetry.

- space-time translation invariance → convolutional architecture
- local gauge symmetry → design group-invariant convolutional filters
- coarse graining & renormalization group → hierarchical convolutions shared weights
- few very, large training examples → rethink minibatching & SGD

Bonus:
Network discovered something **unexpected**, a feature that has a long auto-correlation time.



(c) SLCP

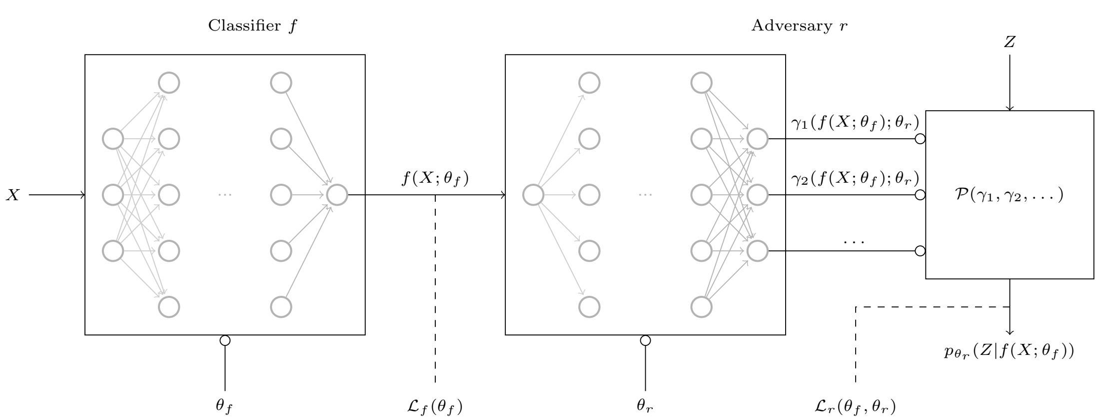
[explainer]

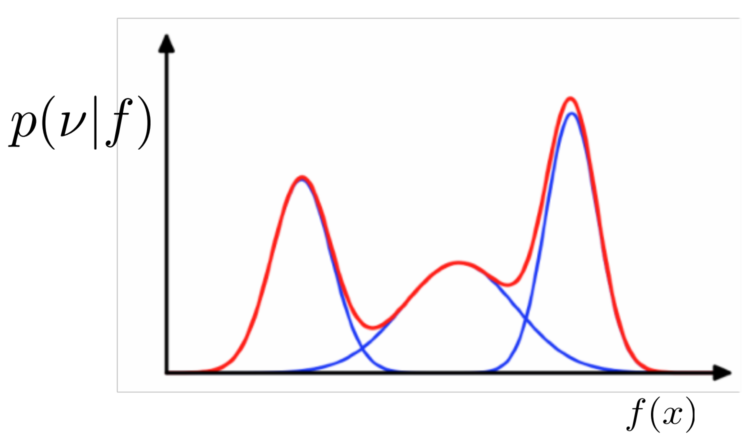
Example:

Systematics Uncertainty
Continuous Domain Adaptation
Fairness on Continuous Attributes

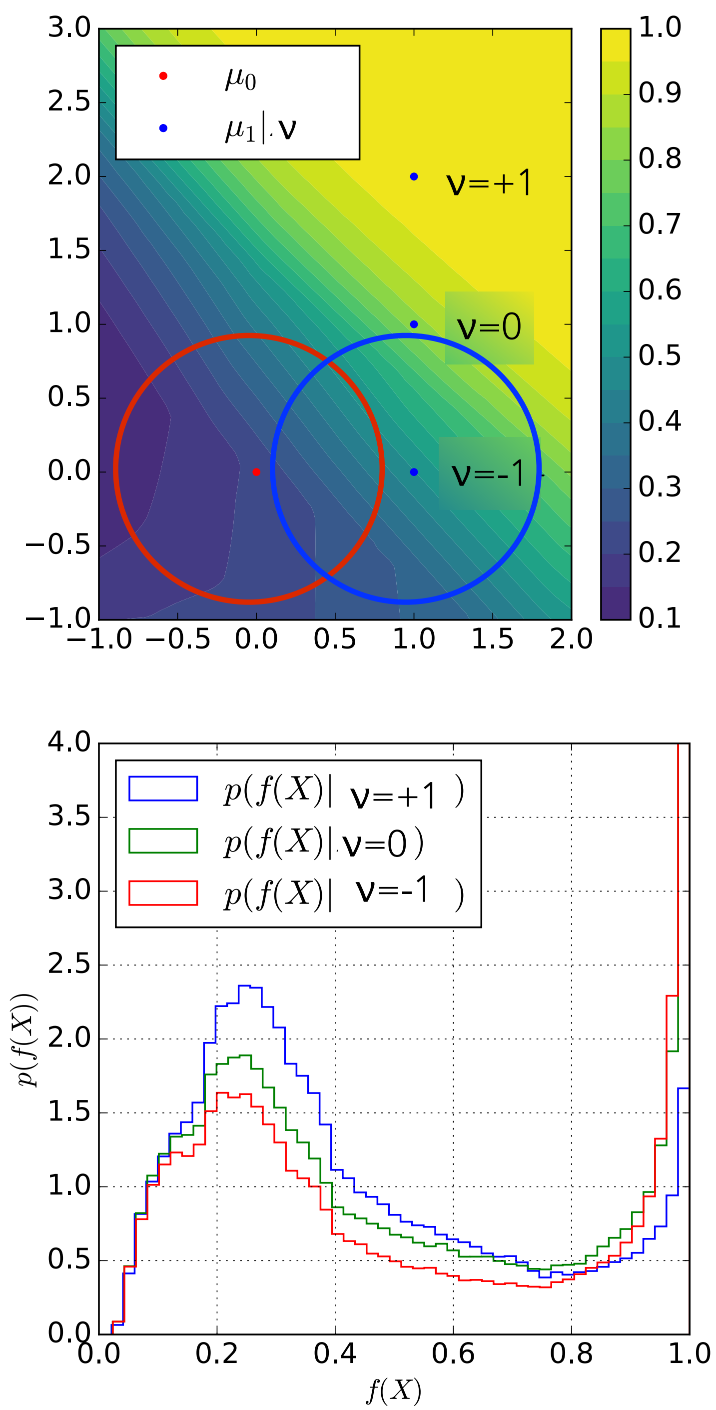
Typically classifier **f(x)** trained to minimize loss **L_f**.

- want classifier output to be insensitive to systematics (nuisance parameter **ν**)
- introduce an **adversary r** that tries to predict ν based on f.
- setup as a minimax game:

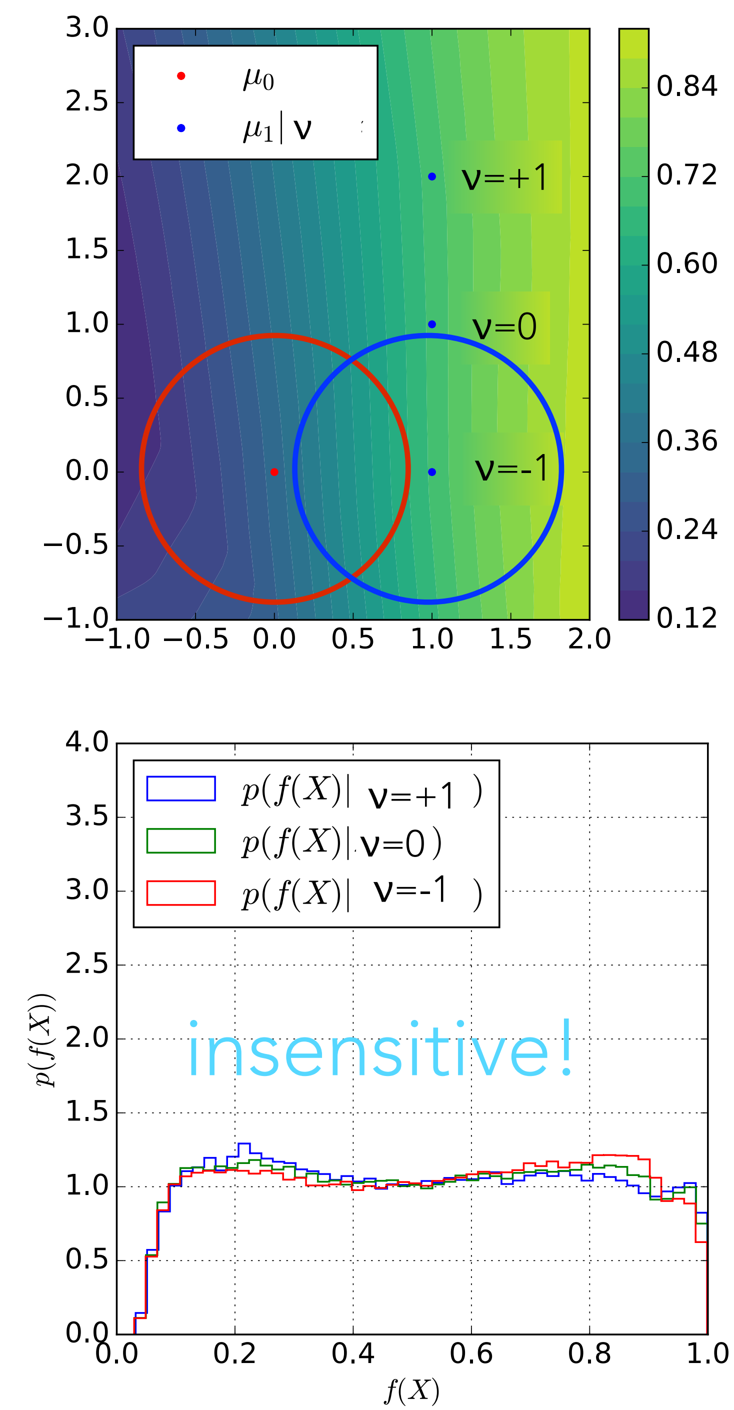




normal training



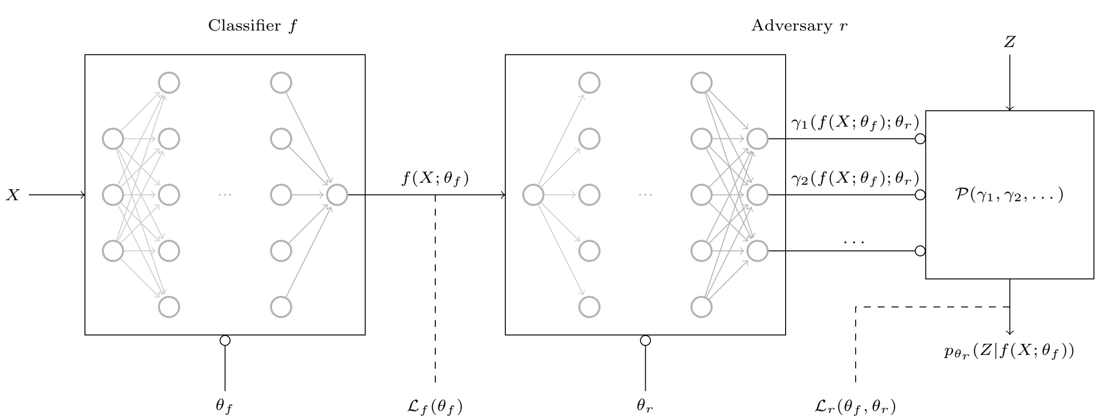
adversarial training

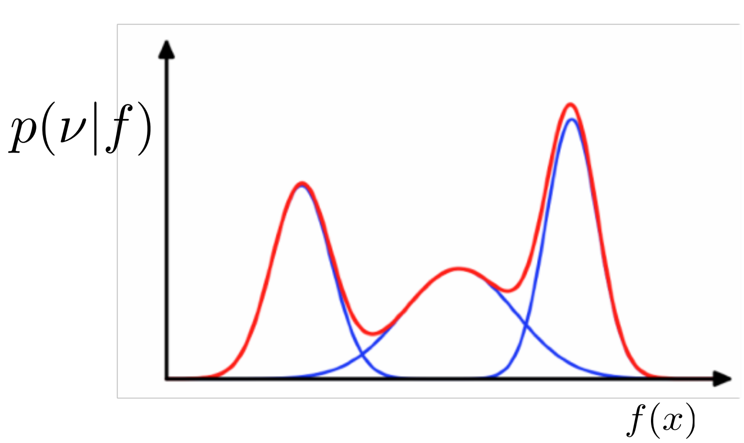


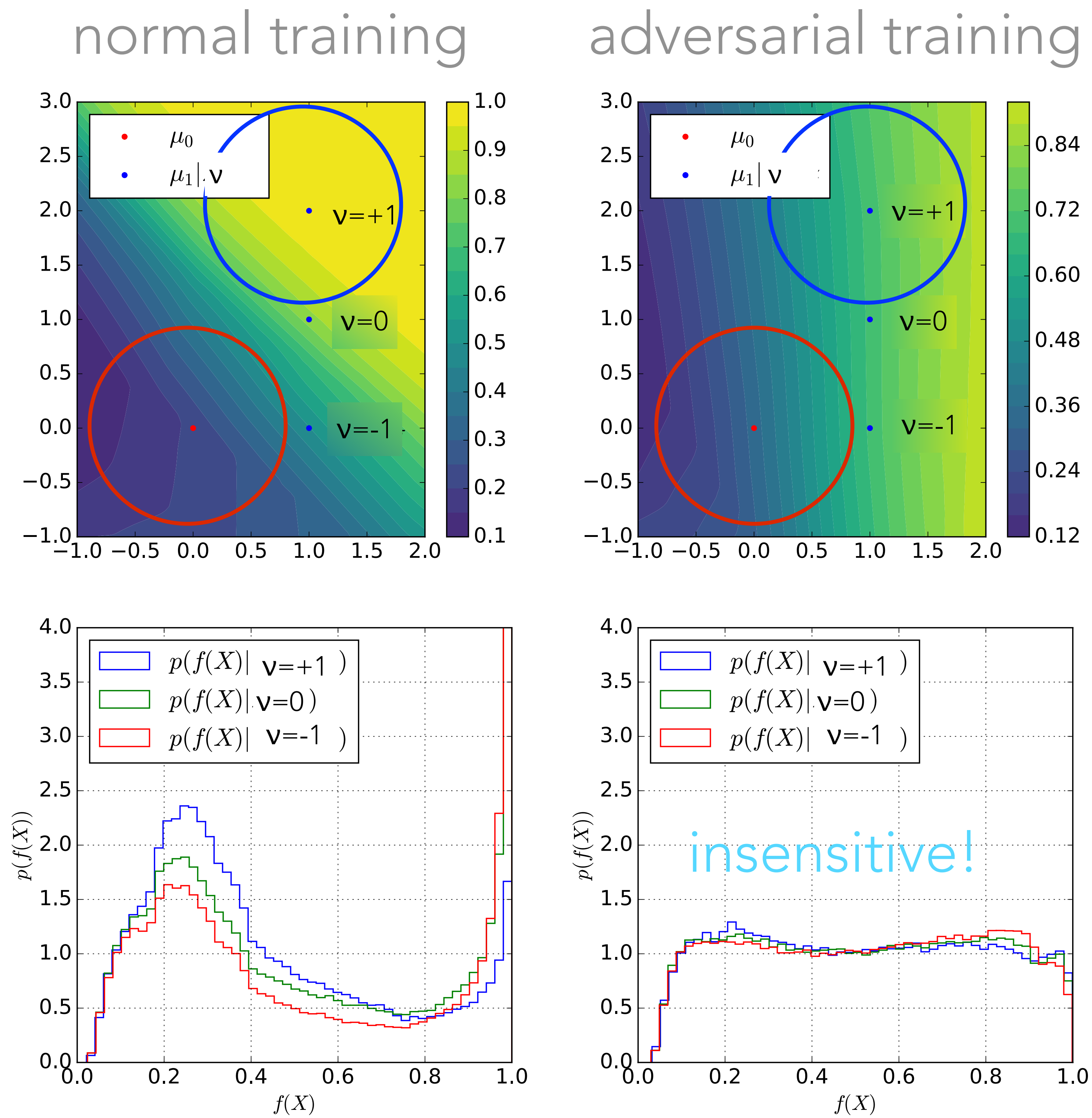
G. Louppe, M. Kagan, K. Cranmer, Learning to Pivot with Adversarial Networks [arXiv:1611.01046]

Typically classifier **f(x)** trained to minimize loss $\mathbf{L_f}$.

- want classifier output to be insensitive to systematics (nuisance parameter $\boldsymbol{\nu}$)
- introduce an **adversary r** that tries to predict ν based on f.
- setup as a minimax game:





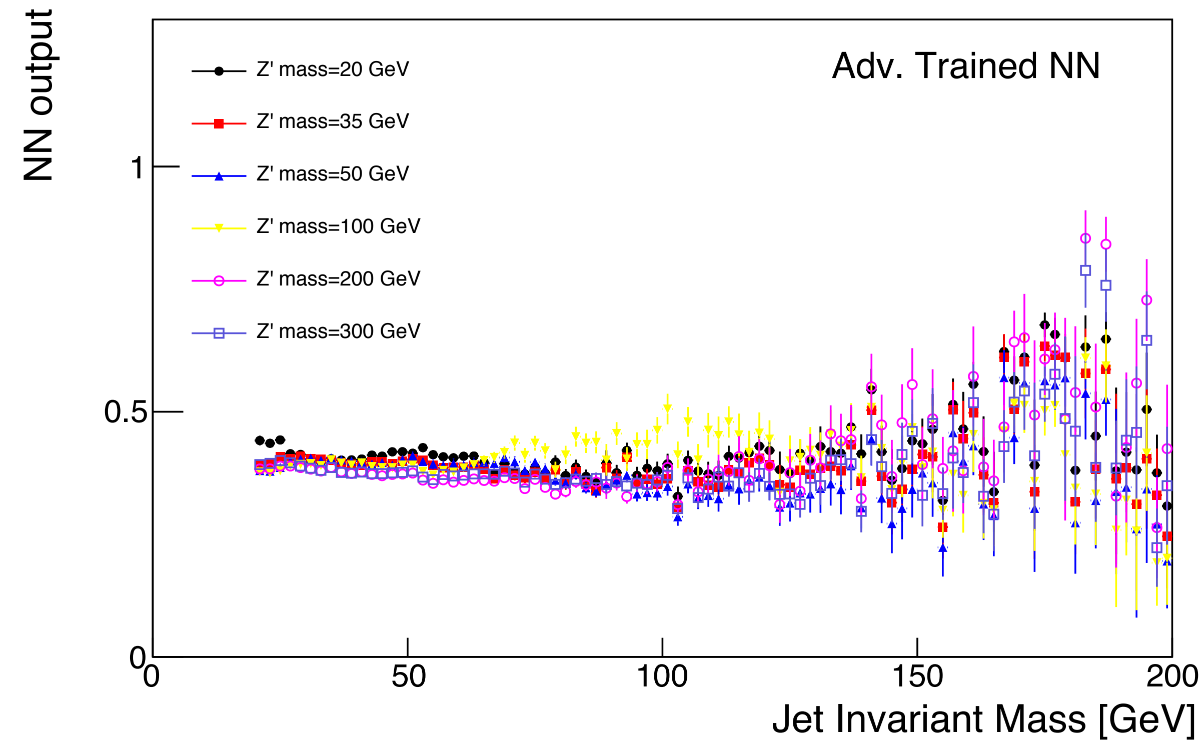


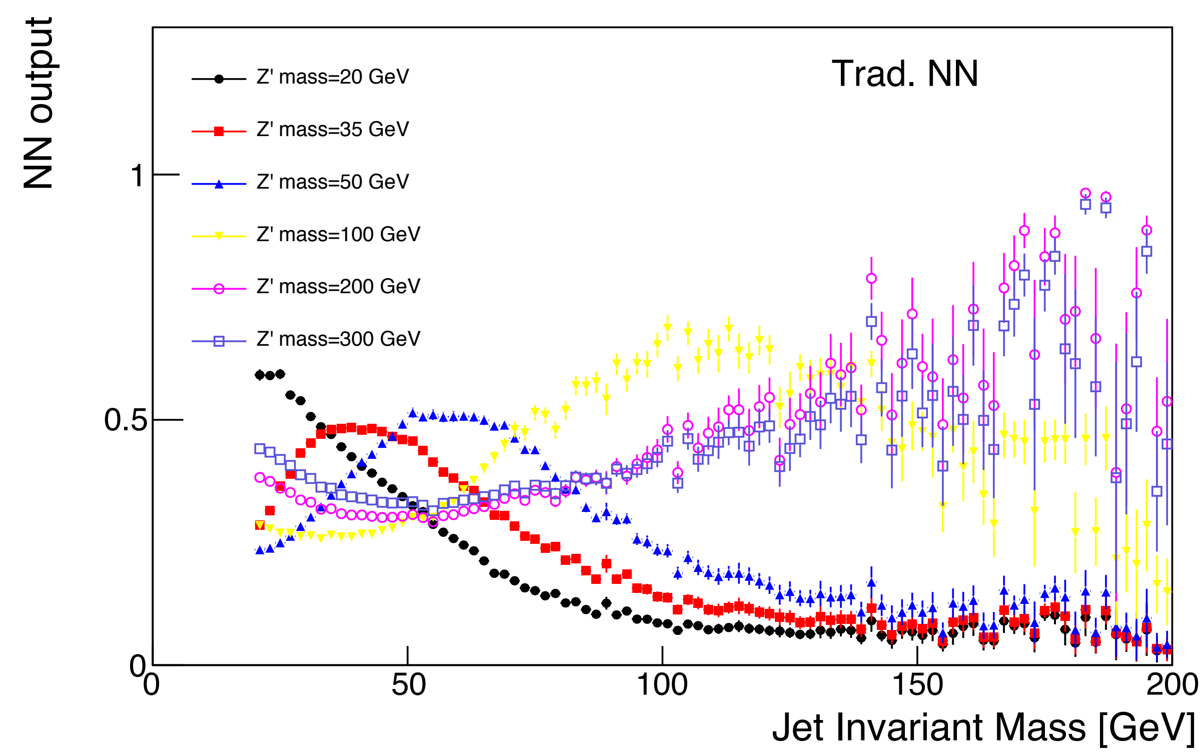
FAIR CLASSIFIERS

K.C, J. Pavez, and G. Louppe, arXiv:1506.02169
P. Baldi, K.C, T. Faucett, P. Sadowski, D. Whiteson arXiv:1601.07913
G. Louppe, M. Kagan, K.C, arXiv:1611.01046
Shimmin, et. al. arXiv:1703.03507

Adversarial approach of "Learning to Pivot" can also be used to train a classifier that is independent from some other continuous variable.

- fairness to continuous attribute
- motivation for doing this is related to robustnesss to uncertainties and interpretability





Example:

Reusable components

“Of course, particle physicists are among the first to realize that nature is compositional.”

