Simulation-Based Inference

ÖAW AI Winter School

Lukas Heinrich

Introduction

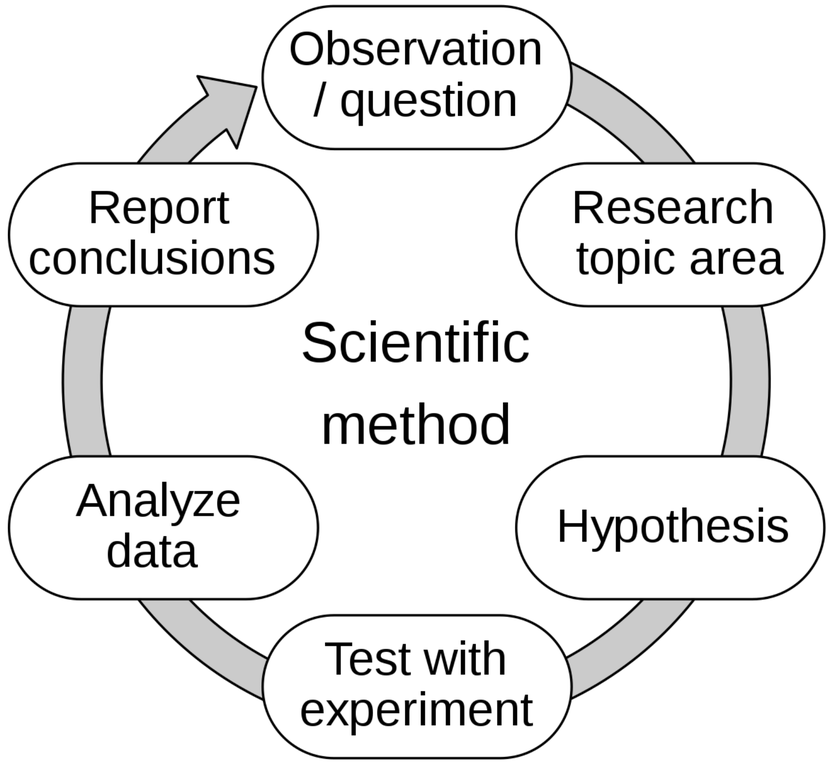
Let’s go back to basics and ask ourselves, and look at the

Scientific Method

and how we analyze data.

Scientific Method

In many ways the scientific method relies on the comparison of hypothesized outcomes with reality

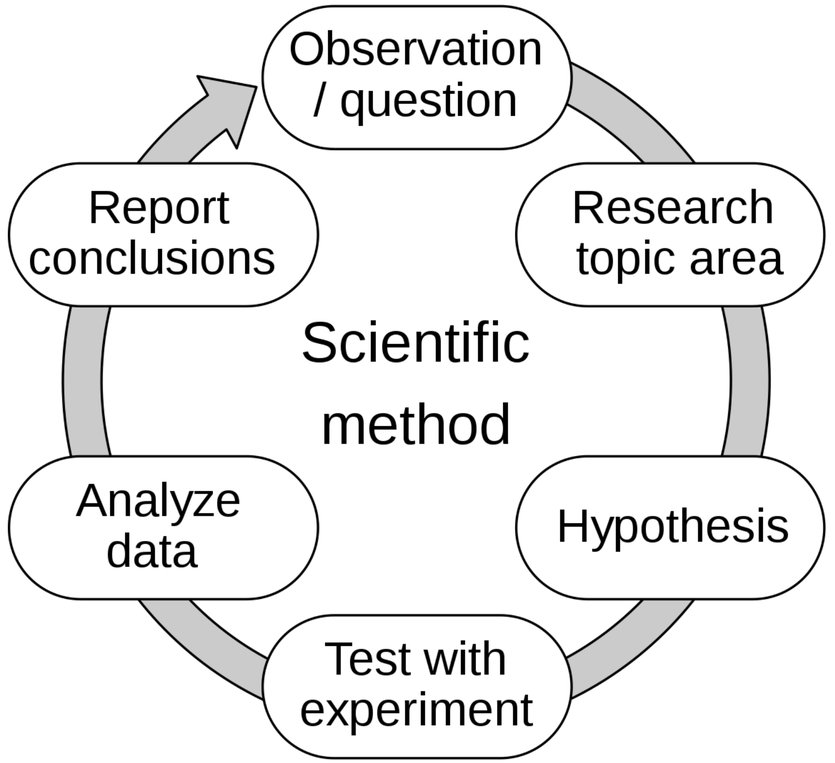


Scientific Method

In many ways the scientific method relies on the comparison of hypothesized outcomes with reality

it's often not the full story, but it's the idealized picture

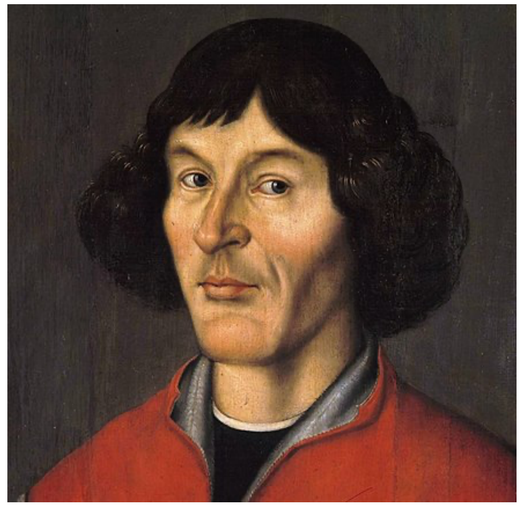
- **sometimes the data is taken without a hypothesis leading to "unexpected results", etc…**



Successes

Claudius Ptolemy

2nd century



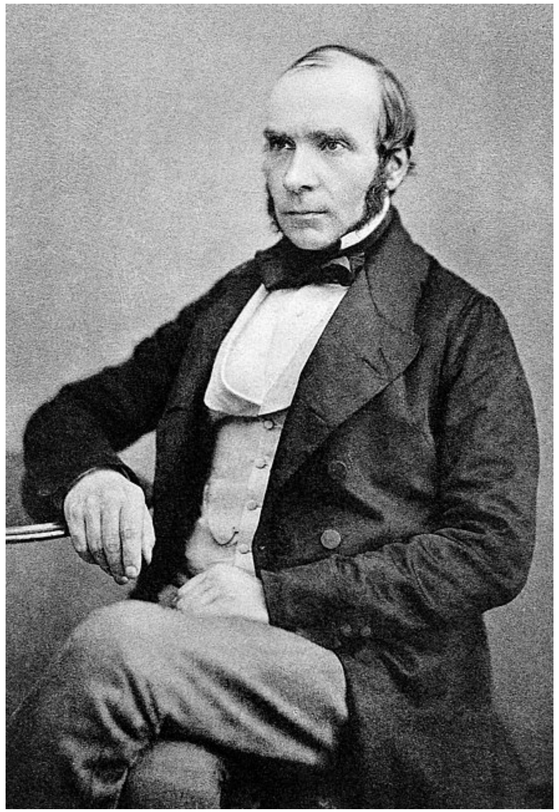
Nikolaus Copernicus

15th century

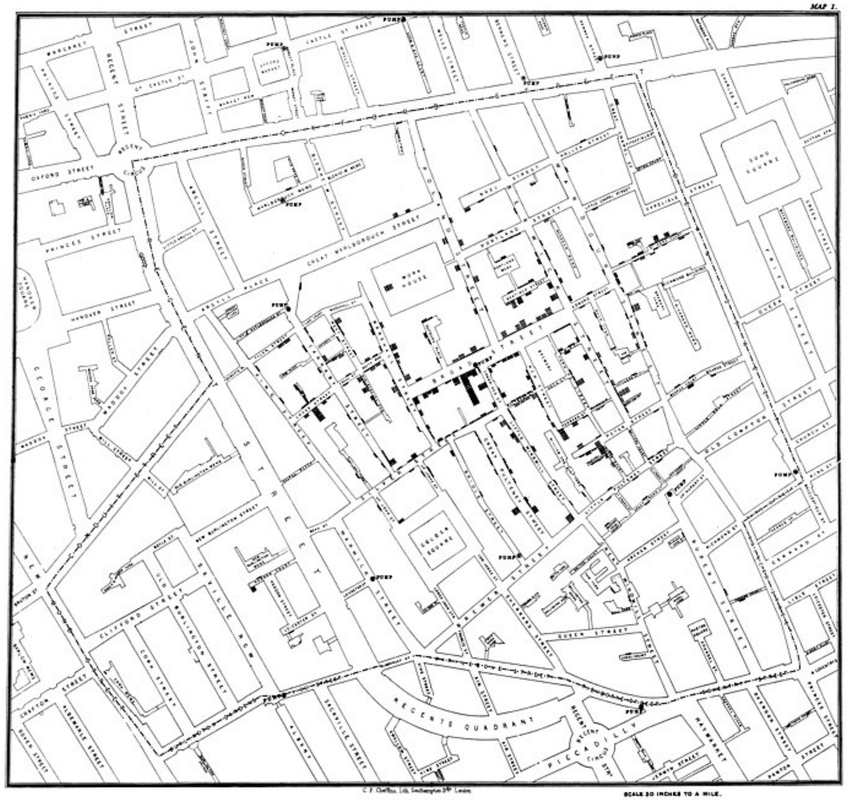
Johannes Kepler

17th century

Successes



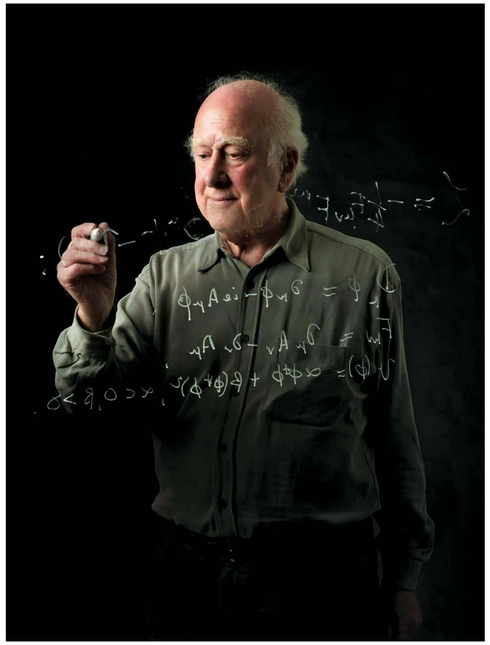
John Snow

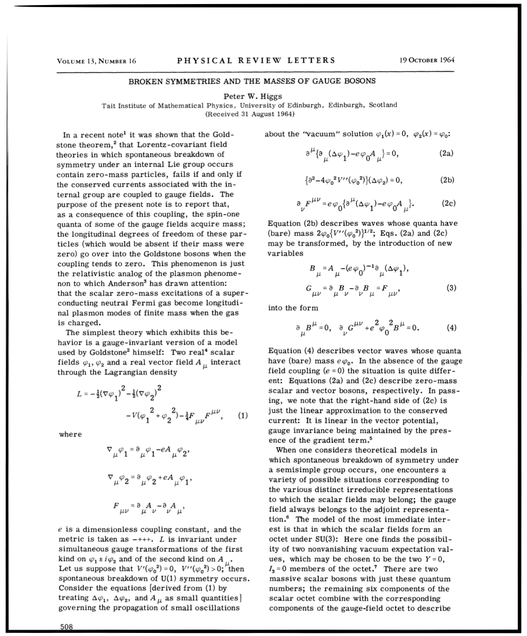


Cholera Outbreak in London

Germ Theory

Successes





VOLUME 13, NUMBER 16 PHYSICAL REVIEW LETTERS 19 OCTOBER 1964

BROKEN SYMMETRIES AND THE MASSES OF GAUGE BOSONS

Peter W. Higgs

Tait Institute of Mathematical Physics, University of Edinburgh, Edinburgh, Scotland

(Received 31 August 1964)

In a recent note[1] it was shown that the Goldstone theorem,[2] that Lorentz-covariant field theories in which spontaneous breakdown of symmetry under an internal Lie group occurs contain zero-mass particles, fails if and only if the conserved currents associated with the internal group are coupled to gauge fields. The purpose of the present note is to report that, as a consequence of this coupling, the spin-one quanta of some of the gauge fields acquire mass; the longitudinal degrees of freedom of these particles (which would be absent if their mass were zero) go over into the Goldstone bosons when the coupling tends to zero. This phenomenon is just the relativistic analog of the plasmon phenomenon to which Anderson[3] has drawn attention: that the scalar zero-mass excitations of a superconducting neutral Fermi gas become longitudinal plasmon modes of finite mass when the gas is charged.

The simplest theory which exhibits this behavior is a gauge-invariant version of a model used by Goldstone[2] himself: Two real[4] scalar fields φ_1, φ_2 and a real vector field A_μ interact through the Lagrangian density

$$L = -\tfrac{1}{2}(\nabla\varphi_1)^2 - \tfrac{1}{2}(\nabla\varphi_2)^2 - V(\varphi_1^2+\varphi_2^2) - \tfrac{1}{4}F_{\mu\nu}F^{\mu\nu}, \quad (1)$$

where

$$\nabla_\mu\varphi_1 = \partial_\mu\varphi_1 - eA_\mu\varphi_2,$$

$$\nabla_\mu\varphi_2 = \partial_\mu\varphi_2 + eA_\mu\varphi_1,$$

$$F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu.$$

e is a dimensionless coupling constant, and the metric is taken as $-+++$. L is invariant under simultaneous gauge transformations of the first kind on $\varphi_1 \pm i\varphi_2$ and of the second kind on A_μ. Let us suppose that $V'(\varphi_0^2) = 0$, $V''(\varphi_0^2) > 0$; then spontaneous breakdown of U(1) symmetry occurs. Consider the equations [derived from (1) by treating $\Delta\varphi_1$, $\Delta\varphi_2$, and A_μ as small quantities] governing the propagation of small oscillations about the "vacuum" solution $\varphi_1(x) = 0$, $\varphi_2(x) = \varphi_0$:

$$\partial^\mu(\partial_\mu(\Delta\varphi_1) - e\varphi_0 A_\mu) = 0, \quad (2a)$$

$$(\partial^2 - 4e\varphi_0^2 V''(\varphi_0^2))(\Delta\varphi_2) = 0, \quad (2b)$$

$$\partial_\mu F^{\mu\nu} = e\varphi_0(\partial^\nu(\Delta\varphi_1) - e\varphi_0 A^\nu). \quad (2c)$$

Equation (2b) describes waves whose quanta have (bare) mass $2e\varphi_0(V''(\varphi_0^2))^{1/2}$; Eqs. (2a) and (2c) may be transformed, by the introduction of new variables

$$B_\mu = A_\mu - (e\varphi_0)^{-1}\partial_\mu(\Delta\varphi_1),$$

$$G_{\mu\nu} = \partial_\mu B_\nu - \partial_\nu B_\mu = F_{\mu\nu}, \quad (3)$$

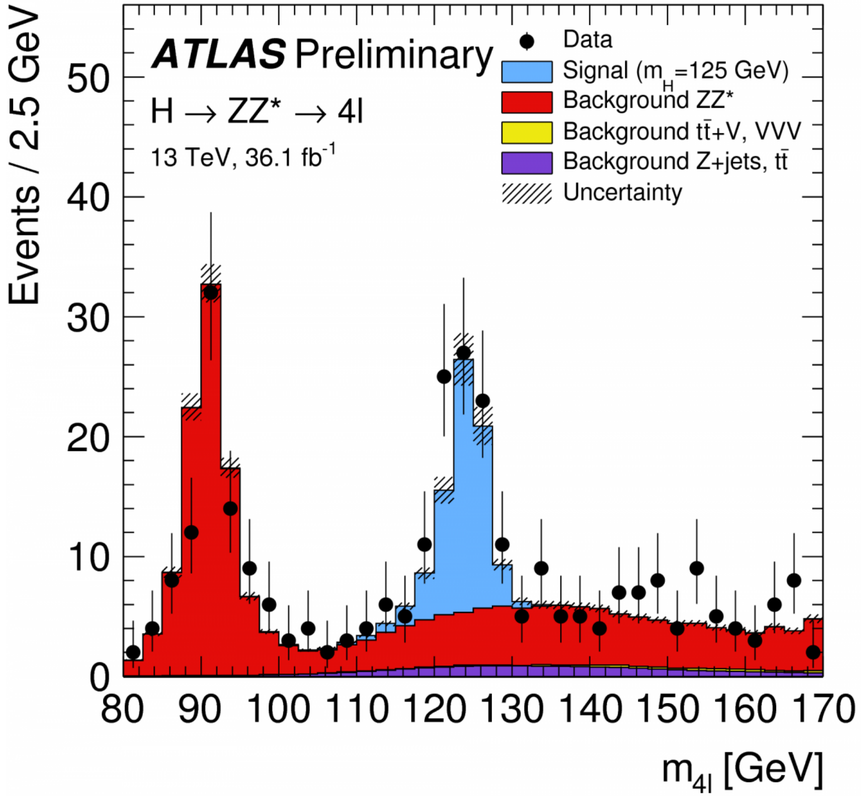
into the form

$$\partial_\mu B^\mu = 0, \quad \partial_\nu G^{\mu\nu} + e^2\varphi_0^2 B^\mu = 0. \quad (4)$$

Equation (4) describes vector waves whose quanta have (bare) mass $e\varphi_0$. In the absence of the gauge field coupling ($e = 0$) the situation is quite different: Equations (2a) and (2c) describe zero-mass scalar and vector bosons, respectively. In passing, we note that the right-hand side of (2c) is just the linear approximation to the conserved current: It is linear in the vector potential, gauge invariance being maintained by the presence of the gradient term.[5]

When one considers theoretical models in which spontaneous breakdown of symmetry under a semisimple group occurs, one encounters a variety of possible situations corresponding to the various distinct irreducible representations to which the scalar fields may belong; the gauge field always belongs to the adjoint representation.[6] The model of the most immediate interest is that in which the scalar fields form an octet under SU(3): Here one finds the possibility of two nonvanishing vacuum expectation values, which may be chosen to be the two $Y = 0$, $I_3 = 0$ members of the octet.[7] There are two massive scalar bosons with just these quantum numbers; the remaining six components of the scalar octet combine with the corresponding components of the gauge-field octet to describe

