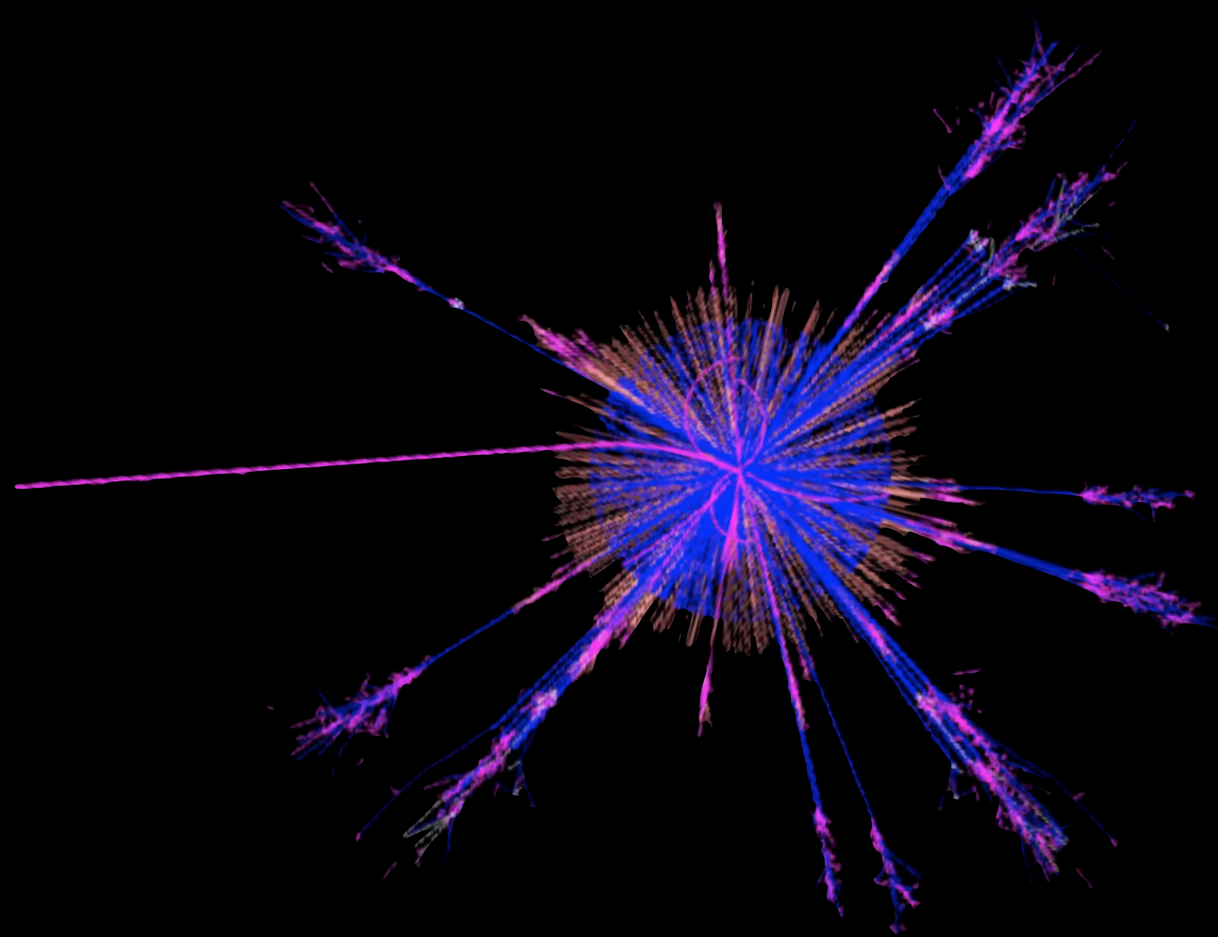
NYU Center for Data Science

Center for Cosmology and particle physics

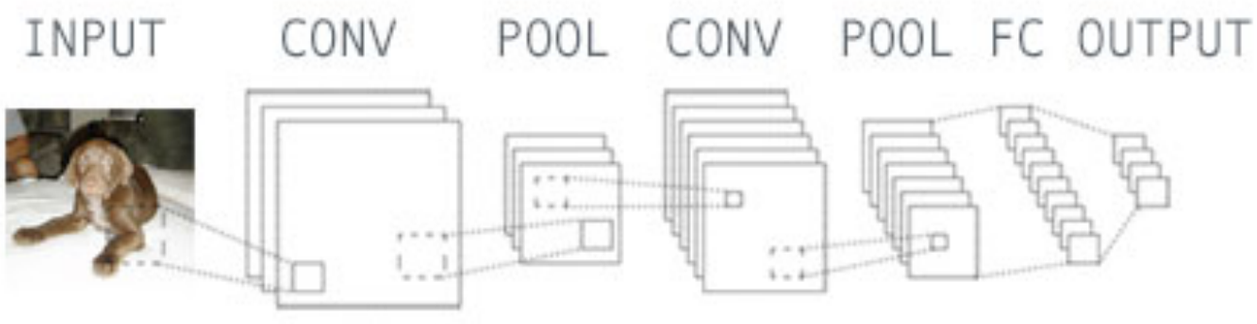
WHAT DOES THE REVOLUTION IN ARTIFICIAL INTELLIGENCE MEAN FOR PHYSICS?

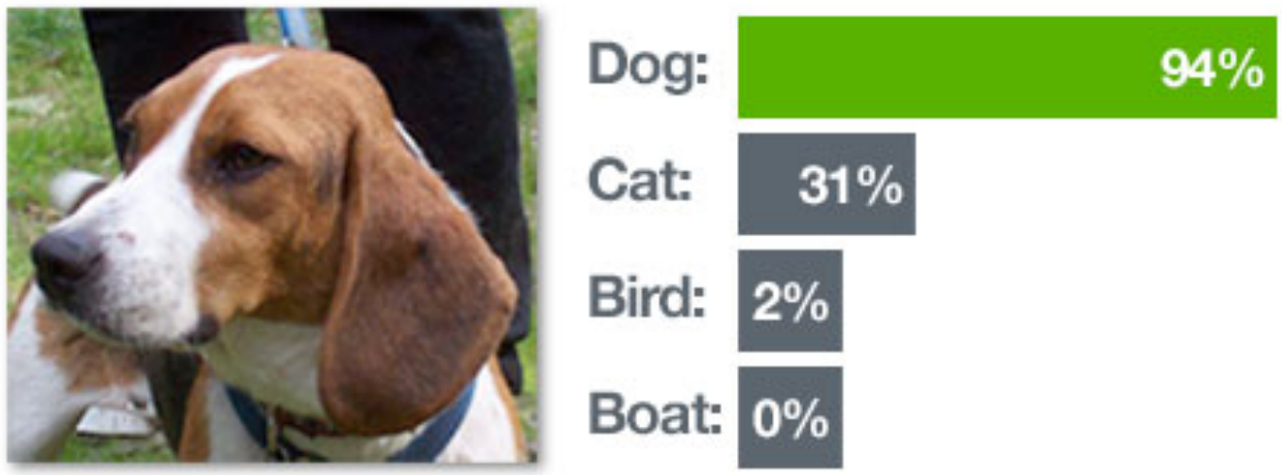


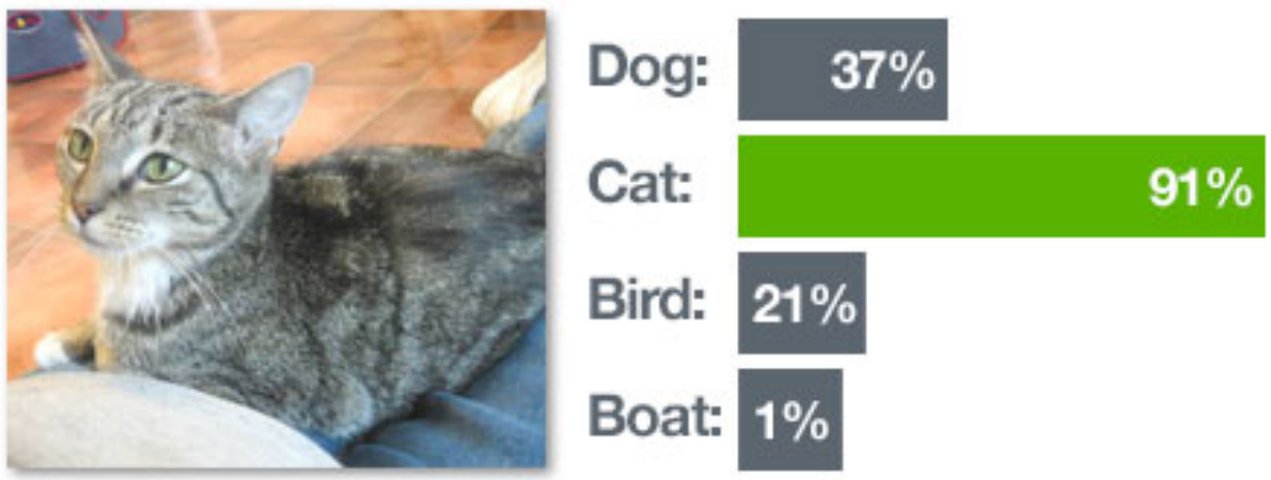
@KyleCranmer
New York University
Department of Physics
Center for Data Science
CILVR Lab

Deep Learning:
A Revolution in AI

IMAGE CLASSIFICATION







2012

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

2015

News & Analysis

Microsoft, Google Beat Humans at Image Recognition

Deep learning algorithms compete at ImageNet challenge

R. Colin Johnson
2/18/2015 03:15 AM EST
14 comments

1 saves
LOGIN TO RATE

CLASSIFICATION → SEGMENTATION

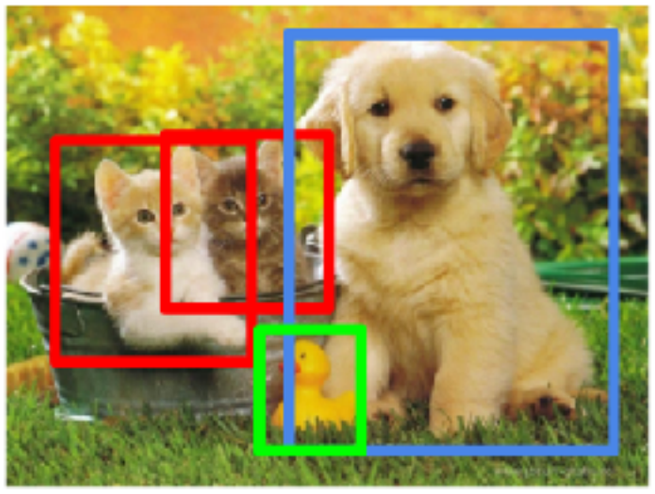
Classification

CAT

Classification + Localization

CAT

Object Detection



CAT, DOG, DUCK

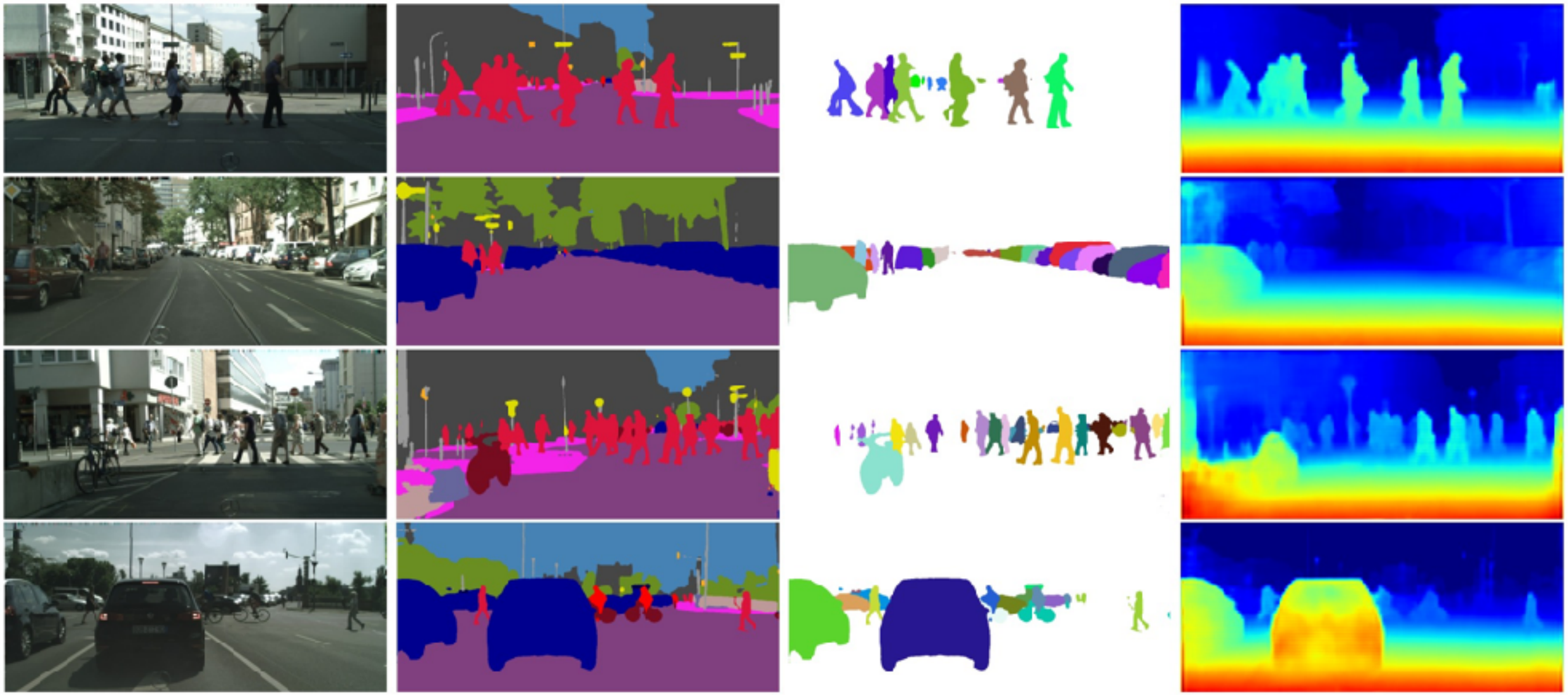
Instance Segmentation

CAT, DOG, DUCK

Single object

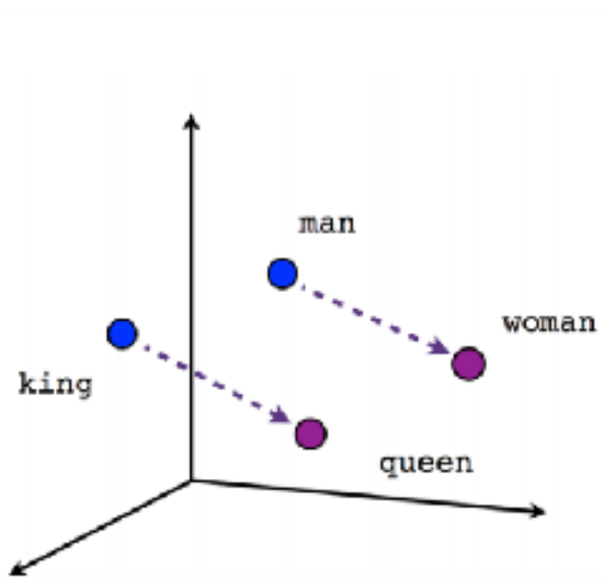
Multiple objects

COMPUTER VISION

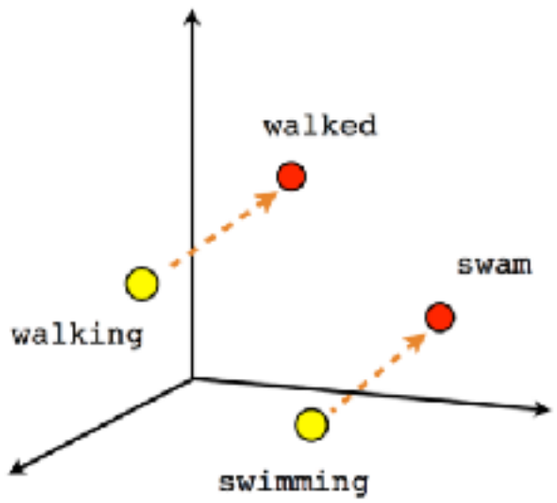


(a) Input image (b) Segmentation output (c) Instance output (d) Depth output

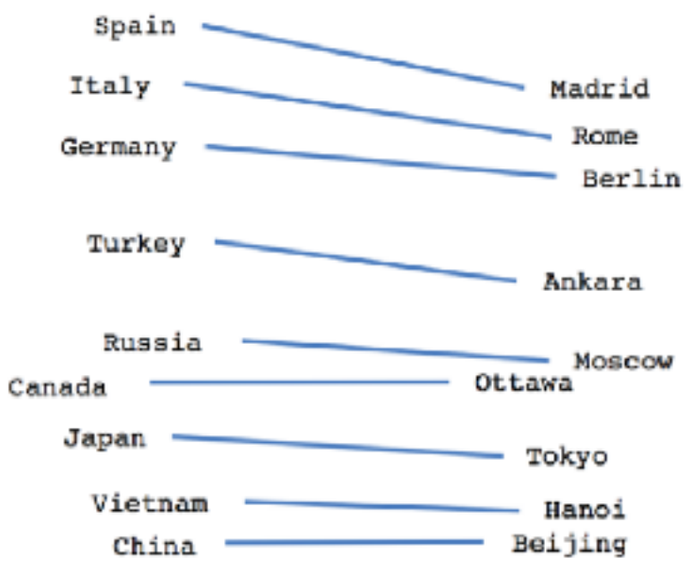
WORD EMBEDDINGS & TRANSLATION



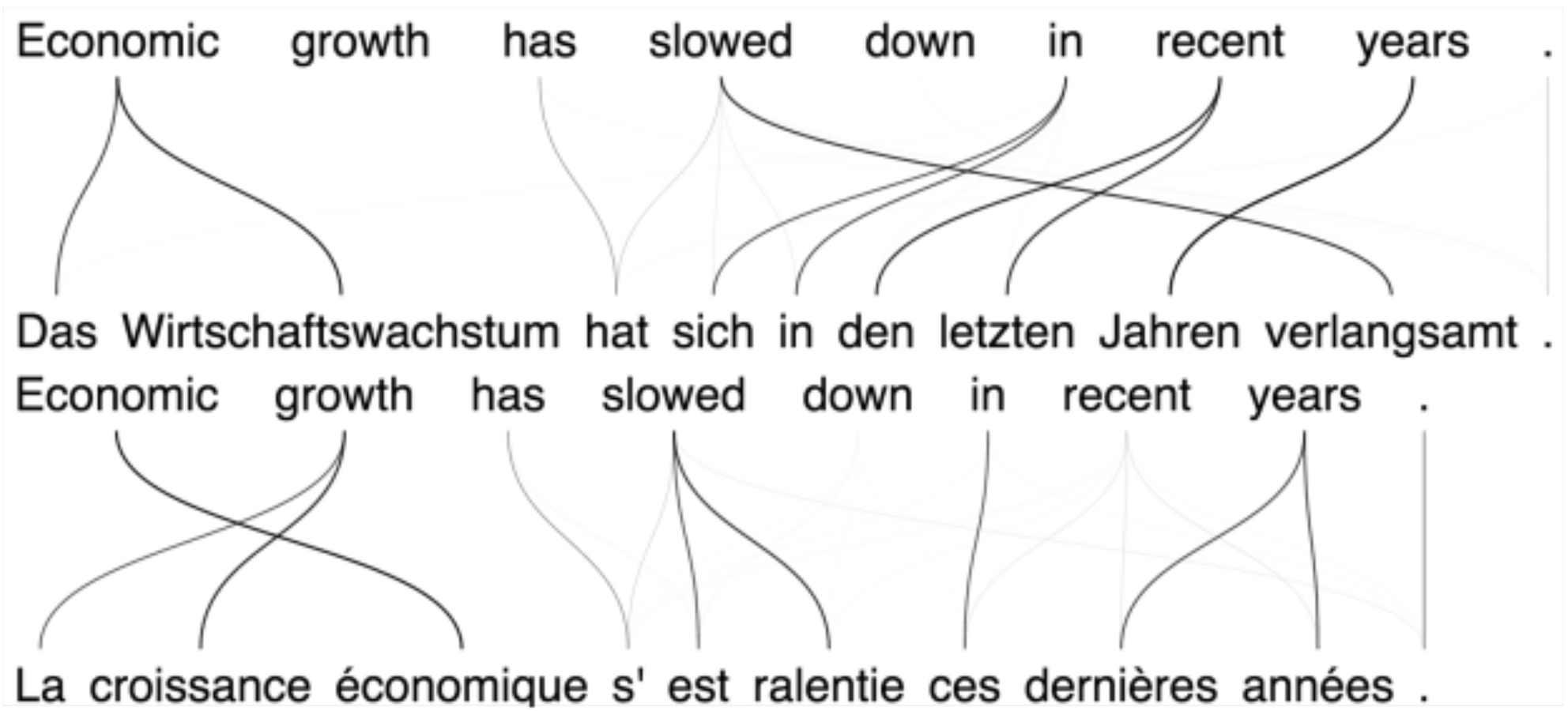
Male-Female



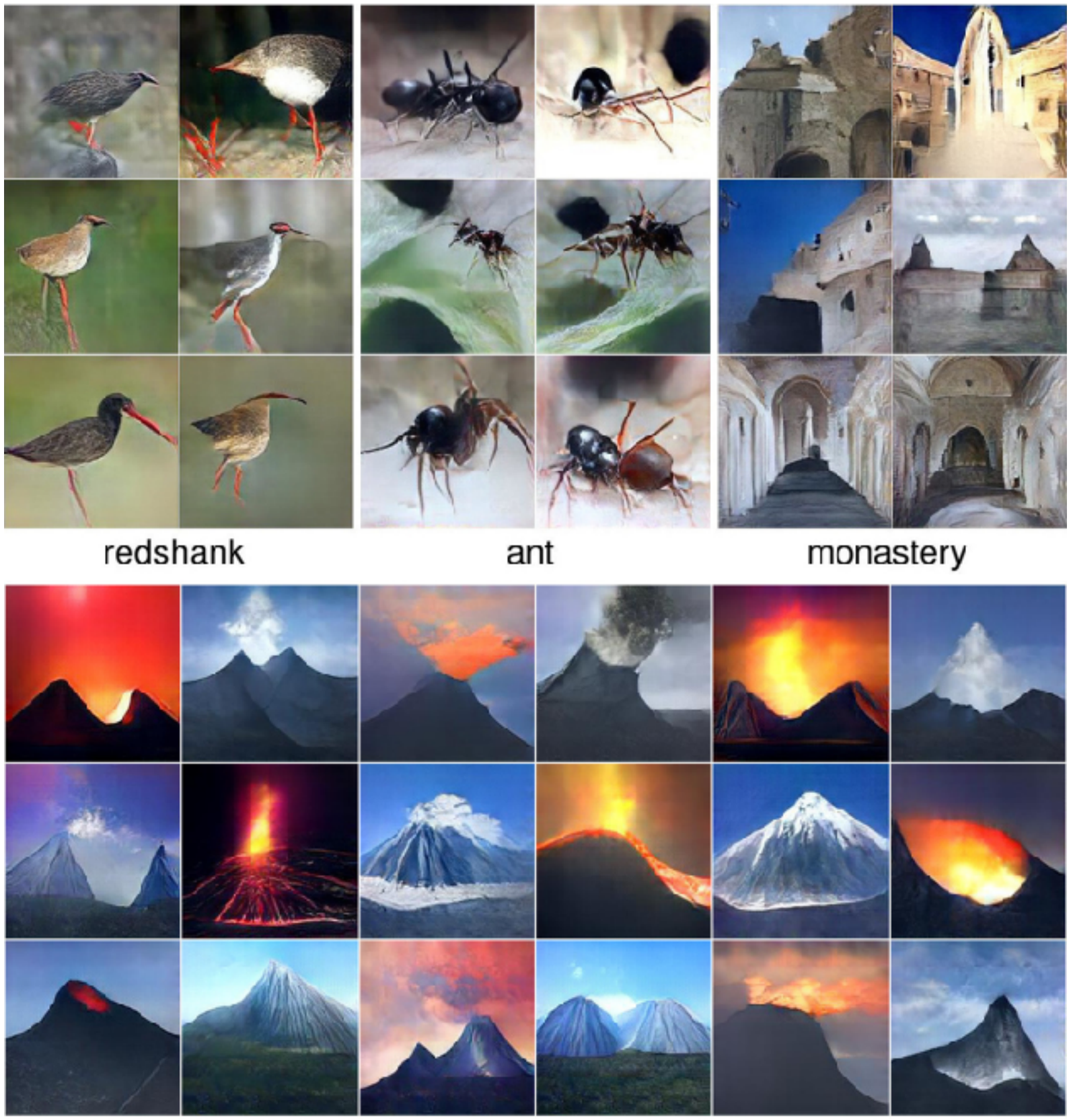
Verb tense



Country-Capital

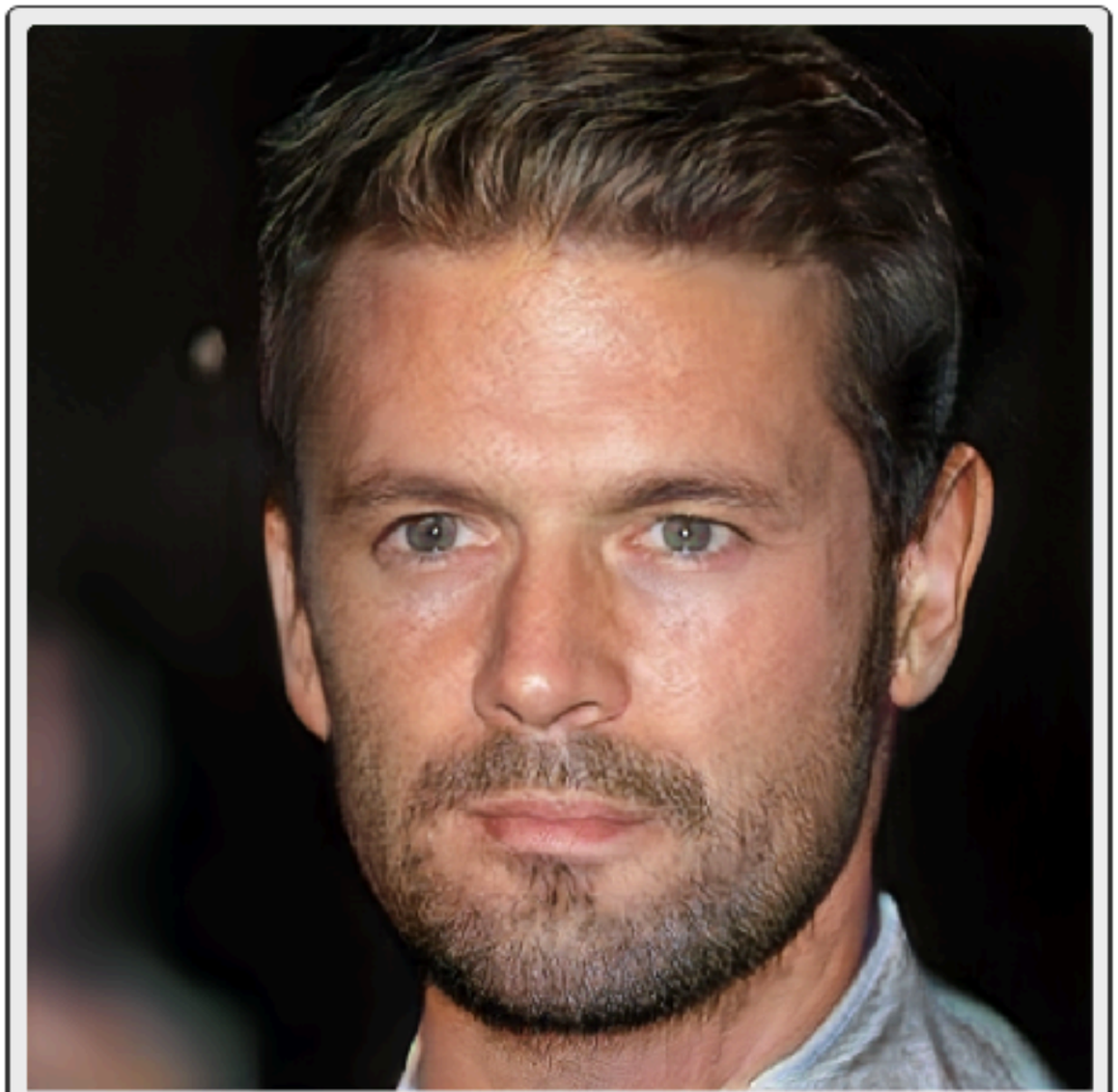


GENERATIVE MODEL FOR IMAGES



How an A.I. 'Cat-and-Mouse Game' Generates Believable Fake Photos

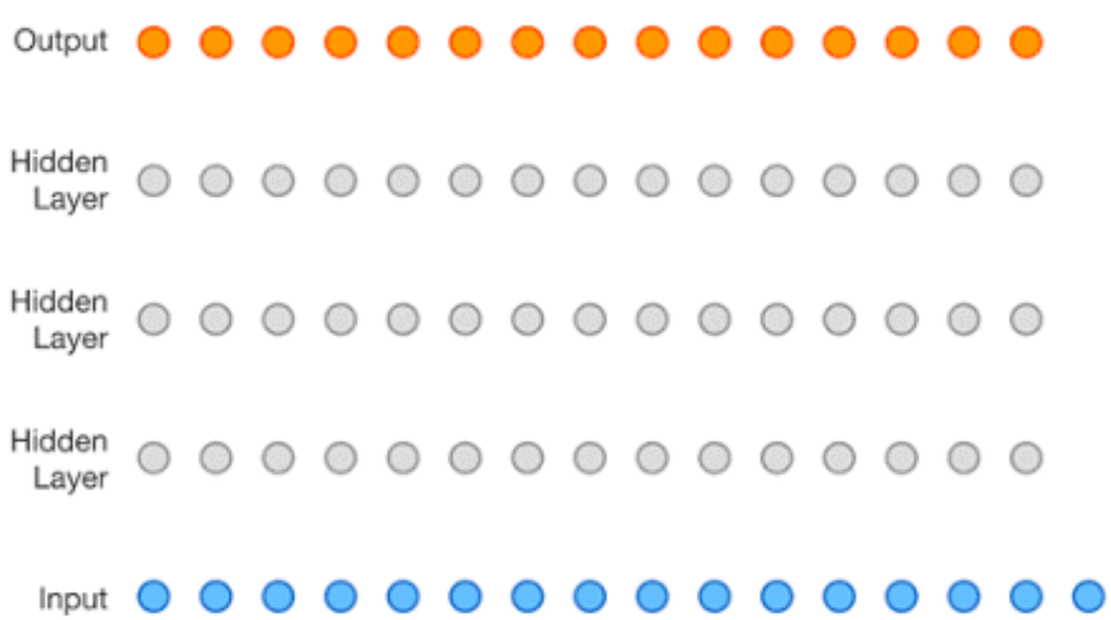
By CADE METZ and KEITH COLLINS JAN. 2, 2018

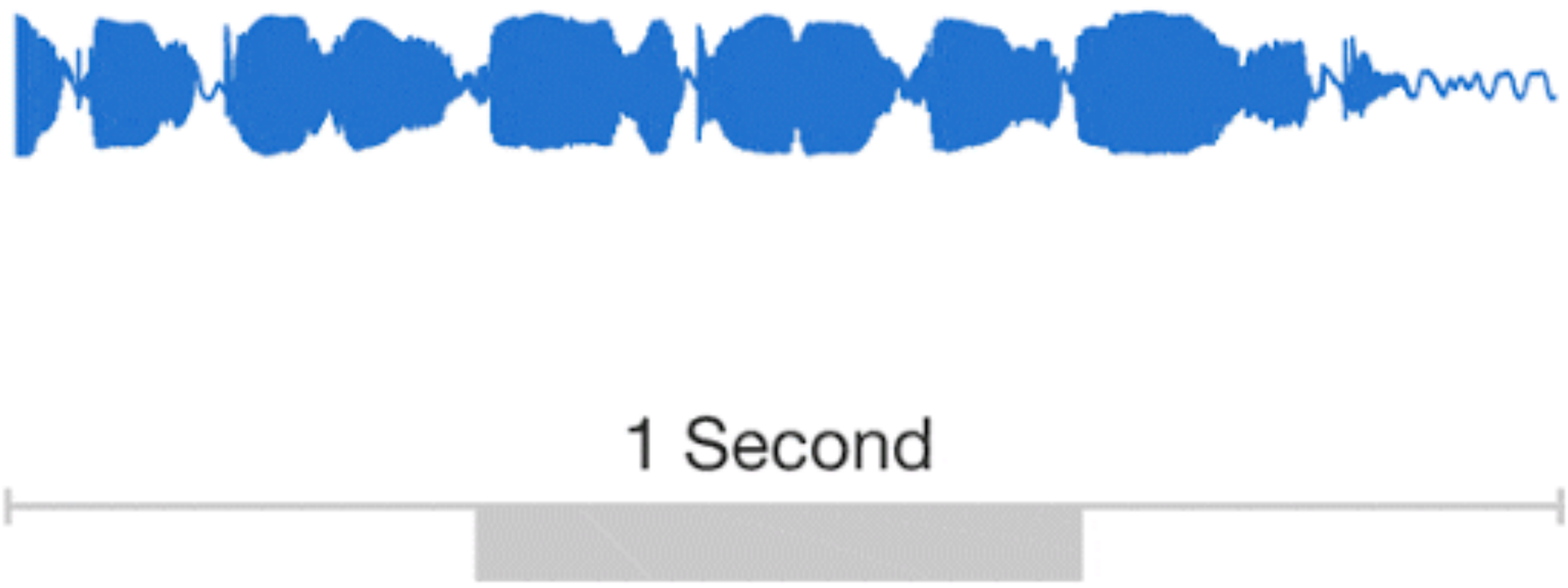


This one is computer-generated

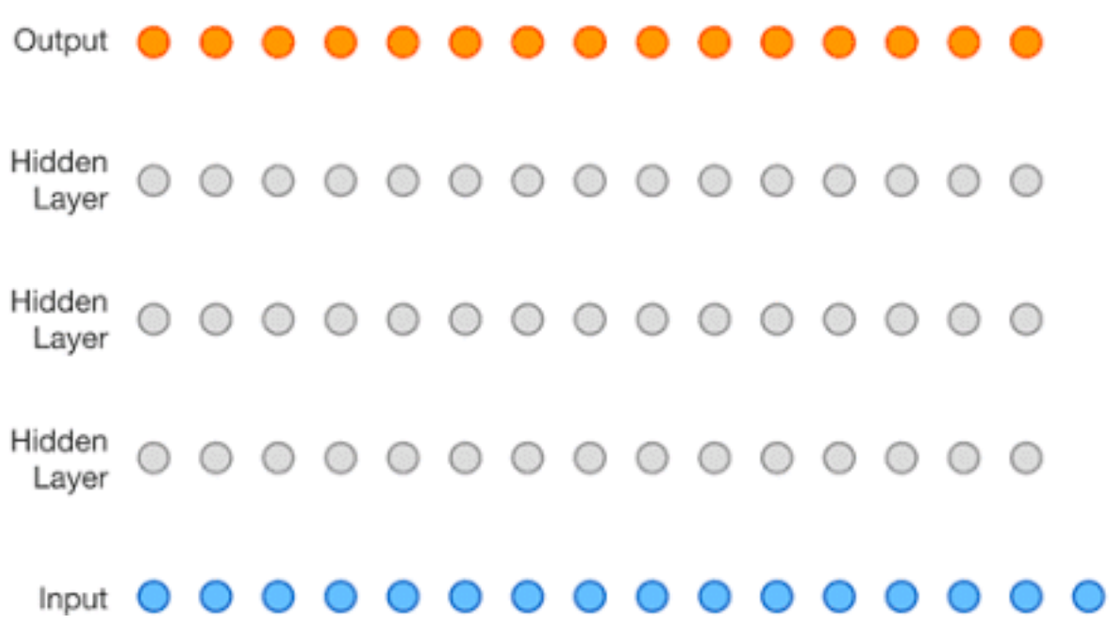
This one is also computer-generated

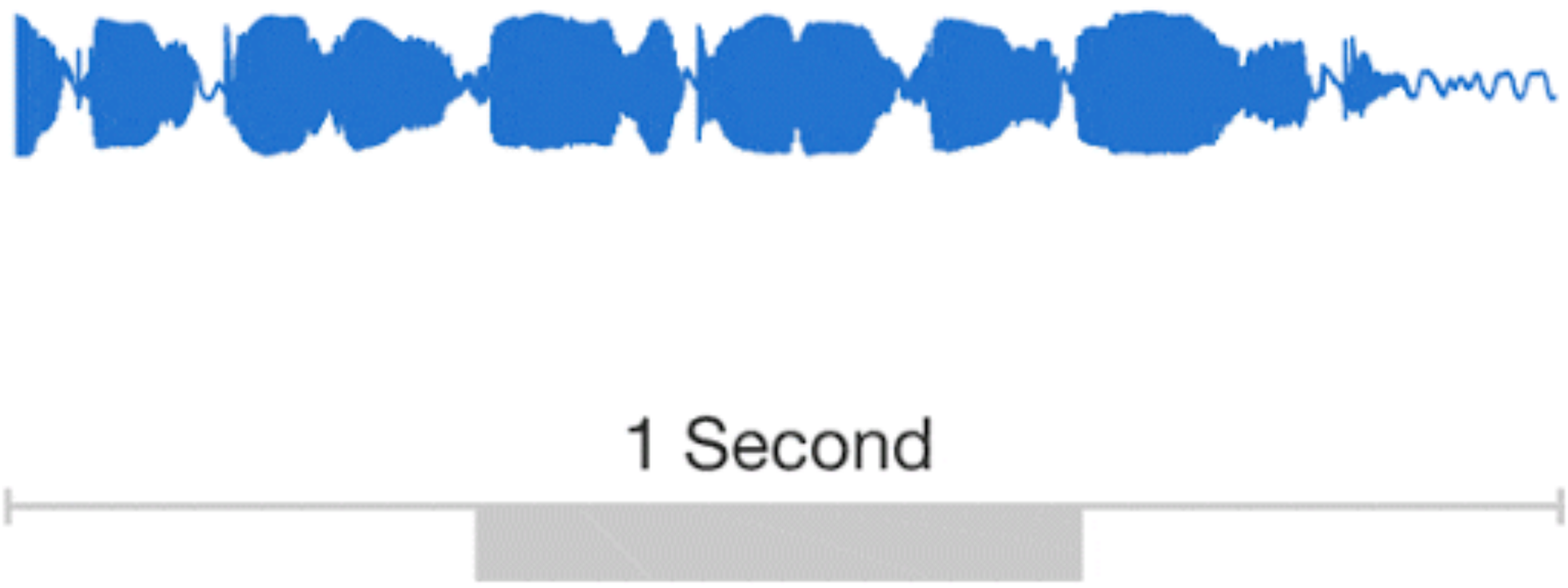
WAVENET: A GENERATIVE MODEL FOR RAW AUDIO



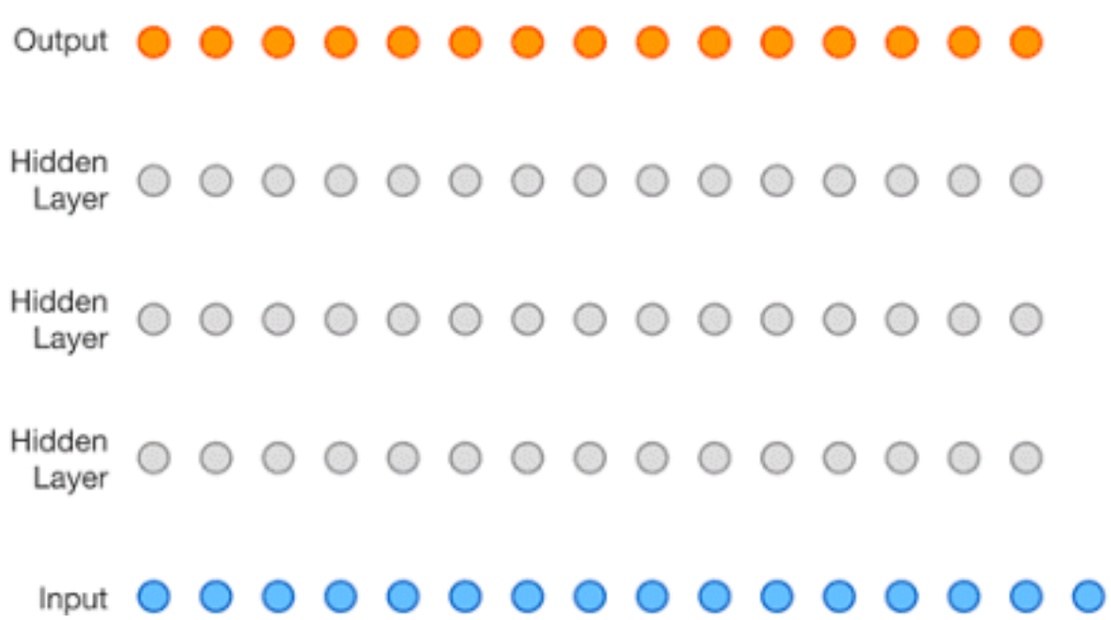


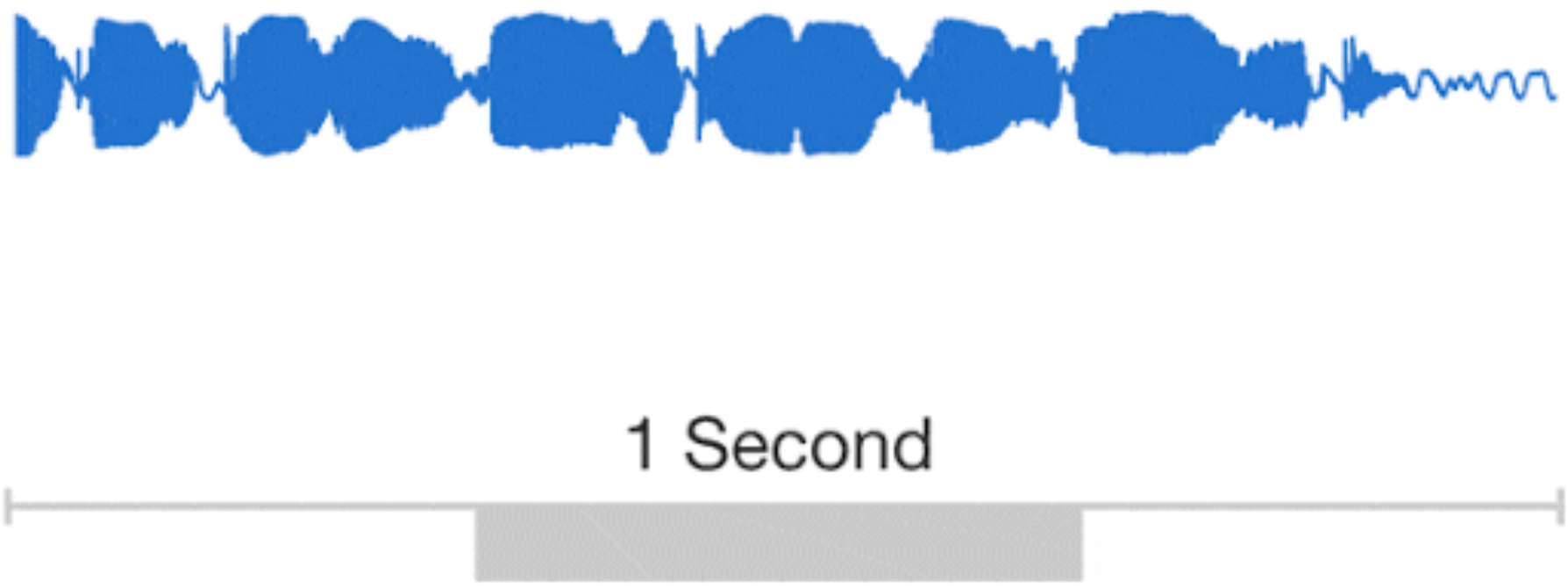
WAVENET: A GENERATIVE MODEL FOR RAW AUDIO



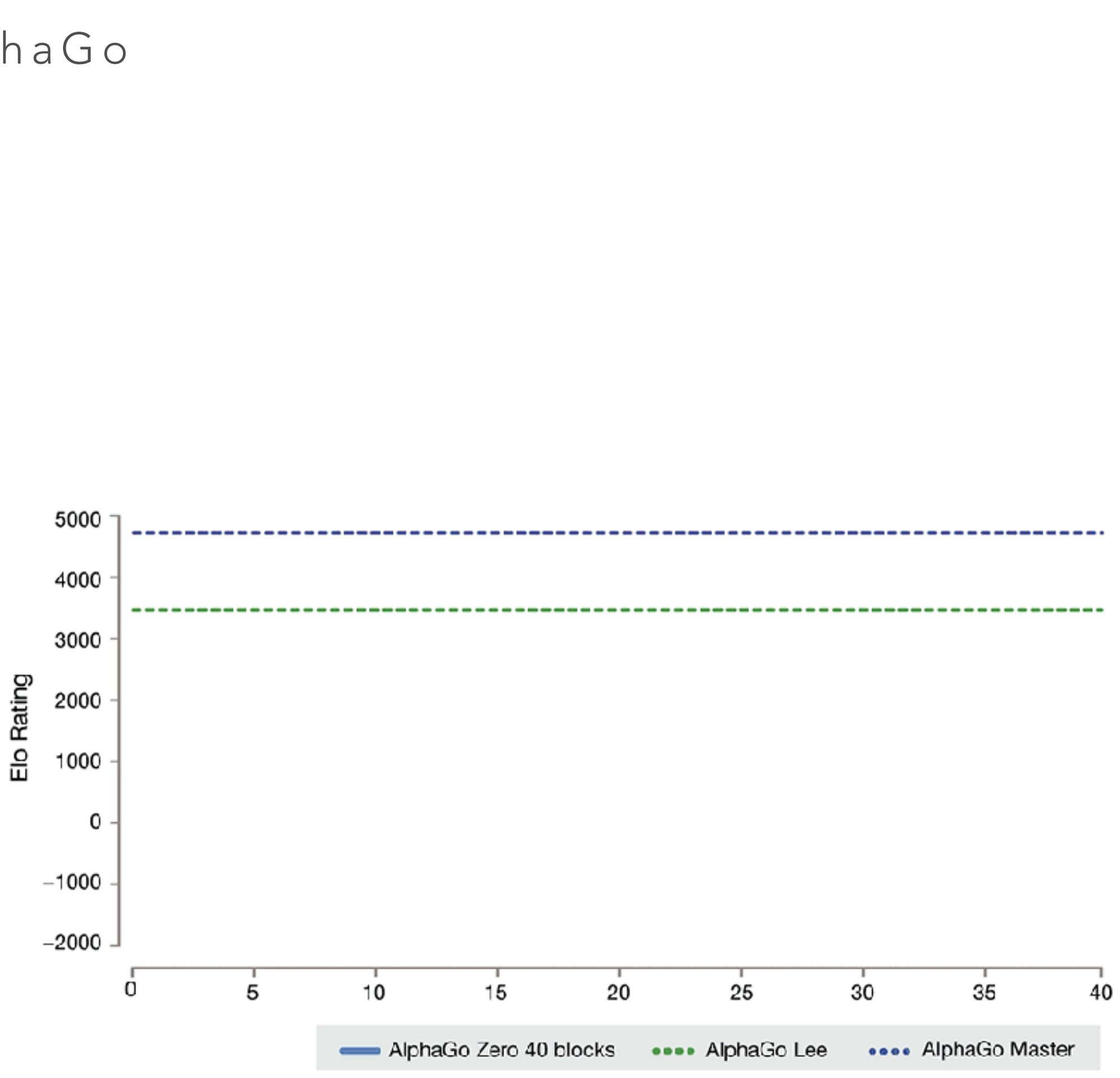


WAVENET: A GENERATIVE MODEL FOR RAW AUDIO

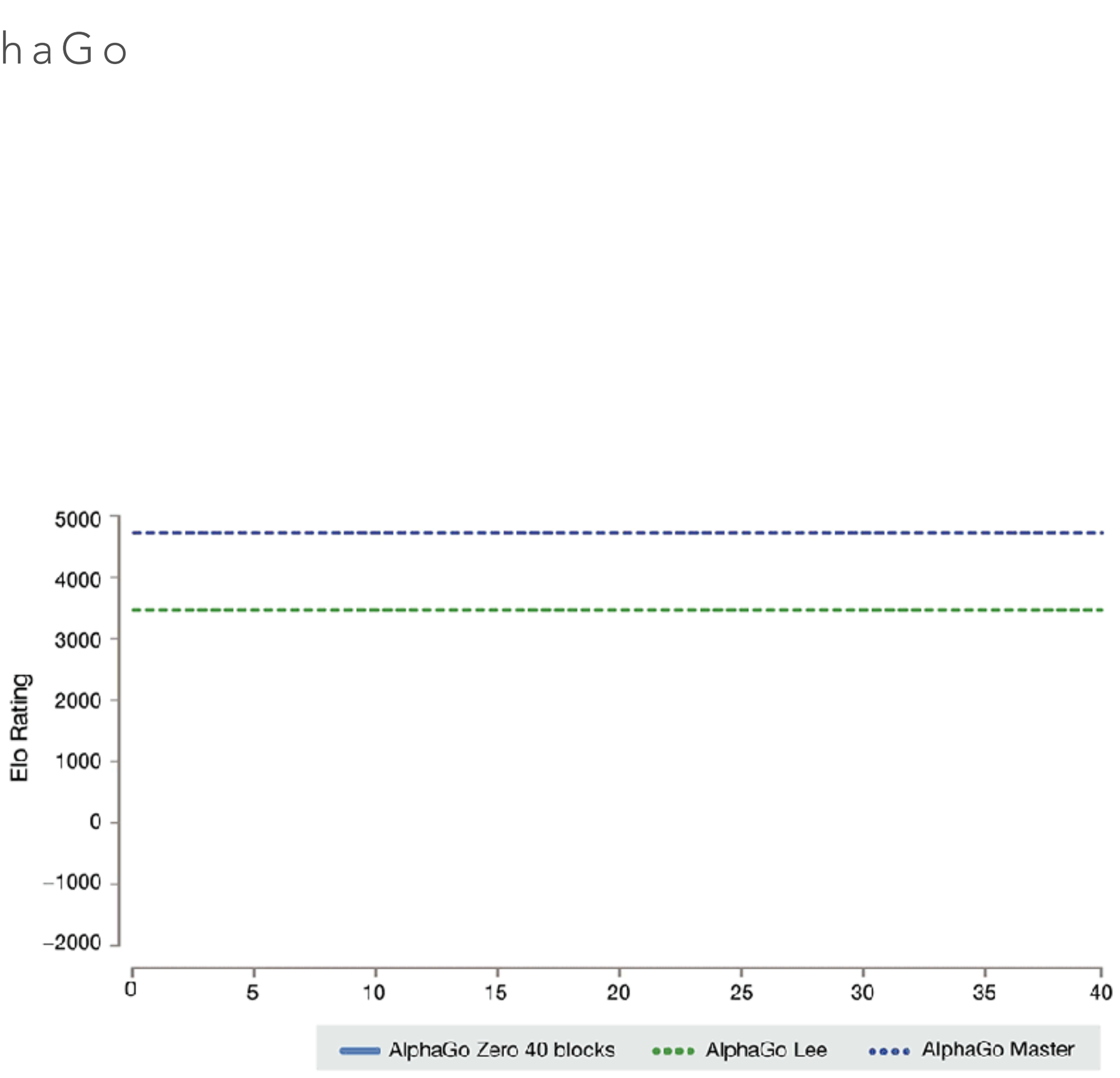




AlphaGo


5000
4000
3000
2000
1000
0
−1000
−2000
Elo Rating
0
5
10
15
20
25
30
35
40
AlphaGo Zero 40 blocks
AlphaGo Lee
AlphaGo Master

AlphaGo


5000
4000
3000
2000
1000
0
−1000
−2000
Elo Rating
0
5
10
15
20
25
30
35
40
AlphaGo Zero 40 blocks
AlphaGo Lee
AlphaGo Master

Why should physicists care?

THE PLAYERS

forward modeling
generation
simulation

PREDICTION

θ
parameters of interest

p(x, z | θ, ν)

x
observed data
simulated data

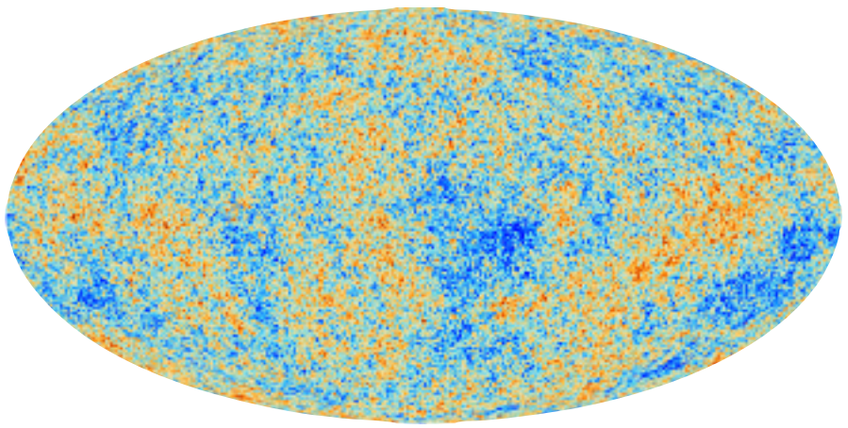
z
latent variables
Monte Carlo truth

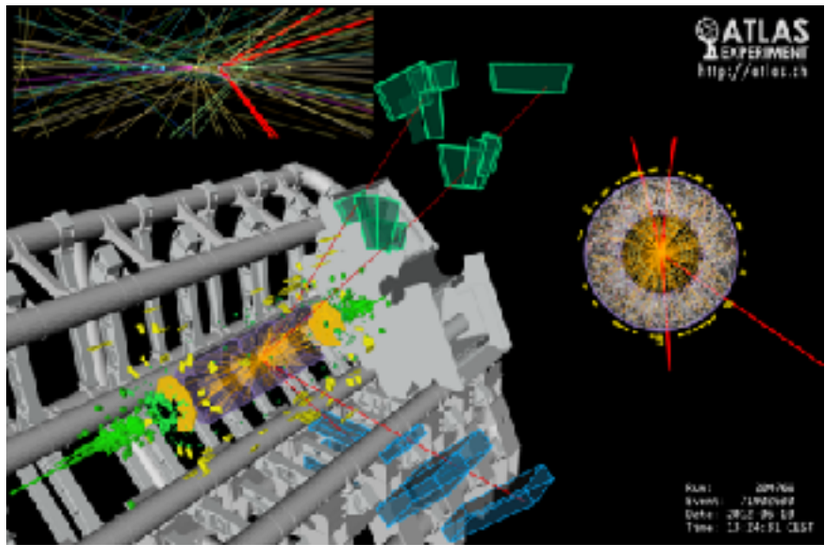
ν
nuisance parameters

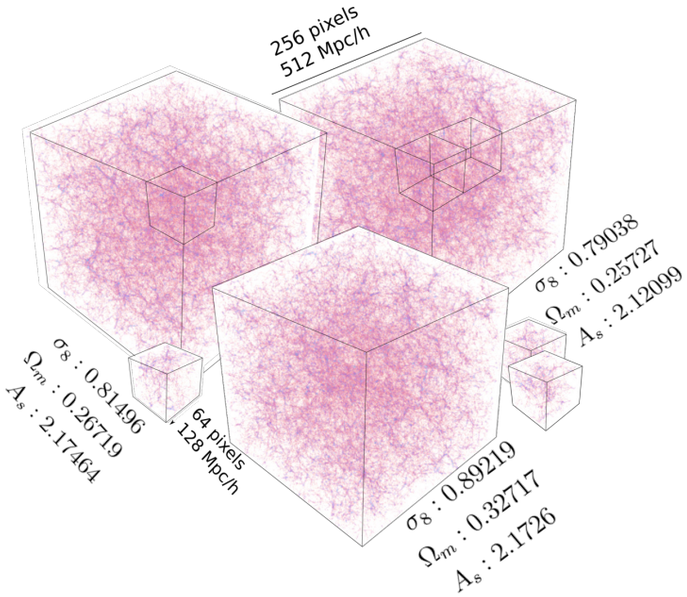
INFERENCE

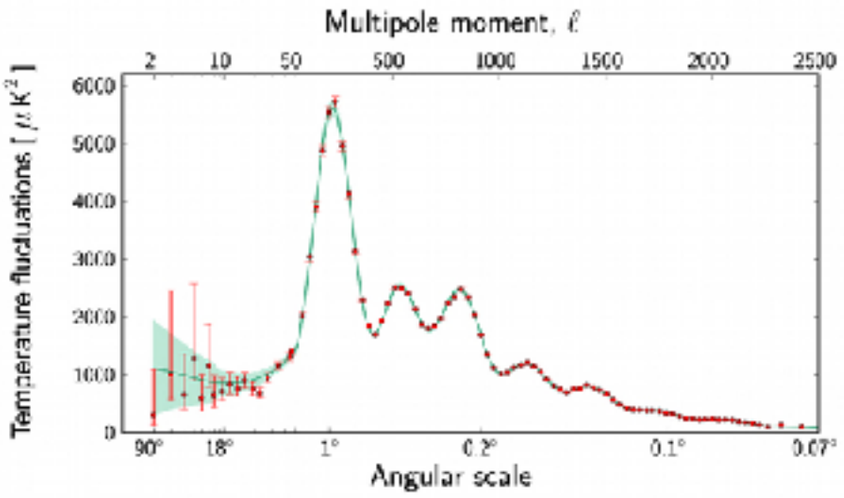
inverse problem
measurement
parameter estimation

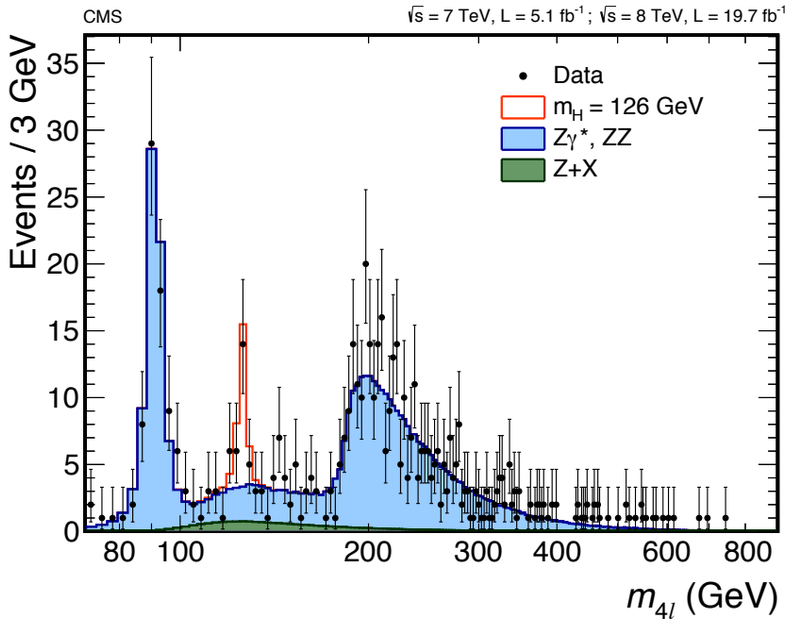
PREDICTION: THE FORWARD MODEL

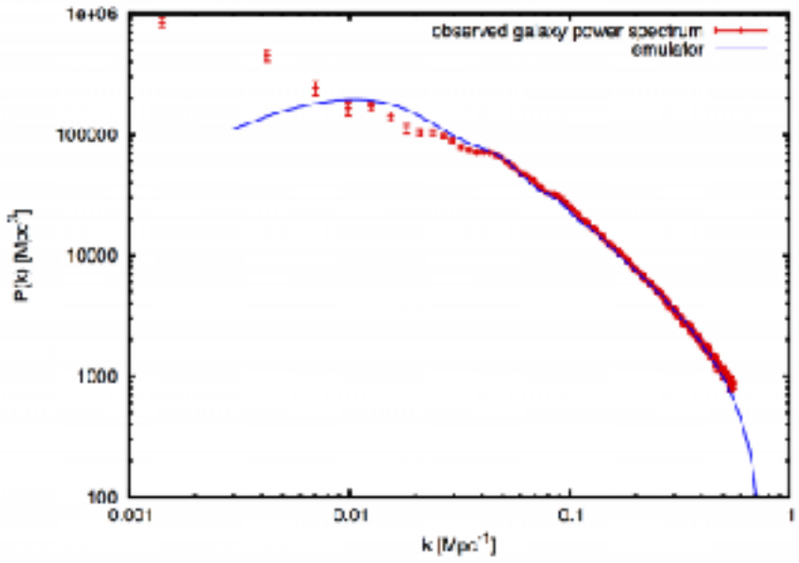












WHY WE SHOULD CARE

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 go way beyond improved classifiers and 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

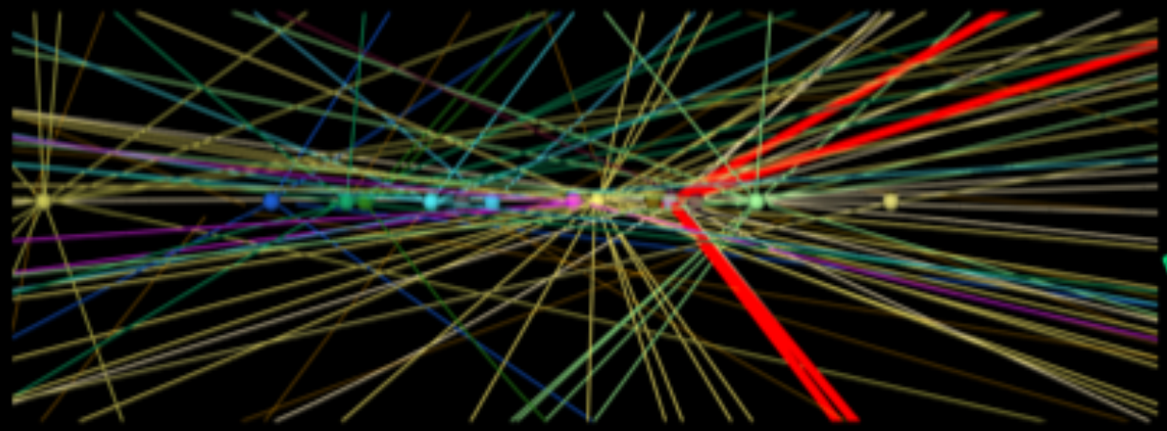
Deep Learning for Physical Sciences

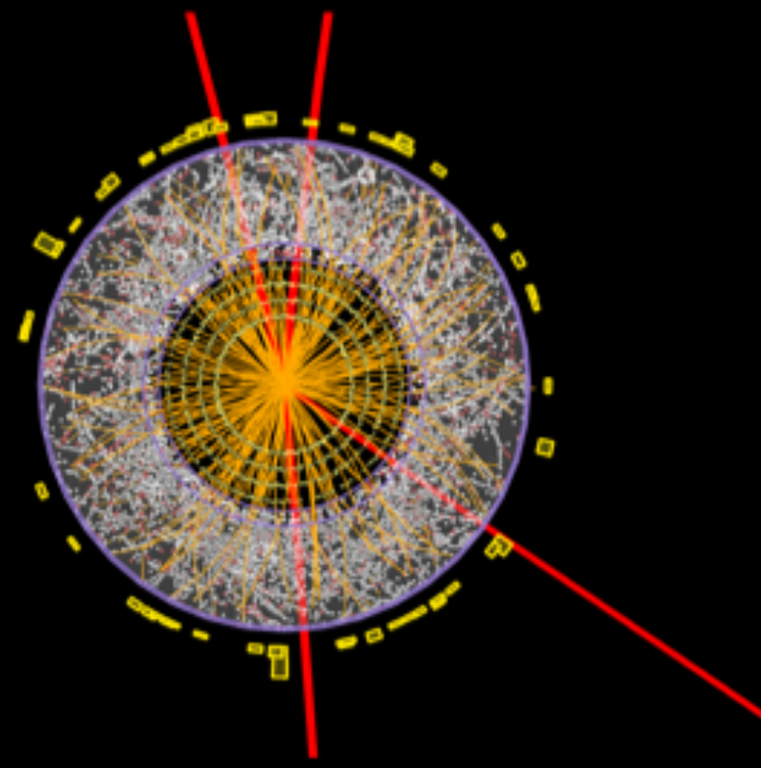
Workshop at the 31st Conference on Neural Information Processing Systems (NIPS)

December 8, 2017

An example

$H \to ZZ \to 4l$

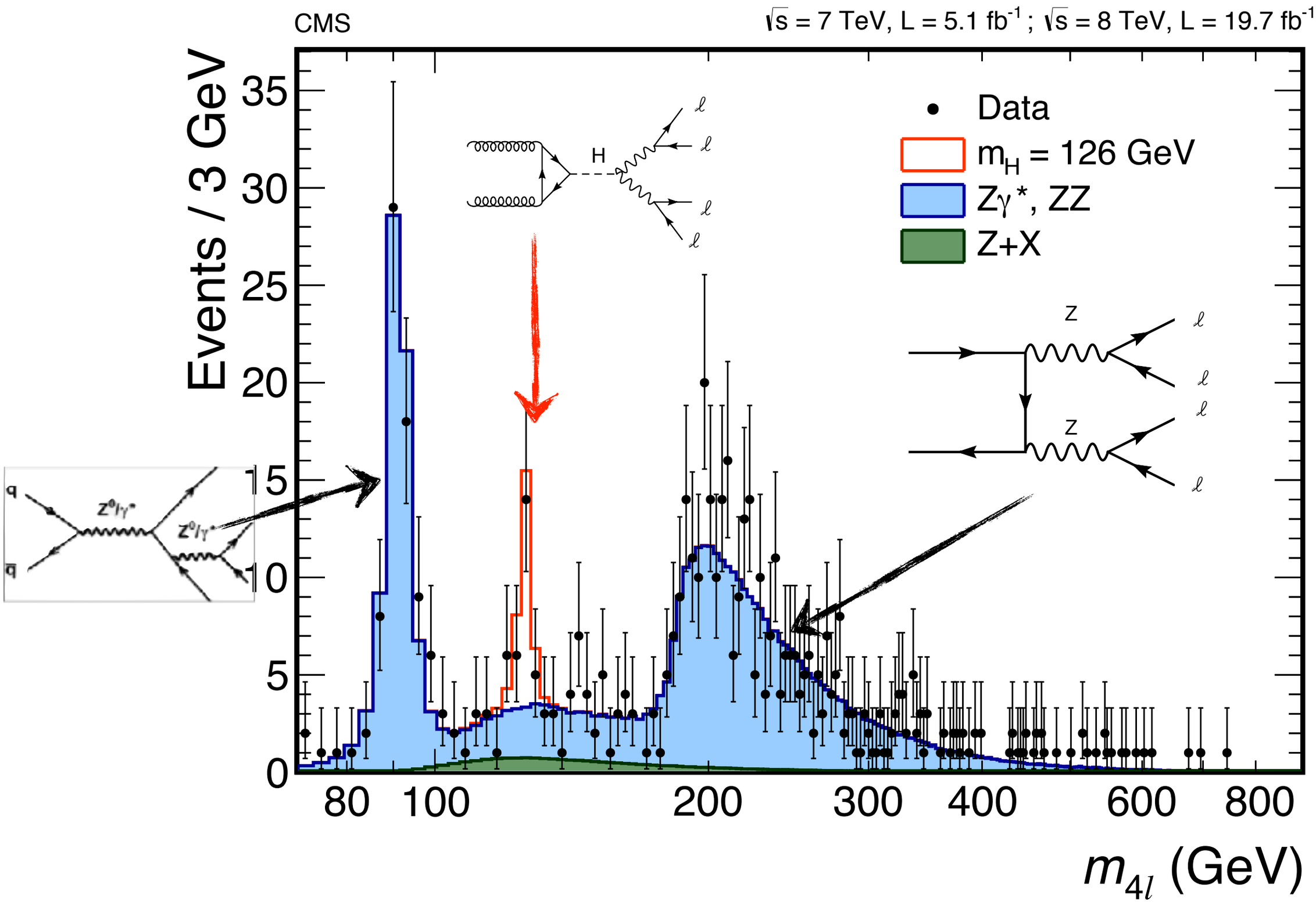




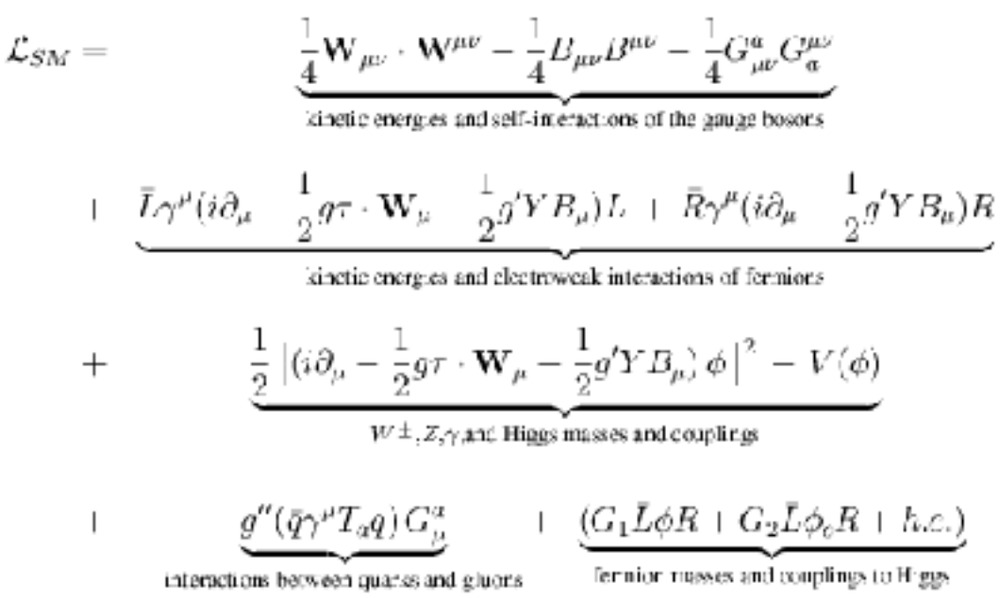
ATLAS EXPERIMENT
http://atlas.ch

Run: 204769
Event: 71902630
Date: 2012-06-10
Time: 13:24:31 CEST

PREDICTIONS FROM SIMULATION



The Forward Model



1) We begin with Quantum Field Theory

The Forward Model

$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$

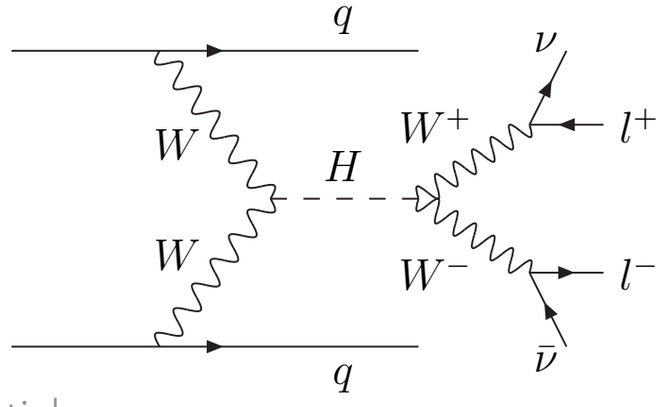
$$+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}}$$

$$+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\phi\right|^{2} - V(\phi)}_{W^{\pm}, Z, \gamma \text{ and Higgs masses and couplings}}$$

$$+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_{1}\bar{L}\phi R + G_{2}\bar{L}\phi_{c}R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$

1) We begin with Quantum Field Theory

2) Theory gives detailed prediction for high-energy collisions



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

The Forward Model

$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$

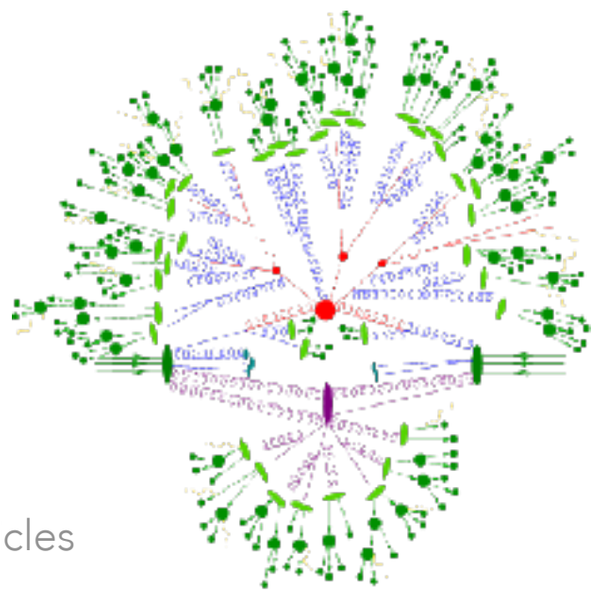
$$+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}}$$

$$+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma,\text{and Higgs masses and couplings}}$$

$$+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_{1}\bar{L}\phi R + G_{2}\bar{L}\phi_{c}R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$

1) We begin with Quantum Field Theory

2) Theory gives detailed prediction for high-energy collisions



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

The Forward Model

$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$

$$+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}}$$

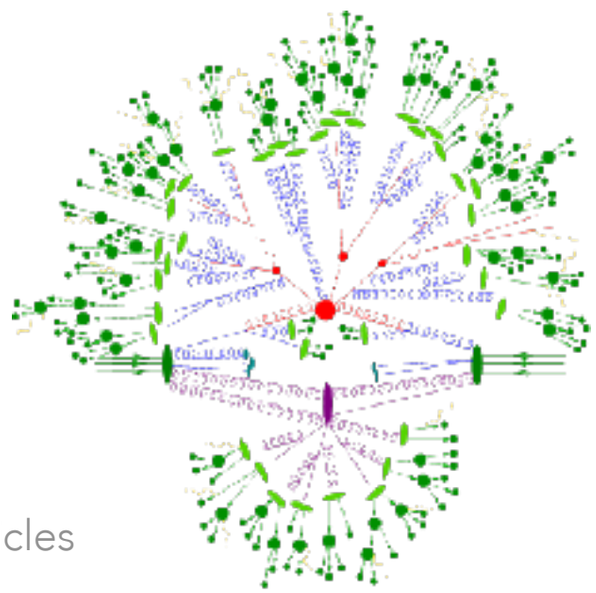
$$+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma \text{ and Higgs masses and couplings}}$$

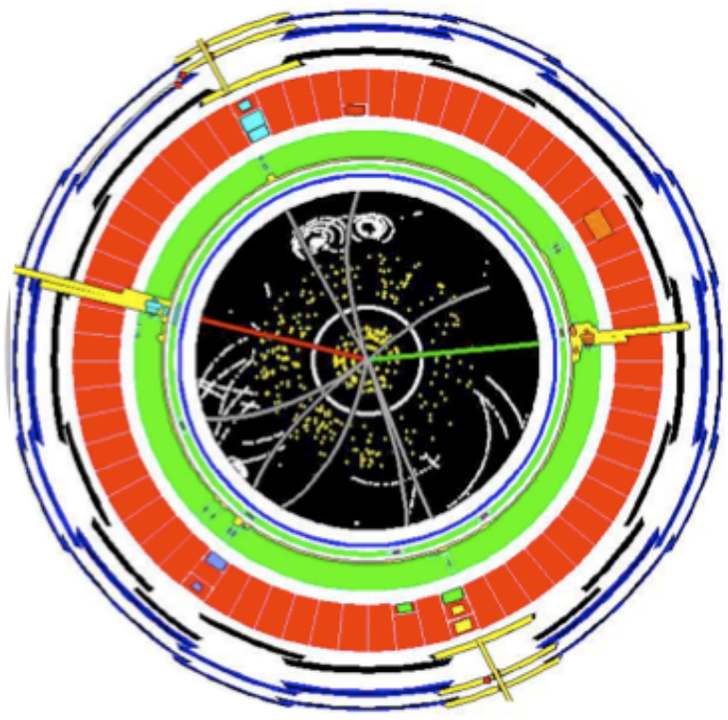
$$+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_{1}\bar{L}\phi R + G_{2}\bar{L}\phi_{c}R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$

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

The Forward Model

$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$

$$+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}}$$

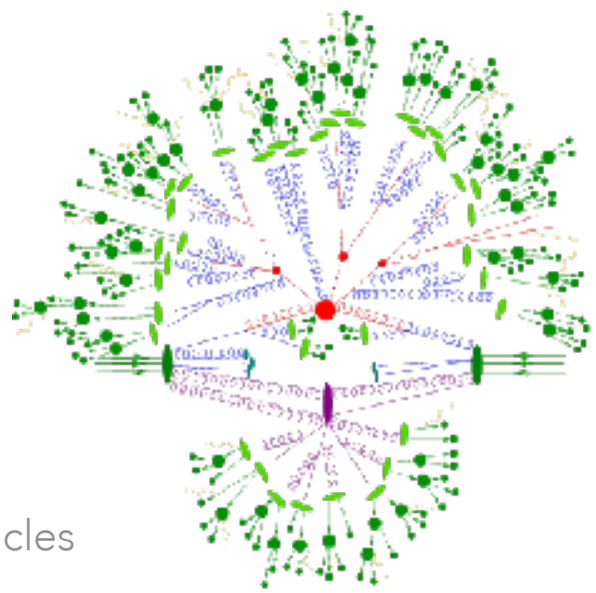
$$+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\phi\right|^{2} - V(\phi)}_{W^{\pm}, Z, \gamma \text{ and Higgs masses and couplings}}$$

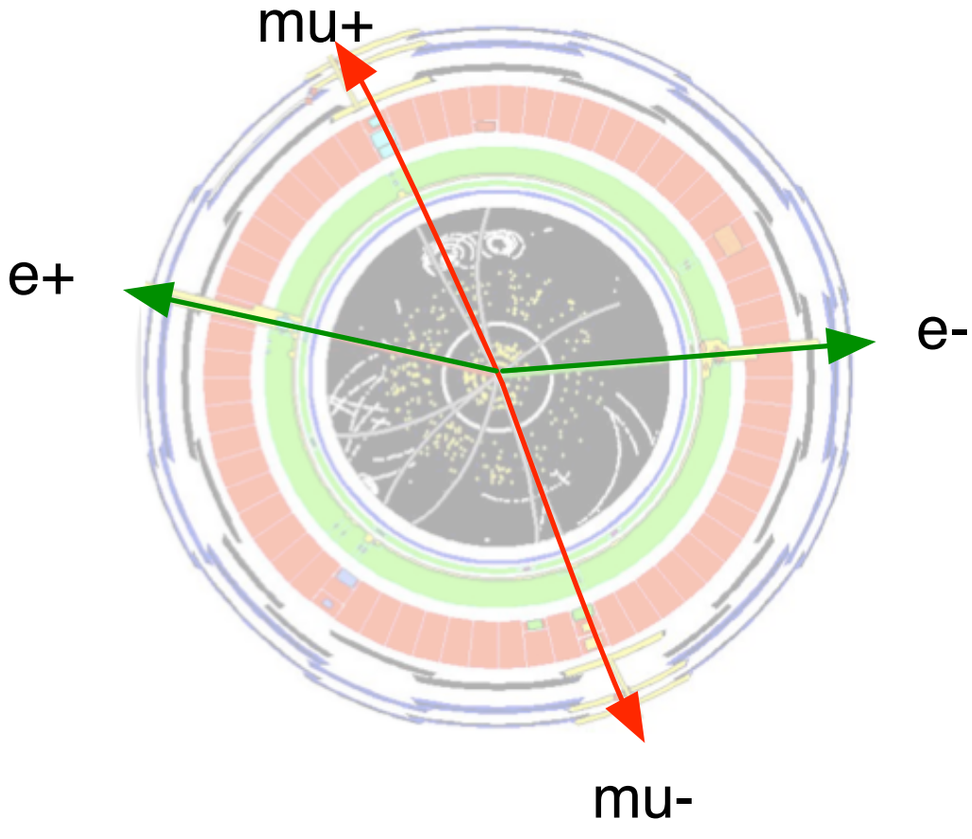
$$+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_1\bar{L}\phi R + G_2\bar{L}\phi_c R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$

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

4) Finally, we run particle identification and feature extraction algorithms on the simulated data as if they were from real collisions.