– YANN LECUN

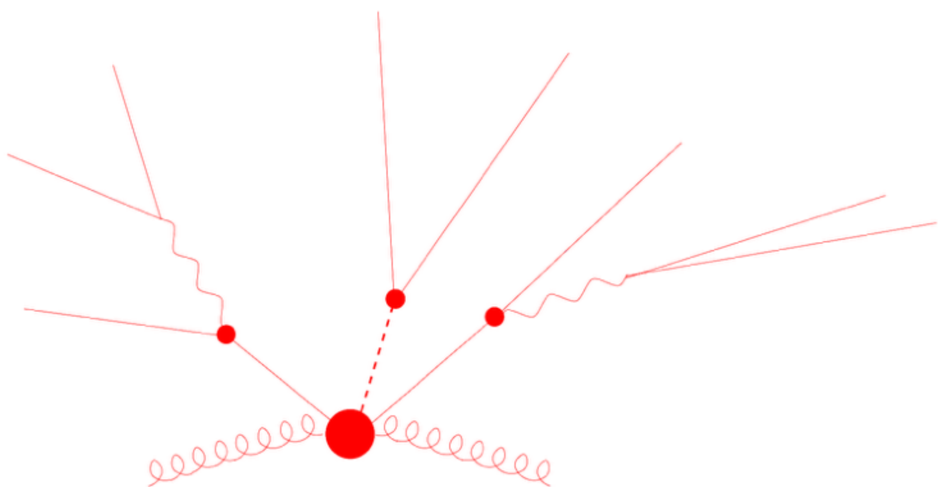
“The world is compositional, or there is a god"

– JASON EISNER

SUB-ATOMIC SCALE

pencil & paper calculable from first principles

$p(z_1 \mid \theta)$

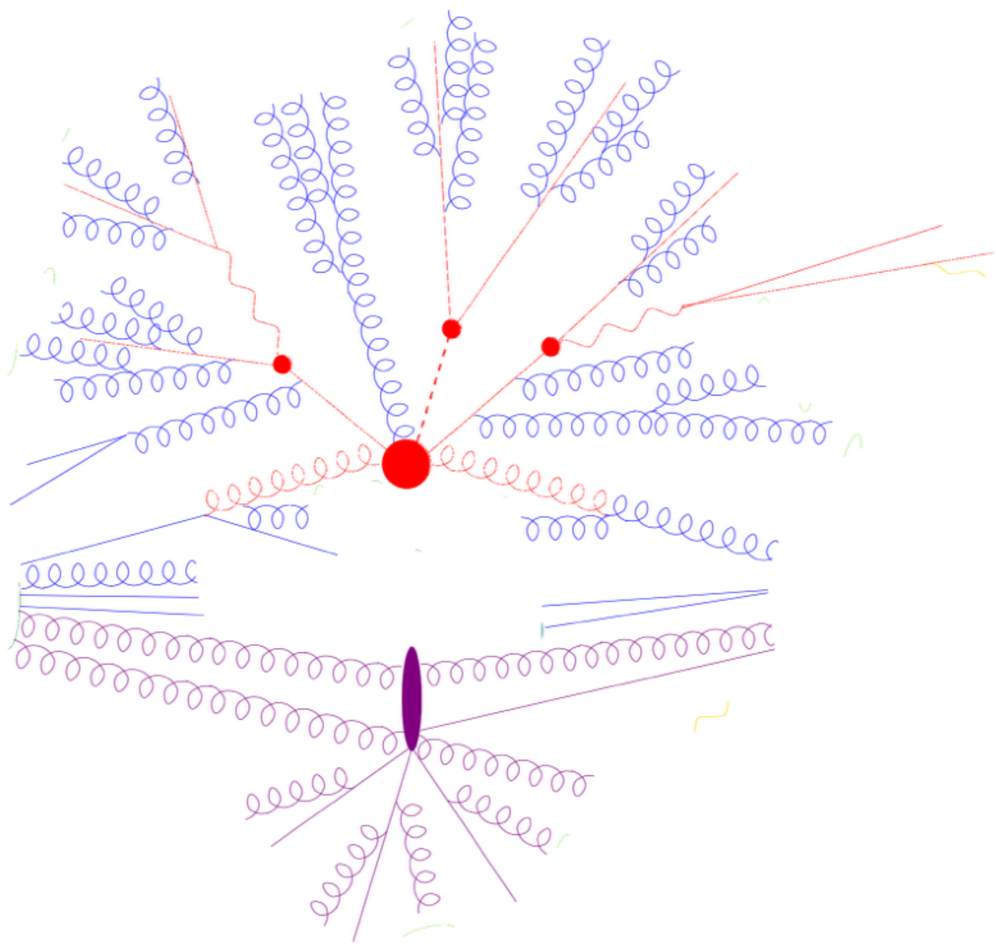


pencil & paper calculable from first principles

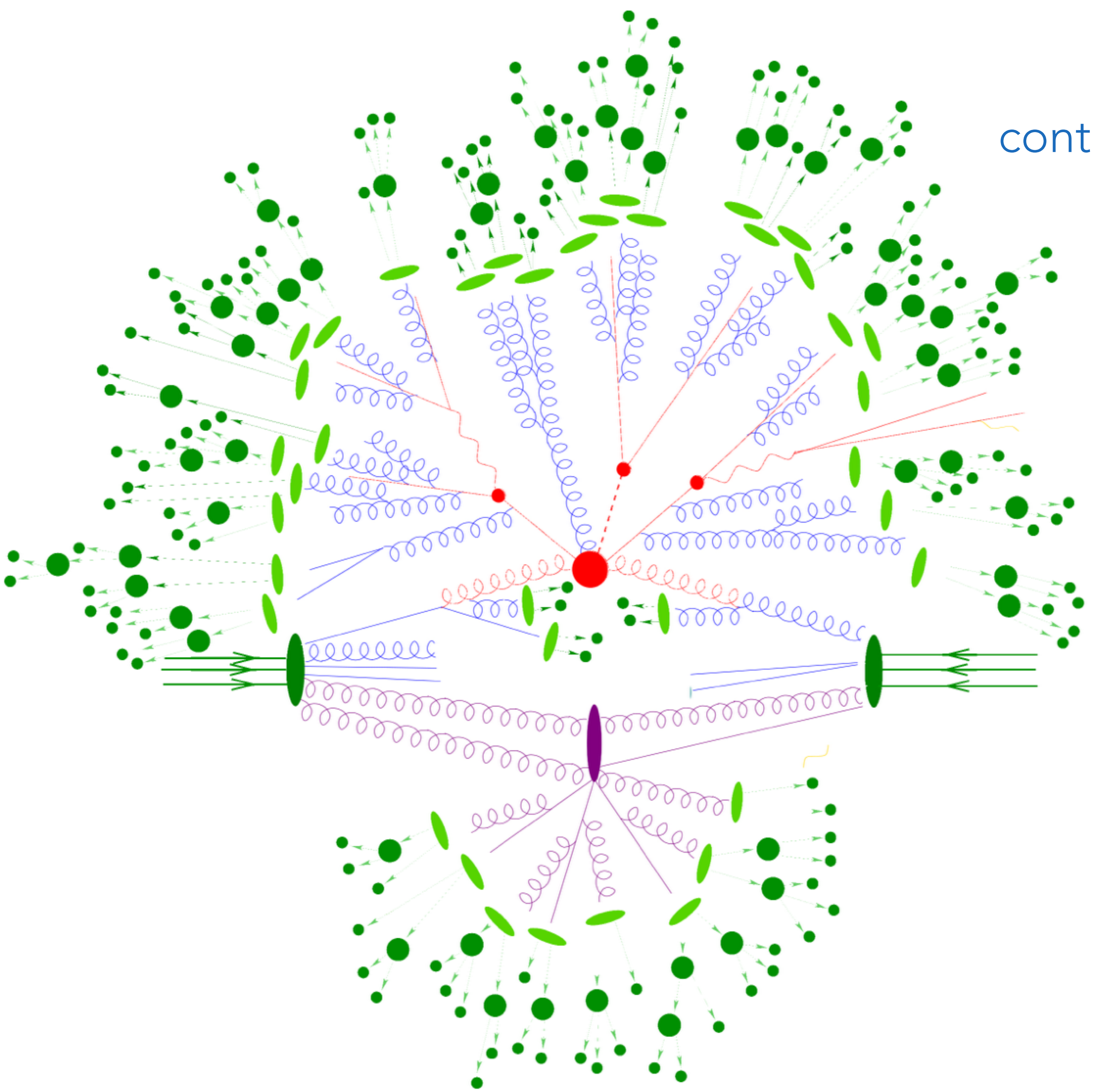
$p(z_1 \mid \theta)$

controlled approximation of first principles

$p(z_2 \mid z_1, \nu_1)$



SUB-ATOMIC SCALE



pencil & paper calculable from first principles

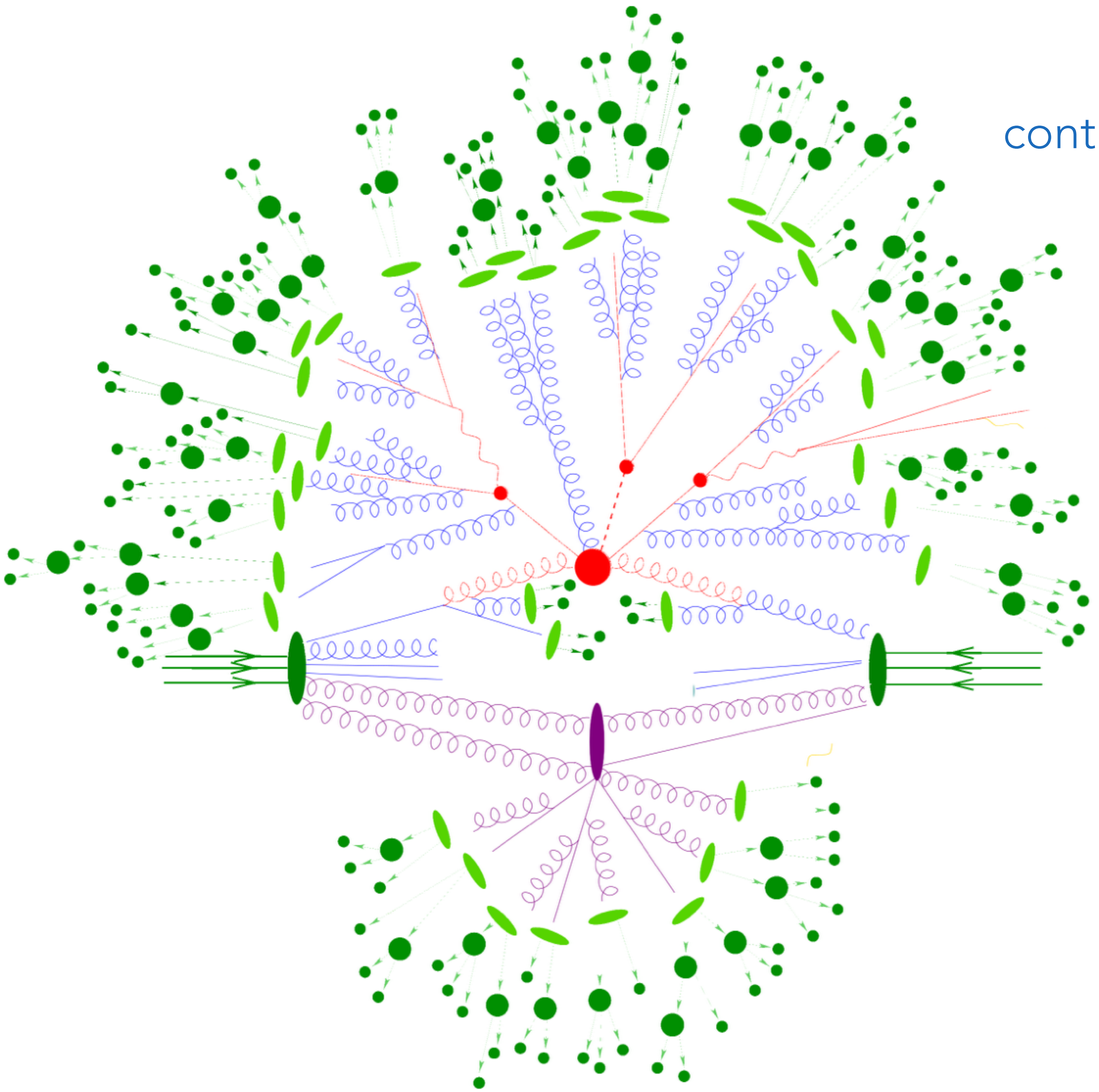
$p(z_1 \mid \theta)$

controlled approximation of first principles

$p(z_2 \mid z_1, \nu_1)$

phenomenological model

$p(z_3 \mid z_2, \nu_2)$



pencil & paper calculable from first principles

$p(z_1 \mid \theta)$

controlled approximation of first principles

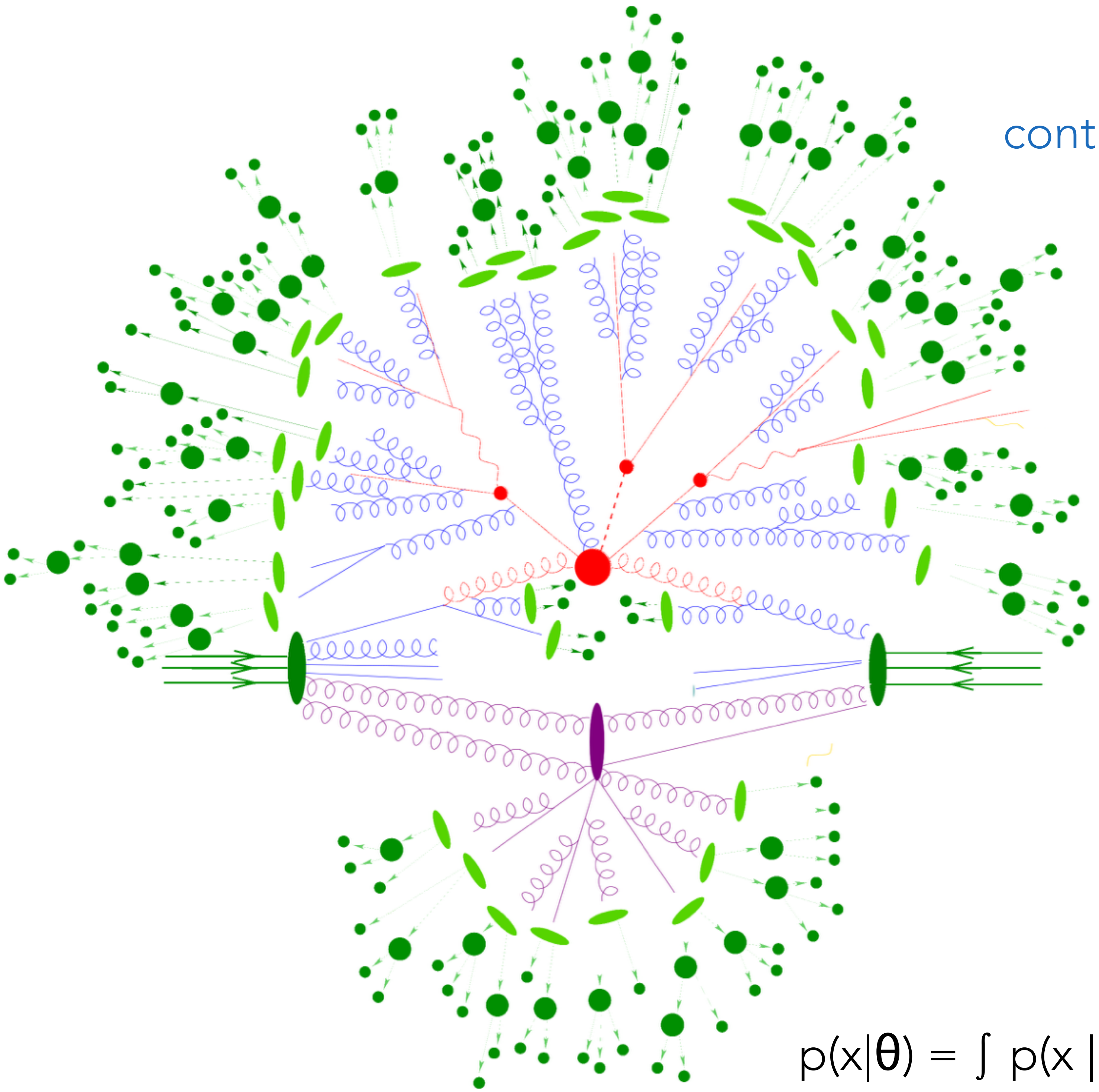
$p(z_2 \mid z_1, \nu_1)$

phenomenological model

$p(z_3 \mid z_2, \nu_2)$

Exploit Markov Property:

SUB-ATOMIC SCALE



pencil & paper calculable from first principles

$p(z_1 \mid \theta)$

controlled approximation of first principles

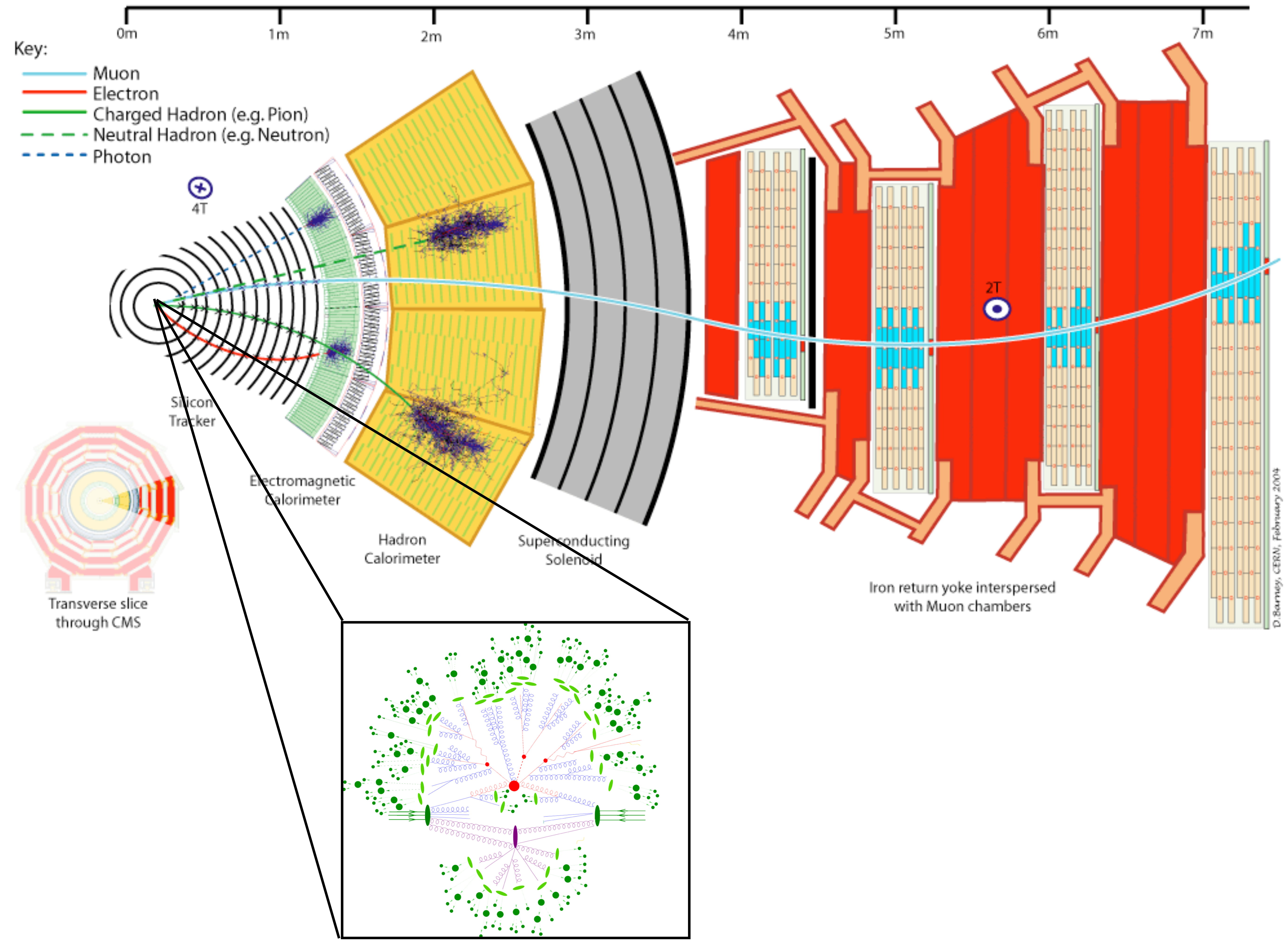
$p(z_2 \mid z_1, \nu_1)$

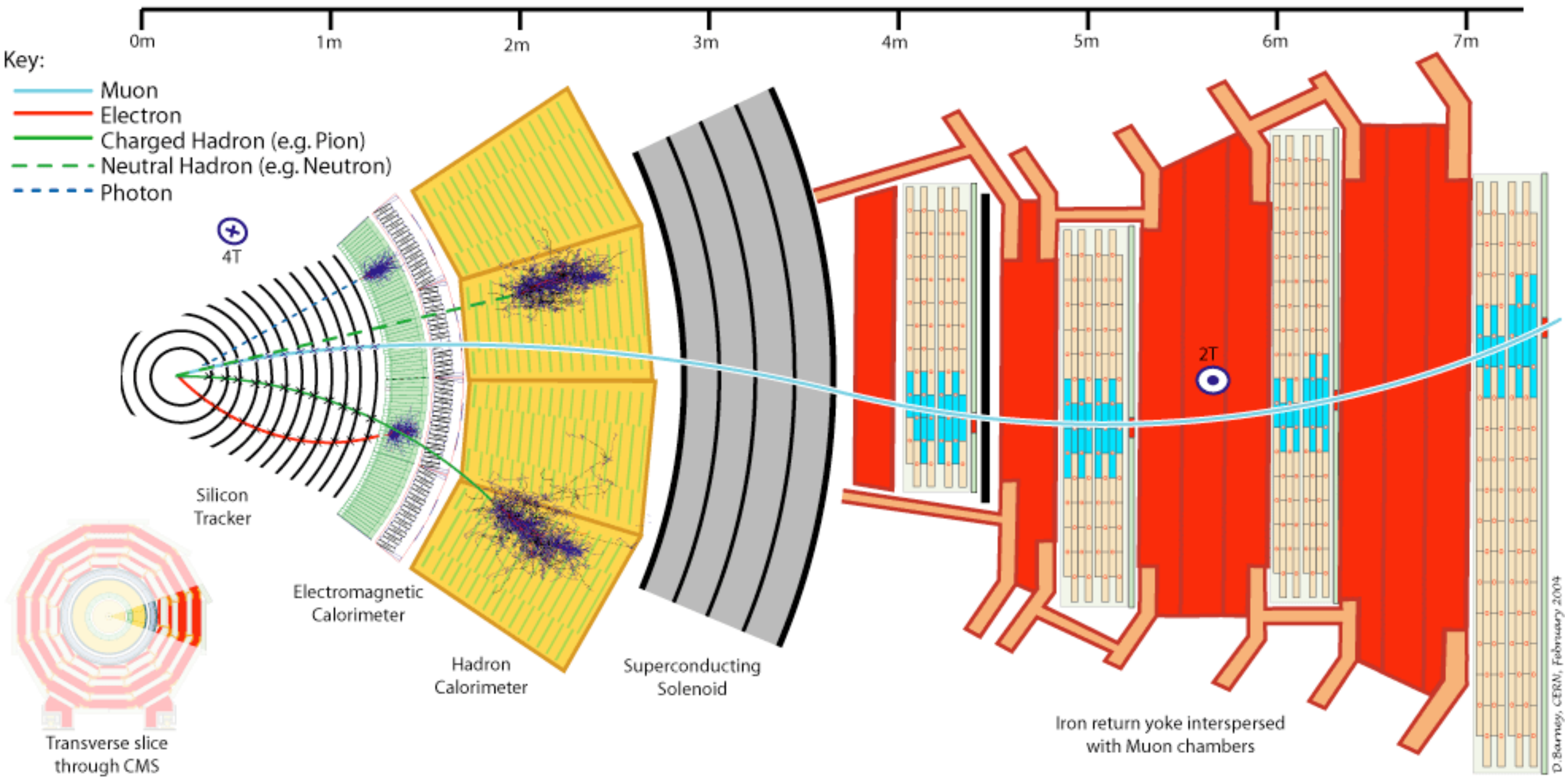
phenomenological model

$p(z_3 \mid z_2, \nu_2)$

Exploit Markov Property:

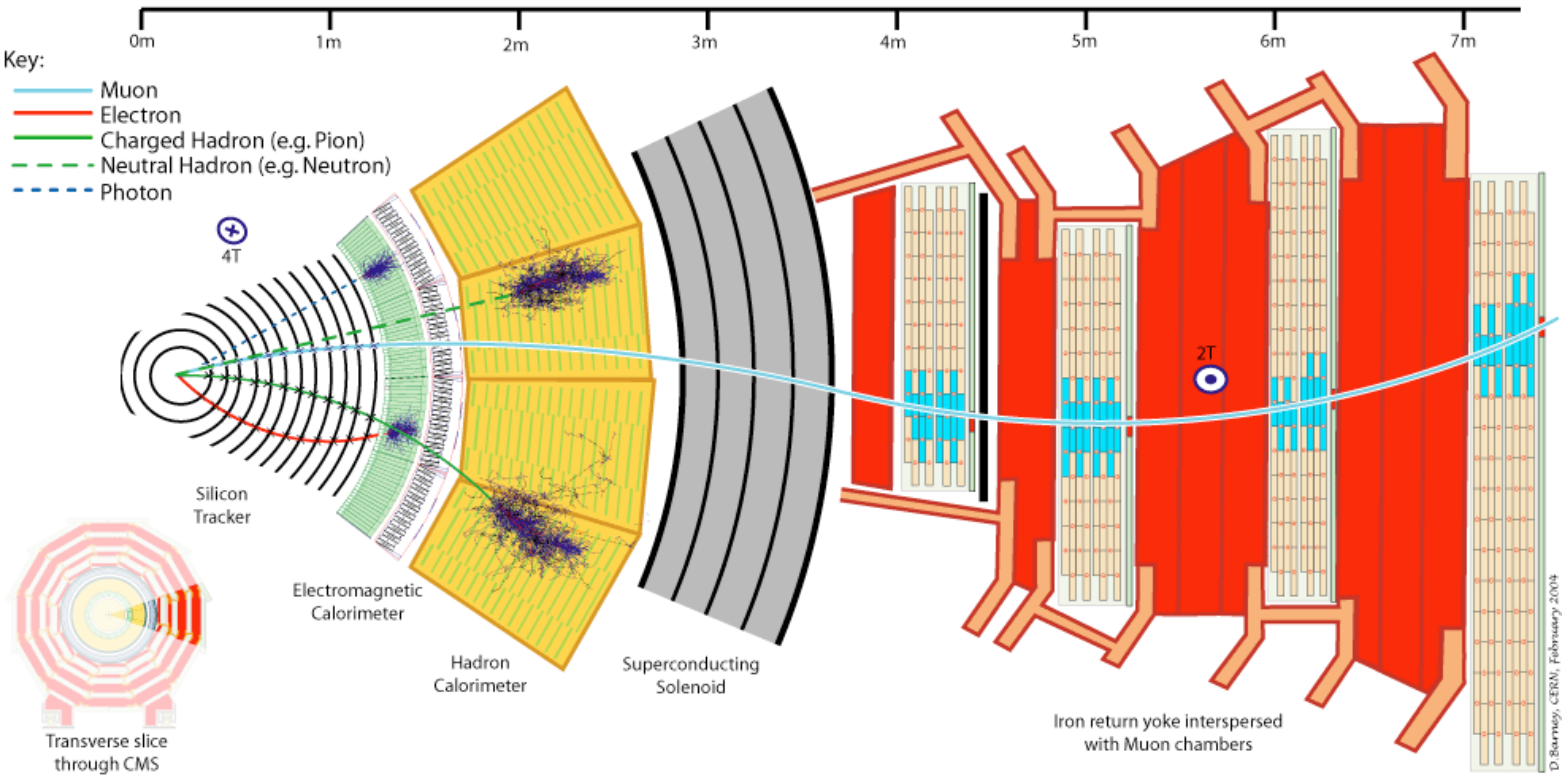
$$p(x|\theta) = \int p(x \mid z_3, \nu_3)\, p(z_3 \mid z_2, \nu_2)\, p(z_2 \mid z_1, \nu_1)\, p(z_1 \mid \theta)\, dz$$


0m
1m
2m
3m
4m
5m
6m
7m
Key:
Muon
Electron
Charged Hadron (e.g. Pion)
Neutral Hadron (e.g. Neutron)
Photon
4T
2T
Silicon Tracker
Electromagnetic Calorimeter
Hadron Calorimeter
Superconducting Solenoid
Iron return yoke interspersed with Muon chambers
Transverse slice through CMS
D.Barney, CERN, February 2004



Detector Simulation $\mathbf{p(x \mid z_3, \nu_3)}$:

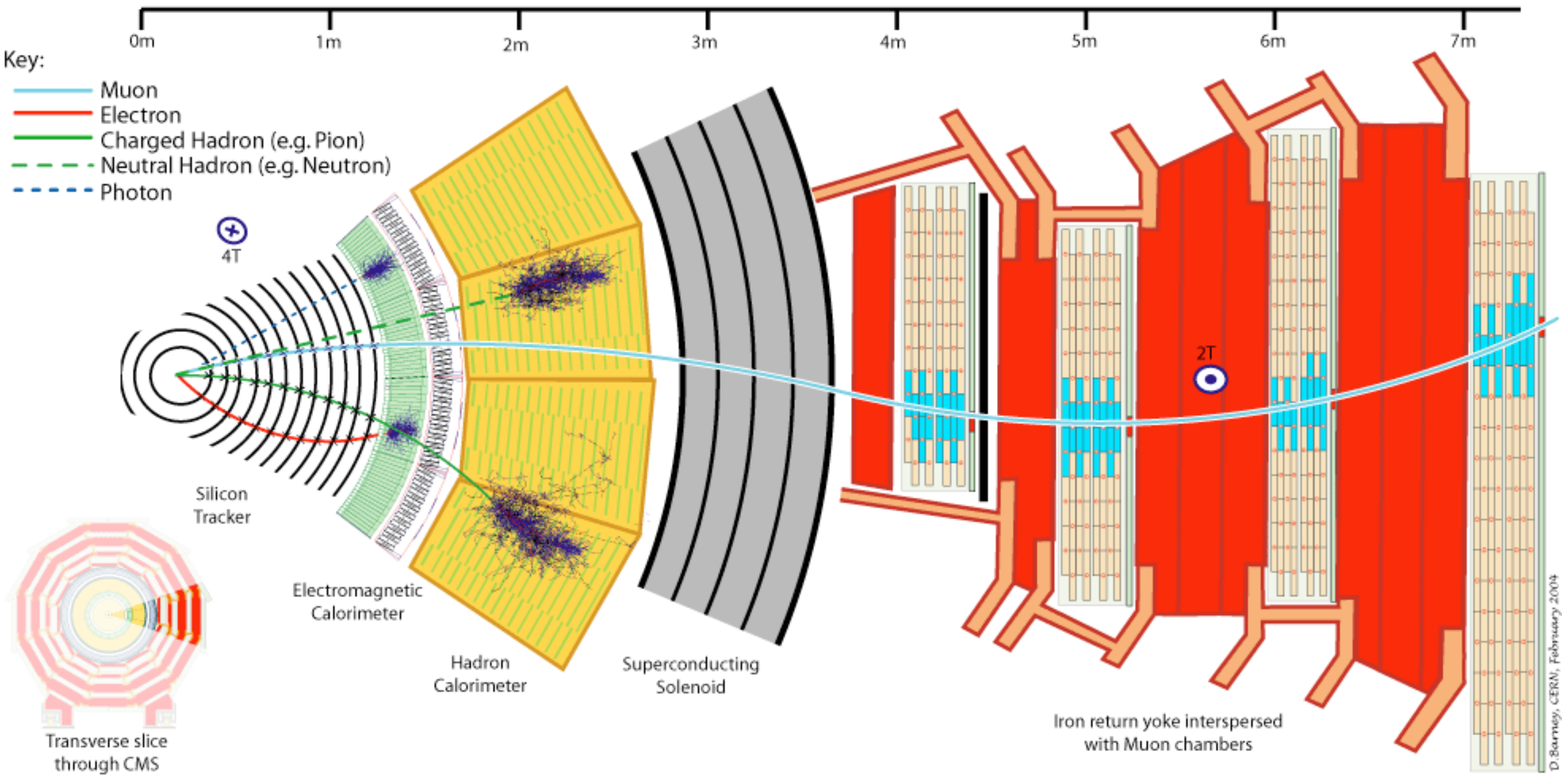
- detailed engineering (CAD)
- in situ measurements of temperature, magnetic field, alignment, calibration constants
- first-principles description of interaction of particles with matter
- look up tables of measured interaction of particles with matter



Detector Simulation **p(x | z_3, ν_3):**

- detailed engineering (CAD)
- in situ measurements of temperature, magnetic field, alignment, calibration constants
- first-principles description of interaction of particles with matter
- look up tables of measured interaction of particles with matter

Exploit Markov Property:

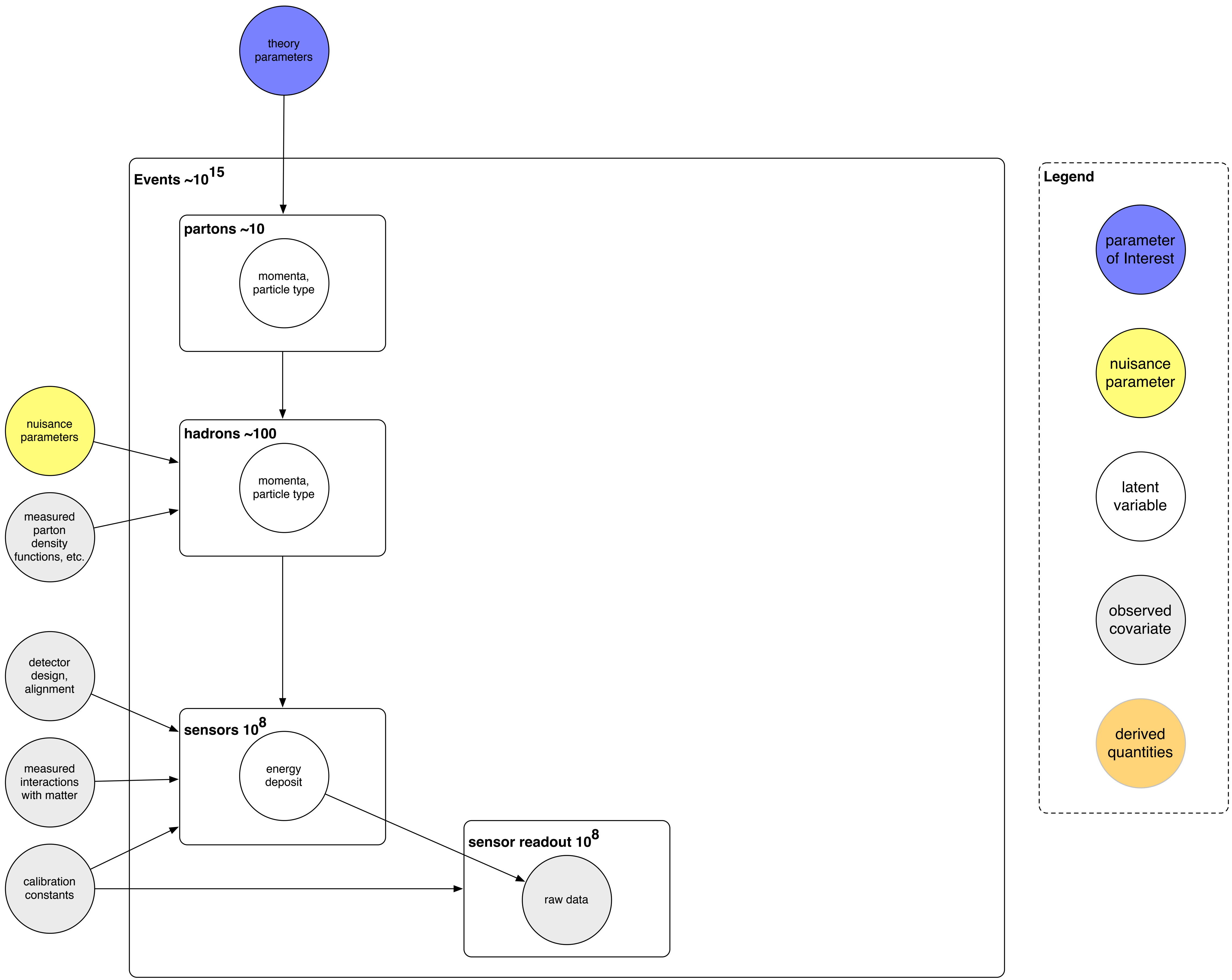


Detector Simulation $\mathbf{p(x \mid z_3, \nu_3)}$:

- detailed engineering (CAD)
- in situ measurements of temperature, magnetic field, alignment, calibration constants
- first-principles description of interaction of particles with matter
- look up tables of measured interaction of particles with matter

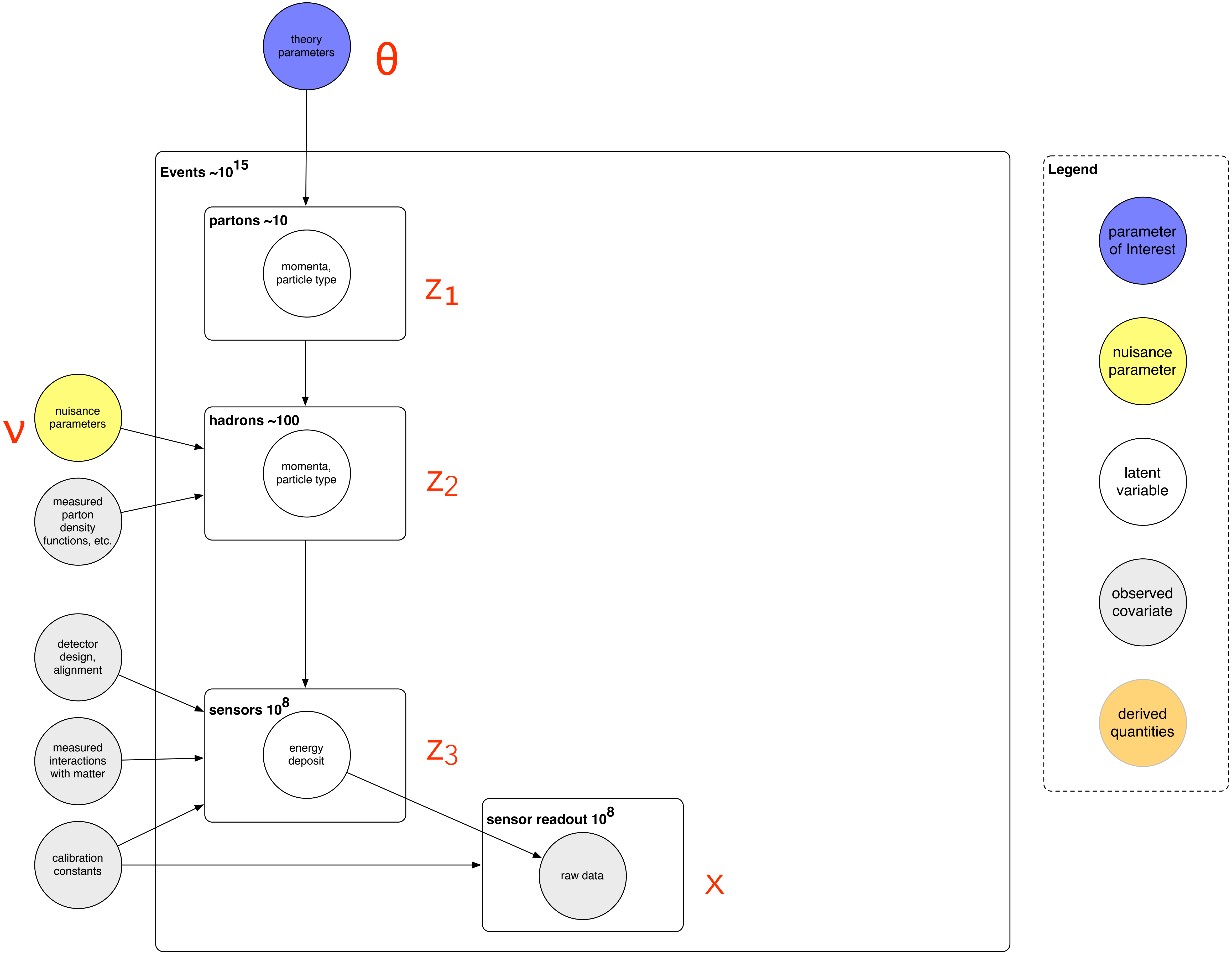
Exploit Markov Property: $p(x|\theta) = \int p(x \mid z_3, \nu_3)\, p(z_3 \mid z_2, \nu_2)\, p(z_2 \mid z_1, \nu_1)\, p(z_1 \mid \theta)\, dz$

SIMULATION



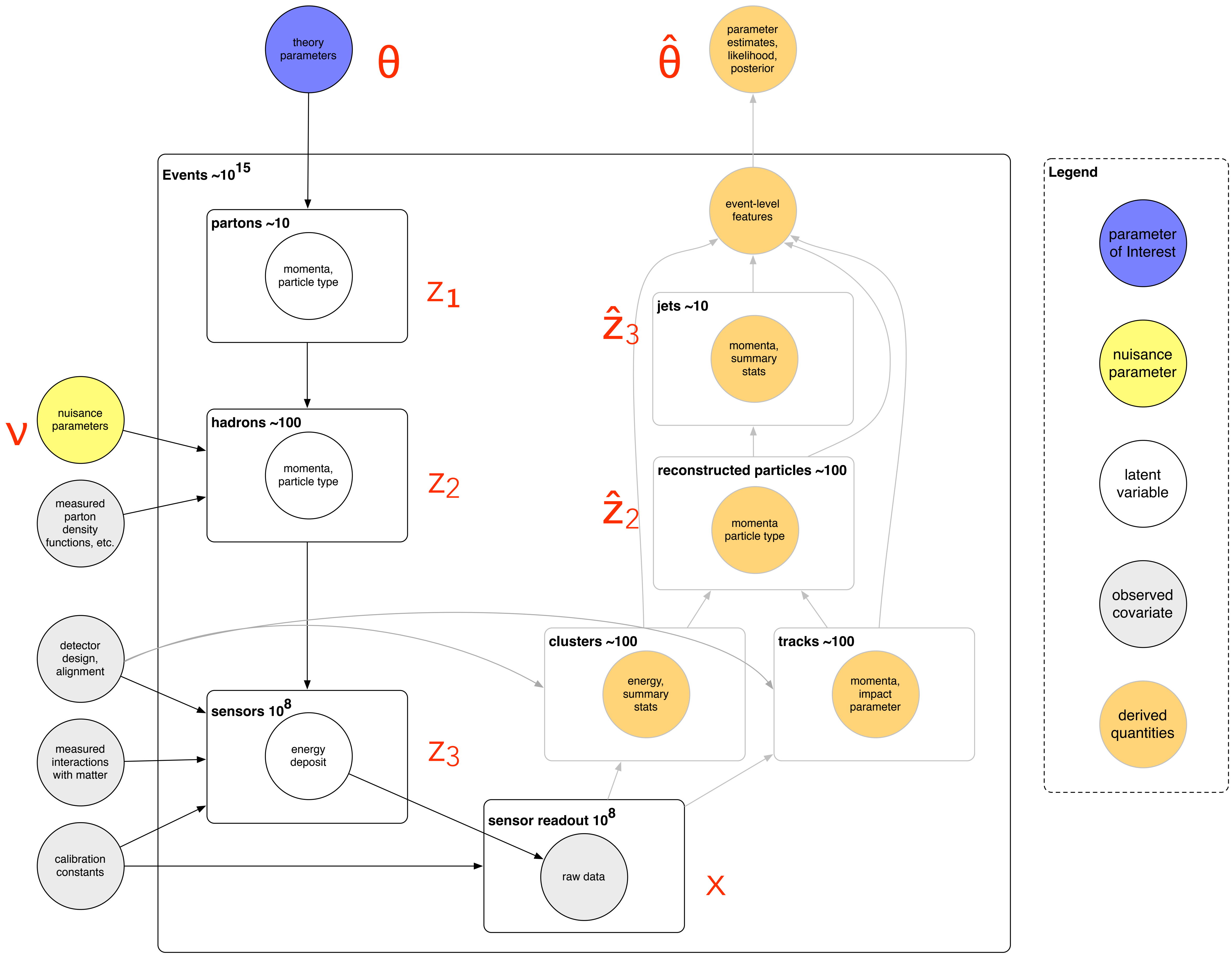
theory parameters
Events ~10^15
partons ~10
momenta, particle type
nuisance parameters
hadrons ~100
momenta, particle type
measured parton density functions, etc.
detector design, alignment
sensors 10^8
energy deposit
measured interactions with matter
calibration constants
sensor readout 10^8
raw data
Legend
parameter of Interest
nuisance parameter
latent variable
observed covariate
derived quantities

SIMULATION

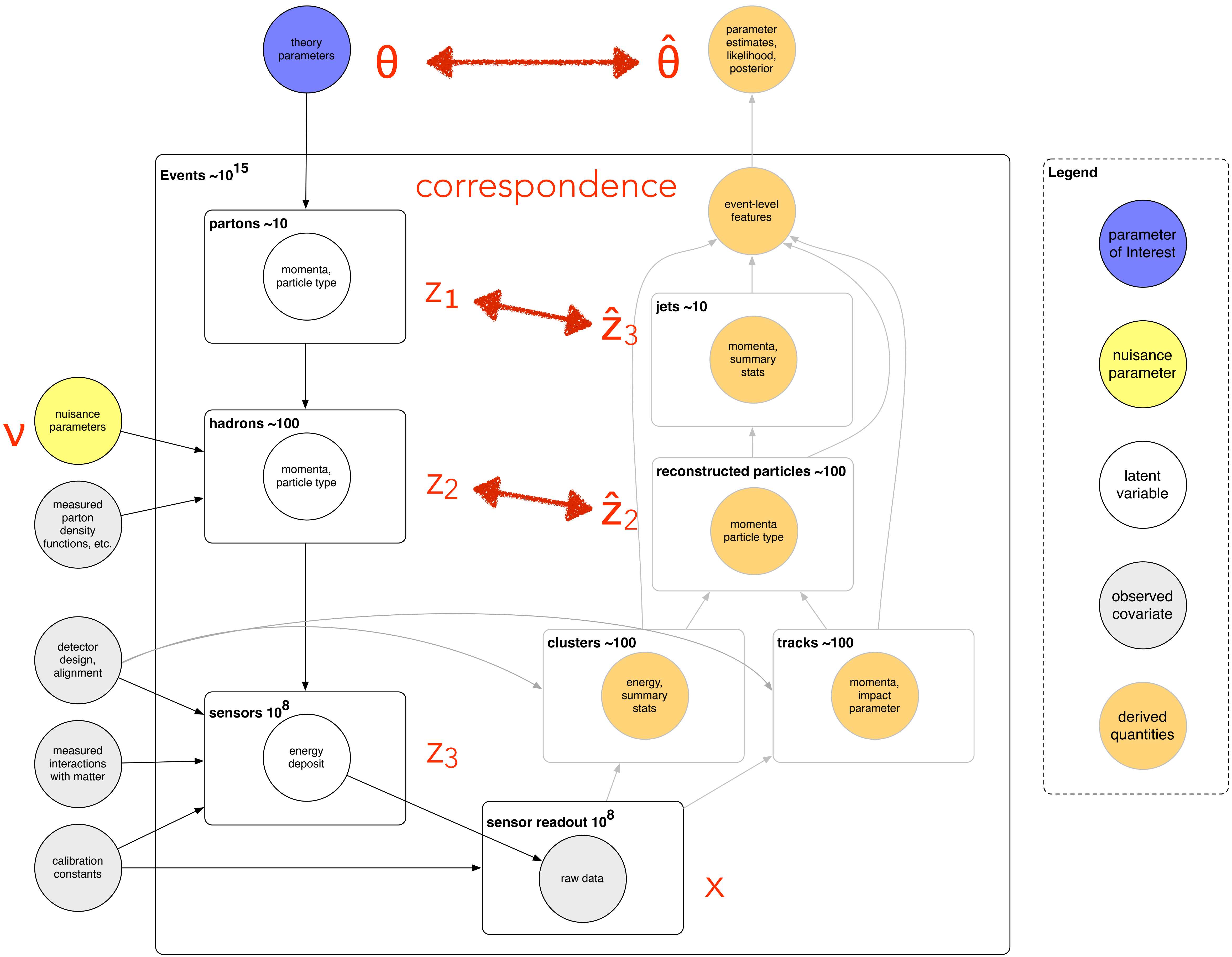


theory parameters
θ
Events ~10^{15}
partons ~10
momenta, particle type
z_1
nuisance parameters
ν
measured parton density functions, etc.
hadrons ~100
momenta, particle type
z_2
detector design, alignment
measured interactions with matter
calibration constants
sensors 10^8
energy deposit
z_3
sensor readout 10^8
raw data
x
Legend
parameter of Interest
nuisance parameter
latent variable
observed covariate
derived quantities

SIMULATION + RECONSTRUCTION



SIMULATION + RECONSTRUCTION



COMPOSITION & REDUCTIONISM

The traditional reconstruction algorithms can be seen as attempt to invert the generative process (point estimate / regression)

- generative model: $\theta \rightarrow z_1 \rightarrow z_2 \rightarrow z_2 \rightarrow x$
- Sequential Inversion: $x \rightarrow \hat{z}_3(x) \rightarrow \hat{z}_2(\hat{z}_3) \rightarrow \hat{z}_1(\hat{z}_2\,)$

Key points:

- can **characterize** & **validate** $p(\hat{z}_1 \mid z_1)$, $p(\hat{z}_2 \mid z_2)$, $p(\hat{z}_3 \mid z_3)$ with simulation
- these components are **reusable** (transfer learning)
  - e.g. an algorithm that looks for electrons in the data (segmentation & classification) and estimates their energy and momentum (regression).
- Provides a notion of "**interpretable**" that is practical and actionable
- **Composition** is at the heart of the **reductionist** paradigm of science

COMPOSITION OF REUSABLE COMPONENTS

How do these fit together?

Combine many of these ideas:

Large model, but **sparsely activated**

Single model to **solve many tasks** (100s to 1Ms)

Dynamically learn and **grow pathways** through large model

Hardware **specialized for ML supercomputing**

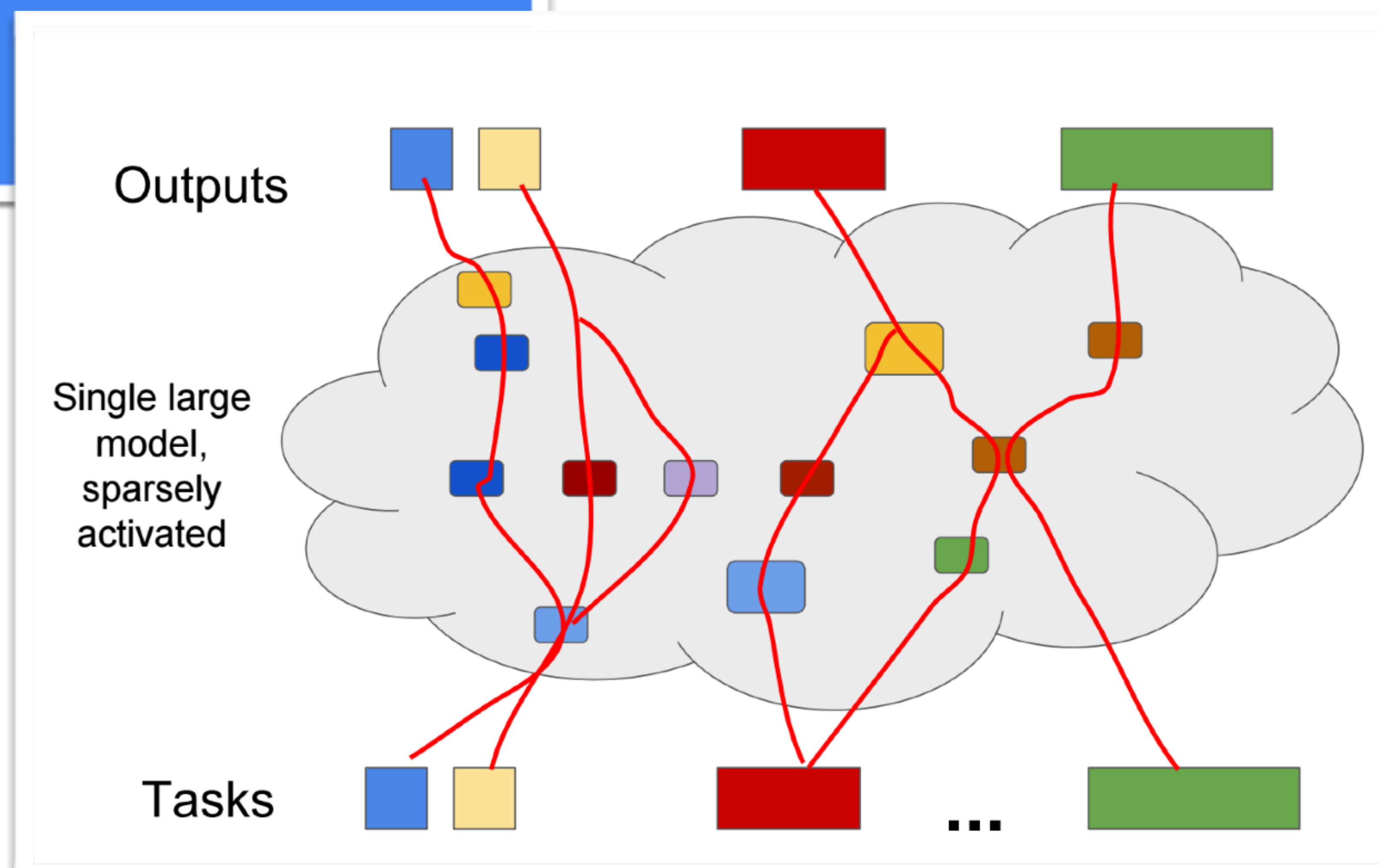
ML for efficient mapping onto this hardware

Google

Kyunghyun Cho

July 10 ·

ML 2.0 at Google



Slides from Jeff Dean of Google Brain @ Jeju July 2017

https://drive.google.com/file/d/0B8z5oUpB2DysZWF1RTFuX1NEZUk/view

DIFFERENTIABLE REDUCTIONISM

The reconstruction algorithms can be seen as attempt to invert the generative process (point estimate / regression) sequentially

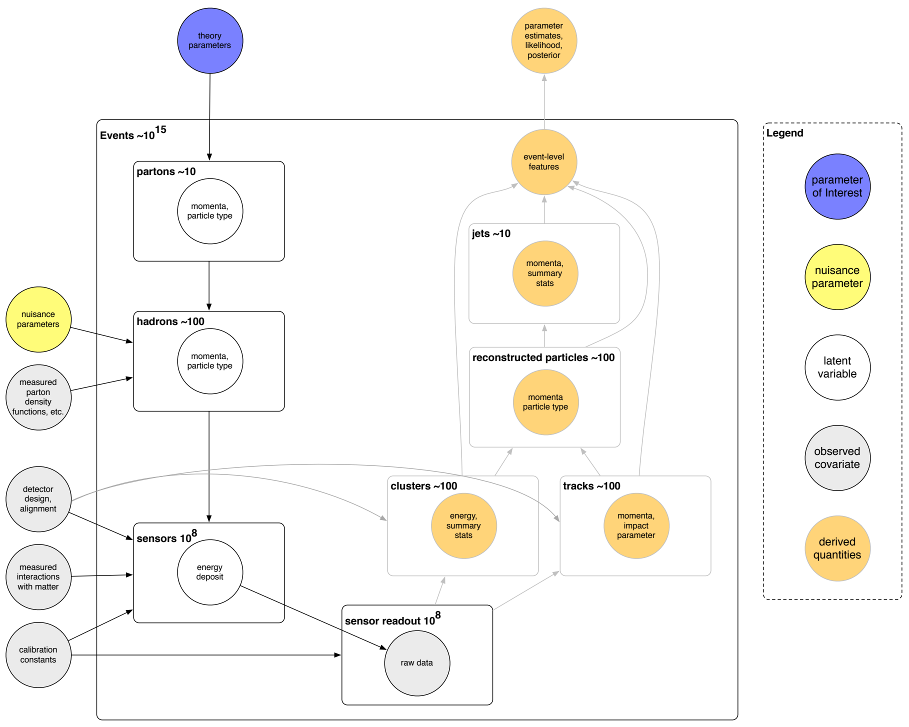
- generative model: $\theta \rightarrow z_1 \rightarrow z_2 \rightarrow z_2 \rightarrow x$
- Sequential Inversion: $x \rightarrow \hat{z}_3(x) \rightarrow \hat{z}_2(\hat{z}_3) \rightarrow \hat{z}_1(\hat{z}_2)$

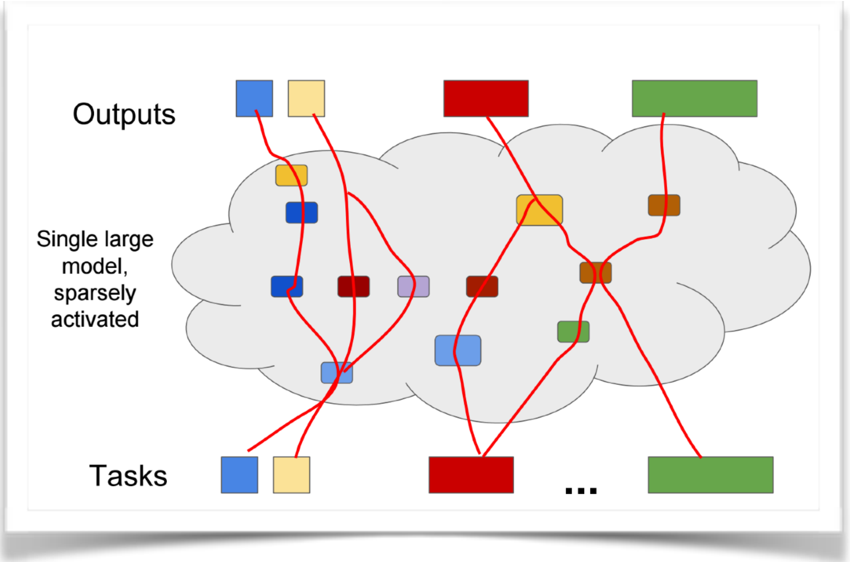
Currently both generative model and inversion algorithms involve hand-engineered, code not developed for auto-diff / back propagation (effectively not differentiable)

- big gain from just reimplementing what we have in a **Differentiable Programming framework**

We can keep the compositional structure and gradually enhance each of the stages of the with deep learning components

- A high-level form of inductive bias (innate structure) on the networks
- jointly optimize & borrow power from all the tasks that use a certain component
  - maintain ability to characterize, validate , and interpret individual components
- transition from deterministic point estimate to probabilistic components for improved uncertainty estimation





CONCLUSION

The developments in machine learning have the potential to effectively bridge the microscopic - macroscopic divide & aid in the inverse problem.