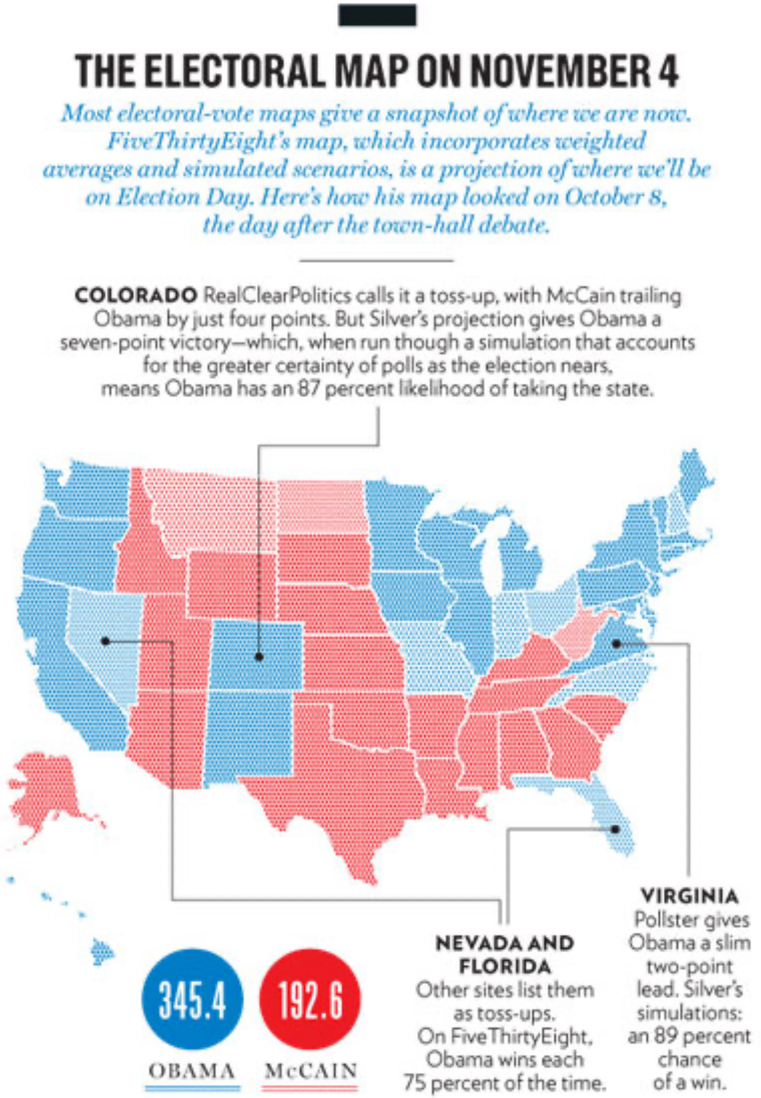
508



Peter Higgs
1964

ATLAS and CMS Experiments at the LHC 2012

Successes



The New York Times

Finding Fame With a Prescient Call for Obama

Give this article

By Stephanie Clifford
Nov. 9, 2008

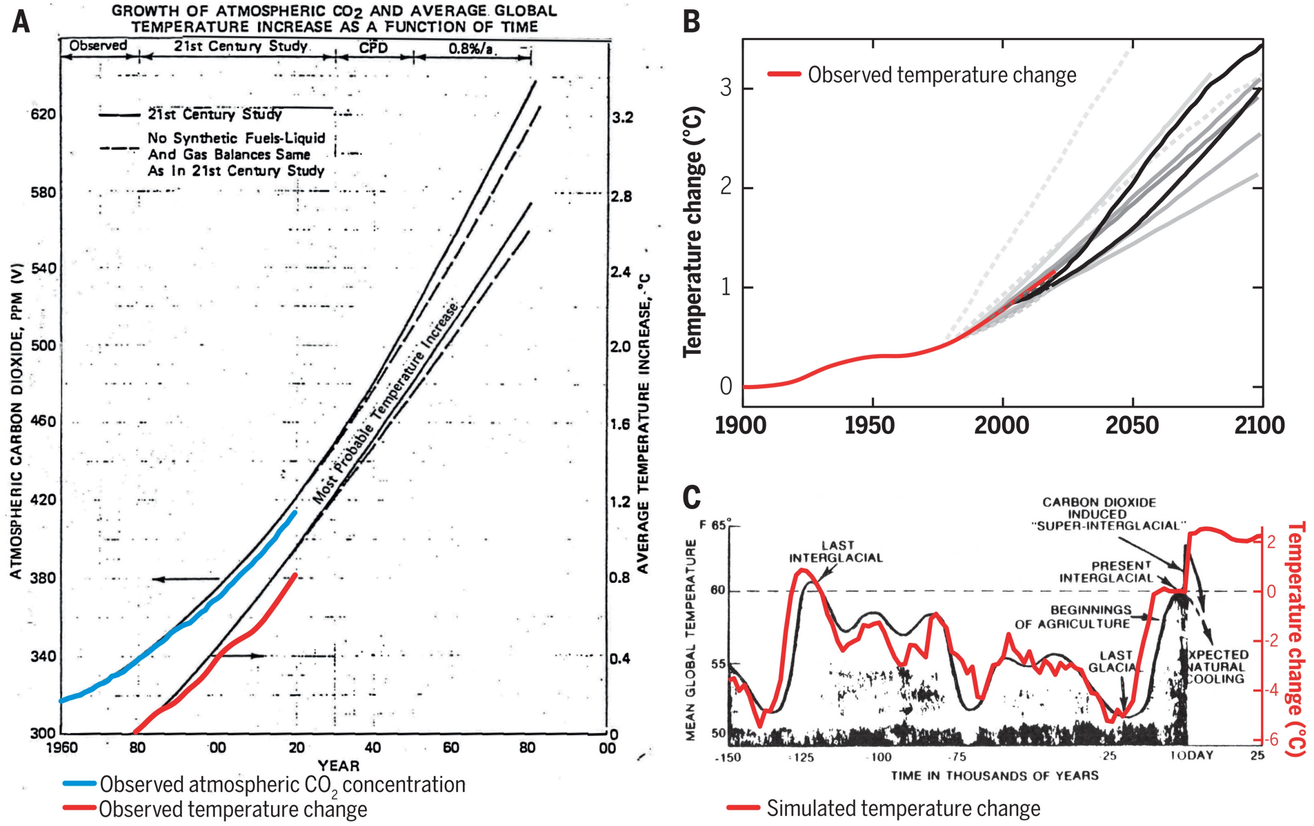
At 9:46 p.m., blogging on his site FiveThirtyEight.com, Nate Silver called the presidential election for Barack Obama. The television networks followed suit about an hour and 15 minutes later after most polls in Western states closed.

Of course, Mr. Silver had a head start: he had forecast that Senator Obama would beat Senator John McCain back in March.

In an election season of unlikely outcomes, Mr. Silver, 30, is perhaps the most unlikely media star to emerge. A baseball statistician who began analyzing political polls only last year, he introduced his site, FiveThirtyEight.com, in March, where he used his own formula to predict federal and state results and run Election Day possibilities based on a host of factors.

Other sites combine polls, notably RealClearPolitics and Pollster, but FiveThirtyEight, which drew almost five million page views on Election Day, has become one of the breakout online stars of the

Successes

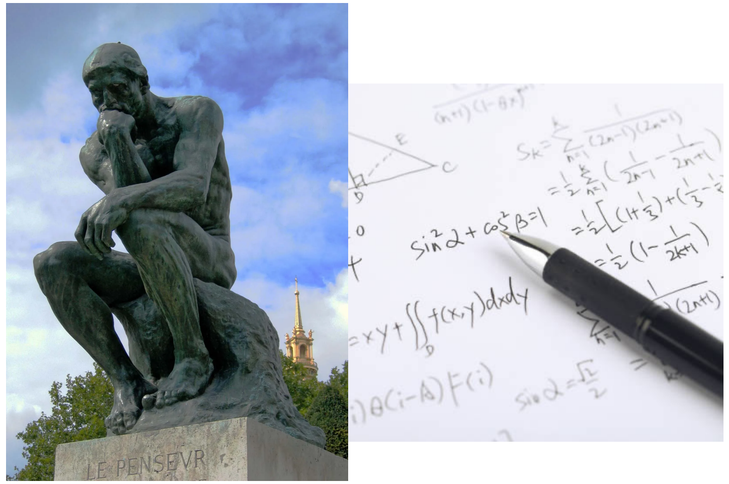


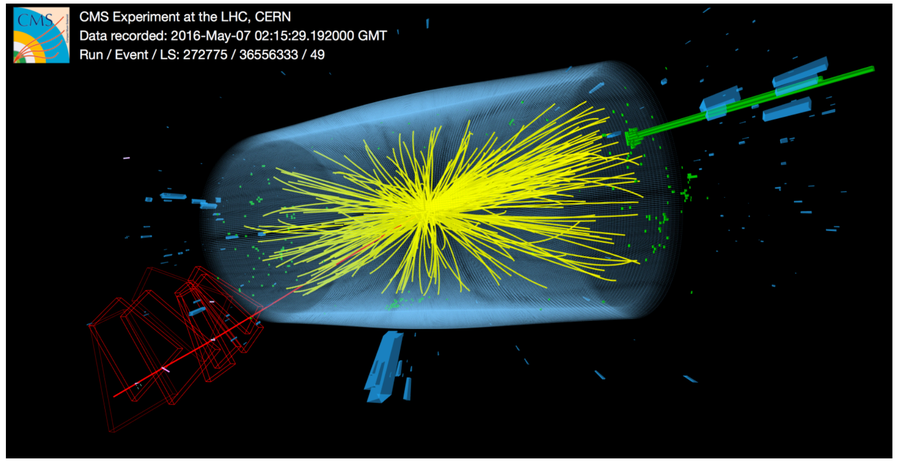
https://www.science.org/doi/10.1126/science.abk0063

Basic Ingredient

Key aspect in science is not only to formulate the theory
but also to **work out what it predicts in the real world**

$$\mathcal{L} = -\frac{1}{4} F_{\mu\nu} F^{\mu\nu} + i\bar{\Psi}\not{D}\Psi + h.c. + \bar{\Psi}_i y_{ij} \Psi_j \phi + h.c. + |D_\mu \phi|^2 - V(\phi)$$





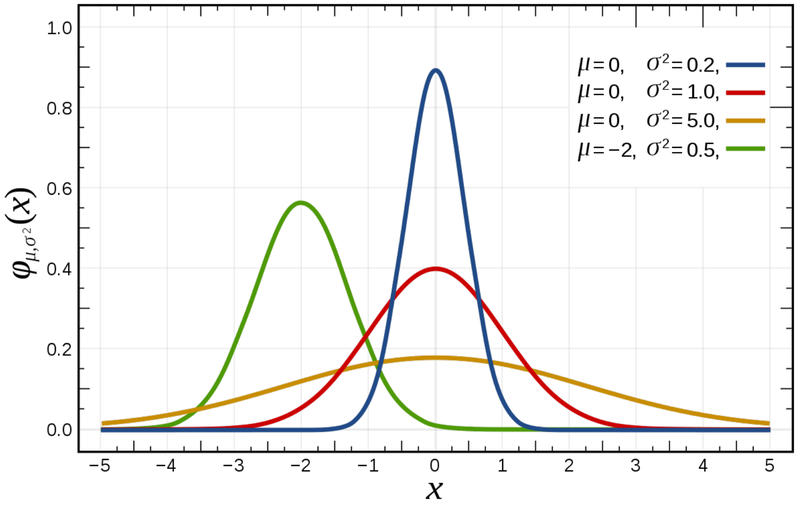
Theory

Testable Predictions

But things are never exactly known

We almost never have exact predictions, due to inherent or effective randomness of the measurement process

Out predictions are usually statistical in nature



$$p(\text{data}\,|\,\text{theory})$$

Bayes Theorem

concrete empirical
prediction lives here

$$p(\text{theory} \mid \text{data}) = \frac{p(\text{data} \mid \text{theory})}{p(\text{data})} p(\text{theory})$$

updated beliefs
about the state
of nature

prior beliefs
about the state
of nature

Tractable Models

Standard statistics (i.e. inference) is concerned with the situation, in which you (think you) know the model

$$p(\text{theory} \mid \text{data})$$

Frequentist Statistics

e.g. Maximum Likelihood Estimates

Bayesian Statistics

e.g. Posterior Estimates

Closed-Form Solvable Models

E.g. for Bayesian Statistics, in certain cases we can solve Bayes' Theorem by hand

Advantage:

very fast

Disadvantage:

rarely reflects our true beliefs

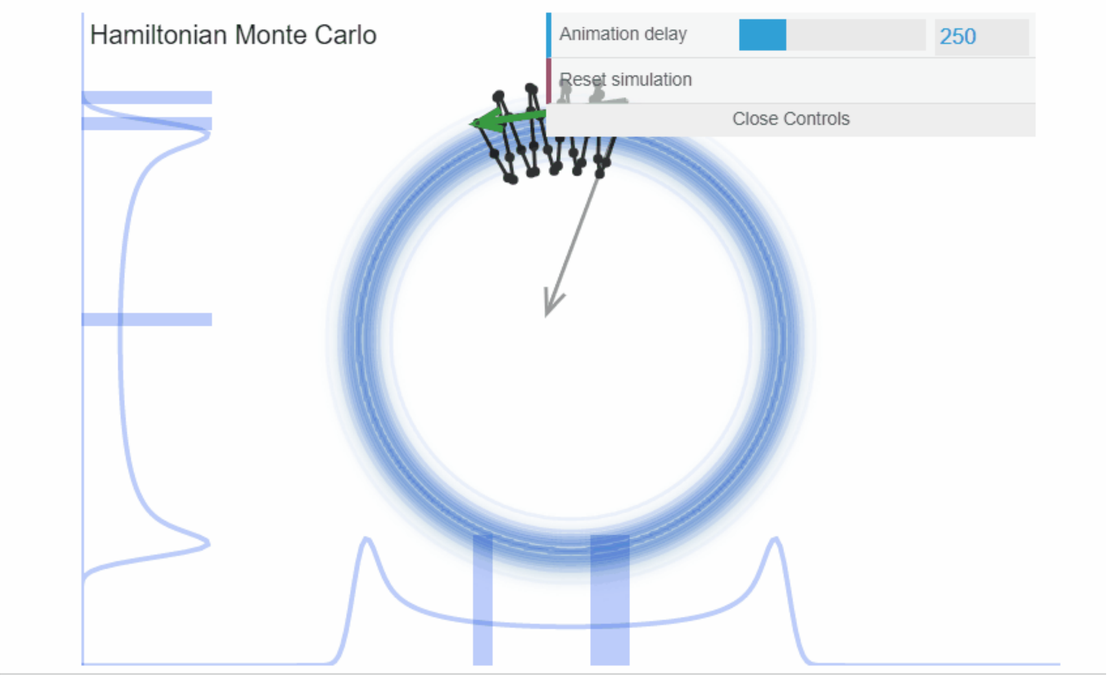
Likelihood	Model parameters	Conjugate prior distribution	Prior hyperparameters	Posterior hyperparameters[note 1]	Interpretation of hyperparameters	Posterior predictive[note 2]
Bernoulli	p (probability)	Beta	$\alpha,\ \beta \in \mathbb{R}$	$\alpha + \sum_{i=1}^{n} x_i,\ \beta + n - \sum_{i=1}^{n} x_i$	α successes, β failures[note 3]	$p(\tilde{x} = 1) = \frac{\alpha'}{\alpha' + \beta'}$
Binomial with known number of trials, m	p (probability)	Beta	$\alpha,\ \beta \in \mathbb{R}$	$\alpha + \sum_{i=1}^{n} x_i,\ \beta + \sum_{i=1}^{n} N_i - \sum_{i=1}^{n} x_i$	α successes, β failures[note 3]	$\mathrm{BetaBin}(\tilde{x} \mid \alpha', \beta')$ (beta-binomial)
Negative binomial with known failure number, r	p (probability)	Beta	$\alpha,\ \beta \in \mathbb{R}$	$\alpha + rn,\ \beta + \sum_{i=1}^{n} x_i$	α total successes, β failures[note 3] (i.e., $\frac{\beta}{r}$ experiments, assuming r stays fixed)	$\mathrm{BetaNegBin}(\tilde{x} \mid \alpha', \beta')$ (beta-negative binomial)
Poisson	λ (rate)	Gamma	$k,\ \theta \in \mathbb{R}$	$k + \sum_{i=1}^{n} x_i,\ \frac{\theta}{n\theta + 1}$	k total occurrences in $\frac{1}{\theta}$ intervals	$\mathrm{NB}\left(\tilde{x} \mid k', \frac{1}{\theta' + 1}\right)$ (negative binomial)
			$\alpha,\ \beta$[note 4]	$\alpha + \sum_{i=1}^{n} x_i,\ \beta + n$	α total occurrences in β intervals	$\mathrm{NB}\left(\tilde{x} \mid \alpha', \frac{\beta'}{1 + \beta'}\right)$ (negative binomial)
Categorical	$\boldsymbol{p}$ (probability vector), k (number of categories; i.e., size of $\boldsymbol{p}$)	Dirichlet	$\boldsymbol{\alpha} \in \mathbb{R}^k$	$\boldsymbol{\alpha} + (c_1, \ldots, c_k)$, where c_i is the number of observations in category i	α_i occurrences of category i[note 3]	$p(\tilde{x} = i) = \frac{\alpha_i'}{\sum_i \alpha_i'} = \frac{\alpha_i + c_i}{\sum_i \alpha_i + n}$
	$\boldsymbol{p}$					

Numeric Solutions (e.g. MCMC)

With increasing computational power, more we started to learn how to solve more realistic problems

MCMC Methods: for known but complex $p(x \mid \theta)$ **we cannot compute $p(\theta \mid x)$ but we can sample from it!**

$$\theta \sim p(\theta \mid x)$$



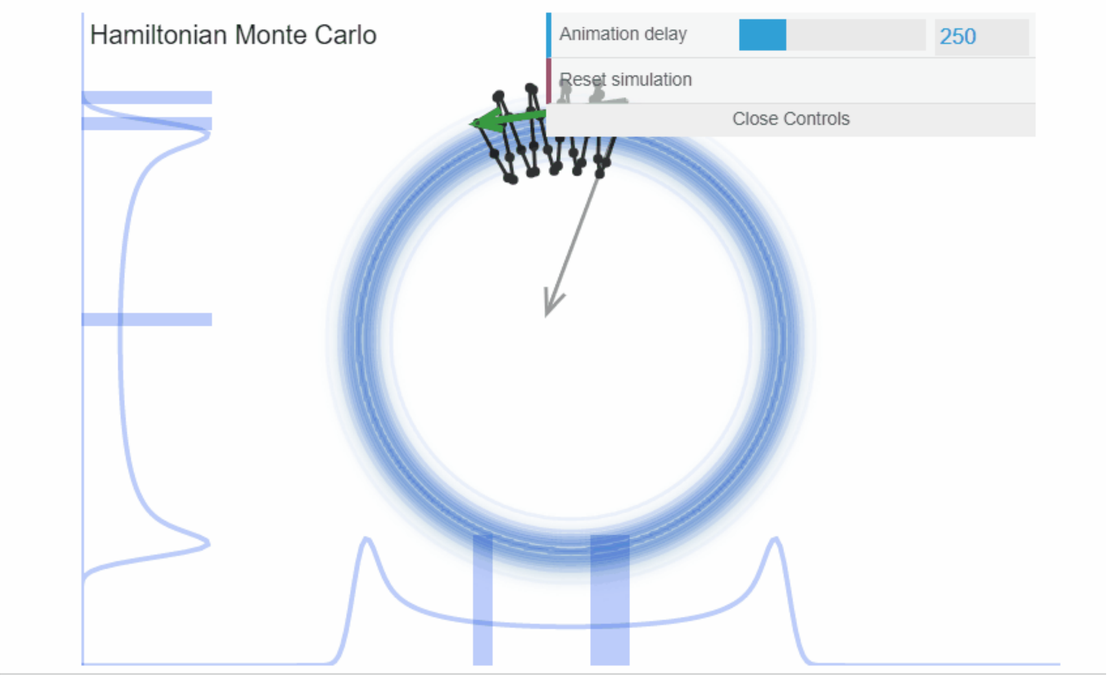
Gradients

We just discussed automatic differentiation. MCMC is even more powerful if you can not only compute $l(\theta) = \log p(x \mid \theta)$

but also its gradient…

$$\nabla_\theta \log p(x \mid \theta)$$

Hamiltonian Monte Carlo



Gradients

We just discus **CMC is even**
more powerful $= \log p(x \mid \theta)$

but also its gra

$\nabla_\theta \log$

Hamiltonian

PyMC

PyMC is a probabilistic programming library for Python that allows users to build Bayesian models with a simple Python API and fit them using Markov chain Monte Carlo (MCMC) methods.

Features

PyMC strives to make Bayesian modeling as simple and painless as possible, allowing users to focus on their problem rather than the methods.

