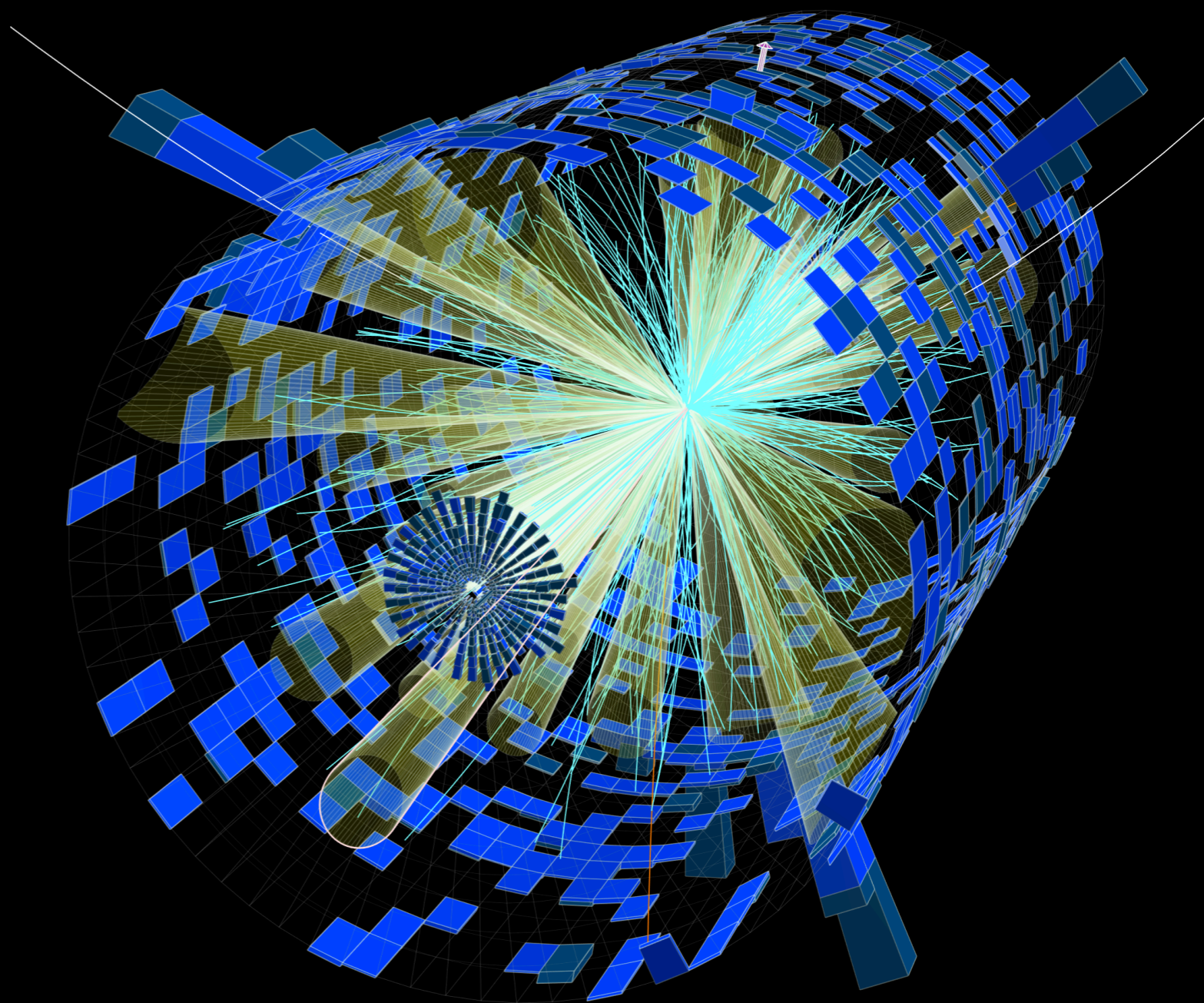




STATISTICS & MACHINE LEARNING



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

Handouts

STATISTICAL DECISION THEORY IN 1 SLIDE

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

$f(x|\theta)$ - statistical model; $\pi(\theta)$ - prior

$\delta: X \rightarrow A$ - **decision rule** (take some action based on observation)

$L: \Theta \times A \rightarrow \mathbb{R}$ - **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)

$\rho(\pi, \delta | x) = E_{\pi(\theta|x)}[L(\theta, \delta(x))]$ - **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.

THUMBNAIL OF THE STATISTICAL PROCEDURE

Follow LHC-HCG Combination Procedures

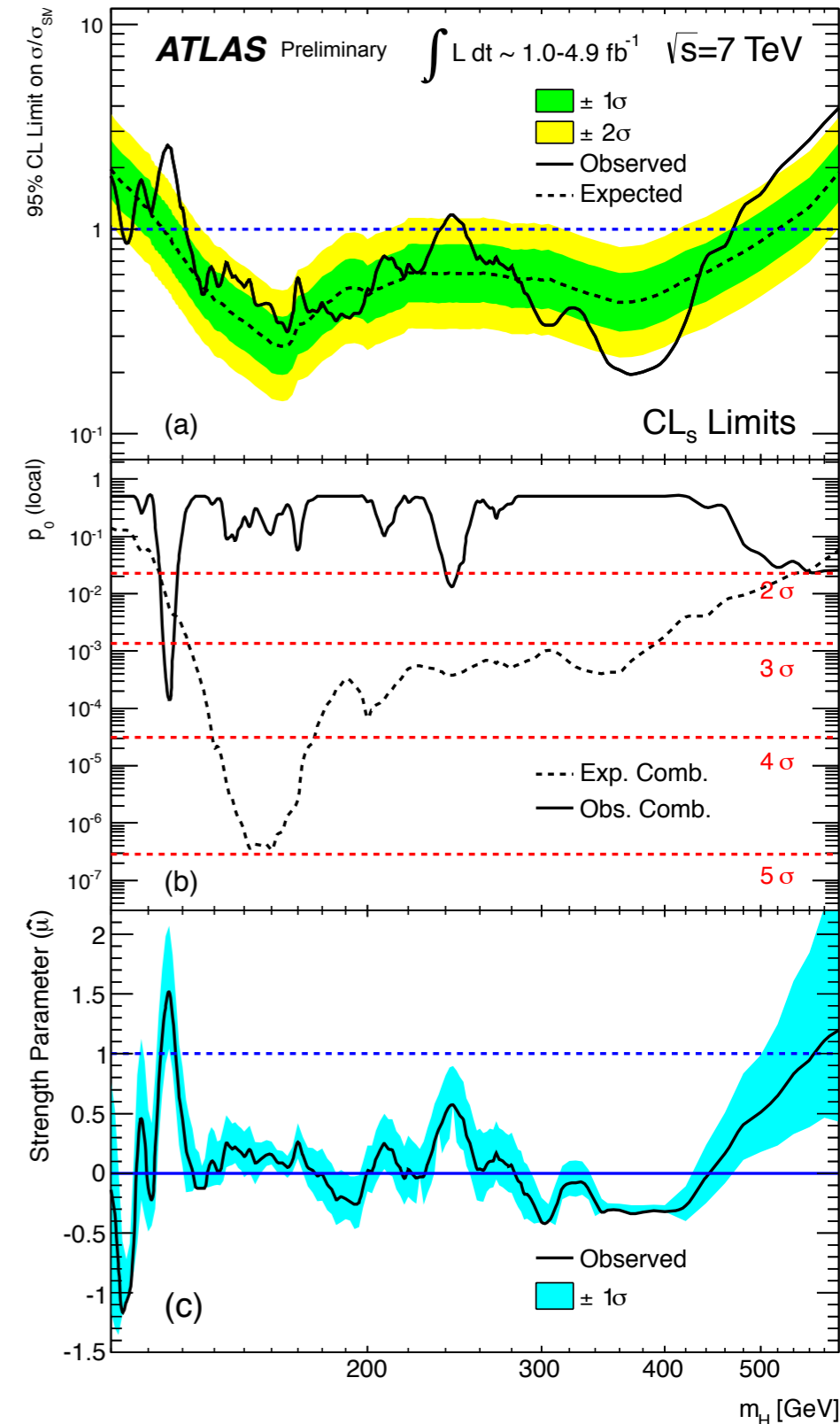
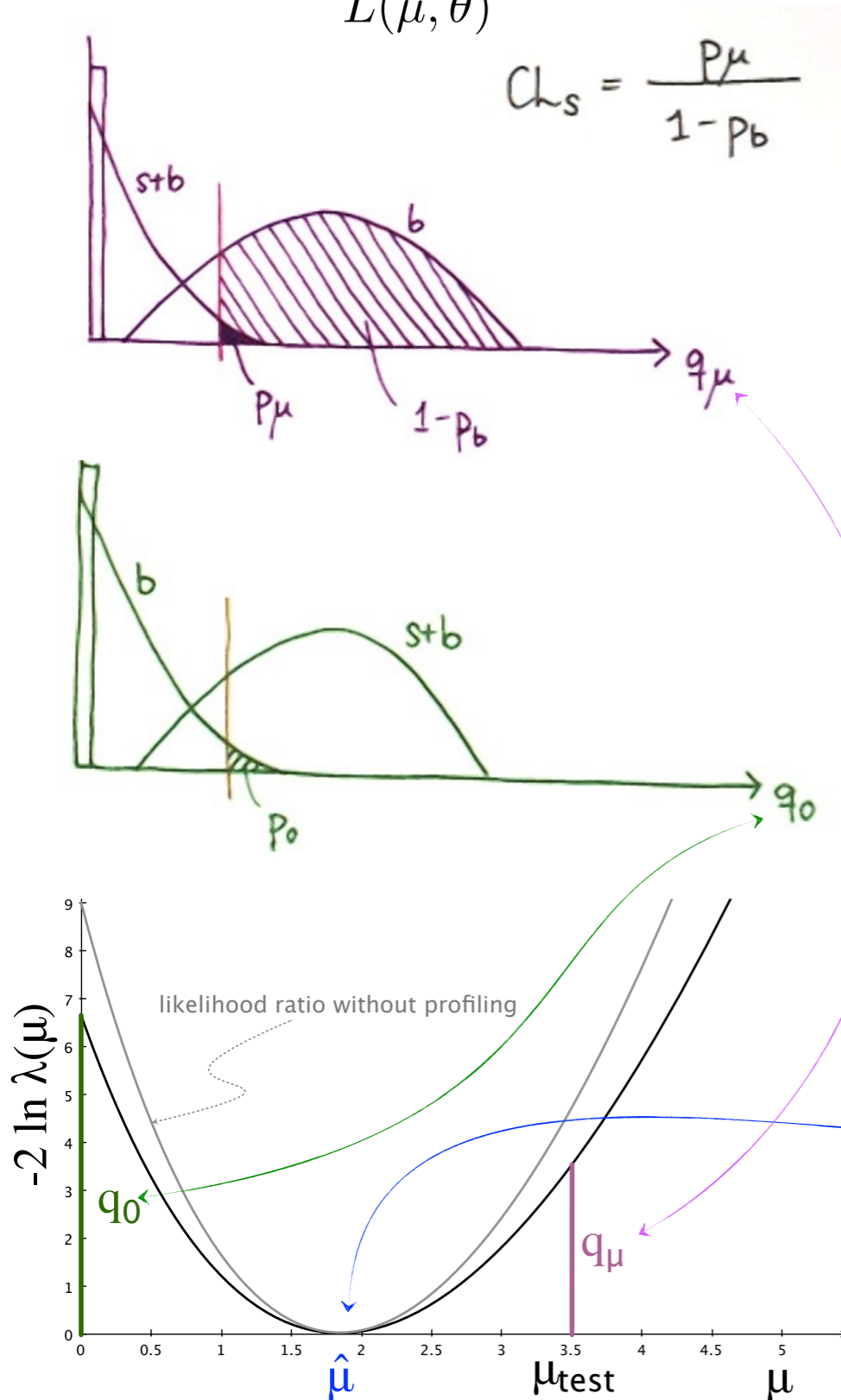
$$\lambda(\mu) = \frac{L(\mu, \hat{\hat{\theta}}(\mu))}{L(\hat{\mu}, \hat{\theta})}$$

$$CL_s = \frac{p_\mu}{1-p_b}$$

CL_s to test
signal
hypothesis

p_0 to test
background
hypothesis

$\hat{\mu}$ to estimate
signal strength



SLIDES FROM THE LAST DAY

My GGI lectures were mainly on the blackboard, but on the last day I showed a selection from these slides.



GGI LECTURES ON THE THEORY OF FUNDAMENTAL INTERACTIONS

Galileo Galilei Institute for Theoretical Physics
Firenze

7-25 JANUARY 2019

PROGRAM

[Schedule \(PDF\)](#)

Lectures and topics

KYLE CRANMER (NYU) [video](#)

GILLES LOUPPE (Liege U.) [video](#)

Statistics and Machine Learning

Lecture 1: [Fundamentals of machine learning](#)

Lecture 2: [Neural networks](#)

Lecture 3: [Generative models](#)

RAPHAEL FLAUGER (UC San Diego)

The Cosmic Microwave Background

JERNEJ KAMENIK (Ljubljana U.)

Flavour Physics and CP Violation

[Home](#)

[Program](#)

[Practical information](#)

[Registered participants](#)

[Photo gallery](#)

[How to reach us](#)

[2014 Program](#)

[2015 Program](#)

[2016 Program](#)

[2017 Program](#)

[2018 Program](#)

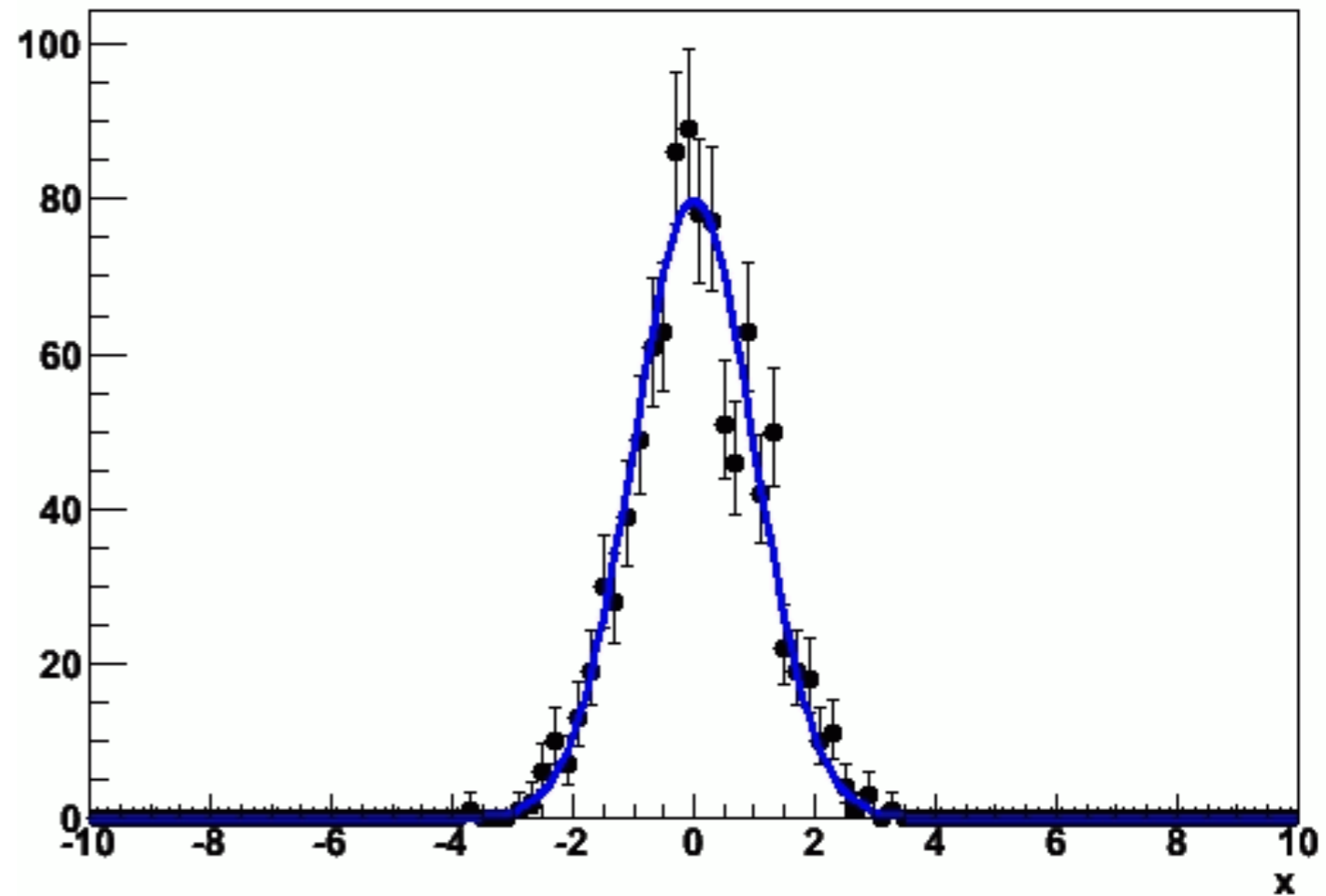
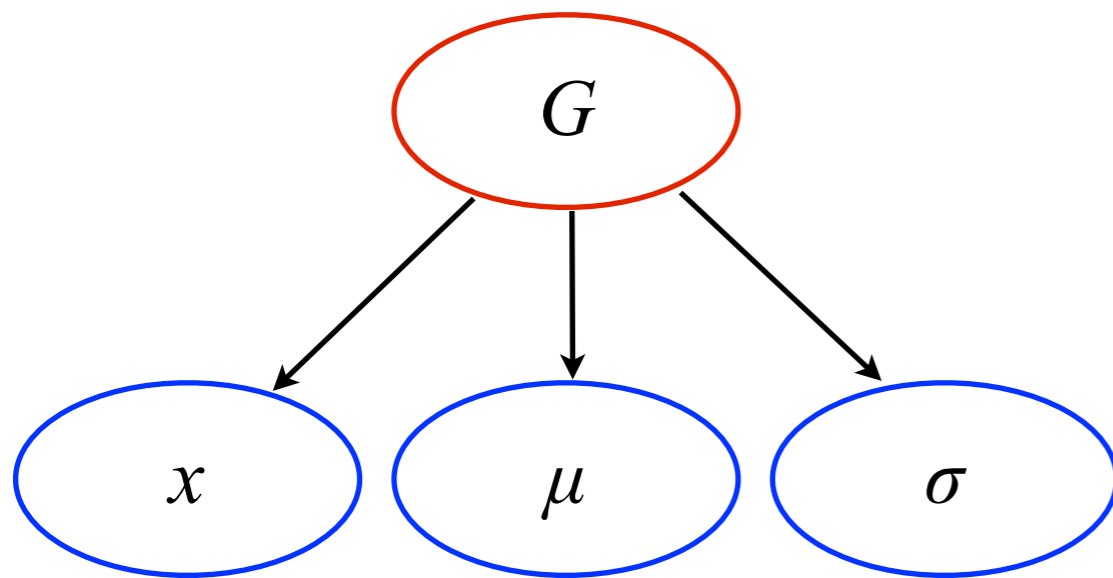
Building a Statistical Model

Systematics & Nuisance Parameters

VISUALIZING PROBABILITY MODELS

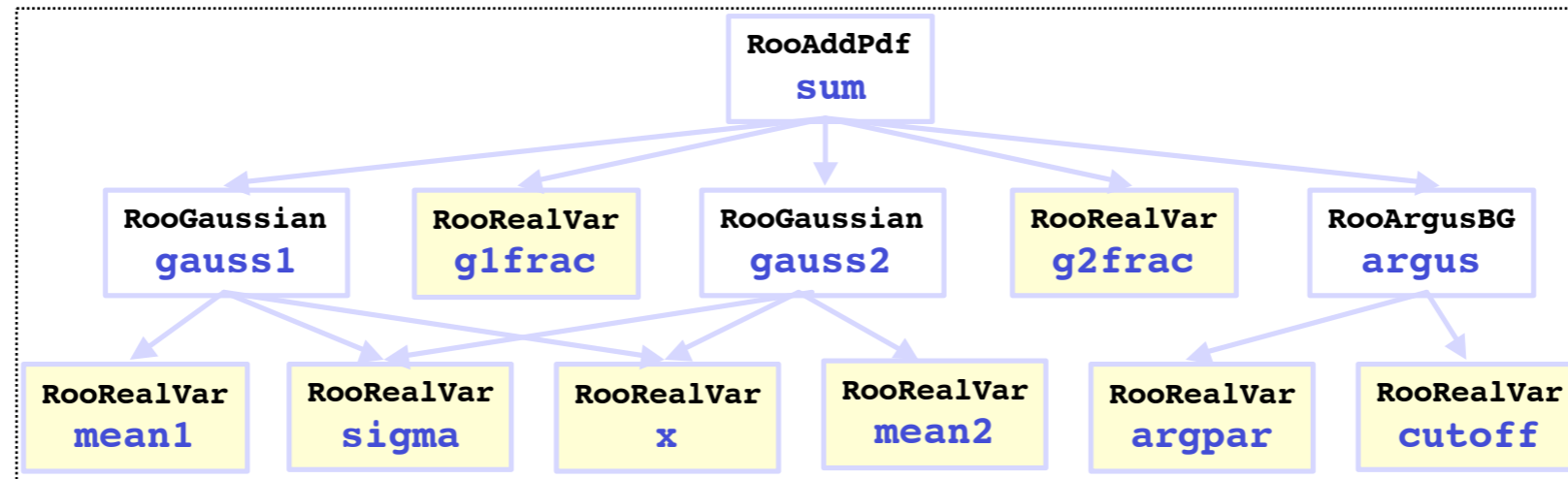
I will represent PDFs graphically as below (directed acyclic graph)

- ▶ eg. a Gaussian $G(x|\mu, \sigma)$ is parametrized by (μ, σ)
- ▶ every node is a real-valued function of the nodes below

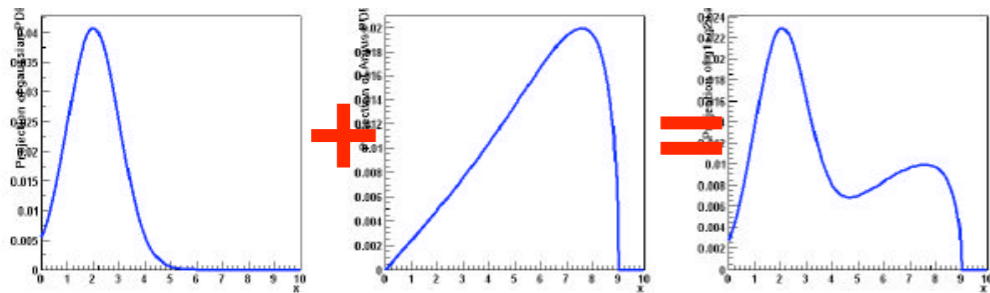


ROOT: A DATA MODELING TOOLKIT

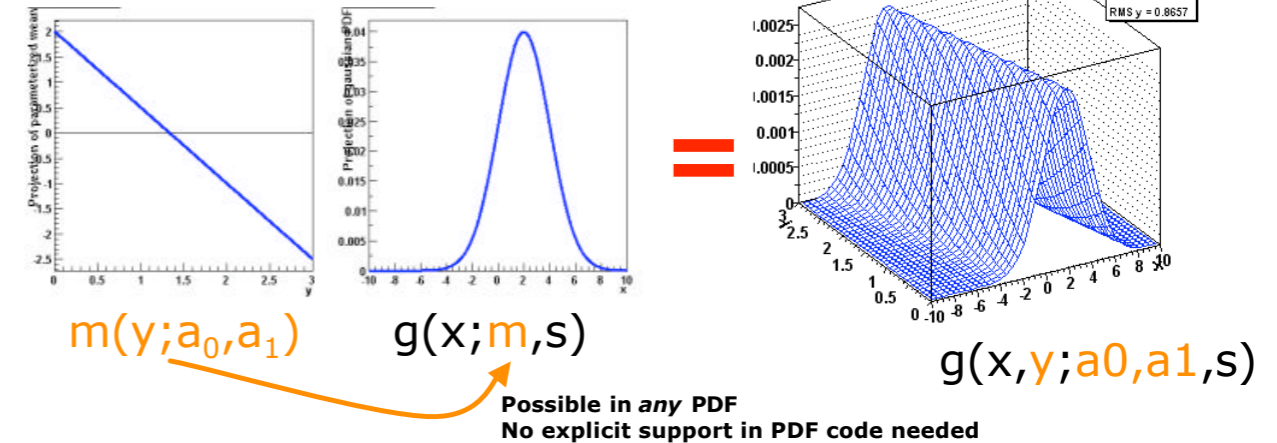
RootFit is a major tool developed at BaBar for data modeling.
RootStats provides higher-level statistical tools based on these PDFs.



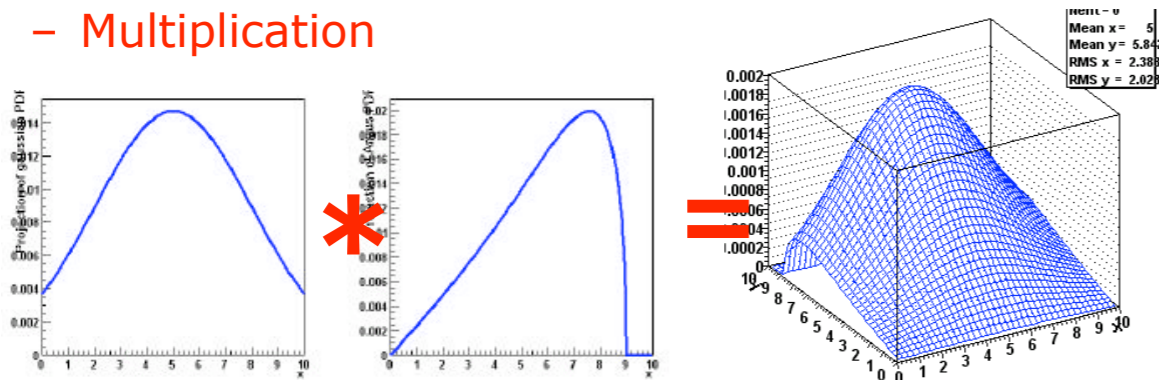
- Addition



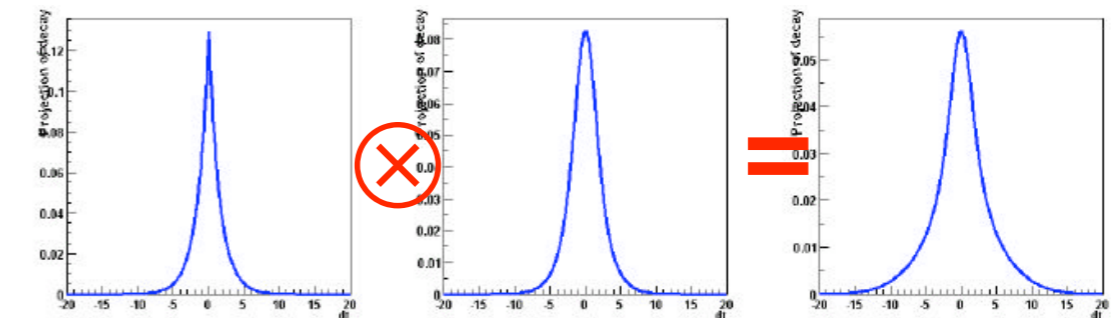
- Composition ('plug & play')



- Multiplication



- Convolution



Wouter Verkerke,

Wouter Verkerke, UCSB

MARKED POISSON PROCESS

Channel: a subset of the data defined by some selection requirements.

- eg. all events with 4 electrons with energy > 10 GeV
- n : number of events observed in the channel
- ν : number of events expected in the channel

Discriminating variable: a property of those events that can be measured and which helps discriminate the signal from background

- eg. the invariant mass of two particles
- $f(x)$: the p.d.f. of the discriminating variable x

$$\mathcal{D} = \{x_1, \dots, x_n\}$$

Marked Poisson Process / Extended Likelihood:

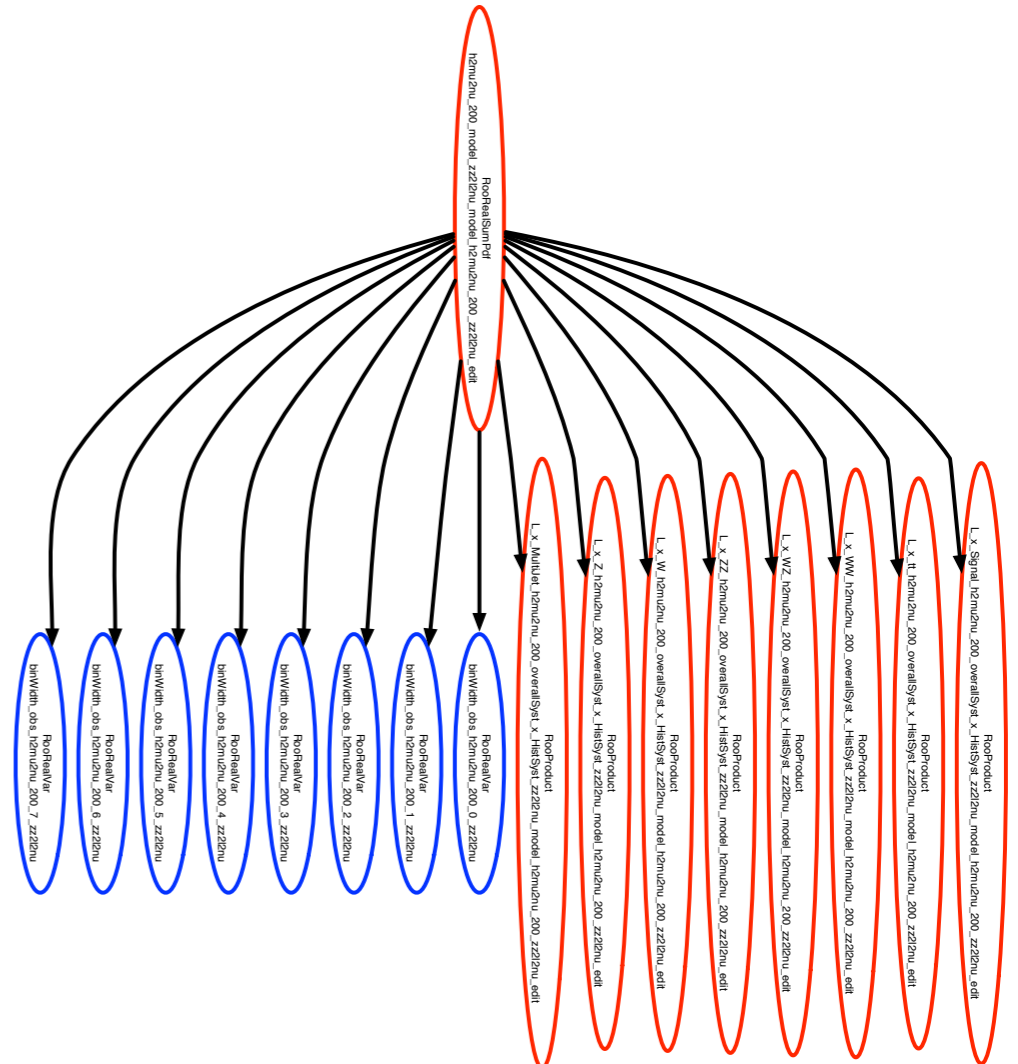
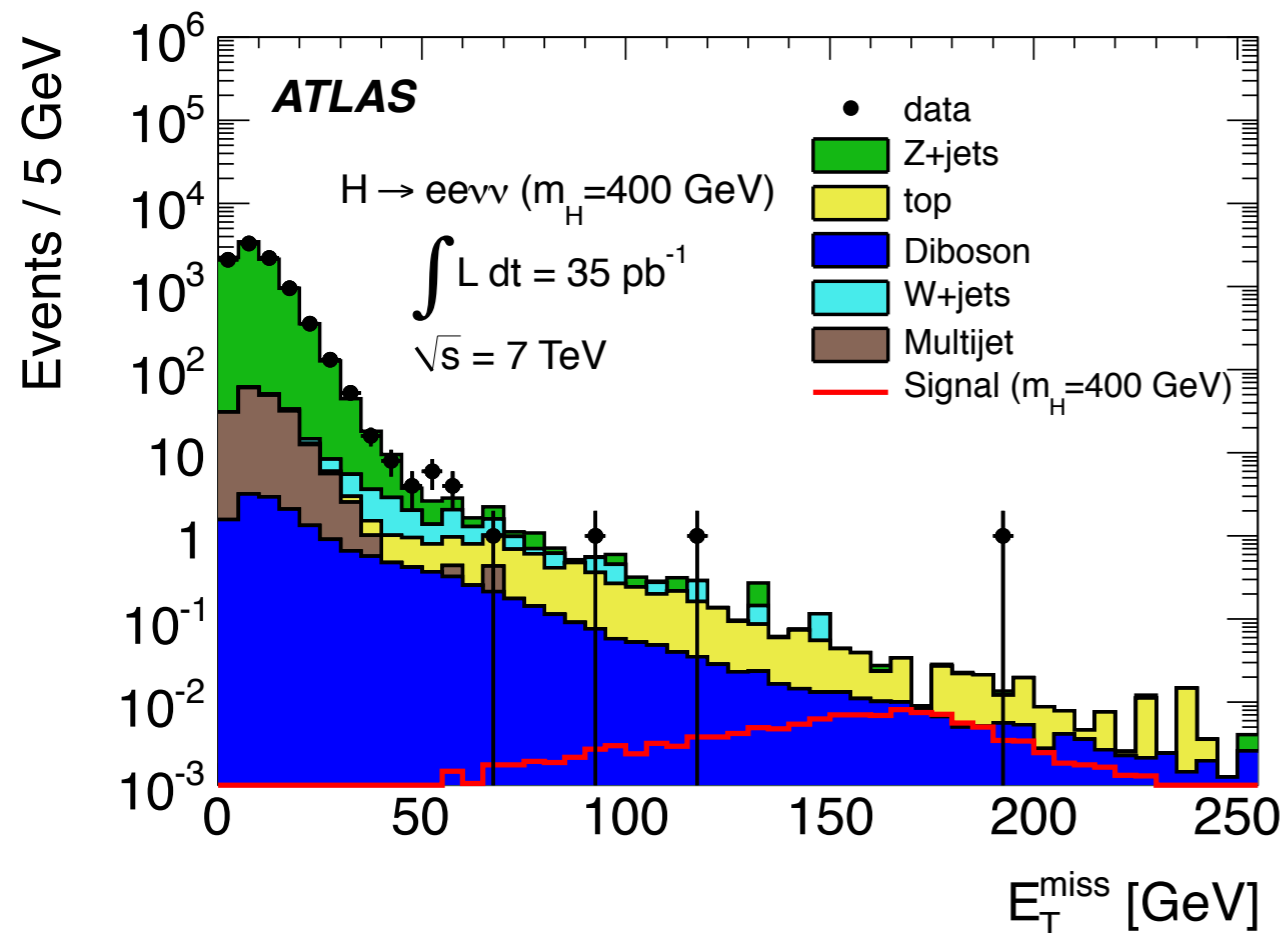
$$\mathbf{f}(\mathcal{D}|\nu) = \text{Pois}(n|\nu) \prod_{e=1}^n f(x_e)$$

MIXTURE MODEL

Sample: a sample of simulated events corresponding to particular type interaction that populates the channel.

- statisticians call this a mixture model

$$f(x) = \frac{1}{\nu_{\text{tot}}} \sum_{s \in \text{samples}} \nu_s f_s(x) , \quad \nu_{\text{tot}} = \sum_{s \in \text{samples}} \nu_s$$



PARAMETRIZING THE MODEL $\boldsymbol{\alpha} = (\mu, \boldsymbol{\theta})$

Parameters of interest (μ): parameters of the theory that modify the rates and shapes of the distributions, eg.

- the mass of a hypothesized particle
- the “signal strength” $\mu=0$ no signal, $\mu=1$ predicted signal rate

Nuisance parameters ($\boldsymbol{\theta}$ or α_p): associated to uncertainty in:

- response of the detector (calibration)
- phenomenological model of interaction in non-perturbative regime

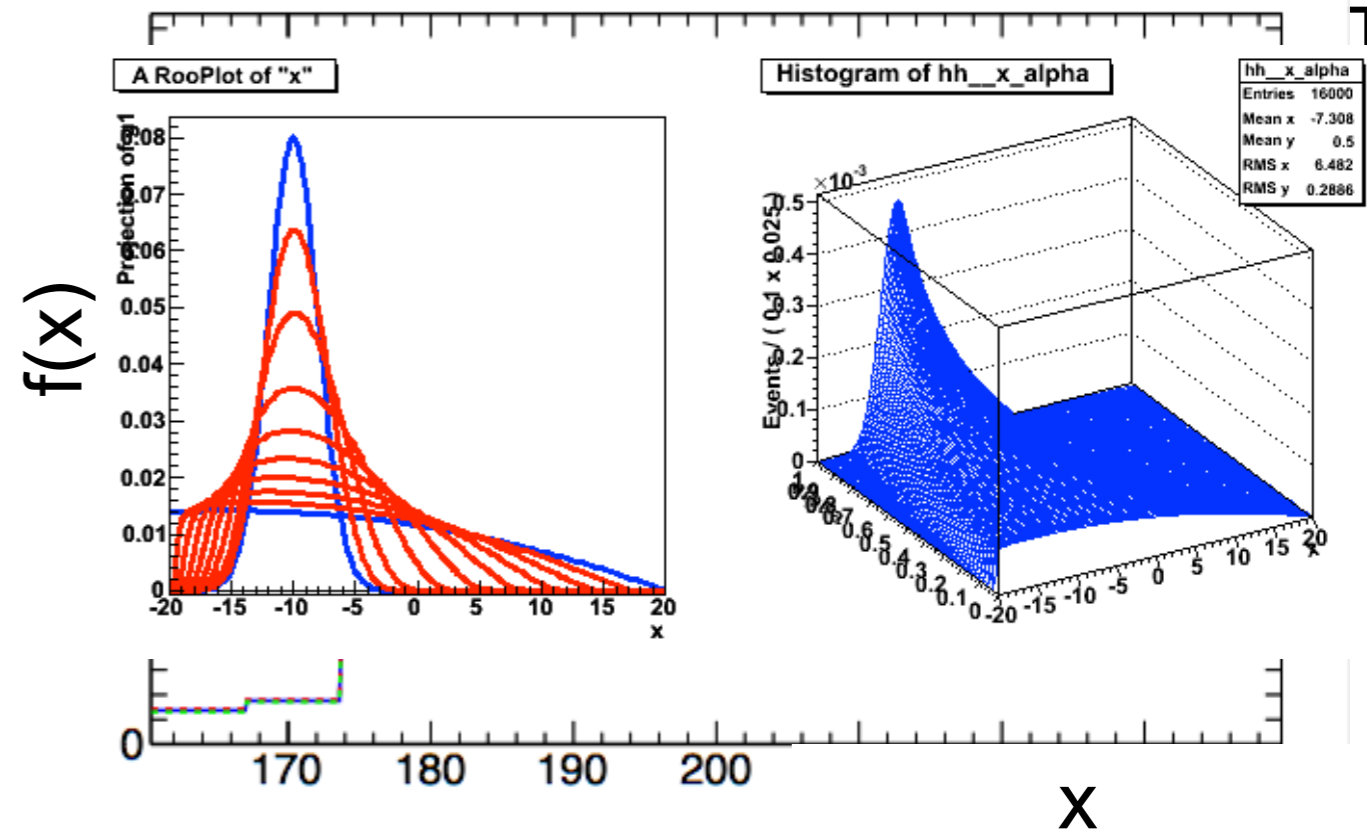
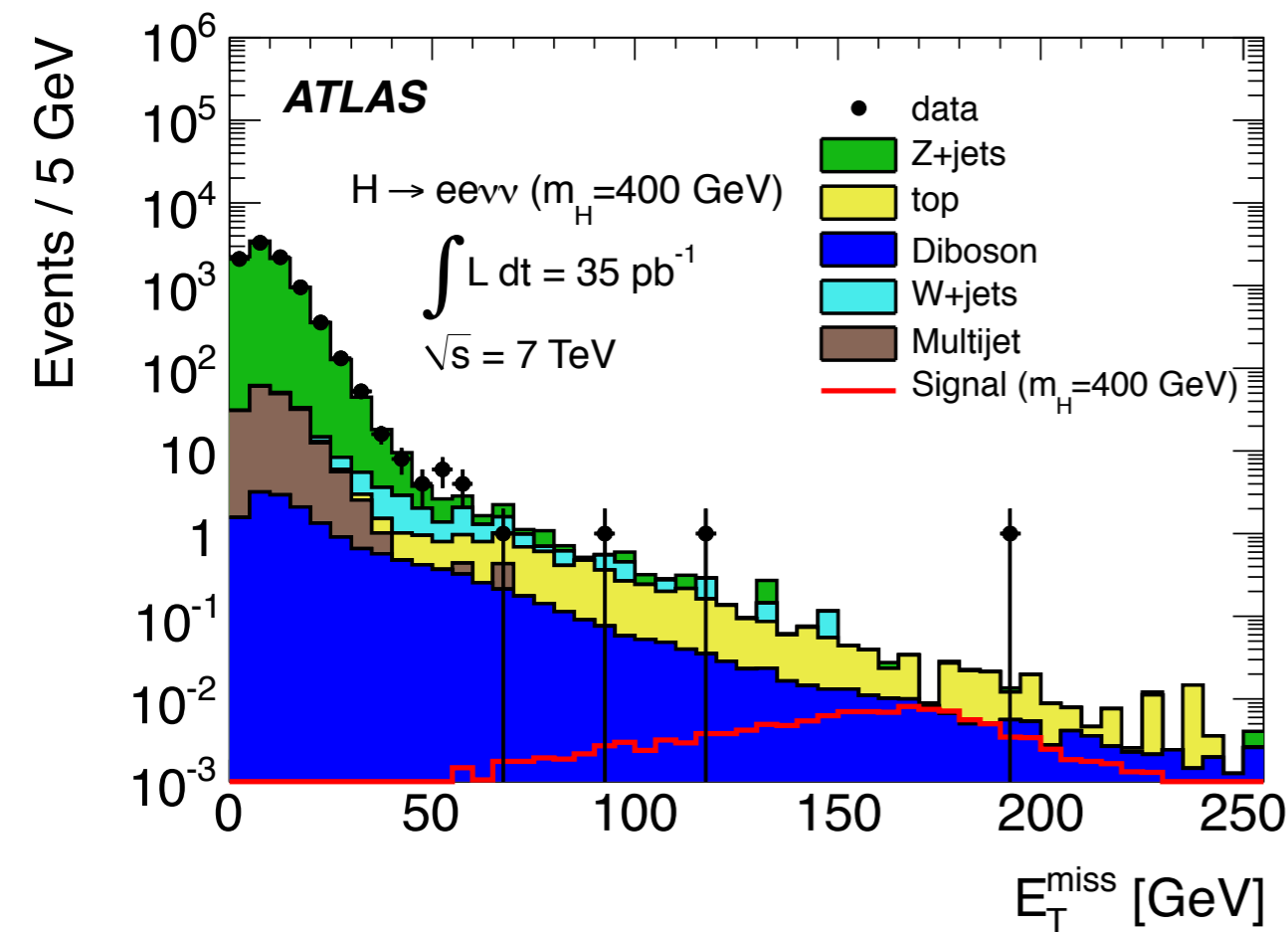
Lead to a parametrized model: $\nu \rightarrow \nu(\boldsymbol{\alpha}), f(x) \rightarrow f(x|\boldsymbol{\alpha})$

$$\mathbf{f}(\mathcal{D}|\boldsymbol{\alpha}) = \text{Pois}(n|\nu(\boldsymbol{\alpha})) \prod_{e=1}^n f(x_e|\boldsymbol{\alpha})$$

