

Constraining effective field theories with machine learning

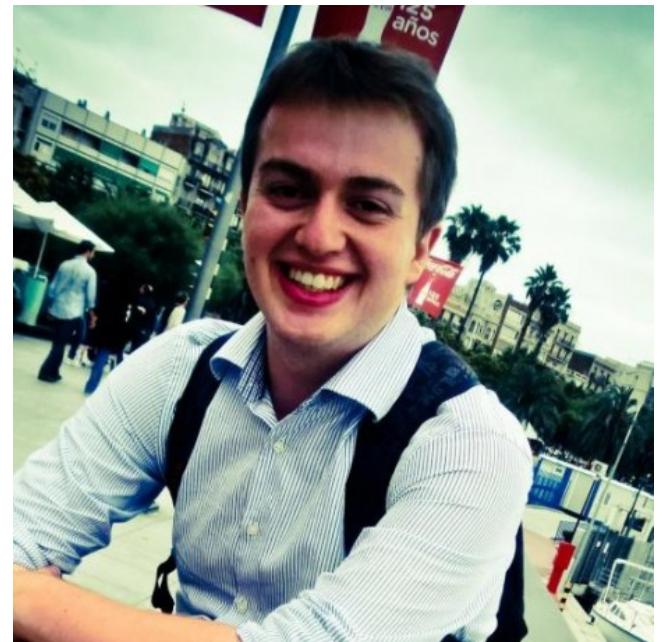
Johann Brehmer

New York University

ACAT 2019, Saas-Fee



Kyle Cranmer



Gilles Louppe



Juan Pavez



Felix Kling



Irina Espejo



Zubair Bhatti



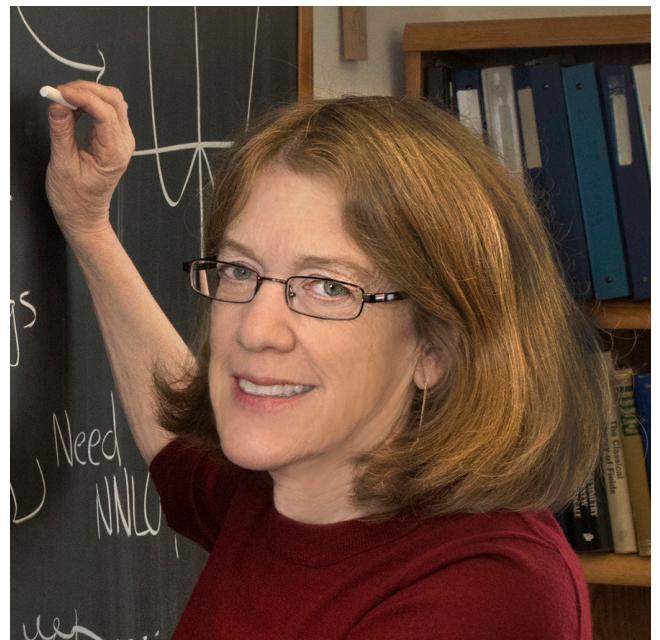
Markus Stoye



Sid Mishra-Sharma



Tilman Plehn



Sally Dawson



Sam Homiller



Josh Ruderman



Duccio Pappadopulo



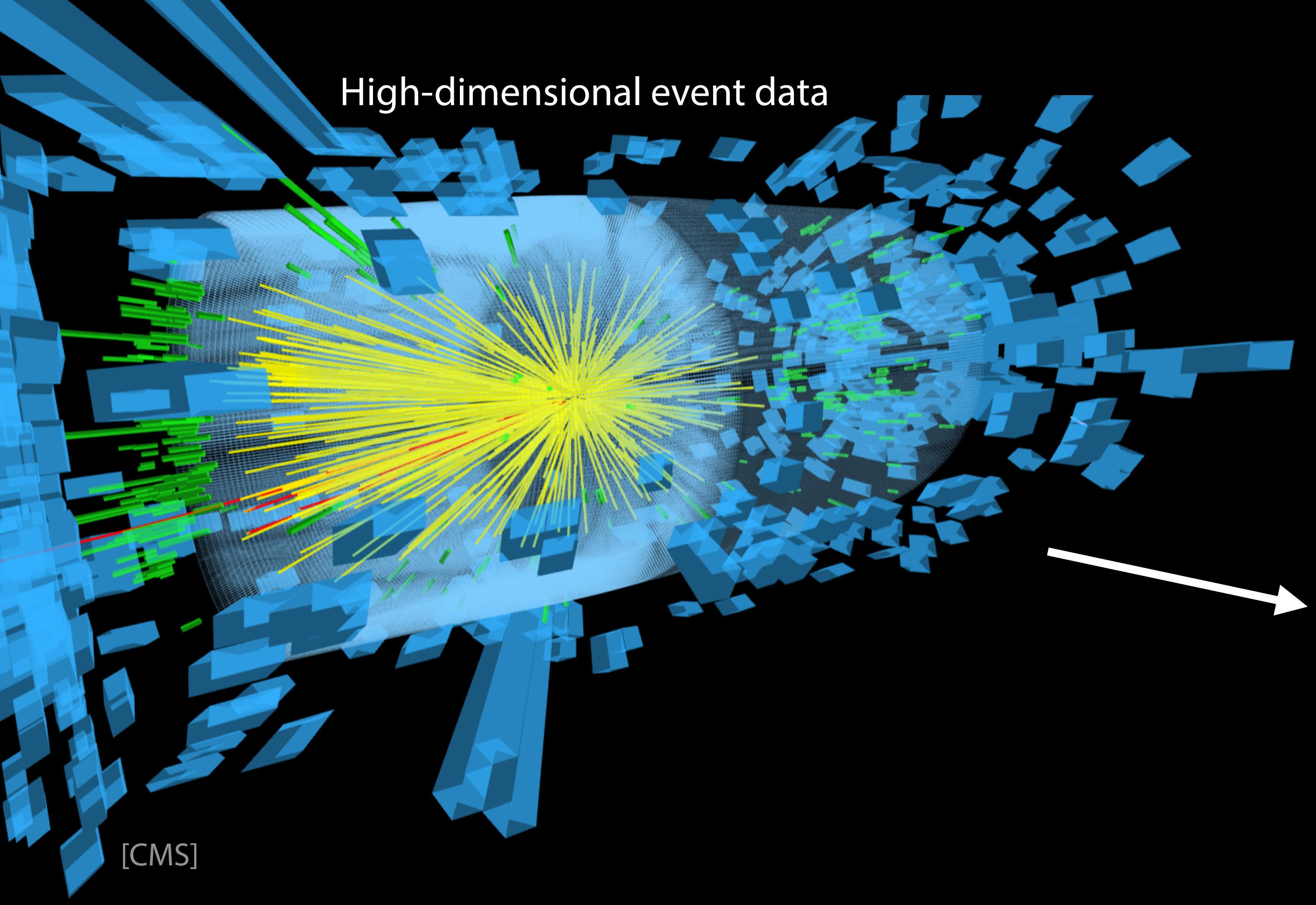
Marco Farina

Thanks to Kyle, Gilles, Felix, Irina, and Sam for material and inspiration for slides!

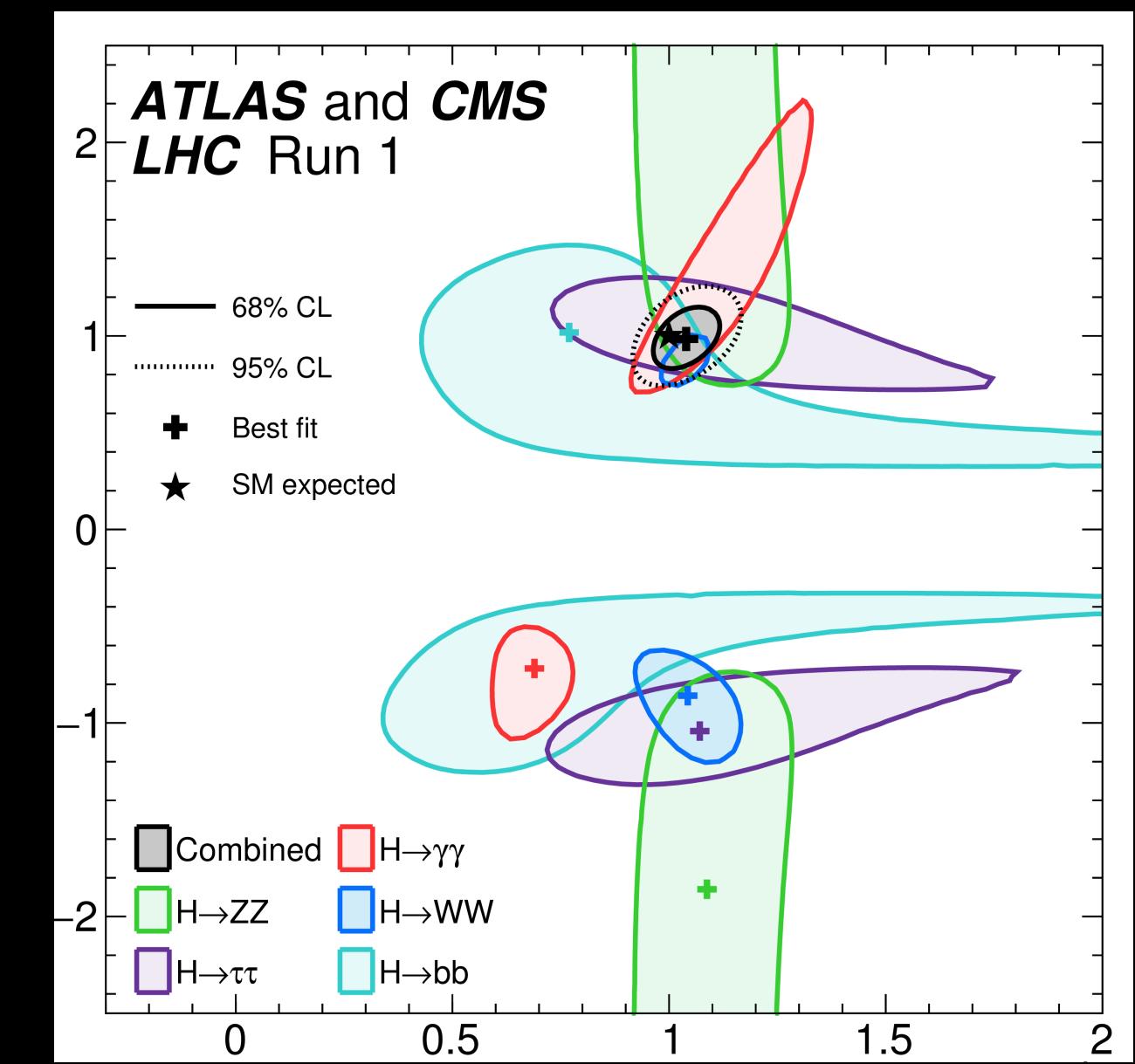


The SCAILFIN Project
scailfin.github.io



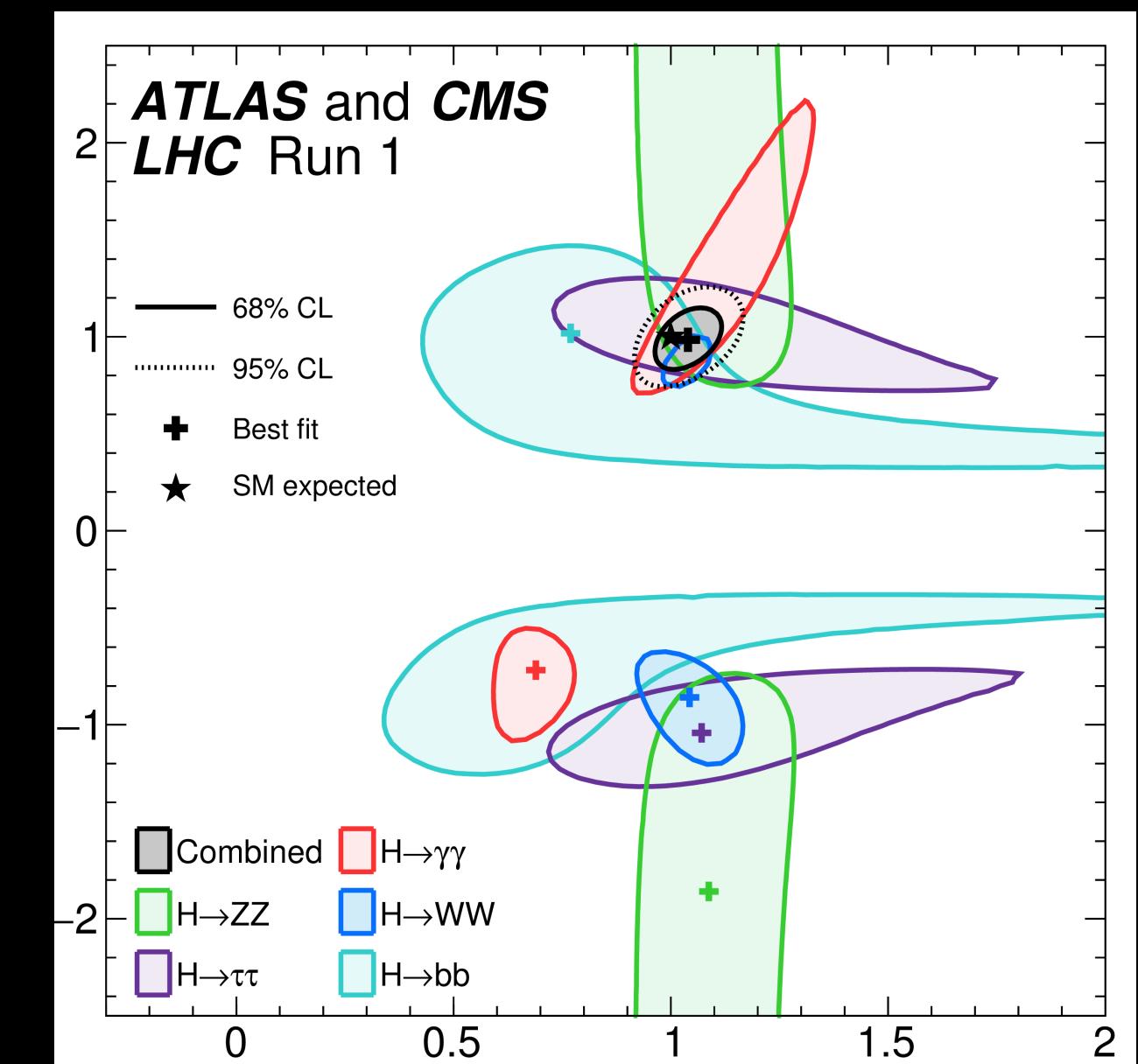
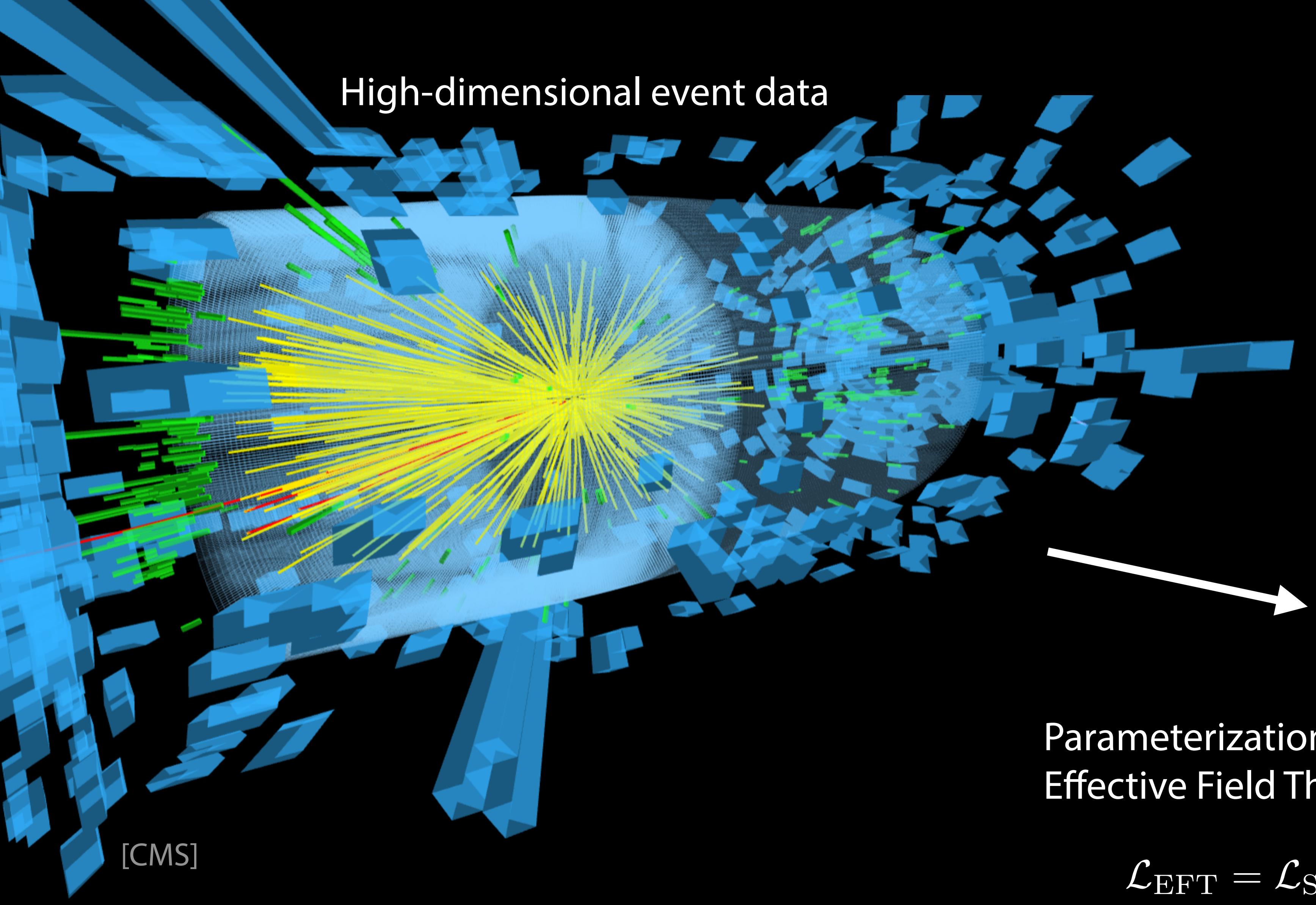


[CMS]



Precision constraints on
indirect effects of new physics

[ATLAS, CMS 1606.022266]



[ATLAS, CMS 1606.022266]

Precision constraints on
indirect effects of new physics

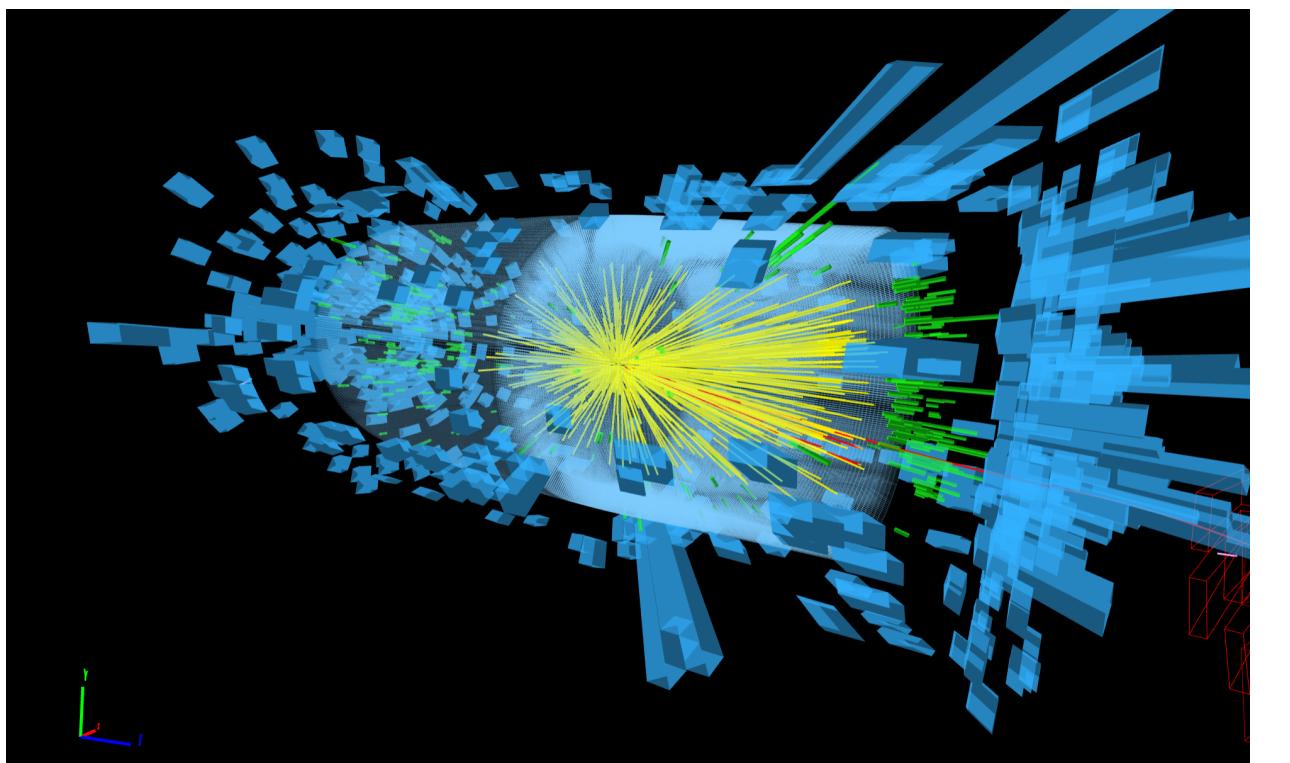
Parameterization in
Effective Field Theory:

$$\mathcal{L}_{\text{EFT}} = \mathcal{L}_{\text{SM}} + \sum_i \frac{f_i}{\Lambda^2} \mathcal{O}_i + \dots$$