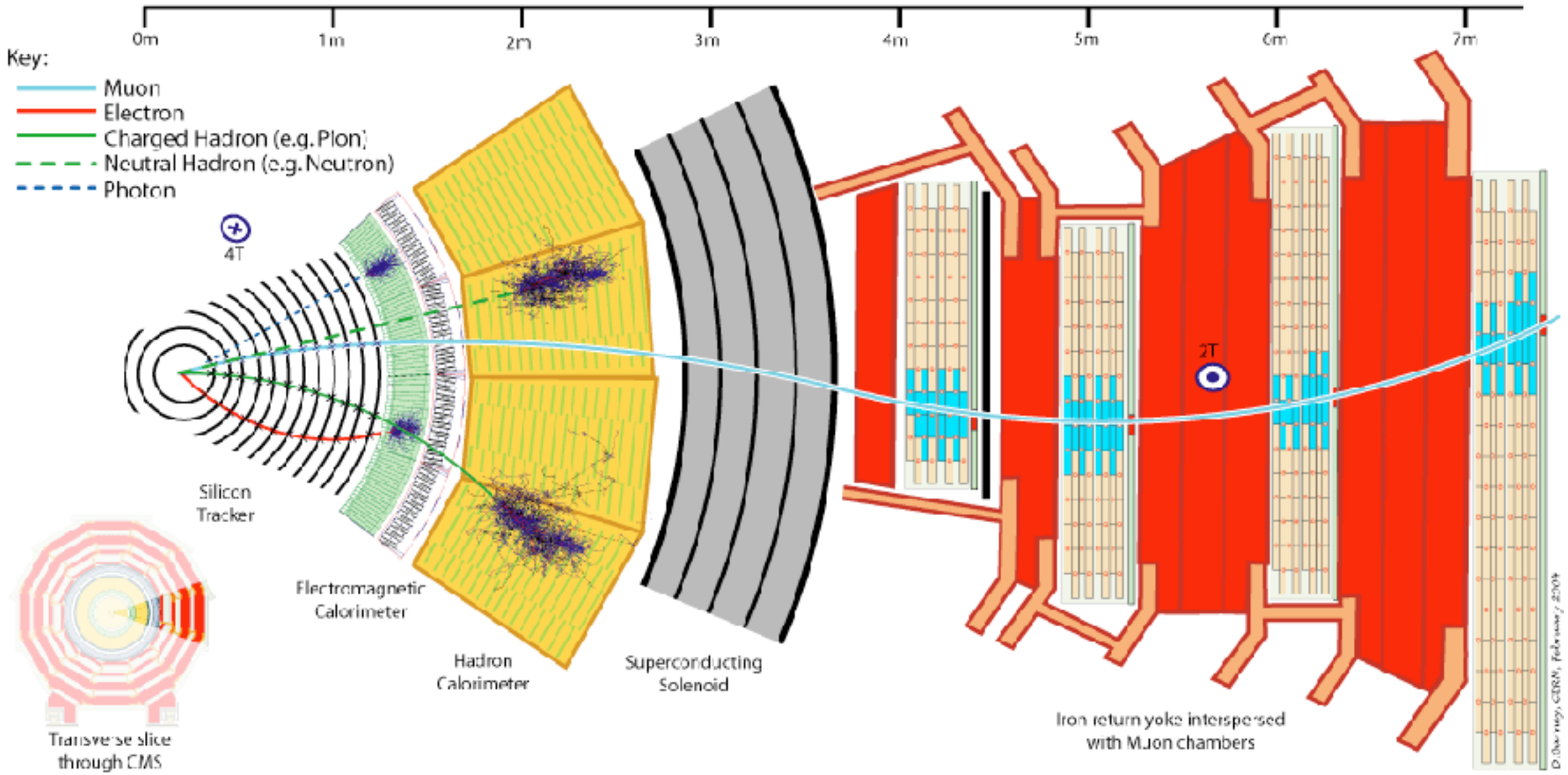
~10-30 features describe interesting part

DETECTOR SIMULATION

Conceptually: Prob(detector response | particles)

Implementation: Monte Carlo integration over micro-physics

Consequence: evaluation of the likelihood is intractable



DETECTOR SIMULATION

Conceptually: Prob(detector response | particles)

Implementation: Monte Carlo integration over micro-physics

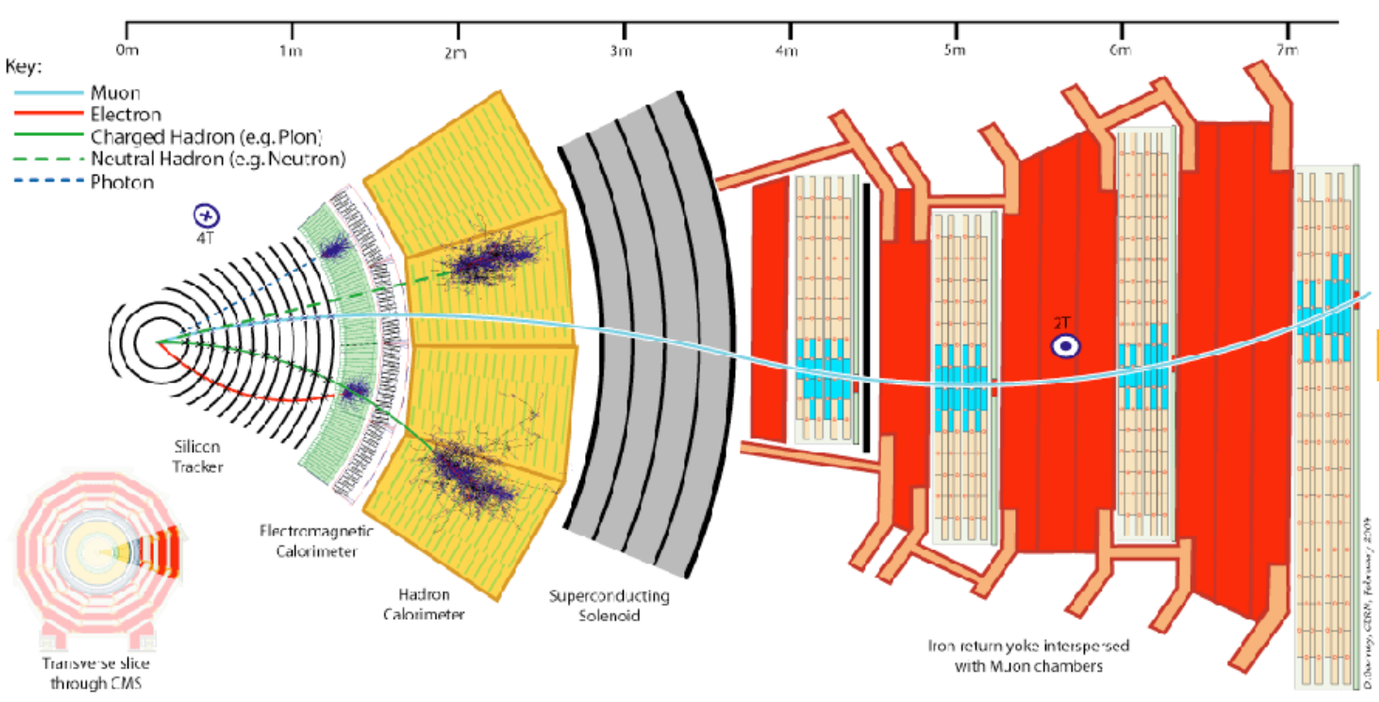
Consequence: evaluation of the likelihood is intractable

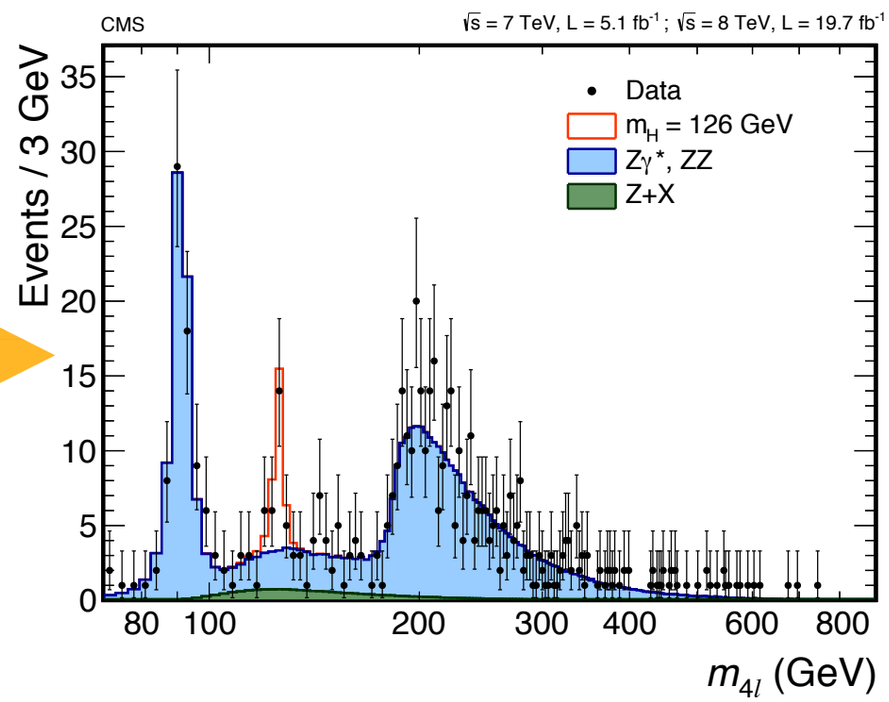
This motivates a new class of algorithms for what is called **likelihood-free inference**, which only require ability to generate samples from the simulation in the "forward mode"

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 $p(x|\theta)$ **approximated** using histograms (univariate density estimation)

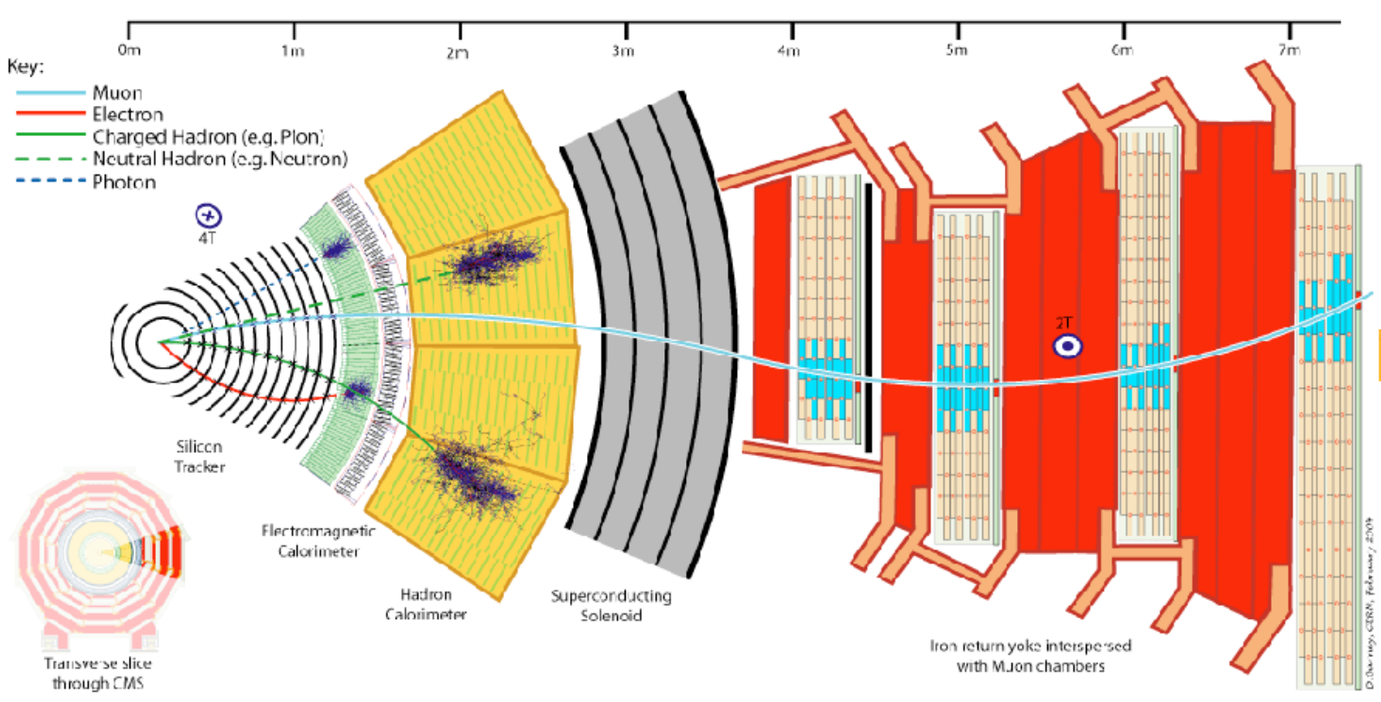


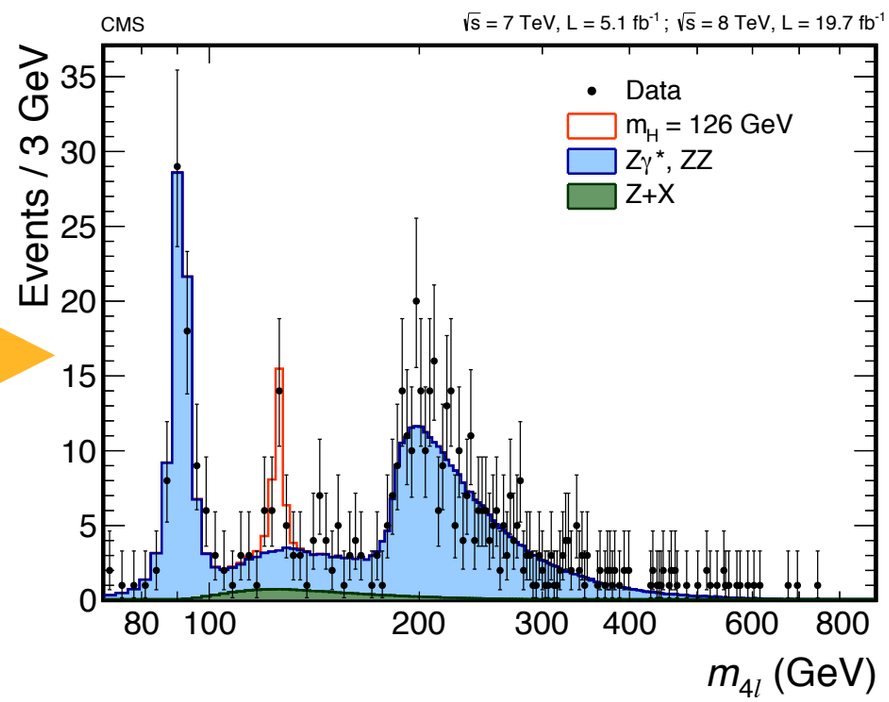


10^8 SENSORS → 1 REAL-VALUED QUANTITY

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 $p(x|\theta)$ **approximated** using histograms (univariate density estimation)



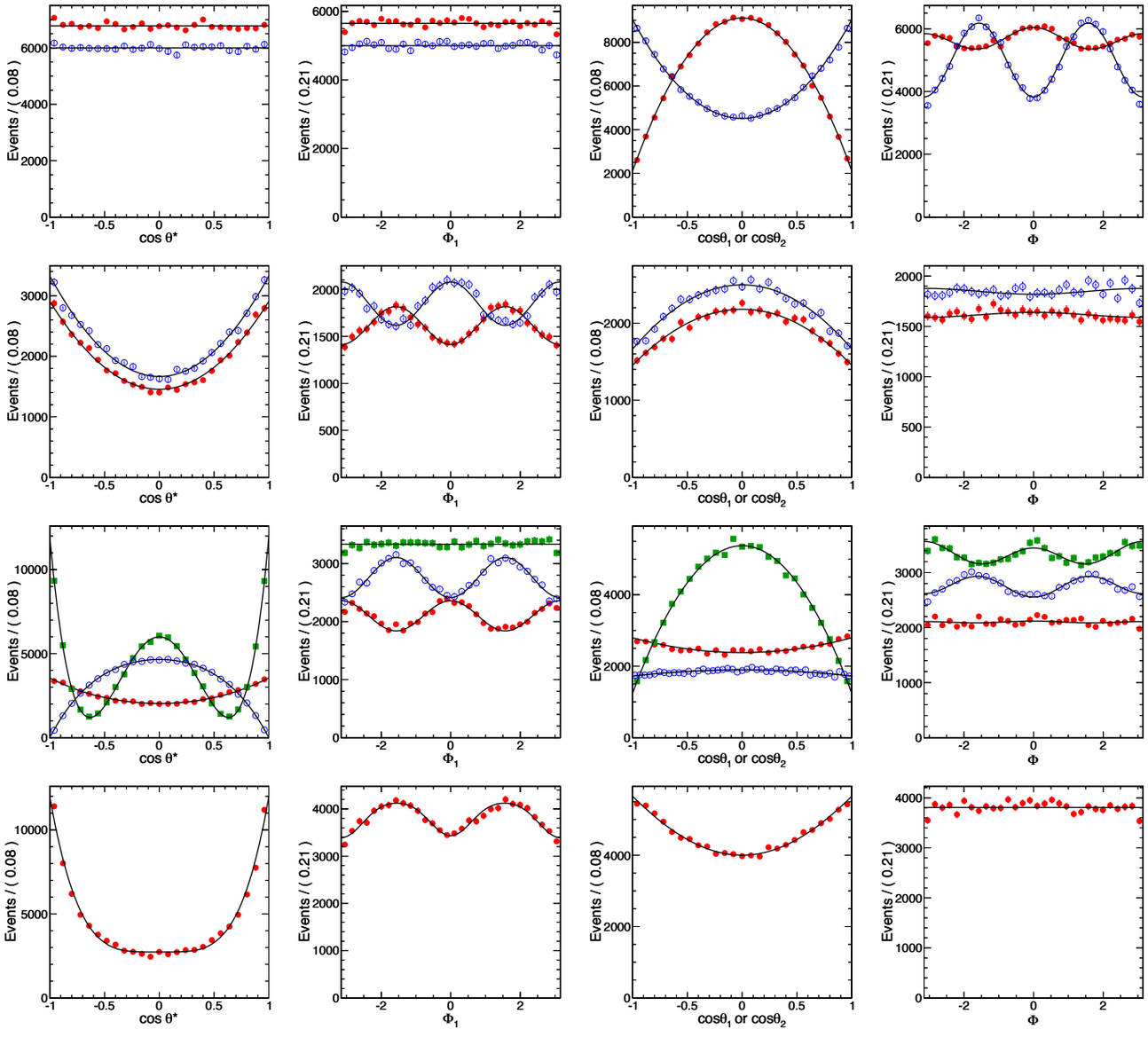


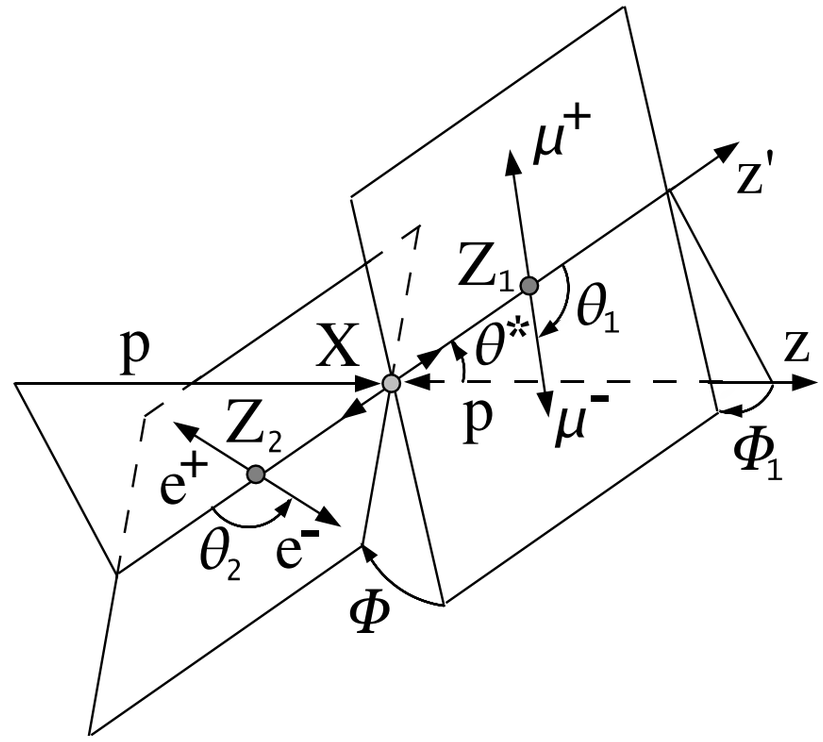
This doesn't scale if x is high dimensional!

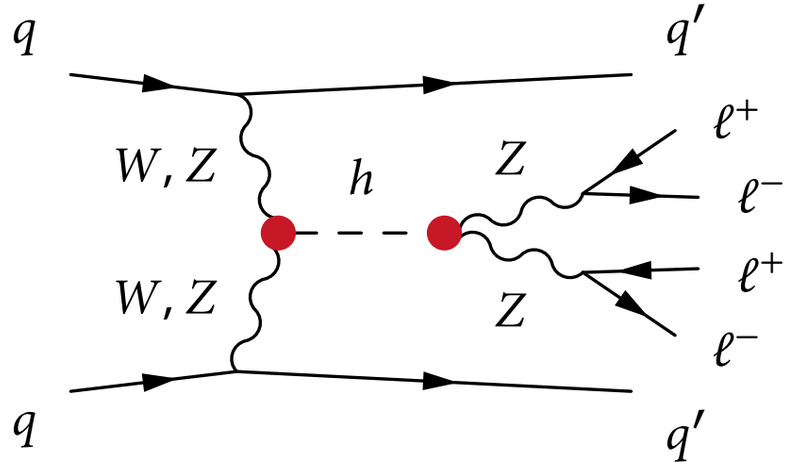
HIGH DIMENSIONAL EXAMPLE

For instance, 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

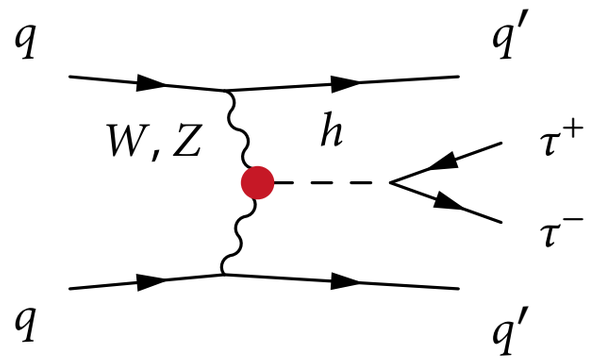






HIGGS EFT

"Better Higgs Measurements Through Information Geometry" [arXiv:1612.05261]



- Theory language: dimension-6 operators of SM EFT, $\mathcal{L} \supset \sum_i \frac{f_i}{\Lambda^2}\mathcal{O}_i$

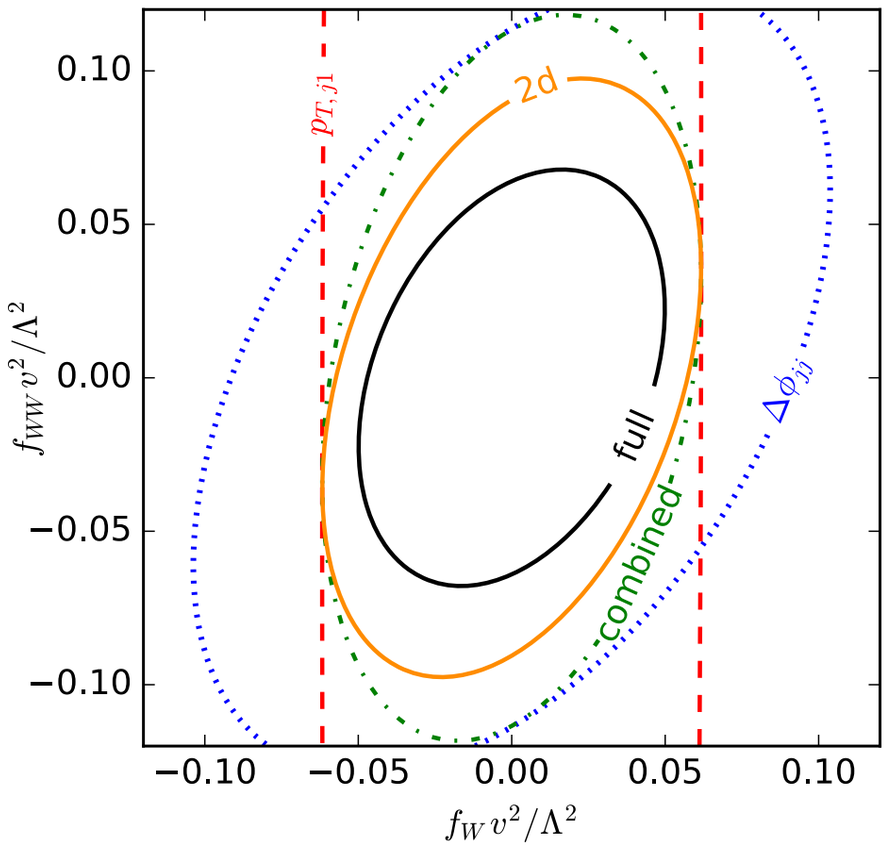
  [W. Buchmuller, D. Wyler 85; K. Hagiwara, S. Ishihara, S. R. Szalapski, D. Zeppenfeld 93; B. Grzadkowski, M. Iskrzynski, M. Misiak, J. Rosiek 1008.4884; …]

  - Total rate: $\mathcal{O}_{\phi,2} = \frac{1}{2}\,\partial^\mu(\phi^\dagger\phi)\,\partial_\mu(\phi^\dagger\phi)$
  - New kinematic structures:

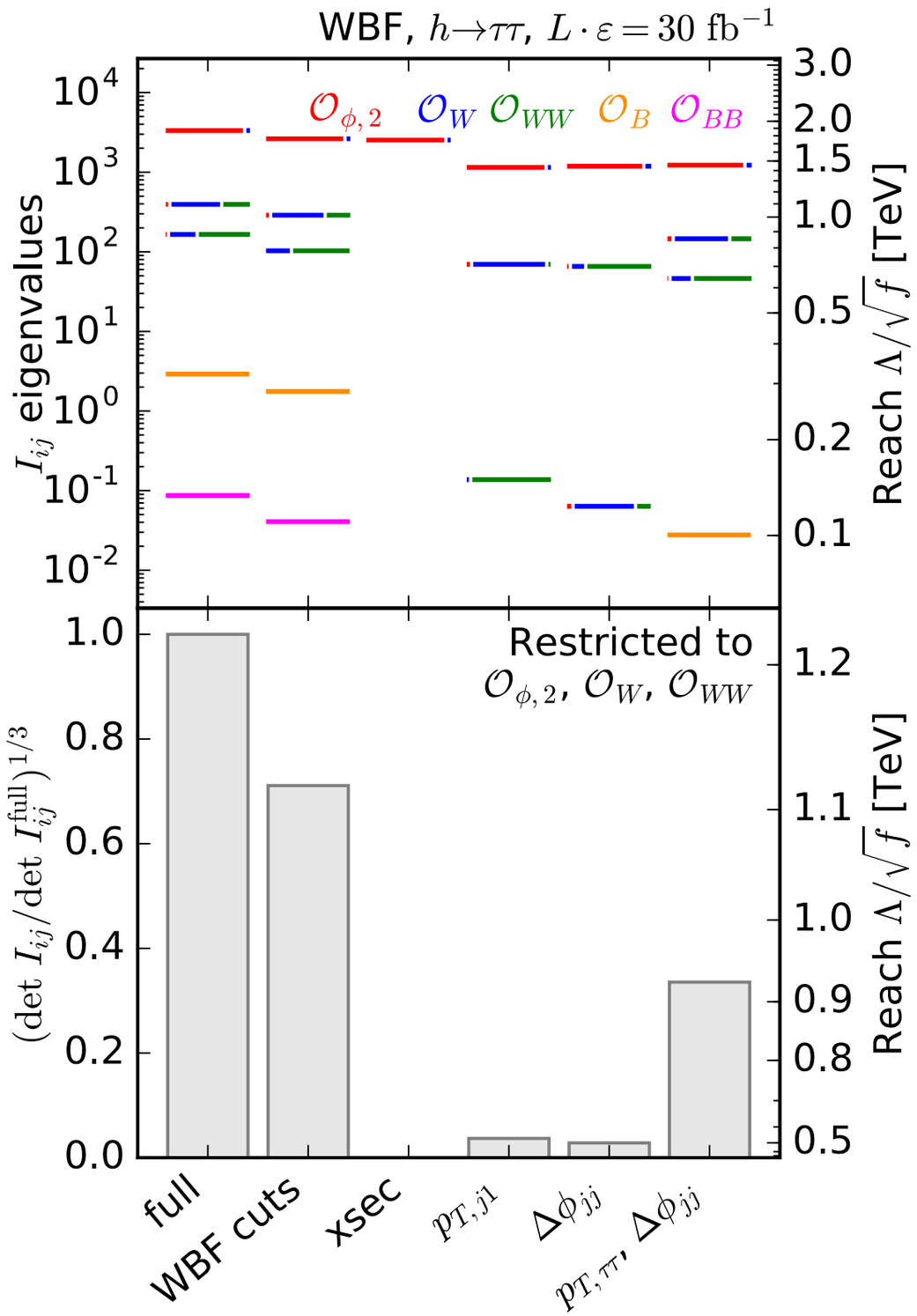
$$\mathcal{O}_B = \mathrm{i}\frac{g}{2}\,(D^\mu\phi^\dagger)(D^\nu\phi)\,B_{\mu\nu} \qquad \mathcal{O}_W = \mathrm{i}\frac{g}{2}\,(D^\mu\phi)^\dagger\sigma^k(D^\nu\phi)\,W^k_{\mu\nu}$$

$$\mathcal{O}_{BB} = -\frac{g'^2}{4}\,(\phi^\dagger\phi)\,B_{\mu\nu}\,B^{\mu\nu} \qquad \mathcal{O}_{WW} = -\frac{g^2}{4}\,(\phi^\dagger\phi)\,W^k_{\mu\nu}\,W^{\mu\nu\,k}$$

  - CP violation: $\mathcal{O}_{W\widetilde{W}} = -\frac{g^2}{4}\,(\phi^\dagger\phi)\,W^k_{\mu\nu}\,\widetilde{W}^{\mu\nu\,k}$
  - Others strongly constrained by EWPD or redundant



Equivalent to 3x more data!



A COMMON THEME

ABC

resources on approximate Bayesian computational methods

Search

Home

Home

This website keeps track of developments in approximate Bayesian computation (ABC) (a.k.a. likelihood-free), a class of computational statistical methods for Bayesian inference under intractable likelihoods. The site is meant to be a resource both for biologists and statisticians who want to learn more about ABC and related methods. Recent publications are under Publications 2012. A comprehensive list of publications can be found under Literature. If you are unfamiliar with ABC methods see the Introduction. Navigate using the menu to learn more.

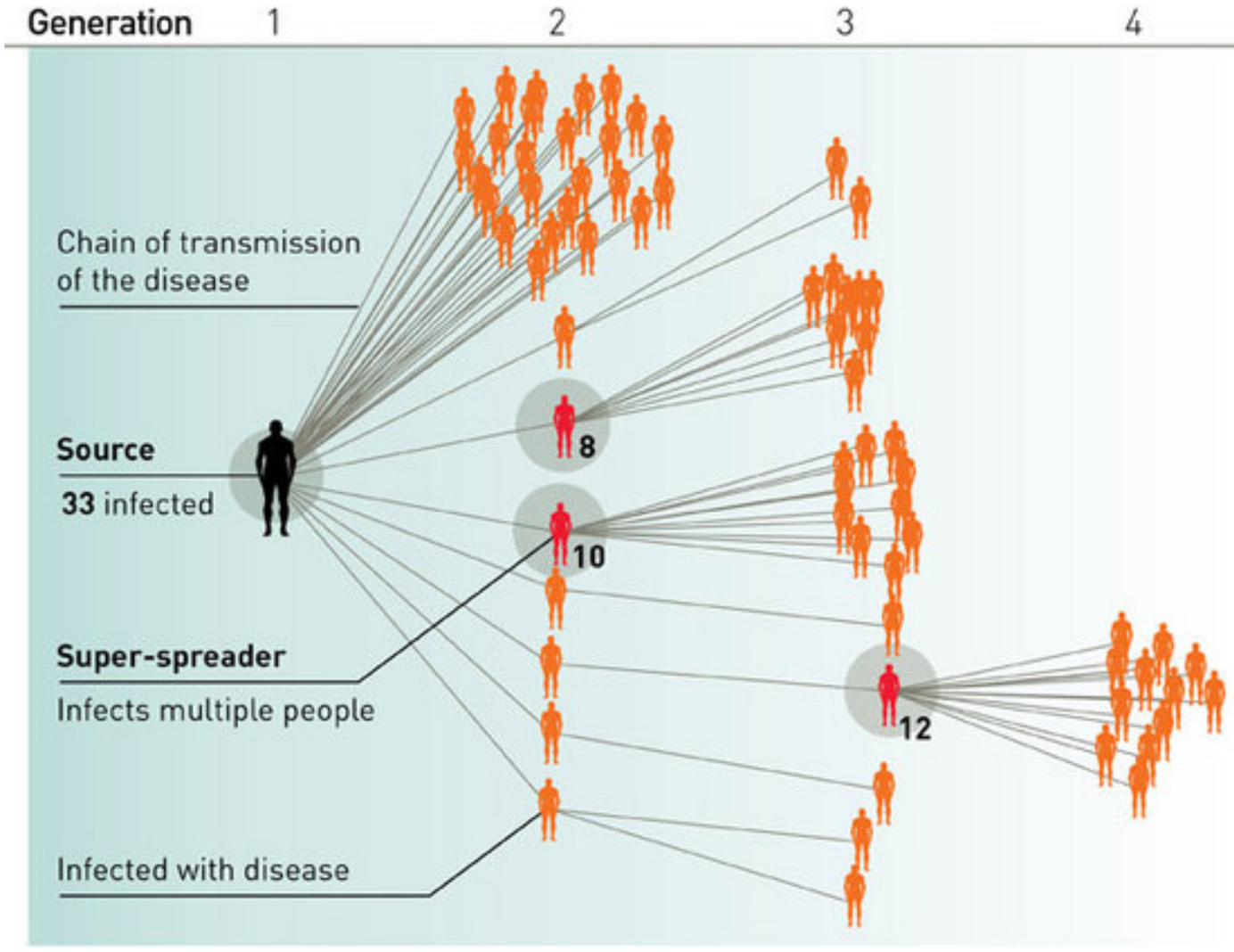
ABC in Montreal | ABC in Montreal (2014)

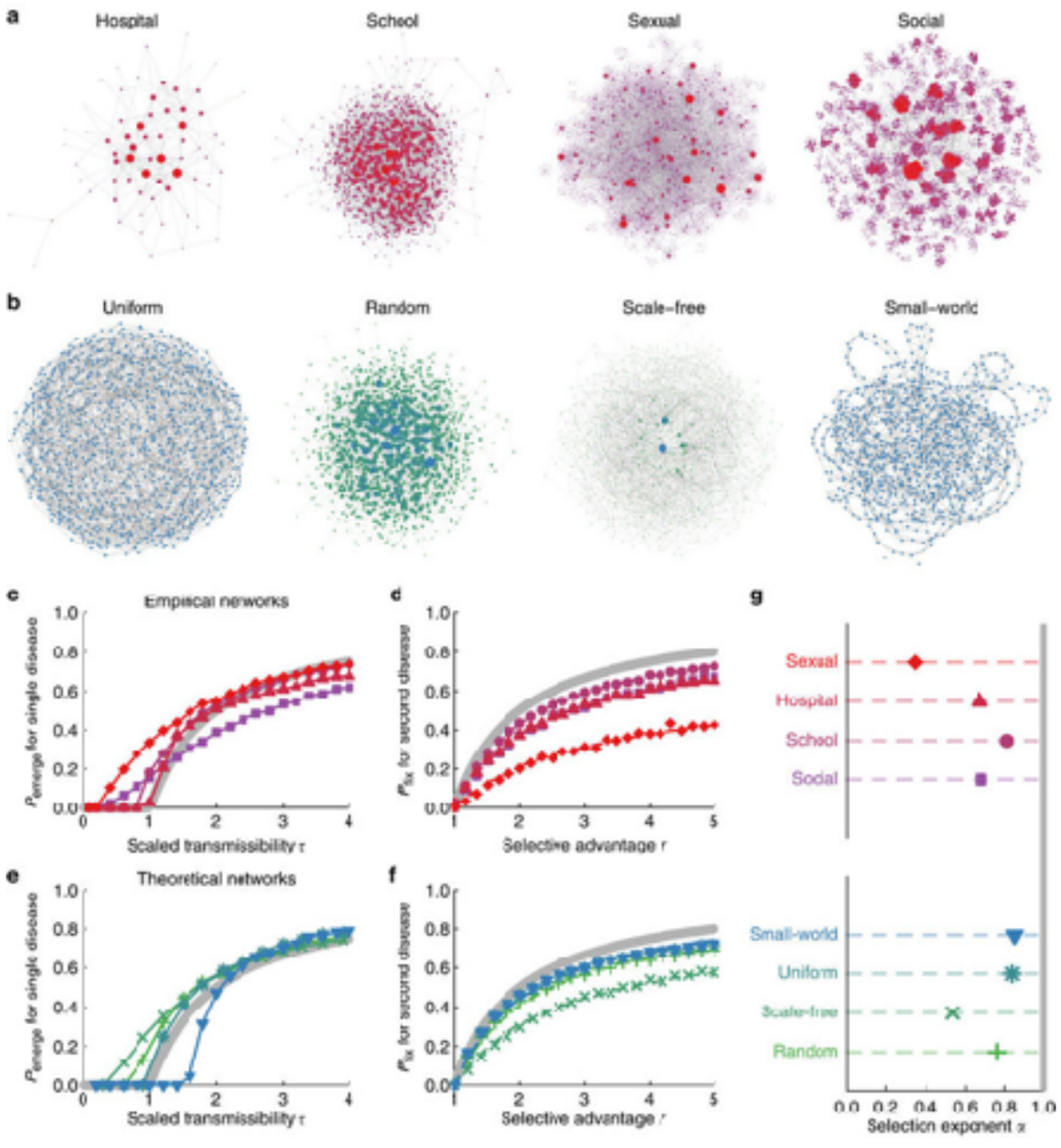
ABC in Montreal

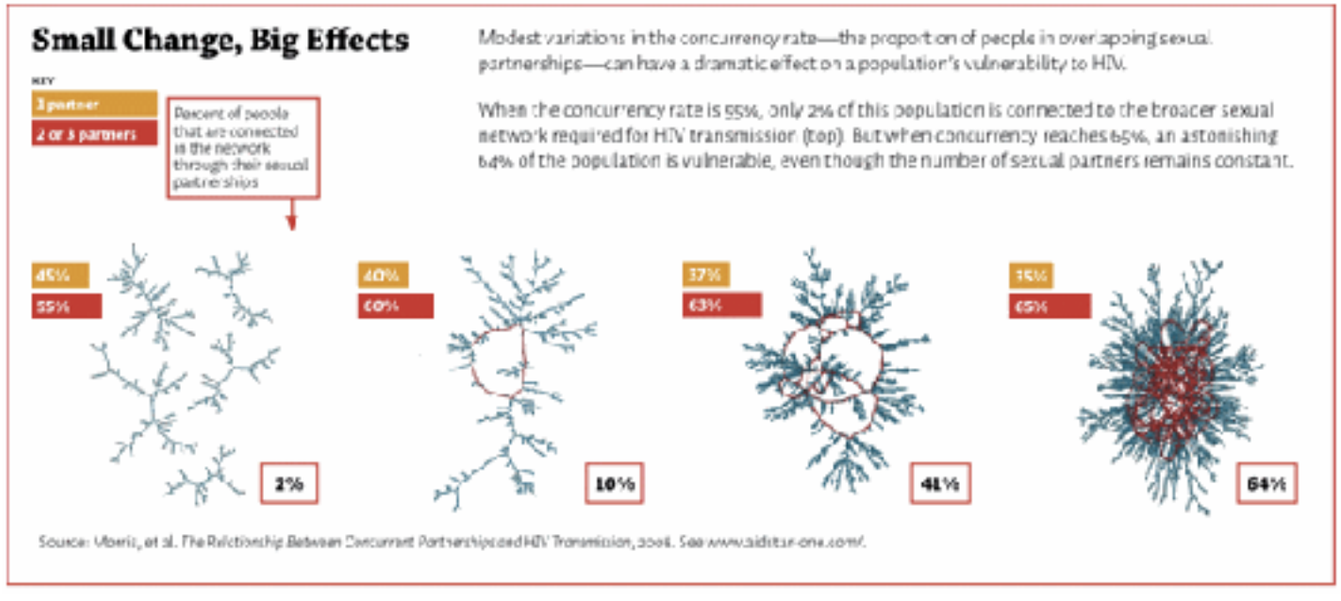
Approximate Bayesian computation (ABC) or likelihood-free (LF) methods have developed mostly beyond the radar of the machine learning community, but are important tools for a large and diverse segment of the scientific community. This is particularly true for systems and population biology, computational neuroscience, computer vision, healthcare sciences, but also many others.

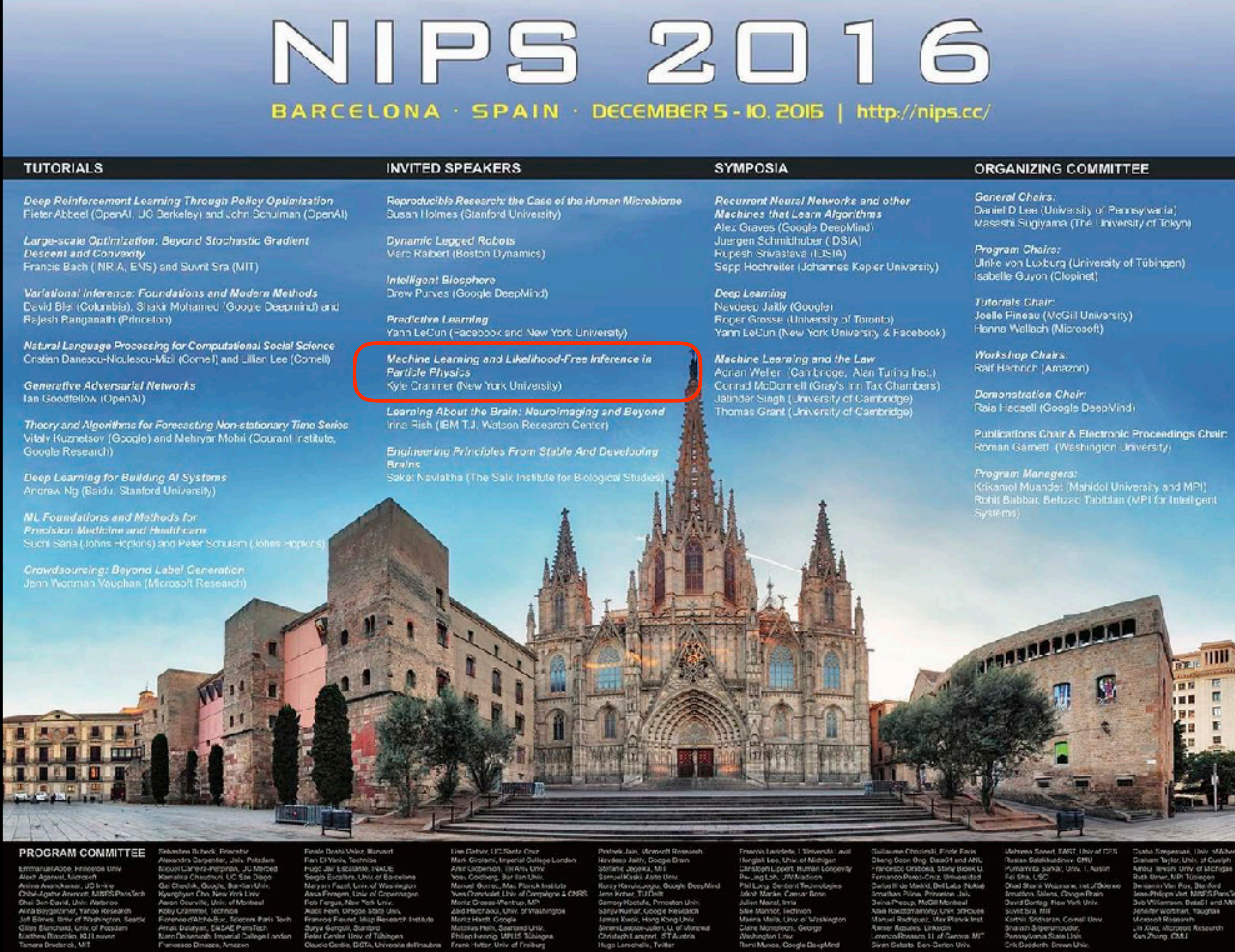
Interaction between the ABC and machine learning community has recently started and contributed to important advances. In general, however, there is still significant room for more intense interaction and collaboration. Our workshop aims at being a place for this to happen.

EPIDEMIOLOGY & POPULATION GENETICS








NIPS 2016
BARCELONA · SPAIN · DECEMBER 5 - 10, 2016 | http://nips.cc/
TUTORIALS
Deep Reinforcement Learning Through Policy Optimization
Pieter Abbeel (OpenAI, UC Berkeley) and John Schulman (OpenAI)
Large-scale Optimization: Beyond Stochastic Gradient Descent and Convexity
Francis Bach (INRIA, ENS) and Suvrit Sra (MIT)
Variational Inference: Foundations and Modern Methods
David Blei (Columbia), Shakir Mohamed (Google Deepmind) and Rajesh Ranganath (Princeton)
Natural Language Processing for Computational Social Science
Cristian Danescu-Niculescu-Mizil (Cornell) and Lillian Lee (Cornell)
Generative Adversarial Networks
Ian Goodfellow (OpenAI)
Theory and Algorithms for Forecasting Non-stationary Time Series
Vitaly Kuznetsov (Google) and Mehryar Mohri (Courant Institute, Google Research)
Deep Learning for Building AI Systems
Andrew Ng (Baidu, Stanford University)
ML Foundations and Methods for Precision Medicine and Healthcare
Suchi Saria (Johns Hopkins) and Peter Schulam (Johns Hopkins)
Crowdsourcing: Beyond Label Generation
Jenn Wortman Vaughan (Microsoft Research)
INVITED SPEAKERS
Reproducible Research: the Case of the Human Microbiome
Susan Holmes (Stanford University)
Dynamic Legged Robots
Marc Raibert (Boston Dynamics)
Intelligent Biosphere
Drew Purves (Google DeepMind)
Predictive Learning
Yann LeCun (Facebook and New York University)
Machine Learning and Likelihood-Free Inference in Particle Physics
Kyle Cranmer (New York University)
Learning About the Brain: Neuroimaging and Beyond
Irina Rish (IBM T.J. Watson Research Center)
Engineering Principles From Stable And Developing Brains
Saket Navlakha (The Salk Institute for Biological Studies)
SYMPOSIA
Recurrent Neural Networks and other Machines that Learn Algorithms
Alex Graves (Google DeepMind)
Juergen Schmidhuber (IDSIA)
Rupesh Srivastava (IDSIA)
Sepp Hochreiter (Johannes Kepler University)
Deep Learning
Navdeep Jaitly (Google)
Roger Grosse (University of Toronto)
Yann LeCun (New York University & Facebook)
Machine Learning and the Law
Adrian Weller (Cambridge, Alan Turing Inst.)
Conrad McDonnell (Gray's Inn Tax Chambers)
Jatinder Singh (University of Cambridge)
Thomas Grant (University of Cambridge)
ORGANIZING COMMITTEE
General Chairs:
Daniel D Lee (University of Pennsylvania)
Masashi Sugiyama (The University of Tokyo)
Program Chairs:
Ulrike von Luxburg (University of Tübingen)
Isabelle Guyon (Clopinet)
Tutorials Chair:
Joelle Pineau (McGill University)
Hanna Wallach (Microsoft)
Workshop Chairs:
Ralf Herbrich (Amazon)
Demonstration Chair:
Raia Hadsell (Google DeepMind)
Publications Chair & Electronic Proceedings Chair:
Roman Garnett (Washington University)
Program Managers:
Krikamol Muandet (Mahidol University and MPI)
Rohit Babbar, Behzad Tabibian (MPI for Intelligent Systems)
PROGRAM COMMITTEE

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:

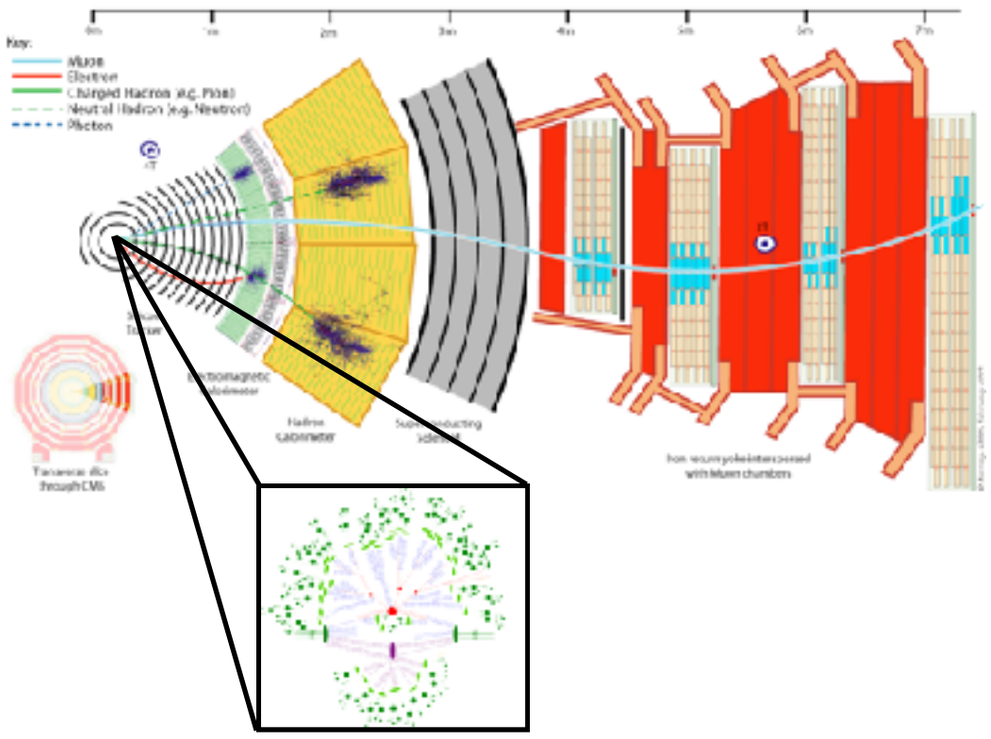
- **Generative adversarial networks** (a NIPS 2016 workshop) are implicit models with an adversarial training scheme.
- Recent advances in **variational inference** (a NIPS 2015 and 2016 workshop) have leveraged implicit models for more accurate approximations.
- **Approximate Bayesian computation** (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.

TWO APPROACHES

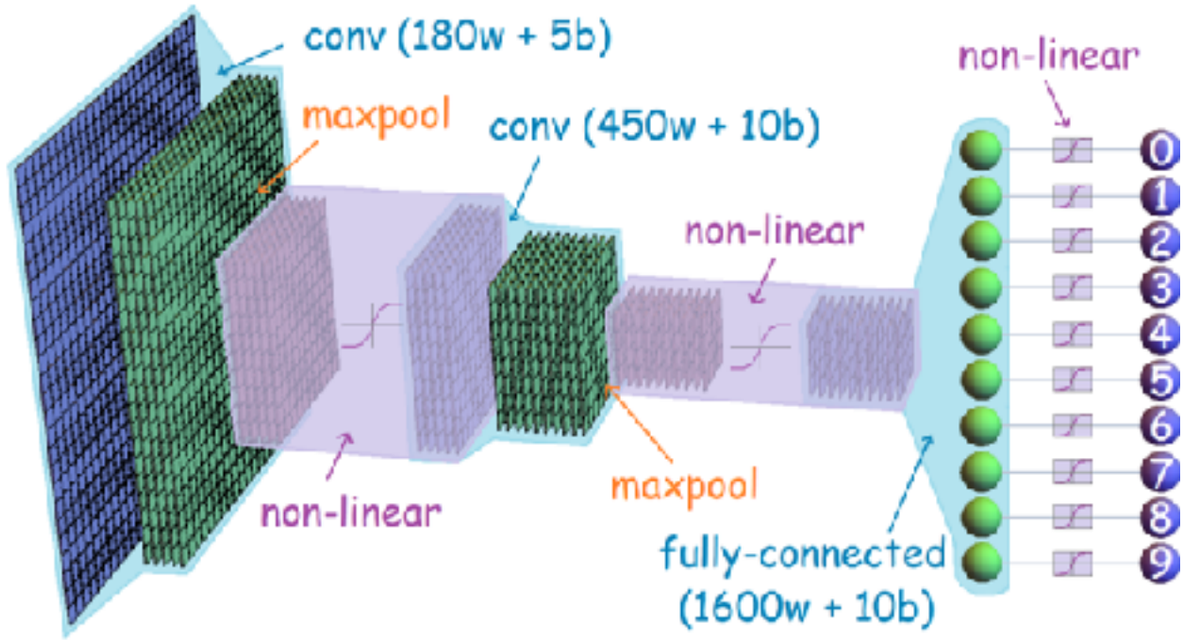
Use simulator

(much more efficiently)



Learn simulator

(with deep learning)



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

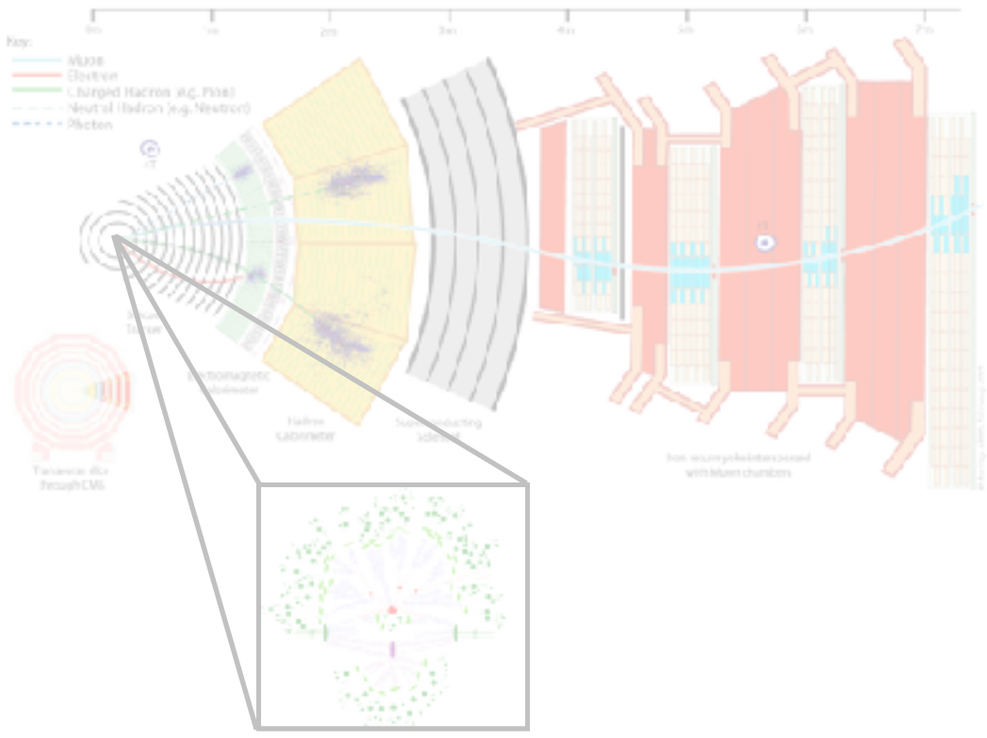
- Generative Adversarial Networks (GANs), Variational Auto-Encoders (VAE)
- Likelihood ratio from classifiers (CARL)
- Autogregressive models, Normalizing Flows

[image credit: A.P. Goucher]

TWO APPROACHES

Use simulator

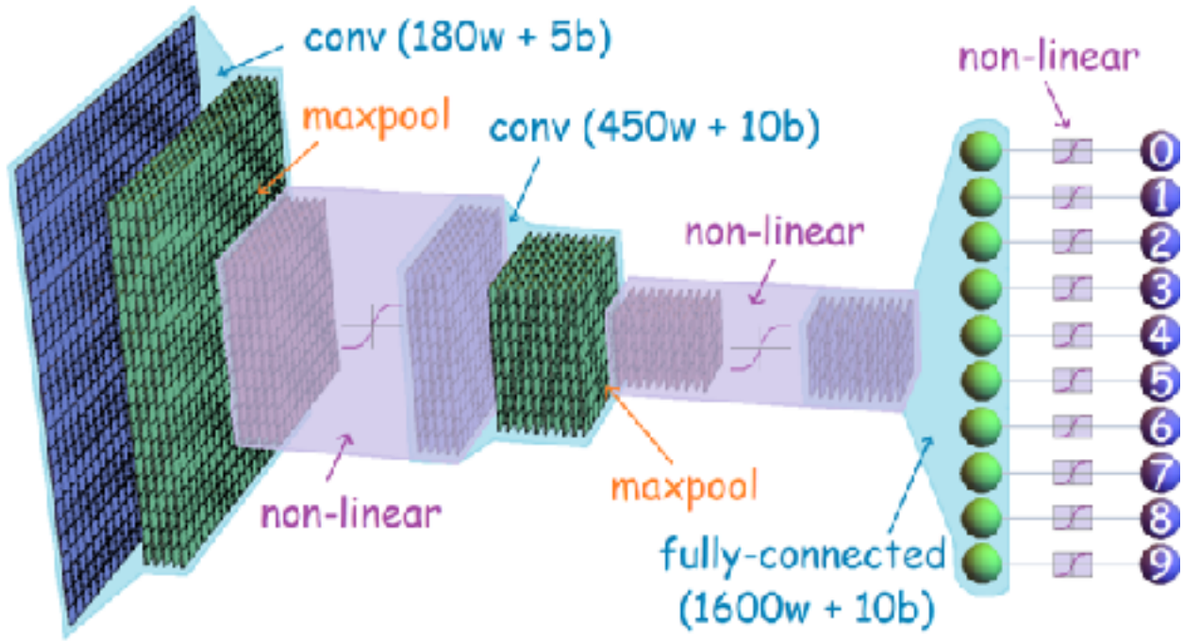
(much more efficiently)



- 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)
- Autogregressive models, Normalizing Flows

[image credit: A.P. Goucher]

THE PLAYERS

forward modeling
generation
simulation

PREDICTION

θ
parameters of interest

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

z
latent variables
Monte Carlo truth

x
observed data
simulated data

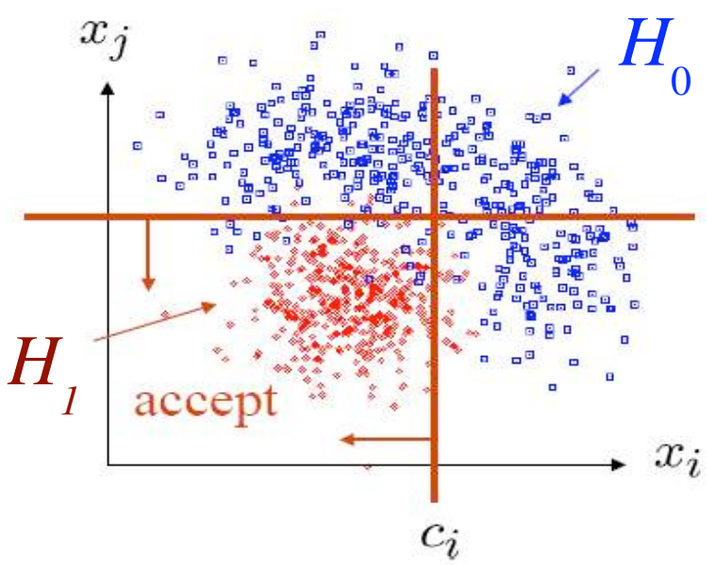
ν
nuisance parameters

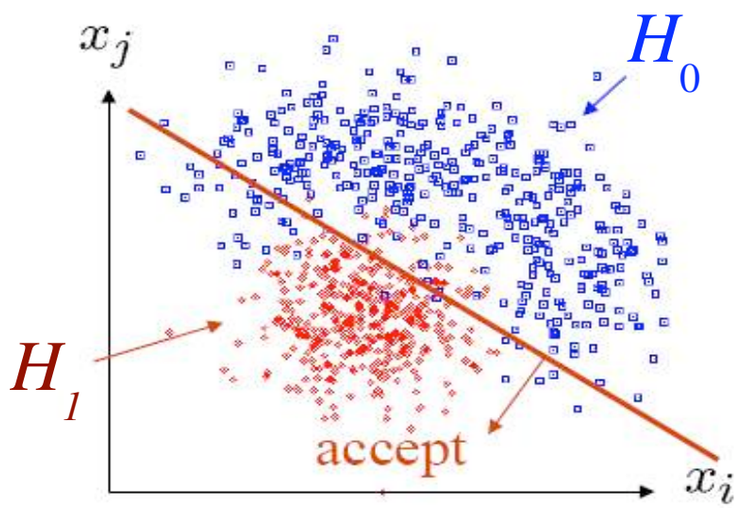
INFERENCE

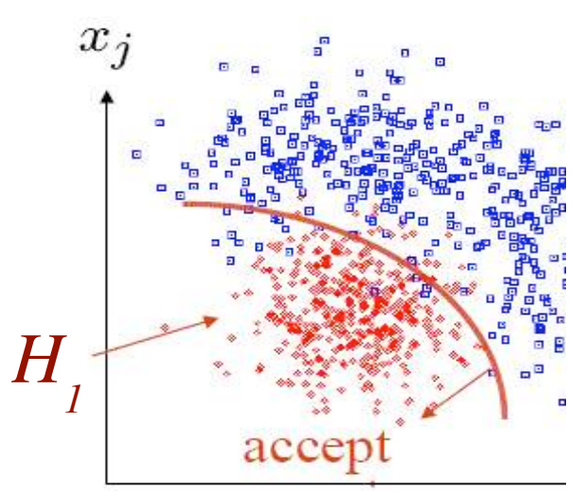
inverse problem
measurement
parameter estimation

HYPOTHESIS TESTING / CLASSIFICATION

If the data are high-dimensional, it's not obvious how to draw the boundary between accept/reject the null hypothesis

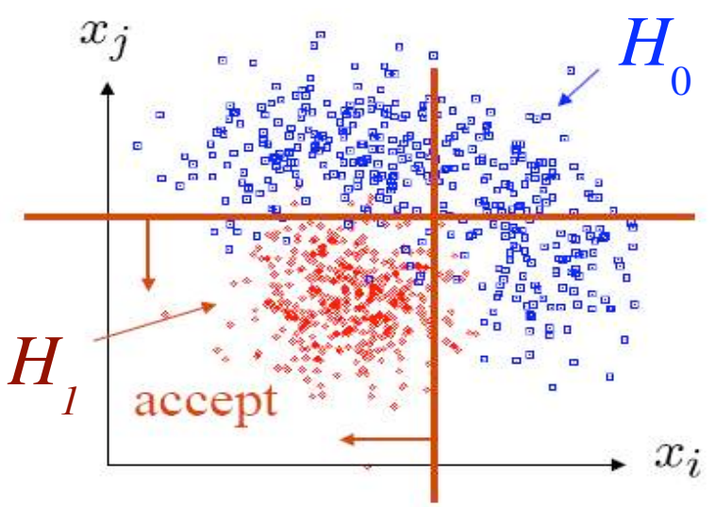


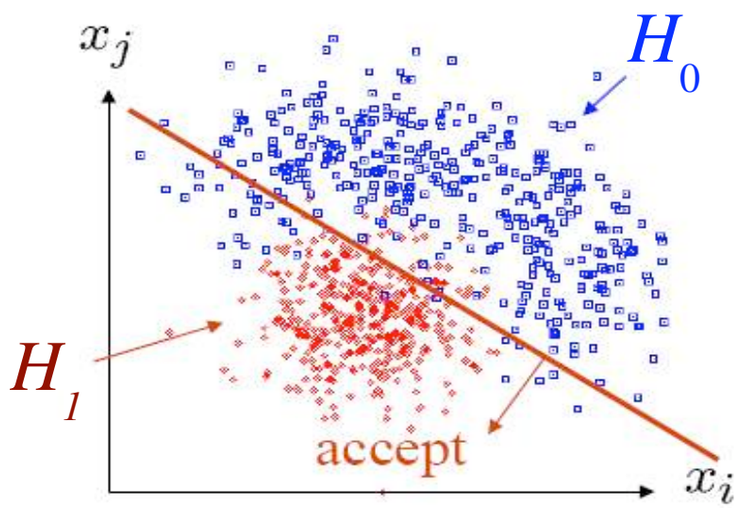


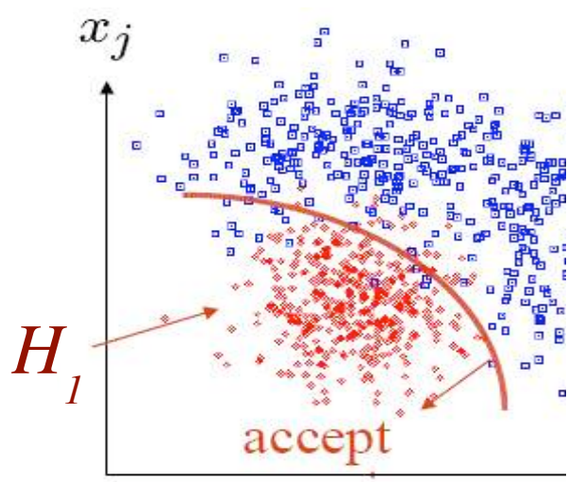


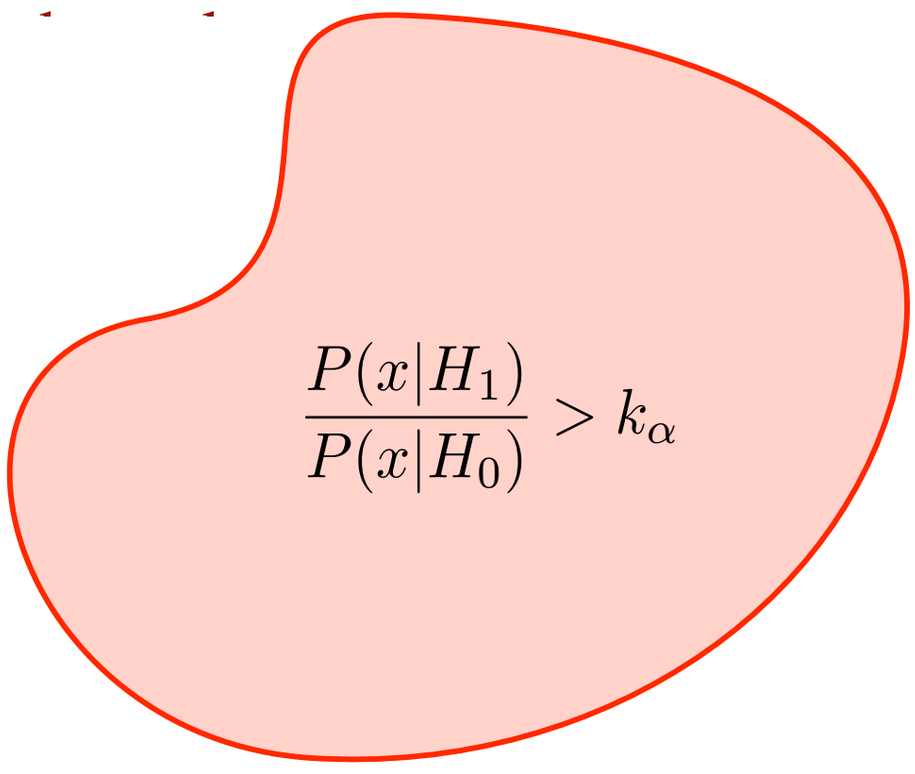
HYPOTHESIS TESTING / CLASSIFICATION

If the data are high-dimensional, it's not obvious how to draw the boundary between accept/reject the null hypothesis









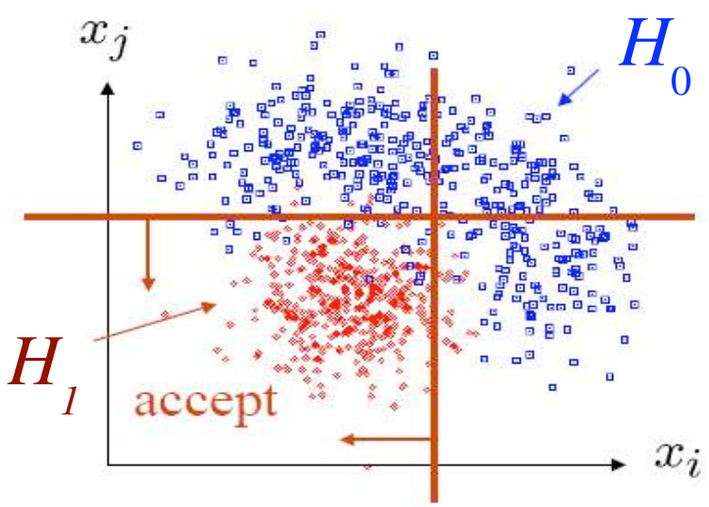
Neyman-Pearson Lemma:

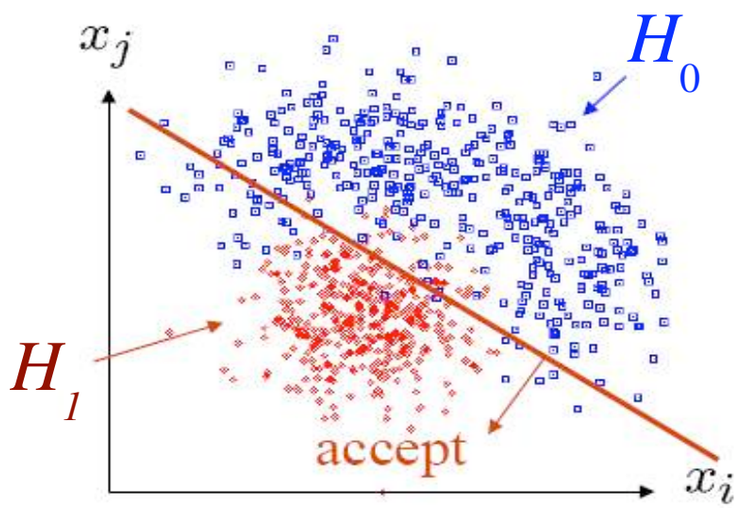
optimal classifier/hypothesis test is a contour of likelihood ratio

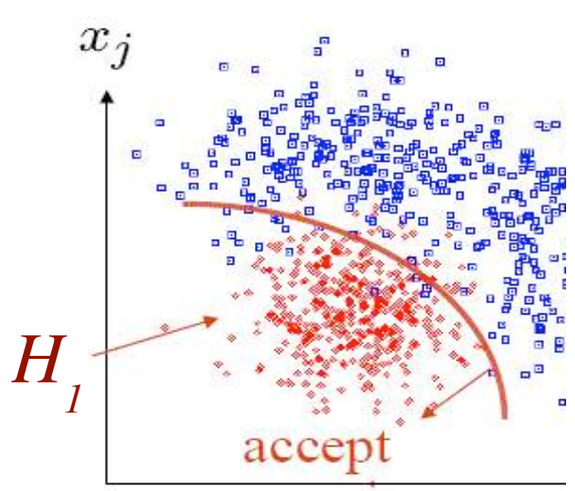
But, If I don't know $P(x|H_1)$ and $P(x|H_0)$ I can't evaluate this likelihood ratio!

HYPOTHESIS TESTING / CLASSIFICATION

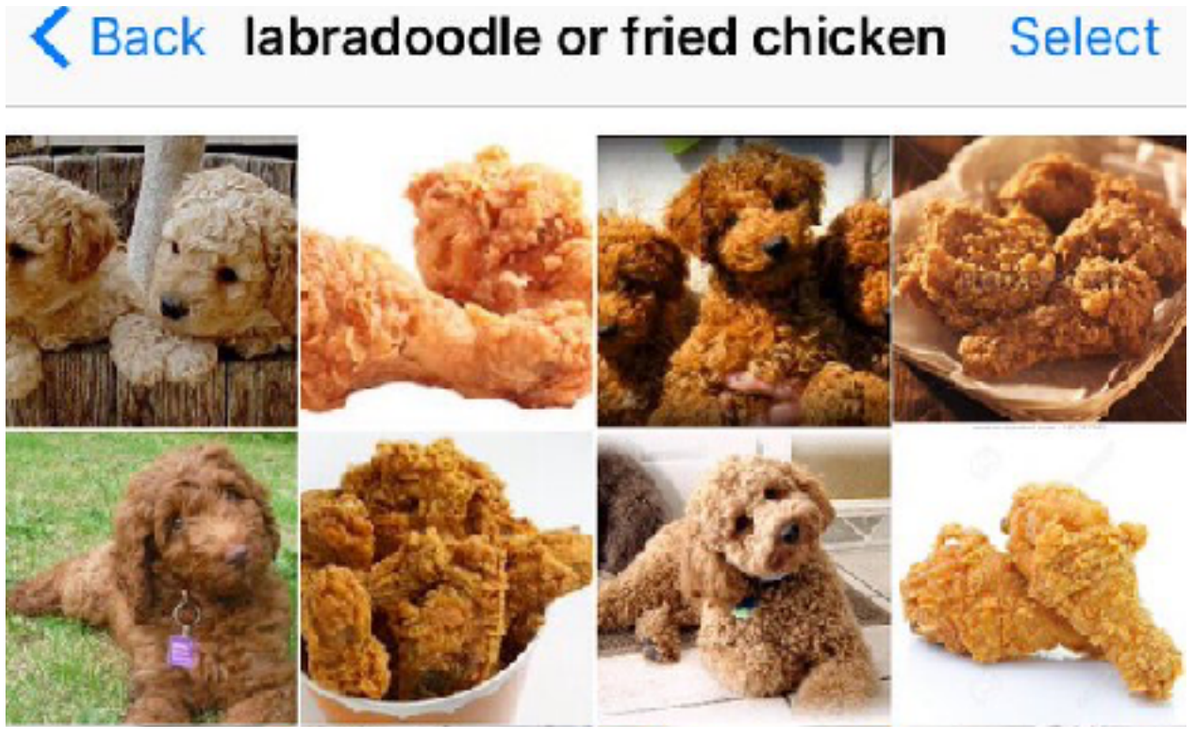
If the data are high-dimensional, it's not obvious how to draw the boundary between accept/reject the null hypothesis

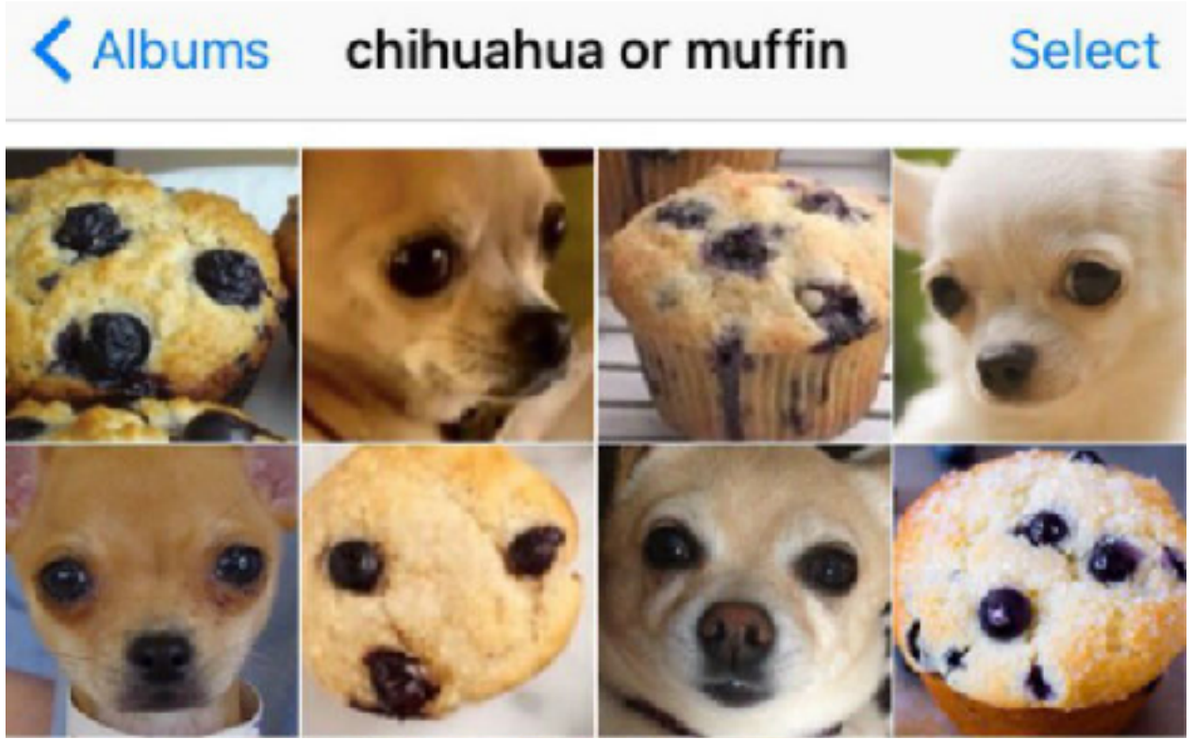




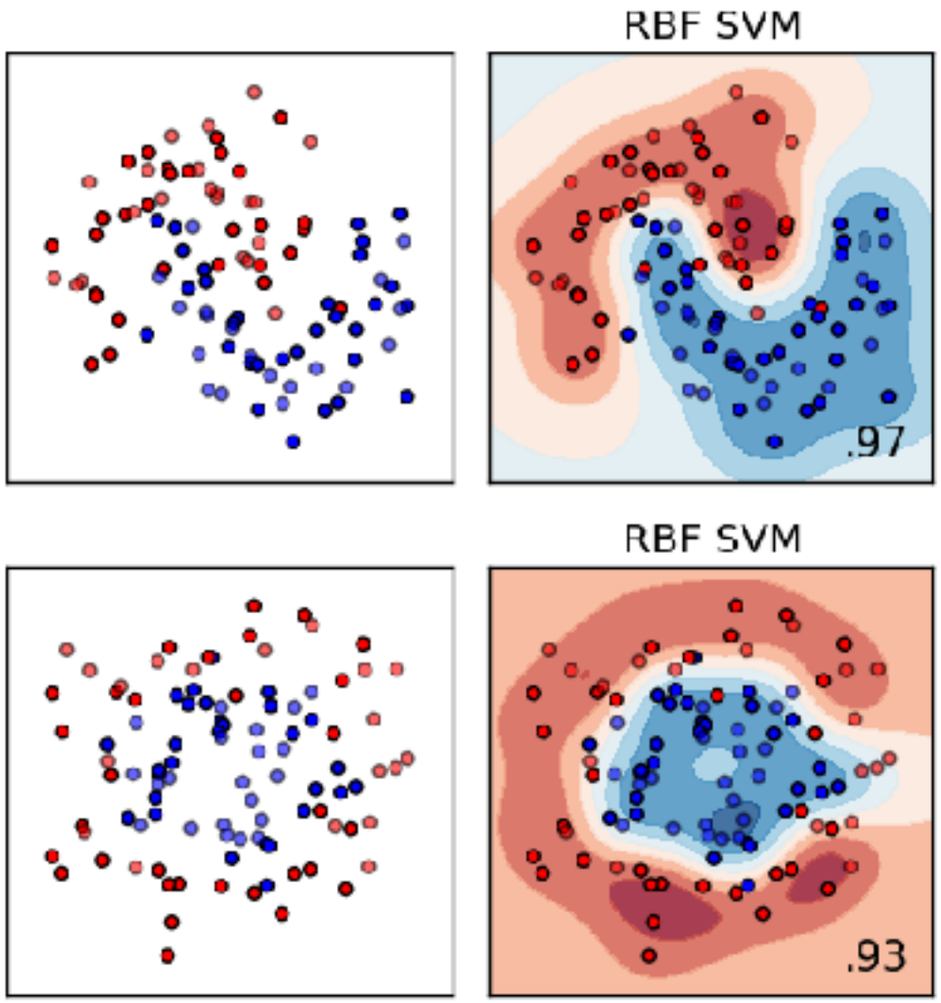


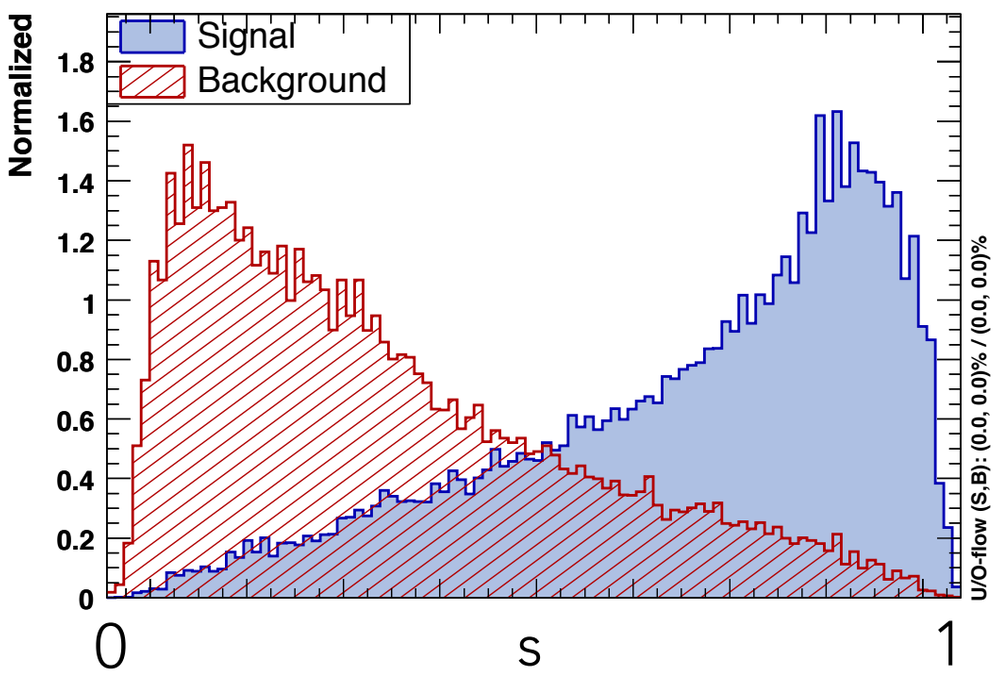
c_i





MACHINE LEARNING: CLASSIFIERS





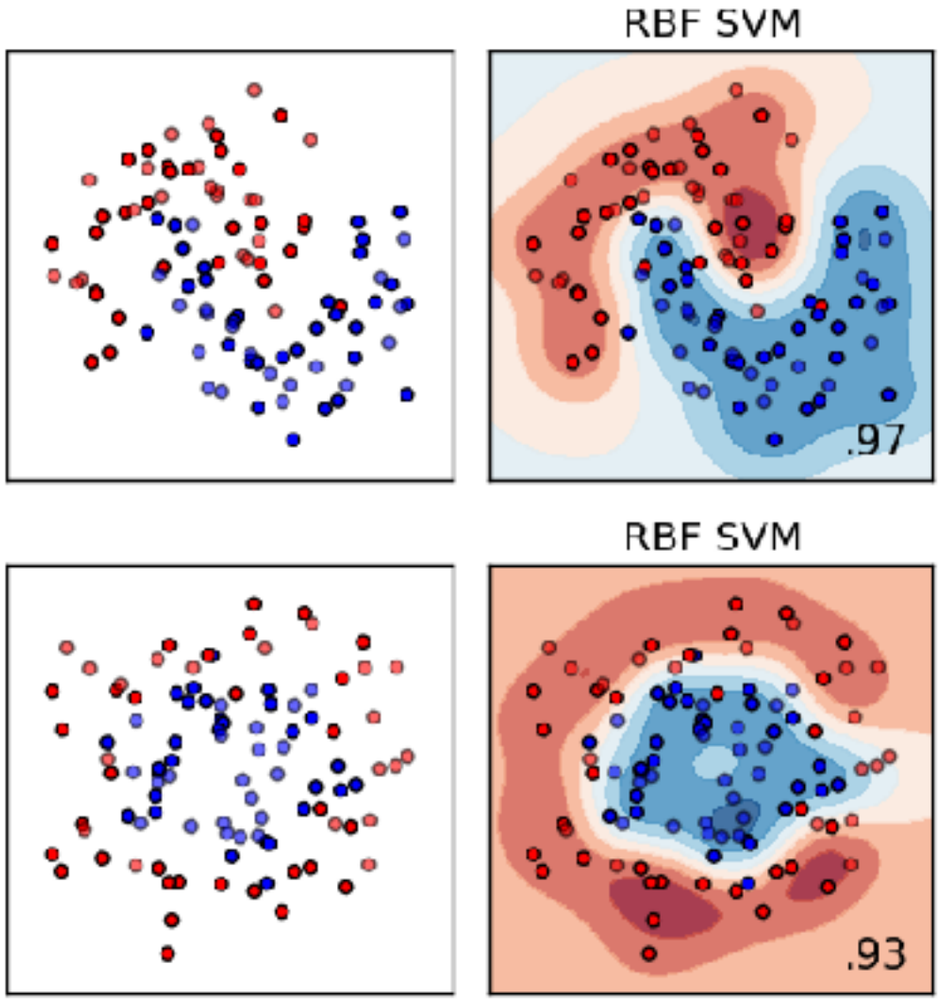
Common to use machine learning classifiers to separate signal (H_1) vs. background (H_0)

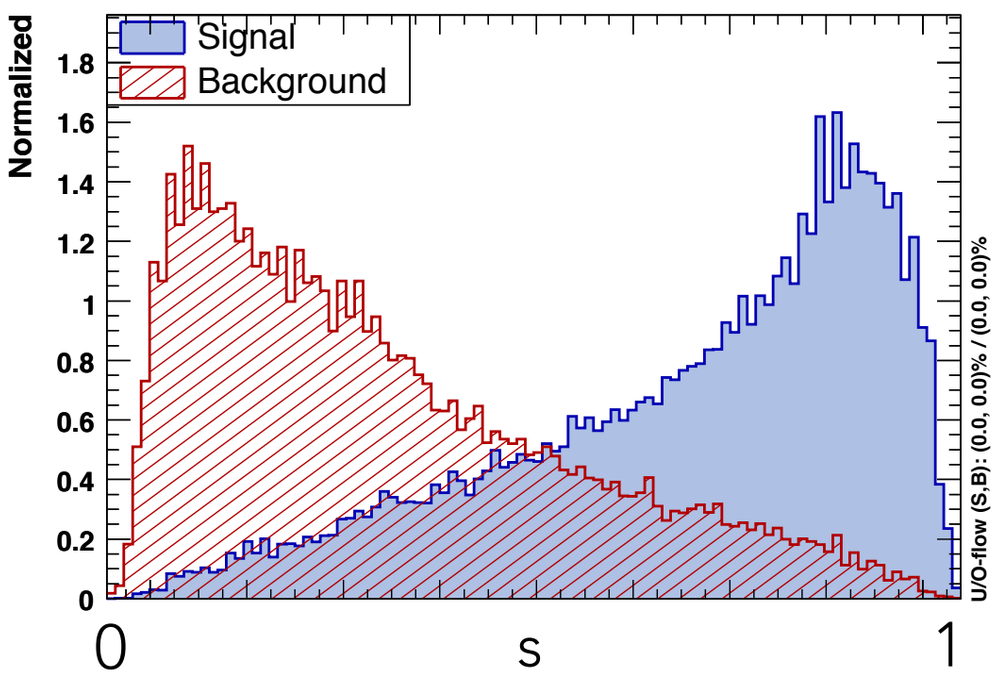
- want a function s: X→ Y that maps signal to y=1 and background to y=0

- **calculus of variations**: find function s(x) that minimizes ***loss***:

$$L[s] = \int p(x|H_0)\,(0 - s(x))^2\,dx + \int p(x|H_1)\,(1 - s(x))^2 dx$$

MACHINE LEARNING: CLASSIFIERS





- **applied calculus of variations**: find function s(x) that minimizes

loss: $$L[s] = \int p(x|H_0)\,(0 - s(x))^2\,dx + \int p(x|H_1)\,(1 - s(x))^2 dx$$

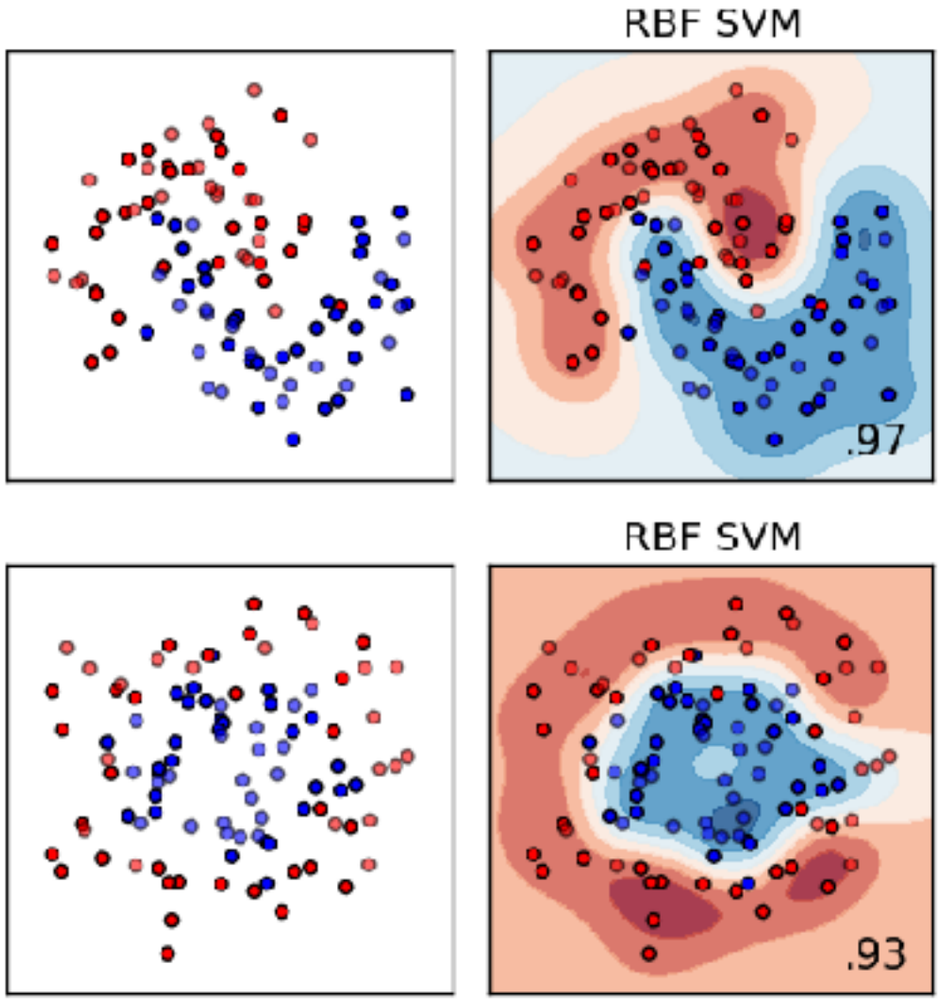
- i.e. approximate the optimal classifier

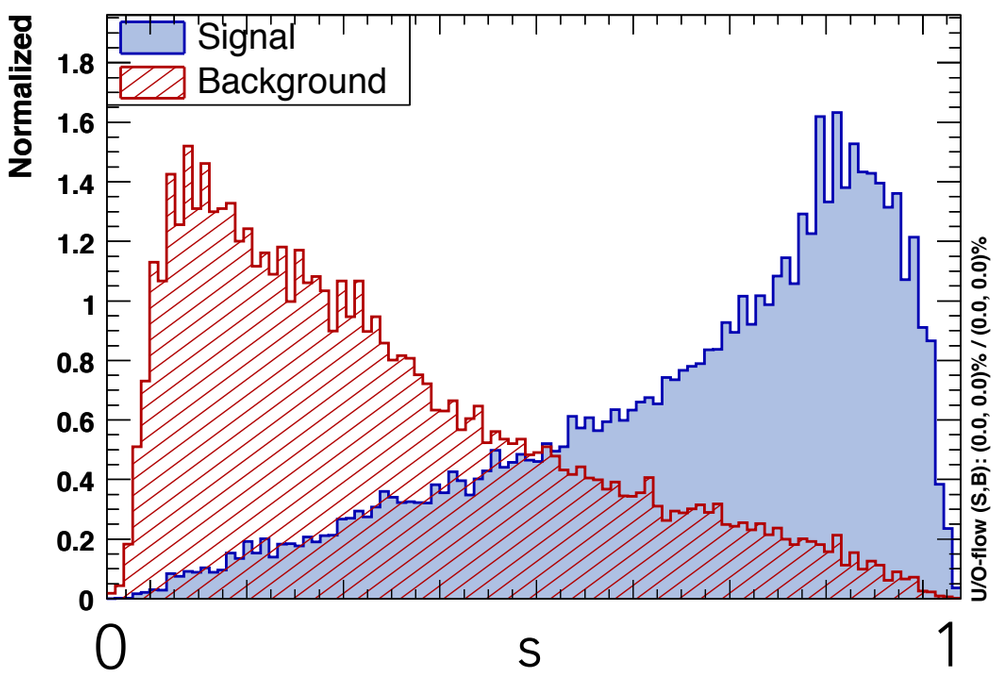
$$s(x) = \frac{p(x|H_1)}{p(x|H_0) + p(x|H_1)}$$

- which is 1-to-1 with the likelihood ratio

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

MACHINE LEARNING: CLASSIFIERS





- **applied calculus of variations**: find function s(x) that minimizes

loss: $$L[s] = \int p(x|H_0)\,(0 - s(x))^2\,dx + \int p(x|H_1)\,(1 - s(x))^2 dx$$

$$\approx \frac{1}{N}\sum_{i=1}^{N}(y_i - s(x_i))^2$$

- i.e. approximate the optimal classifier

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

- which is 1-to-1 with the likelihood ratio

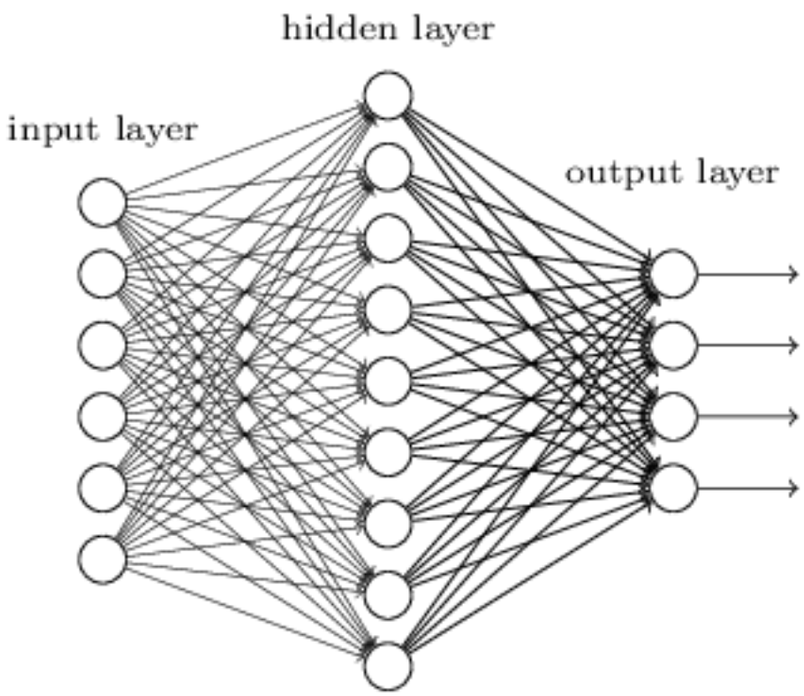
$$\frac{p(x|H_1)}{p(x|H_0)}$$

NN = A HIGHLY FLEXIBLE FAMILY OF FUNCTIONS

In calculus of variations, the optimization is over all functions: $\hat{s} = \underset{s}{\operatorname{argmin}} L[s]$

- In applied calculus of variations, we consider a highly flexible family of functions s_ϕ and optimize: i.e. $\hat{\phi} = \underset{\phi}{\operatorname{argmin}} L[s_\phi]$ and $\hat{s} \approx s_{\hat{\phi}}$
- Think of neural networks as a highly flexible family of functions
- Machine learning also includes non-convex optimization algorithms that are effective even with millions of parameters!