INCORPORATING SYSTEMATIC EFFECTS

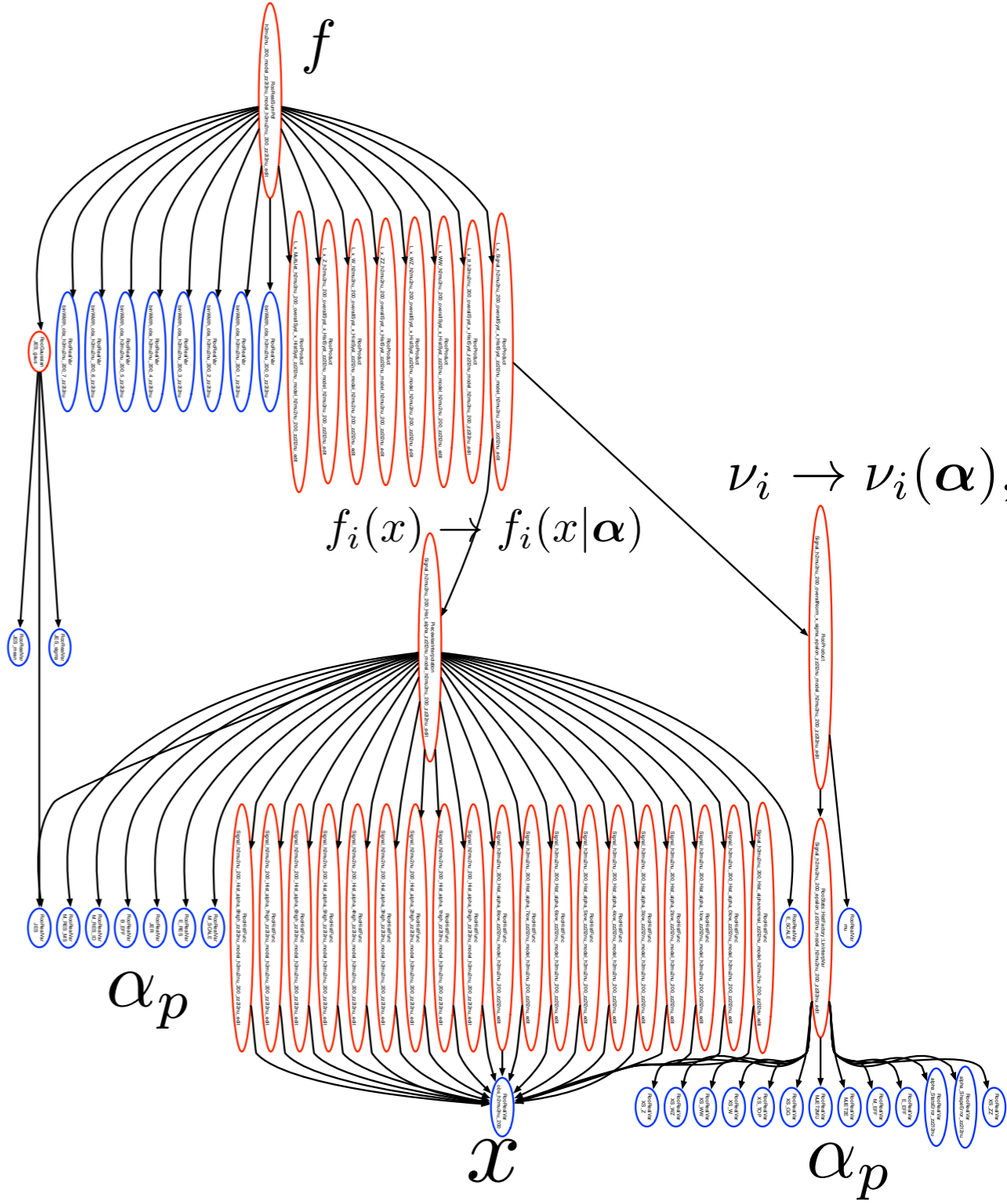
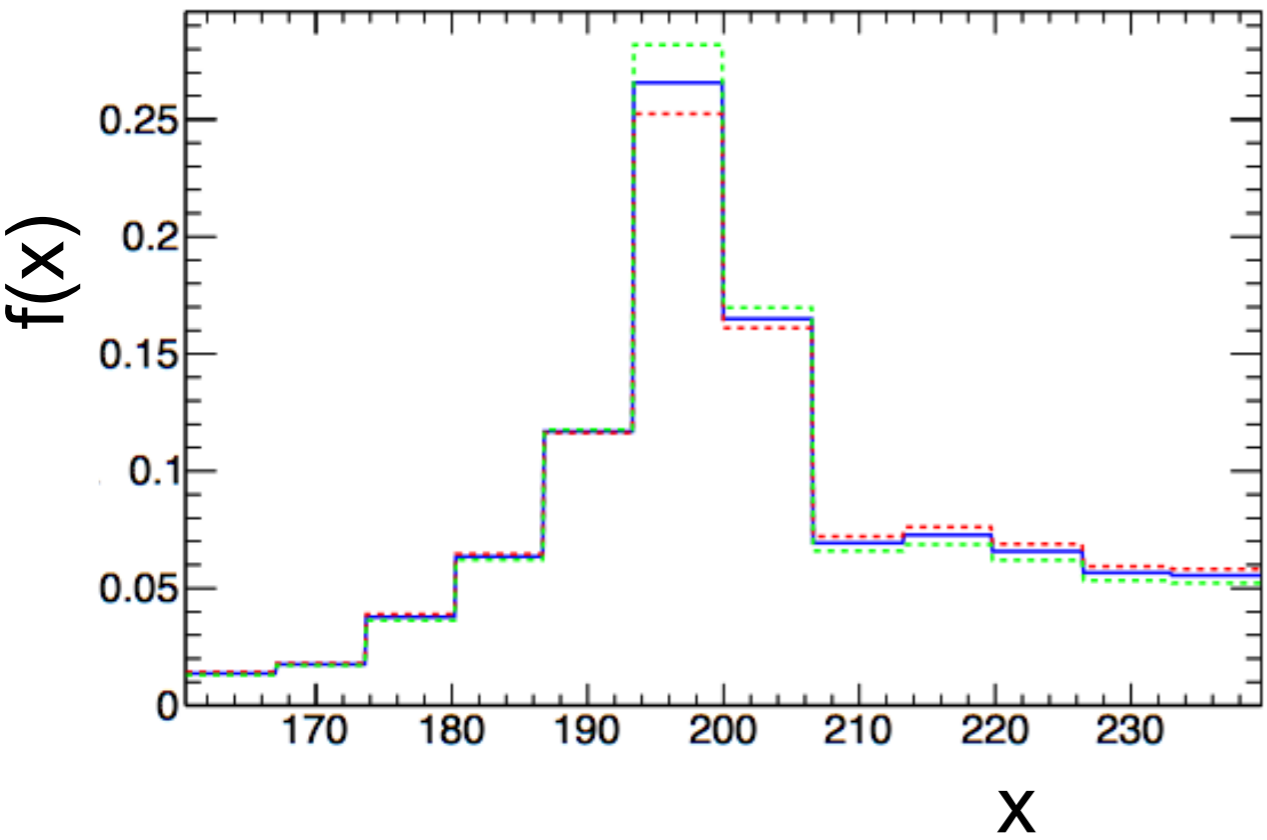
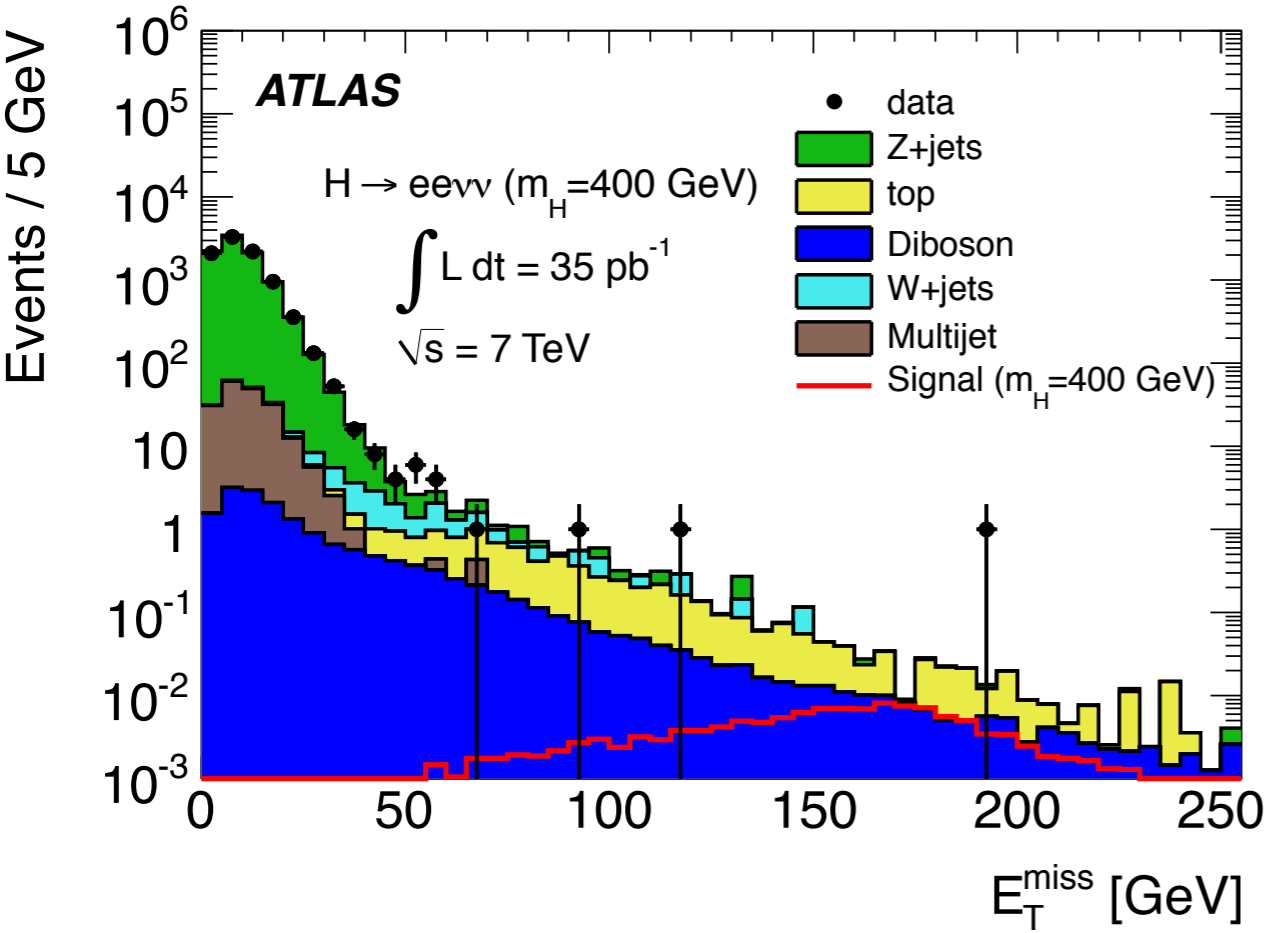
Tabulate effect of individual variations of sources of systematic uncertainty

- typically one at a time evaluated at nominal and “ $\pm 1 \sigma$ ”
- use some form of interpolation to parametrize p^{th} variation in terms of **nuisance parameter** α_p



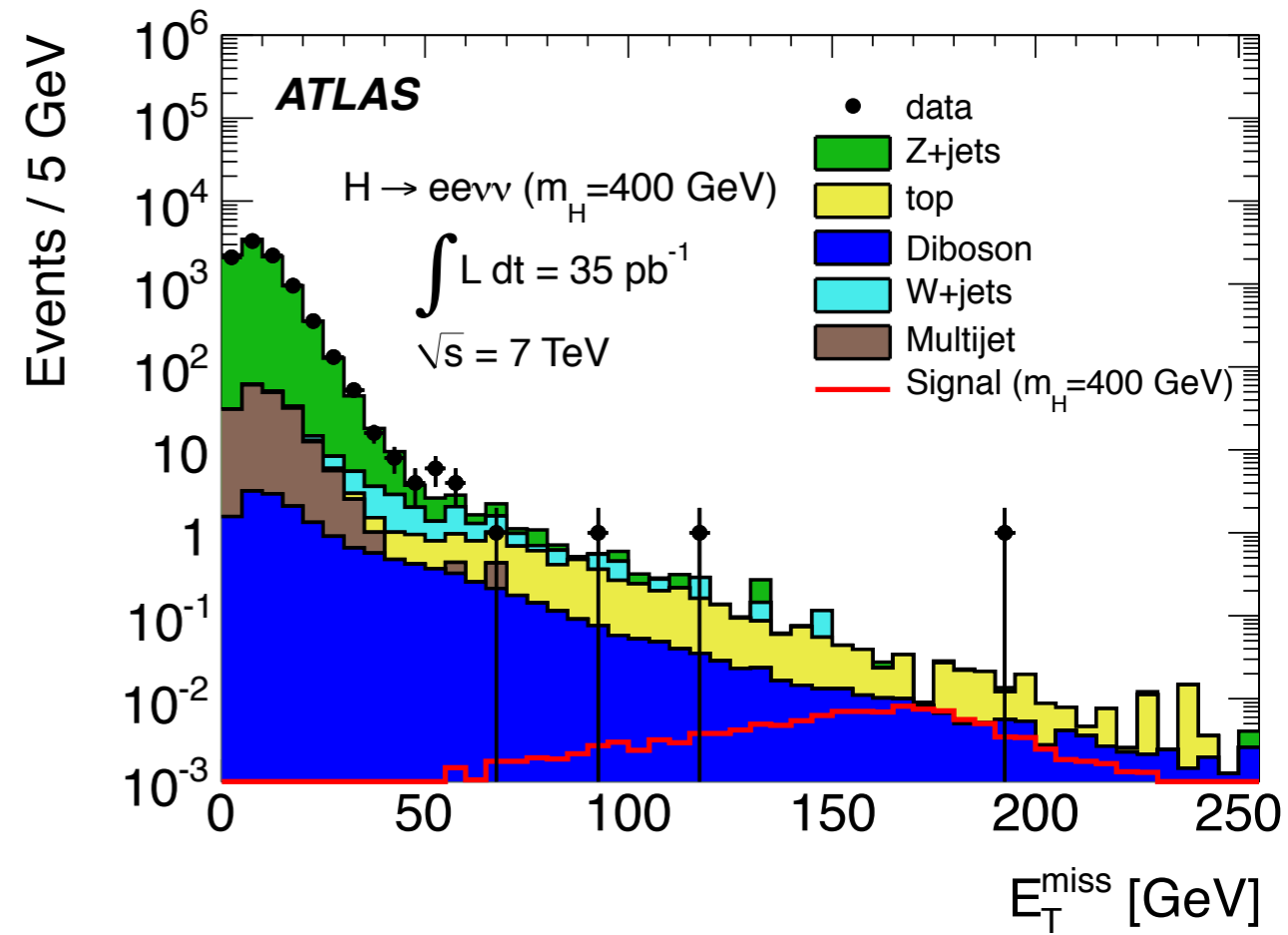
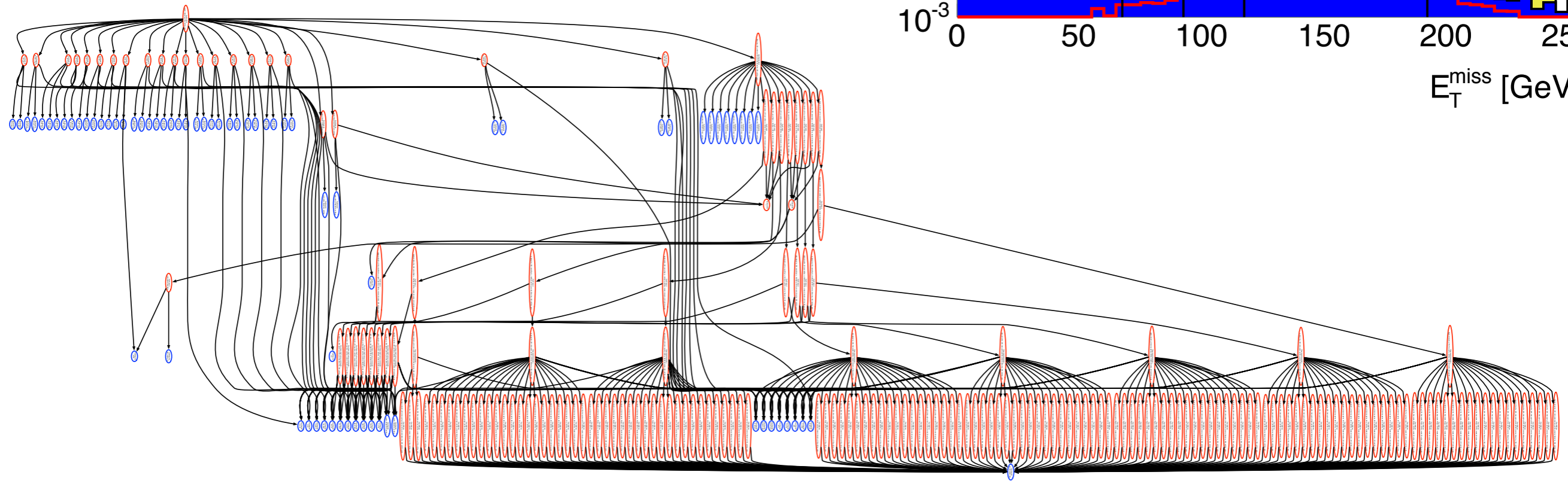
$$\mathbf{f}(\mathcal{D}|\boldsymbol{\alpha}) = \text{Pois}(n|\nu(\boldsymbol{\alpha})) \prod_{e=1}^n f(x_e|\boldsymbol{\alpha})$$

VISUALIZING THE MODEL FOR ONE CHANNEL



VISUALIZING THE MODEL FOR ONE CHANNEL

After parametrizing each component of the mixture model, the pdf for a single channel might look like this



SIMULTANEOUS MULTI-CHANNEL MODEL

Simultaneous Multi-Channel Model: Several disjoint regions of the data are modeled simultaneously. Identification of common parameters across many channels requires coordination between groups such that meaning of the parameters are really the same.

$$\mathbf{f}_{\text{sim}}(\mathcal{D}_{\text{sim}}|\boldsymbol{\alpha}) = \prod_{c \in \text{channels}} \left[\text{Pois}(n_c | \nu_c(\boldsymbol{\alpha})) \prod_{e=1}^{n_c} f_c(x_{ce} | \boldsymbol{\alpha}) \right]$$

where $\mathcal{D}_{\text{sim}} = \{\mathcal{D}_1, \dots, \mathcal{D}_{c_{\text{max}}}\}$

Control Regions: Some channels are not populated by signal processes, but are used to constrain the nuisance parameters

- attempt to describe systematics in a statistical language
- Prototypical Example: “on/off” problem with unknown ν_b

$$\mathbf{f}(n, m | \mu, \nu_b) = \underbrace{\text{Pois}(n | \mu + \nu_b)}_{\text{signal region}} \cdot \underbrace{\text{Pois}(m | \tau \nu_b)}_{\text{control region}}$$

CONSTRAINT TERMS

Often detailed statistical model for auxiliary measurements that measure certain nuisance parameters are not available.

- ▶ one typically has MLE for α_p , denoted a_p and standard error

Constraint Terms: are idealized pdfs for the MLE.

$$f_p(a_p|\alpha_p) \quad \text{for } p \in \mathbb{S}$$

- ▶ common choices are Gaussian, Poisson, and log-normal
- ▶ New: careful to write constraint term a frequentist way
- ▶ Previously: $\pi(\alpha_p|a_p) = f_p(a_p|\alpha_p)\eta(\alpha_p)$ with uniform η

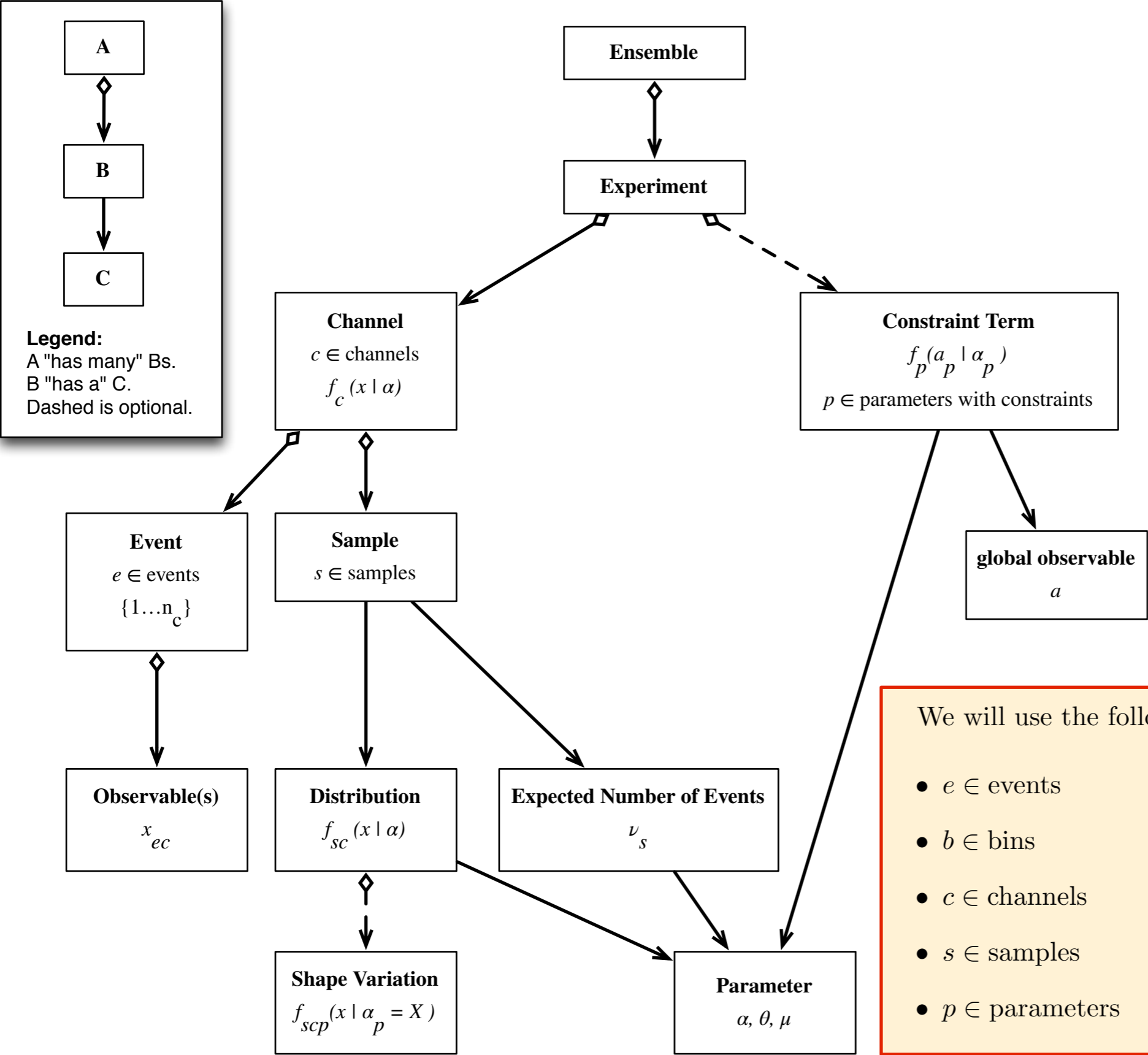
Simultaneous Multi-Channel Model with constraints:

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

where

$$\mathcal{D}_{\text{sim}} = \{\mathcal{D}_1, \dots, \mathcal{D}_{c_{\text{max}}}\}, \quad \mathcal{G} = \{a_p\} \quad \text{for } p \in \mathbb{S}$$

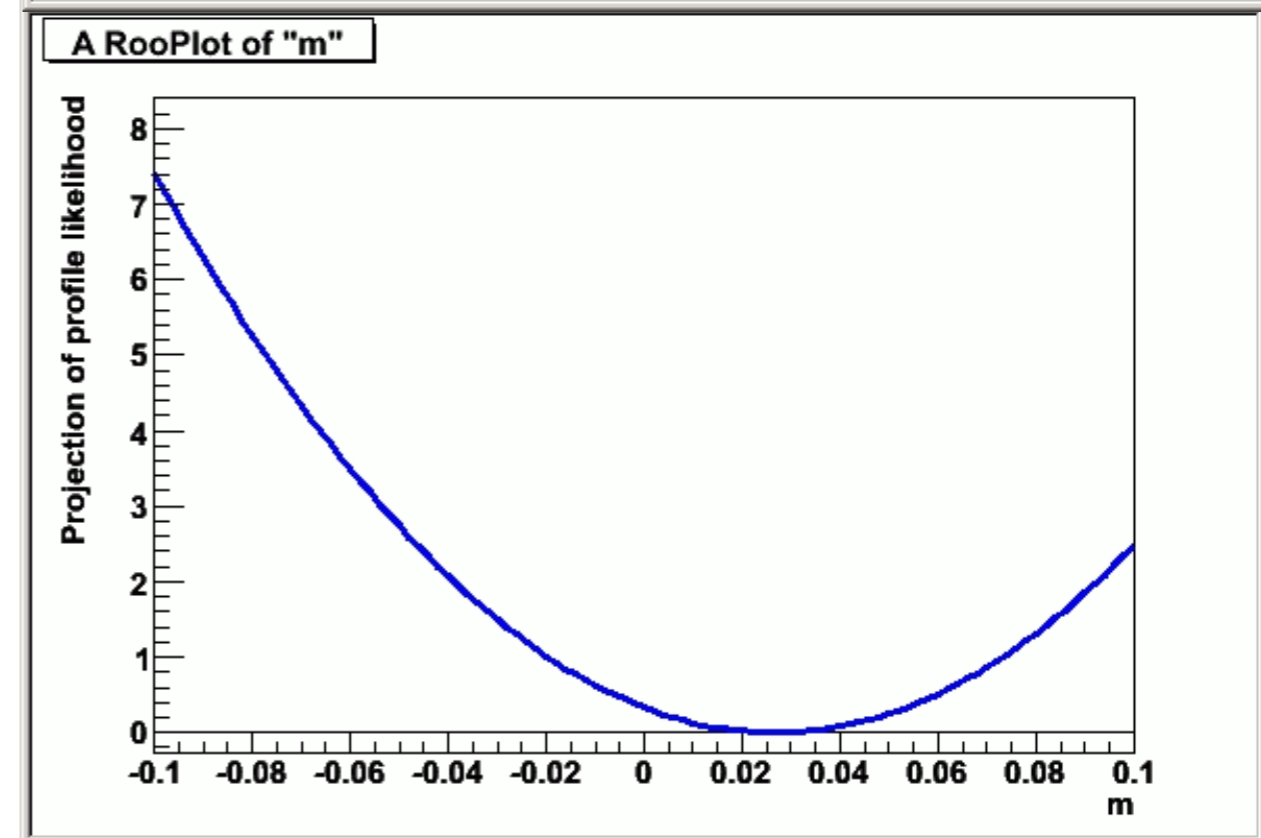
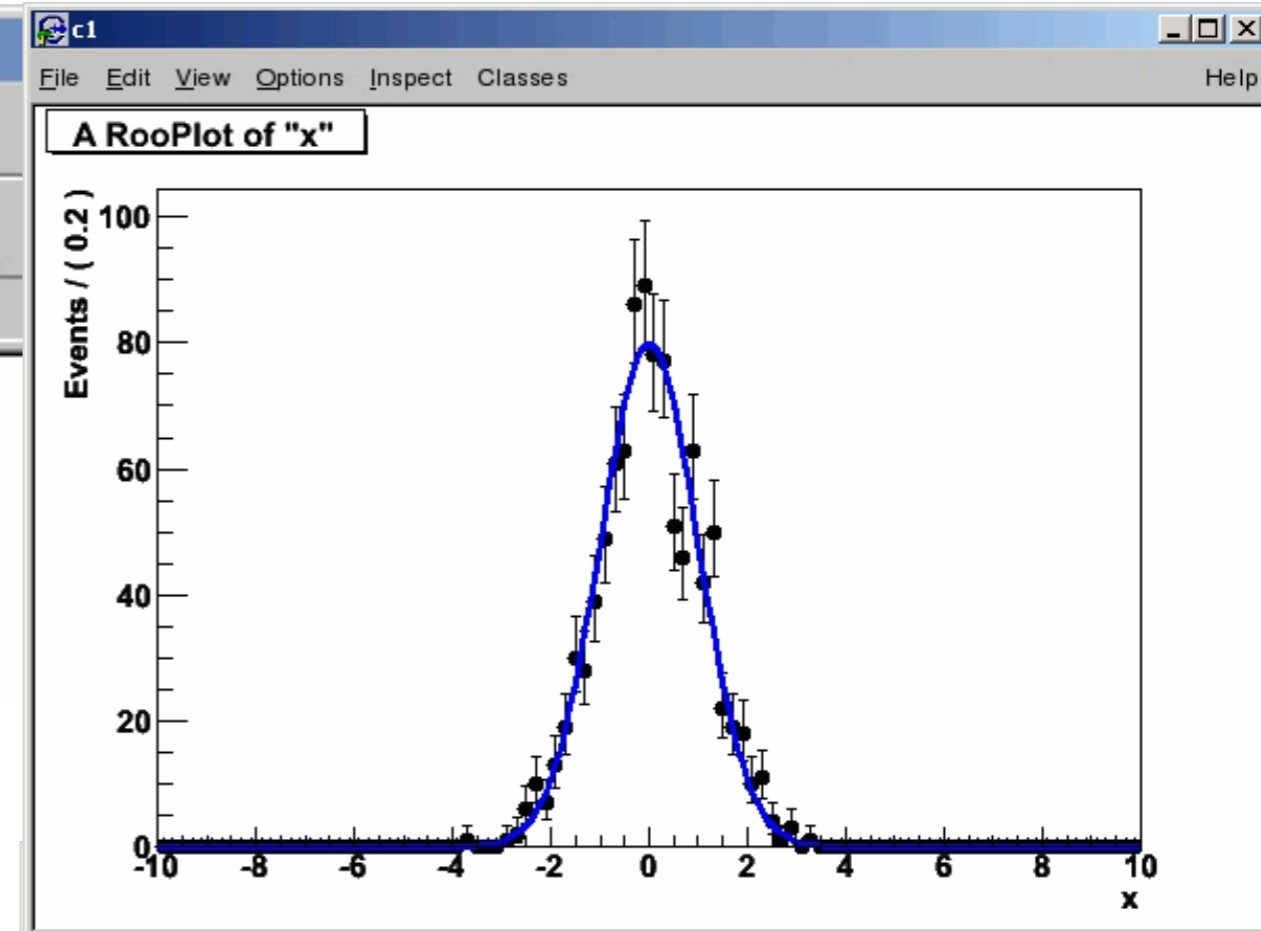
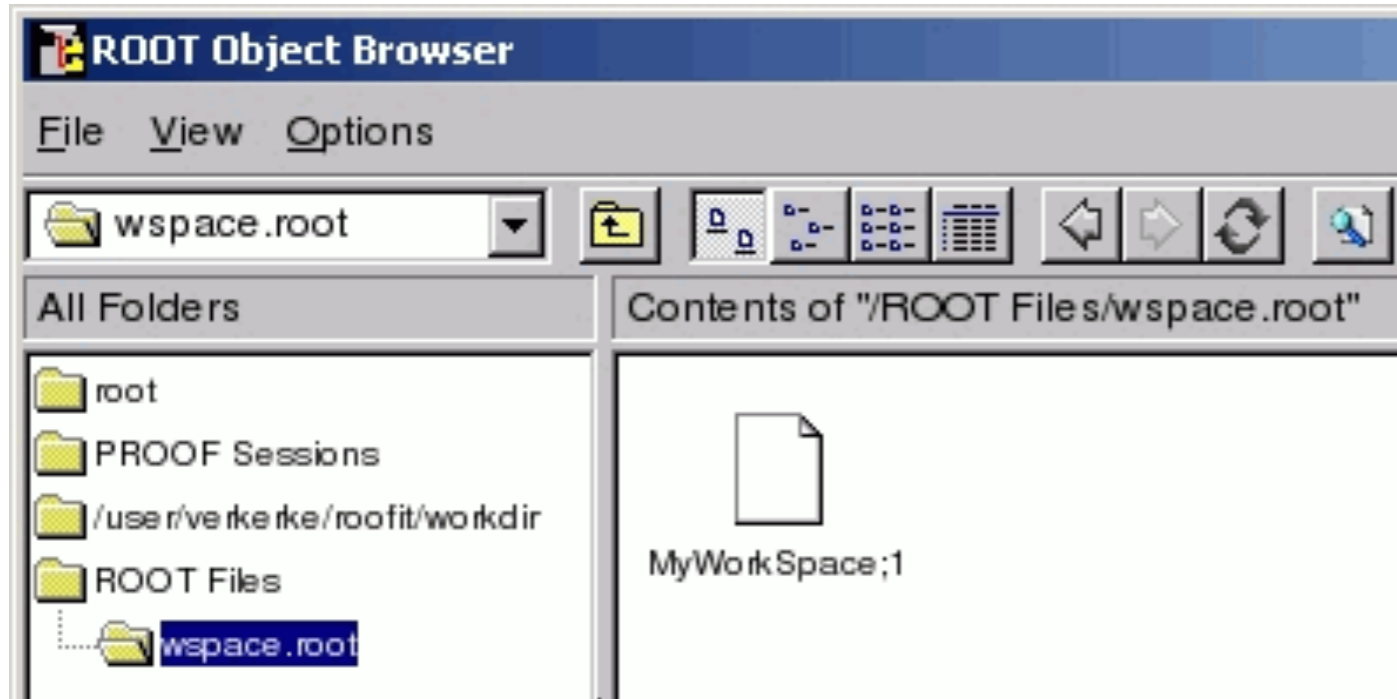
CONCEPTUAL BUILDING BLOCKS



We will use the following mnemonic index conventions:

- $e \in \text{events}$
- $b \in \text{bins}$
- $c \in \text{channels}$
- $s \in \text{samples}$
- $p \in \text{parameters}$

EXAMPLE OF DIGITAL PUBLISHING



RooFit's Workspace now provides the ability to save in a ROOT file the full likelihood model, any priors you might want, and the minimal data necessary to reproduce likelihood function.

Need this for combinations, as p-value is not sufficient information for a proper combination.

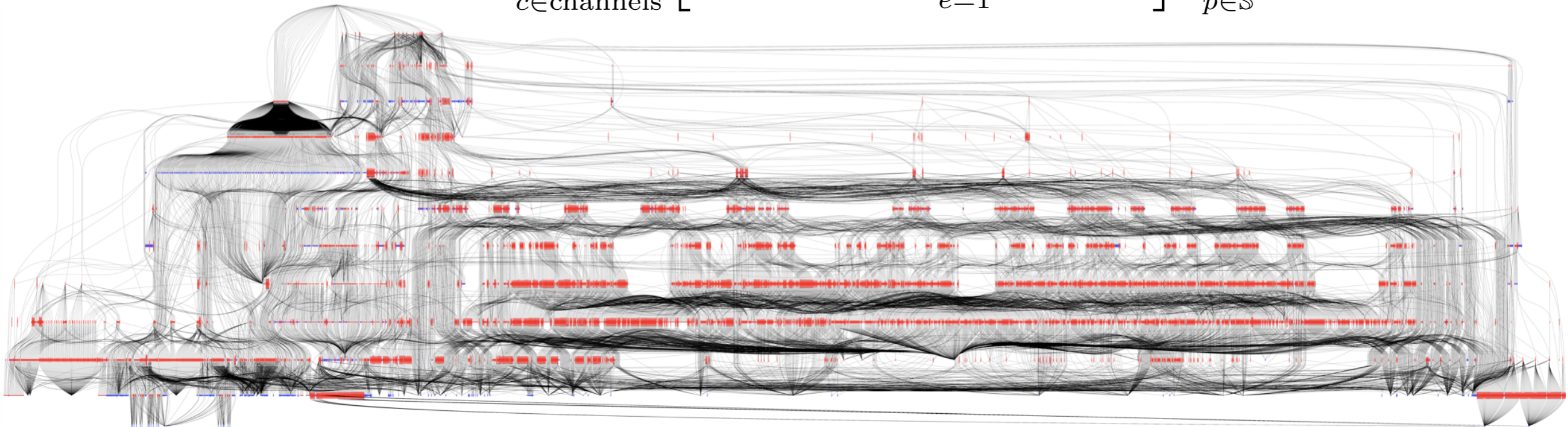
VISUALIZING THE COMBINED MODEL

State of the art: At the time of the discovery, the combined Higgs search included 100 disjoint channels and >500 nuisance parameters

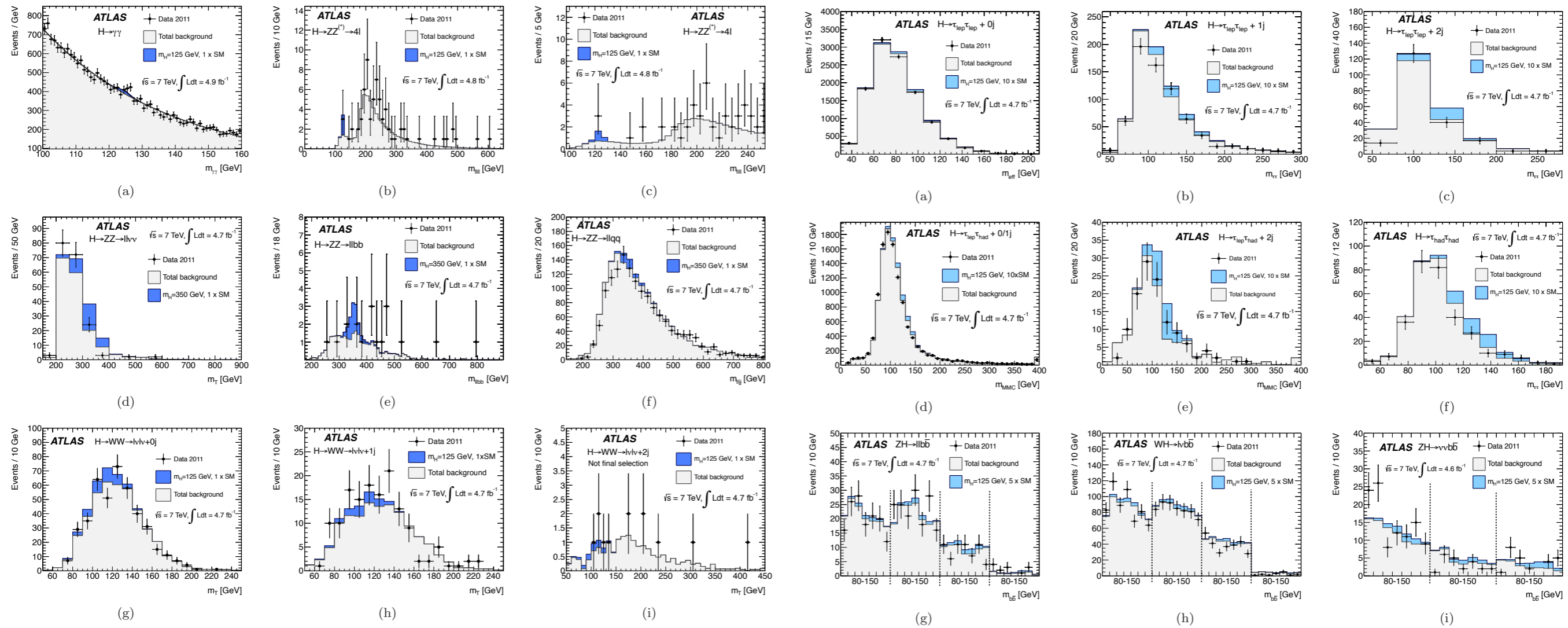
RooFit / RooStats: is the modeling language (C++) which provides technologies for collaborative modeling

- provides technology to publish likelihood functions digitally
- and more, it's the full model so we can also generate pseudo-data

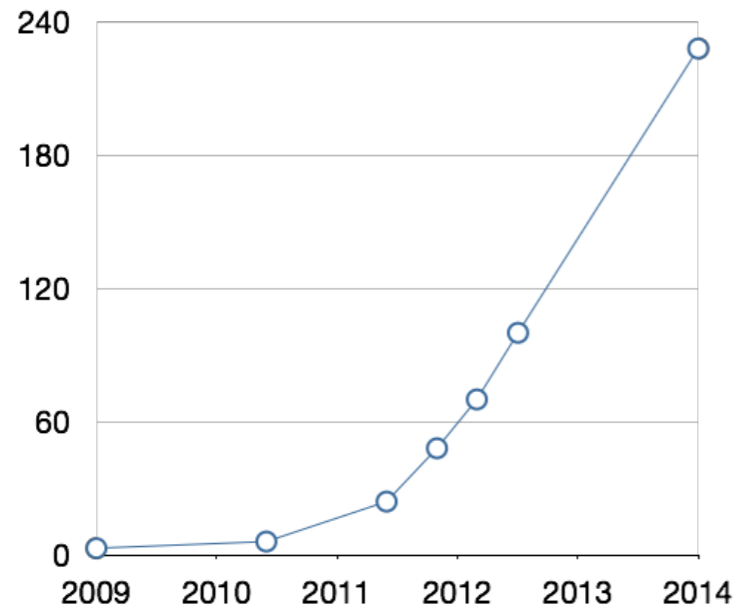
$$\mathbf{f}_{\text{tot}}(\mathcal{D}_{\text{sim}}, \mathcal{G} | \boldsymbol{\alpha}) = \prod_{c \in \text{channels}} \left[\text{Pois}(n_c | \nu_c(\boldsymbol{\alpha})) \prod_{e=1}^{n_c} f_c(x_{ce} | \boldsymbol{\alpha}) \right] \cdot \prod_{p \in \mathbb{S}} f_p(a_p | \alpha_p)$$



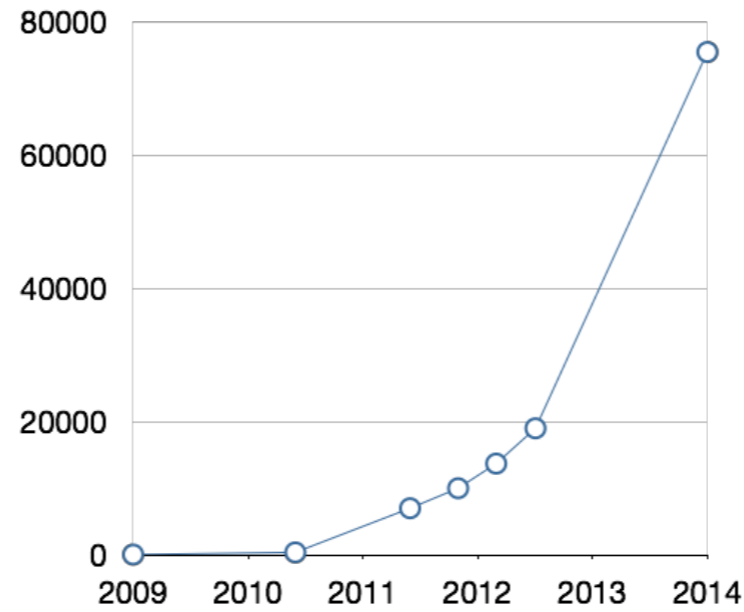
EVOLUTION OF MODEL COMPLEXITY



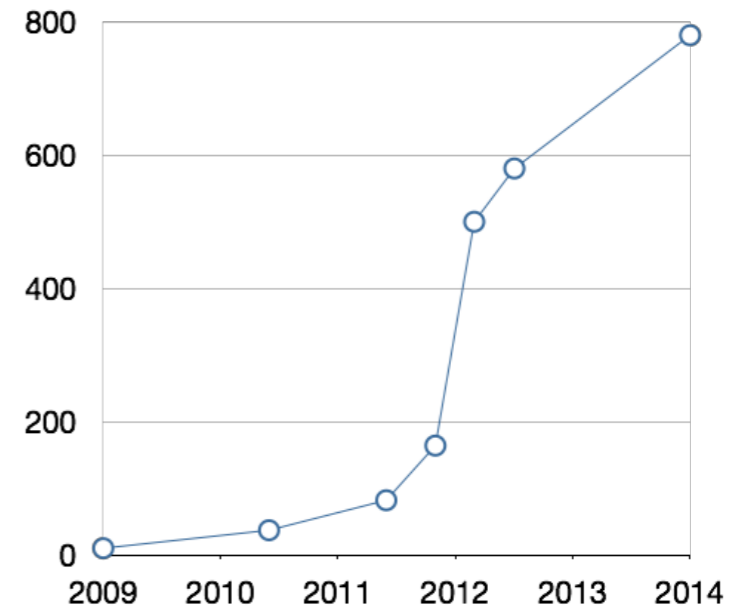
Number of Datasets Combined



Number of Model Components

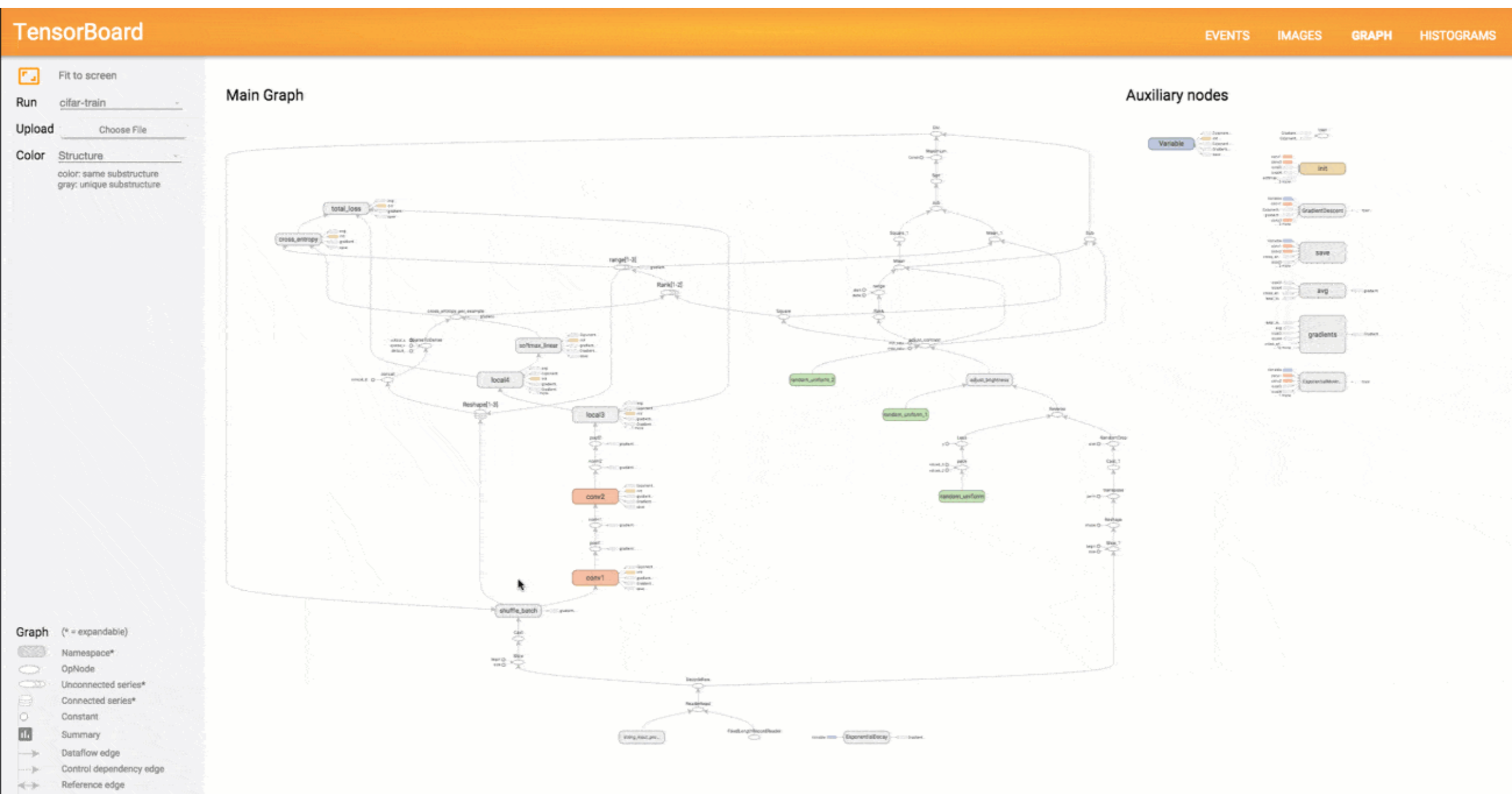


Number of Parameters in Likelihood

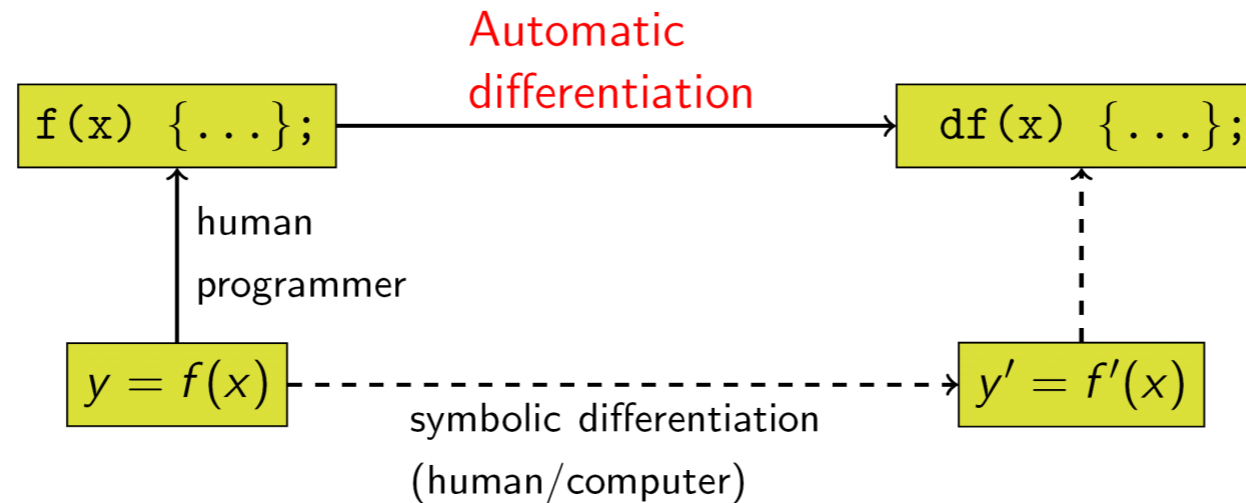


TENSORBOARD

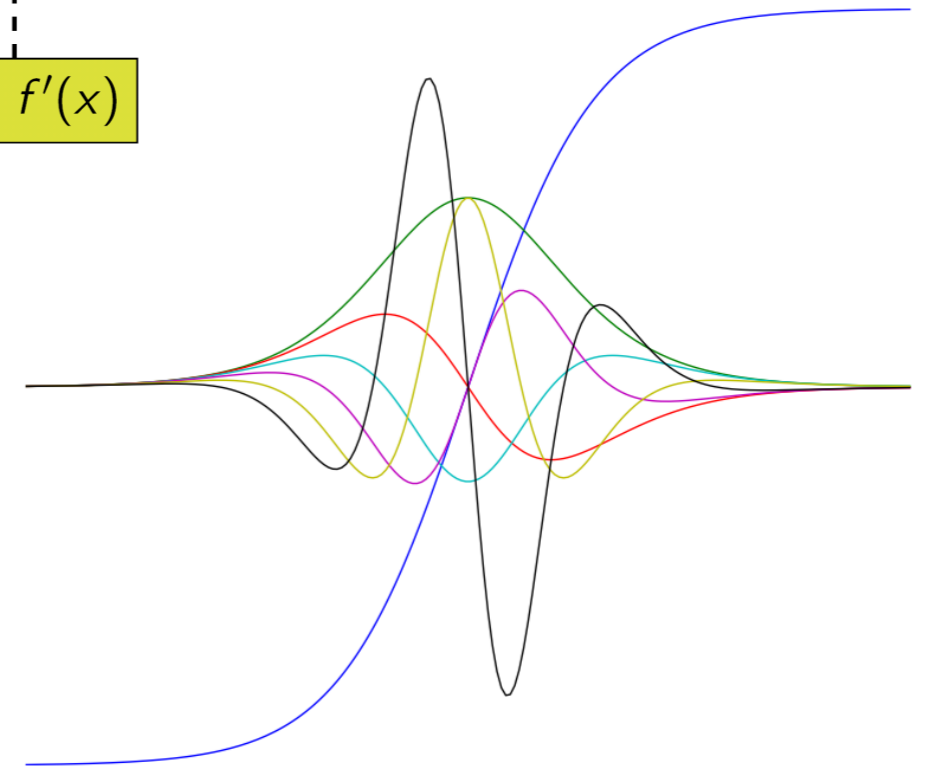
Modern Machine Learning tools like TensorFlow express the model in a similar way as a Directed Acyclic Graph (DAG)