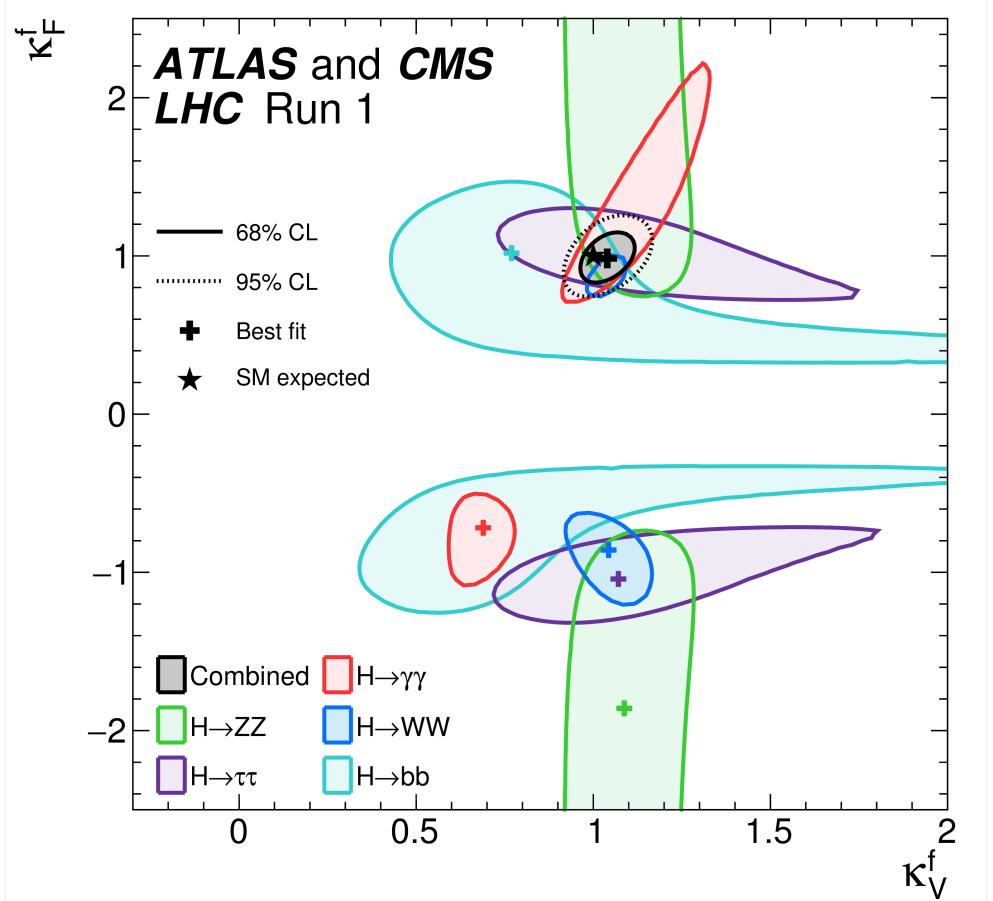
10s to 1000s “universal”
parameters to measure

systematic expansion of
new physics around
Standard Model

Takeaways

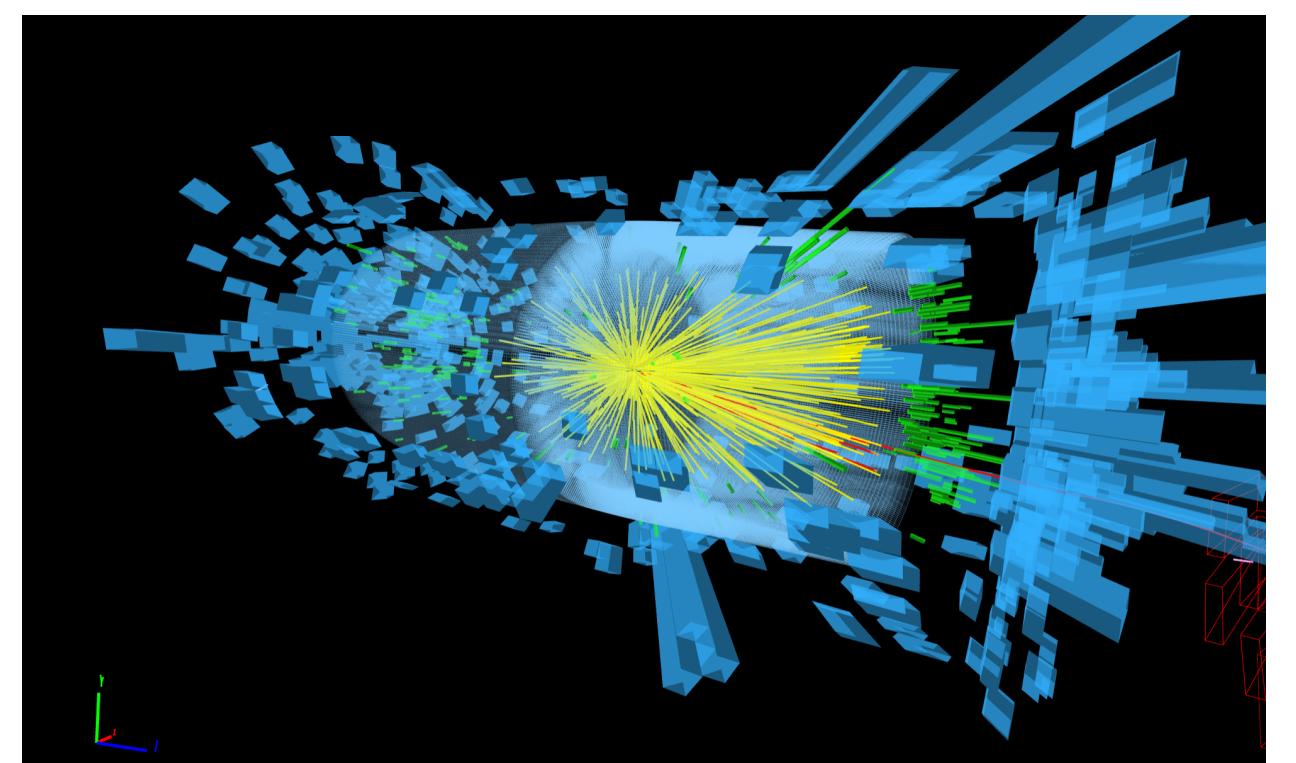


High-dimensional
event data x



Constraints on
parameters θ

Takeaways

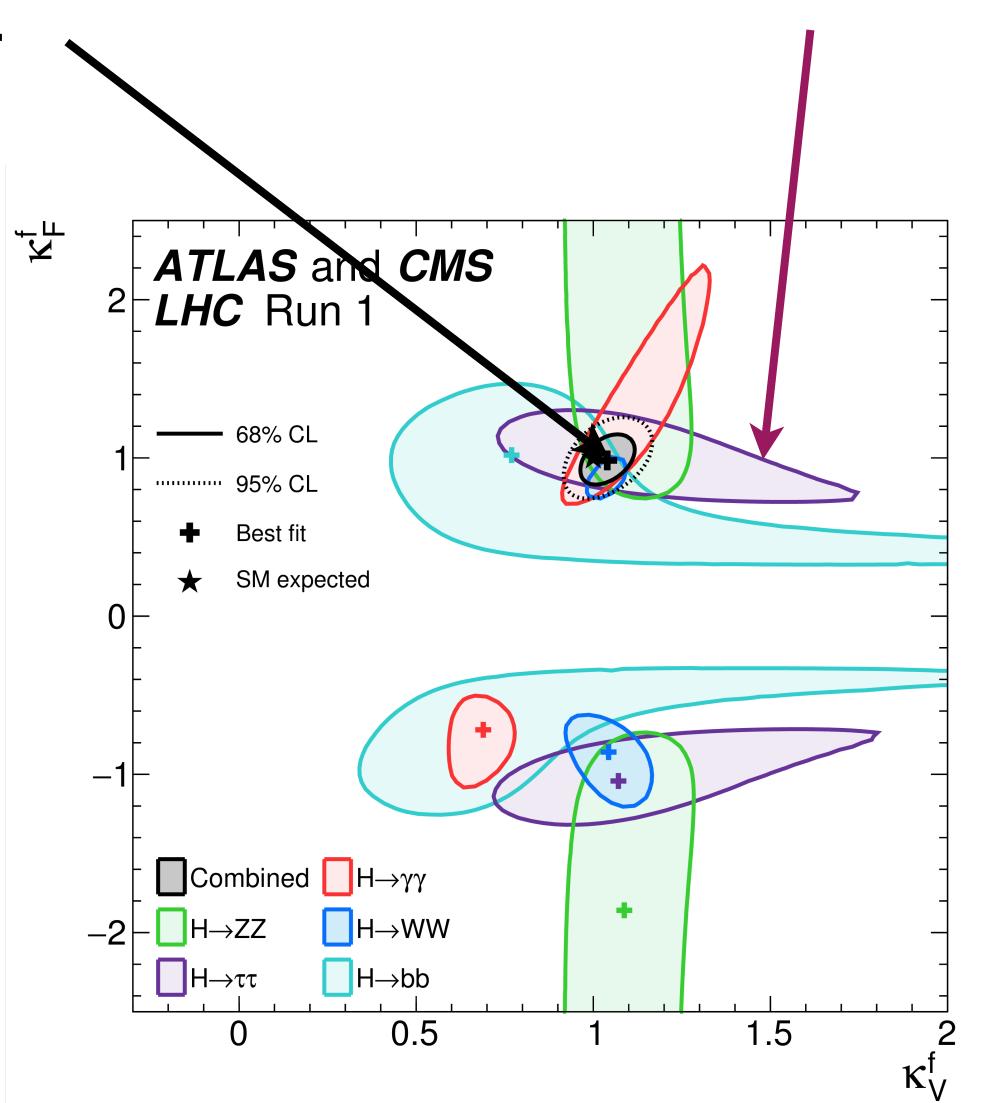


High-dimensional
event data x



Likelihood function
 $p(x|\theta)$

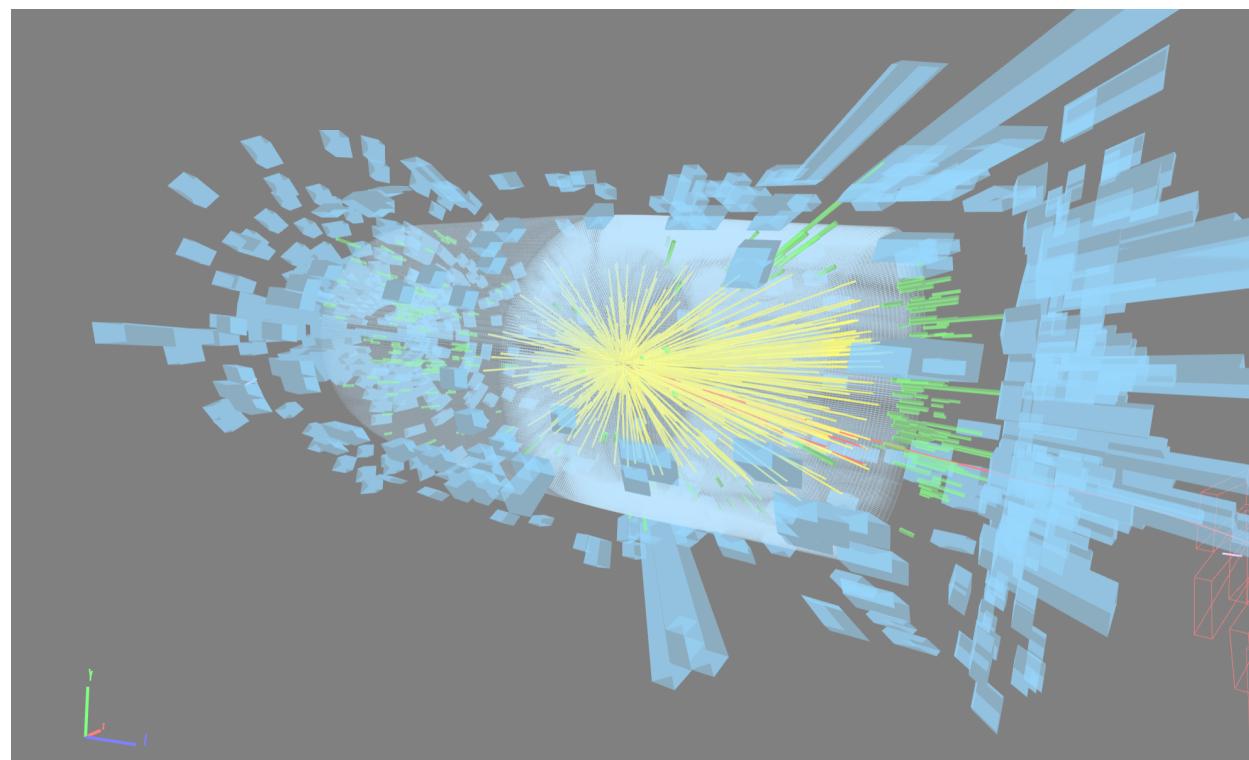
Maximum-likelihood
estimator



Confidence limits based
on likelihood ratio tests

Constraints on
parameters θ

Takeaways

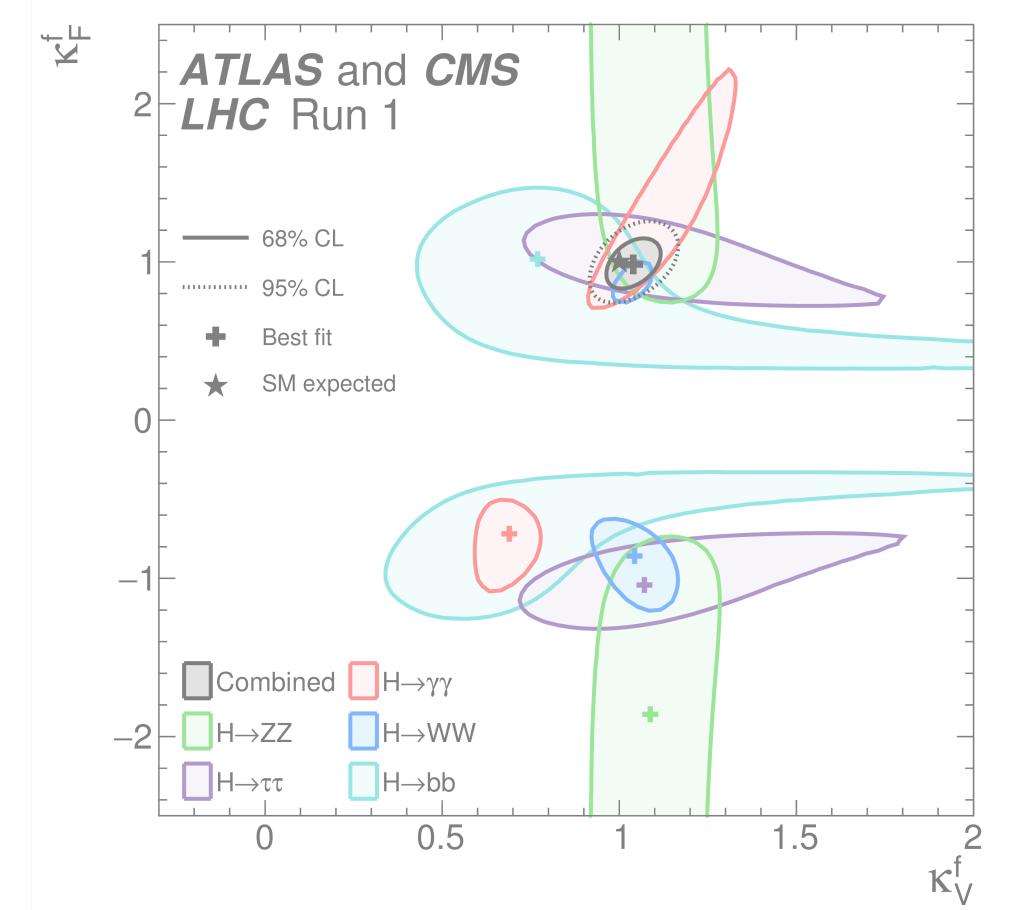


High-dimensional
event data x

Surprisingly, when we want to use high-dimensional data and have to deal with the detector response, we do not have a good way to calculate the likelihood.