Here is what sets it apart:

- **Modern**: Includes state-of-the-art inference algorithms, including MCMC (NUTS) and variational inference (ADVI).
- **User friendly**: Write your models using friendly Python syntax. Learn Bayesian modeling from the many example notebooks.
- **Fast**: Uses Aesara as its computational backend to compile to C and JAX, run your models on the GPU, and benefit from complex graph-optimizations.

Bad News

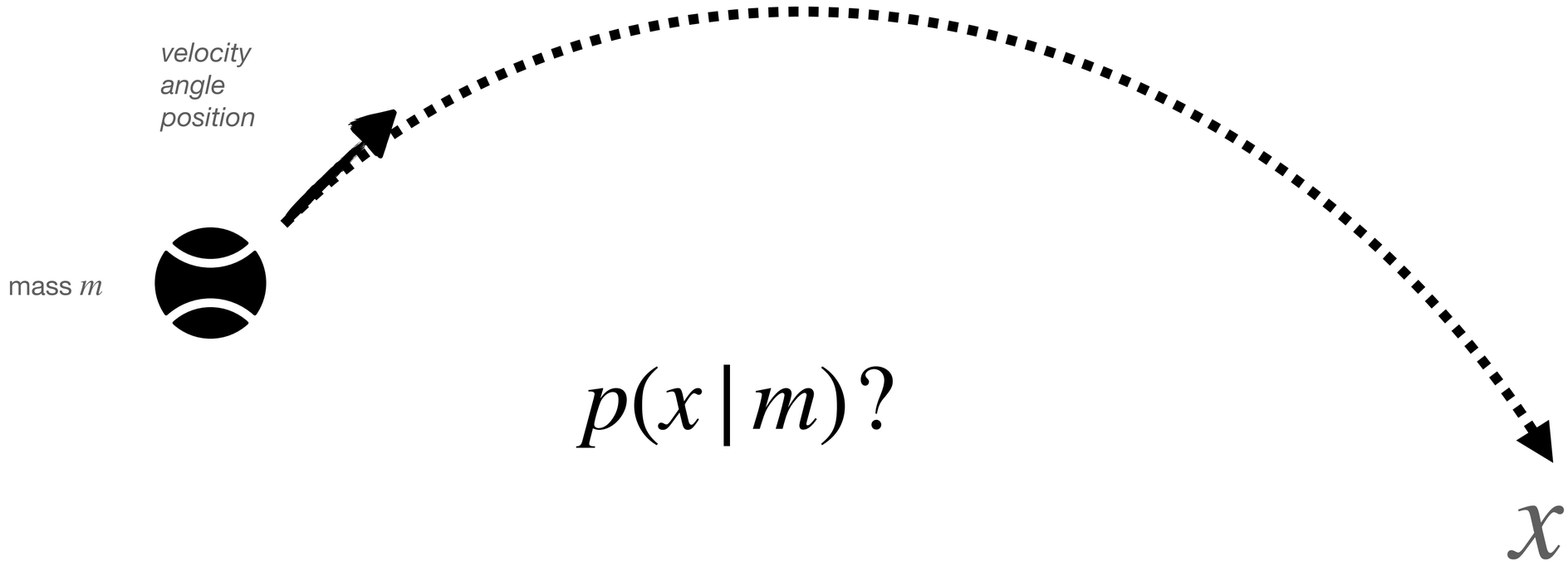
But what if … we didn't know how to compute

$$p(\text{data} \mid \text{theory})$$

(this happens much more often than we'd like)

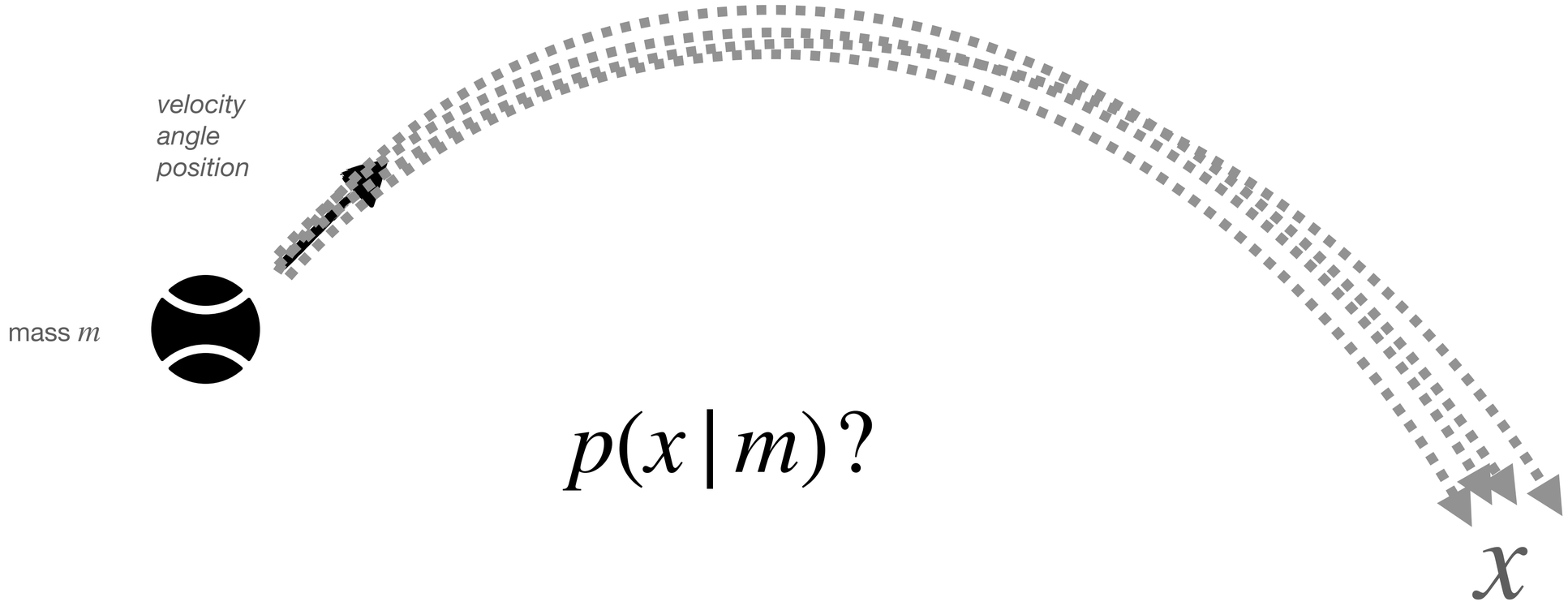
Example

What's the probability for a ball to measure a ballistic ball launched at an angle at a final position x ?



Example

What's the probability for a ball to land at position x ?



$$p(x \mid m)?$$

Example

Need to integrate over all possible unobserved paths for a given final outcome

$$p(x \mid m) = \int p(x \mid f_{\text{physics}}(v, \alpha, \ldots)) p(v, \alpha, \ldots \mid m)?$$

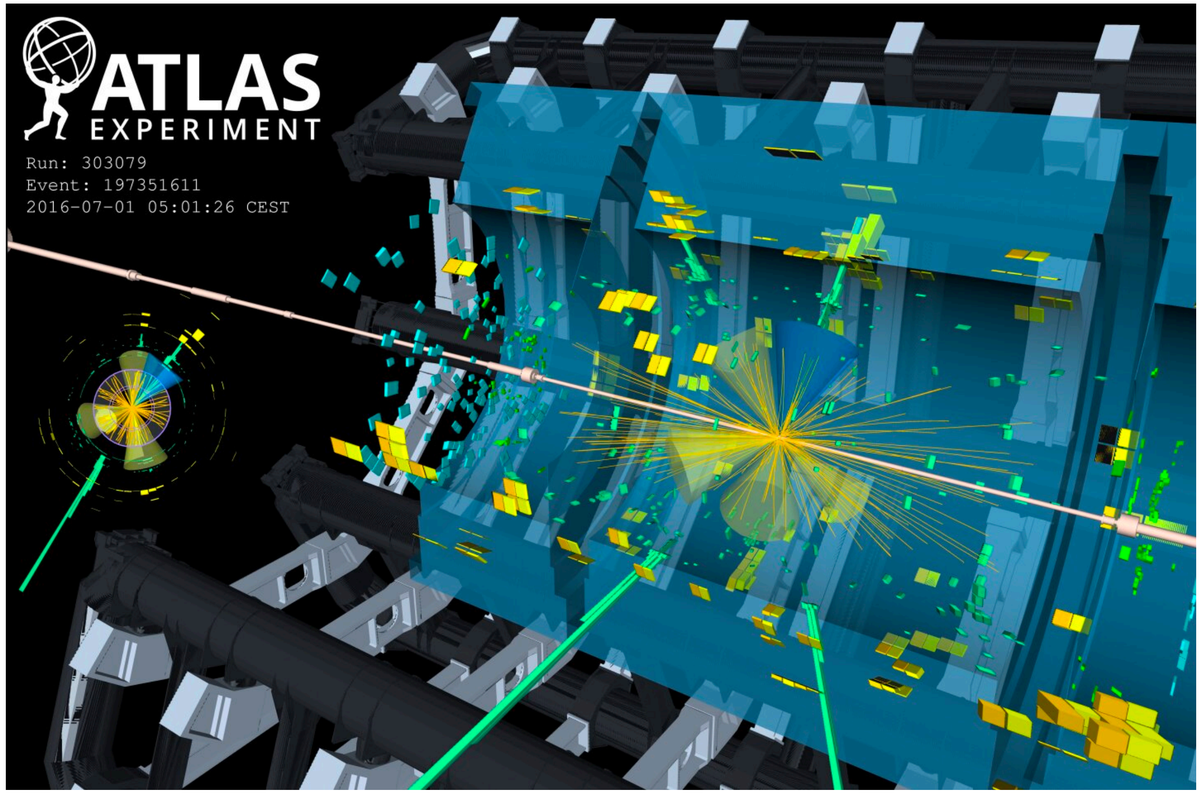
measured position is likely distributed over theoretically calculated one

predicted trajectory of ball based on physics

probabilities to launch at given speed / velocity

A sligthly more complicated example…

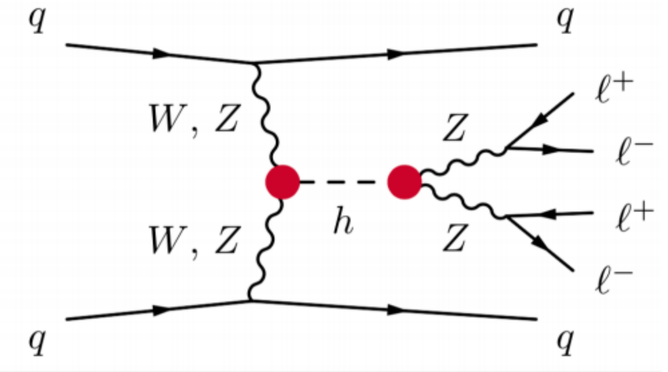
Given a Higgs mass, what do the 100M sensors in ATLAS read?



Why?

Because there is a huge gap between theory and data we need to bridge.

Core Interaction (Matrix Elements)

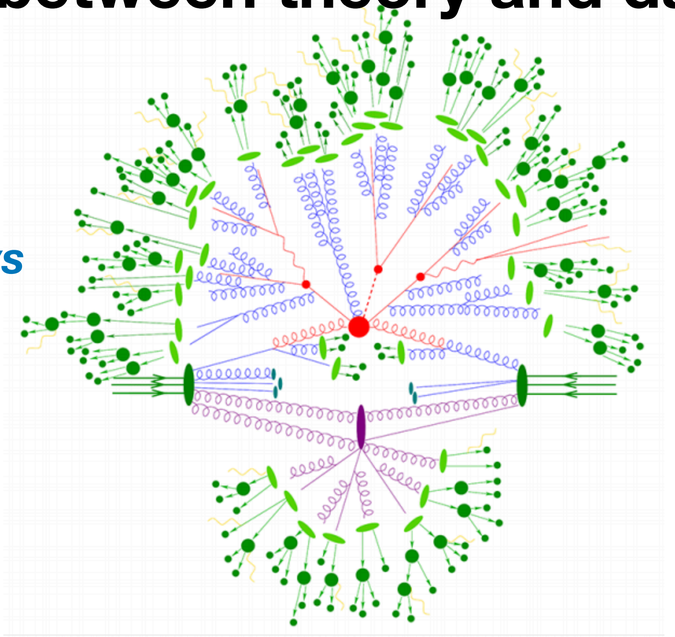


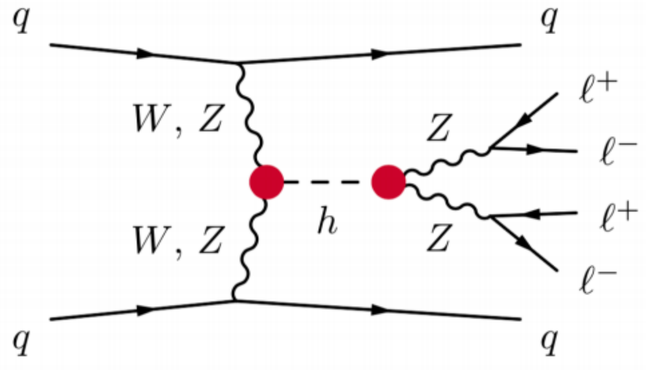
$$p(z_p \mid \theta)$$

Why?

Because there is a huge gap between theory and data we need to bridge.

Secondary, Tertiary, n-ary decays
(hadronization, parton shower)



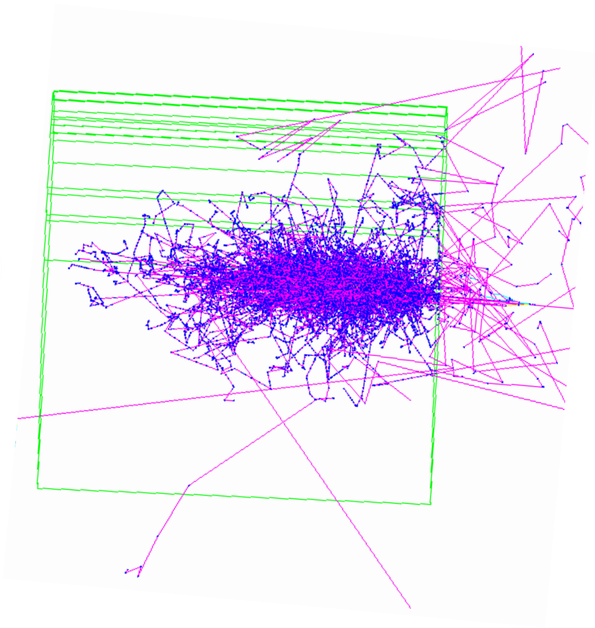


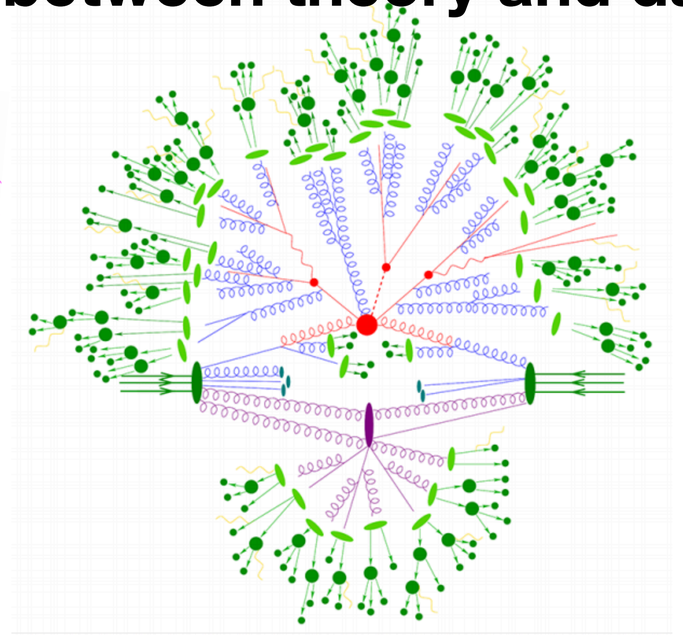
$$p(z_h \mid z_p)\ p(z_p \mid \theta)$$

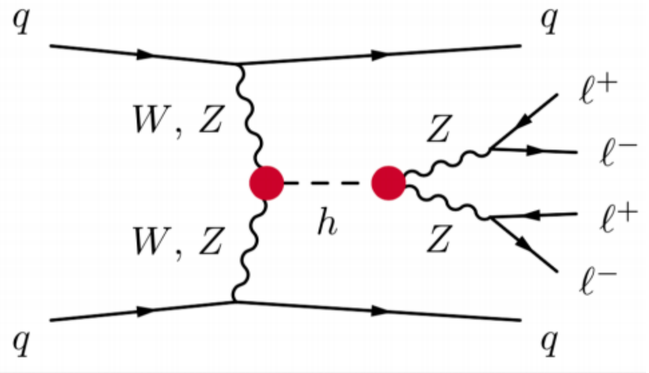
Why?

Because there is a huge gap between theory and data we need to bridge.

sensor interaction (particle transport)





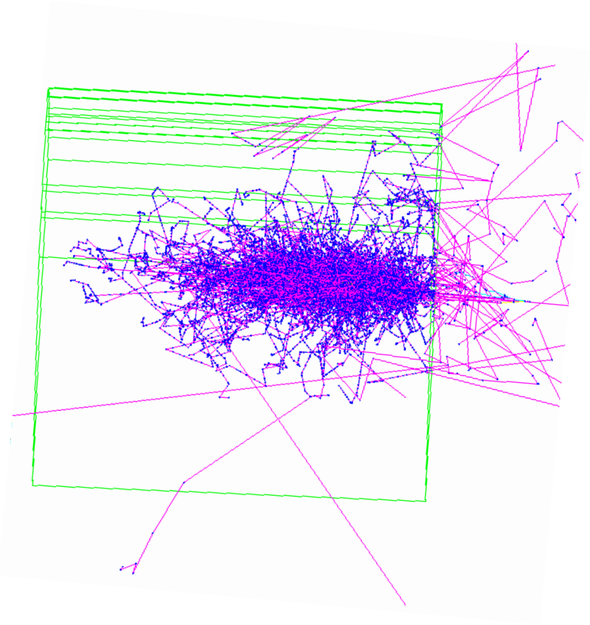


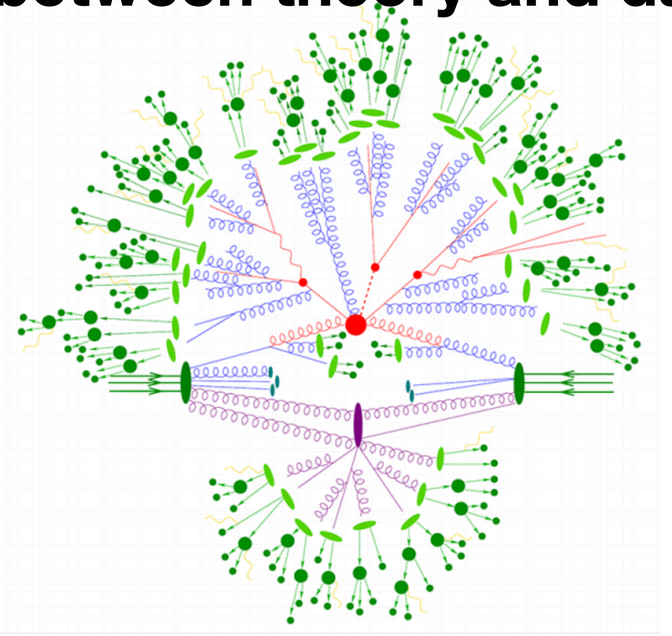
$$p(x \mid z_h)\ p(z_h \mid z_p)\ p(z_p \mid \theta)$$

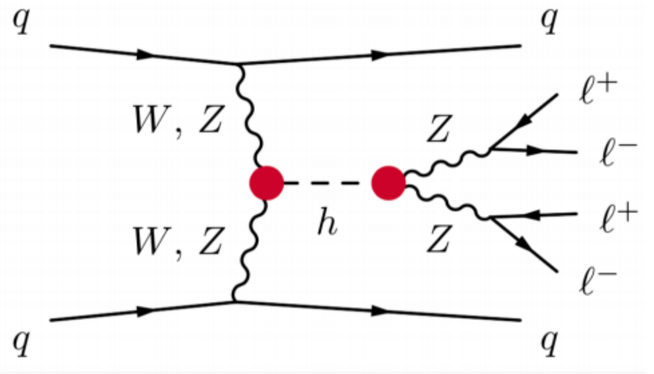
Why?