- leverage expert knowledge of the generative process
- learn surrogates that extract relevant features for inference task

Several strategies to incorporate domain knowledge into the model

- starting point: migrate current code bases to differentiable programming framework
- gradually replace components with deep learning

Helpful to establish more actionable notions of “interpretability”

COLLABORATORS


Gilles Louppe
U. Liège


Kyunghyun Cho


Joan Bruna


Brenden Lake


Meghan Frate


Juan Pavez


Tilman Plehn


Johann Brehmer


Isaac Henrion


Lukas Heinrich


Heiko Müller


Tim Head


Michael Kagan


David Rousseau


Peter Sadowski


Daniel Whiteson


Pierre Baldi


Lezcano Casado


Atılım Güneş Baydin
University of Oxford

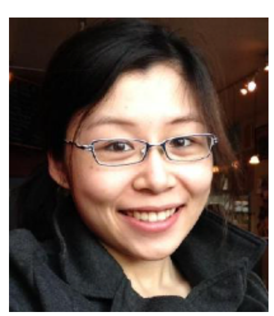

Prabhat
NERSC, Berkeley Lab


Wahid Bhimji
NERSC, Berkeley Lab

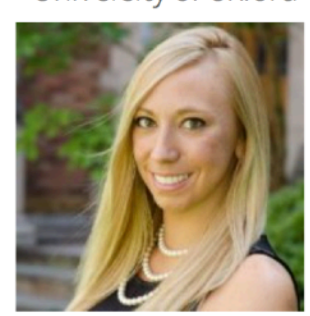

Frank Wood
University of Oxford

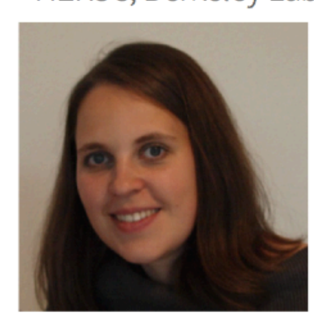

Phiala Shanahan

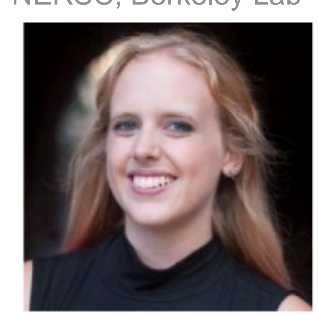

William Detmold

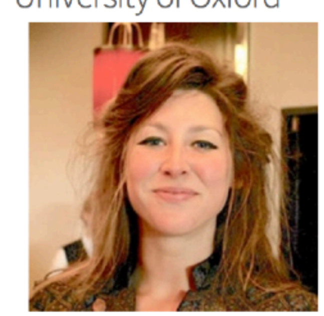

Karen Ng


Tuan Anh Le


Michela Paganini
Yale University


Daniela Huppenkothen
New York University


Savannah Thais
Yale University

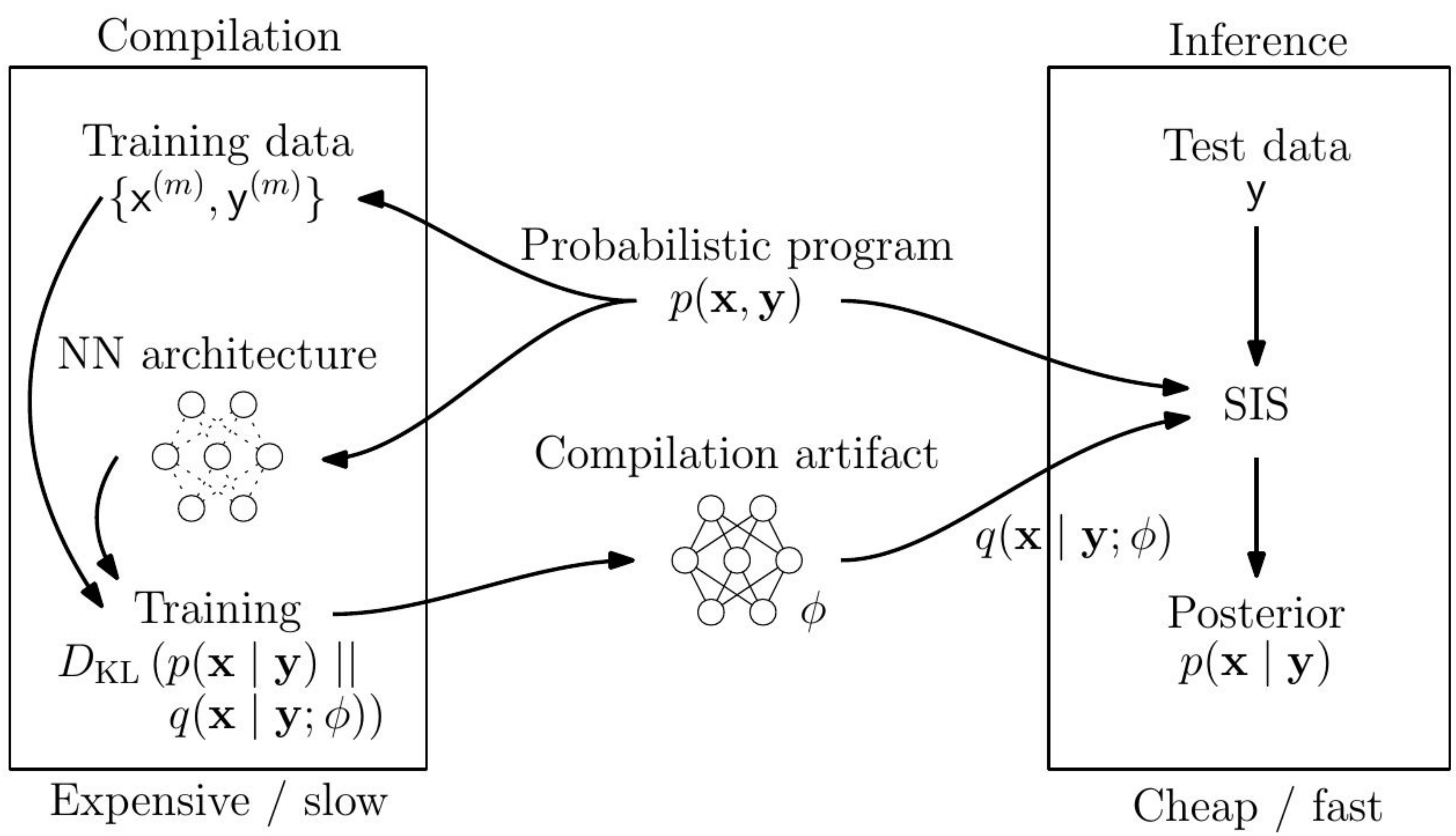

Ruth Angus
Columbia University

Backup

In short: hijack the random number generators and use NN's to perform a *very* smart type of importance sampling

Input: an inference problem denoted in a universal PPL (Anglican, CPProb)

Output: a trained inference network, or "compilation artifact" (Torch, PyTorch)



Le, Baydin and Wood. Inference Compilation and Universal Probabilistic Programming. AISTATS 2017. *arXiv:1610.09900*

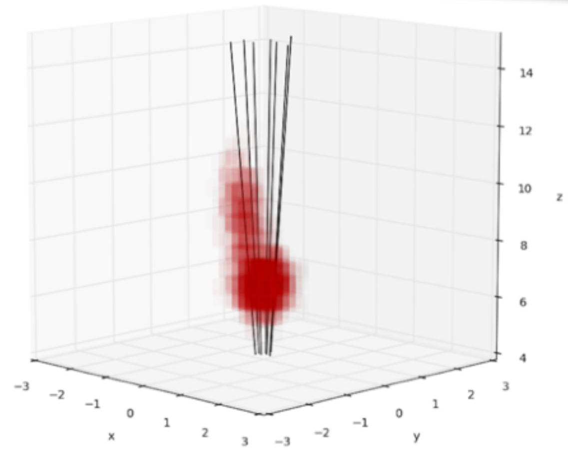
IN PROGRESS: C++, SHERPA, GEANT4

Mario Lezcano Casado, Atılım Güneş Baydin, Tuan Anh Le, Frank Wood*
Department of Engineering Science
University of Oxford
{lezcano,gunes,tuananh,fwood}@robots.ox.ac.uk

Lukas Heinrich, Gilles Louppe, Kyle Cranmer
Department of Physics & Center for Data Science
New York University
{kyle.cranmer,lukas.heinrich,g.louppe}@cern.ch

Wahid Bhimji, Prabhat
Lawrence Berkeley National Laboratory
{wbhimji,prabhat}@lbl.gov

Karen Ng
Intel
karen.y.ng@intel.com



Probabilistic programming with C++

Our new tool: CPProb
https://github.com/probprog/cpprob

Instrumenting C++ code to allow tools like SHERPA and GEANT run with inference compilation

```
void linear_regression(const std::array<std::pair<RealType, RealType>, N> & points) {
    using boost::random::normal_distribution;

    auto normal = normal_distribution<RealType>{0, 10};
    const auto a = cpprob::sample(normal, true);
    const auto b = cpprob::sample(normal, true);

    for (const auto & point : points) {
        auto likelihood = normal_distribution<RealType>{a * point.first + b, 1};
        cpprob::observe(likelihood, point.second);
    }
    cpprob::predict(a);
    cpprob::predict(b);
}
```

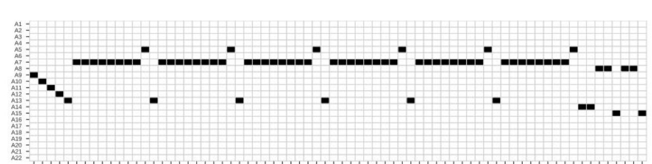
```
SHERPA::Hadron_Decays::Treat(ATOOLS::Blob_List*, double&)+0x709
SHERPA::Event_Handler::IterateEventPhases(SHERPA::eventtype::code&, double&)+0x1b2
SHERPA::Event_Handler::GenerateHadronDecayEvent(SHERPA::eventtype::code&)+0x979
```

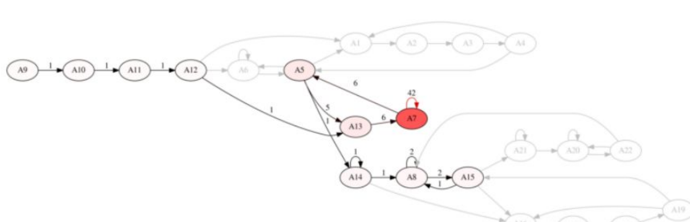
A case study in SHERPA & GEANT

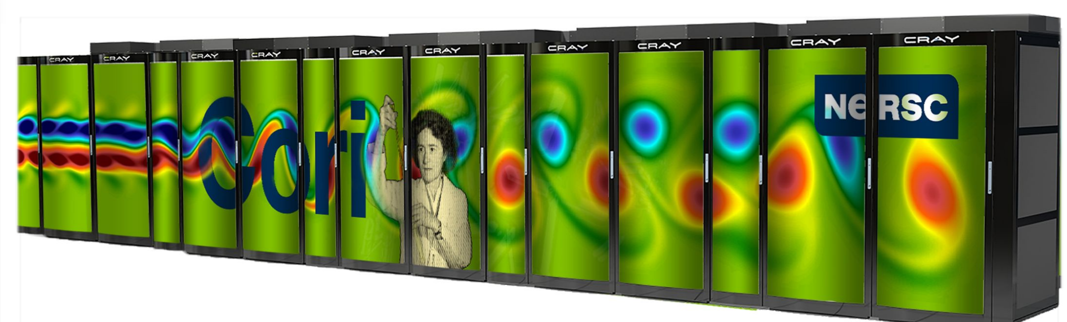
Probabilistic program analytics
allows us to **pinpoint "interesting" addresses** in execution traces
and corresponding **C++ code within SHERPA**

4.4.24 Unique trace T24

Length 72







NERSC, Lawrence Berkeley National Lab

Our current tools:

- CPProb
  - A new C++ PPL coupled with large-scale simulations using, e.g., SHERPA and GEANT
- PyTorch inference compilation backend
  - Dynamic computation graphs for NN artifacts

Designed to run on Cori at NERSC using Shifter

docker

shifterimg -v pull docker:gbaydin/pytorch-infcomp:latest
shifterimg -v pull docker:gbaydin/sherpa-infcomp-full:latest

GANs FOR PHYSICS

CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks

Michela Paganini[a,b], **Luke de Oliveira**[a], **and Benjamin Nachman**[a]

[a] *Lawrence Berkeley National Laboratory, 1 Cyclotron Rd, Berkeley, CA, 94720, USA*
[b] *Department of Physics, Yale University, New Haven, CT 06520, USA*

E-mail: michela.paganini@yale.edu, lukedeoliveira@lbl.gov, bnachman@cern.ch

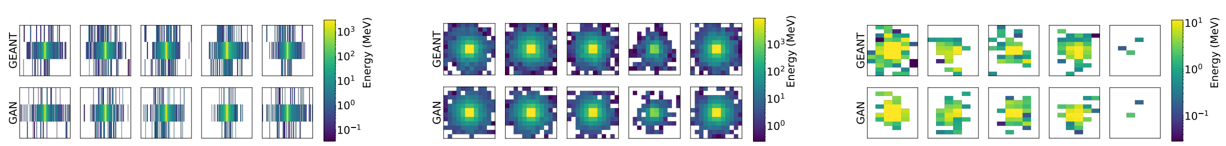


Figure 9: Five randomly selected e^+ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

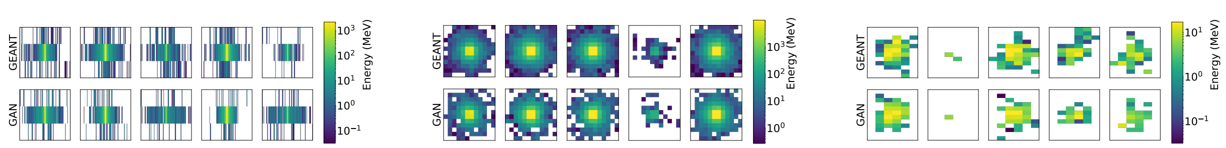


Figure 10: Five randomly selected γ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

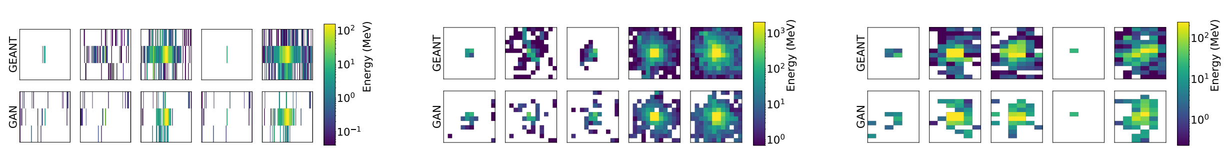
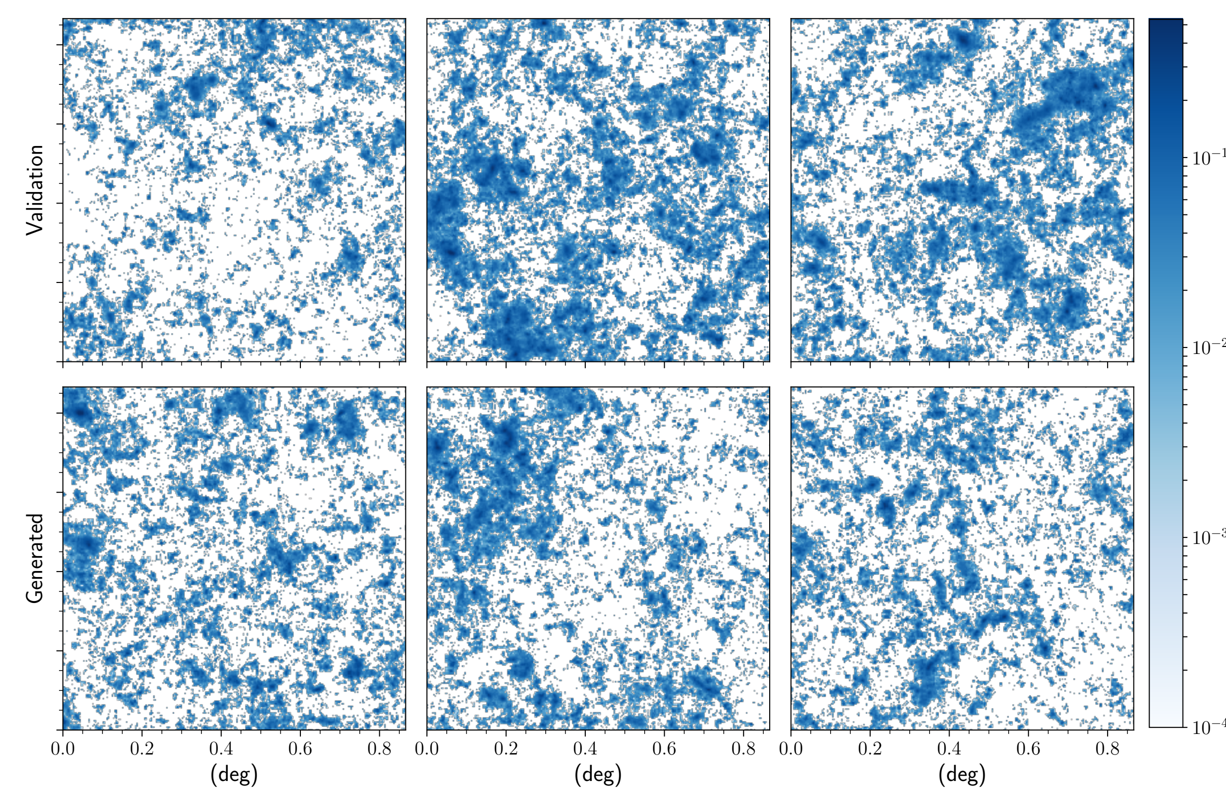


Figure 11: Five randomly selected π^+ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

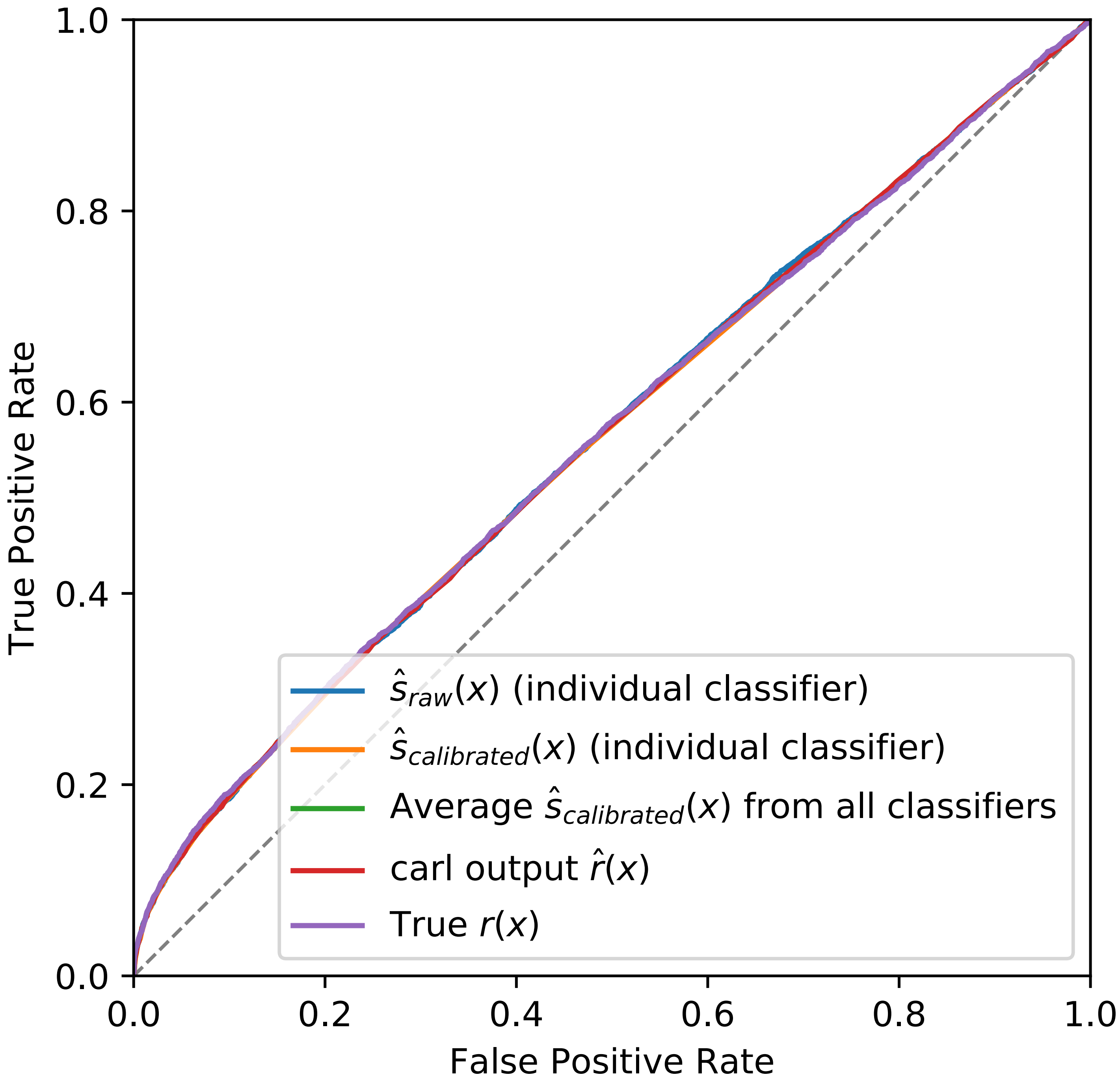
Creating Virtual Universes Using Generative Adversarial Networks