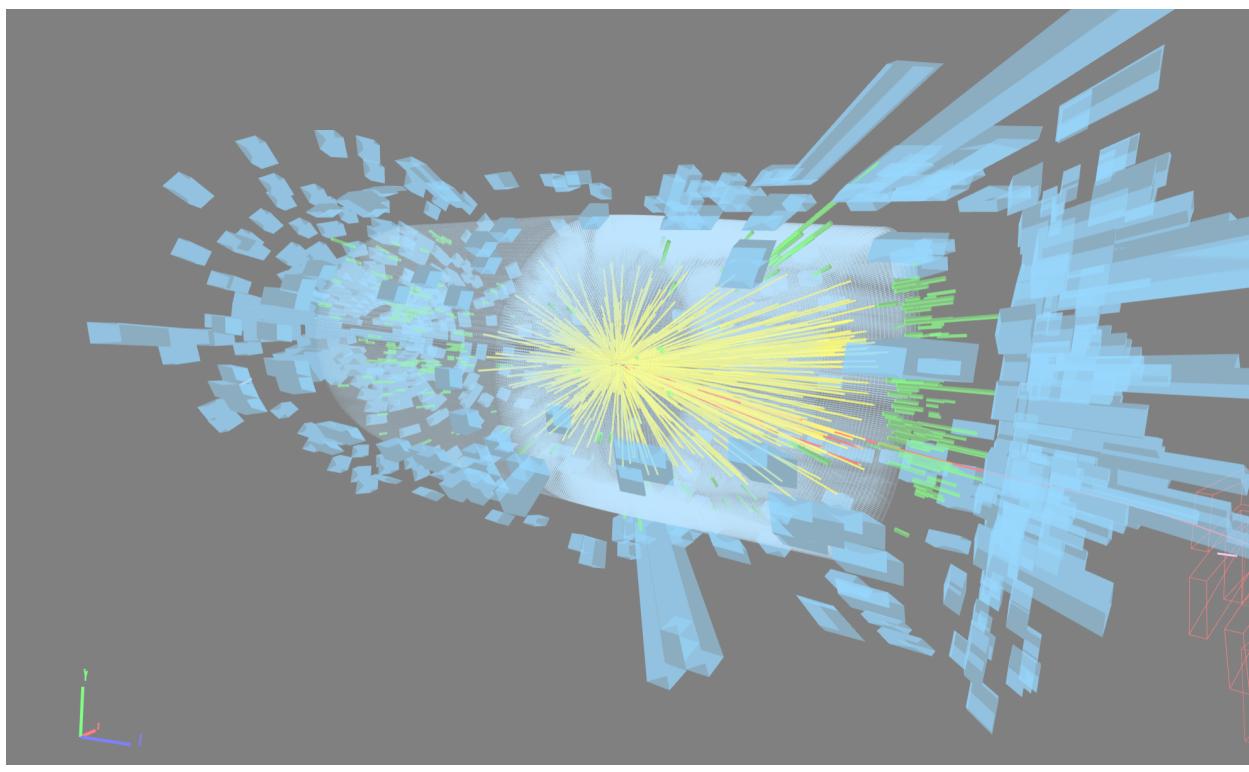


Likelihood function
 $p(x|\theta)$

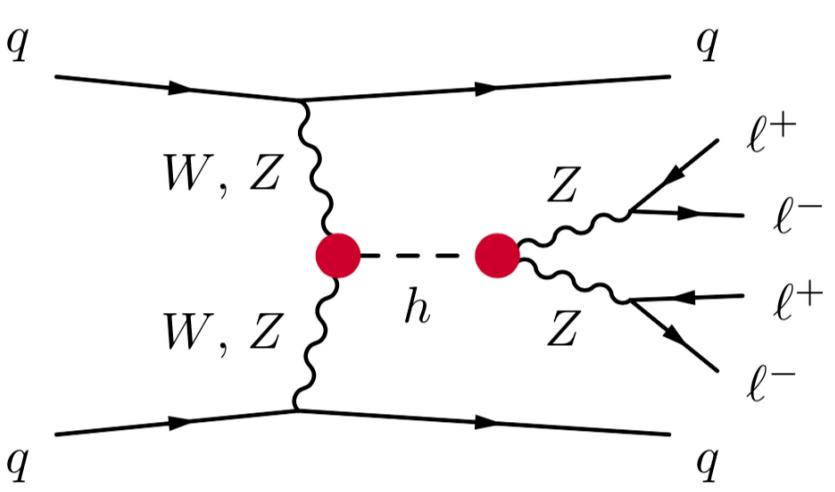


Constraints on
parameters θ

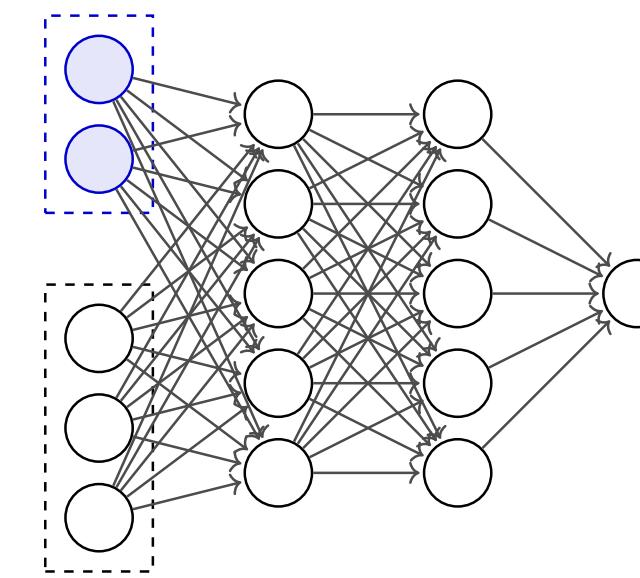
Takeaways



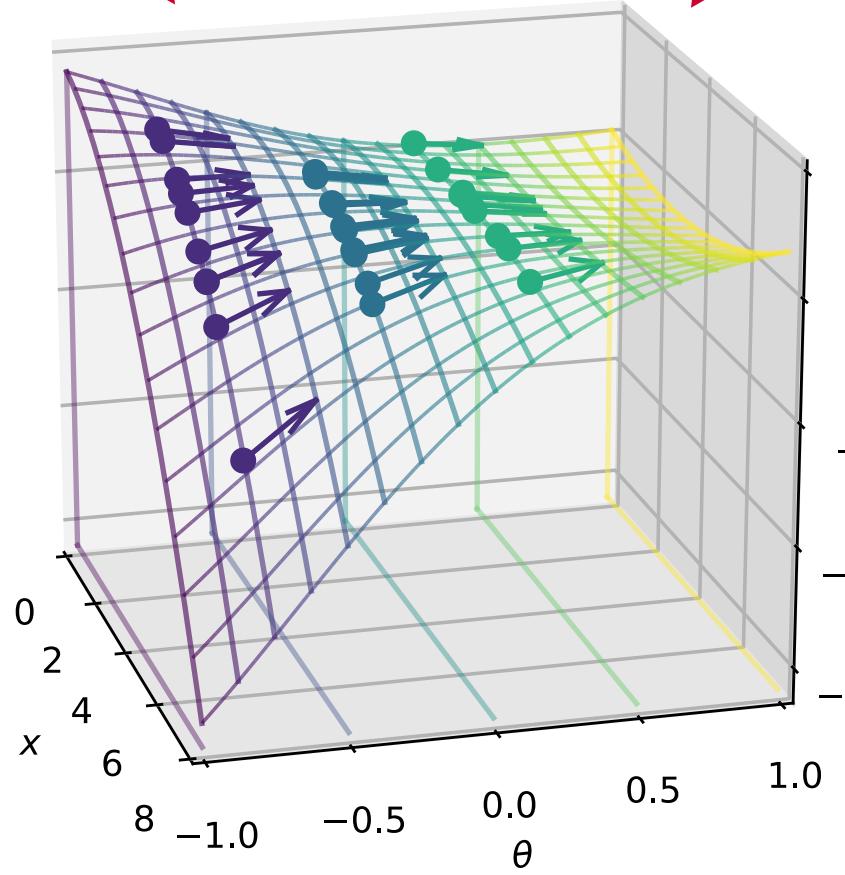
High-dimensional
event data x



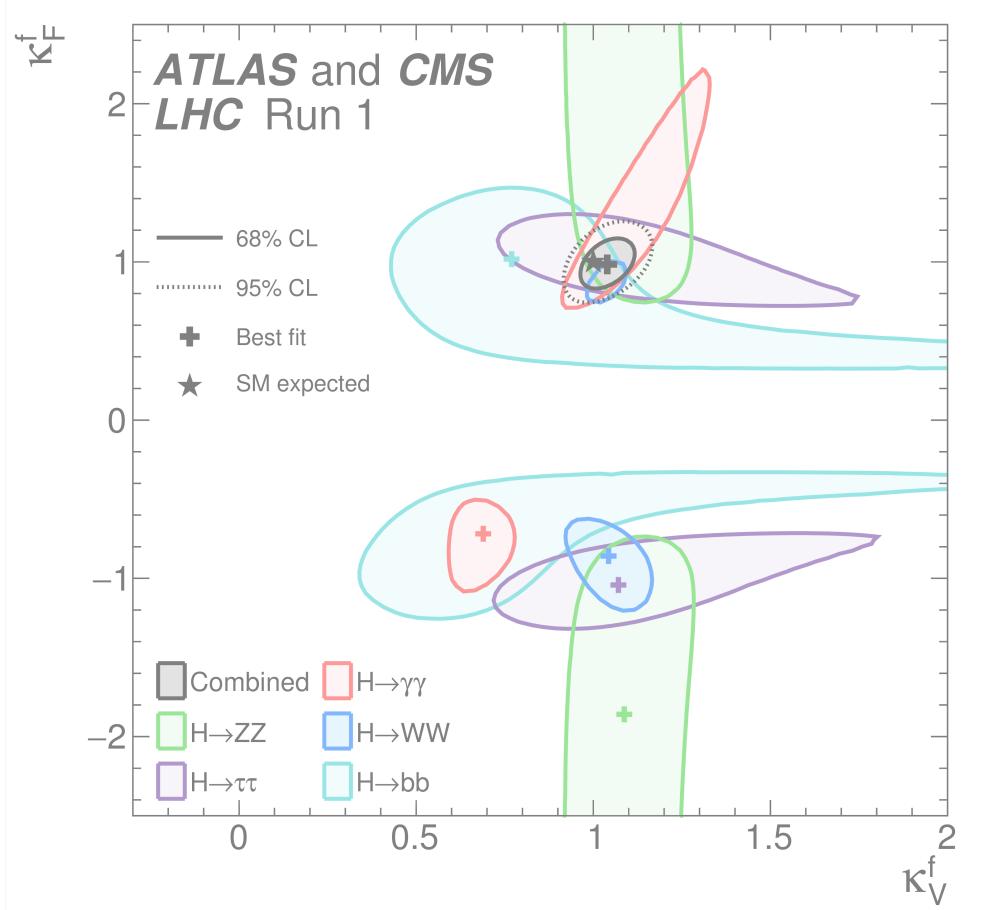
Physics insight:
matrix element information



Machine learning



Estimator of the
likelihood $p(x|\theta)$



Constraints on
parameters θ

LHC measurements as a likelihood-free inference problem

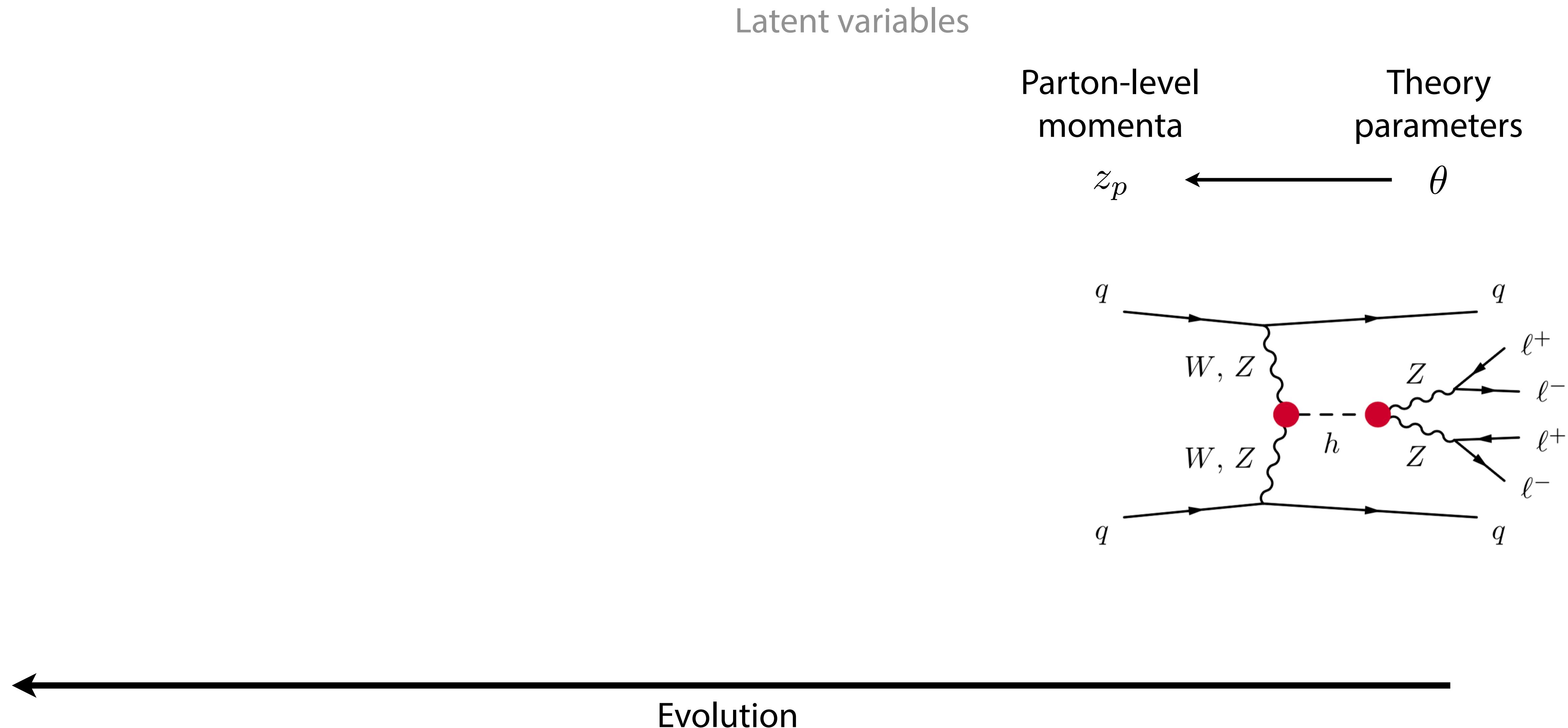
Modelling particle physics processes

Theory
parameters
 θ

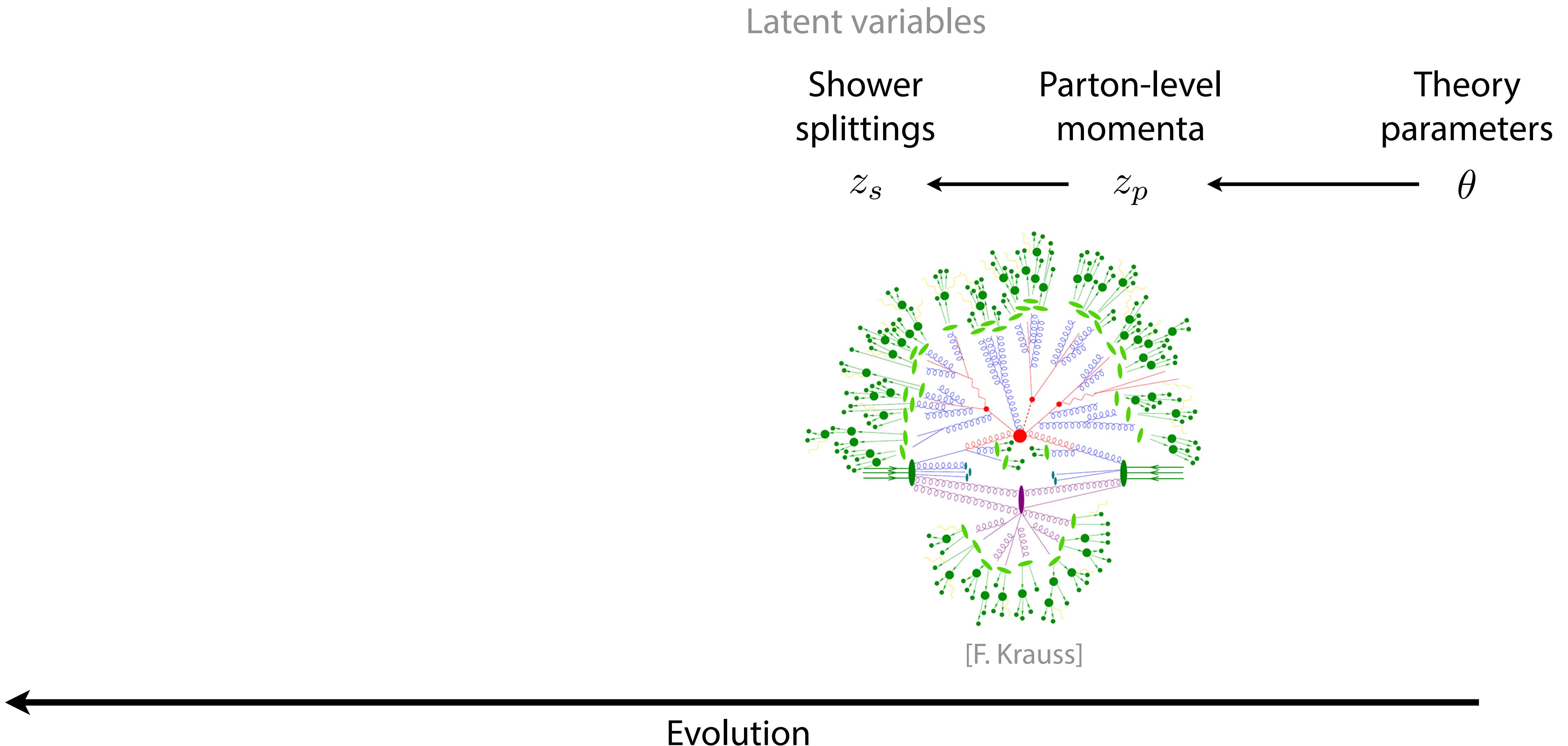


Evolution

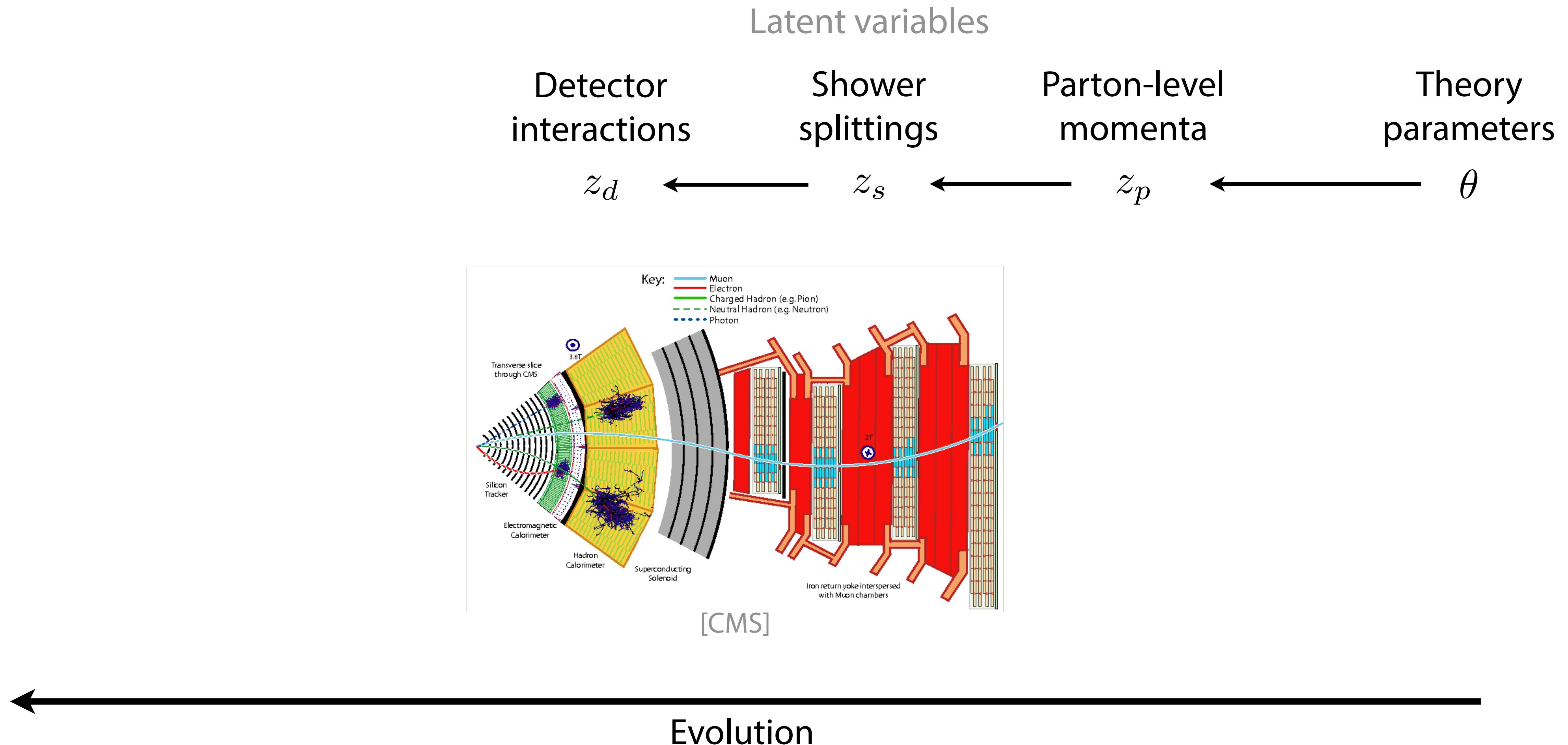
Modelling particle physics processes



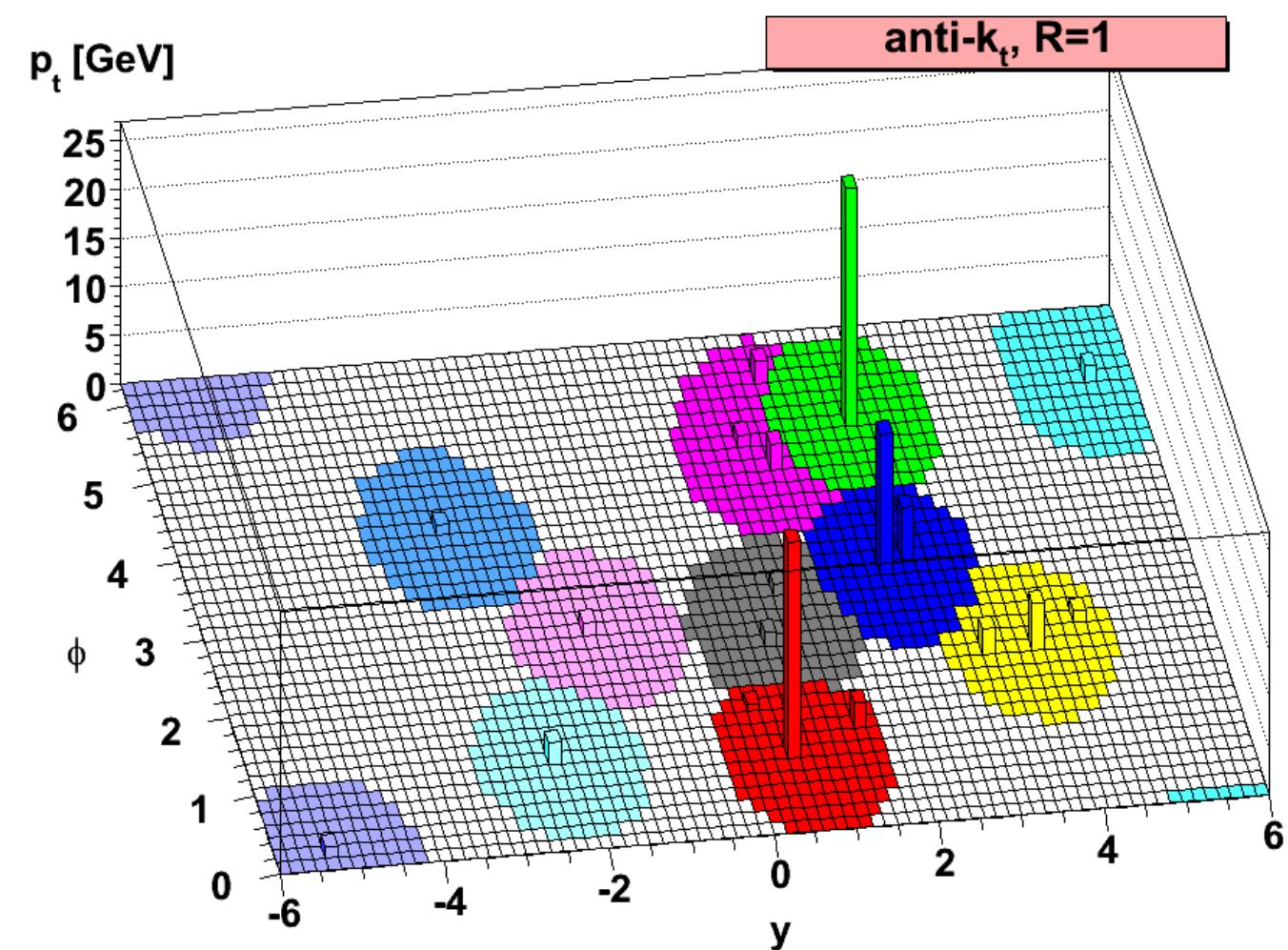
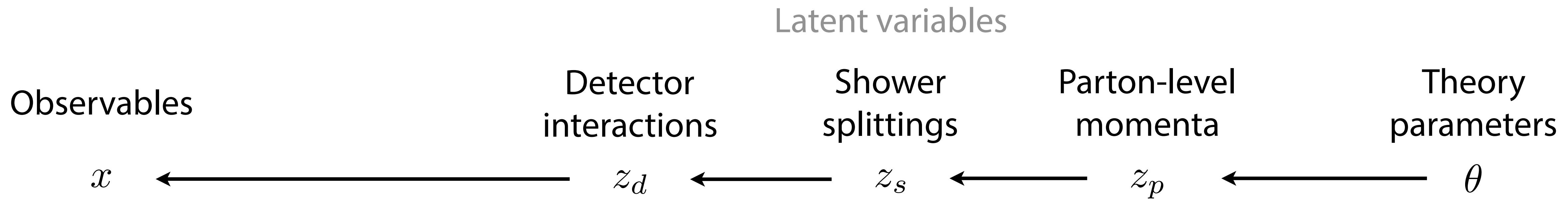
Modelling particle physics processes