AUTOMATIC DIFFERENTIATION



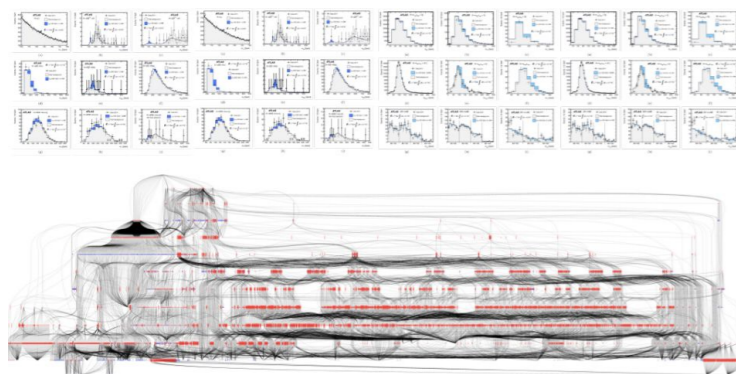
```
>>> import autograd.numpy as np # Thinly-wrapped numpy
>>> from autograd import grad   # The only autograd function you may ever need
>>>
>>> def tanh(x):                # Define a function
...     y = np.exp(-2.0 * x)
...     return (1.0 - y) / (1.0 + y)
...
>>> grad_tanh = grad(tanh)      # Obtain its gradient function
>>> grad_tanh(1.0)              # Evaluate the gradient at x = 1.0
0.41997434161402603
>>> (tanh(1.0001) - tanh(0.9999)) / 0.0002 # Compare to finite differences
0.41997434264973155
```



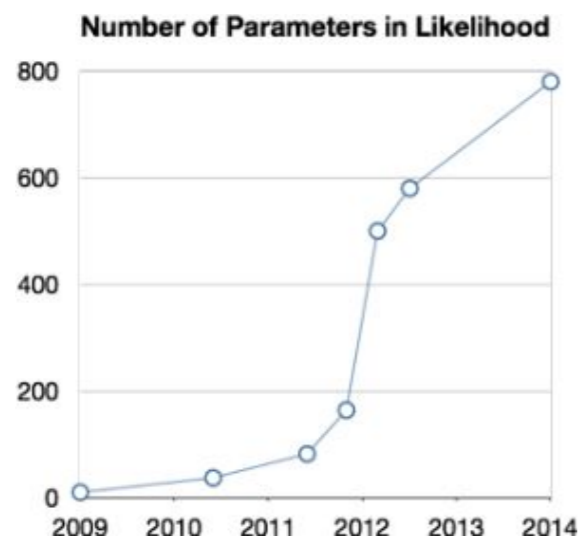
We can continue to differentiate as many times as we like, and use numpy's vectorization of scalar-valued functions across many different input values:

```
>>> from autograd import elementwise_grad as egrad # for functions that vectorize over inputs
>>> import matplotlib.pyplot as plt
>>> x = np.linspace(-7, 7, 200)
>>> plt.plot(x, tanh(x),
...         x, egrad(tanh)(x), # first derivative
...         x, egrad(egrad(tanh))(x), # second derivative
...         x, egrad(egrad(egrad(tanh)))(x), # third derivative
...         x, egrad(egrad(egrad(egrad(tanh)))(x), # fourth derivative
...         x, egrad(egrad(egrad(egrad(egrad(tanh)))(x), # fifth derivative
...         x, egrad(egrad(egrad(egrad(egrad(egrad(tanh)))(x)) # sixth derivative
>>> plt.show()
```

Probabilistic programming frameworks



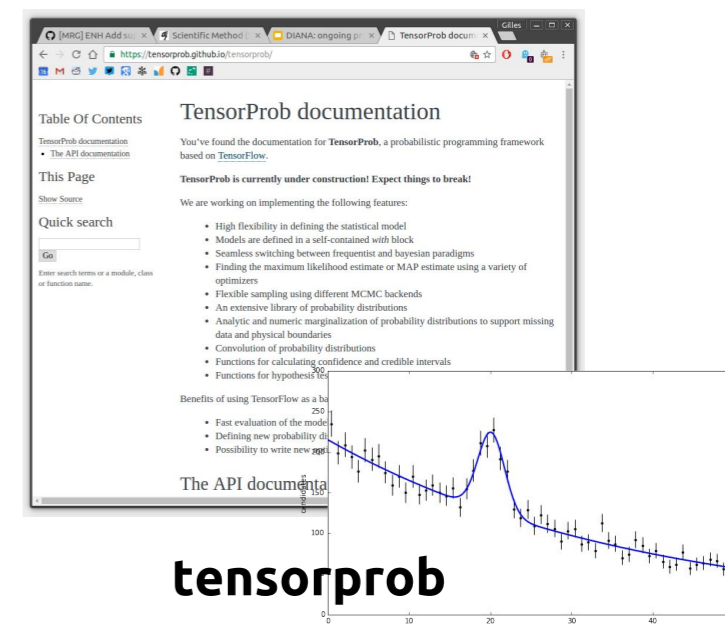
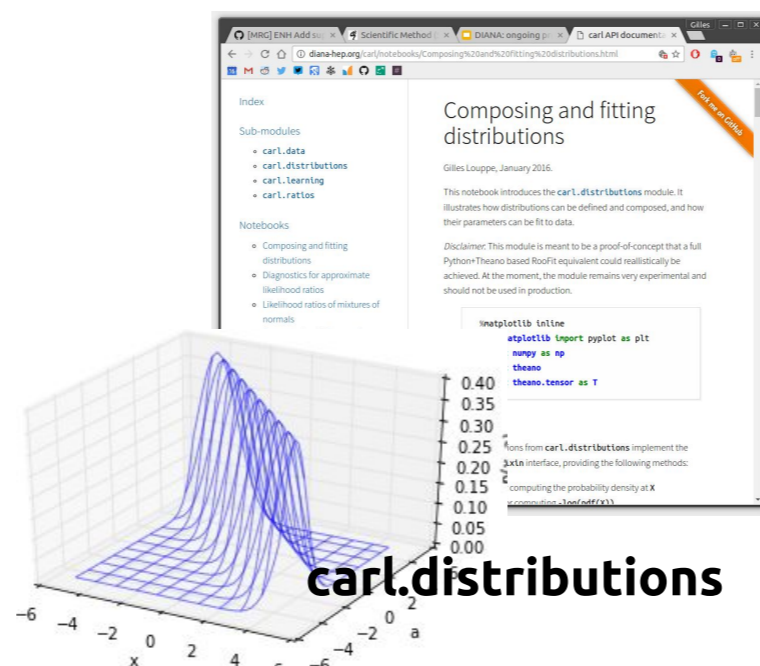
$$f_{\text{tot}}(\mathcal{D}_{\text{sim}}, \mathcal{G}|\alpha) = \prod_{c \in \text{channels}} \left[\text{Pois}(n_c | \nu_c(\alpha)) \prod_{e=1}^{n_c} f_c(x_{ce} | \alpha) \right] \cdot \prod_{p \in \mathcal{S}} f_p(a_p | \alpha_p)$$



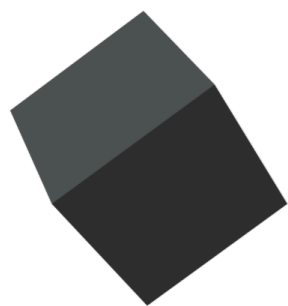
RooFit serves us well, but shows limits in terms of **scalability**.

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.



Edward



A library for probabilistic modeling, inference, and criticism.

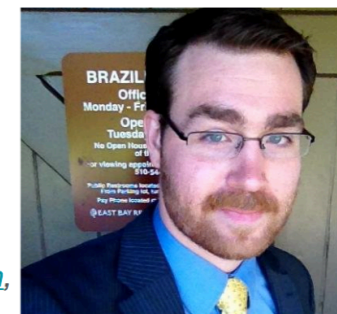
Edward is a Python library for probabilistic modeling, inference, and criticism. It is a testbed for fast experimentation and research with probabilistic models, ranging from classical hierarchical models on small data sets to complex deep probabilistic models on large data sets. Edward fuses three fields: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

It supports **modeling** with



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

Dustin Tran



Matthew Feickert

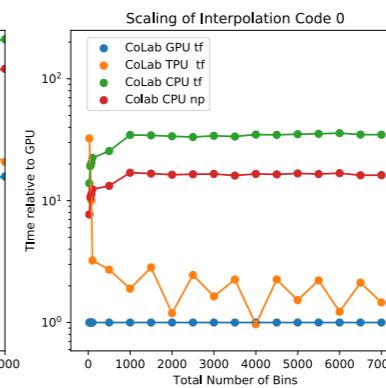
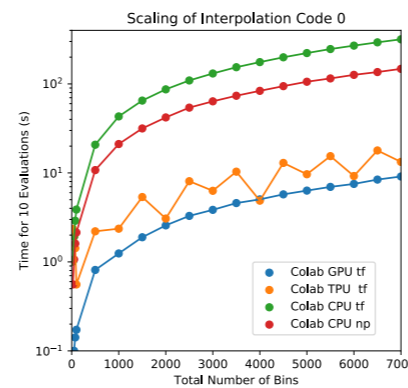
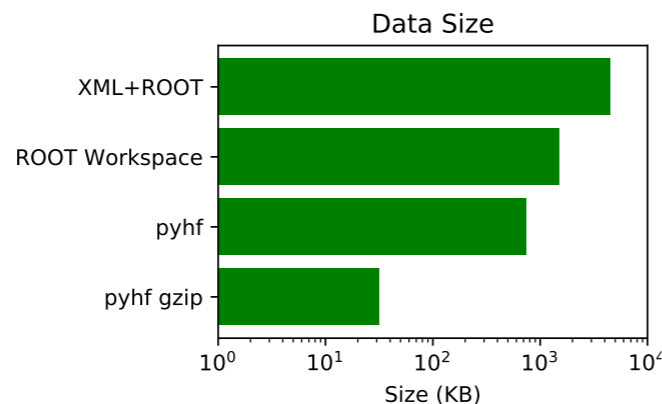
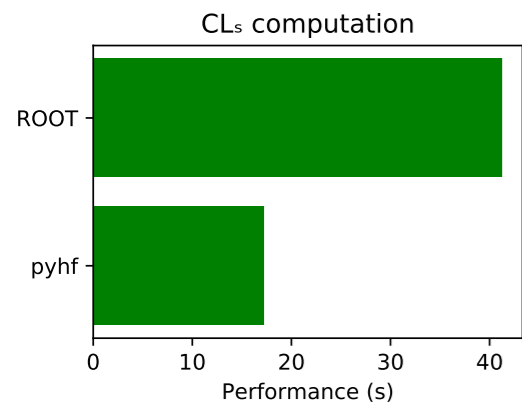
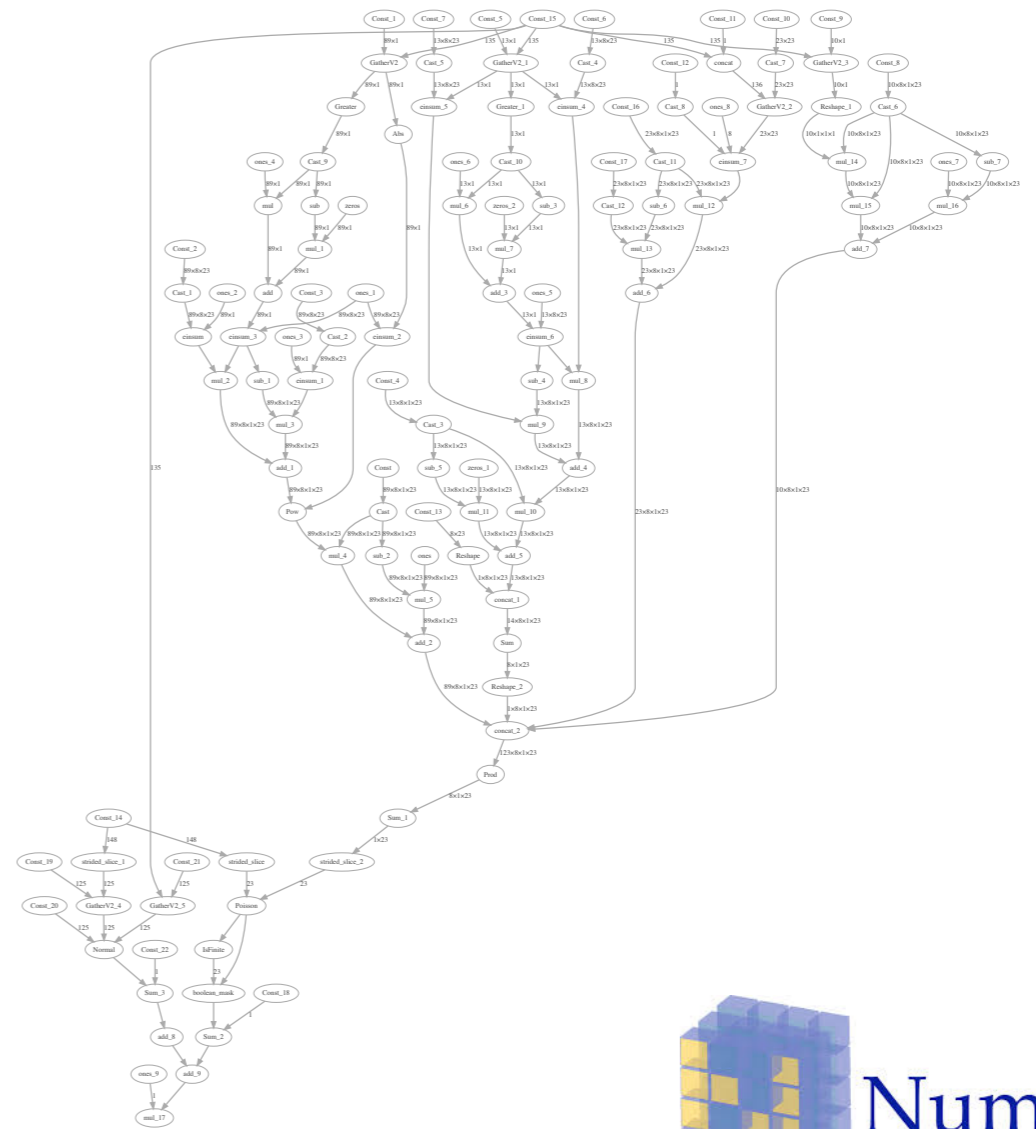
High Energy Physics Ph.D. Candidate
Southern Methodist University
matthew.feickert@cern.ch or mfeickert@smu.edu
GitHub: [matthewfeickert](https://github.com/matthewfeickert) @HEPfeickert

pyhf: modern implementation of HEP likelihood computations

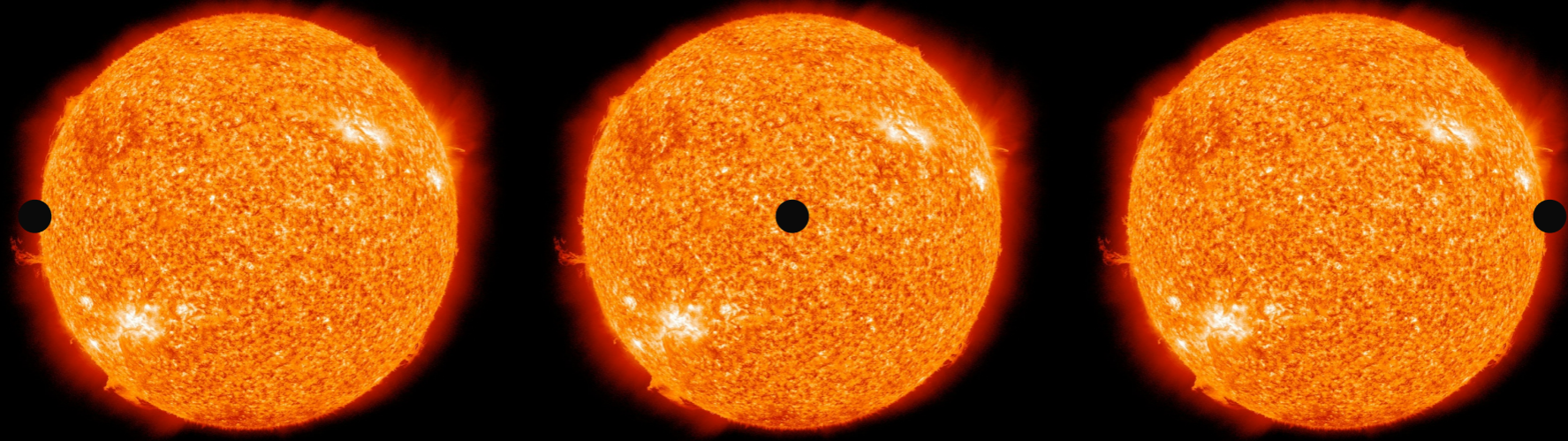
- **easy-to-publish likelihoods**
- **make likelihoods fast to compute**

Likelihoods as computational graphs of array computations

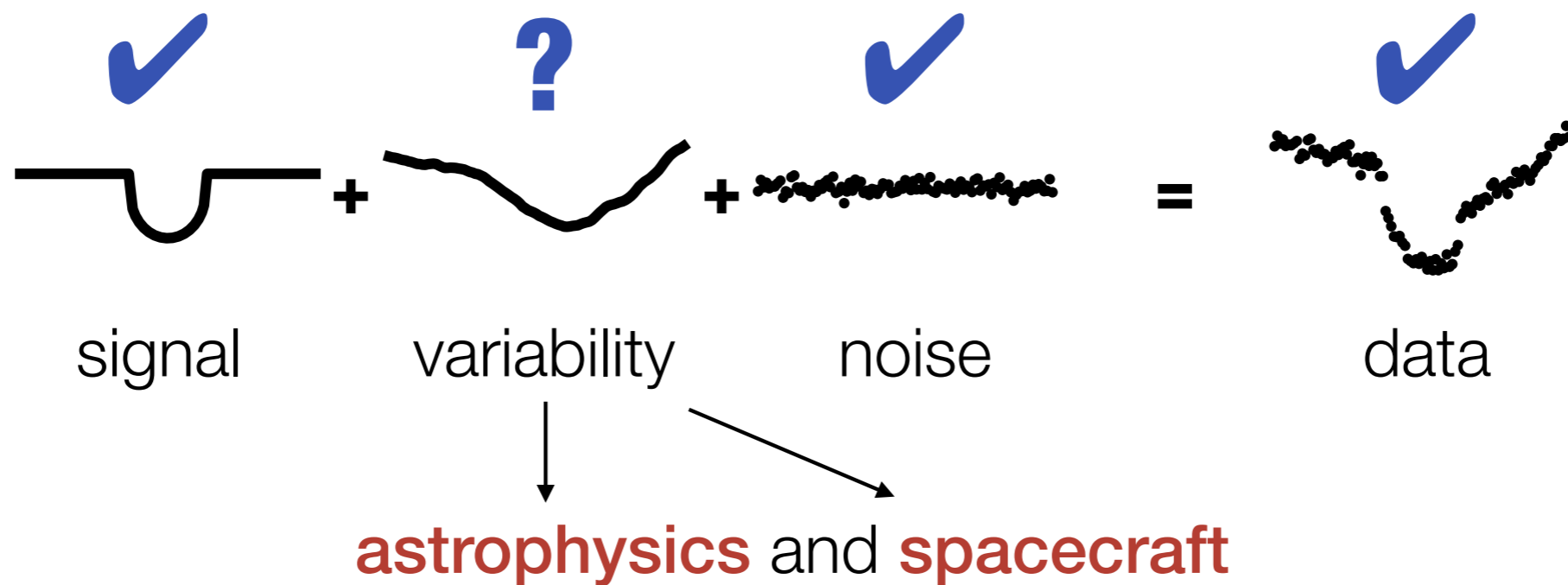
- **automatic gradients**
- **compute on specialized hardware (GPU / TPU)**
- **graph structure allows distribution across machines — stat. combinations**



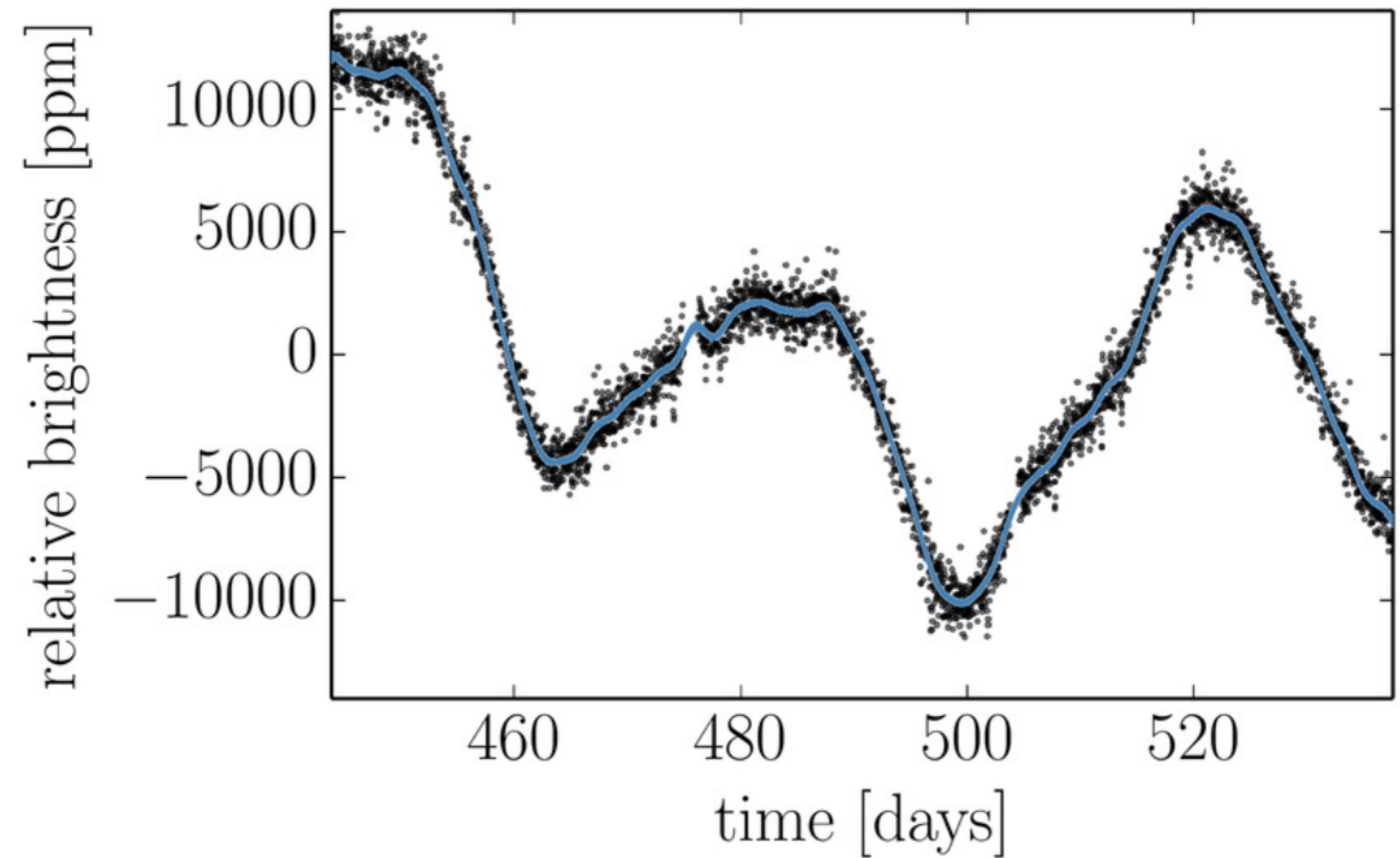
Gaussian Processes



The **anatomy** of a **transit** observation



AN EXOPLANET EXAMPLE



the data are drawn from one

HUGE*

Gaussian

* the dimension is the number of data points.

GAUSSIAN PROCESSES

$$\log p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\sigma}, \boldsymbol{\theta}, \boldsymbol{\alpha}) = -\frac{1}{2} [\mathbf{y} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})]^{\mathrm{T}} K_{\boldsymbol{\alpha}}(\mathbf{x}, \boldsymbol{\sigma})^{-1} [\mathbf{y} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})] \\ - \frac{1}{2} \log \det K_{\boldsymbol{\alpha}}(\mathbf{x}, \boldsymbol{\sigma}) - \frac{N}{2} \log 2 \pi$$

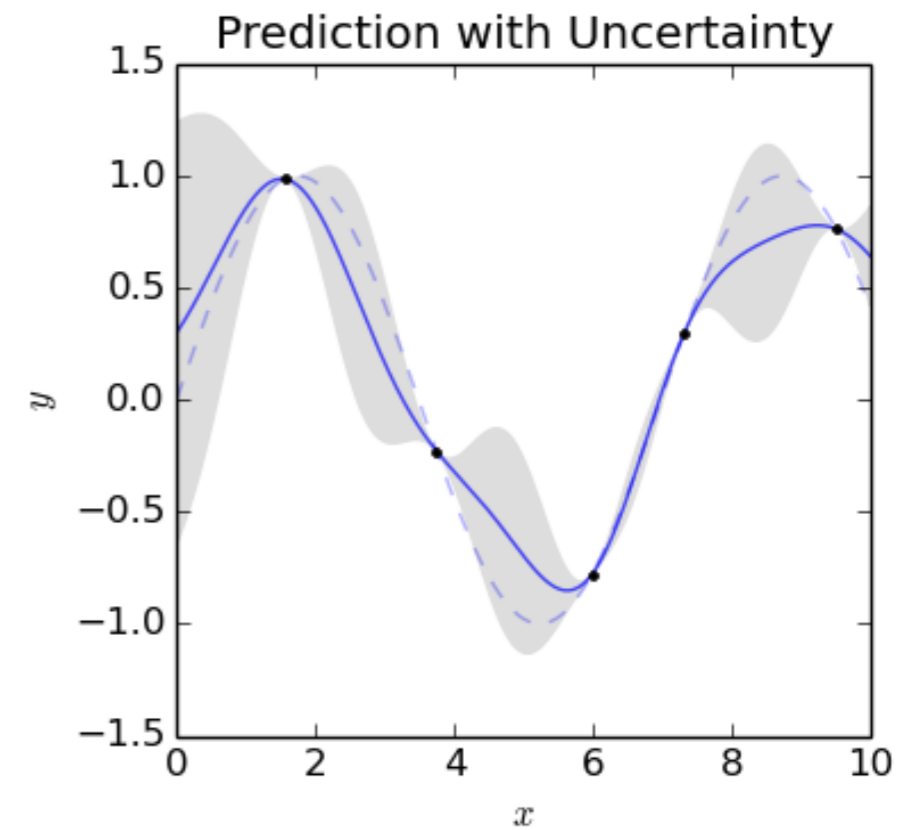
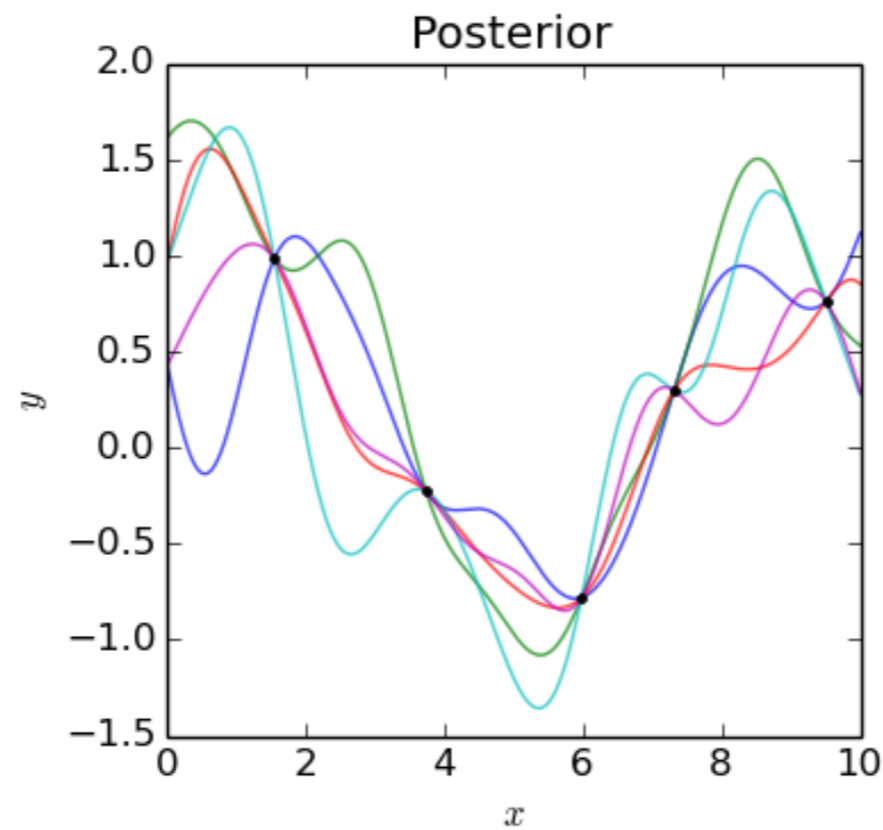
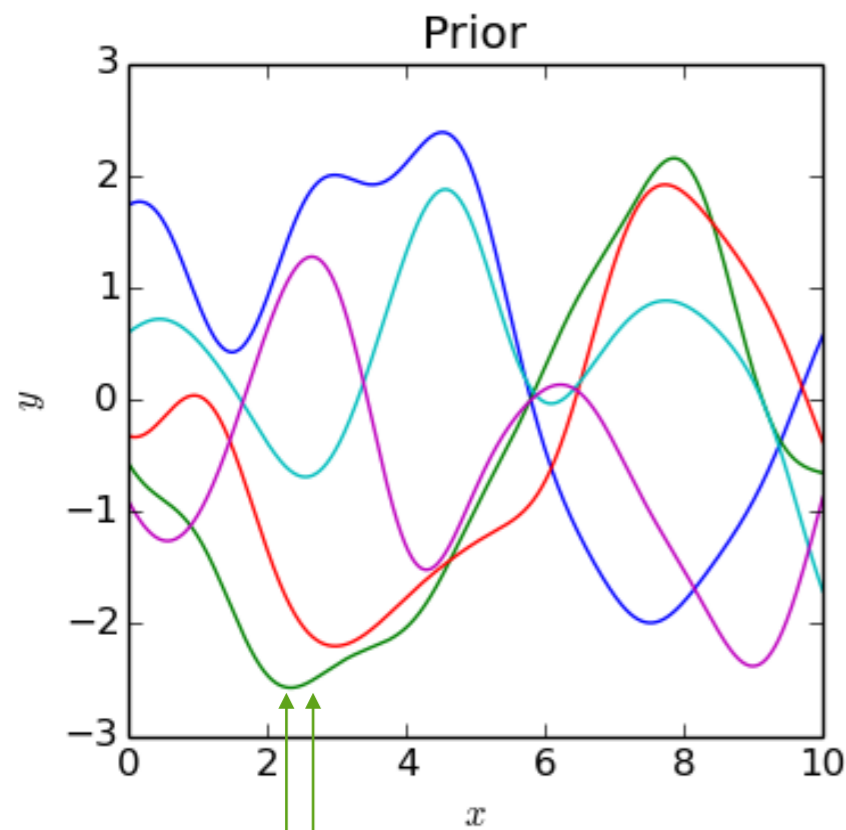
where

$$[K_{\boldsymbol{\alpha}}(\mathbf{x}, \boldsymbol{\sigma})]_{ij} = \sigma_i^2 \delta_{ij} + \underbrace{k_{\boldsymbol{\alpha}}(x_i, x_j)}$$

kernel function

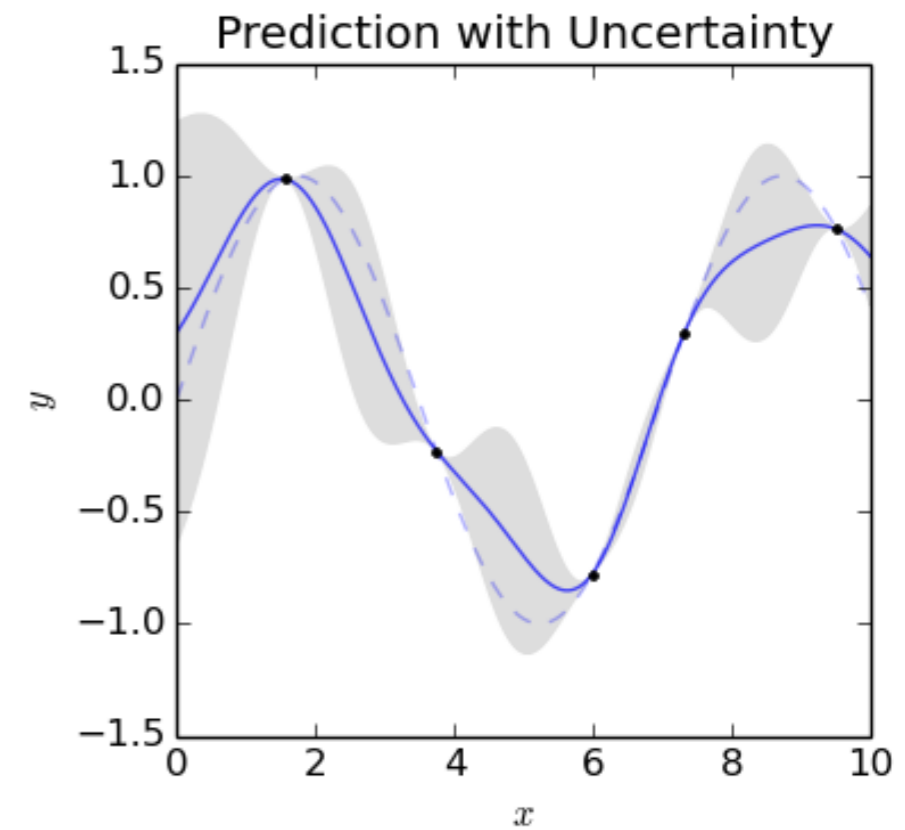
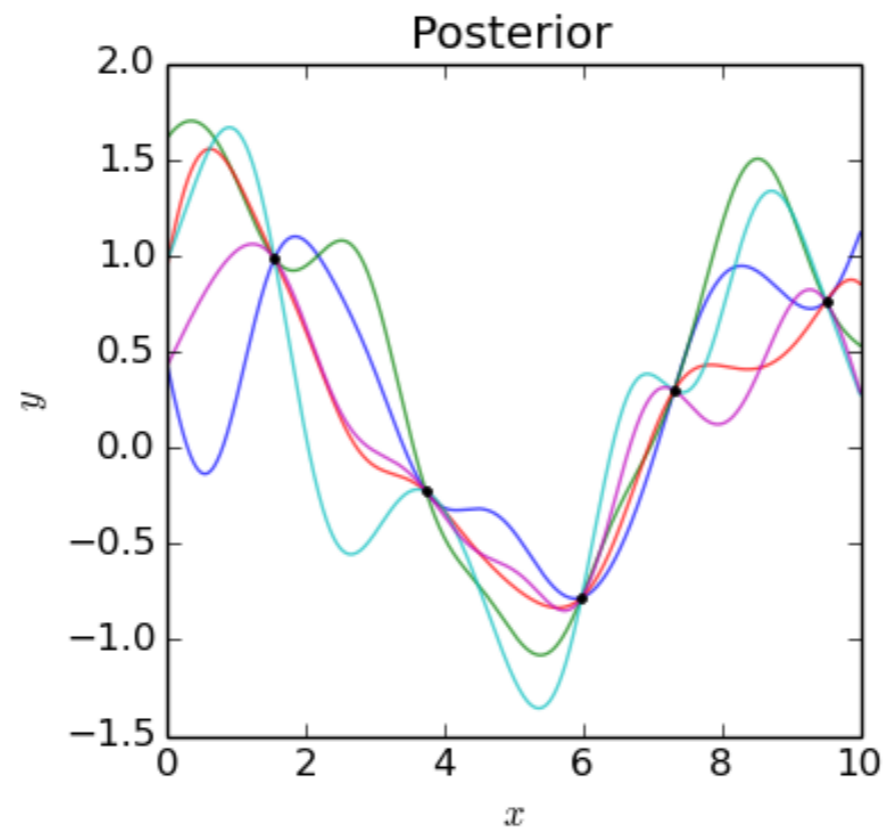
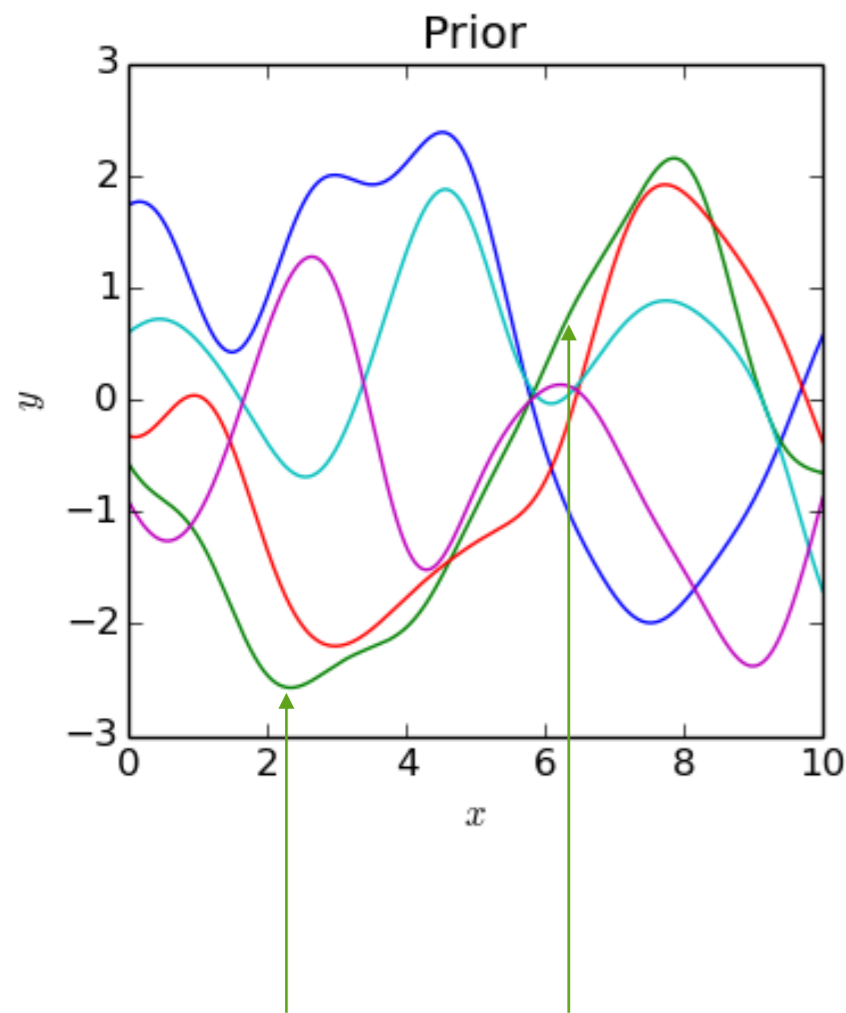
(where the magic happens)

https://en.wikipedia.org/wiki/Gaussian_process



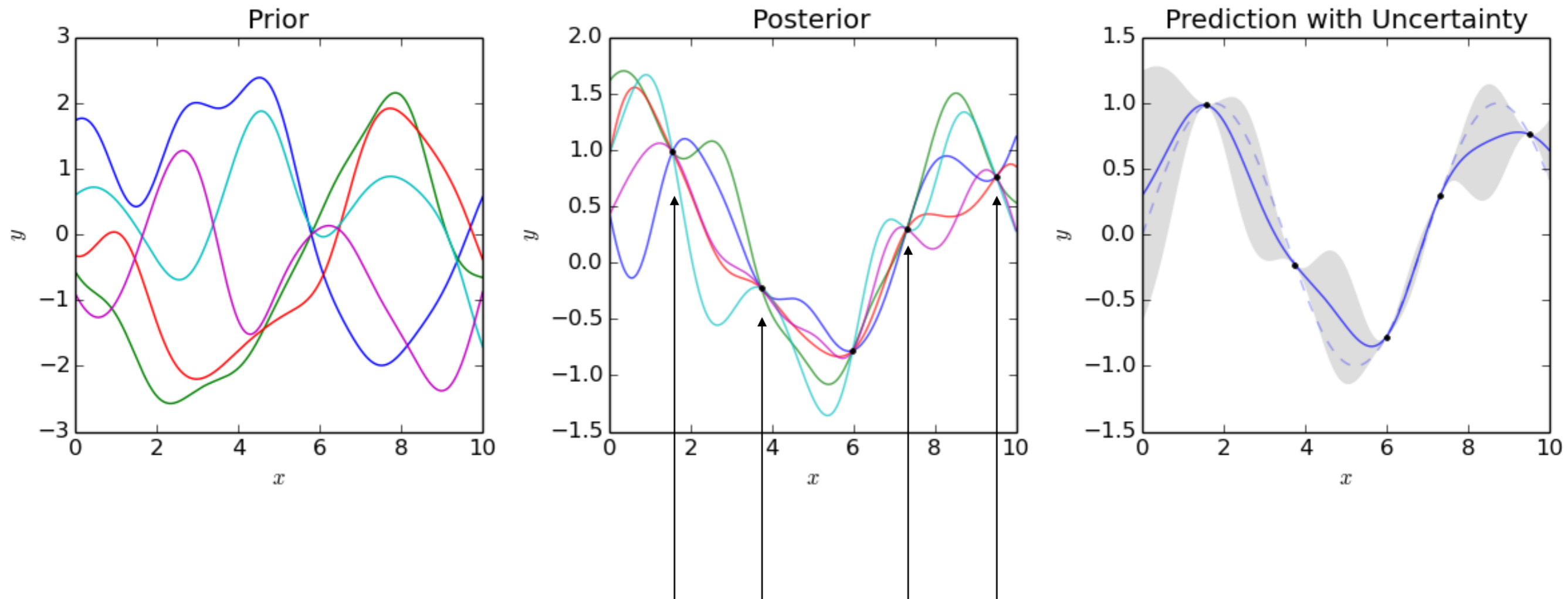
function values $y(x_1)$ &
 $y(x_2)$ are highly
correlated for close by
points

https://en.wikipedia.org/wiki/Gaussian_process



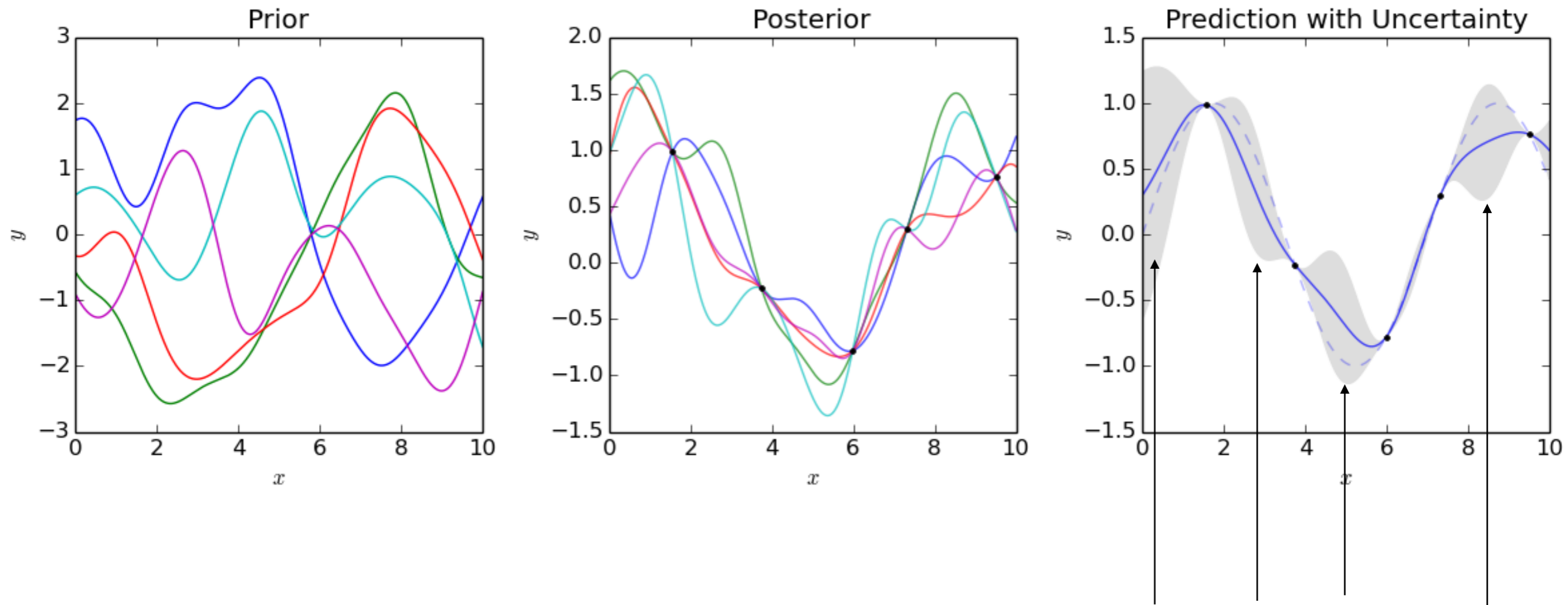
function values $y(x_1)$ &
 $y(x_2)$ are \sim uncorrelated
for far-away points

https://en.wikipedia.org/wiki/Gaussian_process



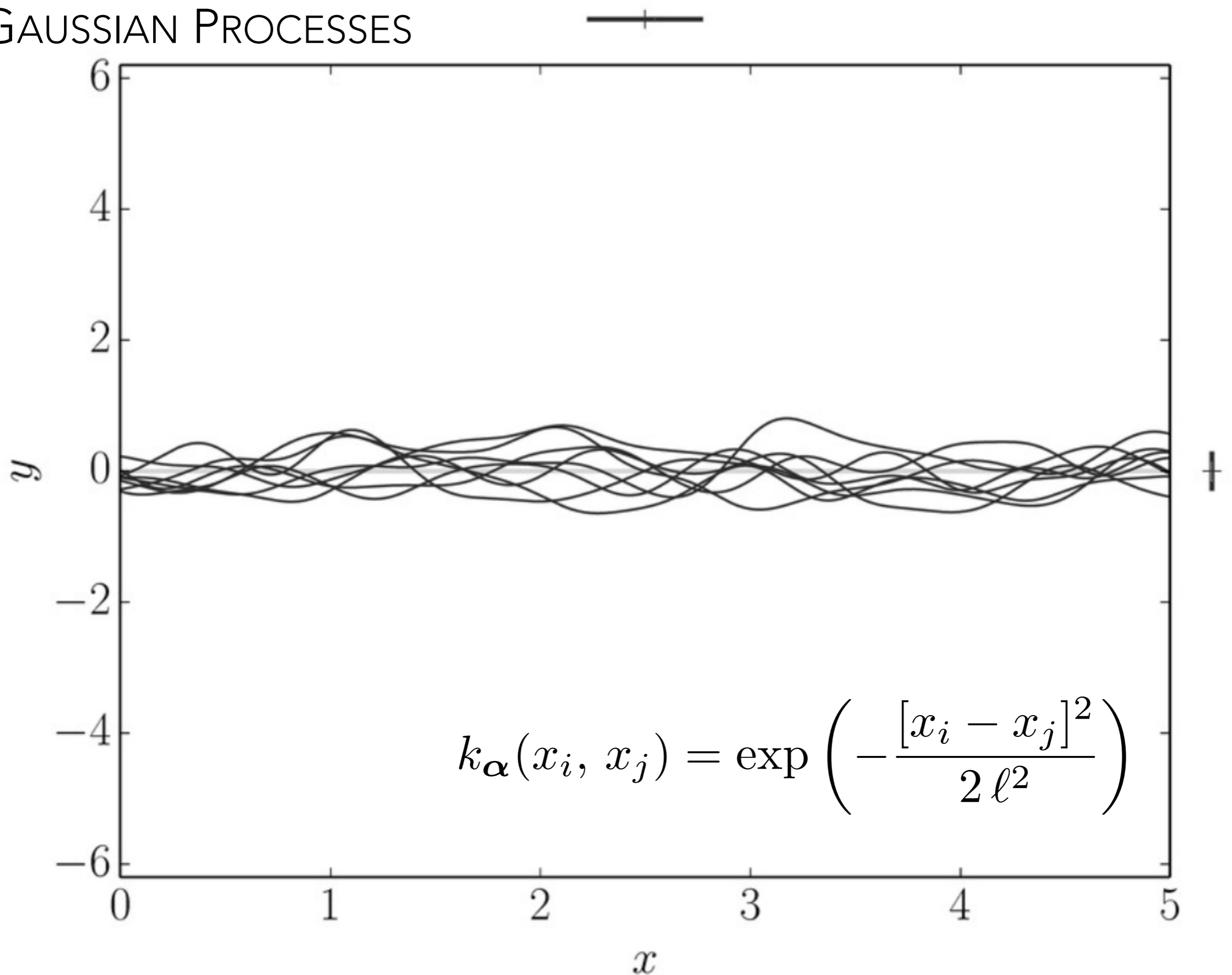
if value of function is known at some
points, the gaussian process must go
through them
(can include uncertainty in function value)

https://en.wikipedia.org/wiki/Gaussian_process



GP can interpolate and extrapolate.
Uncertainty on interpolation and
extrapolation grows as you move further
from known points

GAUSSIAN PROCESSES



LEARN MORE

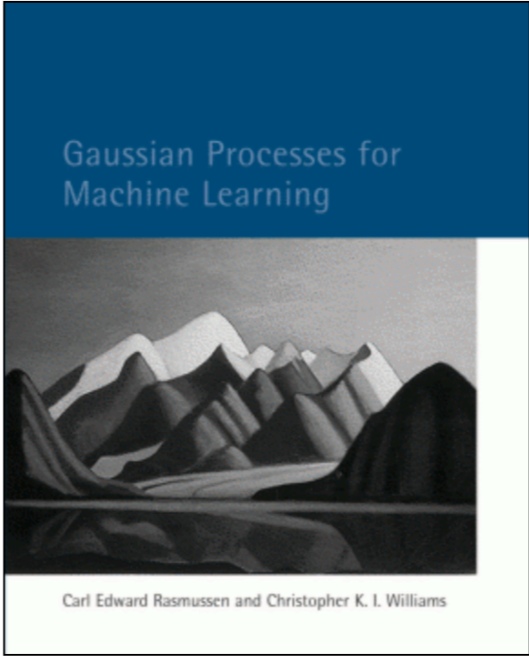
gaussianprocess.org

DiscoveryLinks ▾ Higgs ▾ RooStats ▾ ALEPH ▾ Apple ▾ News ▾ Life Stuff ▾ Wikipedia, ATLAS inSpire dblp Theory&Practice ▾ HCG ▾ Evernote Web Equation job CERN inSpire >>

G N G G G De Division of P... for Jet Physi... isaachenrion... learning-qcd... Submit Test I... cweniger/sw... Statistics For... Gaussian Pro... +

Gaussian Processes for Machine Learning

Carl Edward Rasmussen and Christopher K. I. Williams
The MIT Press, 2006. ISBN 0-262-18253-X.