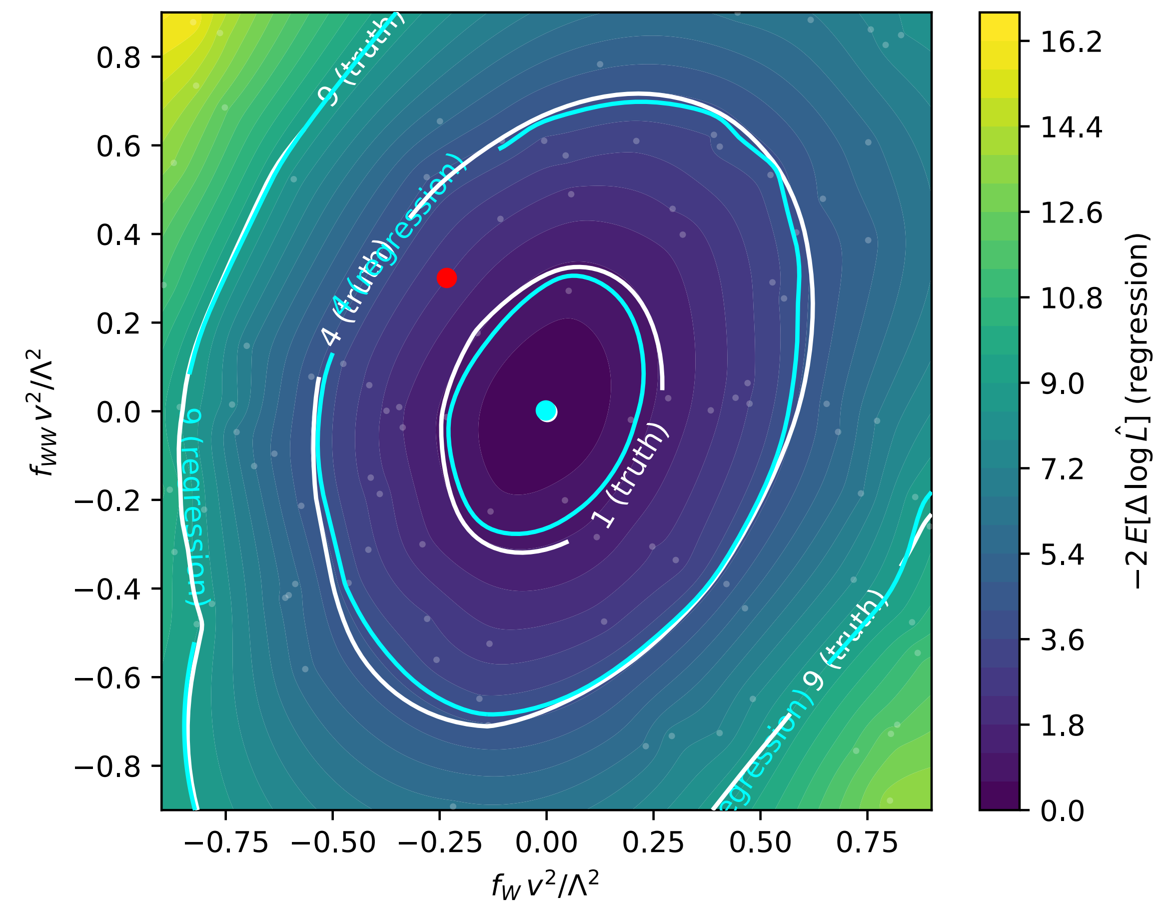
Mustafa Mustafa*[1], Deborah Bard[1], Wahid Bhimji[1], Rami Al-Rfou[2], and Zarija Lukić[1]

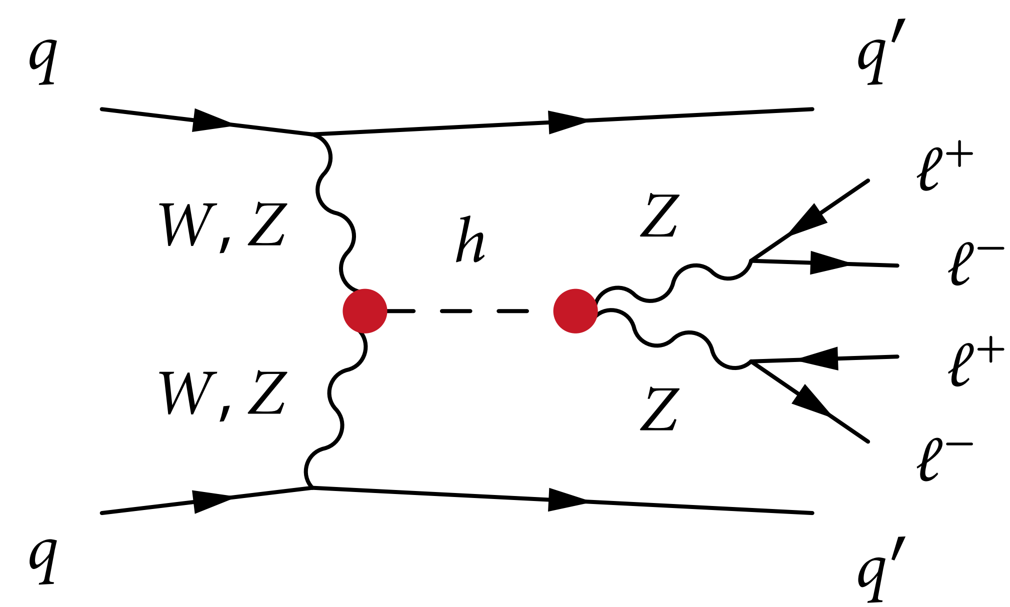
[1]Lawrence Berkeley National Laboratory, Berkeley, CA 94720
[2]Google Research, Mountain View, CA 94043



SEPARABILITY



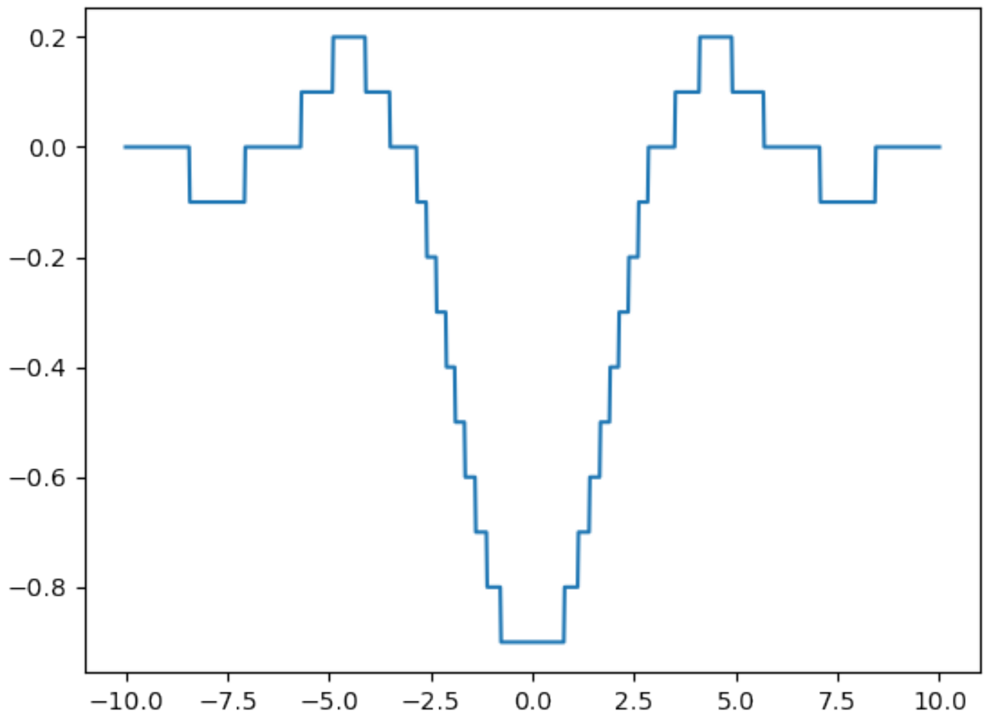




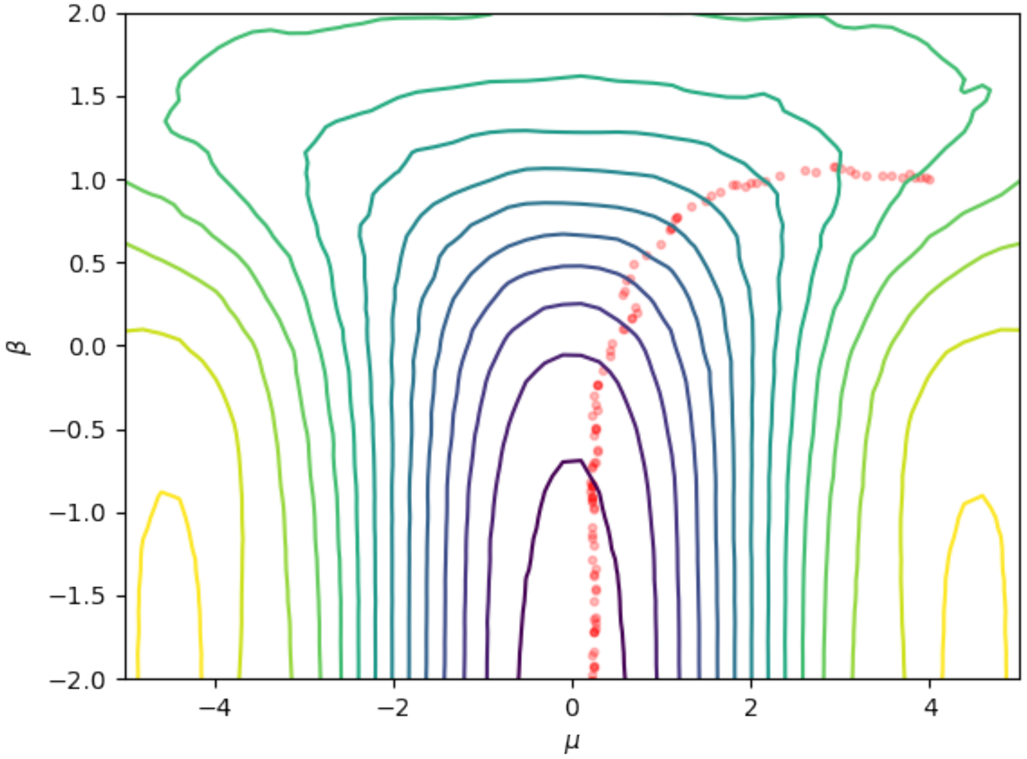
VARIATIONAL OPTIMIZATION

$$\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \leq \mathbb{E}_{\boldsymbol{\theta} \sim q(\boldsymbol{\theta}|\boldsymbol{\psi})}[f(\boldsymbol{\theta})] = U(\boldsymbol{\psi})$$

$$\nabla_{\boldsymbol{\psi}} U(\boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{\theta} \sim q(\boldsymbol{\theta}|\boldsymbol{\psi})}[f(\boldsymbol{\theta}) \nabla_{\boldsymbol{\psi}} \log q(\boldsymbol{\theta}|\boldsymbol{\psi})]$$



Piecewise constant $-\frac{\sin(\mathbf{x})}{\mathbf{x}}$



$q(\boldsymbol{\theta}|\boldsymbol{\psi} = (\mu, \beta)) = \mathcal{N}(\mu, e^{\beta})$

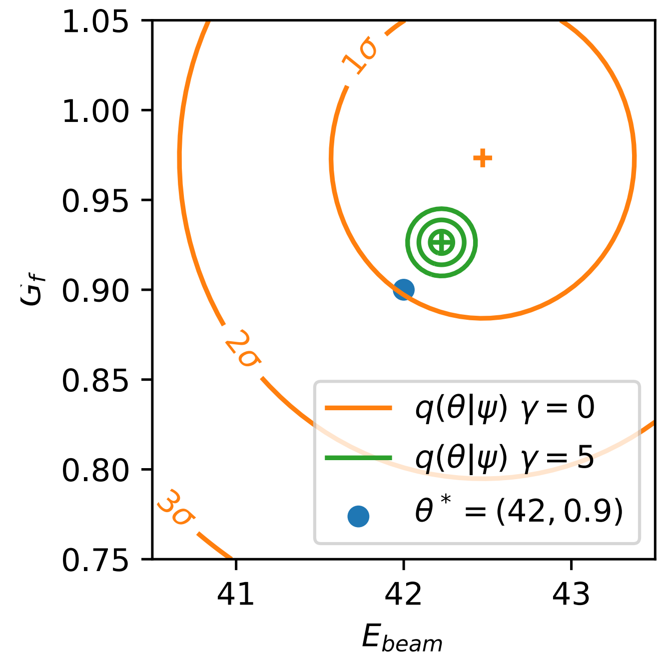
Like a GAN, but generative model is non-differentiable and the parameters of simulator have meaning

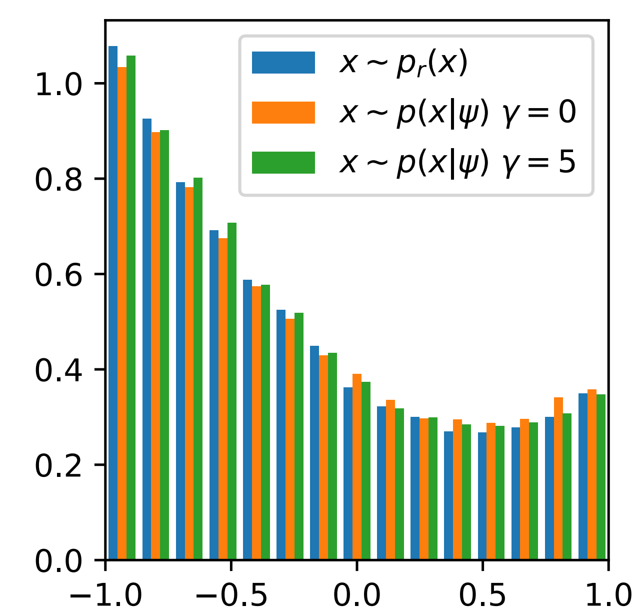
- Replace the generative network with a non-differentiable forward simulator $g(\mathbf{z};\boldsymbol{\theta})$.
- With VO, optimize upper bounds of the adversarial objectives:

$$U_d = \mathbb{E}_{\boldsymbol{\theta}\sim q(\boldsymbol{\theta}|\boldsymbol{\psi})}[\mathcal{L}_d] \quad (1)$$

$$U_g = \mathbb{E}_{\boldsymbol{\theta}\sim q(\boldsymbol{\theta}|\boldsymbol{\psi})}[\mathcal{L}_g] \quad (2)$$

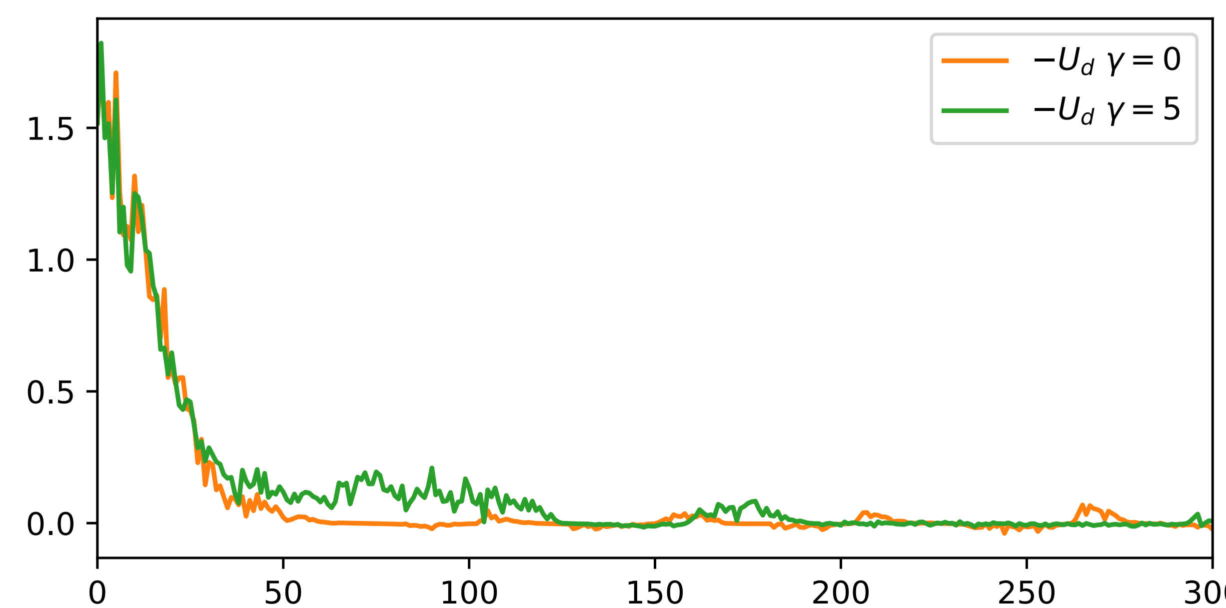
respectively over $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$.





Effectively sampling from marginal model

$$\mathbf{x} \sim q(\mathbf{x}|\boldsymbol{\psi}) \equiv \boldsymbol{\theta} \sim q(\boldsymbol{\theta}|\boldsymbol{\psi}), \mathbf{z} \sim p(\mathbf{z}|\boldsymbol{\theta}), \mathbf{x} = g(\mathbf{z};\boldsymbol{\theta})$$

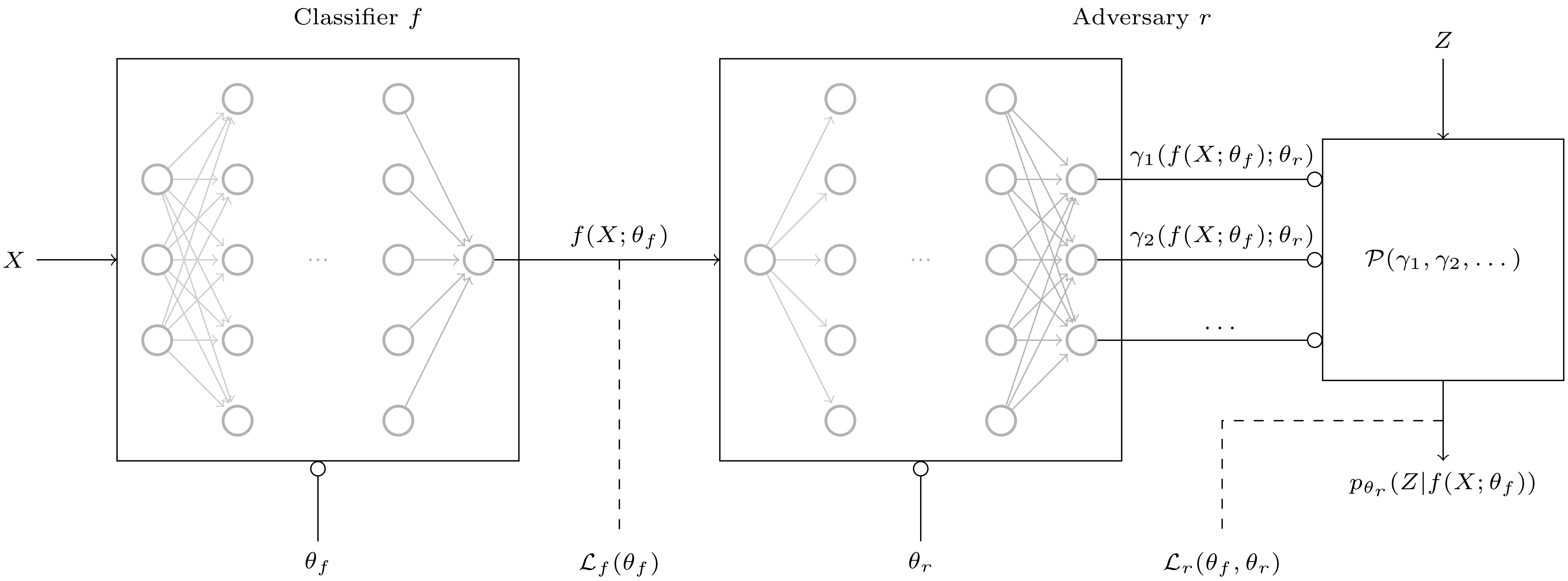


We use Wasserstein distance, as in WGAN

G. Louppe & K.C. arXiv:1707.07113

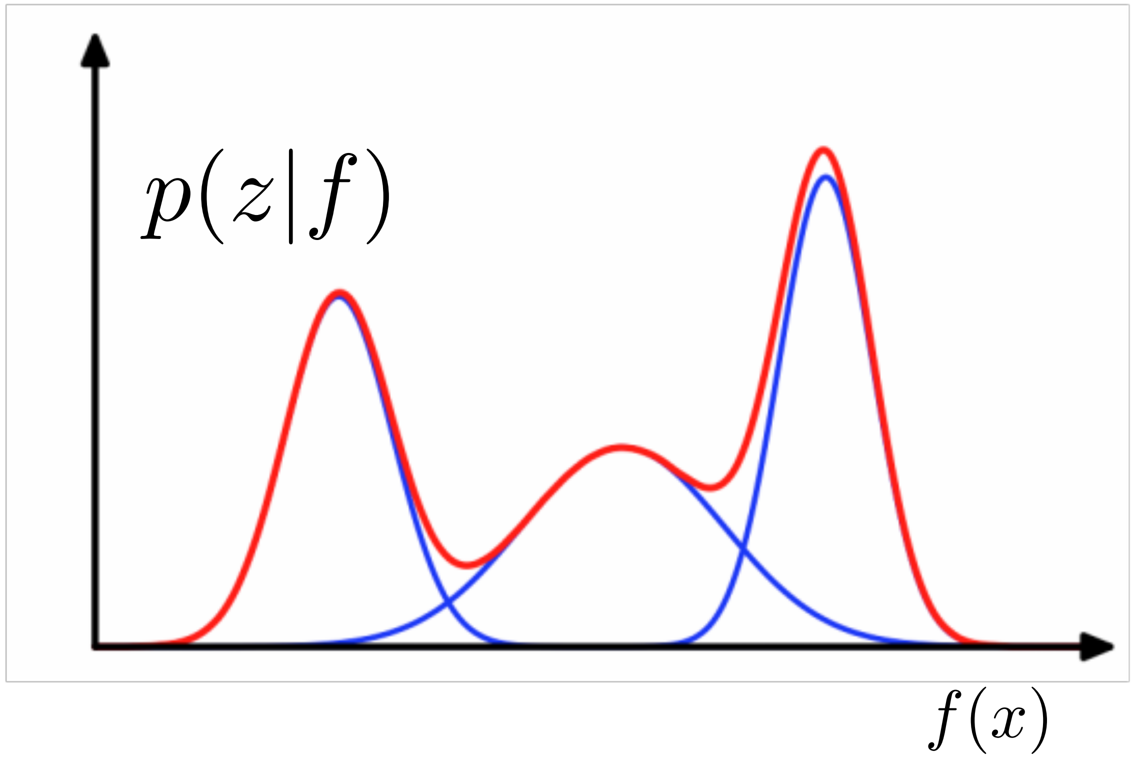
THE ADVERSARIAL MODEL

G. Louppe, M. Kagan, K. Cranmer, arXiv:1611.01046



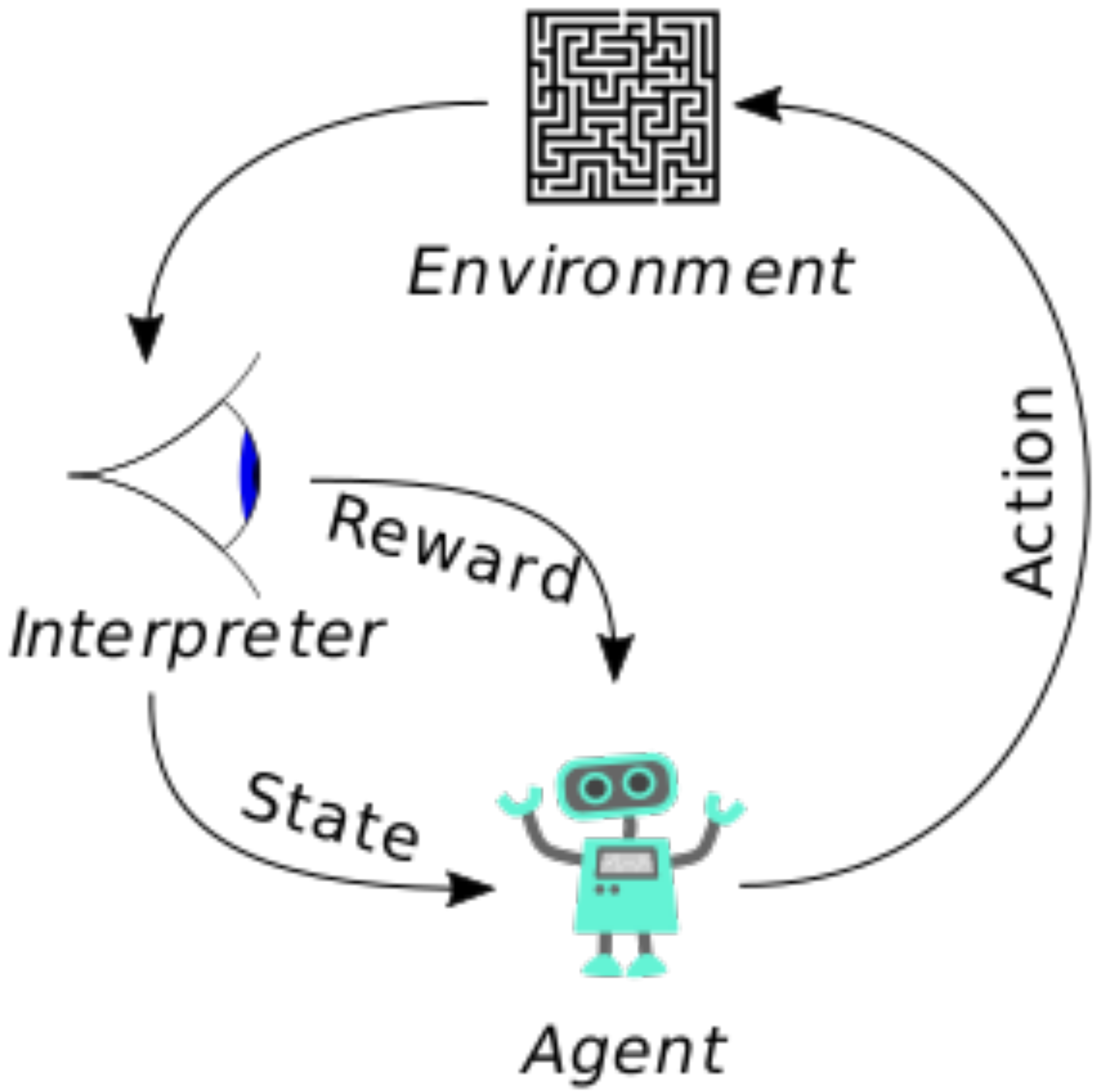
the γ_1, γ_2, … are the mean, standard deviation, and amplitude for the Gaussian Mixture Model.

