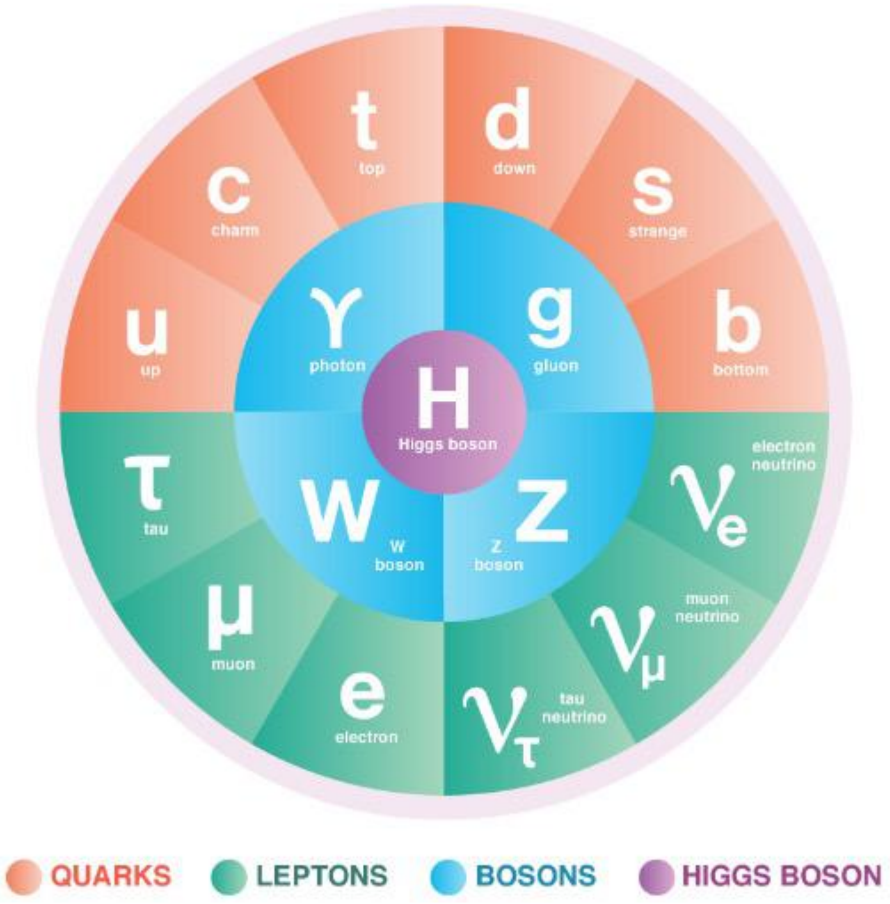
Using Gradients to Get More Out of High Energy Physics

Michael Kagan, SLAC

CMU STAMPS Seminar
November 10, 2023

High Energy Physics – What We Know



$$
\begin{gathered}
-\tfrac{1}{2}\partial_\nu g^a_\mu \partial_\nu g^a_\mu - g_s f^{abc}\partial_\mu g^a_\nu g^b_\mu g^c_\nu - \tfrac{1}{4}g_s^2 f^{abc} f^{ade} g^b_\mu g^c_\nu g^d_\mu g^e_\nu + \\
\tfrac{1}{2} i g_s^2 (\bar{q}^\sigma_i \gamma^\mu q^\sigma_j) g^a_\mu + \bar{G}^a \partial^2 G^a + g_s f^{abc}\partial_\mu \bar{G}^a G^b g^c_\mu - \partial_\nu W^+_\mu \partial_\nu W^-_\mu - \\
M^2 W^+_\mu W^-_\mu - \tfrac{1}{2}\partial_\nu Z^0_\mu \partial_\nu Z^0_\mu - \tfrac{1}{2c_w^2} M^2 Z^0_\mu Z^0_\mu - \tfrac{1}{2}\partial_\mu A_\nu \partial_\mu A_\nu - \tfrac{1}{2}\partial_\mu H \partial_\mu H - \\
\tfrac{1}{2} m_h^2 H^2 - \partial_\mu \phi^+ \partial_\mu \phi^- - M^2 \phi^+ \phi^- - \tfrac{1}{2}\partial_\mu \phi^0 \partial_\mu \phi^0 - \tfrac{1}{2c_w^2} M \phi^0 \phi^0 - \beta_h [\tfrac{2M^2}{g^2} + \\
\tfrac{2M}{g} H + \tfrac{1}{2}(H^2 + \phi^0\phi^0 + 2\phi^+\phi^-)] + \tfrac{2M^4}{g^2}\alpha_h - i g c_w [\partial_\nu Z^0_\mu (W^+_\mu W^-_\nu - \\
W^+_\nu W^-_\mu) - Z^0_\nu (W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + Z^0_\mu (W^+_\nu \partial_\nu W^-_\mu - \\
W^-_\nu \partial_\nu W^+_\mu)] - i g s_w [\partial_\nu A_\mu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - A_\nu (W^+_\mu \partial_\nu W^-_\mu - \\
W^-_\mu \partial_\nu W^+_\mu) + A_\mu (W^+_\nu \partial_\nu W^-_\mu - W^-_\nu \partial_\nu W^+_\mu)] - \tfrac{1}{2} g^2 W^+_\mu W^-_\mu W^+_\nu W^-_\nu + \\
\tfrac{1}{2} g^2 W^+_\mu W^-_\nu W^+_\mu W^-_\nu + g^2 c_w^2 (Z^0_\mu W^+_\mu Z^0_\nu W^-_\nu - Z^0_\mu Z^0_\mu W^+_\nu W^-_\nu) + \\
g^2 s_w^2 (A_\mu W^+_\mu A_\nu W^-_\nu - A_\mu A_\mu W^+_\nu W^-_\nu) + g^2 s_w c_w [A_\mu Z^0_\nu (W^+_\mu W^-_\nu - \\
W^+_\nu W^-_\mu) - 2 A_\mu Z^0_\mu W^+_\nu W^-_\nu] - g\alpha [H^3 + H\phi^0\phi^0 + 2H\phi^+\phi^-] - \\
\tfrac{1}{8} g^2 \alpha_h [H^4 + (\phi^0)^4 + 4(\phi^+\phi^-)^2 + 4(\phi^0)^2\phi^+\phi^- + 4H^2\phi^+\phi^- + 2(\phi^0)^2 H^2] - \\
g M W^+_\mu W^-_\mu H - \tfrac{1}{2} g \tfrac{M}{c_w^2} Z^0_\mu Z^0_\mu H - \tfrac{1}{2} i g [W^+_\mu (\phi^0 \partial_\mu \phi^- - \phi^- \partial_\mu \phi^0) - \\
W^-_\mu (\phi^0 \partial_\mu \phi^+ - \phi^+ \partial_\mu \phi^0)] + \tfrac{1}{2} g [W^+_\mu (H \partial_\mu \phi^- - \phi^- \partial_\mu H) - W^-_\mu (H \partial_\mu \phi^+ - \\
\phi^+ \partial_\mu H)] + \tfrac{1}{2} g \tfrac{1}{c_w} (Z^0_\mu (H \partial_\mu \phi^0 - \phi^0 \partial_\mu H) - i g \tfrac{s_w^2}{c_w} M Z^0_\mu (W^+_\mu \phi^- - W^-_\mu \phi^+) + \\
i g s_w M A_\mu (W^+_\mu \phi^- - W^-_\mu \phi^+) - i g \tfrac{1-2c_w^2}{2c_w} Z^0_\mu (\phi^+ \partial_\mu \phi^- - \phi^- \partial_\mu \phi^+) + \\
i g s_w A_\mu (\phi^+ \partial_\mu \phi^- - \phi^- \partial_\mu \phi^+) - \tfrac{1}{4} g^2 W^+_\mu W^-_\mu [H^2 + (\phi^0)^2 + 2\phi^+\phi^-] - \\
\tfrac{1}{4} g^2 \tfrac{1}{c_w^2} Z^0_\mu Z^0_\mu [H^2 + (\phi^0)^2 + 2(2s_w^2 - 1)^2 \phi^+\phi^-] - \tfrac{1}{2} g^2 \tfrac{s_w^2}{c_w} Z^0_\mu \phi^0 (W^+_\mu \phi^- + \\
W^-_\mu \phi^+) - \tfrac{1}{2} i g^2 \tfrac{s_w^2}{c_w} Z^0_\mu H (W^+_\mu \phi^- - W^-_\mu \phi^+) + \tfrac{1}{2} g^2 s_w A_\mu \phi^0 (W^+_\mu \phi^- + \\
W^-_\mu \phi^+) + \tfrac{1}{2} i g^2 s_w A_\mu H (W^+_\mu \phi^- - W^-_\mu \phi^+) - g^2 \tfrac{s_w}{c_w} (2c_w^2 - 1) Z^0_\mu A_\mu \phi^+ \phi^- - \\
g^1 s_w^2 A_\mu A_\mu \phi^+ \phi^- - \bar{e}^\lambda (\gamma\partial + m_e^\lambda) e^\lambda - \bar{\nu}^\lambda \gamma\partial \nu^\lambda - \bar{u}^\lambda_j (\gamma\partial + m_u^\lambda) u^\lambda_j - \\
\bar{d}^\lambda_j (\gamma\partial + m_d^\lambda) d^\lambda_j + i g s_w A_\mu [-(\bar{e}^\lambda \gamma^\mu e^\lambda) + \tfrac{2}{3}(\bar{u}^\lambda_j \gamma^\mu u^\lambda_j) - \tfrac{1}{3}(\bar{d}^\lambda_j \gamma^\mu d^\lambda_j)] + \\
\tfrac{ig}{4c_w} Z^0_\mu [(\bar{\nu}^\lambda \gamma^\mu (1+\gamma^5) \nu^\lambda) + (\bar{e}^\lambda \gamma^\mu (4s_w^2 - 1 - \gamma^5) e^\lambda) + (\bar{u}^\lambda_j \gamma^\mu (\tfrac{4}{3}s_w^2 - \\
1 - \gamma^5) u^\lambda_j) + (\bar{d}^\lambda_j \gamma^\mu (1 - \tfrac{8}{3}s_w^2 - \gamma^5) d^\lambda_j)] + \tfrac{ig}{2\sqrt{2}} W^+_\mu [(\bar{\nu}^\lambda \gamma^\mu (1+\gamma^5) e^\lambda) + \\
(\bar{u}^\lambda_j \gamma^\mu (1+\gamma^5) C_{\lambda\kappa} d^\kappa_j)] + \tfrac{ig}{2\sqrt{2}} W^-_\mu [(\bar{e}^\lambda \gamma^\mu (1+\gamma^5) \nu^\lambda) + (\bar{d}^\kappa_j C^\dagger_{\lambda\kappa} \gamma^\mu (1 + \\
\gamma^5) u^\lambda_j)] + \tfrac{ig}{2\sqrt{2}} \tfrac{m_e^\lambda}{M} [-\phi^+ (\bar{\nu}^\lambda (1-\gamma^5) e^\lambda) + \phi^- (\bar{e}^\lambda (1+\gamma^5) \nu^\lambda)] - \\
\tfrac{g}{2} \tfrac{m_e^\lambda}{M} [H(\bar{e}^\lambda e^\lambda) + i\phi^0 (\bar{e}^\lambda \gamma^5 e^\lambda)] + \tfrac{ig}{2M\sqrt{2}} \phi^+ [-m_d^\kappa (\bar{u}^\lambda_j C_{\lambda\kappa} (1-\gamma^5) d^\kappa_j) + \\
m_u^\lambda (\bar{u}^\lambda_j C_{\lambda\kappa} (1+\gamma^5) d^\kappa_j] + \tfrac{ig}{2M\sqrt{2}} \phi^- [m_d^\lambda (\bar{d}^\lambda_j C^\dagger_{\lambda\kappa} (1+\gamma^5) u^\kappa_j) - m_u^\kappa (\bar{d}^\lambda_j C^\dagger_{\lambda\kappa} (1 - \\
\gamma^5) u^\kappa_j] - \tfrac{g}{2} \tfrac{m_u^\lambda}{M} H (\bar{u}^\lambda_j u^\lambda_j) - \tfrac{g}{2} \tfrac{m_d^\lambda}{M} H (\bar{d}^\lambda_j d^\lambda_j) + \tfrac{ig}{2} \tfrac{m_u^\lambda}{M} \phi^0 (\bar{u}^\lambda_j \gamma^5 u^\lambda_j) - \\
\tfrac{ig}{2} \tfrac{m_d^\lambda}{M} \phi^0 (\bar{d}^\lambda_j \gamma^5 d^\lambda_j) + \bar{X}^+ (\partial^2 - M^2) X^+ + \bar{X}^- (\partial^2 - M^2) X^- + \bar{X}^0 (\partial^2 - \\
\tfrac{M^2}{c_w^2}) X^0 + \bar{Y} \partial^2 Y + i g c_w W^+_\mu (\partial_\mu \bar{X}^0 X^- - \partial_\mu \bar{X}^+ X^0) + i g s_w W^+_\mu (\partial_\mu \bar{Y} X^- - \\
\partial_\mu \bar{X}^+ Y) + i g c_w W^-_\mu (\partial_\mu \bar{X}^- X^0 - \partial_\mu \bar{X}^0 X^+) + i g s_w W^-_\mu (\partial_\mu \bar{X}^- Y - \\
\partial_\mu \bar{Y} X^+) + i g c_w Z^0_\mu (\partial_\mu \bar{X}^+ X^+ - \partial_\mu \bar{X}^- X^-) + i g s_w A_\mu (\partial_\mu \bar{X}^+ X^+ - \\
\partial_\mu \bar{X}^- X^-) - \tfrac{1}{2} g M [\bar{X}^+ X^+ H + \bar{X}^- X^- H + \tfrac{1}{c_w^2} \bar{X}^0 X^0 H] + \\
\tfrac{1-2c_w^2}{2c_w} i g M [\bar{X}^+ X^0 \phi^+ - \bar{X}^- X^0 \phi^-] + \tfrac{1}{2c_w} i g M [\bar{X}^0 X^- \phi^+ - \bar{X}^0 X^+ \phi^-] + \\
i g M s_w [\bar{X}^0 X^- \phi^+ - \bar{X}^0 X^+ \phi^-] + \tfrac{1}{2} i g M [\bar{X}^+ X^+ \phi^0 - \bar{X}^- X^- \phi^0]
\end{gathered}
$$

Image source: Symmetry Magazine

High Energy Physics – Big Questions

Why is the Higgs so light?

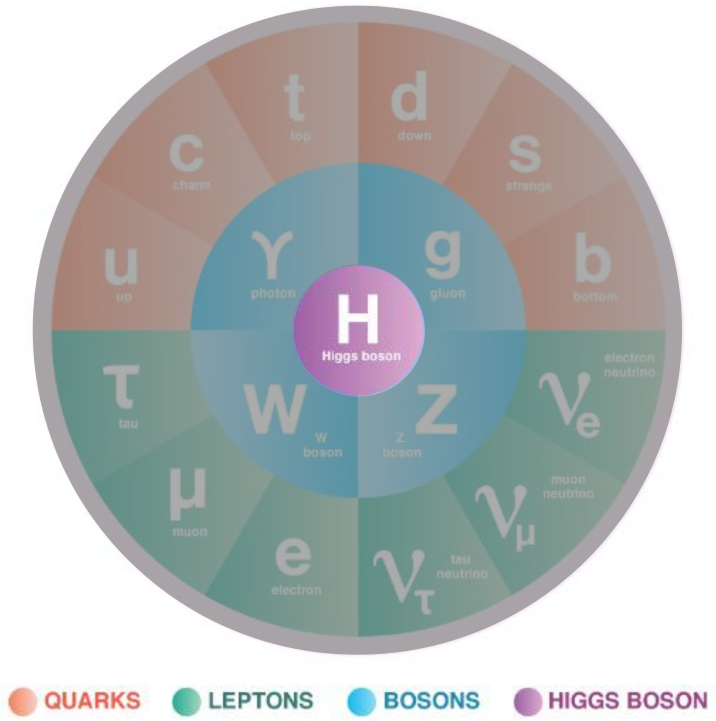


Image source: Symmetry Magazine

What is Dark Matter?
What is Dark Energy?

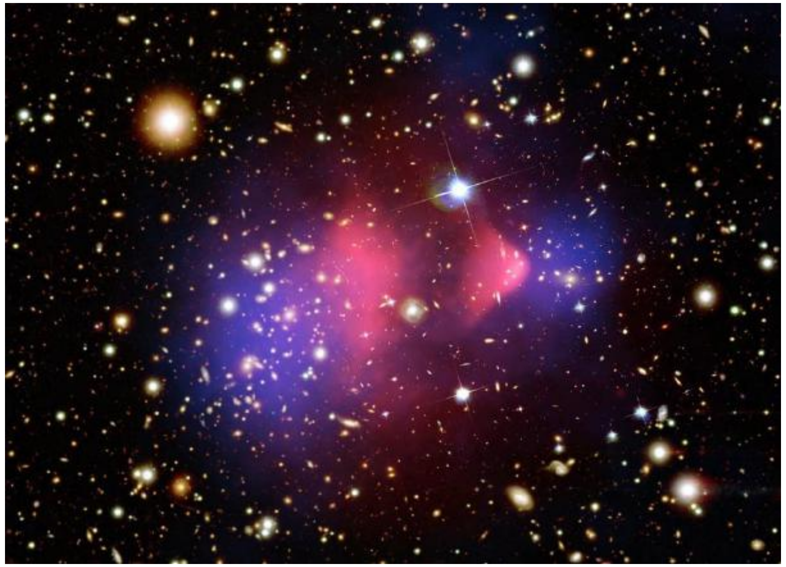


Image source: NASA/CXC/CFA/ M.MARKEVITCH

Why is there more matter than anti-matter in the universe?