Shallow neural network



Deep neural network

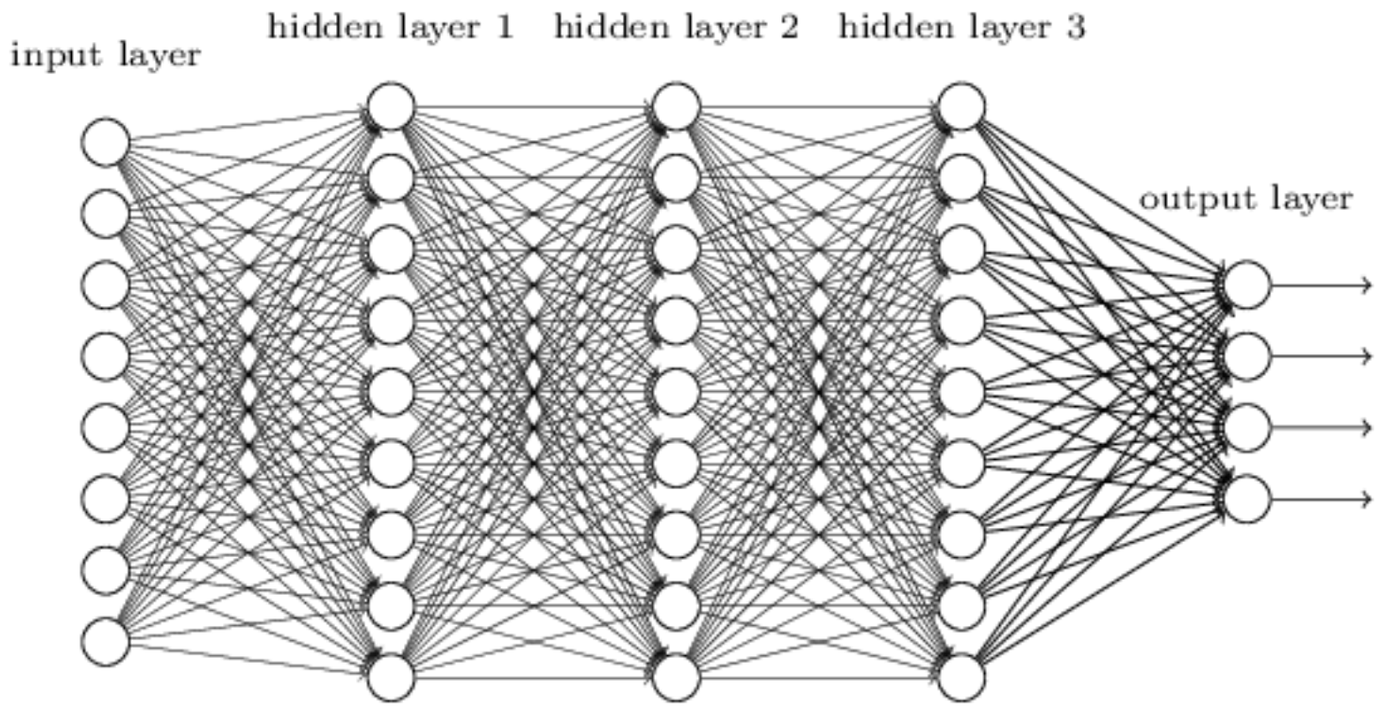


image credit: Michael Nielsen

PARAMETRIZED CLASSIFIERS

We showed 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 Θ. 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: http://arxiv.org/abs/1506.02169

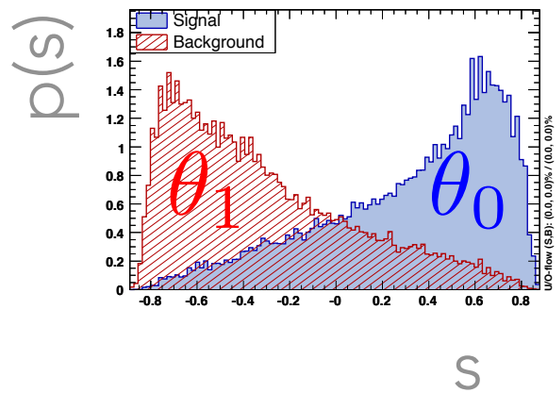
LIKELIHOOD RATIO TESTS

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 $s\colon X \to \mathbb{R}$ has the same level sets as the likelihood ratio

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

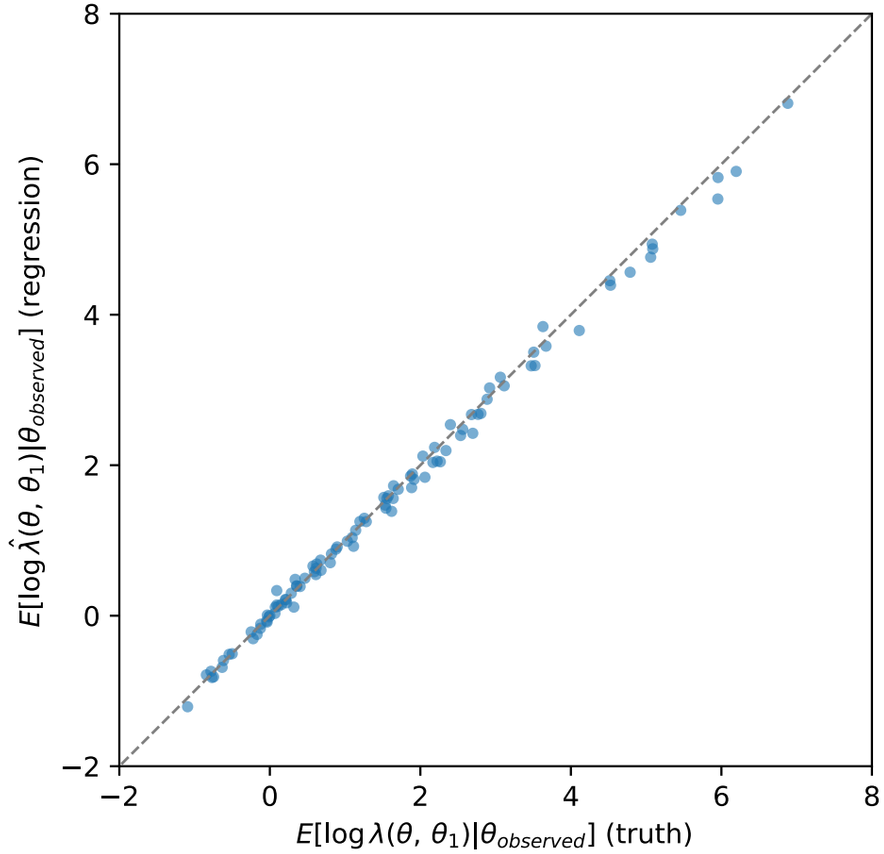
Estimating the density of $s(x;\theta_0,\theta_1)$ via the simulator calibrates the ratio.

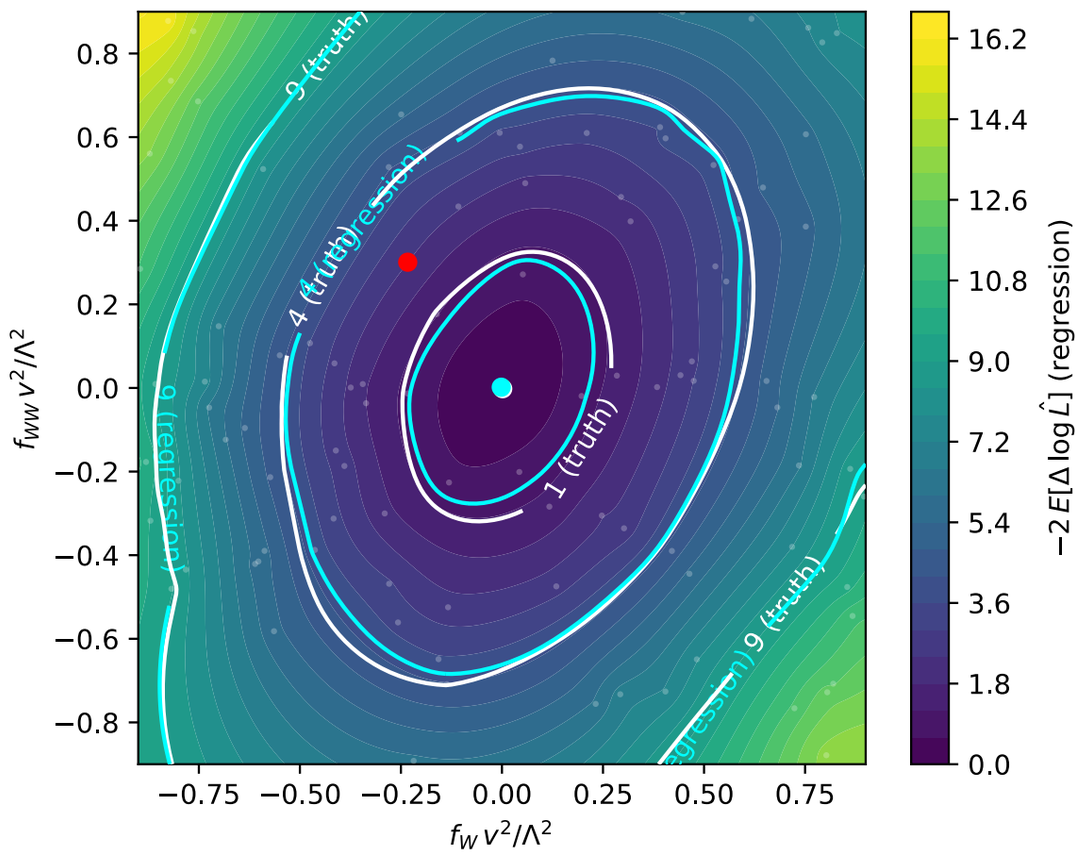
K.C., G. Louppe, J. Pavez: http://arxiv.org/abs/1506.02169

LEARNING THE HIGGS EFT LIKELIHOOD

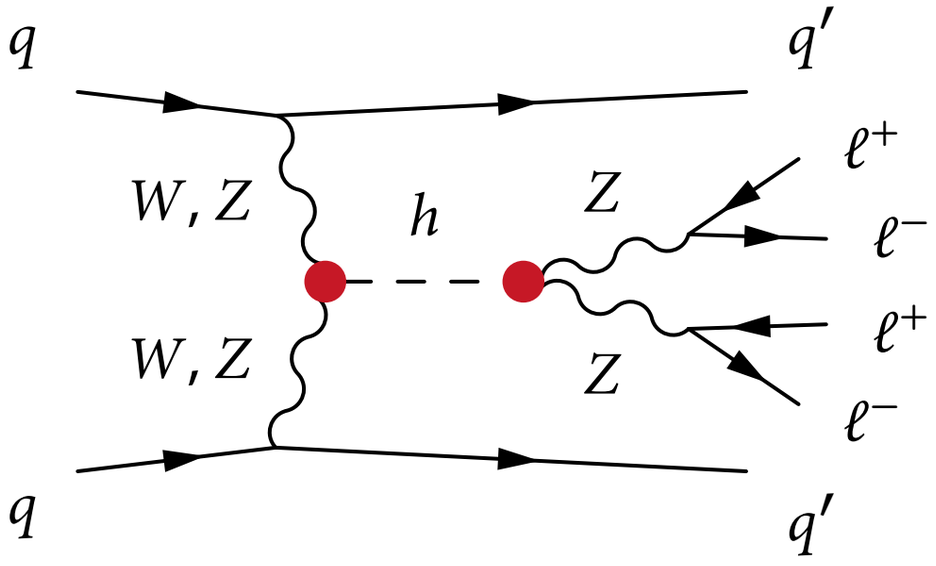
(based on 16-D fully differential cross-section)

Estimated likelihood





True likelihood



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]

THE PLAYERS

forward modeling
generation
simulation

PREDICTION

θ
parameters of interest

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

x
observed data
simulated data

z
latent variables
Monte Carlo truth

ν
nuisance parameters

INFERENCE

inverse problem
measurement
parameter estimation

Generative Models

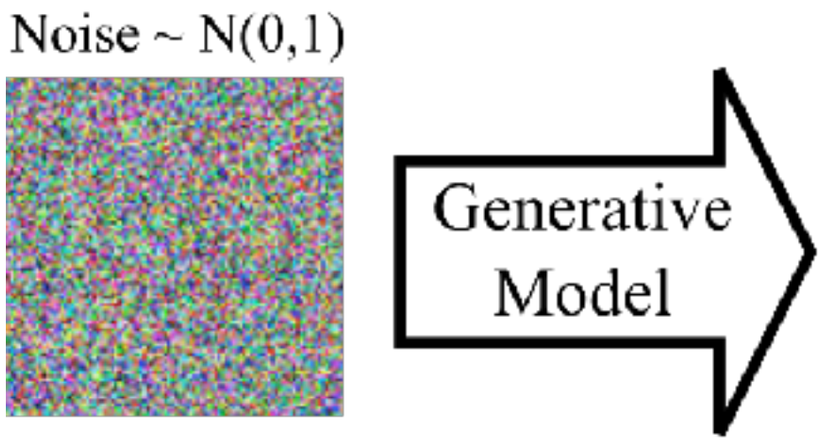
"What I cannot create, I do not understand."

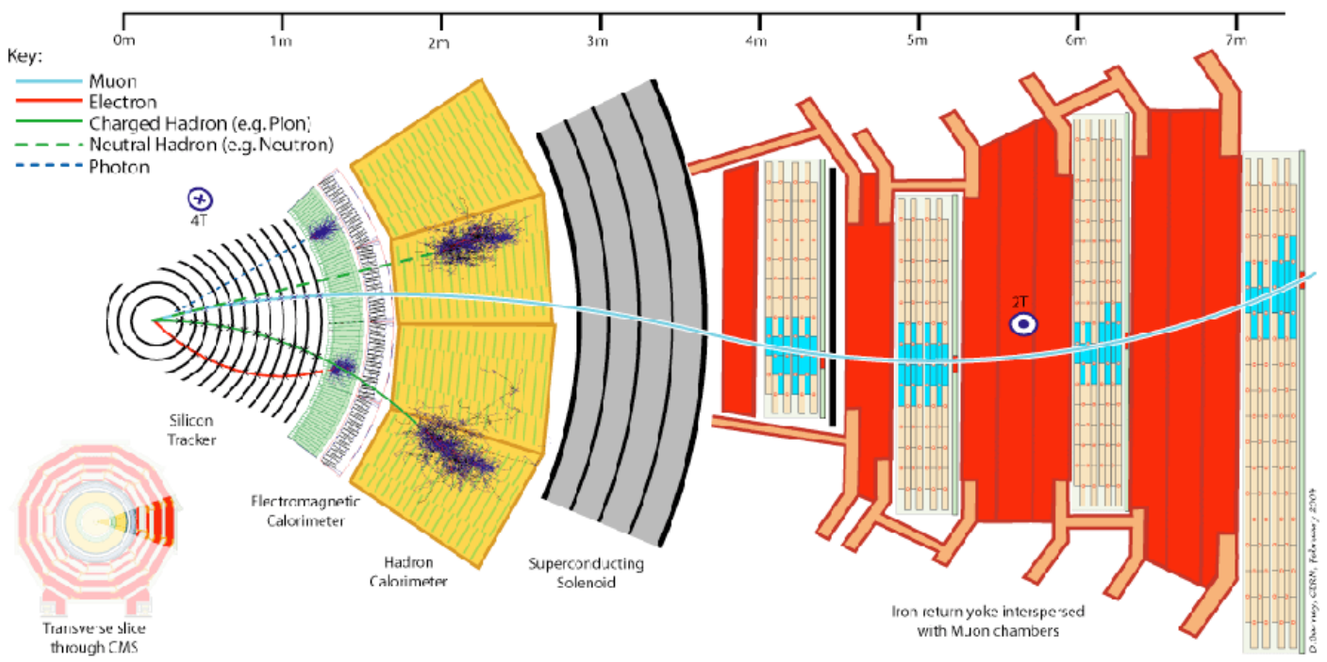
—RICHARD FEYNMAN

LEARNING THE GENERATIVE MODEL

Z

X

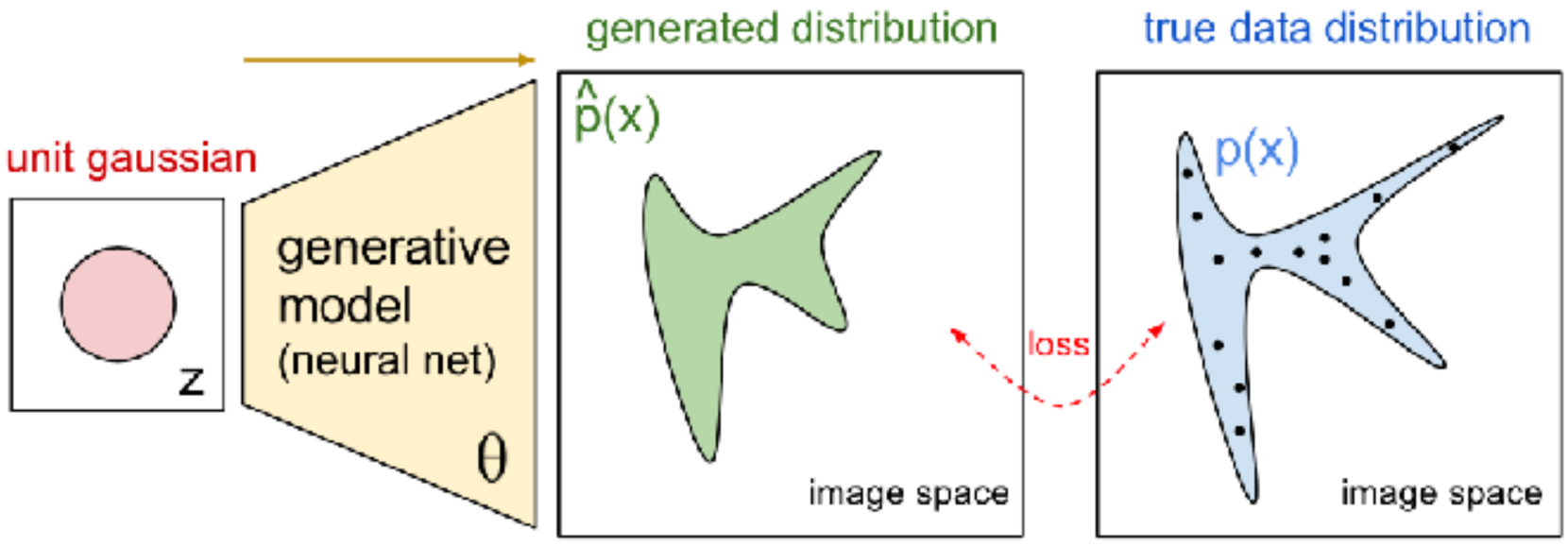




http://torch.ch/blog/2015/11/13/gan.html

GENERATIVE ADVERSARIAL NETWORKS

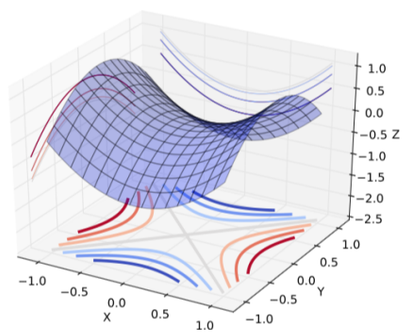
Goodfellow, et al arXiv:1406.2661
slide Gilles Louppe



Leo is G Tom is D

- Two-player game:
  - a discriminator D,
  - a generator G;
- D is a classifier $\mathcal{X} \mapsto \{0, 1\}$ that tries to distinguish between
  - a sample from the data distribution ($D(\mathbf{x}) = 1$, for $\mathbf{x} \sim p_{\text{data}}$),
  - and a sample from the model distribution ($D(G(\mathbf{z})) = 0$, for $\mathbf{z} \sim p_{\text{noise}}$);
- G is a generator $\mathcal{Z} \mapsto \mathcal{X}$ trained to produce samples $G(\mathbf{z})$ (for $\mathbf{z} \sim p_{\text{noise}}$) that are difficult for D to distinguish from data.

$$(D^*, G^*) = \max_D \min_G V(D, G).$$



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

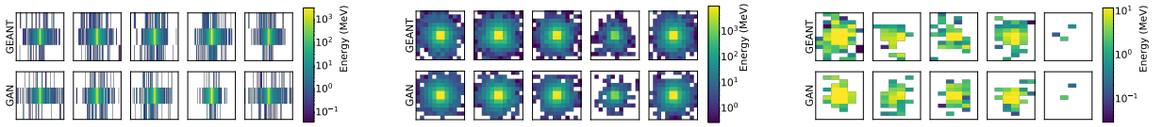


Figure 9: Five randomly selected e^+ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

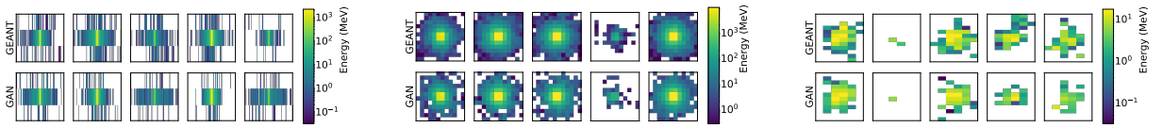


Figure 10: Five randomly selected γ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

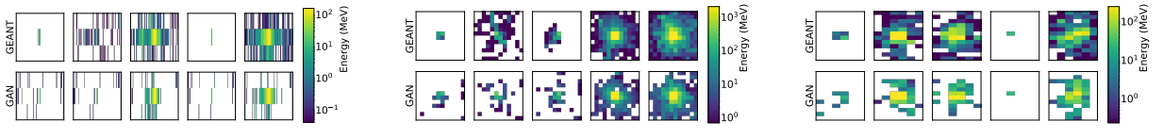
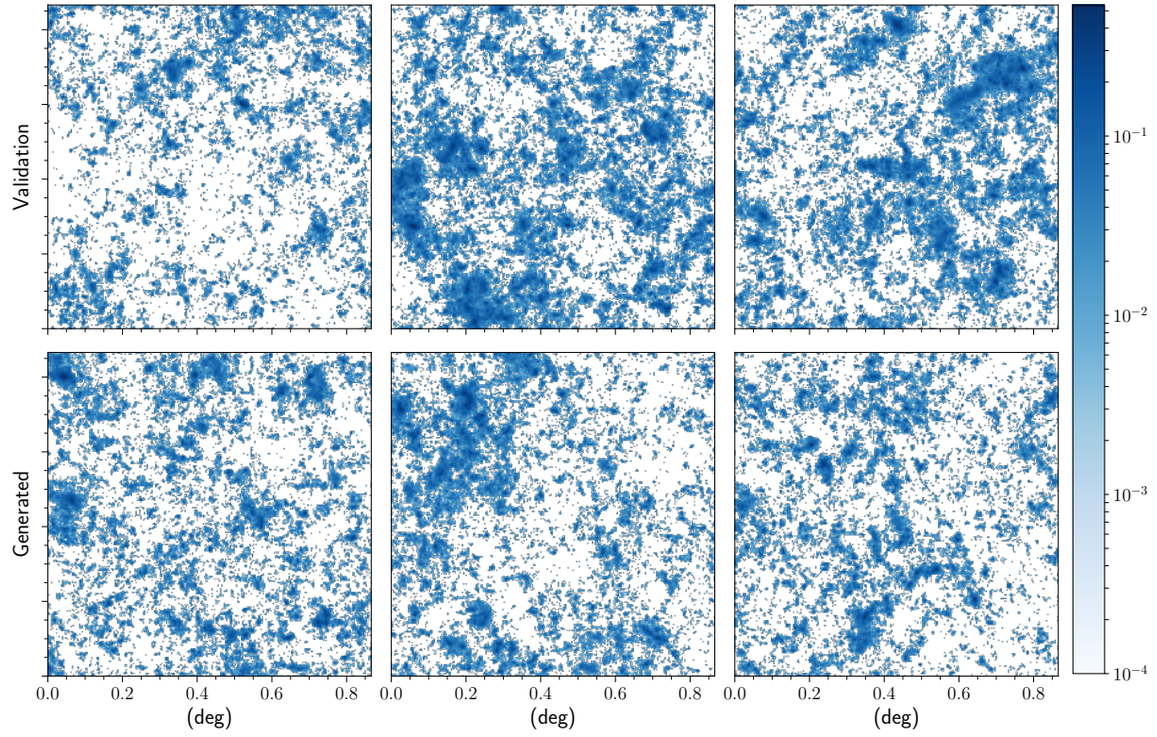


Figure 11: Five randomly selected π^+ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

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



GENERATIVE MODELS FOR CALIBRATION

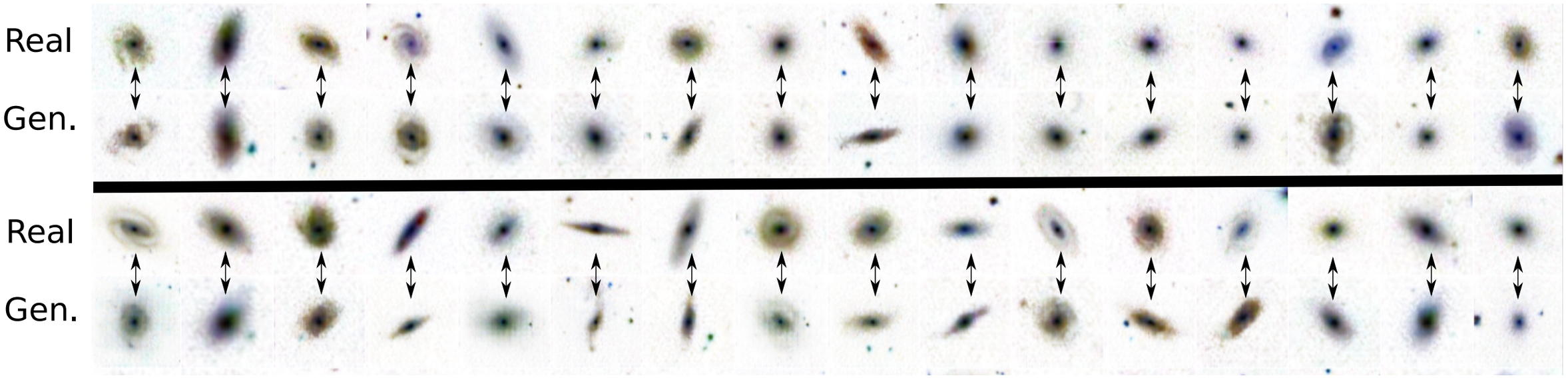
Use of generative models of galaxy images to help calibrate down-stream analysis in next-generation surveys.

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.**

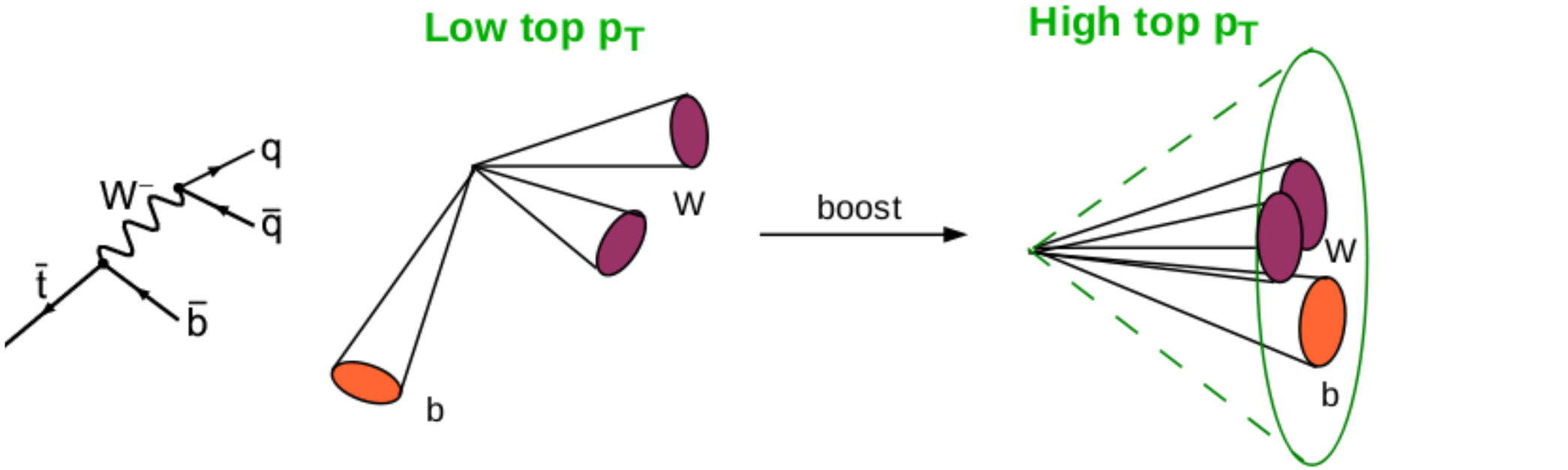


Physics-Aware Machine Learning

(injecting our physics knowledge)

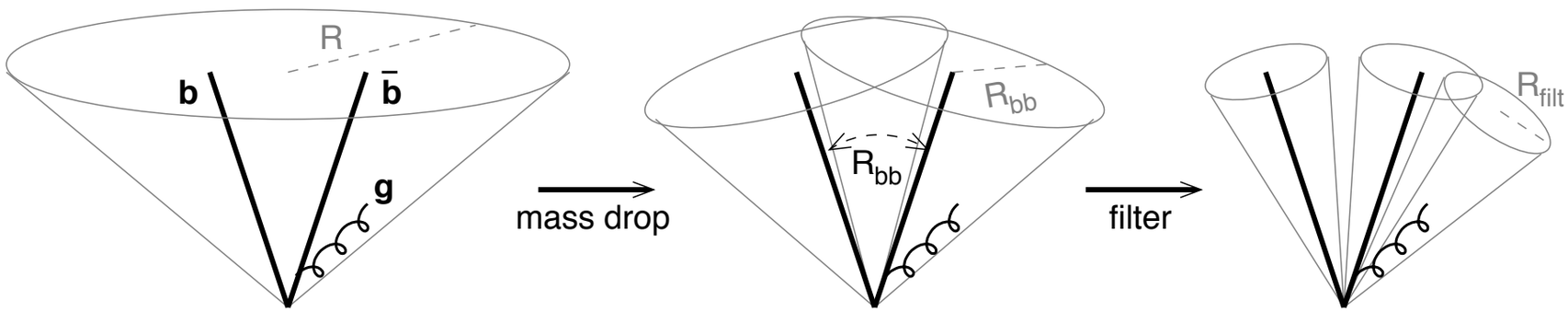
JET SUBSTRUCTURE

Many scenarios for physics Beyond the Standard Model include highly boosted W, Z, H bosons or top quarks



Identifying these rests on subtle substructure inside jets

- an enormous number of theoretical effort in developing observables and techniques to tag jets like this



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

ATLAS
EXPERIMENT

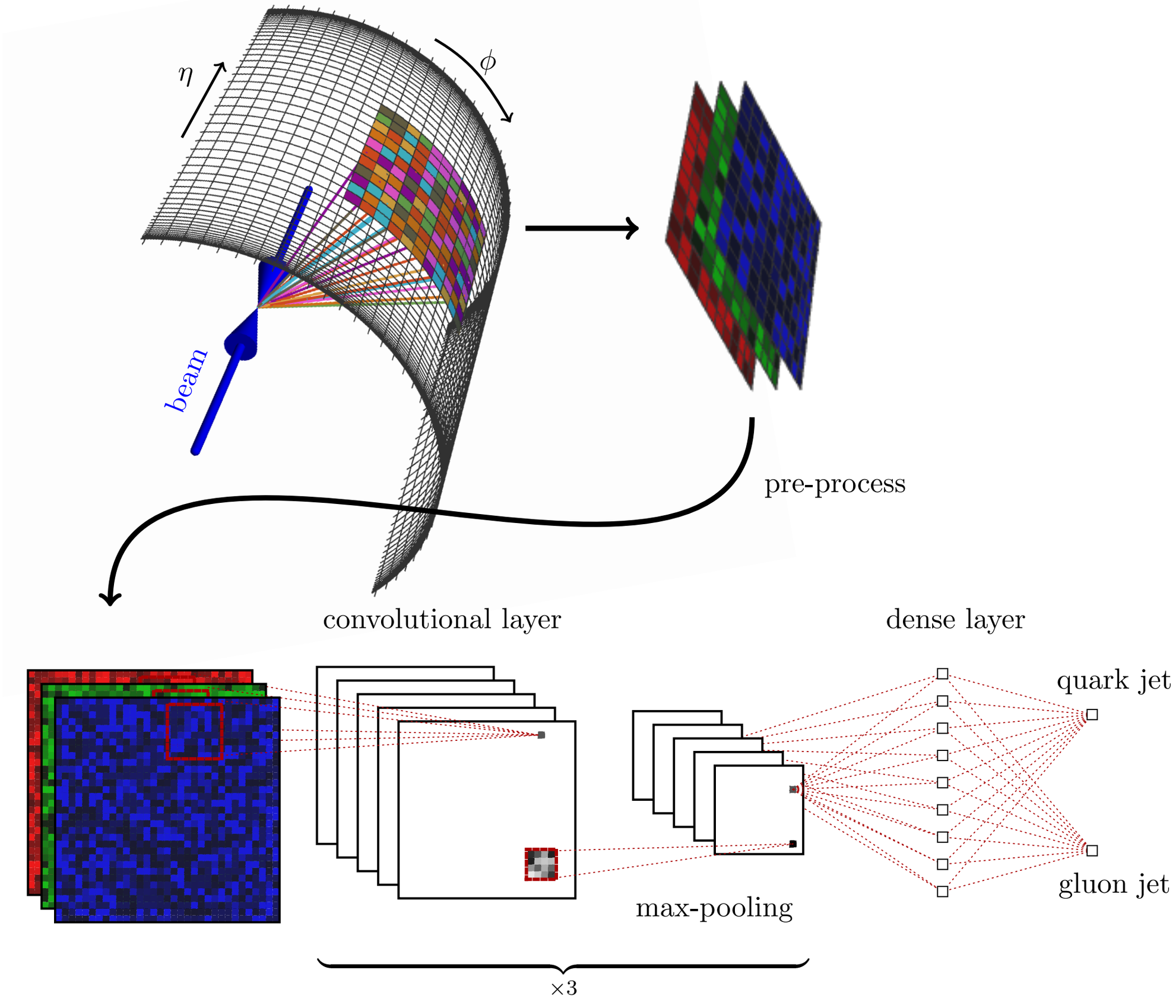
JET IMAGES

image: Komiske, Metodiev, Schwartz arxiv:1612.01551

Oliveira, et. al arXiv:1511.05190

Whiteson, et al arXiv:1603.09349

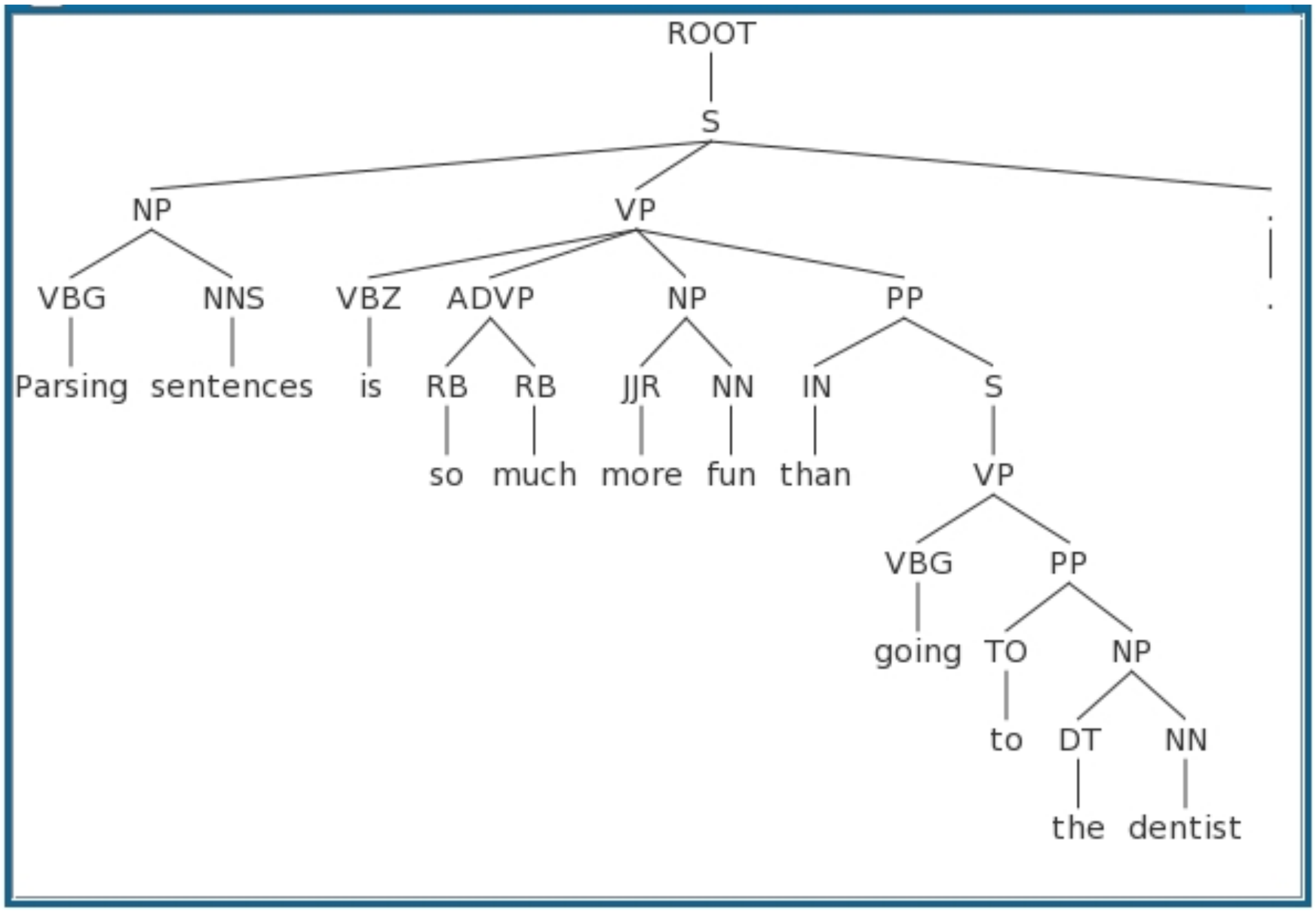
Barnard, et al arXiv:1609.00607



FROM IMAGES TO SENTENCES

Recursive Neural Networks showing great performance for Natural Language Processing tasks

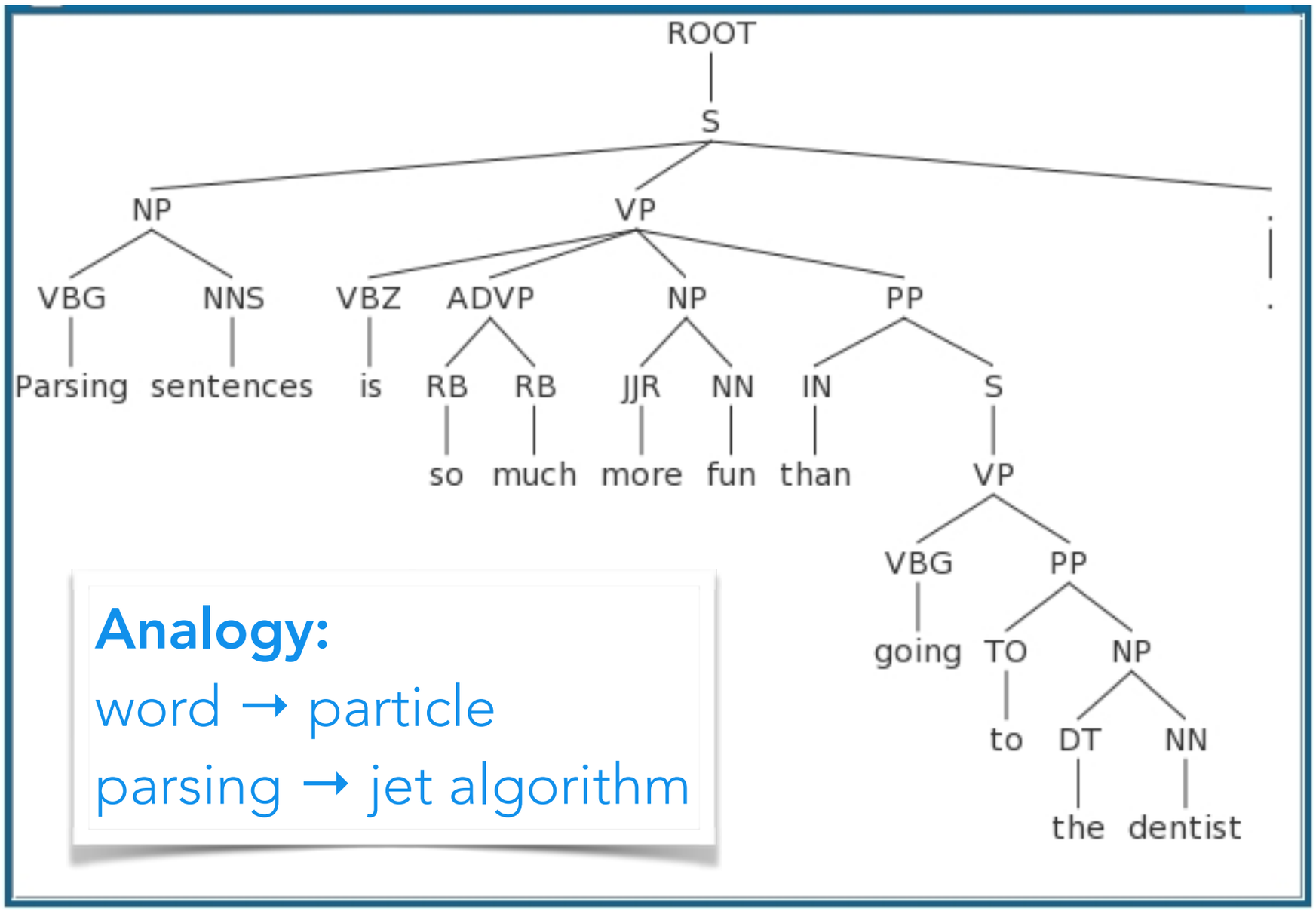
- neural network's topology given by parsing of sentence!



FROM IMAGES TO SENTENCES

Recursive Neural Networks showing great performance for Natural Language Processing tasks

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

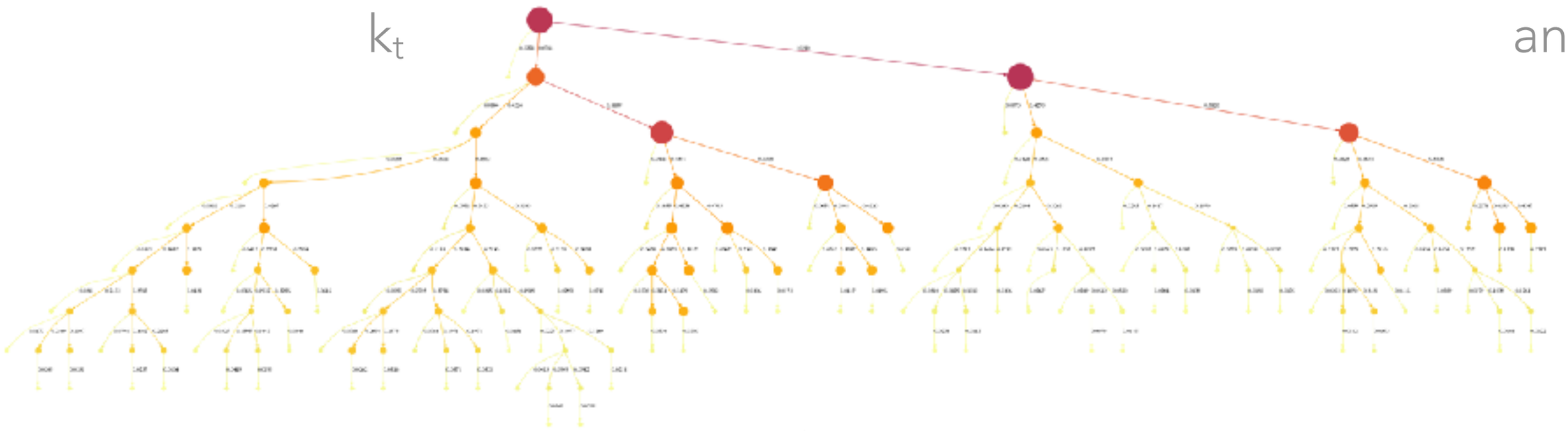


Analogy:

word → particle

parsing → jet algorithm

QCD-INSPIRED RECURSIVE NEURAL NETWORKS

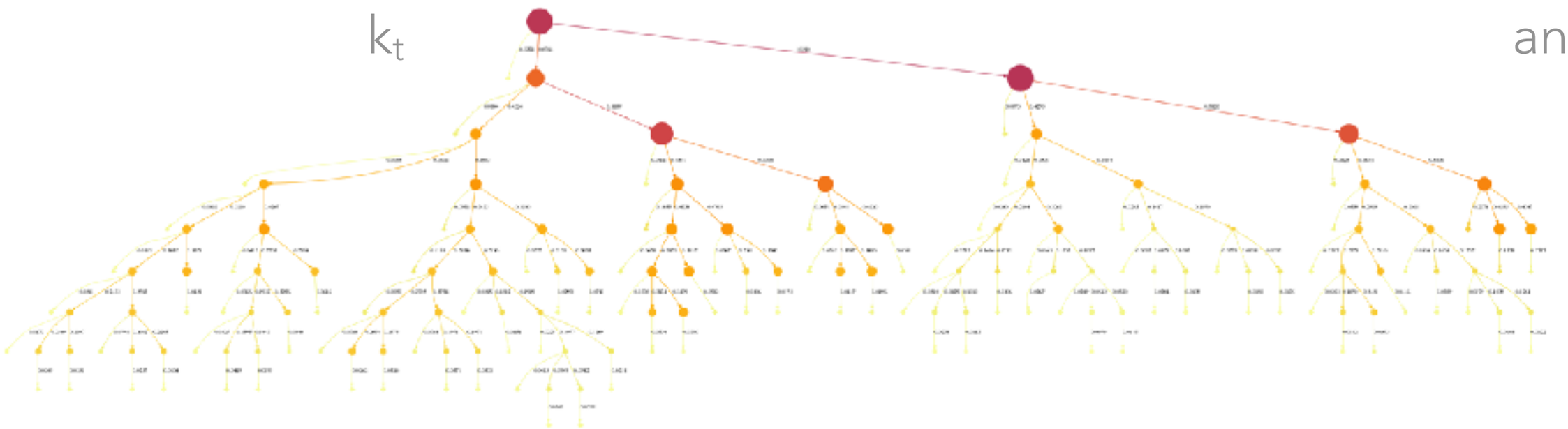


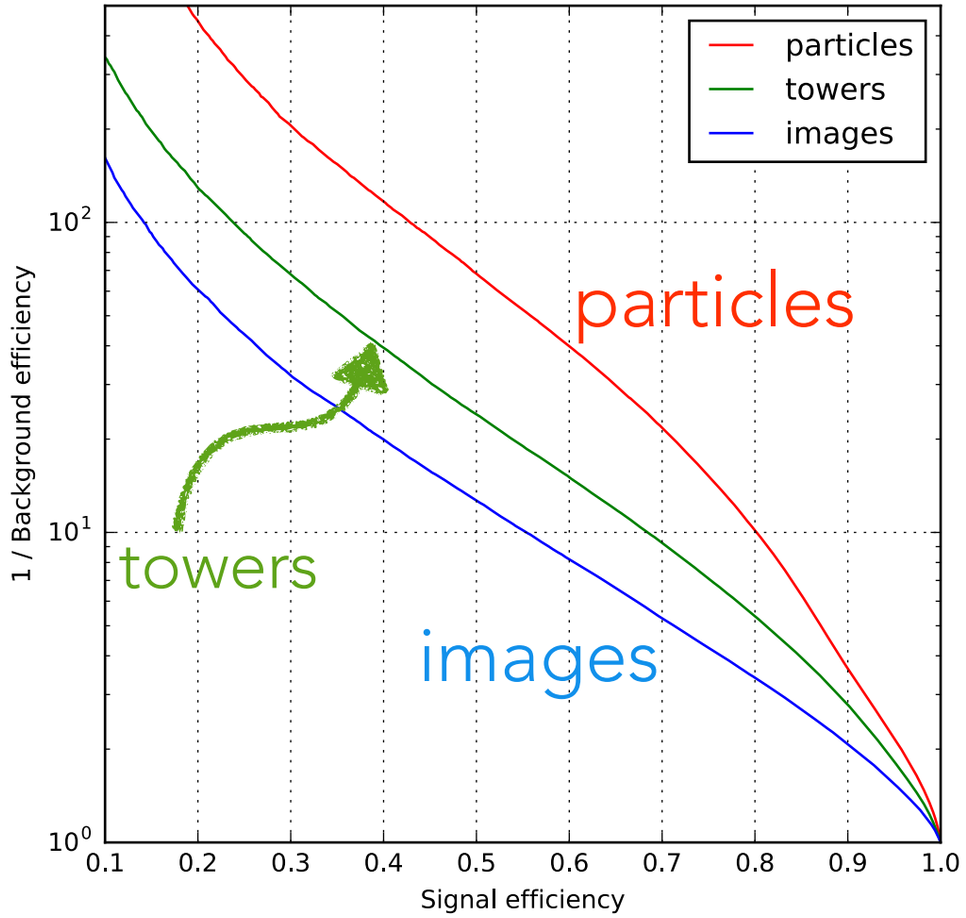
Work with Gilles Louppe, Kyunghyun Cho, Cyril Becot

- Use sequential recombination jet algorithms to provide network topology (**on a per-jet basis**)
- path towards ML models with good theoretical 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 Isaac Henrion & Joan Bruna using graph conv nets
follow up: Taoli Cheng arXiv:1711.02633

QCD-INSPIRED RECURSIVE NEURAL NETWORKS



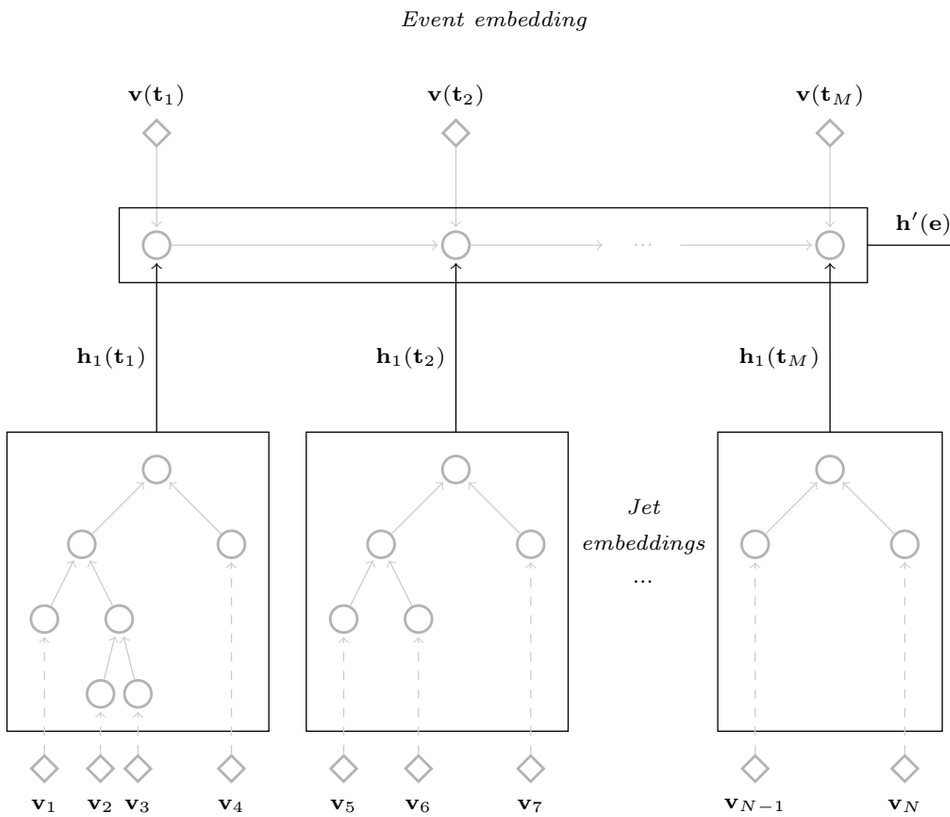


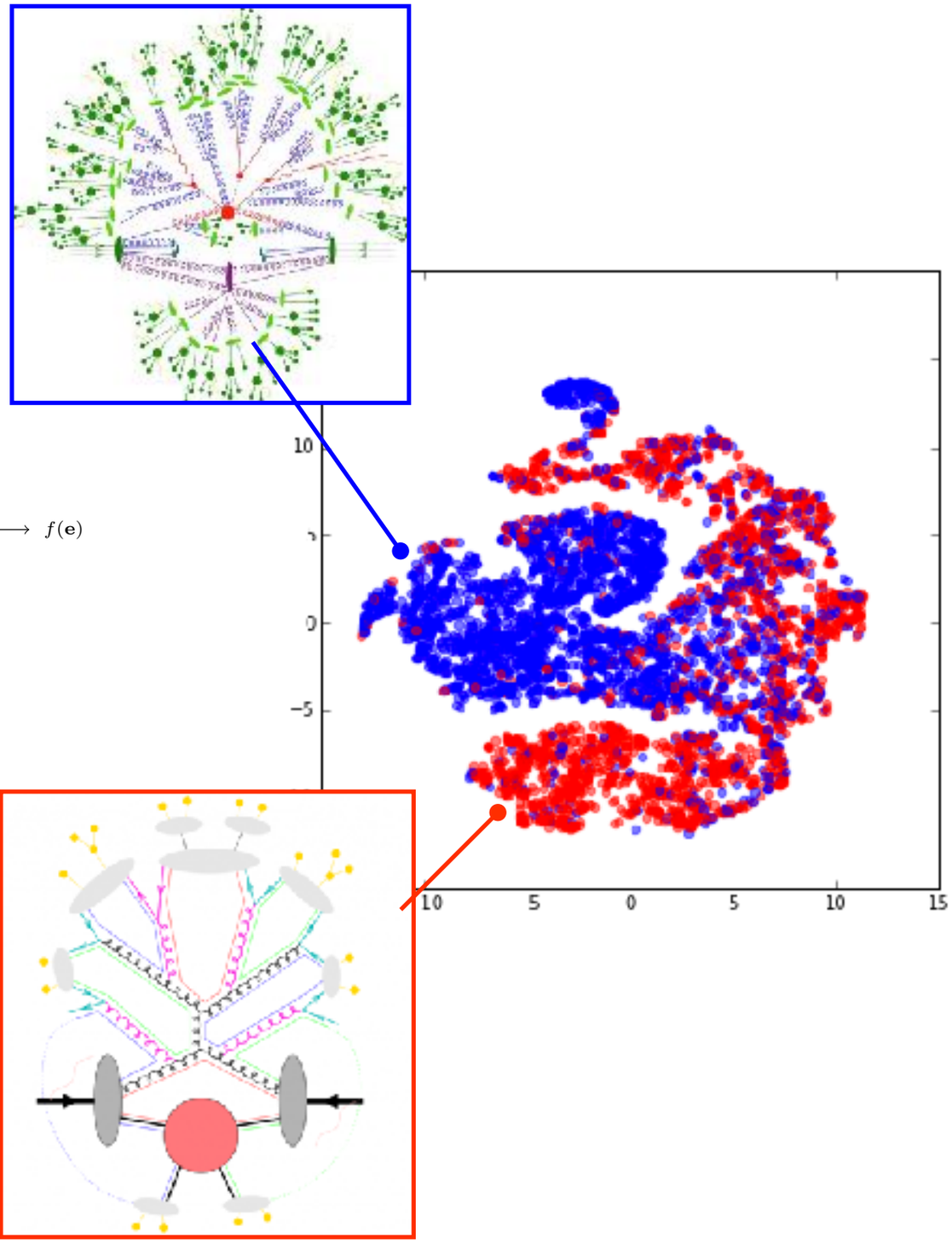
- W-jet tagging example using data from Dawe, et al arXiv:1609.00607
- down-sampling by projecting into images looses information
- RNN needs much less data to train!

HIERARCHICAL MODEL FOR THE ENTIRE EVENT

particle embedding → jet embedding → event embedding → classifier

It scales!





arXiv:1702.00748;
follow up: graph conv nets & T.Cheng arXiv:1711.02633

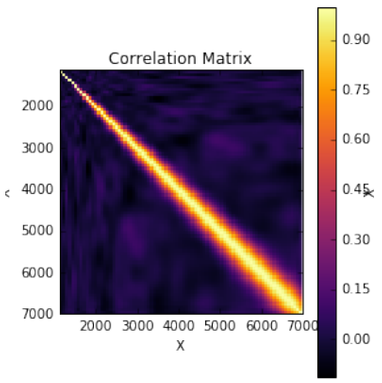
PHYSICS-AWARE MACHINE LEARNING

We can inject our knowledge of physics into the variational family

… in some cases we can extract physics as well!

Physics-aware Gaussian Processes

arXiv:1709.05681

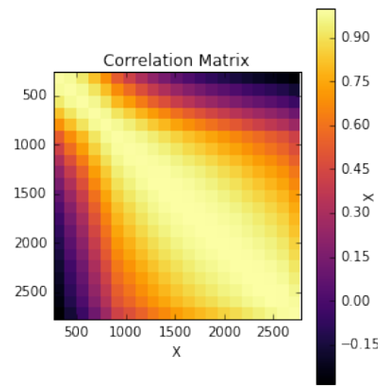


Final Kernel =

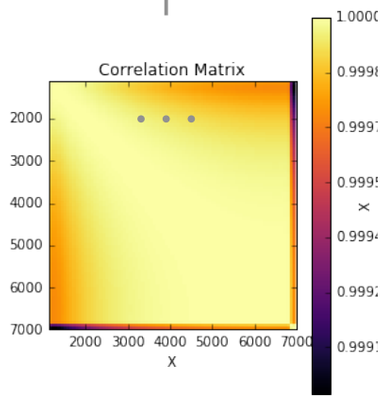
Poisson fluctuations

=

+ Mass Resolution



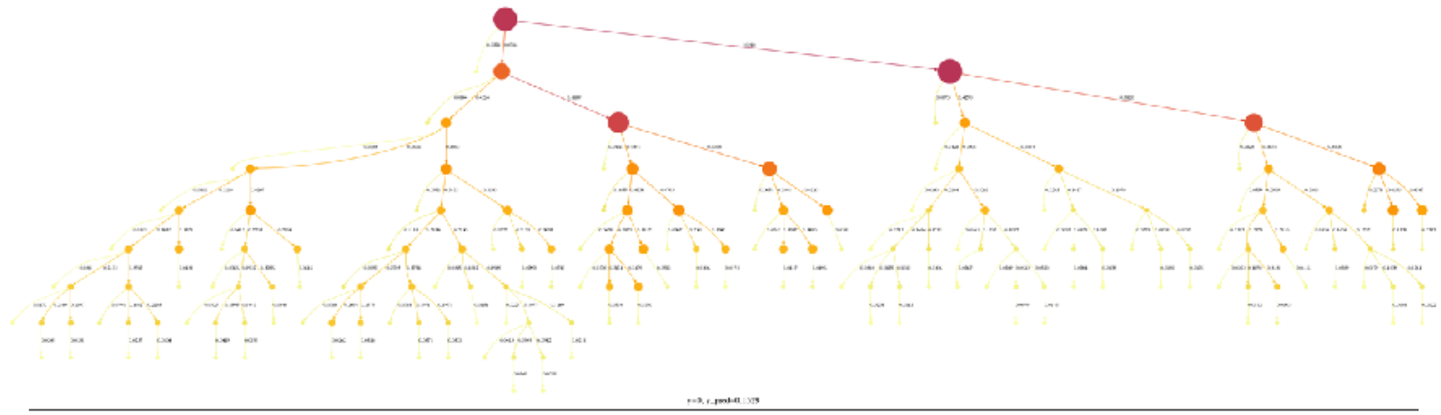
+



+ Jet Energy Scale

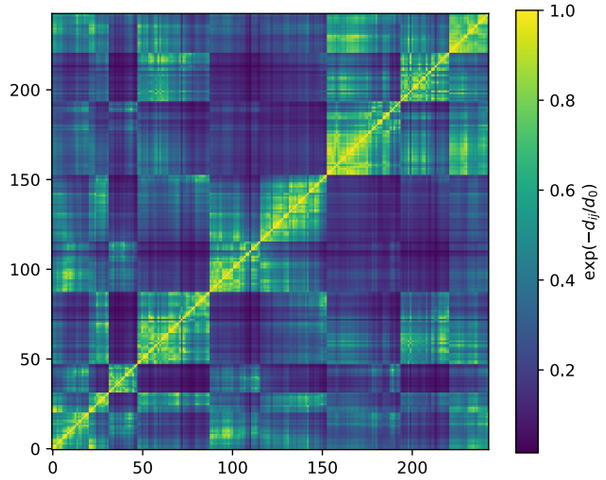
QCD-Aware recursive neural networks

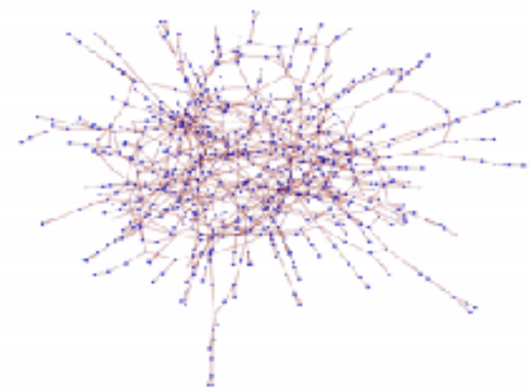
arXiv:1702.00748



QCD-Aware graph convolutional neural networks

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





$$d_{ii'}^{\alpha} = \min(p_{ti}^{2\alpha}, p_{ti'}^{2\alpha}) \frac{\Delta R_{ii'}^2}{R^2}$$

Systematics

G. Louppe, M. Kagan, K.C, arXiv:1611.01046
Shimmin, et. al. arXiv:1703.03507
E Metodiev, B Nachman, J Thaler, arXiv:1708.02949
Dery, Nachman, Rubbo, Schwartzman arXiv:1702.00414

There is a lot of activity in addressing systematics in context of ML

Parametrized classifier allows us to learn dependence on nuisance parameters, which we can profile or marginalize later

- various diagnostics developed to check that dependence

"**Learn to pivot**" is a new technique using adversarial training so that machine learning model is robust / invariant to changes in nuisance parameter

- can also be used to "decorrelate" the classifier output with a an observed variable like invariant mass so can still use sidebands

- related to ML topics: "domain adaptation" & "fairness"

Recently, **weakly supervised** used to train classifier on real data assuming some control regions in data with different proportions among the classes

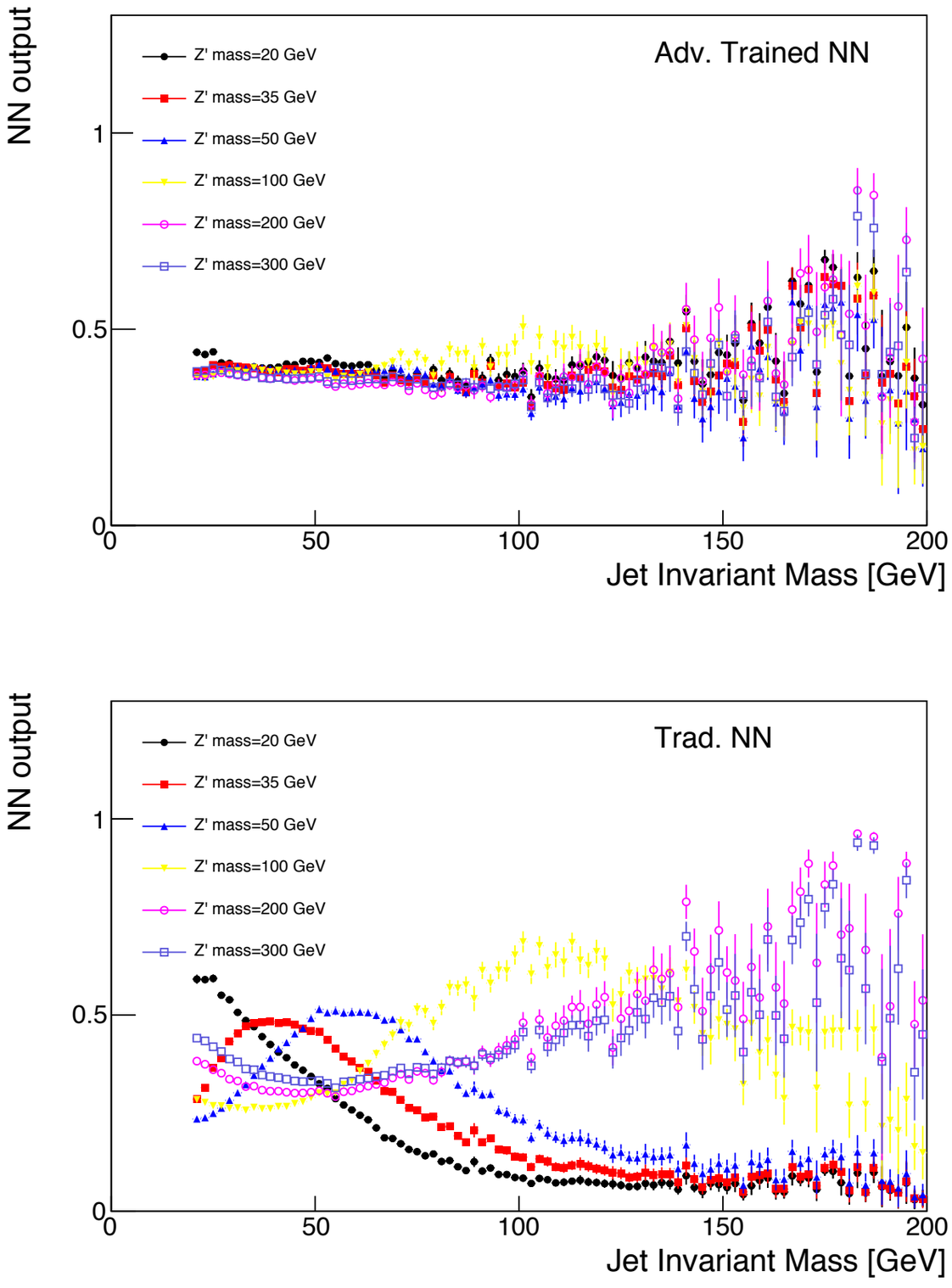
- different approaches depending on if those proportions are known or not

DECORRELATED TAGGERS

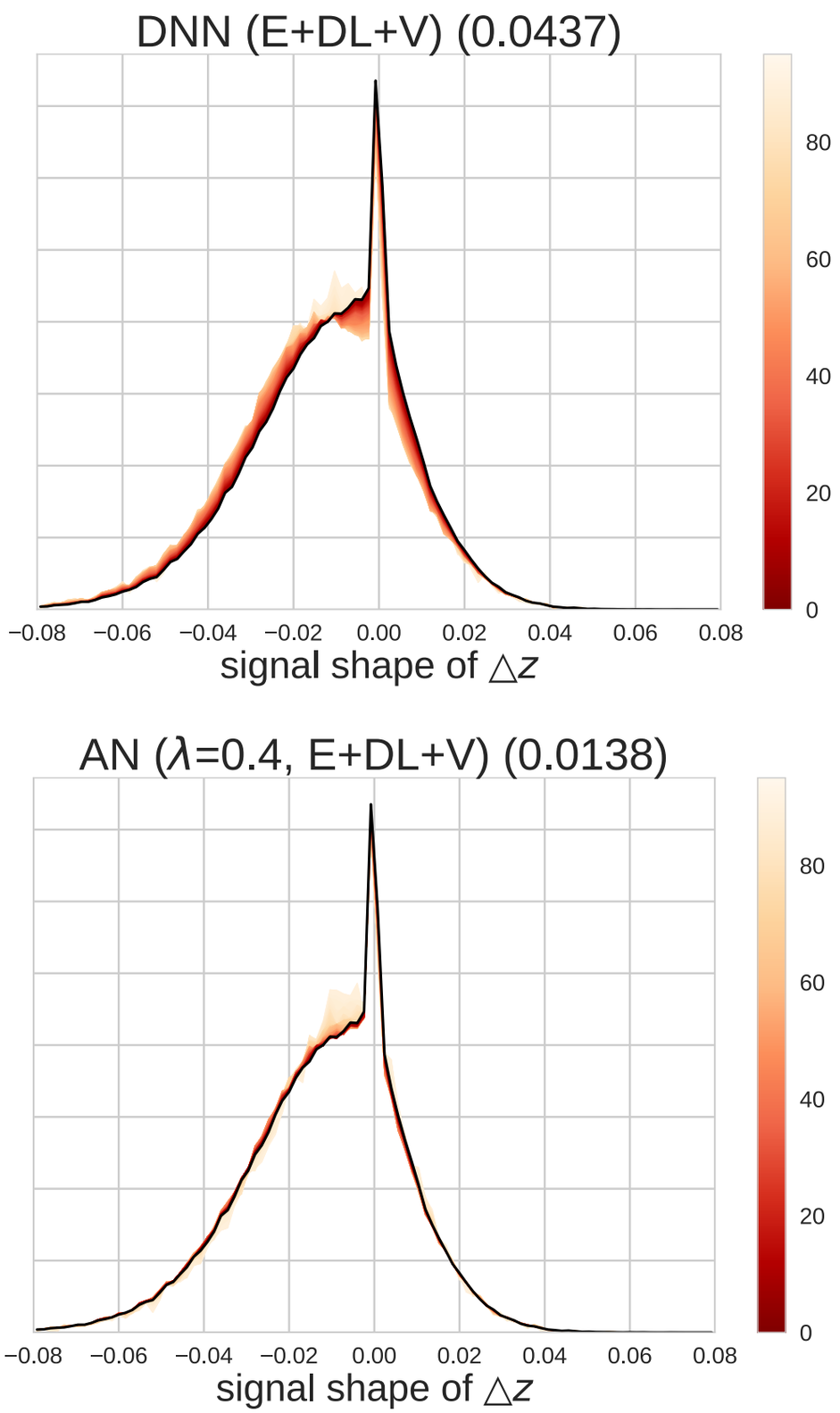
K.C, J. Pavez, and G. Louppe, arXiv:1506.02169
P. Baldi, K.C, T. Faucett, P. Sadowski, D. Whiteson arXiv:1601.07913
G. Louppe, M. Kagan, K.C, arXiv:1611.01046
Shimmin, et. al. arXiv:1703.03507

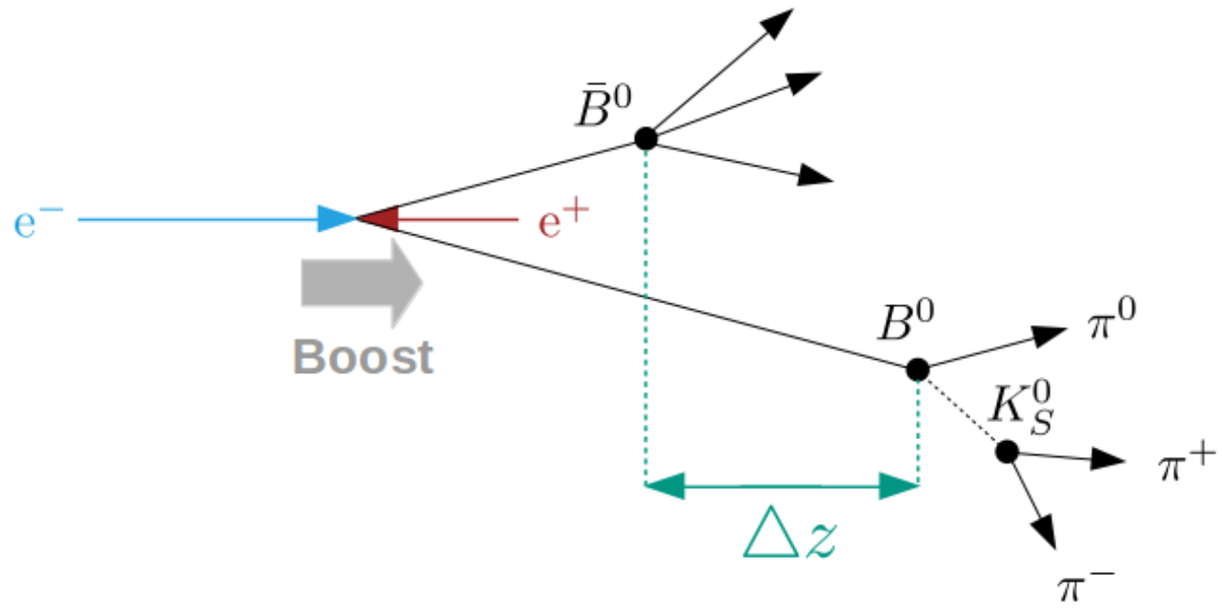
Adversarial approach of "Learning to Pivot" can also be used to train a classifier that is "decorrelated" with some other variable.