- the neural network takes in f and predicts γ_1, γ_2, …



Reinforcement / Active Learning
+ Likelihood Free Inference

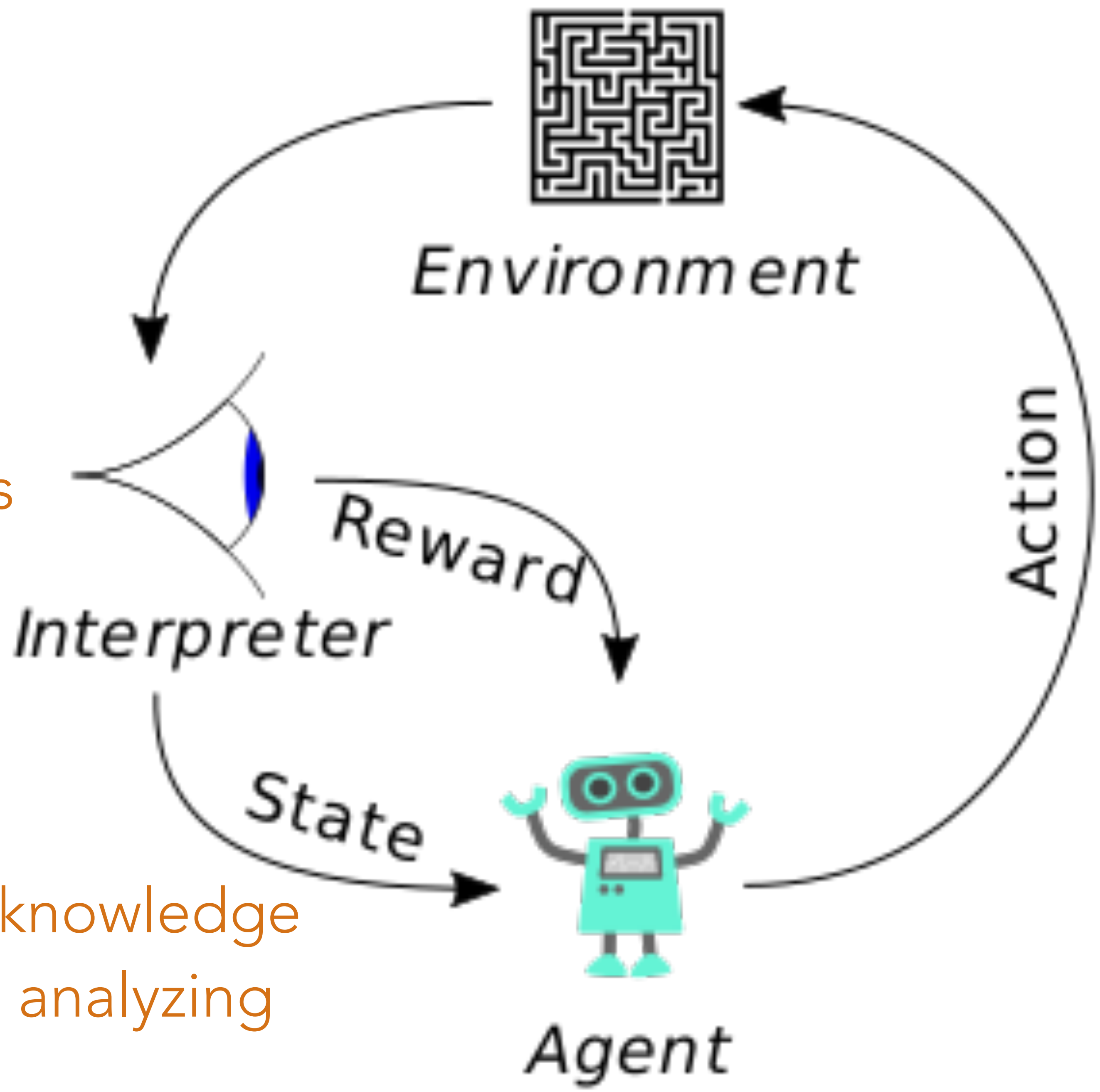
Scientist trying to decide what experiment to do next



Scientist trying to decide what experiment to do next

perform experiment,
gather data

statistical analysis



decide which experiment to perform

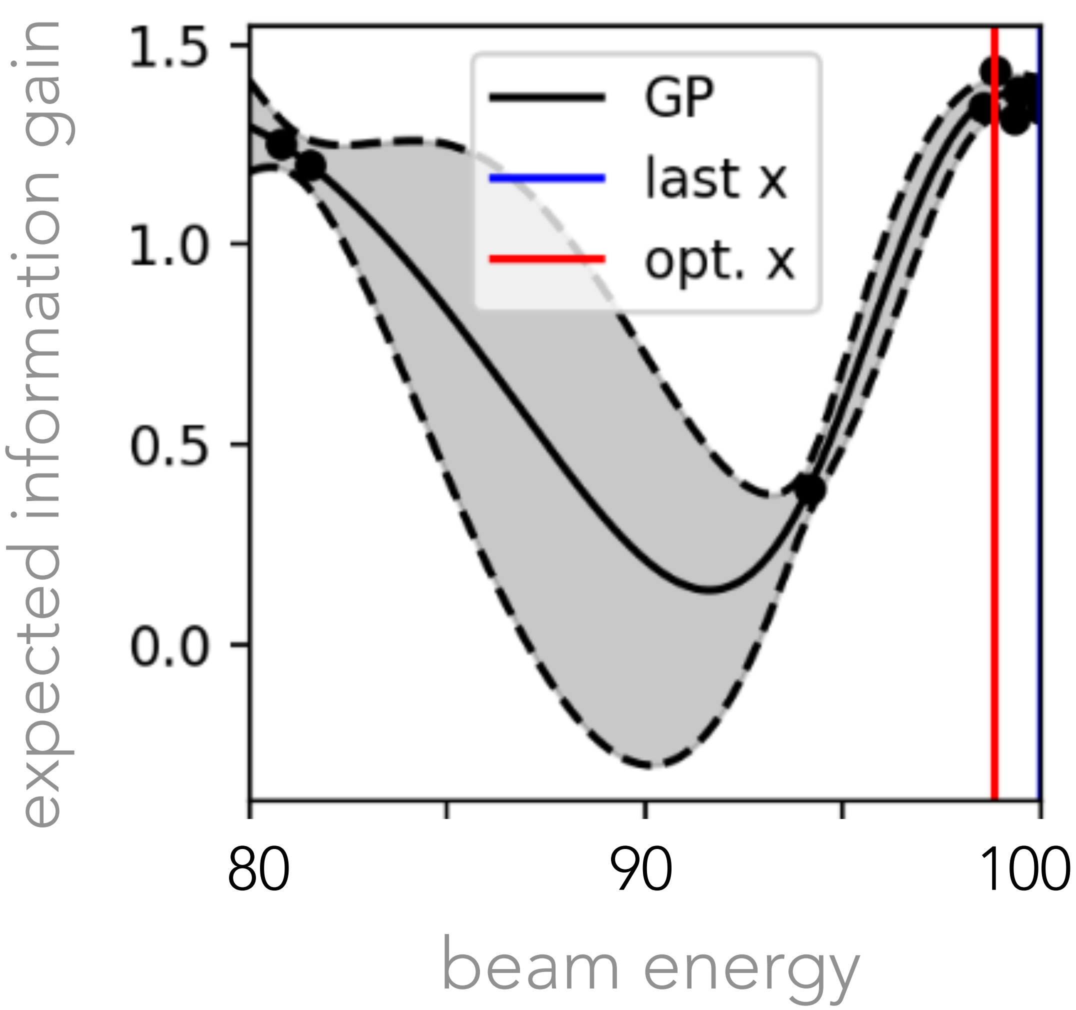
updated knowledge based on analyzing data

OPTIMIZING EXPERIMENTS

https://github.com/cranmer/active_sciencing

Proof-of-principle algorithm can:

- measure parameter of theory (eg. Weinberg angle in Standard Model of particle Physics) from raw data
- optimize experiment (eg. beam energy) for most sensitive measurement



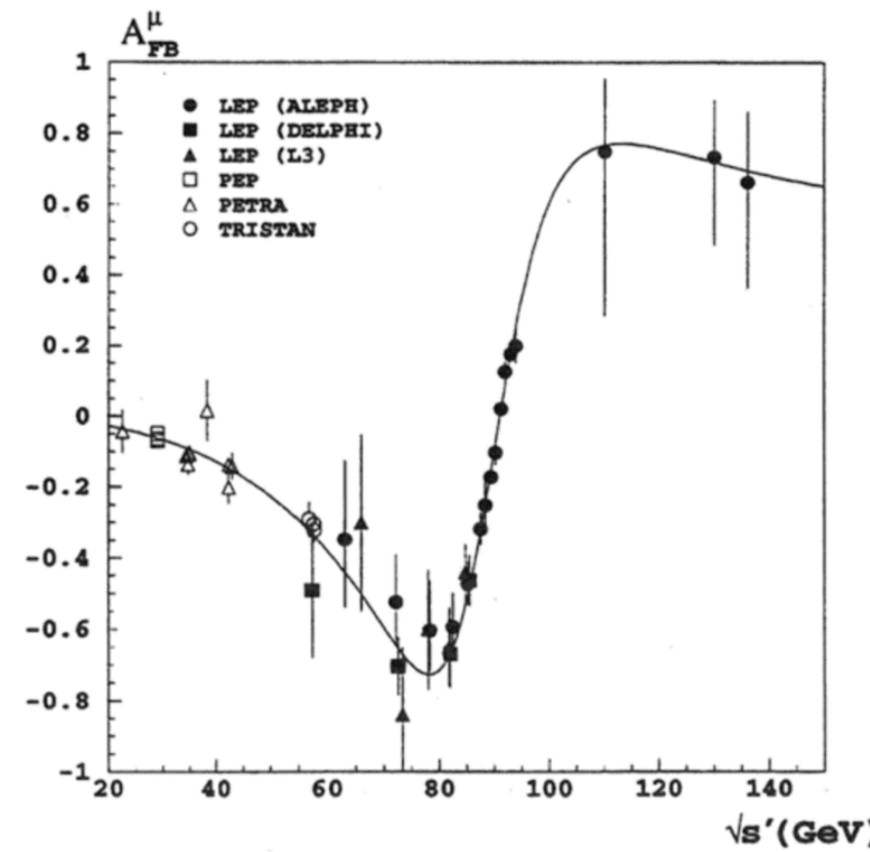
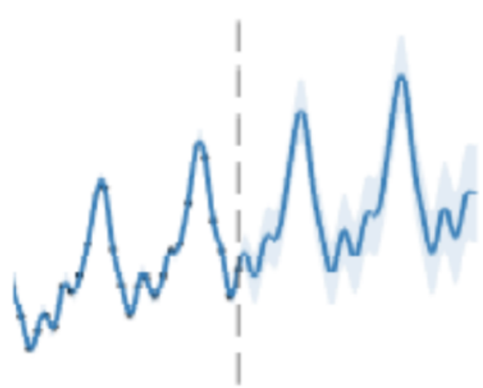


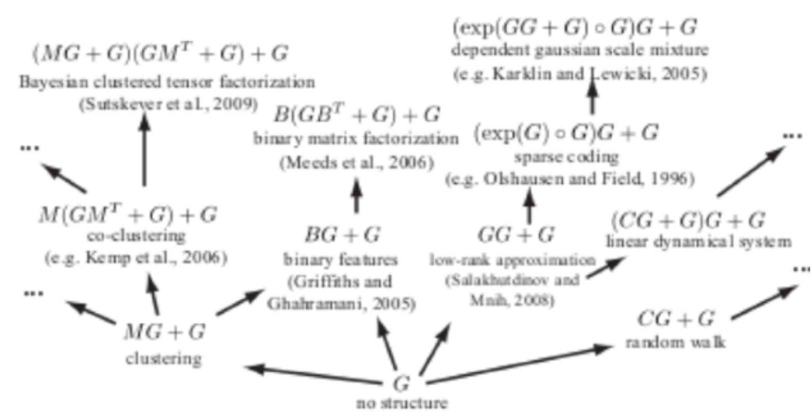
Figure 2: Measured forward-backward asymmetries of muon-pair production compared with the model independent fit results.

Physics goes into the construction of a "Kernel" that defines M.L. model

- Vocabulary of kernels + grammar for composition = powerful modeling

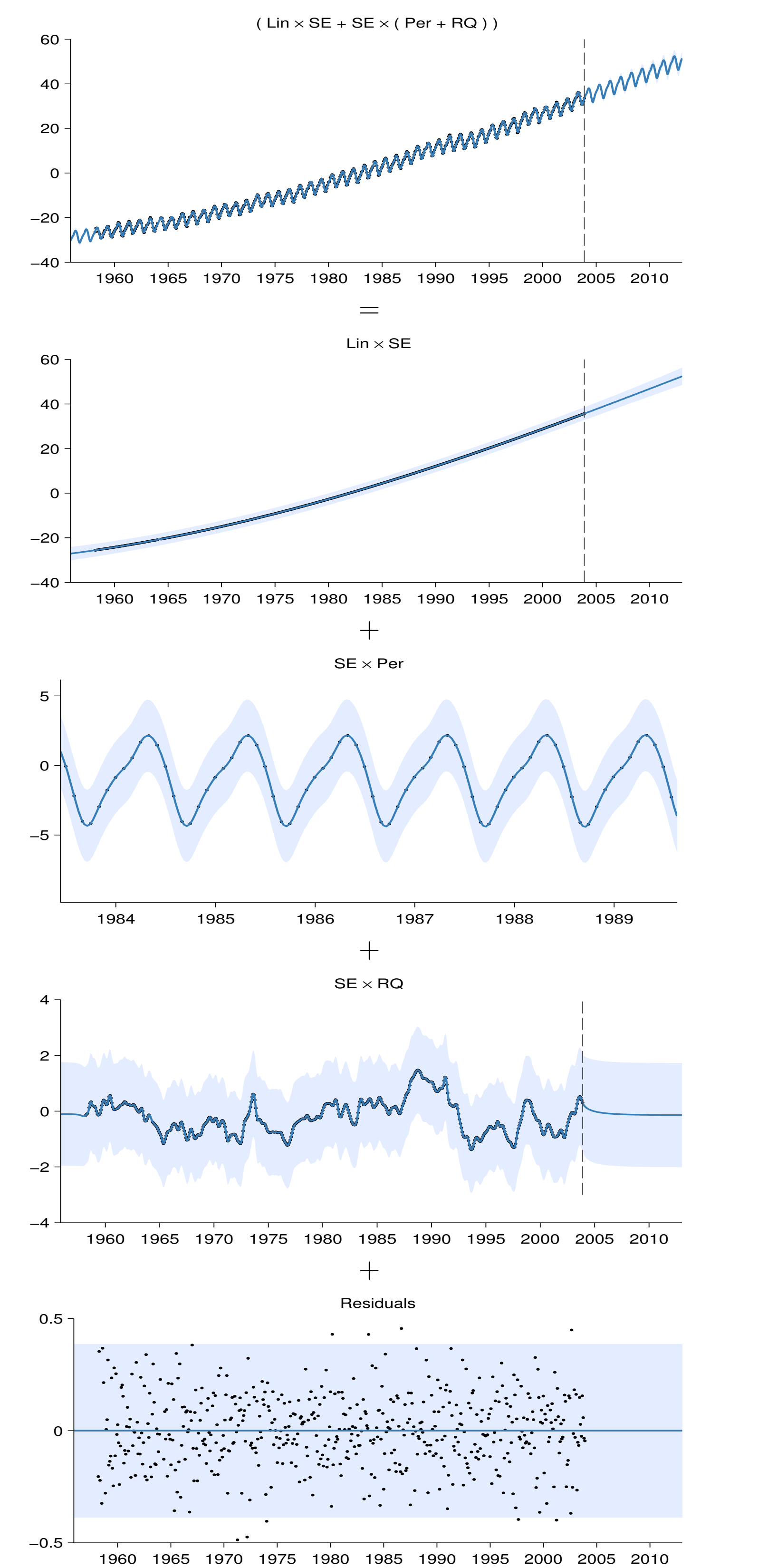


Structure Discovery in Nonparametric Regression through Compositional Kernel Search
David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, Zoubin Ghahramani
International Conference on Machine Learning, 2013
pdf | code | poster | bibtex



Exploiting compositionality to explore a large space of model structures
Roger Grosse, Ruslan Salakhutdinov, William T. Freeman, Joshua B. Tenenbaum
Conference on Uncertainty in Artificial Intelligence, 2012
pdf | code | bibtex

Mauna Loa atmospheric CO_2



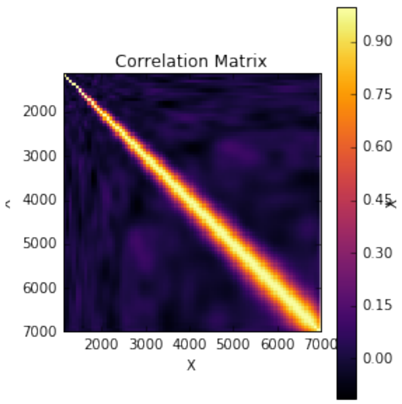
PHYSICS-AWARE MACHINE LEARNING

We can **inject** our knowledge of physics into the machine learning models!

We can **extract** knowledge learned from the data!

Physics-aware Gaussian Processes

arXiv:1709.05681

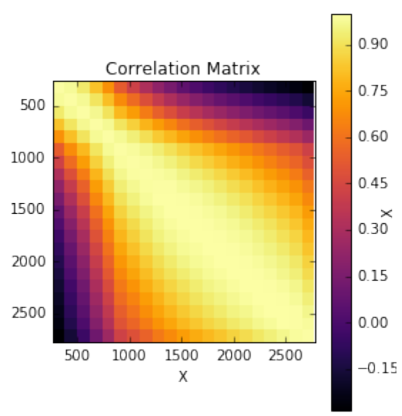


Final Kernel =

Poisson fluctuations

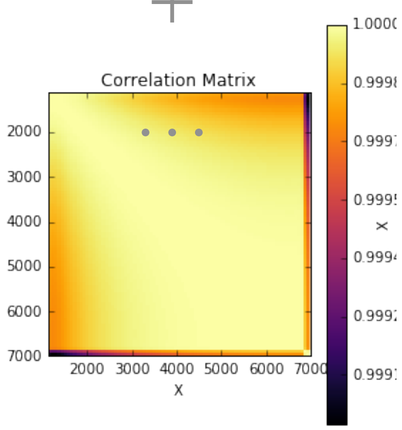
=

+ Mass Resolution



+ Parton Density Functions

+

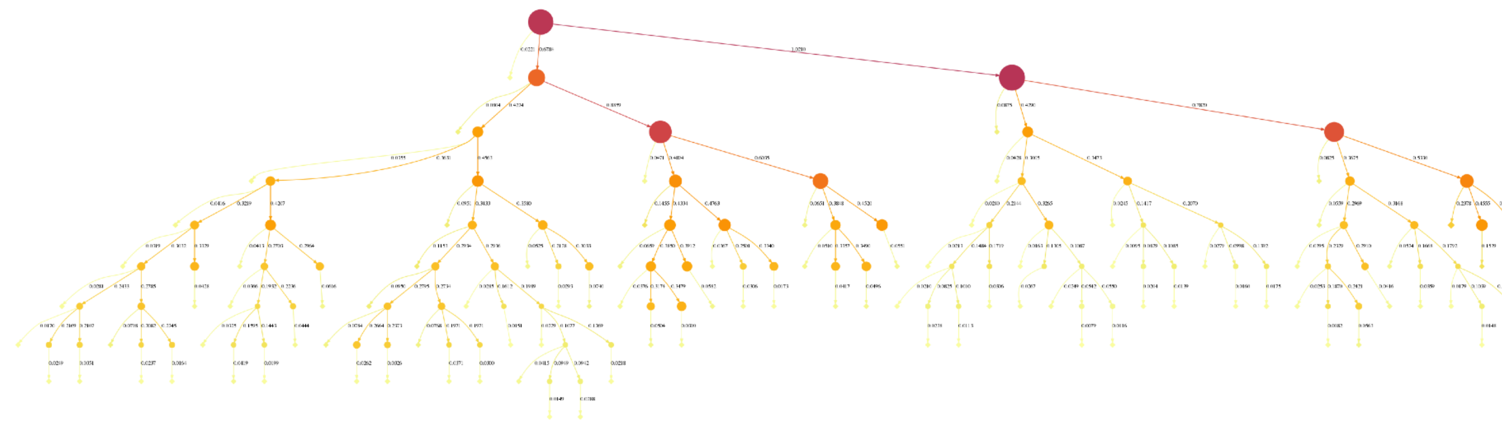


+ Jet Energy Scale

+

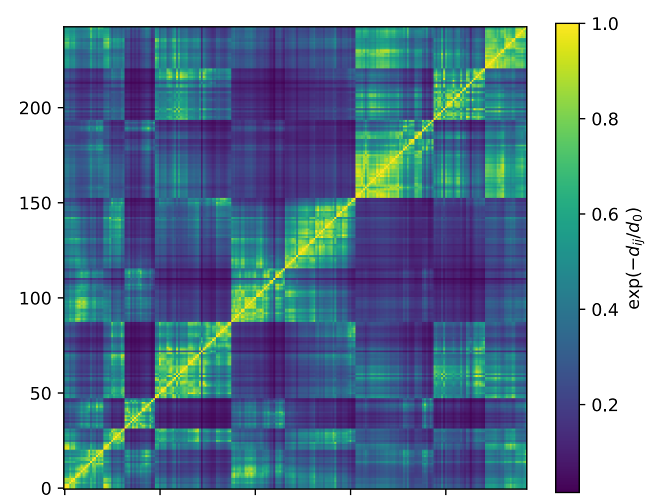
QCD-Aware recursive neural networks

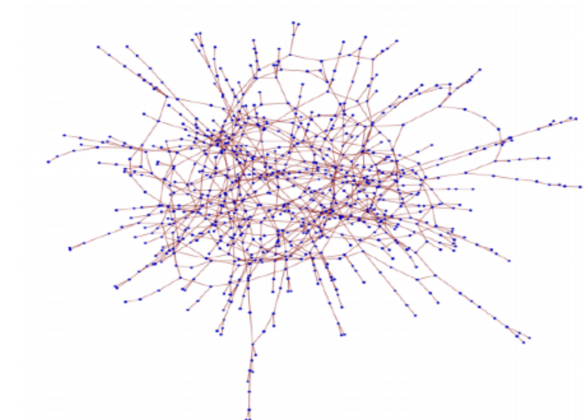
arXiv:1702.00748



QCD-Aware graph convolutional neural networks

NIPS2017 workshop [http://bit.ly/2AkwYRG]

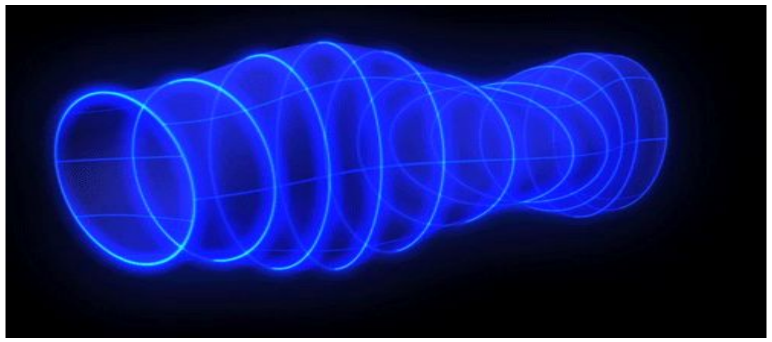


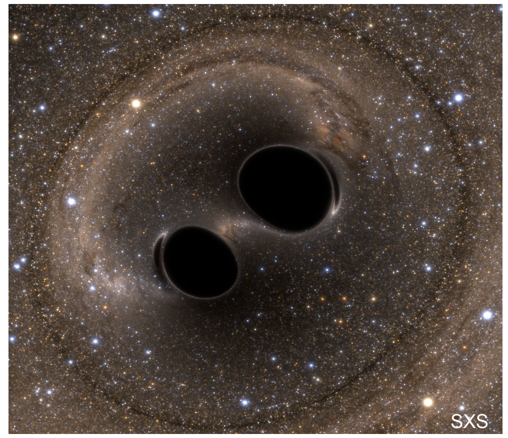


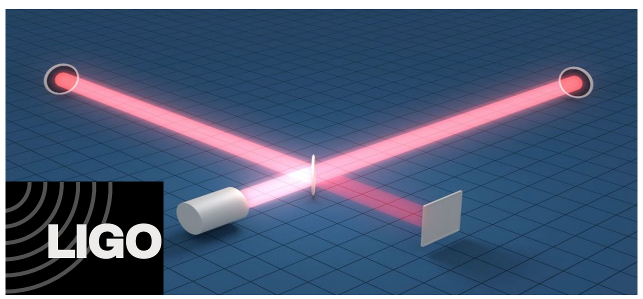
$$d^{\alpha}_{ii'} = \min(p^{2\alpha}_{ti}, p^{2\alpha}_{ti'})\frac{\Delta R^2_{ii'}}{R^2}$$

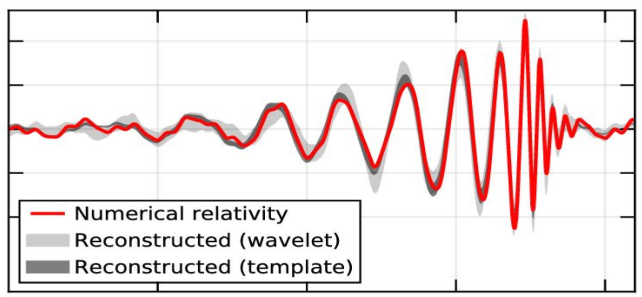
GRAVITATIONAL WAVES & NEUTRINOS

Gravitational Waves









2

Source: ligo.org

Live Demo:
www.tiny.cc/DLGW

Detecting GW150914

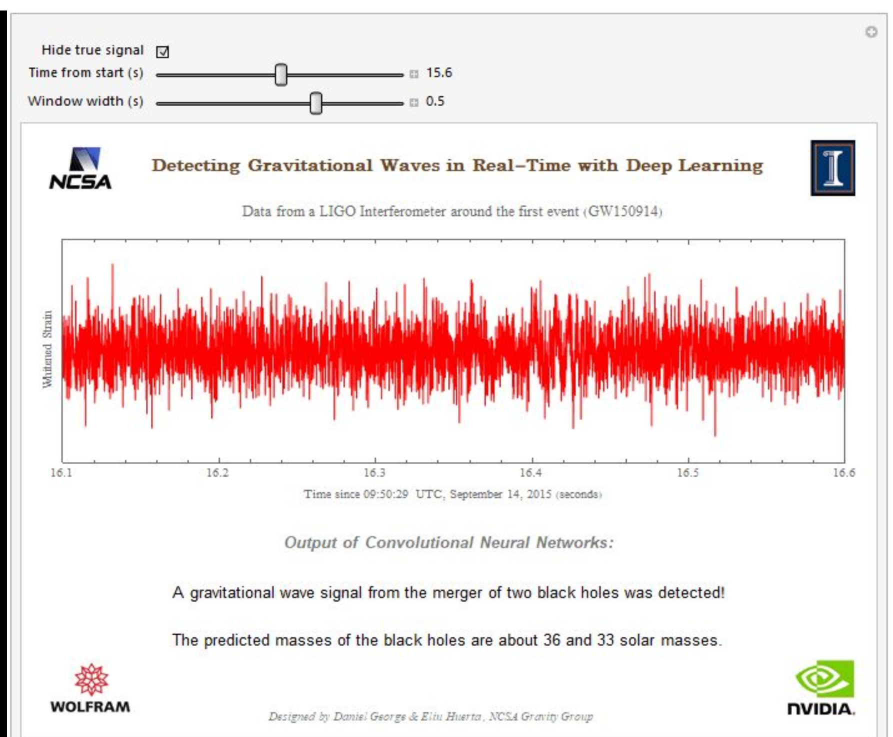
Data not included in training

Trained with only non-spinning, non-eccentric simulations

~1s to analyze 4096s of data.