Modelling particle physics processes



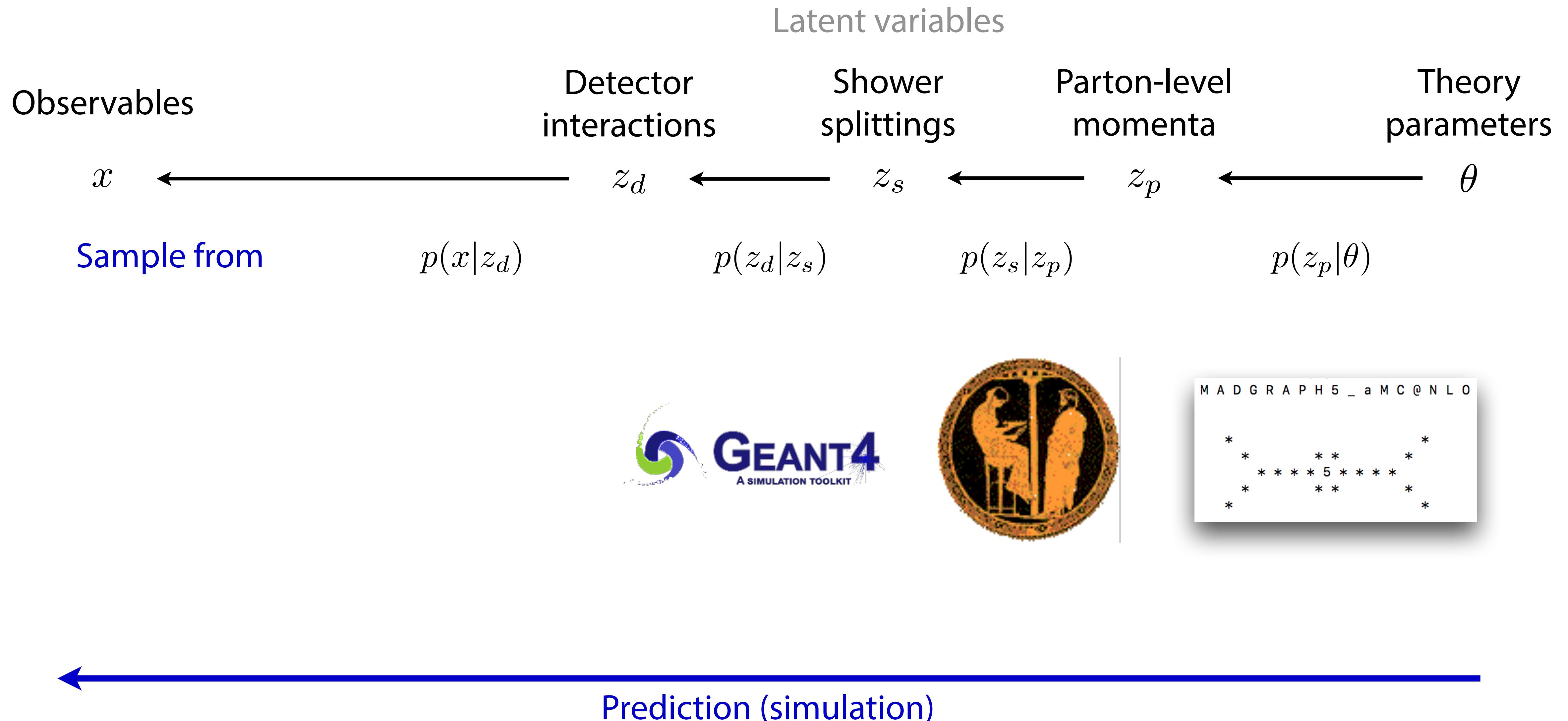
Modelling particle physics processes



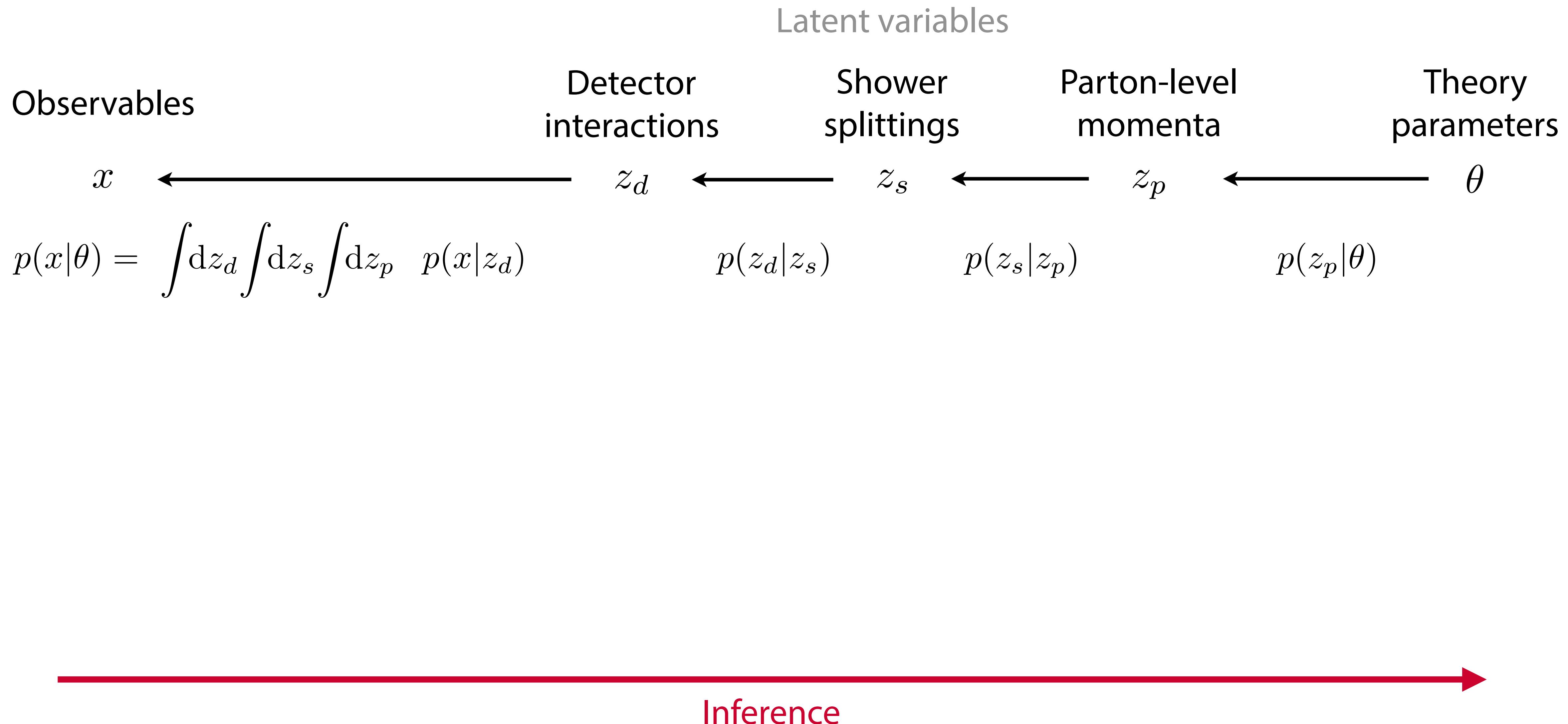
[M. Cacciari, G. Salam, G. Soyez 0802.1189]

Evolution

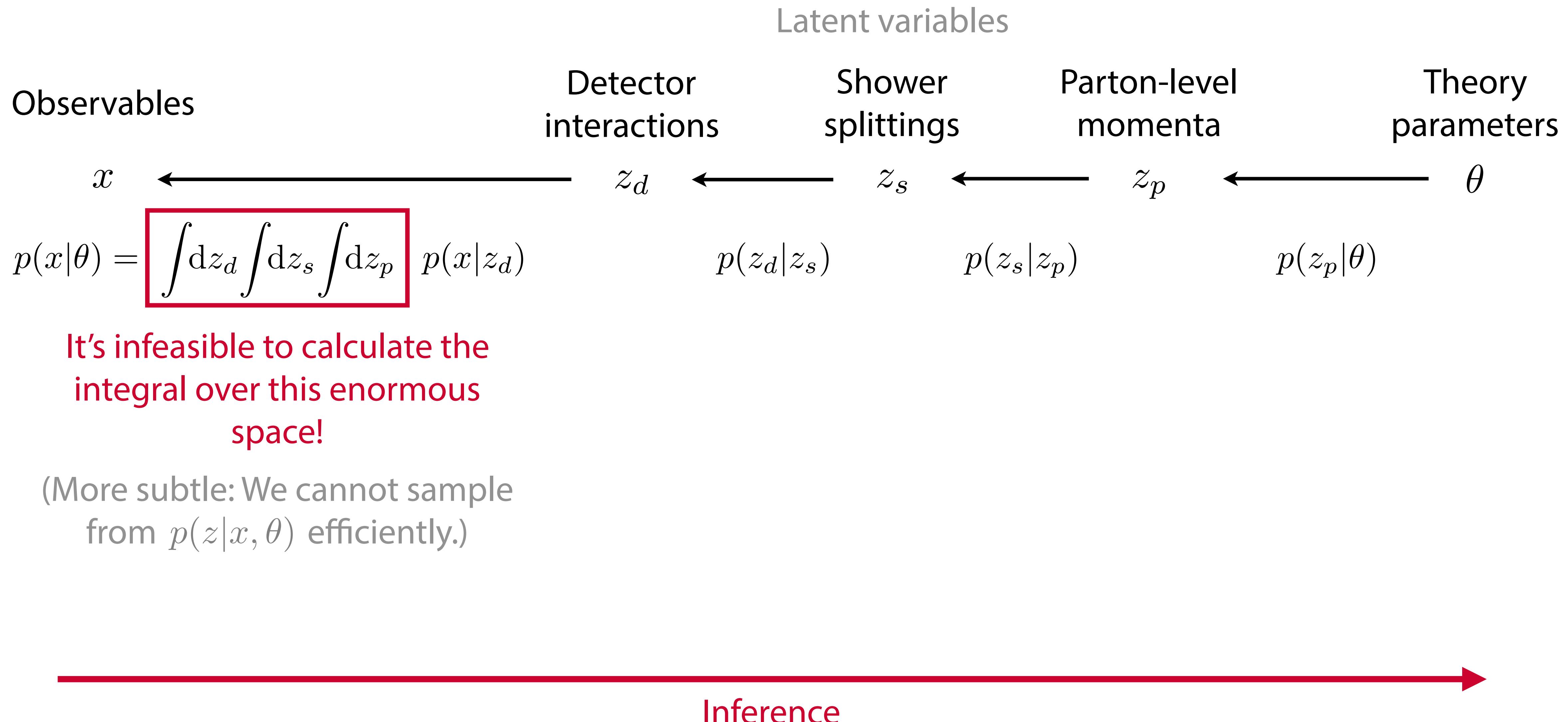
Modelling particle physics processes



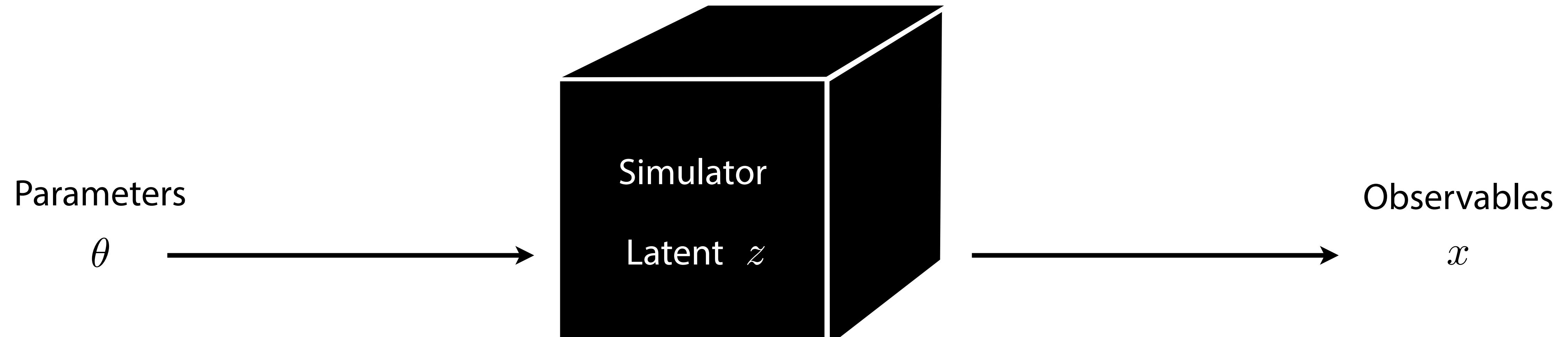
Modelling particle physics processes



Modelling particle physics processes



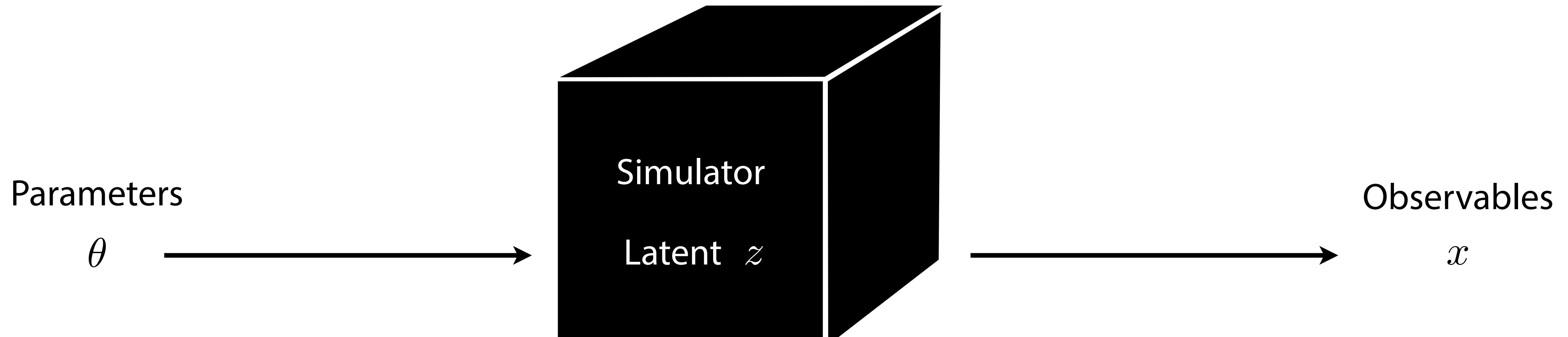
Likelihood-free inference / implicit models



Prediction:

- Well-understood mechanistic model
- Simulator can generate samples

Likelihood-free inference / implicit models



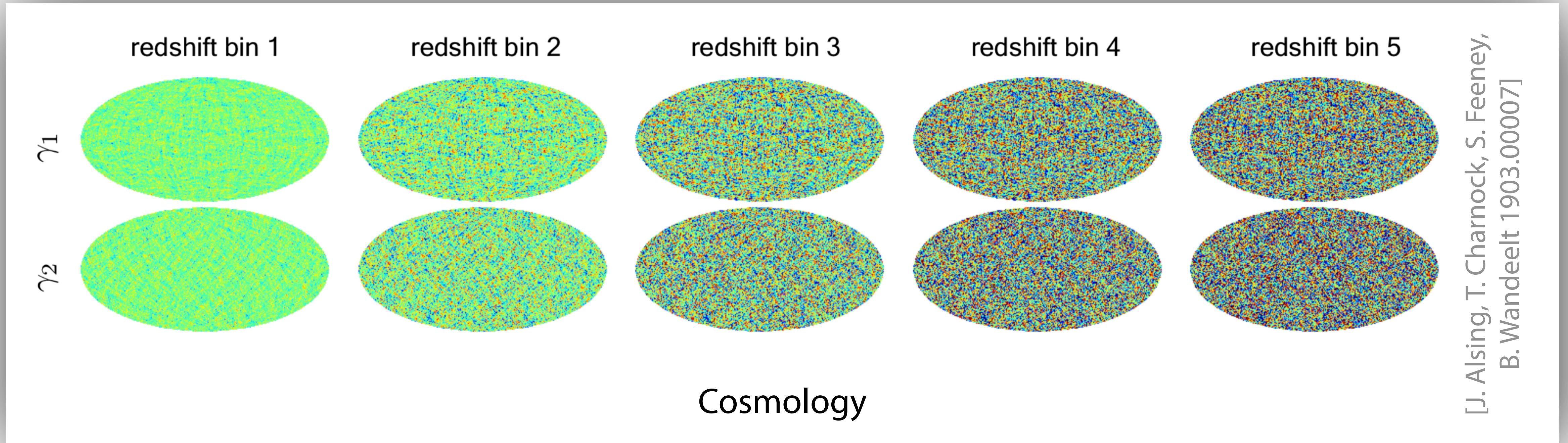
Prediction:

- Well-understood mechanistic model
- Simulator can generate samples

Inference:

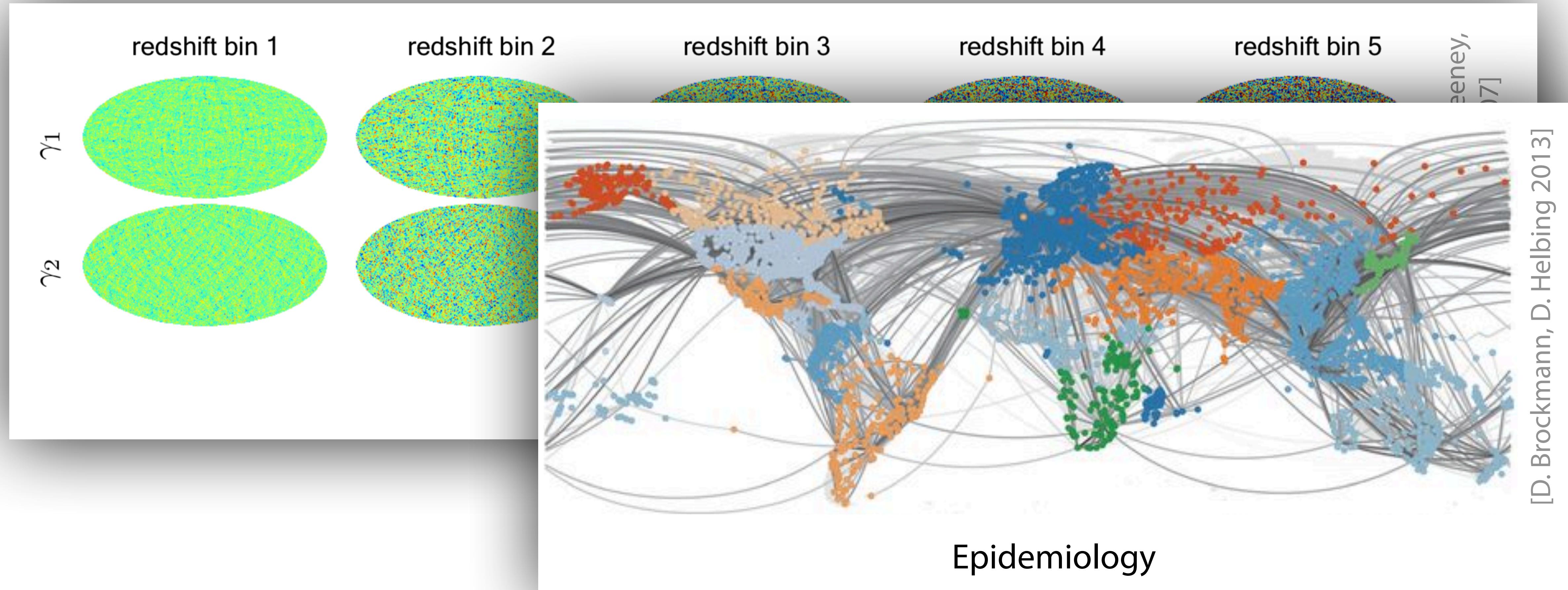
- Likelihood function $p(x|\theta)$ is intractable
- Inference based on estimator $\hat{p}(x|\theta)$

A thriving research field

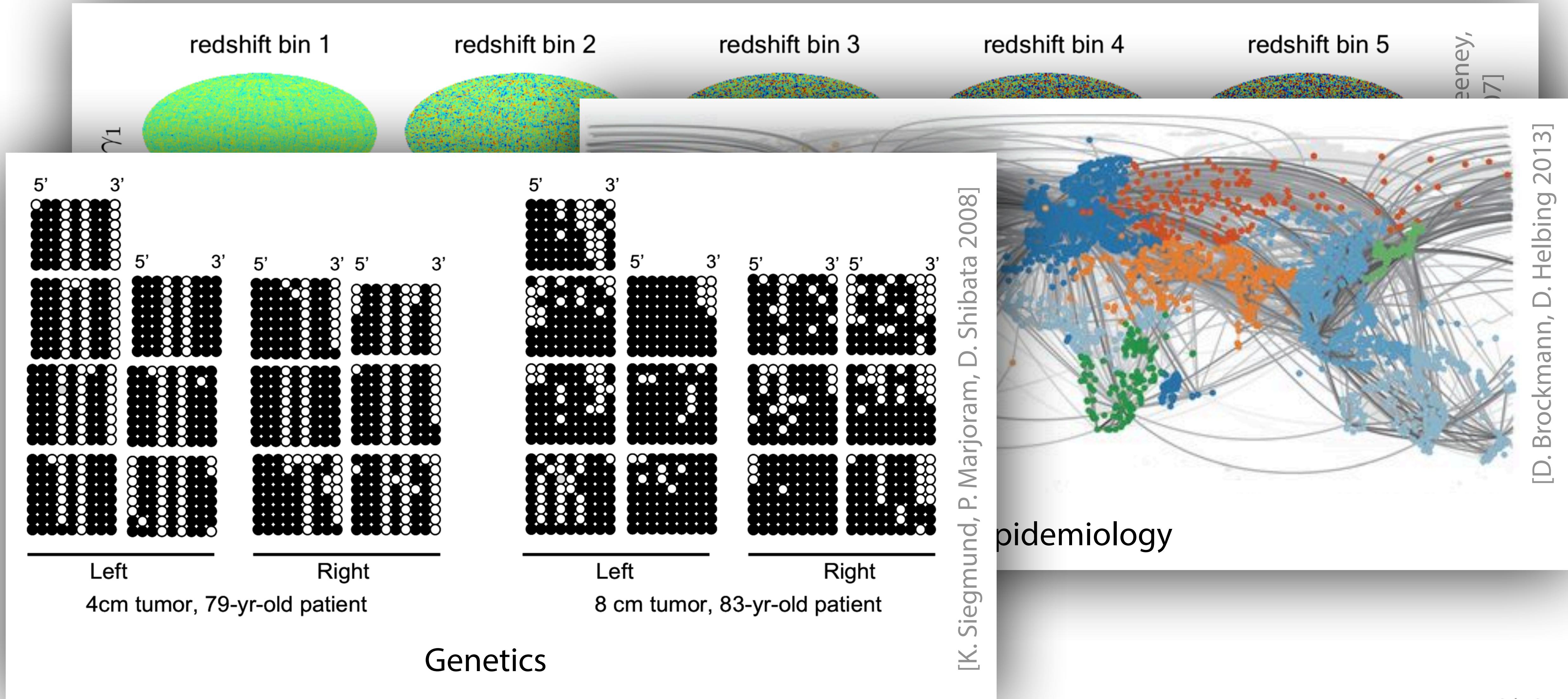


[J. Alsing, T. Charnock, S. Feeney,
B. Wandelt 1903.00007]