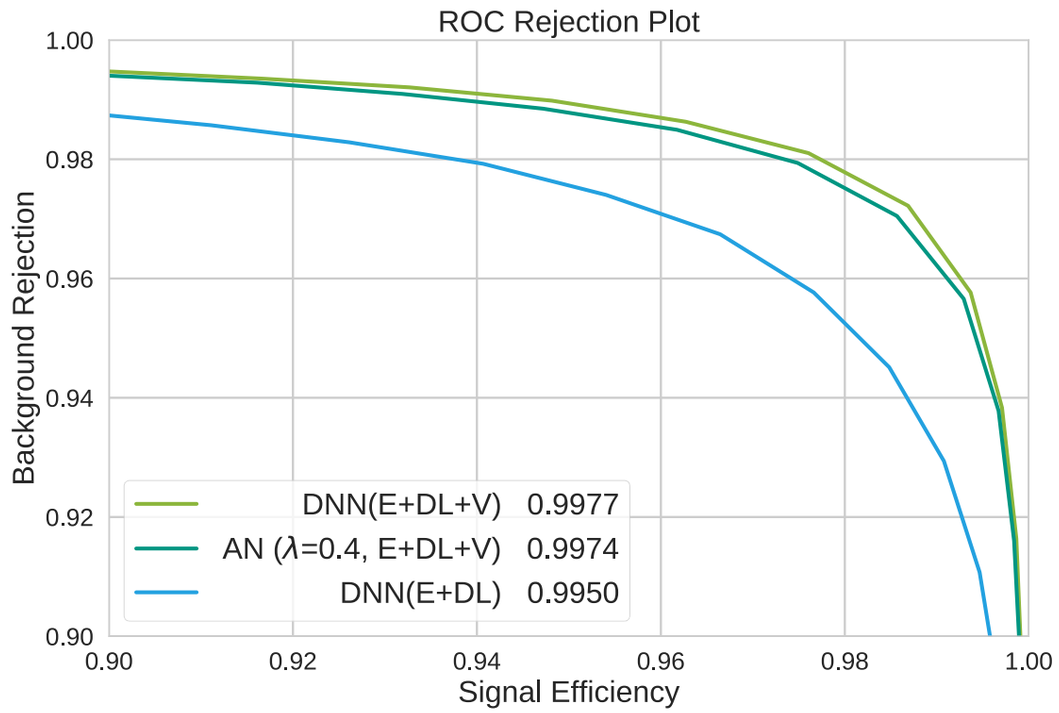
- want "jet taggers" that are decorrelated with jet invariant mass
- so that analysis can still search for a bump using jet invariant mass
- avoids sculpting background



DECORRELATION IN BELLE II







Active Sciencing

https://github.com/cranmer/active_sciencing

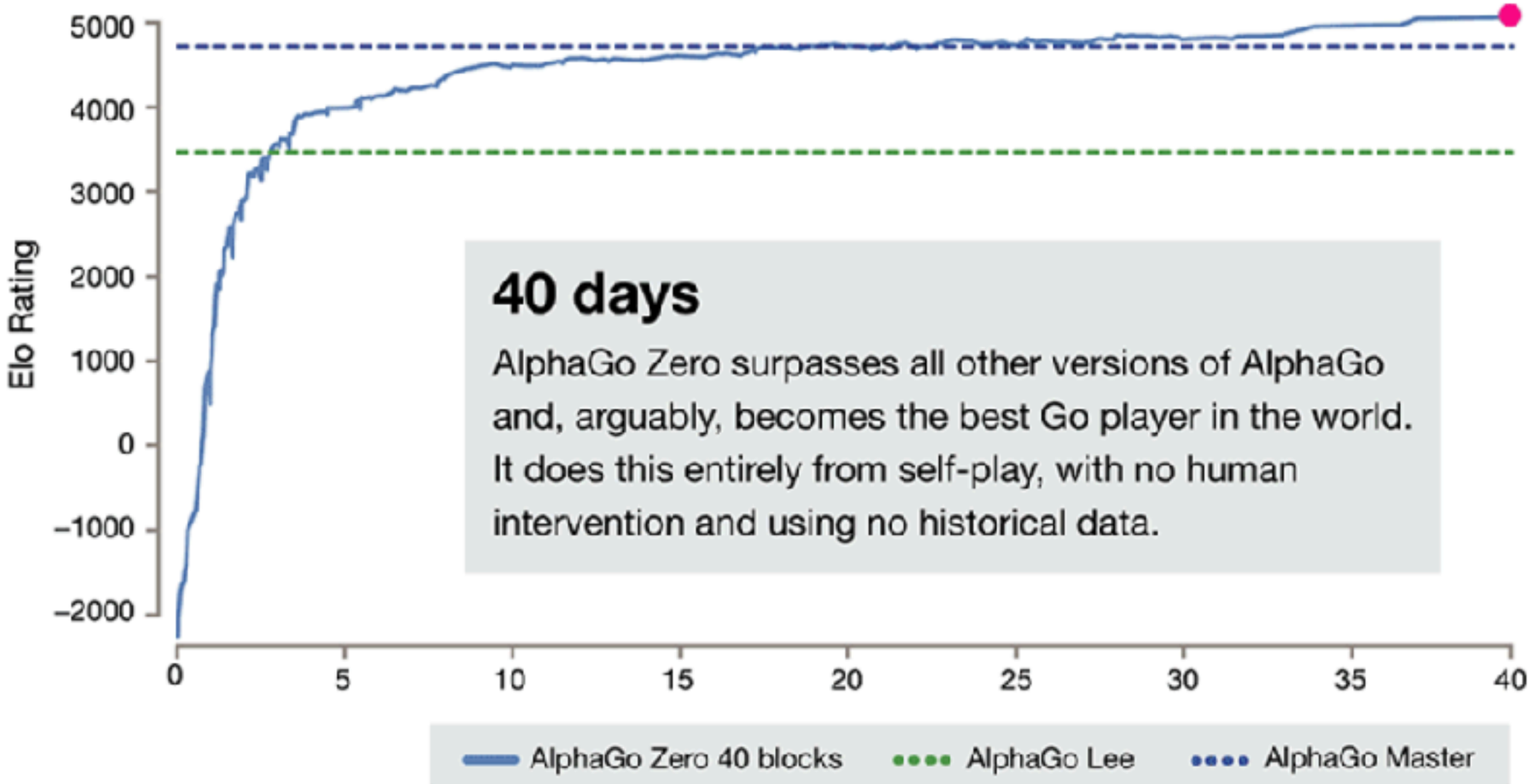
AlphaGo

(68) at (61)

Captured Stones

70 hours

AlphaGo Zero plays at super-human level. The game is disciplined and involves multiple challenges across the board.

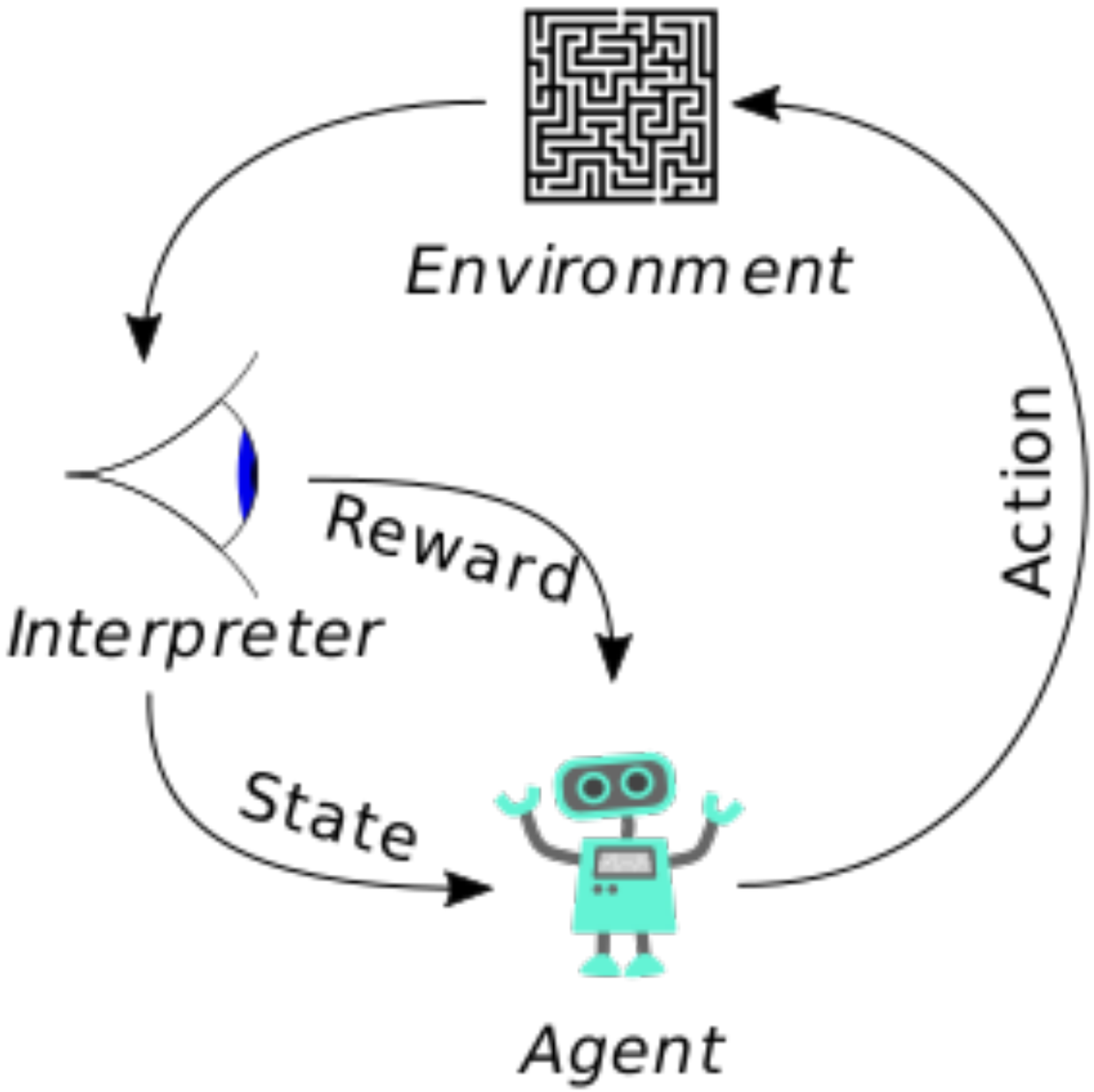


40 days

AlphaGo Zero surpasses all other versions of AlphaGo and, arguably, becomes the best Go player in the world. It does this entirely from self-play, with no human intervention and using no historical data.

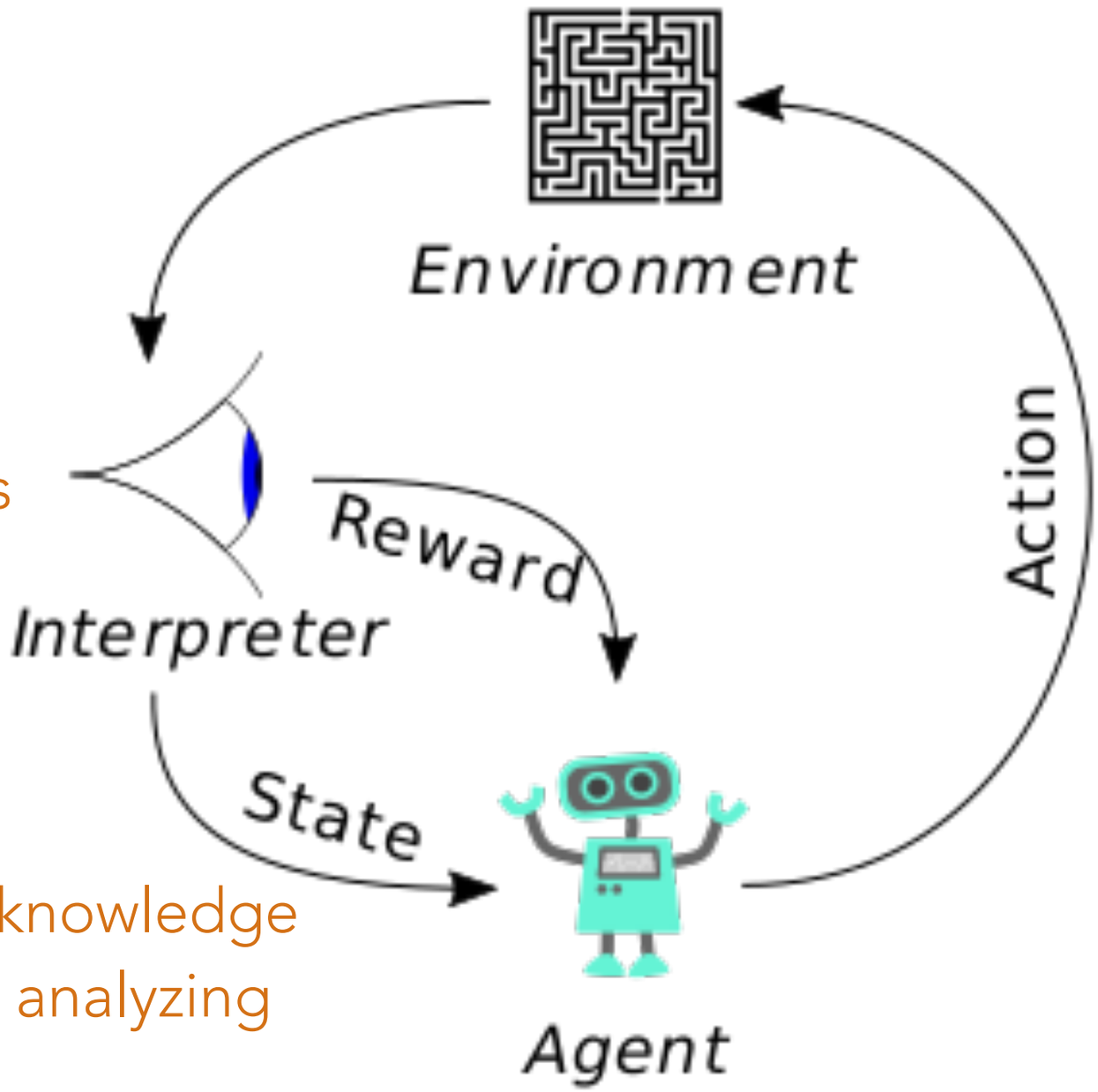
REINFORCEMENT LEARNING & SCIENTIFIC METHOD

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

AN EXAMPLE

Say we want to measure the Weinberg angle

- experiments are $e^+e^- \rightarrow \mu^+\mu^-$ at various $\sqrt{s}$ and beam polarization

- data are 4-momenta p_{μ^+} & p_{μ^-} without knowing forward-backward asymmetry is interesting observable

Can we use likelihood-free inference to:

- estimate θ_W from p_{μ^+} & p_{μ^-} generated from simulator?

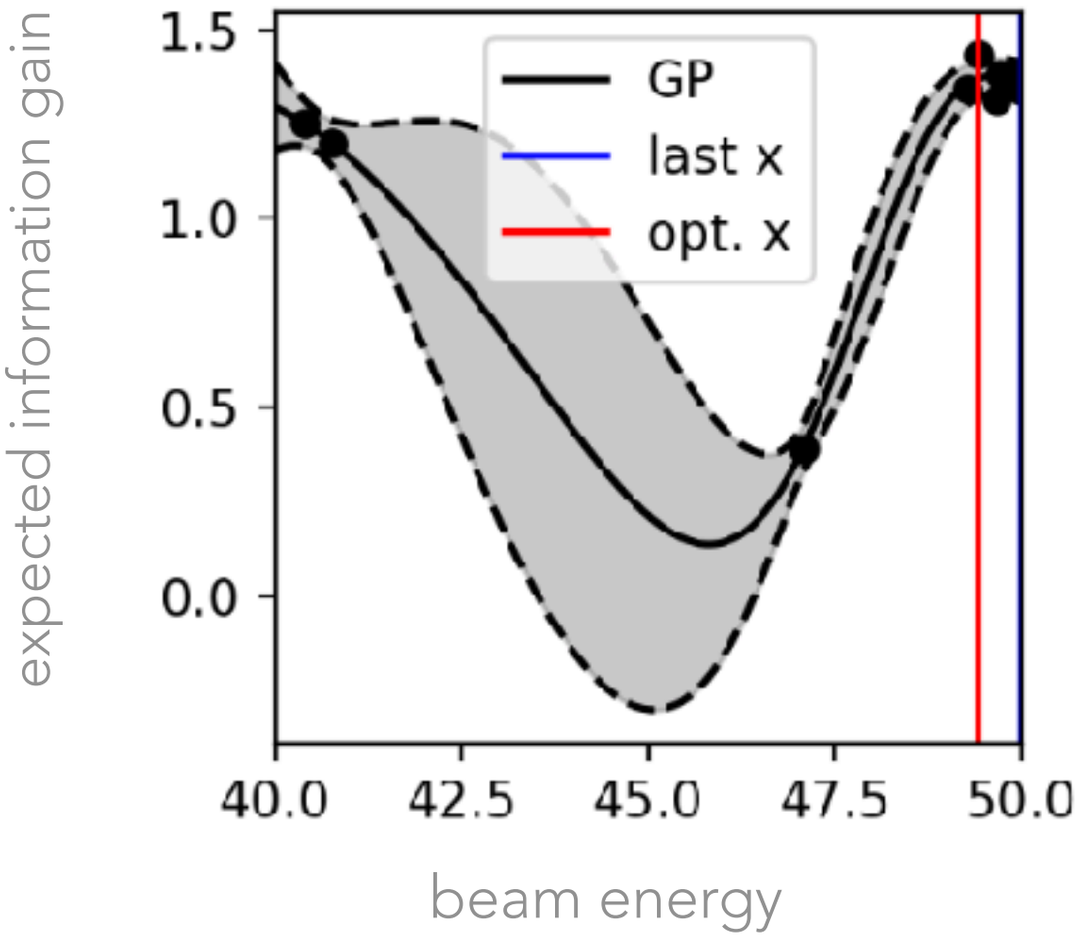
- decide which $\sqrt{s}$ and beam polarization are optimal for this measurement?

ACTIVE SCIENCING DEMO

Input:

- workflow for performing “real” experiment that returns data
- workflow for running simulator given parameters of theory and experimental configuration

Automated system can measure the Weinberg angle and optimize beam energy (eg. just above or below $M_Z/2$) just from using simulator



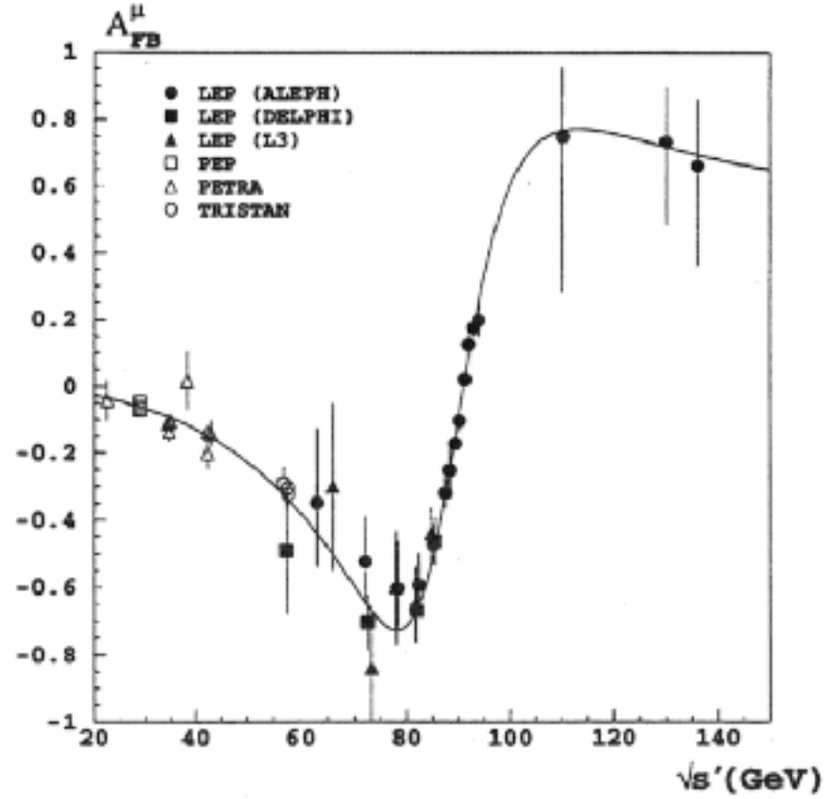


Figure 2: Measured forward-backward asymmetries of muon-pair production compared with the model independent fit results.

CONCLUSIONS

The developments in machine learning and AI go way beyond improved classifiers and have the potential to revolutionize high energy & nuclear physics

- likelihood-free inference and generative models are two particularly exciting areas

Our understanding of how to leverage our prior physics knowledge while letting machine learning do what it's good at is maturing.

- ability to inject and extract physics knowledge from models
- exploit hierarchical structure of events
- robustness to systematic uncertainty in simulation; training on data

Harnessing the full potential of these techniques will require deep integration into our software and changes to our computing models

Backup / Reference

RESOURCES

HEP-ML resources

- https://github.com/iml-wg/HEP-ML-Resources

ML for Jet Physics workshop:

- https://indico.physics.lbl.gov/indico/event/546/timetable/#20171211

Lectures on Physics, Statistics, and Machine Learning

- 288 slides!
- https://agenda.irmp.ucl.ac.be/conferenceDisplay.py?confId=2658

Deep Learning for Physical Sciences workshop at NIPS

- http://dl4physicalsciences.github.io

STATISTICAL TASKS & LEARNING PARADIGMS

Statistical Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

STATISTICAL TASKS & LEARNING PARADIGMS

Statistical Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

STATISTICAL TASKS & LEARNING PARADIGMS

Statistical Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

STATISTICAL TASKS & LEARNING PARADIGMS

Statistical Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

STATISTICAL TASKS & LEARNING PARADIGMS

Statistical Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Systematics

SYSTEMATICS

So far training was based on supervised learning algorithms

- training data came from simulation (or some control sample where class labels or class proportions are known)

Probably the biggest worry for the use of machine learning is what if the data used for training does not represent real data

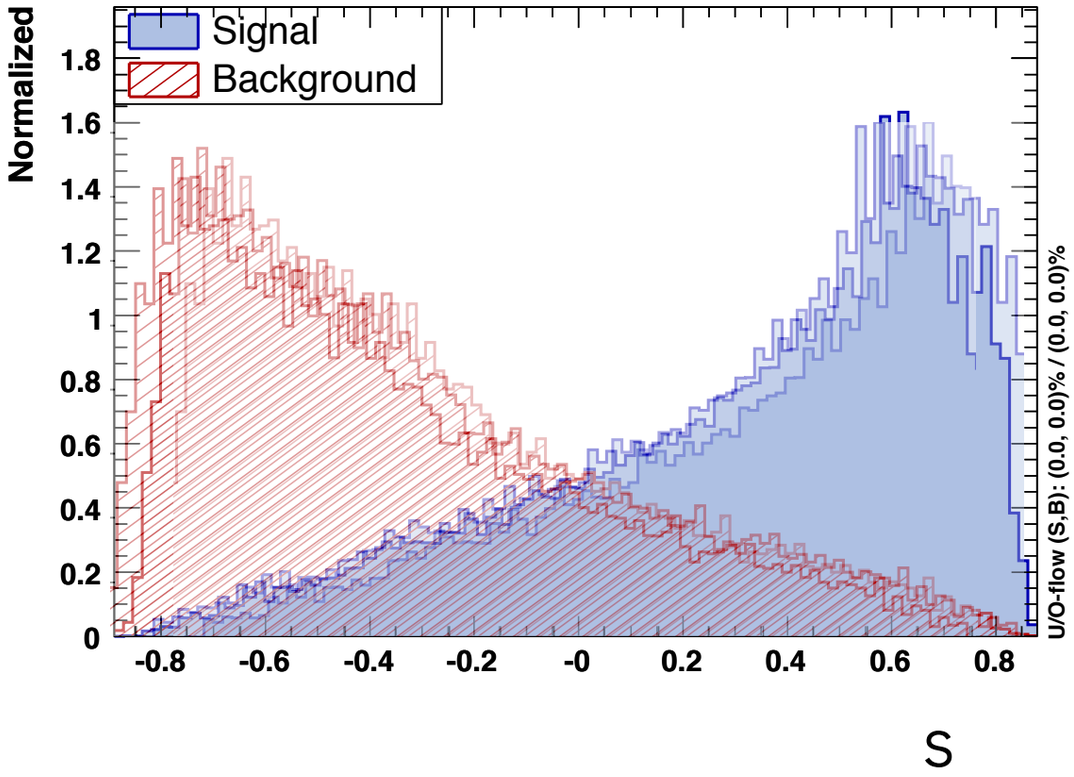
- e.g. we know the simulation isn't perfect
- this is already a problem for "traditional" techniques
- typically we "propagate uncertainty"

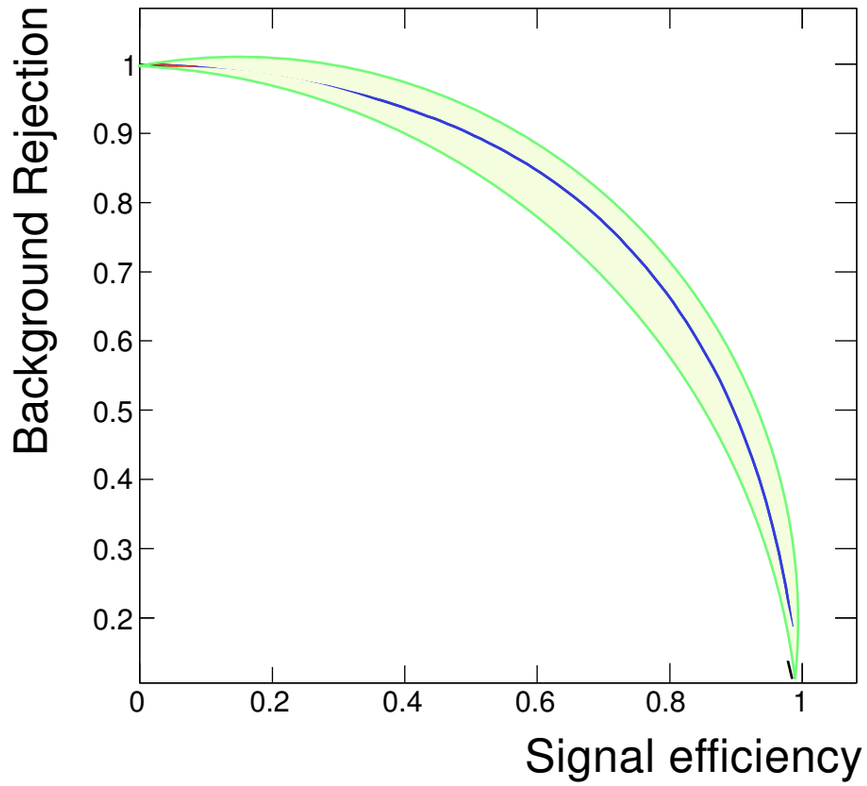
Alternatively, try to learn from real data (limited in applicability)

INCORPORATING SYSTEMATICS INTO CLASSIFIER

We **propagate uncertainty** in the distribution of the **input x** through to the distribution of the **output s(x)**

- Introduce "nuisance parameters" ν corresponding to unknown calibration constants, detector response, etc. Effect of the systematics encoded in $p_0(x|\nu)$
- Typically, the classifier s(x) is trained on data from nominal ν_0
- use systematic variations $p_0(x|\nu)$ & $p_1(x|\nu)$ for input features & propagate through s(x)



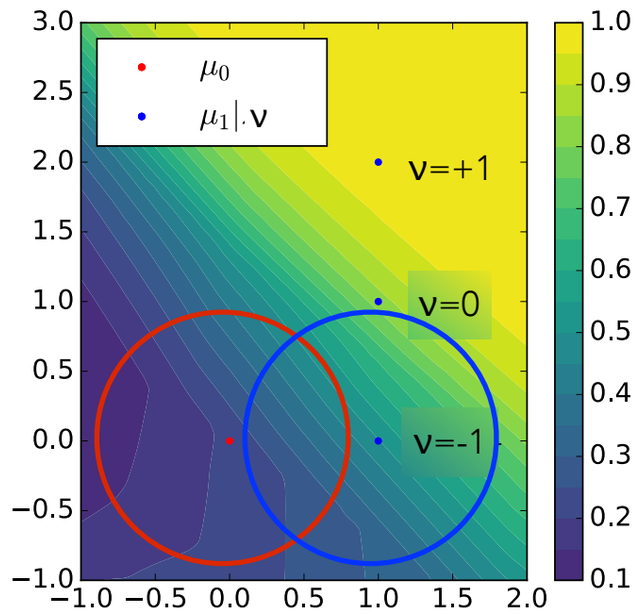


LEARNING TO PIVOT WITH ADVERSARIAL NETWORKS

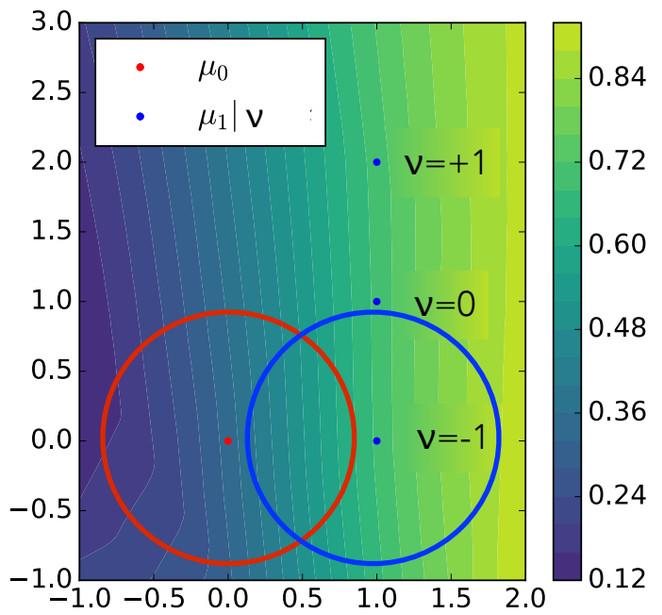
Typically classifier **s(x)** trained to minimize loss **L**.

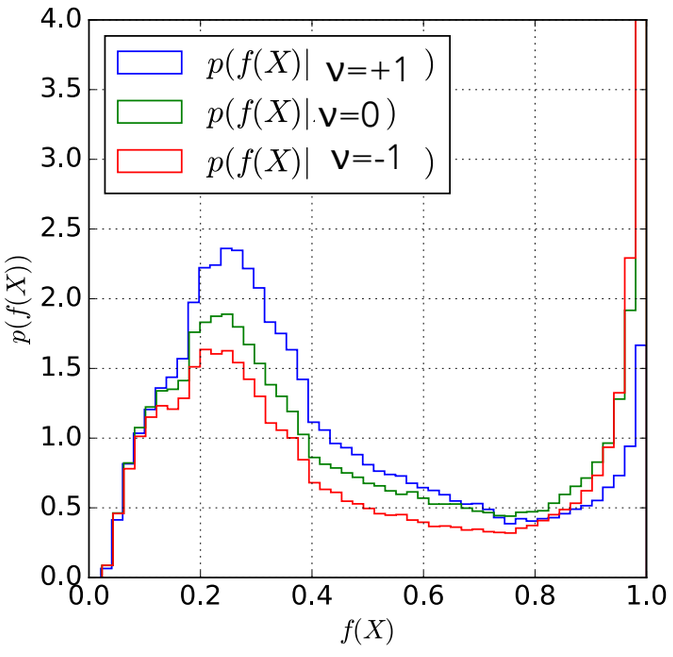
- want classifier output to be insensitive to systematics (nuisance parameter **ν**)
- introduce an **adversarial network** that tries to predict ν based on s.
- Loss is penalized if adversary succeeds
- provides training procedure that allows for **tradeoff** between traditional classification accuracy and **robustness to systematics**

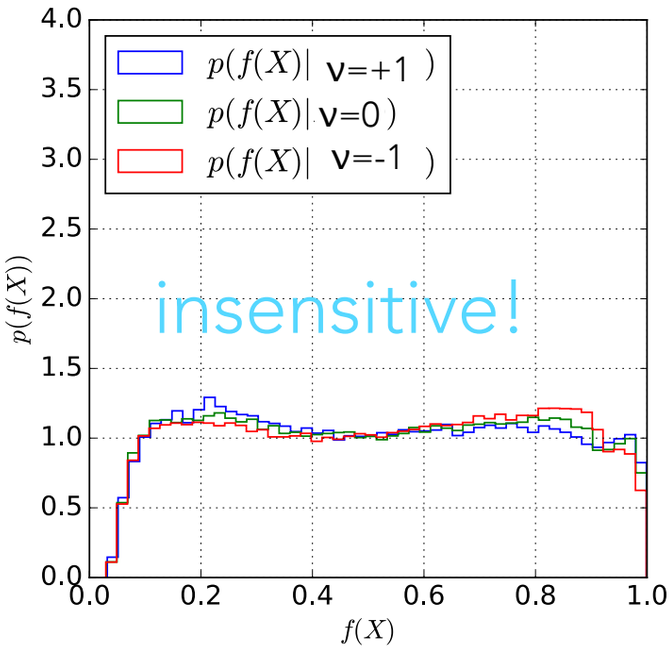
normal training



adversarial training







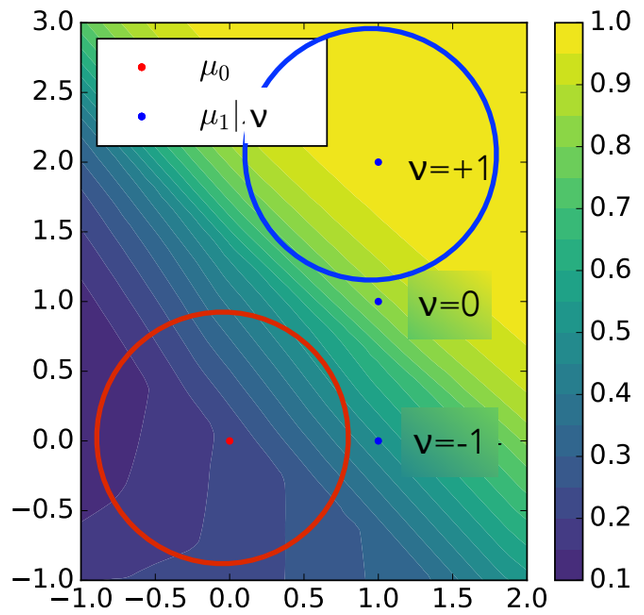
G. Louppe, M. Kagan, K. Cranmer, arXiv:1611.01046

LEARNING TO PIVOT WITH ADVERSARIAL NETWORKS

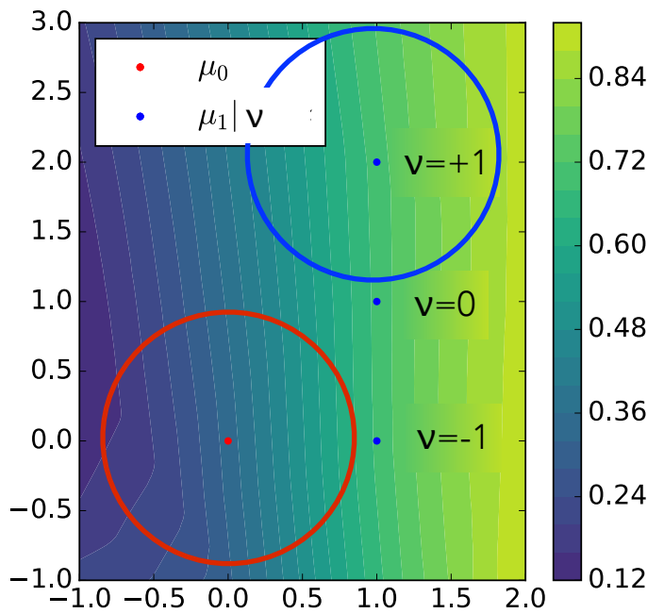
Typically classifier **s(x)** trained to minimize loss **L**.

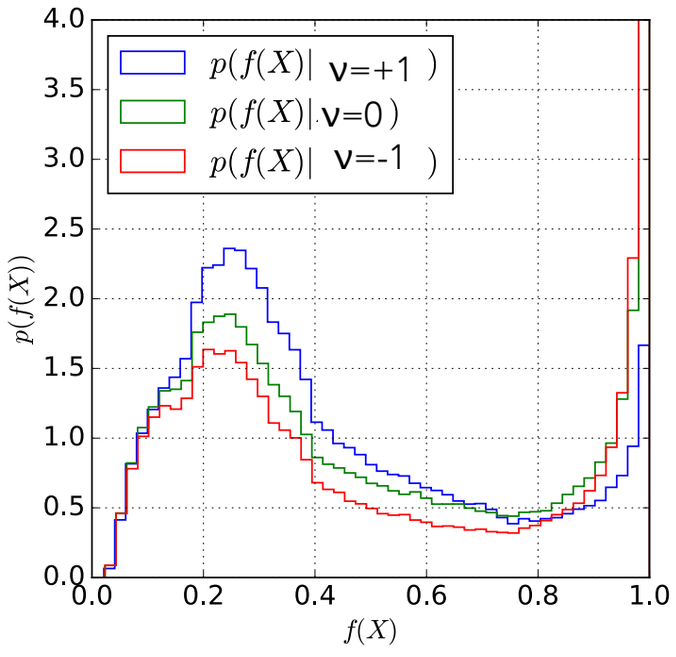
- want classifier output to be insensitive to systematics (nuisance parameter **ν**)
- introduce an **adversarial network** that tries to predict ν based on s.
- Loss is penalized if adversary succeeds
- provides training procedure that allows for **tradeoff** between traditional classification accuracy and **robustness to systematics**

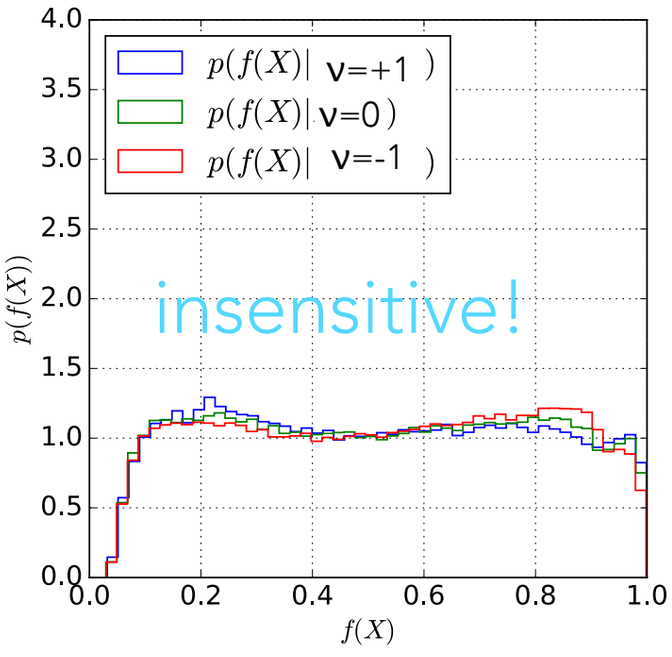
normal training



adversarial training

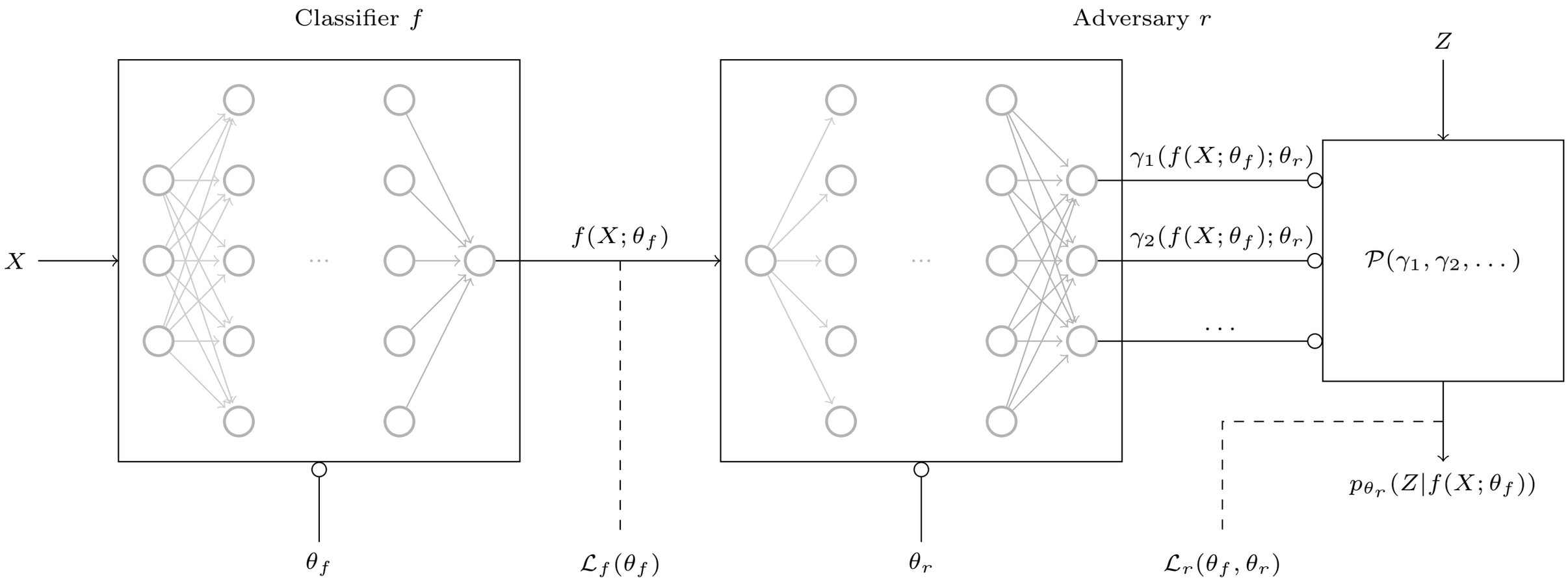






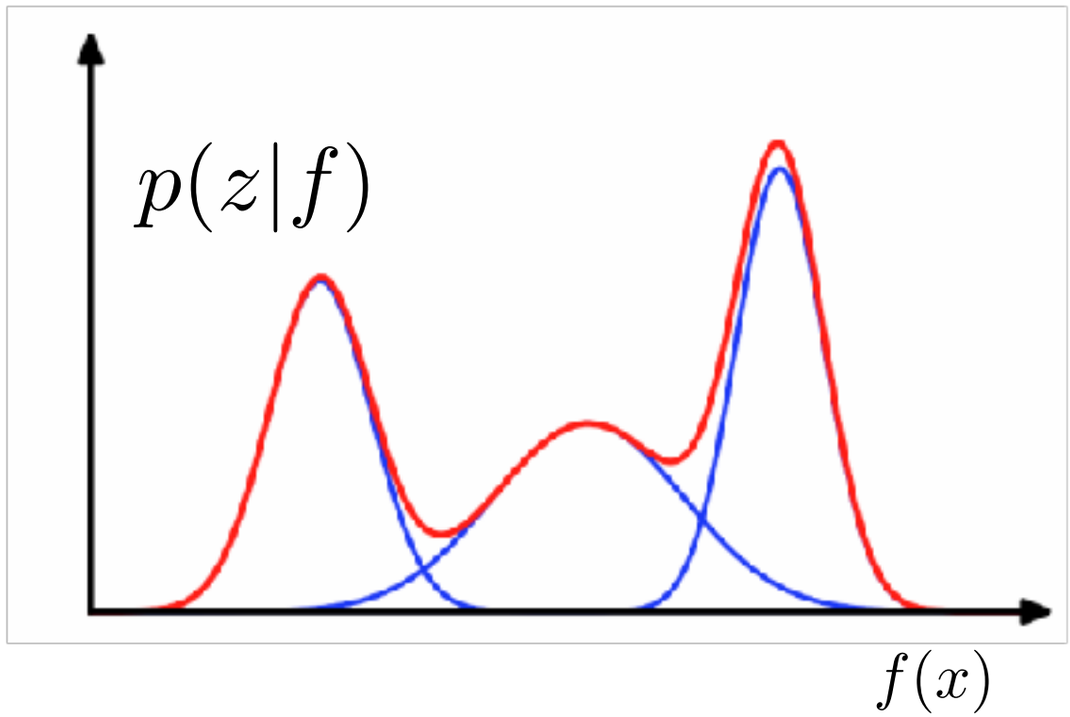
G. Louppe, M. Kagan, K. Cranmer, arXiv:1611.01046

THE ADVERSARIAL MODEL



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, …



AN EXAMPLE

Technique allows us to tune **λ**, the tradeoff between classification power and robustness to systematic uncertainty

An example:

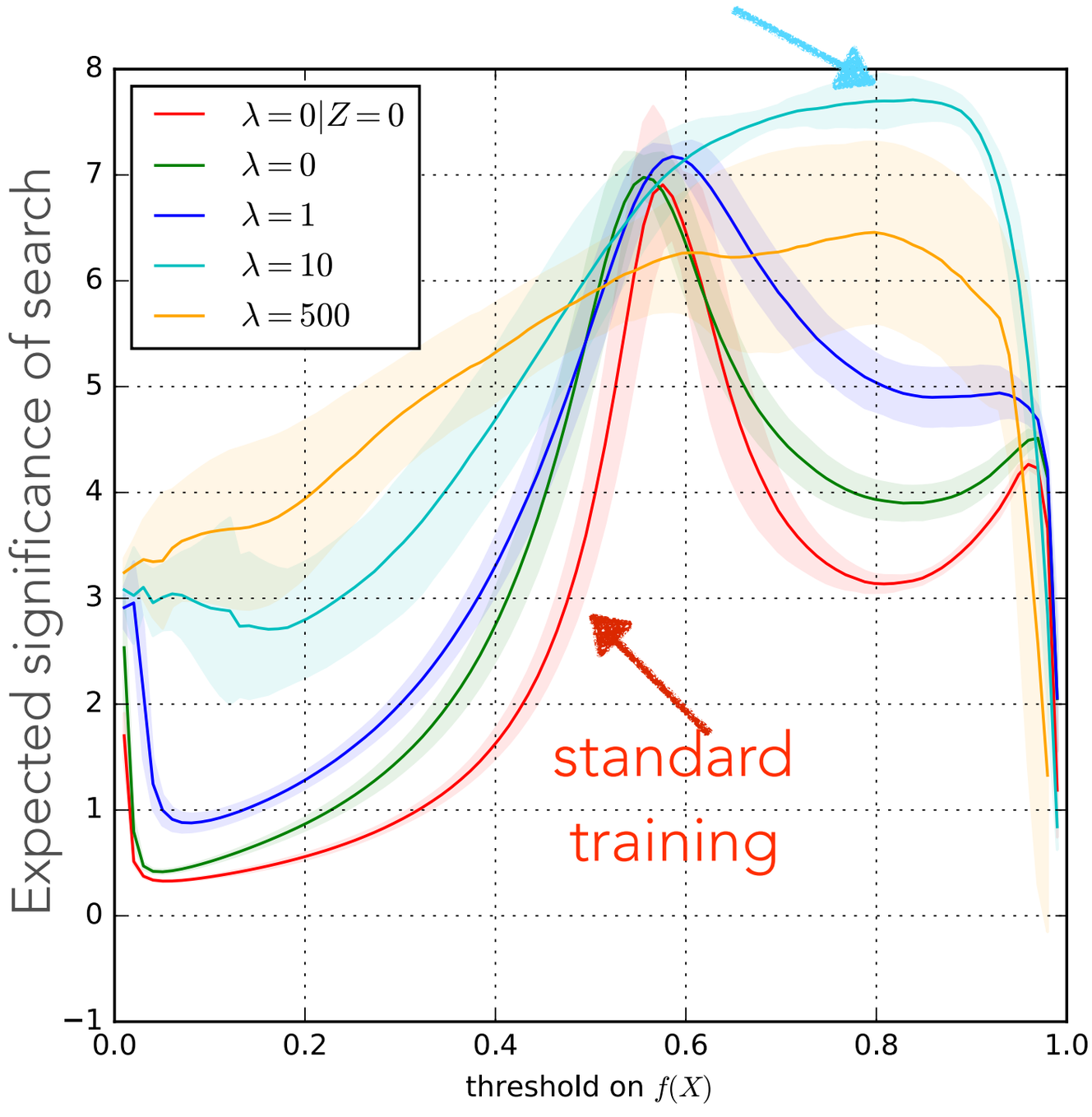
background: 1000 QCD jets

signal: 100 boosted W's

Train W vs. QCD classifier

Pileup as source of uncertainty

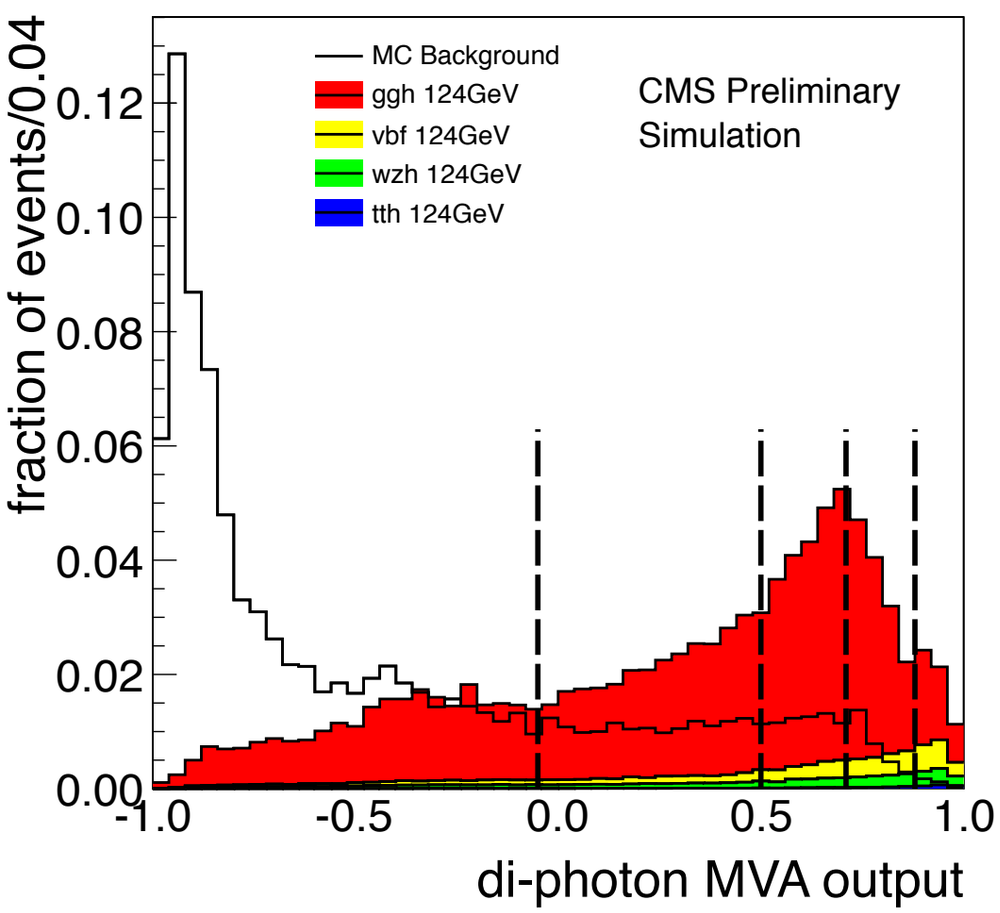
Simple cut-and-count analysis with background uncertainty.

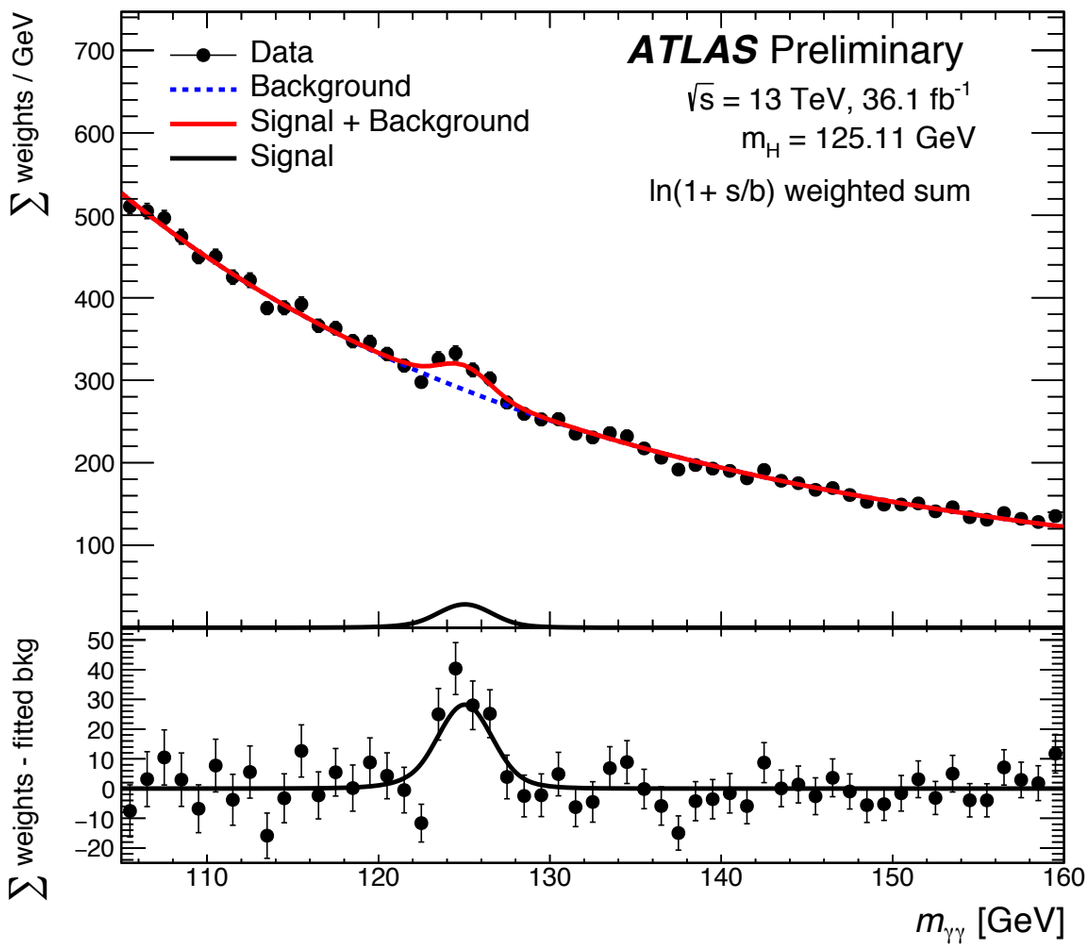


BUMP HUNTING WITH ML

Even if we are comfortable with systematics, often physicists want to base analysis on a variable that is "interpretable"

- Can we combine the two? Use a classifier to improve signal/background but still base search on a simple bump hunt?



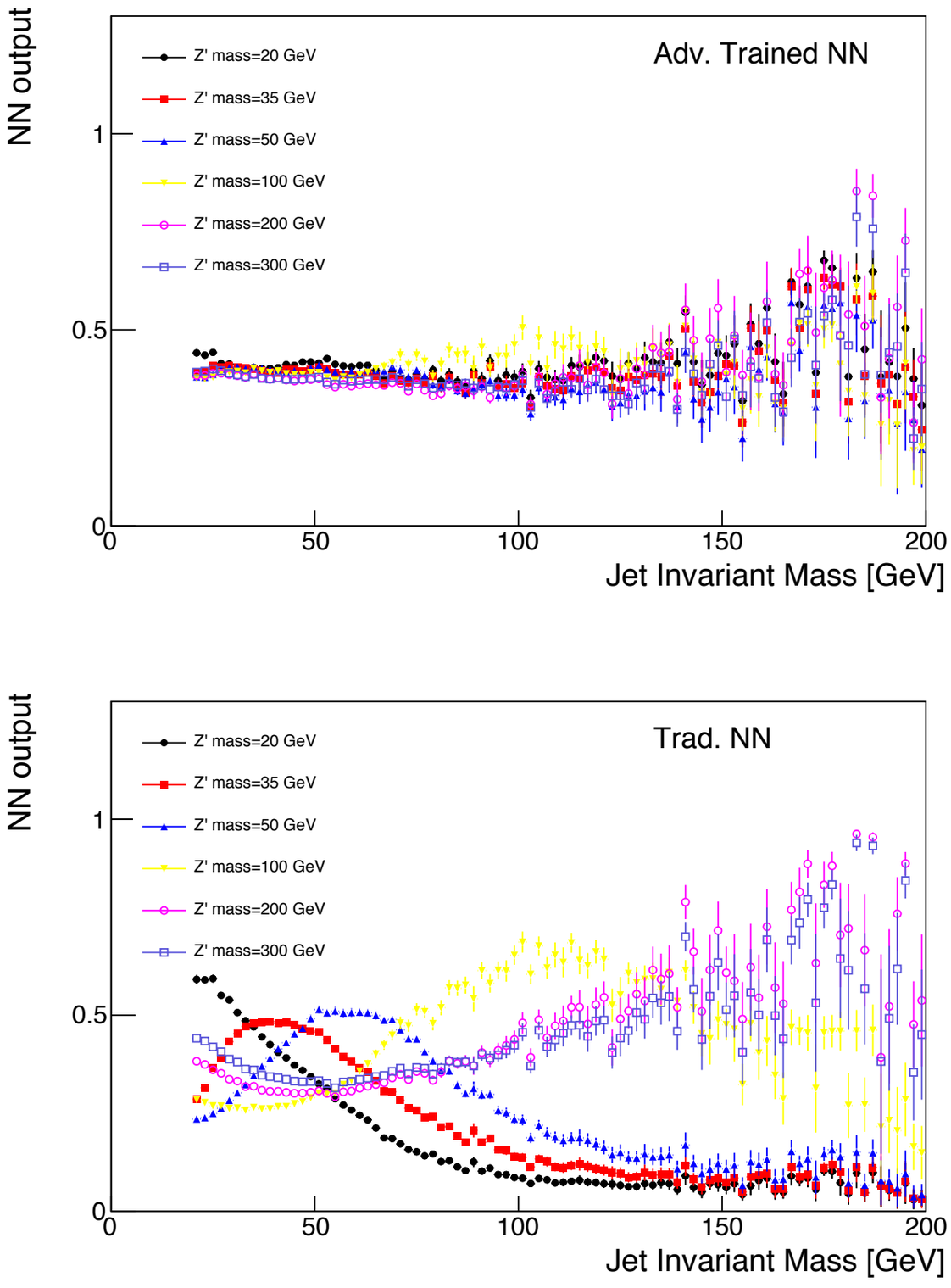


DECORRELATED TAGGERS

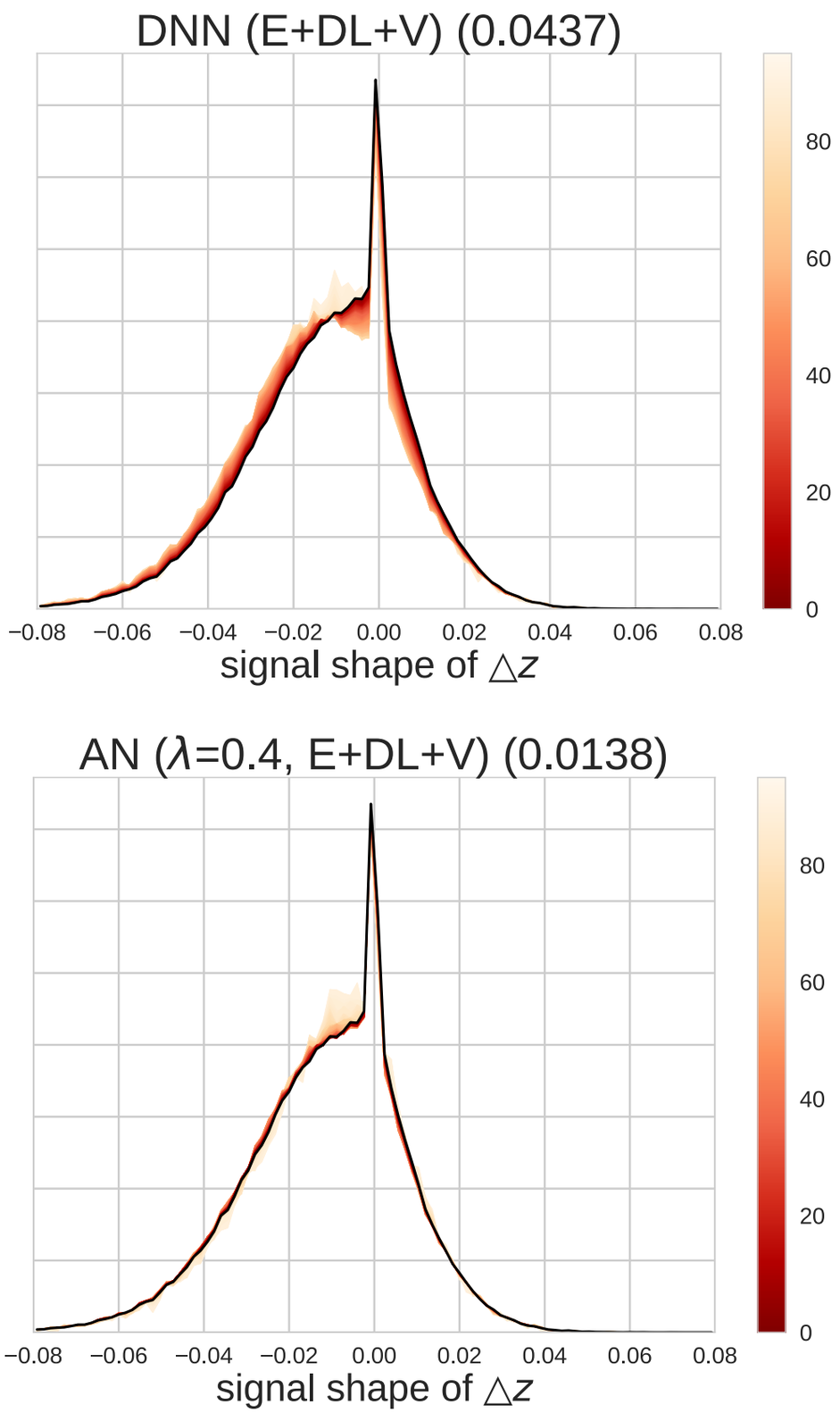
K.C, J. Pavez, and G. Louppe, arXiv:1506.02169
P. Baldi, K.C, T. Faucett, P. Sadowski, D. Whiteson arXiv:1601.07913
G. Louppe, M. Kagan, K.C, arXiv:1611.01046
Shimmin, et. al. arXiv:1703.03507

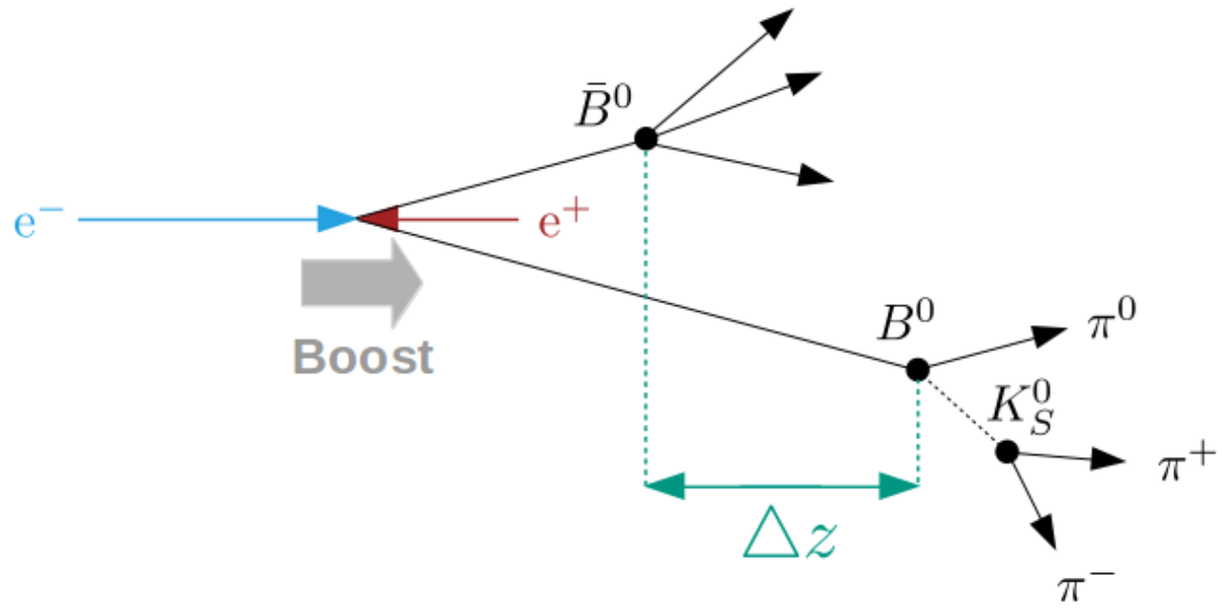
Adversarial approach of "Learning to Pivot" can also be used to train a classifier that is "decorrelated" to some other variable.

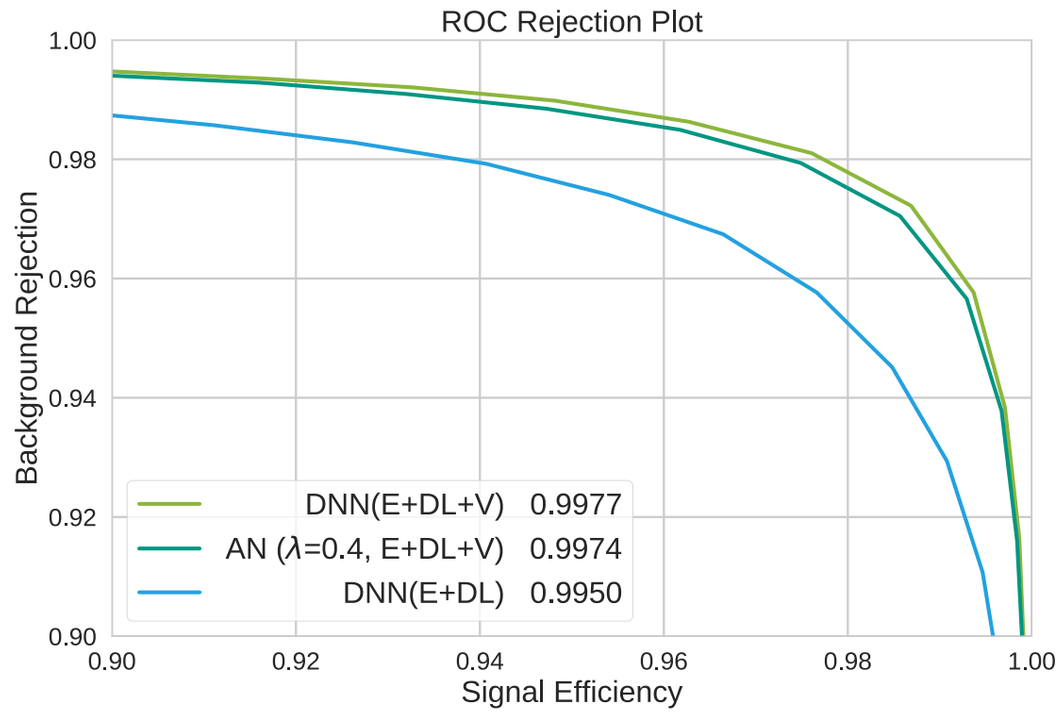
- want jet taggers that are decorrelated with jet invariant mass
- so that analysis can still search for a bump using jet invariant mass
- avoids sculpting background



DECORRELATION IN BELLE II







Parameter Estimation

Parameter Estimation

- Let us focus on (1) first. $\{\mathbf{X}^1, \ldots, \mathbf{X}^L\} \sim p^* \text{ iid.}$
- Suppose $p^* = p_{\theta^*} \text{ for some } \theta^*.$
- Two main approaches for parameter estimation:
  - Maximum Likelihood Estimation:

$$E(\theta) = \log p(\{\mathbf{X}^1, \ldots, \mathbf{X}^L\} \mid \theta) = \sum_{l<L} \log p(\mathbf{X}^l \mid \theta)$$

$$\hat{\theta}_{MLE} = \arg\max_{\theta} E(\theta)$$

  - Under appropriate assumptions, $\hat{\theta}_{MLE}$ is
    - consistent (as sample size grows, $\hat{\theta}_{MLE} \to \theta^* \text{ (in probability)}$
    - asymptotically efficient (no other consistent estimator has lower asymptotic mean-squared error).
  - However, in general this estimation is computationally intractable.

If we could approximate the conditional density p(x|θ), then we could

- approximate the maximum likelihood estimate

- construct frequentist confidence intervals or Bayesian credible intervals

Cramér-Rao Bound

The minimum variance bound on an estimator is given by the Cramér-Rao inequality:

- simple univariate case:

$$\mathrm{Var}[\hat{\theta}|\theta] = E[(\hat{\theta} - E[\theta|\theta])^2]\,|\theta]$$

- For an unbiased estimator the Cramér-Rao bound states

$$\mathrm{Var}[\hat{\theta}|\theta] \geq \frac{1}{I(\theta)}$$

- where $I(\theta)$ is the Fisher information

$$(\mathcal{I}\,(\theta))_{i,j} = \mathrm{E}\left[\frac{\partial}{\partial\theta_i}\ln f(X;\theta)\frac{\partial}{\partial\theta_j}\ln f(X;\theta)\middle|\theta\right].$$

- General form for multiple parameters:

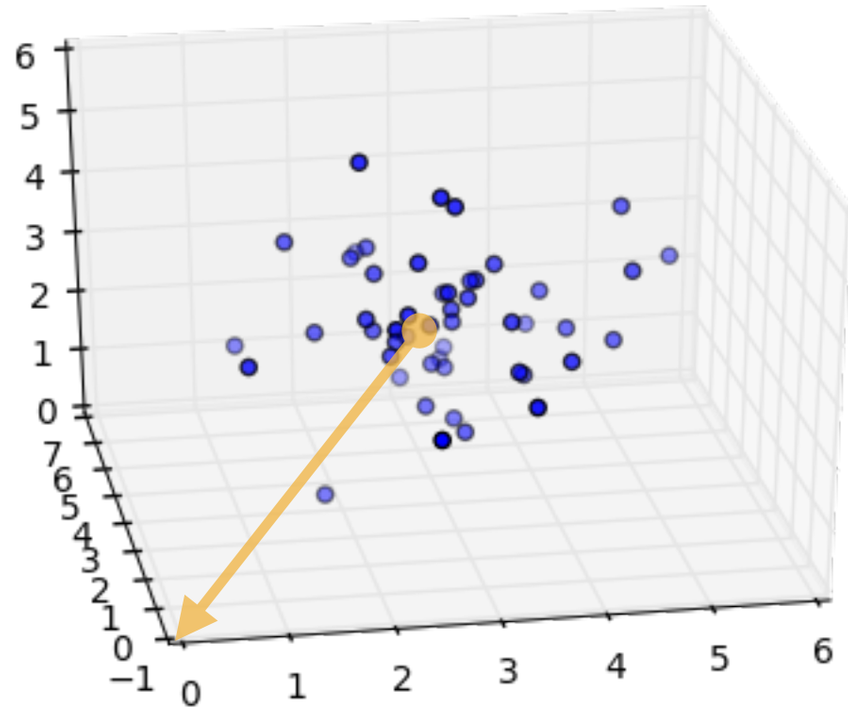
$$\mathrm{cov}[\hat{\theta}|\theta]_{ij} \geq I^{-1}_{ij}(\theta)$$

Maximum Likelihood Estimators *asymptotically* reach this bound

JAMES-STEIN ESTIMATOR

Consider a standard multivariate Gaussian distribution for $\vec{x}$ in n dimensions centered around $\vec{\mu}$

$$f(\vec{x}|\vec{\mu}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x_i - \mu_i)^2}{2}\right) .$$



Goal: minimize mean-squared error

$$MSE[\hat{\vec{\mu}}] = E[||\hat{\vec{\mu}} - \vec{\mu}||^2])$$

MLE (unbiased)

$$\hat{\vec{\mu}}_{MLE} = \overline{x} = \frac{1}{m} \sum_{j=1}^{m} \vec{x}_j$$

James-Stein (weird)

$$\hat{\mu}_{JS} = \left(1 - \frac{n-2}{||\bar{x}||^2}\right) \bar{x}$$

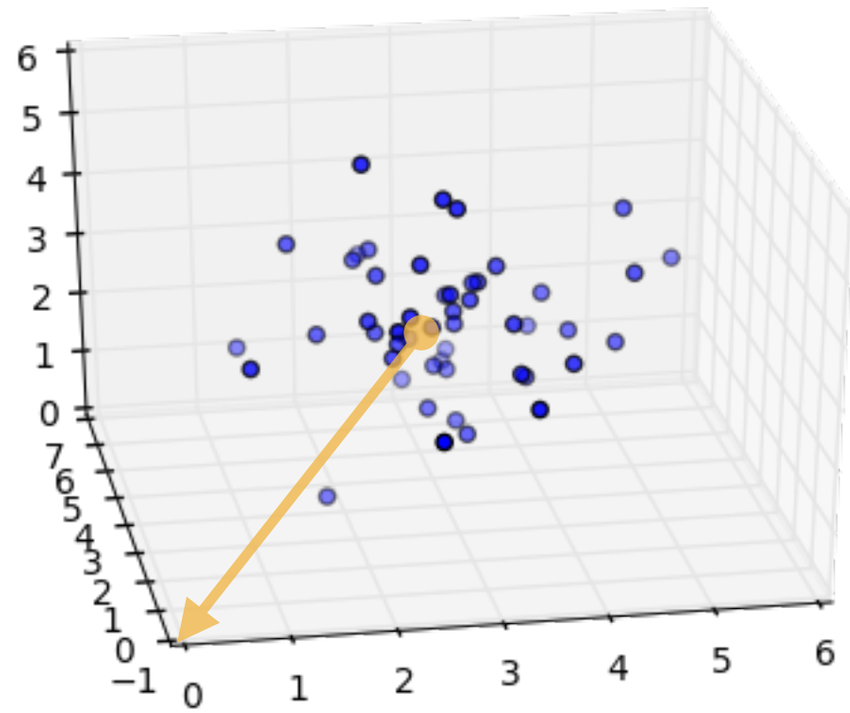
JAMES-STEIN ESTIMATOR

The James-Stein estimator seems like a horrible suggestion

$$\hat{\mu}_{JS} = \left(1 - \frac{n-2}{||\bar{x}||^2}\right)\bar{x}$$

- clearly biased (MLE is not)
- shifts towards origin is not translationally invariant

$x \rightarrow x' = x+\Delta$

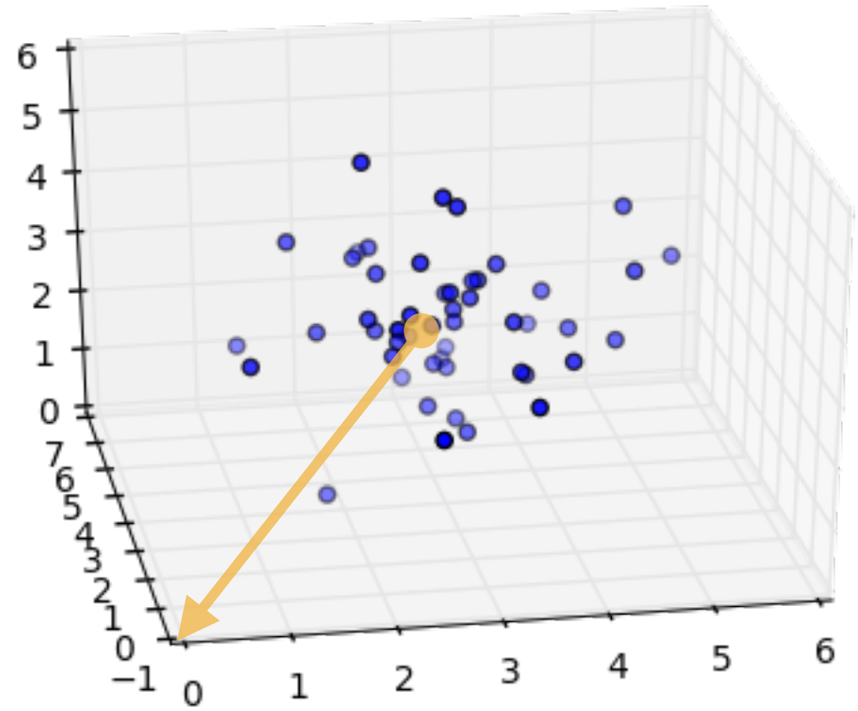


JAMES-STEIN ESTIMATOR

The James-Stein estimator seems like a horrible suggestion

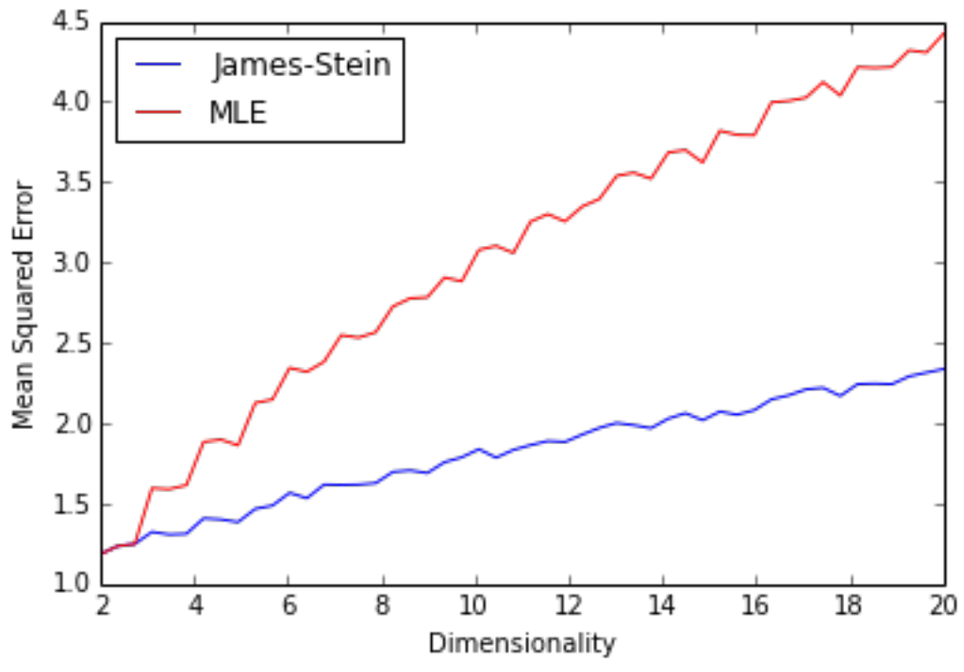
$$\hat{\mu}_{JS} = \left(1 - \frac{n-2}{||\bar{x}||^2}\right)\bar{x}$$

- clearly biased (MLE is not)
- shifts towards origin is not translationally invariant
  $x \rightarrow x' = x+\Delta$



Yet, it has smaller mean squared error than MLE for n>2 !

- it "dominates" the MLE



BIAS/VARIANCE TRADEOFF

We introduced Bias and Variance of estimators

$$\mathrm{Var}[\hat{\mu}|\mu] = E[(\hat{\mu} - E[\mu|\mu])^2]\,|\mu]$$

Most physicist are allergic to the idea of a biased estimator

- try to find unbiased estimator with smallest variance
- hence importance of Cramér-Rao bound

But what if we just want to minimize the mean-squared error?

$$MSE[\hat{\mu}|\mu] = E[(\hat{\mu} - \mu)^2]\,|\mu]$$

it decomposes like this

$$MSE[\hat{\mu}|\mu] = \mathrm{Var}[\hat{\mu}|\mu] + (\mathrm{Bias}[\hat{\mu}|\mu])^2$$

So it encodes some relative weight to bias and variance. Think harder!

Statistical Decision Theory

STATISTICAL DECISION THEORY IN 1 SLIDE

Θ - States of nature; X - possible observations; A - action to be taken

f(x|θ) - statistical model; π(θ) - prior

δ: X → A - **decision rule** (take some action based on observation)

L: Θ x A → $\mathbb{R}$ - **loss function**, real-valued function true parameter and action

$R(\theta,\delta) = E_{f(x|\theta)}[L(\theta, \delta)]$ - **risk**

- A decision δ* rule **dominates** a decision rule δ if and only if $R(\theta,\delta^*) \leq R(\theta,\delta)$ for all θ, and the inequality is strict for some θ.
- A decision rule is **admissible** if and only if no other rule dominates it; otherwise it is inadmissible

$r(\pi, \delta) = E_{\pi(\theta)}[R(\theta,\delta)]$ - **Bayes risk** (expectation over θ w.r.t. prior and possible observations)

$\rho(\pi, \delta \mid x) = E_{\pi(\theta|x)}[L(\theta,\delta(x))]$ - **expected loss** (expectation over θ w.r.t. posterior $\pi(\theta|x)$)

- δ' is a (generalized) Bayes rule if it minimizes the expected loss
- under mild conditions every admissible rule is a (generalized) Bayes rule (**with respect to some prior** —possibly an improper one—that favors distributions where that rule achieves low risk). Thus, in frequentist decision theory it is sufficient to consider only (generalized) Bayes rules.
- Conversely, while Bayes rules with respect to proper priors are virtually always admissible, generalized Bayes rules corresponding to improper priors need not yield admissible procedures. Stein's example is one such famous situation.

The longest highest bridge in the world

- Optimality theory: Data X. Model $f(x|\theta), \theta \in \Theta$.
- Decision problem: observe X, make decision $d(X)$.
- Lose $L(d(X), \theta)$ – real valued.
- Judge quality of $d(X)$ by long run average risk:

$$R(d, \theta) = \langle L(d(X), \theta \rangle_\theta = \mathrm{E}\left[L(d(X), \theta|\theta\right].$$

- Key idea: admissibility.
- Procedure d_1 is better than d_2 if, for *all* θ,

$$R(d_1, \theta) < R(d_2, \theta).$$

- We call d_2 *inadmissible*.

Admissibility and Bayes

Theorem

Every admissible procedure is Bayes.

Theorem

Every Bayes procedure is admissible

Written separately because neither is quite right.
But meaning is – sensible procedures need to be Bayes.
Not always an easy restriction to impose – but wise, in my view, to remember.

Famous example – Neyman-Pearson lemma

- Data X with density f_0 or f_1.
- Decision: observe X guess which density. Hypothesis testing.
- Loss: 1 if wrong, 0 if right.
- Risk is

$$(P_0(\text{Reject}), P_1(\text{Accept}))$$

- Neyman Pearson say minimize second component subject to constraint on first.

Implied Priors

- Langrange multipliers. Minimize

$$P_1(\text{Accept}) + \lambda P_0(\text{Reject}) = \beta + \lambda\alpha.$$

- Same as Bayes for prior $P(f_1 \text{ true}) = 1/(1+\lambda)$.
- Then adjust prior (λ) to find Bayes procedure which satisfies constraint.
- Notice that $\lambda/(1+\lambda) = P(H_o)$.
- Procedure implies (at least one) prior.