[[Contents](#) | [Software](#) | [Datasets](#) | [Errata](#) | [Authors](#) | [Order](#)]

Gaussian processes (GPs) provide a principled, practical, probabilistic approach to learning in kernel machines. GPs have received increased attention in the machine-learning community over the past decade, and this book provides a long-needed systematic and unified treatment of theoretical and practical aspects of GPs in machine learning. The treatment is comprehensive and self-contained, targeted at researchers and students in machine learning and applied statistics.

The book deals with the supervised-learning problem for both regression and classification, and includes detailed algorithms. A wide variety of covariance (kernel) functions are presented and their properties discussed. Model selection is discussed both from a Bayesian and a classical perspective. Many connections to other well-known techniques from machine learning and statistics are discussed, including support-vector machines, neural networks, splines, regularization networks, relevance vector machines and others. Theoretical issues including learning curves and the PAC-Bayesian framework are treated, and several approximation methods for learning with large datasets are discussed. The book contains illustrative examples and exercises, and code and datasets are available on the Web. Appendixes provide mathematical background and a discussion of Gaussian Markov processes.

The book is available for [download](#) in electronic format.

<http://www.gaussianprocess.org/gpml/>

Parametrized Function
vs.
Gaussian Process

[Information](#)[References \(44\)](#)[Citations \(0\)](#)[Files](#)[Plots](#)

Modeling Smooth Backgrounds and Generic Localized Signals with Gaussian Processes

Meghan Frate, Kyle Cranmer, Saarik Kalia, Alexander Vandenberg-Rodes, Daniel Whiteson

Sep 17, 2017 - 14 pages

e-Print: [arXiv:1709.05681](https://arxiv.org/abs/1709.05681) [physics.data-an] | [PDF](#)

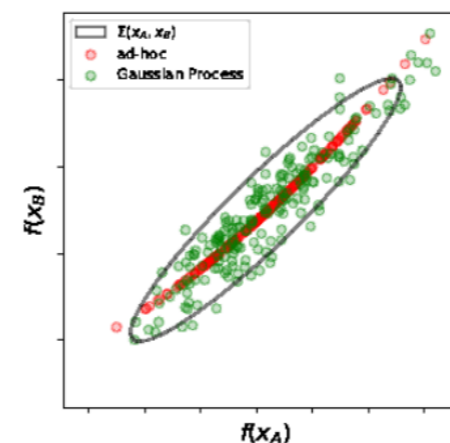
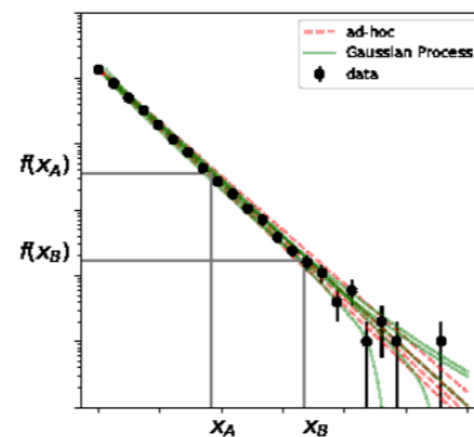
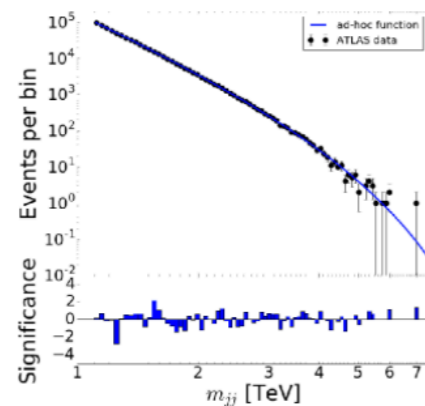
Abstract (arXiv)

We describe a procedure for constructing a model of a smooth data spectrum using Gaussian processes rather than the historical parametric description. This approach considers a fuller space of possible functions, is robust at increasing luminosity, and allows us to incorporate our understanding of the underlying physics. We demonstrate the application of this approach to modeling the background to searches for dijet resonances at the Large Hadron Collider and describe how the approach can be used in the search for generic localized signals.

Note: *Temporary entry*

Note: 14 pages, 16 figures

Keyword(s): INSPIRE: [background](#) | [CERN LHC Coll](#) | [dijet](#) | [resonance](#) | [data analysis method](#) | [Gauss model](#) | [statistics](#) | [statistical analysis](#)



[Show more plots](#)

Record added 2017-09-19, last modified 2017-10-07

PARAMETRIC FUNCTION VS. GP

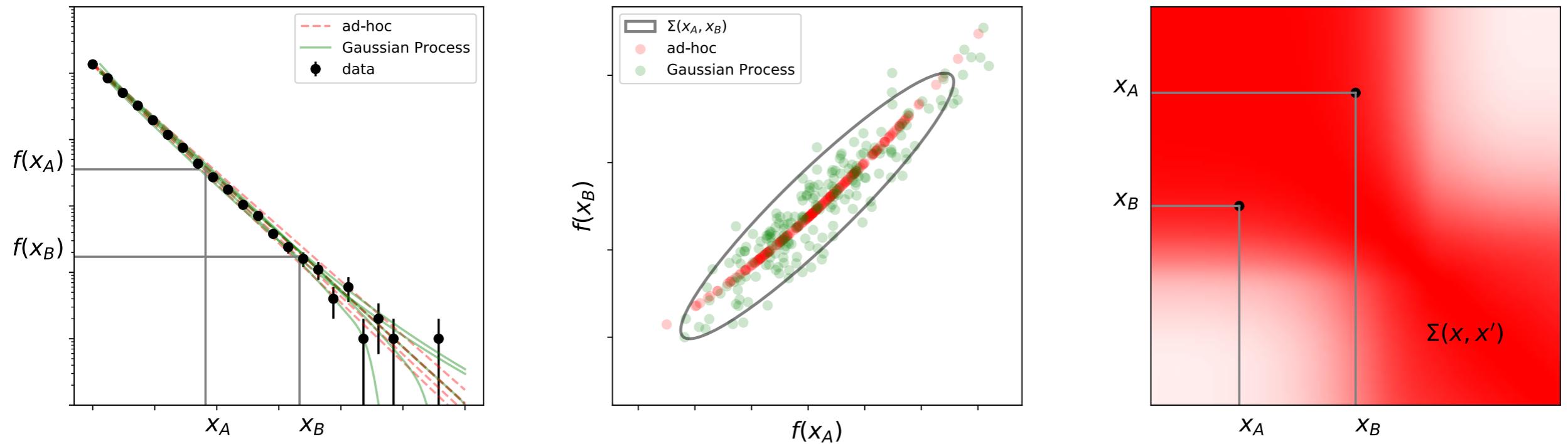


FIG. 2: Schematic of the relationship between an ad-hoc function and the GP. An example toy dataset is shown (left) with samples from the posterior for an ad-hoc 1-parameter function (red) and a GP (green). Each posterior sample is an entire curve $f(x)$, which corresponds to a particular point in the (center) plane of $f(x_A)$ vs. $f(x_B)$. The red dots for the ad-hoc 1-parameter function trace out a 1-dimensional curve, which reveals how the function is overly-rigid. In contrast, the green dots from the GP relax the assumptions and fill a correlated multivariate Gaussian (with covariance indicated by the black ellipse). The covariance kernel $\Sigma(x, x')$ for the GP is shown (right) with $\Sigma(x_A, x_B)$ corresponding to the black ellipse of the center panel.

MOAR DATA!

GP fits the background well, and continues to as we add more data. Parametric function no longer fits well

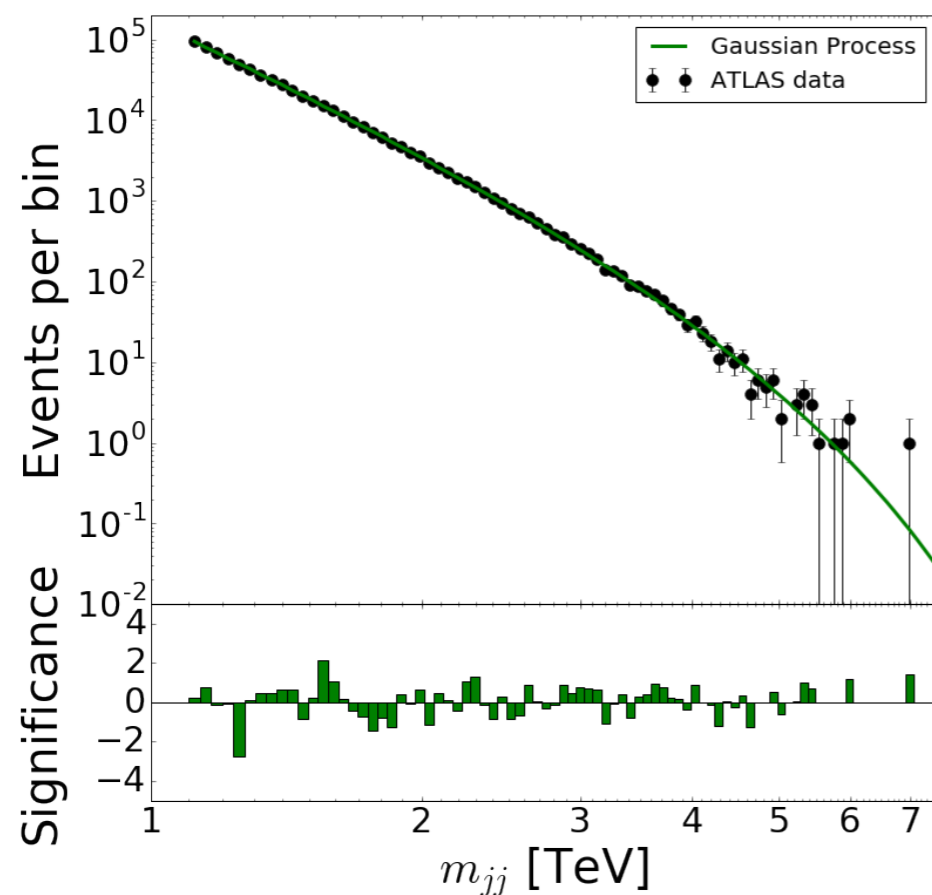
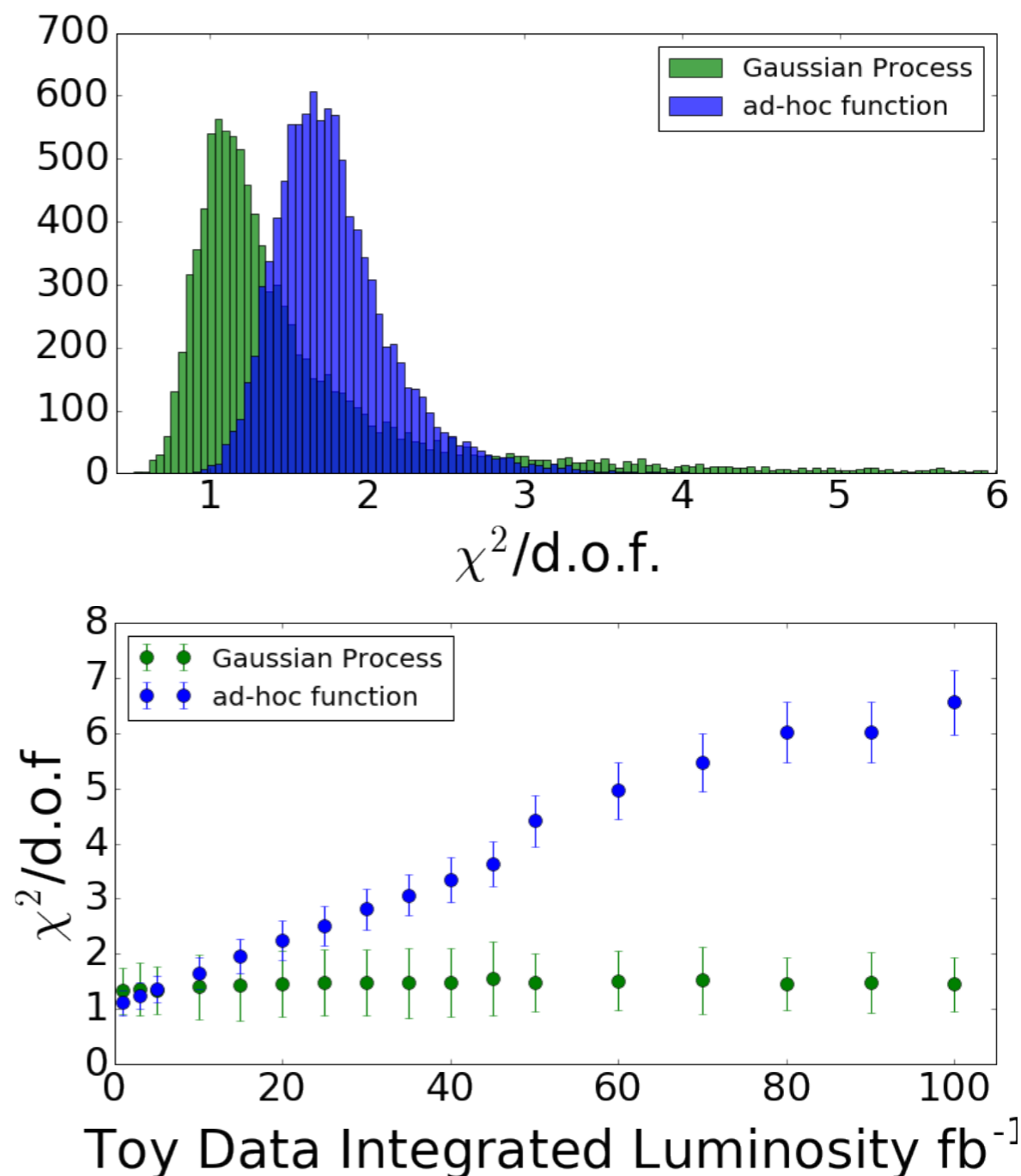


FIG. 5: Invariant mass of dijet pairs reported by ATLAS [15] in proton-proton collisions at $\sqrt{s} = 13$ TeV with integrated luminosity of 3.6 fb^{-1} . The green line shows the resulting Gaussian process background model. The bottom pane shows the significance of the residual between the data and the GP model.

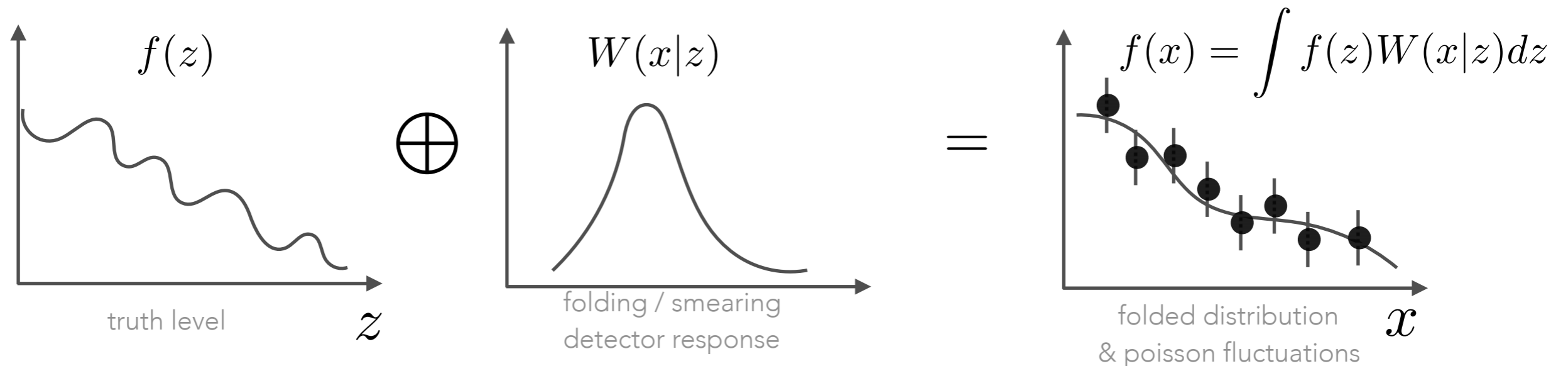


Physically Motivated Kernels

CONNECTION TO UNFOLDING

If the truth level distribution $f(z)$ is a Gaussian Process with kernel $\Sigma(z, z')$, then the reconstructed distribution $f(x)$ is also a Gaussian process with $\Sigma(x, x')$

$$\Sigma(x, x') = \int \int dz dz' \Sigma(z, z') W(x, z) W(x', z') \quad (8)$$



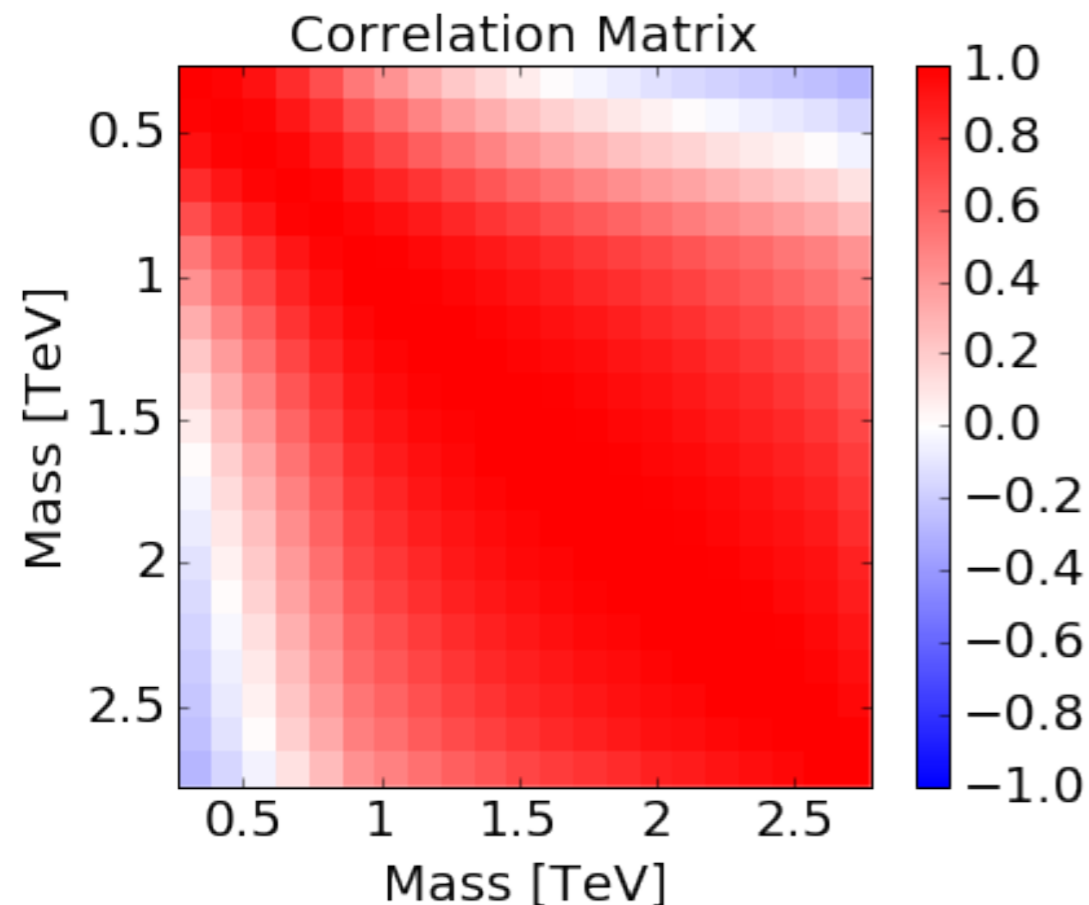
If we are making predictions with Monte Carlo, truth level distribution $f(z)$ is usually known exactly.

To think of $f(z)$ as a Gaussian Process, we need some notion of uncertainty (eg. parton density functions, higher-order corrections, renormalization/factorization scales)

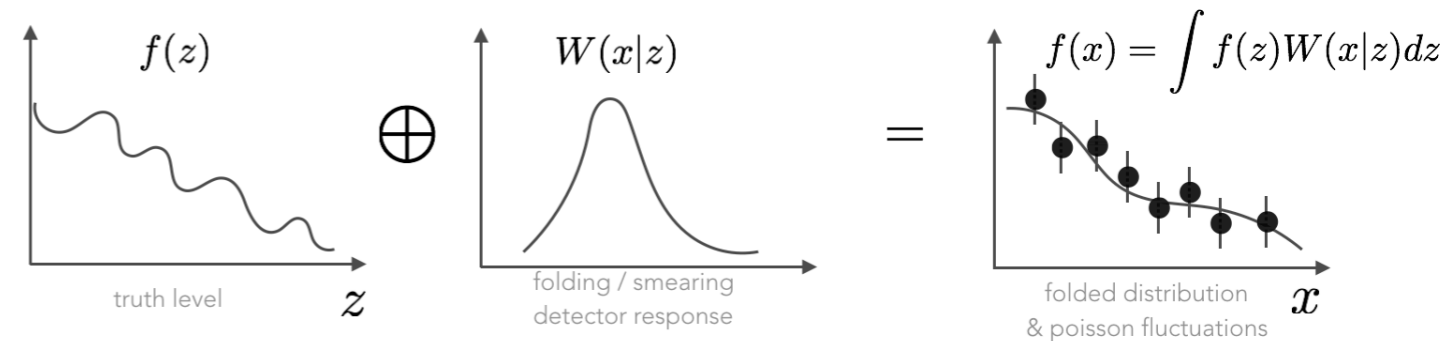
In unfolding, we often don't want to make assumptions about $f(z)$... it could be anything. But regularization in unfolding is equivalent to choosing a kernel for $f(z)$.

Even in extreme case where we assume no smoothness in $f(z)$, $f(x)$ has to be smooth due to detector resolution.

EXAMPLE: PDF UNCERTAINTIES



$$\Sigma(x, x') = \int \int dz dz' \Sigma(z, z') W(x, z) W(x', z') \quad (8)$$

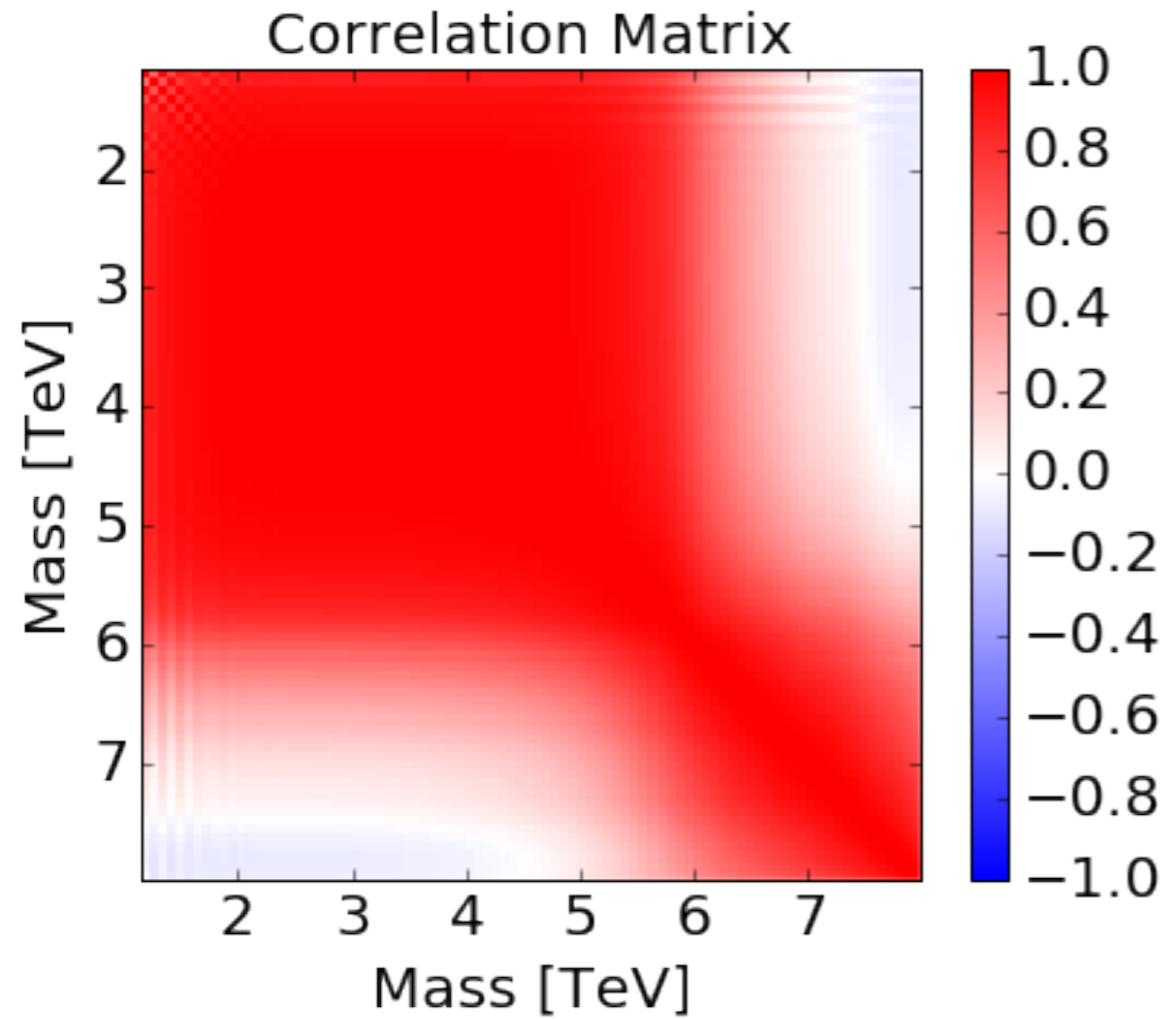


Here we focused on truth level distribution $f(z)$.

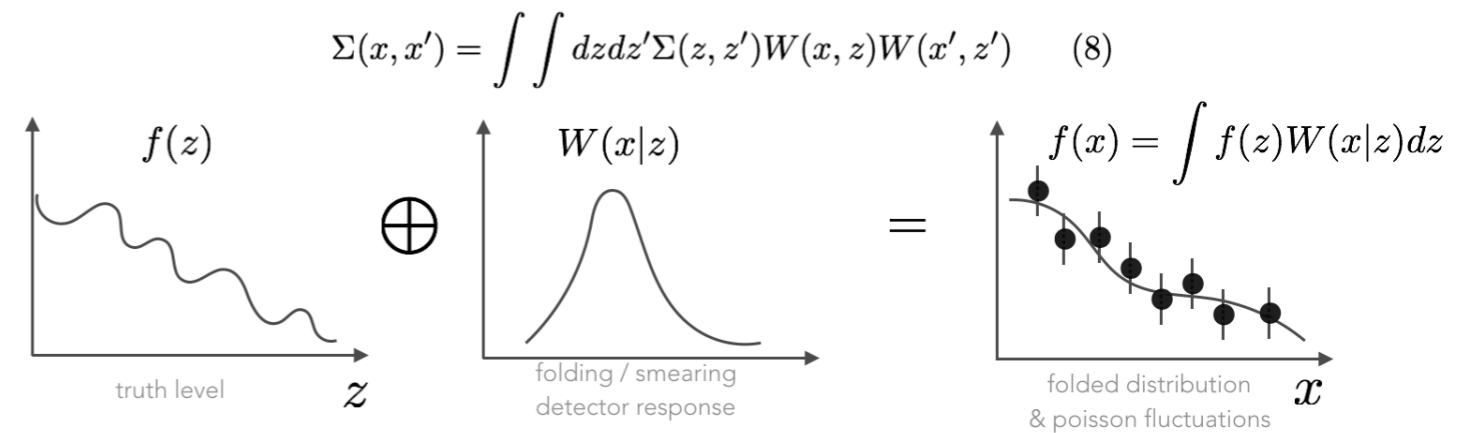
Used dijet spectrum predicted at NLO with POWHEG-BOX and look into PDF uncertainties from NNPDF3

This is the PDF uncertainty in the truth distribution expressed as a Gaussian Process Kernel.

EXAMPLE: JET ENERGY SCALE



Take for example, the jet energy scale (JES) uncertainty. As described in Refs. [17, 27] the ATLAS JES uncertainty is only a few percent for jets with p_T of around 1 TeV where data are plentiful, while the limited size of observed examples for higher- p_T jets requires an alternate approach to estimating the JES. The resulting JES uncertainty therefore grows rapidly with m_{jj} and has an impact of at most 15% [27]. To illustrate the covariance due to the JES uncertainty, consider a simplified two-parameter model for the impact on the m_{jj} distribution: $J(z, \theta) = 1 + 15\% \theta_1 z^4 + 5\% \theta_2 (1 - z)$, where z is the true dijet invariant mass and $z_{\max} = 7$ TeV. We use the best fit 3-parameter fit as a proxy for $f(z)$ and fold in the smearing $W(x|z, \theta) = \text{Gaus}(x|zJ(z/z_{\max}, \theta), \sigma_x)$, where $\sigma_x = 2\%z$ is the dijet invariant mass resolution [17]. By assuming a uniform prior and an appropriate scaling for θ , we sample from the posterior $\text{Gaus}(\theta_1|0, 1)\text{Gaus}(\theta_2|0, 1)$ and propagate the uncertainty in θ through to the predicted bin counts $\hat{f}(\mathbf{x}|\theta)$ as in Eqs. 4 and 5. This allows us to explicitly build the covariance matrix Σ using the simulation shown in Fig. 3. As expected, we see a roughly block-diagonal structure defined by low and high mass regions.

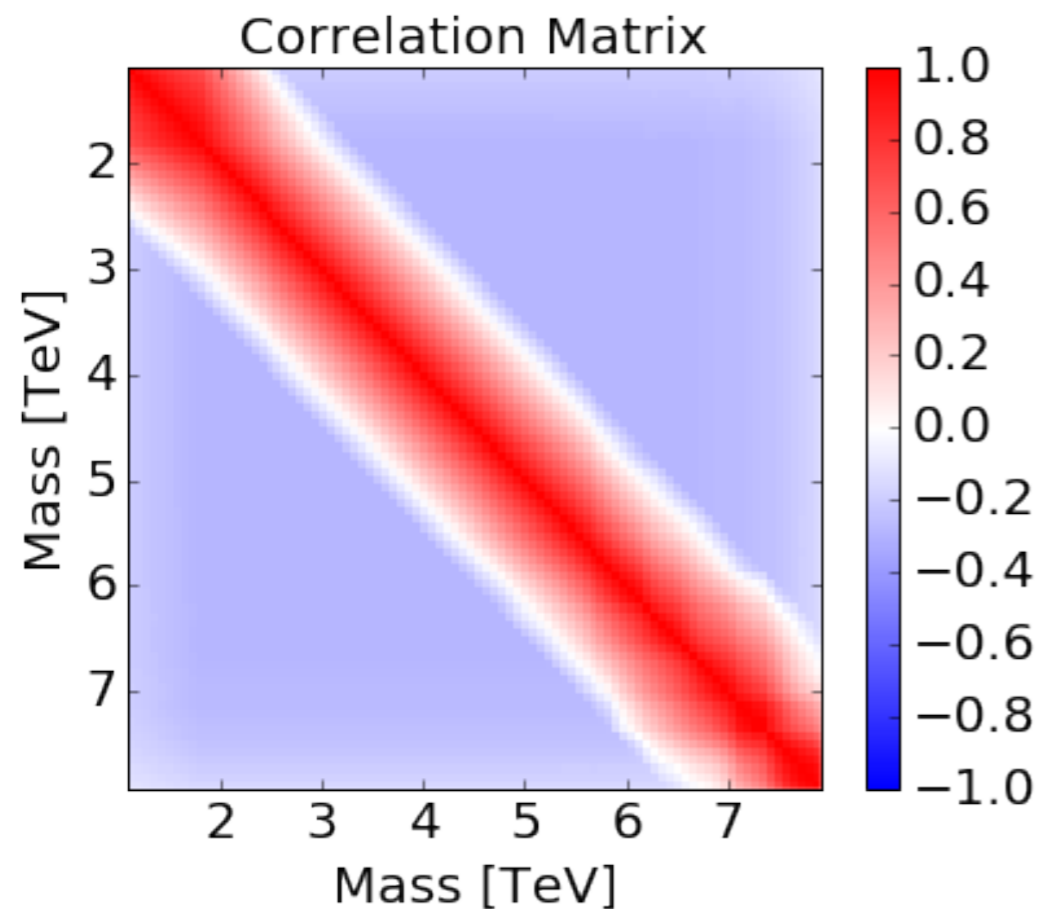
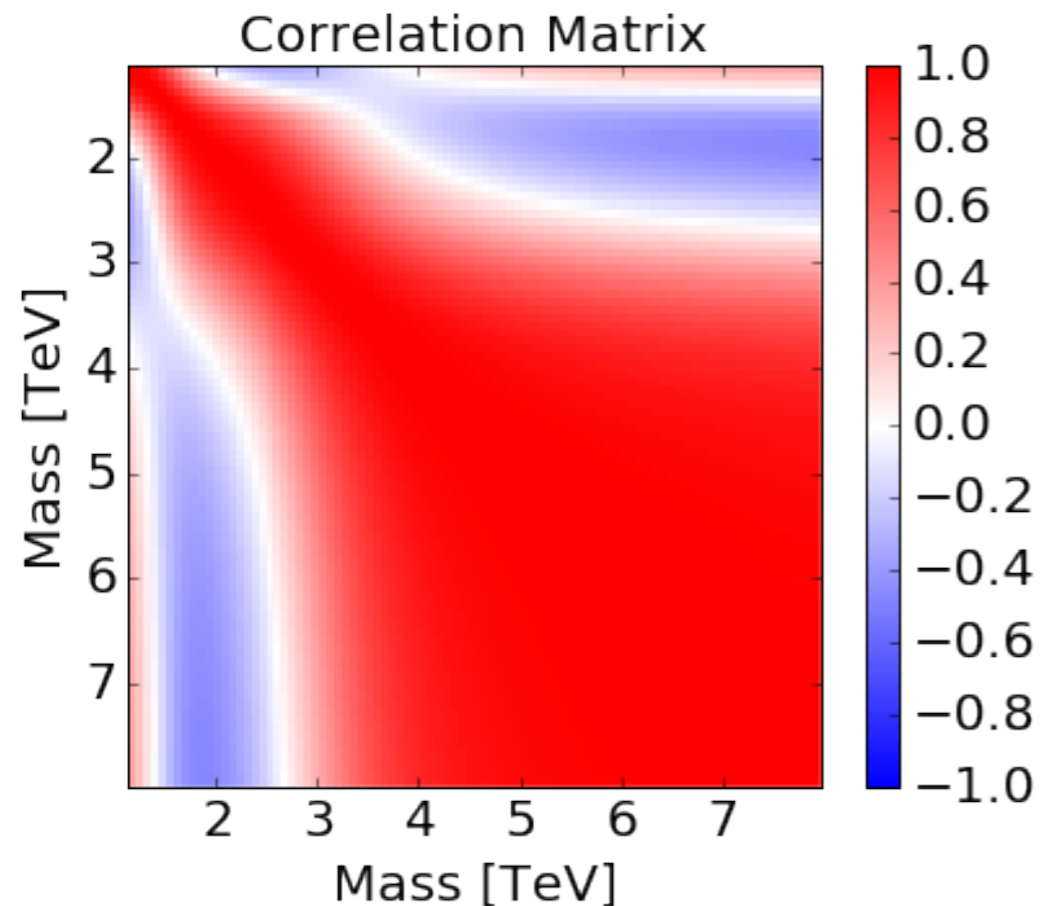


Even if the truth-level distribution is known, the folding matrix may not be known exactly.

Example: consider a jet energy scale with 2 nuisance parameters, where one parameter dominantly affects low- p_T jets (in situ) and the other high- p_T jets (limited stats for in situ).

Propagate uncertainty in jet energy scale to reconstructed m_{jj} spectrum, obtain covariance kernel.

EXAMPLE: TRADITIONAL DIJET

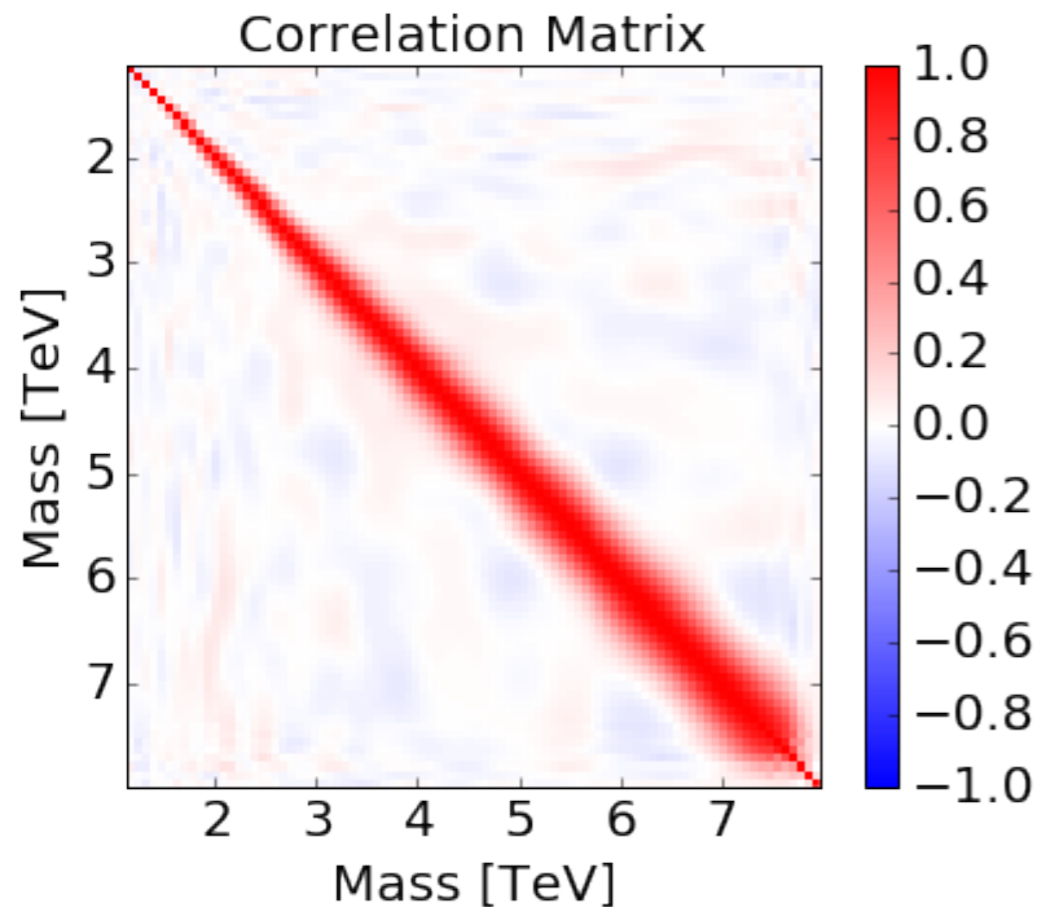


We can also think of the covariance structure for current fitting strategies.

- top: 3-parameter dijet function
- bottom: sliding window (SWIFT)

These are post-fit covariance plots.

POST-FIT PARAMETRIZED DIJET KERNEL



In addition to kernels constructed bottom-up from first-principles, we can also construct parametrized kernels using some intuition.

GPs adapt to the data very well, so even simple exponential-squared kernels often work fine.