Because there is a huge gap between theory and data we need to bridge.

sensor interaction (particle transport)







all of this is quantum-mechanical and thus random, we only observe x at the end → integrate

$$p(x \mid \theta) = \int dz \;\; p(x \mid z_h)\; p(z_h \mid z_p)\; p(z_p \mid \theta)$$

Example

More generally, often $p(x \mid \theta)$ is intractable because of hidden unobserved dynamics / stochastic processes

- **histories of how the data came to be**
- **intractable even if we know (in principle) all the details (!)**

Example

More generally, often $p(x \mid \theta)$ is intractable because of hidden unobserved dynamics / stochastic processes

Mathematically: intractable integral

$$p(x \mid \theta) = \int \mathrm{d}\theta \; p(x \mid z) p(z \mid \theta)$$

possible measurements
given a concrete history

possible history
given external parameters

Now what?

If we cannot calculate the values of $p(x \mid \theta)$ how could we possibly do statistics ? Are we doomed?

No! Let's take a closer look at $p(x \mid \theta)$

The Two Faces of $p(x)$

Info about $p(x)$? What is $p(x)$? It's two things at once:

The Two Faces of $p(x)$

Info about $p(x)$? What is $p(x)$? It's two things at once:

A formula

$$\mathbb{R}^2 \rightarrow \mathbb{R}$$

$$p_{\mu,\Sigma}(x) = \frac{1}{\sqrt{|2\pi\Sigma|}} exp\left(-\frac{1}{2}(x-\mu)^T\Sigma(x-\mu)\right)$$

Evaluating the Probability for a given sample

The Two Faces of $p(x \mid \theta)$

Info about $p(x)$? What is $p(x)$? It's two things at once:

A process

$$\rightarrow \mathbb{R}^2$$

$$x \sim p(x \mid \theta)$$

Generating new samples from randomness

A formula

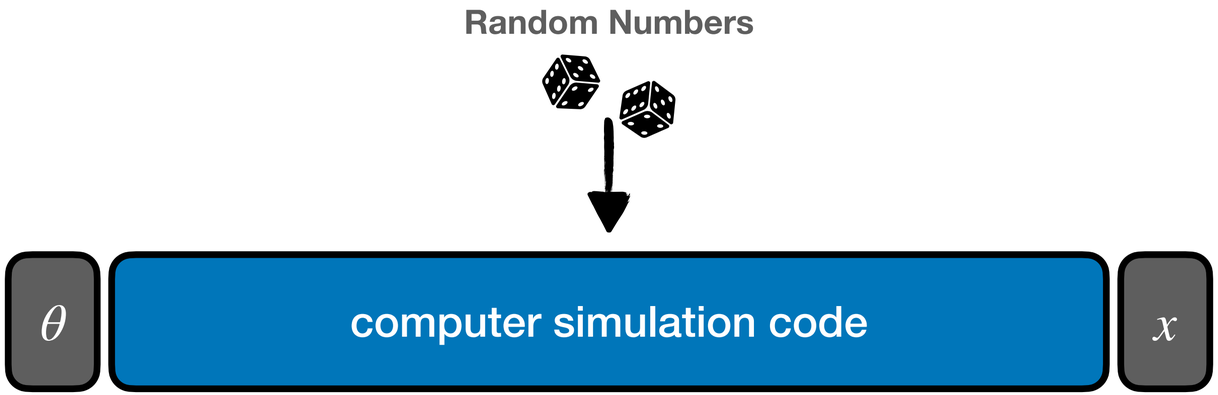
$$\mathbb{R}^2 \rightarrow \mathbb{R}$$

$$p_{\mu,\Sigma}(x) = \frac{1}{\sqrt{|2\pi\Sigma|}} exp\left(-\frac{1}{2}(x-\mu)^T\Sigma(x-\mu)\right)$$

Evaluating the Probability for a given sample

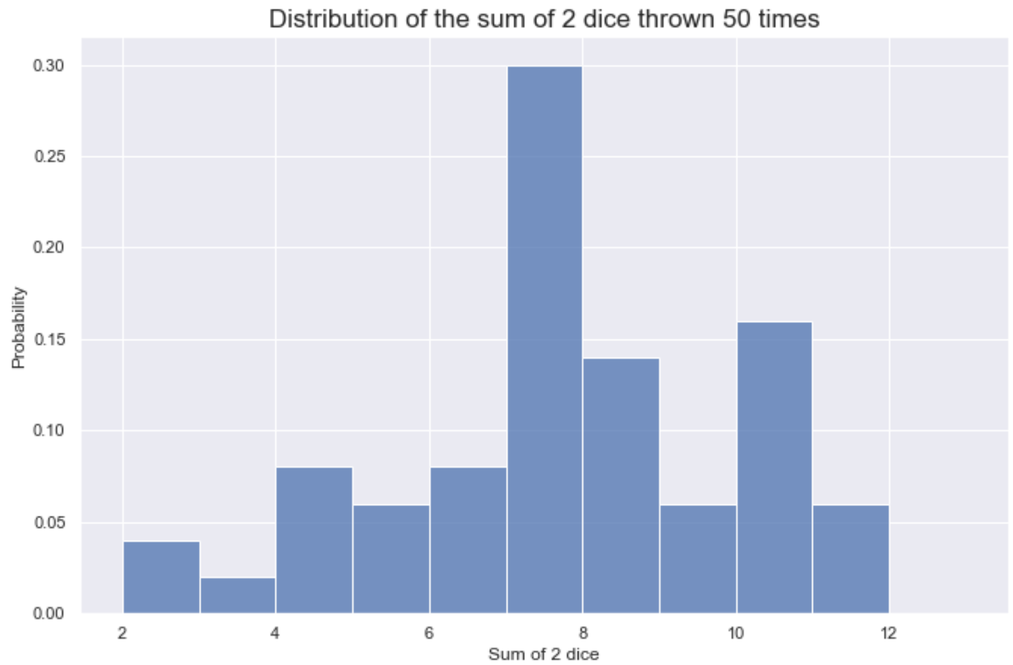
Writing Samplers of Physical Experiments

If we understand the mechanics of the data-generating process it's feasible to write "stochastic simulators" that implicitly implement $p(x \mid \theta)$



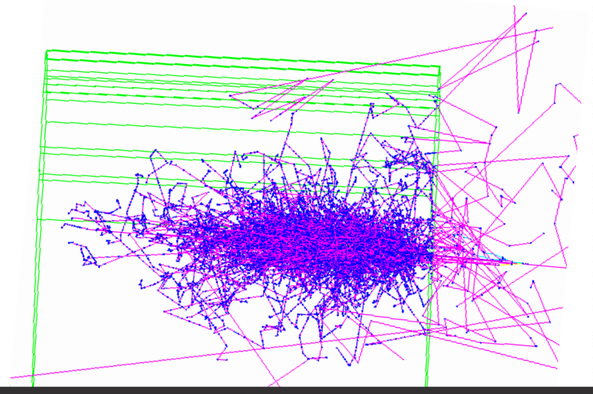
Example:

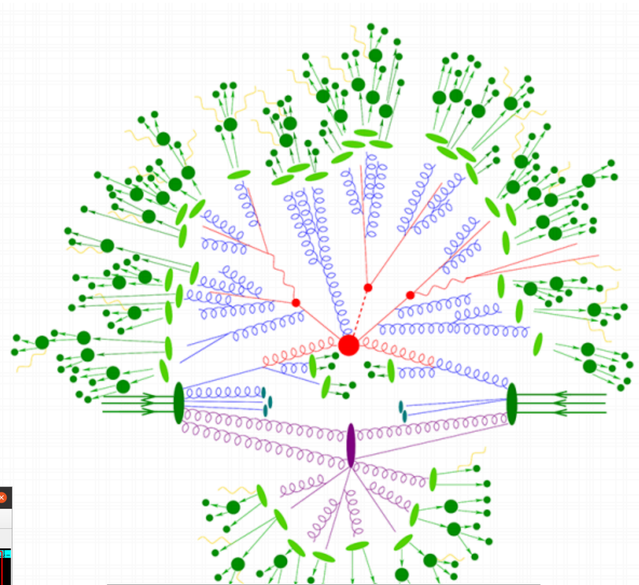
Without calculating anything, we can easily "approximately" determine $p(x \mid \theta)$ by analyzing samples from $x \sim p(x \mid \theta)$

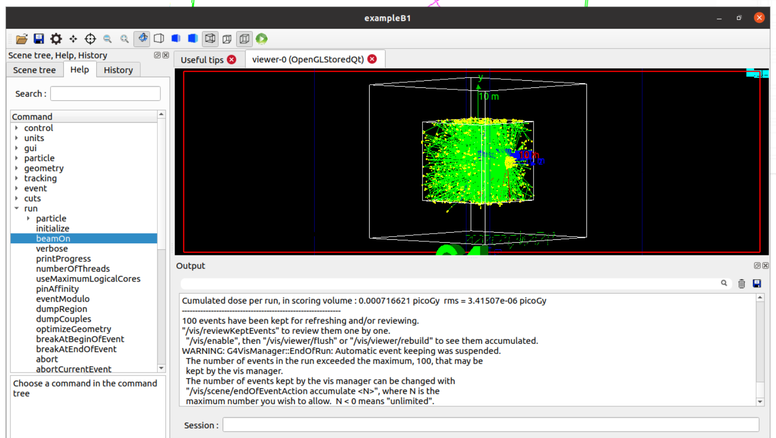


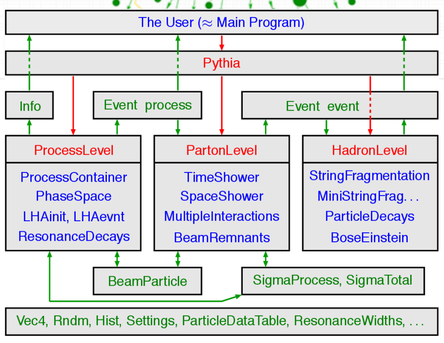
Simulators are Everywhere

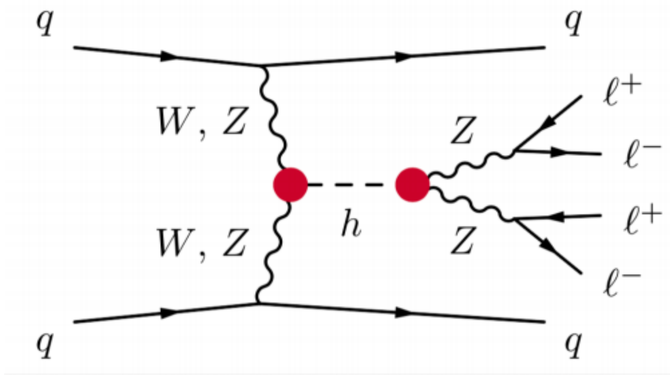
Stochastic Simulation Programs are how we can model complex physical systems. The Science lives in Software

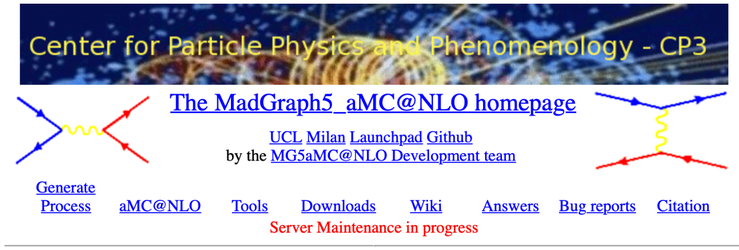




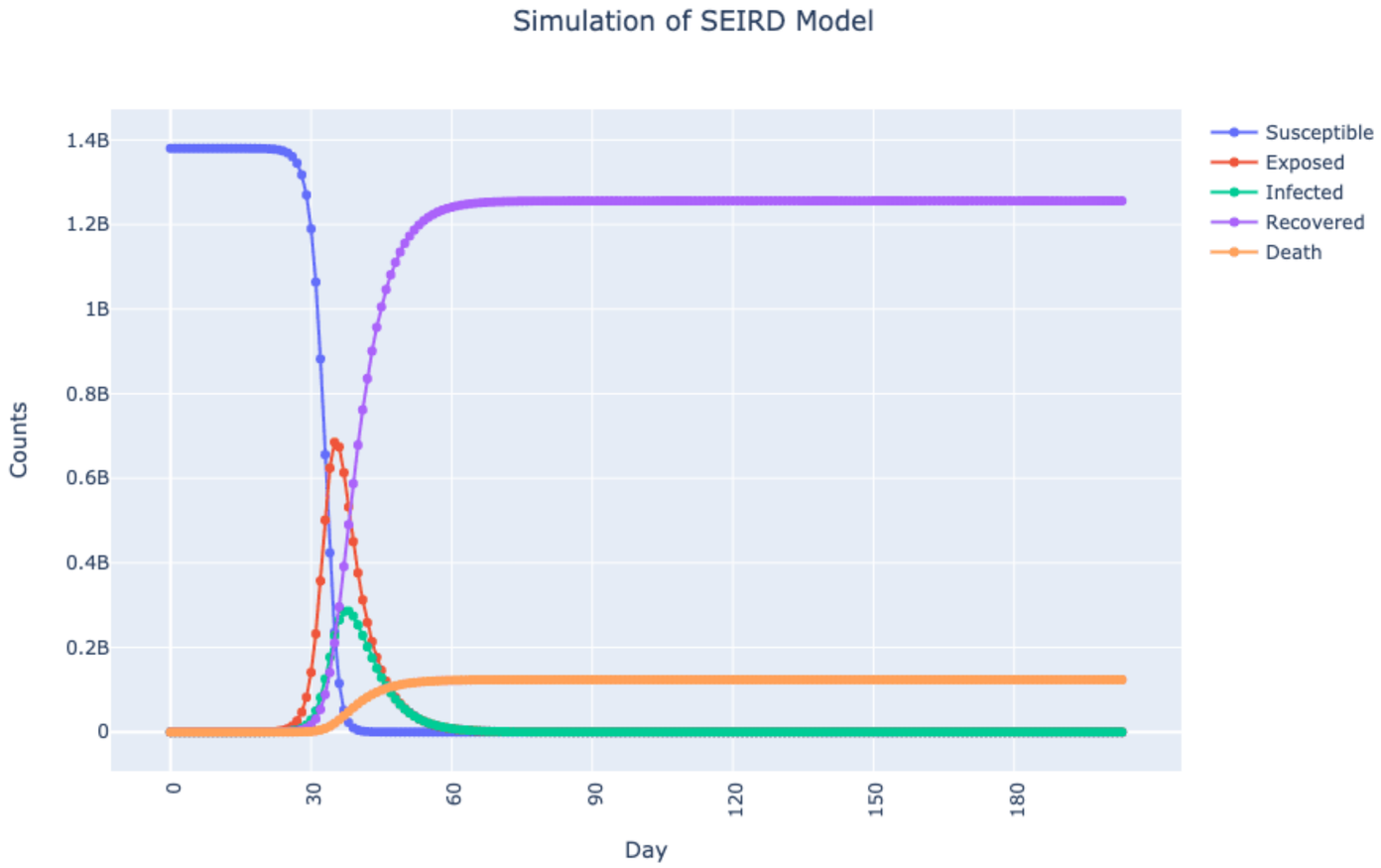








Simulators are Everywhere



Epidemiology Simulation

Simulation-based Inference

Simulation-based Inference:

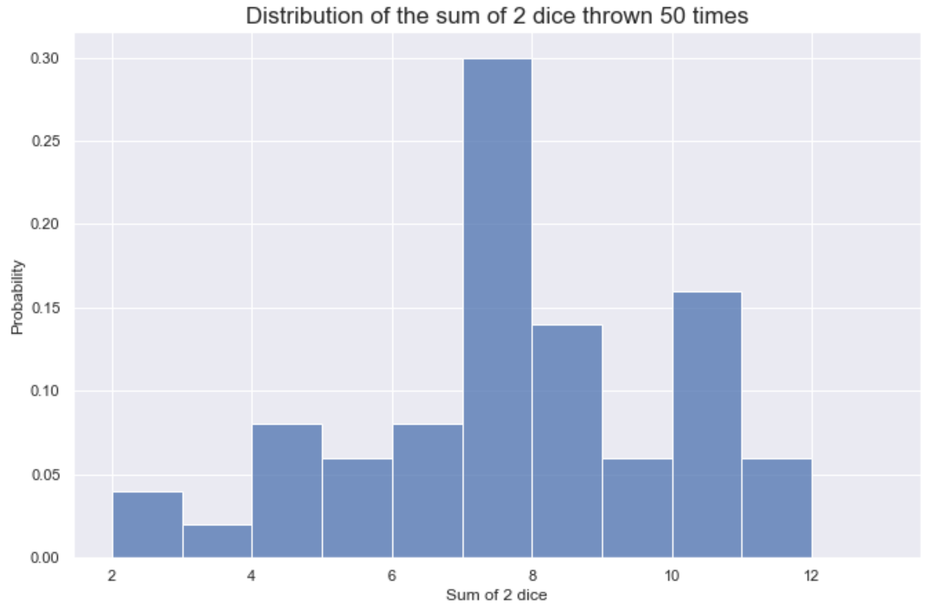
statistical techniques to apply when

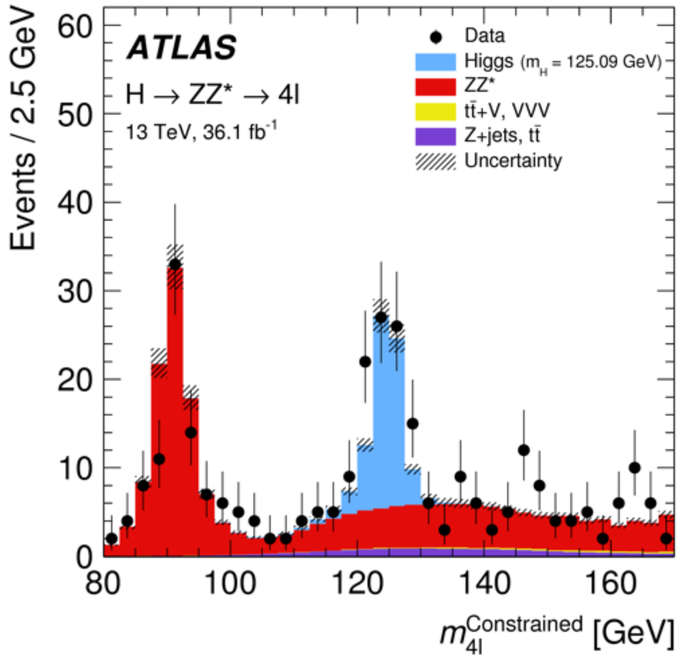
- you don't have a tractable likelihood $L(\theta) = p(x \mid \theta)$
- but do have a simulator that can give you faithful samples $x \sim p(x \mid \theta)$

Simulation-based Inference

Simplest method: density estimation

If we don't have $p(x \mid \theta)$ then approximate $\hat{p}(x \mid \theta)$ it as best as possible - then do standard statistics