Generalization

$$R(T_i) \leqslant R_{emp}(T_i) + \frac{\ln N - \ln \eta}{l}\left(1 + \sqrt{1 + \frac{2R_{emp}(T_i)l}{\ln N - \ln \eta}}\right)$$

ALL YOUR BAYES ARE BELONG TO US

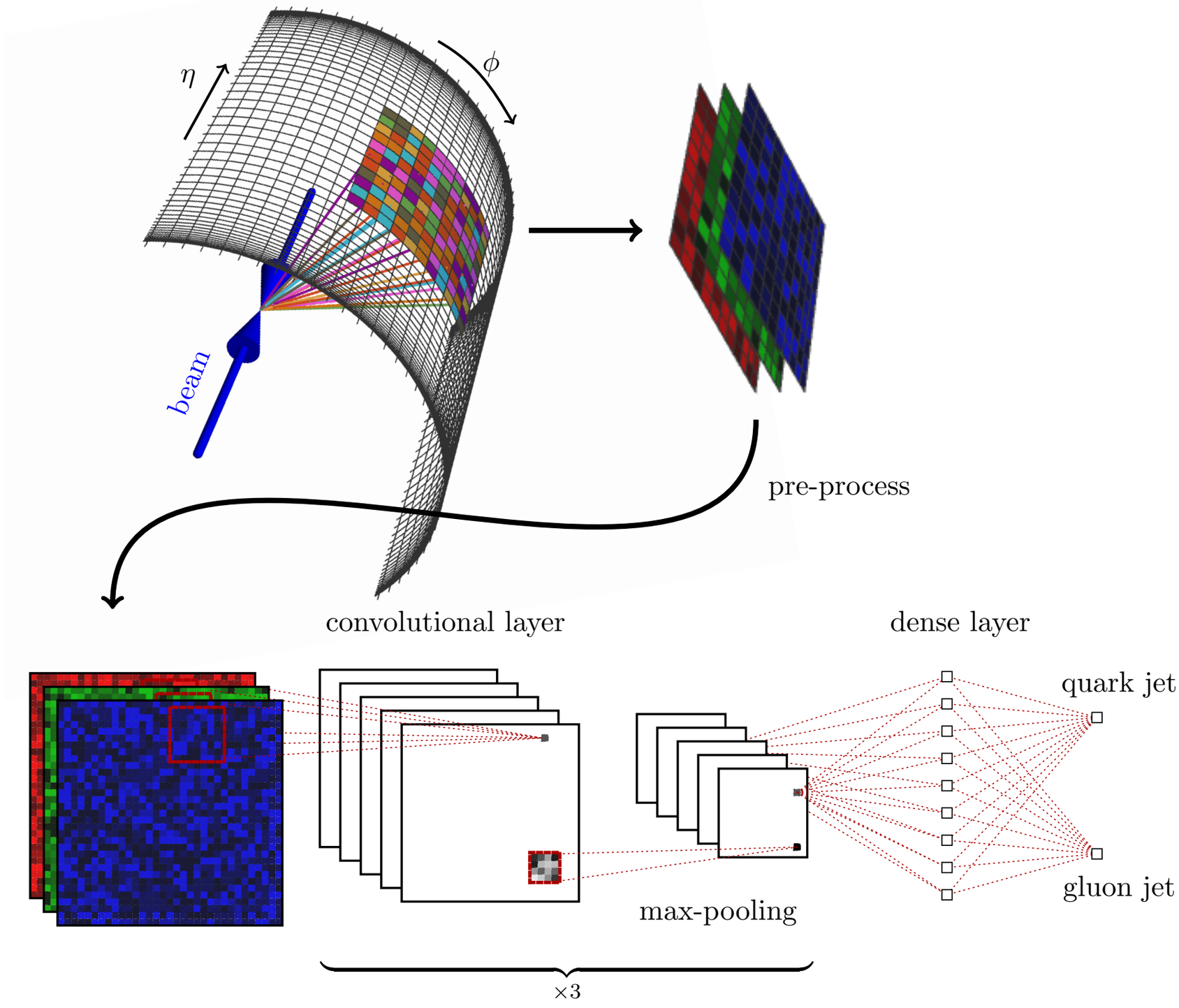
Physics-Aware Machine Learning

JET IMAGES

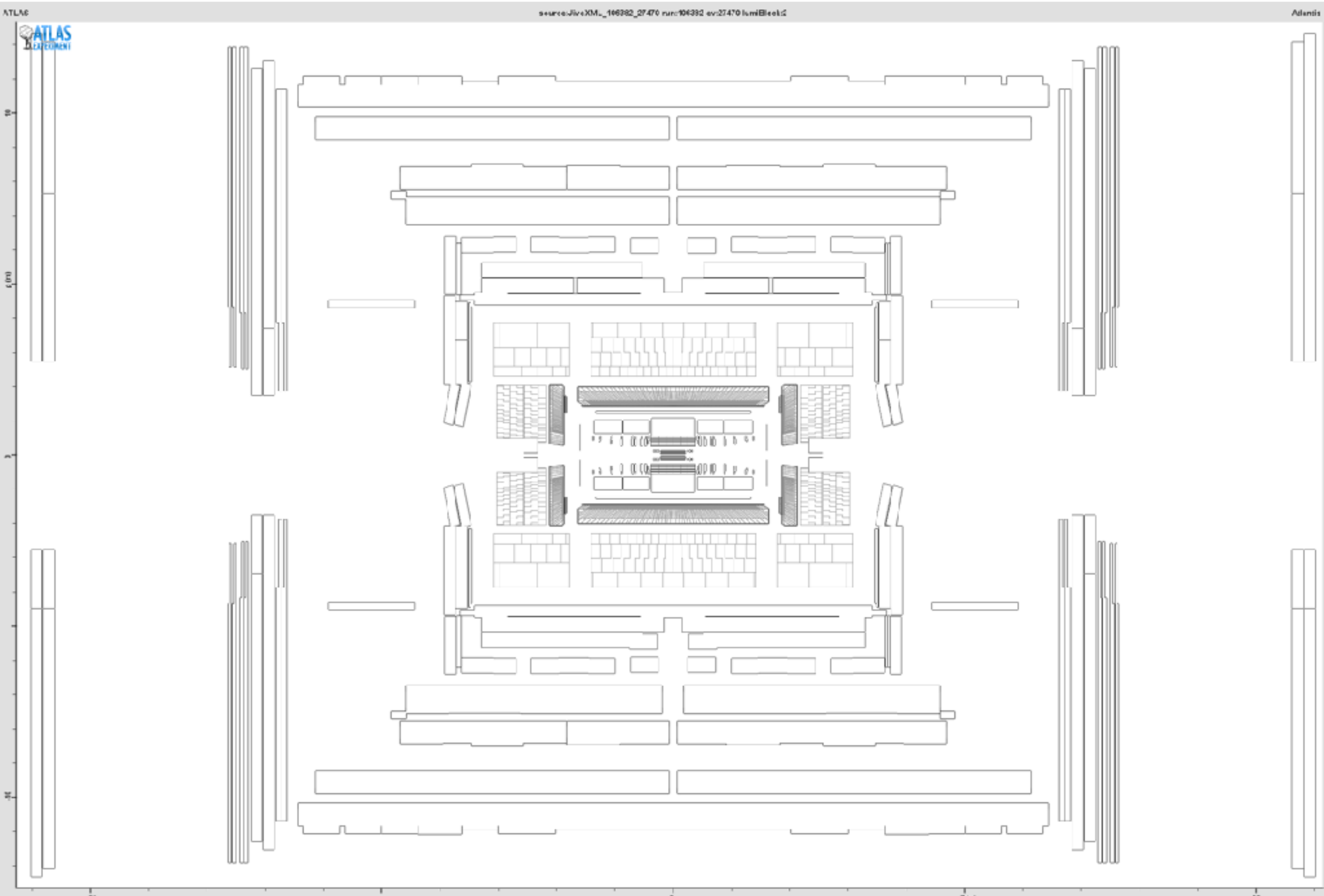
image: Komiske, Metodiev, Schwartz arxiv:1612.01551
Oliveira, et. al arXiv:1511.05190
Whiteson, et al arXiv:1603.09349
Barnard, et al arXiv:1609.00607

Jet = Collimated spray of particles coming from quark / gluon

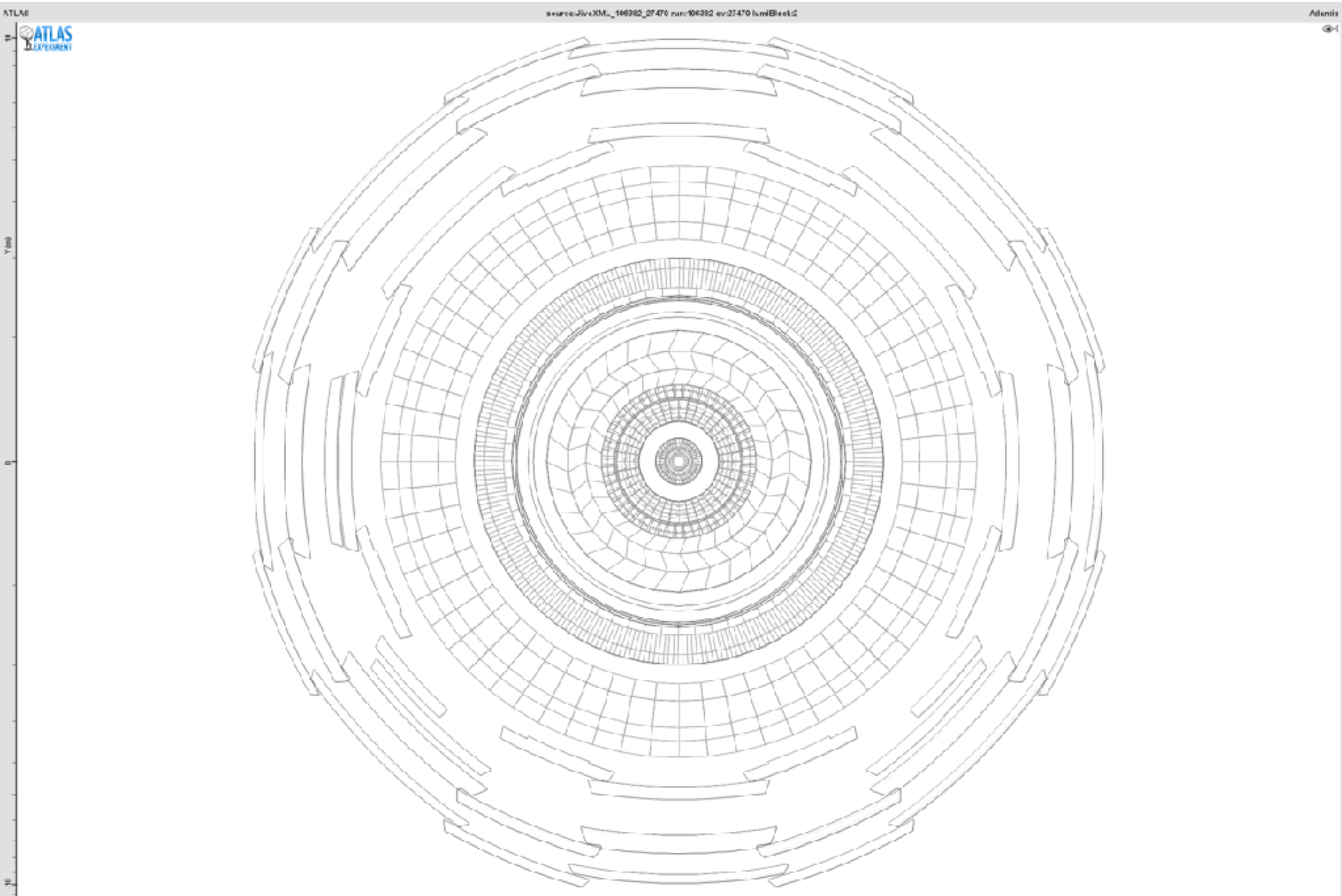
"We supplement this construction by adding color to the images, with **red, green and blue** intensities given by the transverse momentum in **charged** particles, transverse momentum in **neutral** particles, and pixel-level charged particle **counts**."



NON-UNIFORM GEOMETRY



NON-UNIFORM GEOMETRY



HOW CAN WE IMPROVE?

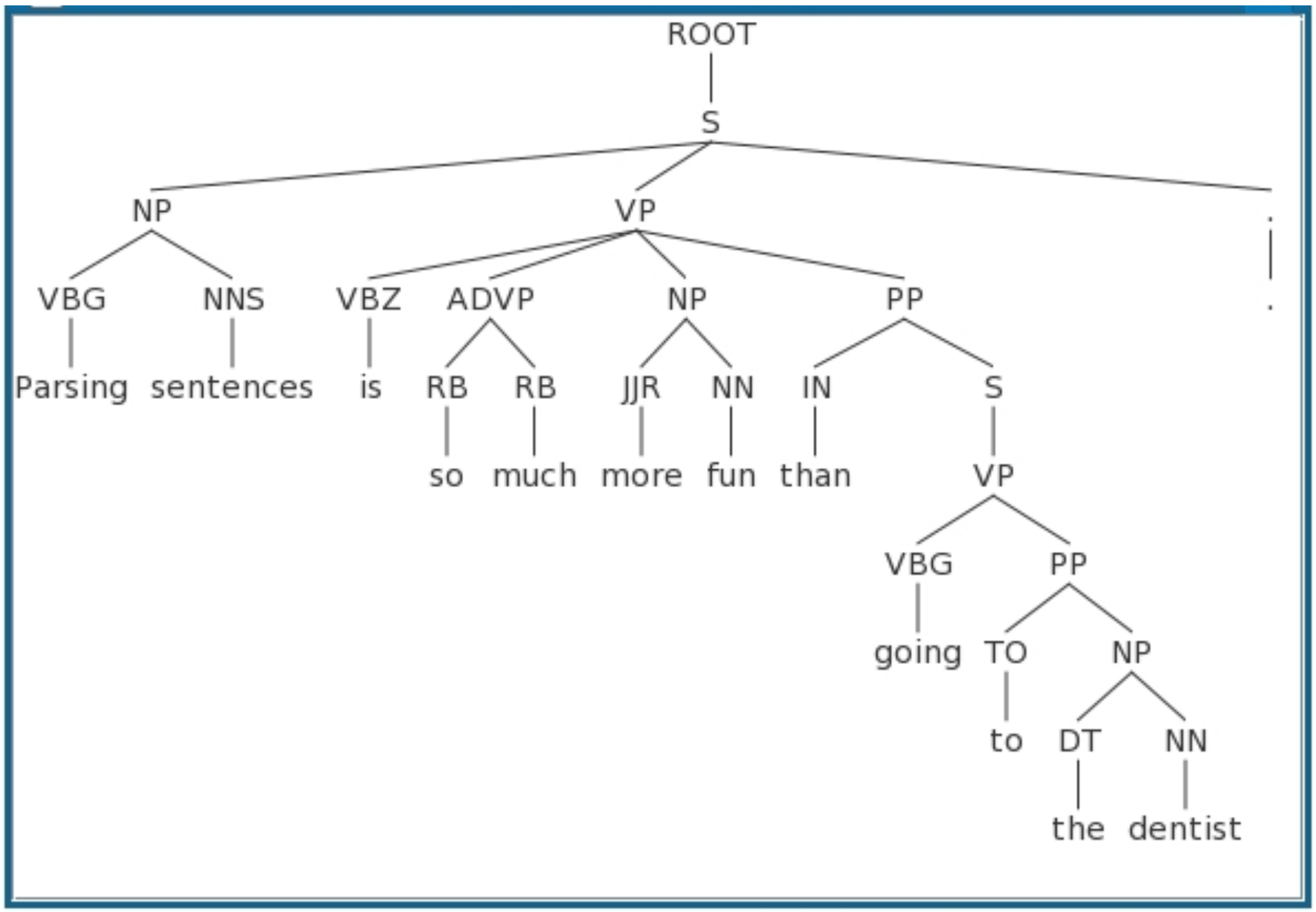
Image based approaches are doing well, but….

- would be nice to be able to work with a variable length input
  - avoid pre-processing into a regular-grid (eg. non-uniform calorimeters)
  - avoid representing empty pixels (sparse input)
- would be nice if classifier had nice theoretical properties
  - infrared & collinear safety, robustness to pileup, etc.
- would be nice to be more data efficient, most image-based networks use a LOT of training data.

FROM IMAGES TO SENTENCES

Recursive Neural Networks showing great performance for Natural Language Processing tasks

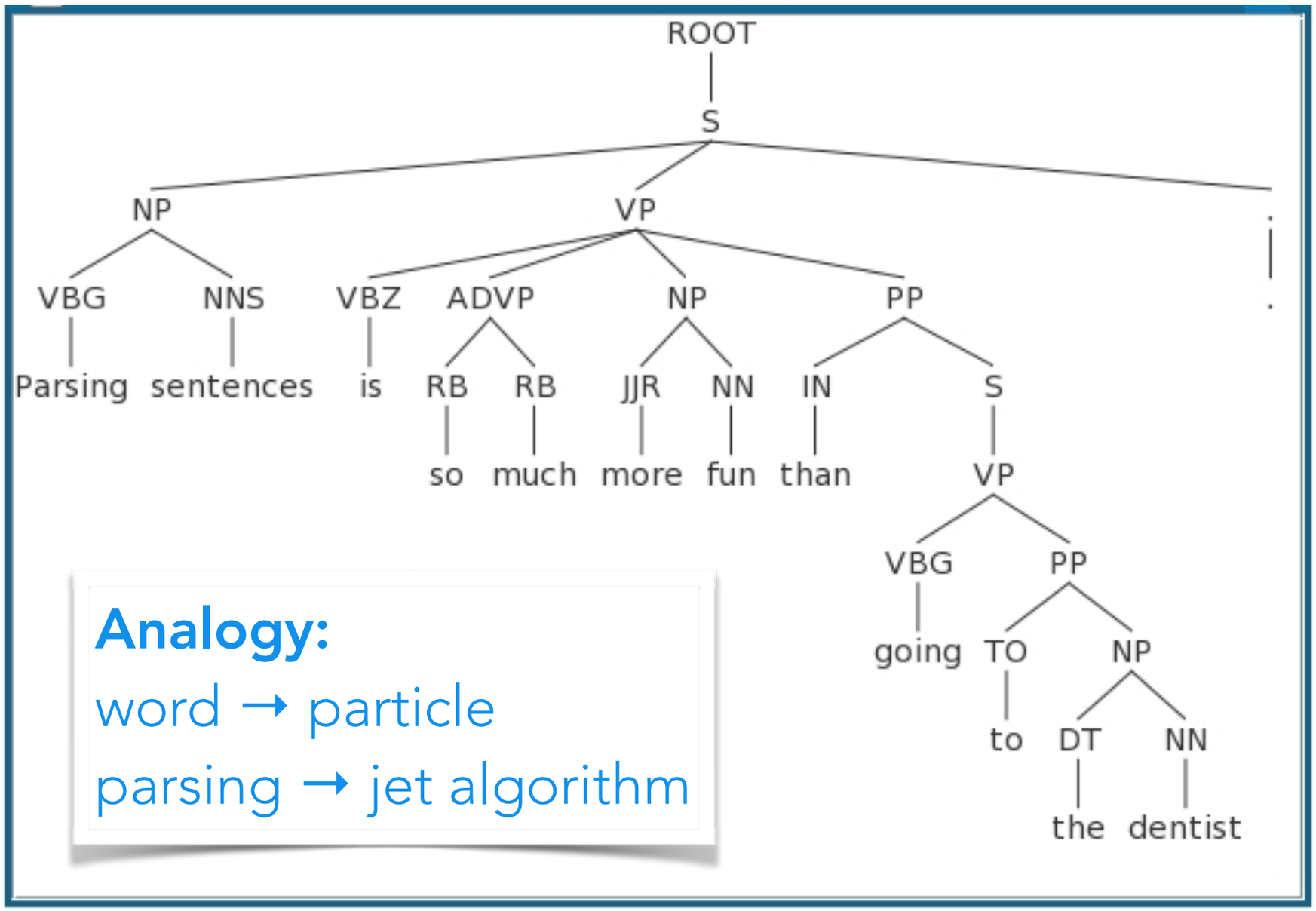
- neural network's topology given by parsing of sentence!



FROM IMAGES TO SENTENCES

Recursive Neural Networks showing great performance for Natural Language Processing tasks

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

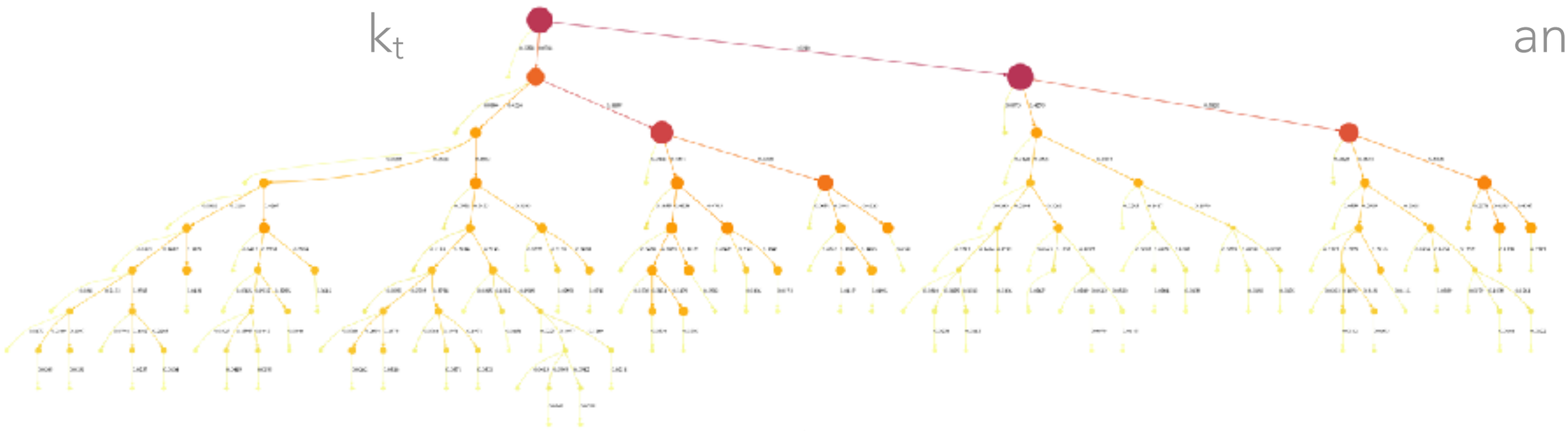


Analogy:

word → particle

parsing → jet algorithm

QCD-INSPIRED RECURSIVE NEURAL NETWORKS

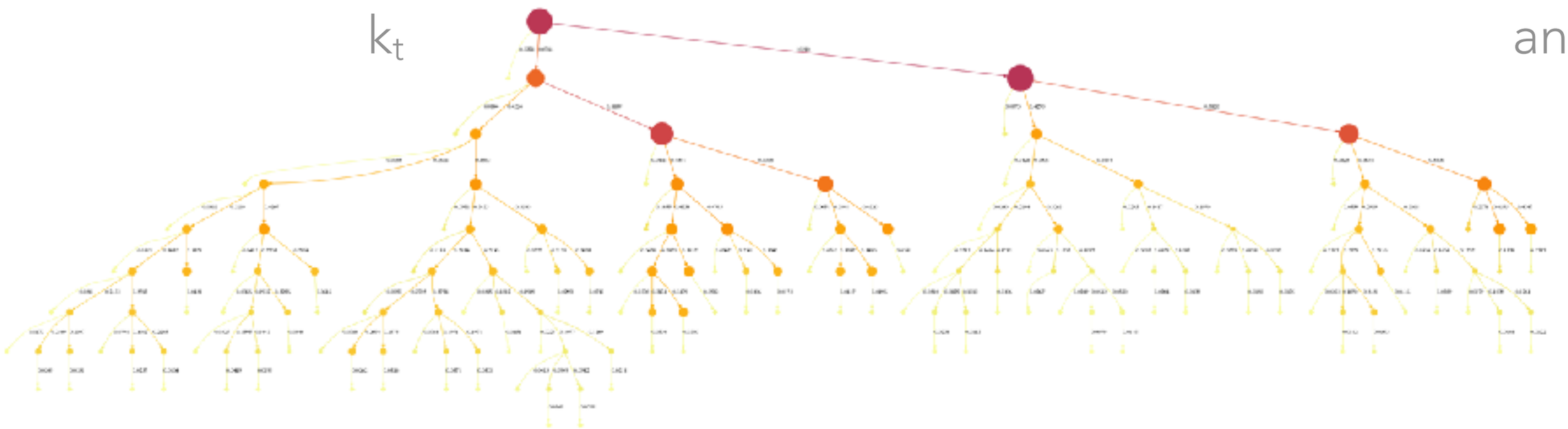


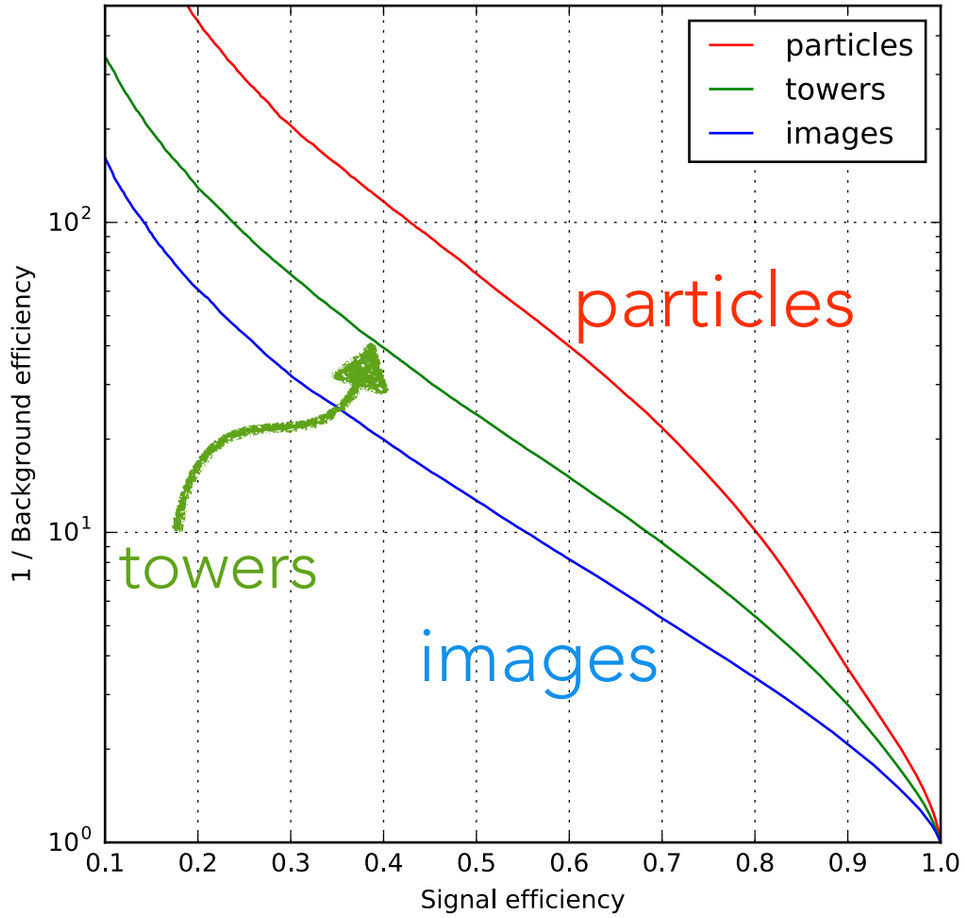
Work with Gilles Louppe, Kyunghyun Cho, Cyril Becot

- Use sequential recombination jet algorithms to provide network topology (**on a per-jet basis**)
- path towards ML models with good theoretical 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



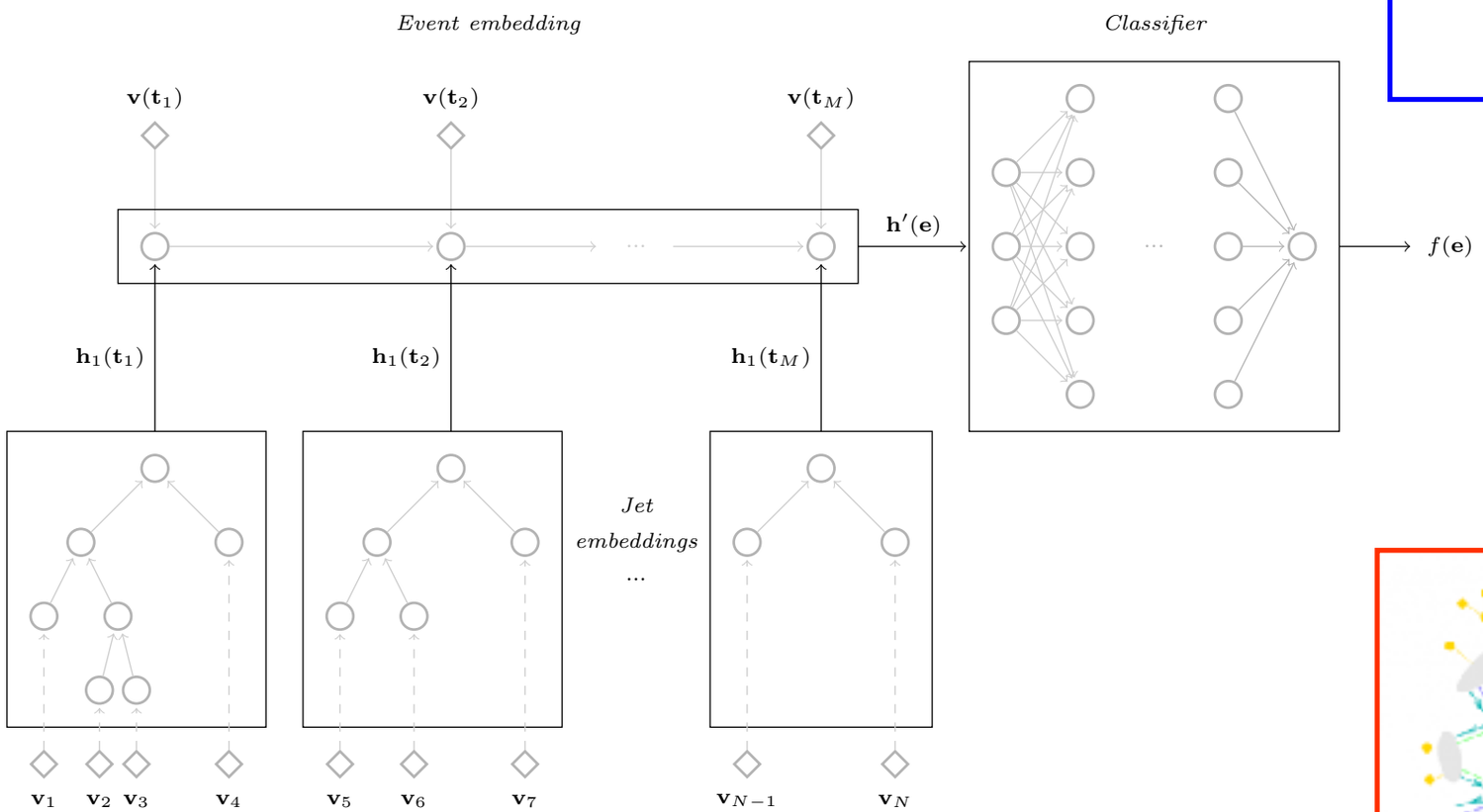


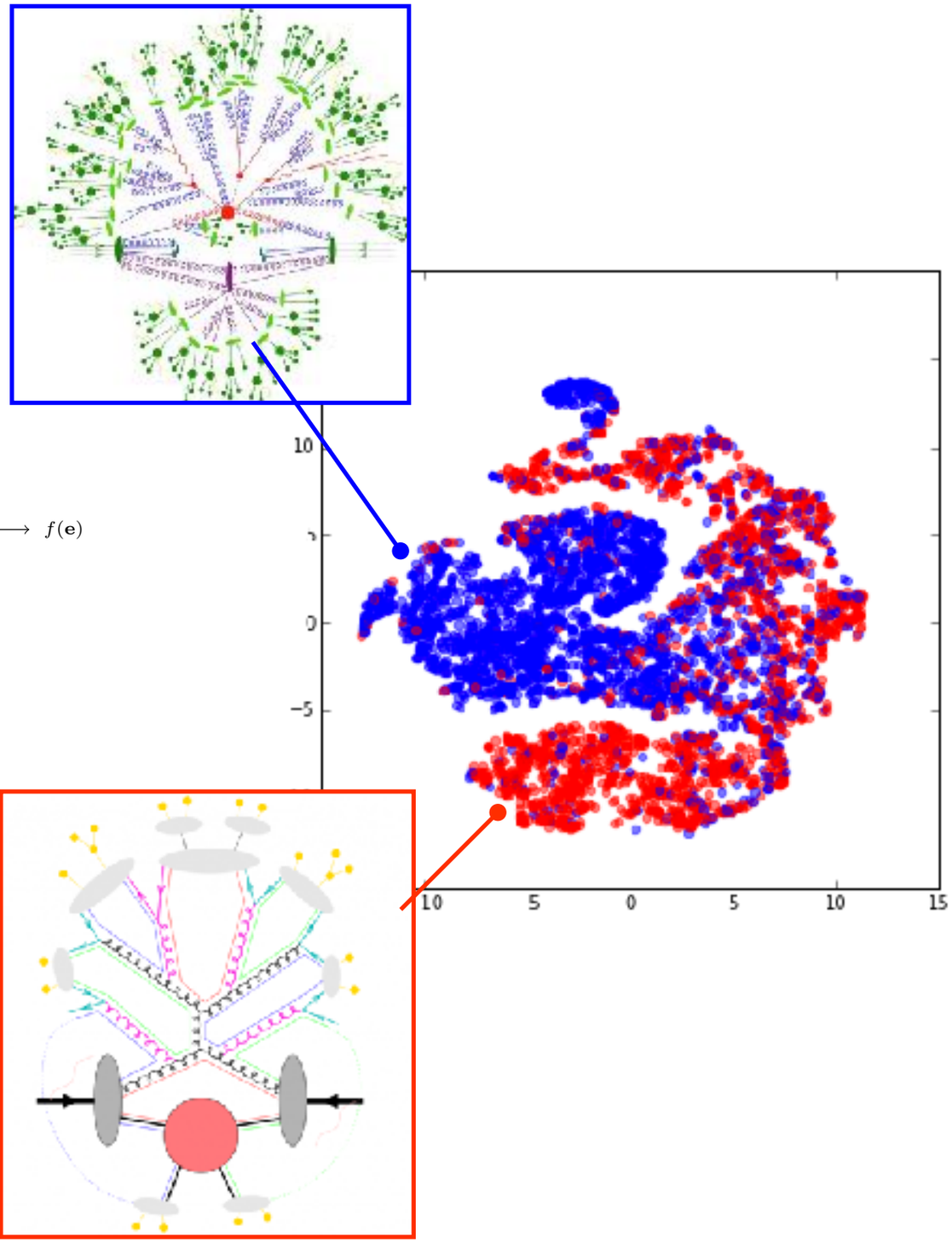
- W-jet tagging example using data from Dawe, et al arXiv:1609.00607
- down-sampling by projecting into images looses information
- RNN needs much less data to train!

HIERARCHICAL MODEL FOR THE ENTIRE EVENT

particle embedding → jet embedding → event embedding → classifier

It scales!





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

TASKS & LEARNING PARADIGMS

Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

TASKS & LEARNING PARADIGMS

Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

TASKS & LEARNING PARADIGMS

Tasks:

- Classification
- Regression
- Density Estimation
- Statistical Inference
- Decision Making

Learning Paradigms:

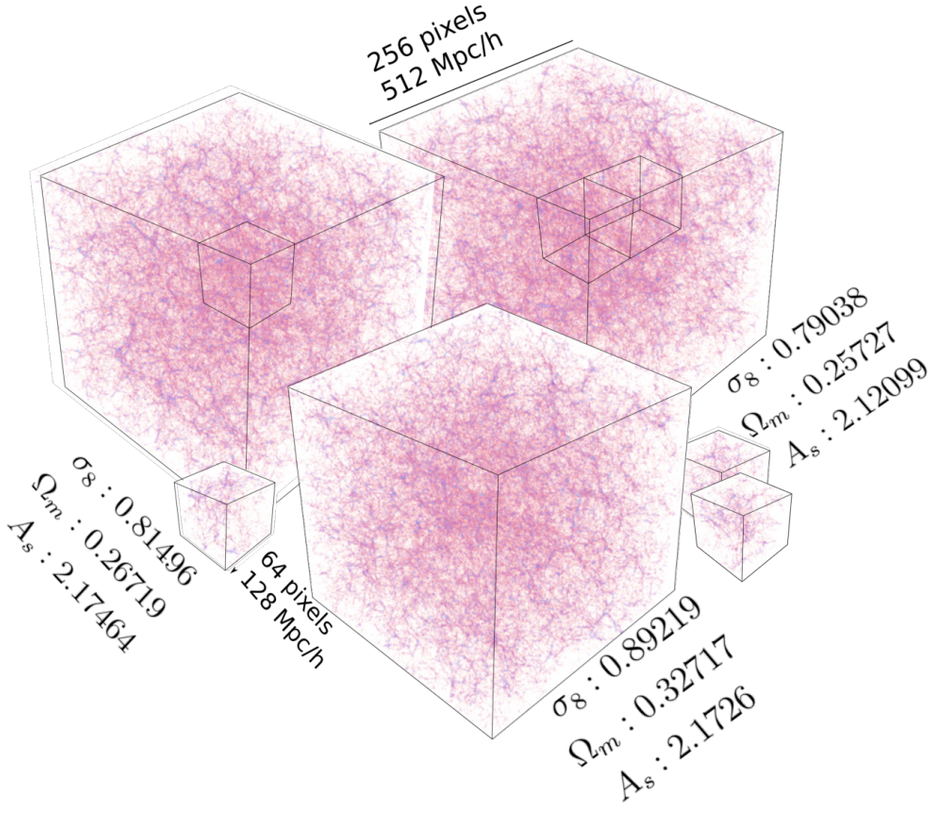
- Supervised Learning
- Weakly Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

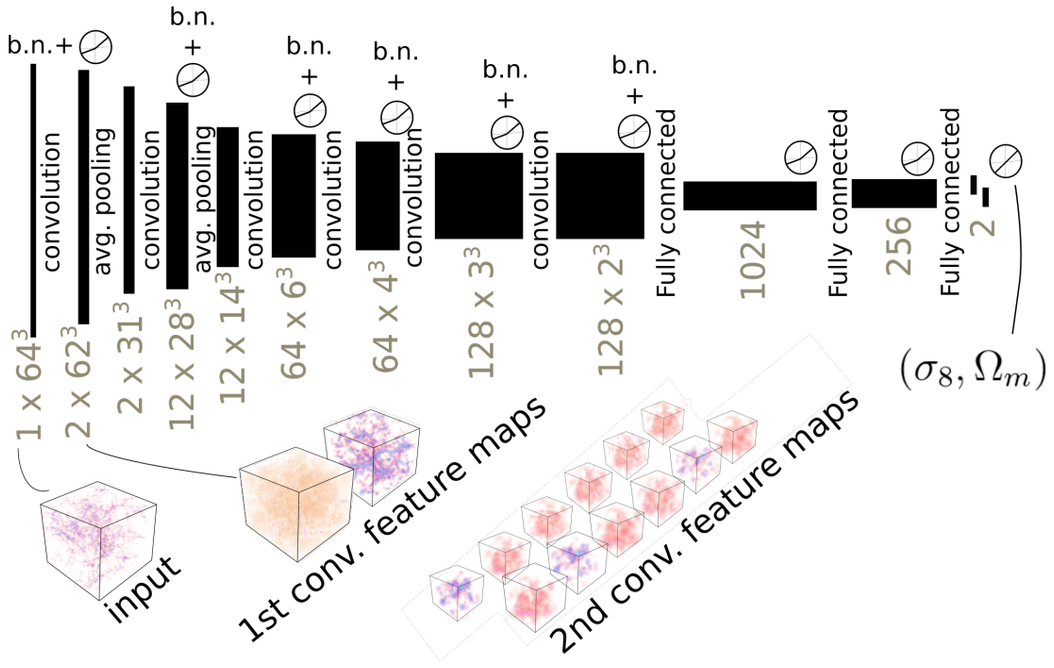
REGRESSION FOR Ω_M, σ_8

Estimating Cosmological Parameters from the Dark Matter Distribution

Siamak Ravanbakhsh⋆ MRAVANBA@CS.CMU.EDU
Junier Oliva⋆ JOLIVA@CS.CMU.EDU
Sebastien Fromenteau† SFROMENT@ANDREW.CMU.EDU
Layne C. Price† LAYNEP@ANDREW.CMU.EDU
Shirley Ho† SHIRLEYH@ANDREW.CMU.EDU
Jeff Schneider⋆ JEFF.SCHNEIDER@CS.CMU.EDU
Barnabás Póczos⋆ BAPOCZOS@CS.CMU.EDU

⋆ School of Computer Science, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213, USA
† McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Carnegie 5000 Forbes Ave., Pittsburgh, PA 15213, USA





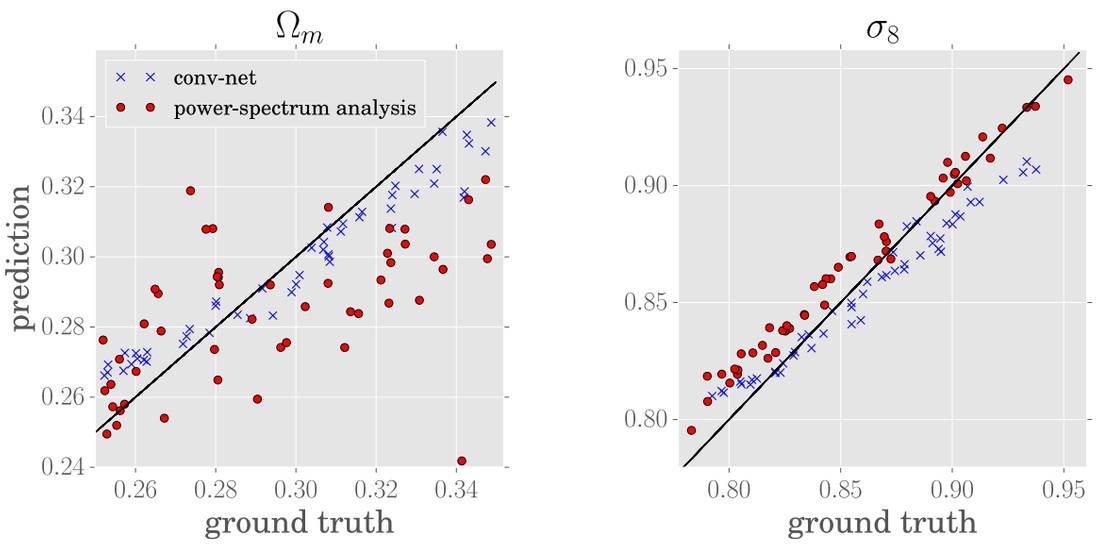
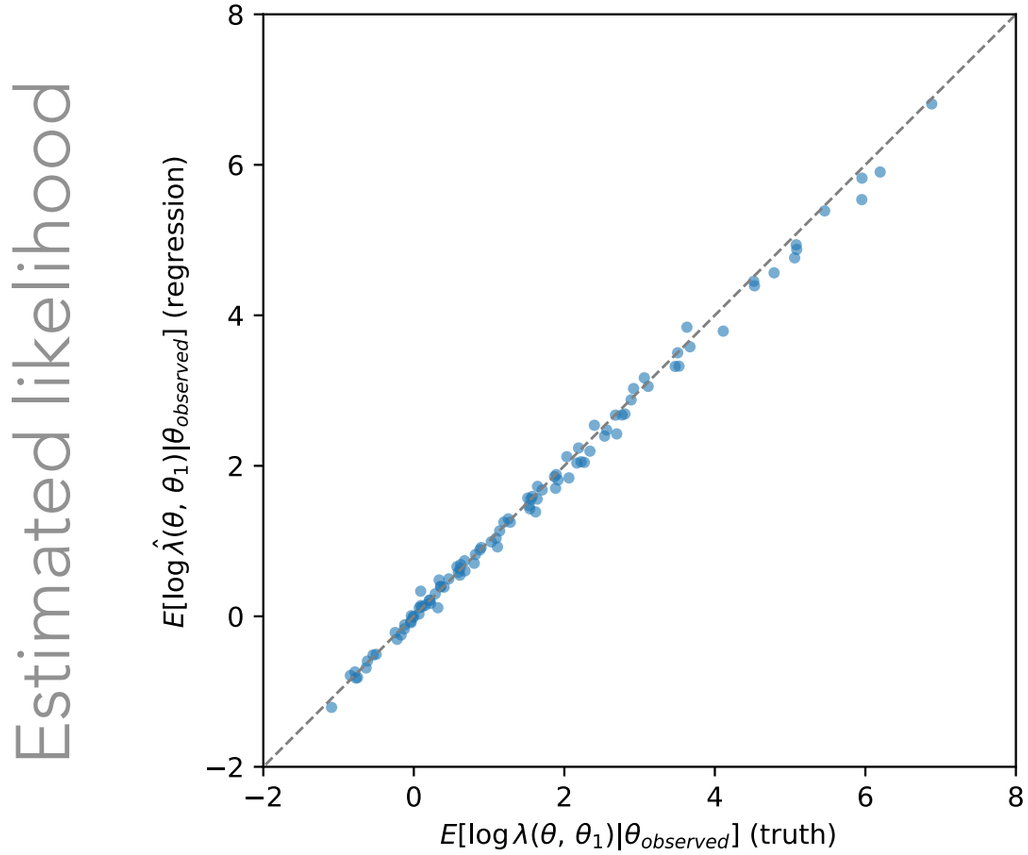
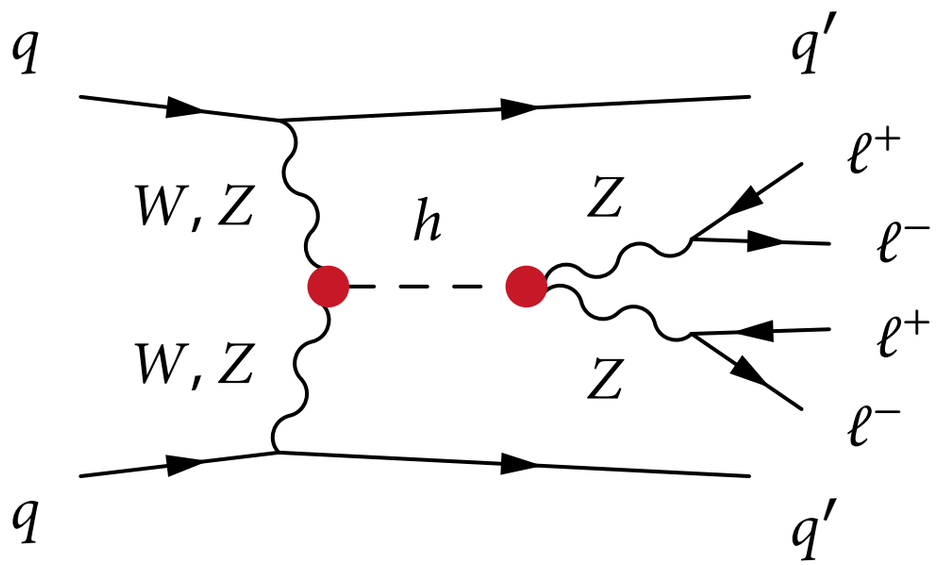


Figure 2. Prediction and ground truth of Ω_m and σ_8 using 3D conv-net and analysis of the power-spectrum on 50 test cube instances.

LEARNING A 16 DIM LIKELIHOOD



True likelihood



Hierarchical Modeling
&
Components

WHAT IS THE OBJECTIVE?

ML: What is the problem you are trying to solve?

Physicist: [eventually describes problem and formalizes objective]

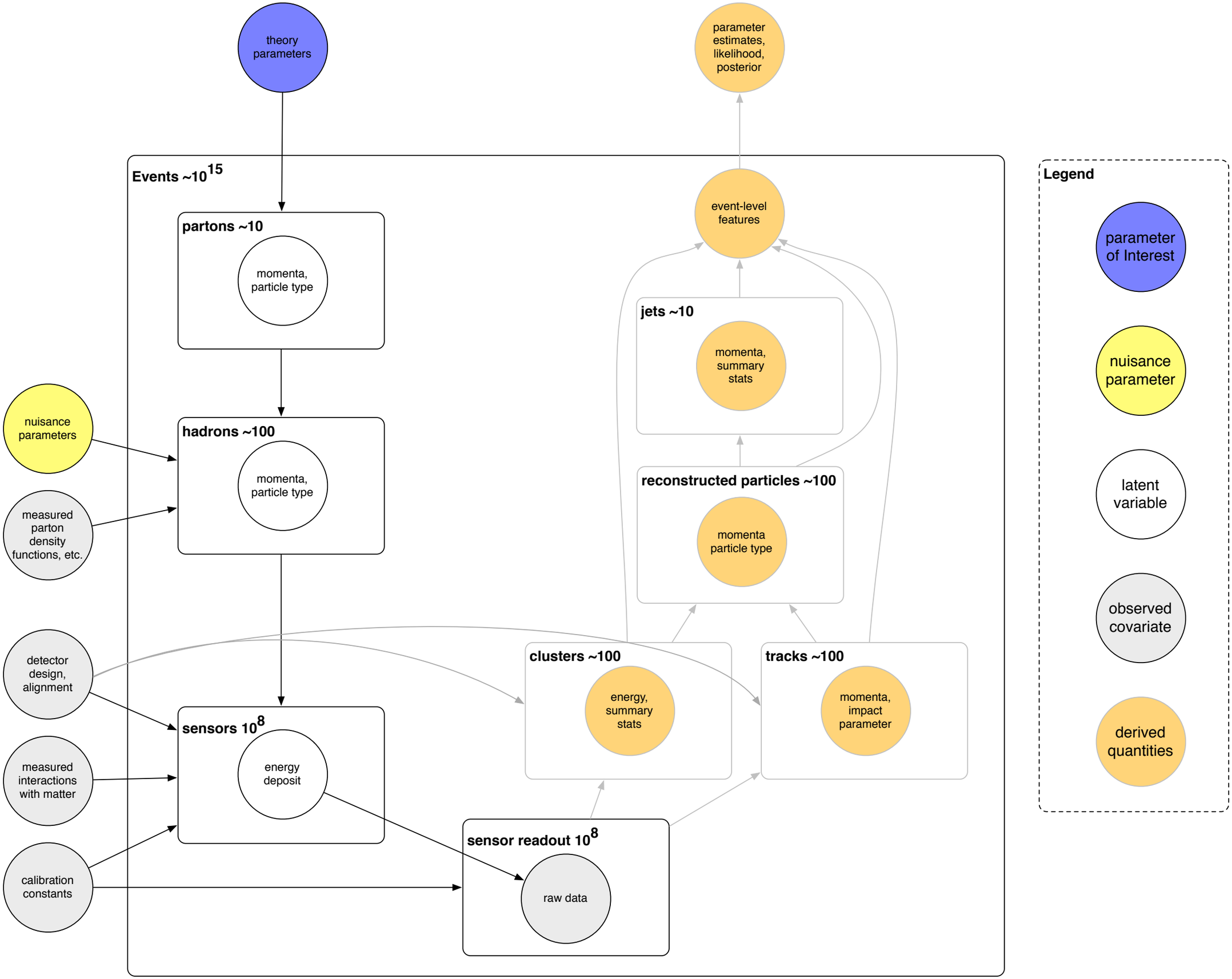
ML: Ok, well let's optimize this directly …

Physicist: but, I also want….

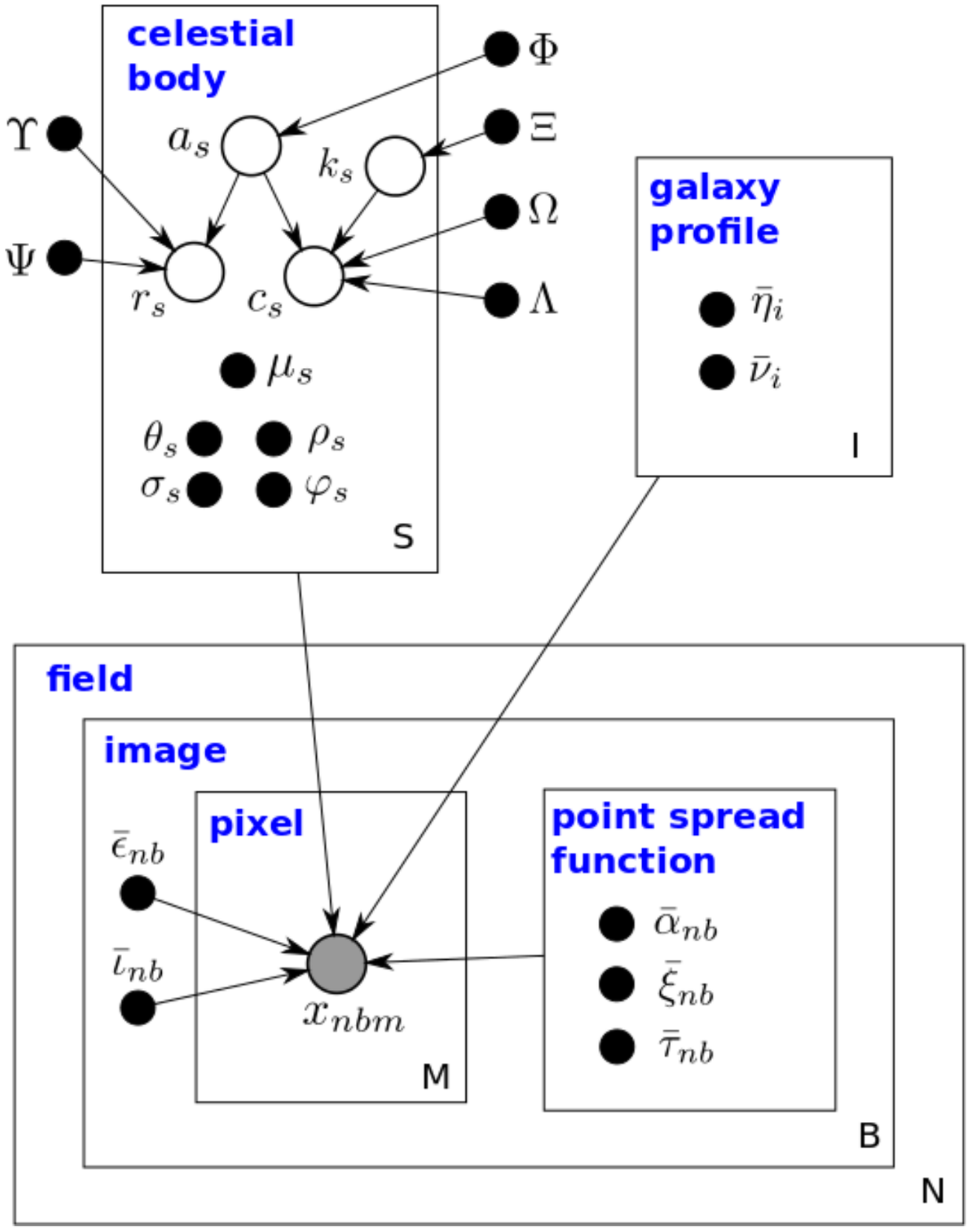
Used to criticize physicists for constantly changing problem statement, but traditional approach to physics problems has many advantages

- modular, reusable components (facilitates transfer learning, "ML2.0")
- interpretable & individually validated
- a form of structural regularization

FULL SIMULATION + RECONSTRUCTION



HIERARCHICAL GRAPHICAL MODELS IN ASTRONOMY



Celeste: Variational inference for a generative model of astronomical images

Regler, et al 33rd ICMLR, 2016

ML2.0?

How do these fit together?

Combine many of these ideas:

Large model, but **sparsely activated**

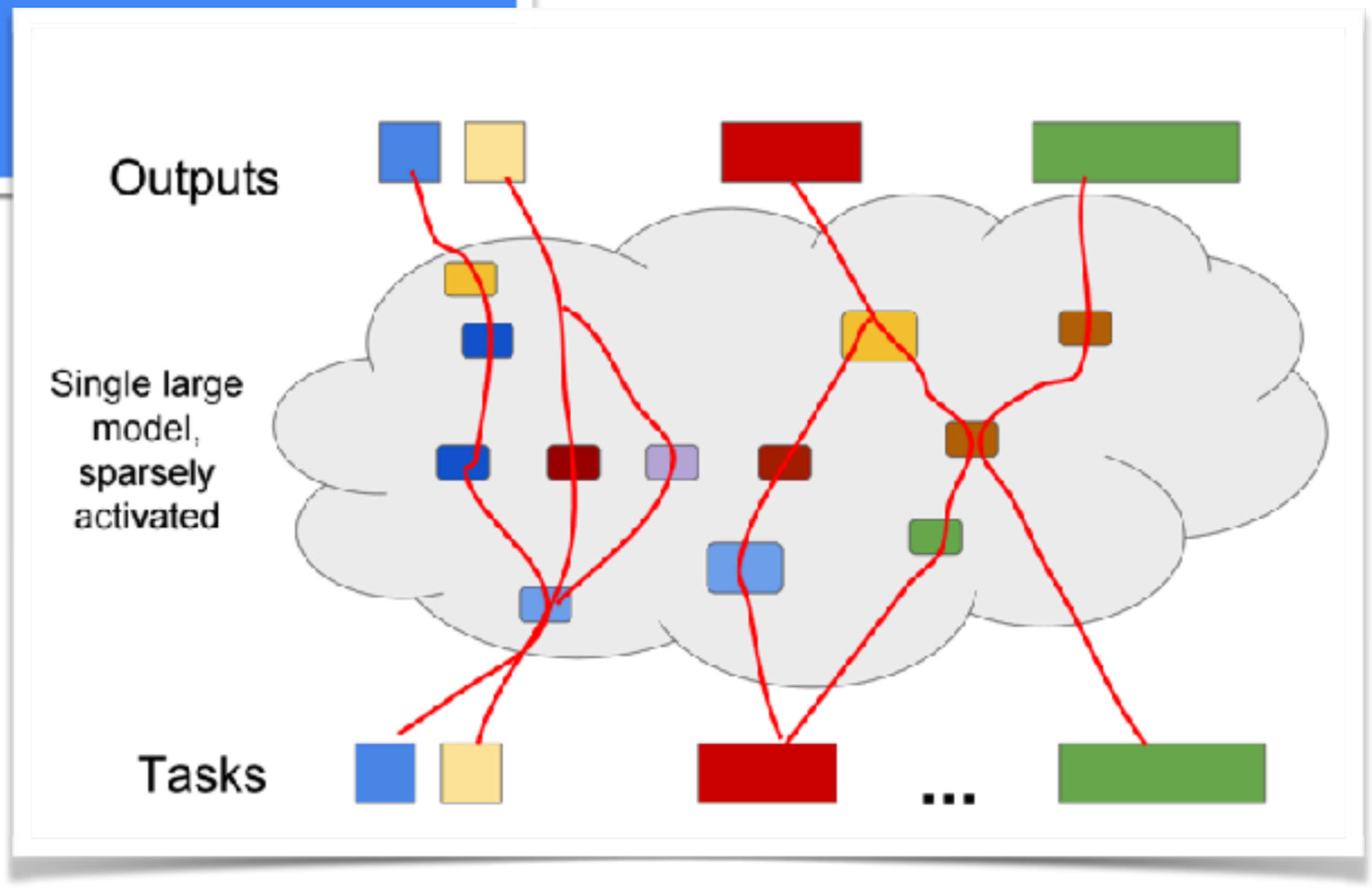
Single model to **solve many tasks** (100s to 1Ms)

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

Hardware **specialized for ML supercomputing**

ML for efficient mapping onto this hardware

Google



Slides from Jeff Dean of Google Brain @ Jeju July 2017

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

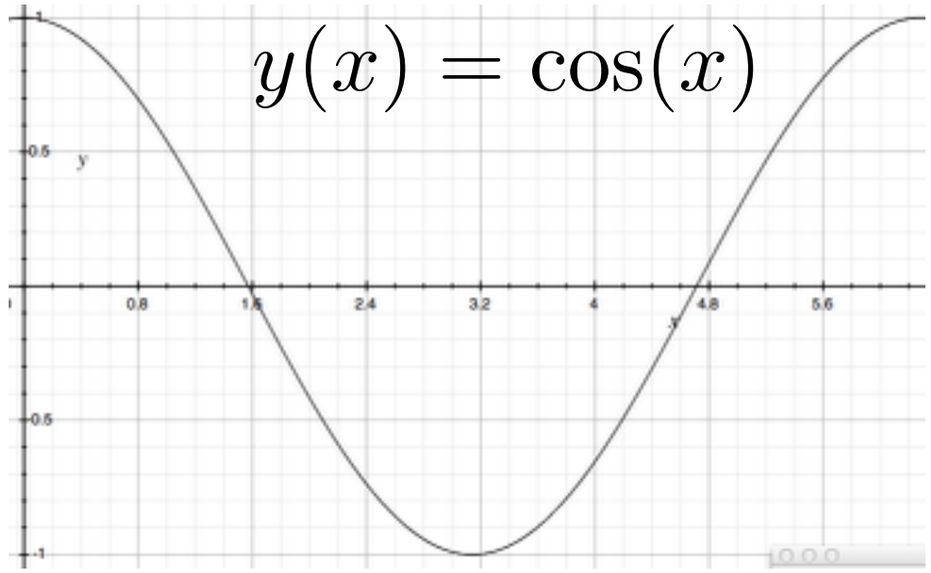
TRANSFORMATION PROPERTIES:

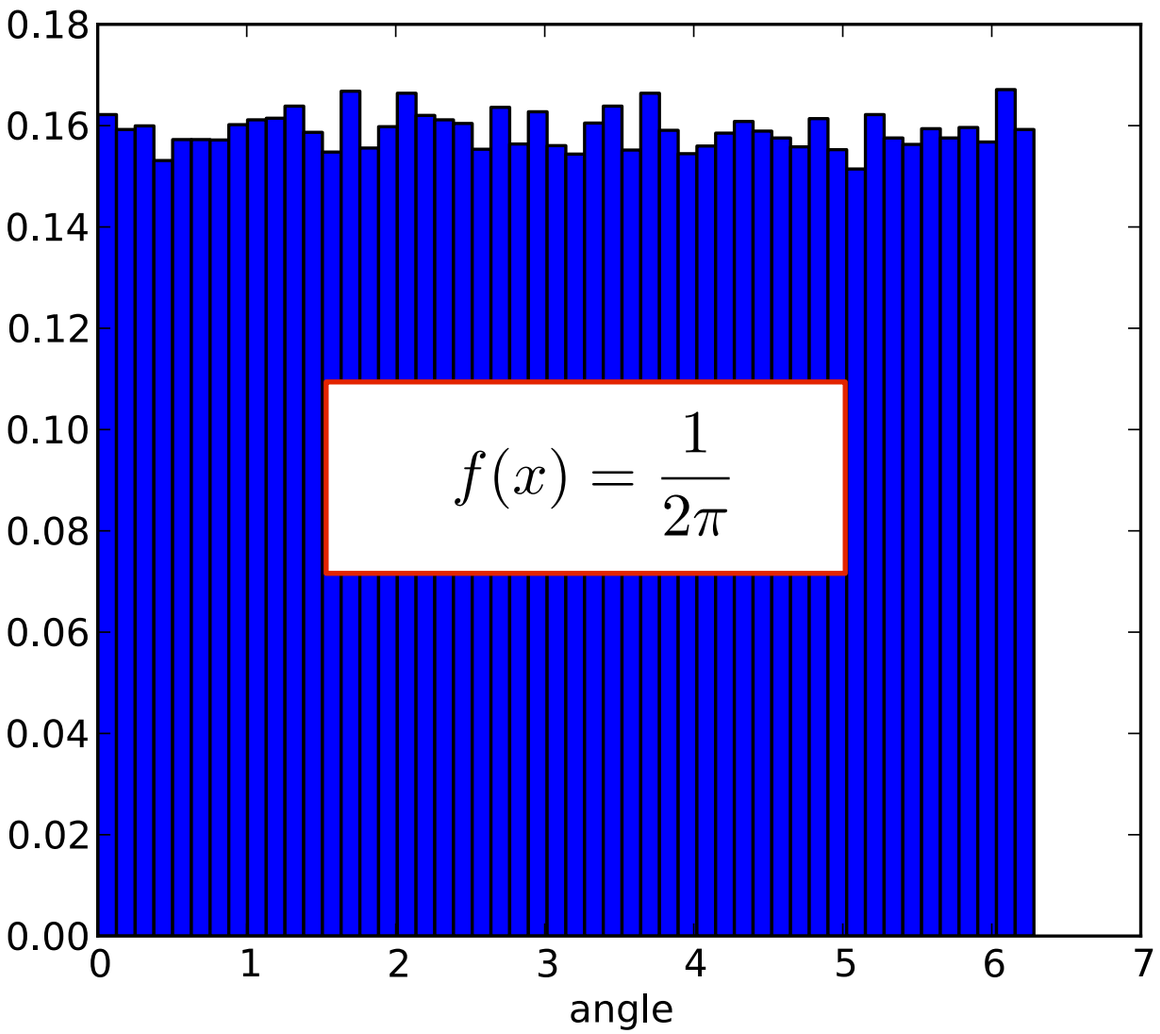
PDF VS. LIKELIHOOD

An example

if x has a uniform distribution [0,2π) ,

what is distribution of y=cos(x)?

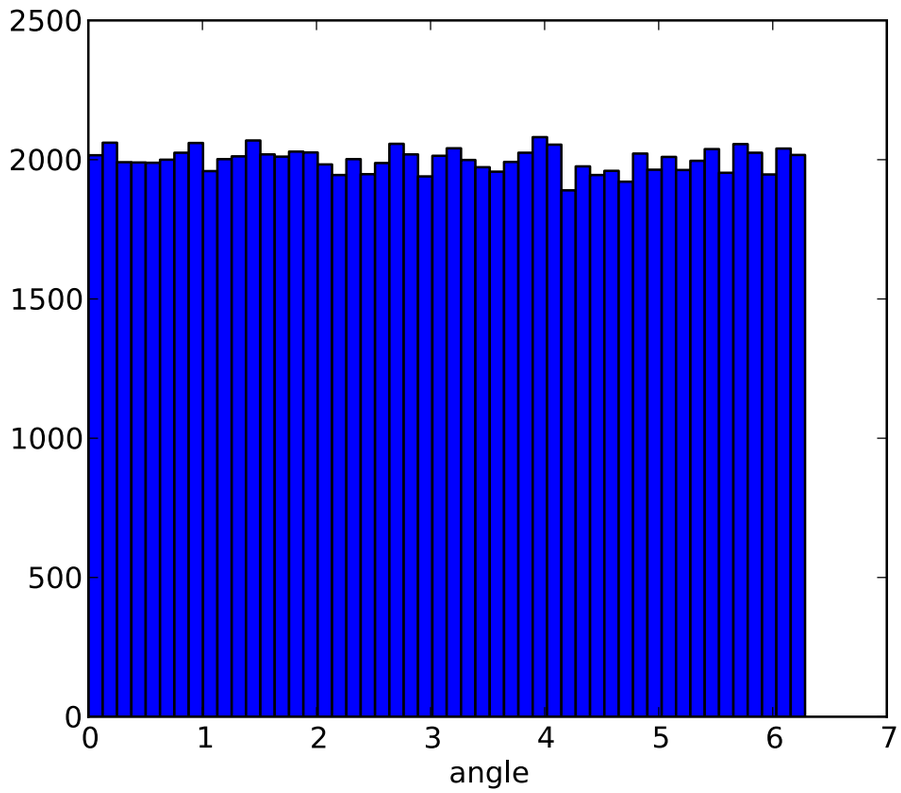




CHANGE OF VARIABLES

What happens with x→ cos(x)

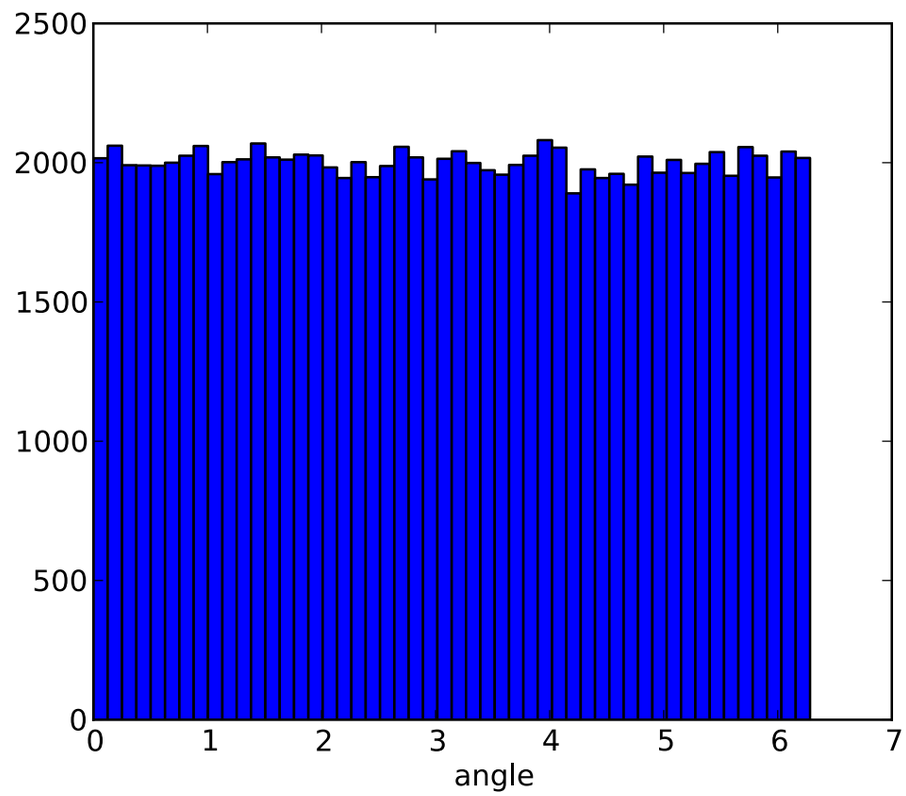
```python
import numpy as np
import matplotlib.pyplot as plt

N_MC=100000  # number of Monte Carlo Experiments
nBins = 50 # number of bins for Histograms

data_x, data_y = [],[]  #lists that will hold x and y

# do experiments
for i in range(N_MC):
    # generate observation for x
    x = np.random.uniform(0,2*np.pi)

    y = np.cos(x)
    data_x.append(x)
    data_y.append(y)

#setup figures
fig = plt.figure(figsize=(13,5))
fig_x = fig.add_subplot(1,2,1)
fig_y = fig.add_subplot(1,2,2)

fig_x.hist(data_x,nBins)
fig_x.set_xlabel('angle')

fig_y.hist(data_y,nBins)
fig_y.set_xlabel('cos(angle)')

plt.show()
```

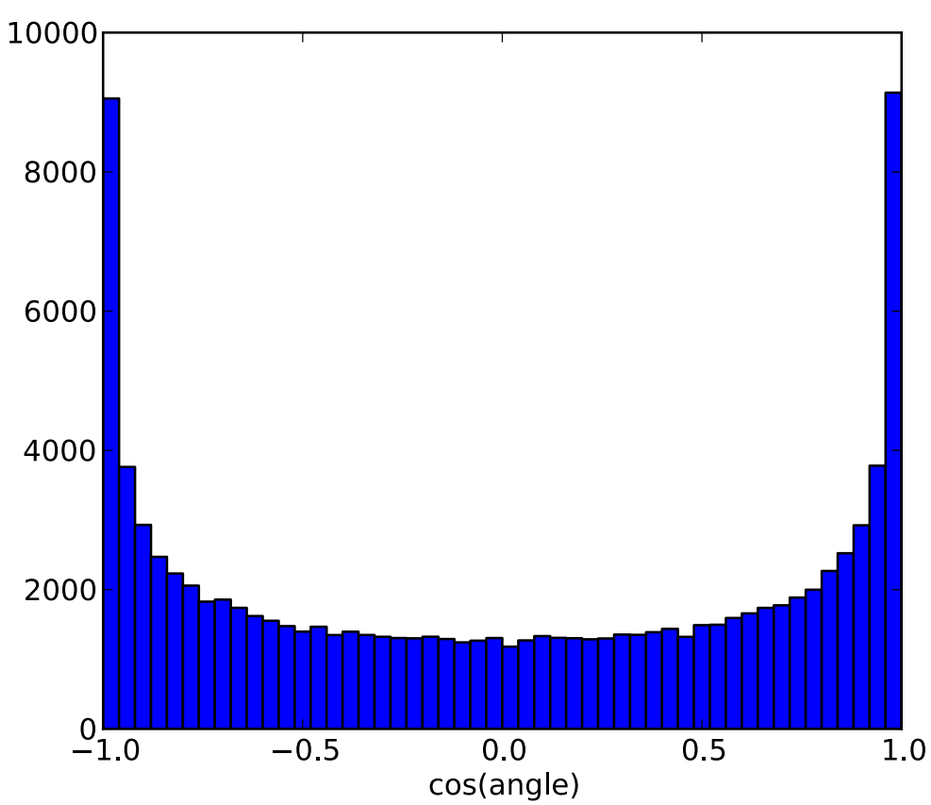


CHANGE OF VARIABLES

What happens with x→ cos(x)

```
1   import numpy as np
2   import matplotlib.pyplot as plt
3
4   N_MC=100000  # number of Monte Carlo Experiments
5   nBins = 50 # number of bins for Histograms
6
7   data_x, data_y = [],[]  #lists that will hold x and y
8
9   # do experiments
10  for i in range(N_MC):
11      # generate observation for x
12      x = np.random.uniform(0,2*np.pi)
13
14      y = np.cos(x)
15      data_x.append(x)
16      data_y.append(y)
17
18  #setup figures
19  fig = plt.figure(figsize=(13,5))
20  fig_x = fig.add_subplot(1,2,1)
21  fig_y = fig.add_subplot(1,2,2)
22
23  fig_x.hist(data_x,nBins)
24  fig_x.set_xlabel('angle')
25
26  fig_y.hist(data_y,nBins)
27  fig_y.set_xlabel('cos(angle)')
28
29  plt.show()
```





CHANGE OF VARIABLES

If *f(x)* is the pdf for *x* and *y(x)* is a change of variables, then the pdf *g(y)* must satisfy

$$P(x_a < x < x_b) \equiv \int_{x_a}^{x_b} f(x)dx = \int_{y(x_a)}^{y(x_b)} g(y)dy \equiv P(y(x_a) < y < y(x_b))$$

We can rewrite the integral on the right

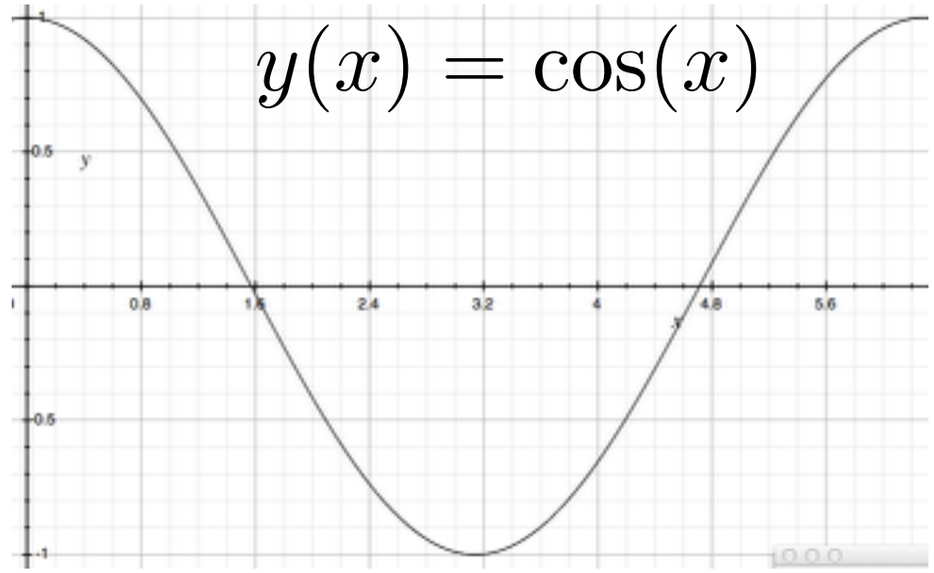
$$\int_{y(x_a)}^{y(x_b)} g(y)dy = \int_{x_a}^{x_b} g(y(x)) \left|\frac{dy}{dx}\right| dx$$

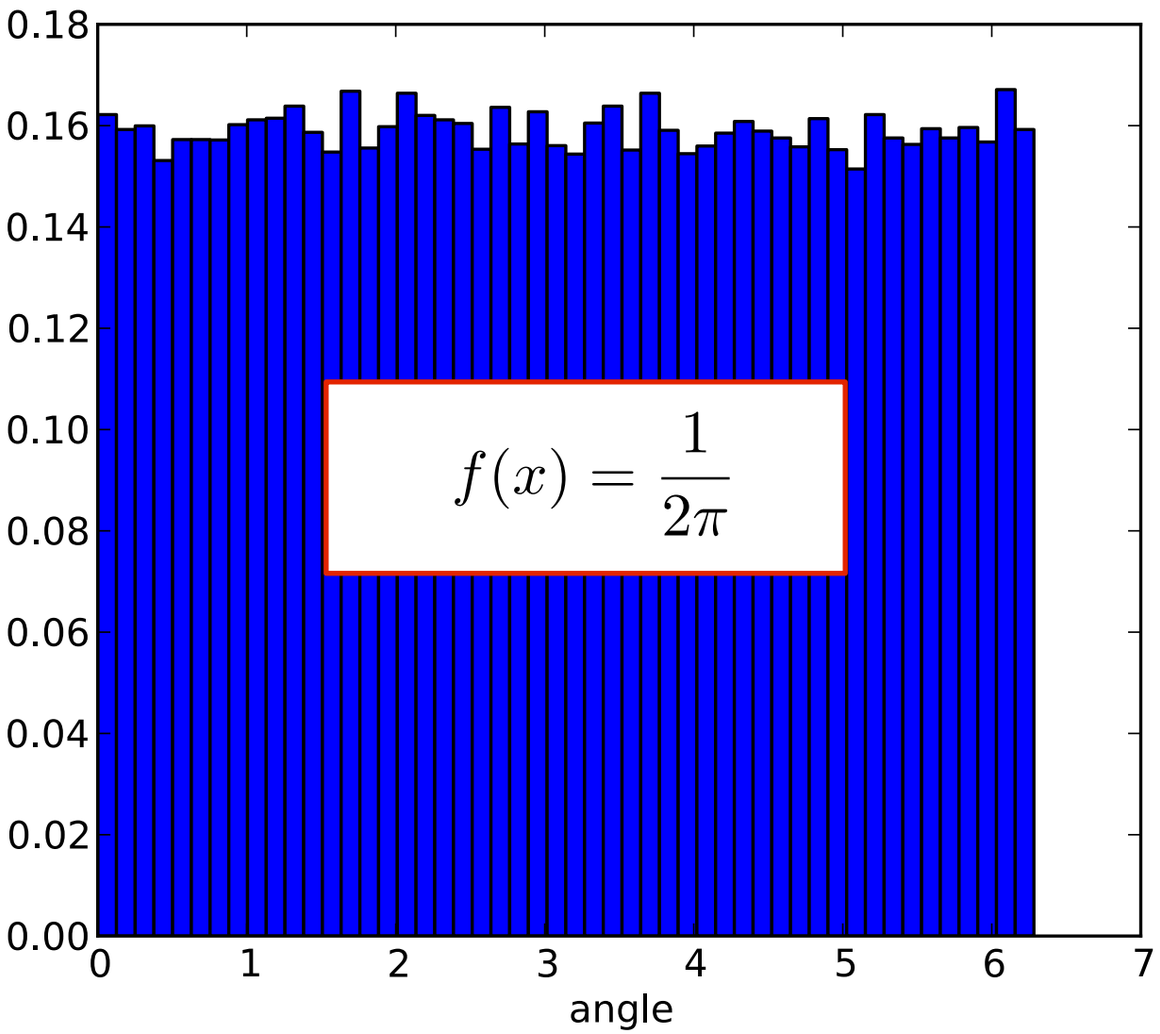
therefore, the two pdfs are related by a Jacobian factor

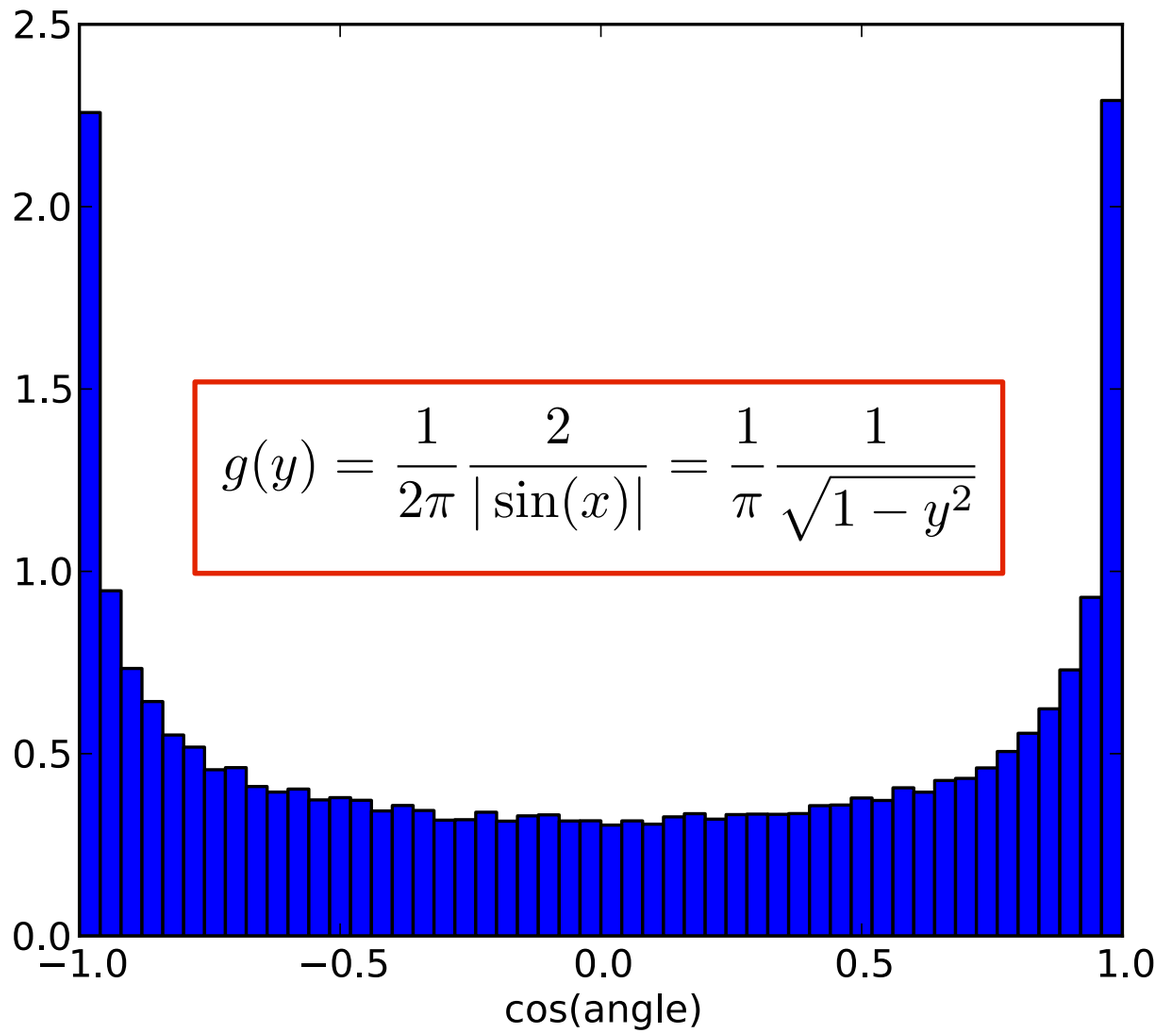
$$f(x) = g(y) \left|\frac{dy}{dx}\right|$$

An example

$$f(x) = g(y)\left|\frac{dy}{dx}\right|$$

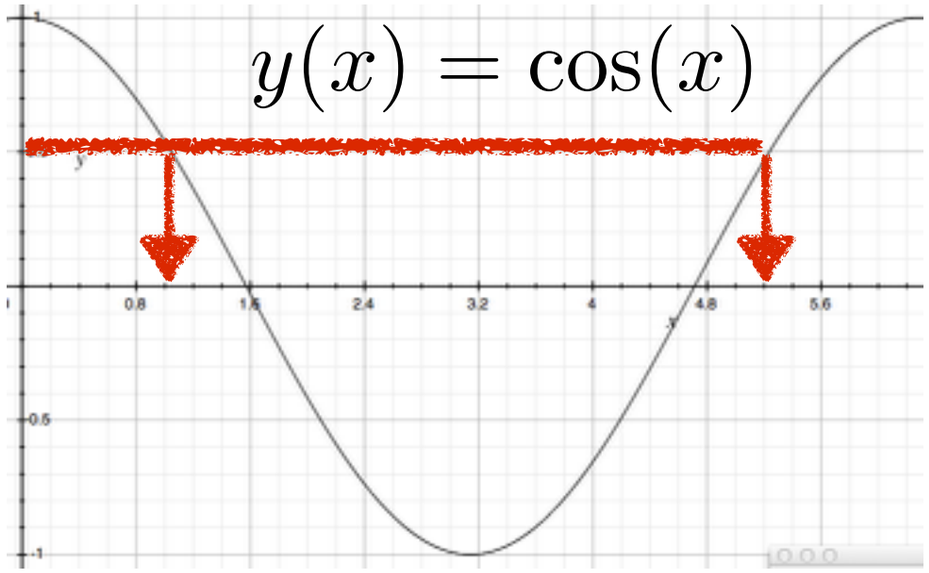


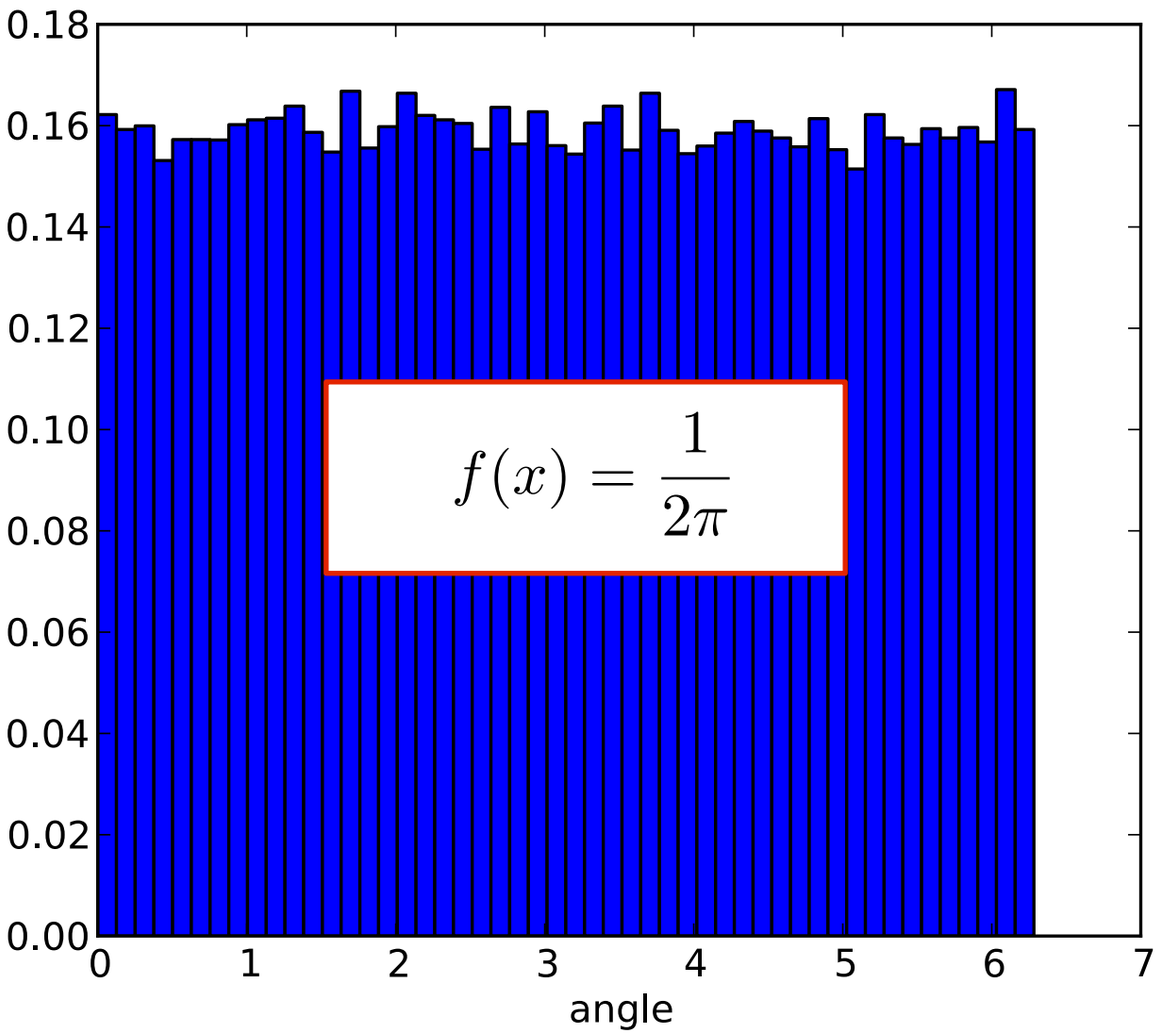


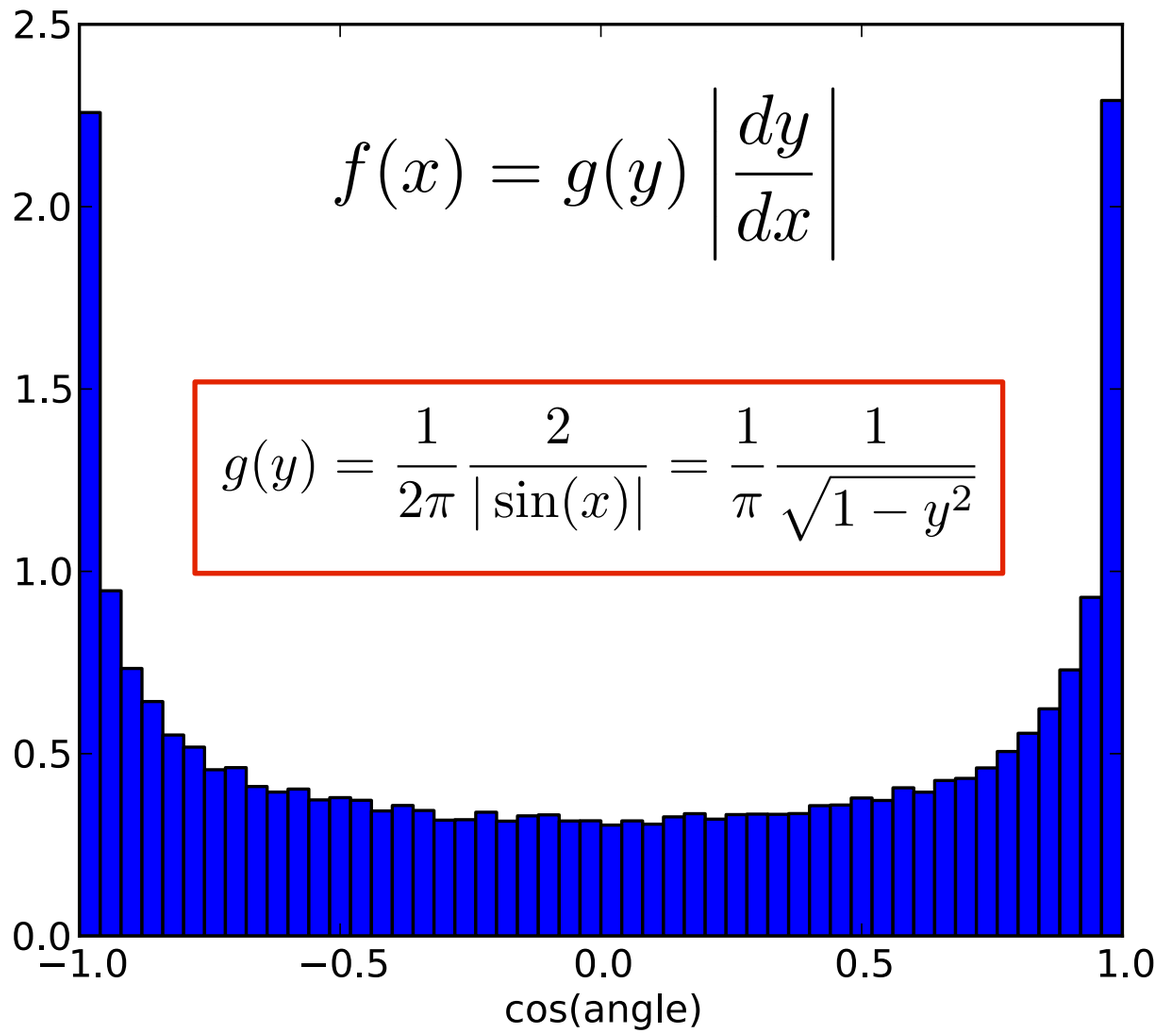


An example

I am glossing over the fact that the map is not 1-to-1. Different values of x, map into same value of y. We will need to sum/integrate over them. Here it is easy, but in general this may become intractable… need inverse map







Change of variable x, change of parameter θ

- **For pdf p(x|θ) and change of variable from x to y(x):**

  p(y(x)|θ) = p(x|θ) / |dy/dx|.