Masses correct within error bars

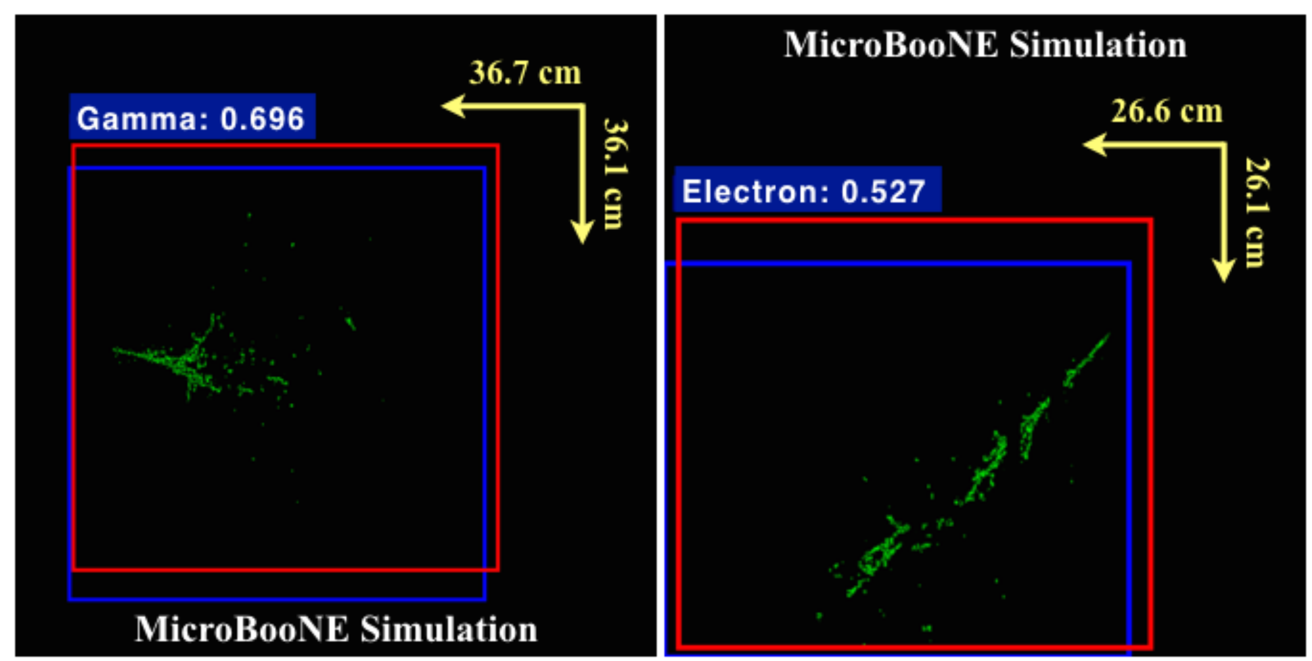
No False Alarms with two detectors!



Convolutional Neural Networks Applied to Neutrino Events in a Liquid Argon Time Projection Chamber

---

MicroBooNE Collaboration



JUAN CARRASQUILLA @ NIPS

Juan Carrasquilla

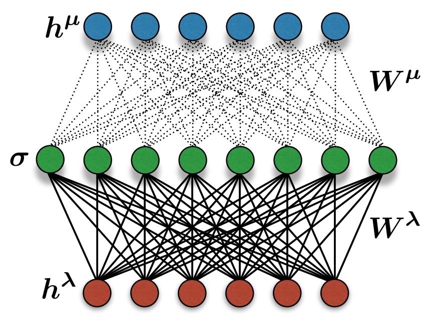
RESTRICTED BOLTZMANN MACHINE WAVE FUNCTION

RBM probability distribution:

$$p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma}) = \mathrm{e}^{\sum_j b_j^{\lambda}\sigma_j + \sum_i \log\left(1+\mathrm{e}^{c_i^{\lambda}+\sum_j W_{ij}^{\lambda}\sigma_j}\right)}$$

RBM wavefunction:

$$\psi_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\boldsymbol{\sigma}) = \sqrt{\frac{p_{\boldsymbol{\lambda}}(\boldsymbol{\sigma})}{Z_{\boldsymbol{\lambda}}}}\,\mathrm{e}^{i\phi_{\boldsymbol{\mu}}(\boldsymbol{\sigma})} \qquad \phi_{\boldsymbol{\mu}} = \log p_{\boldsymbol{\mu}}(\boldsymbol{\sigma})$$



Widespread use of RBMs to solve many-body physics:

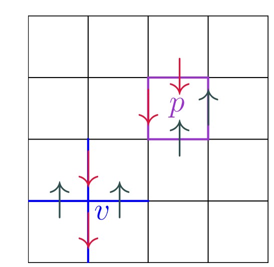
- Variational ansatz for quantum wave-functions (Carleo & Troyer, Science 2017)
- Exact representation of topological states (Deng, Li & Das Sarma, arXiv 2016)
- Accelerate Monte Carlo simulations (Huang & Wang, PRB 2017)
- Topological quantum error correction (GT & Melko, PRL)
- and more . . .

But other choices for the neural network are also possible (CNN, MLP etc)

Torlai, Mazzola, Carrasquilla, Troyer, Melko and Carleo 1703:05334

NEURAL-NETWORK QUANTUM STATE TOMOGRAPHY FOR LARGE MANY-BODY SYSTEMS

KITAEV'S TORIC CODE GROUND STATE

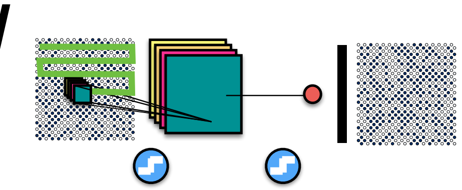


$$H = -J_p \sum_p \prod_{i\in p} \sigma_i^z - J_v \sum_v \prod_{i\in v} \sigma_i^x$$

$$|\Psi_{\mathrm{TC}}\rangle \propto \lim_{\beta\to\infty} \sum_{\sigma_1,\ldots,\sigma_N} e^{\frac{\beta}{2} J \sum_p \prod_{i\in p} \sigma_i^z} |\sigma_1,\ldots,\sigma_N\rangle$$

PEPS : F. Verstraete, M. M. Wolf, D. Perez-Garcia, J. I. Cirac Phys. Rev. Lett. 96, 220601 (2006).

$$O_{\mathrm{cold}}(\sigma_1,\ldots,\sigma_N) \propto \lim_{\beta\to\infty} \exp \beta J \sum_p \prod_{i\in p} \sigma_i^z$$

$| \Psi >= \sum \sqrt{}$  $| \quad >$

J. Carrasquilla and R. G. Melko. Nature Physics 13, 431–434 (2017)
Dong-Ling Deng et al Phys. Rev. X 7, 021021 (2017)
Jing Chen, Song Cheng, Haidong Xie, Lei Wang, Tao Xiang arXiv:1701.04831 RBMs

NEURAL-NETWORK QUANTUM STATE TOMOGRAPHY FOR LARGE MANY-BODY SYSTEMS

GENERATIVE MODELS FOR CALIBRATION

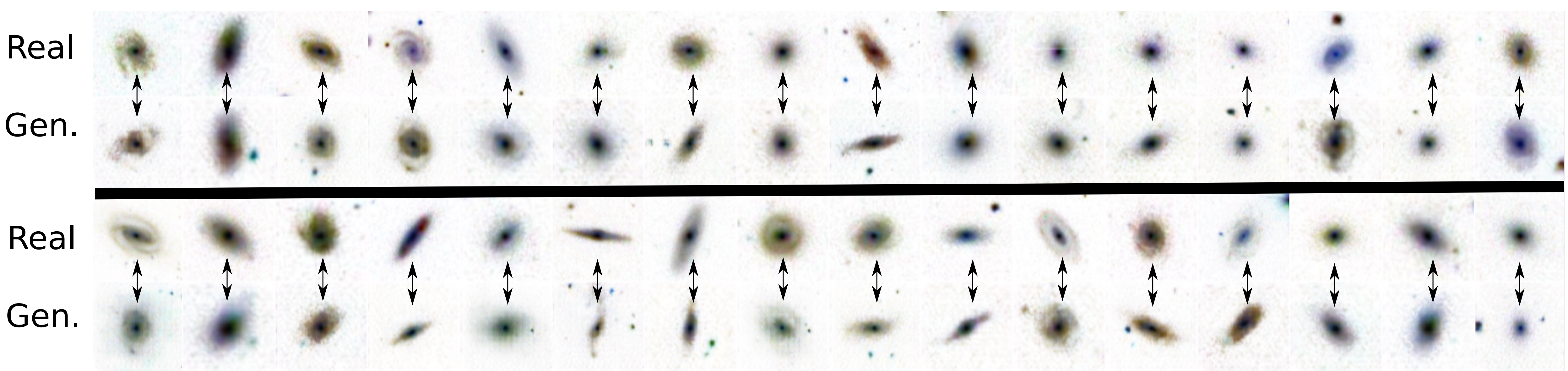
Use of generative models of galaxy images to help calibrate down-stream analysis in next-generation surveys.

Enabling Dark Energy Science with Deep Generative Models of Galaxy Images

Siamak Ravanbakhsh[1], François Lanusse[2], Rachel Mandelbaum[2], Jeff Schneider[1], and Barnabás Póczos[1]

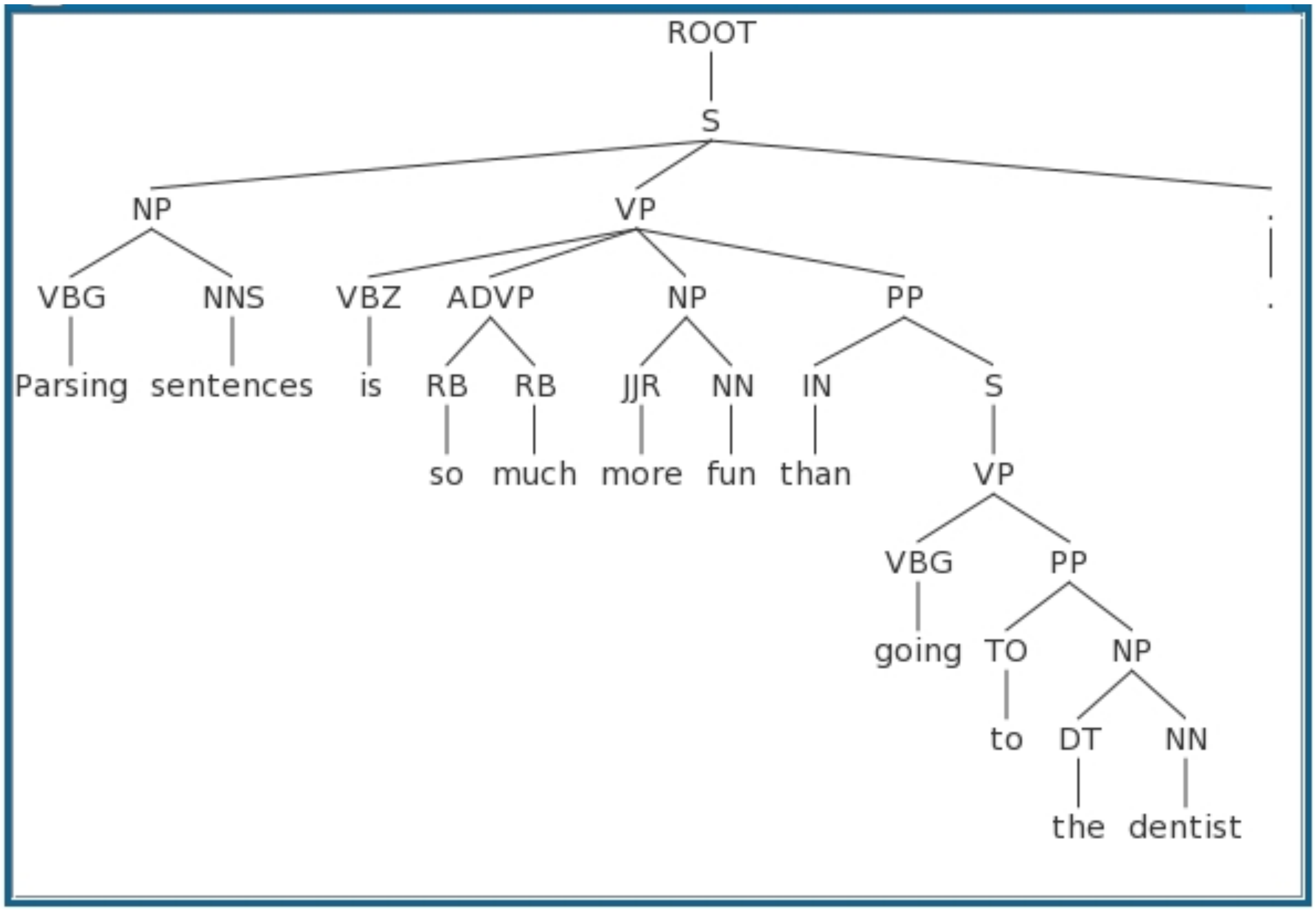
[1]*School of Computer Science, Carnegie Mellon University*
[2]*McWilliams Center for Cosmology, Carnegie Mellon University*

***Abstract*—Understanding the nature of dark energy, the mysterious force driving the accelerated expansion of the Universe, is a major challenge of modern cosmology. The next generation of cosmological surveys, specifically designed to address this issue, rely on accurate measurements of the apparent shapes of distant galaxies. However, shape measurement methods suffer from various unavoidable biases and therefore will rely on a precise calibration to meet the accuracy requirements of the science analysis. This calibration process remains an open challenge as it requires large sets of high quality galaxy images. To this end, we study the application of deep conditional generative models in generating realistic galaxy images. In particular we consider variations on conditional variational autoencoder and introduce a new adversarial objective for training of conditional generative networks. Our results suggest a reliable alternative to the acquisition of expensive high quality observations for generating the calibration data needed by the next generation of cosmological surveys.**



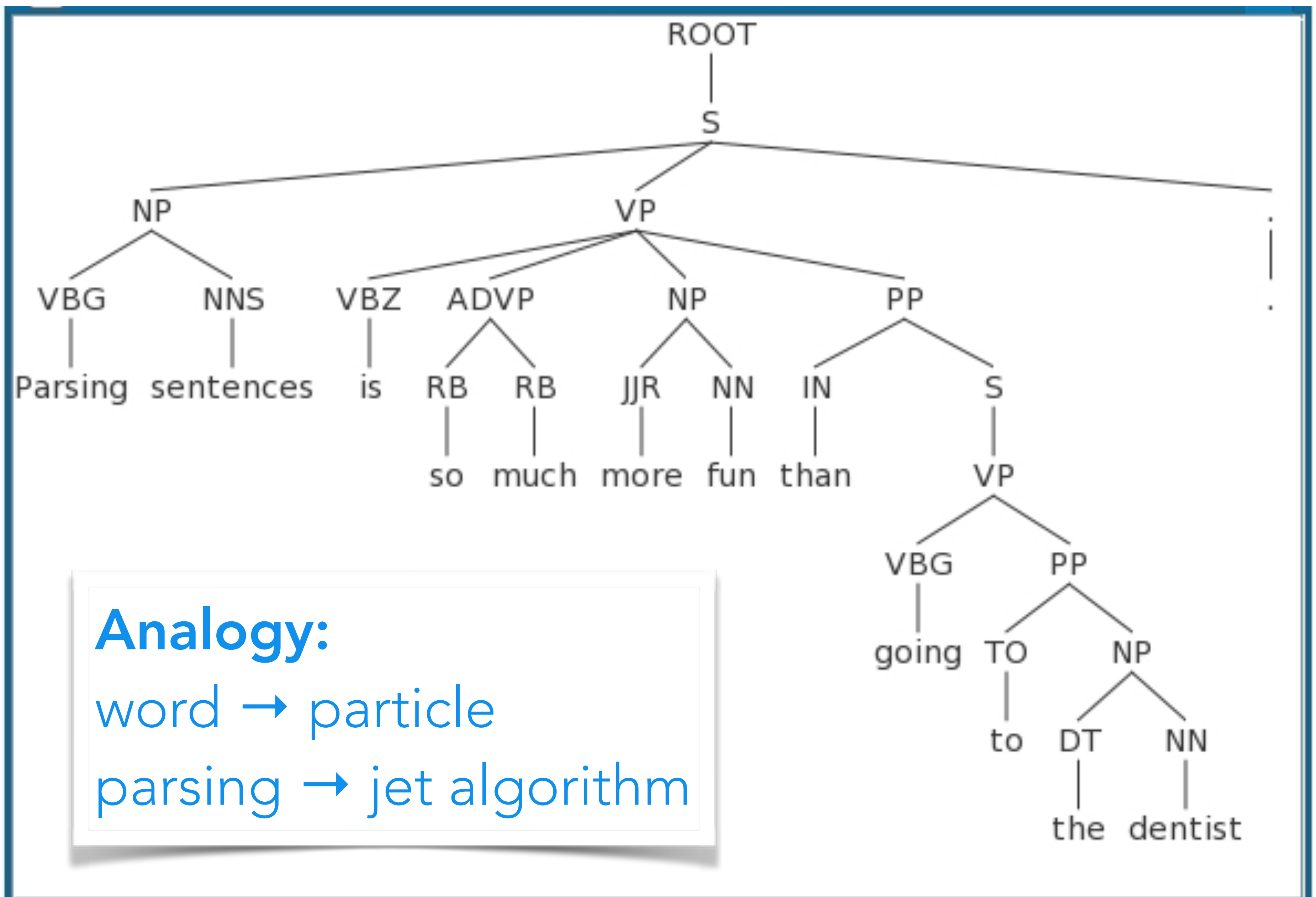
Recursive Neural Networks showing great performance for Natural Language Processing tasks

- neural network's topology given by parsing of sentence!



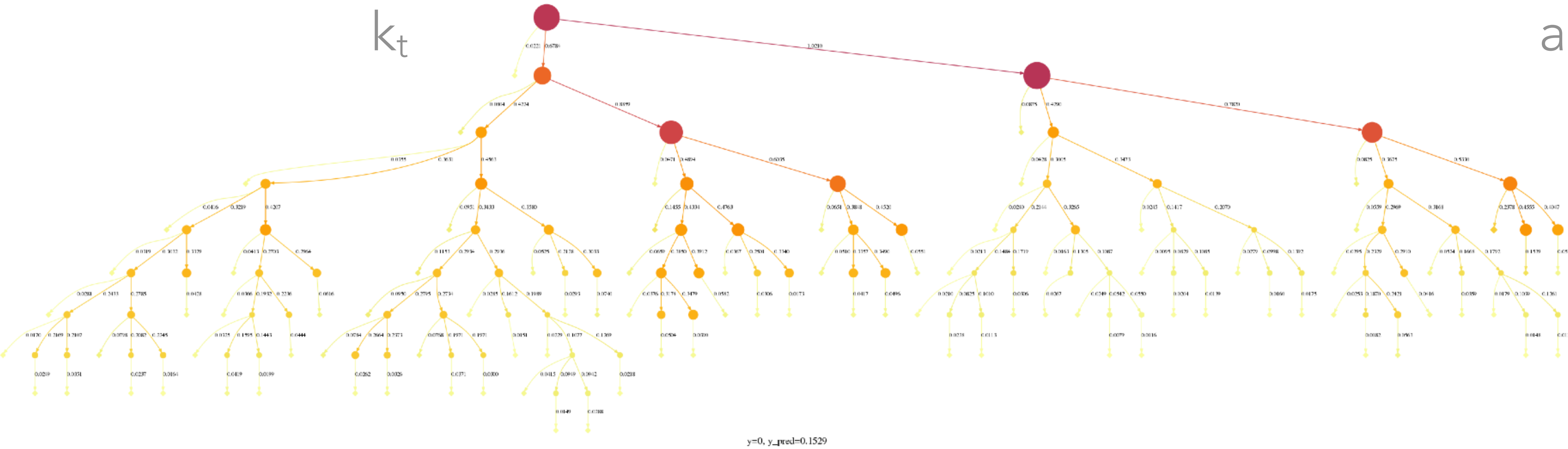
Recursive Neural Networks showing great performance for Natural Language Processing tasks

- neural network's topology given by parsing of sentence!

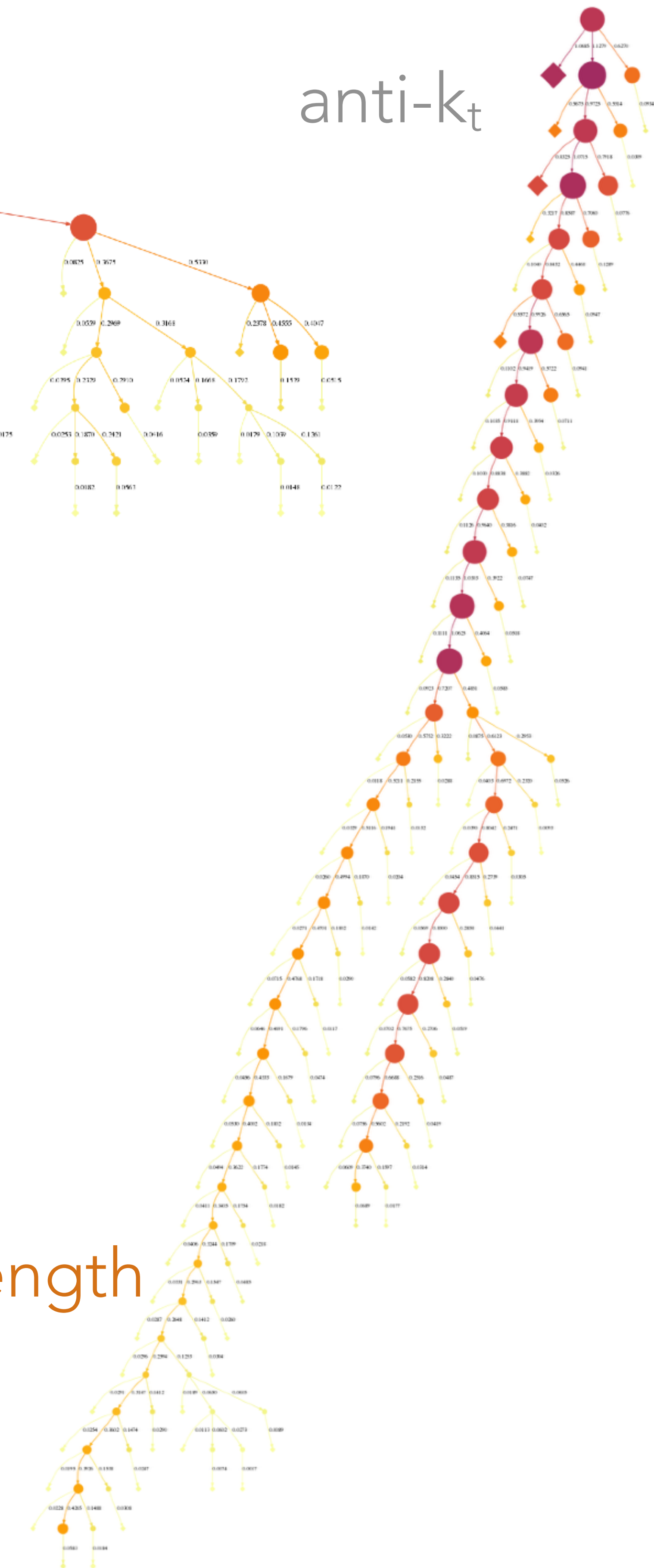


QCD-INSPIRED RECURSIVE NEURAL NETWORKS

k_t



anti-k_t

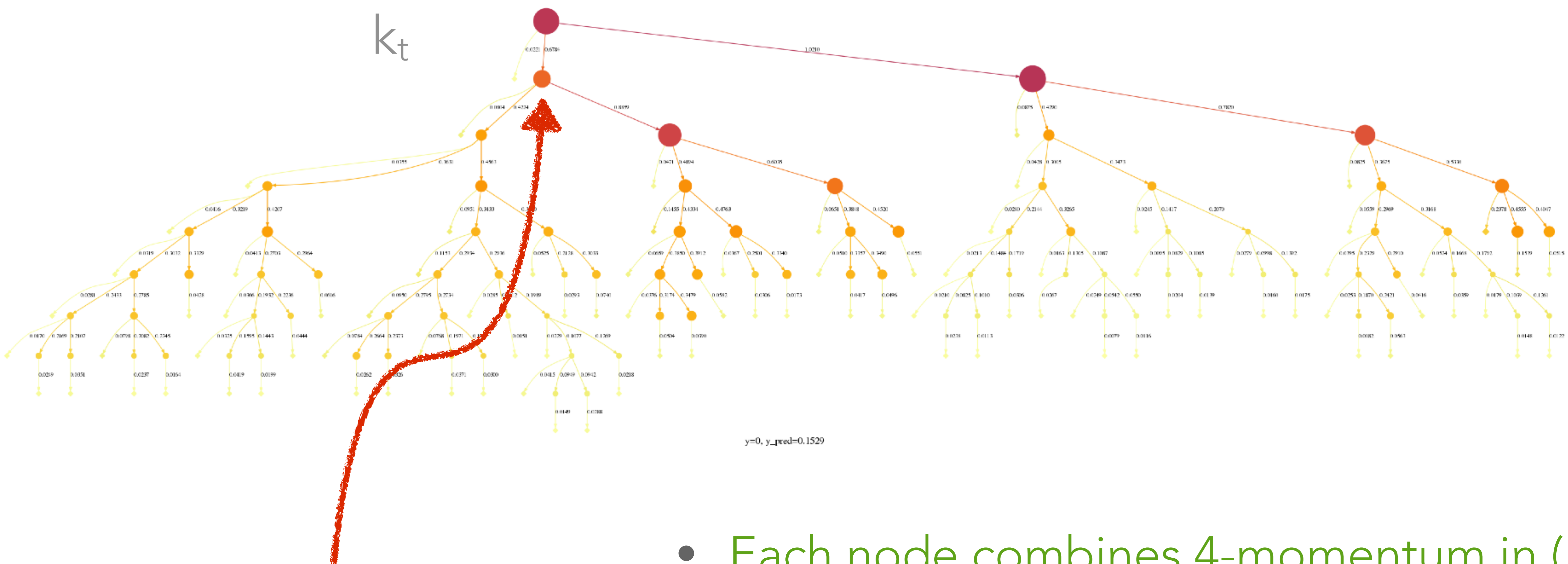


Work with Gilles Louppe, Kyunghyun Cho, Cyril Becot

- Use sequential recombination jet algorithms to provide network topology (on a **per-jet basis**)
- path towards ML models with good physics properties
- Top node of recursive network provides a fixed-length **embedding** of a jet that can be fed to a classifier

arXiv:1702.00748 & follow up work with Joan Bruna using graph conv nets

QCD-INSPIRED RECURSIVE NEURAL NETWORKS



$$\mathbf{h}_k^{\text{jet}} = \begin{cases} \mathbf{u}_k & \text{if } k \text{ is a leaf} \\ \mathbf{z}_H \odot \tilde{\mathbf{h}}_k^{\text{jet}} + \mathbf{z}_L \odot \mathbf{h}_{k_L}^{\text{jet}} + & \text{otherwise} \\ \hookrightarrow \mathbf{z}_R \odot \mathbf{h}_{k_R}^{\text{jet}} + \mathbf{z}_N \odot \mathbf{u}_k & \end{cases}$$

$$\mathbf{u}_k = \sigma\left(W_u g(\mathbf{o}_k) + b_u\right)$$

$$\mathbf{o}_k = \begin{cases} \mathbf{v}_{i(k)} & \text{if } k \text{ is a leaf} \\ \mathbf{o}_{k_L} + \mathbf{o}_{k_R} & \text{otherwise} \end{cases}$$

$$\tilde{\mathbf{h}}_k^{\text{jet}} = \sigma\left(W_{\tilde{h}} \begin{bmatrix} \mathbf{r}_L \odot \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{r}_R \odot \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{r}_N \odot \mathbf{u}_k \end{bmatrix} + b_{\tilde{h}}\right)$$