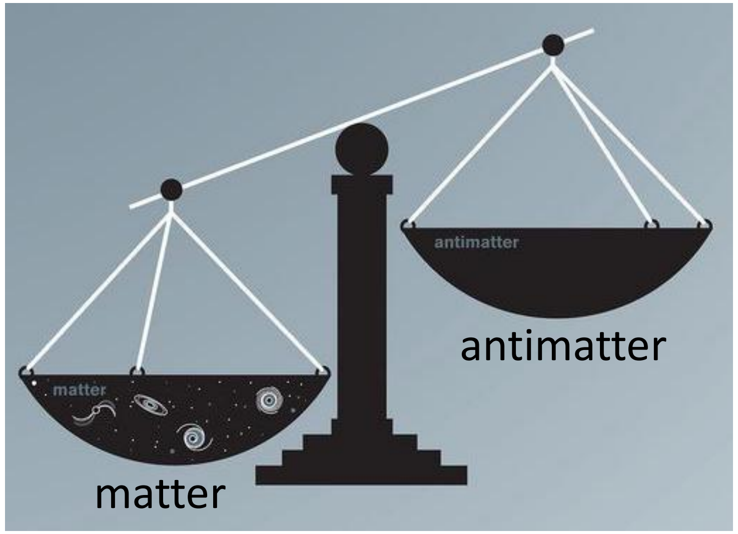
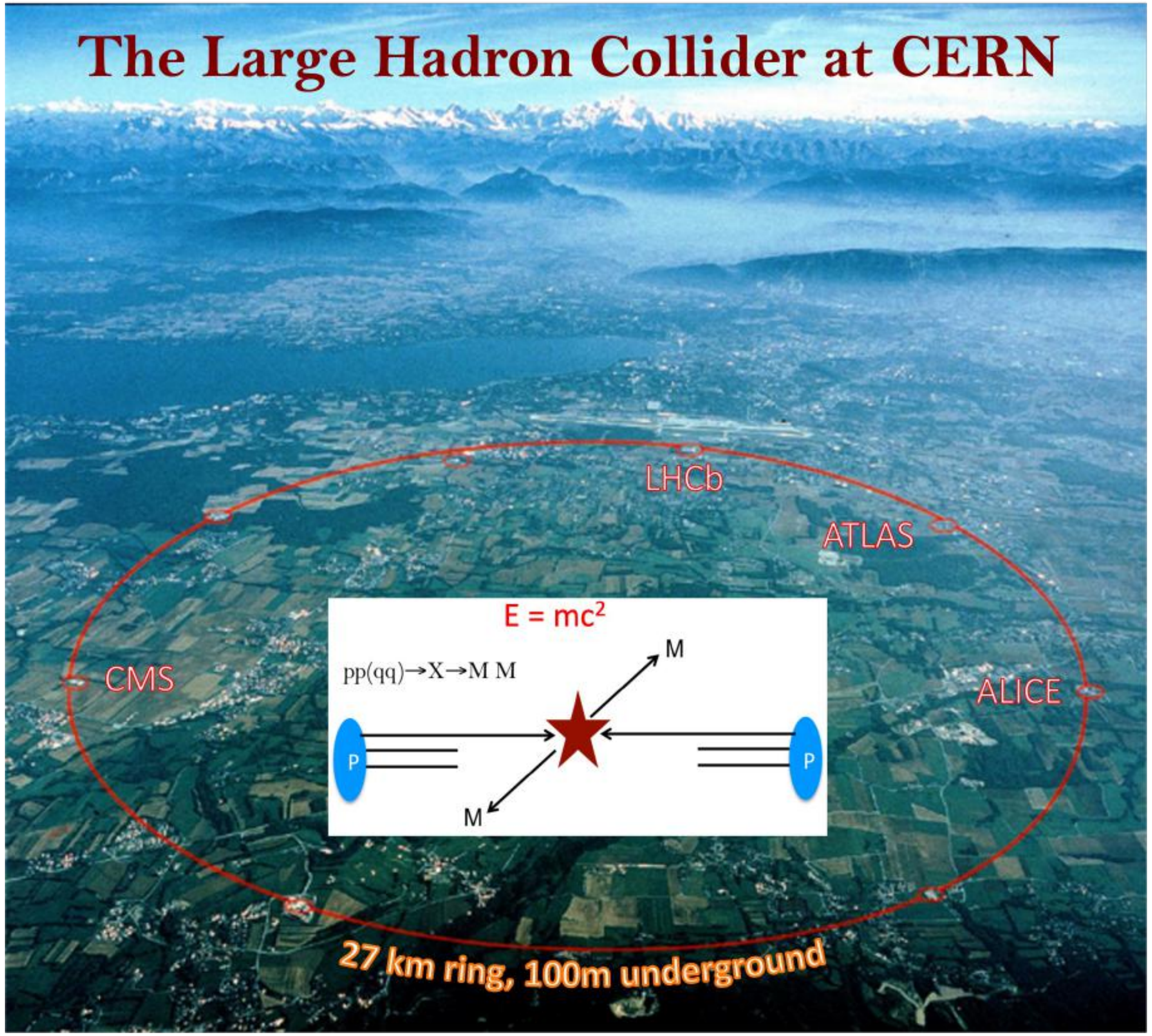
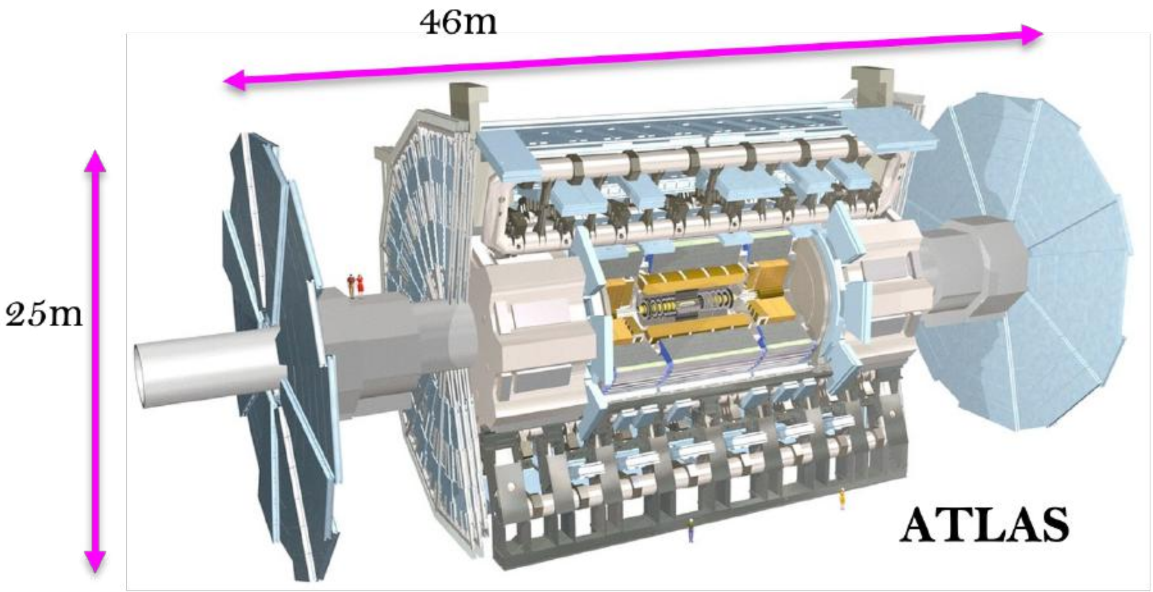
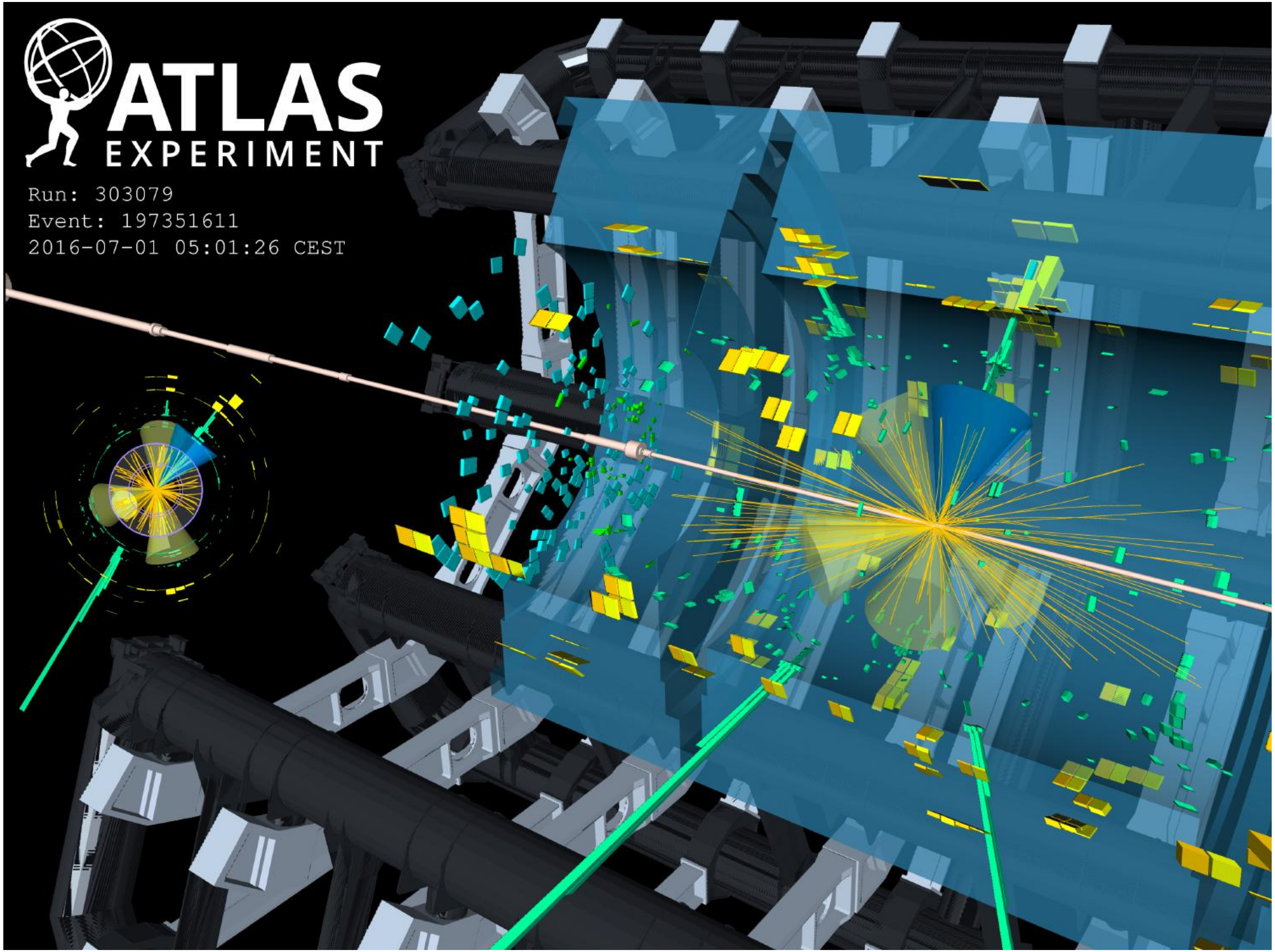


Image source: Symmetry Magazine

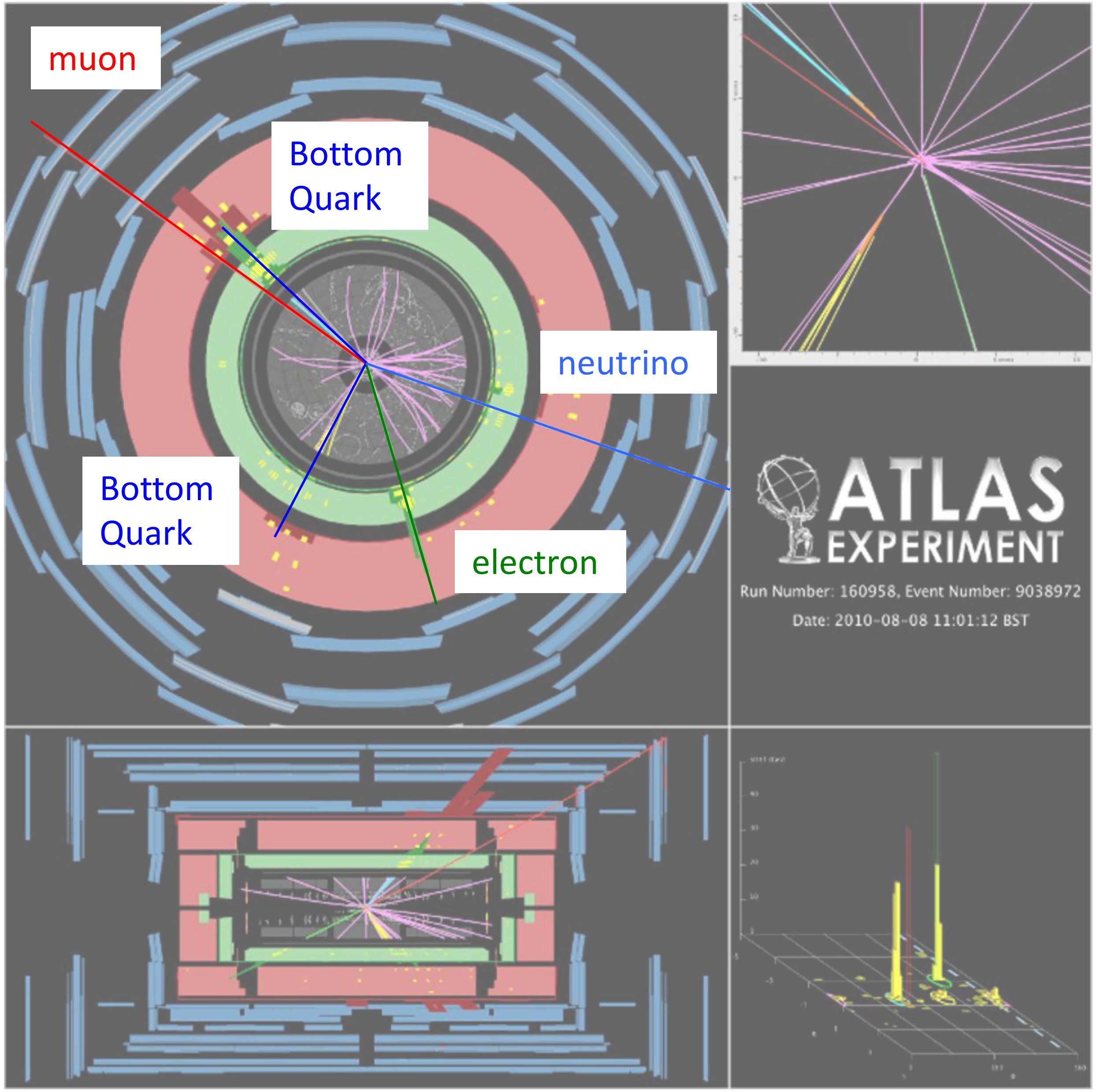
Studying Physics at the Smallest Scales

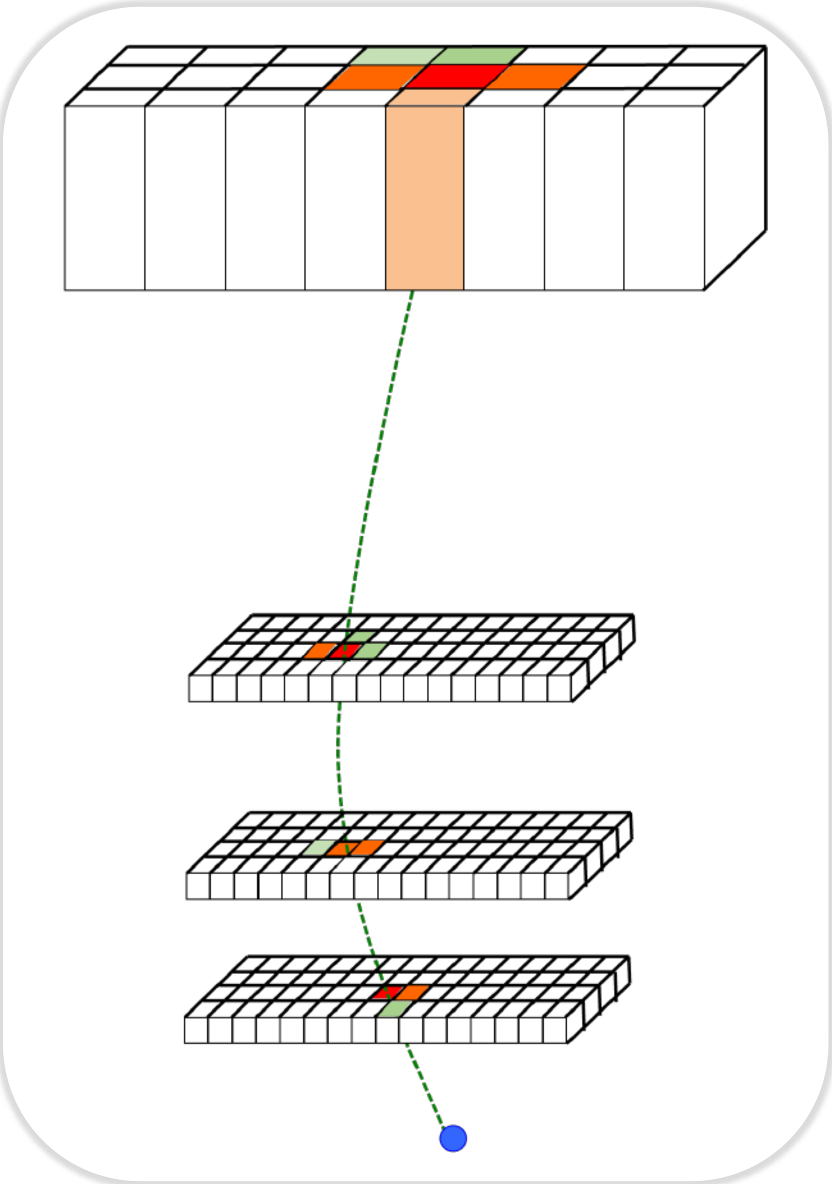




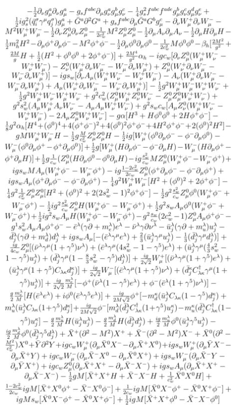

ATLAS
EXPERIMENT
Run: 303079
Event: 197351611
2016-07-01 05:01:26 CEST

Studying Collisions

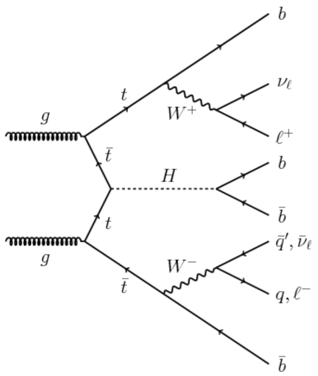




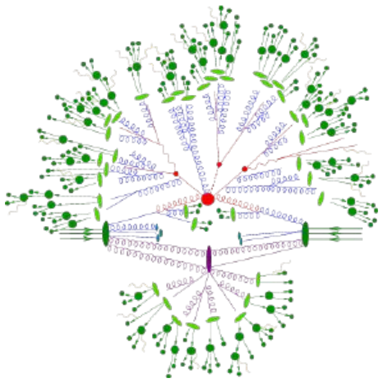
How doe we do all this? Simulations



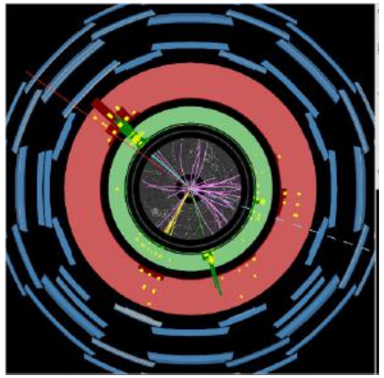
O(20) Fundamental physics parameters θ



O(10) particles



O(100) particles



O(10^8) detector elements

Deep knowledge of data generation process

Likelihood intractable,
but can simulate with high-fidelity simulators

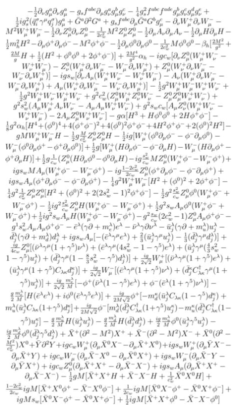
Data

Parameters of interest

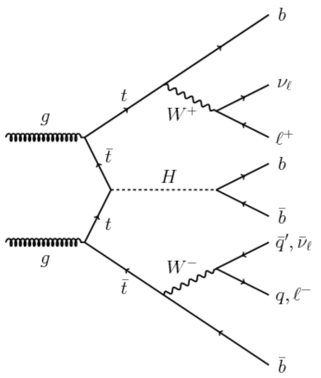
$$p(x|\theta) = \int p(x|z)p(z|\theta)dz$$