  Jacobian modifies probability *density*, guaranties that

  P($y(x_1) < y < y(x_2)$) = P($x_1 < x < x_2$), i.e., that

  ***Probabilities* are invariant under change of variable x.**

  - **Mode of probability *density* is *not* invariant (so, e.g., criterion of maximum probability density is ill-defined).**
  - **Likelihood *ratio* is invariant under change of variable x. (Jacobian in denominator cancels that in numerator).**

- **For likelihood $\mathcal{L}(\theta)$ and reparametrization from θ to u(θ):**

  $\mathcal{L}(\theta) = \mathcal{L}(u(\theta))$ **(!).**

  - **Likelihood $\mathcal{L}(\theta)$ is invariant under reparametrization of parameter θ (reinforcing fact that $\mathcal{L}$ is *not* a pdf in θ).**

Probability Integral Transform

Consider a specific change of variables related to the cumulative for some arbitrary *f(x)*

$$y(x) = \int_{-\infty}^{x} f(x')dx'$$

Using our general change of variables formula:

$$f(x) = g(y)\left|\frac{dy}{dx}\right|$$

We find for this case the Jacobian factor is

$$\left|\frac{dy}{dx}\right| = f(x)$$

Thus $g(y) = 1$

Probability Integral Transform

"…seems likely to be one of the most fruitful conceptions introduced into statistical theory in the last few years"
– Egon Pearson (1938)

Given continuous x ∈ (a,b), and its pdf p(x), let

$$y(x) = \int_a^x p(x')\,dx' \ .$$

Then y ∈ (0,1) and p(y) = 1 (uniform) for all y. (!)

So there always exists a metric in which the pdf is uniform.

***Many* issues become more clear (or trivial) after this transformation*. (If x is discrete, some complications.)**

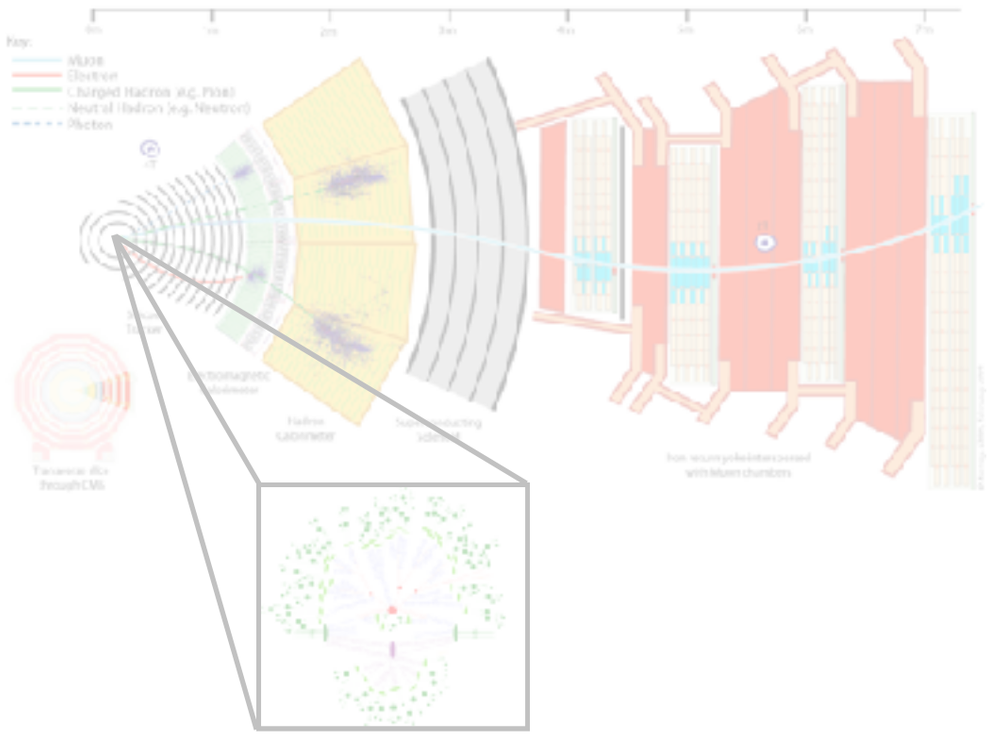
The specification of a Bayesian prior pdf p(μ) for parameter μ is equivalent to the choice of the metric f(μ) in which the pdf is uniform. This is a *deep* issue, not always recognized as such by users of flat prior pdf's in HEP!

***And the inverse transformation provides for efficient M.C. generation of p(x) starting from RAN().**

TWO APPROACHES

Use simulator

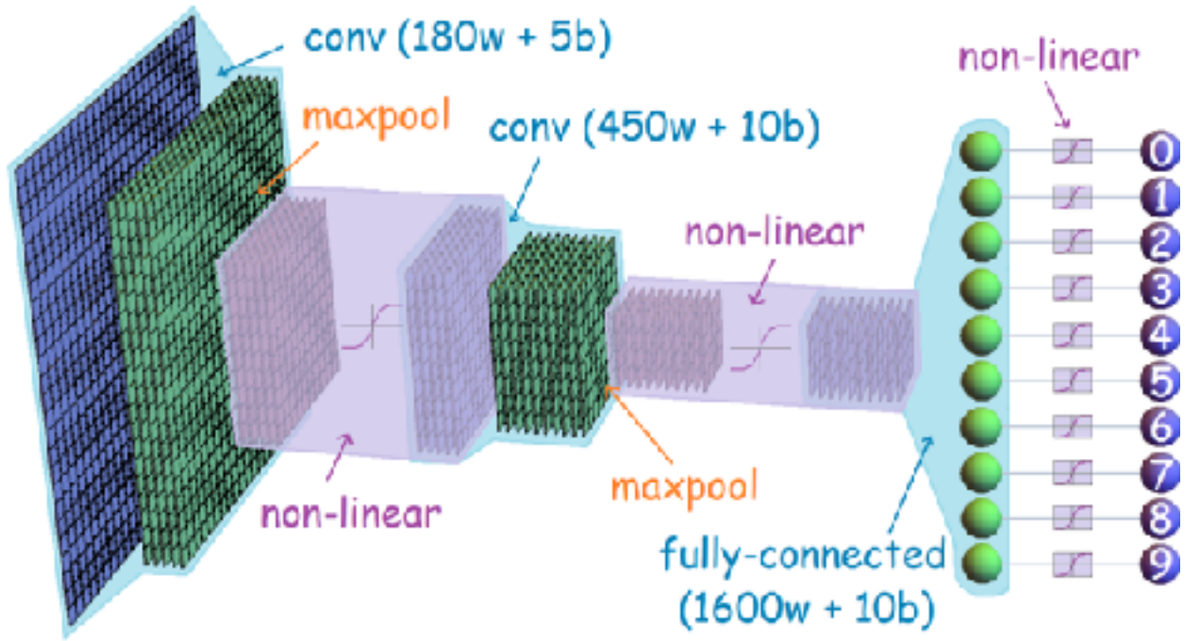
(much more efficiently)



- 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)
- Autogregressive models, Normalizing Flows

[image credit: A.P. Goucher]

DENSITY ESTIMATION VIA CALCULUS OF VARIATIONS

What function r(x) minimizes the "cross-entropy" loss?

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)}\, dx$$

- Subject to $\int r(x)dx = 1$

DENSITY ESTIMATION VIA CALCULUS OF VARIATIONS

What function r(x) minimizes the "cross-entropy" loss?

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)}\, dx \;\approx\; \frac{1}{N}\sum_{i=1}^{N} \log r(x_i)$$

- Subject to $\int r(x)dx = 1$

DENSITY ESTIMATION VIA CALCULUS OF VARIATIONS

What function r(x) minimizes the "cross-entropy" loss?

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)}\, dx \;\approx\; \frac{1}{N}\sum_{i=1}^{N} \log r(x_i)$$

- Subject to $\int r(x)dx = 1$

Euler-Lagrange Equation w/ Lagrange-multiplier

$$L[r,\lambda] = F(x,r) + \lambda r(x)$$

$$\underbrace{\frac{d}{dx}\left(\frac{\delta L}{\delta r'}\right)}_{=0} - \frac{\delta L}{\delta r} = 0 \qquad \frac{\delta L}{\delta r} = 0 = \frac{-p(x)}{r(x)} + \lambda$$

$$r(x) = p(x)/\lambda$$

imposing the constraint gives $\lambda = 1$ thus $r(x) = p(x)$

How do we create complicated probability densities p(x) that are tractable

and

are normalized such that $\int p(x)\, dx = 1$?

BIJECTIONS

K.C. & G. Louppe: http://beta.briefideas.org/ideas/5c2f74aedbf3618ca180382e393c7617

If I have a bijection: $f : X \to Z$

and an arbitrary tractable density on Z: $p(z)$

Then density on X follows from a simple change of variables

$$p(x) = p(f_\phi(x)) \left| \det \left(\frac{\partial f_\phi(x)}{\partial x_T} \right) \right|$$

Now construct neural networks f_ϕ that are bijections & optimize "cross entropy" loss

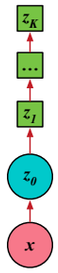
If it is a bijection, I can generate samples of x from inverse transformation $f^{-1}(z)$

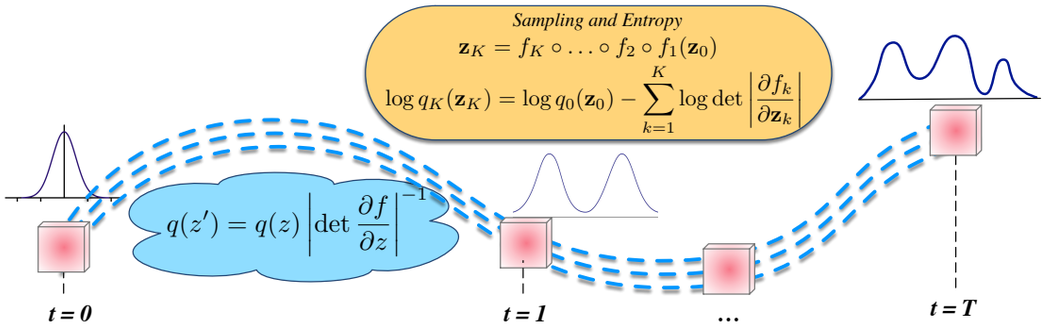
ENGINEERING BIJECTIONS

Approximations using Change-of-variables

Exploit the rule for change of variables for random variables:

- Begin with an initial distribution $q_0(\mathbf{z}_0|\mathbf{x})$.
- Apply a sequence of K invertible functions f_k.





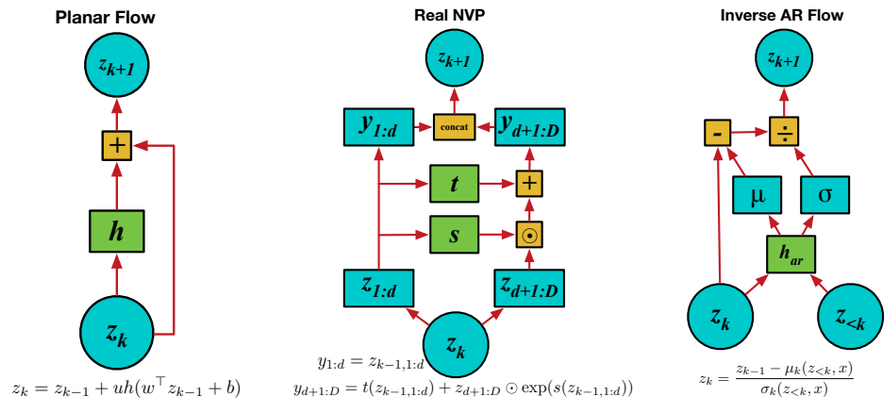
Distribution flows through a sequence of invertible transforms

[Rezende and Mohamed, 2015]

Choice of Transformation Function

$$\mathscr{L} = \mathbb{E}_{q_0(\mathbf{z}_0)}[\log p(\mathbf{x}, \mathbf{z}_K)] - \mathbb{E}_{q_0(\mathbf{z}_0)}[\log q_0(\mathbf{z}_0)] - \mathbb{E}_{q_0(\mathbf{z}_0)}\left[\sum_{k=1}^{K} \log \det \left| \frac{\partial f_k}{\partial \mathbf{z}_k} \right|\right]$$

- Begin with a fully-factorised Gaussian and improve by change of variables.
- Triangular Jacobians allow for computational efficiency.



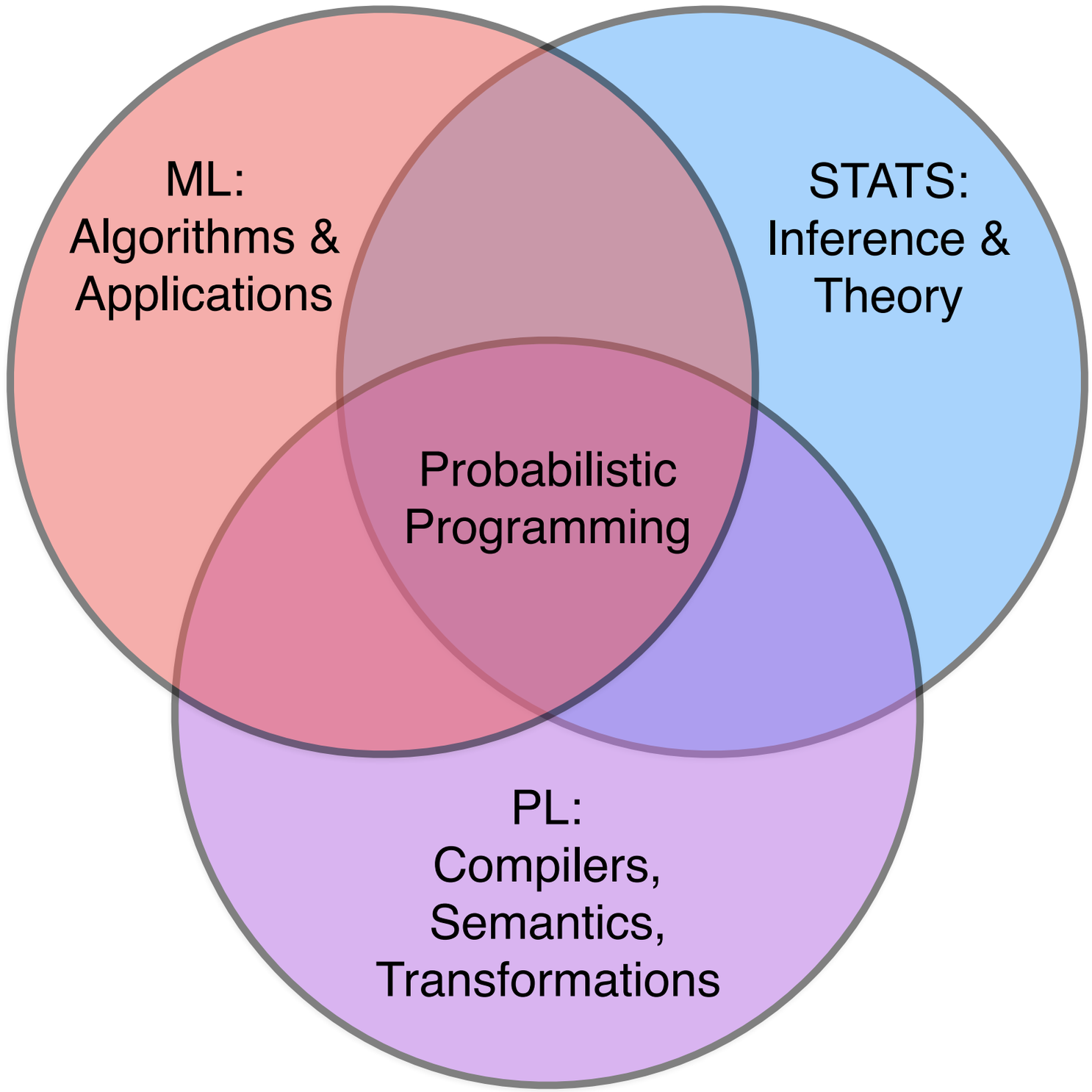
[Rezende and Mohamed, 2016; Dinh et al., 2016; Kingma et al., 2016]

Linear time computation of the determinant and its gradient.

Probabilistic Programming:
Inverting the simulation

(very ambitious)

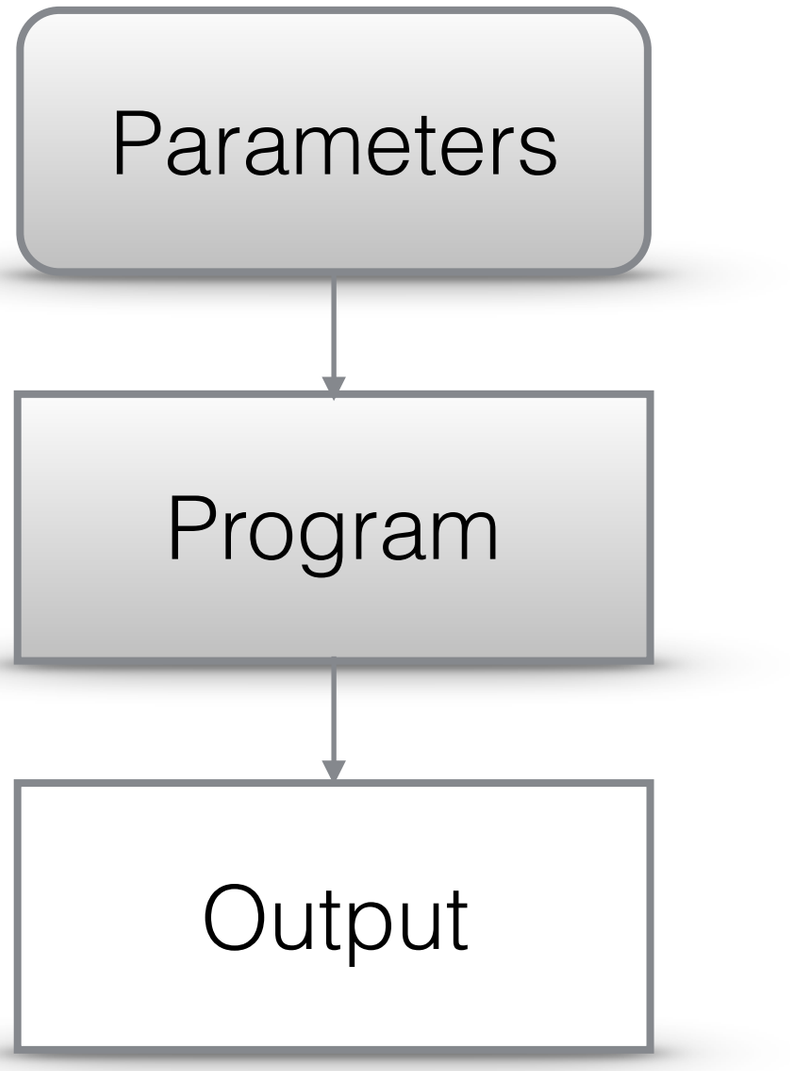
Probabilistic Programming



Intuition

[slides, Frank Wood]

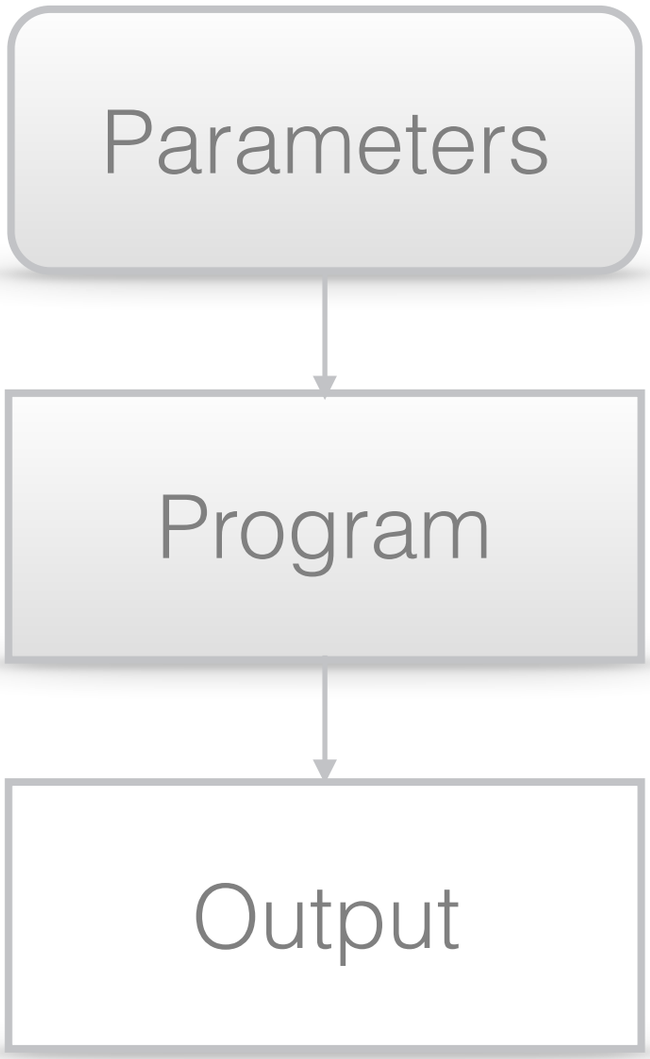
Intuition



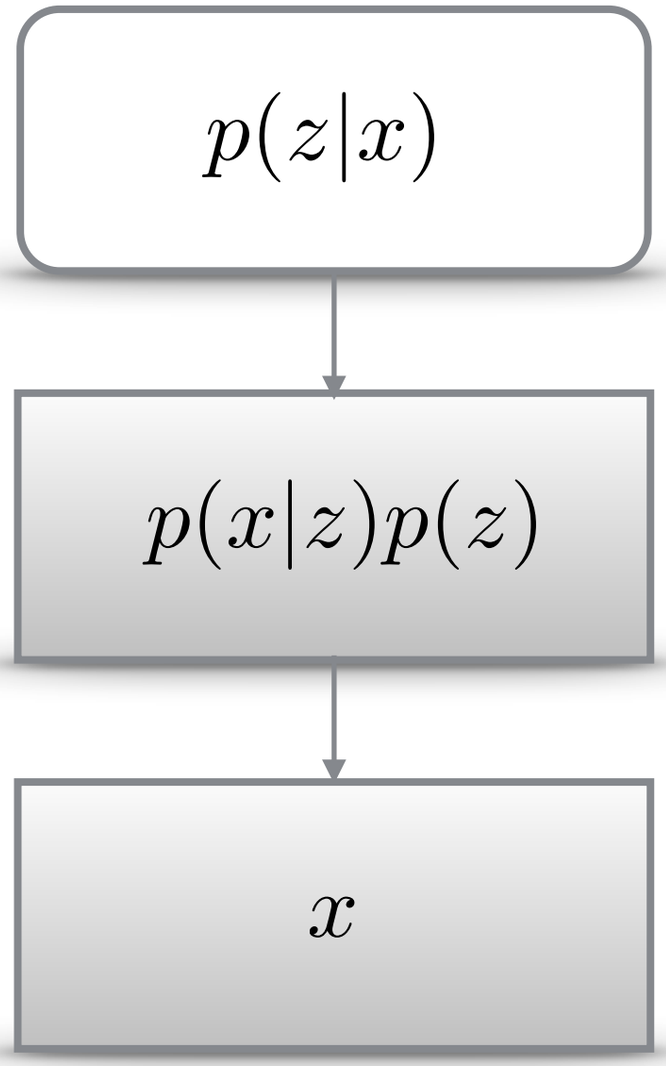
CS

[slides, Frank Wood]

Intuition



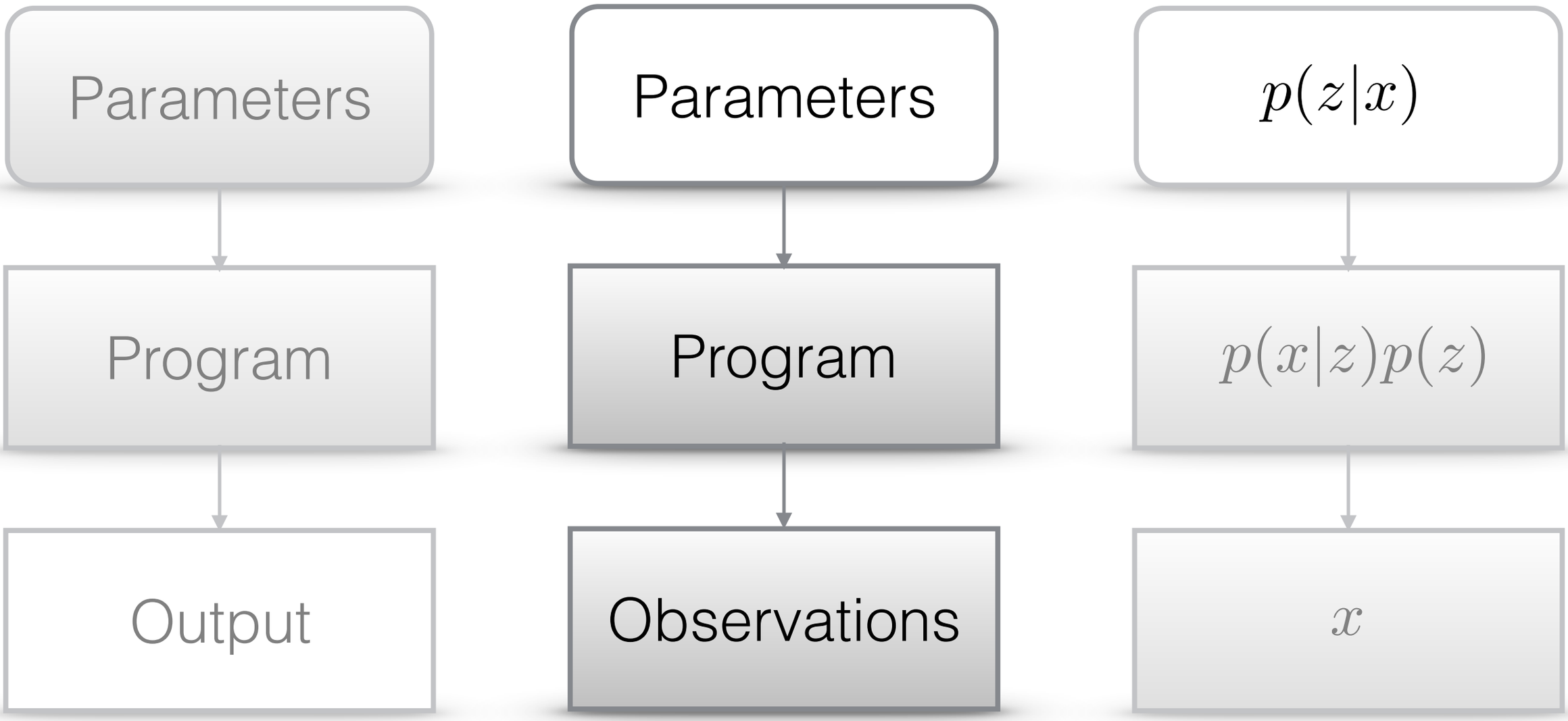
CS



Statistics

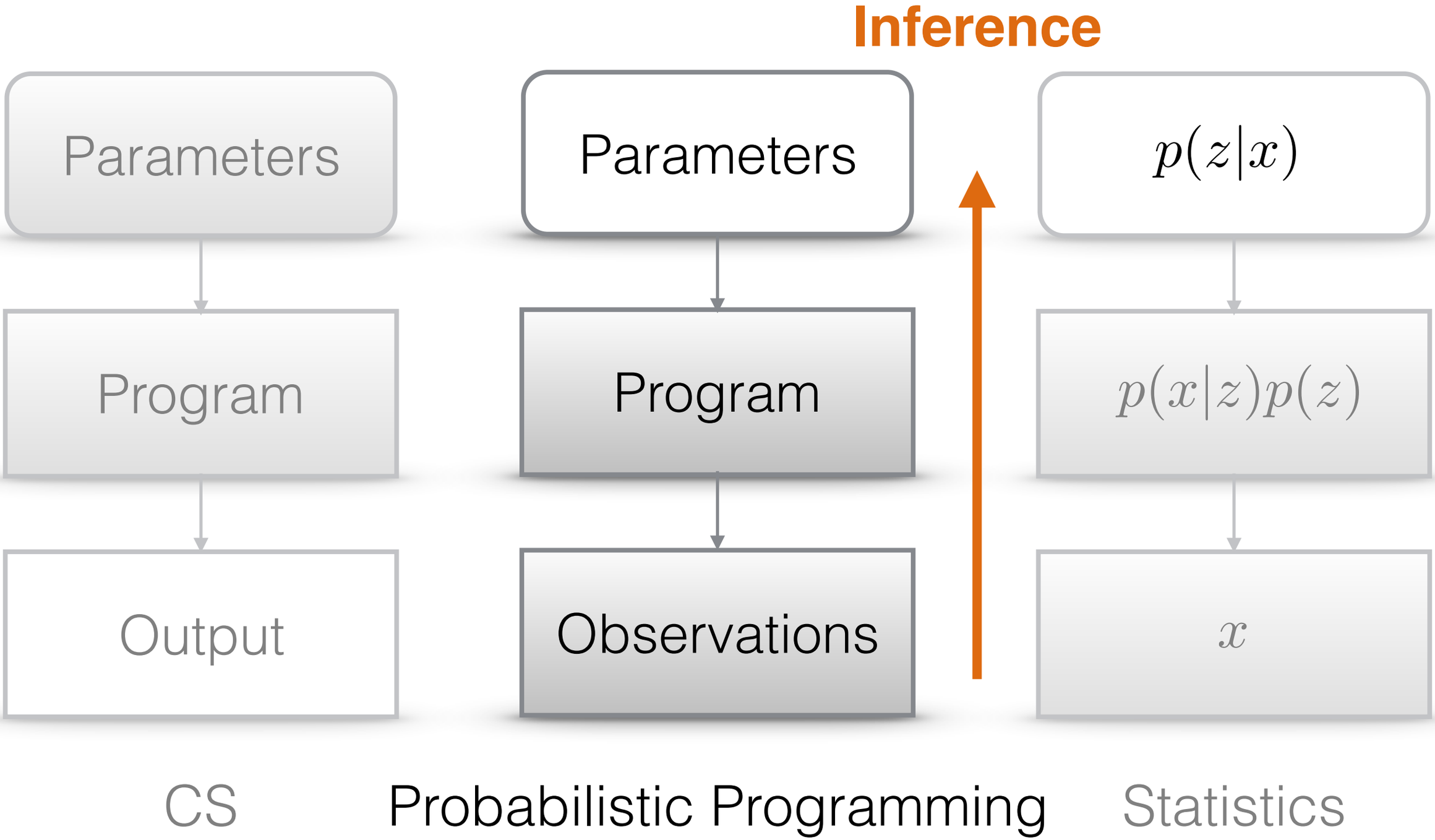
[slides, Frank Wood]

Intuition



[slides, Frank Wood]

Intuition



[slides, Frank Wood]

CAPTCHA breaking

Observation

Posterior Samples

Generative Model

```
(defquery captcha
 [image num-chars tol]
 (let [[w h] (size image)
       ;; sample random characters
       num-chars (sample
                  (poisson num-chars))
       chars (repeatedly
              num-chars sample-char)]
  ;; compare rendering to true image
  (map (fn [y z]
         (observe (normal z tol) y))
       (reduce-dim image)
       (reduce-dim (render chars w h)))
 ;; predict captcha text
 {:text
  (map :symbol (sort-by :x chars))}))
```

Mansinghka,, Kulkarni, Perov, and Tenenbaum
“Approximate Bayesian image interpretation using generative probabilistic graphics programs." NIPS (2013).

CAPTCHA breaking

Observation

Posterior Samples

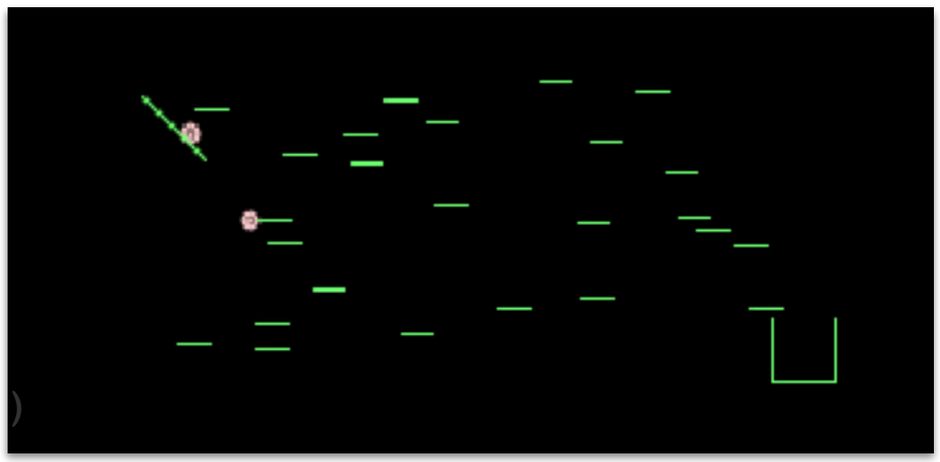
Generative Model

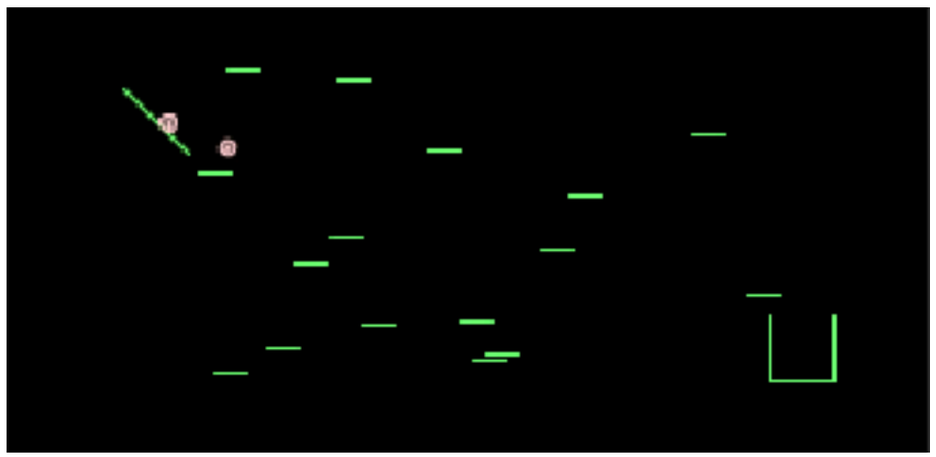
```
(defquery captcha
 [image num-chars tol]
 (let [[w h] (size image)
       ;; sample random characters
       num-chars (sample
                  (poisson num-chars))
       chars (repeatedly
              num-chars sample-char)]
  ;; compare rendering to true image
  (map (fn [y z]
         (observe (normal z tol) y))
       (reduce-dim image)
       (reduce-dim (render chars w h)))
 ;; predict captcha text
 {:text
  (map :symbol (sort-by :x chars))}))
```

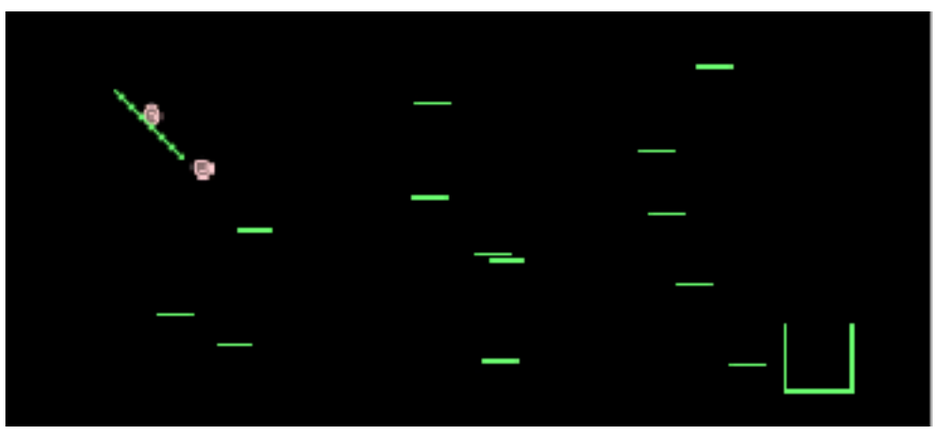
Mansinghka,, Kulkarni, Perov, and Tenenbaum
“Approximate Bayesian image interpretation using generative probabilistic graphics programs." NIPS (2013).

ANALOGY: RANDOM BUMPERS ~ RANDOM CALORIMETER SHOWER

```
(defquery arrange-bumpers []
    (let [number-of-bumpers (sample (poisson 20))
          bumpydist (uniform-continuous 0 10)
          bumpxdist (uniform-continuous -5 14)
          bumper-positions (repeatedly
                            number-of-bumpers
                            #(vector (sample bumpxdist)
                                     (sample bumpydist)))

          ;; code to simulate the world
          world (create-world bumper-positions)
          end-world (simulate-world world)
          balls (:balls end-world)

          ;; how many balls entered the box?
          num-balls-in-box (balls-in-box end-world)]

      {:balls balls
       :num-balls-in-box num-balls-in-box
       :bumper-positions bumper-positions}))
```





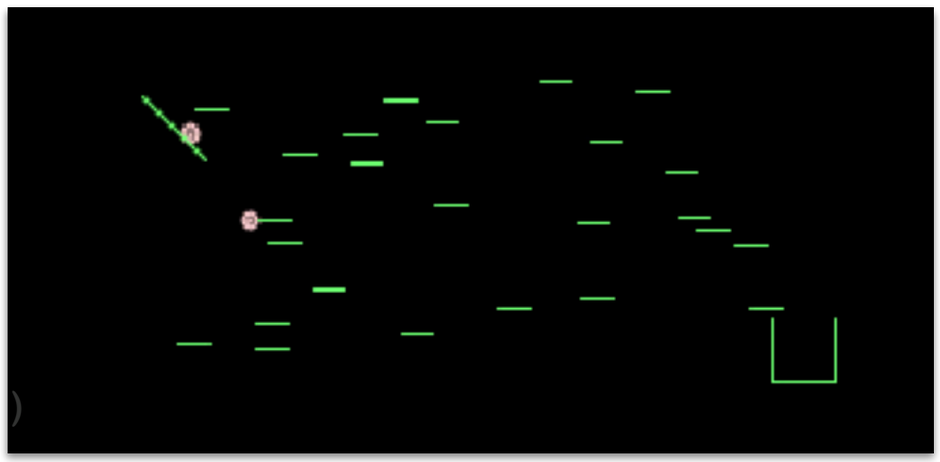


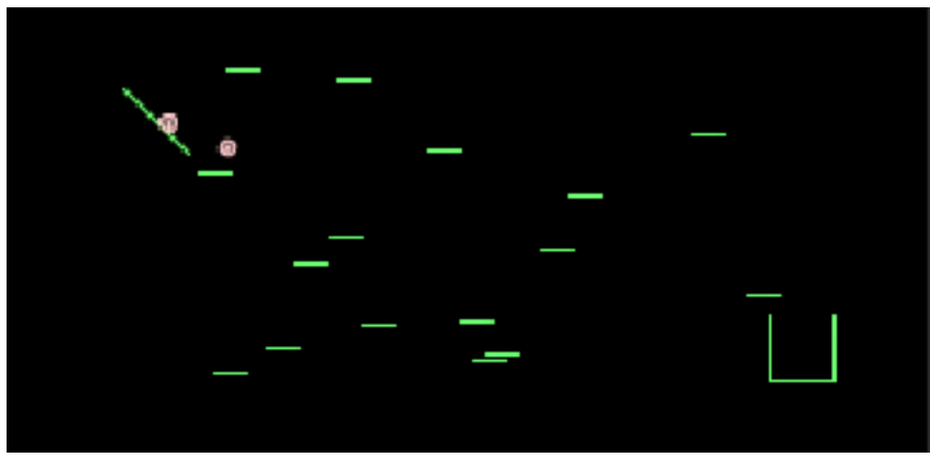
3 examples generated from simulator

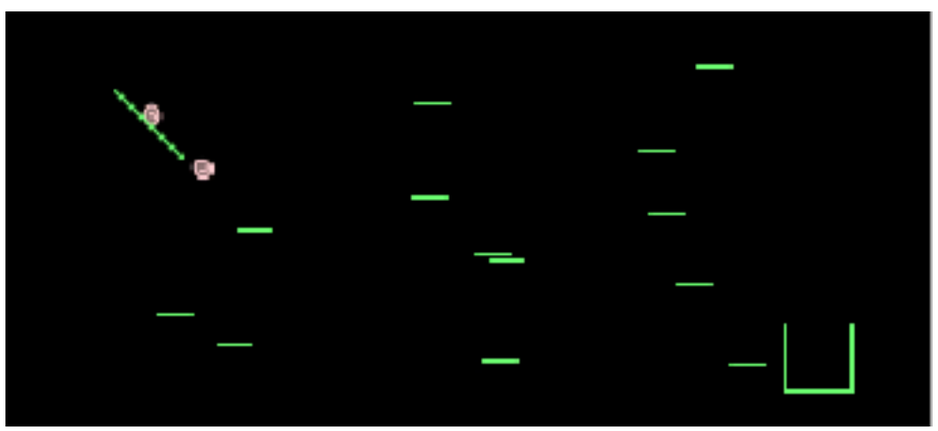
[slides, Frank Wood]

ANALOGY: RANDOM BUMPERS ~ RANDOM CALORIMETER SHOWER

```
(defquery arrange-bumpers []
    (let [number-of-bumpers (sample (poisson 20))
          bumpydist (uniform-continuous 0 10)
          bumpxdist (uniform-continuous -5 14)
          bumper-positions (repeatedly
                            number-of-bumpers
                            #(vector (sample bumpxdist)
                                     (sample bumpydist)))

          ;; code to simulate the world
          world (create-world bumper-positions)
          end-world (simulate-world world)
          balls (:balls end-world)

          ;; how many balls entered the box?
          num-balls-in-box (balls-in-box end-world)]

      {:balls balls
       :num-balls-in-box num-balls-in-box
       :bumper-positions bumper-positions}))
```





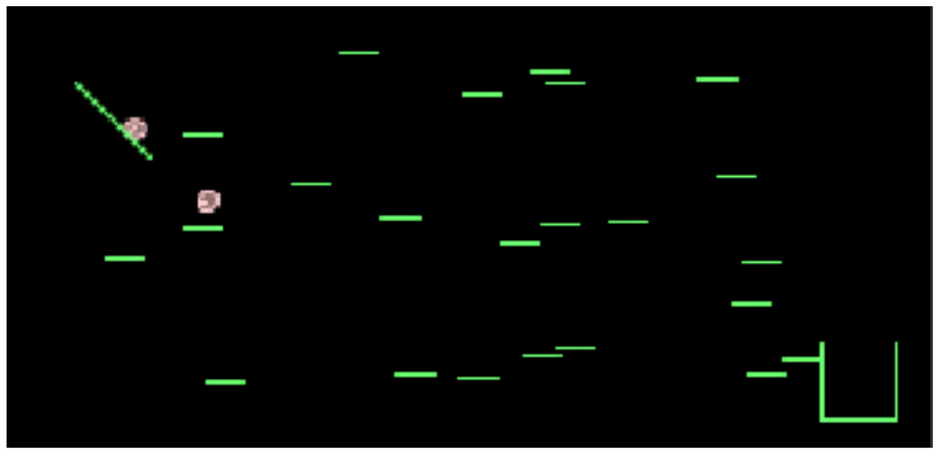


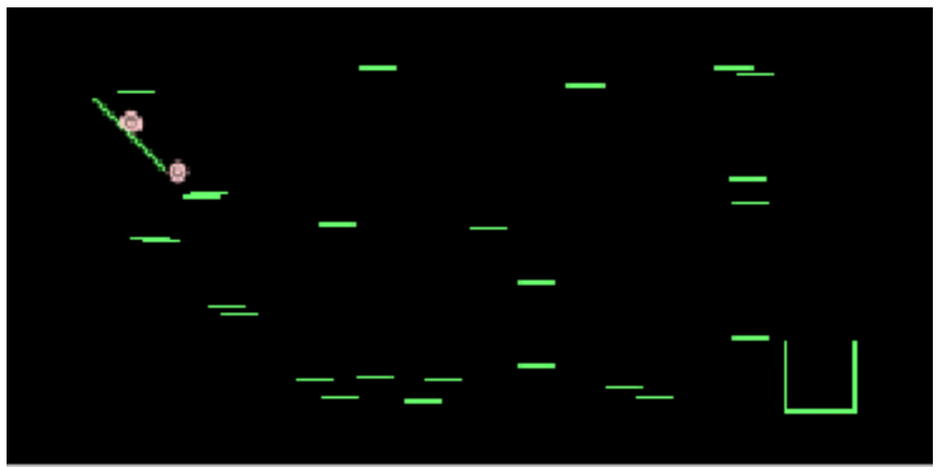
3 examples generated from simulator

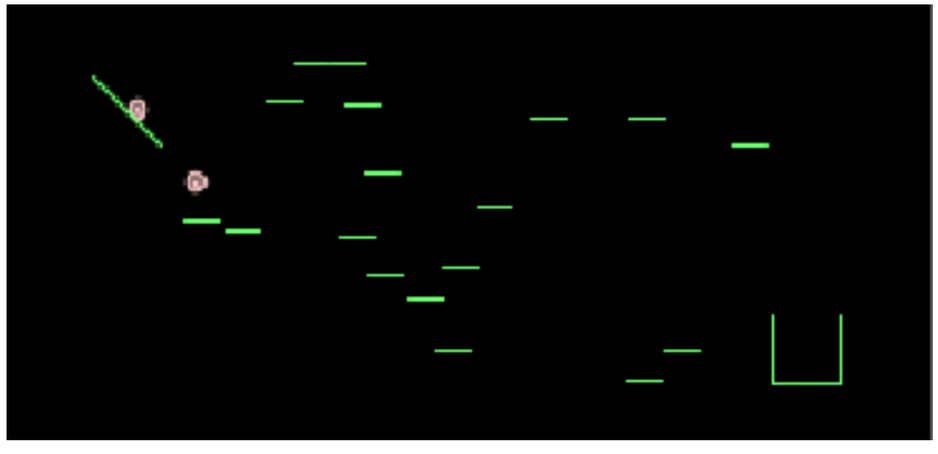
[slides, Frank Wood]

UNDERSTANDING THE TAILS OF DISTRIBUTIONS

```
(defquery arrange-bumpers []
    (let [number-of-bumpers (sample (poisson 20))
          bumpydist (uniform-continuous 0 10)
          bumpxdist (uniform-continuous -5 14)
          bumper-positions (repeatedly
                            number-of-bumpers
                            #(vector (sample bumpxdist)
                                     (sample bumpydist)))

          ;; code to simulate the world
          world (create-world bumper-positions)
          end-world (simulate-world world)
          balls (:balls end-world)

          ;; how many balls entered the box?
          num-balls-in-box (balls-in-box end-world)

          obs-dist (normal 4 0.1)]

    (observe obs-dist num-balls-in-box)
```





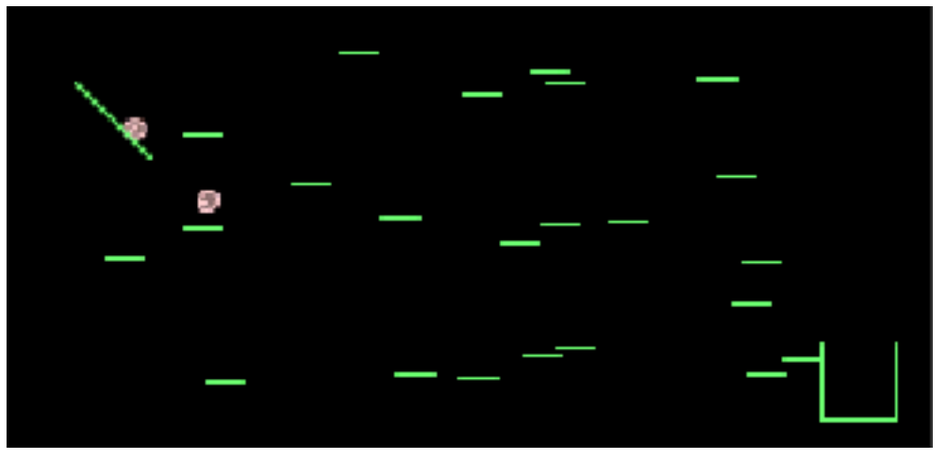


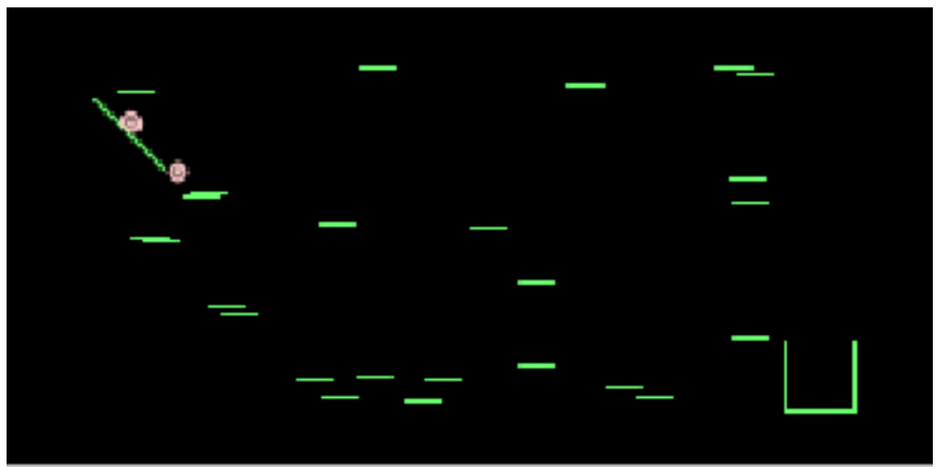
3 examples generated from simulator
conditioned on ~20% of balls land in box
(~ given observed energy deposits)

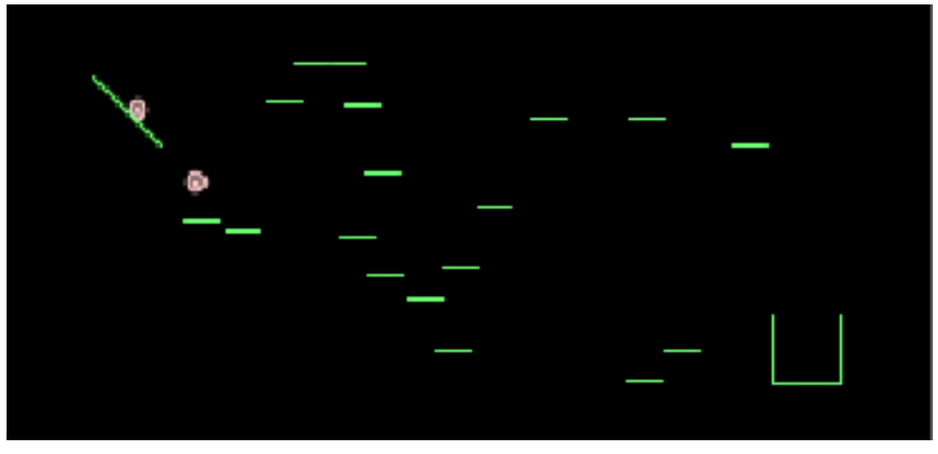
[slides, Frank Wood]

UNDERSTANDING THE TAILS OF DISTRIBUTIONS

```
(defquery arrange-bumpers []
    (let [number-of-bumpers (sample (poisson 20))
          bumpydist (uniform-continuous 0 10)
          bumpxdist (uniform-continuous -5 14)
          bumper-positions (repeatedly
                            number-of-bumpers
                            #(vector (sample bumpxdist)
                                     (sample bumpydist)))

          ;; code to simulate the world
          world (create-world bumper-positions)
          end-world (simulate-world world)
          balls (:balls end-world)

          ;; how many balls entered the box?
          num-balls-in-box (balls-in-box end-world)

          obs-dist (normal 4 0.1)]

    (observe obs-dist num-balls-in-box)
```







3 examples generated from simulator
conditioned on ~20% of balls land in box
(~ given observed energy deposits)

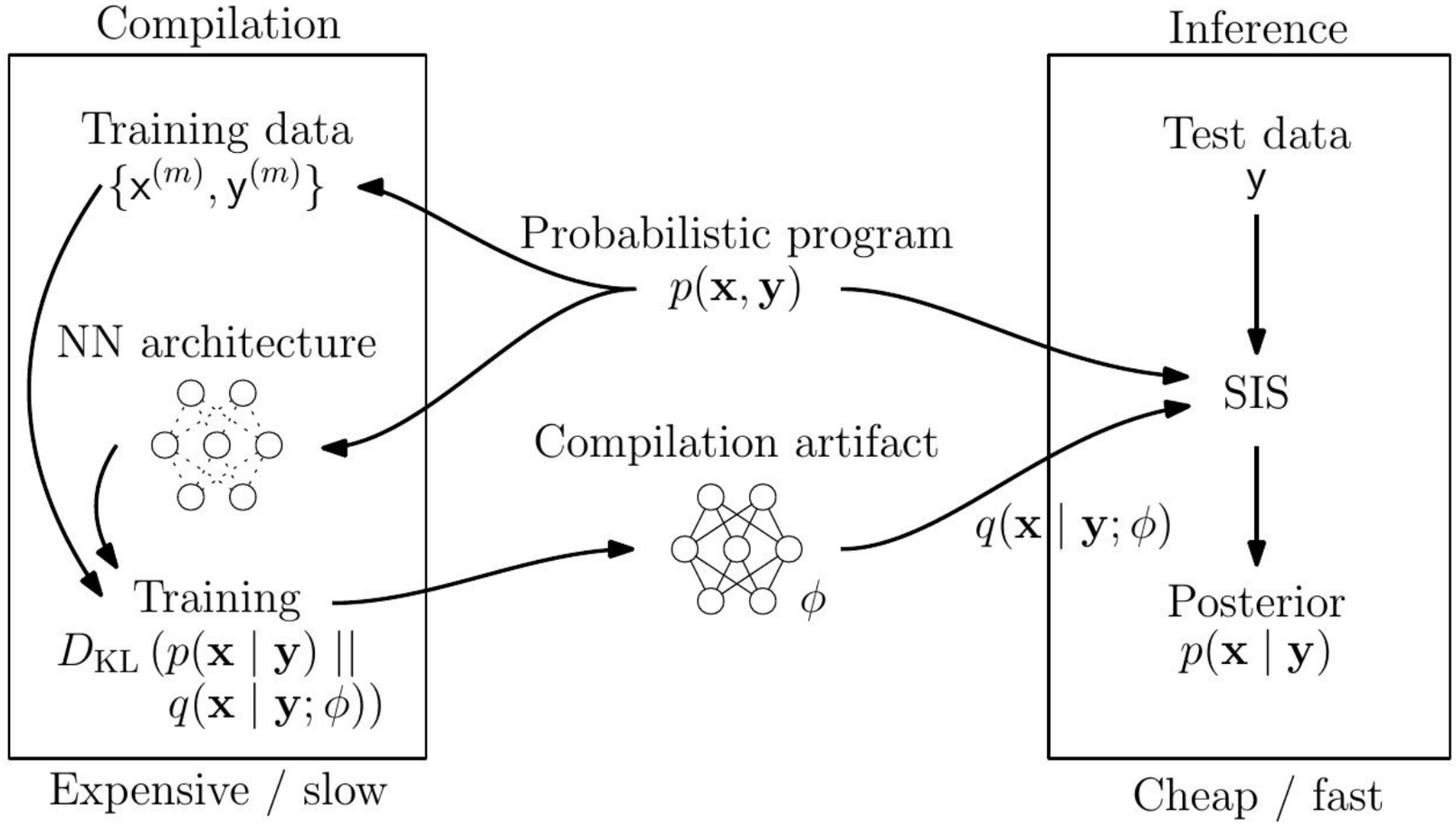
[slides, Frank Wood]

HOW DOES IT WORK?