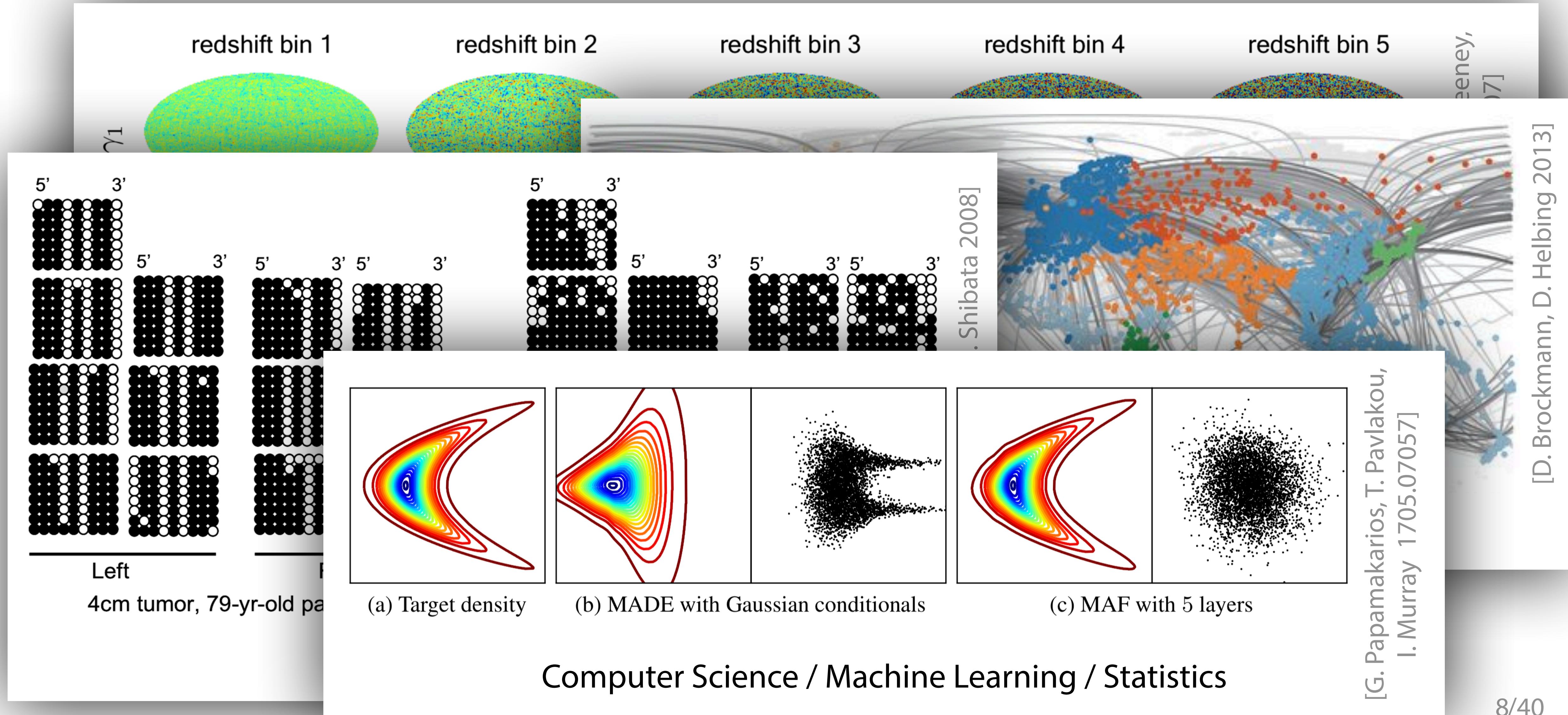
A thriving research field



A thriving research field

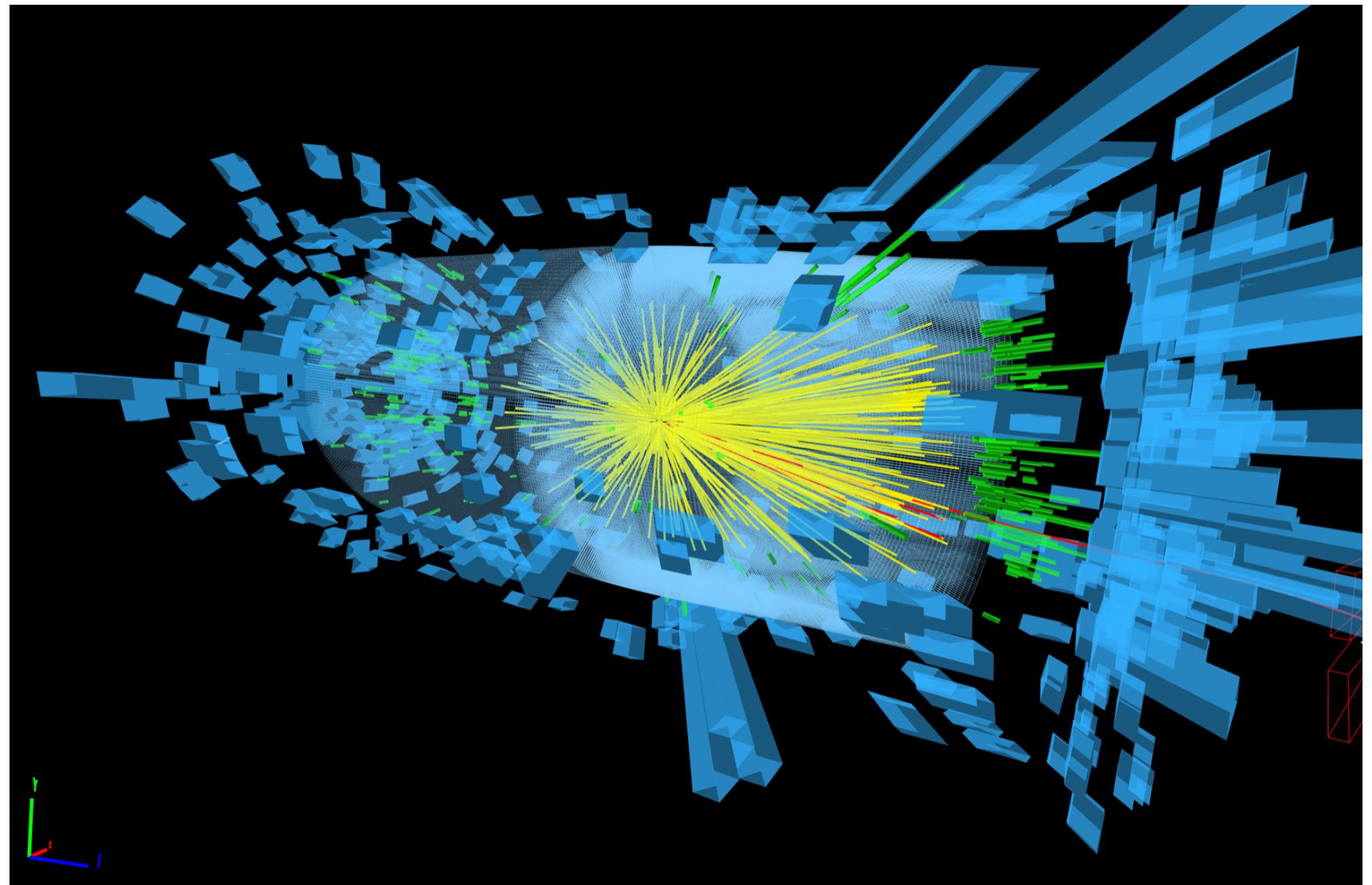


A thriving research field



Why has that not stopped us before?

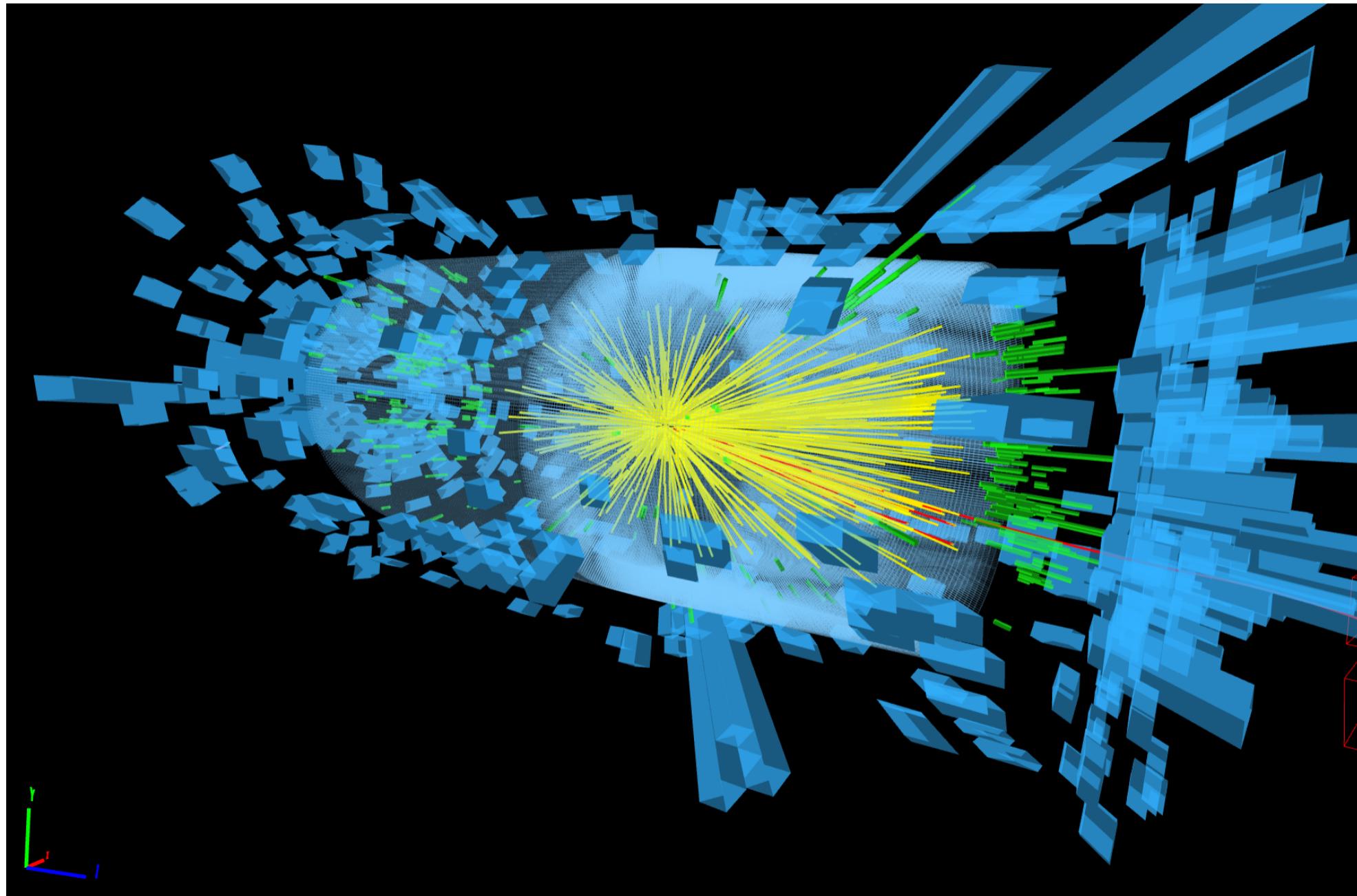
Solve it with summary statistics



High-dimensional event data x

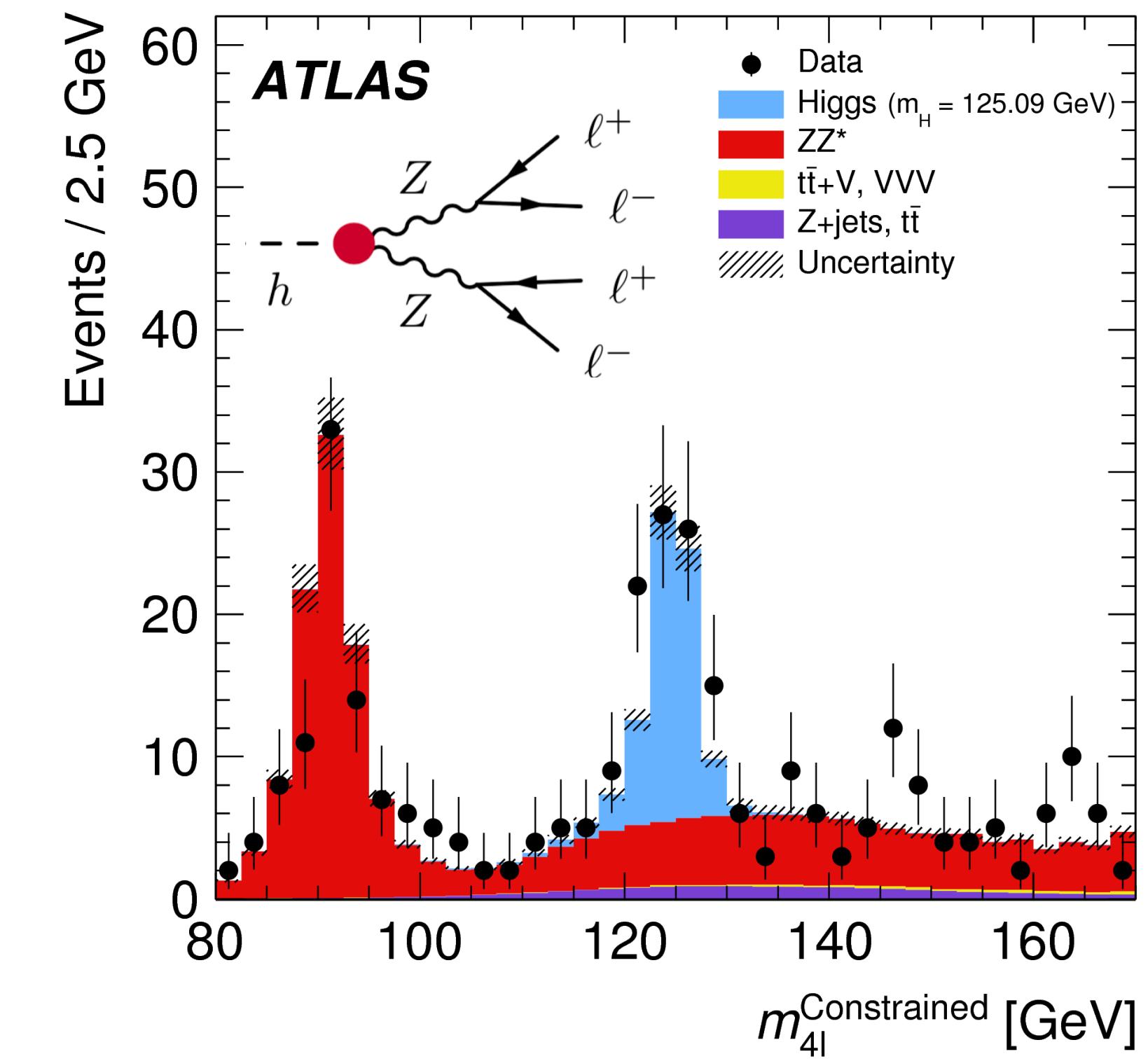
$p(x|\theta)$ cannot be calculated

Solve it with summary statistics



High-dimensional event data x

$p(x|\theta)$ cannot be calculated



One or two summary statistics x'

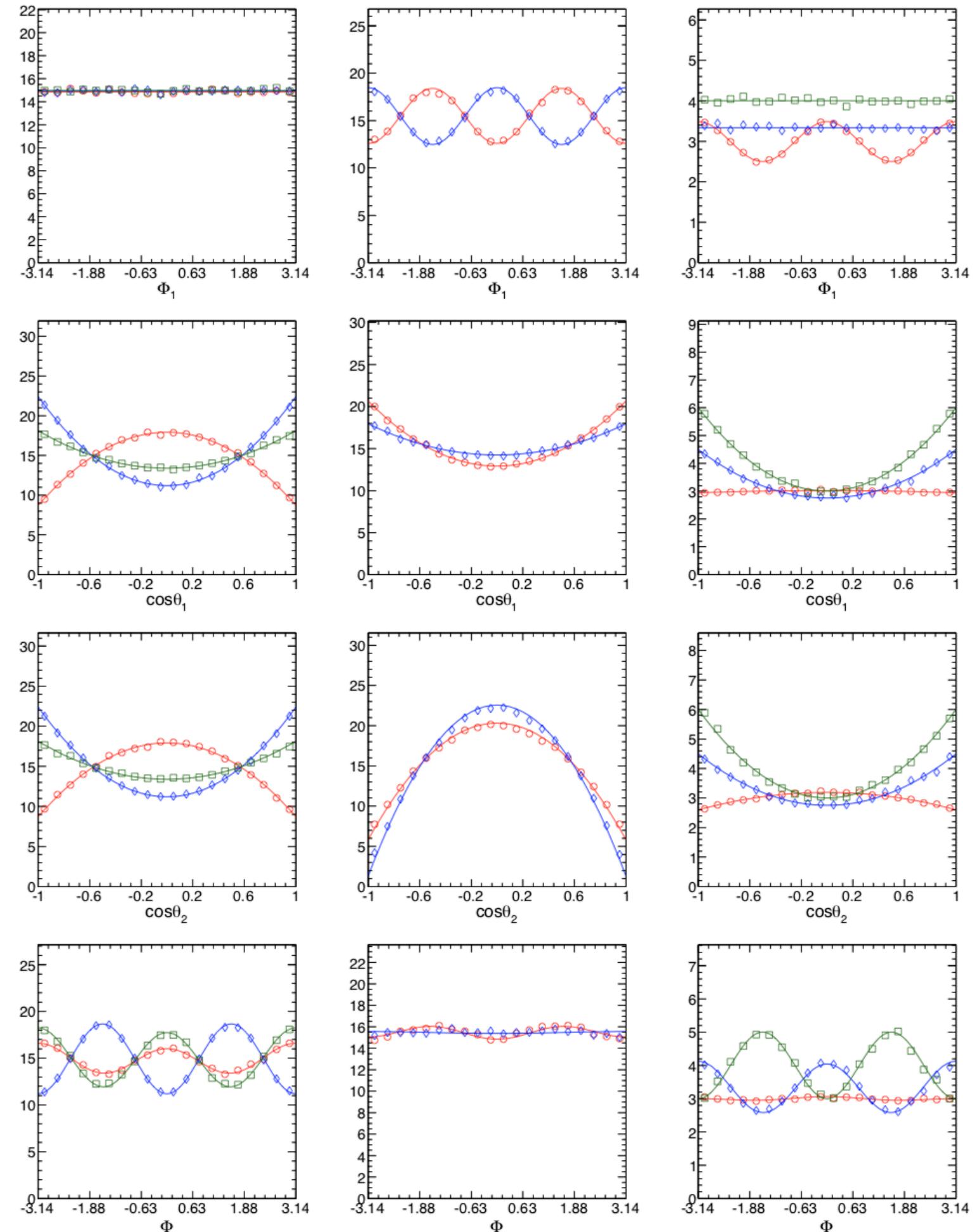
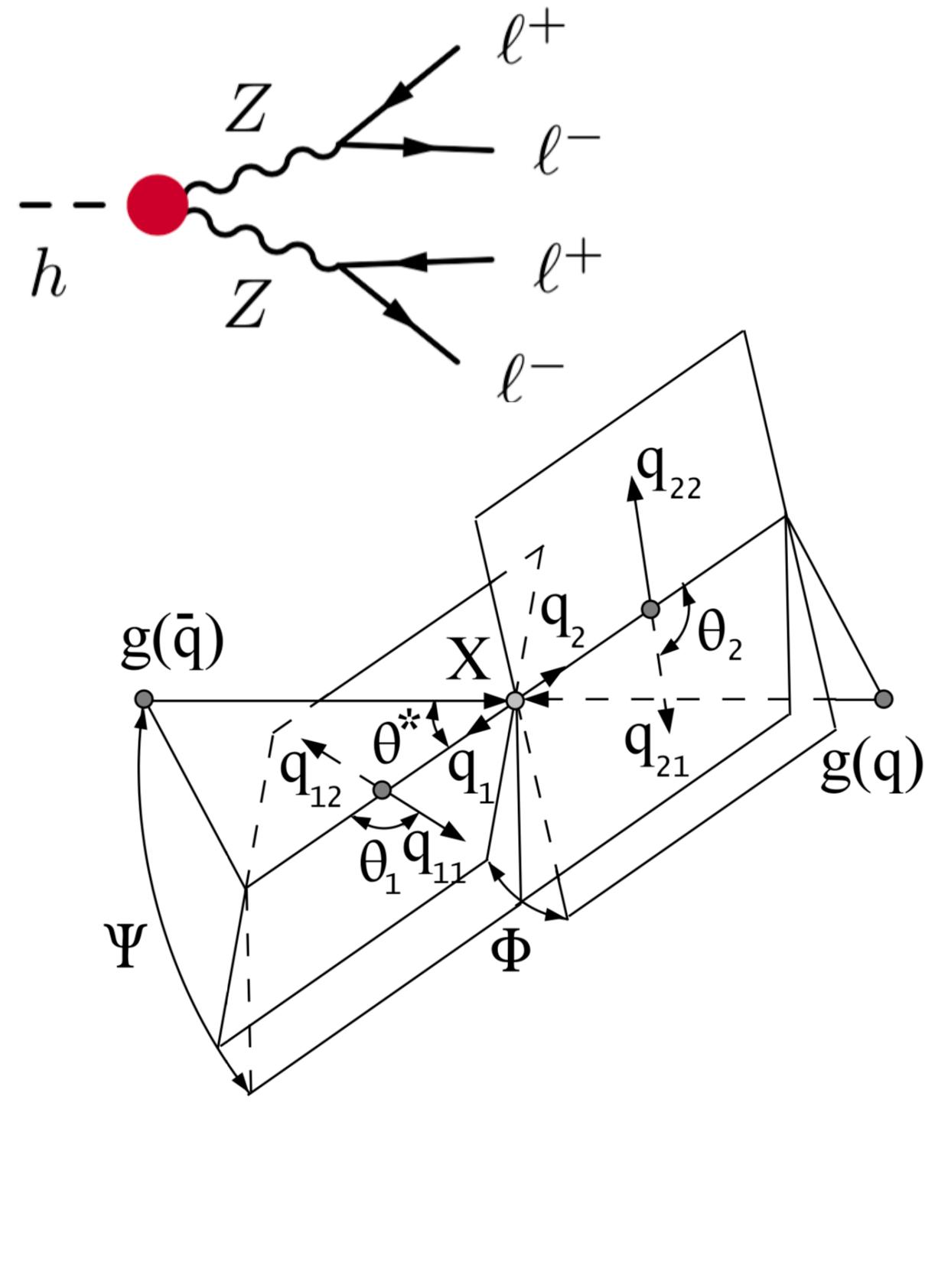
$p(x'|\theta)$ can be estimated
with histograms, KDE, ...