Latent variables
O(10^8) Dimensional

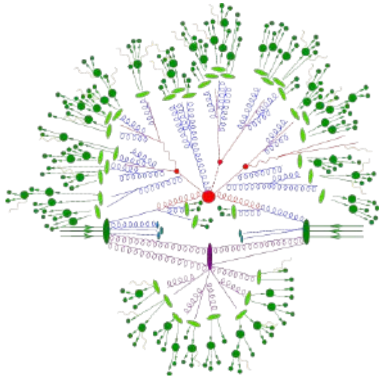
Data Analysis “Inverts” this Process



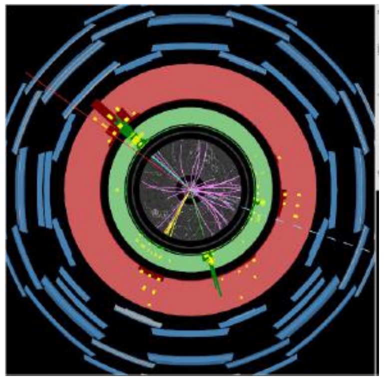
O(20) Fundamental physics parameters θ



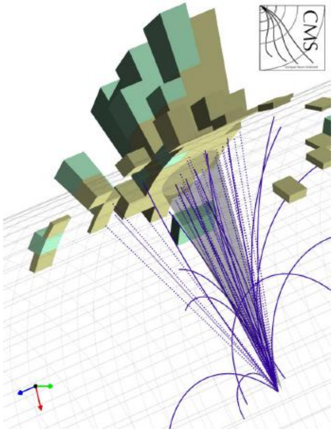
O(10) particles



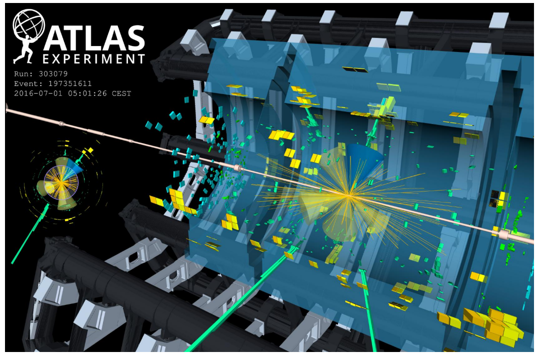
O(100) particles



O(10^8) detector elements

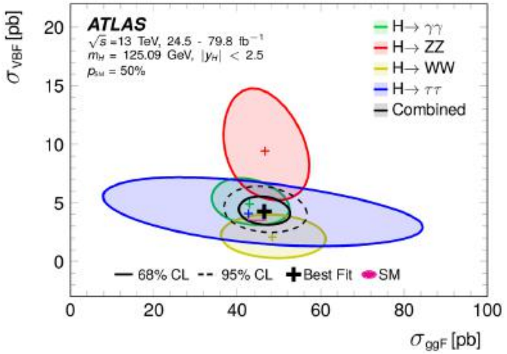


Reconstruct particles



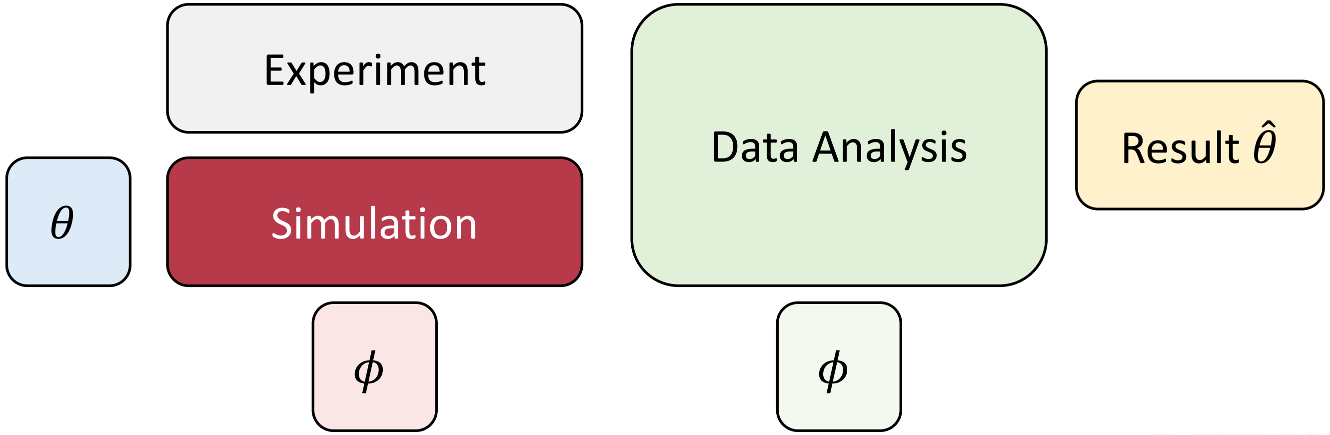
Reconstruct & Select interesting events

1909.02845



Statistical Inference

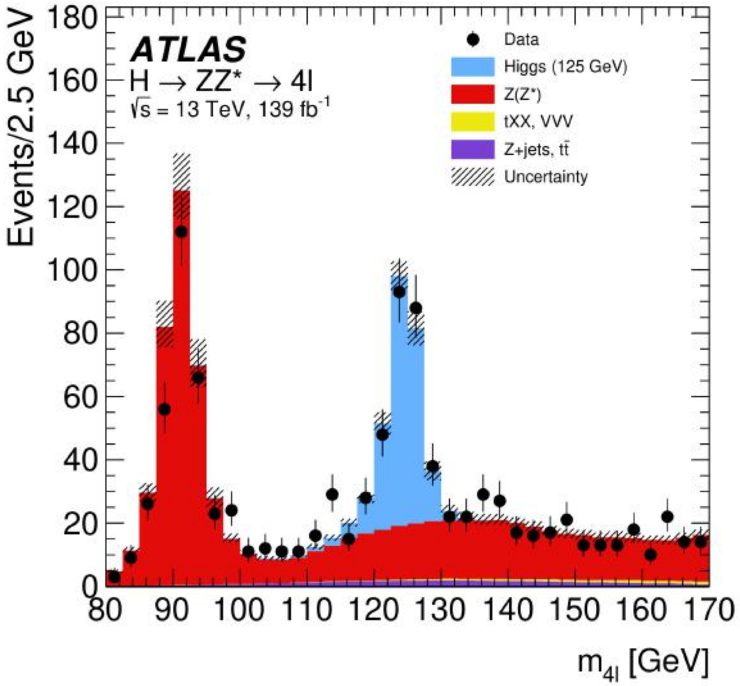
Data Analysis Workflow



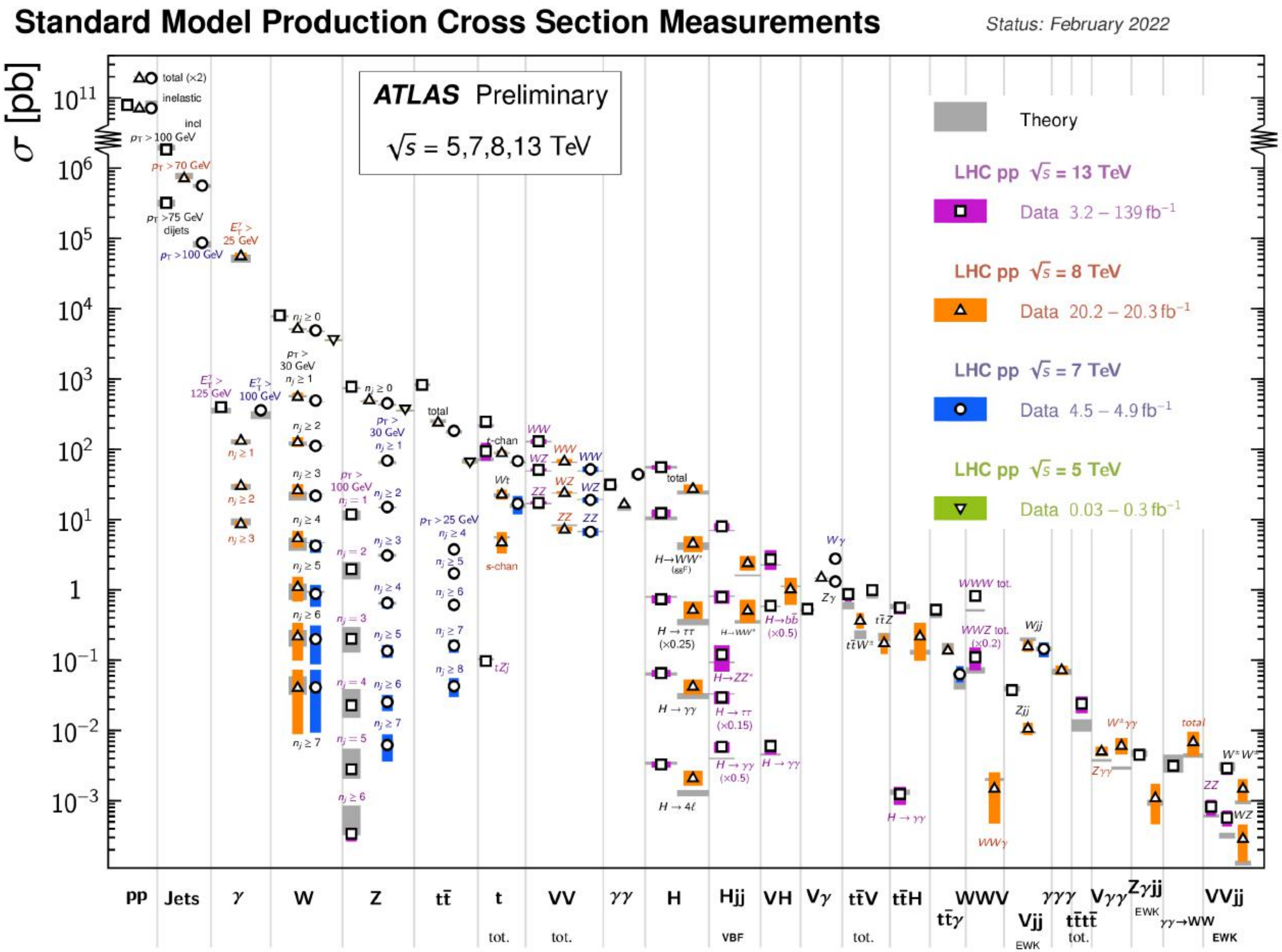
Summarize: Reduce 100M → 1 informative number

Statistical Inference: Compare simulation & data

EPJC 80 (2020) 942



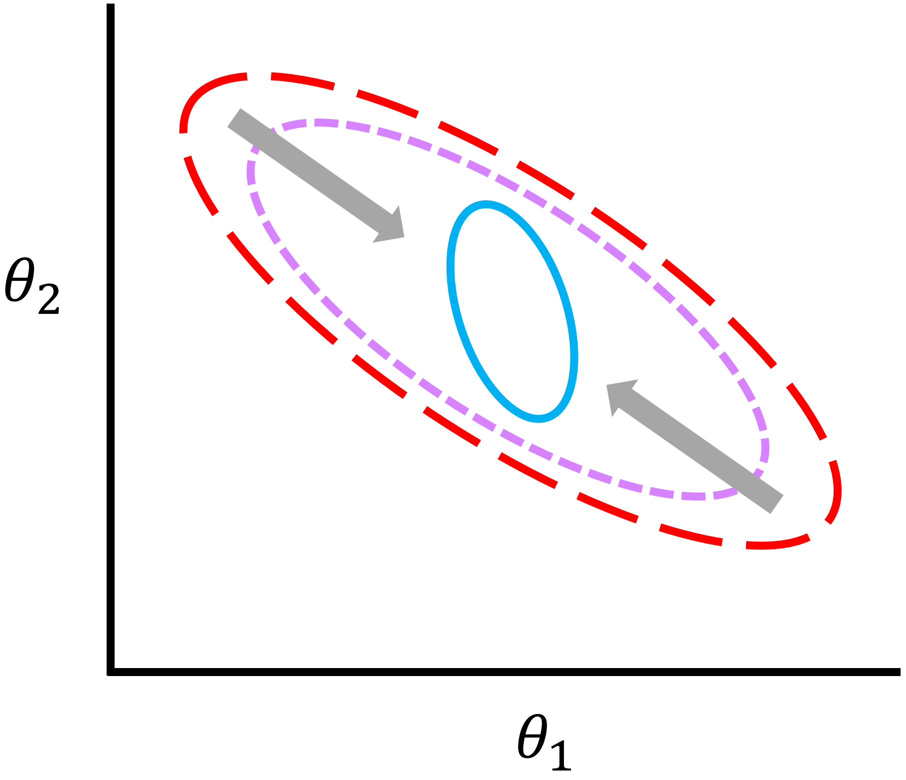
This work very well!



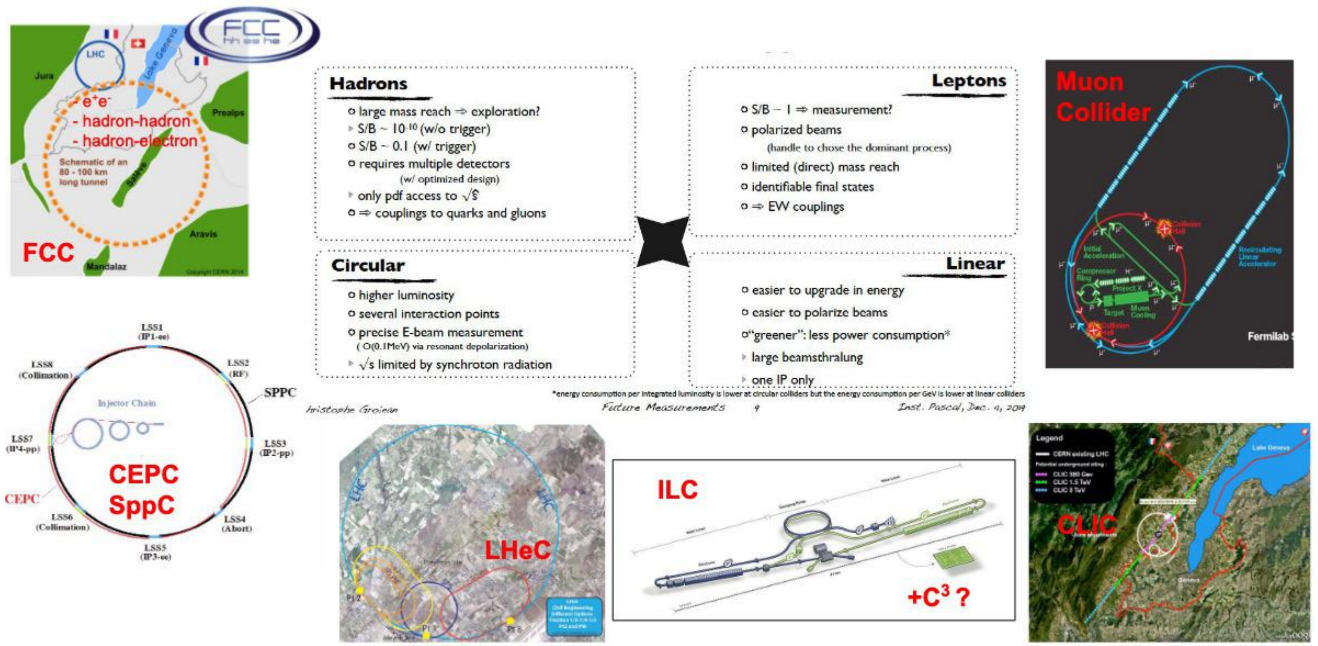
ATL-PHYS-PUB-2022-009

But we want to get the most out of our data!

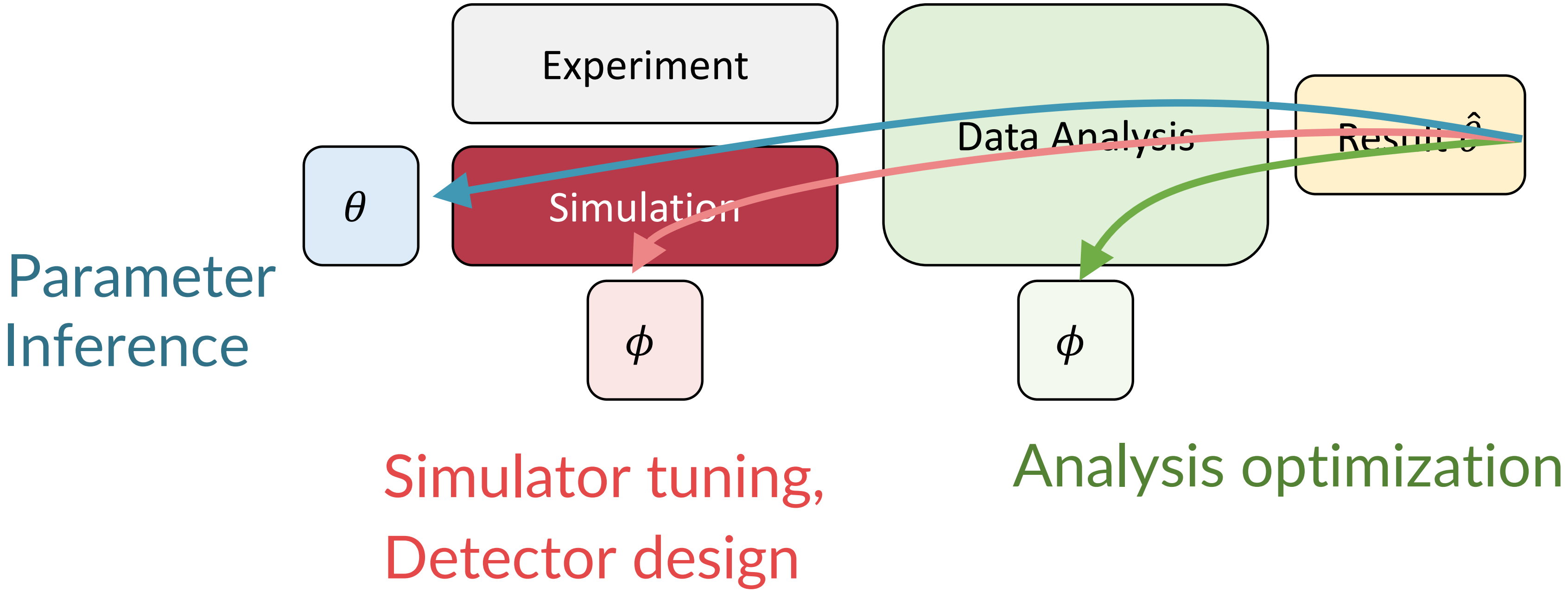
Improve Confidence Intervals



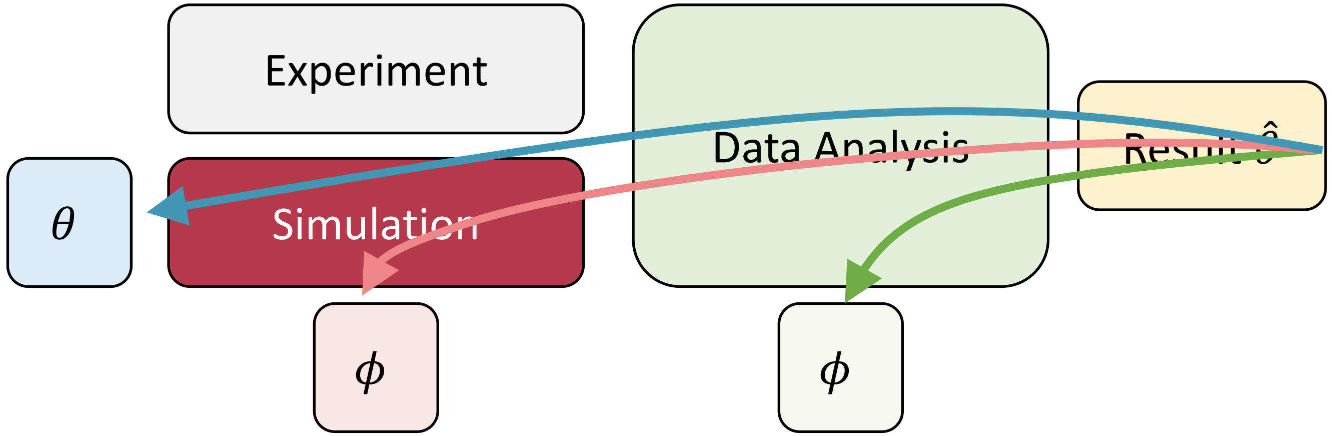
Which collider? Which Detectors?