For our dijet studies, we used a “Gibbs kernel”, which has length scale $l(x)$ and amplitude vary with x

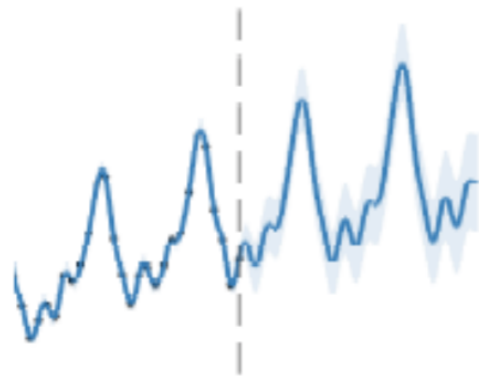
- plot shows post-fit covariance kernel

$$\Sigma(x, x') = Ae^{\frac{d-(x+x')}{2a}} \sqrt{\frac{2l(x)l(x')}{l(x)^2+l(x')^2}} e^{\frac{-(x-x')^2}{l(x)^2+l(x')^2}}$$

FUTURE DIRECTIONS

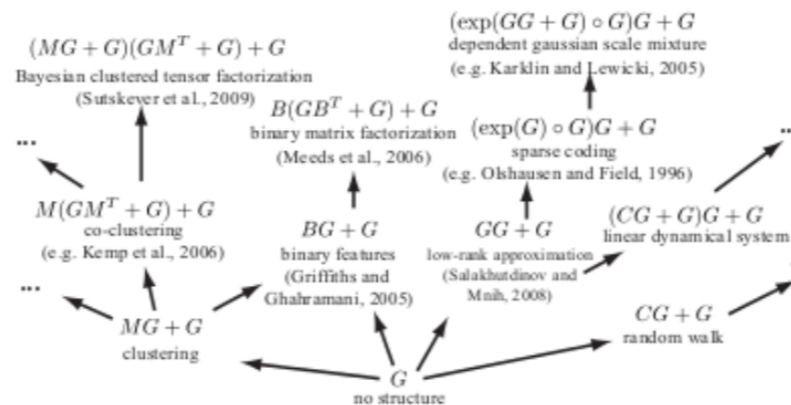
Vocabulary of kernels + grammar for composition

- physics goes into the construction of a "Kernel" that describes covariance of data



Structure Discovery in Nonparametric Regression through Compositional Kernel Search

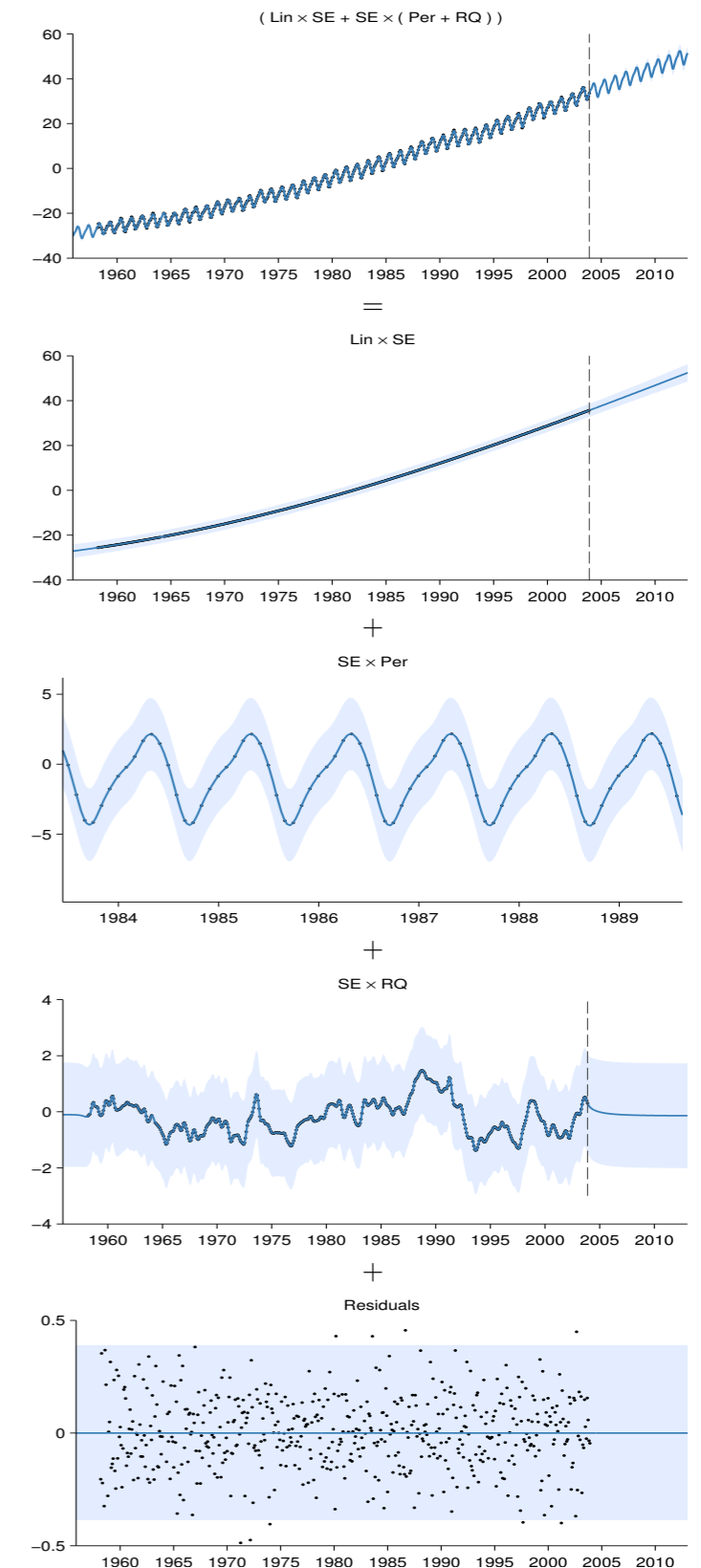
David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, Zoubin Ghahramani
International Conference on Machine Learning, 2013
[pdf](#) | [code](#) | [poster](#) | [bibtex](#)



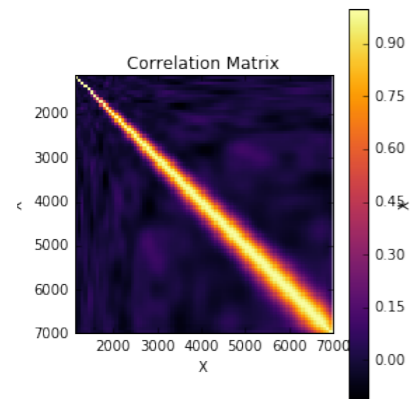
Exploiting compositionality to explore a large space of model structures

Roger Grosse, Ruslan Salakhutdinov, William T. Freeman, Joshua B. Tenenbaum
Conference on Uncertainty in Artificial Intelligence, 2012
[pdf](#) | [code](#) | [bibtex](#)

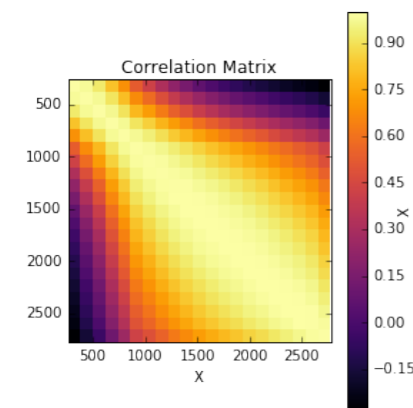
Mauna Loa atmospheric CO₂



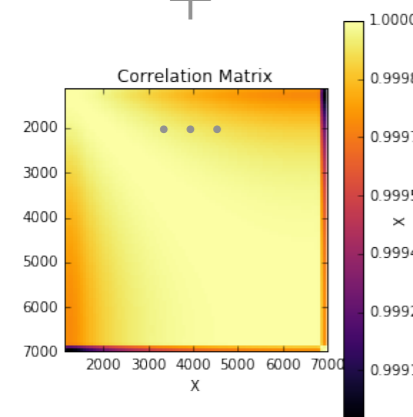
Instead of fitting the dijet spectrum with an ad hoc 3-5 parameter function, use GP with kernel motivated from physics



=



+



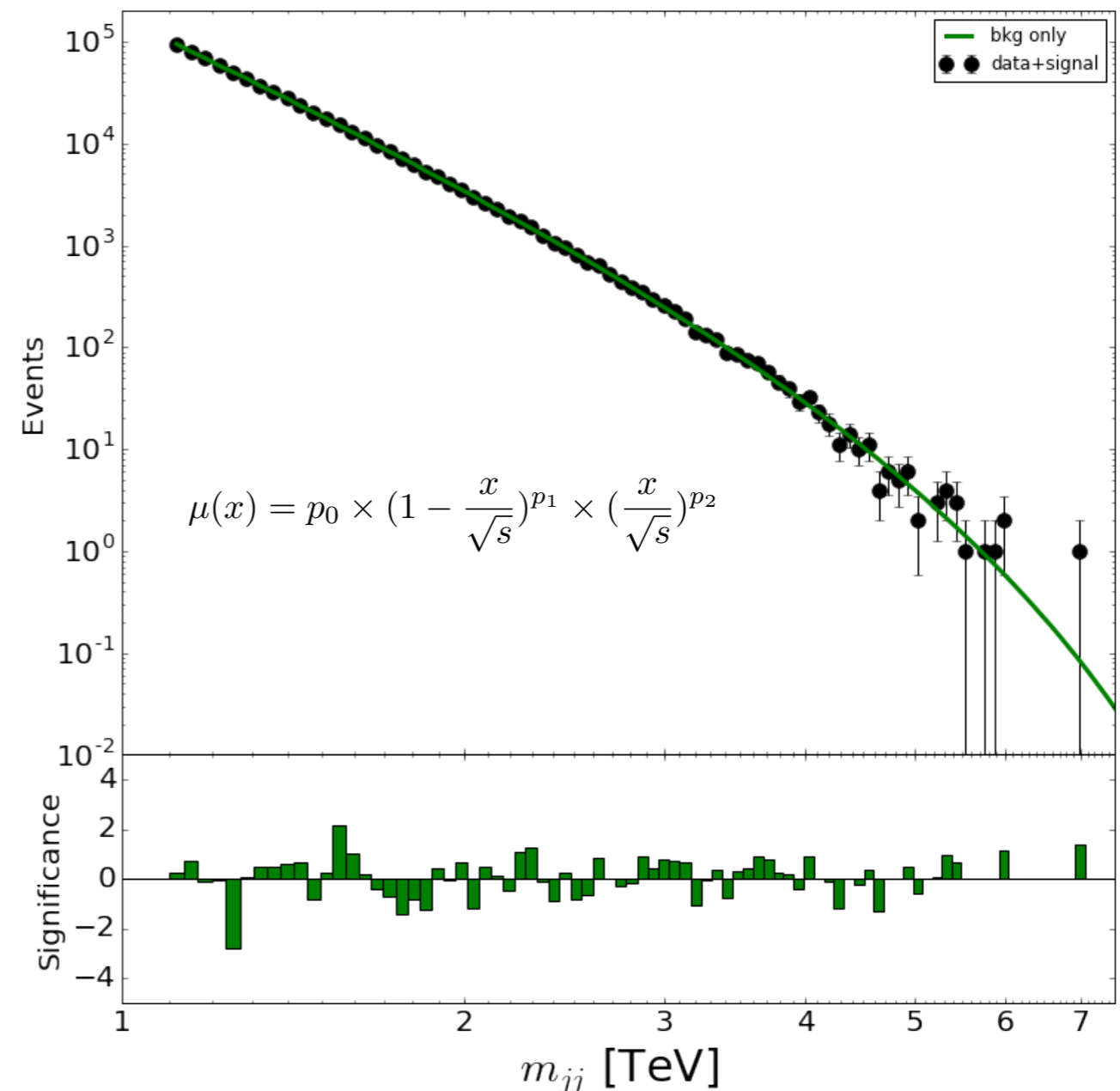
+

...

Final Kernel =
Poisson stats
+ Mass Resolution

+ Parton Density
Functions

+ Jet Energy Scale

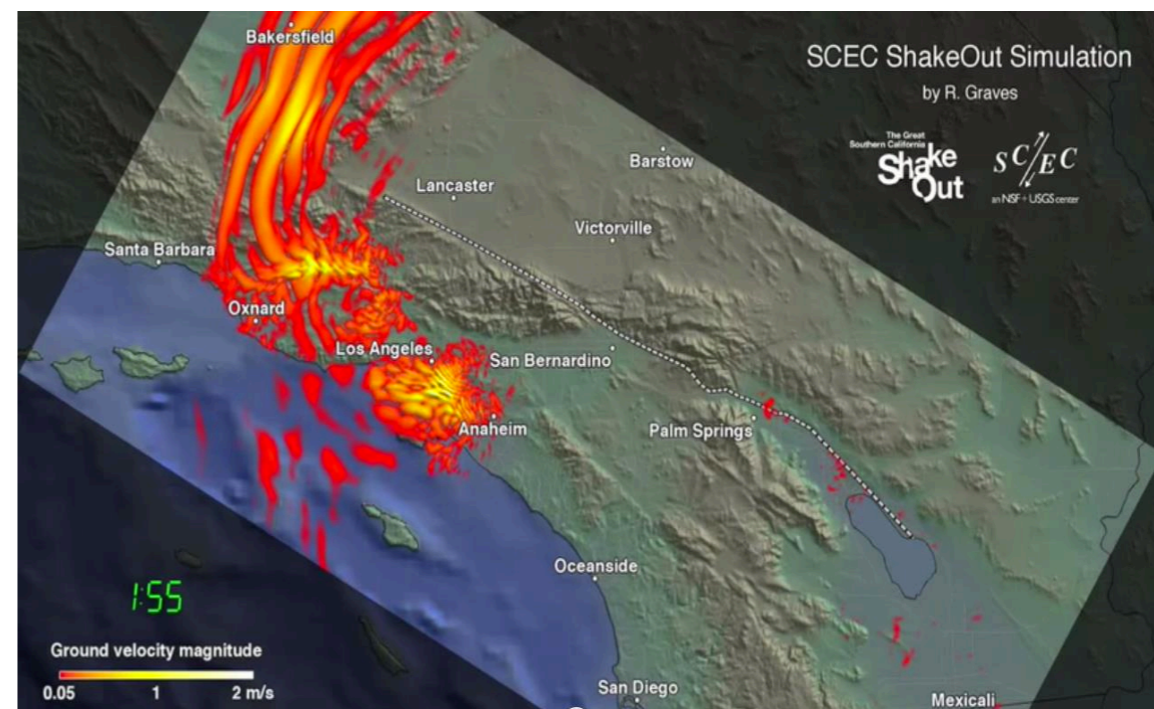
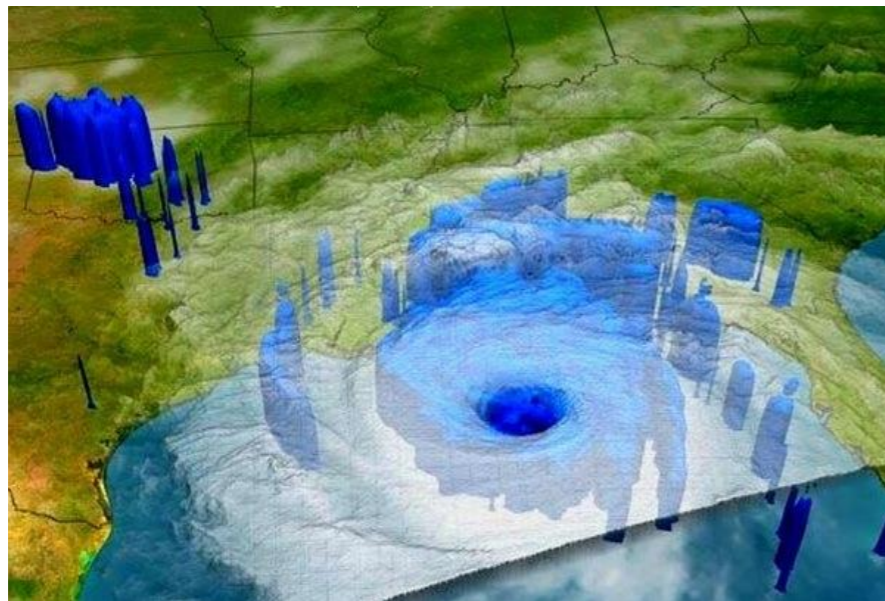
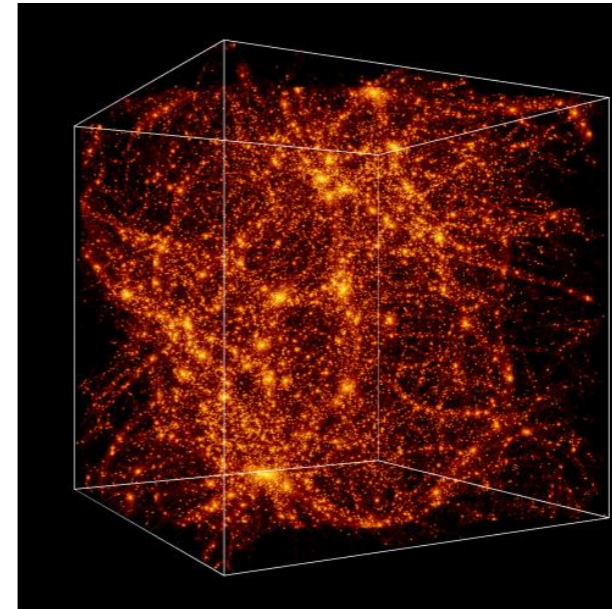
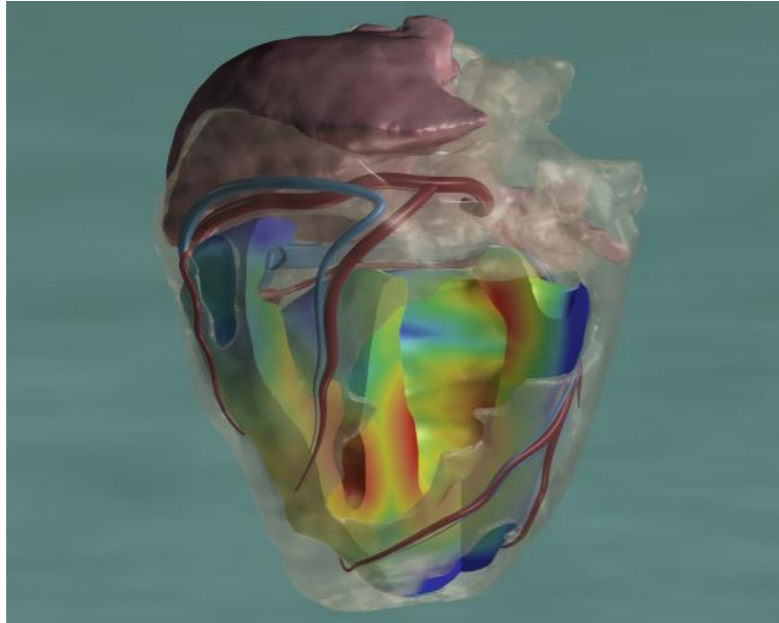


Likelihood-Free Inference



Max
Welling

Generative Models: Simulators



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.

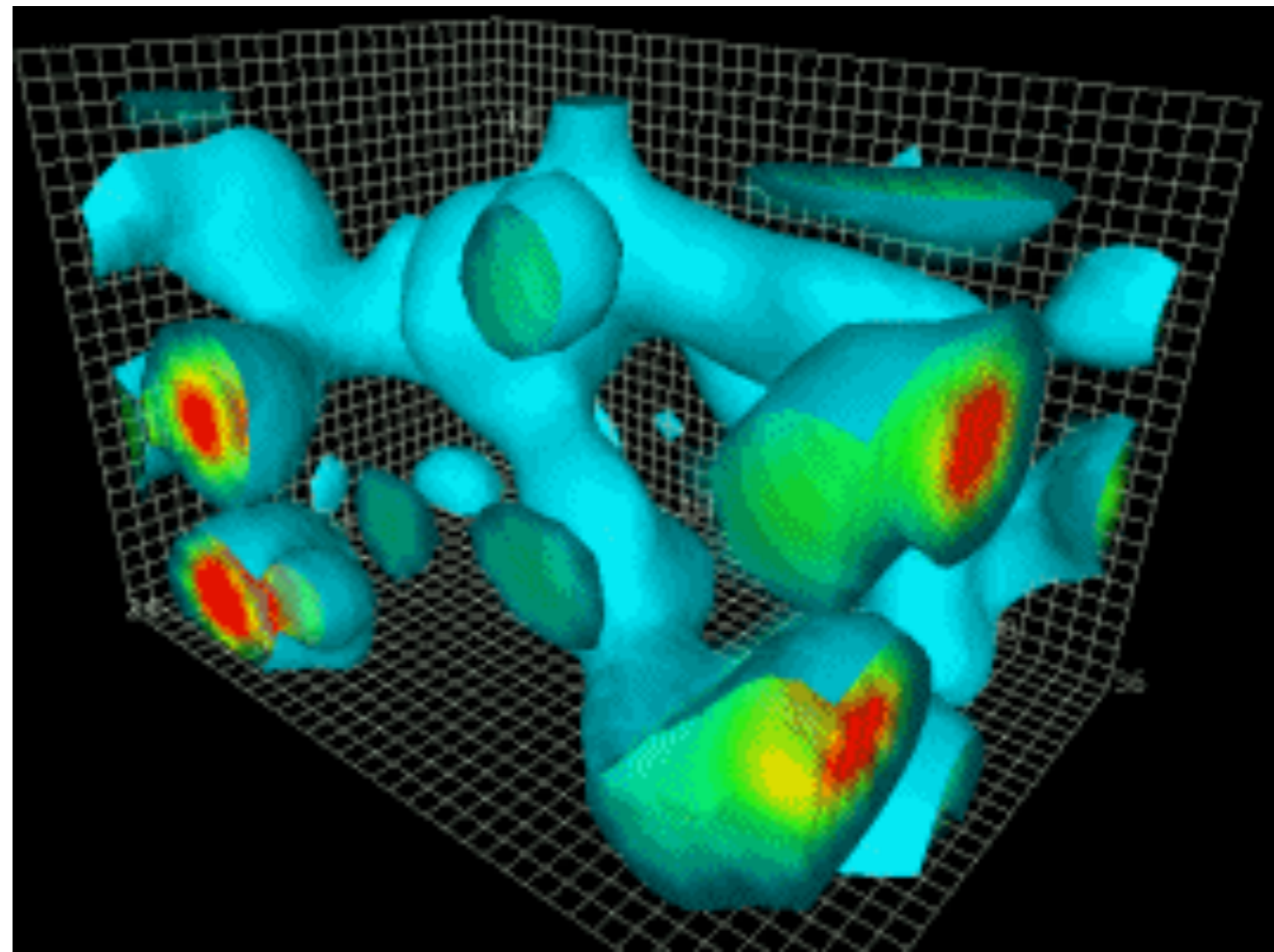
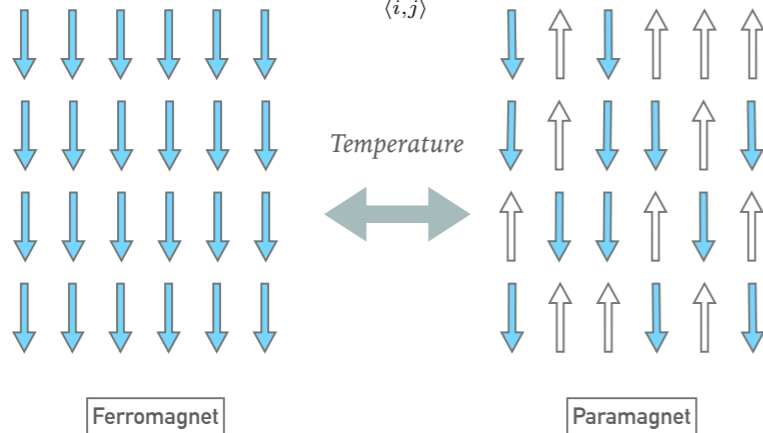
- they can provide effective statistical models for macroscopic phenomena that are tied back to the low-level microscopic (reductionist) model
- generative models and likelihood-free inference are two particularly exciting areas

LATTICE FIELD THEORY

PHASES, PHASE TRANSITIONS, AND THE ORDER PARAMETER

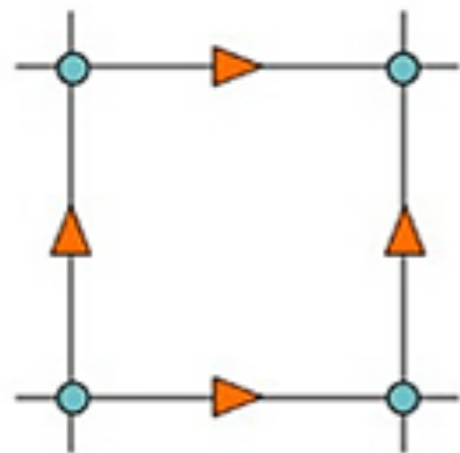
Ising ferromagnet in two dimensions

$$E = -J \sum_{\langle i,j \rangle} \sigma_i \sigma_j$$

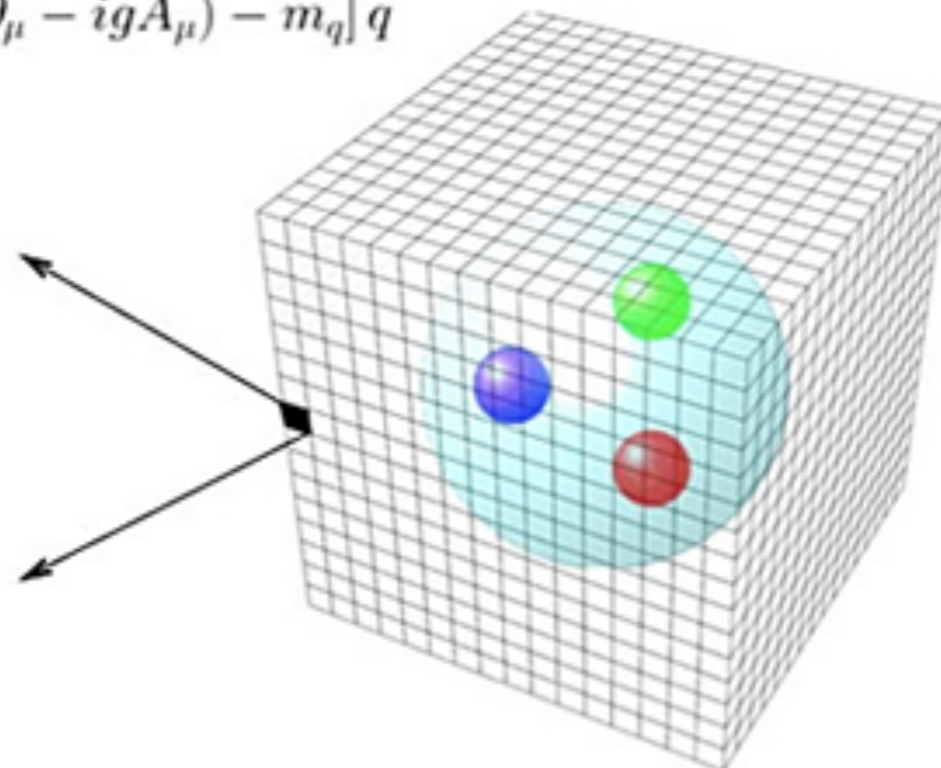


QCD Lagrangian

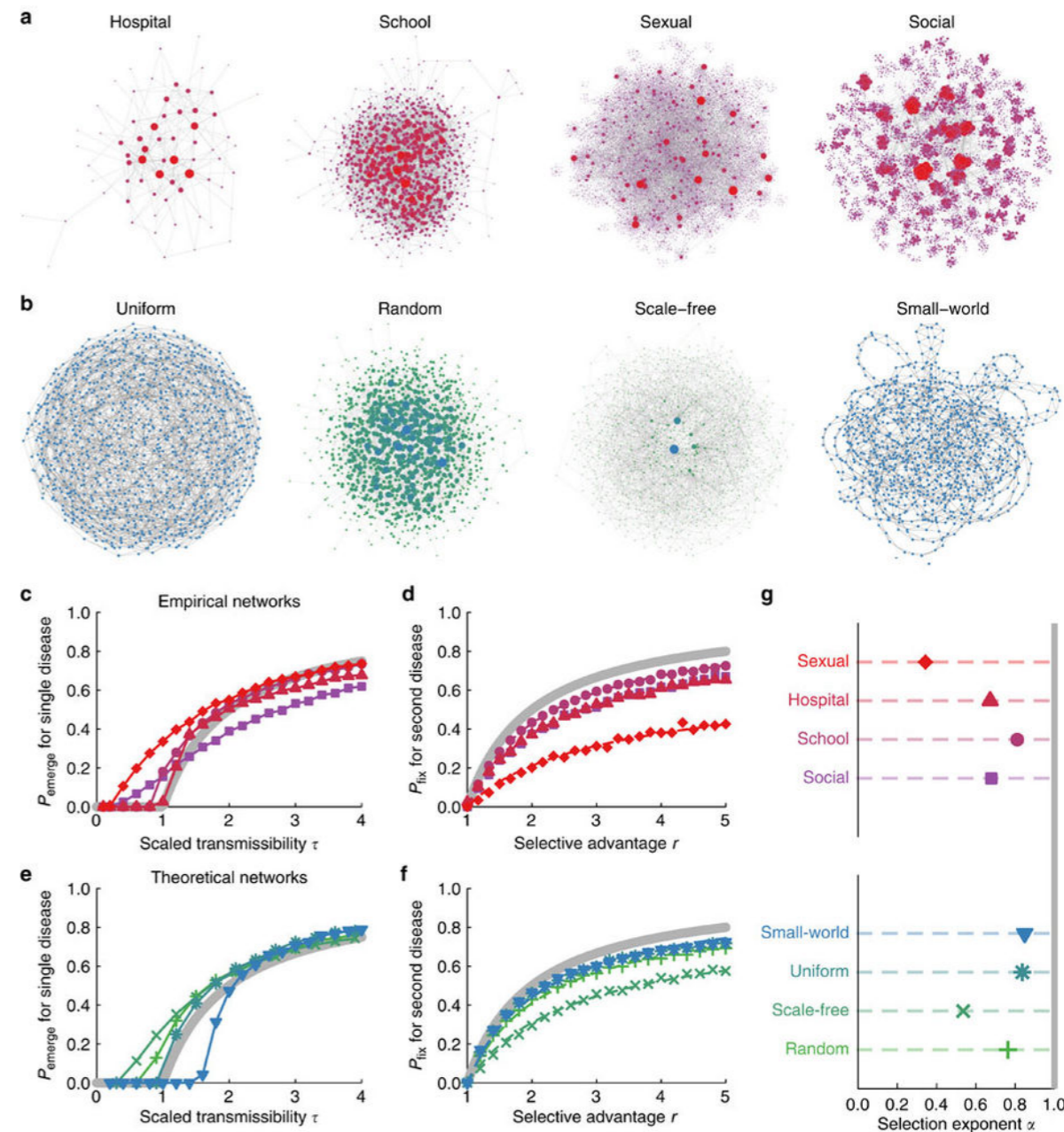
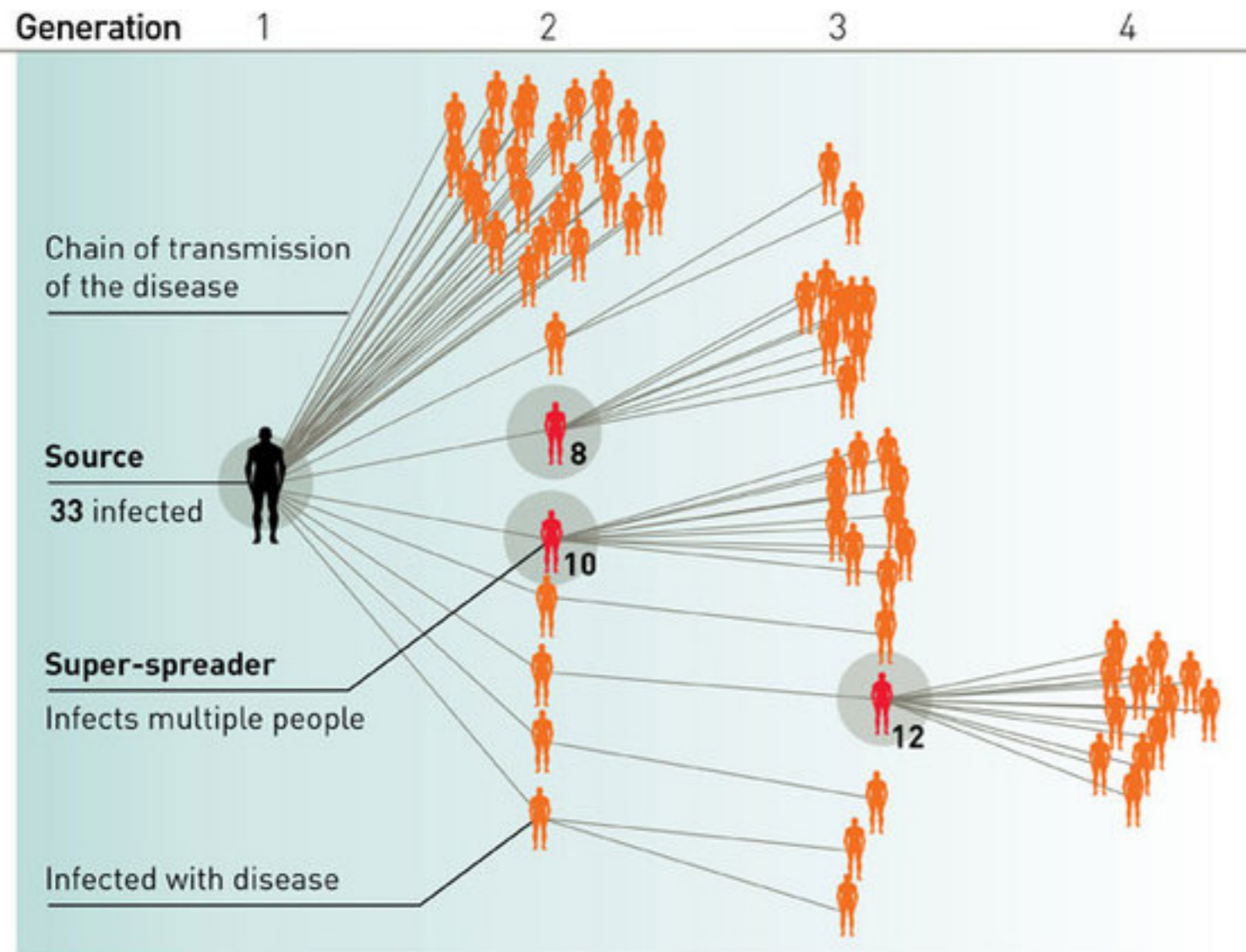
$$\mathcal{L} = -\frac{1}{4} F^{\mu\nu} F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q} [i\gamma^\mu (\partial_\mu - igA_\mu) - m_q] q$$



● quark ▲ gluon



EPIDEMIOLOGY & POPULATION GENETICS



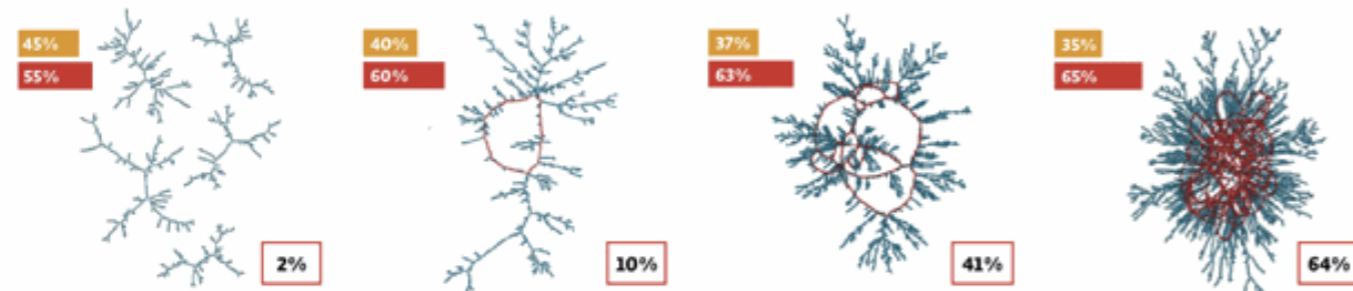
Small Change, Big Effects

KEY
1 partner
2 or 3 partners

Percent of people that are connected in the network through their sexual partnerships

Modest variations in the concurrency rate—the proportion of people in overlapping sexual partnerships—can have a dramatic effect on a population's vulnerability to HIV.

When the concurrency rate is 55%, only 2% of this population is connected to the broader sexual network required for HIV transmission (top). But when concurrency reaches 65%, an astonishing 64% of the population is vulnerable, even though the number of sexual partners remains constant.



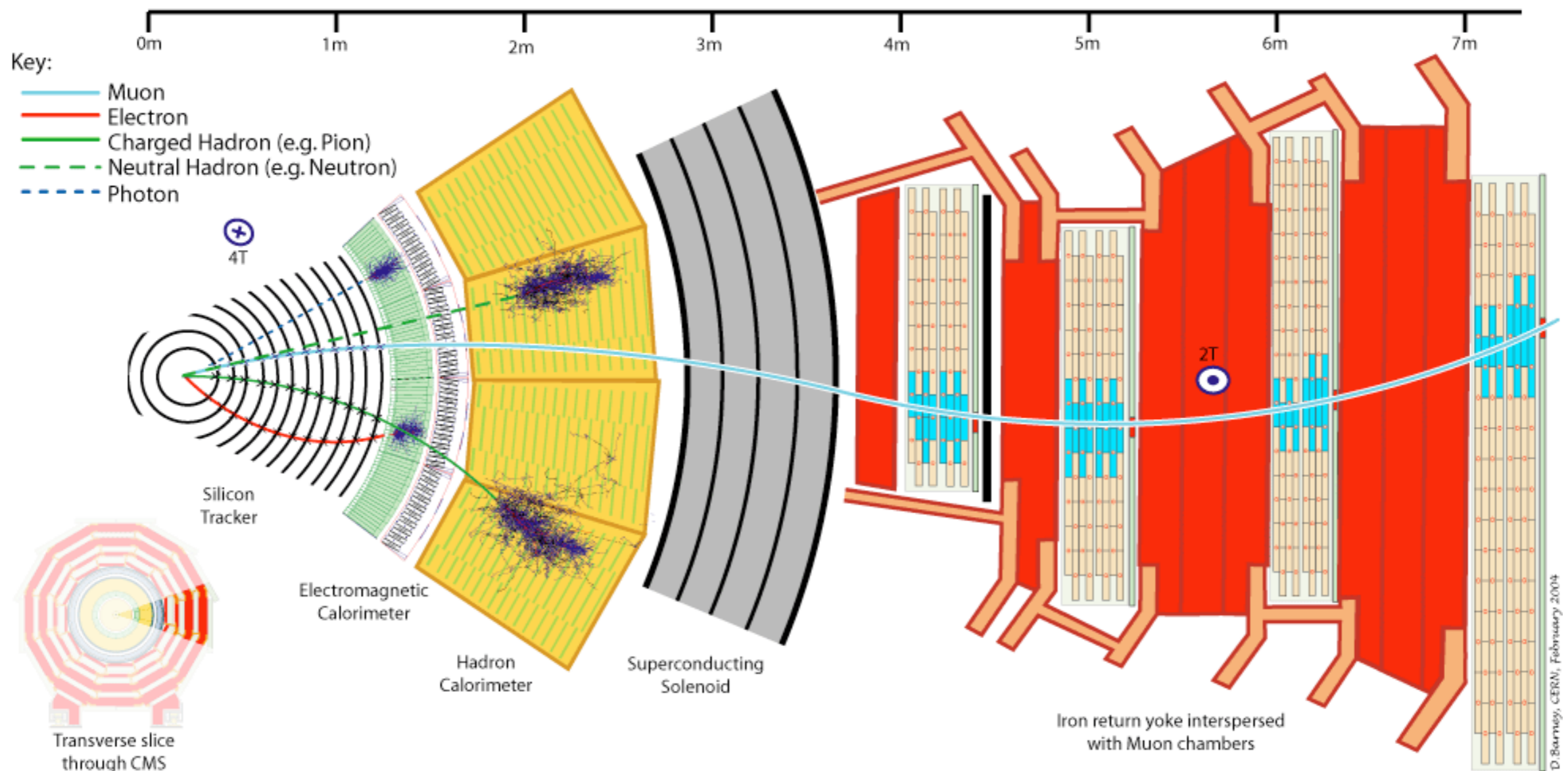
Source: Morris, et al. The Relationship Between Concurrent Partnerships and HIV Transmission, 2008. See www.aidsstar-one.com/.

DETECTOR SIMULATION

Conceptually: $\text{Prob}(\text{detector response} \mid \text{particles})$

Implementation: Monte Carlo integration over micro-physics

Consequence: evaluation of the likelihood is intractable



DETECTOR SIMULATION

Conceptually: $\text{Prob}(\text{detector response} \mid \text{particles})$

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”

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.

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.

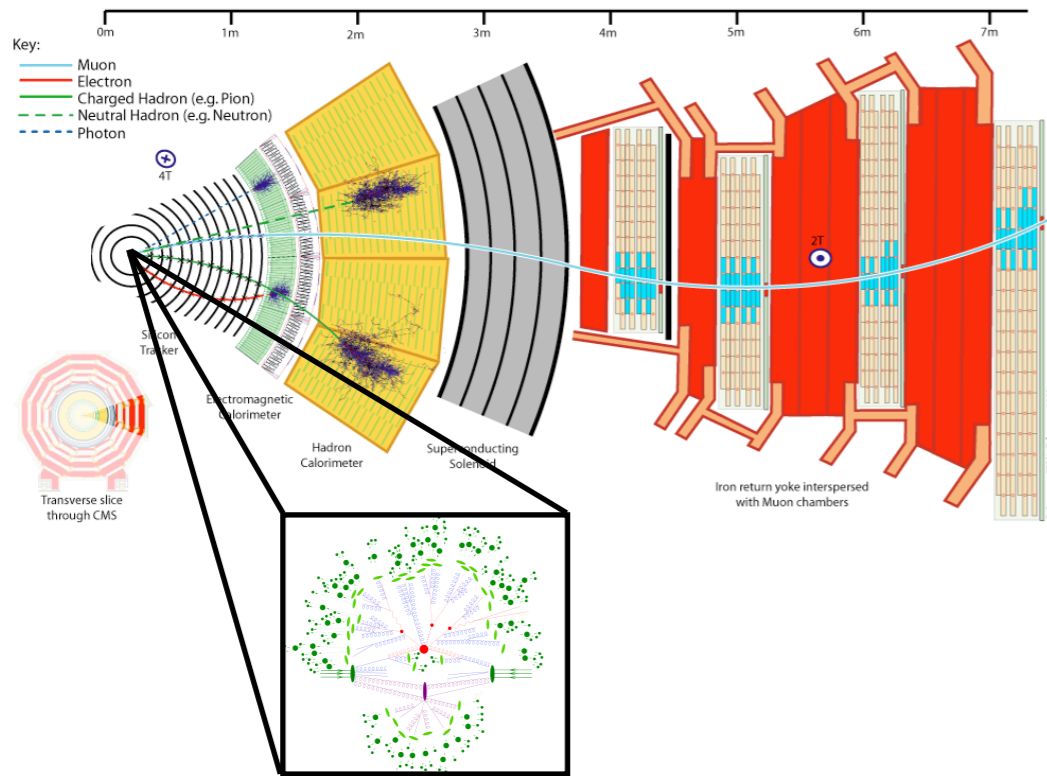
Of particular interest at this workshop is to unite fields that work on implicit models. For example:

- **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.

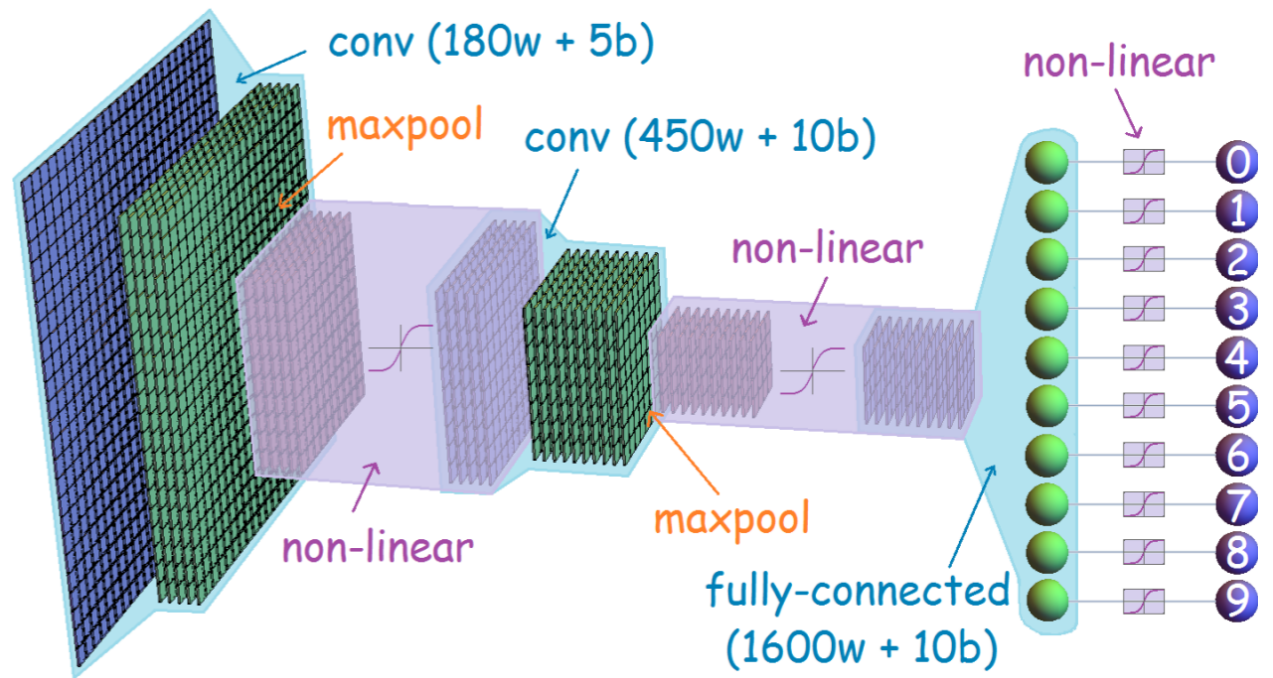
APPROACHES TO LIKELIHOOD-FREE INFERENCE

Use simulator
(much more efficiently)



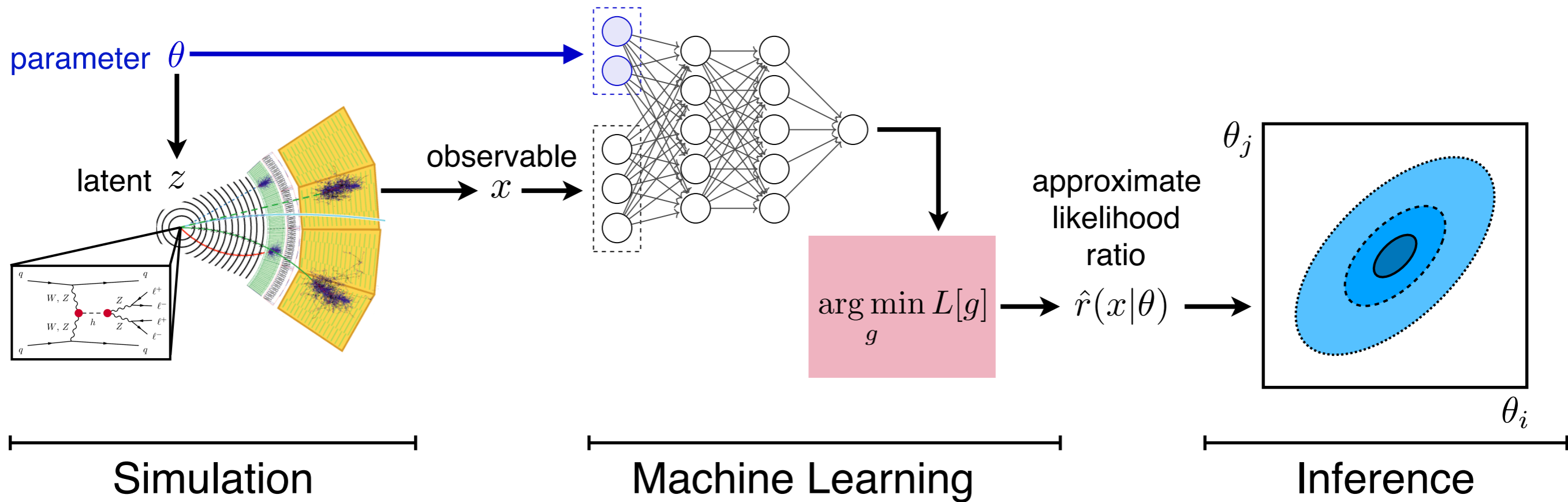
- Approximate Bayesian Computation (ABC)
- Probabilistic Programming
- Adversarial Variational Optimization (AVO)

Learn simulator
(with deep learning)