Summary statistics for EFT measurements?

- Choosing summary statistics x' is difficult and problem-dependent
- Often there is no single good standard variable — compressing to any x' loses information!
[JB, K. Cranmer, F. Kling, T. Plehn 1612.05261;
JB, F. Kling, T. Plehn , T. Tait 1712.02350]
- Ideally: analyze high-dimensional x including all correlations (“fully differential cross section”)

Summary statistics for EFT measurements?

- Choosing summary statistics x' is difficult and problem-dependent
 - Often there is no single good standard variable — compressing to any x' loses information!
- [JB, K. Cranmer, F. Kling, T. Plehn 1612.05261;
 JB, F. Kling, T. Plehn , T. Tait 1712.02350]
- Ideally: analyze high-dimensional x including all correlations (“fully differential cross section”)

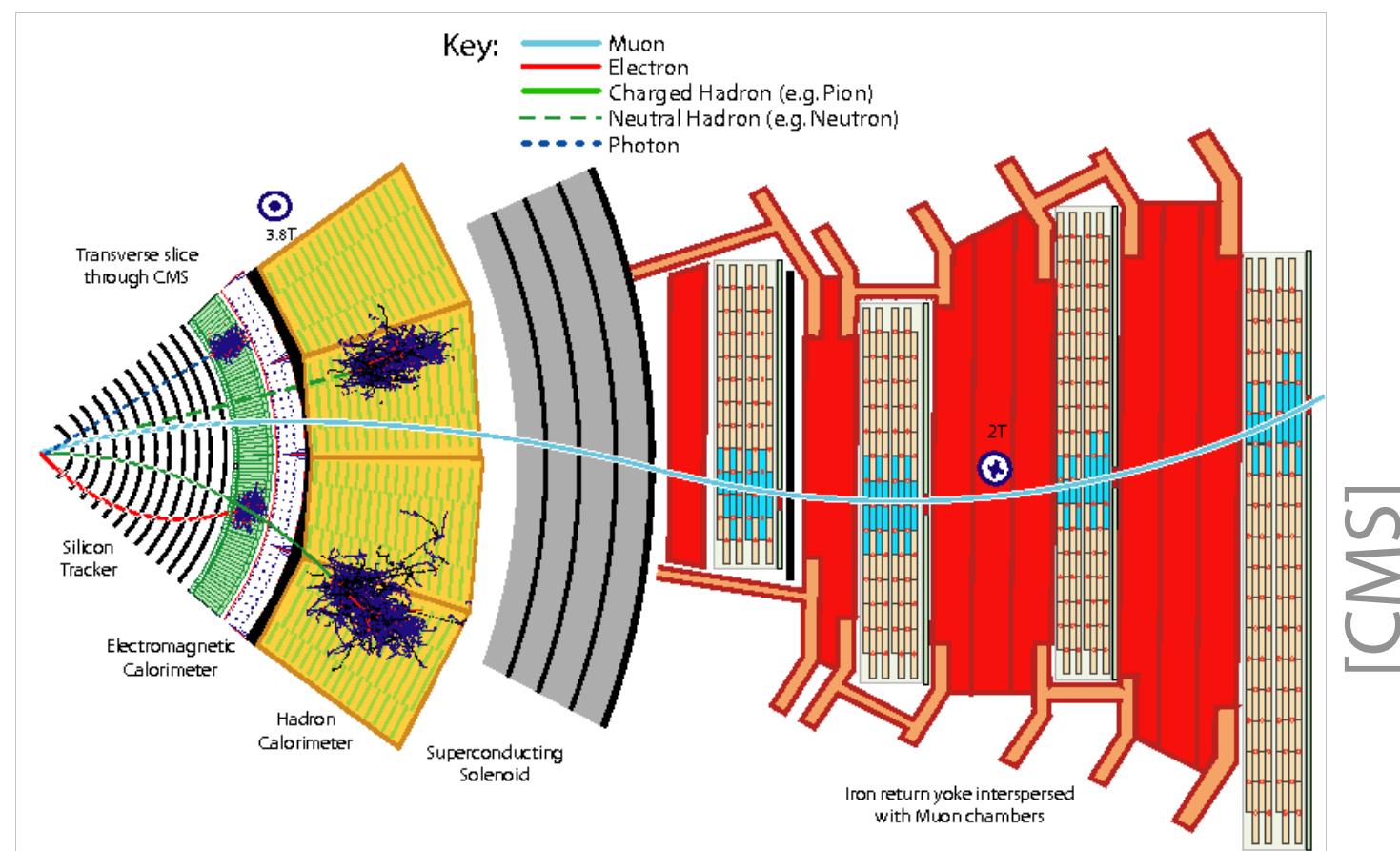


[Bolognesi et al. 1208.4018]

Solve it by approximating the integral

- Problem: high-dim. integral over shower / detector trajectories

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



Solve it by approximating the integral

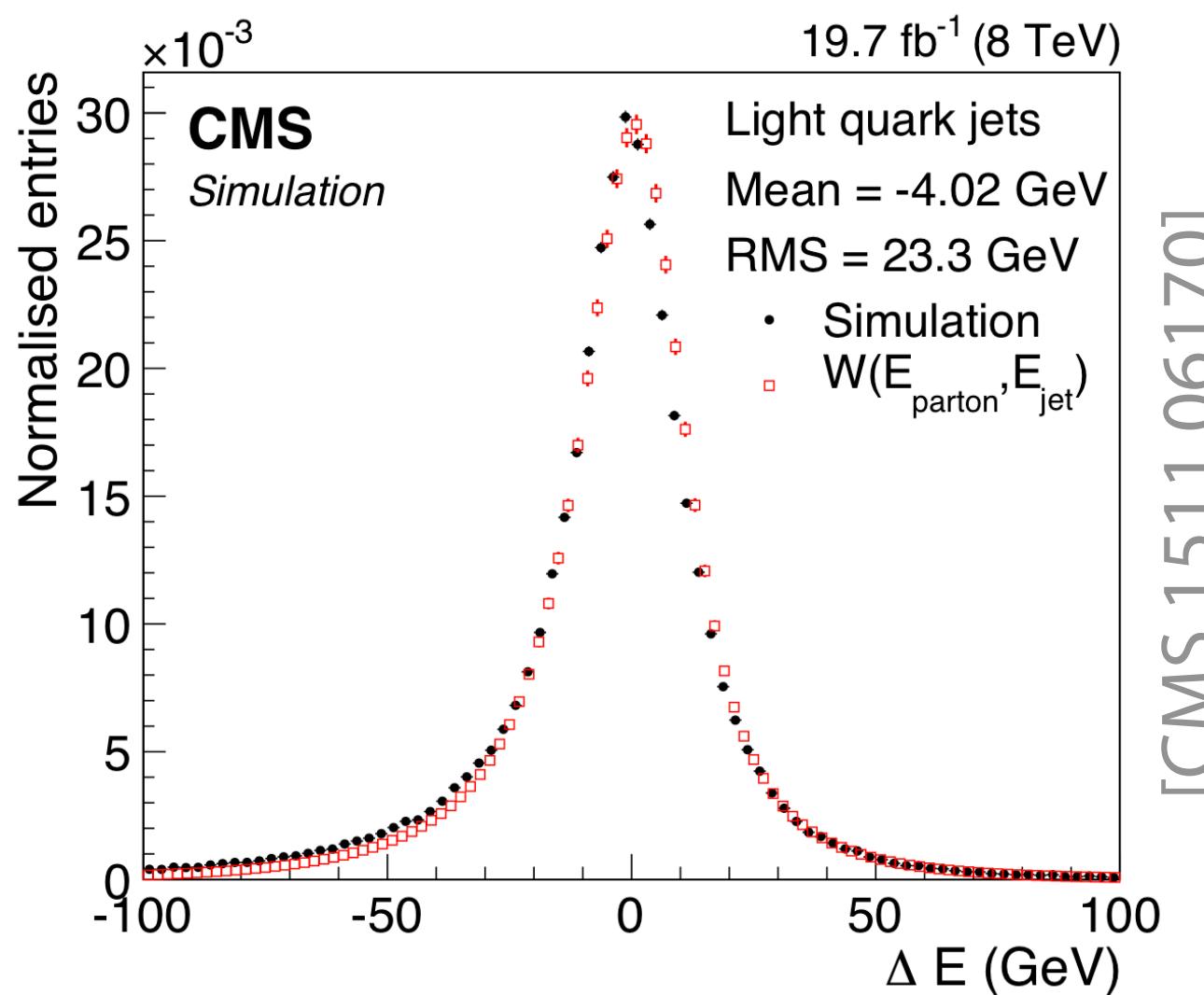
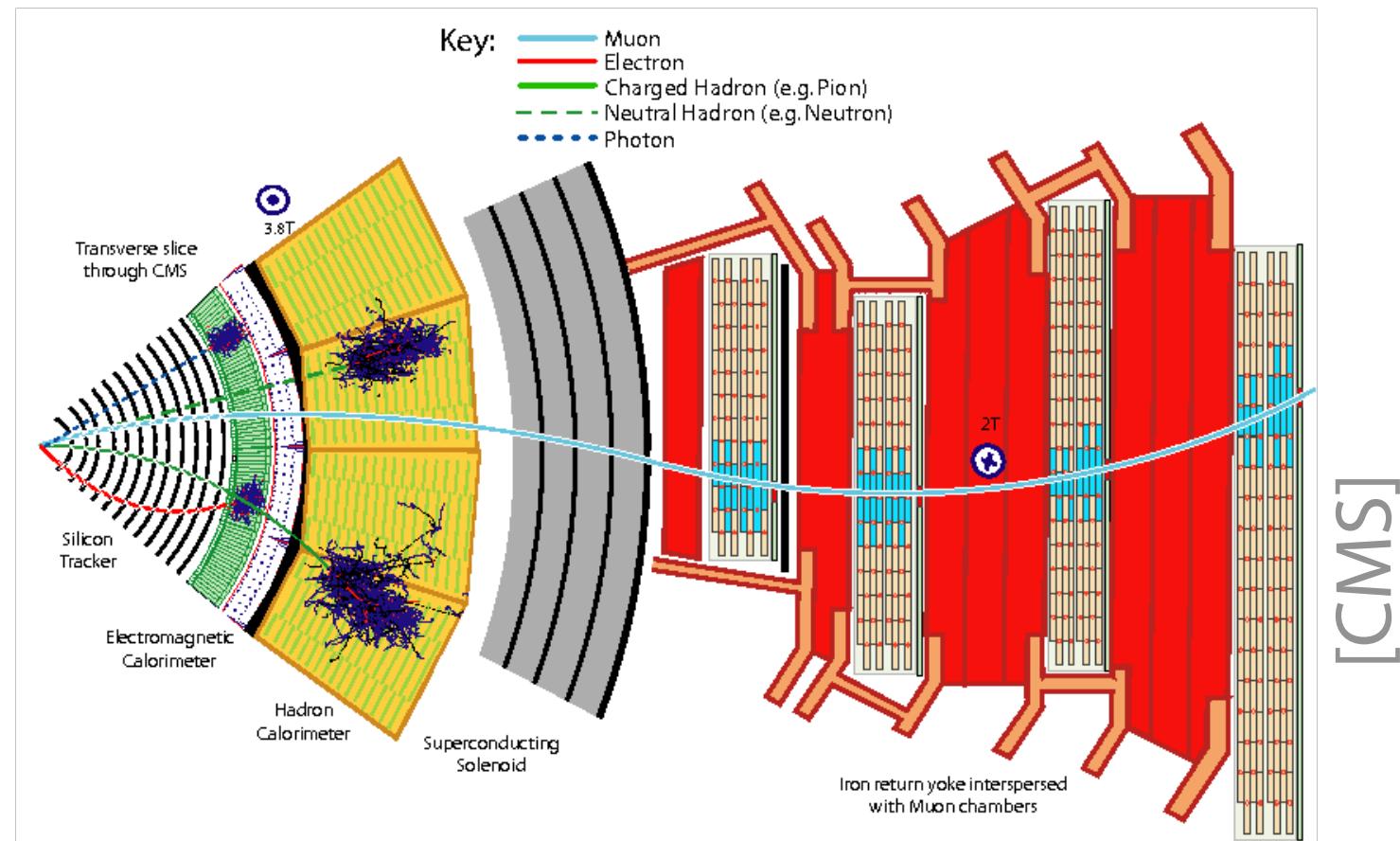
- Problem: high-dim. integral over **shower / detector trajectories**

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

- Matrix Element Method (MEM): [K. Kondo 1988]

- approximate **shower + detector effects** into **transfer function** $\hat{p}(x|z_p)$
- explicitly calculate remaining integral

$$\hat{p}(x|\theta) = \int dz_p \hat{p}(x|z_p) p(z_p|\theta)$$



[CMS 1511.06170]

Solve it by approximating the integral

- Problem: high-dim. integral over **shower / detector trajectories**

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

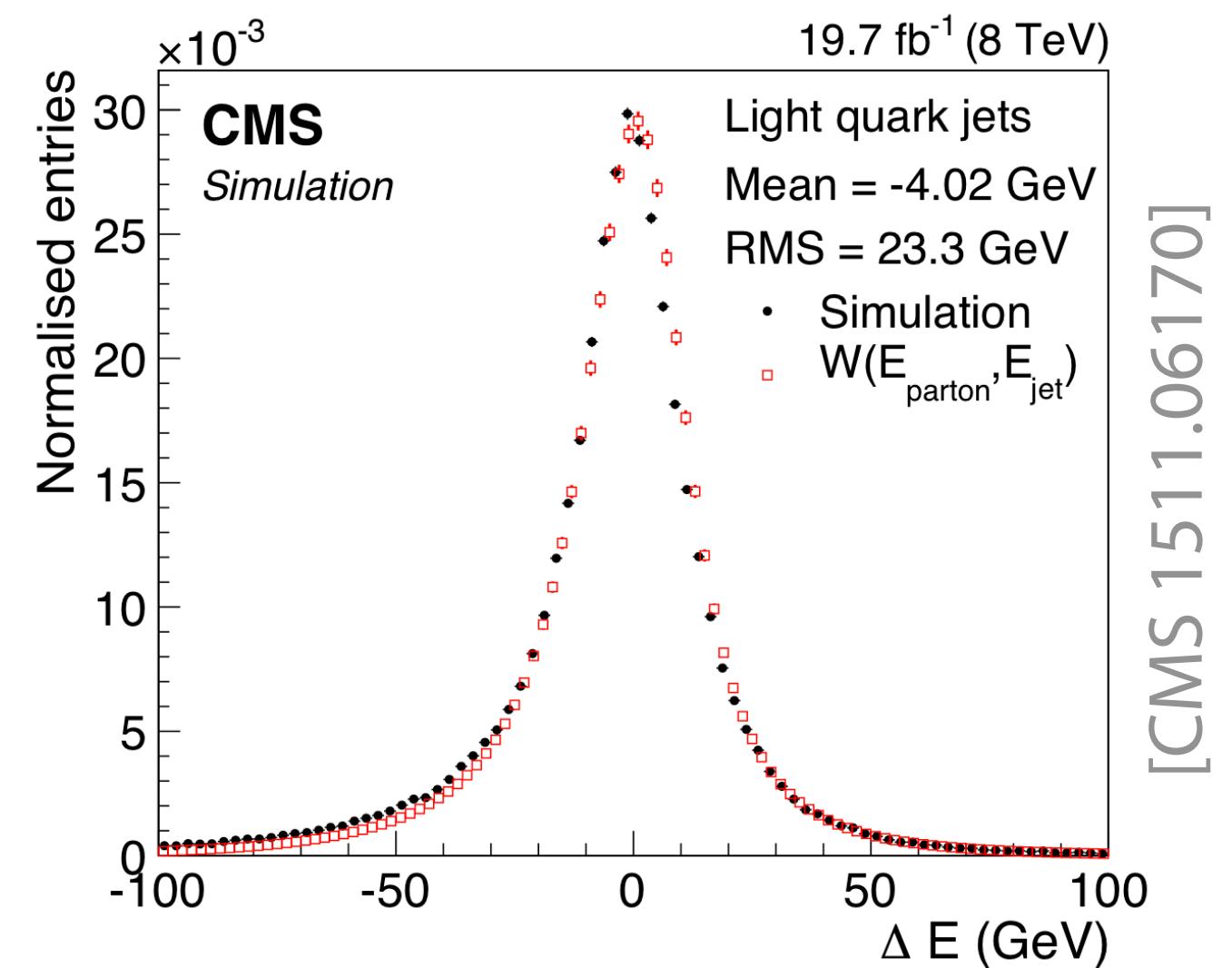
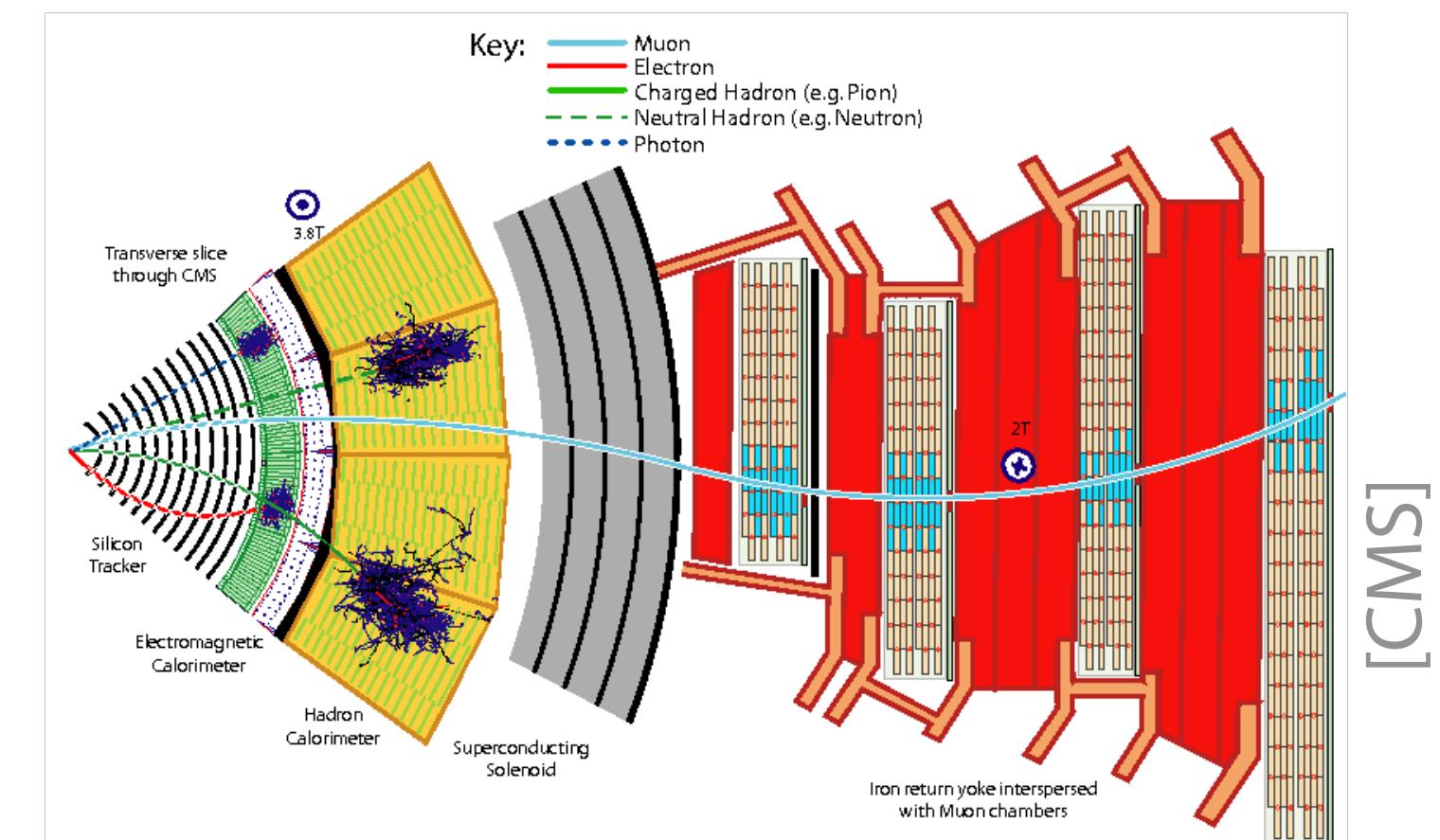
- Matrix Element Method (MEM): [K. Kondo 1988]

- approximate **shower + detector effects** into **transfer function** $\hat{p}(x|z_p)$
- explicitly calculate remaining integral

$$\hat{p}(x|\theta) = \int dz_p \hat{p}(x|z_p) p(z_p|\theta)$$

⇒ Uses matrix-element information, no summary statistics necessary, but:

- ad-hoc transfer functions (what about extra radiation?)
- evaluation still requires calculating an expensive integral



[CMS 1511.06170]

Solve it with machine learning classifiers

[K. Cranmer, J. Pavez, G. Louppe 1506.02169]

Solve it with machine learning classifiers

[K. Cranmer, J. Pavez, G. Louppe 1506.02169]