What are we optimizing

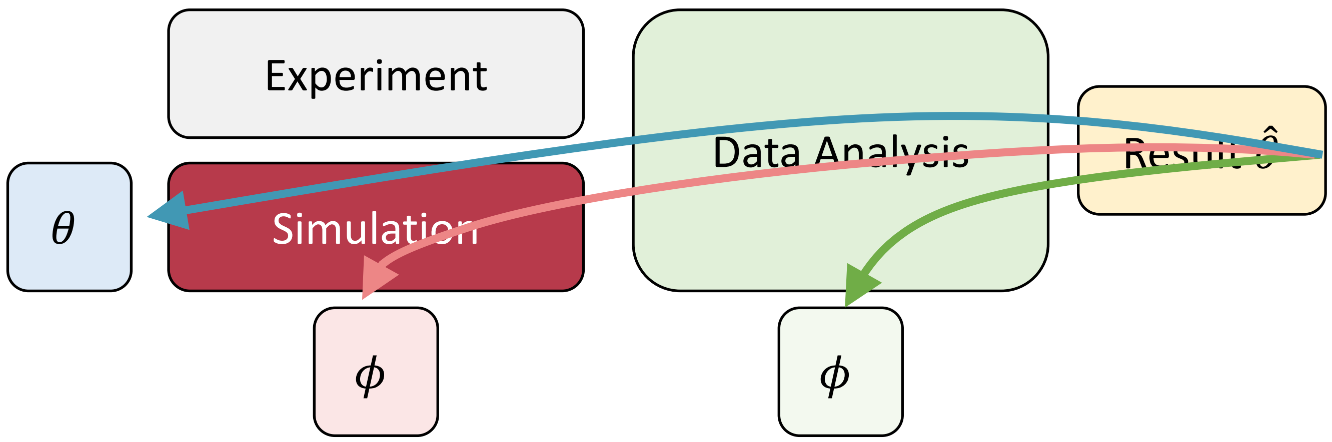


What are we optimizing



$$\min_{\phi} \mathbb{E}[f(x,\phi)] = \min_{\phi} \int f(x,\phi) p_{\phi}(x|\theta) dx$$

What are we optimizing



$$\min_{\phi} \mathbb{E}[f(x,\phi)] = \min_{\phi} \int f(x,\phi) p_{\phi}(x|\theta) dx$$

Statistical Analysis / Design Objective

Realizations of measurements:

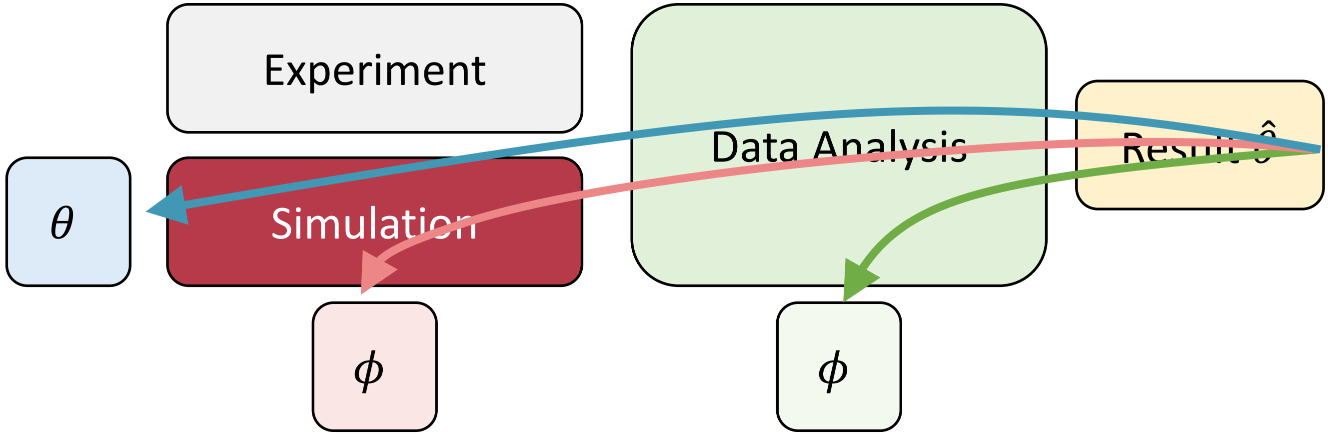
E.g. Simulations

Analysis params / Design params

Probability to see a measurement, e.g.

- Scattering prob.
- Detection prob.

What are we optimizing



$$\min_{\phi} \mathbb{E}[f(x,\phi)] = \min_{\phi} \int f(x,\phi) p_{\phi}(x|\theta) dx$$

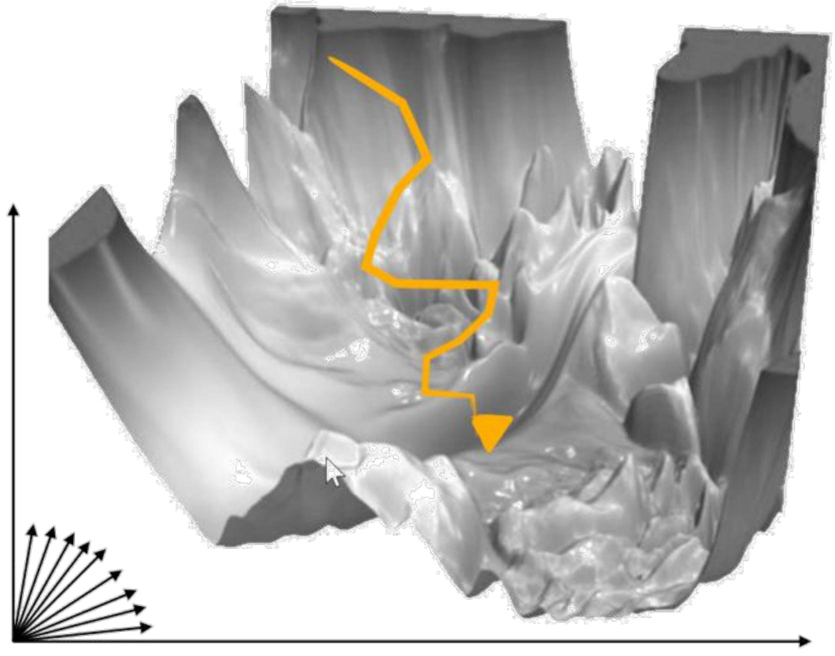
$$\approx \min_{\phi} \frac{1}{N} \sum_{x_i \sim p_{\phi}(x|\theta)} f(x_i,\phi)$$

How do we optimize? Gradient Descent

Deep learning looks very similar, optimizing an objective over parameters of a model using set of samples

Stochastic gradient descent (SGD): go-to optimization algorithm for training deep neural networks with even $O(10^{11})$ parameters

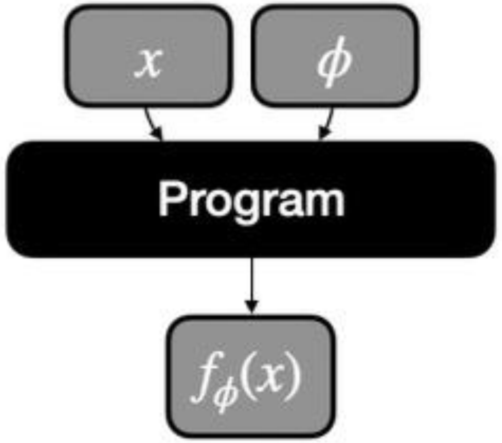
$$\theta \leftarrow \theta - \nabla_\theta L(x, \theta)$$



To deal with hyper-planes in a 14-dimensional space, visualize a 3D space and say 'fourteen' to yourself very loudly.
-G. Hinton

Differentiable Programming

Derivatives for gradient-based optimization come from running **differentiable code** via **automatic differentiation (AD)**



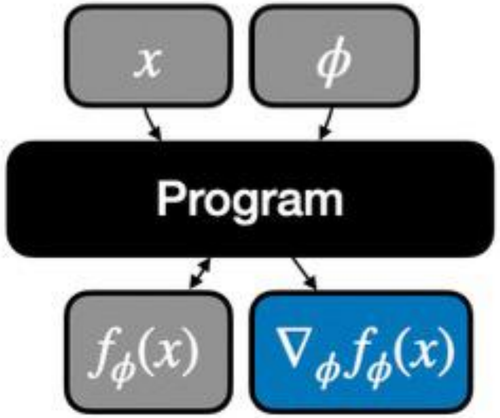


Image credit: L. Heinrich

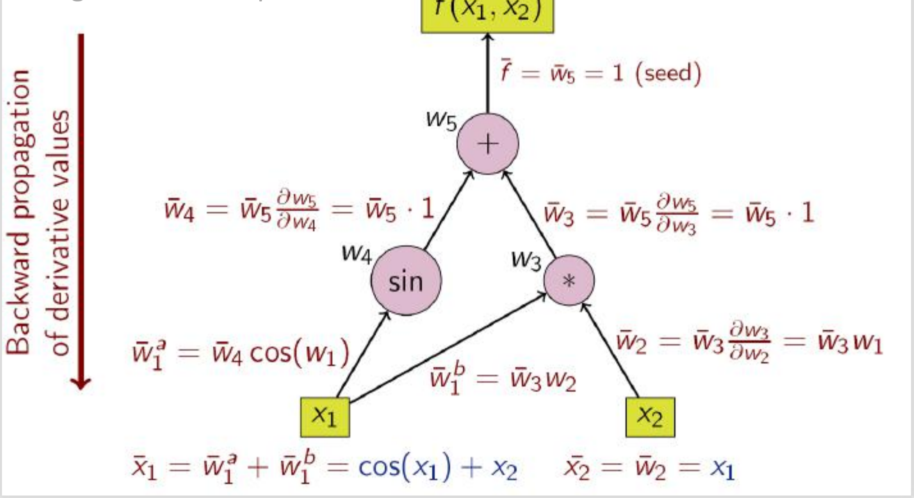


Image credit: Wikipedia

TensorFlow

PyTorch

Differentiable Programming in HEP

Mix **learnable ML modules** with domain-specific computations, e.g. **physics code**, and use the full pipeline as a jointly optimizable entity

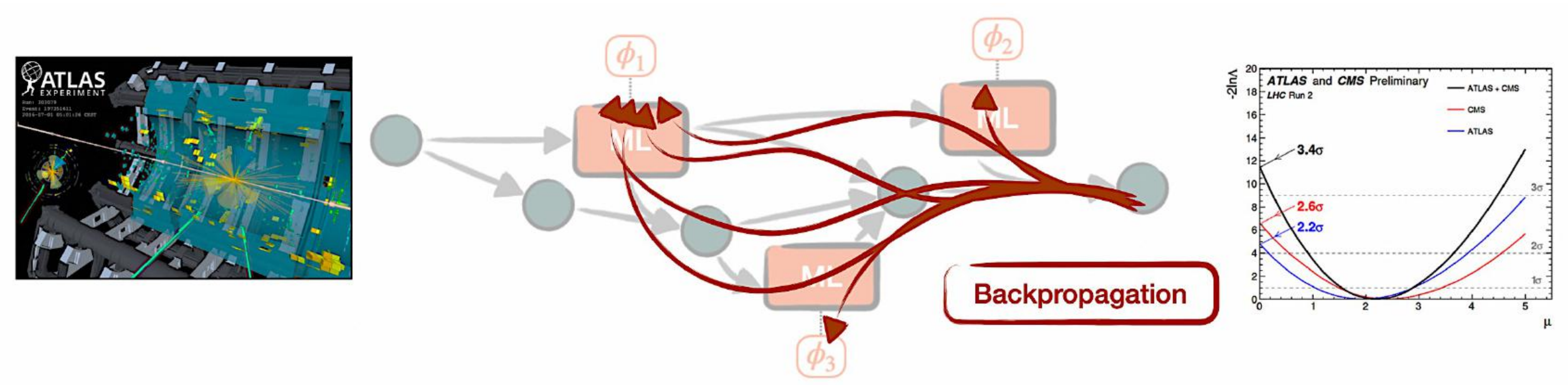
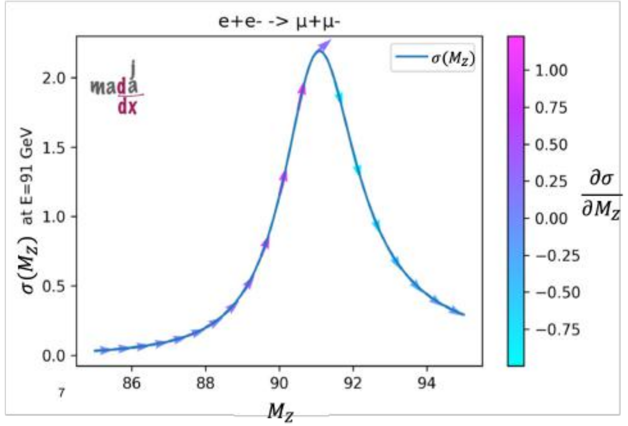


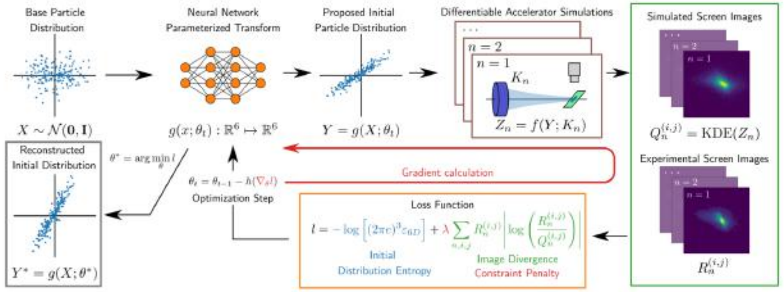
Image credit: L. Heinrich

A Growing Body of Work in HEP



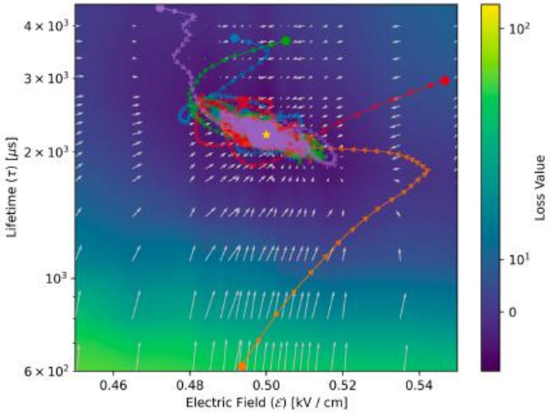
Differentiable Matrix Elements

[Heinrich, **MK**, 2203.00057]



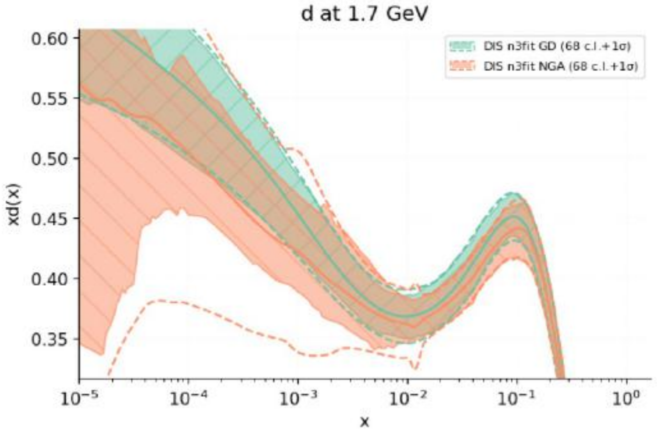
Differentiable Accelerator Simulation

[Roussel, Edelen, 2211.09077]

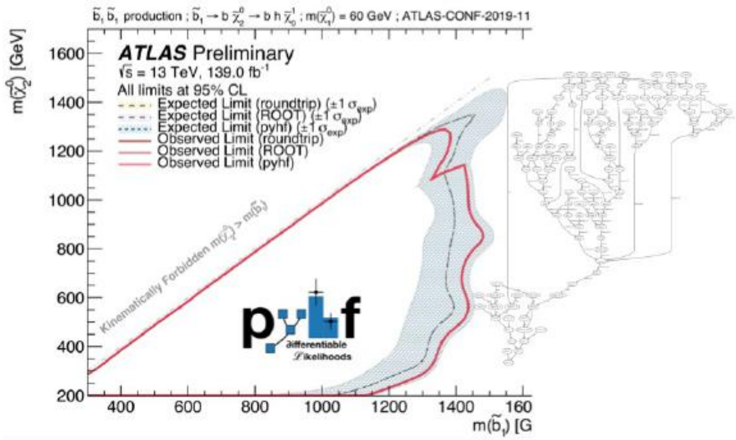


Differentiable LAr TPC simulation

[Gasiorowski, et al., 2309.04639]

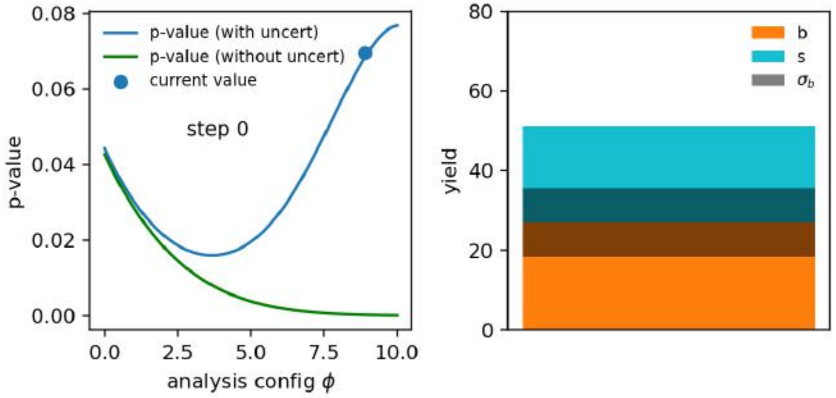


Differentiable Parton Distribution Functions [Ball, et al., 2109.02671]



Differentiable Inference

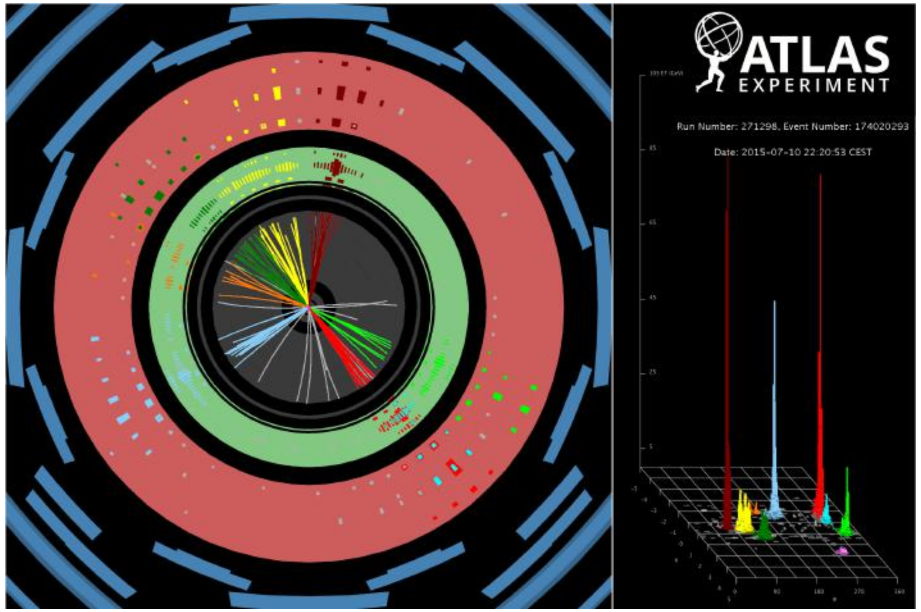
[Feickert, Heinrich, Stark, 2211.15838]