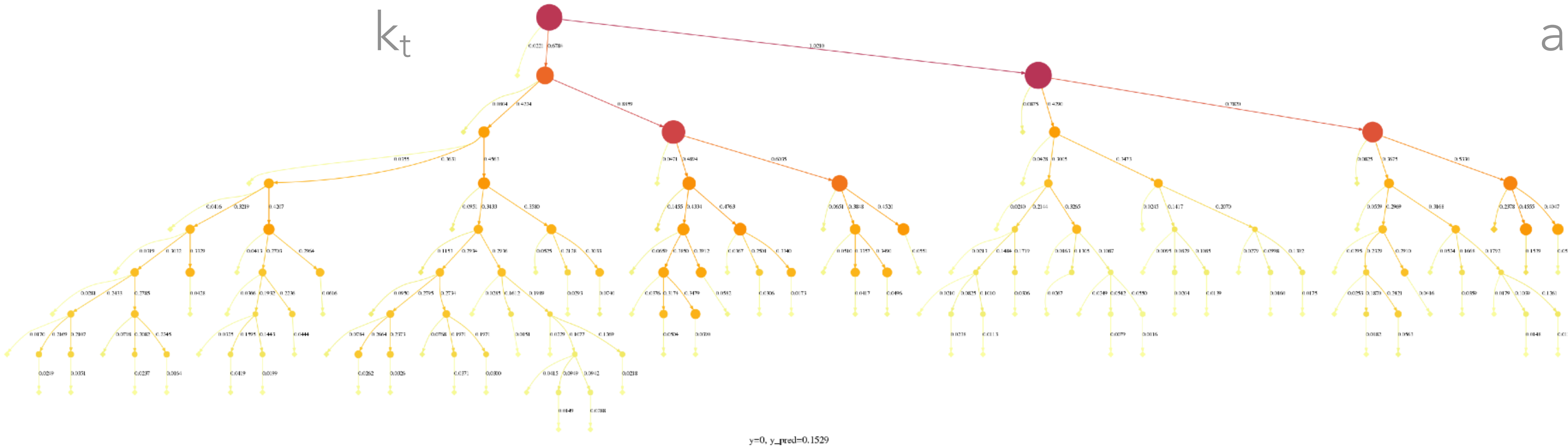
$$\begin{bmatrix} \mathbf{z}_H \\ \mathbf{z}_L \\ \mathbf{z}_R \\ \mathbf{z}_N \end{bmatrix} = \text{softmax}\left(W_z \begin{bmatrix} \tilde{\mathbf{h}}_k^{\text{jet}} \\ \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{u}_k \end{bmatrix} + b_z\right)$$

$$\begin{bmatrix} \mathbf{r}_L \\ \mathbf{r}_R \\ \mathbf{r}_N \end{bmatrix} = \text{sigmoid}\left(W_r \begin{bmatrix} \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{u}_k \end{bmatrix} + b_r\right)$$

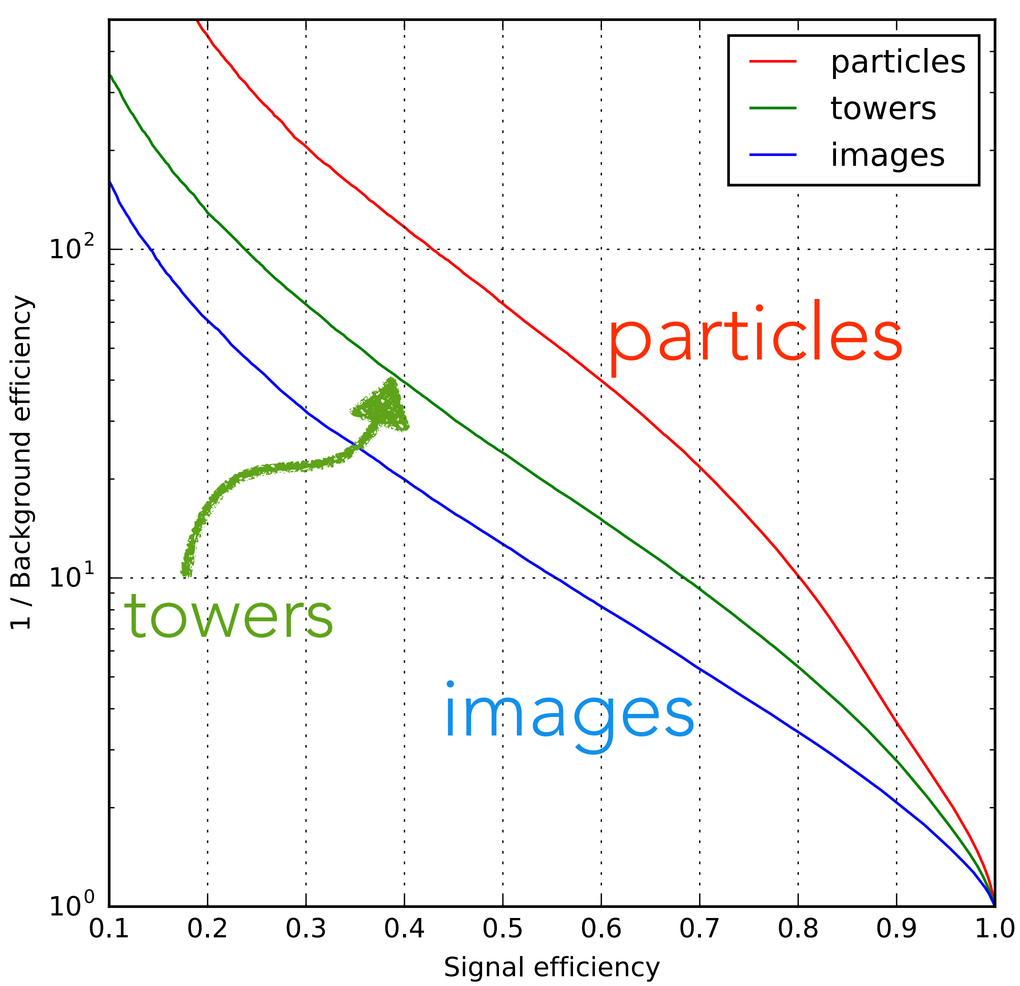
- Each node combines 4-momentum in (E-scheme recombination of o_k) and a non-linear transformation of hidden state of children h_{kL}, $h_{kR} \in \mathbb{R}^{40}$
- Recursively applied (shared weights, Markov)
- "gating" allows for weighting of information of L/R children and for to flow directly along one branch

QCD-INSPIRED RECURSIVE NEURAL NETWORKS

k_t

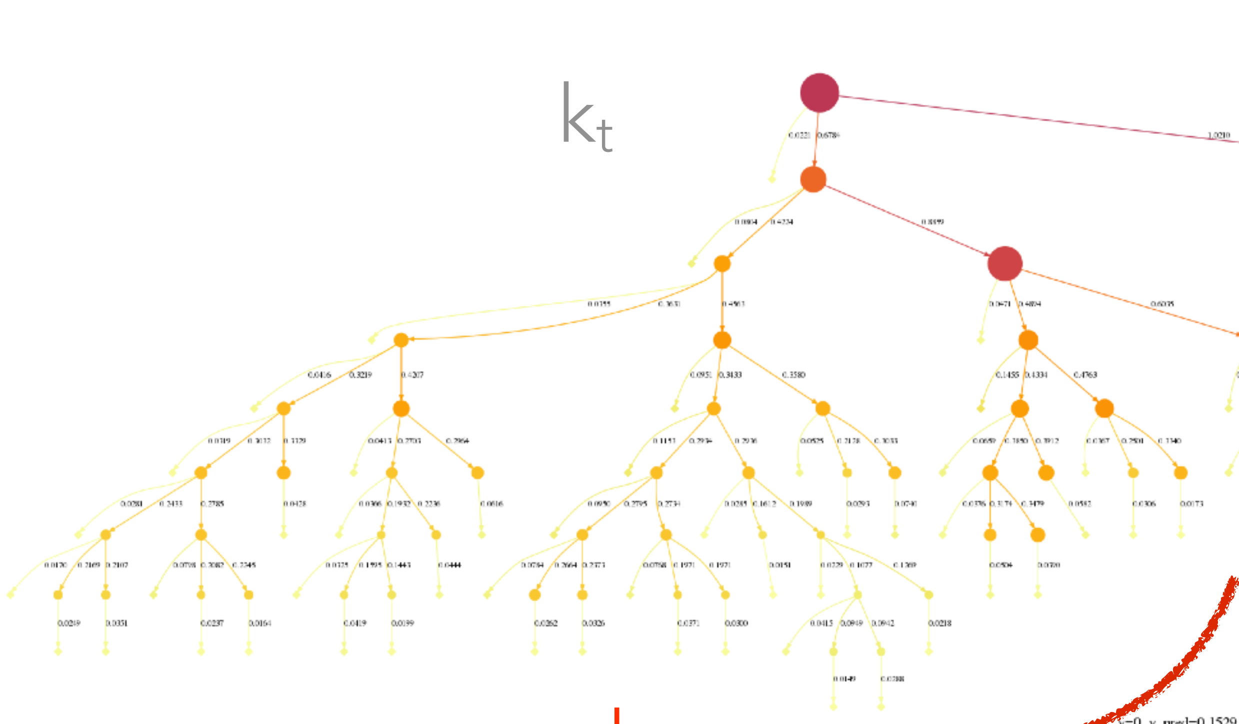


anti-k_t

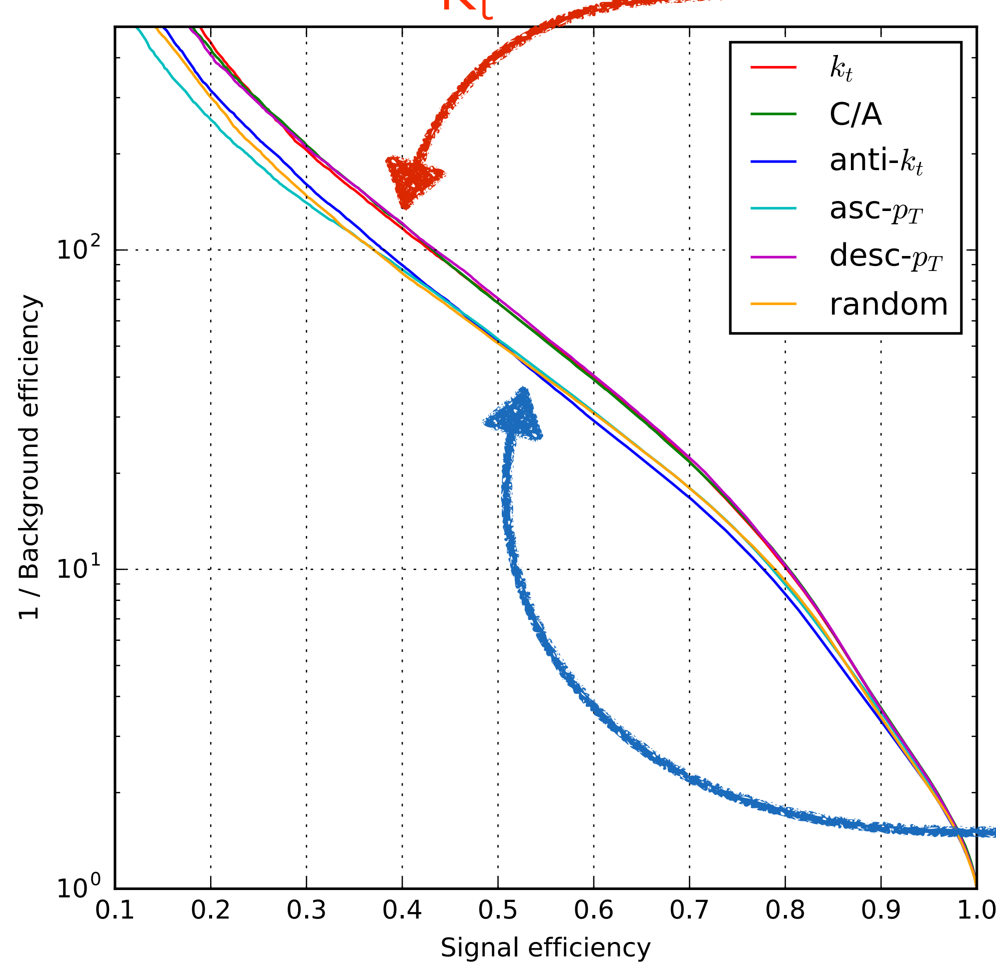


- W-jet tagging example using data from Dawe, et al arXiv:1609.00607
- down-sampling by projecting into images looses information
- RNN needs much less data to train!

QCD-INSPIRED RECURSIVE NEURAL NETWORKS



k_t



- choice of jet algorithm matters
- “gating” improves performance

anti-k_t

JET-LEVEL CLASSIFICATION RESULTS

TABLE I. Summary of jet classification performance for several approaches applied either to particle-level inputs or towers from a DELPHES simulation.

Input	Architecture	ROC AUC	$R_{\epsilon=50\%}$
Projected into images			
towers	MaxOut	**0.8418**	–
towers	k_t	0.8321 ± 0.0025	**12.7 ± 0.4**
towers	k_t (gated)	0.8277 ± 0.0028	12.4 ± 0.3
Without image preprocessing			
towers	τ_{21}	0.7644	6.79
towers	mass + τ_{21}	0.8212	11.31
towers	k_t	0.8807 ± 0.0010	24.1 ± 0.6
towers	C/A	0.8831 ± 0.0010	24.2 ± 0.7
towers	anti-k_t	0.8737 ± 0.0017	22.3 ± 0.8
towers	asc-p_T	0.8835 ± 0.0009	**26.2 ± 0.7**
towers	desc-p_T	**0.8838 ± 0.0010**	25.1 ± 0.6
towers	random	0.8704 ± 0.0011	20.4 ± 0.3
particles	k_t	0.9185 ± 0.0006	68.3 ± 1.8
particles	C/A	**0.9192 ± 0.0008**	68.3 ± 3.6
particles	anti-k_t	0.9096 ± 0.0013	51.7 ± 3.5
particles	asc-p_T	0.9130 ± 0.0031	52.5 ± 7.3
particles	desc-p_T	0.9189 ± 0.0009	**70.4 ± 3.6**
particles	random	0.9121 ± 0.0008	51.1 ± 2.0
With gating (see Appendix A)			
towers	k_t	0.8822 ± 0.0006	25.4 ± 0.4
towers	C/A	0.8861 ± 0.0014	26.2 ± 0.8
towers	anti-k_t	0.8804 ± 0.0010	24.4 ± 0.4
towers	asc-p_T	0.8849 ± 0.0012	27.2 ± 0.8
towers	desc-p_T	**0.8864 ± 0.0007**	**27.5 ± 0.6**
towers	random	0.8751 ± 0.0029	22.8 ± 1.2
particles	k_t	0.9195 ± 0.0009	74.3 ± 2.4
particles	C/A	**0.9222 ± 0.0007**	81.8 ± 3.1
particles	anti-k_t	0.9156 ± 0.0012	68.3 ± 3.2
particles	asc-p_T	0.9137 ± 0.0046	54.8 ± 11.7
particles	desc-p_T	0.9212 ± 0.0005	**83.3 ± 3.1**
particles	random	0.9106 ± 0.0035	50.7 ± 6.7

When working on images:

- recursive network has similar performance to previous approaches

Improved performance when working with calo towers without image pre-processing

- loss of information depends on details of calorimeter, pixelation, etc.

Working on truth-level particles led to a significant improvement

- generically expect information from tracking, particle flow, etc. to be somewhere between towers and truth particle-level

Neural Message Passing for Jet Physics

Isaac Henrion, Johann Brehmer, Joan Bruna, Kyunghyun Cho, Kyle Cranmer, Gilles Louppe, Gaspar Rochette

Courant Institute & Center for Data Science

NYU

Paper: https://dl4physicalsciences.github.io/files/nips_dlps_2017_29.pdf
Talk: https://dl4physicalsciences.github.io/files/nips_dlps_2017_slides_henrion.pdf

Message Passing Neural Network

Algorithm 1 Message passing neural network

Require: $N \times D$ nodes $\mathbf{x}$, adjacency matrix A

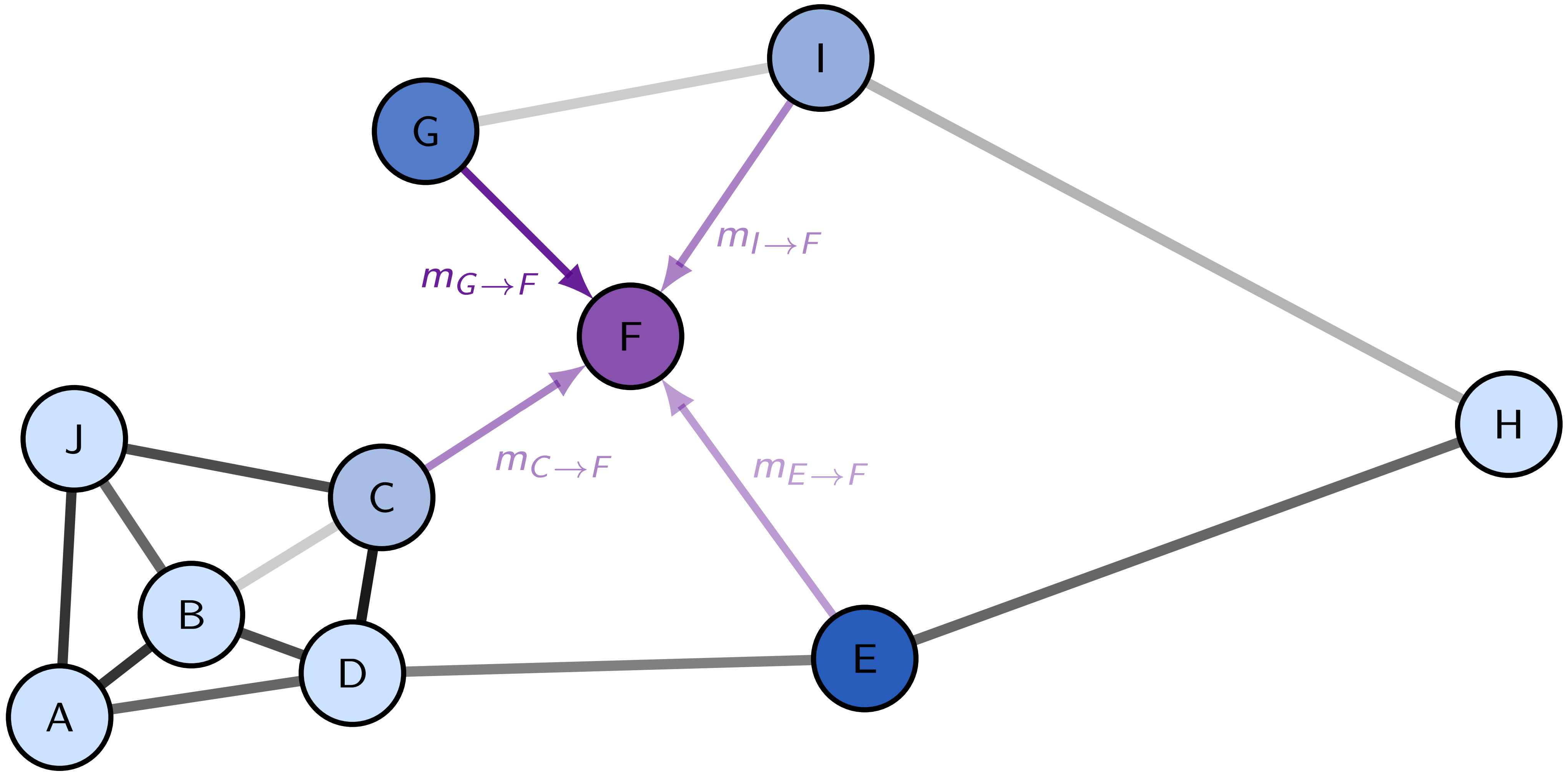
$\mathbf{h} \leftarrow \text{Embed}(\mathbf{x})$
for $t = 1, \ldots, T$ **do**
$\mathbf{m} \leftarrow \text{Message}(A, \mathbf{h})$
$\mathbf{h} \leftarrow \text{VertexUpdate}(\mathbf{h}, \mathbf{m})$
end for
$\mathbf{r} = \text{Readout}(\mathbf{h})$
return $\text{Classify}(\mathbf{r})$

Adjacency matrix generalizes receptive field of convolution kernel

Vertex Update like pooling

Iterations like layers of a CNN

Graph neural networks



$$\tilde{m}_j^t = f(h_j^{t-1})$$
$$m_{j\to i}^t = \sigma(A_{ij}\tilde{m}_j^t)$$
$$h_i^t = \mathrm{GRU}(h_i^{t-1}, \Sigma_j m_{j\to i}^t)$$

Message Passing Neural Network

Algorithm 2 Message passing neural network

Require: $N \times D$ array of jet constituents $\mathbf{x}$

$\mathbf{h} \leftarrow \text{Embed}(\mathbf{x})$

for $t = 1, \ldots, T$ **do**

$A \leftarrow \text{AdjacencyMatrix}_t(\mathbf{h})$

$\mathbf{m} \leftarrow \text{Message}_t(A, \mathbf{h})$

$\mathbf{h} \leftarrow \text{VertexUpdate}_t(\mathbf{h}, \mathbf{m})$

end for

$\mathbf{r} = \text{Readout}(\mathbf{h})$

return $\text{Classify}(\mathbf{r})$

Difference from Alg 1:

new weights for each iteration (layer) of message passing

A problem with the adjacency matrix

Question

Where does adjacency matrix come from?

Answer 1

Use a physics-inspired adjacency matrix.

BONUS: import physics knowledge

Answer 2

Learn the adjacency matrix from the data.

BONUS: export physics algorithm

Very interesting:

adjacency matrix can be **interpreted** like a kT, C/A, anti-kT

Once learned, can **export** the adjacency function for other uses

Provides bi-directional **interface** between ML and jet physics.

Learning the adjacency matrix

$$F(h, h') = v^\top (h + h') + b$$

$$s_{ij}^t = F(h_i^{t-1}, h_j^{t-1})$$

$$A_{ij}^t = \frac{\exp\{s_{ij}^t\}}{\sum_k \exp\{s_{ik}^t\}} \qquad \text{(directed)}$$

$$A_{\text{sym}} = \frac{1}{2}\left(A + A^\top\right) \qquad \text{(undirected)}$$

This is a simple starting point, not motivated by physics

Classification results

1/FPR @ TPR = 50%

Model	Iterations	$R_{\epsilon=50\%}$
Rec-NN (no gating)	1	70.4 ± 3.6
Rec-NN (gating)	1	$\mathbf{83.3 \pm 3.1}$
MPNN (directed)	1	89.4 ± 3.5
MPNN (directed)	2	$\mathbf{98.3 \pm 4.3}$
MPNN (directed)	3	85.9 ± 8.5
MPNN (identity)	3	74.5 ± 5.2
Relation Network	1	67.7 ± 6.8

Significant improvement on W vs. QCD tagging!

This is with a learned adjacency matrix

what did it learn? Is that adjacency matrix useful?

we are working MPNN with QCD-motivated adjacency matrix