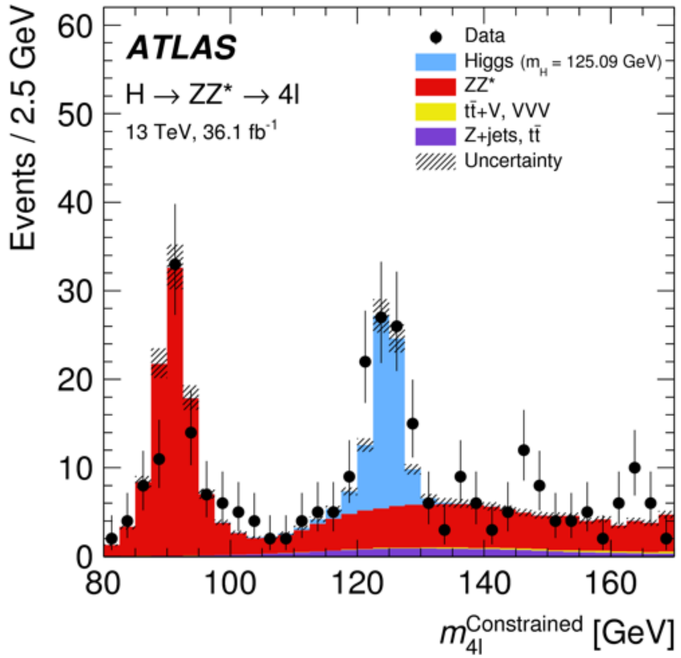
Simulation-based Inference

Simplest method: density estimation

If we don't have $p(x \mid \theta)$ then approximate $\hat{p}(x \mid \theta)$ it as best as possible - then do standard statistics

Advantage: no change in methodology

Disadvantage: density estimation in high dimensions is very difficult



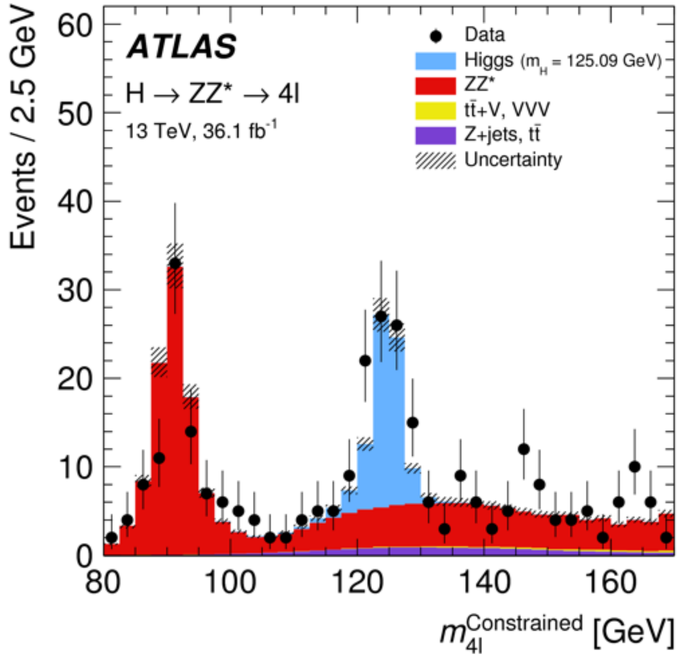
Simulation-based Inference

Simplest method: density estimation

Solution for high-dimensions: find good **summary statistics**

powerful functions $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}^m \; m \ll n$

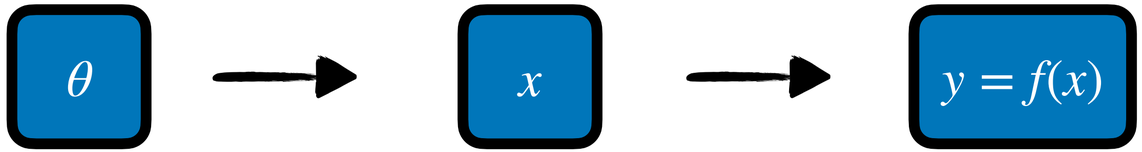
That retain the relevant information and "filter out" the noise as best as possible



Data Processing Inequality

This approach relies crucially on the quality of $f(x)$]

At best it is as good as the real data “sufficient statistics”, but usually a lossy process

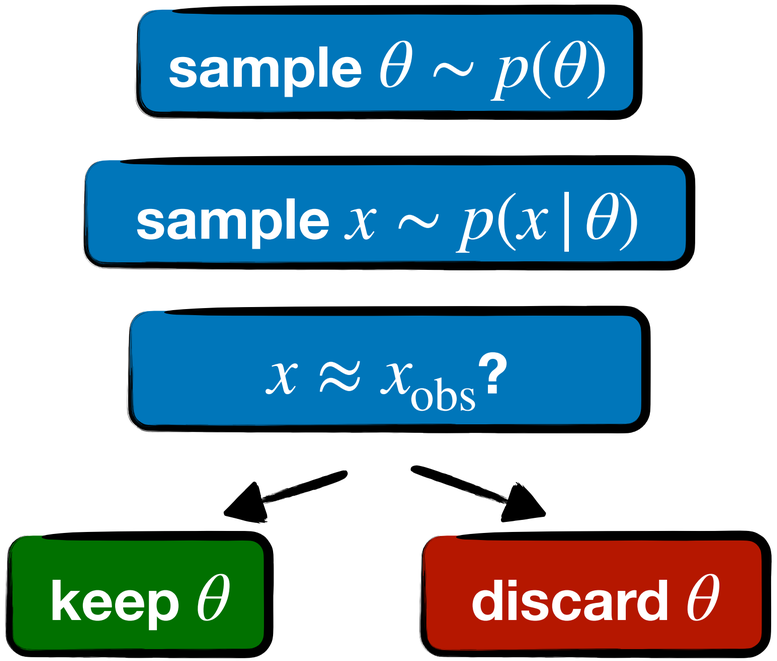


$$I(x;\theta) \geq I(y;\theta)$$

Brute-Force SBI

Alternatively, you can brute force e.g. posterior estimation with samples using Approximate Bayesian Computation

Simple Recipe to sample $\theta \sim p(\theta \mid x)$



Brute-Force SBI

Alternatively, you can brute force e.g. posterior estimation with samples using Approximate Bayesian Computation

Simple Recipe to sample $\theta \sim p(\theta \,|\, x)$

$$p_\epsilon(\theta \,|\, x_o) = \int \delta(\,|\, x - x_o \,|\, < \epsilon) p(x \,|\, \theta) p(\theta)$$

Advantage: no density estimation

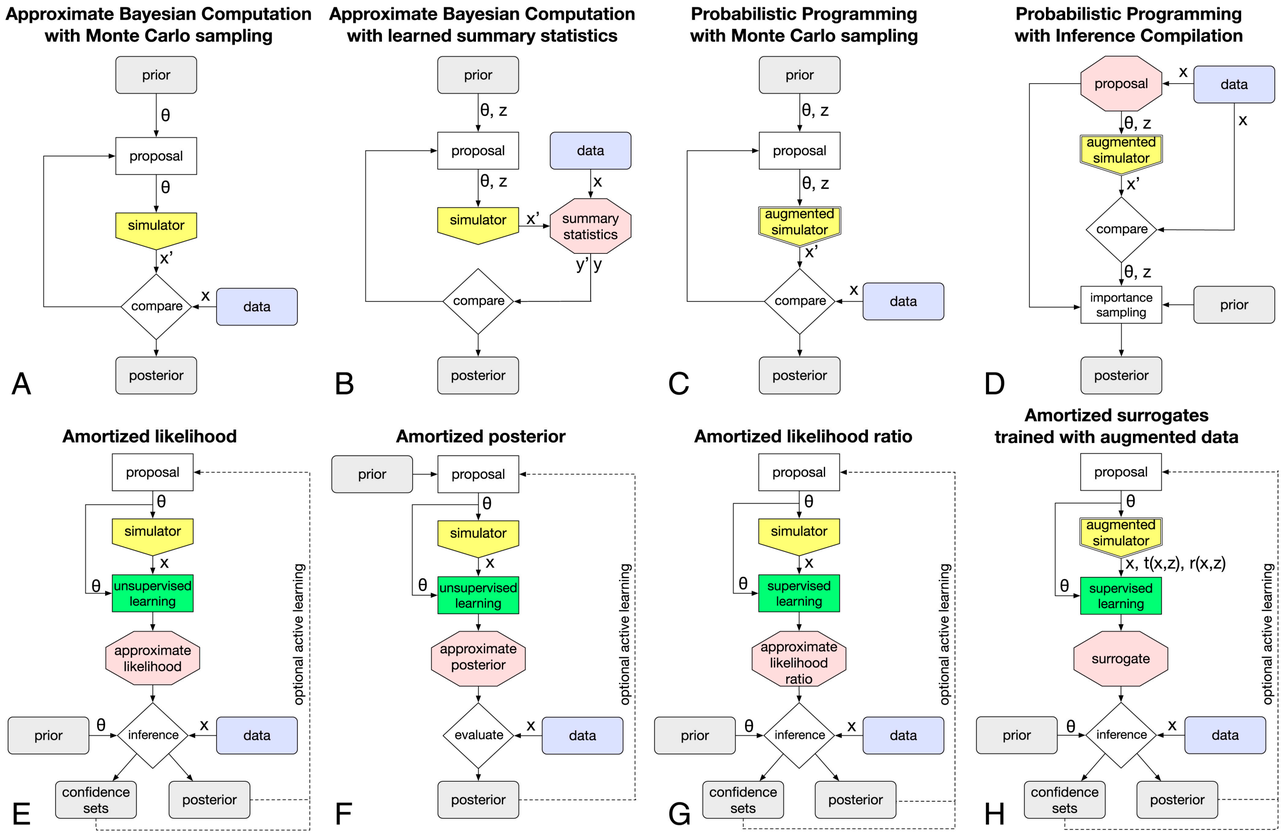
Disadvantage: very inefficient or very approximate

Simulation-based Inference with ML

More Recently there has been a big push to utilize Machine Learning for Simulation-based Inference

An entire family of methods forming, that combine domain-simulators with ML for higher fidelity, efficiency, etc..

Can only scratch the surface



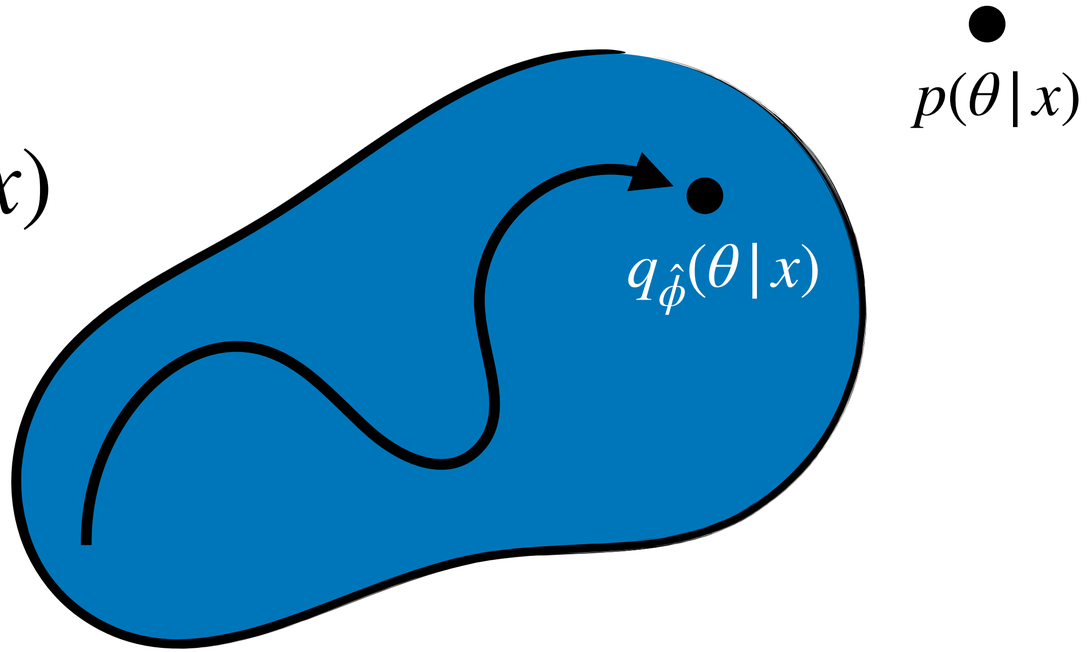
Simplest Approach: Variational Inference

Learning the posterior $p(\theta \mid x)$ is exactly what supervised machine learning does.

It does to by proposing a family of candidate posteriors, parametrized by ϕ: $q_\phi(\theta \mid x)$

Goal: Optimize ϕ such that $q_\phi(\theta \mid x)$

is a good approx. for all x



Simplest Approach: Variational Inference

Want to minimize

$$L(x) = D_{\mathrm{KL}}(p(\theta \,|\, x) \,||\, q_\phi(\theta \,|\, x)) = \mathbb{E}_{\theta|x} \log \frac{p(\theta \,|\, x)}{q_\phi(\theta \,|\, x)}$$

Nice try: no way to sample from $p(\theta \,|\, x)$ ....

Power of Amortization

Amazingly if we try to simultaneously solve for all x the problem becomes tractable

$$L' = \mathbb{E}_x L(x) = \mathbb{E}_x \mathbb{E}_{\theta|x}[-\log q_\phi(\theta \,|\, x)] = \mathbb{E}_{x,\theta}[-\log q(\theta \,|\, x)]$$

just pairs of samples of data and label (θ, x)

This is exactly the "Cross-Entropy" loss in ML!

i.e. ML will converge to the true $p(\theta \,|\, x)$ in the limit of infinite data and this purely based on samples!

Limitations

The main limitation of "standard ML" is that the families of candidate posteriors $q_\phi(\theta \mid x)$ is usually very limited

- Classification: $q_\phi(\theta \mid x) = \text{Bernoulli}(\theta \mid p = f_\phi(x))$
- Regression: $q_\phi(\theta \mid x) = \text{Normal}(\theta \mid \mu = f_\phi(x), \sigma = \sigma_0)$

True posteriors are likely much more complicated

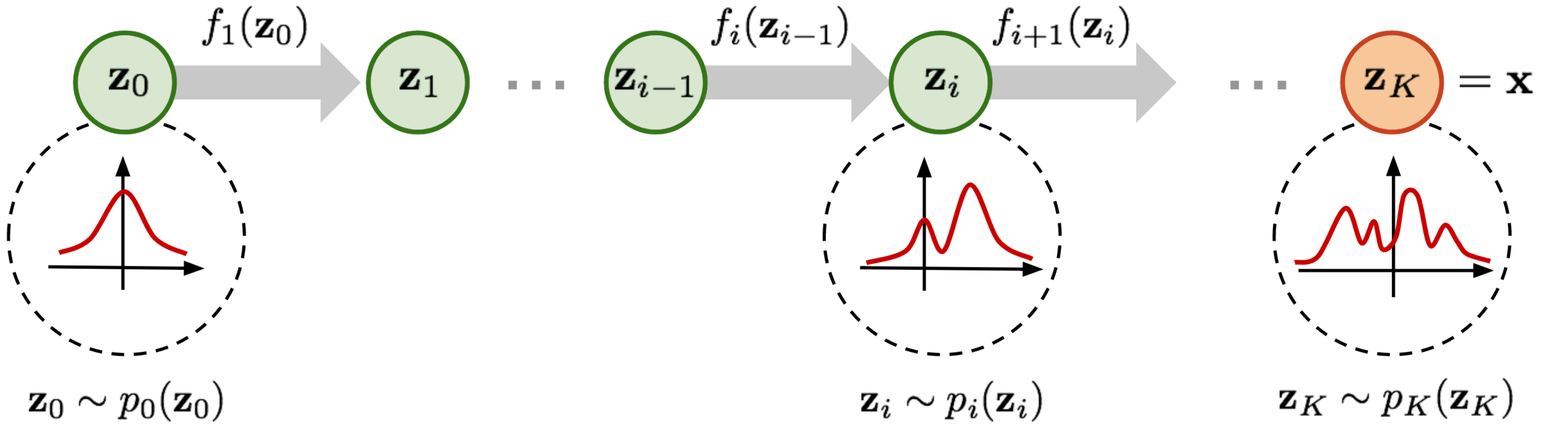
Complexifying $q_\phi(\theta \mid x)$

We can go beyond simple “scaffolding” densities (NNs used to just predict parameters of well known density families) using normalizing flows.

Neural Posterior Estimation

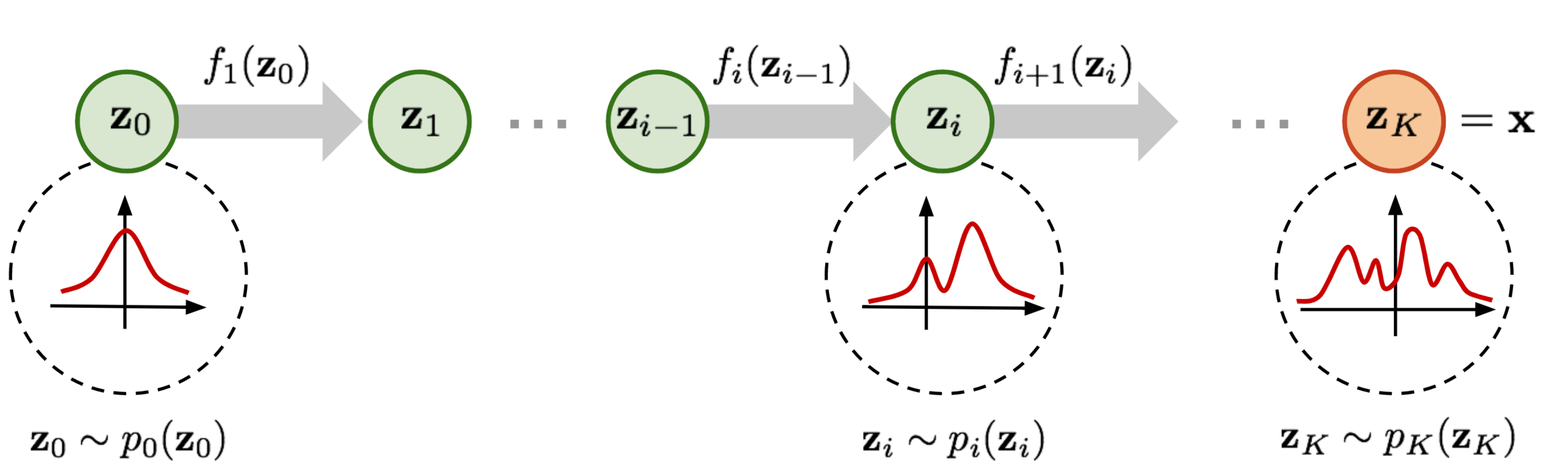
Normalizing Flows

Normalizing Flows are an expressive density modelling method based on a sequence of learnable bijections



Normalizing Flows

Training standard supervised-learning but with normalizing flows gives an immediate recipe for “simulation-based inference”



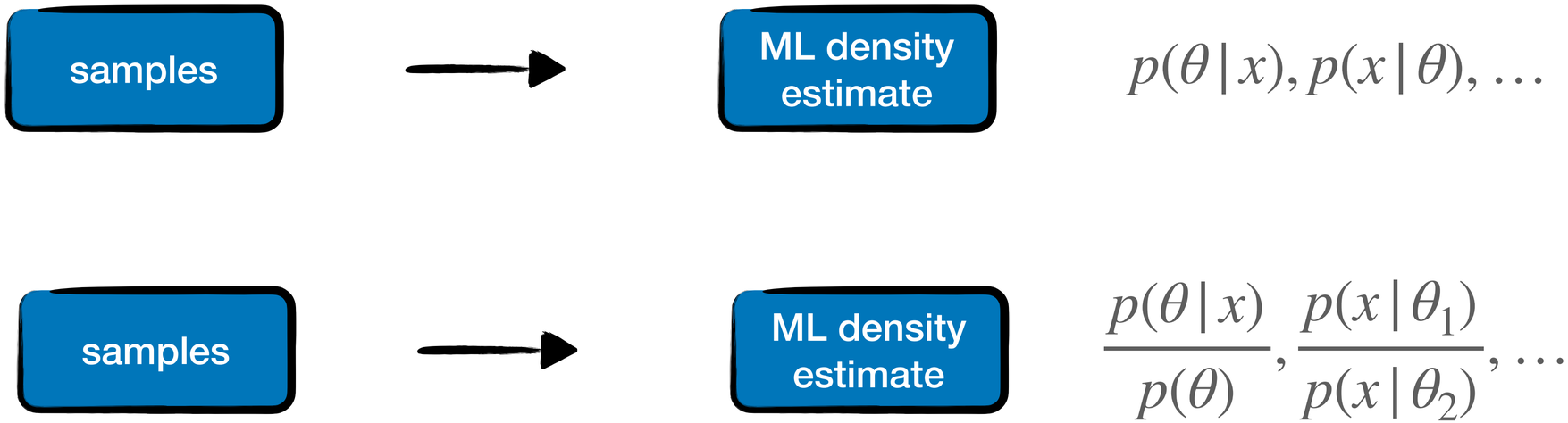
Disadvantage: density estimation is still a hard problem

Density Ratio Estimation

An alternative to density estimation is …

Density Ratio Estimation

Insight: a lot of standard statistiscs doesn't really need a fully normalized density. Un- (or arbitrarily) normalized is fine.