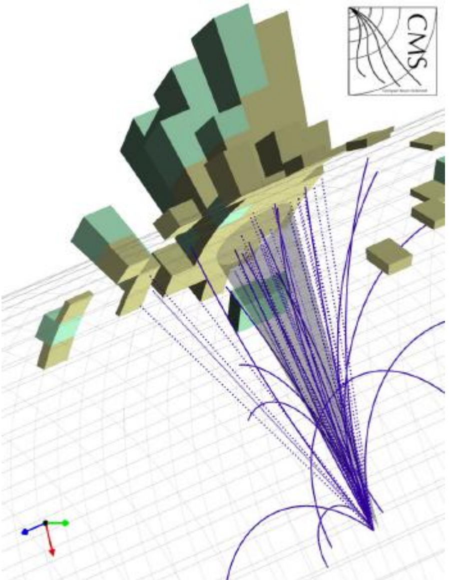


Differentiable Analysis + Inference

[Simpson, Heinrich, 2203.05570]

Example: Jet Classification





Jet = Unordered set of particles

Each particles has a list of features:

Particle = {momentum, direction, position, … }

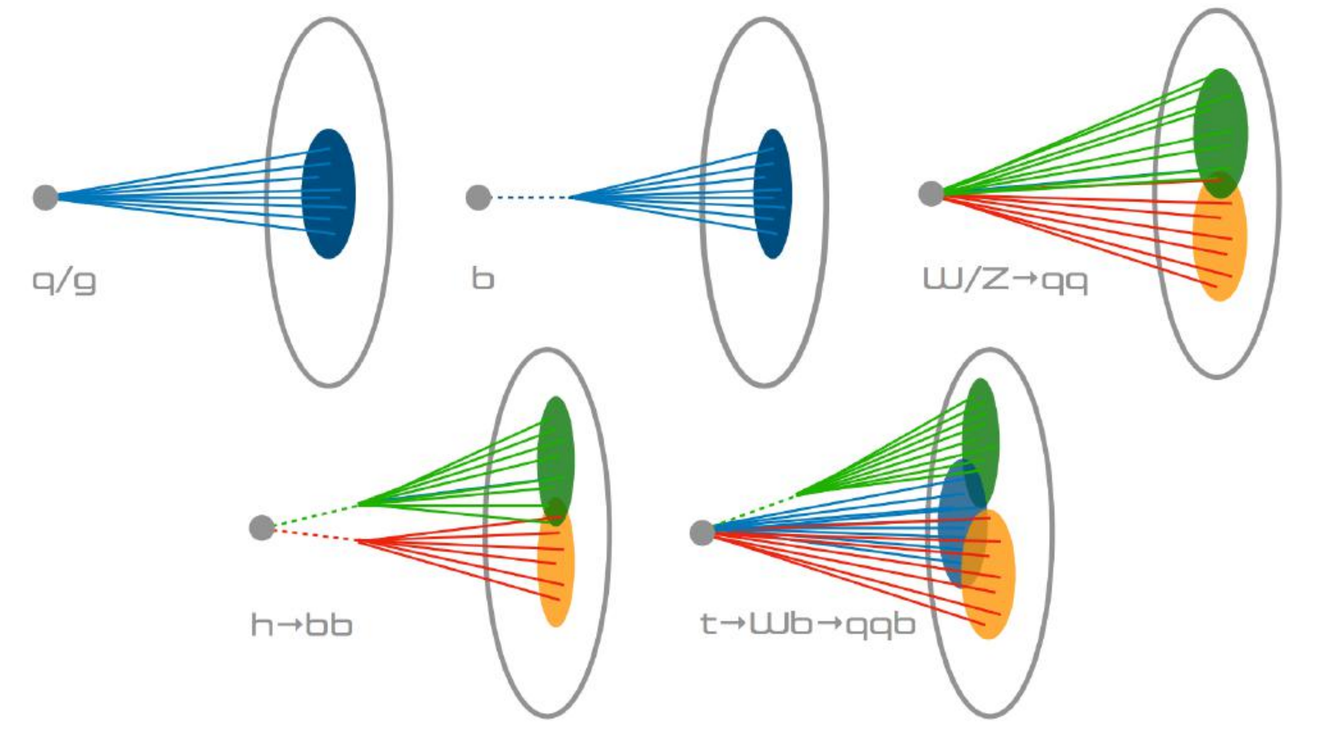
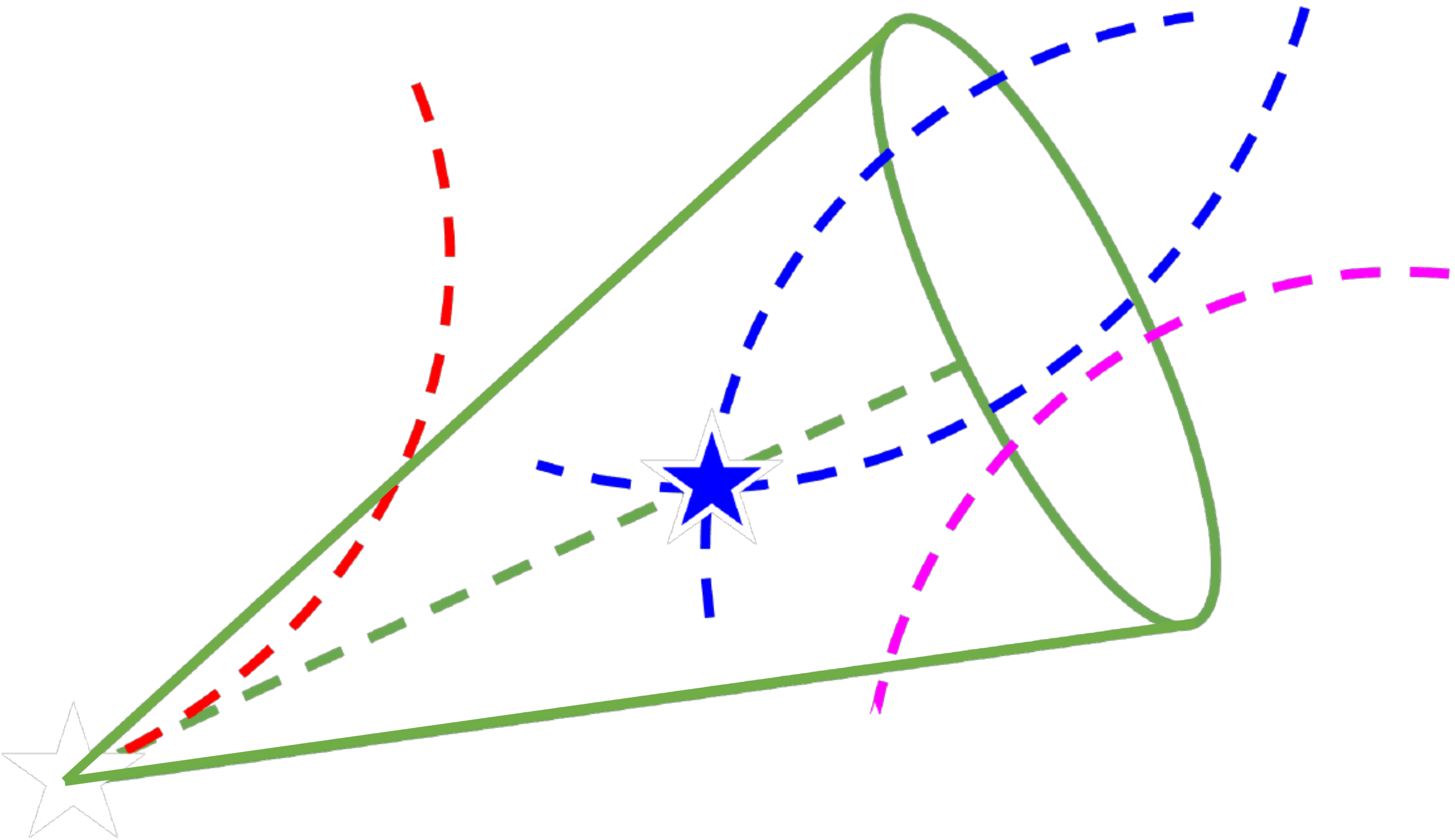


Image Credit: [1909.12285](1909.12285)

Differentiable Vertexing

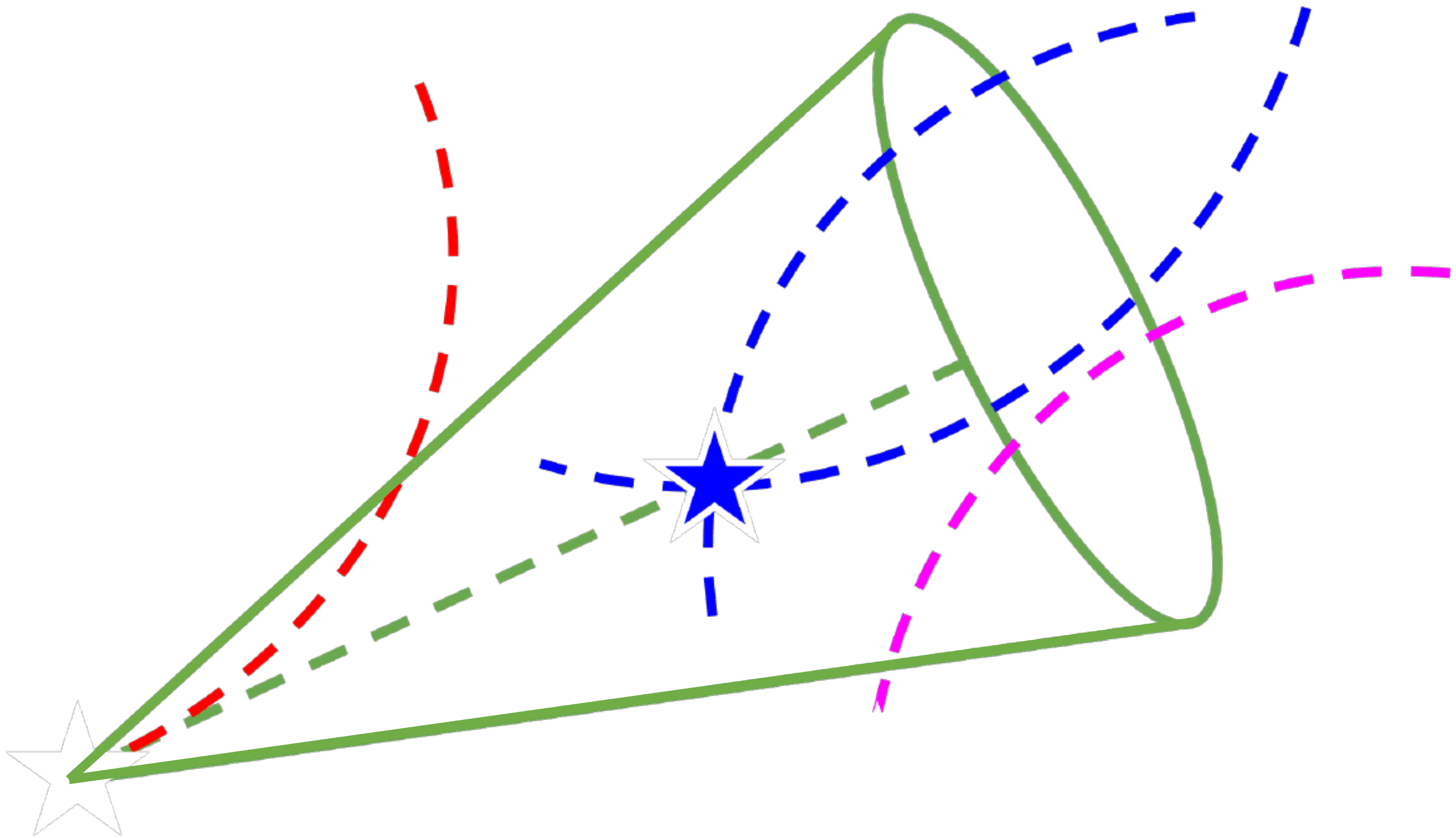


Vertex (★)

$$v^* = \arg\min_{v} \chi^2(v, \alpha)$$

Smith, Ochoa, Inacio, Shoemaker, **MK**, 2310.12804

Differentiable Vertexing



Vertex (★)

$$v^* = \arg\min_v \chi^2(v, \alpha)$$

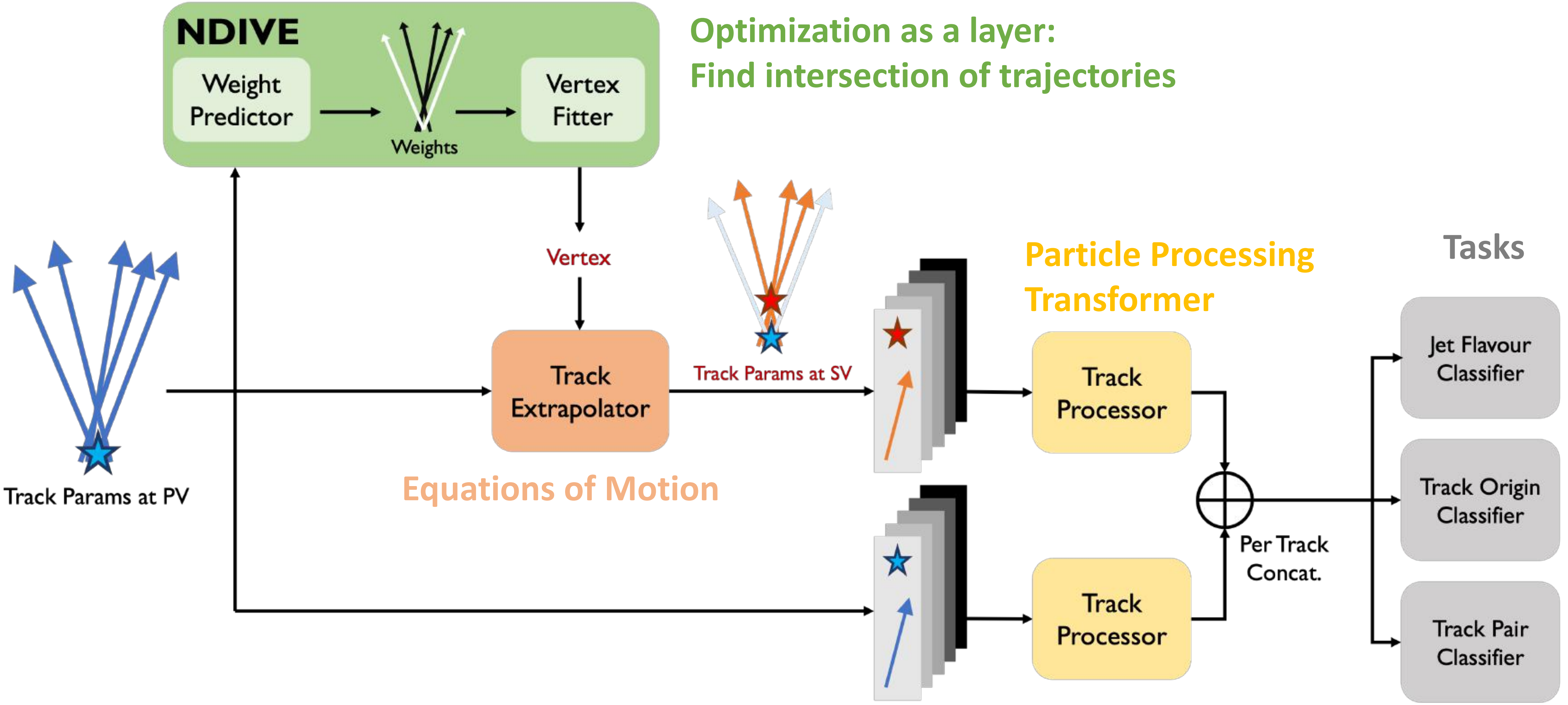
Gradients from implicit differentiation

$$\mathcal{G} \equiv \left.\frac{\partial \chi^2(v, \alpha)}{\partial v}\right|_{v^*} = 0$$

$$\frac{\partial v^*}{\partial \alpha} = -\left(\frac{\partial \mathcal{G}}{\partial v}\right)^{-1} \left(\frac{\partial \mathcal{G}}{\partial \alpha}\right)$$

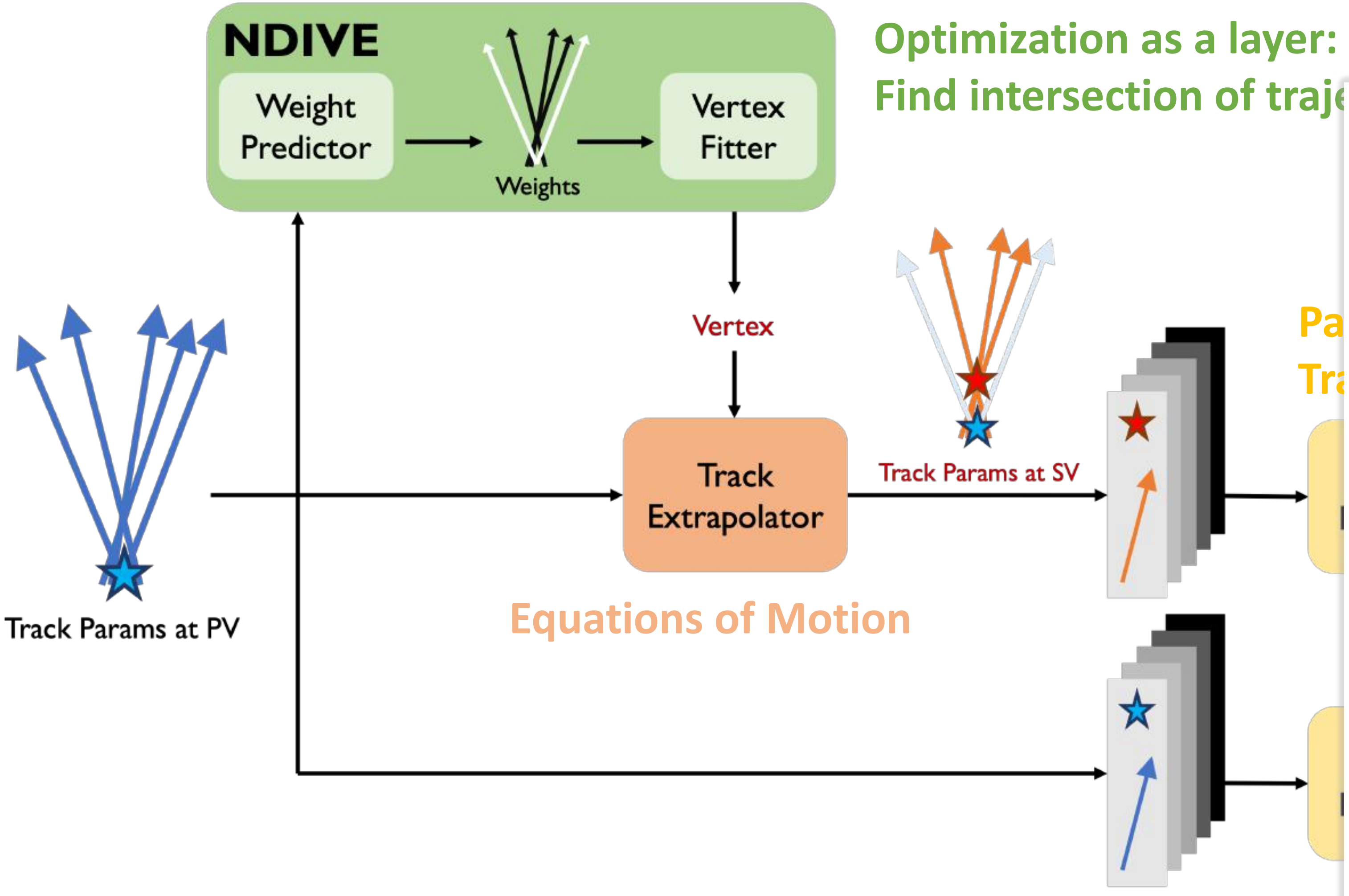
Smith, Ochoa, Inacio, Shoemaker, **MK**, 2310.12804