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*

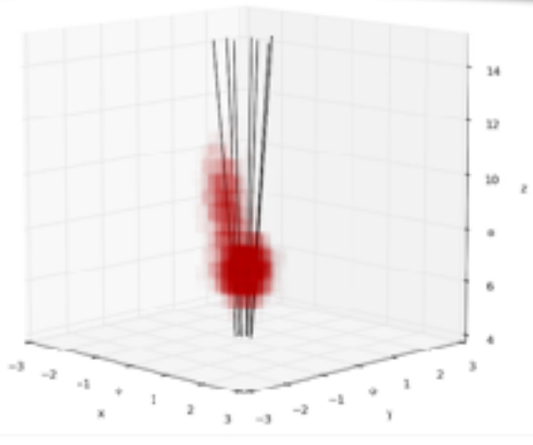
IN PROGRESS: C++, SHERPA, GEANT4

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);
}
```

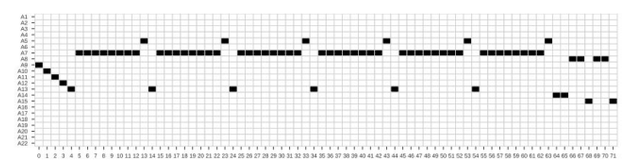
```
SHERPA::Hadron_Decays::Treat(ATOOLS::Blob_List*, double&)+0x709
SHERPA::Event_Handler::IterateEventPhases(SHERPA::eventtype::code&, double&)+0x1b2
SHERPA::Event_Handler::GenerateHadronDecayEvent(SHERPA::eventtype::code&)+0x979
```

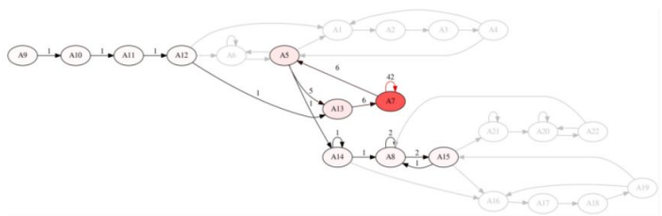
A case study in SHERPA & GEANT

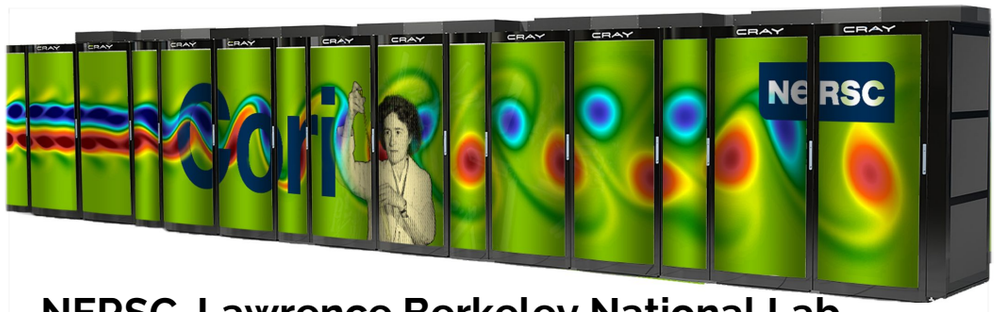
Probabilistic program analytics
allows us to **pinpoint "interesting" addresses** in execution traces
and corresponding **C++ code within SHERPA**

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

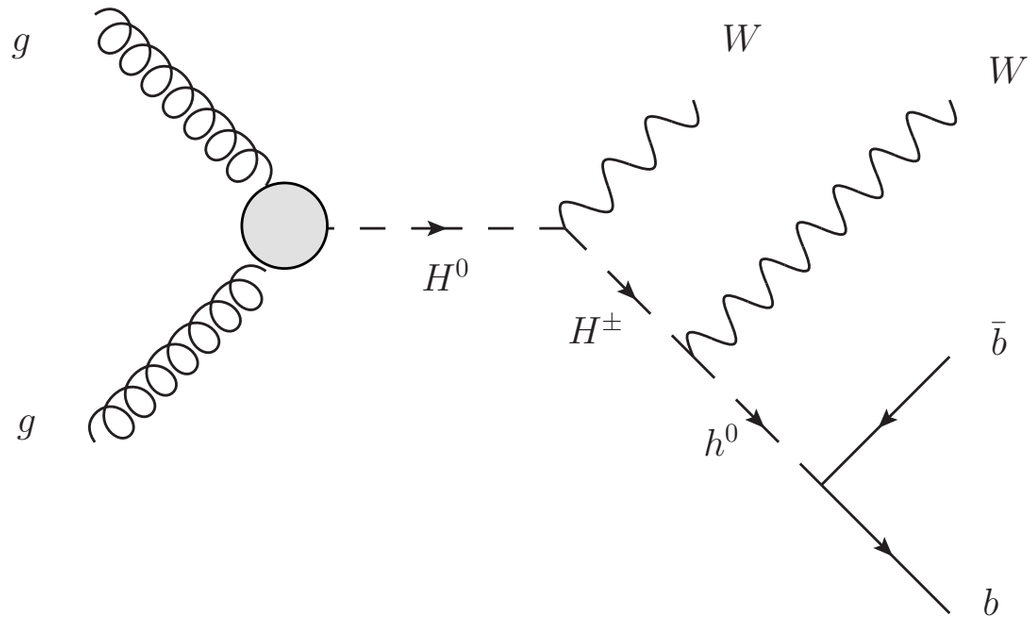
shifterimg -v pull docker:gbaydin/pytorch-infcomp:latest
shifterimg -v pull docker:gbaydin/sherpa-infcomp-full:latest

docker

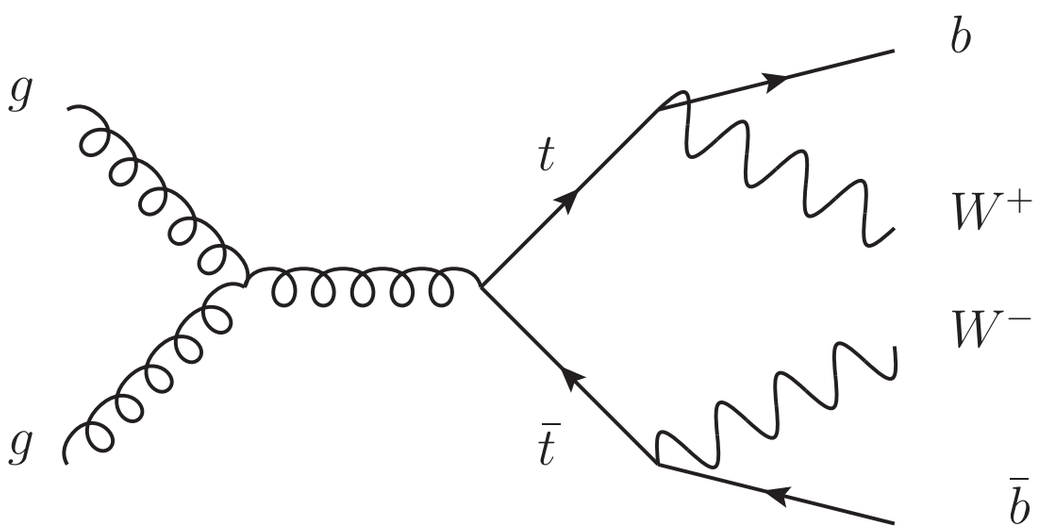
Examples

DEEP LEARNING IN HEP

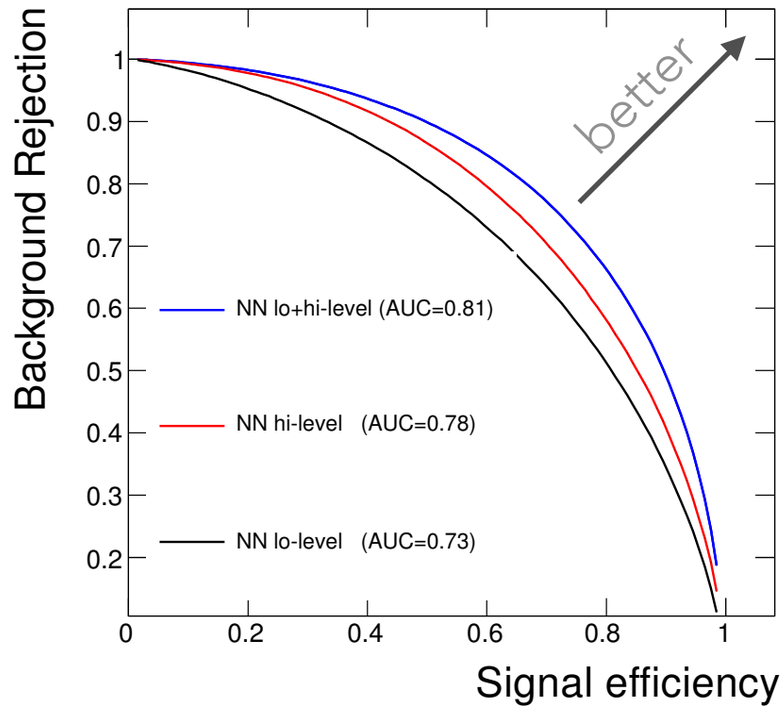
Baldi, Sadowski,Whiteson
arxiv:1402.4735



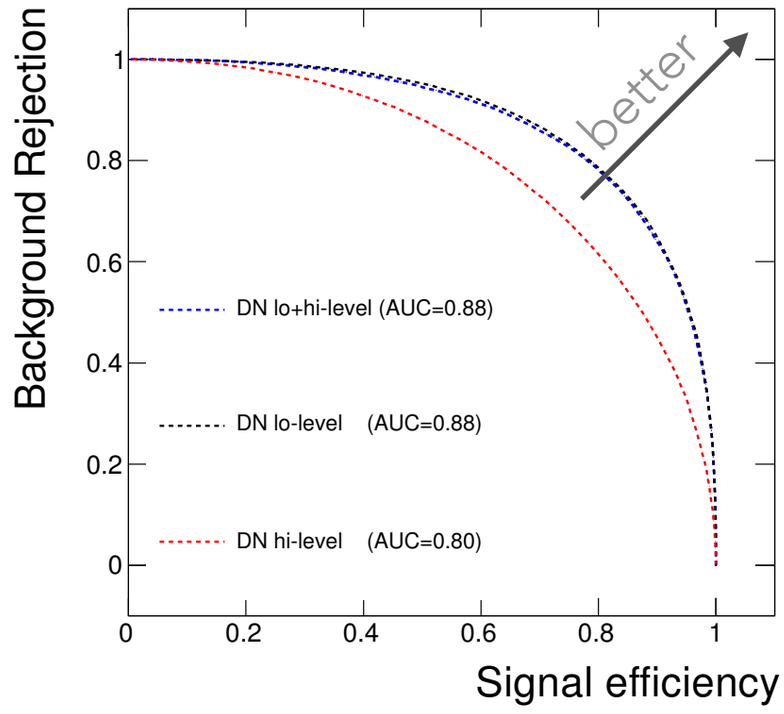
(a)



(b)



(a)



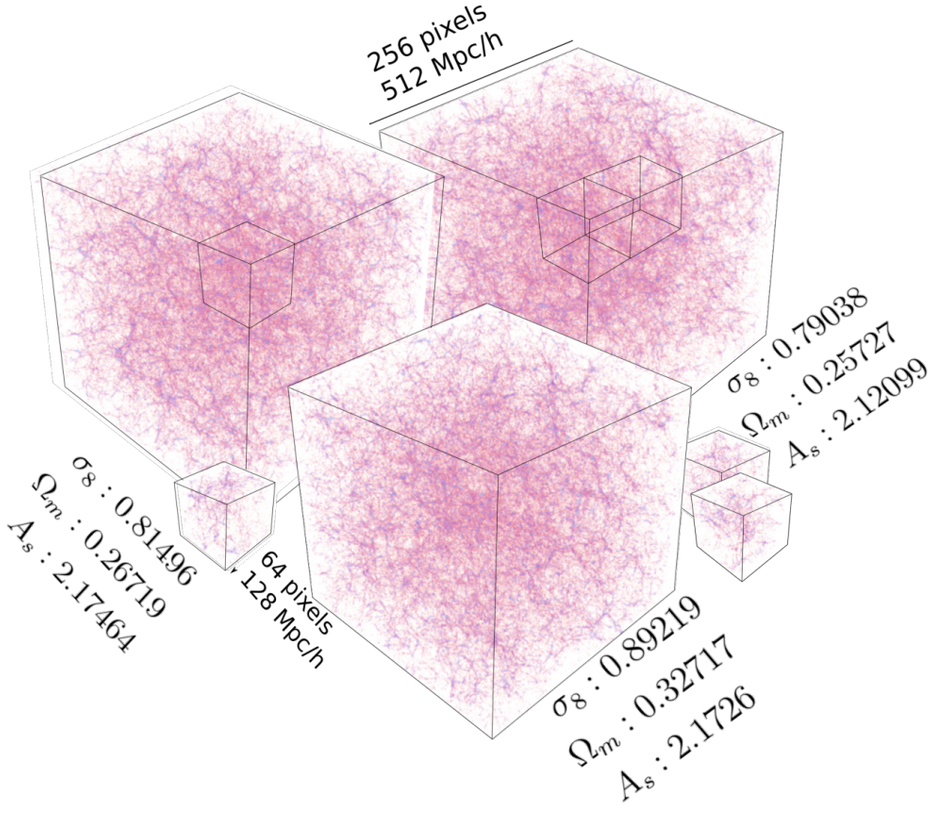
(b)

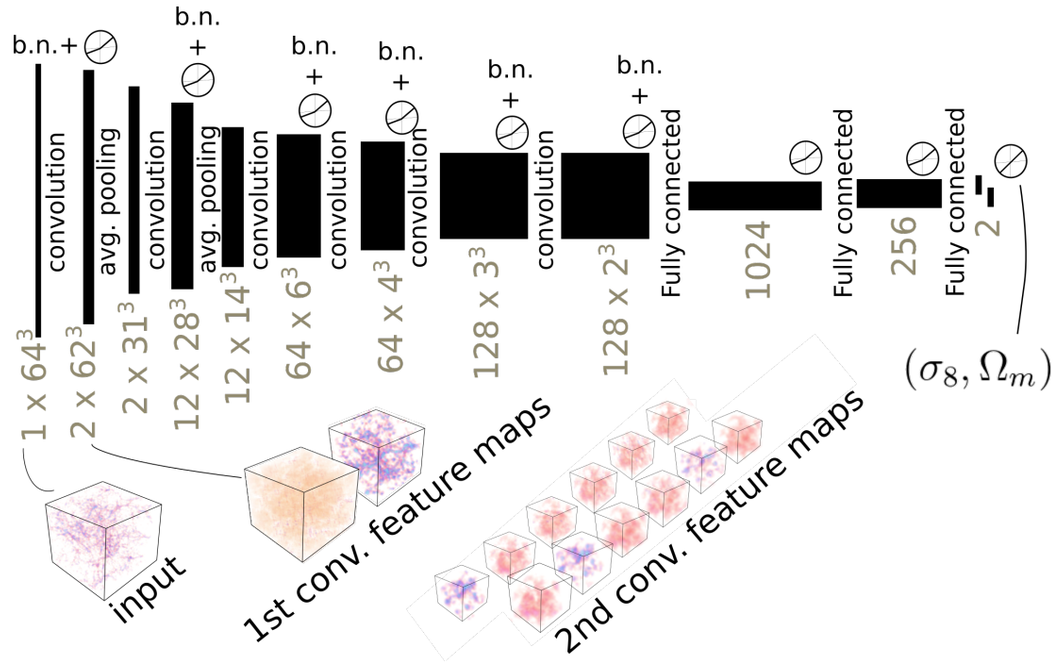
REGRESSION FOR Ω_M, σ_8

Estimating Cosmological Parameters from the Dark Matter Distribution

Siamak Ravanbakhsh⋆ MRAVANBA@CS.CMU.EDU
Junier Oliva⋆ JOLIVA@CS.CMU.EDU
Sebastien Fromenteau† SFROMENT@ANDREW.CMU.EDU
Layne C. Price† LAYNEP@ANDREW.CMU.EDU
Shirley Ho† SHIRLEYH@ANDREW.CMU.EDU
Jeff Schneider⋆ JEFF.SCHNEIDER@CS.CMU.EDU
Barnabás Póczos⋆ BAPOCZOS@CS.CMU.EDU

⋆ School of Computer Science, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213, USA
† McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Carnegie 5000 Forbes Ave., Pittsburgh, PA 15213, USA





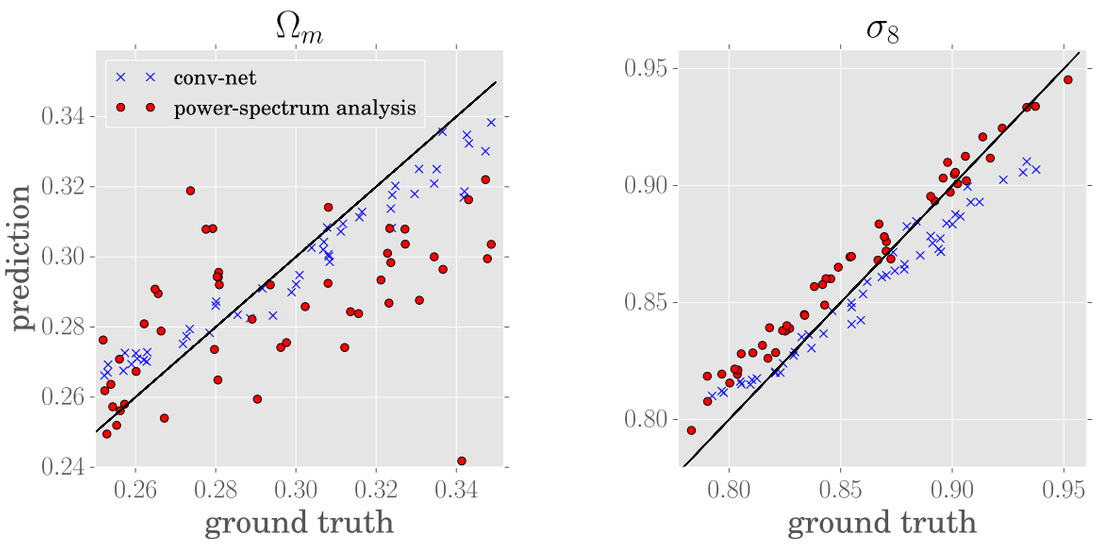


Figure 2. Prediction and ground truth of Ω_m and σ_8 using 3D conv-net and analysis of the power-spectrum on 50 test cube instances.

NEUTRINOS

Applications of Convolutional Neutral Networks and Deep Learning to the Intensity Frontier

NOvA

- CVN-based event classification is used to categorize events as cosmic-ray, neutral-current, νe-charged-current, νμ-CC, and ντ-CC.
- CVN-based algorithms contributed one of the most important improvements in our recent electron neutrino appearance results; equivalent to a 30% increase in statistics.
- Now exploring impacts of these methods elsewhere in reconstruction: clustering, prong-by-prong particle identification, and energy estimation.

A convolutional neural network neutrino event classifier

A. Aurisano, A. Radovic, D. Rocco, A. Himmel, M.D. Messier, E. Niner, G. Pawloski, F. Psihas, A. Sousa and P. Vahle

Published 1 September 2016 • © 2016 IOP Publishing Ltd and Sissa Medialab srl

Journal of Instrumentation, Volume 11, September 2016

Constraints on Oscillation Parameters from ν_e Appearance and ν_μ Disappearance in NOvA

P. Adamson et al. (NOvA Collaboration)

Phys. Rev. Lett. 118, 231801 – Published 5 June 2017

MicroBooNE

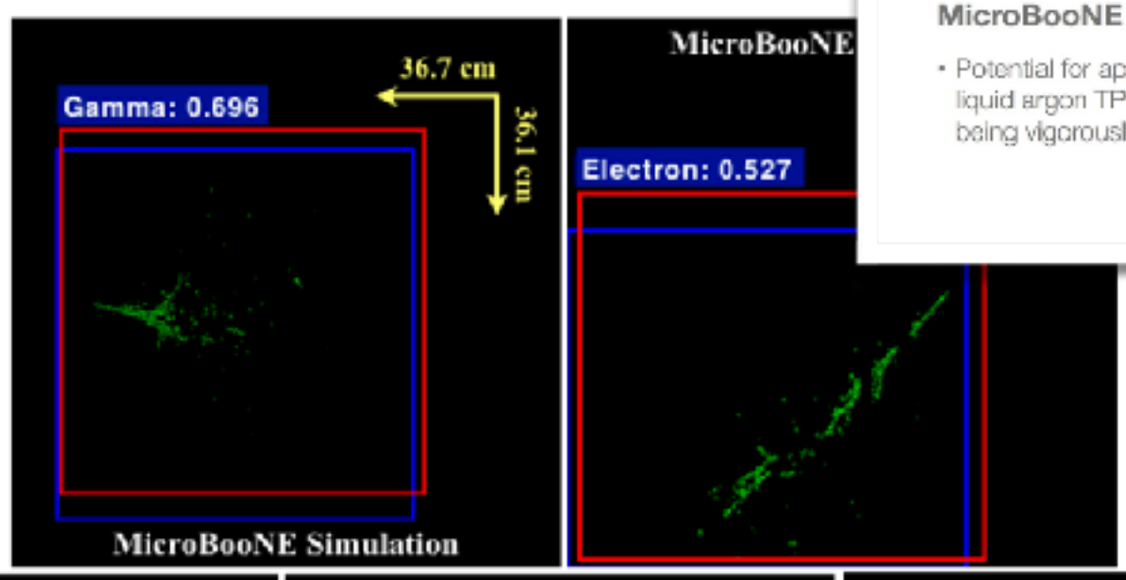
- Potential for applications to highly-granular liquid argon TPC detectors is huge and is being vigorously explored.

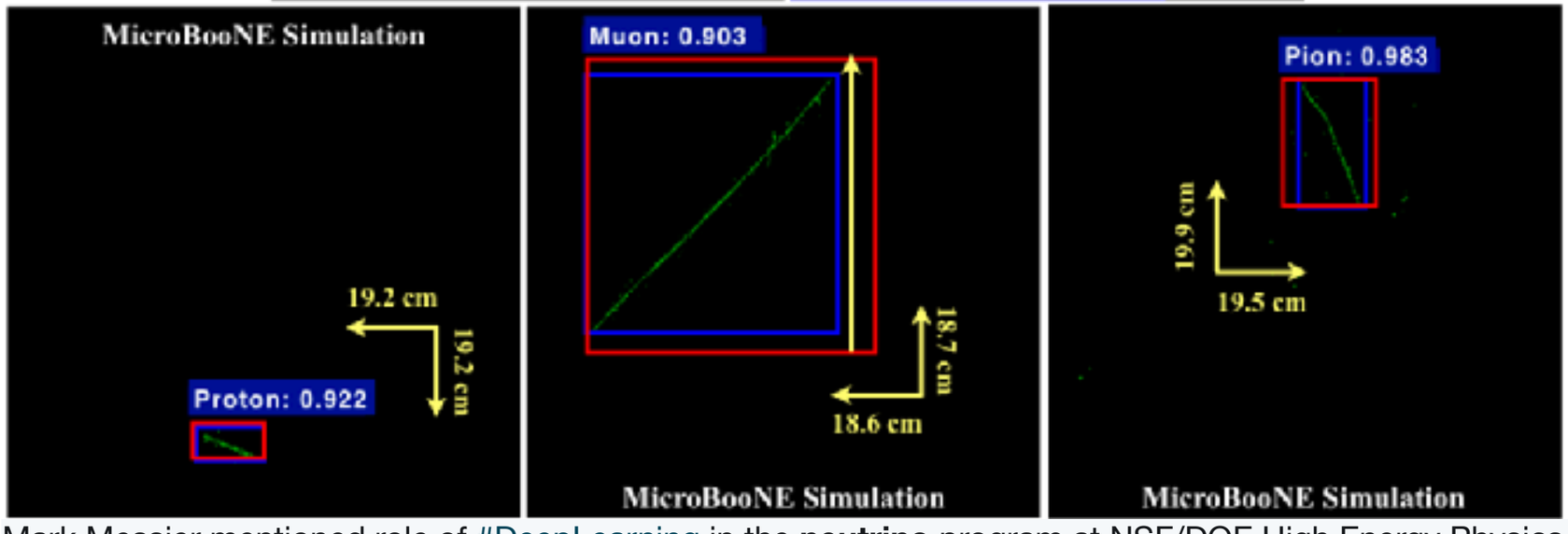
Convolutional neural networks applied to neutrino events in a liquid argon time projection chamber

R. Acciarri, C. Adams, R. An, J. Asaadi, M. Auger, L. Bagby, B. Ballew, G. Barr, M. Bass, F. Bay

Published 14 March 2017 • © 2017 IOP Publishing Ltd and Sissa Medialab srl

Journal of Instrumentation, Volume 12, March 2017





Mark Messier mentioned role of #DeepLearning in the **neutrino** program at NSF/DOE High Energy Physics Advisory Panel

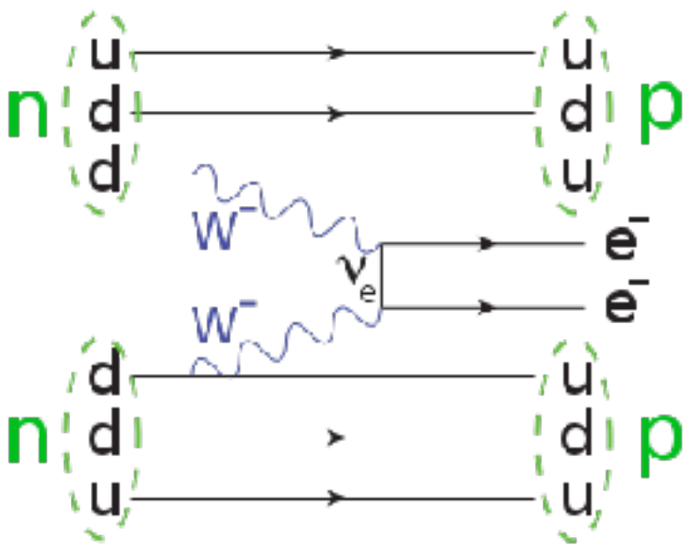
NEUTRINOLESS DOUBLE BETA DECAY

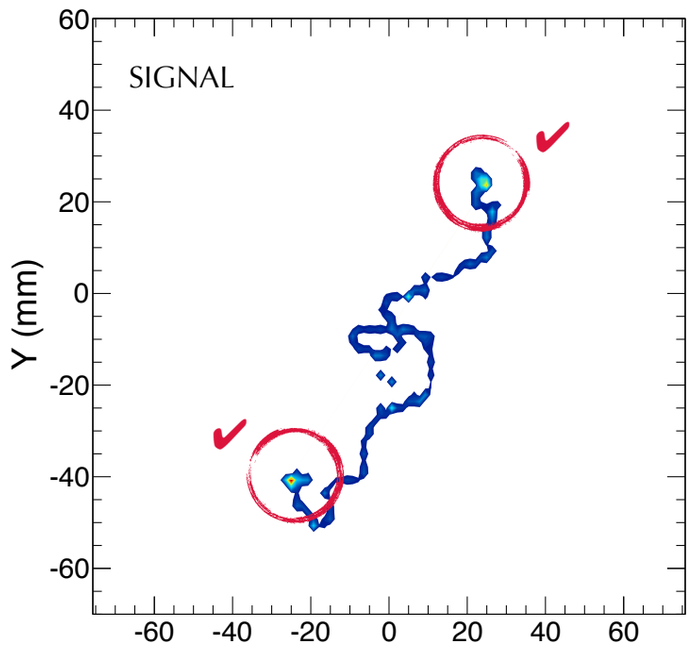
Background rejection in NEXT using deep neural networks

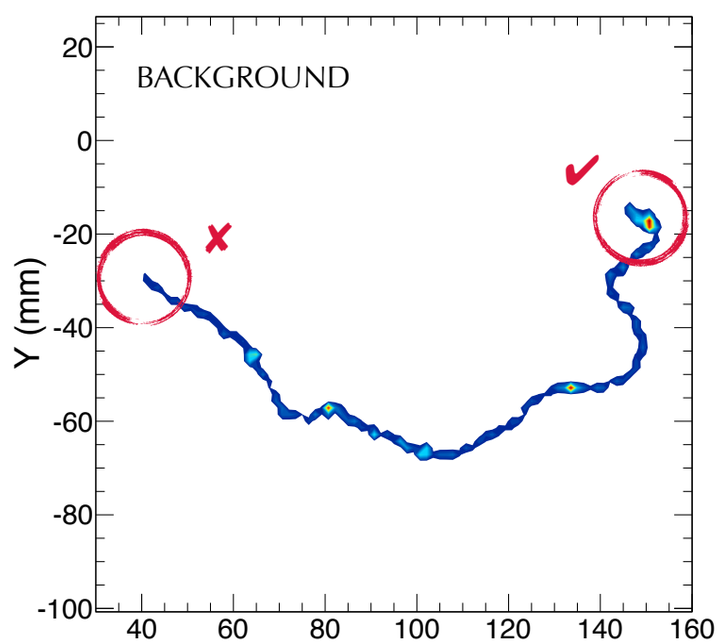
next **arXiv:1609.06202**

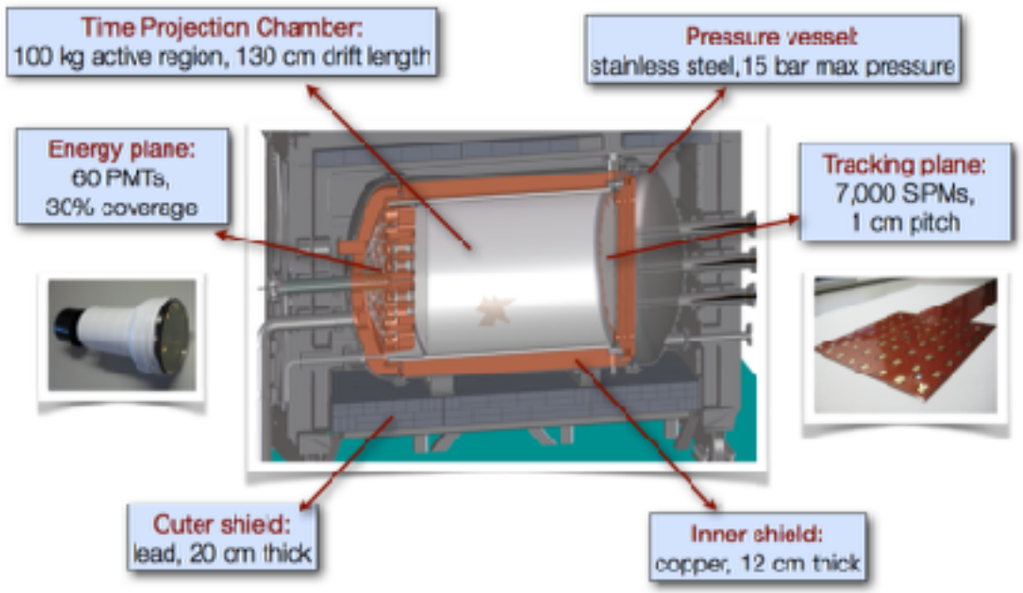
NEXT collaboration

NEXT is a high pressure Xenon (HPXe) Time Projection Chamber (TPC).



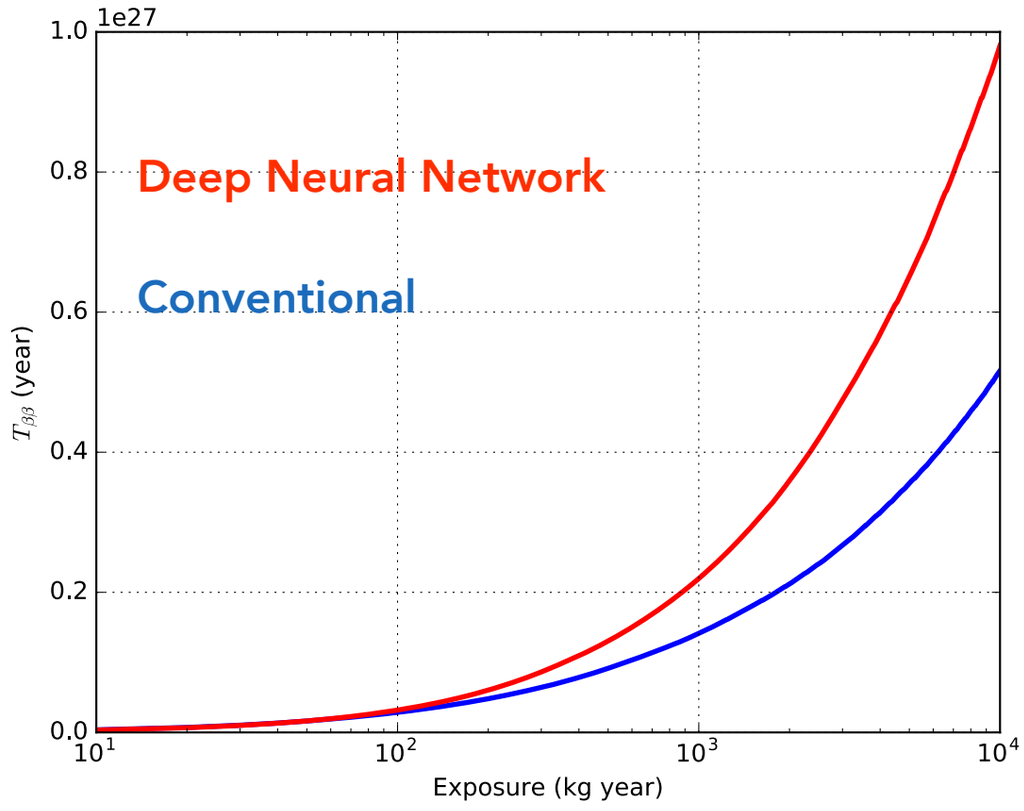






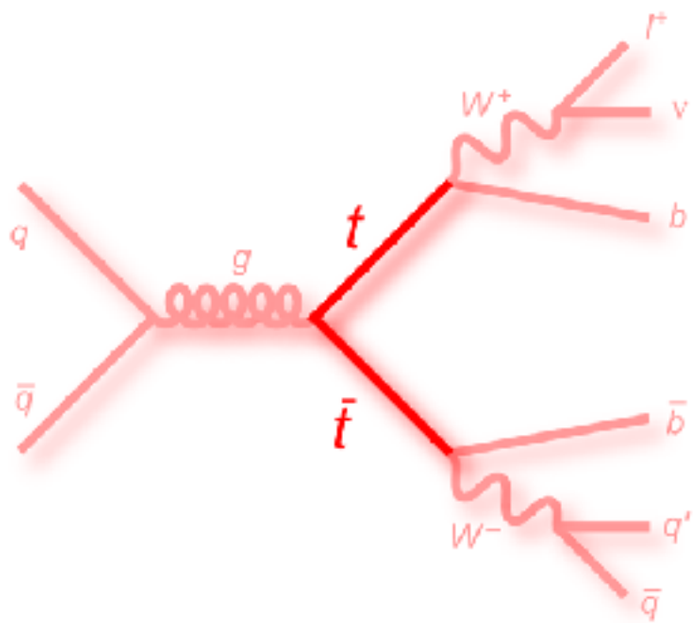
NEXT-100

Neutrinoless double beta decay; from 2017.

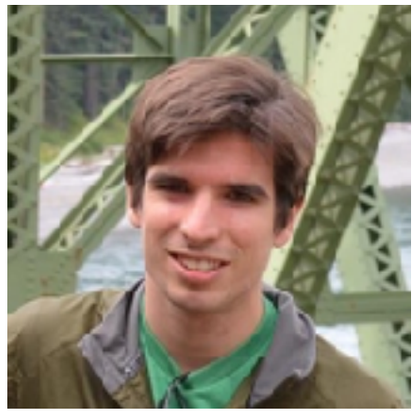


PARAMETRIZED CLASSIFIERS WITH DNN

Example: $Z' \rightarrow t\bar{t}$

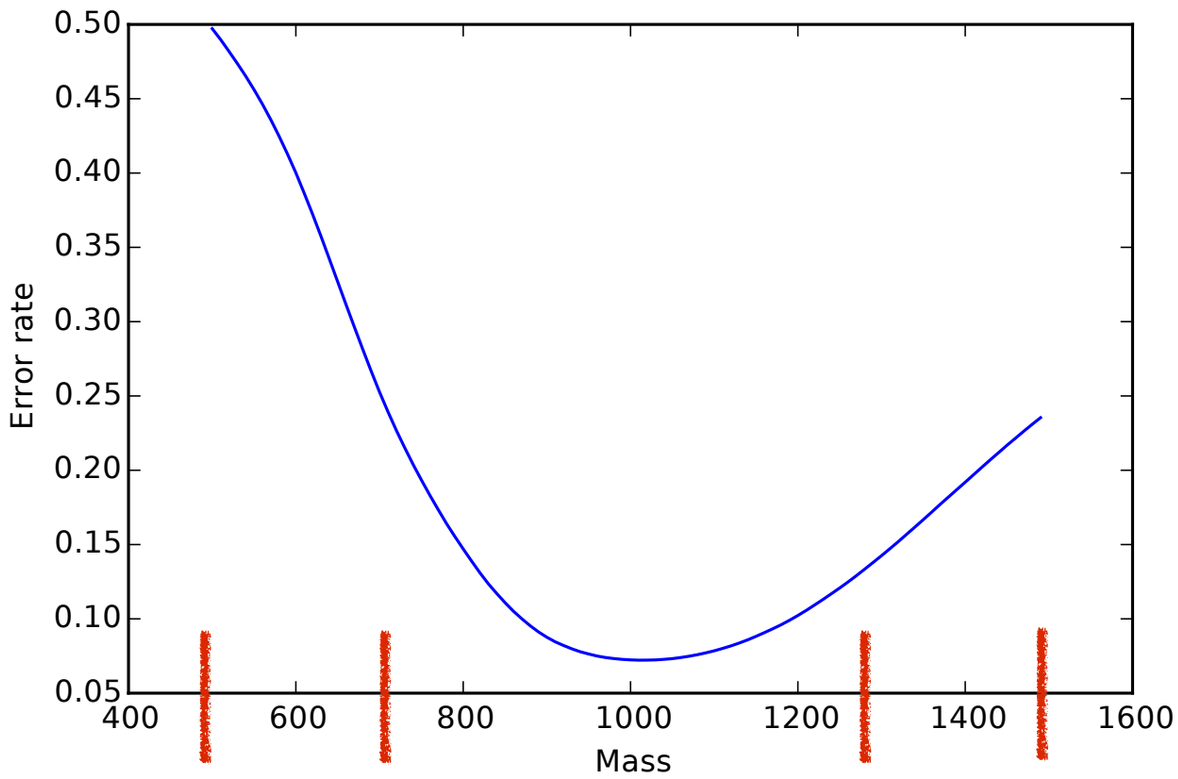


together with:




Peter Sadowski , Daniel Whiteson, Pierre Baldi, Taylor Faucett

The networks were trained on 28 features: 22 low-level, 5 high-level, and the mass



Train at $m_{Z'}$=500,750,1250,1500 GeV

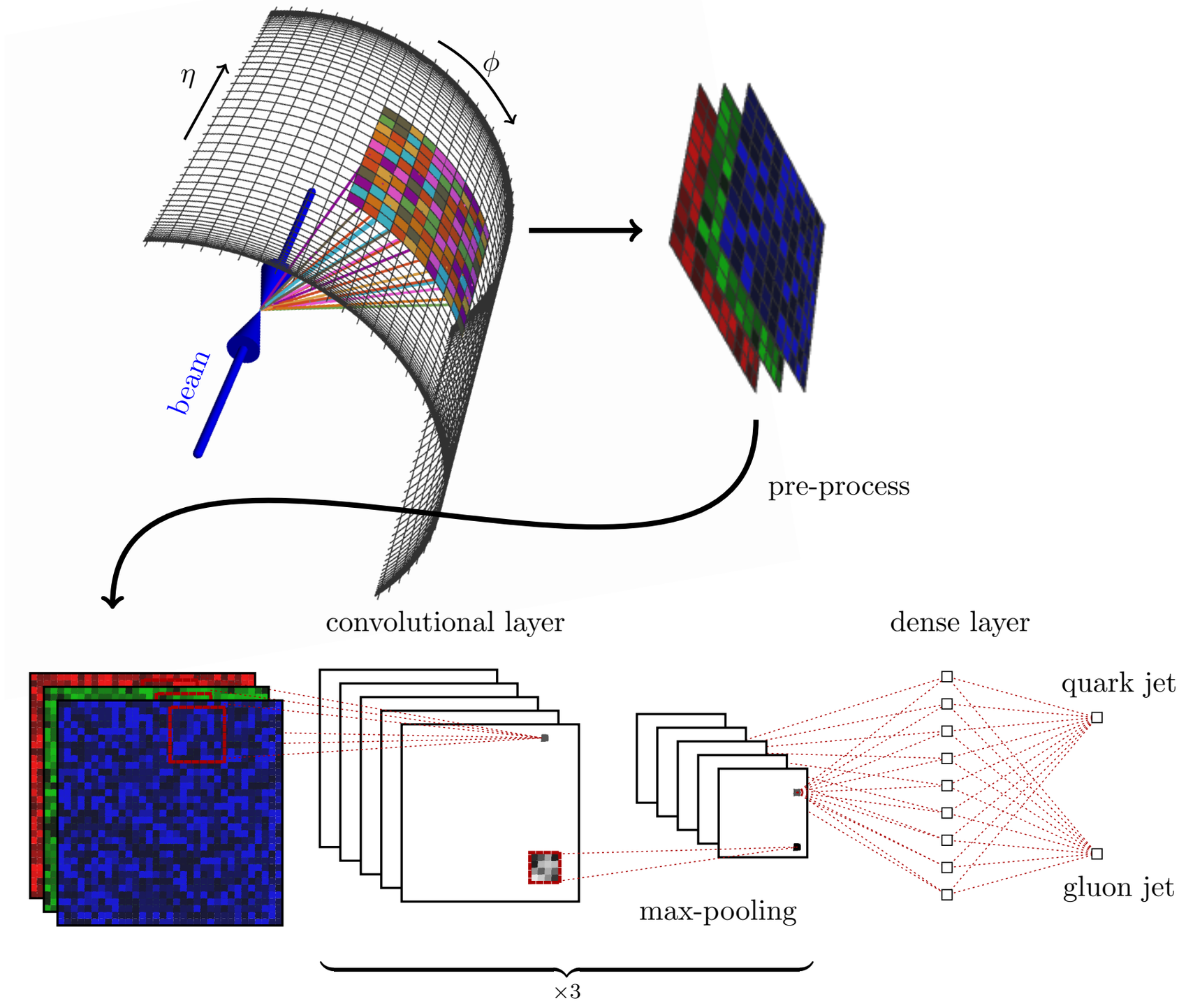
Almost identical performance to dedicated training at $m_{Z'}$=1000 GeV

JET IMAGES

image: Komiske, Metodiev, Schwartz arxiv:1612.01551
Oliveira, et. al arXiv:1511.05190
Whiteson, et al arXiv:1603.09349
Barnard, et al arXiv:1609.00607

Jet = Collimated spray of particles coming from quark / gluon

"We supplement this construction by adding color to the images, with **red, green and blue** intensities given by the transverse momentum in **charged** particles, transverse momentum in **neutral** particles, and pixel-level charged particle **counts**."



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

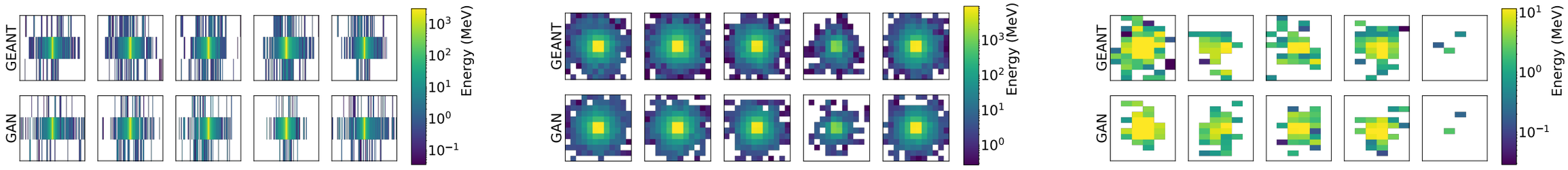


Figure 9: Five randomly selected e^+ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

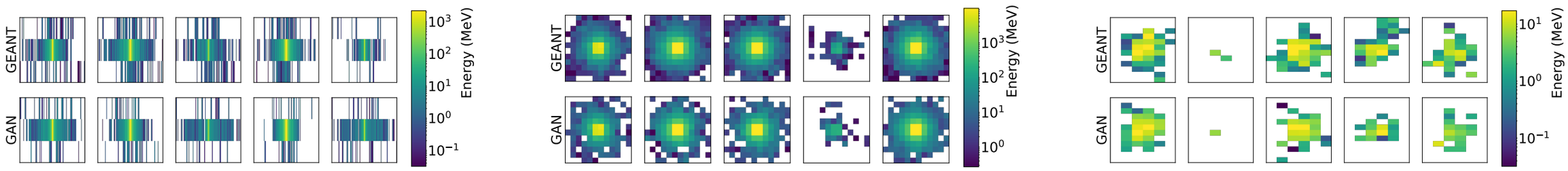


Figure 10: Five randomly selected γ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

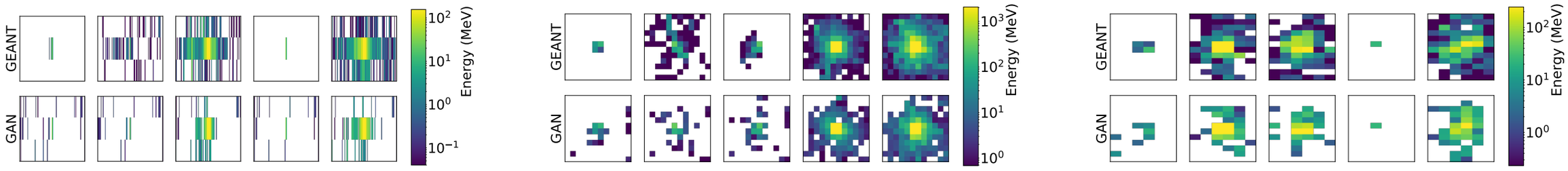
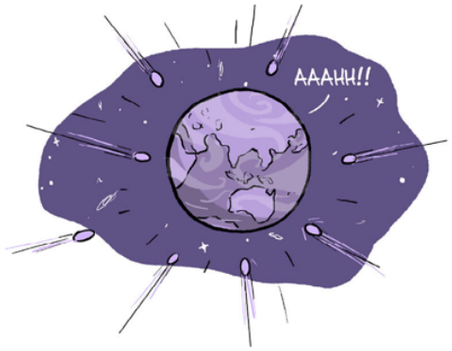


Figure 11: Five randomly selected π^+ showers per calorimeter layer from the training set (top) and the five nearest neighbors (by euclidean distance) from a set of CaloGAN candidates.

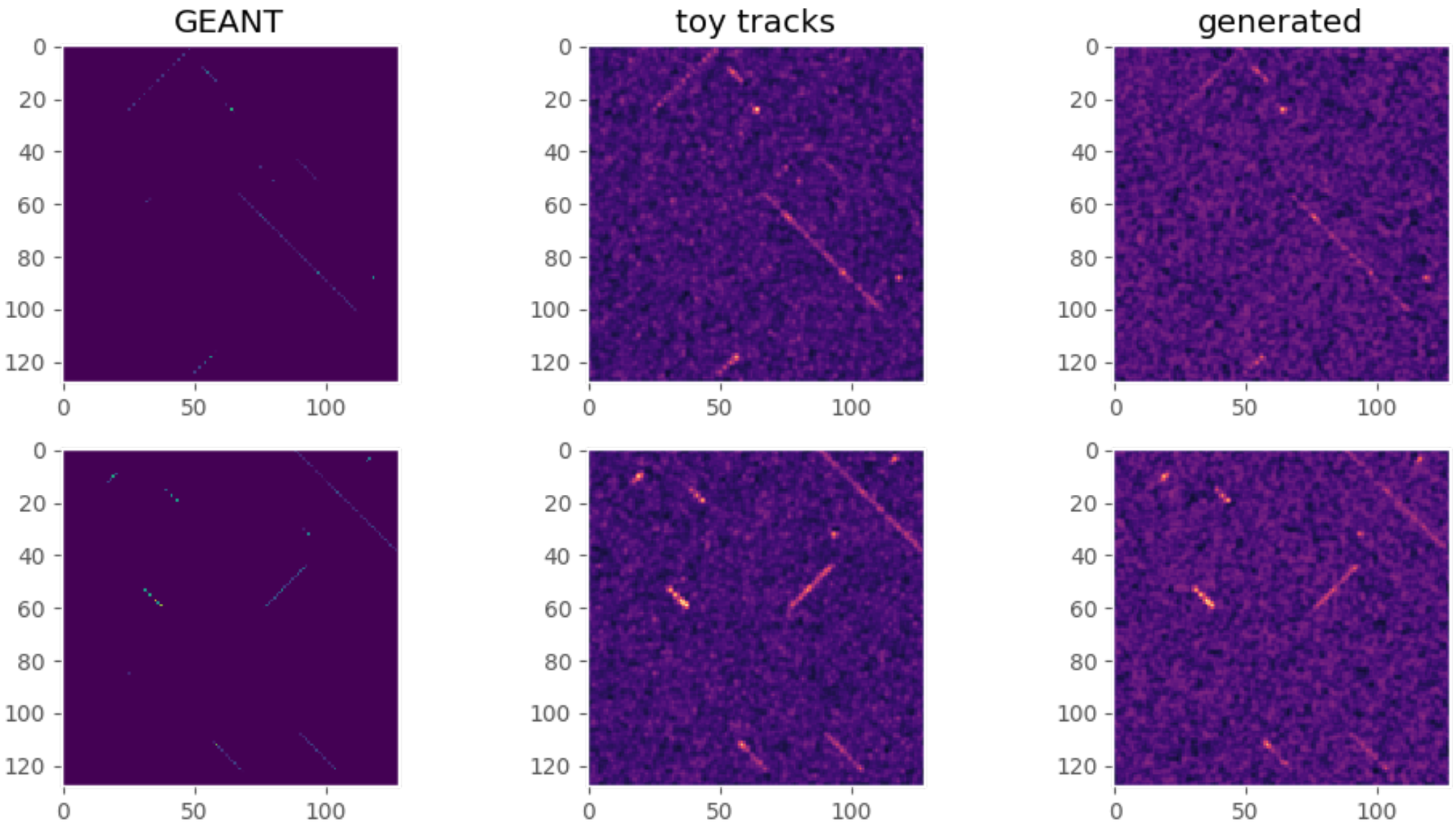
GANs COSMIC RAYS

CRAYFIS in a nutshell



Illustrations from the book "We have no idea" by D. Whiteson, J. Cham.

Maxim Borisyak[1], Chase Shimmin[3], Andy Nelson[2], Andrey Ustyuzhanin[1]

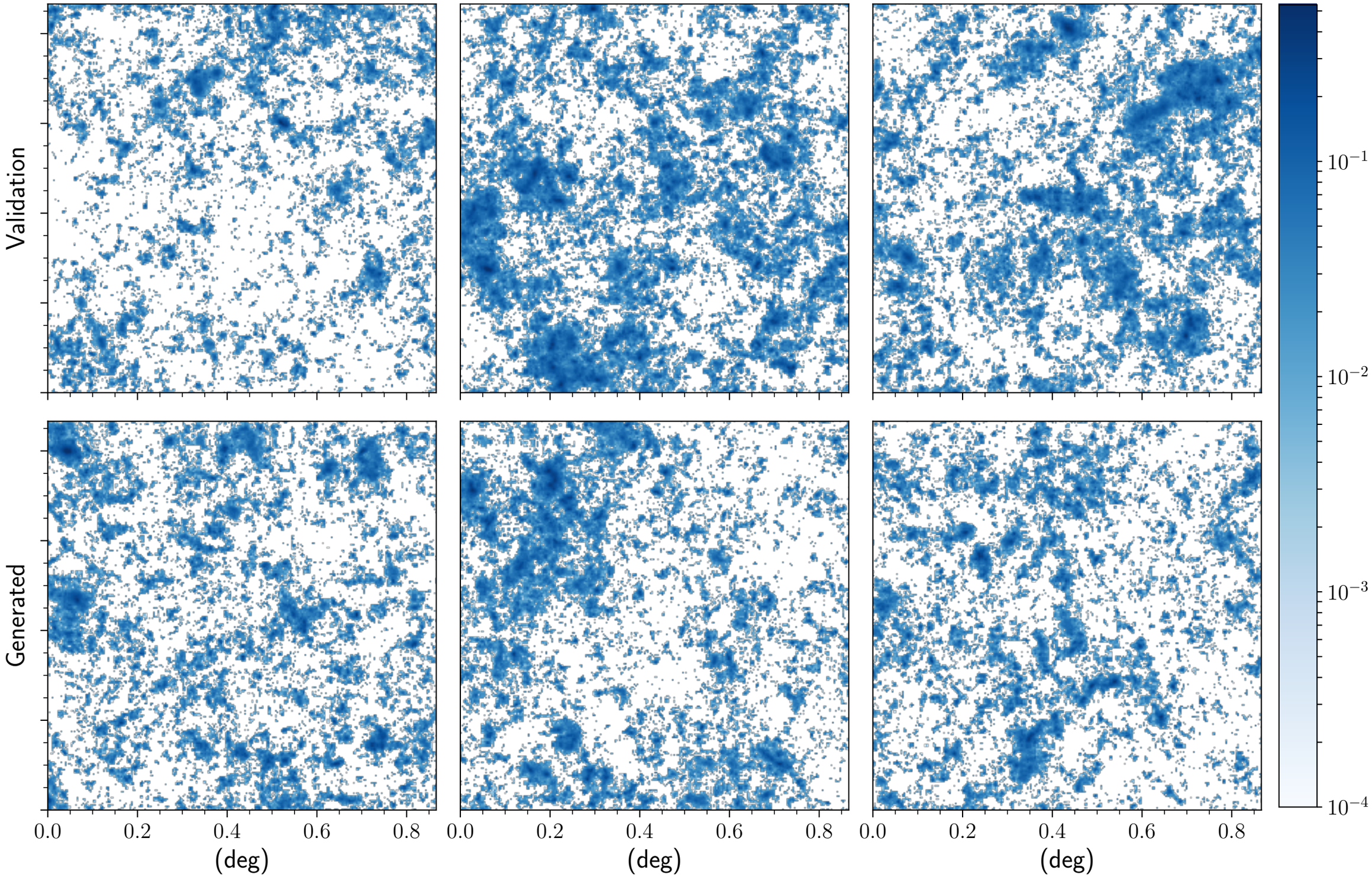


GANs FOR PHYSICS

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



GENERATIVE MODELS FOR CALIBRATION

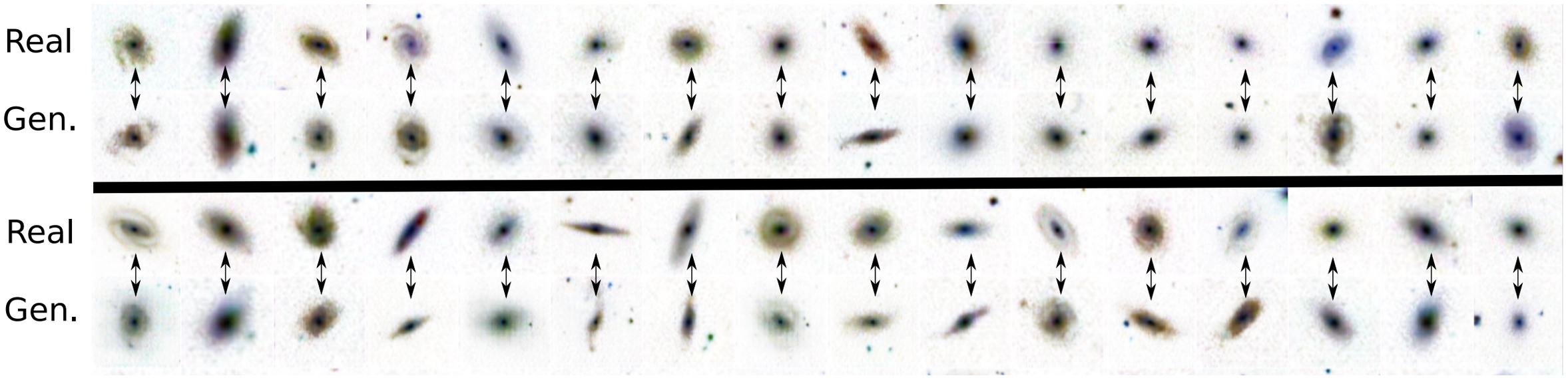
Use of generative models of galaxy images to help calibrate down-stream analysis in next-generation surveys.

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.**



Active Sciencing

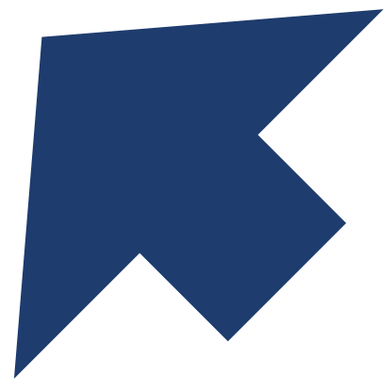
https://github.com/cranmer/active_sciencing

SYNTHESIS

active learning / sequential design / black box optimization

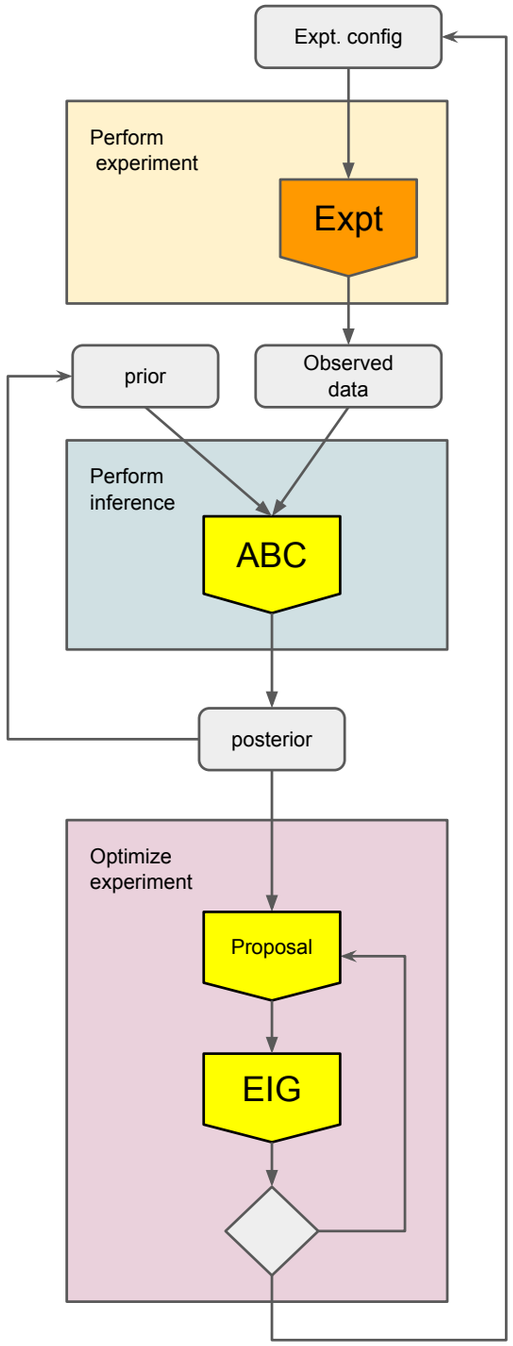
Active Sciencing

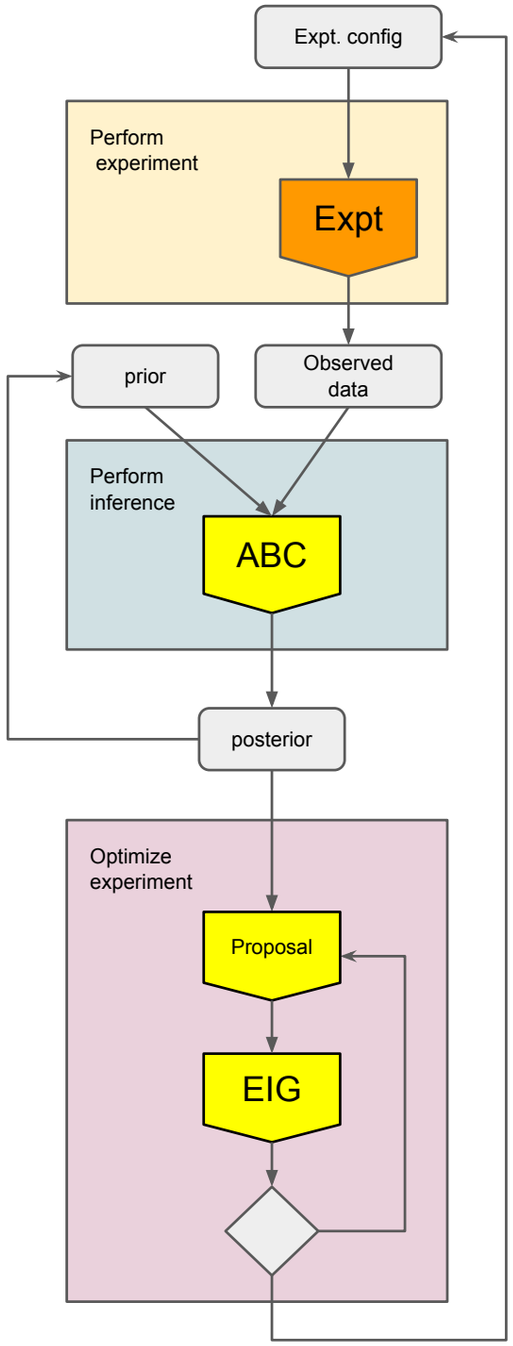
reusable workflows

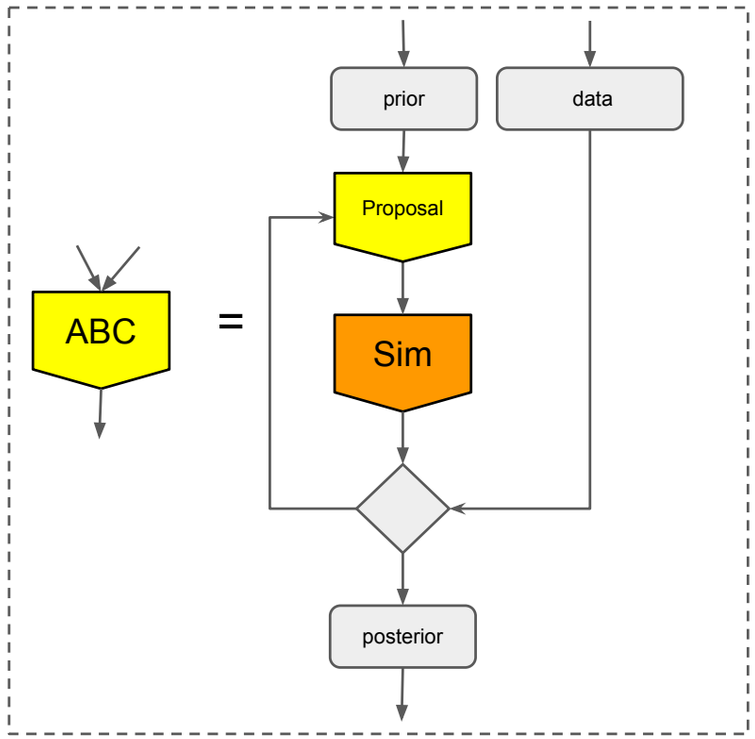


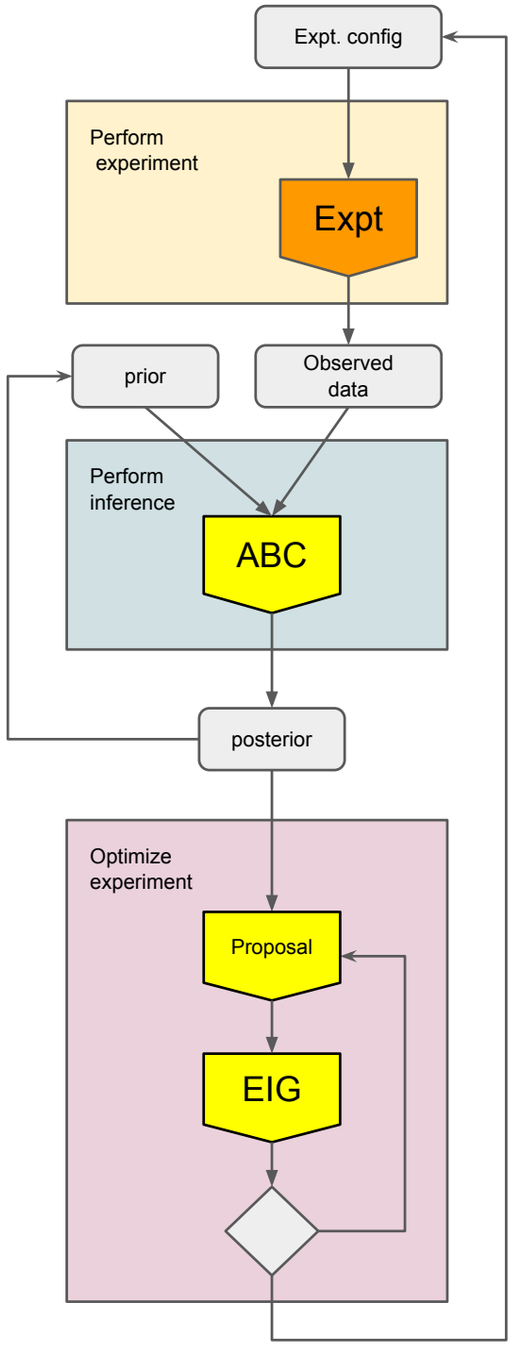
simulation-based
inference engines

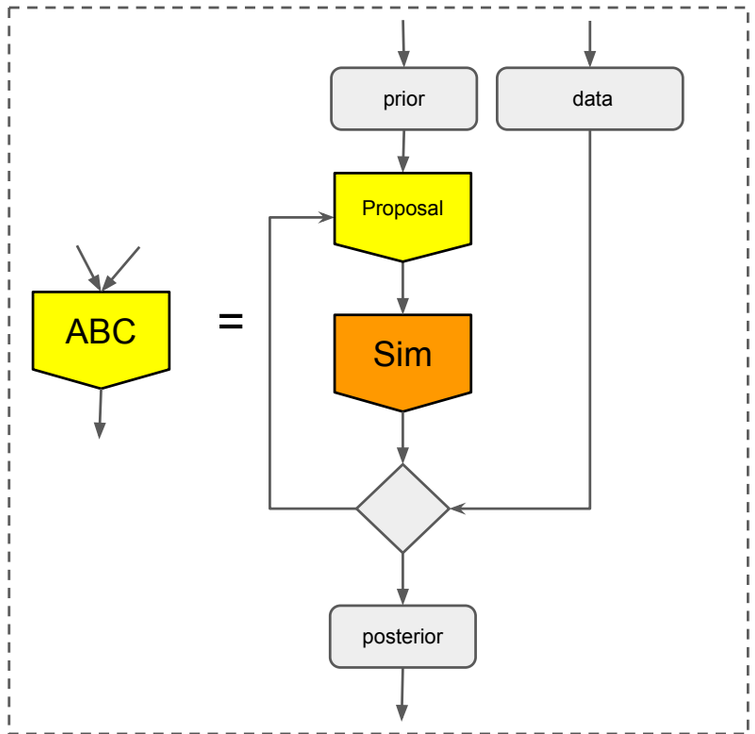
https://github.com/cranmer/active_sciencing

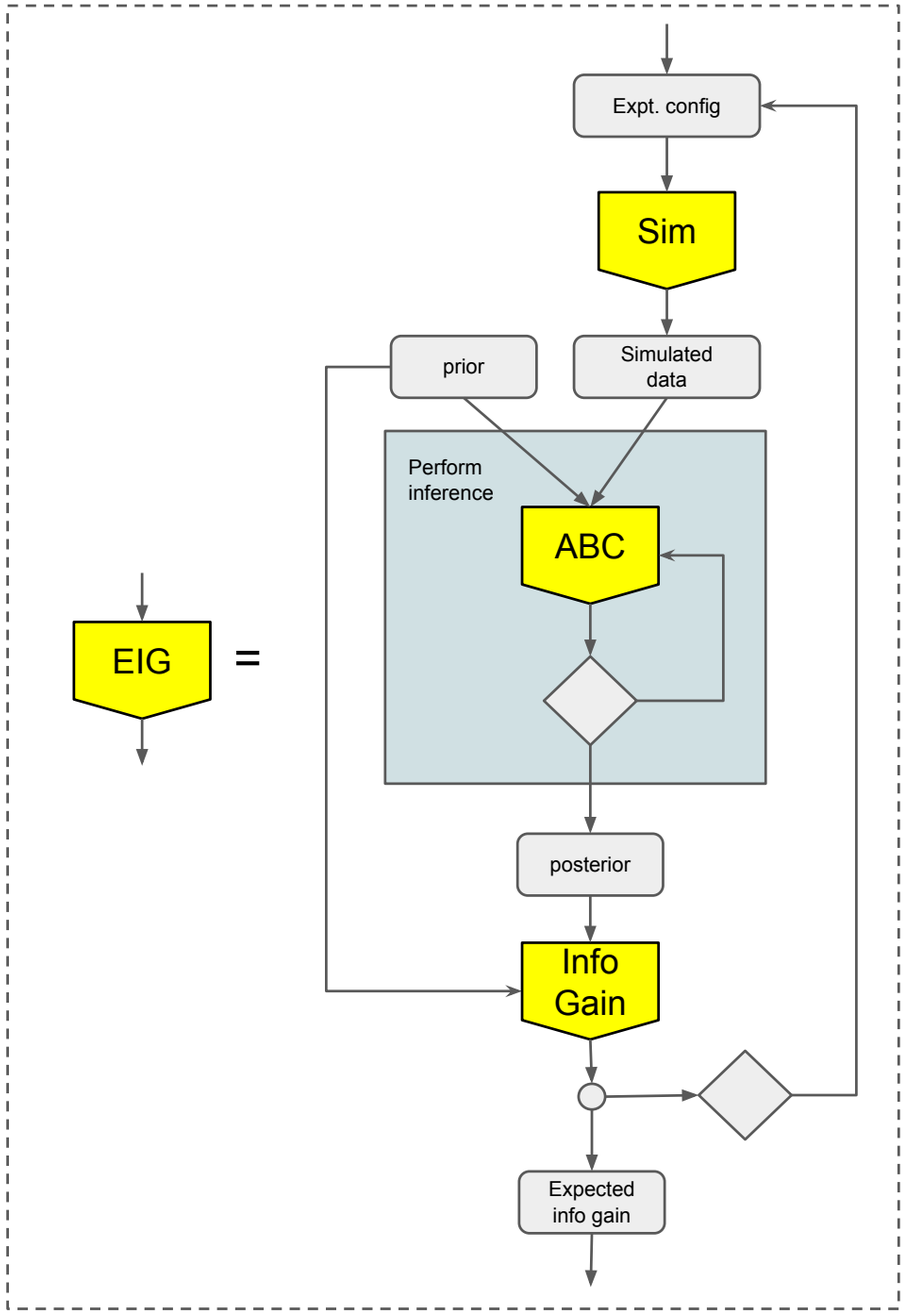

Expt. config
Perform
experiment
Expt
prior
Observed
data
Perform
inference
ABC
posterior
Optimize
experiment
Proposal
EIG


Expt. config
Perform
experiment
Expt
prior
Observed
data
Perform
inference
ABC
posterior
Optimize
experiment
Proposal
EIG


prior
data
Proposal
ABC
=
Sim
posterior


Expt. config
Perform
experiment
Expt
prior
Observed
data
Perform
inference
ABC
posterior
Optimize
experiment
Proposal
EIG


prior
data
Proposal
ABC
=
Sim
posterior

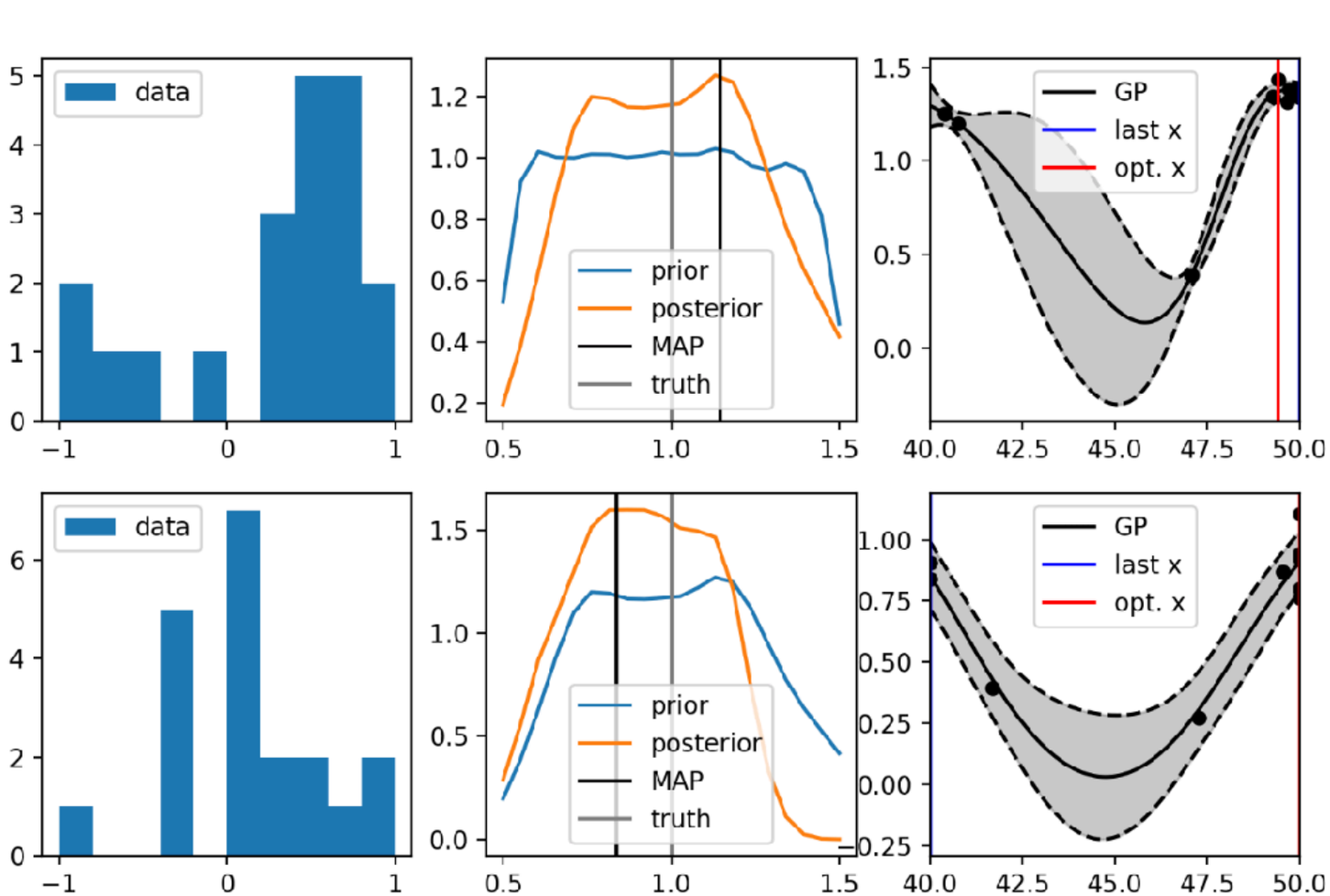

Expt. config
Sim
prior
Simulated
data
Perform
inference
ABC
EIG
=
posterior
Info
Gain
Expected
info gain

ACTIVE SCIENCING DEMO

Input:

- workflow for performing “real” experiment that returns data
- workflow for running simulator given parameters of theory and experimental configuration

Demo shows use of likelihood-free inference technique & Bayesian Optimization to measure the Weinberg angle and optimize beam energy (eg. just above or below $M_Z/2$)



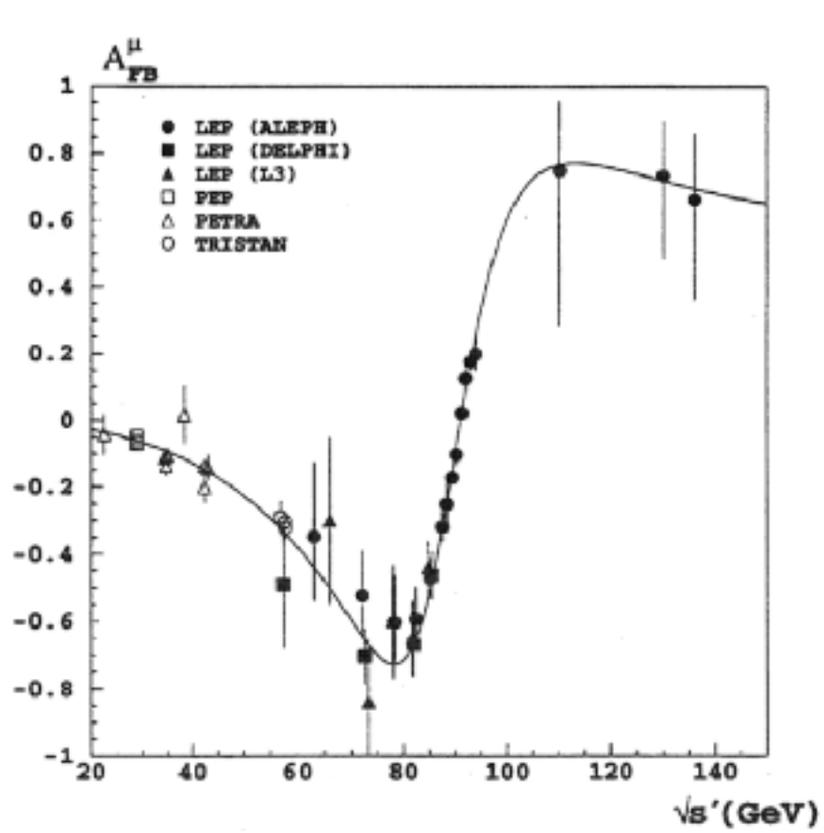


Figure 2: Measured forward-backward asymmetries of muon-pair production compared with the model independent fit results.

ENCAPSULATING THE SIMULATION

https://github.com/lukasheinrich/weinberg-test

README.md

Run HEP workflows from the web.

by Kyle Cranmer and Lukas Heinrich

An example notebook on how to generate simulated high energy physics collision events using the generator package MadGraph. Simulated datasets obtained from this notebook can then be used to train and evaluate the performance of generative models for physics.

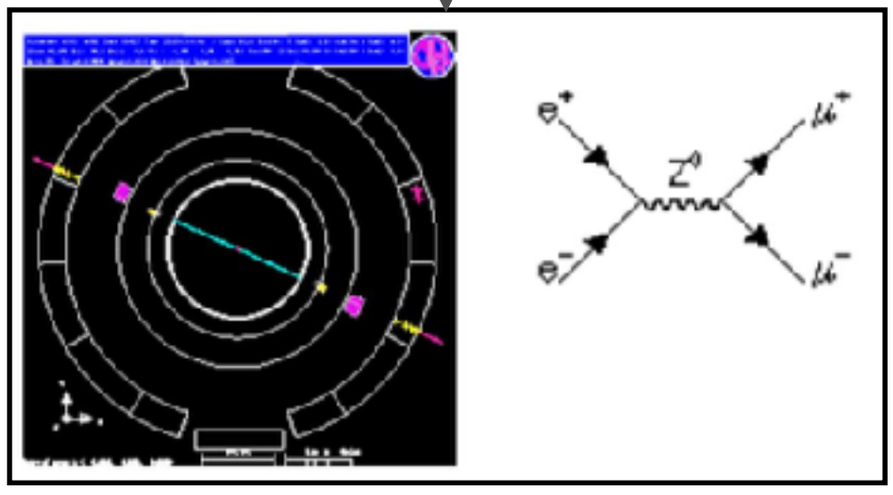
Usage:

This repository has been equipped with a Dockerfile to encapsulate its software environment. It can be used with the mybinder service to launch an ephemeral jupyter notebook server to run the notebook.

Click on the below badge and open the notebook `acage.ipynb`.

launch binder

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



other electroweak parameters. This can be shown with Eq. (2.96), giving

$$A^{f}_{FB}(s) \simeq A^{f}_{FB}(m_Z^2) + \frac{(s-m_Z^2)}{s}\frac{3\pi\alpha(s)}{\sqrt{2}\mathrm{G_F}m_Z^2}\frac{2Q_eQ_fg_{Ae}g_{Af}}{(g_{Ve}^2+g_{Ae}^2)(g_{Vf}^2+g_{Af}^2)} \ . \quad (8.30)$$

ENCAPSULATING THE SIMULATION

https://github.com/lukasheinrich/weinberg-test

README.md

Run HEP workflows from the web.

by Kyle Cranmer and Lukas Heinrich

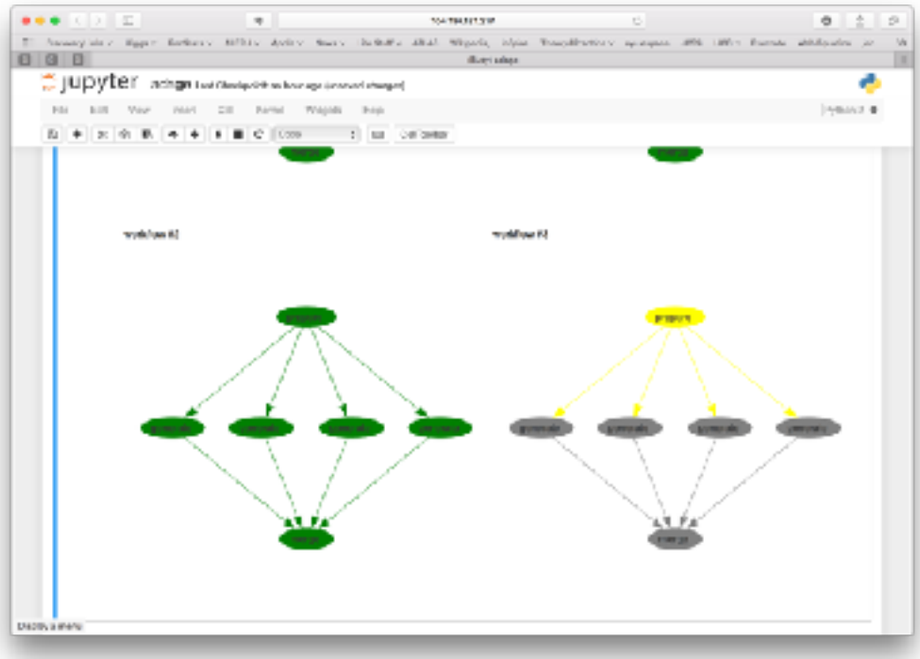
An example notebook on how to generate simulated high energy physics collision events using the generator package MadGraph. Simulated datasets obtained from this notebook can then be used to train and evaluate the performance of generative models for physics.

Usage:

This repository has been equipped with a Dockerfile to encapsulate its software environment. It can be used with the mybinder service to launch an ephemeral jupyter notebook server to run the notebook.

Click on the below badge and open the notebook `acage.ipynb`.

launch binder

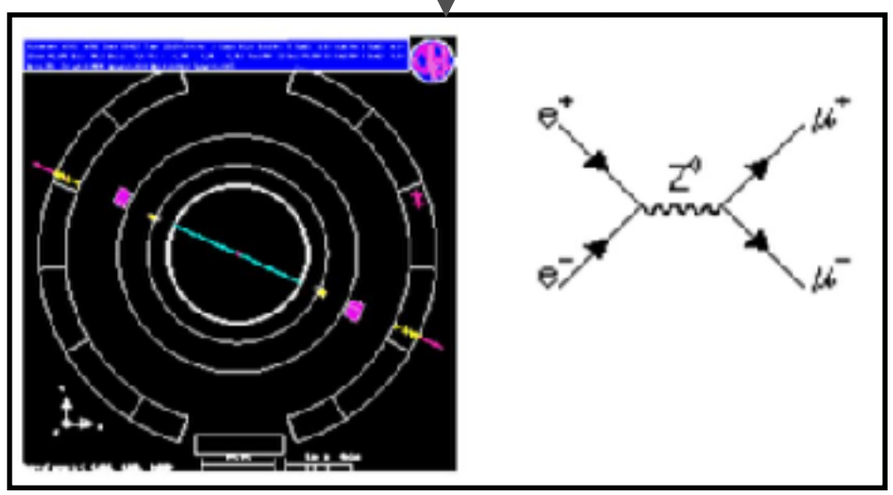


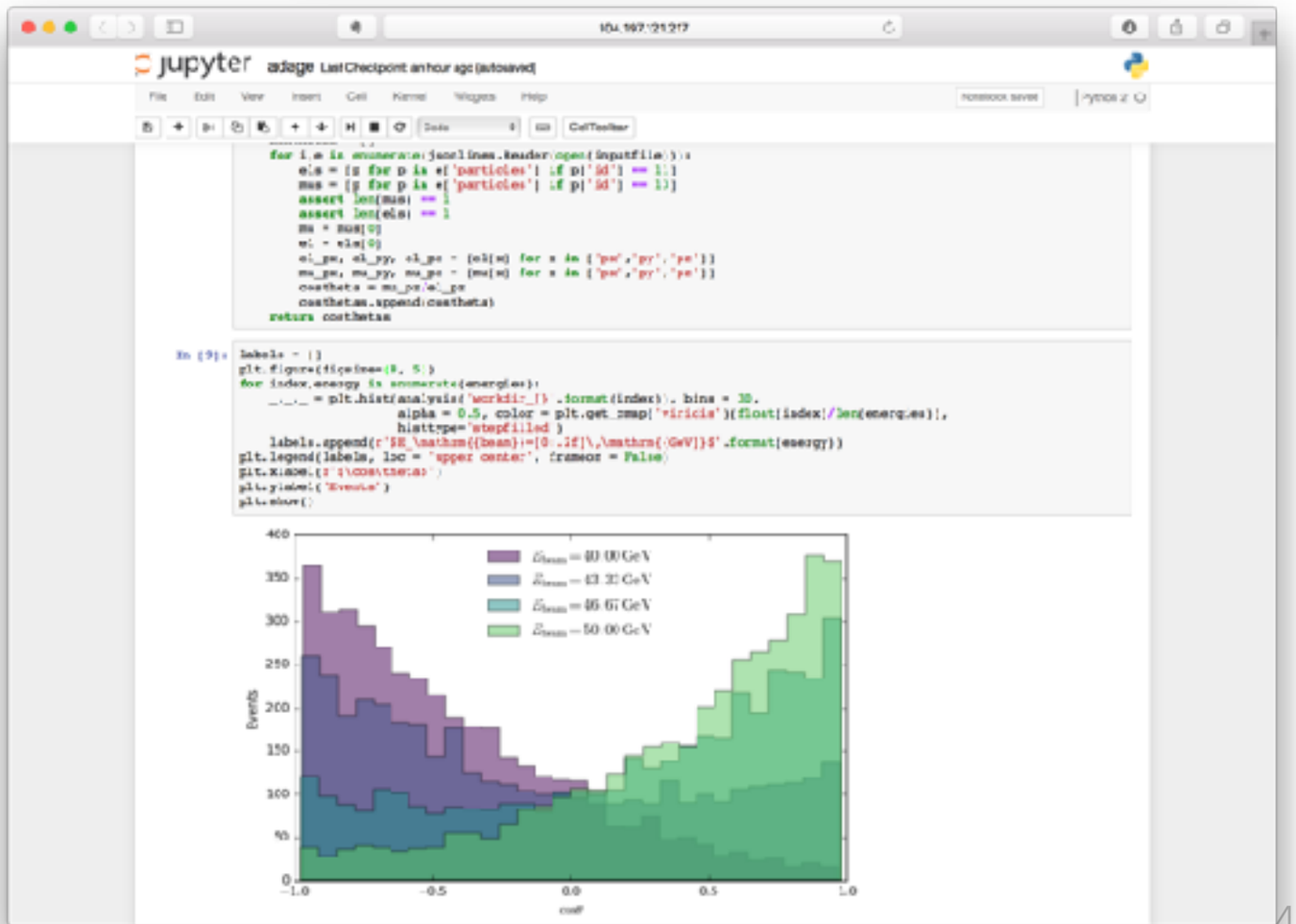
$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$

$$+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}}$$

$$+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma,\text{and Higgs masses and couplings}}$$

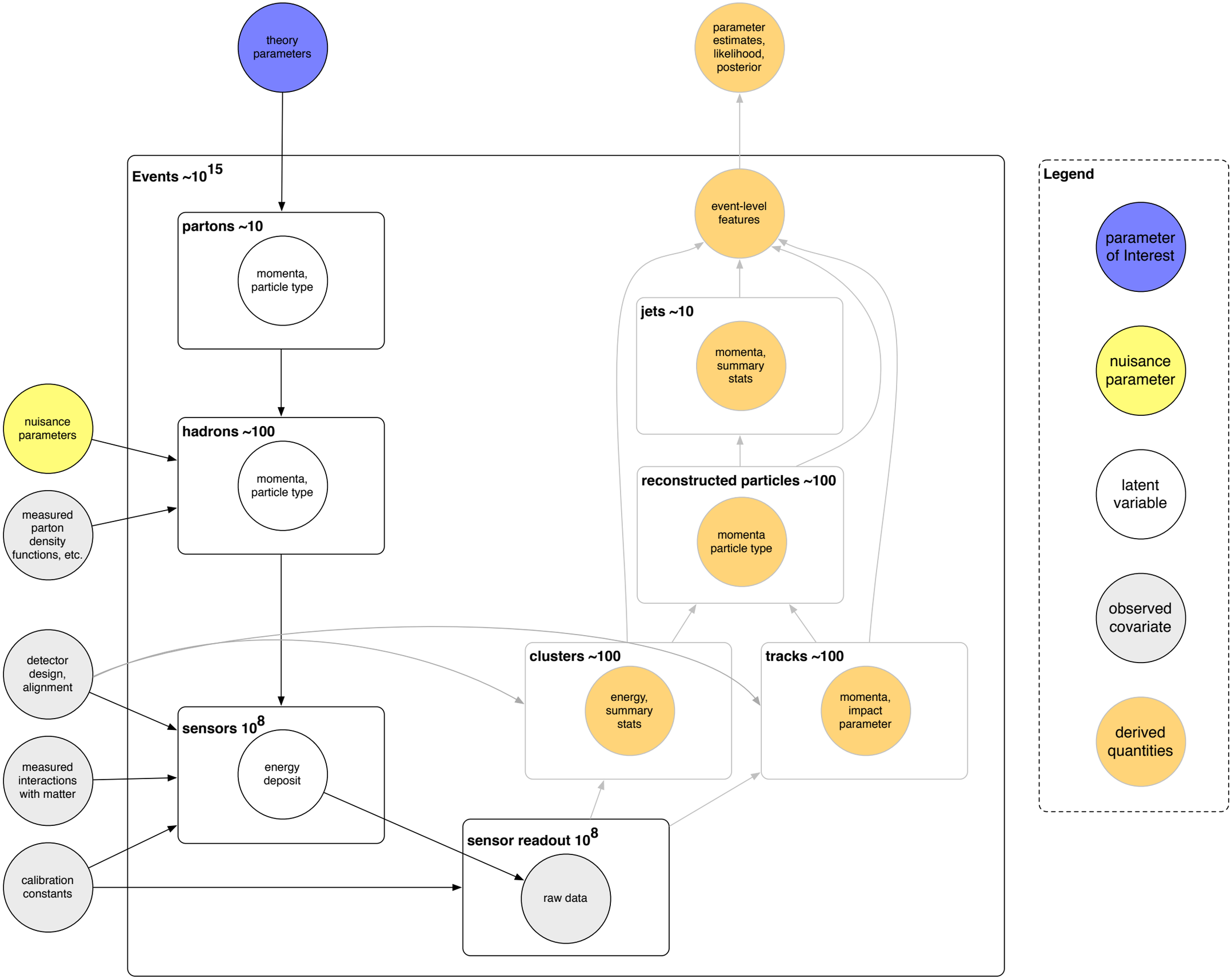
$$+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_1\bar{L}\phi R + G_2\bar{L}\phi_c R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$



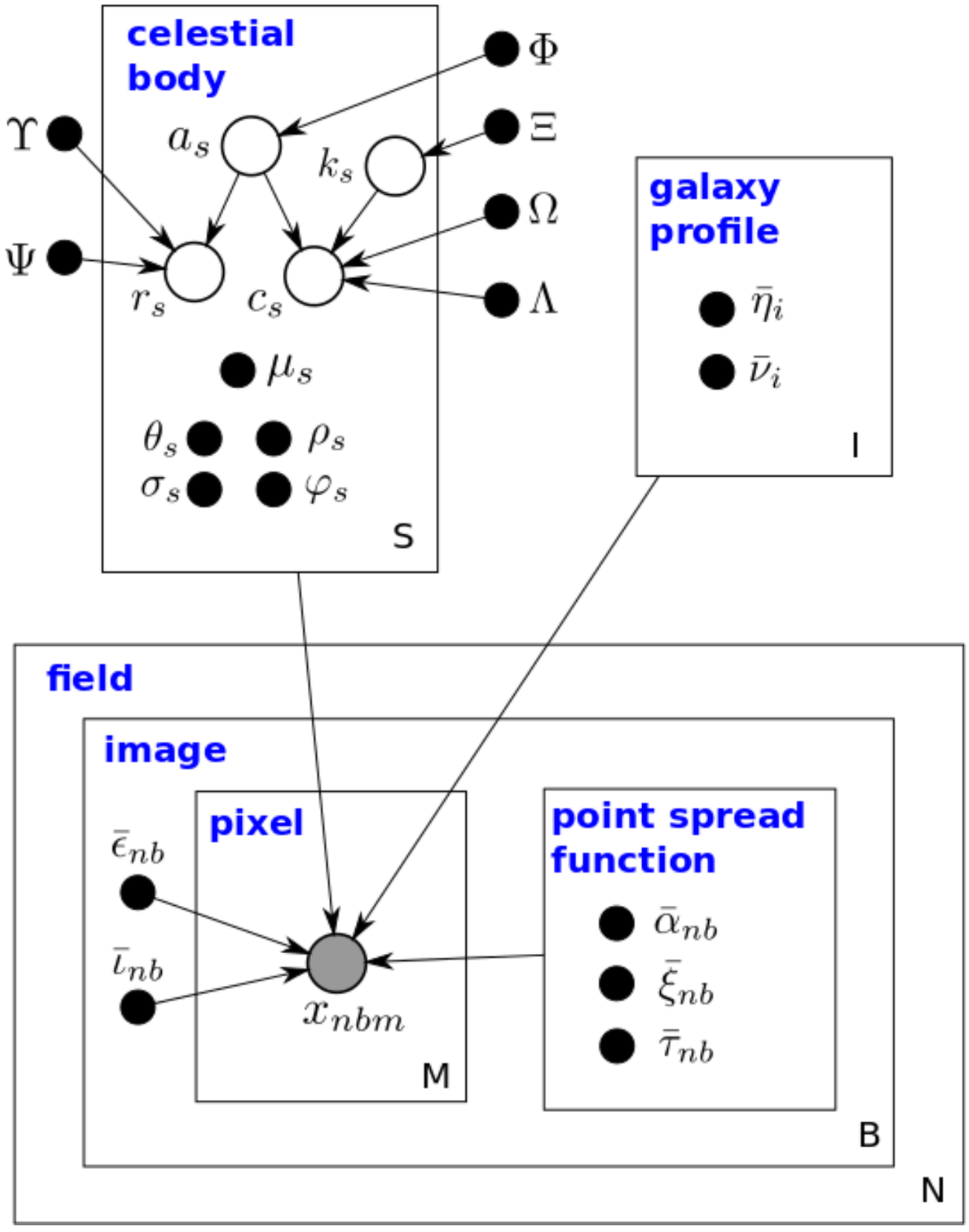


Misc

FULL SIMULATION + RECONSTRUCTION



HIERARCHICAL GRAPHICAL MODELS IN ASTRONOMY



Celeste: Variational inference for a generative model of astronomical images

Regler, et al 33rd ICMLR, 2016

ML2.0?

How do these fit together?

Combine many of these ideas:

Large model, but **sparsely activated**

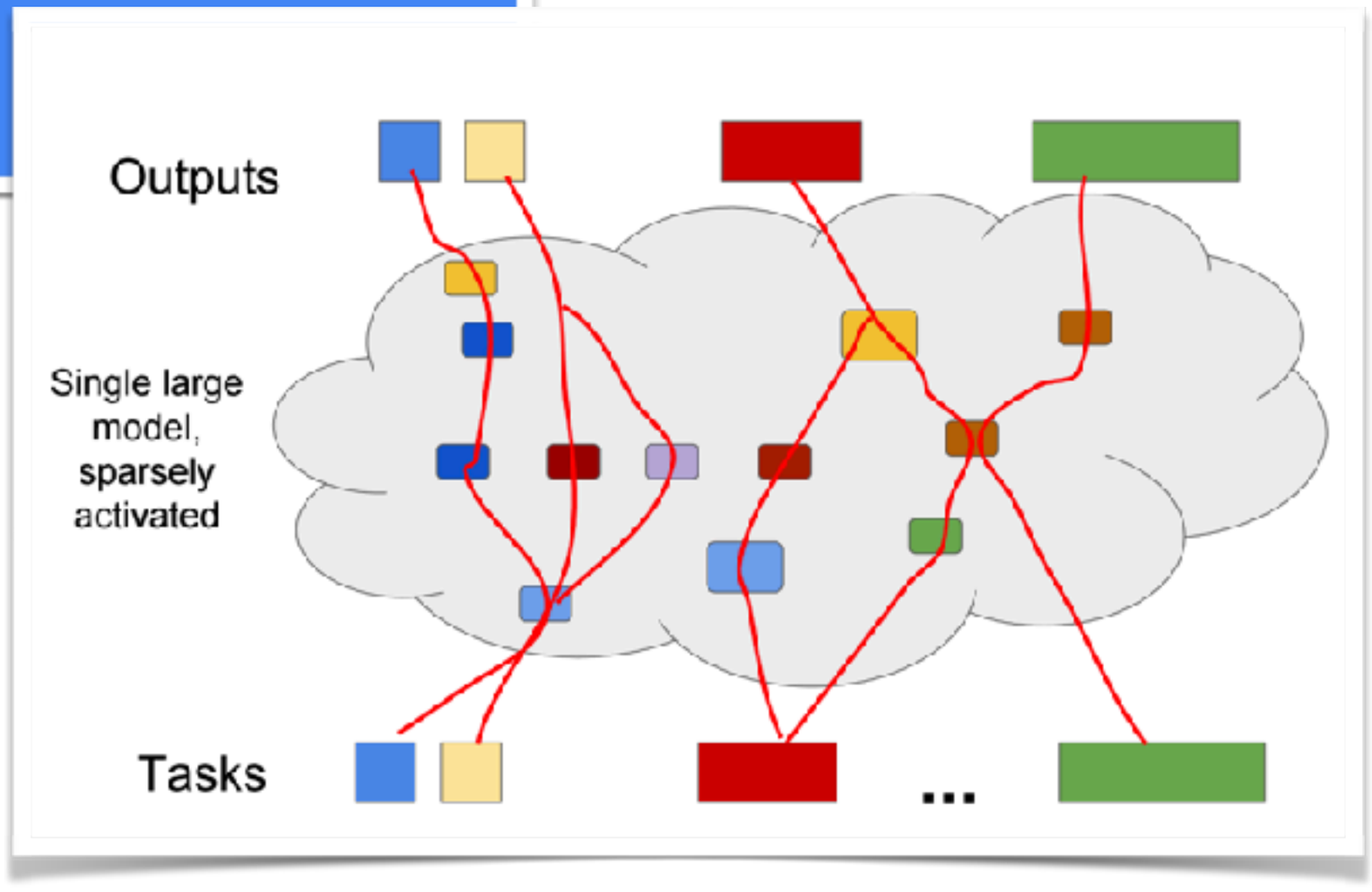
Single model to **solve many tasks** (100s to 1Ms)

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

Hardware **specialized for ML supercomputing**

ML for efficient mapping onto this hardware

Google

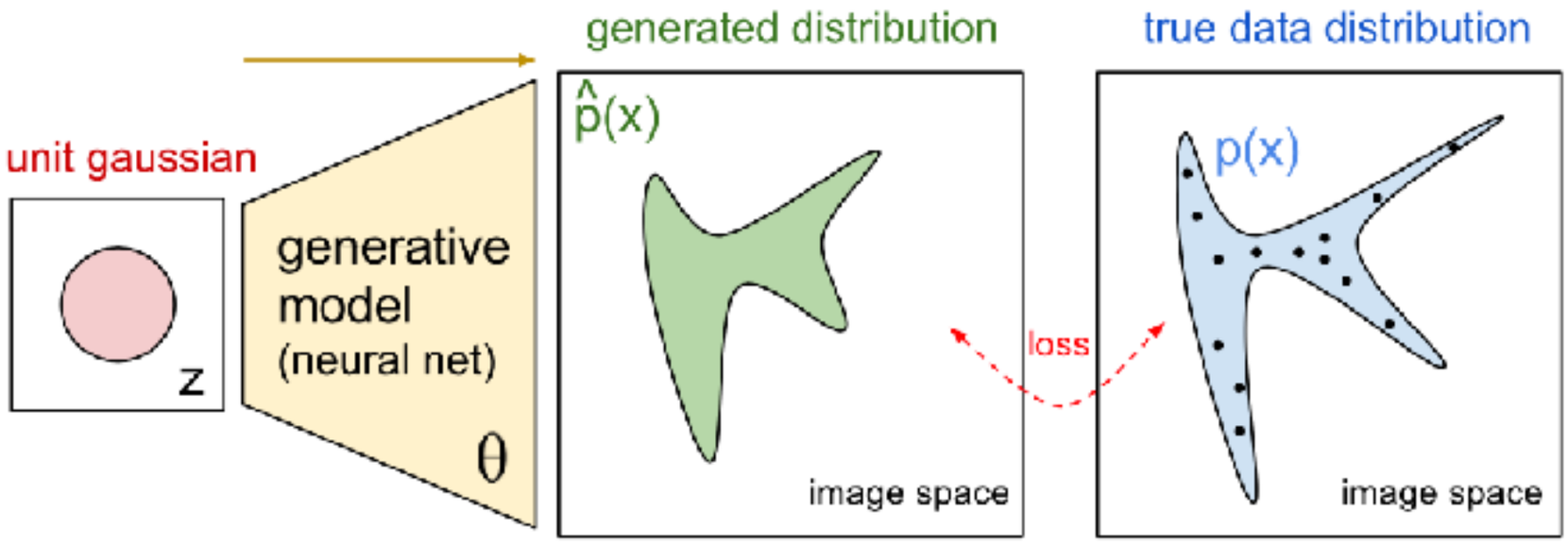


Slides from Jeff Dean of Google Brain @ Jeju July 2017

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

GENERATIVE ADVERSARIAL NETWORKS

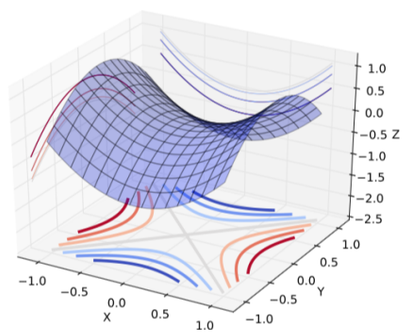
Goodfellow, et al arXiv:1406.2661
slide Gilles Louppe



Leo is G Tom is D

- Two-player game:
  - a discriminator D,
  - a generator G;
- D is a classifier $\mathcal{X} \mapsto \{0, 1\}$ that tries to distinguish between
  - a sample from the data distribution ($D(\mathbf{x}) = 1$, for $\mathbf{x} \sim p_{\text{data}}$),
  - and a sample from the model distribution ($D(G(\mathbf{z})) = 0$, for $\mathbf{z} \sim p_{\text{noise}}$);
- G is a generator $\mathcal{Z} \mapsto \mathcal{X}$ trained to produce samples $G(\mathbf{z})$ (for $\mathbf{z} \sim p_{\text{noise}}$) that are difficult for D to distinguish from data.

$$(D^*, G^*) = \max_D \min_G V(D, G).$$



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 minimax 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 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**!

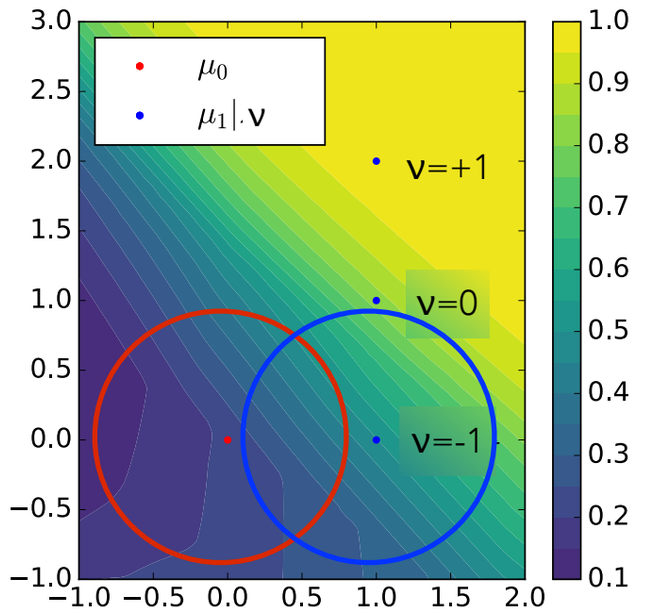
Allows us to efficiently fit / **tune simulation** with stochastic gradient techniques!

LEARNING TO PIVOT WITH ADVERSARIAL NETWORKS

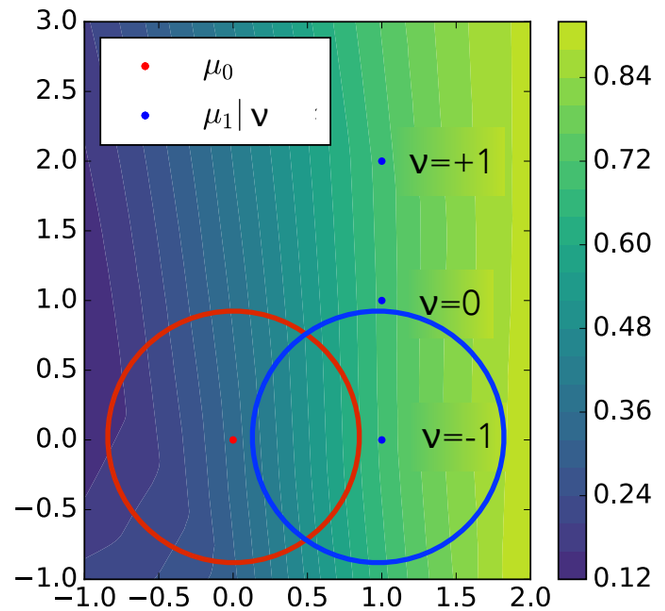
Typically classifier **f(x)** trained to minimize loss $\mathbf{L_f}$.

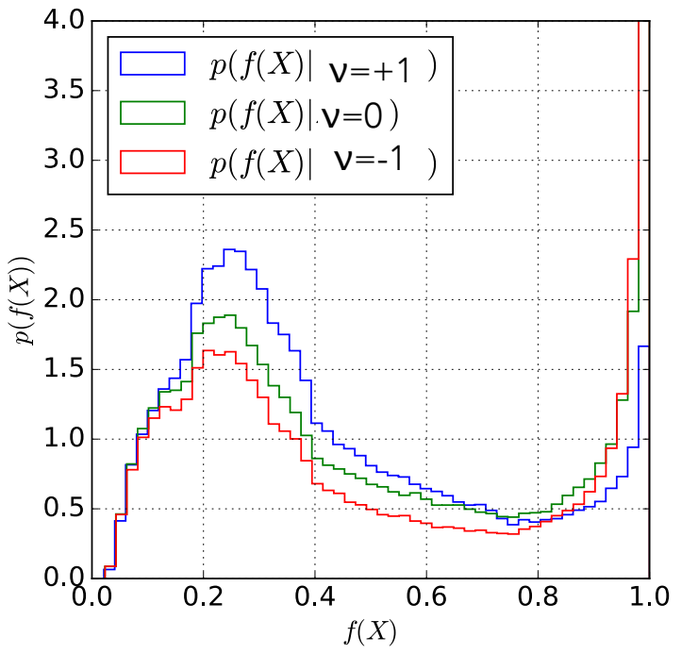
- want classifier output to be insensitive to systematics (nuisance parameter **ν**)
- introduce an **adversary r** that tries to predict ν based on f.
- provides training procedure that allows for **tradeoff** between traditional classification accuracy and **robustness to systematics**

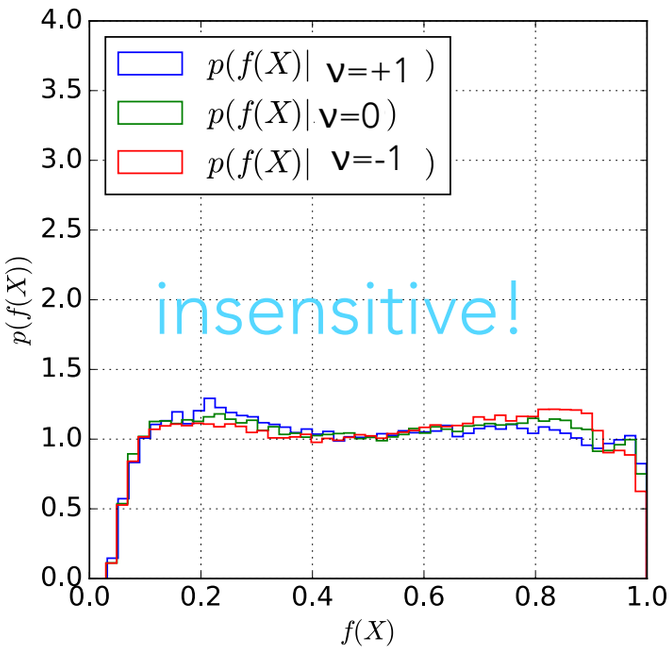
normal training



adversarial training







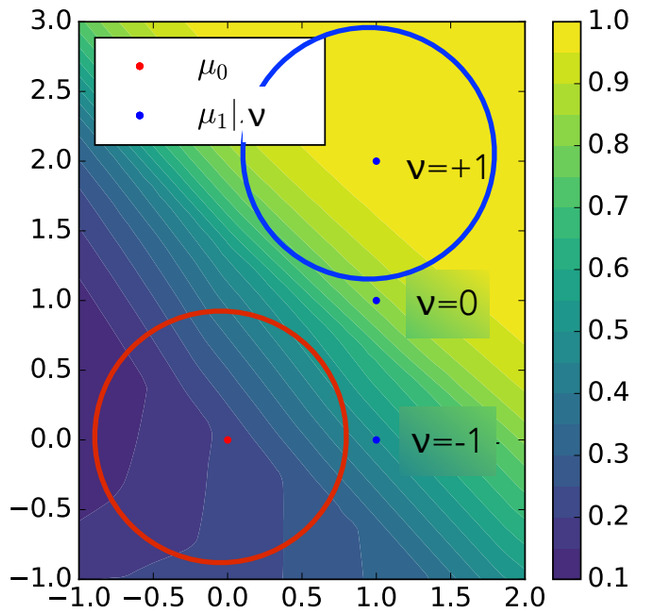
G. Louppe, M. Kagan, K. Cranmer, arXiv:1611.01046

LEARNING TO PIVOT WITH ADVERSARIAL NETWORKS

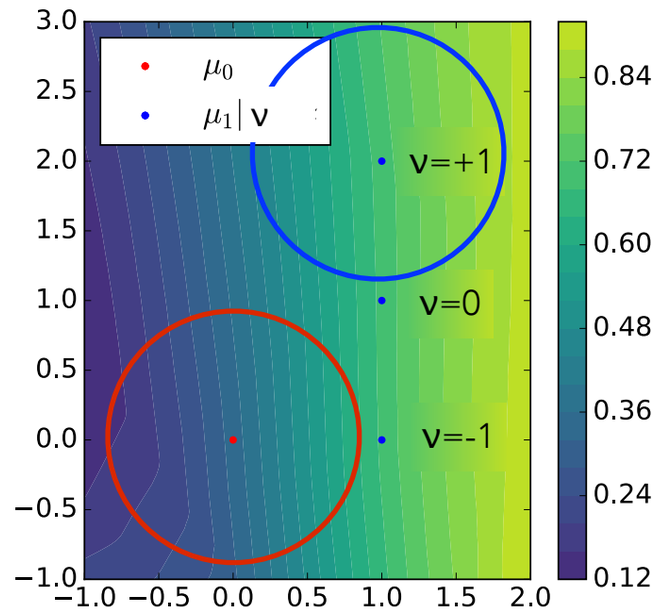
Typically classifier **f(x)** trained to minimize loss $\mathbf{L_f}$.

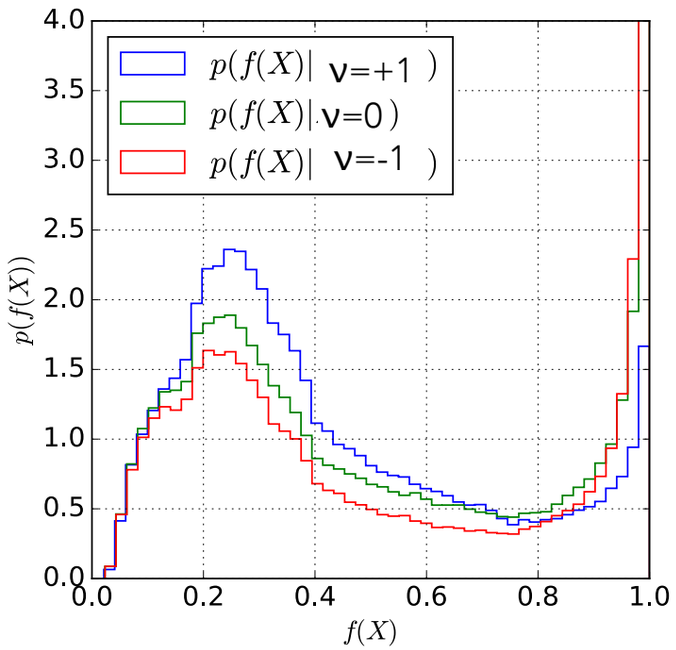
- want classifier output to be insensitive to systematics (nuisance parameter **ν**)
- introduce an **adversary r** that tries to predict ν based on f.
- provides training procedure that allows for **tradeoff** between traditional classification accuracy and **robustness to systematics**

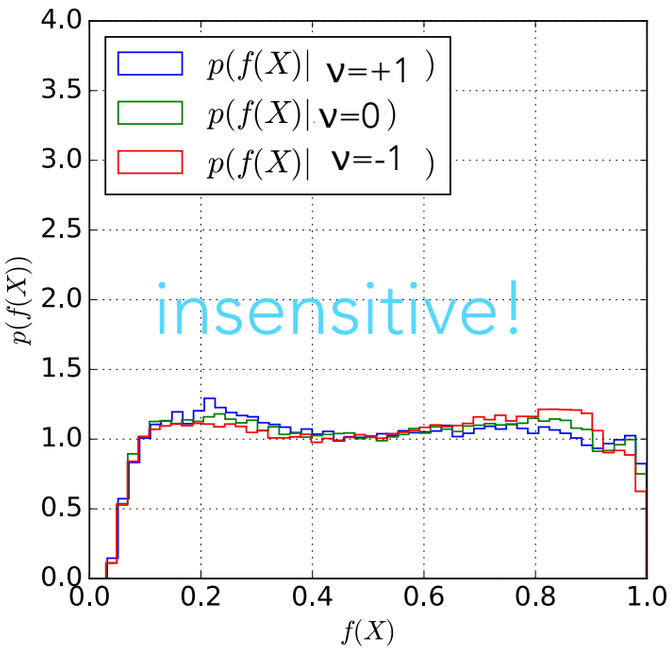
normal training



adversarial training







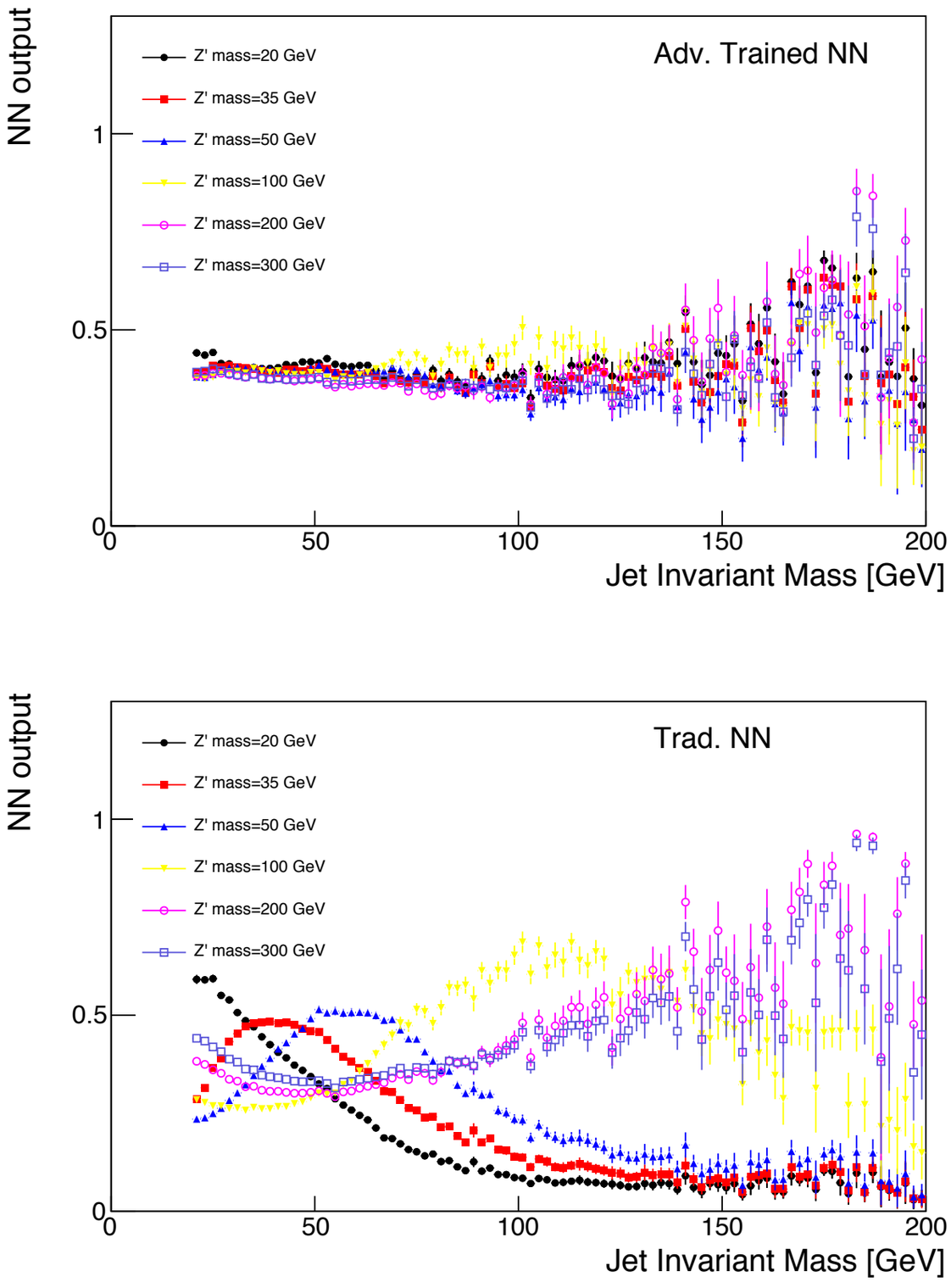
G. Louppe, M. Kagan, K. Cranmer, arXiv:1611.01046

DECORRELATED TAGGERS

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 "decorrelated" to some other variable.

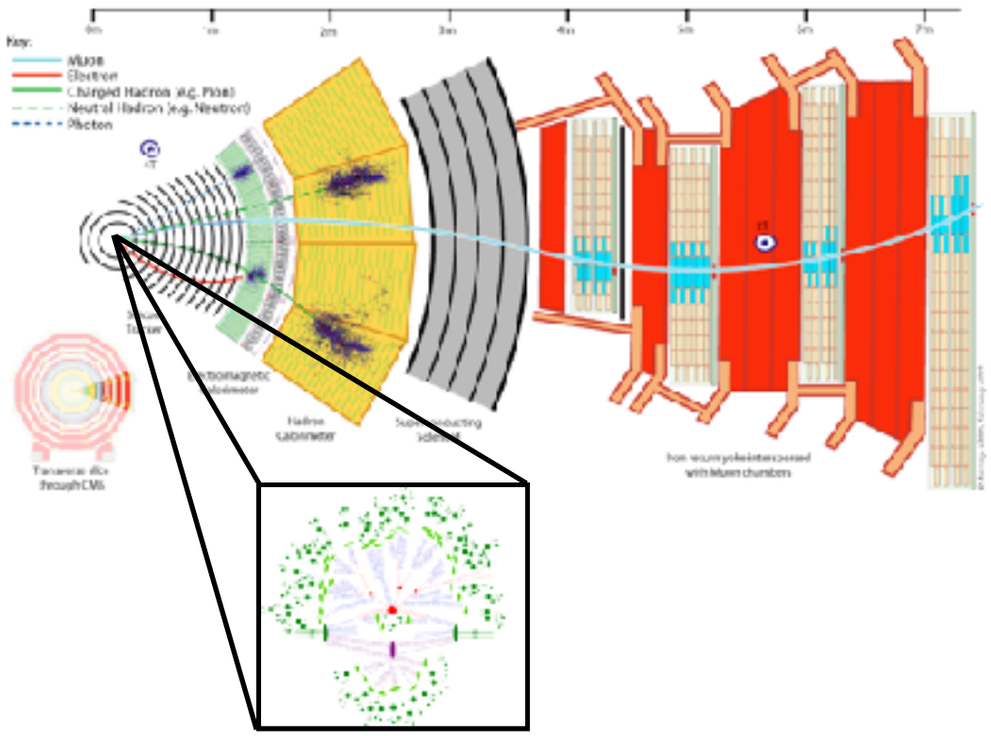
- want jet taggers that are decorrelated with jet invariant mass
- so that analysis can still search for a bump using jet invariant mass
- avoids sculpting background



TWO APPROACHES

Use simulator

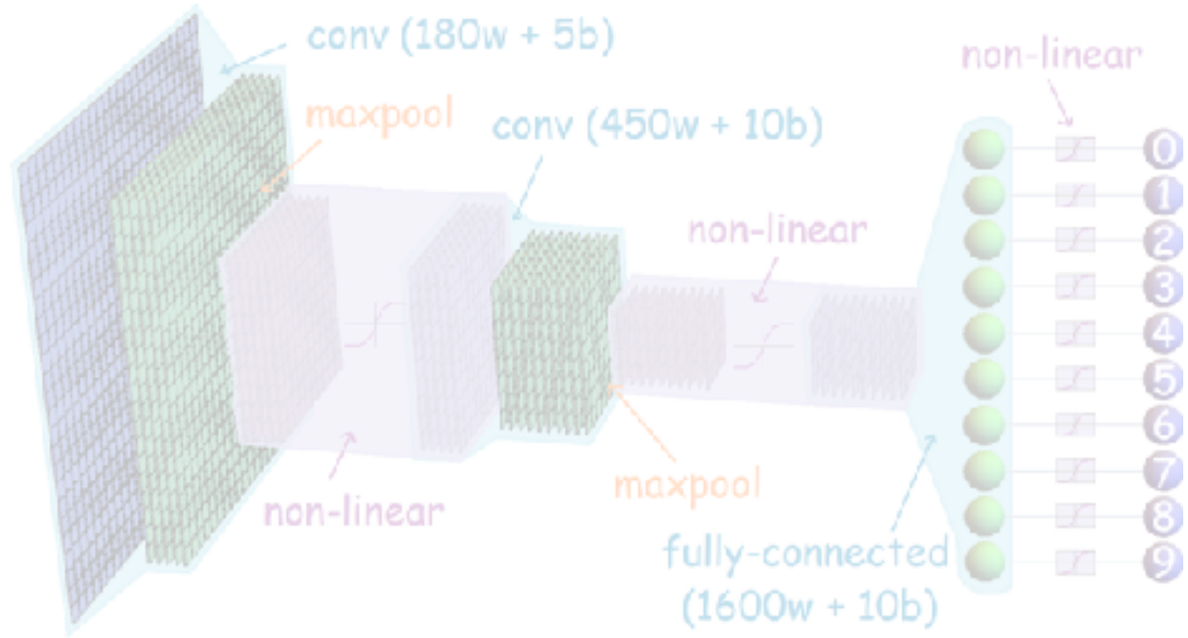
(much more efficiently)



- 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)
- Autogregressive models, Normalizing Flows

[image credit: A.P. Goucher]

'Likelihood-Free' Inference ← exact Bayesian Computation

Rejection Algorithm

- Draw θ from prior $\pi(\cdot)$
- Accept θ with probability $\pi(D \mid \theta)$

Accepted θ are independent draws from the posterior distribution, $\pi(\theta \mid D)$.

If the likelihood, $\pi(D|\theta)$, is unknown:

'Mechanical' Rejection Algorithm

- Draw θ from $\pi(\cdot)$
- Simulate $X \sim f(\theta)$ from the computer model
- Accept θ if $D = X$, i.e., if computer output equals observation

The acceptance rate is $\int \mathbb{P}(D|\theta)\pi(\theta)\mathrm{d}\theta = \mathbb{P}(D)$.

*From Richard Wilkinson's talk at Data science @LHC

Rejection ABC

If $\mathbb{P}(D)$ is small (or D continuous), we will rarely accept any θ. Instead, there is an approximate version:

Uniform Rejection Algorithm

- Draw θ from $\pi(\theta)$
- Simulate $X \sim f(\theta)$
- Accept θ if $\rho(D, X) \leq \epsilon$

ϵ reflects the tension between computability and accuracy.

- As $\epsilon \to \infty$, we get observations from the prior, $\pi(\theta)$.
- If $\epsilon = 0$, we generate observations from $\pi(\theta \mid D)$.

For reasons that will become clear later, we call this *uniform-ABC*.

*From Richard Wilkinson's talk at Data science @LHC

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 minimax 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 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!

‘Likelihood-Free’ Inference ← exact Bayesian Computation