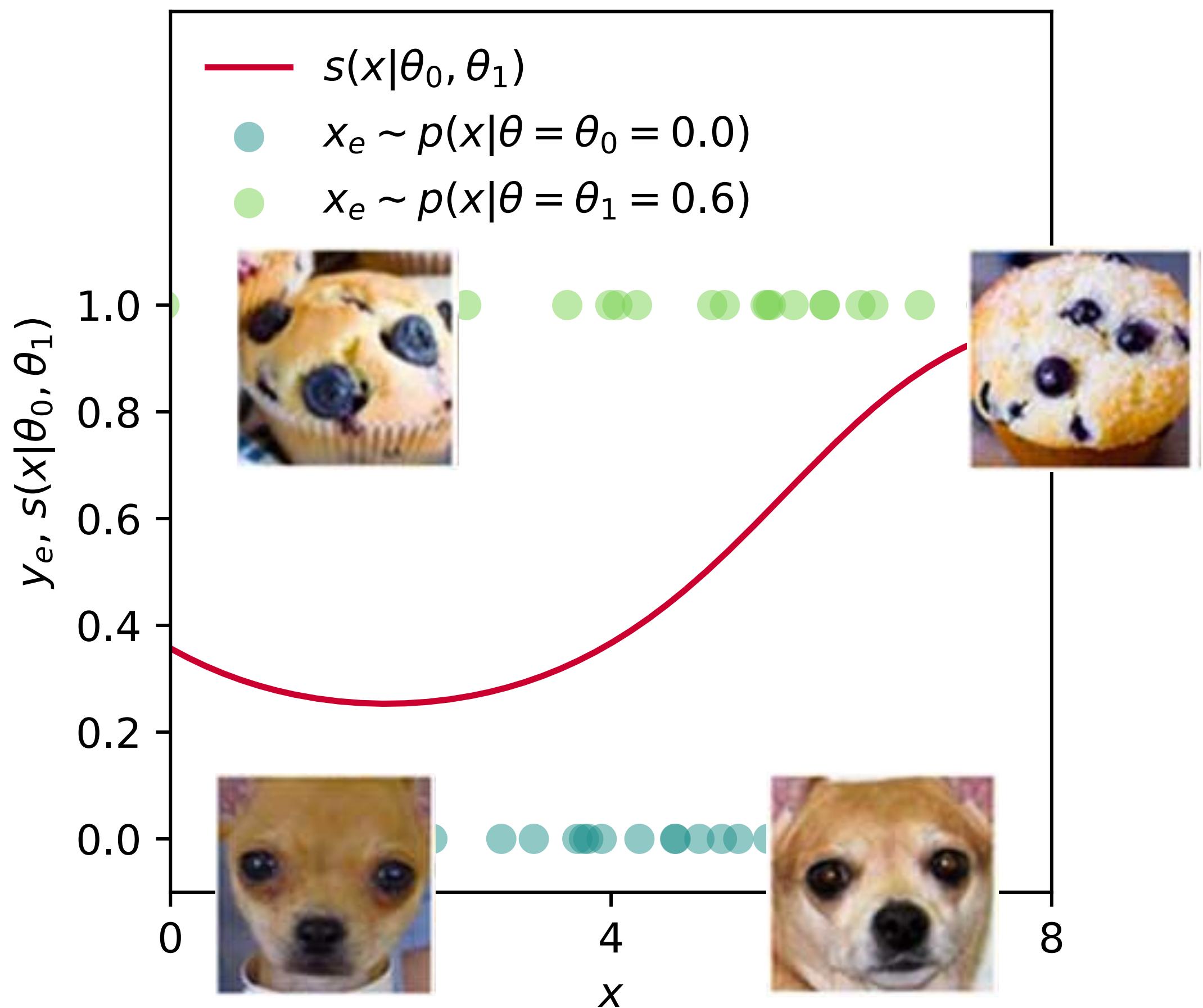


[M. Yao, idea for analogy: K. Cranmer]

Solve it with machine learning classifiers

[K. Cranmer, J. Pavez, G. Louppe 1506.02169]

- Train neural network (BDT, ...) to tell $x \sim p(x|\theta_0)$ from $x \sim p(x|\theta_1)$
- Classifier output $\hat{s}(x)$ is closer to 0 for θ_0 -like events (closer to 1 for θ_1 -like events)



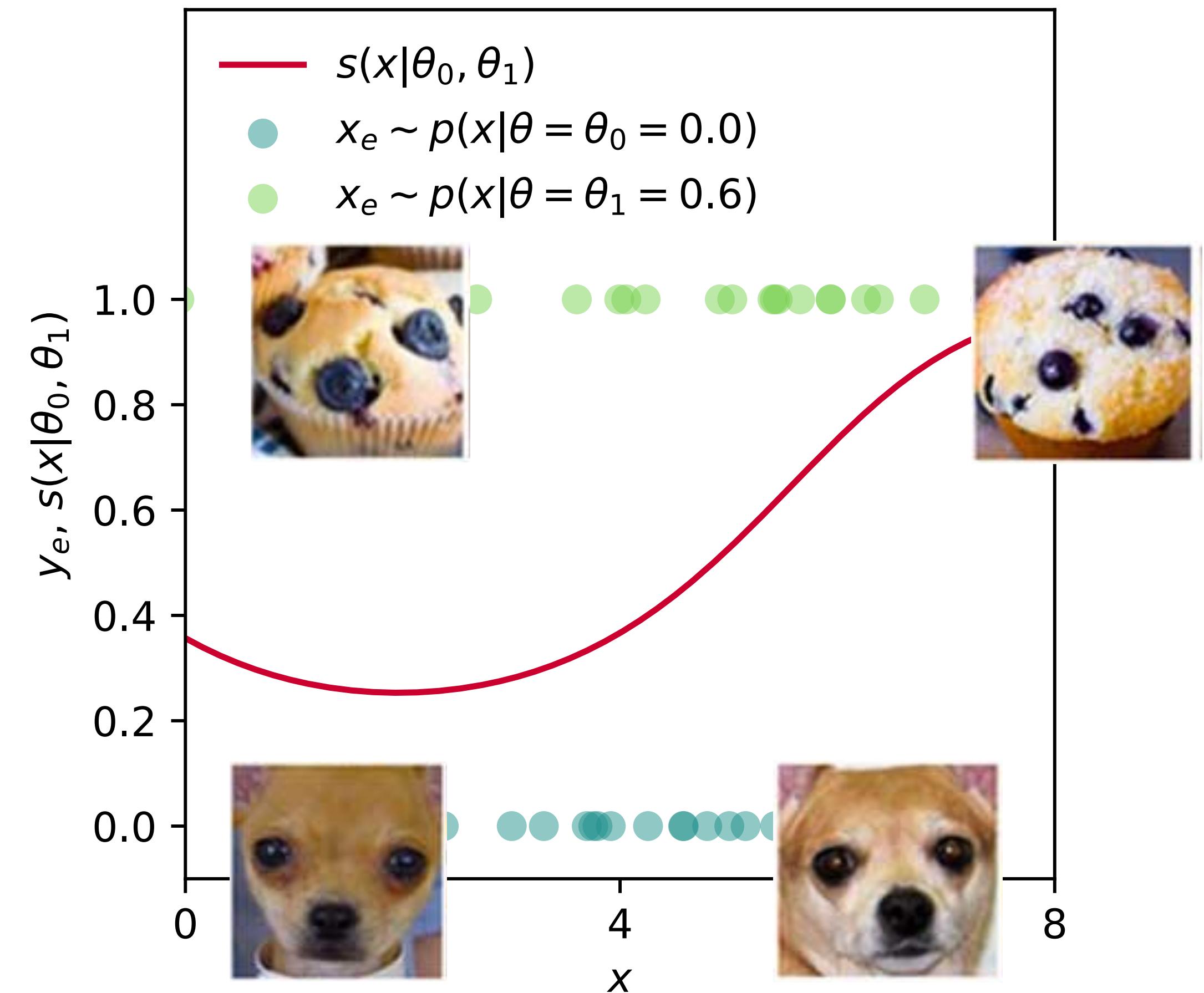
Solve it with machine learning classifiers

[K. Cranmer, J. Pavez, G. Louppe 1506.02169]

- Train neural network (BDT, ...) to tell $x \sim p(x|\theta_0)$ from $x \sim p(x|\theta_1)$
- Classifier output $\hat{s}(x)$ is closer to 0 for θ_0 -like events (closer to 1 for θ_1 -like events)
- CARL: Transform classifier output function $\hat{s}(x)$ into estimator for the likelihood ratio

$$r(x) \equiv p(x|\theta_0)/p(x|\theta_1)$$

(calibrate estimator with histograms of NN output)



⇒ No summary statistics necessary, very fast evaluation... but may require large training samples

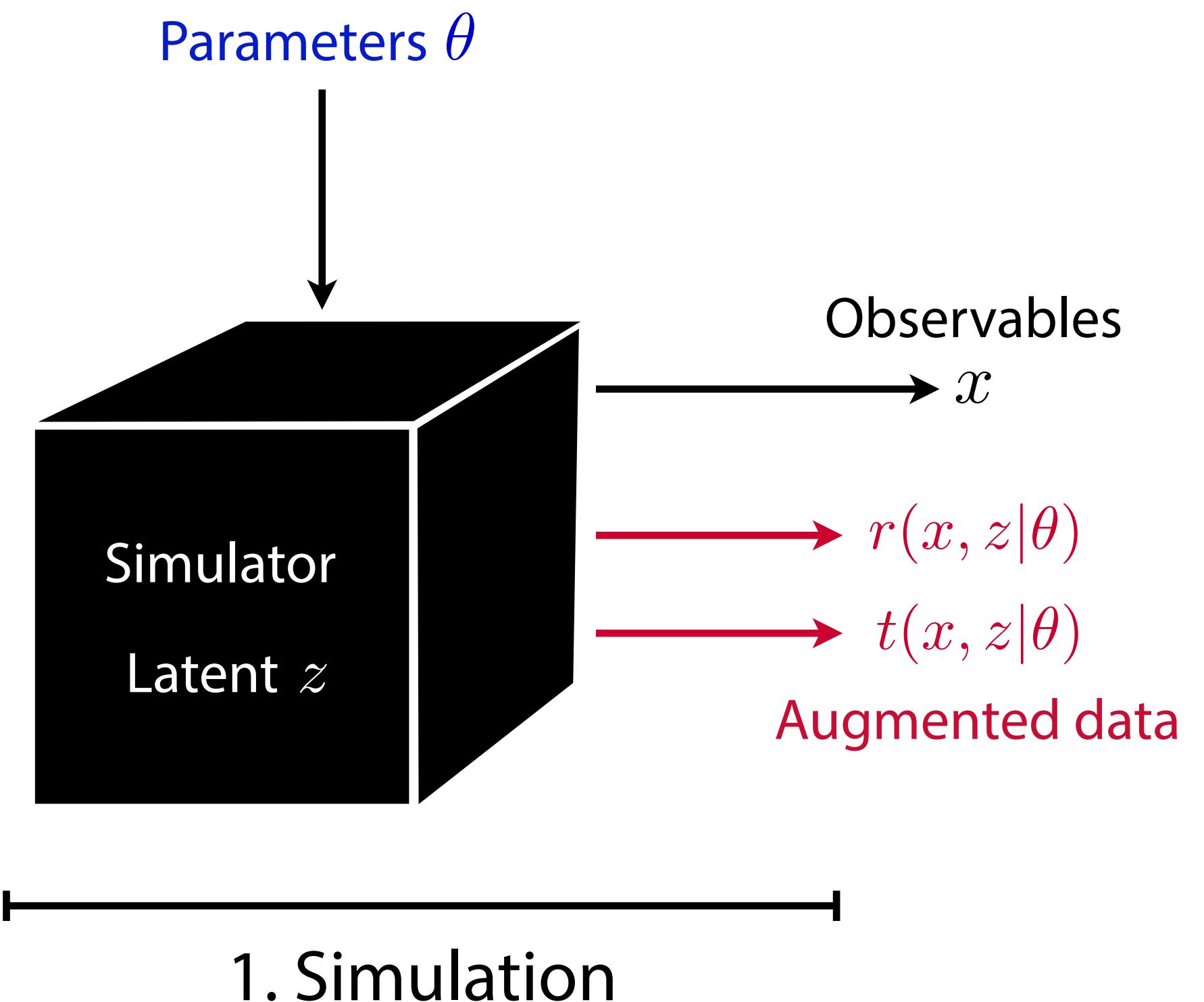
What if we could estimate the likelihood...

- for high-dimensional observables, including correlations?
like MEM: no need to pick summary statistics
- including state-of-the-art shower and detector models?
allowing for extra radiation, no need for transfer functions
- in microseconds?
amortized inference: train once, then always evaluate fast
- requiring less training examples than established machine learning methods?
using matrix element information: “ML version of MEM”

A new approach: “Mining gold” from the simulator

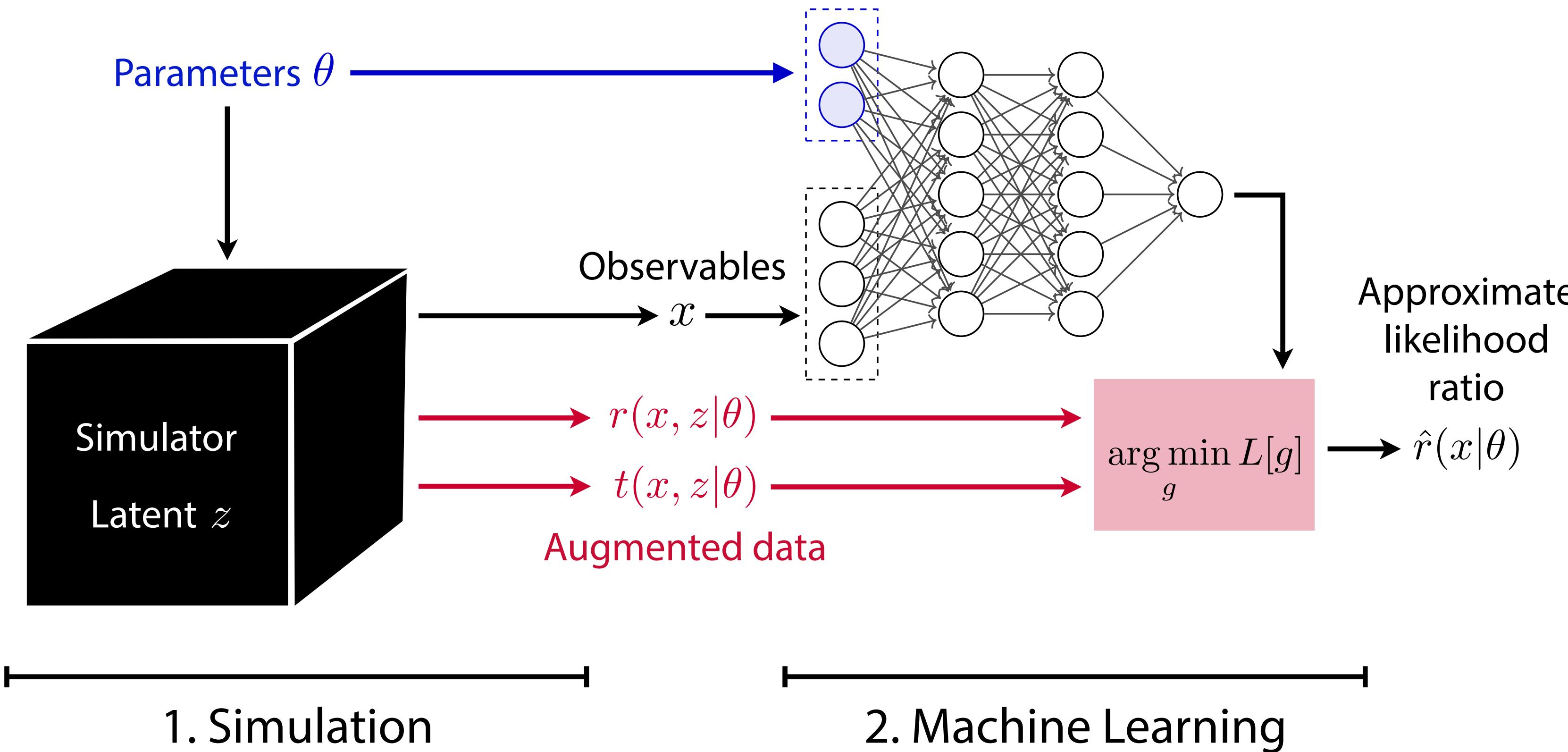
[JB, K. Cranmer, G. Louppe, J. Pavez 1805.00013, 1805.00020, 1805.12244]

Bird's-eye view



“Mining gold”: Extract additional information from simulator

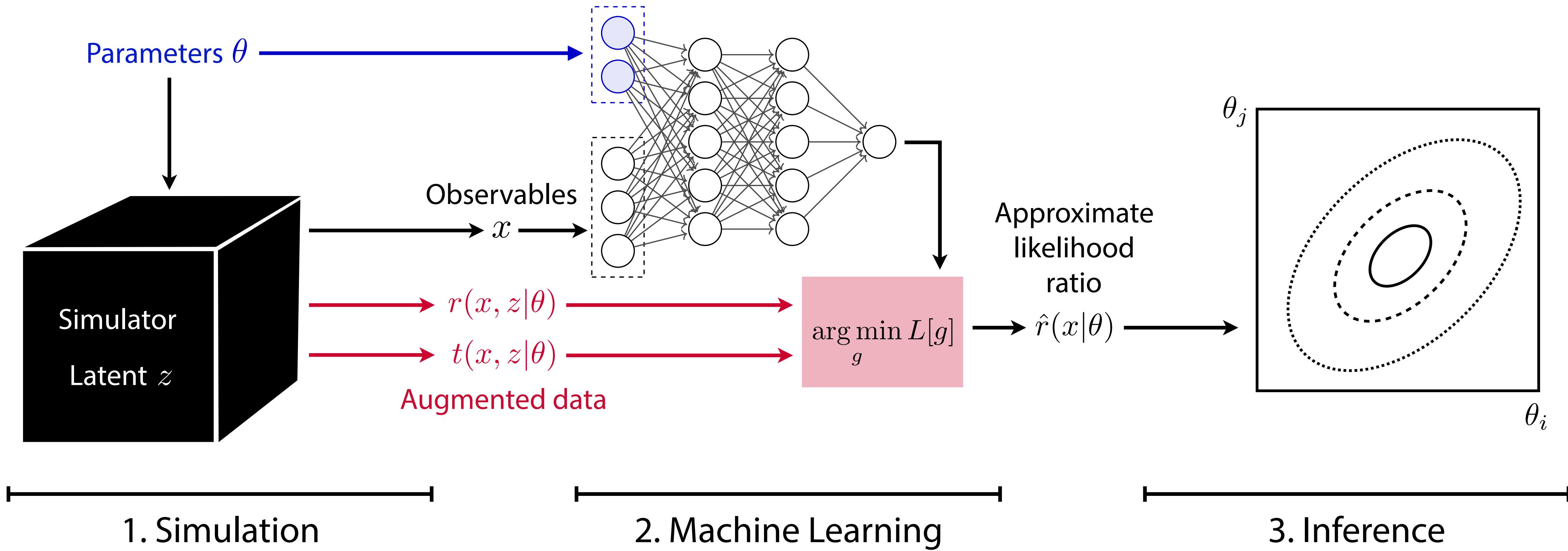
Bird's-eye view



“Mining gold”: Extract additional information from simulator

Use this information to train estimator for likelihood ratio

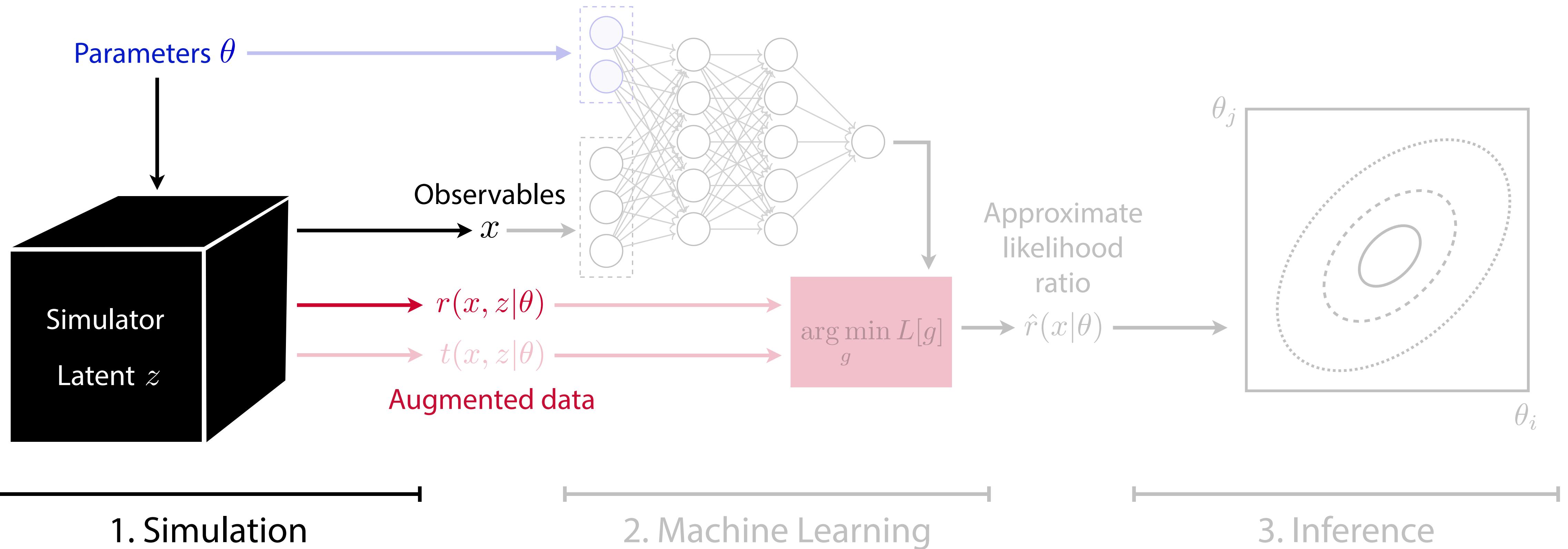
Bird's-eye view



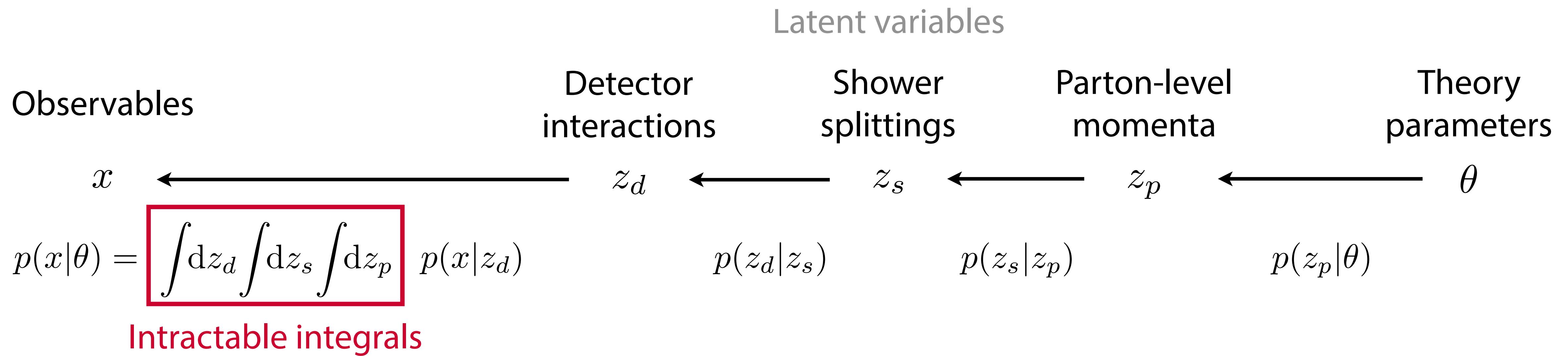
“Mining gold”: Extract additional information from simulator