Going back to $f(x)$

We actually know what the best possible summary statistic is for e.g. deciding between two hypotheses

It's the likelihood ratio, due to the Neyman-Pearson Lemma

$$t(x) = \frac{p(x \mid \theta_0)}{p(x \mid \theta_1)}$$

Can we find it without knowing $p(x \mid \theta)$?

Yes we can!

Let's consider binary classification between θ_0 and θ_1

We know: BXE minimization leads to a the true posterior (in the asymptotic limit)

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

Yes we can!

Let's consider binary classification between θ_0 and θ_1

We know: BXE minimization leads to a the true posterior (in the asymptotic limit)

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

Yes we can!

Let's consider binary classification between θ_0 and θ_1

We know: BXE minimization leads to a the true posterior (in the asymptotic limit)

$$p(\theta_1 \,|\, x) = \frac{1}{1 + \frac{p(x \,|\, \theta_0)}{p(x \,|\, \theta_1)}}$$

Yes we can!

Let's consider binary classification between θ_0 and θ_1

We know: BXE minimization leads to a the true posterior (in the asymptotic limit)

$$p(\theta_1 \,|\, x) = \frac{1}{1 + \dfrac{p(x \,|\, \theta_0)}{p(x \,|\, \theta_1)}}$$

"The Likelihood Ratio Trick"

binary classification ↔ learn l'hood ratio

1/Likelihood Ratio

Yes we can!

Let's consider binary classification between θ_0 and θ_1

We know: BXE minimization leads to a the true posterior (in the asymptotic limit)

$$p(\theta_1 \,|\, x) = \frac{1}{1 + \exp(-\log(LR))}$$

Yes we can!

Let's consider binary classification between θ_0 and θ_1

We know: BXE minimization leads to a the true posterior (in the asymptotic limit)

$$p(\theta_1 \,|\, x) = \text{sigmoid}(\log \text{LR})$$

The pre-activation in the last NN layer is actually an estimate of the log-likelihood ratio!

Not very surprising

We actually know what the best possible summary statistic is for e.g. deciding between two hypotheses

It's the likelihood ratio, due to the Neyman-Pearson Lemma

$$t(x) = \frac{p(x \mid \theta_0)}{p(x \mid \theta_1)}$$

So training a ML algorithm to decide, will learn the same thing

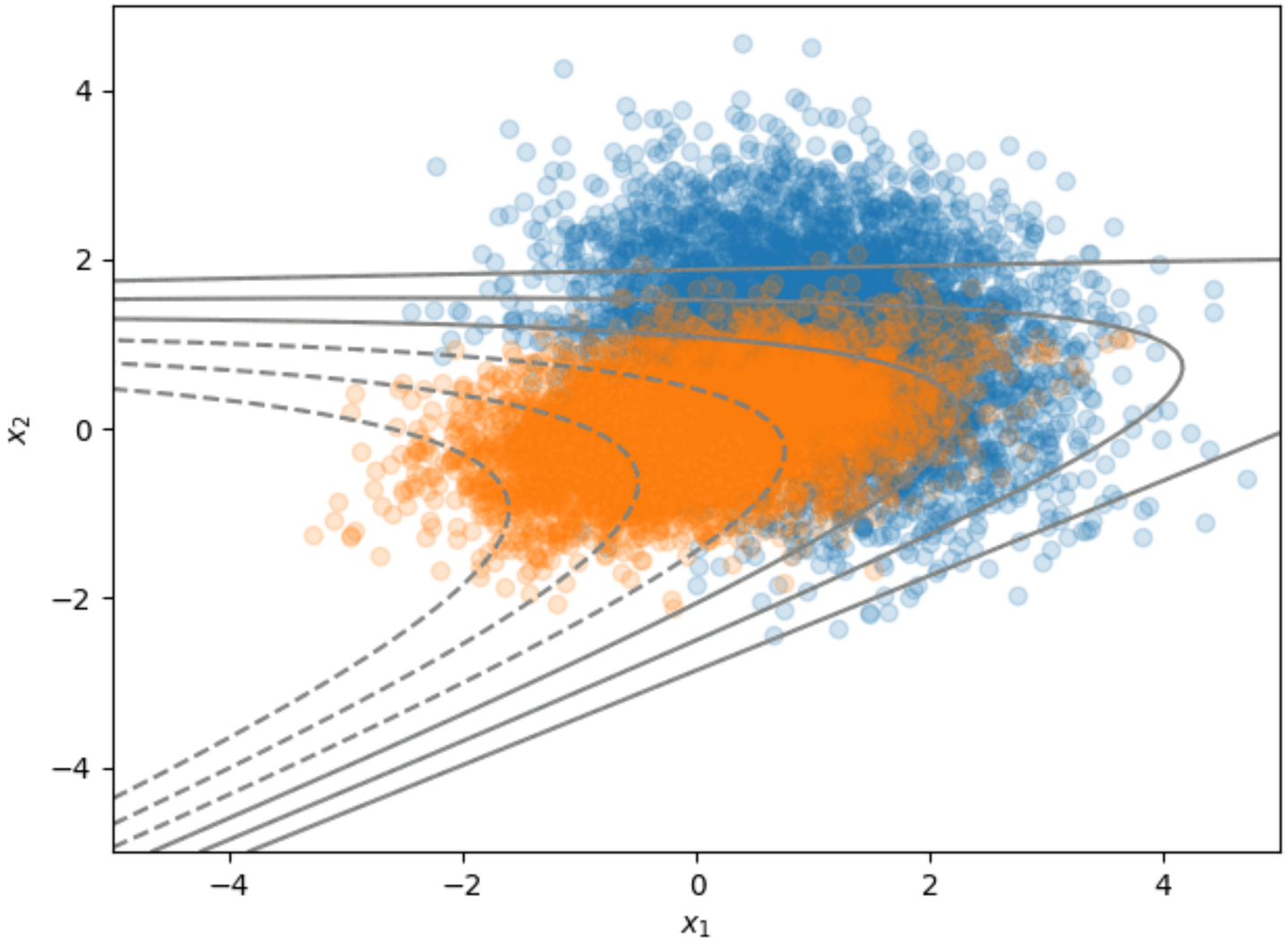
Recovering LR from a Classifier

We can recover the Likelihood Ratio from the ML model by just rearranging the formula

$$y = q_\phi(\theta_1 \,|\, x) = f_\phi(x) \approx \frac{1}{1 + \frac{1}{LR}}$$

$$LR \approx \frac{y}{1 - y}$$

Likelihood Ratio Trick - Visualized



The Value of Density Ratios

With the Likelihood-Ratio trick in our bag, we have a nice way to learn the likelihood ratio for a high-dimensional parameter space

We now want $A(x, \theta) = \dfrac{p(x \mid \theta)}{p(x \mid \theta_0)}$ **instead of** $A(x) = \dfrac{p(x \mid \theta_1)}{p(x \mid \theta_0)}$

For most purposes $A(x, \theta)$ **is just as useful as** $p(x \mid \theta)$

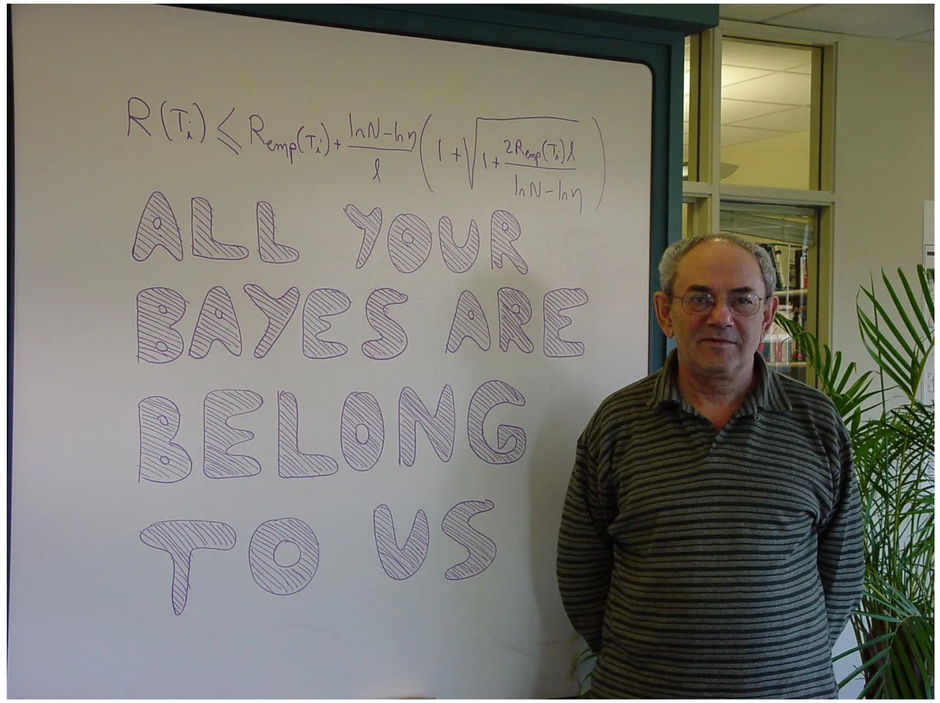
- **MLE estimation, MCMC, HMC, you name it…**

The Value of Density Ratios

Insight:

**"When solving a problem of interest,
do not solve a more general problem
as an intermediate step" -Vladimir Vapnik**

**(with density estimation we did try to
solve a more general problem)**



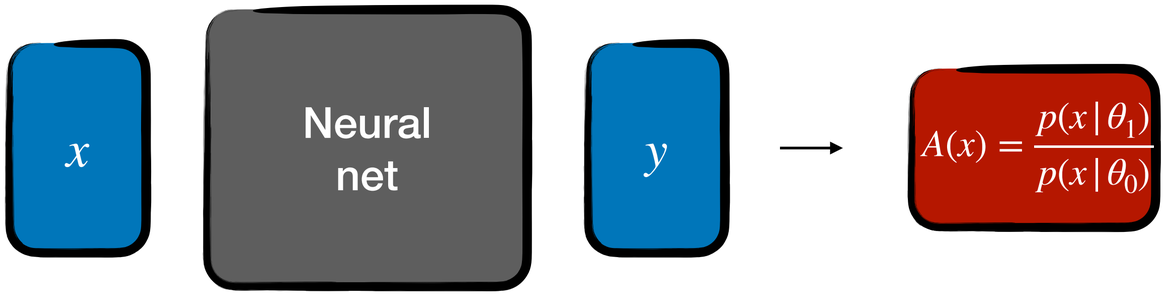
Parametrized Neural Network

We can use our classificaton-as-LR estimation trick to learn our target function $A(x, \theta)$ by

- **conditioning (or parametrizing) out neural network**
- **expanding our classification training regime**

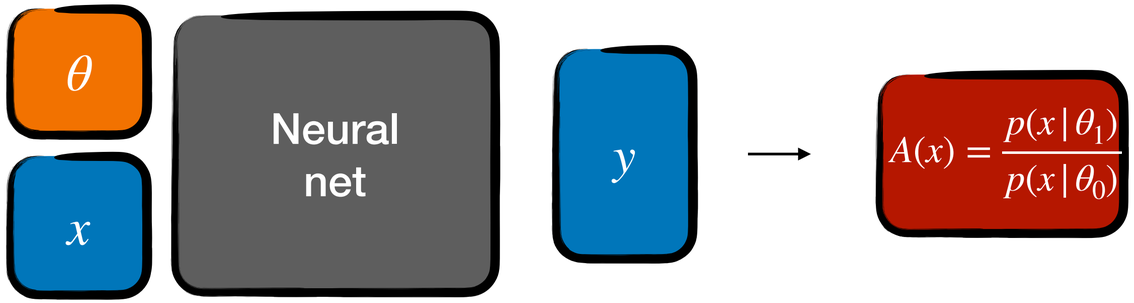
Conditioning the Neural Net

Normally the neural network just gets the data it's supposed to classify



Conditioning the Neural Net

But now we extend it by also giving it an additional input θ to turn $A(x) \rightarrow A(x, \theta)$



The input θ is signaling to the NN, which classification problem we are solving at any given time

Parametrized Training

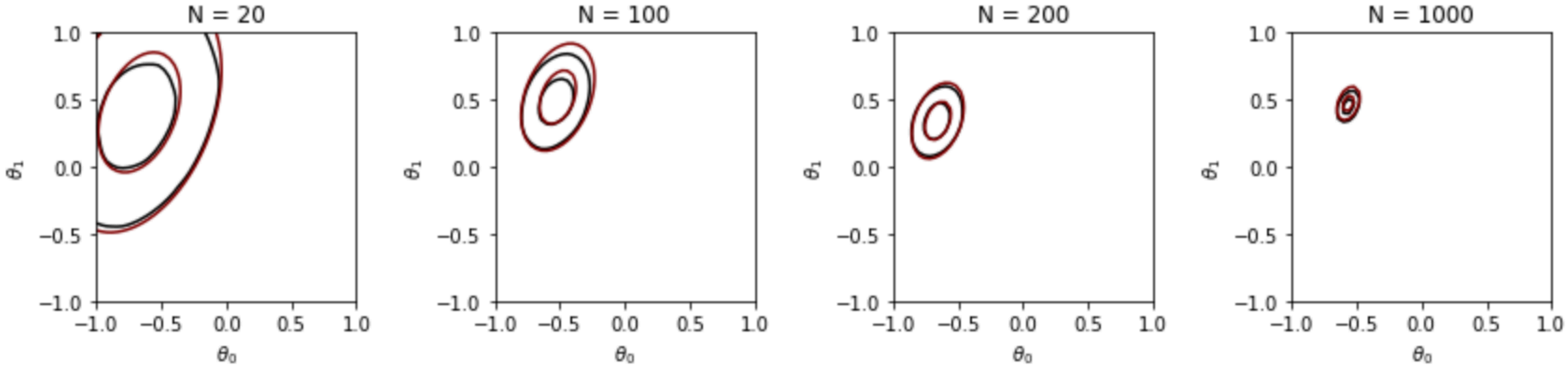
We can now have the following training loop

- sample a $\theta \sim p(\theta)$
- set up a classification problem for θ
  - sample “class 0” from simulator at θ_1: $x \sim p(x \mid \theta_1)$
  - sample “class 1” from simulator at θ: $x \sim p(x \mid \theta)$
  - do one train step in binary classification by running the network at value θ (will converge to likelihood ratio)
- rinse & repeat

We will do this as an exercise!

Example Results

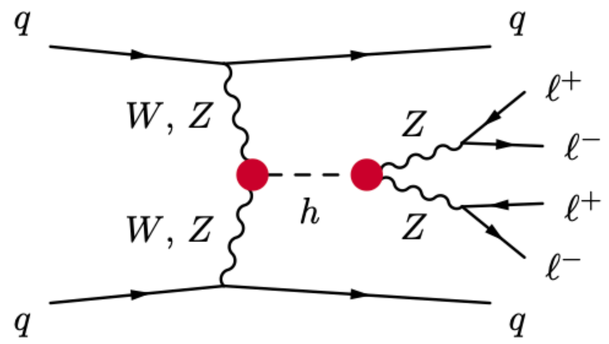
With sufficient data, we can learn $A(x, \theta) = p(x \mid \theta)/C$ to an impressive precision.



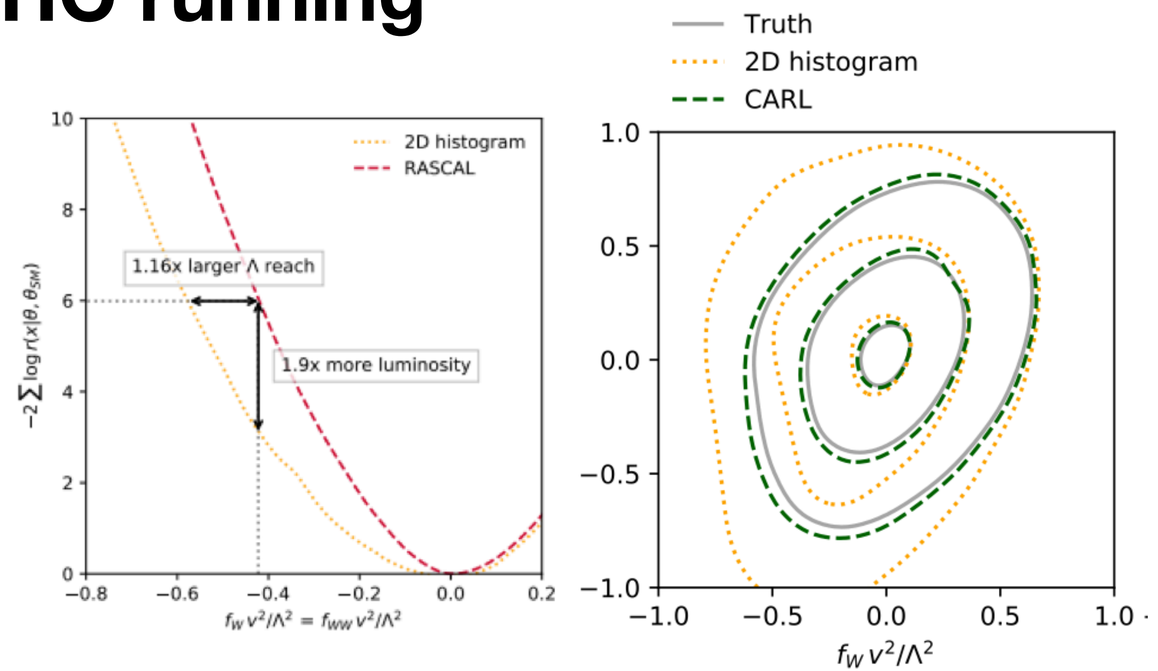
Note: no(!) data processing inequality, we use the full data!

Impact, e.g. at the LHC

The difference between summary statistics (hand-chosen $f(x)$) and "optimal observables" (i.e. LR estimates) can be equivalent to many years of LHC running



Studying the Higgs Boson in detail at the LHC



Extensions

The procedure to learn density ratios through classification is very general. Can be applied to many different scenarios

E.g.

$$p(\theta \mid x) = \frac{p(x \mid \theta)}{p(x)} p(\theta)$$

Extensions

The procedure to learn density ratios through classification is very general. Can be applied to many different scenarios

E.g.