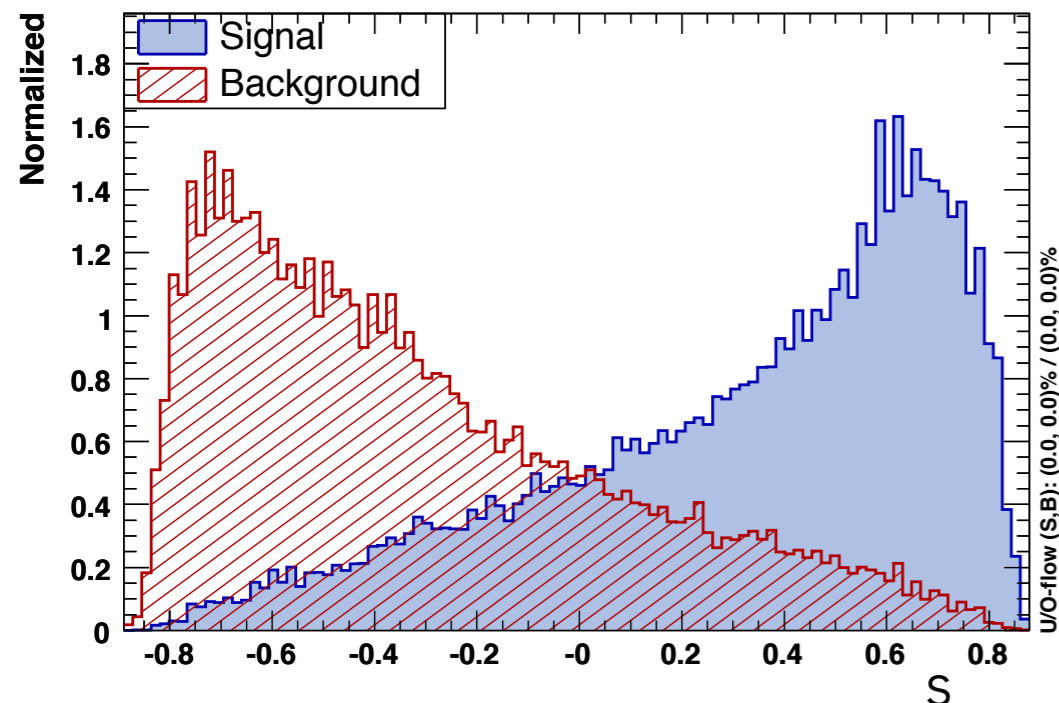
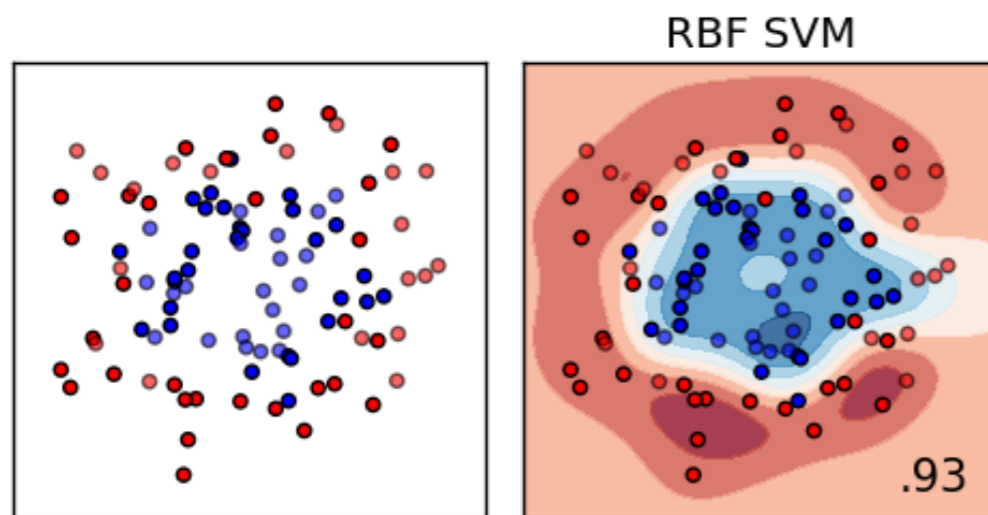
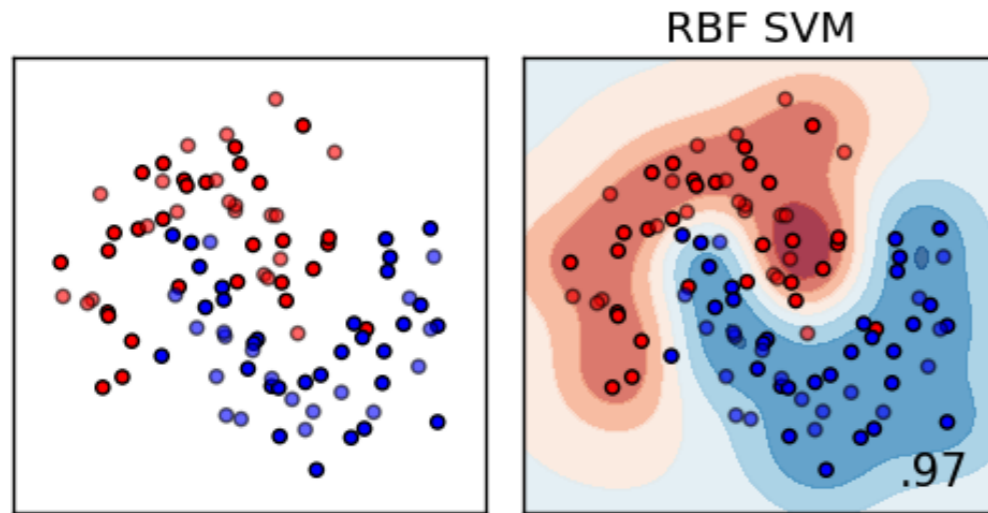


- Generative Adversarial Networks (GANs), Variational Auto-Encoders (VAE)
- Likelihood ratio from classifiers (CARL)
- Autoregressive models, Normalizing Flows

LEARNING FROM SIMULATED DATA



LIKELIHOOD RATIO TRICK



- **binary classifier**: 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

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

EXTENDING THE LIKELIHOOD RATIO TRICK

A binary classifier approximates

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

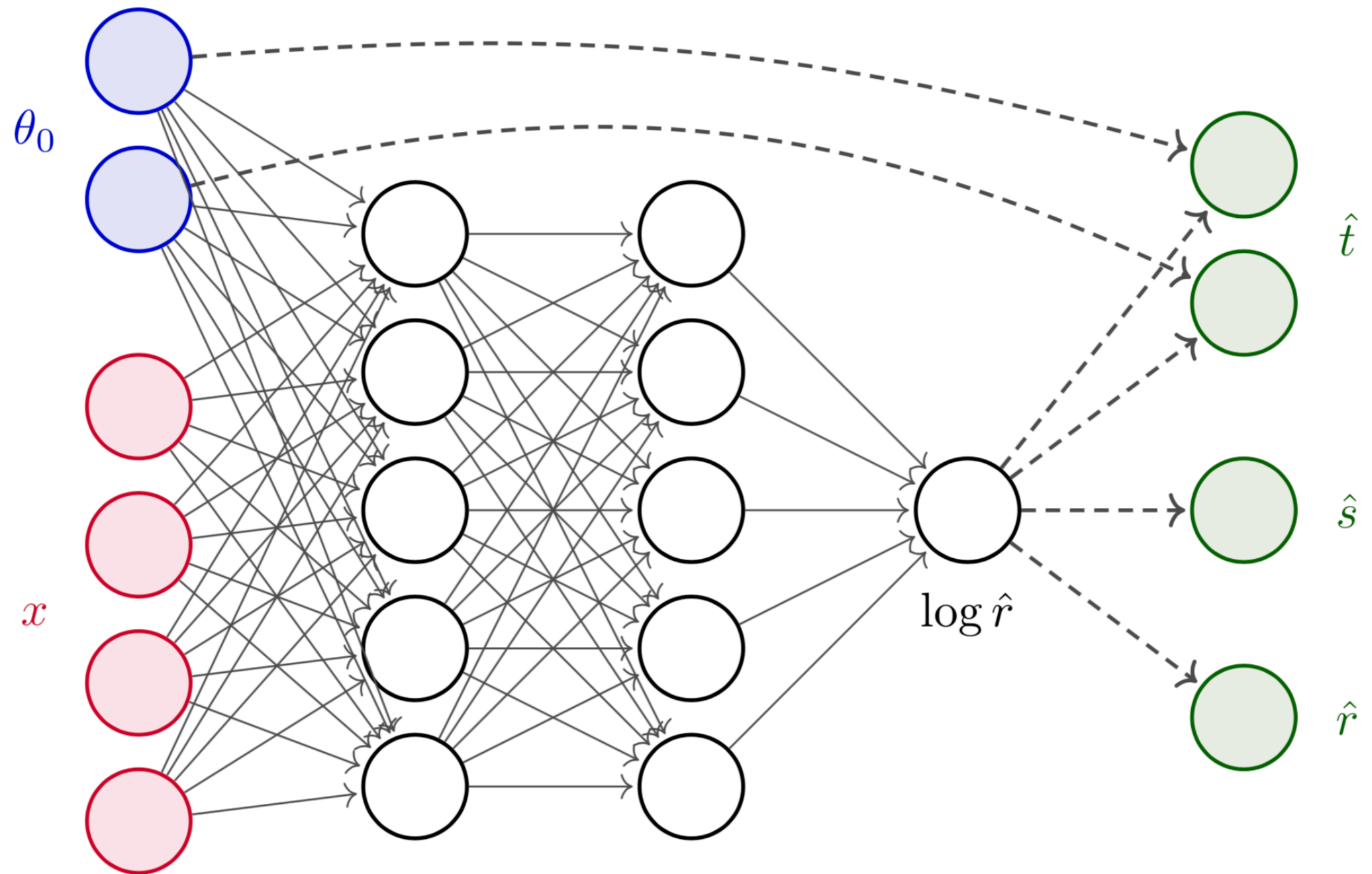
Which is one-to-one with the likelihood ratio

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

Can do the same thing for any two points θ_0 & θ_1 in parameter space Θ . I call this a **parametrized classifier**

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

PARAMETERIZED CLASSIFIERS



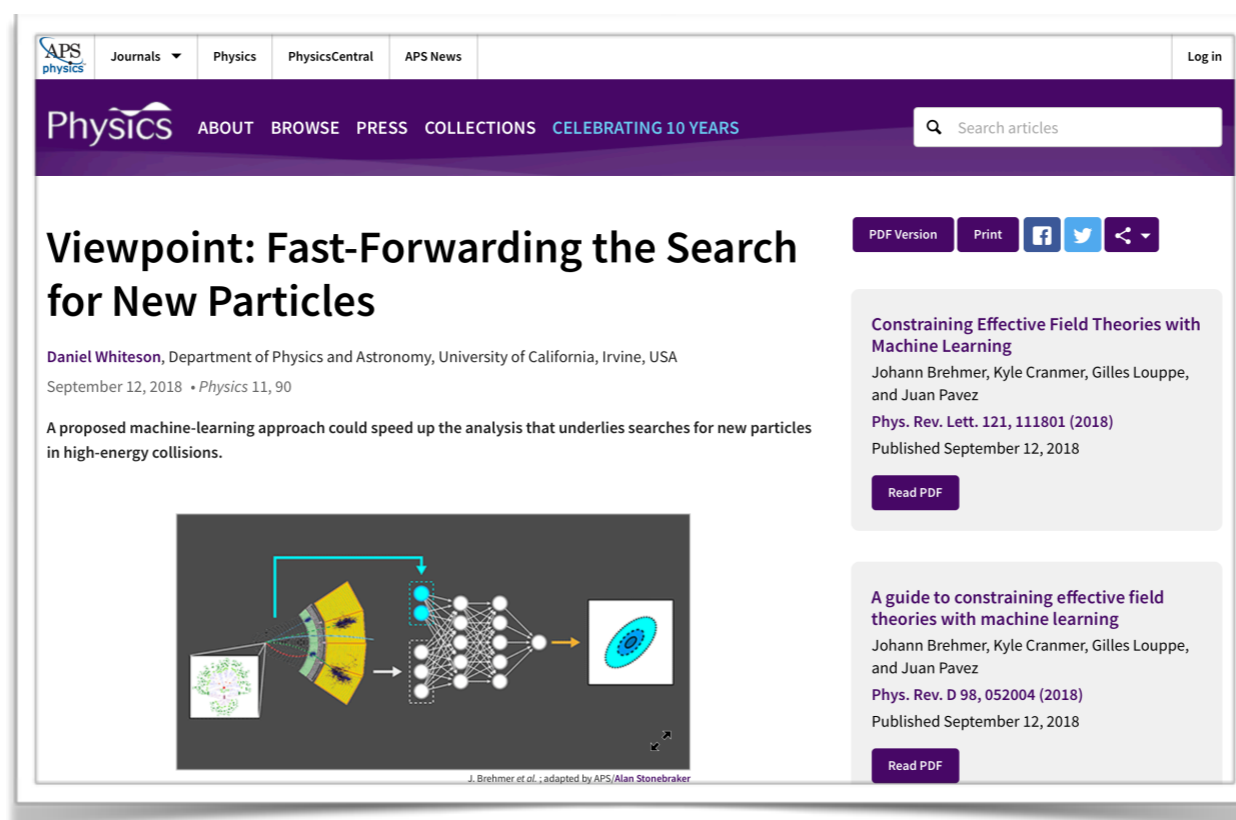
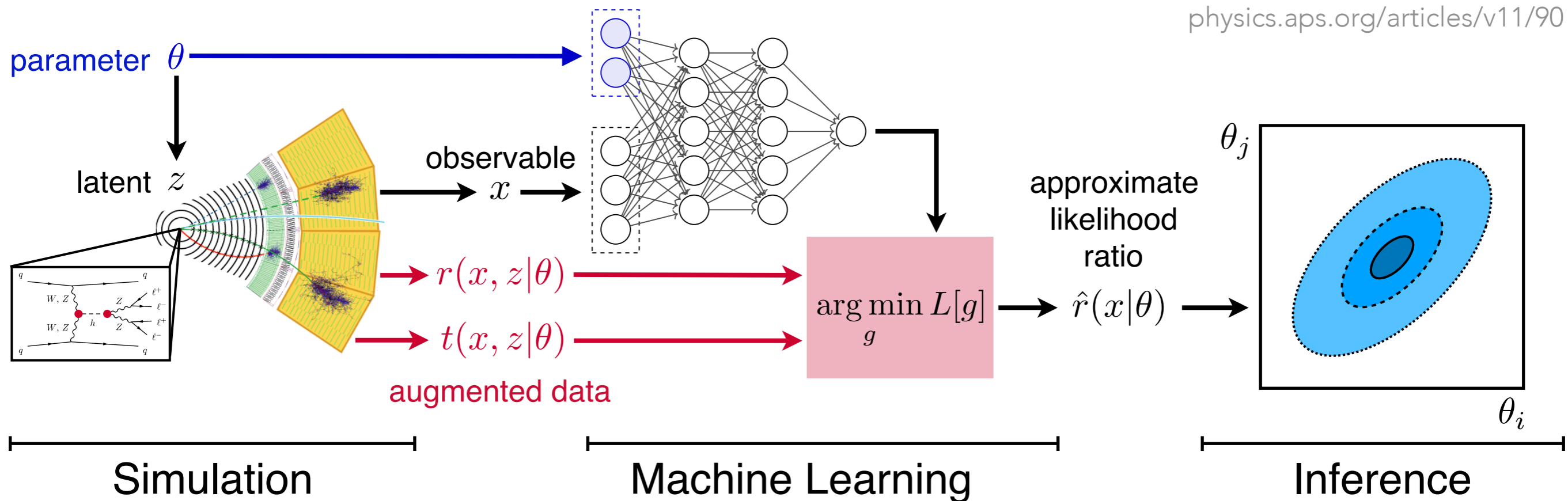
LEARNING WITH AUGMENTED DATA

arXiv:1805.12244

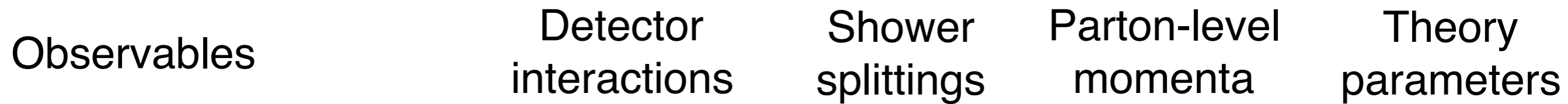
PRL, arXiv:1805.00013

PRD, arXiv:1805.00020

physics.aps.org/articles/v11/90



MINING GOLD FROM THE SIMULATION



$$x \longleftarrow z_d \longleftarrow z_s \longleftarrow z_p \longleftarrow \theta$$

$$p(x|\theta) = \int dz_d \int dz_s \int dz_p p(x|z_d)$$

$$p(z_d|z_s)$$

$$p(z_s|z_p)$$

$$p(z_p|\theta)$$

This parton-level likelihood can be evaluated!

⇒ We can calculate the “joint” likelihood ratio conditional on a specific evolution:

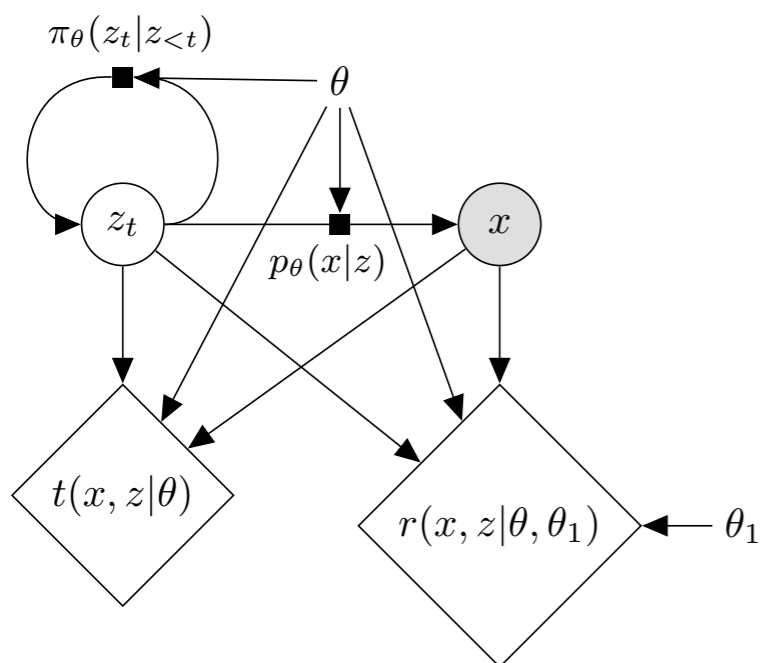
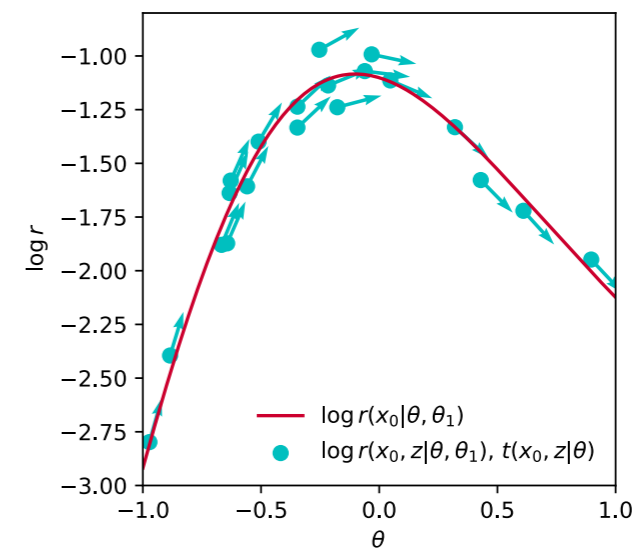
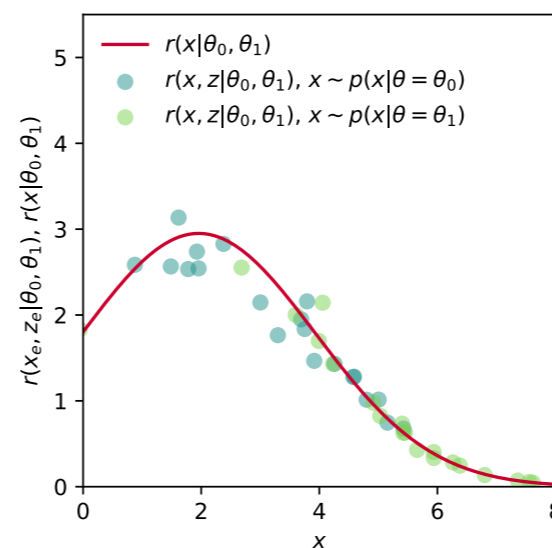
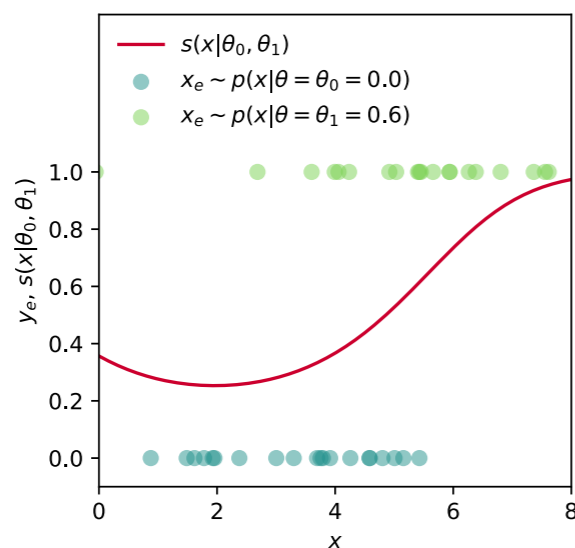
$$r(x, z|\theta_0, \theta_1) \equiv \frac{p(x, z_d, z_s, z_p|\theta_0)}{p(x, z_d, z_s, z_p|\theta_1)} = \frac{p(x|z_d)}{p(x|z_d)} \frac{p(z_d|z_s)}{p(z_d|z_s)} \frac{p(z_s|z_p)}{p(z_s|z_p)} \frac{p(z_p|\theta_0)}{p(z_p|\theta_1)}$$

(“How much more likely is this specific evolution assuming θ_0 compared to θ_1 ?”)

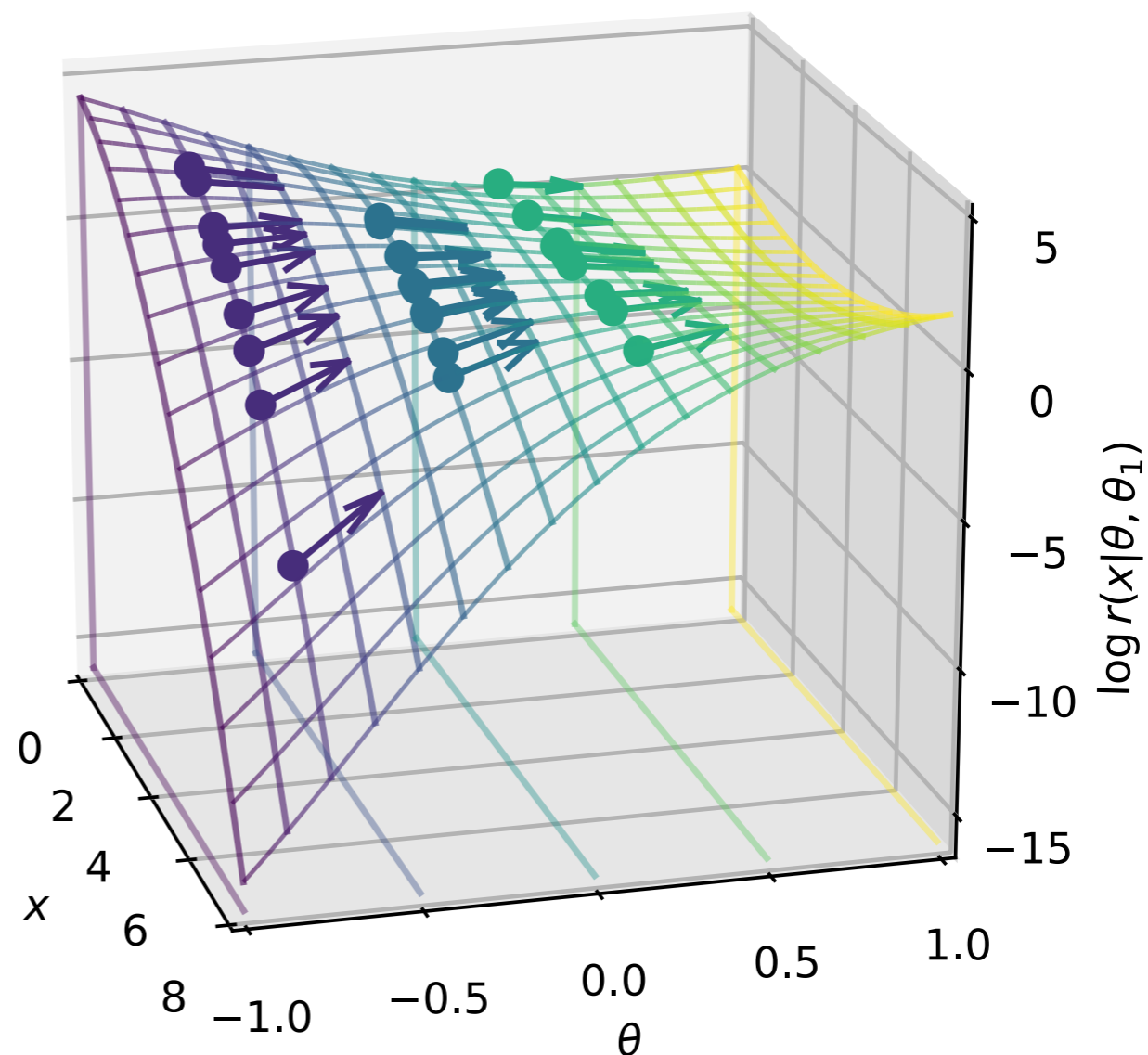
⇒ Similarly, we can calculate the joint score

$$t(x, z|\theta_0) \equiv \nabla_{\theta} \log p(x, z_d, z_s, z|\theta)$$

PUTTING IT ALL TOGETHER



can think of simulator
as policy π_θ in language of
reinforcement learning



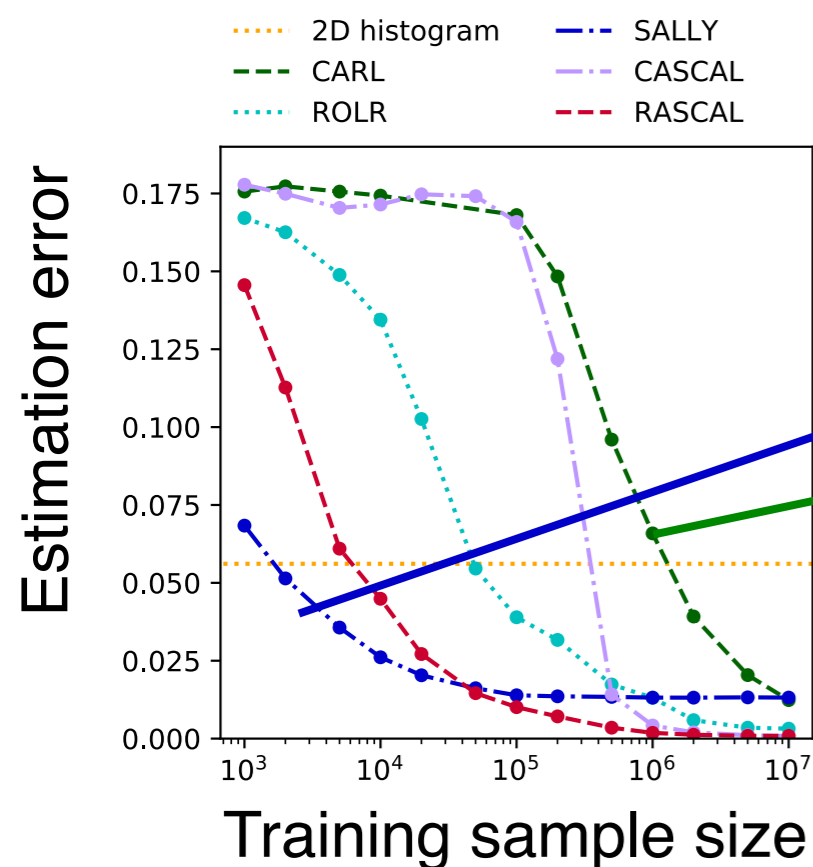
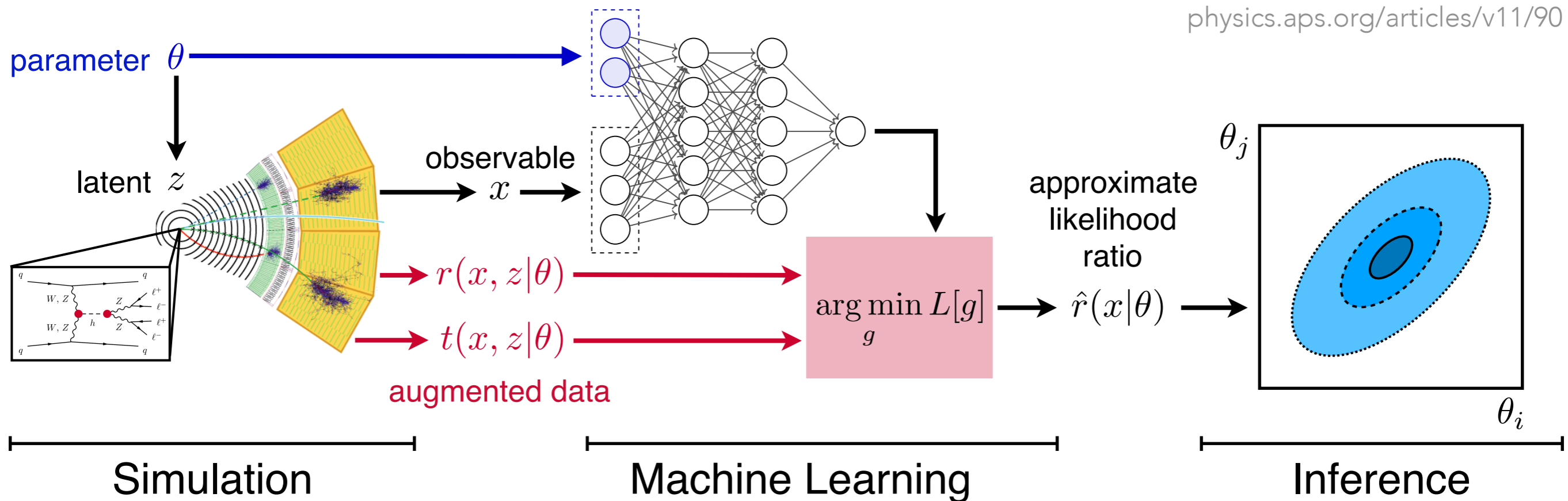
LEARNING WITH AUGMENTED DATA

arXiv:1805.12244

PRL, arXiv:1805.00013

PRD, arXiv:1805.00020

physics.aps.org/articles/v11/90

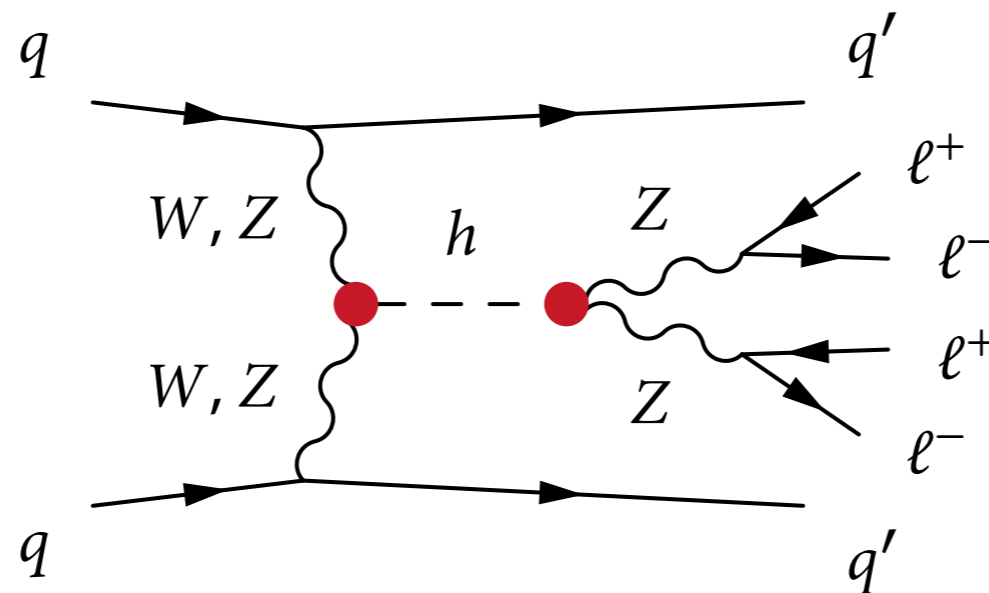
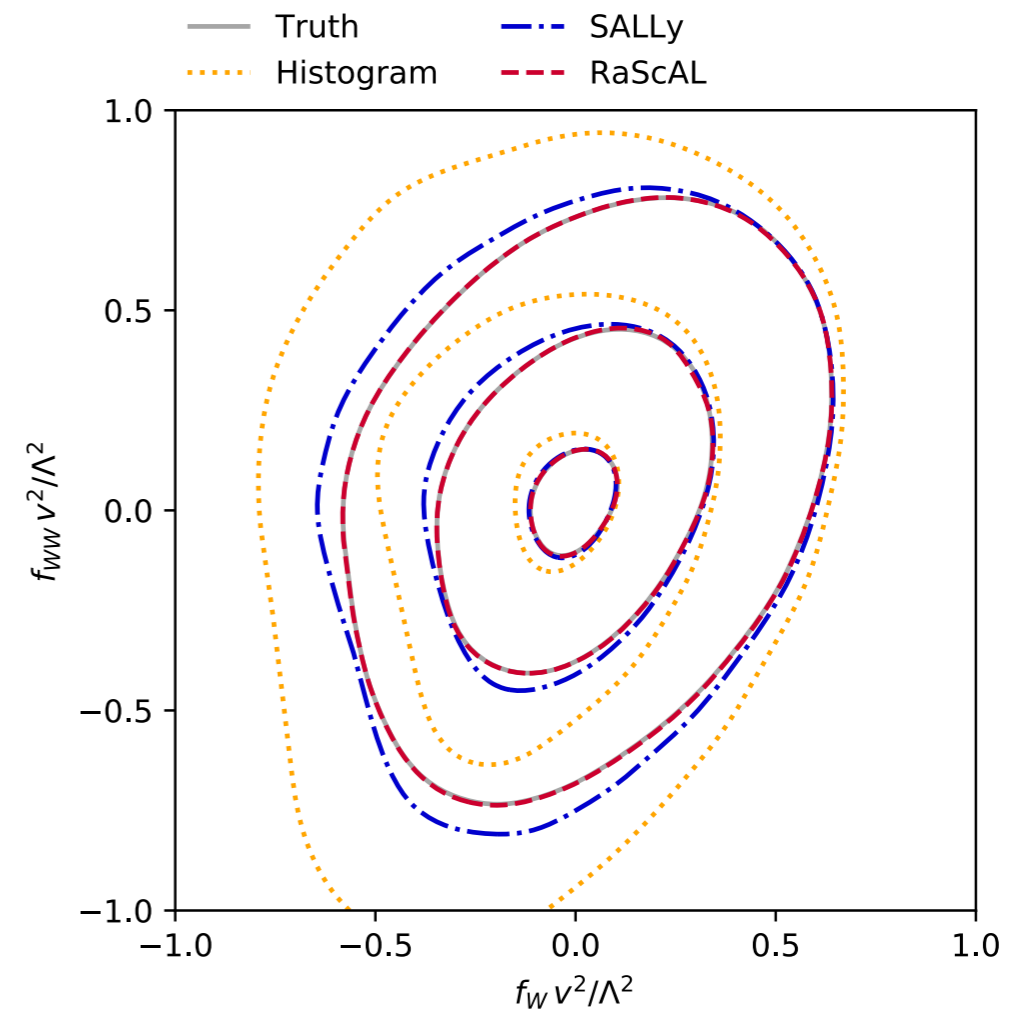
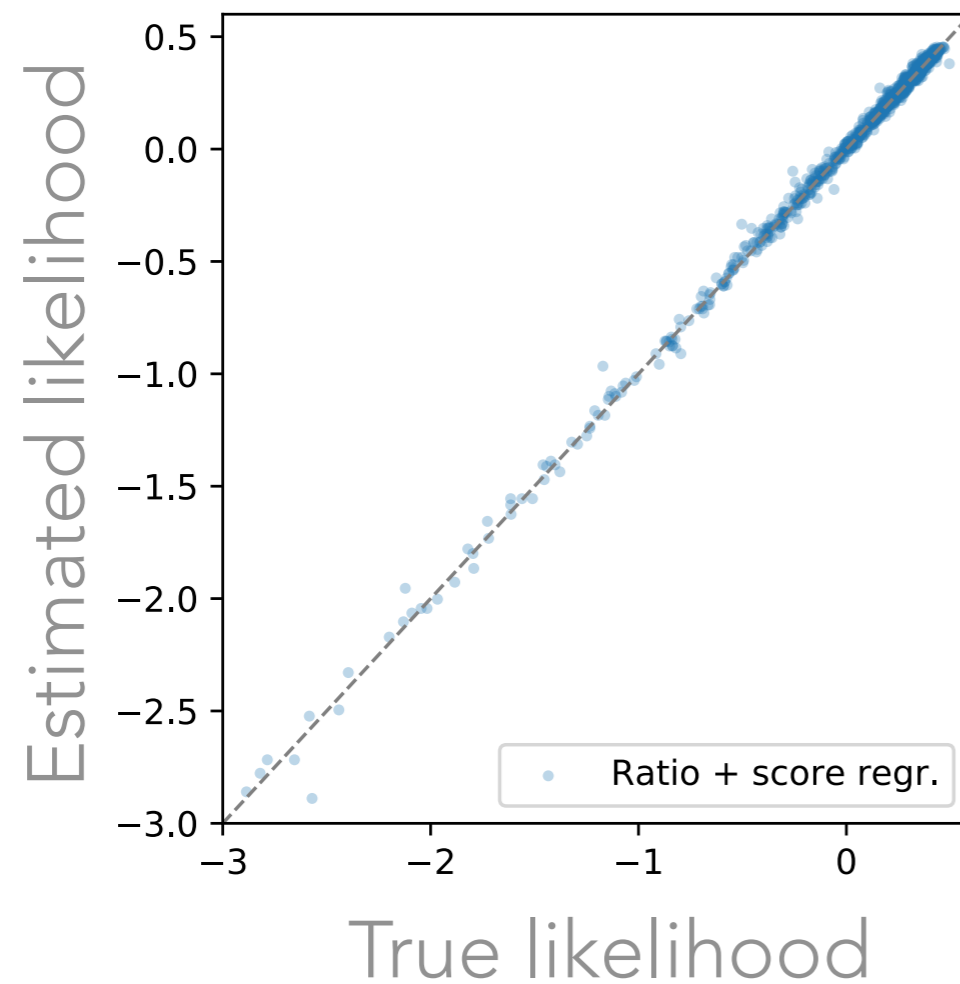


New techniques
require less data than
without augmented data

Traditional Histogram

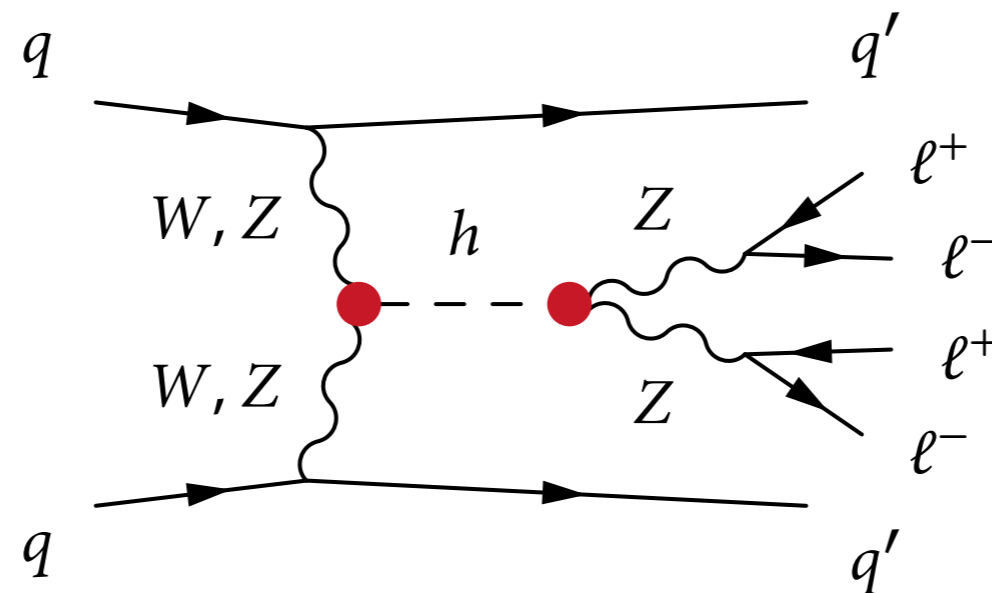
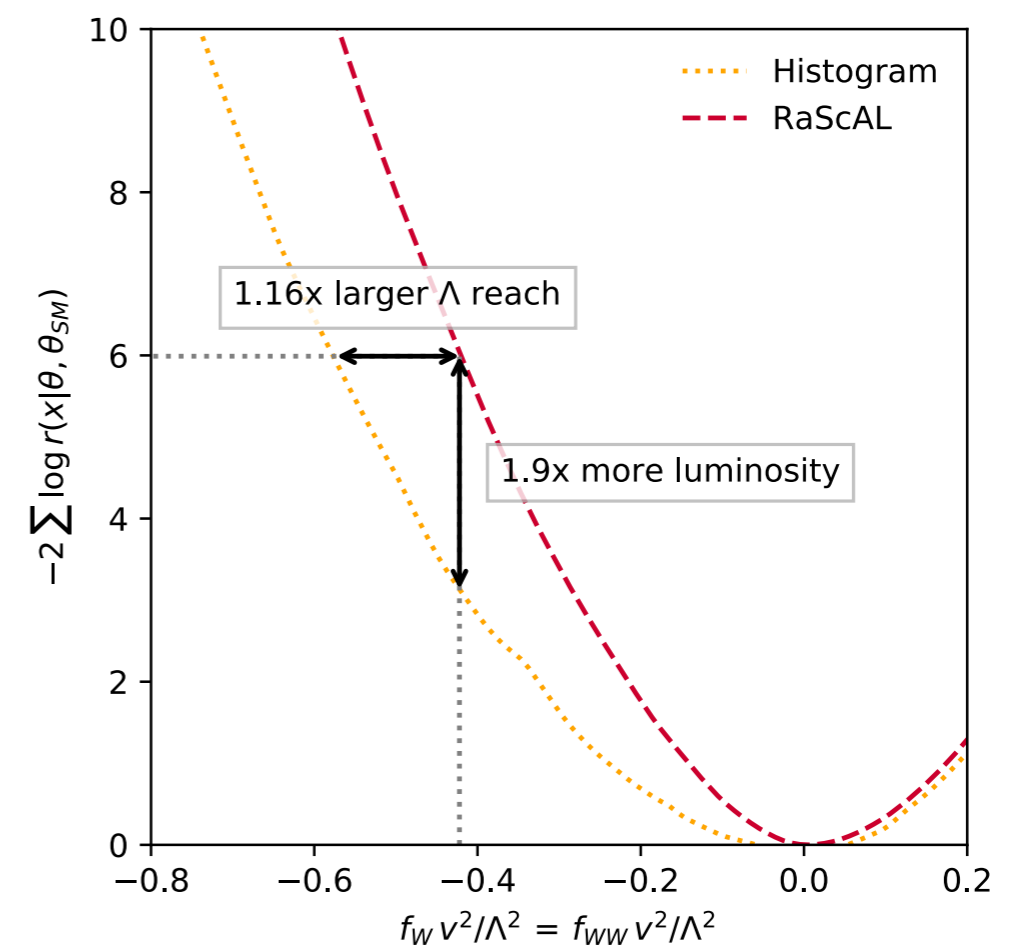
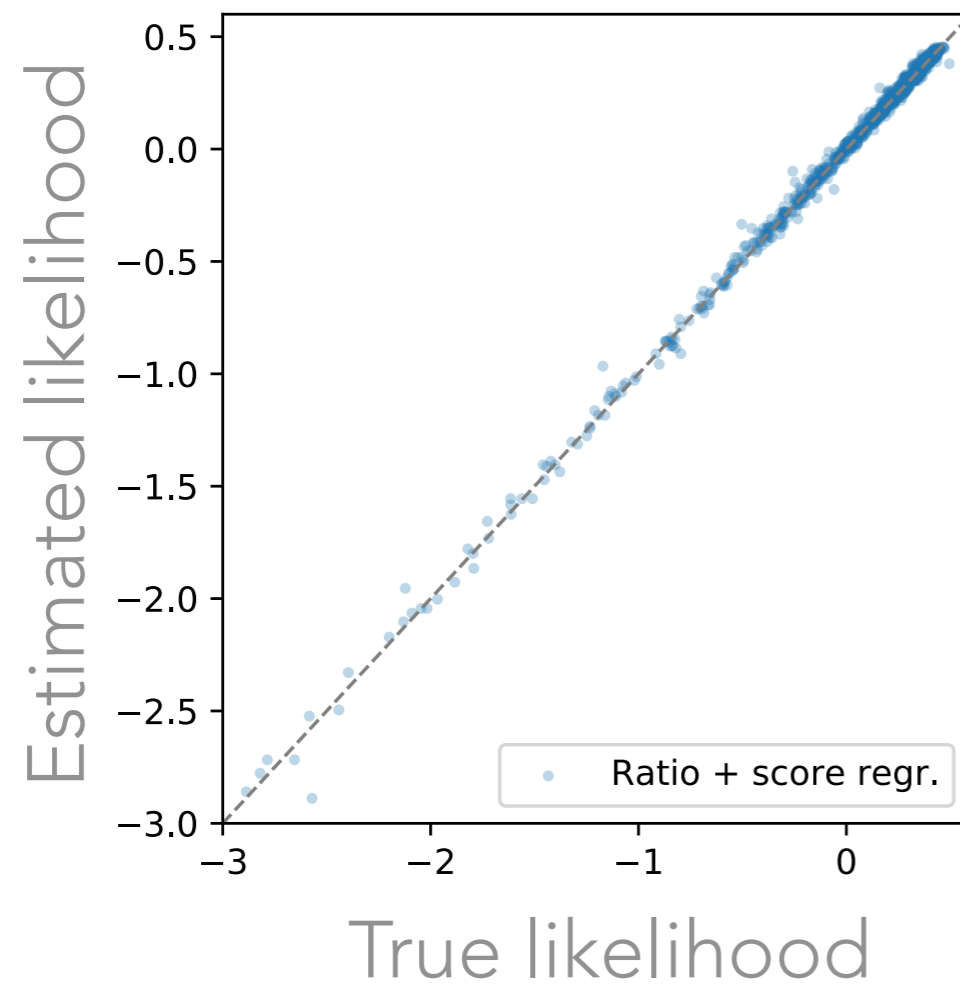
MACHINE LEARNING THE HIGGS EFFECTIVE FIELD THEORY

(based on a 42-Dim observation \mathbf{x})



MACHINE LEARNING THE HIGGS EFFECTIVE FIELD THEORY

(based on a 42-Dim observation \mathbf{x})



Physics-Aware Machine Learning

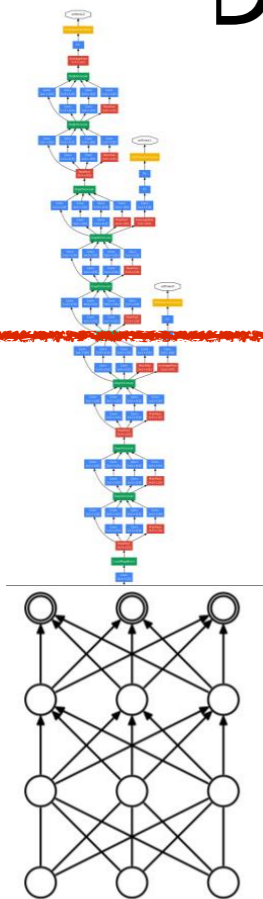
(choosing the variational family)

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



Max Welling

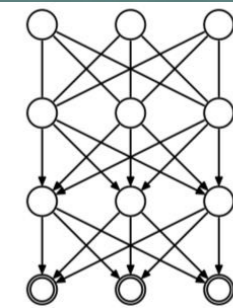
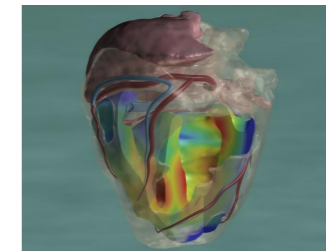
Discriminative or Generative?



-Deep Learning
-Kernel Methods
-Random Forests
-Boosting

Exciting

-Bayesian Networks
-Probabilistic Programs
-Simulator Models



- Advantages discriminative models:
 - Flexible map from input to target (low bias)
 - Efficient training algorithms available
 - Solve the problem you are evaluating on.
 - Very successful and accurate!