Adding Domain Knowledge with Differentiable Programming



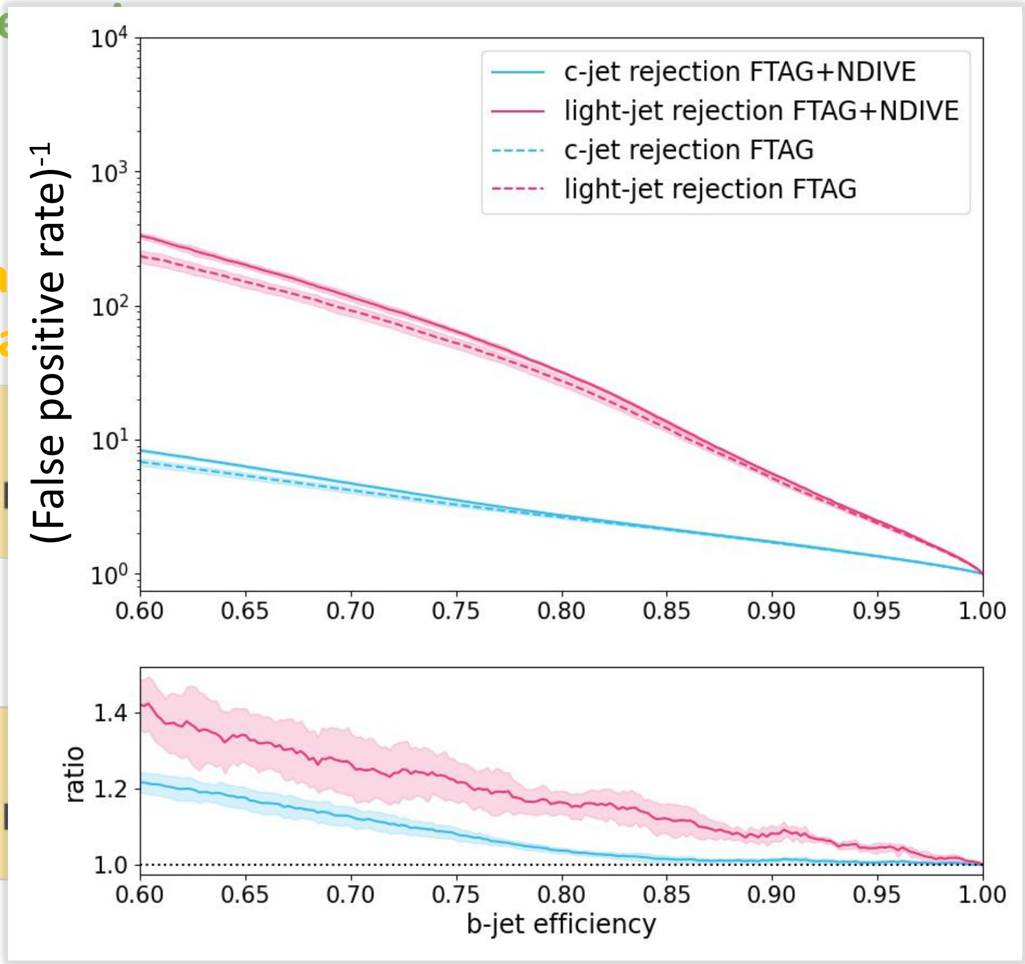
Smith, Ochoa, Inacio, Shoemaker, **MK**, [2310.12804](2310.12804)

Adding Domain Knowledge with Differentiable Programming



Optimization as a layer:
Find intersection of traje

Adding Physics Helps!



Smith, Ochoa, Inacio, Shoemaker, **MK**, 2310.12804

What happens if there is stochasticity inside the model?

Automatic Differentiation & ML

AD is great for differentiating deterministic functions like

$$y = f(x)$$

ML requires differentiating expectation values

$$\frac{\partial}{\partial \phi} \mathbb{E}_{p(x)}[f(x,\phi)] = \frac{\partial}{\partial \phi} \int f(x,\phi) p(x) dx$$

No param dependence in the distribution $p(\cdot)$

Automatic Differentiation & ML

AD is great for differentiating deterministic functions like

$$y = f(x)$$

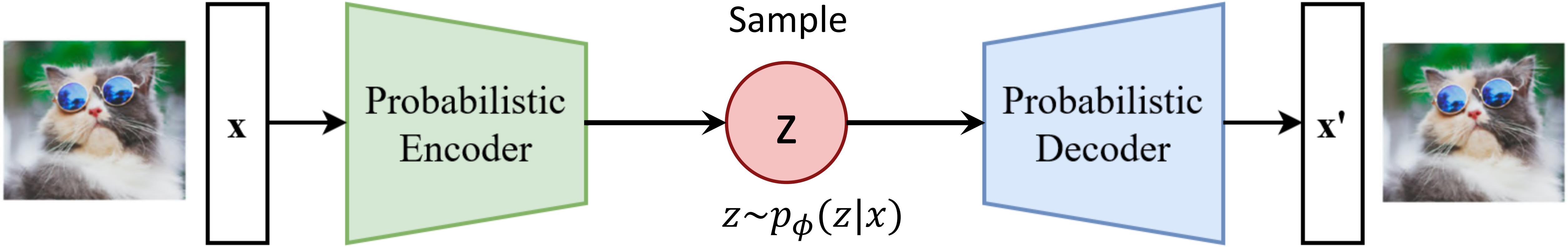
ML requires differentiating expectation values

$$\frac{\partial}{\partial \phi}\mathbb{E}_{p(x)}[f(x,\phi)] = \frac{\partial}{\partial \phi}\int f(x,\phi)p(x)dx$$

"Easy" Case

$$= \int \frac{\partial f(x,\phi)}{\partial \phi} p(x)dx = \mathbb{E}_{p(x)}\left[\frac{\partial f(x,\phi)}{\partial \phi}\right]$$

Harder Case: Variational Autoencoder

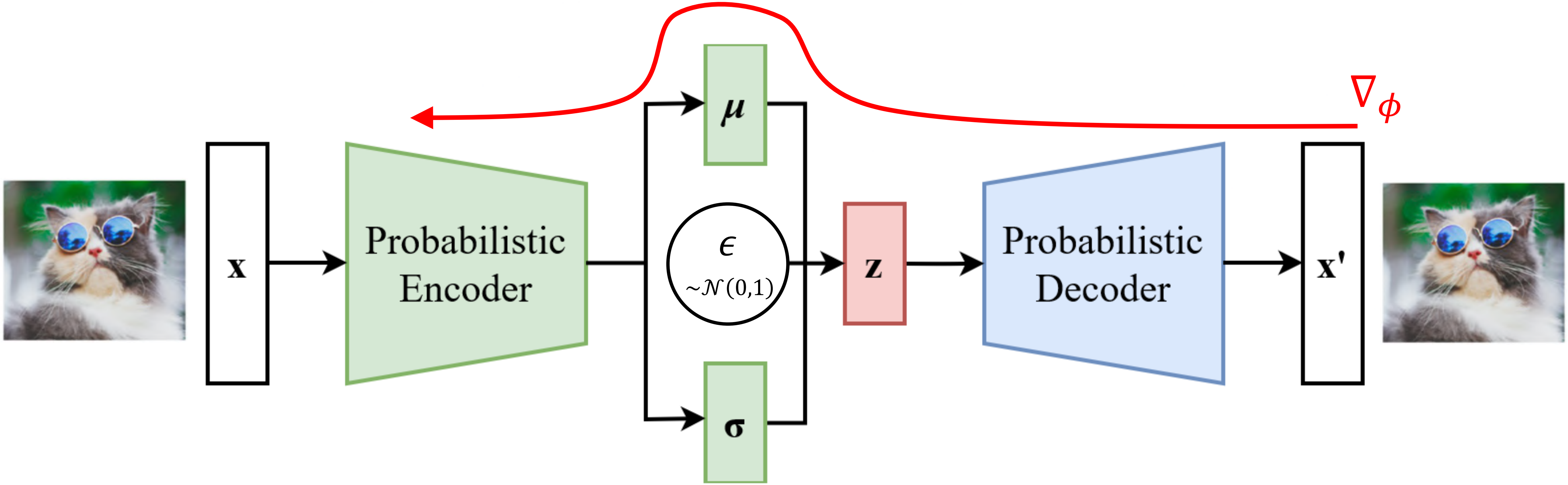


param dependence in $q(\cdot)$

$$L = \mathbb{E}_{q_\phi(z|x)}[\log p(x|z)] - \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z|x)}{p(z)}\right]$$

Kingma, Welling, 1312.6114
Rezende, Mohamed, Wierstra, 1401.4082

Harder Case: Variational Autoencoder



$$L = \mathbb{E}_{p(\epsilon)}[\log p(x|z(\epsilon,\phi))] - \mathbb{E}_{p(\epsilon)}\left[\log \frac{q_\phi(z(\epsilon,\phi)|x)}{p(z(\epsilon,\phi))}\right]$$

Kingma, Welling, [1312.6114]

Rezende, Mohamed, Wierstra, [1401.4082]

Reparameterization Trick

Separate parameters from stochasticity

$\boldsymbol{x} \sim \boldsymbol{p_\theta}(\boldsymbol{x}) \quad \rightarrow \quad$ **rewrite** $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{\epsilon}, \boldsymbol{\theta})$ **with** $\boldsymbol{\epsilon} \sim \boldsymbol{p}(\boldsymbol{\epsilon})$

Example:

$x \sim \mathcal{N}(\mu, \sigma) \quad \rightarrow \quad x = g(\epsilon, \mu, \sigma) = \epsilon * \sigma + \mu$ with $\epsilon \sim \mathcal{N}(0,1)$

$$\frac{d}{d\theta}\mathbb{E}_{p_\theta(x)}[f(x)] = \frac{d}{d\theta}\mathbb{E}_{p(\epsilon)}\left[f\left(g(\epsilon, \theta)\right)\right] = \mathbb{E}_{p(\epsilon)}\left[\frac{df}{dg}\frac{dg}{d\theta}\right]$$

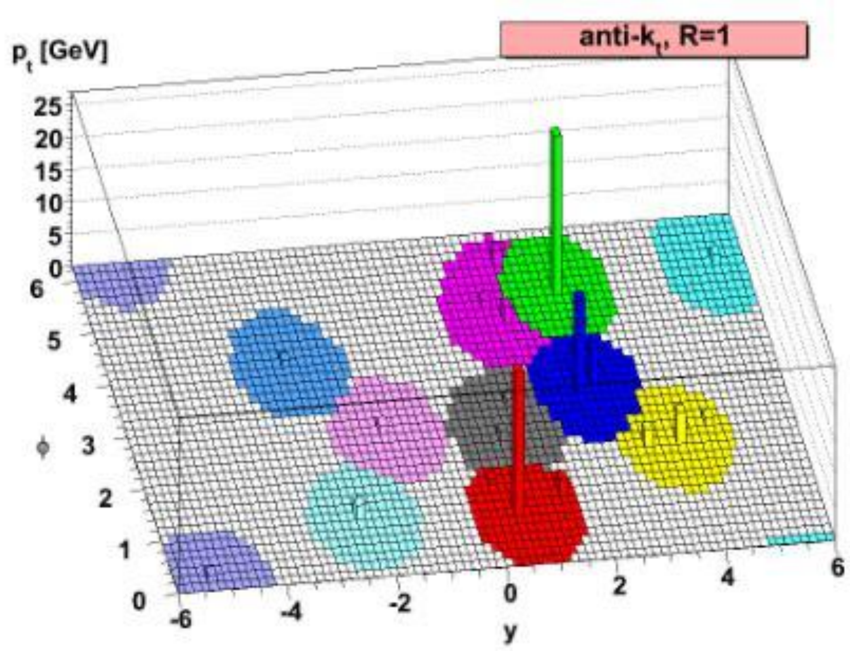
Is that all we need?

A Problem: Discrete Random Variables & Choices

Discrete random variables and discrete choices are all over HEP

Branching / Showering Processes

Clustering Algorithms

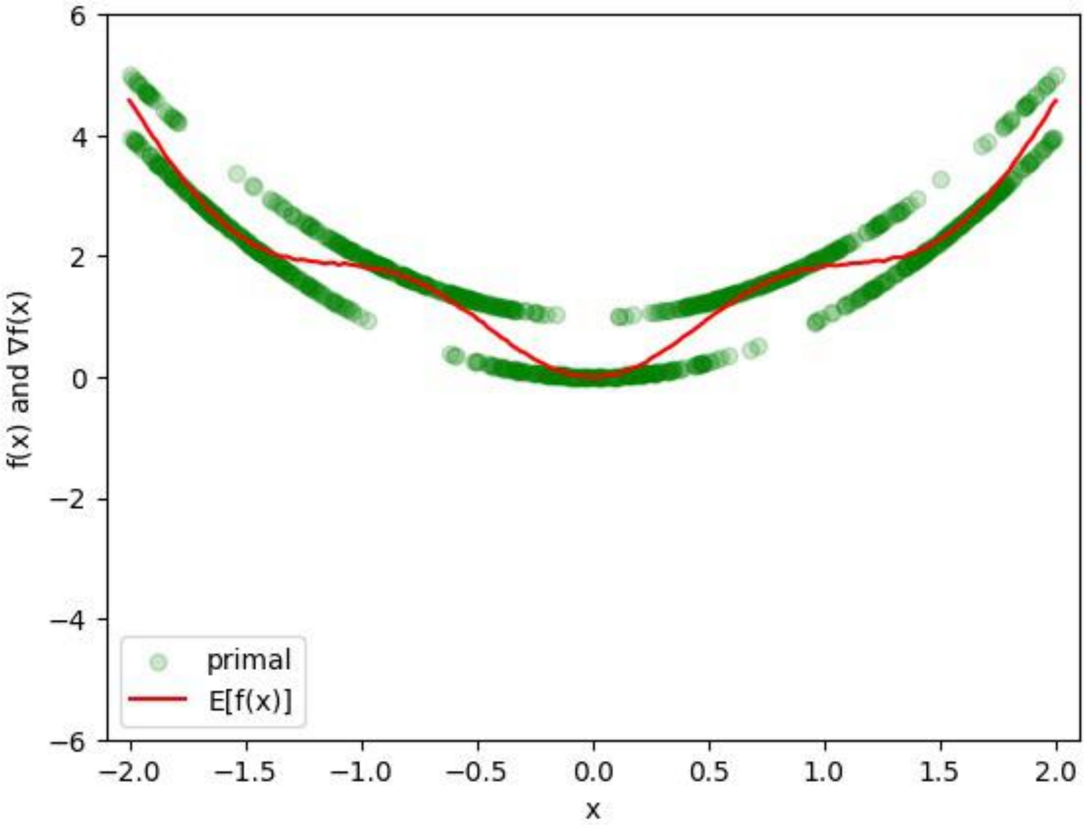


Programs with Discrete Randomness

```
def f(x):
    theta = (sin(2.*x))**2
    b = bernoulli(theta)
    g = x*x
    return g+b
```

Bernoulli parameter θ depends on x

$f(x) = x^2 + b \qquad b \sim Bern(\theta = \sin^2(2x))$



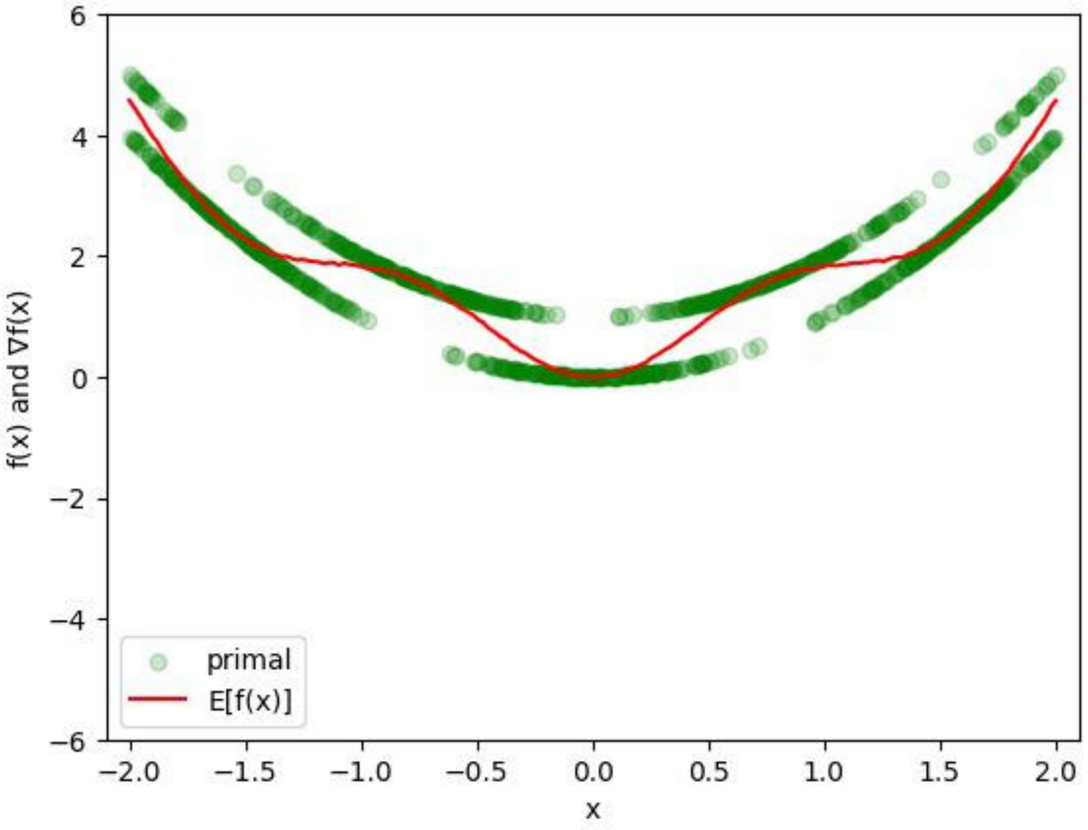
Programs with Discrete Randomness


```
def f(x):
    theta = (sin(2.*x))**2
    b = bernoulli(theta)
    g = x*x
    return g+b
```

$$\mathbb{E}_b[f(x)] = x^2 + \sin^2 2x$$

$$\nabla_x \mathbb{E}_b[f(x)] = 2x + 4\sin(2x)\cos(2x)$$

$f(x) = x^2 + b \qquad b \sim Bern(\theta = \sin^2(2x))$



Even if a program contains discrete randomness,
expected value can be smooth and have a well-defined derivative