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)p(\theta)}p(\theta)$$

Extensions

The procedure to learn density ratios through classification is very general. Can be applied to many different scenarios

E.g.

Look Ma! A Density Ratio

$$p(\theta \mid x) = \frac{p(x, \theta)}{p(x)p(\theta)} p(\theta)$$

The key difference between the posterior and the prior is captured by the density ratio of the joint distribution vs the product of the marginals

Extensions

We can sample from both!

- **joint density** (x, θ)**:** $\theta \sim p(\theta) \rightarrow x \sim p(x \mid \theta)$
- **marginal** (x, θ)**:** $\theta \sim p(\theta) \quad \theta' \sim p(\theta) \rightarrow x \sim p(x \mid \theta')$

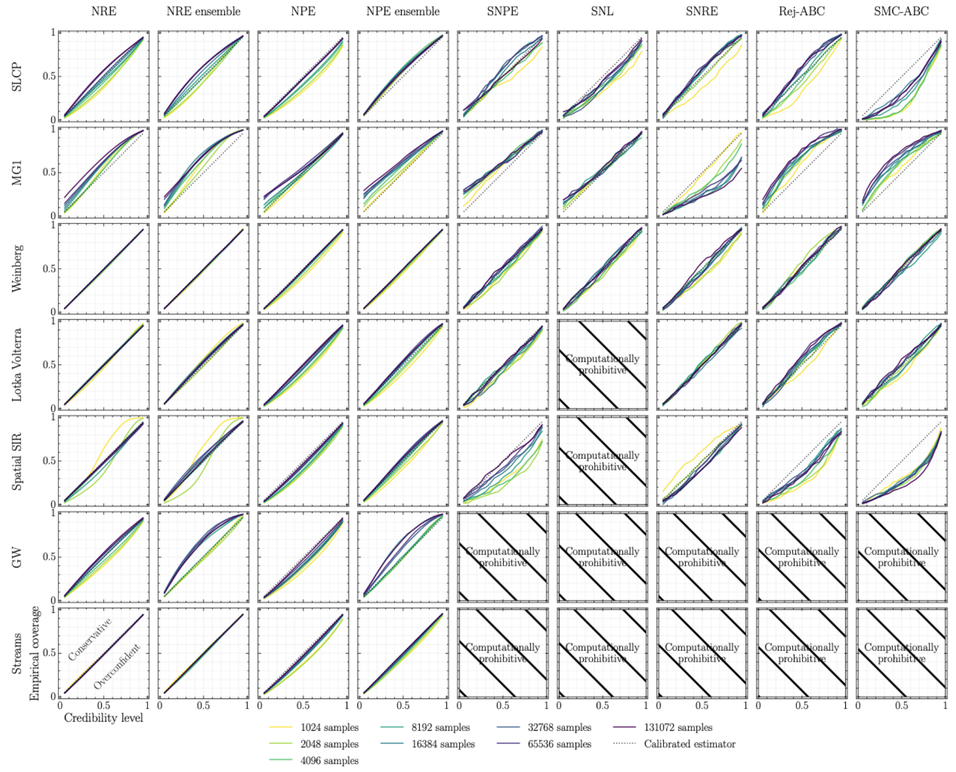
And with this we can setup the same classification regime to learn

$$\frac{p(x, \theta)}{p(x)p(\theta)}$$

Neural Ratio Estimation (NRE) (for Bayes)

Benchmarks

Methods for Simulation-based inference behave well in different regimes. Extensive Benchmarking to guide you.



https://arxiv.org/pdf/2110.06581

Software Tools for SBI

sbi Search mackelab/sbi v0.21.0 ☆373 87

`sbi` : simulation-based inference

`sbi` : A Python toolbox for simulation-based inference.

```
prior = BoxUniform(low=zeros(2), high=2*ones(2))   # Box prior [0,2]x[0,2]
def simulator(θ): return θ + 0.1*randn_like(θ)     # Gaussian in 2D
posterior = infer(simulator, prior, method='SNPE', num_simulations=500)
```

```
Running 500 simulations.: 100%|██████████| 500/500 [00:00<00:00, 57141.55it/s]
Training neural network. Epochs trained:  90
```

```
samples = posterior.sample((1000,), x=observed)
```

Table of contents

- Motivation and approach
- Publications
  - Posterior estimation (SNPE)
  - Likelihood-estimation (SNLE)
  - Likelihood-ratio-estimation (SNRE)
  - Utilities

Inference can be run in a single line of code:

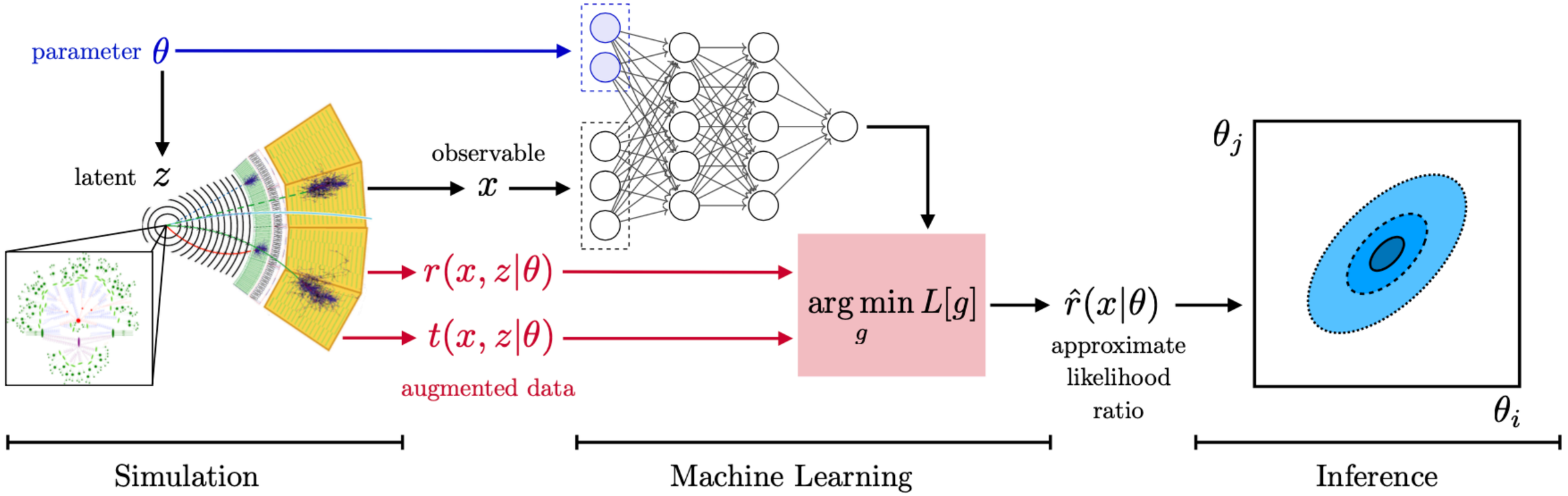
```
posterior = infer(simulator, prior, method='SNPE', num_simulations=1000)
```

- To learn about the general motivation behind simulation-based inference, and the inference methods included in `sbi`, read on below.
- For example applications to canonical problems in neuroscience, browse the recent research article Training density estimators to identify mechanistic models of neural dynamics.

Display a menu

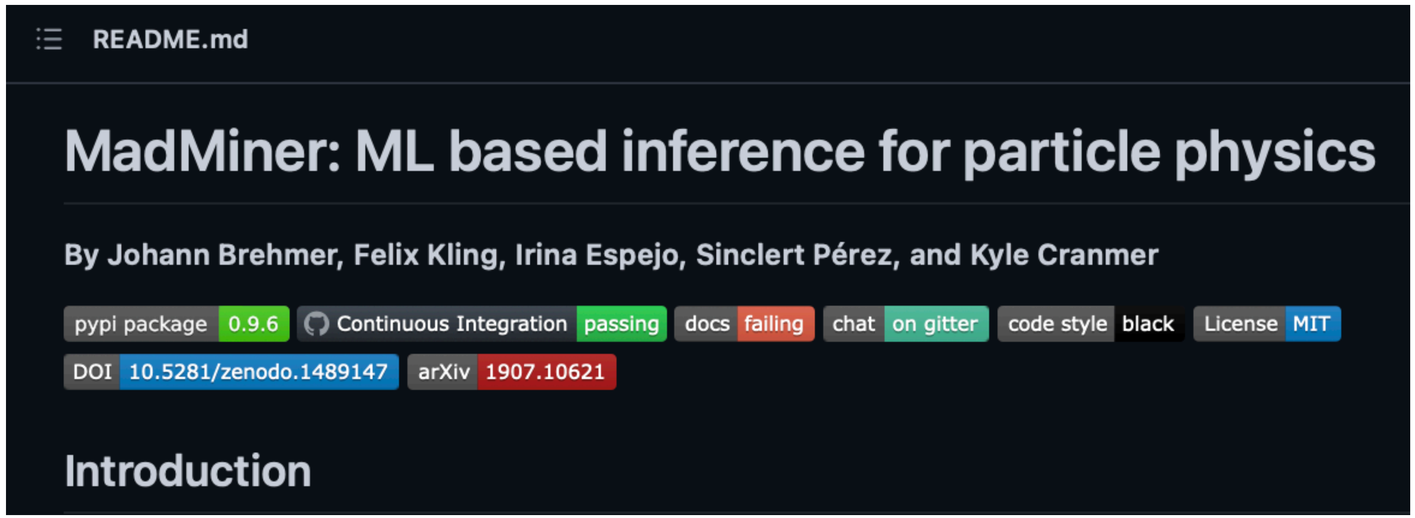
```
prior = BoxUniform(low=zeros(2), high=2*ones(2))   # Box prior [0,2]x[0,2]
def simulator(θ): return θ + 0.1*randn_like(θ)     # Gaussian in 2D
```

In Particle Physics



In Particle Physics

MadMiner: a package for end-to-end SBI in particle physics



https://madminer.readthedocs.io/en/latest/?badge=latest

Bonus: Data Efficiency & Mining Gold

Sounds too good to be true?

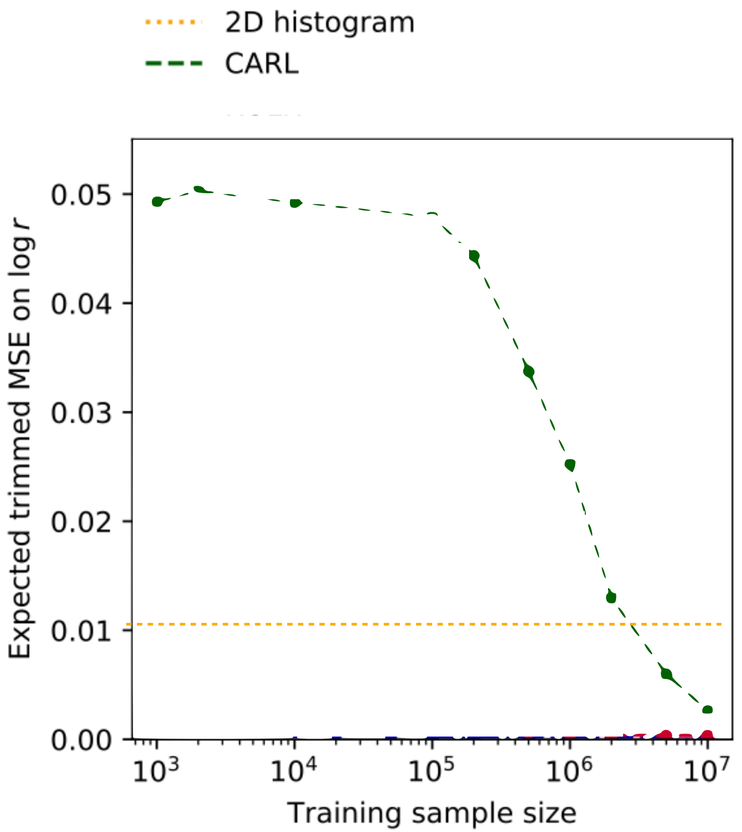
This is nice in theory. But of course there is a catch.

This requires an very large amount of MC to become better than the traditional approach.

Higgs Example:

- **10 Million samples per theory point**
- **O(10)-O(100) possible theories**

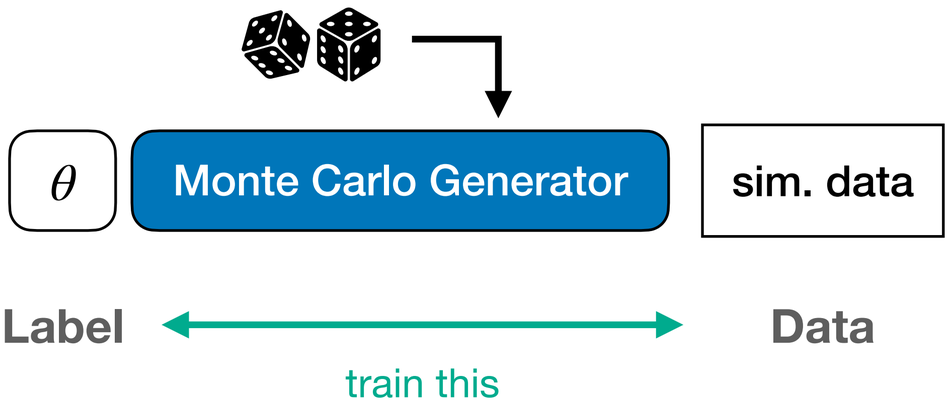
Can be computationally prohibitive

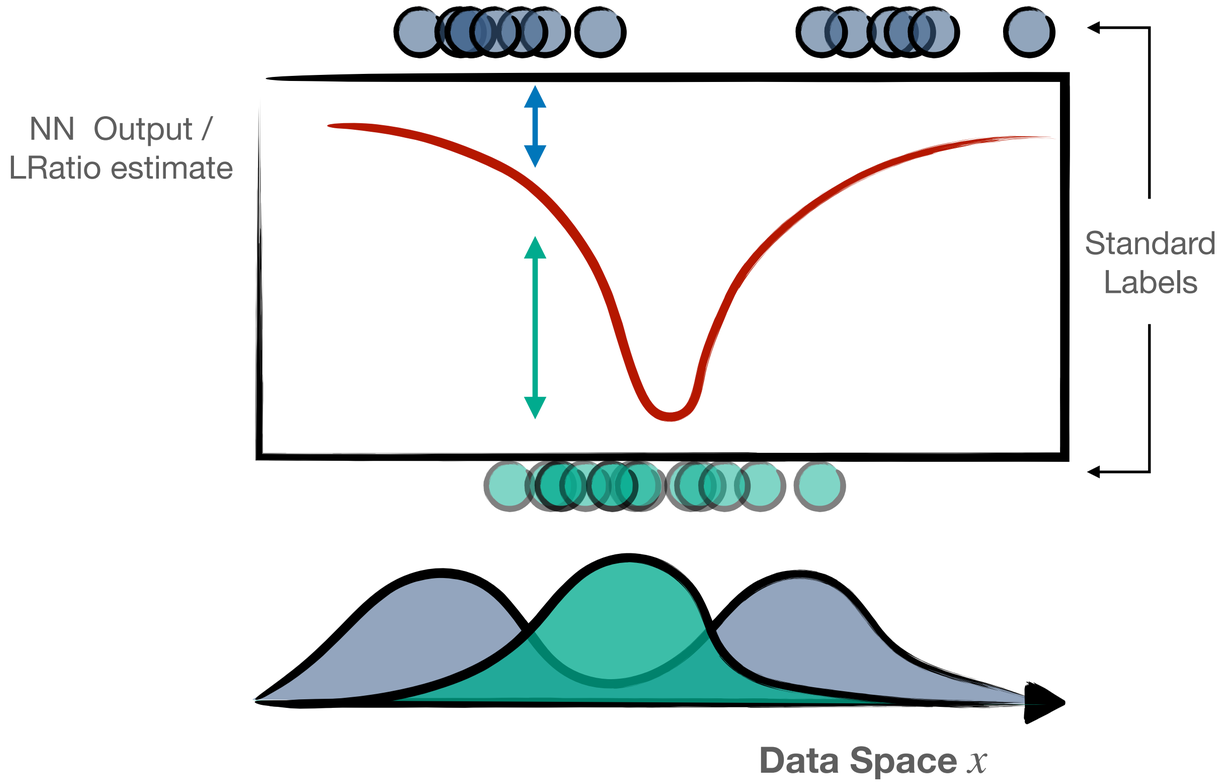


Increasing Data Efficiency

The reason training converges slowly is again distance between theory and data.

Training = “tug of war” between samples



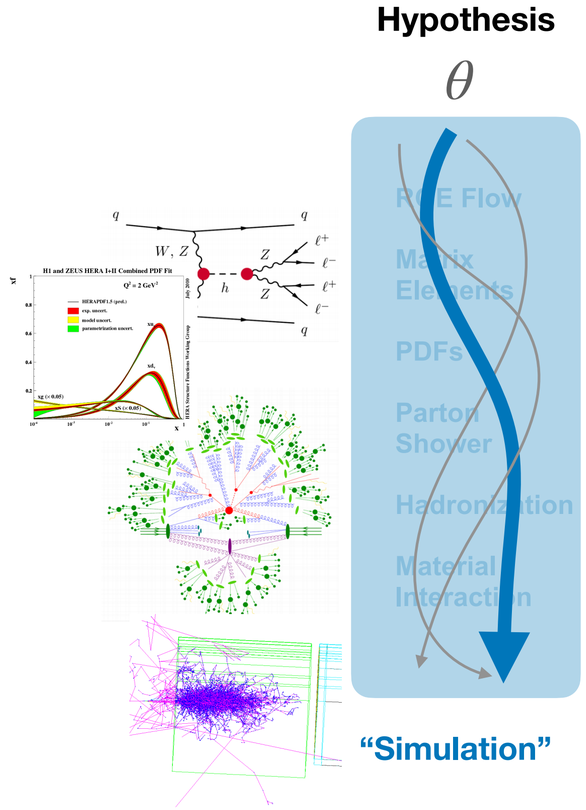


More tricks

To increase data efficiency we can use more tricks. Remember:

→ We can calculate $p(x, z)$ for any given path through our MC.