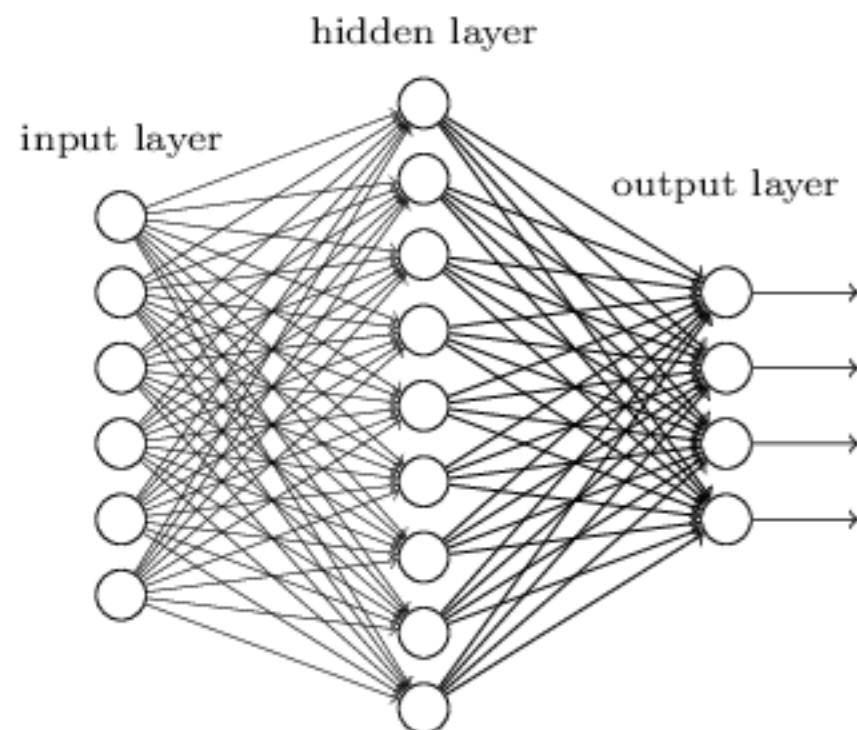
- Advantages generative models:
 - Inject expert knowledge
 - Model causal relations
 - Interpretable
 - Data efficient
 - More robust to domain shift
 - Facilitate un/semi-supervised learning

NN = A HIGHLY FLEXIBLE FAMILY OF FUNCTIONS

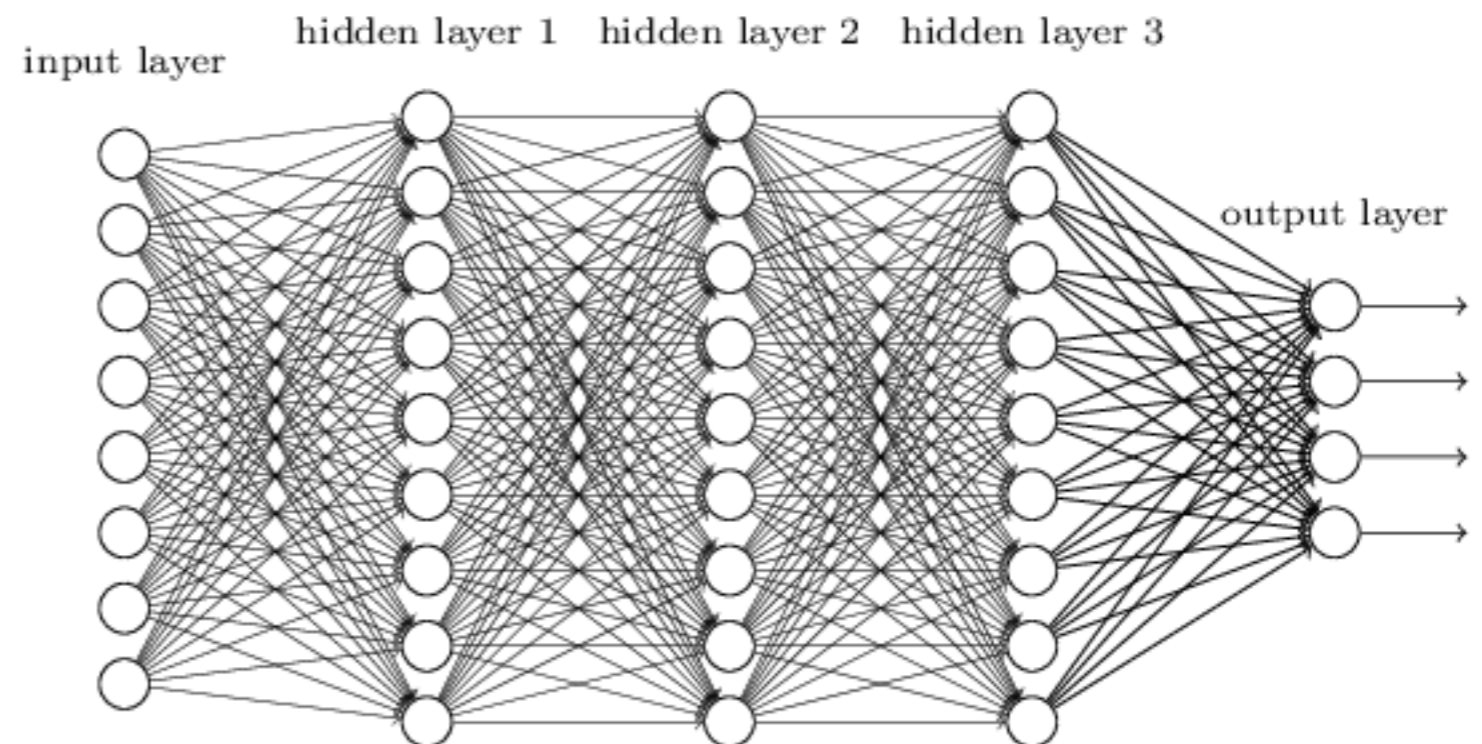
In calculus of variations, the optimization is over all functions: $\hat{s} = \operatorname{argmin}_s L[s]$

- In applied calculus of variations, we consider a highly flexible family of functions s_θ and optimize: i.e. $\hat{\theta} = \operatorname{argmin}_\theta L[s_\theta]$ $\hat{s} \approx s_{\hat{\theta}}$
- 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



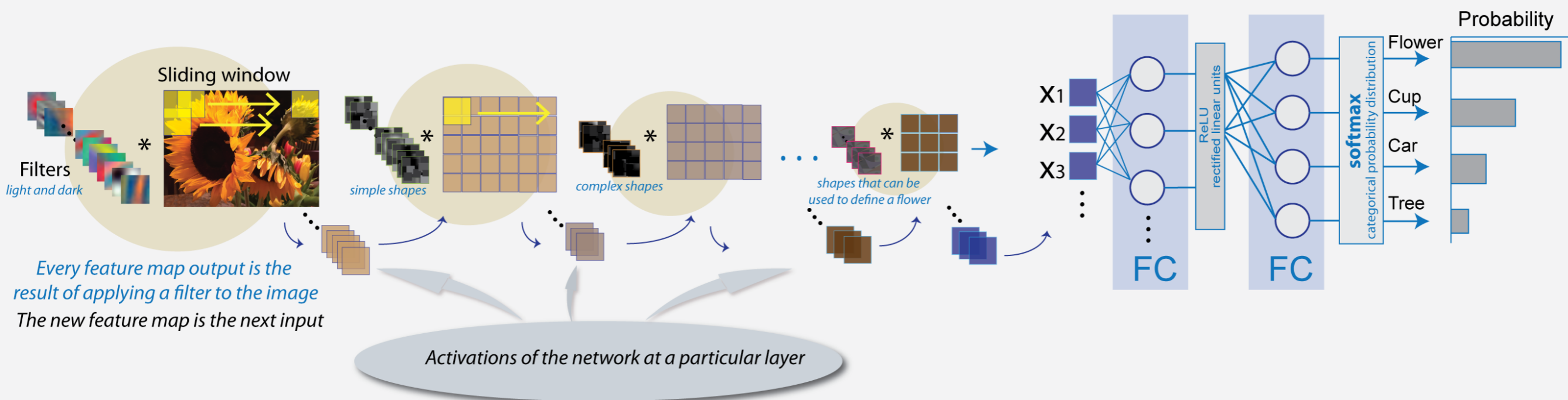
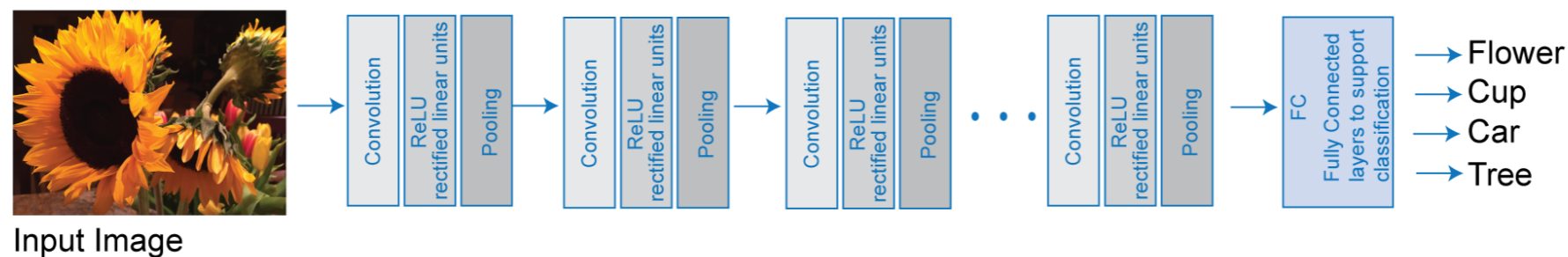
Deep neural network



CONVOLUTIONAL NEURAL NETWORKS

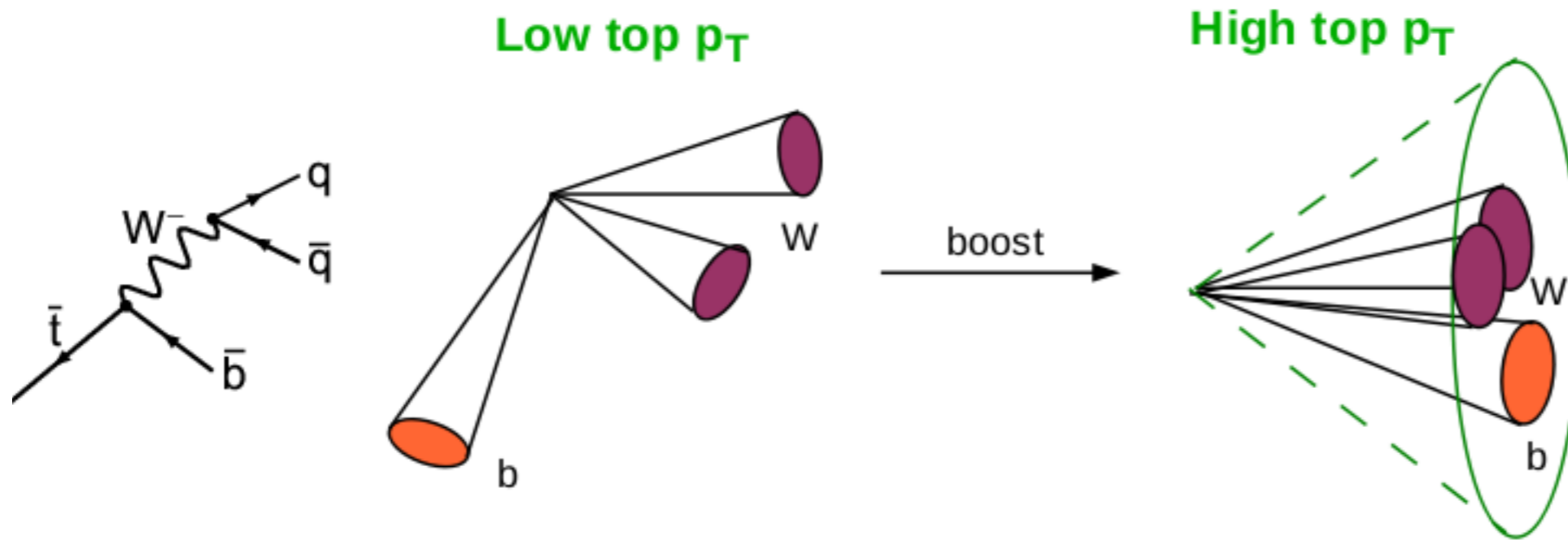
Another major idea of deep learning: convolutional filters

- the world is compositional \Rightarrow hierarchical architecture
- images are translationally invariant \Rightarrow shared weights



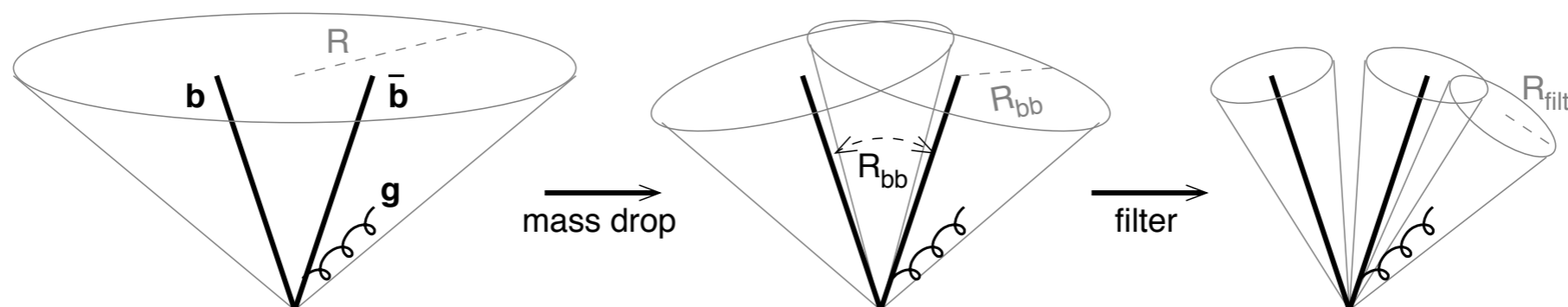
JET SUBSTRUCTURE

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



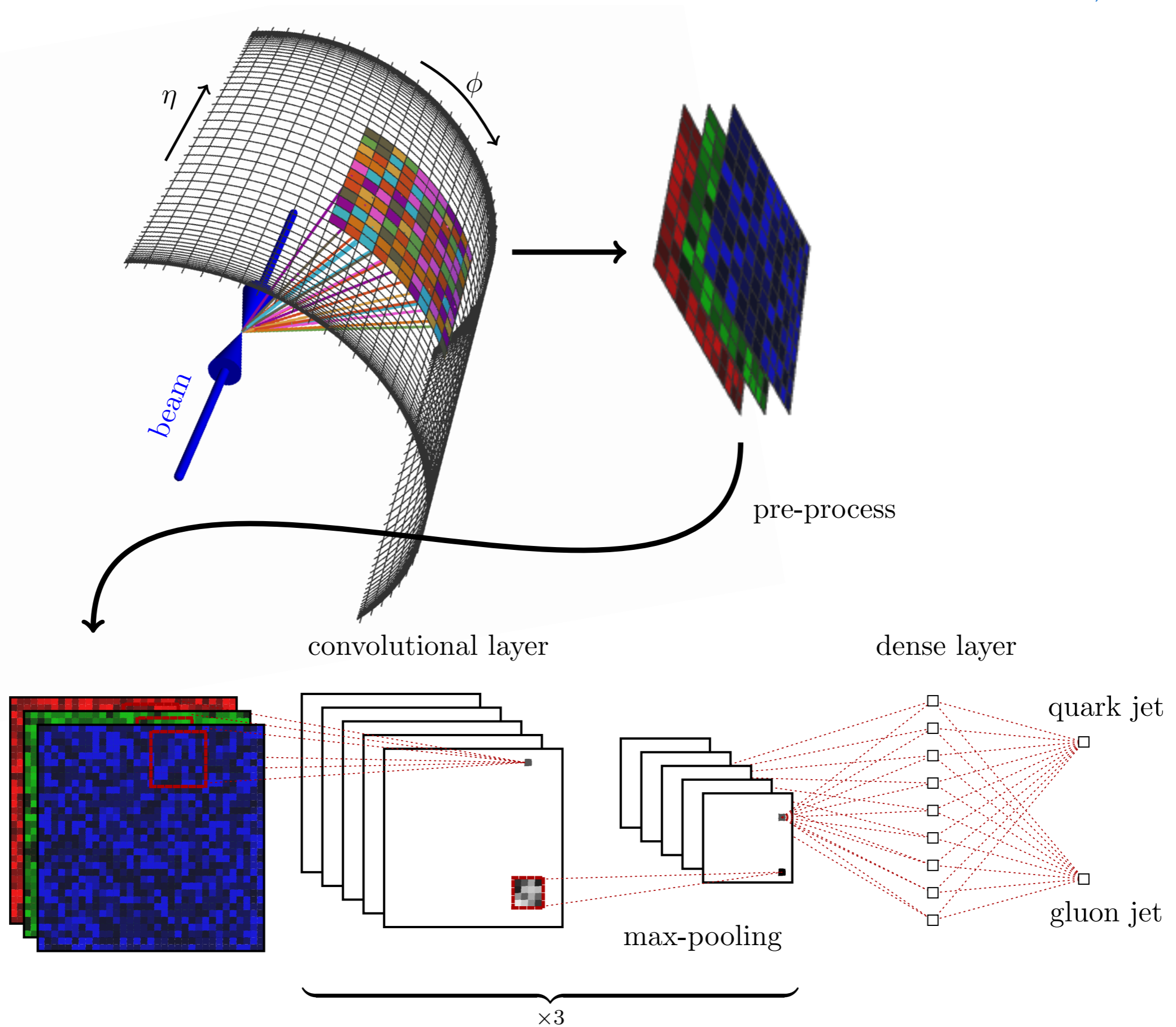
JET IMAGES

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 IMAGES

Oliveira, et. al arXiv:1511.05190

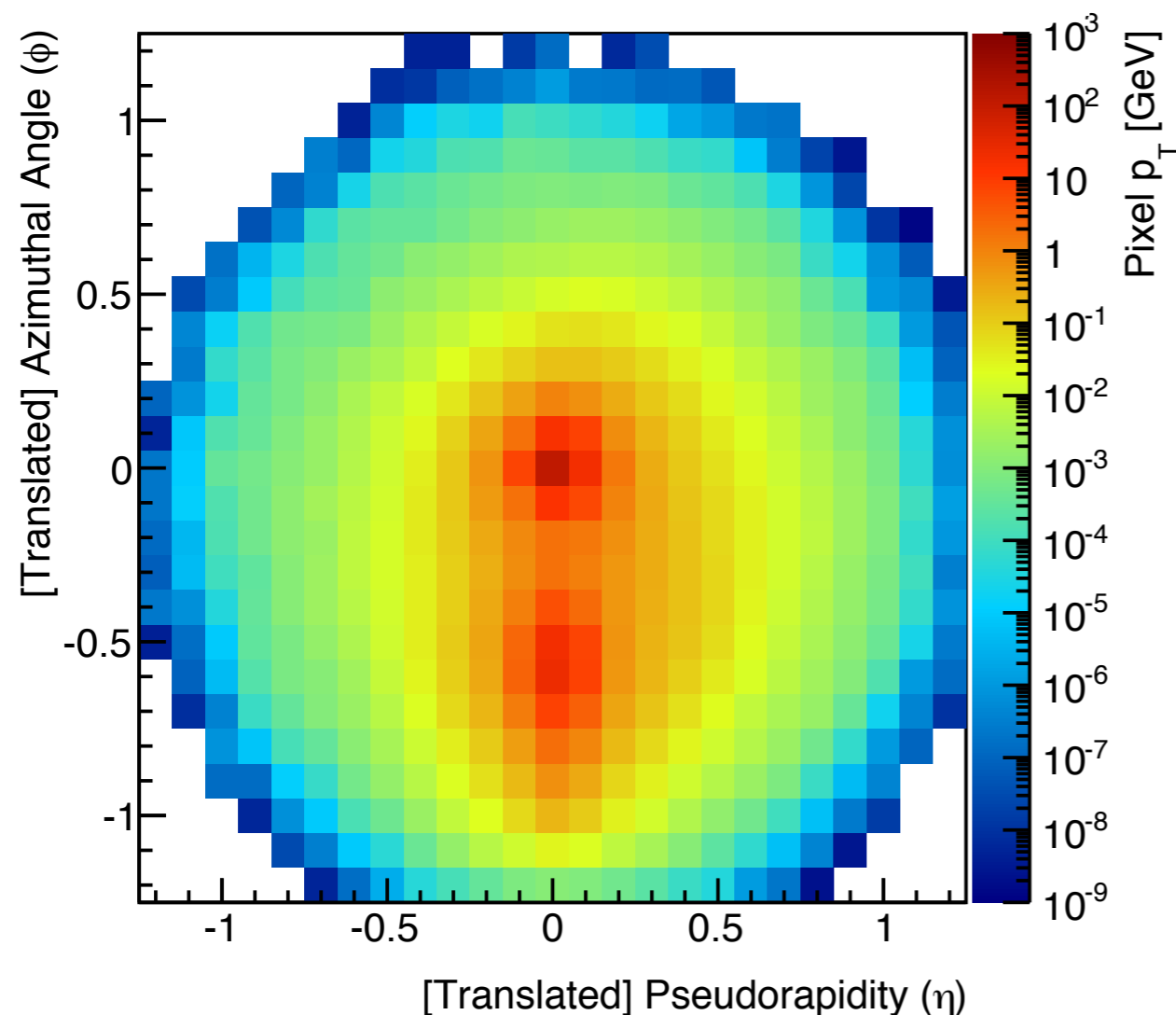
Whiteson, et al arXiv:1603.09349

Dawe, et al arXiv:1609.00607

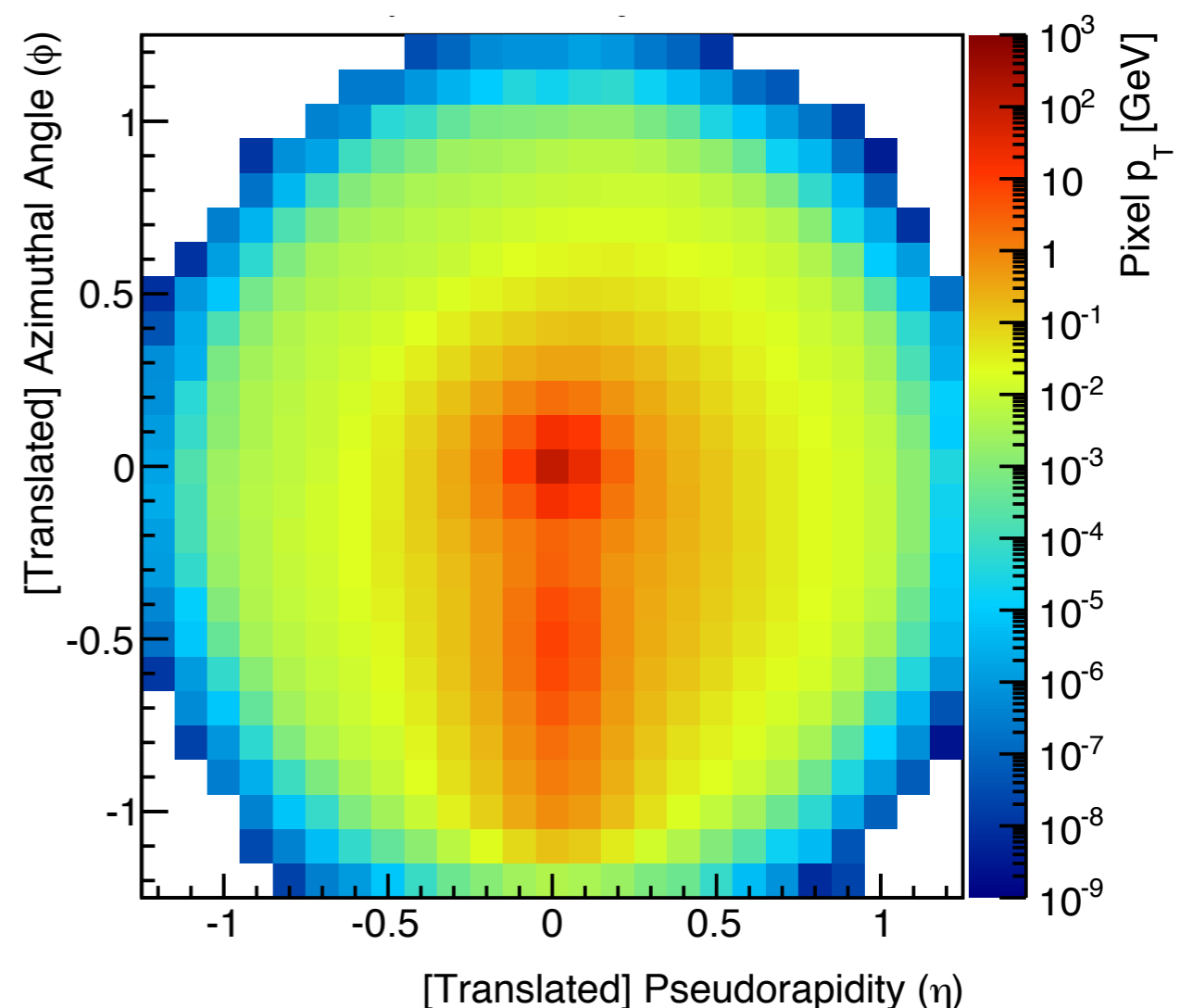
Apply deep learning algorithms to classify to “jet images”

- good results (based on fast simulation & idealized uniform calorimeter)
- preprocessed to mod out symmetries in the data
- discretization into images loses information

Average Boosted W Jet ($y=1$)



Average QCD Jet ($y=0$)



JETS AS A GRAPH

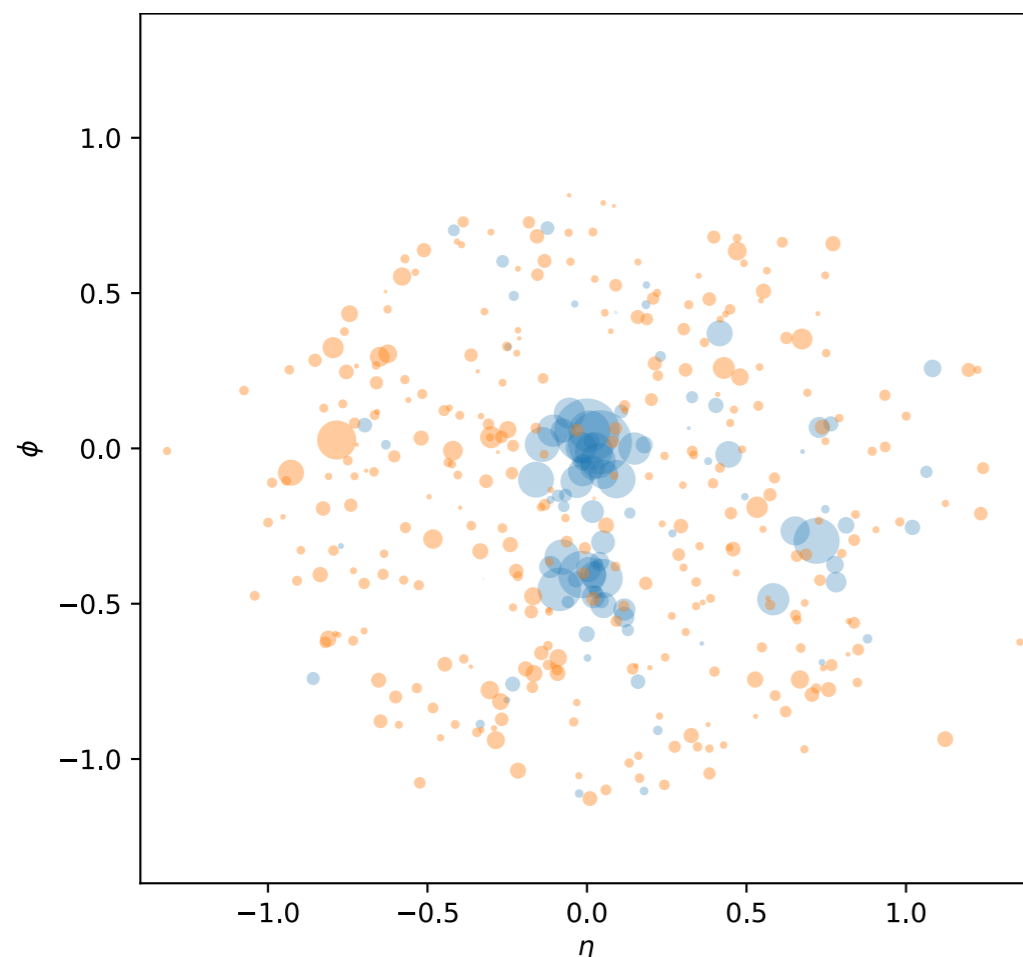
Using message passing neural networks over a fully connected graph on the particles

- Two approaches for adjacency matrix for edges
 - inject physics knowledge by using d_{ij} of jet algorithms
 - learn adjacency matrix and export new jet algorithm

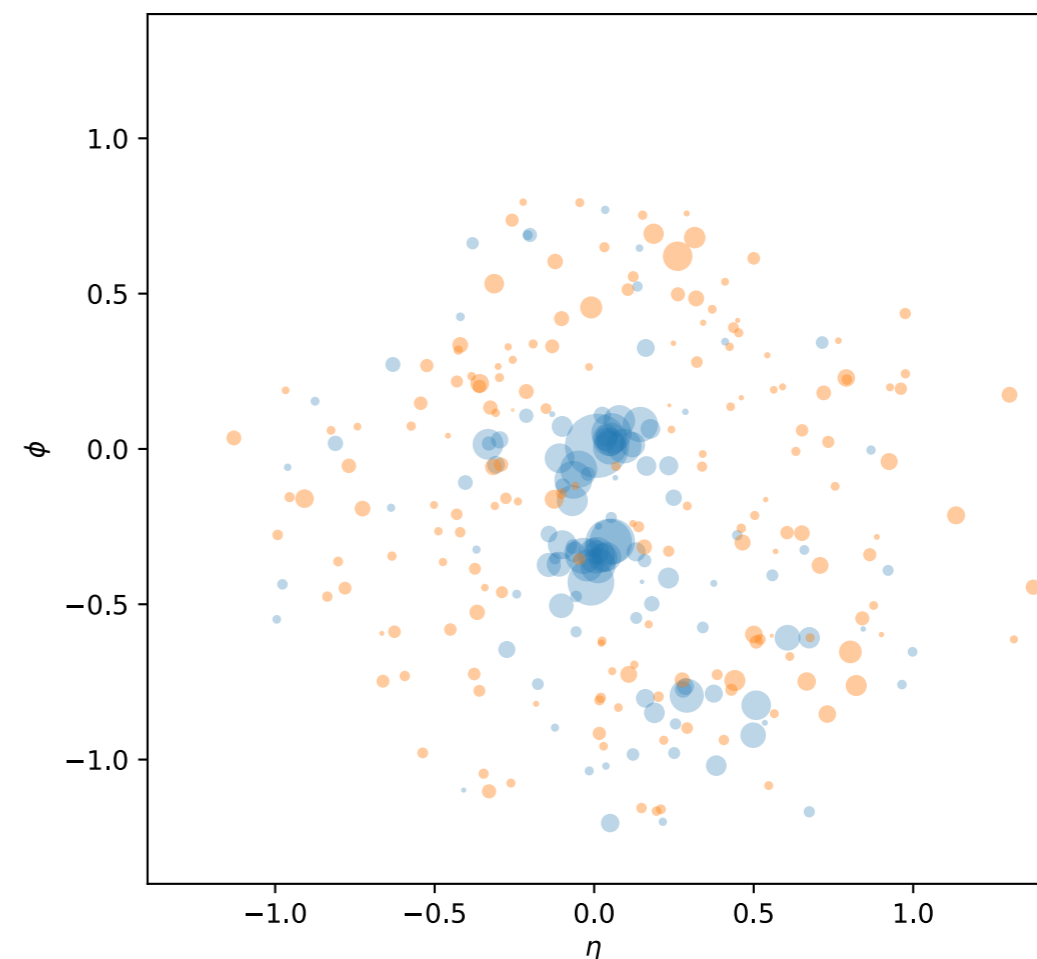


Isaac Henrion

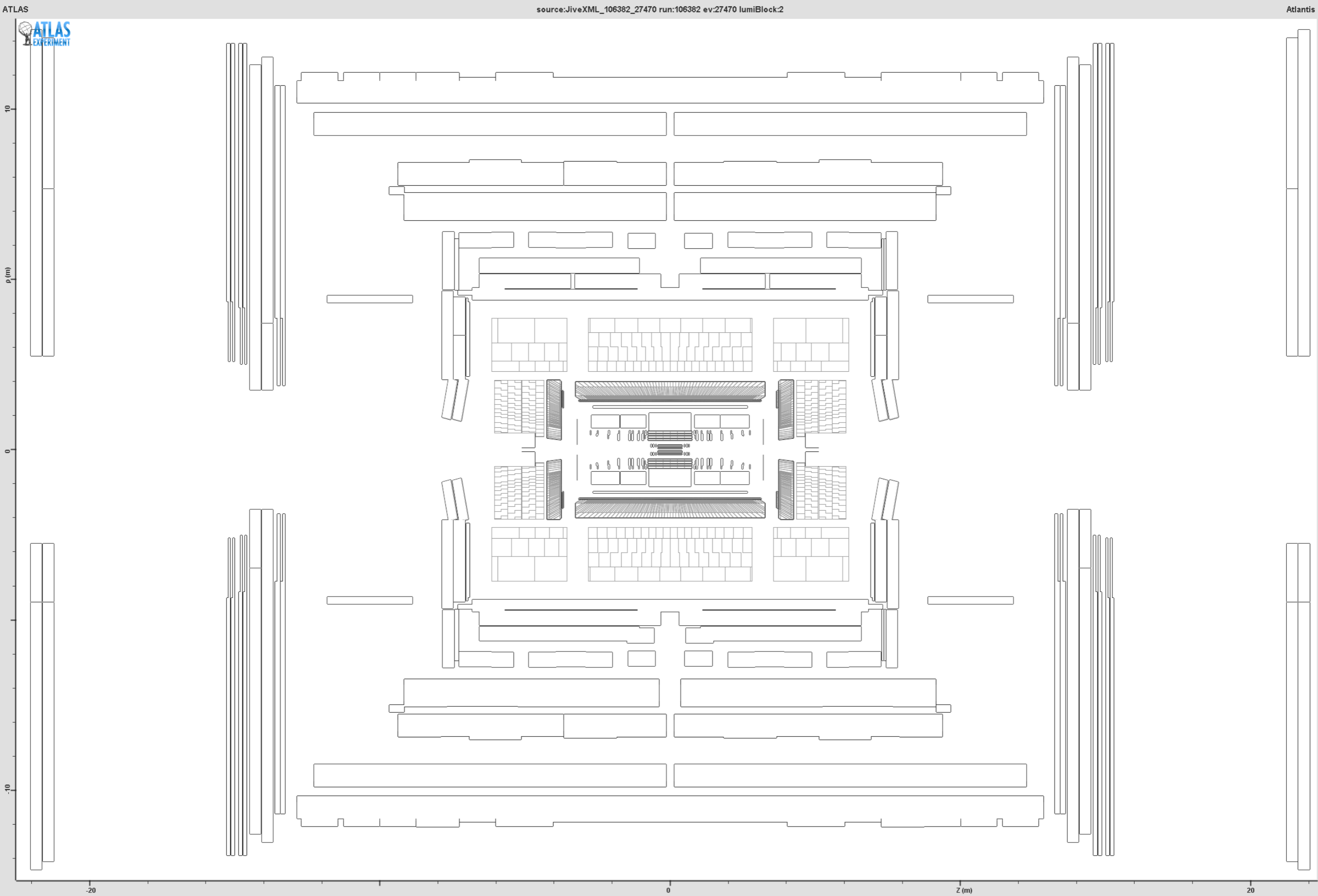
Example Boosted W Jet ($y=1$)



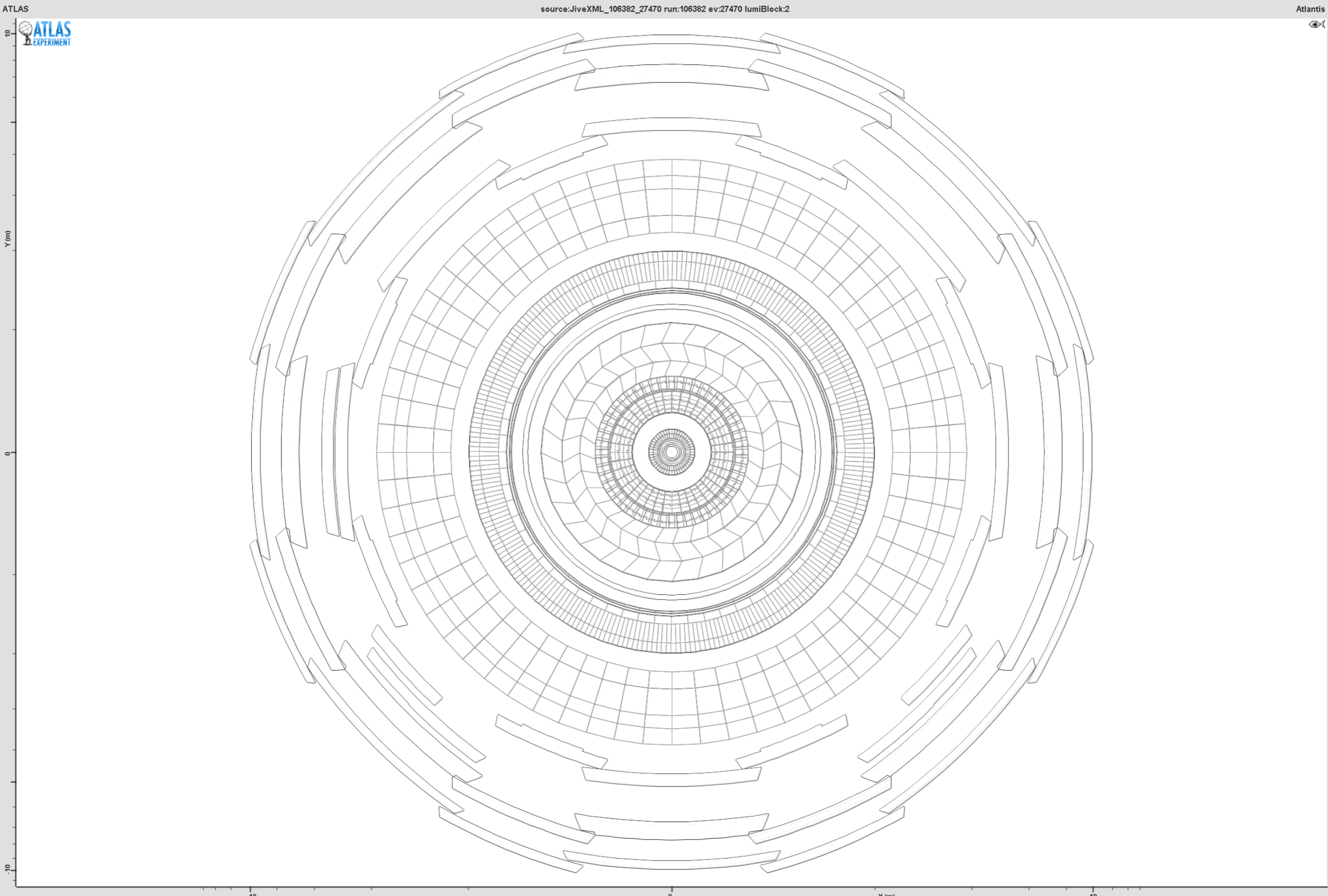
Example QCD Jet ($y=0$)



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!

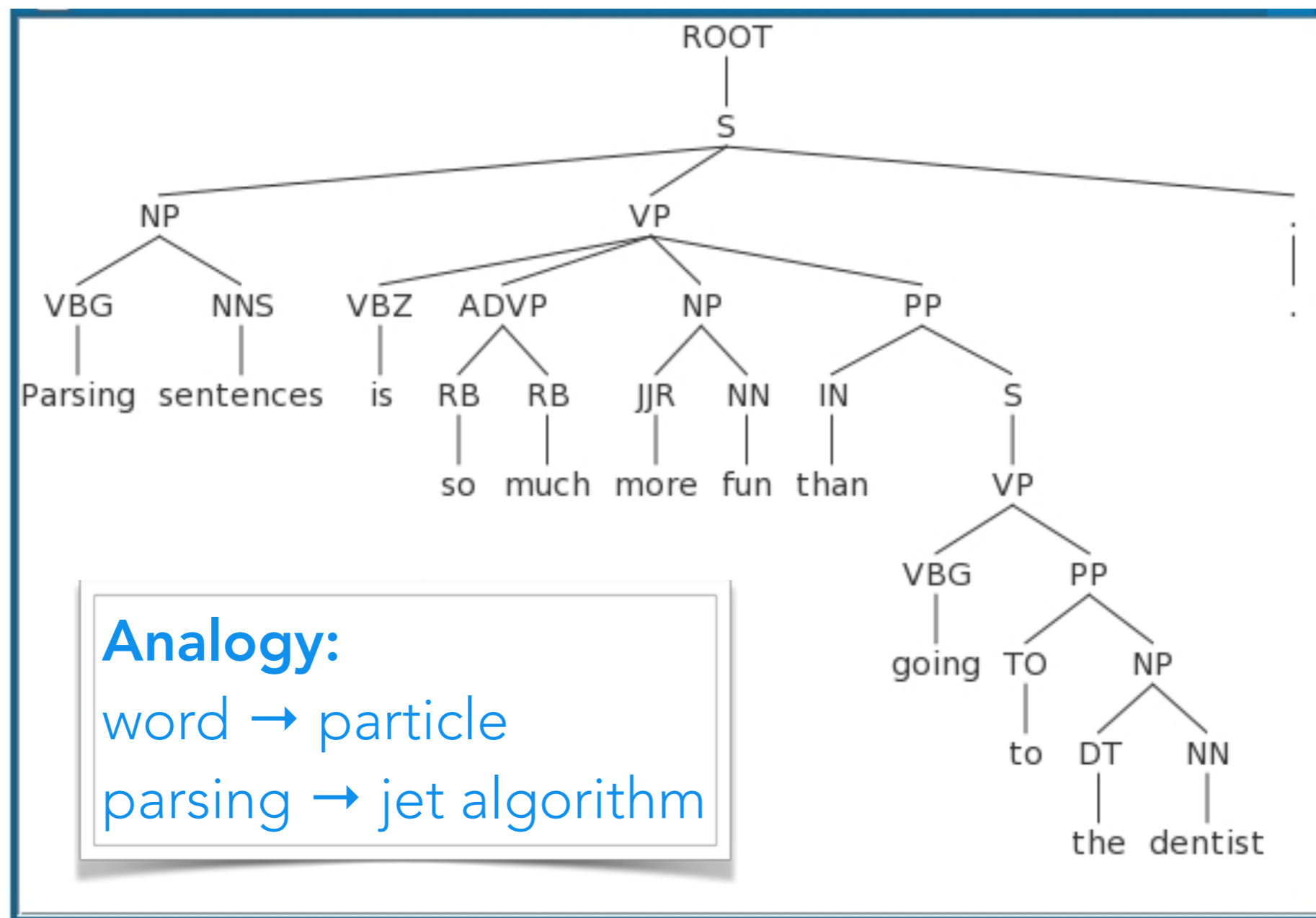



Figure 1: A diagram illustrating the structure of the proposed model. The diagram shows a hierarchical tree structure for k_t and $\text{anti-}k_t$. The k_t tree has a root node (red) with two children (orange), which further branch into more nodes (yellow). The $\text{anti-}k_t$ tree is a single vertical chain of nodes (red, orange, yellow). The diagram is labeled with k_t and $\text{anti-}k_t$. Below the diagram, it says $y=0, y_{\text{pred}}=0.1529$.