Programs with Discrete Randomness

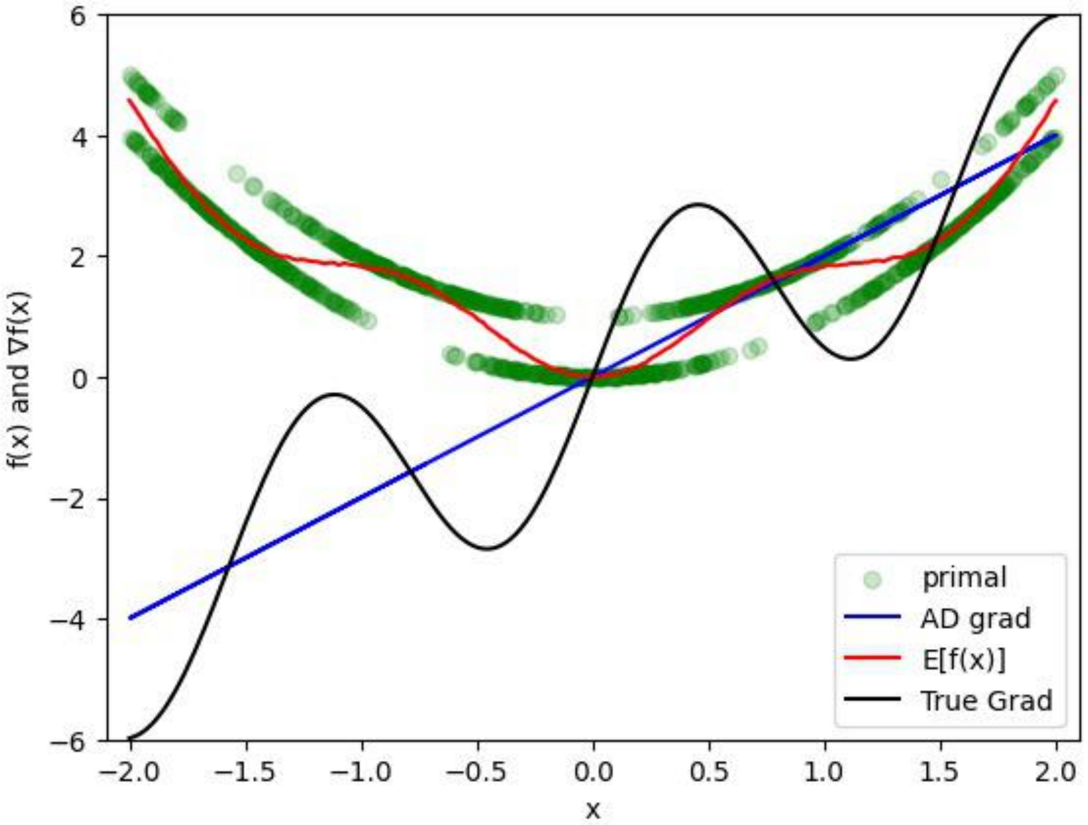

```
def f(x):
    theta = (sin(2.*x))**2
    b = bernoulli(theta)
    g = x*x
    return g+b
```

$$\mathbb{E}_b[f(x)] = x^2 + \sin^2 2x$$

$$\nabla_x \mathbb{E}_b[f(x)] = 2x + 4\sin(2x)\cos(2x)$$

AD Gradient: $grad\big(f_i(x)\big) \to 2x_i$

$f(x) = x^2 + b \qquad b \sim Bern(\theta = \sin^2(2x))$



Standard AD tools don't know how to handle discrete randomness that depends on the parameter of differentiation → ***We need another approach***

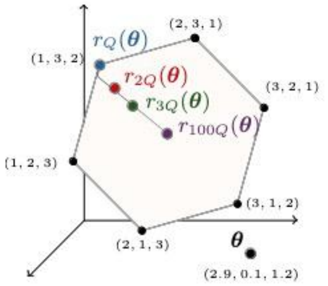
Derivatives for Discrete Randomness

Do Some Work, Get Better Derivatives
- Score Functions
- Stochastic AD

Related work: Smooth perturbation analysis

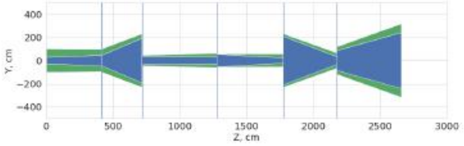
Approximate Derivatives
- Smoothing / Relaxations
- Surrogates

Example: Differentiable ranking & sorting



2002.08871

HEP Example: Surrogates for SHiP magnet design



MK, et al. 2002.04632

Numerical Derivatives
- Finite Differences

$$\frac{f(x+\epsilon)-f(x)}{\epsilon}$$

Don't Use Derivatives
- Gradient-Free Methods

Bayesian Opt., Genetic Algs, …

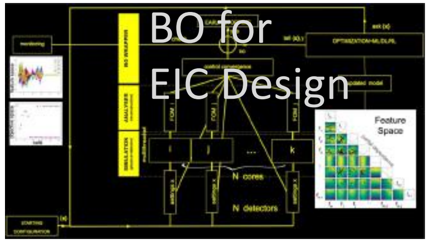


Figure credit: C. Fanelli

Score Functions / REINFORCE

$$\nabla_\theta \mathbb{E}_{p_\theta(x)}[f(x)] = \int \nabla_\theta p_\theta(x) f(x) dx$$

Score Functions / REINFORCE

$$\nabla_\theta \mathbb{E}_{p_\theta(x)}[f(x)] = \int p_\theta(x) f(x) \nabla_\theta \log p_\theta(x)\, dx$$

because:

$$\nabla_\theta \log p_\theta(x) = \frac{1}{p_\theta(x)} \nabla_\theta p_\theta(x)$$

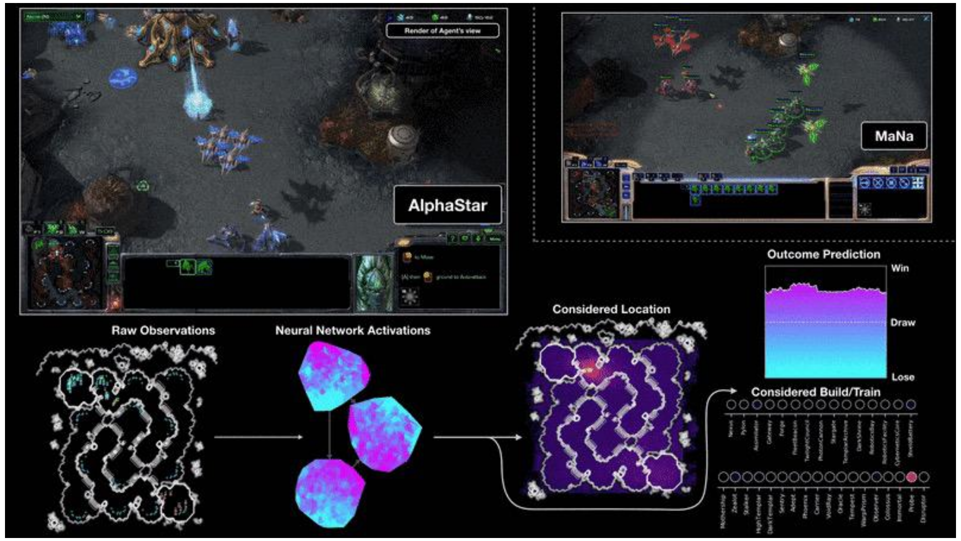
Score Functions / REINFORCE

$$\nabla_\theta \mathbb{E}_{p_\theta(x)}[f(x)] = \mathbb{E}_{p_\theta(x)}[f(x)\nabla_\theta \log p_\theta(x)]$$

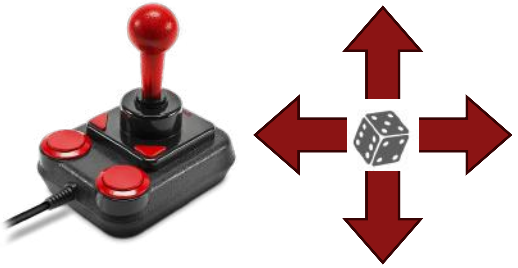
Gradient estimator used in Reinforcement Learning

Works with discrete x and even non-differentiable $f(\cdot)$

Requires tracking probabilities $\log p_\theta(x)$ throughout program



AlphaStar Vinyals et al. 2019



Stochastic AD

$$\frac{d}{d\theta}\mathbb{E}_{p_\theta}[f(x)] = \mathbb{E}_{p_\theta}[\delta + \beta(y - x)]$$

Standard AD — Weight — Alternative value of rv

Recently, *Arya et al.* extended fwd-mode AD to discrete-stochastic environments

Importantly, this includes a ***composition rule*** for how to combine weights β step-by-step along the computation chain

2210.08572

Automatic Differentiation of Programs with Discrete Randomness

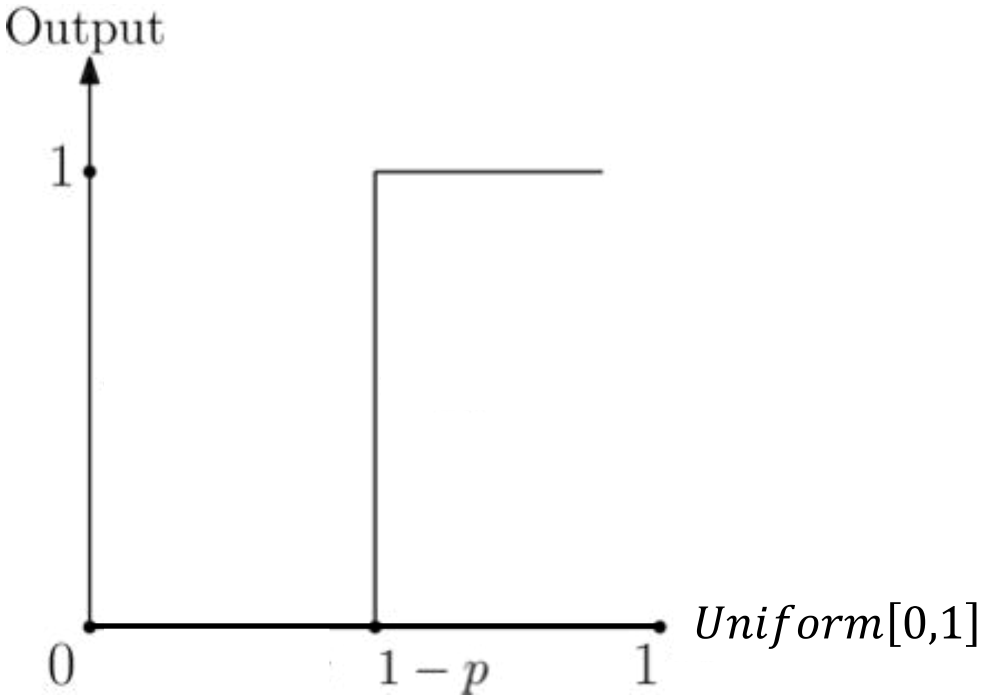
Gaurav Arya
Massachusetts Institute of Technology, USA
aryag@mit.edu

Moritz Schauer
Chalmers University of Technology, Sweden
University of Gothenburg, Sweden
smoritz@chalmers.se

Intuition

Can use the inversion method to reparameterize discrete random variables

$$x \sim Bernoulli(p) \quad \rightarrow \quad \begin{aligned} &\omega \sim Uniform[0,1] \\ &x = \begin{cases} 1 & if\ \omega > 1-p \\ 0 & Otherwise. \end{cases} \end{aligned}$$



Arya et al., 2210.08572

Intuition

Can use the inversion method to reparameterize discrete random variables

$$x \sim Bernoulli(p) \quad \rightarrow \quad \begin{aligned} &\omega \sim Uniform[0,1] \\ &x = \begin{cases} 1 & if\ \omega > 1-p \\ 0 & Otherwise. \end{cases} \end{aligned}$$

$$\mathbb{E}[b] = \int 1_{[\omega > 1-p]} p(\omega) d\omega = \int_{1-p}^{1} d\omega$$

Intuition

Can use the inversion method to reparameterize discrete random variables

$$x \sim Bernoulli(p) \quad \rightarrow \quad \begin{gathered} \omega \sim Uniform[0,1] \\ x = \begin{cases} 1 & if\ \omega > 1 - p \\ 0 & Otherwise. \end{cases} \end{gathered}$$

$$\mathbb{E}[b] = \int 1_{[\omega > 1-p]} p(\omega) d\omega = \int_{1-p}^{1} d\omega$$

Standard AD on Monte Carlo expectation of this program would still be wrong

$$grad_p \left(\frac{1}{N} \sum_i [1\ if\ (\omega_i > 1 - p)\ else\ 0] \right) = 0$$

Intuition

Can use the inversion method to reparameterize discrete random variables

$$x \sim Bernoulli(p) \quad \rightarrow \quad \omega \sim Uniform[0,1]$$

$$x = \begin{cases} 1 & if\ \omega > 1-p \\ 0 & Otherwise. \end{cases}$$

$$\mathbb{E}[b] = \int 1_{[\omega > 1-p]} p(\omega) d\omega = \int_{1-p}^{1} d\omega$$

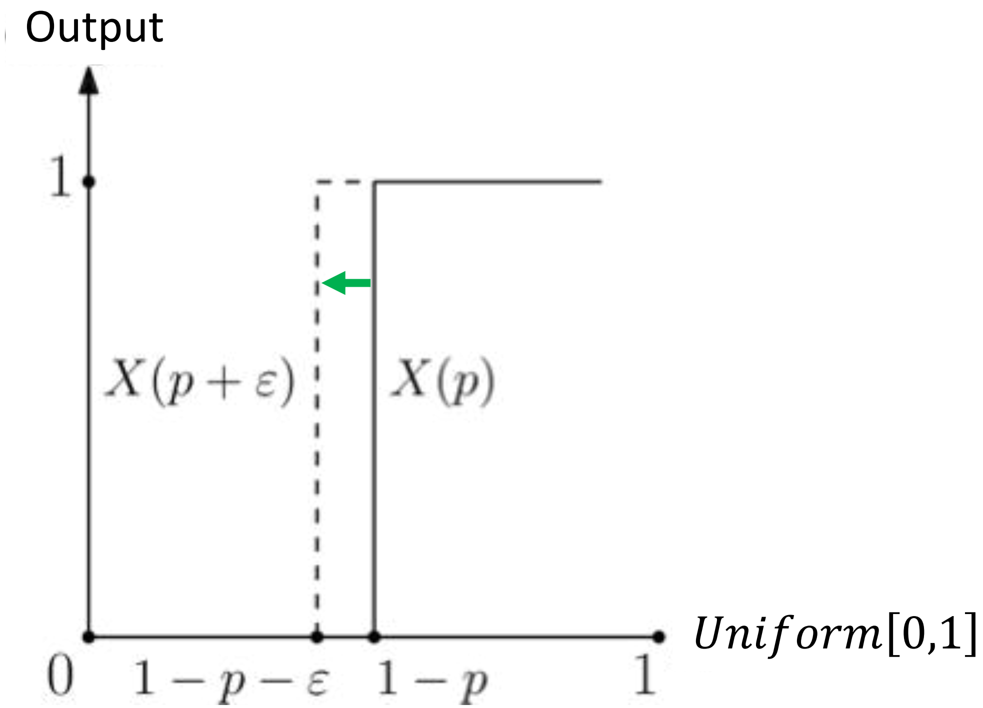
Param. dependence in integration bounds

Correct derivative must account for boundary dependence → *Leibniz Rule*

Intuition

Can use the inversion method to reparameterize discrete random variables

$$x \sim Bernoulli(p) \quad \rightarrow \quad \begin{array}{l} \omega \sim Uniform[0,1] \\ x = \begin{cases} 1 & if\ \omega > 1-p \\ 0 & Otherwise. \end{cases} \end{array}$$

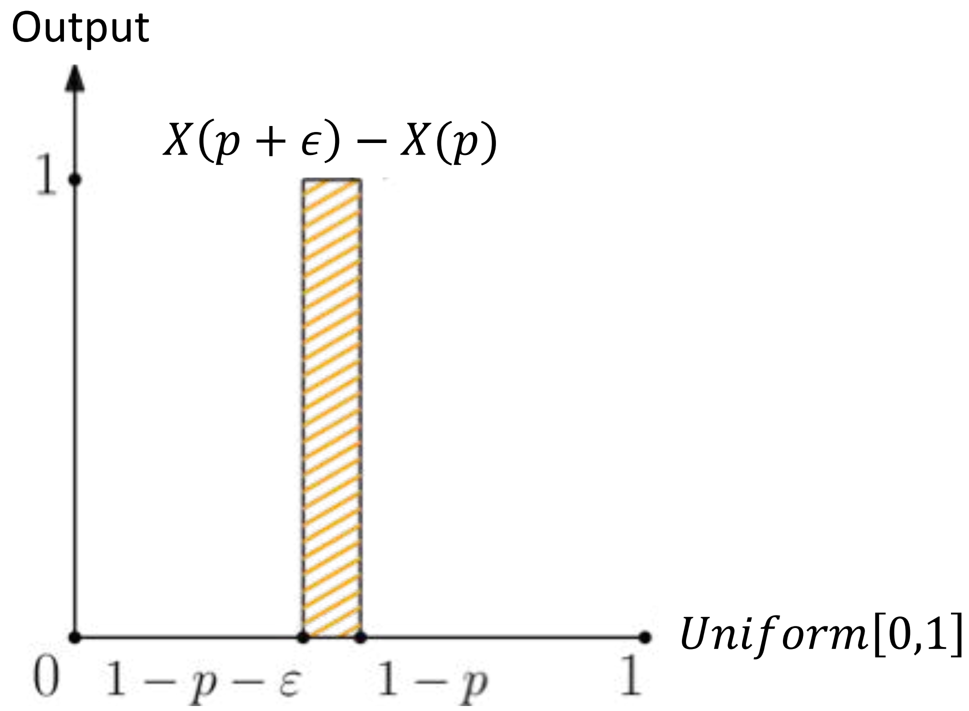


Arya et al., [2210.08572](2210.08572)

Intuition

Can use the inversion method to reparameterize discrete random variables

$$x \sim Bernoulli(p) \quad \rightarrow \quad \begin{array}{l} \omega \sim Uniform[0,1] \\ x = \begin{cases} 1 & if\ \omega > 1-p \\ 0 & Otherwise. \end{cases} \end{array}$$



Arya et al., 2210.08572

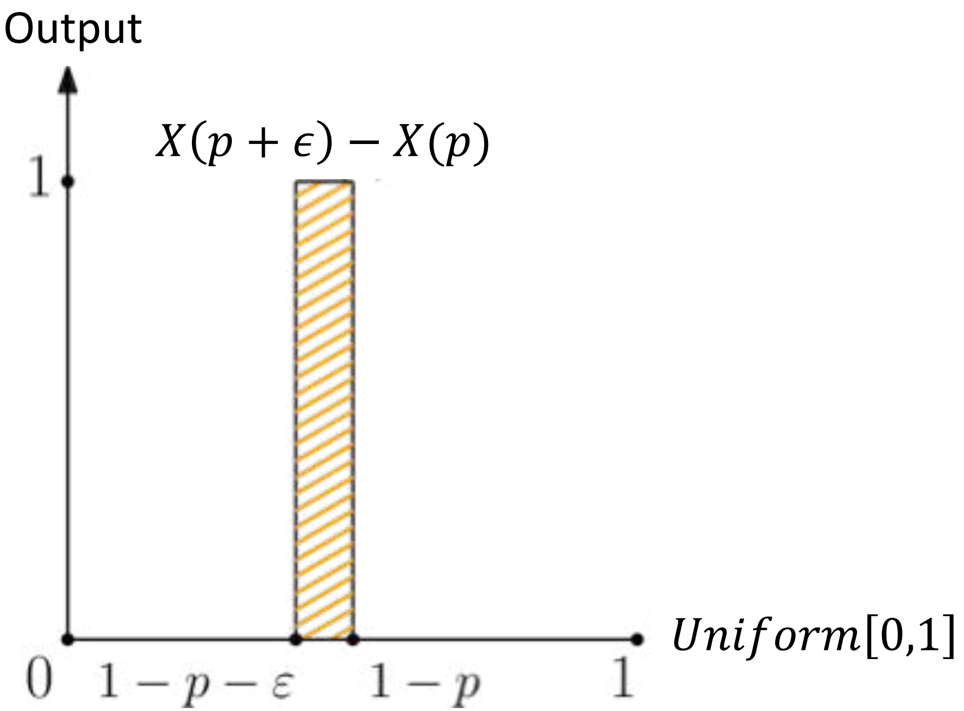
Stochastic AD

$$\frac{d}{d\theta}\mathbb{E}_{p_\theta}[f(x)] = \mathbb{E}_{p_\theta}[\delta + \beta(y - x)]$$