→ it's just the integral that kills us



$$p(x \mid \theta) = \int dz \; \frac{p(x \mid z_h)\; p(z_h \mid z_p)\; p(z_p \mid \theta)}{p(x, z \mid \theta)}$$

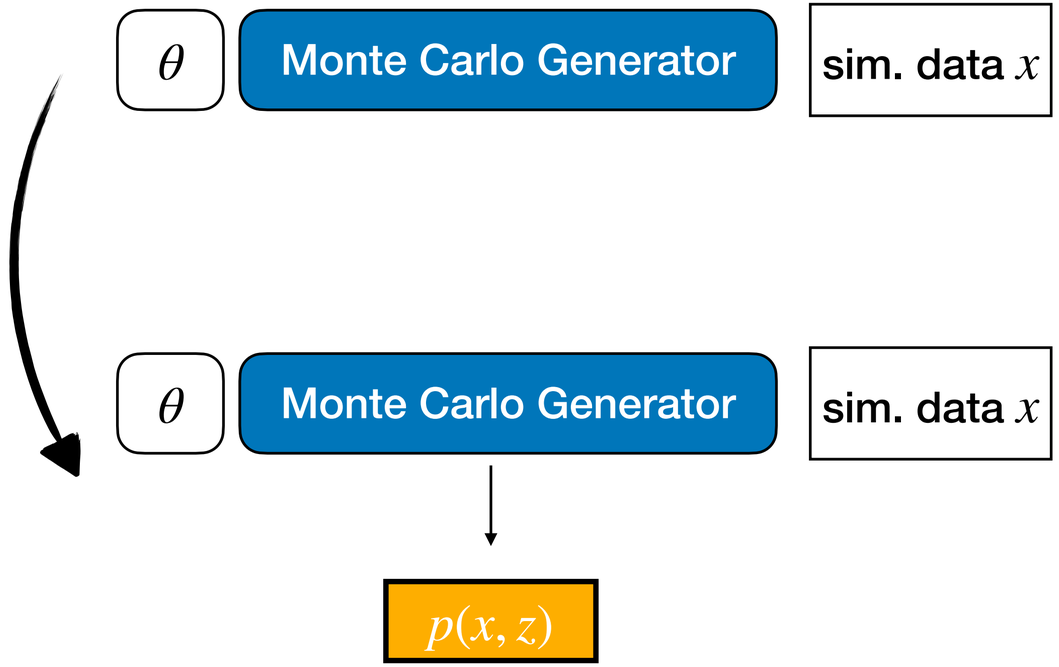
NOT OK

OK

Mining Gold

By adapting our MC to not only spit out the simulated event **but also the "path probability"** $p(x, z \mid \theta)$ we are extracting valuable information for ML training

→ exploit our physics knowledge in ML training



"Mining Gold"

Interesting Property

Turns out: the true likelihood ratio, and the path-dependent ratio are closely related

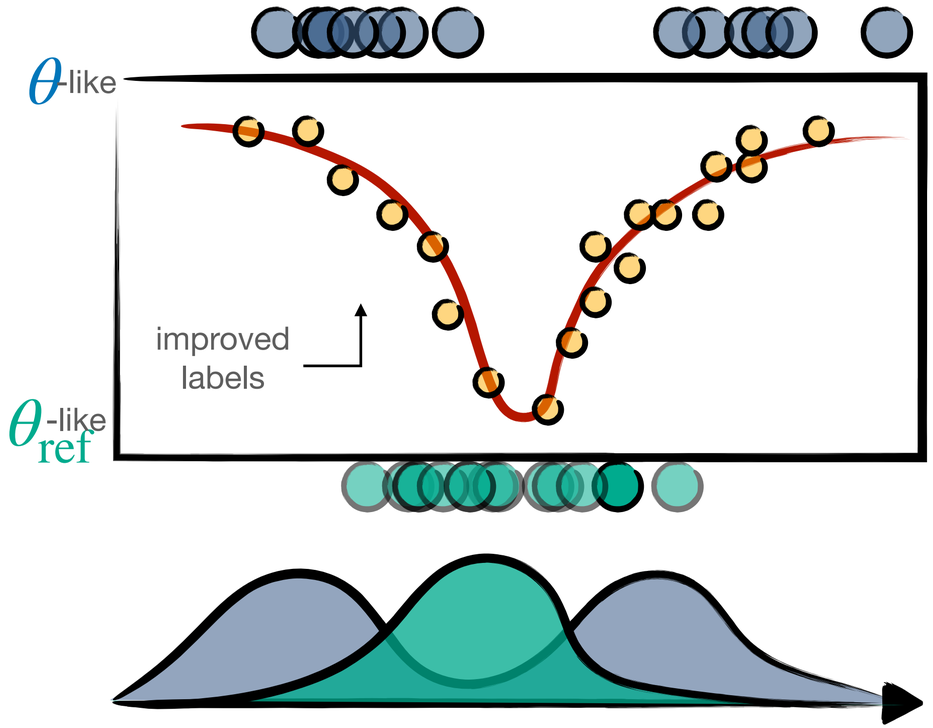
NOT OK $\dfrac{p(x \mid \theta)}{p(x \mid \theta_0)}$ OK $\dfrac{p(x, z \mid \theta)}{p(x, z \mid \theta_0)}$

But…

$$\frac{p(x \mid \theta)}{p(x \mid \theta_0)} = \mathbb{E}_{p(z|x)} \frac{p(x, z \mid \theta_1)}{p(x, z \mid \theta_0)}$$

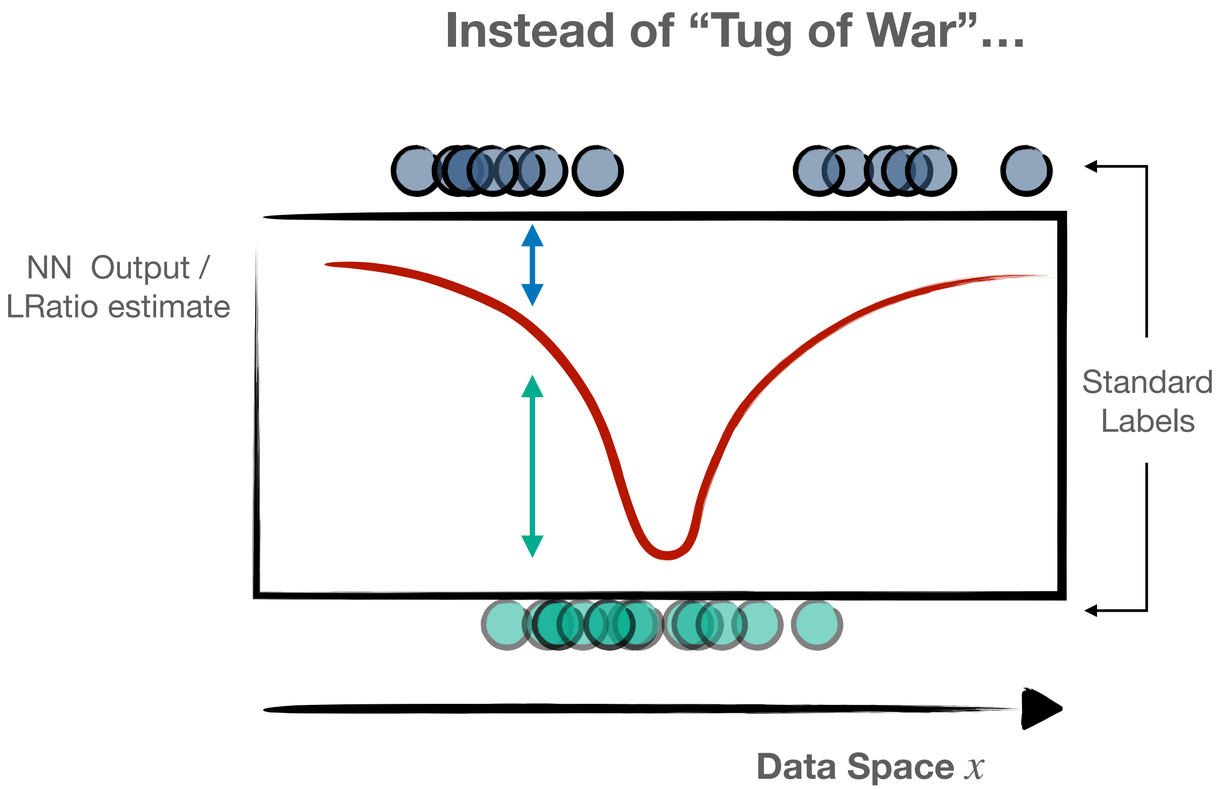
Interesting Property

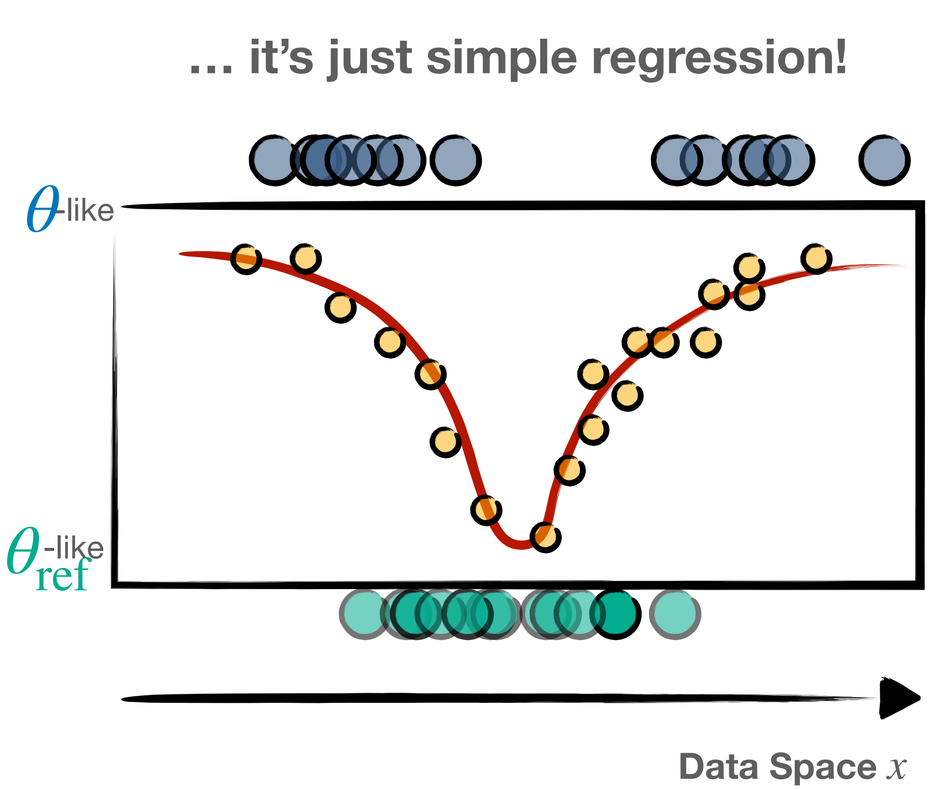
The paths through the simulation chain add “theory noise” but in expectation average out to be the true likelihood ratio!



Better Training

That means: by supercharging our MC generators w/ “Mining Gold” we can extract much much better training data.

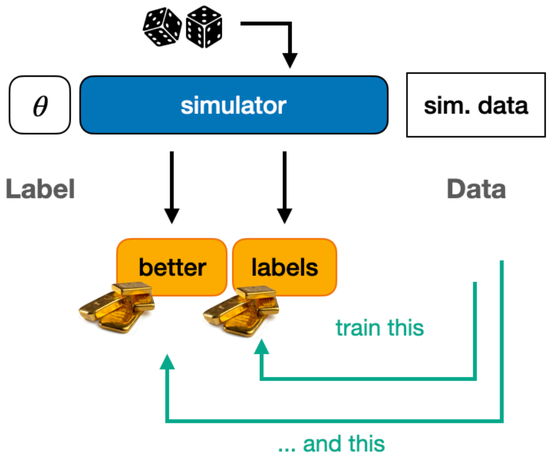


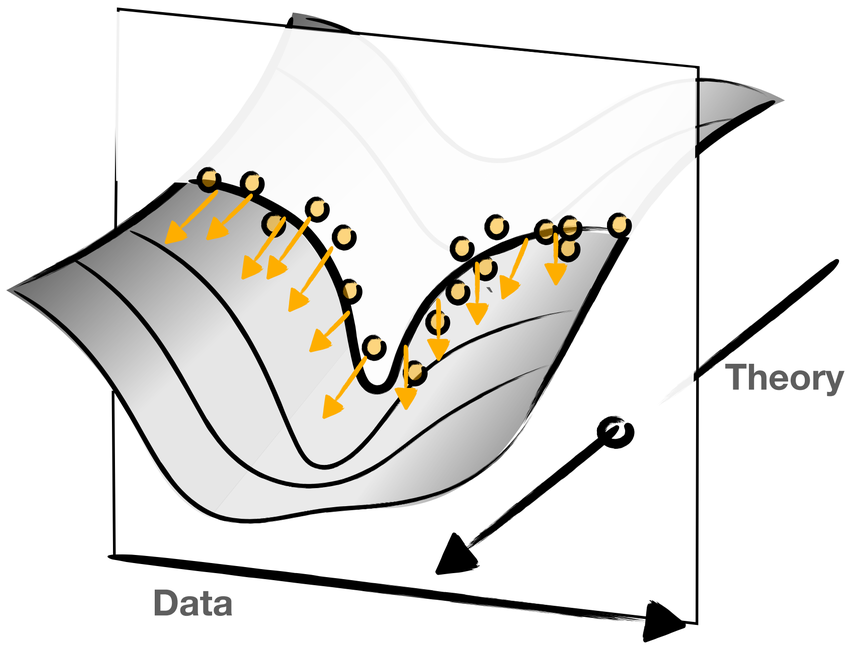


Gradients for the win

Idea: derivatives of our noisy labels w.r.t theory parameters are additional powerful labels we can extract from the simulator…

$$\nabla \log p(x, z \mid \theta)$$

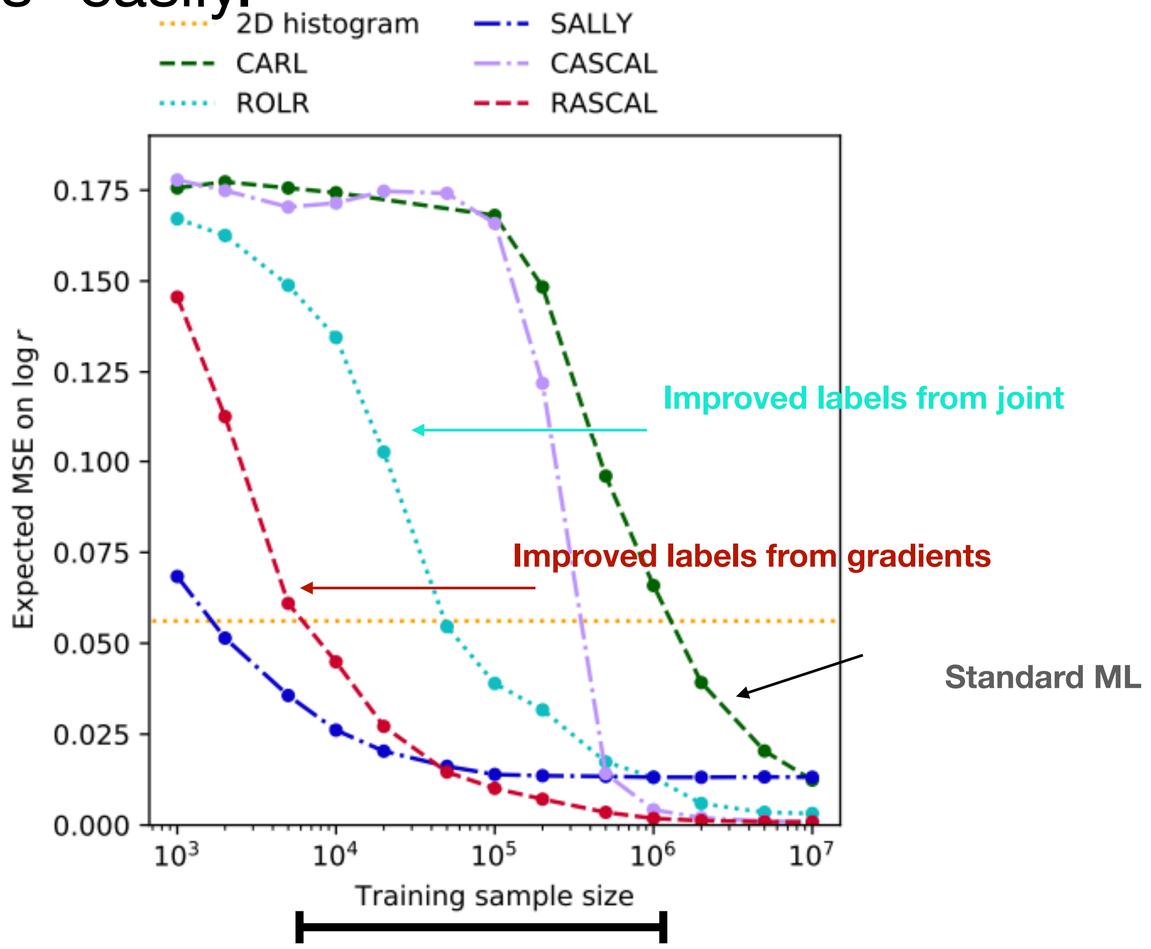




Needs a Differentiable Simulator!

Impressive Gains in Data Efficiency

This gives us the ability to train with 100x-1000x less training data.
Beat “traditional analysis” easily.

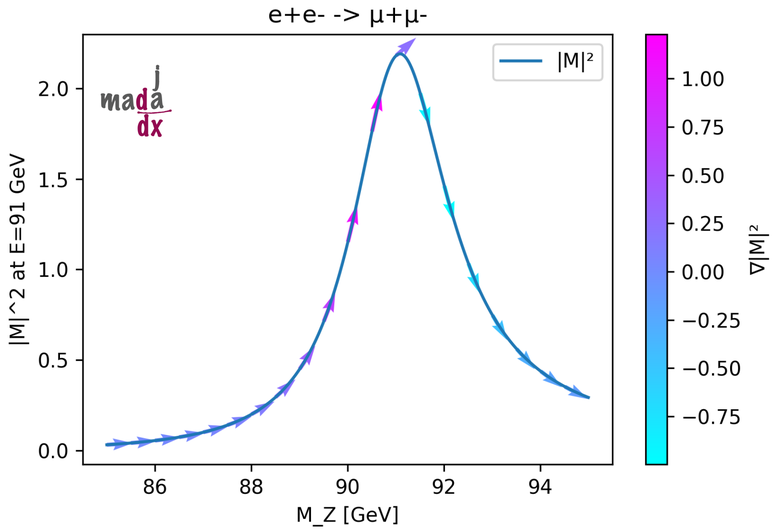


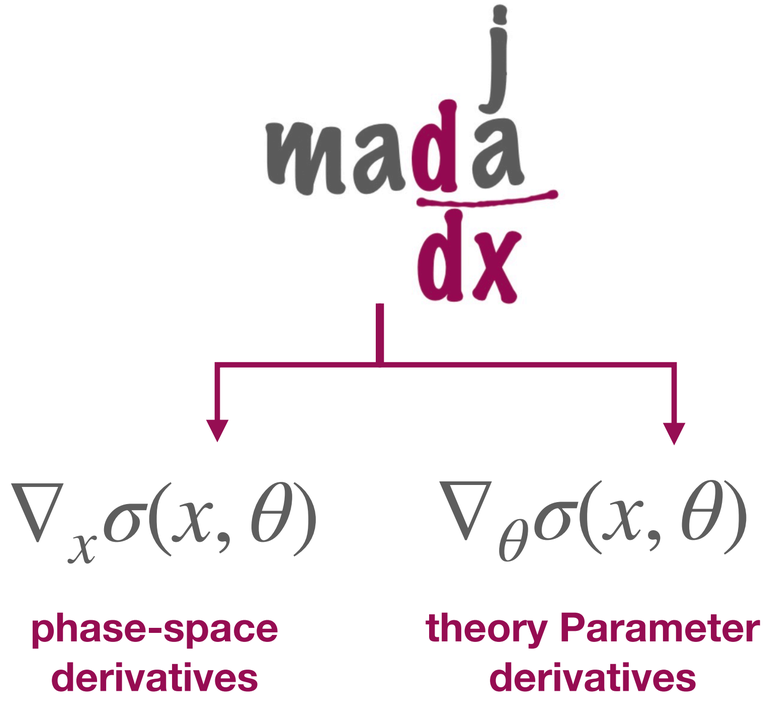
Differentiable Simulators

For these more advanced methods, it's key to have differentiable simulators in order to extract the max. amount of information from the generating process.

As we've seen: a wide & active field.

In particle physics pursued e.g. through **MadJax**





Summary

Advanced scientific predictions are rarely computable using pen & paper.

The rise of numerical and stochastic simulation

For these situations a new breed of statistical methods is emerging that fuses ML with domain simulators to extract reliable statistical inferences despite lack of likelihood

End.