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

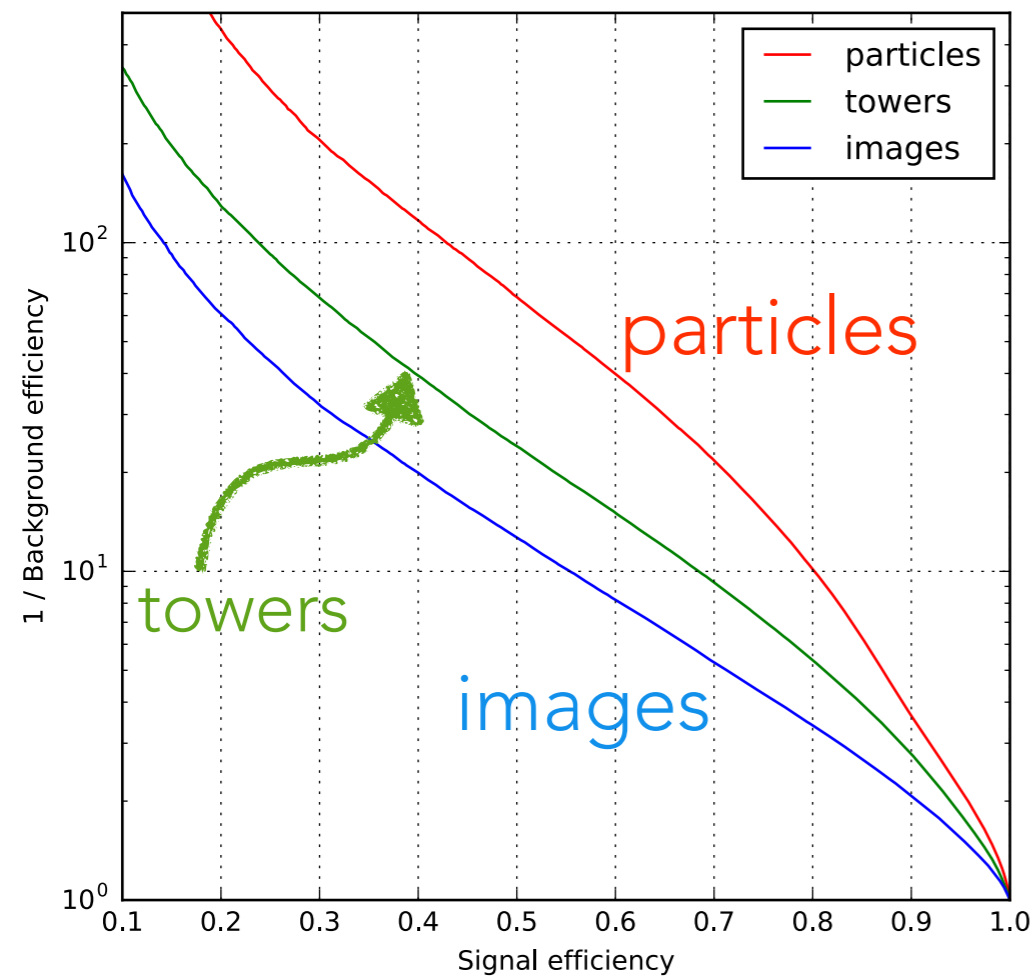
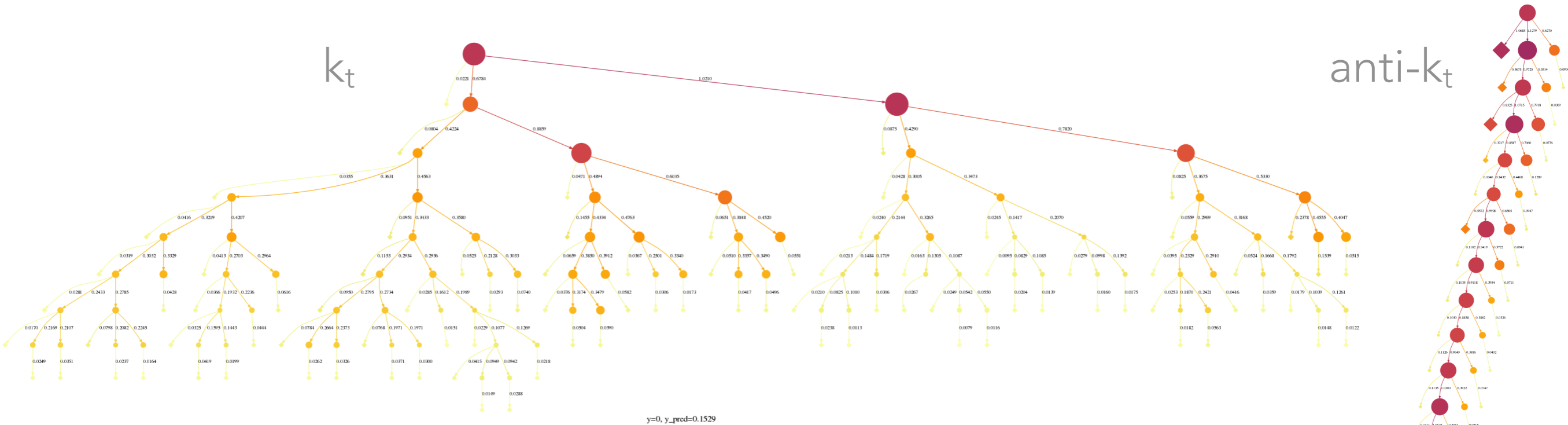


y=0, y_pred=0.2148

- 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



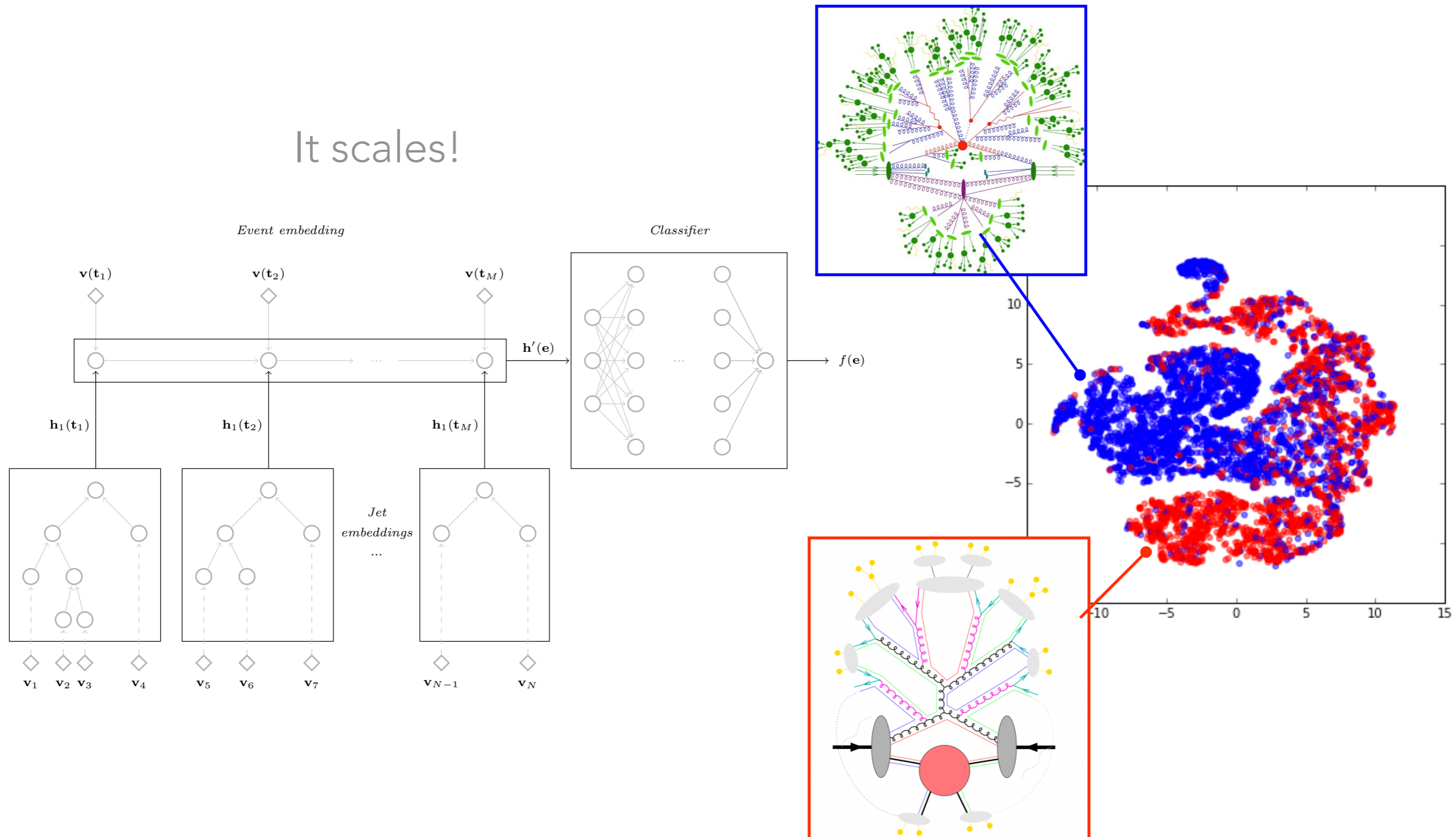
- W-jet tagging example using data from Dawe, et al arXiv:1609.00607
- down-sampling by projecting into images loses information
- RNN needs much less data to train!



HIERARCHICAL MODEL FOR THE ENTIRE EVENT

particle embedding \rightarrow jet embedding \rightarrow event embedding \rightarrow classifier

It scales!

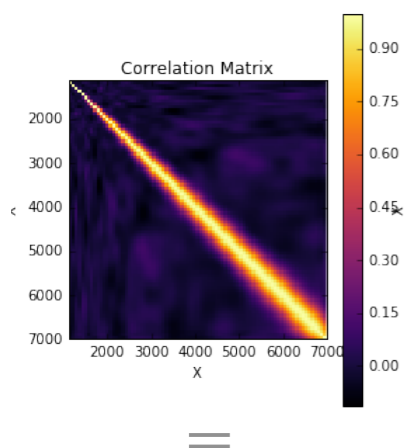


PHYSICS-AWARE MACHINE LEARNING

We can inject our knowledge of physics into the variational family

Physics-aware Gaussian Processes

arXiv:1709.05681



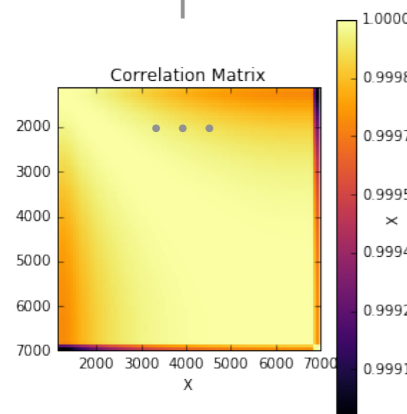
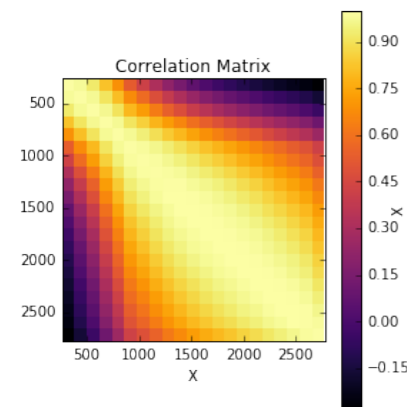
Final Kernel =

Poisson fluctuations

+ Mass Resolution

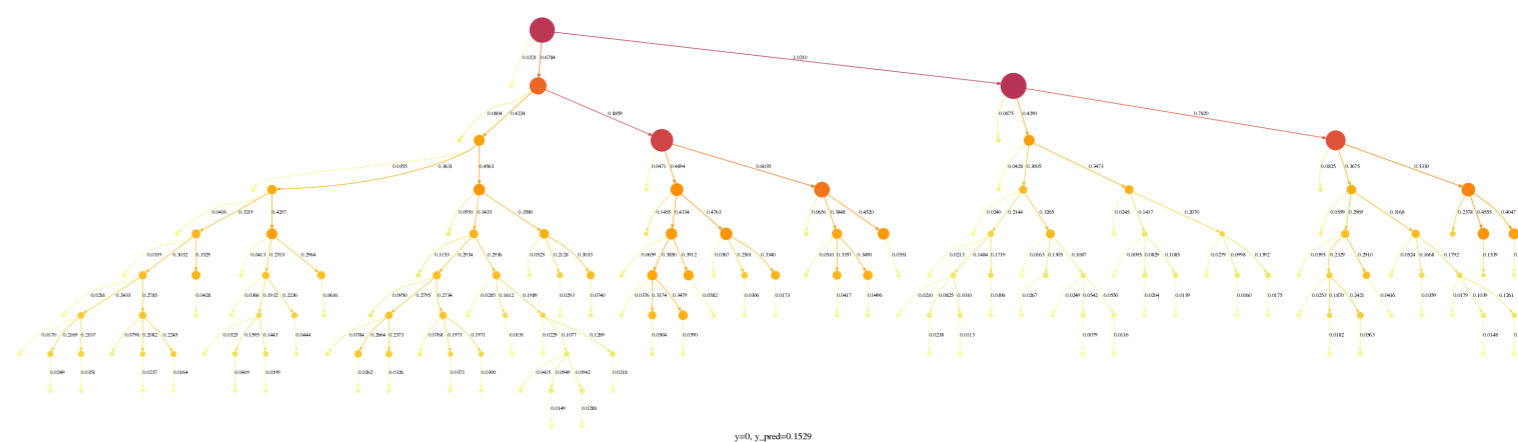
+ Parton Density Functions

+ Jet Energy Scale



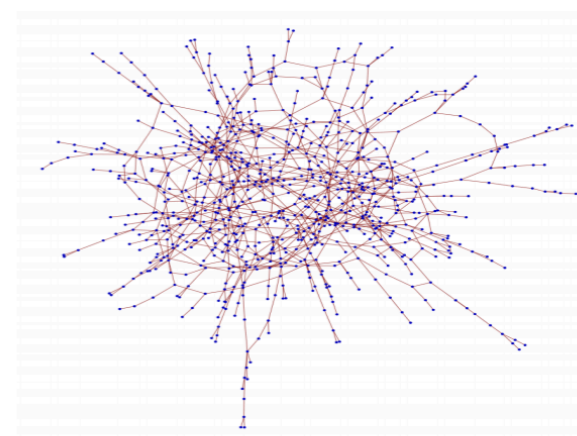
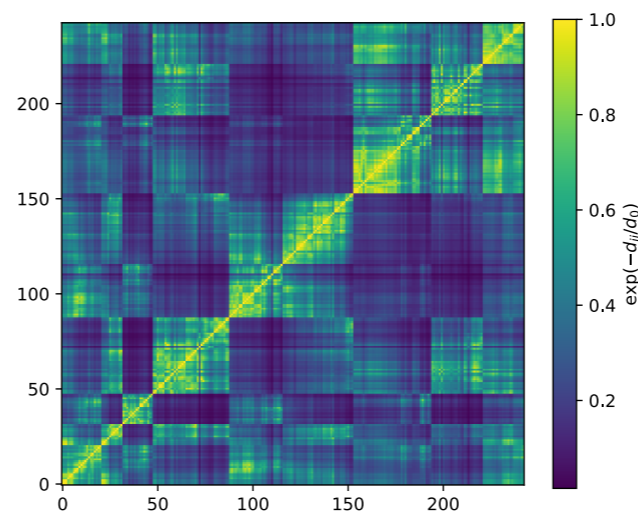
QCD-Aware recursive neural networks

arXiv:1702.00748



QCD-Aware graph convolutional neural networks

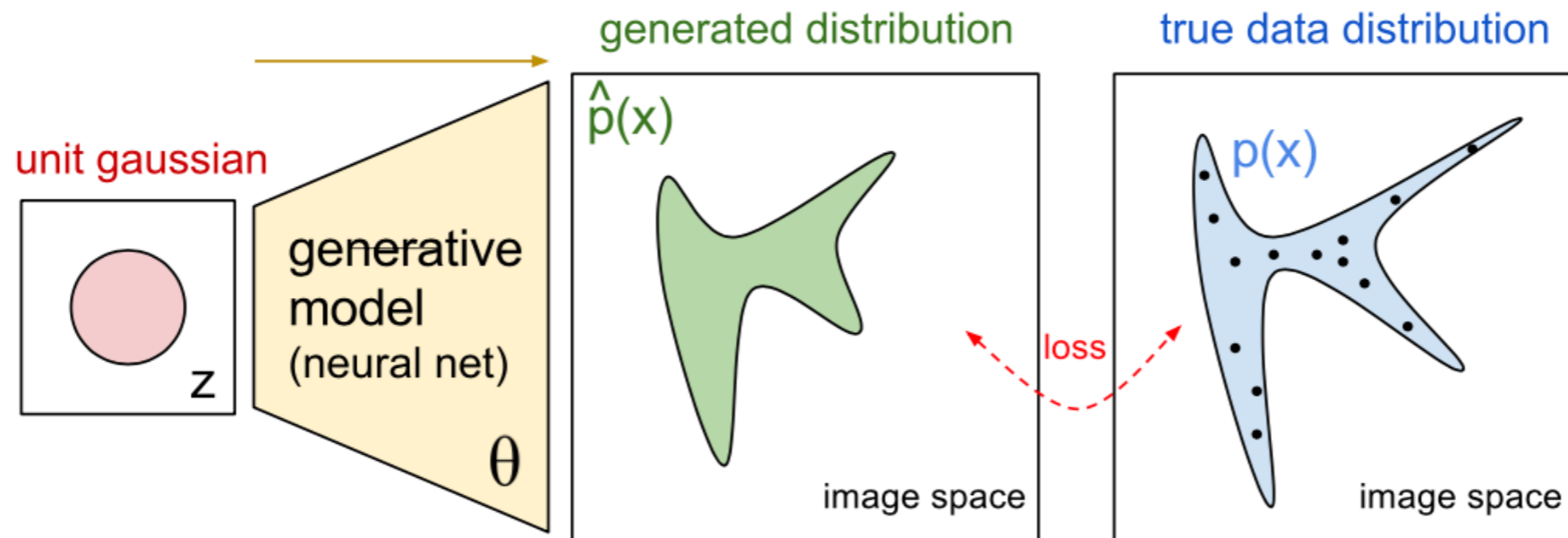
NIPS2017 workshop



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

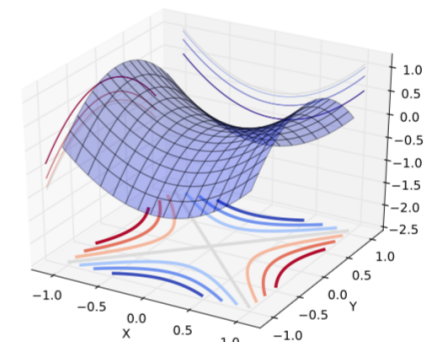
Adversarial Training (not just for GANs)

GENERATIVE ADVERSARIAL NETWORKS



- 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.

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



Leo is G

Tom is D

Adversarial Variational Optimization of Non-Differentiable Simulators

Gilles Louppe¹ and Kyle Cranmer¹¹New 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 min-max problem by minimizing variational upper bounds of the two adversarial objectives. Effectively, the procedure results in learning a proposal distribution over simulator parameters, such that the corresponding marginal distribution of the generated data matches the observations. We present results of the method with simulators producing both discrete and continuous data.

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!**

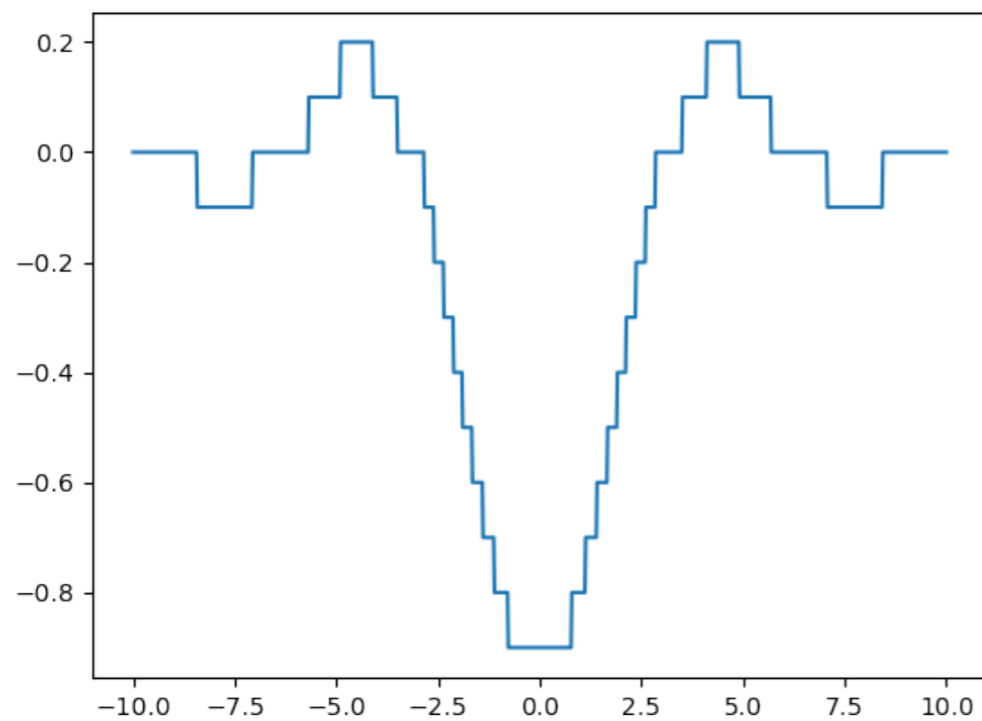
Allows us to efficiently fit / **tune simulation** with stochastic gradient techniques!

Leo is G Tom is D

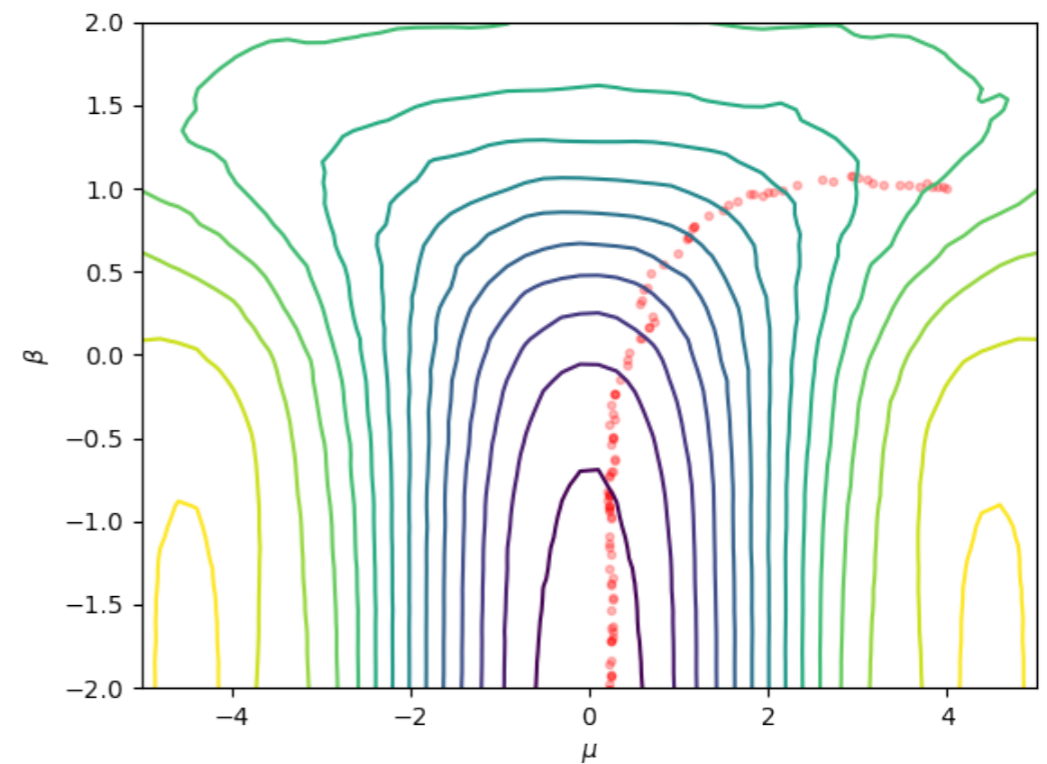
VARIATIONAL OPTIMIZATION

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

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



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



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



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

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

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

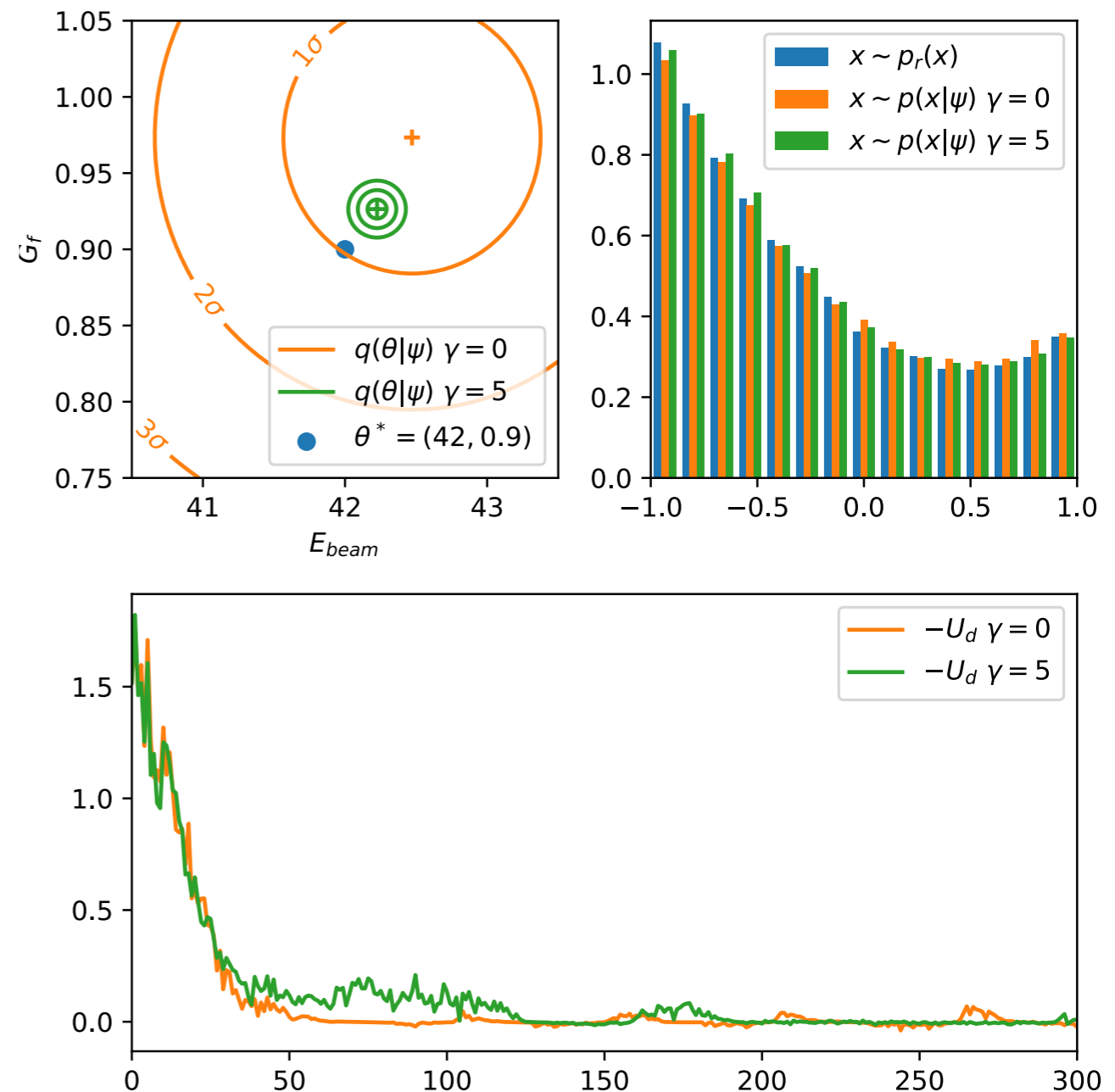
$$U_g = \mathbb{E}_{\boldsymbol{\theta} \sim q(\boldsymbol{\theta}|\boldsymbol{\psi})} [\mathcal{L}_g] \quad (2)$$

respectively over ϕ and ψ .

Effectively sampling from marginal model

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

We use Wasserstein distance, as in WGAN



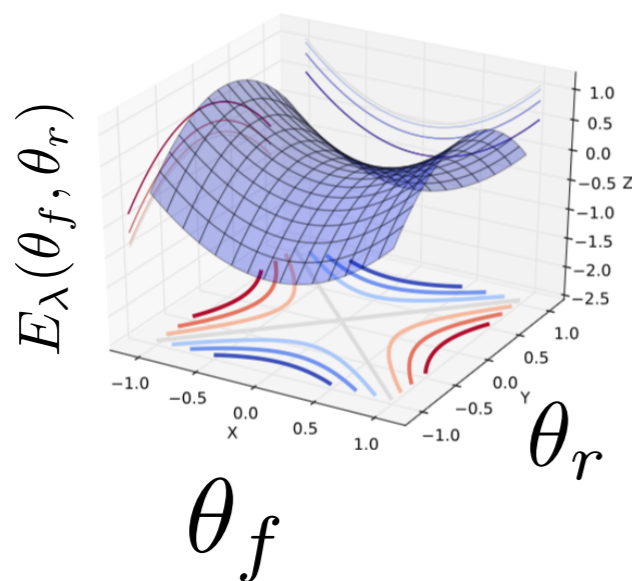
LEARNING TO PIVOT WITH ADVERSARIAL NETWORKS

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

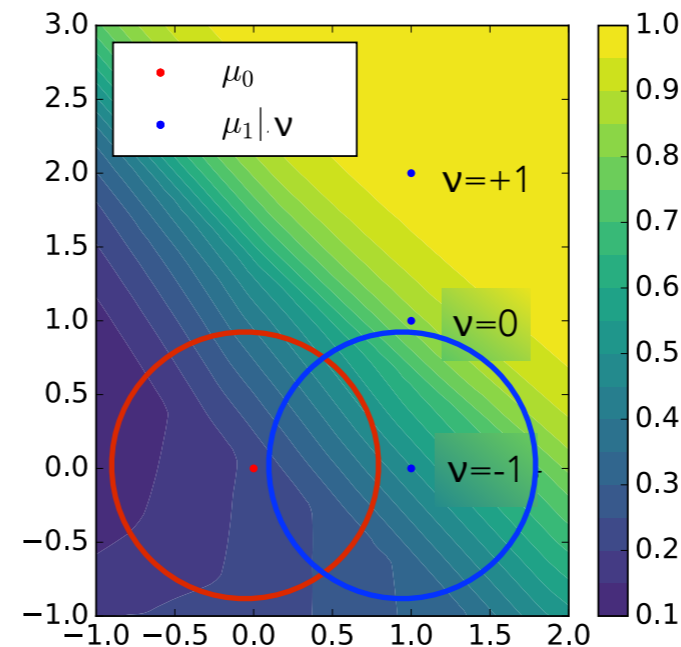
- want classifier output to be insensitive to systematics (nuisance parameter \mathbf{v})
- introduce an **adversary** \mathbf{r} that tries to predict \mathbf{v} based on \mathbf{f} .
- setup as a minimax game:

$$\hat{\theta}_f, \hat{\theta}_r = \arg \min_{\theta_f} \max_{\theta_r} E(\theta_f, \theta_r).$$

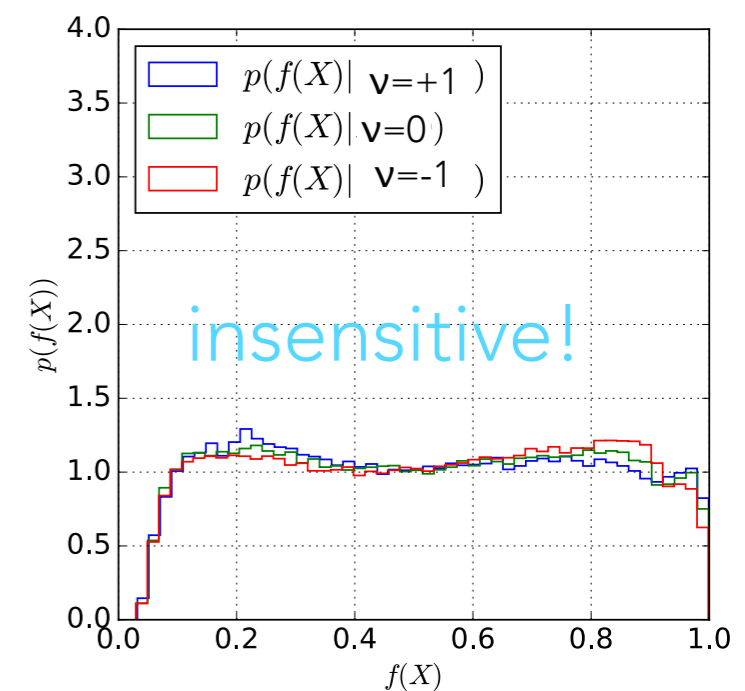
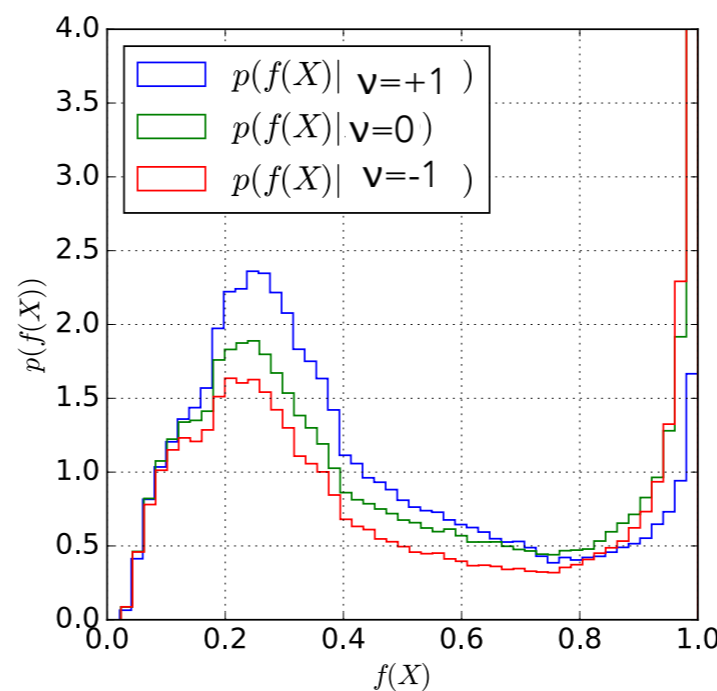
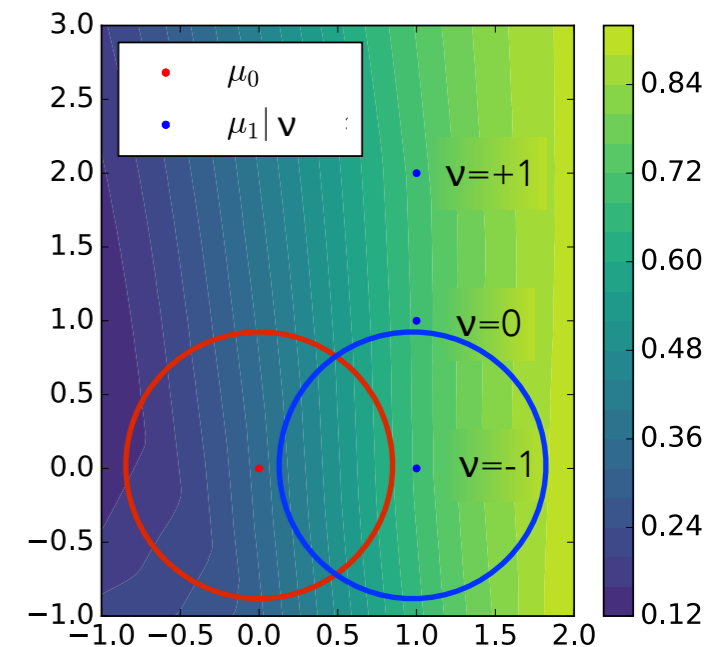
$$E_\lambda(\theta_f, \theta_r) = \mathcal{L}_f(\theta_f) - \lambda \mathcal{L}_r(\theta_f, \theta_r)$$



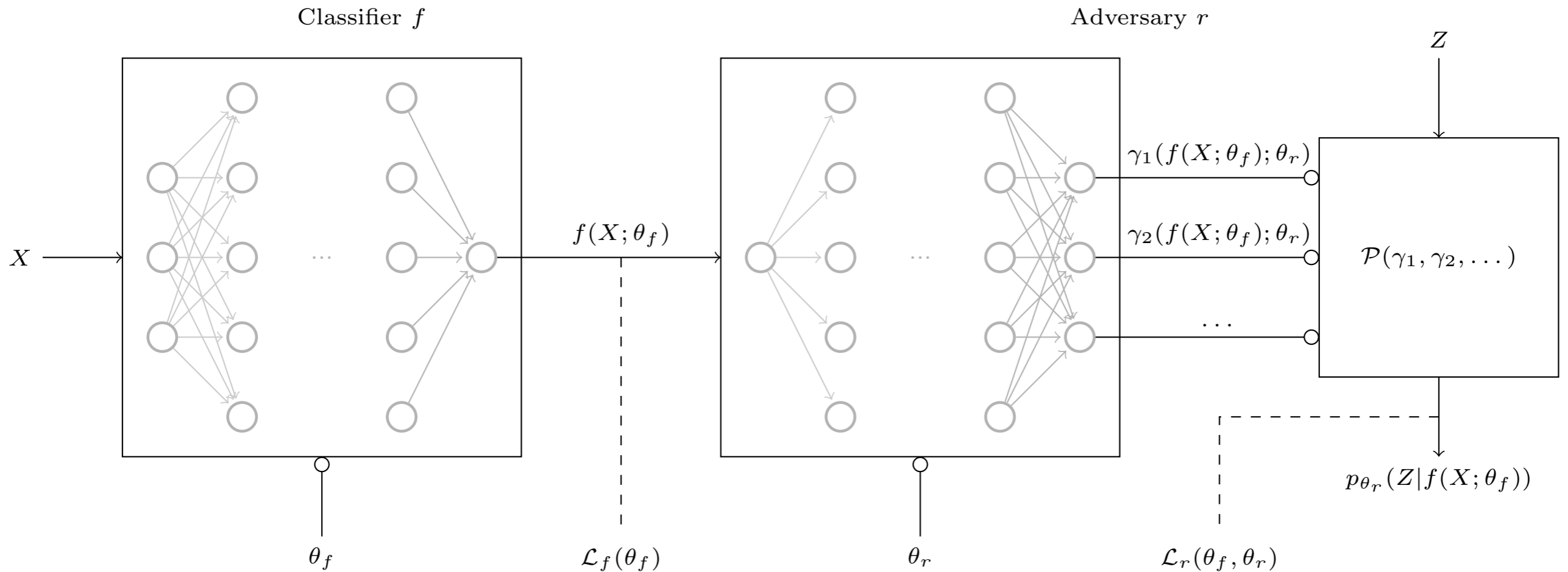
normal training



adversarial training

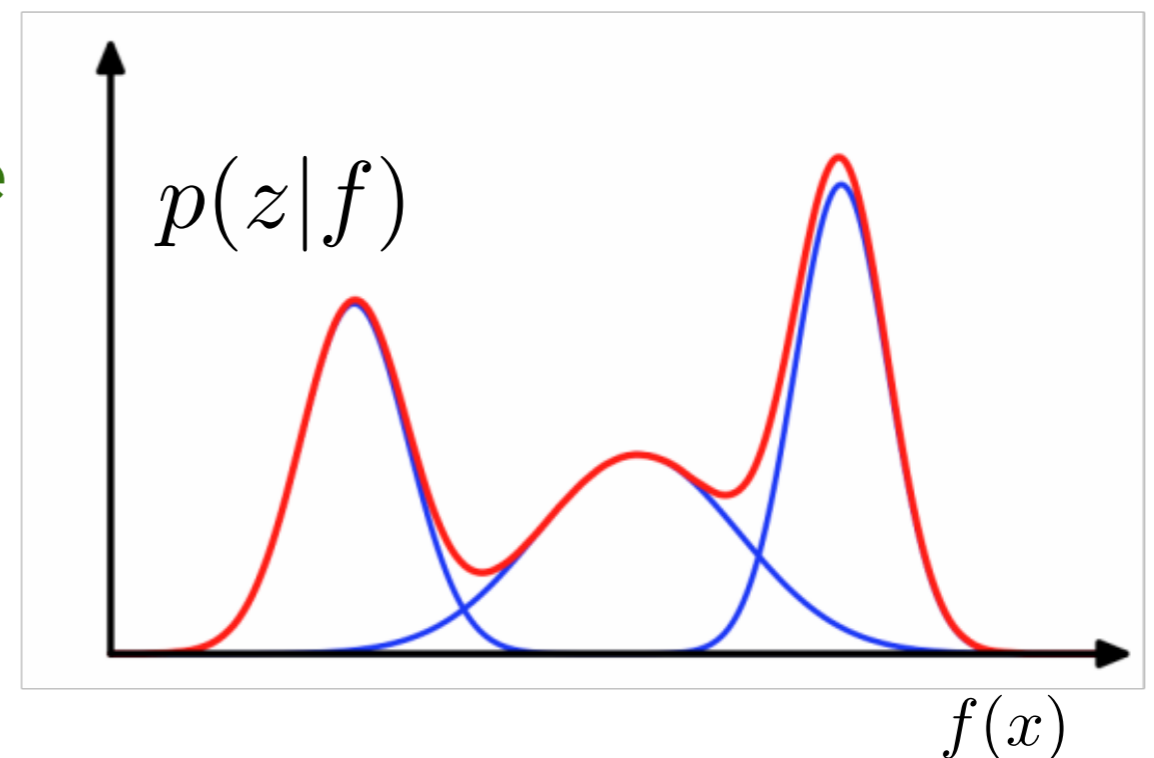


THE ADVERSARIAL MODEL



the $\gamma_1, \gamma_2, \dots$ are the mean, standard deviation, and amplitude for the Gaussian Mixture Model.

- the neural network takes in f and predicts $\gamma_1, \gamma_2, \dots$



AN EXAMPLE

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

An example:

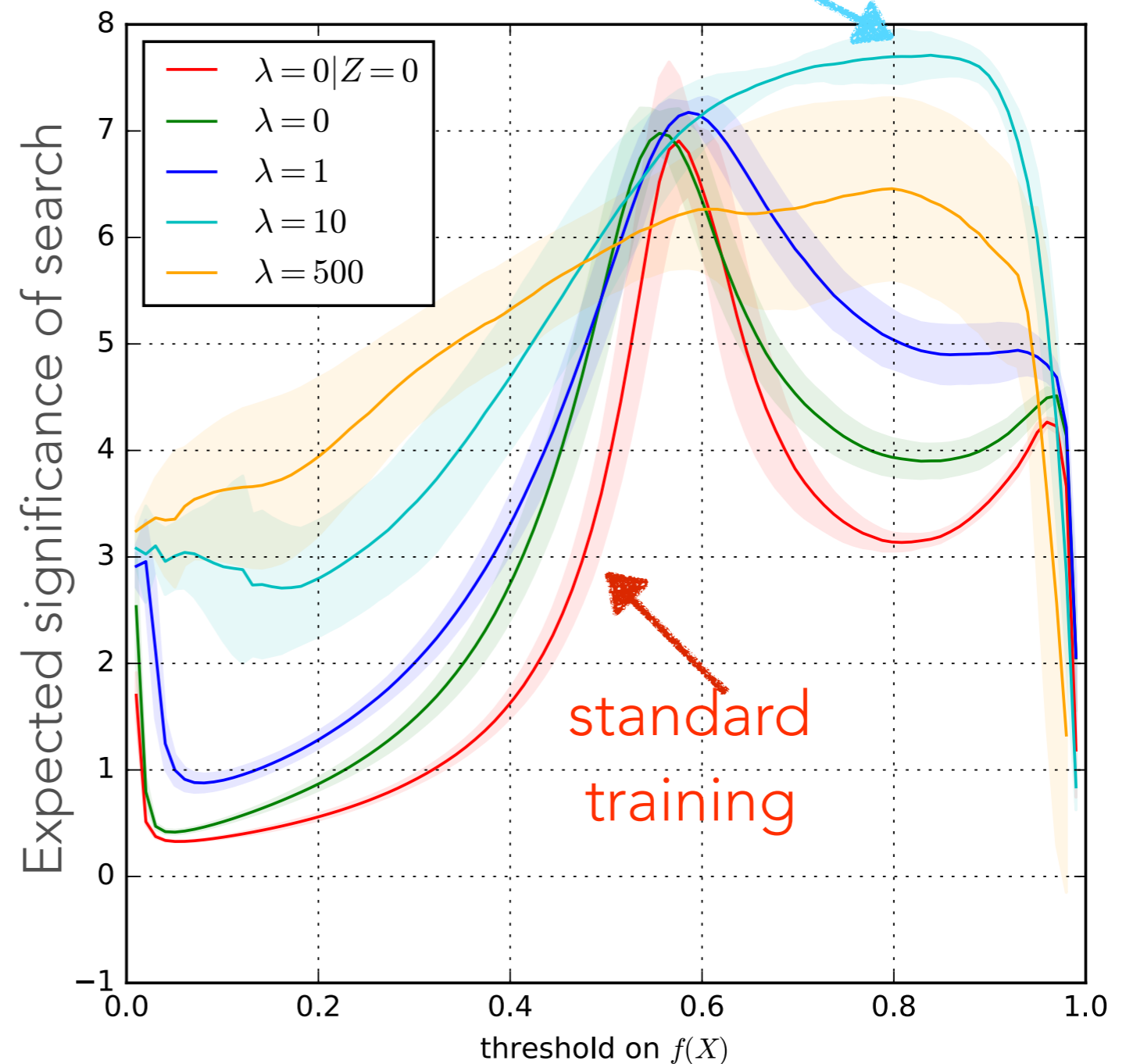
background: 1000 QCD jets
signal: 100 boosted W's

Train W vs. QCD classifier

Pileup as source of uncertainty

Simple cut-and-count analysis with background uncertainty.

optimal tradeoff of classification vs. & robustness

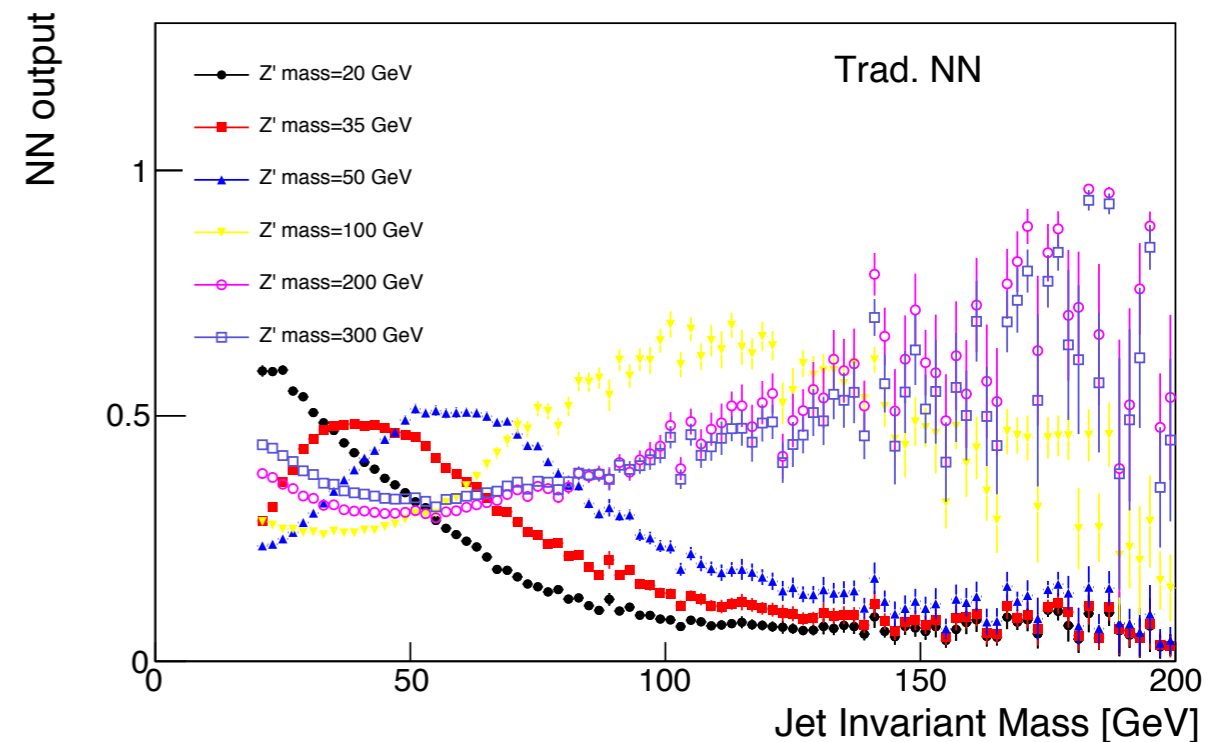
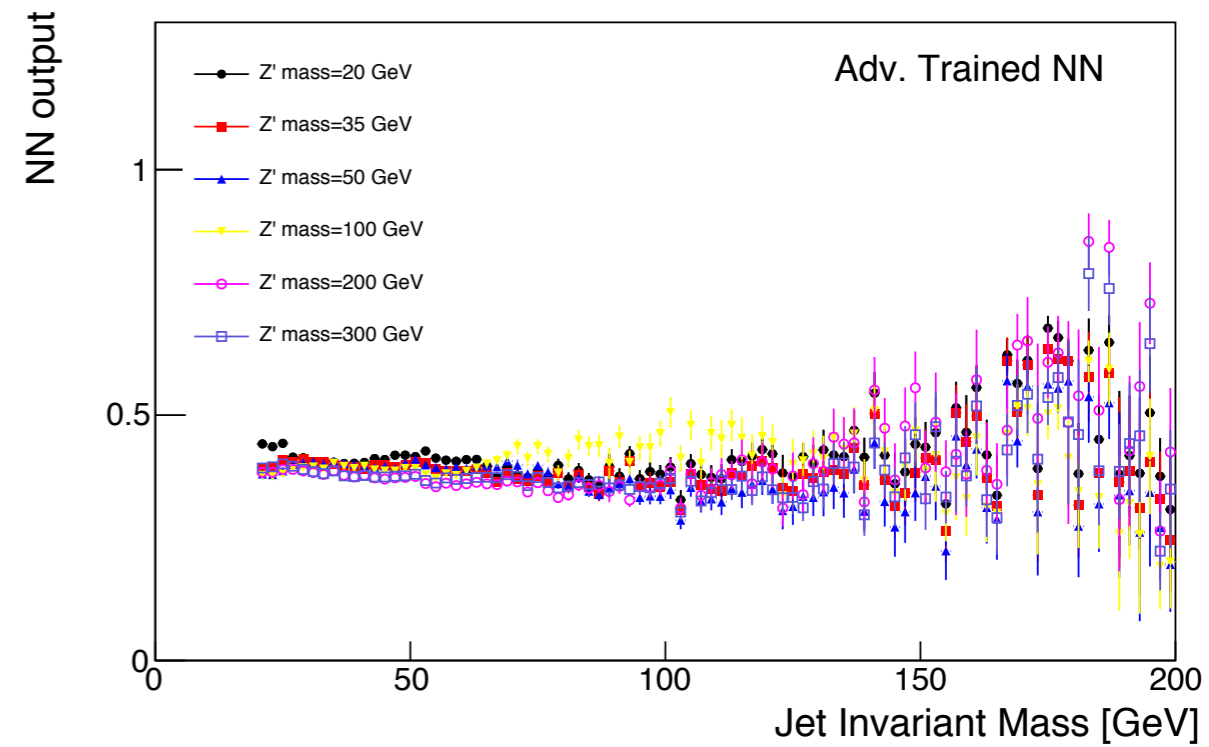


DECORRELATED TAGGERS

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