The weight β accounts for the derivative of the probability of a jump in program

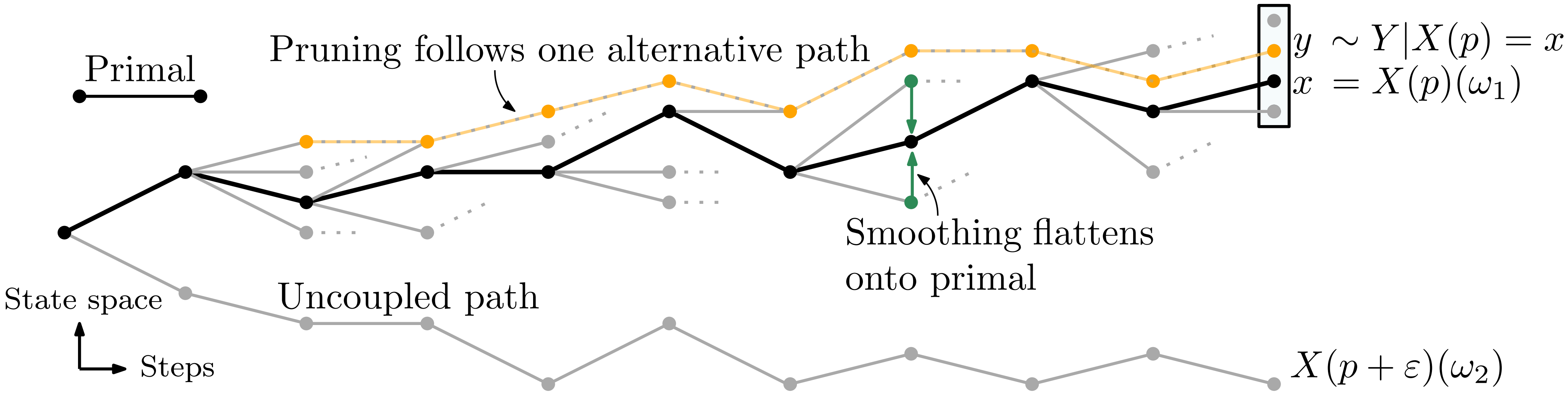
Equivalently, the weight accounts for the boundary derivative

In many cases: $\beta = \frac{\partial_\theta CDF_\theta(X(\theta))}{PDF_\theta(X(\theta))}$



Arya et al., 2210.08572

Automatic Differentiation



Correlated paths → low variance

$\mathcal{O}(1)$ unbiased forward mode AD

Arya et al., 2210.08572

Are these methods useful?

Toy Shower

Simplified particle shower:

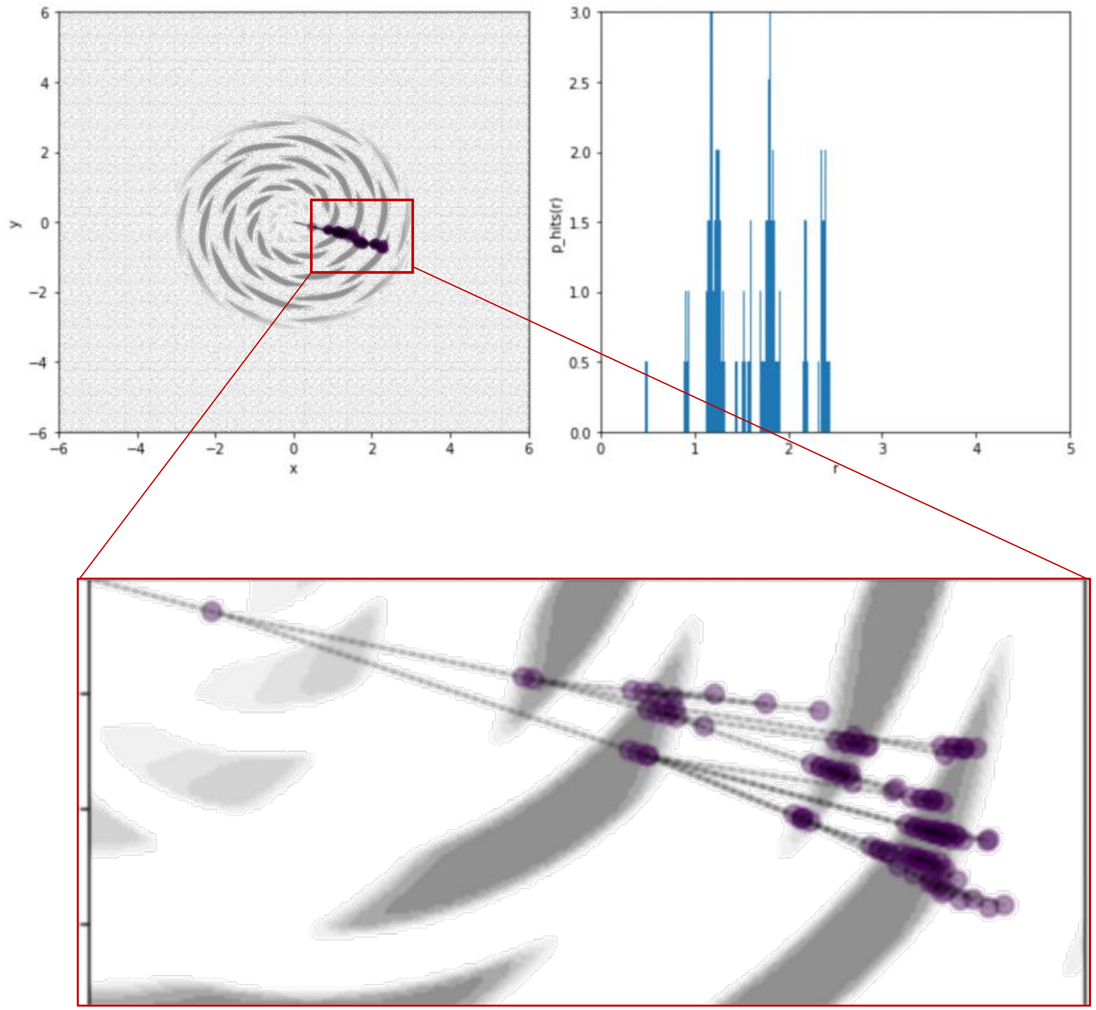
Including Energy loss and splitting

Design parameter:

Radial distance of material

Design goal:

Specify average shower depth



MK, Heinrich, 2308.16680

Toy Shower

Simplified particle shower:

Including Energy loss and splitting

Design parameter:

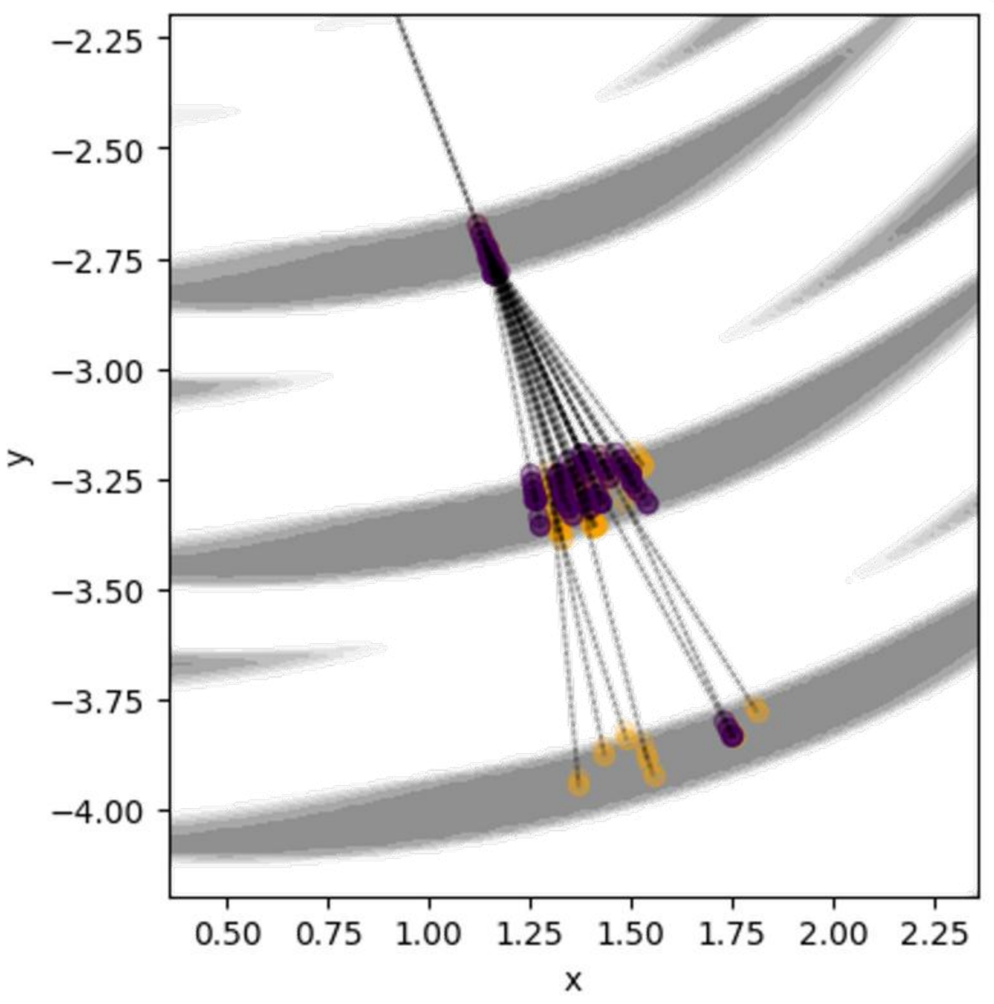
Radial distance of material

Design goal:

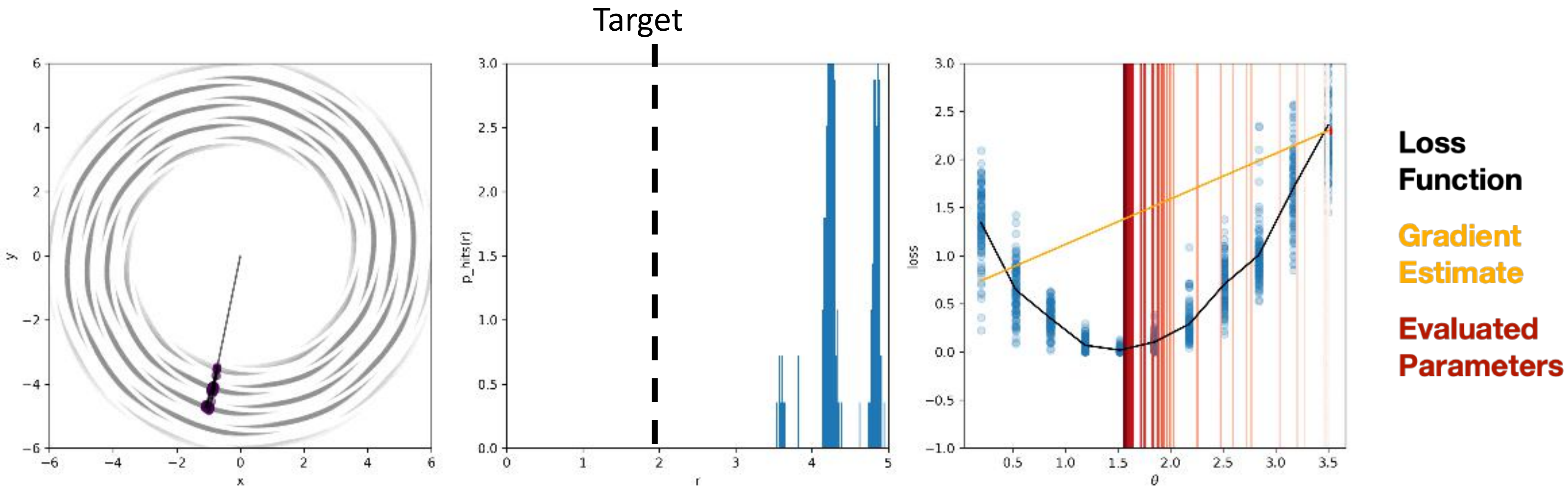
Specify average shower depth

Dedicated implementation of Stochastic AD

→ Can generate "alternative showers"



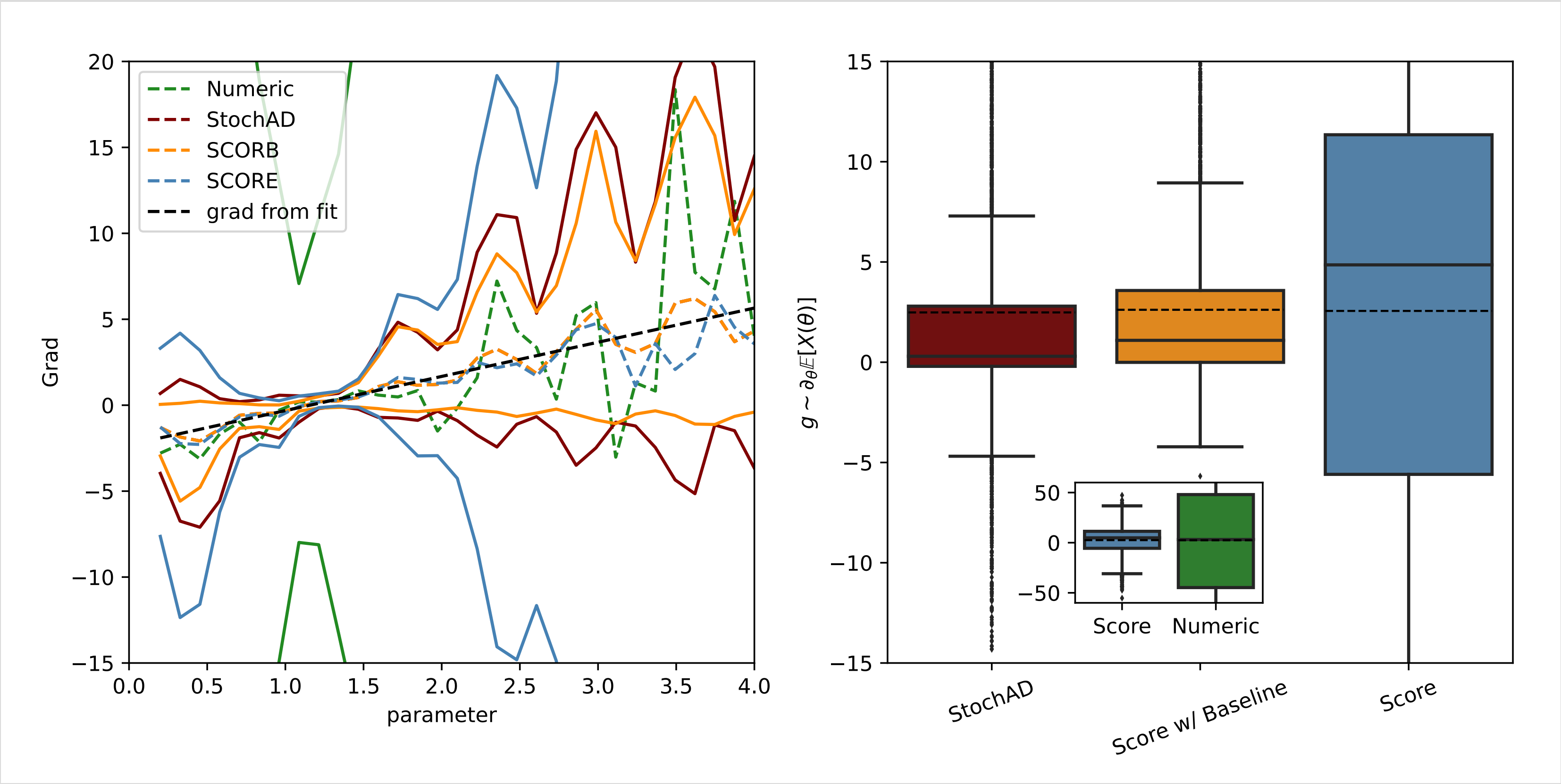
Example Optimization

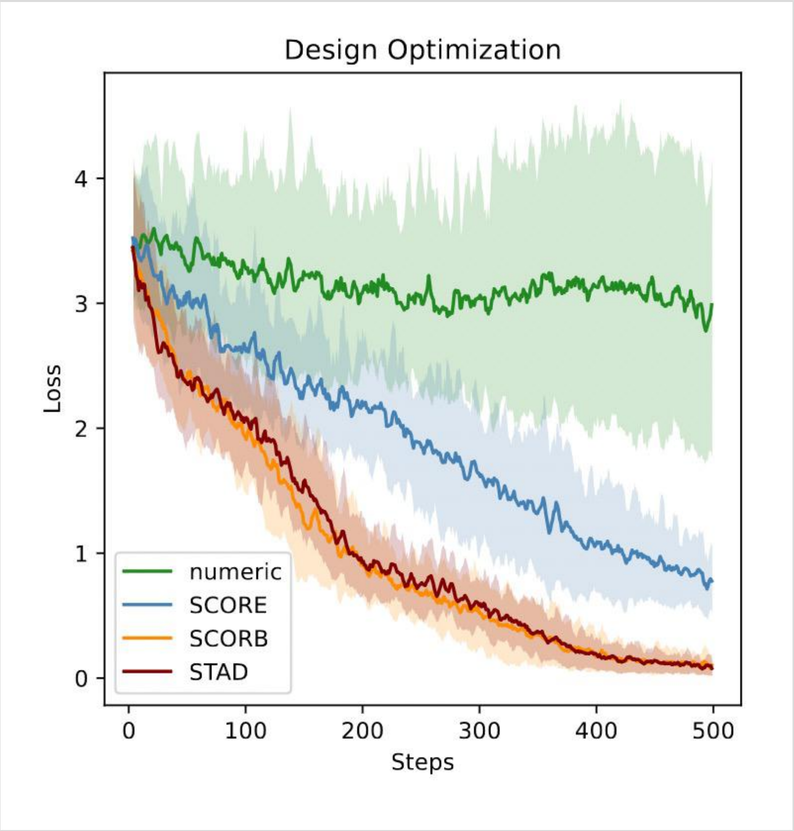


Gradients are noisy but in right direction (on average) → optimization works!

Comparisons

Both score function and Stochastic AD have reasonable variance of gradients





MK, Heinrich, 2308.16680

Summary

LHC and future colliders present unique opportunities

Need to make the most of it!

Differentiable programming provides powerful method to

Optimize our data analysis and simulation pipelines

Embed physics knowledge in our ML tools

Not everything is easily differentiable,

lots of challenges along the way…

especially for programs with discrete randomness