Rejection Algorithm

- Draw θ from prior $\pi(\cdot)$
- Accept θ with probability $\pi(D \mid \theta)$

Accepted θ are independent draws from the posterior distribution, $\pi(\theta \mid D)$.

If the likelihood, $\pi(D|\theta)$, is unknown:

‘Mechanical’ Rejection Algorithm

- Draw θ from $\pi(\cdot)$
- Simulate $X \sim f(\theta)$ from the computer model
- Accept θ if $D = X$, i.e., if computer output equals observation

The acceptance rate is $\int \mathbb{P}(D|\theta)\pi(\theta)\mathrm{d}\theta = \mathbb{P}(D)$.

*From Richard Wilkinson’s talk at Data science @LHC

Rejection ABC

If $\mathbb{P}(D)$ is small (or D continuous), we will rarely accept any θ. Instead, there is an approximate version:

Uniform Rejection Algorithm

- Draw θ from $\pi(\theta)$
- Simulate $X \sim f(\theta)$
- Accept θ if $\rho(D, X) \leq \epsilon$

ϵ reflects the tension between computability and accuracy.

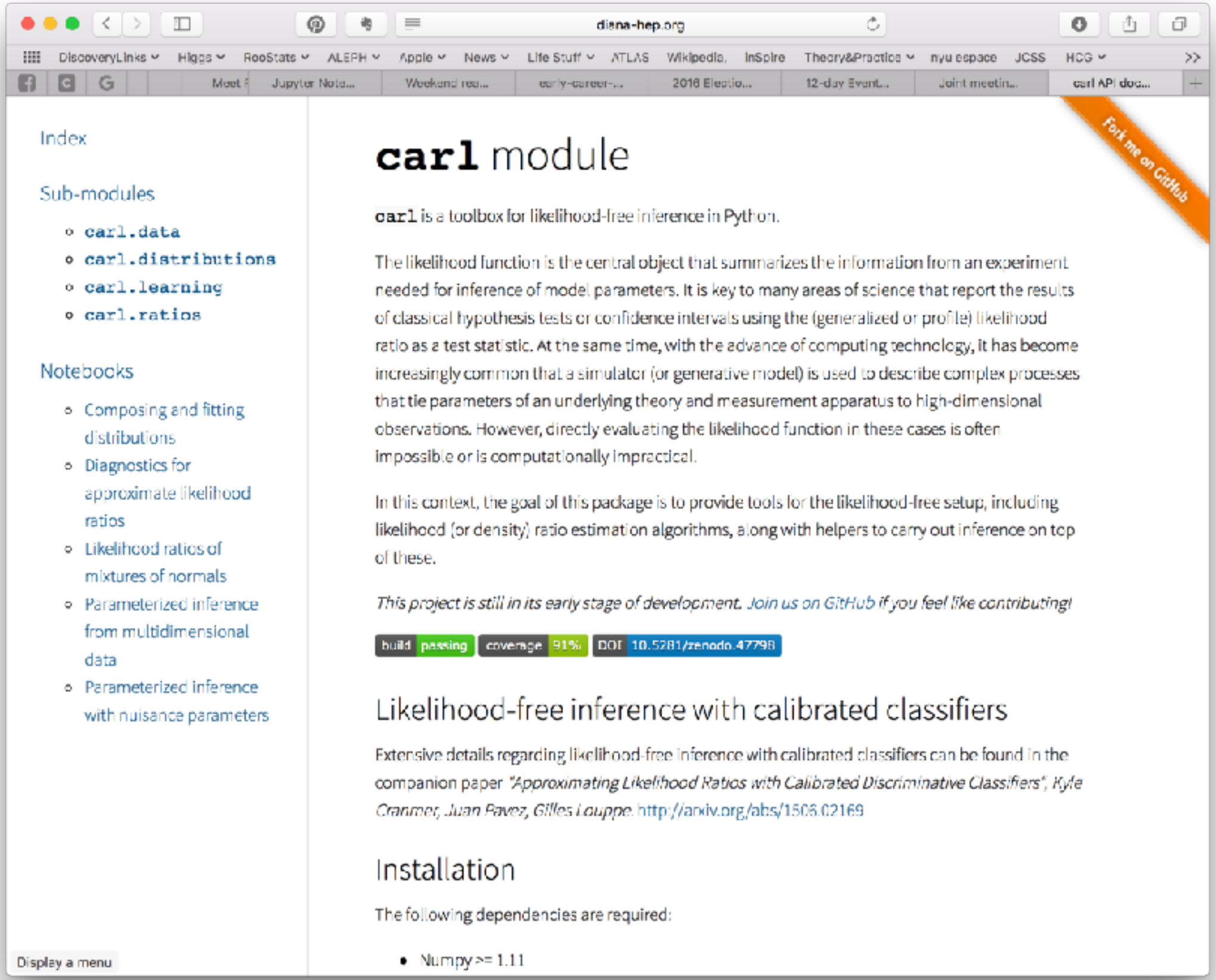
- As $\epsilon \to \infty$, we get observations from the prior, $\pi(\theta)$.
- If $\epsilon = 0$, we generate observations from $\pi(\theta \mid D)$.

For reasons that will become clear later, we call this *uniform-ABC*.

*From Richard Wilkinson's talk at Data science @LHC

CARL SOFTWARE

http://diana-hep.org/carl/



BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

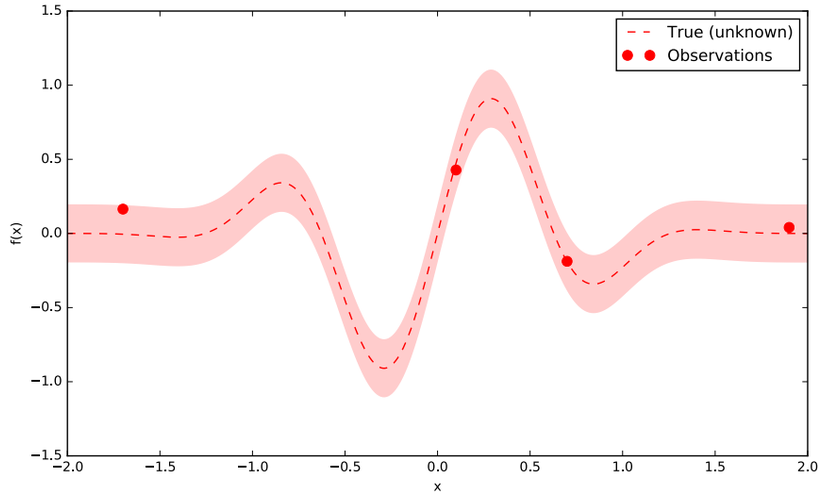
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Where shall we sample next?



BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

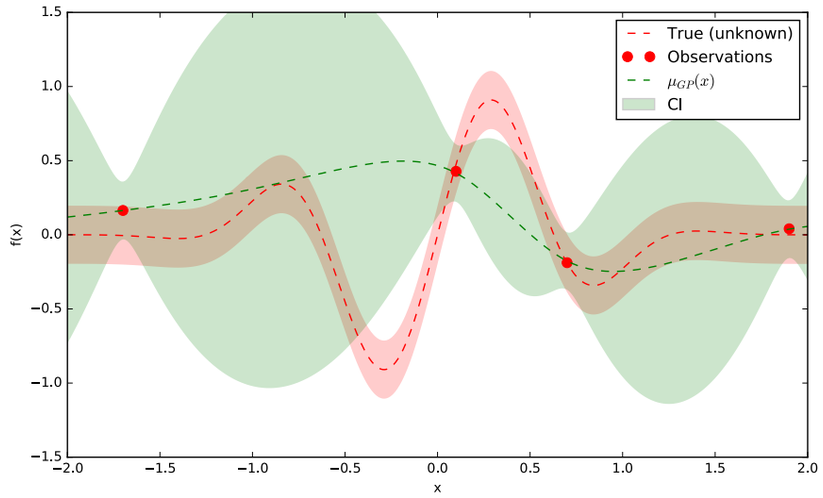
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

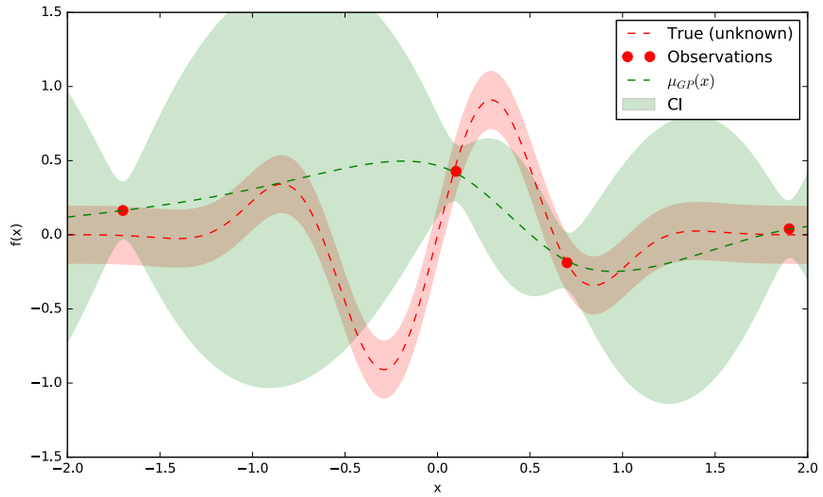
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

Acquisition functions

Acquisition functions u(x) specify which sample x should be tried next:

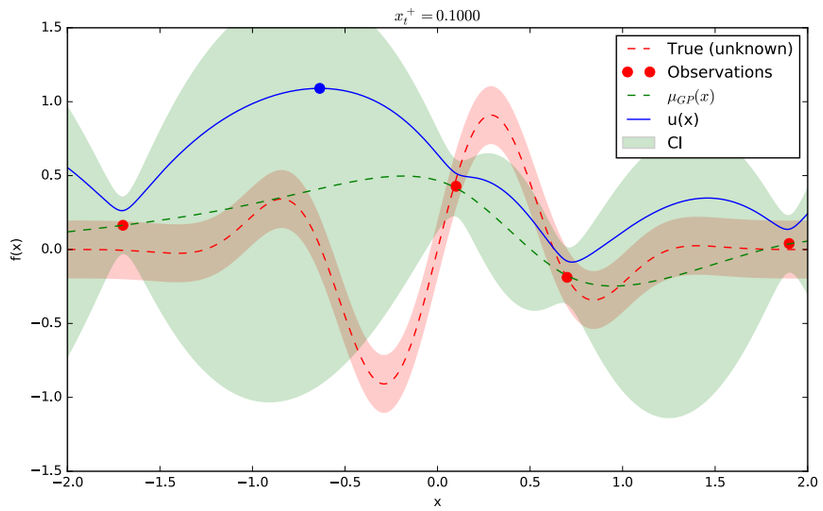
- Upper confidence bound $\text{UCB}(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$;
- Probability of improvement $\text{PI}(x) = P(f(x) \geq f(x_t^+) + \kappa)$;
- Expected improvement $\text{EI}(x) = \mathbb{E}[f(x) - f(x_t^+)]$;
- ... and many others.

where x_t^+ is the best point observed so far.

In most cases, acquisition functions provide knobs (e.g., κ) for controlling the exploration-exploitation trade-off.

- Search in regions where $\mu_{GP}(x)$ is high (exploitation)
- Probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration)

Plugging everything together ($t = 0$)



$$x_{t+1} = \arg\max_x \text{UCB}(x)$$

BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

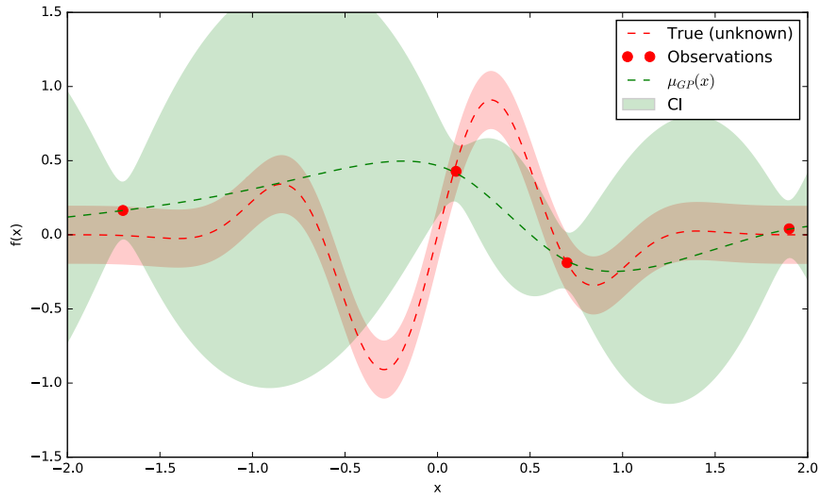
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

Acquisition functions

Acquisition functions u(x) specify which sample x should be tried next:

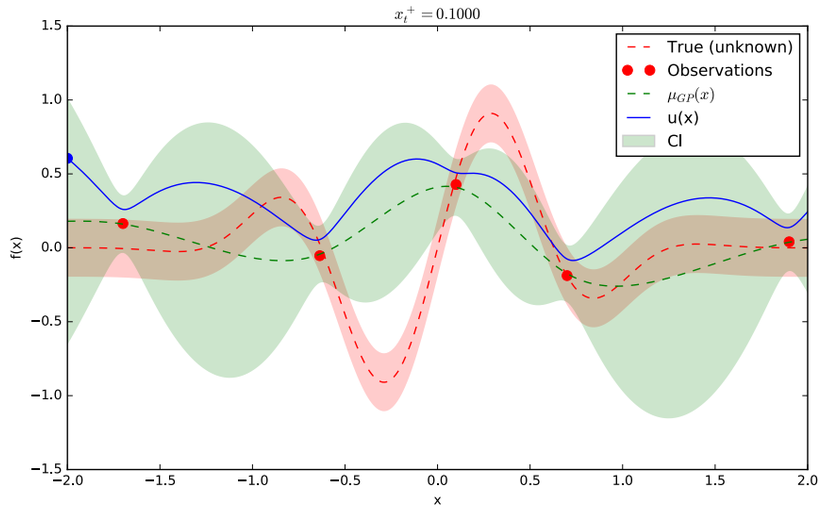
- Upper confidence bound $\text{UCB}(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$;
- Probability of improvement $\text{PI}(x) = P(f(x) \geq f(x_t^+) + \kappa)$;
- Expected improvement $\text{EI}(x) = \mathbb{E}[f(x) - f(x_t^+)]$;
- ... and many others.

where x_t^+ is the best point observed so far.

In most cases, acquisition functions provide knobs (e.g., κ) for controlling the exploration-exploitation trade-off.

- Search in regions where $\mu_{GP}(x)$ is high (exploitation)
- Probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration)

... and repeat until convergence ($t = 1$)



BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

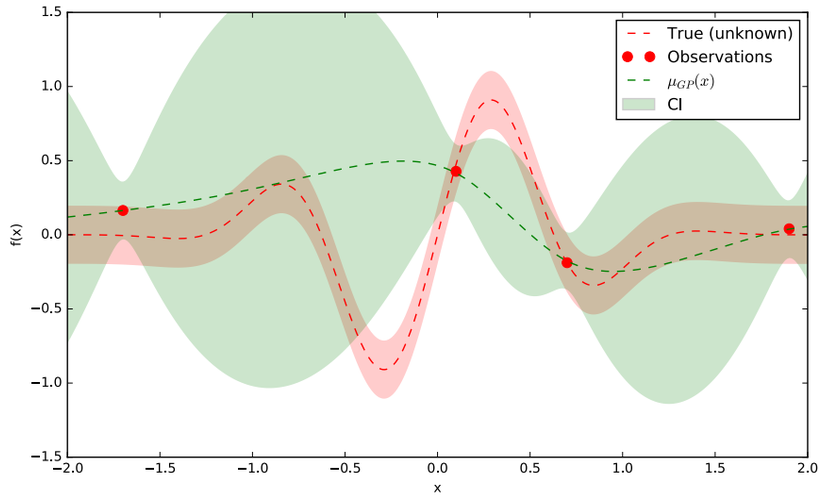
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

Acquisition functions

Acquisition functions u(x) specify which sample x should be tried next:

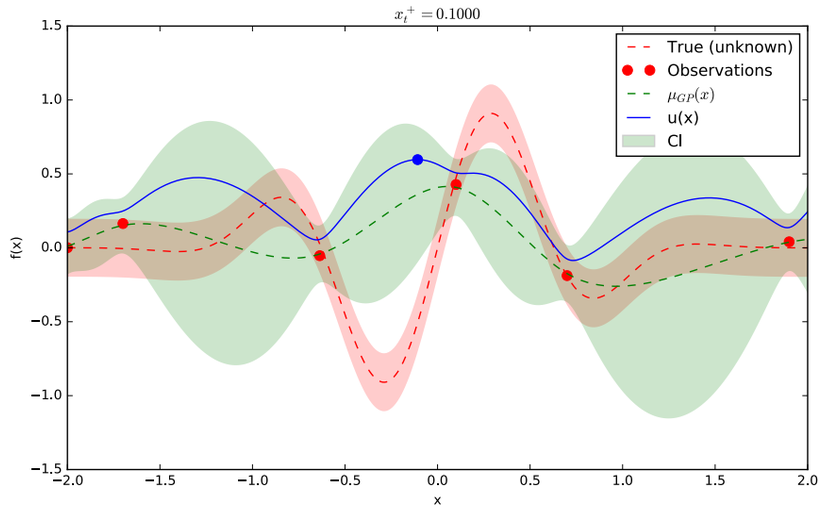
- Upper confidence bound $\text{UCB}(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$;
- Probability of improvement $\text{PI}(x) = P(f(x) \geq f(x_t^+) + \kappa)$;
- Expected improvement $\text{EI}(x) = \mathbb{E}[f(x) - f(x_t^+)]$;
- ... and many others.

where x_t^+ is the best point observed so far.

In most cases, acquisition functions provide knobs (e.g., κ) for controlling the exploration-exploitation trade-off.

- Search in regions where $\mu_{GP}(x)$ is high (exploitation)
- Probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration)

... and repeat until convergence ($t = 2$)



BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

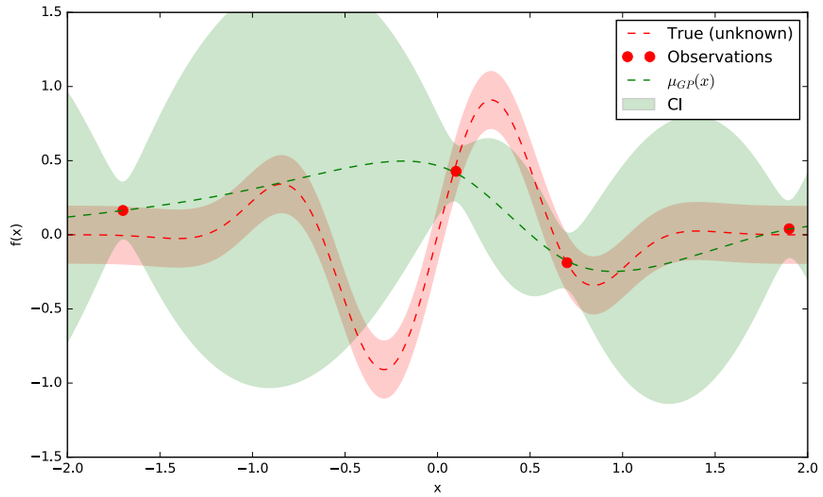
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

Acquisition functions

Acquisition functions u(x) specify which sample x should be tried next:

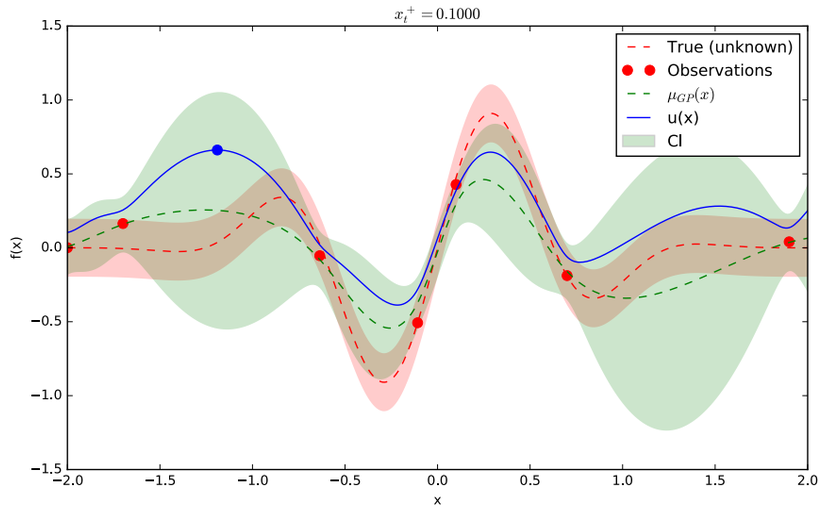
- Upper confidence bound $\text{UCB}(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$;
- Probability of improvement $\text{PI}(x) = P(f(x) \geq f(x_t^+) + \kappa)$;
- Expected improvement $\text{EI}(x) = \mathbb{E}[f(x) - f(x_t^+)]$;
- ... and many others.

where x_t^+ is the best point observed so far.

In most cases, acquisition functions provide knobs (e.g., κ) for controlling the exploration-exploitation trade-off.

- Search in regions where $\mu_{GP}(x)$ is high (exploitation)
- Probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration)

... and repeat until convergence ($t = 3$)



BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

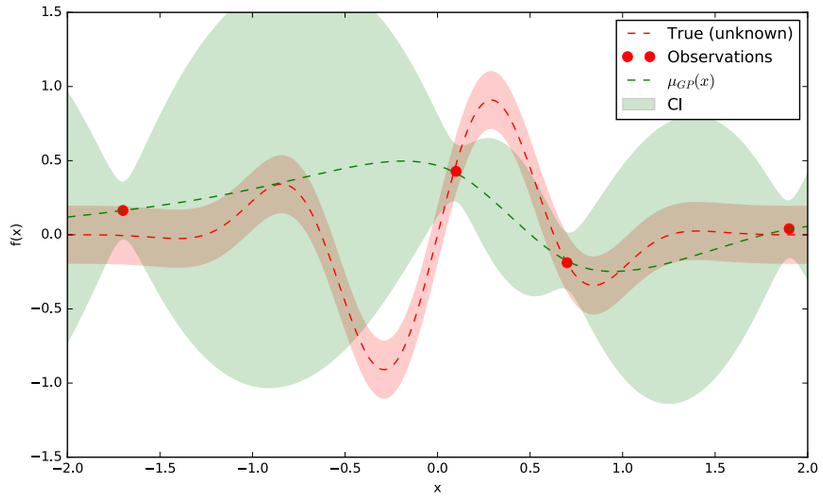
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

Acquisition functions

Acquisition functions u(x) specify which sample x should be tried next:

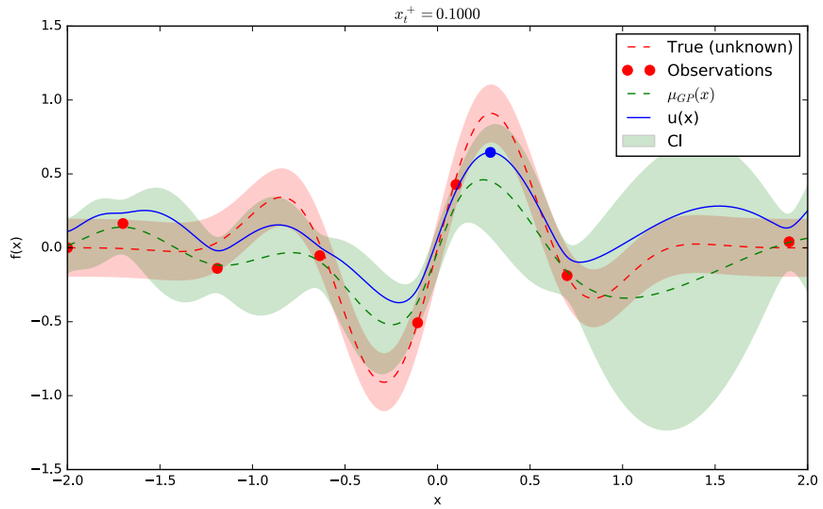
- Upper confidence bound $\text{UCB}(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$;
- Probability of improvement $\text{PI}(x) = P(f(x) \geq f(x_t^+) + \kappa)$;
- Expected improvement $\text{EI}(x) = \mathbb{E}[f(x) - f(x_t^+)]$;
- ... and many others.

where x_t^+ is the best point observed so far.

In most cases, acquisition functions provide knobs (e.g., κ) for controlling the exploration-exploitation trade-off.

- Search in regions where $\mu_{GP}(x)$ is high (exploitation)
- Probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration)

... and repeat until convergence ($t = 4$)



BAYES OPT IN A NUTSHELL

[slides from Gilles Louppe]

Bayesian optimisation

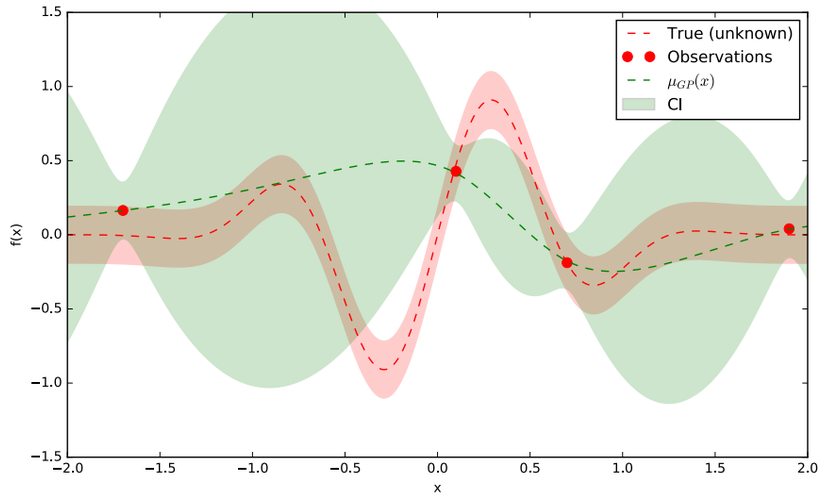
for $t = 1 : T$,

1. Given observations (x_i, y_i) for $i = 1 : t$, build a probabilistic model for the objective f.
   - Integrate out all possible true functions, using Gaussian process regression.
2. Optimise a cheap utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = \arg\max_x u(x)$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Build a probabilistic model for the objective function



This gives a posterior distribution over functions that could have generated the observed data.

Acquisition functions

Acquisition functions u(x) specify which sample x should be tried next:

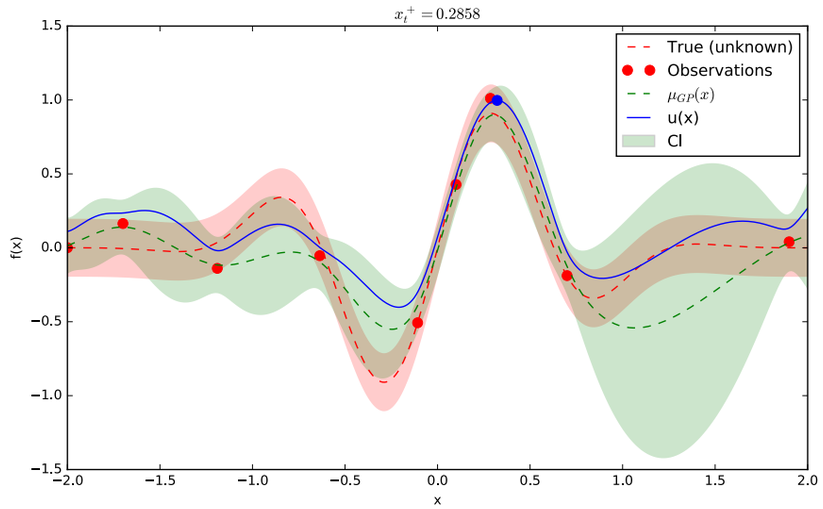
- Upper confidence bound $\text{UCB}(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$;
- Probability of improvement $\text{PI}(x) = P(f(x) \geq f(x_t^+) + \kappa)$;
- Expected improvement $\text{EI}(x) = \mathbb{E}[f(x) - f(x_t^+)]$;
- ... and many others.

where x_t^+ is the best point observed so far.

In most cases, acquisition functions provide knobs (e.g., κ) for controlling the exploration-exploitation trade-off.

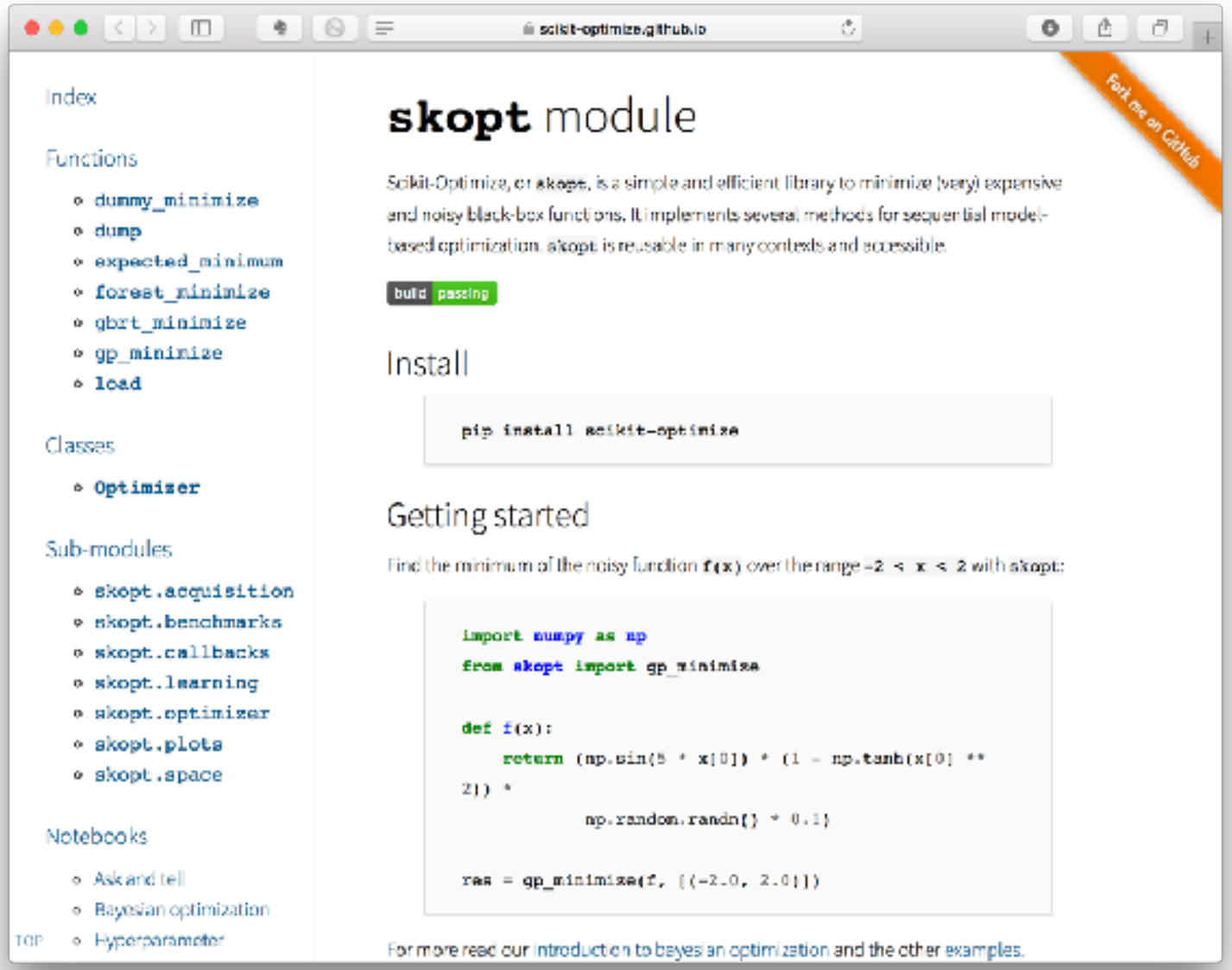
- Search in regions where $\mu_{GP}(x)$ is high (exploitation)
- Probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration)

... and repeat until convergence ($t = 5$)



OPTIMIZATION SOFTWARE

- Python
  - Spearmint https://github.com/JasperSnoek/spearmint
  - GPyOpt https://github.com/SheffieldML/GPyOpt
  - RoBO https://github.com/automl/RoBO
  - scikit-optimize https://github.com/MechCoder/scikit-optimize (work in progress)
- C++
  - MOE https://github.com/yelp/MOE



GitHub Repo for previous slides:

https://github.com/glouppe/talk-bayesian-optimisation

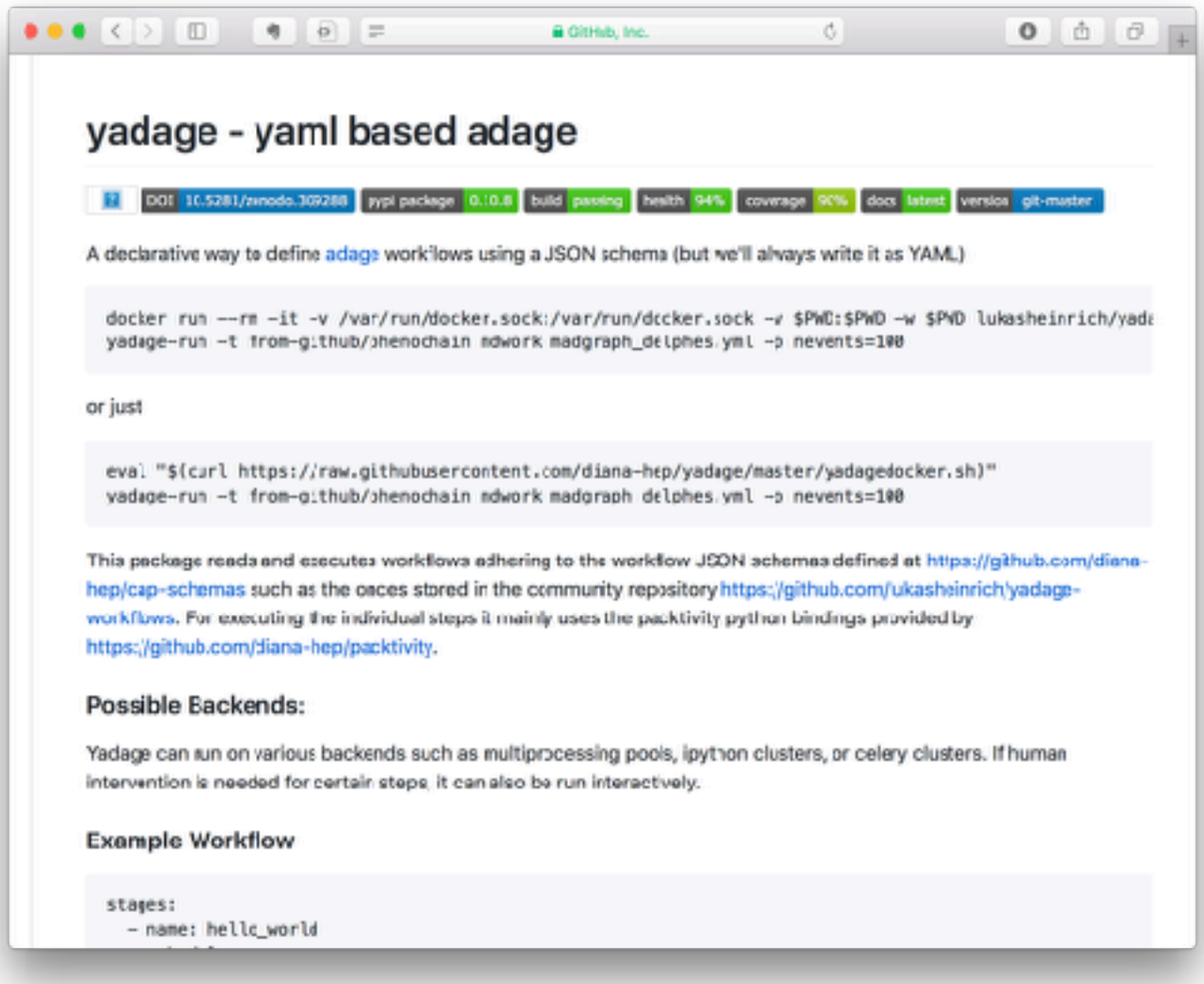
Yadage and Packtivity – analysis preservation using parametrized workflows

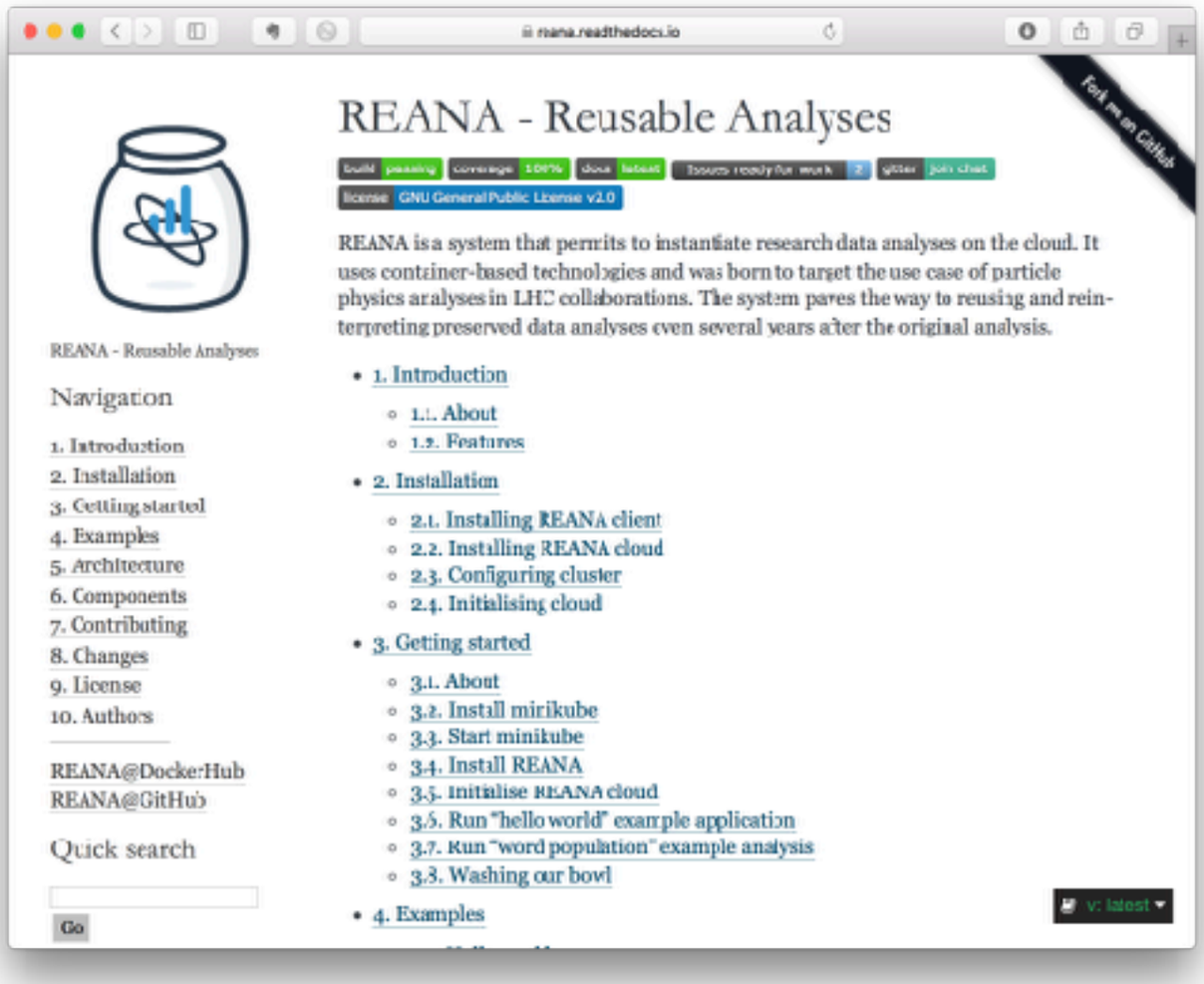
Kyle Cranmer[1] **and Lukas Heinrich**[1]

[1] Department of Physics, New York University, New York, USA

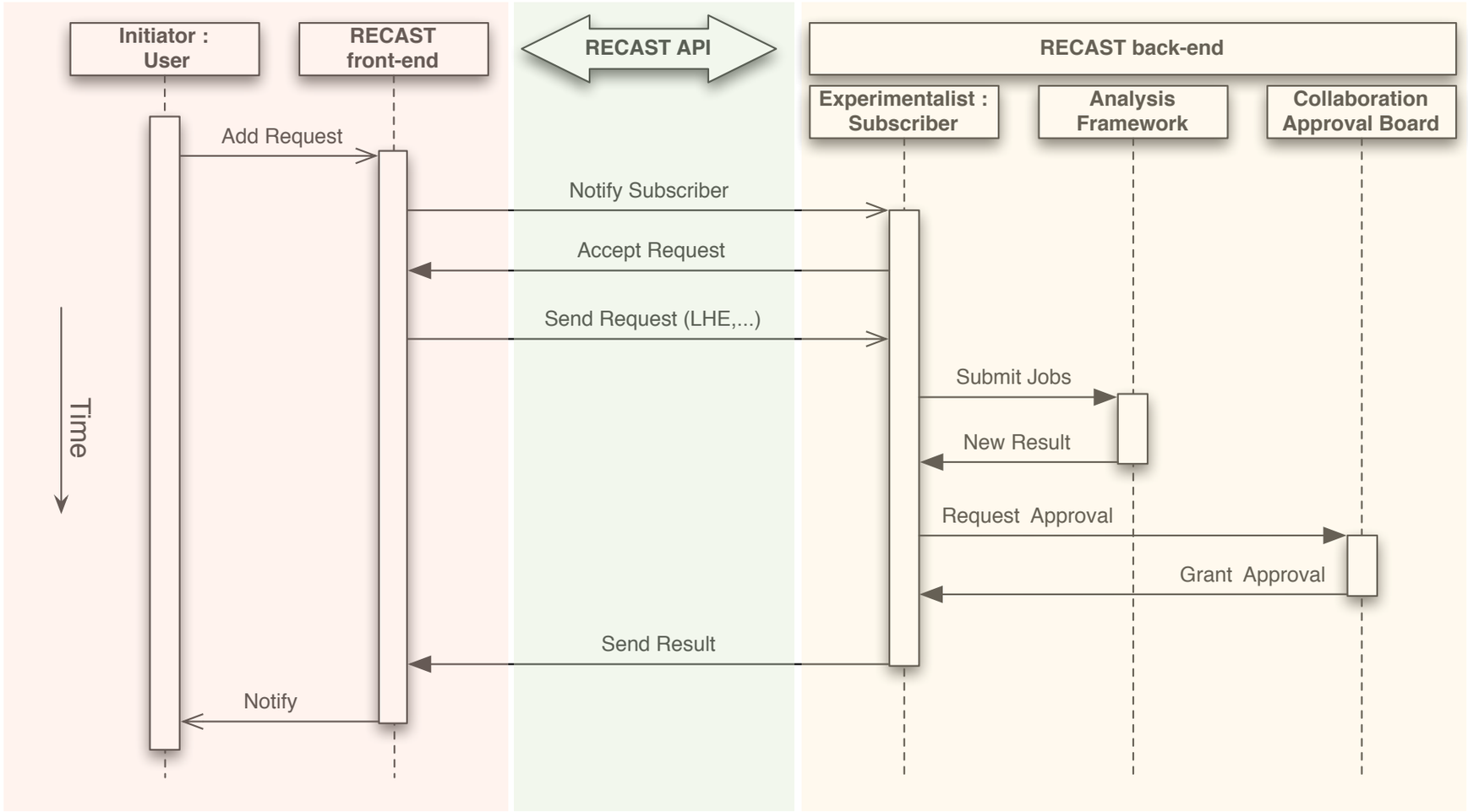
E-mail: `lukas.heinrich@cern.ch`

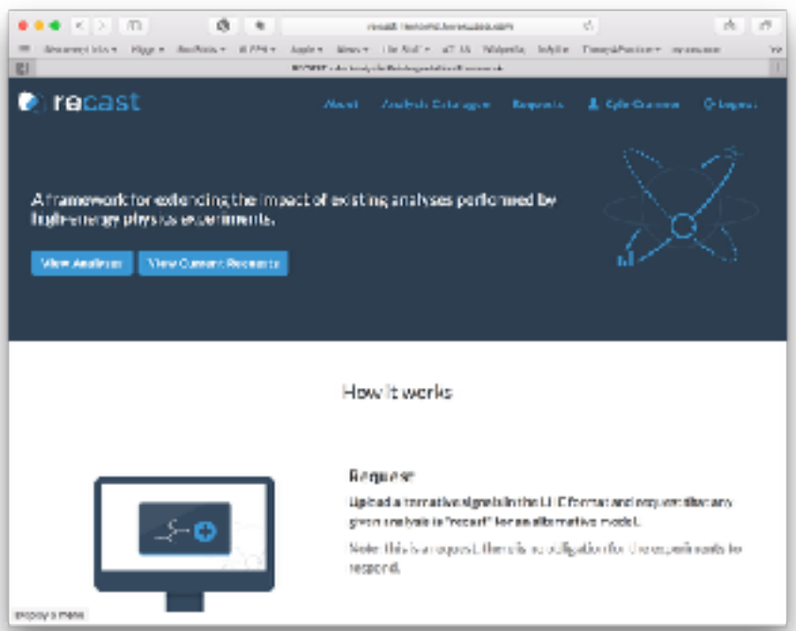
Abstract. Preserving data analyses produced by the collaborations at LHC in a parametrized fashion is crucial in order to maintain reproducibility and re-usability. We argue for a declarative description in terms of individual processing steps – "packtivities" – linked through a dynamic directed acyclic graph (DAG) and present an initial set of JSON schemas for such a description and an implementation – "yadage" – capable of executing workflows of analysis preserved via Linux containers.



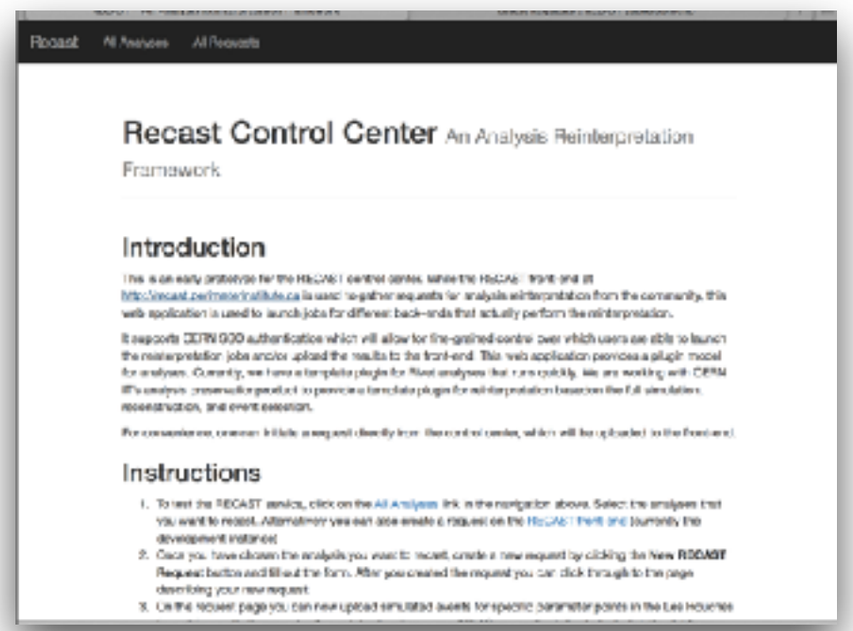


https://arxiv.org/pdf/1706.01878.pdf

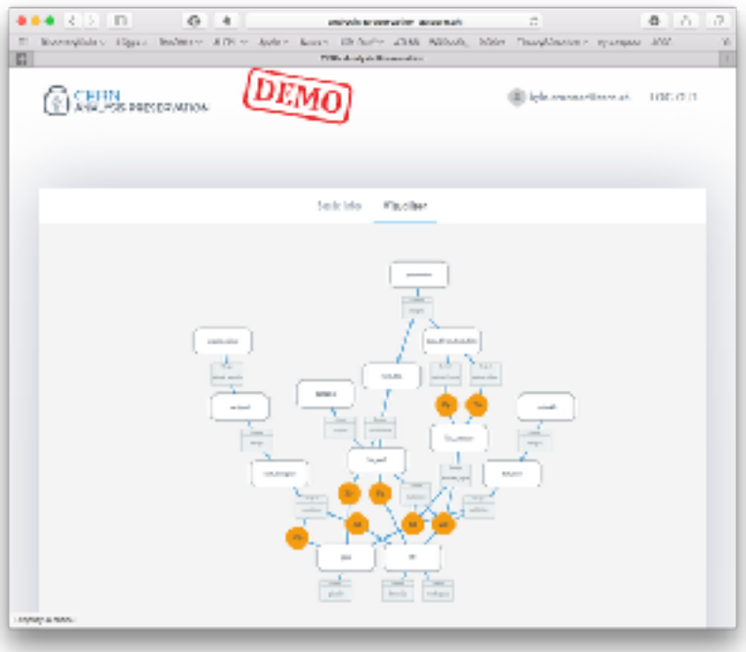




Front-End: public facing
collects requests



Control Center: not public, uses CERN auth.,
oversees processing of jobs on back-end



CERN Analysis Preservation:
Stores workflows, provides back-end
computing resources

Create standalone simulation tools to facilitate collaboration between HEP and machine learning community

By *Pierre Baldi, Peter Sadowski, Daniel Whiteson, Christian Lorenz Müller, Michael Williams, Lukas Heinrich, Steven Schramm, Maurizio Pierini, Sergei Gleyzer, Amir Farbin, jean-roch vlimant, Tim Head, Juan Pavez, Peter Elmer, Balázs Kégl, Andrey Ustyuzhanin, Vladimir Gligorov, Gilles Louppe, Kyle Cranmer*

dslhc | machinelearning | datascience | open data | simulation

Discussions at recent workshops have made it clear that one of the key barriers to collaboration between high energy physics and the machine learning community is access to training data. Recent successes in data sharing through the HiggsML and Flavours of Physics Kaggle challenges have borne much fruit, but required significant effort to coordinate.

While static simulated datasets are useful for challenges, in the course of investigating new machine learning techniques it is advantageous to be able to generate training data on demand (e.g. Refs. 1, 2, 3).
Therefore we recommend efforts be made to produce the ingredients required to facilitate such collaboration:

- Specific challenges for HEP experiments should be fully specified such that minimal domain-specific knowledge is required to attack them.
- Stand-alone simulators should be made open source. They should be developed to be easy to use without domain-specific expertise, while still being representative of real experimental challenges. Such a simulation will permit non-HEP researchers to generate realistic HEP datasets for training and testing. These simulators could range from truth-level simulation of a hard scattering to fast simulation like Delphes, to full GEANT4 simulation of sensor arrays.
- Performance metrics (objective functions) and operational constraints should be defined to evaluate proposed solutions.

Authors

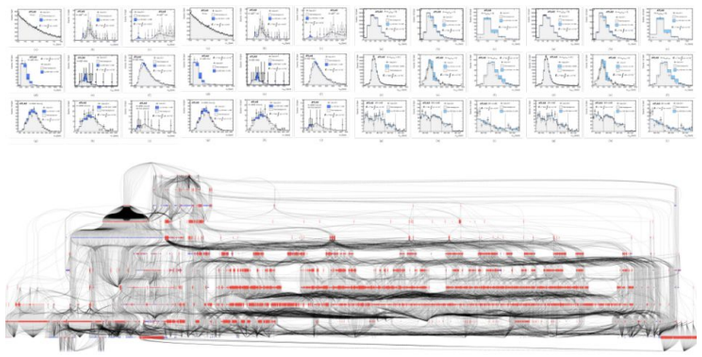
Pierre Baldi, Peter Sadowski, Daniel Whiteson, Christian Lorenz Müller, Michael Williams, Lukas Heinrich, Steven Schramm, Maurizio Pierini, Sergei Gleyzer, Amir Farbin, jean-roch vlimant, Tim Head, Juan Pavez, Peter Elmer, Balázs Kégl, Andrey Ustyuzhanin, Vladimir Gligorov, Gilles Louppe, Kyle Cranmer

Metadata

DOI 10.5281/zenodo.46864

Published: 26 Feb, 2016

Probabilistic programming frameworks

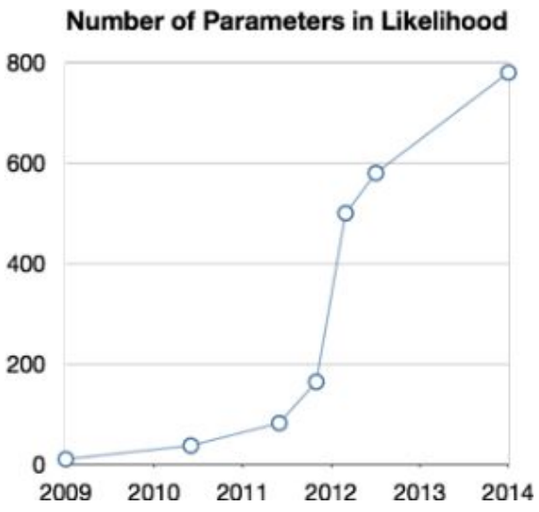


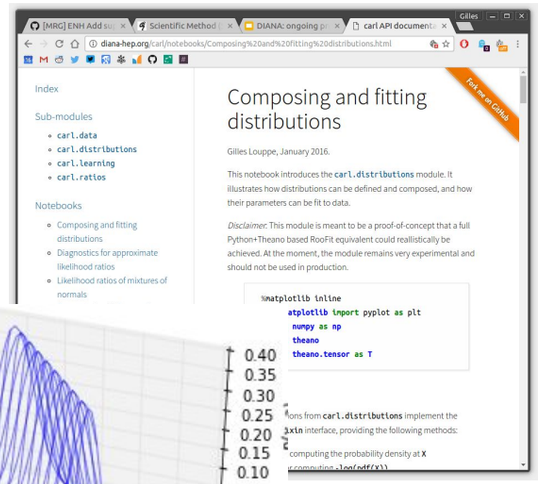
$$\mathbf{f}_{\text{tot}}(\mathcal{D}_{\text{sim}}, \mathcal{G}|\boldsymbol{\alpha}) = \prod_{c \in \text{channels}} \left[\text{Pois}(n_c|\nu_c(\boldsymbol{\alpha})) \prod_{e=1}^{n_c} f_c(x_{ce}|\boldsymbol{\alpha}) \right] \cdot \prod_{p \in \mathcal{S}} f_p(a_p|\alpha_p)$$

RooFit serves us well, but shows limits in terms of **scalability**.

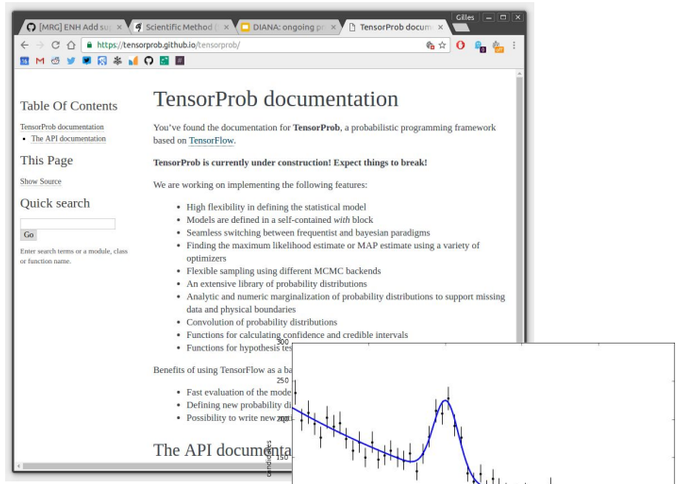
Using a data flow graph framework, RooFit would be **distributed**, **GPU-enabled** and automatically **differentiable**.

Feasibility? Certainly **within reach!** As illustrated by our tentative proof-of-concepts carl.distributions [Gilles Louppe] and tensorprob [Igor Babuschkin, now at DeepMind]. See also Edward.





carl.distributions



tensorprob

Edward

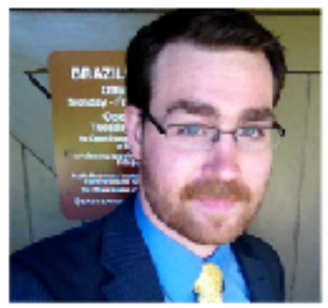
A library for probabilistic modeling, inference, and criticism.

Edward is a Python library for probabilistic modeling, inference, and criticism. It is a testbed for fast experimentation and research with probabilistic models, ranging from classical hierarchical models on small data sets to complex deep probabilistic models on large data sets. Edward fuses three fields: Bayesian statistics and machine learning, deep learning, and probabilistic programming.

It supports **modeling** with

Dustin Tran

Ph.D. Student
Columbia University
dustin@cs.columbia.edu (@dustinvtran,
http://dustintran.com



Matthew Feickert

High Energy Physics Ph.D. Candidate
Southern Methodist University
matthew.feickert@cern.ch or mfeickert@smu.edu
GitHub: matthewfeickert @HEPfeickert

THANKS!

Physicists:

Lukas Heinrich,
Cyril Becot,
Daniel Whiteson,
Michael Kagan,
Johann Brehmer,
Taylor Faucett
David Rousseau

ML/AI:

Gilles Louppe,
Juan Pavez,
Kyunghyun Cho,
Brenden Lake,
Pierre Baldi,
Peter Sadowski,
Uri Shalit,
Joan Bruna,
Yann LeCun,
Balázs Kégl
Cecile Germain

NYU CDS Masters Students

Xinyi Gong,
Zihao Wang,
Lanyu Shang,
Alex Pine,
Israel Malkin,
Charlie Guthrie,
Manoj Kumar,
Phil Yeres,
Michele Ceru
Hao Liu,
Li Ke,
Yuhao Zhao

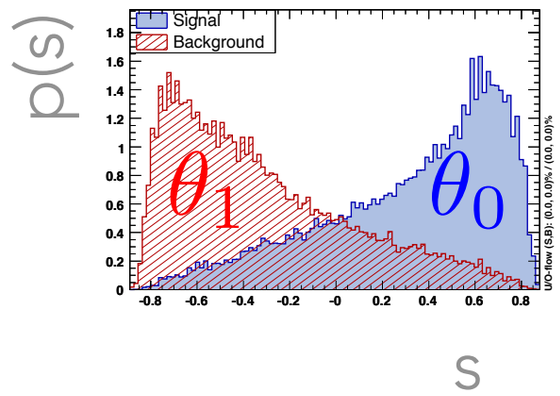
CARL

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: http://arxiv.org/abs/1506.02169

Physics / ML Dictionary

A correspondence between thermodynamics and inference

Colin H. LaMont and Paul A. Wiggins
*Departments of Physics, Bioengineering and Microbiology, University of Washington, Box 351560. 3910 15th Avenue Northeast, Seattle, WA 98195, USA**

Thermodynamics			**Statistics**	
Quantity:	**Interpretation:**		**Quantity:**	**Interpretation:**
$\beta = T^{-1}$	Inverse temperature	$\leftrightarrow$	N	Sample size
$\boldsymbol{\theta}$	State variables/vector	$\leftrightarrow$	$\boldsymbol{\theta}$	Model parameters
X^N	Quenched disorder	$\leftrightarrow$	X^N	Observations
$E_X(\boldsymbol{\theta})$	State energy	$\leftrightarrow$	$\hat{H}_X(\boldsymbol{\theta})$	Cross entropy estiamtor
E_0	Disorder-averaged ground state energy	$\leftrightarrow$	$\overline{H}_0$	Shannon entropy
$\rho(\boldsymbol{\theta})$	Density of states	$\leftrightarrow$	$\varpi(\boldsymbol{\theta})$	Prior
$\mathfrak{Z}$	Partition function	$\leftrightarrow$	Z	Evidence
$\mathfrak{Z}^{-1}\rho\ \exp -\beta E_X$	Normalized Boltzmann weight	$\leftrightarrow$	$\varpi(\boldsymbol{\theta}\|X^N)$	Posterior
$F = -\beta^{-1}\log\mathfrak{Z}$	Free energy	$\leftrightarrow$	$F = -N^{-1}\log Z$	Minus-log-evidence
$U = -\partial_\beta \beta F$	Average energy	$\leftrightarrow$	$U = -\partial_N N F$	Minus-log-prediction
$C = -\beta^2\partial_\beta^2 \beta F$	Heat capacity	$\leftrightarrow$	$C = -N^2\partial_N^2 N F$	*Learning capacity*
$S = \beta^2\partial_\beta F$	Gibbs entropy	$\leftrightarrow$	$S = N^2\partial_N F$	*Complexity*

TABLE I. **Thermodynamic-Bayesian correspondence.** The top half of the table lists the correspondences that can be determined directly from the definition of the marginal likelihood as the partition function. The lower half of the table lists the implied thermodynamic expressions and their existing or proposed statistical interpretation.