Use this information to train estimator for likelihood ratio

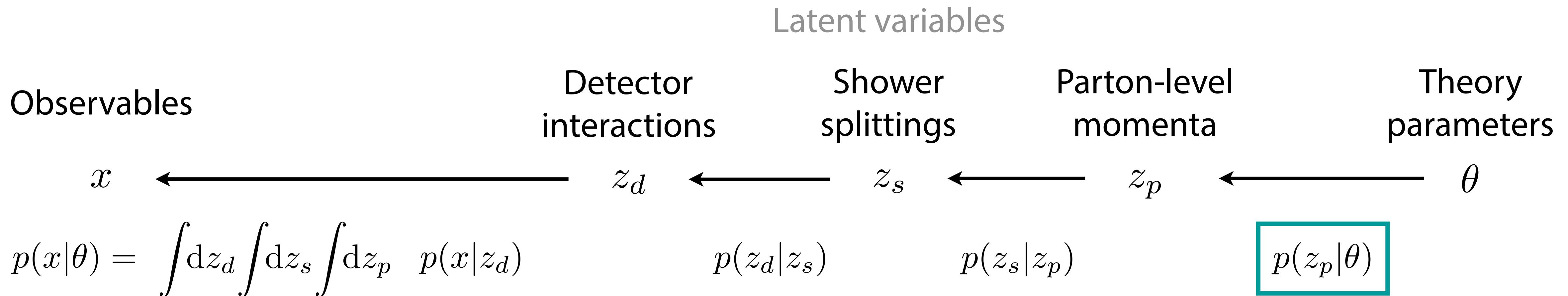
Limit setting with standard hypothesis tests



Mining gold from the simulator



Mining gold from the simulator

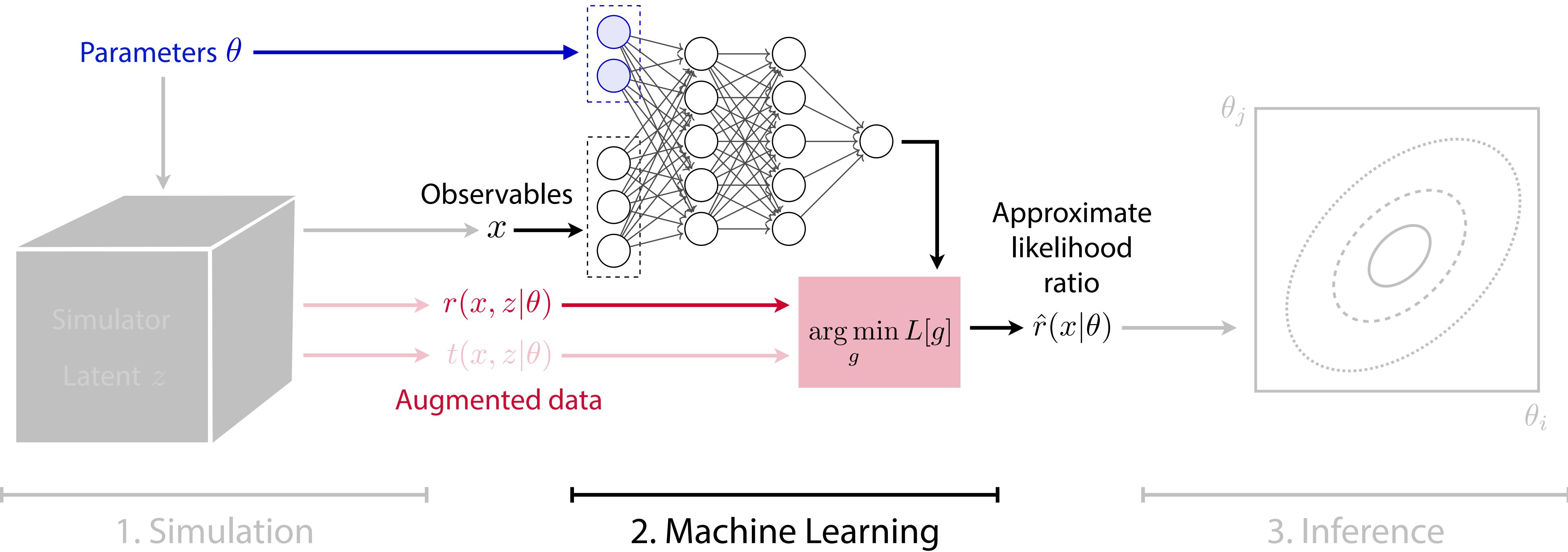


Parton-level likelihood
is given by matrix element
and can be evaluated!

⇒ For each simulated event, we can calculate the **joint likelihood ratio** which depends on the specific evolution of the simulation:

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

$$\frac{p(z_p|\theta_0)}{p(z_p|\theta_1)} \sim \frac{|\mathcal{M}(z_p|\theta_0)|^2}{|\mathcal{M}(z_p|\theta_1)|^2}$$



The value of gold

We can calculate the **joint likelihood ratio**

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



(“How much more likely is this simulated event, including all intermediate states, for θ_0 compared to θ_1 ? ”)

We want the **likelihood ratio function**

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

(“How much more likely is the observation x for θ_0 compared to θ_1 ? ”)

The value of gold

We can calculate the joint likelihood ratio

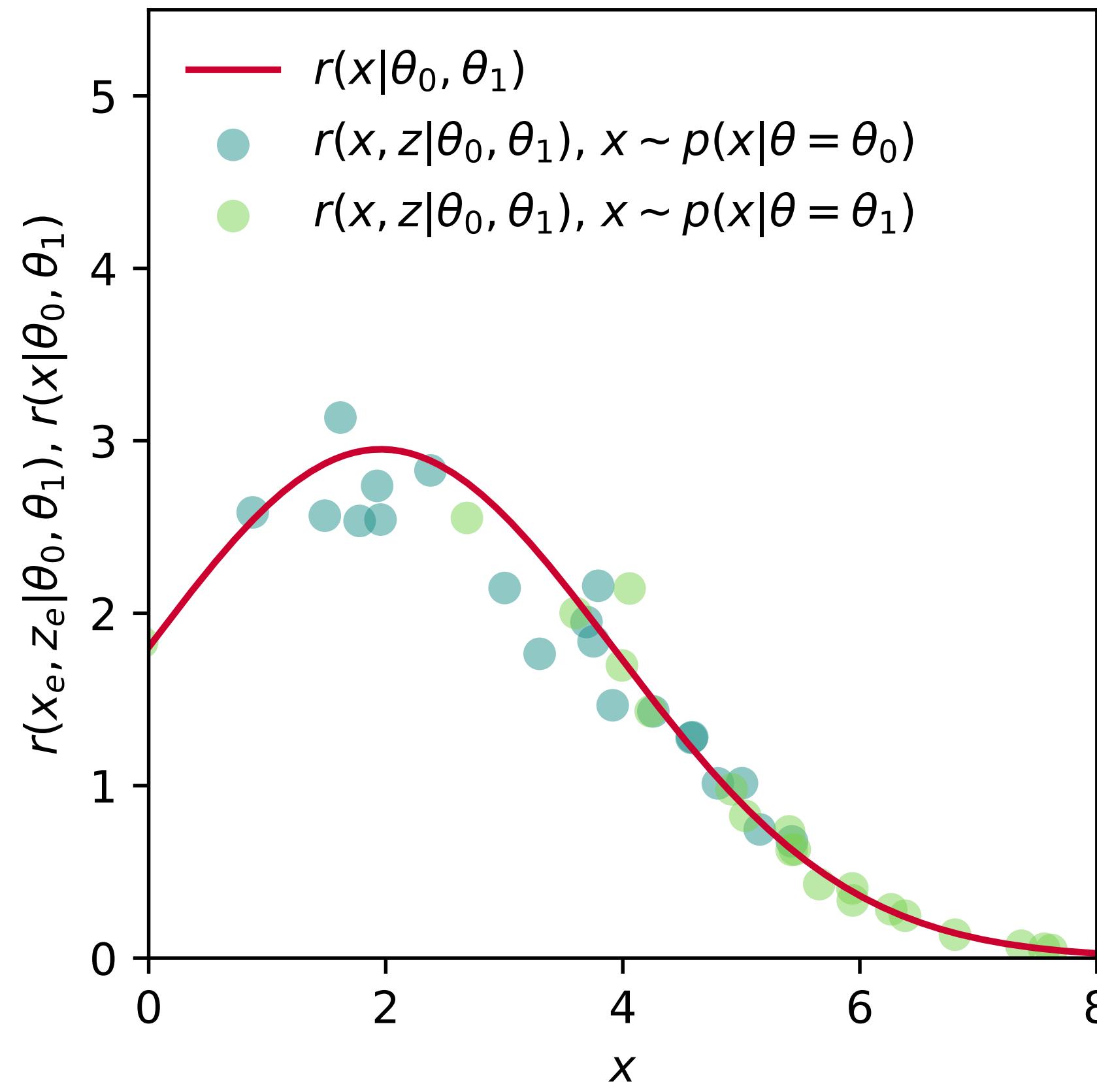
$$r(x, z | \theta_0, \theta_1) \equiv \frac{p(x, z_d, z_s, z_p | \theta_0)}{p(x, z_d, z_s, z_p | \theta_1)}$$



$r(x, z | \theta_0, \theta_1)$ are scattered around $r(x | \theta_0, \theta_1)$

We want the likelihood ratio function

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



The value of gold

We can calculate the joint likelihood ratio

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

With $r(x, z|\theta_0, \theta_1)$, we define a functional like

$$L_r[\hat{r}(x|\theta_0, \theta_1)] = \int dx \int dz p(x, z|\theta_1) \left[(\hat{r}(x|\theta_0, \theta_1) - r(x, z|\theta_0, \theta_1))^2 \right].$$

It is minimized by

$$r(x|\theta_0, \theta_1) = \arg \min_{\hat{r}(x|\theta_0, \theta_1)} L_r[\hat{r}(x|\theta_0, \theta_1)]!$$

(And we can sample from $p(x, z|\theta)$ by running the simulator.)

We want the likelihood ratio function

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

Machine learning = applied calculus of variations

So to get a good estimator of the likelihood ratio, we need to minimize a functional numerically:

$$r(x|\theta_0, \theta_1) = \arg \min_{\hat{r}(x|\theta_0, \theta_1)} L_r[\hat{r}(x|\theta_0, \theta_1)]$$

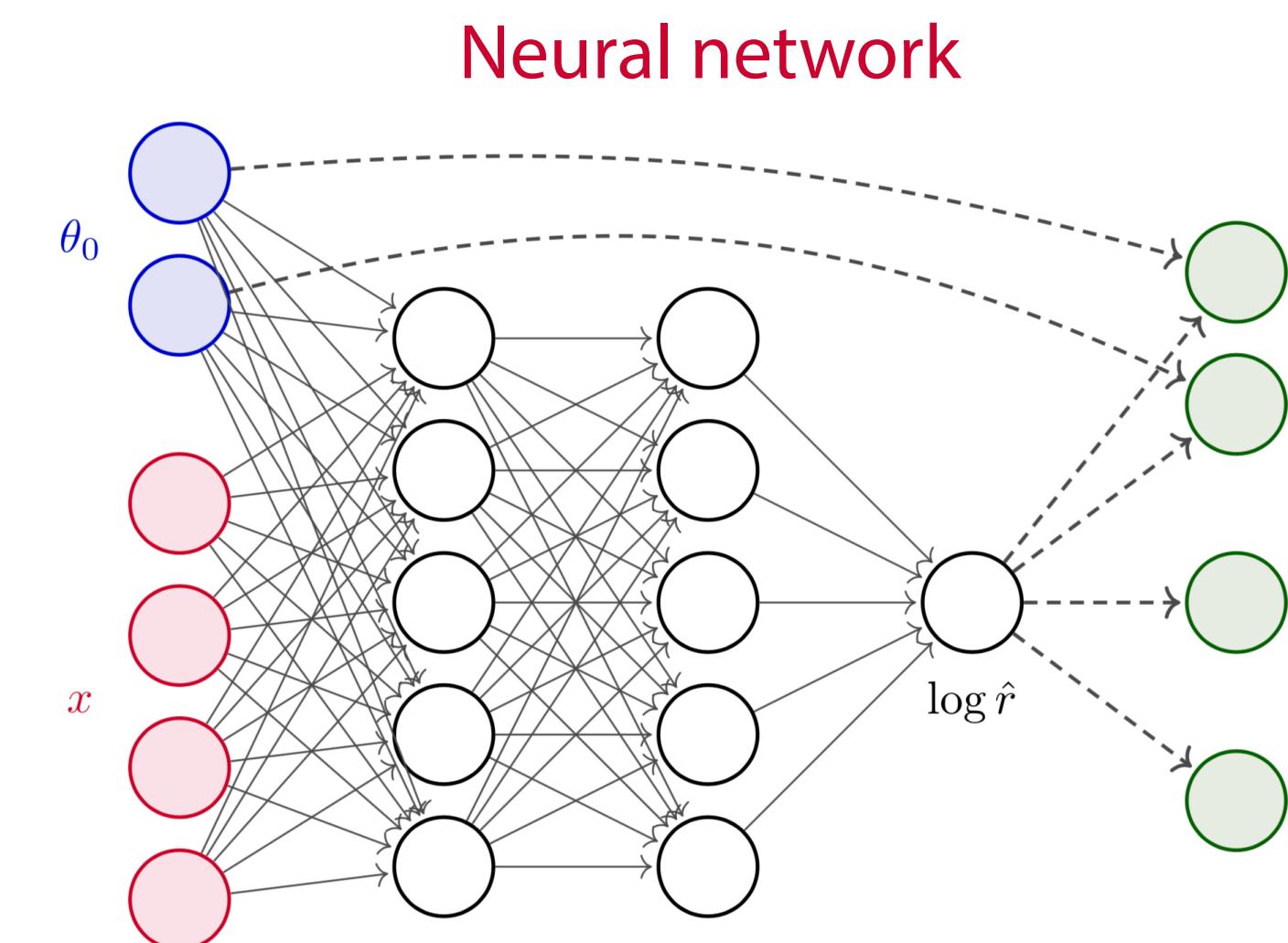
Variational family Extremization Functional with integral

```
graph TD; A[Variational family] --> B[Extremization]; B --> C[Functional with integral];
```

Machine learning = applied calculus of variations

So to get a good estimator of the likelihood ratio, we need to minimize a functional numerically:

This is where machine learning comes in!



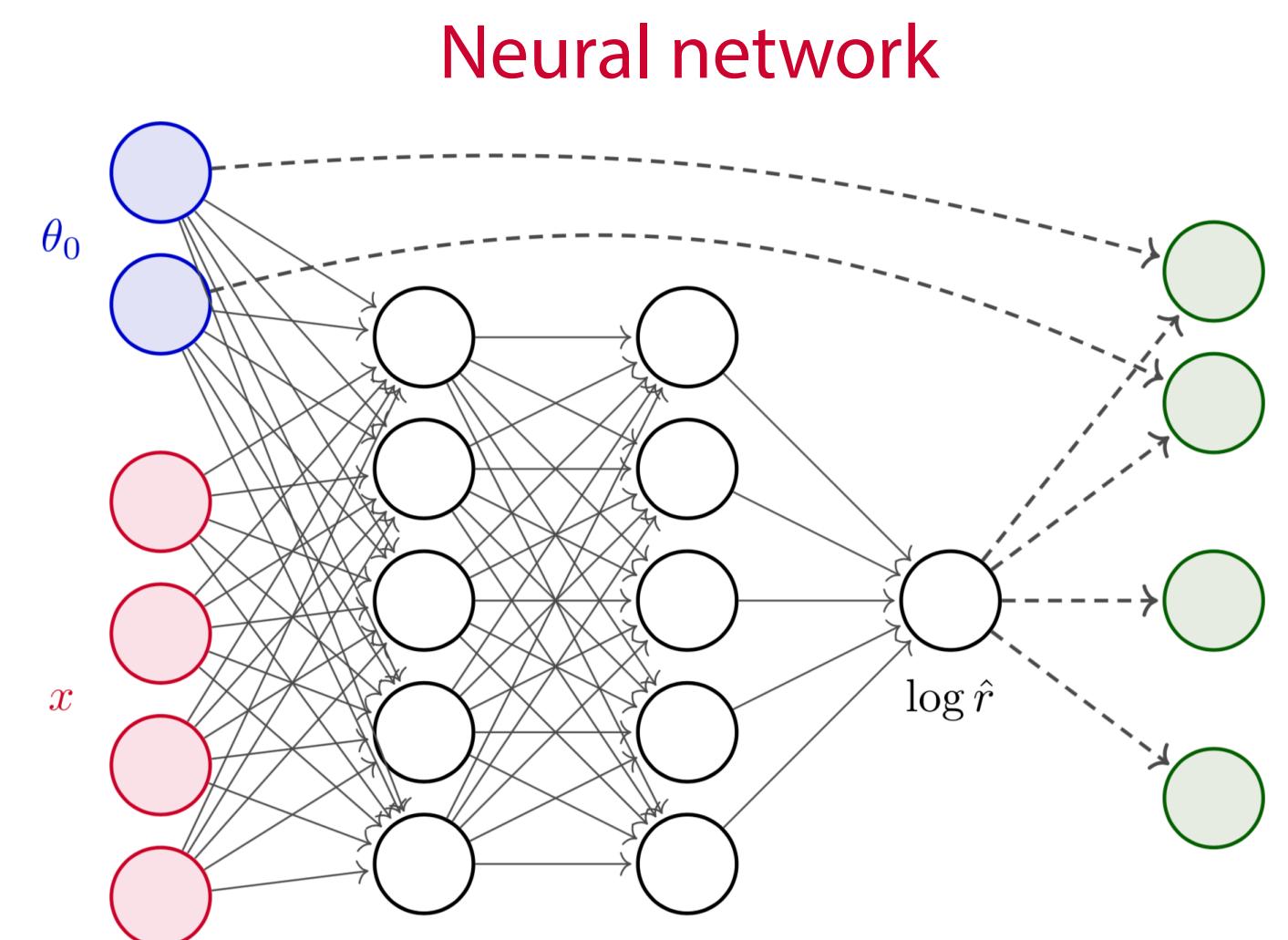
$$r(x|\theta_0, \theta_1) = \arg \min_{\hat{r}(x|\theta_0, \theta_1)} L_r[\hat{r}(x|\theta_0, \theta_1)]$$

Variational family
Extremization
Functional with integral
Loss function with finite sum over samples
Stochastic gradient descent

Machine learning = applied calculus of variations

So to get a good estimator of the likelihood ratio, we need to minimize a functional numerically:

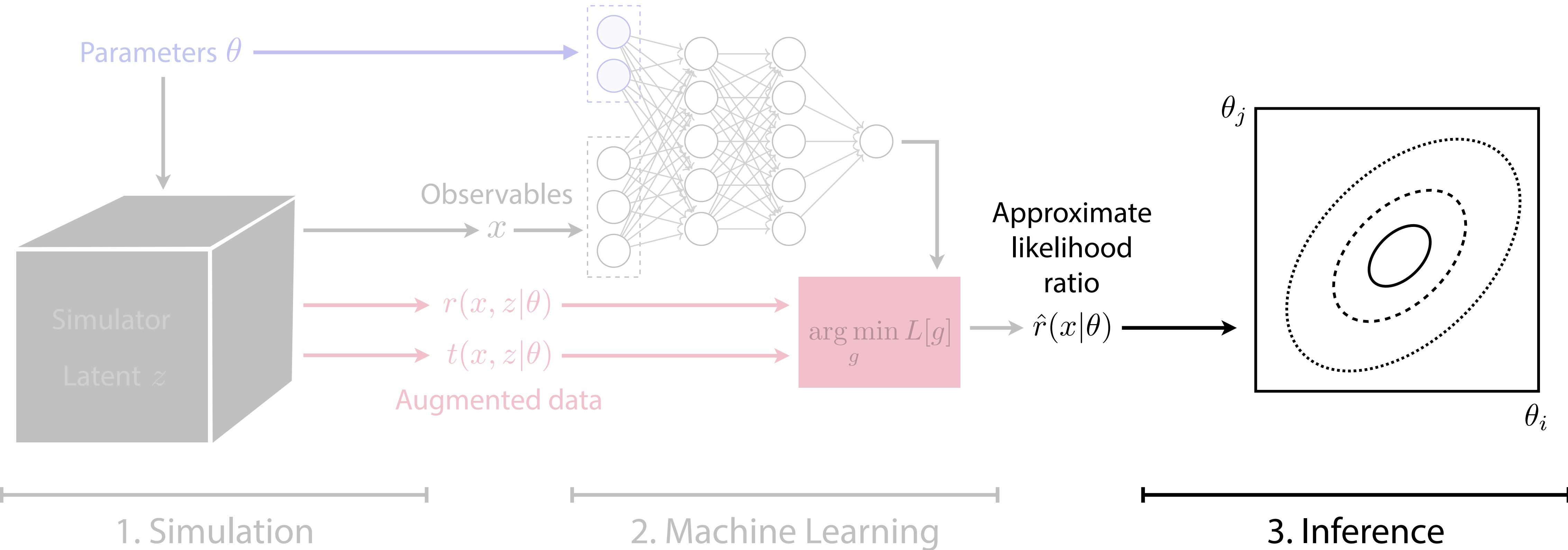
This is where machine learning comes in!

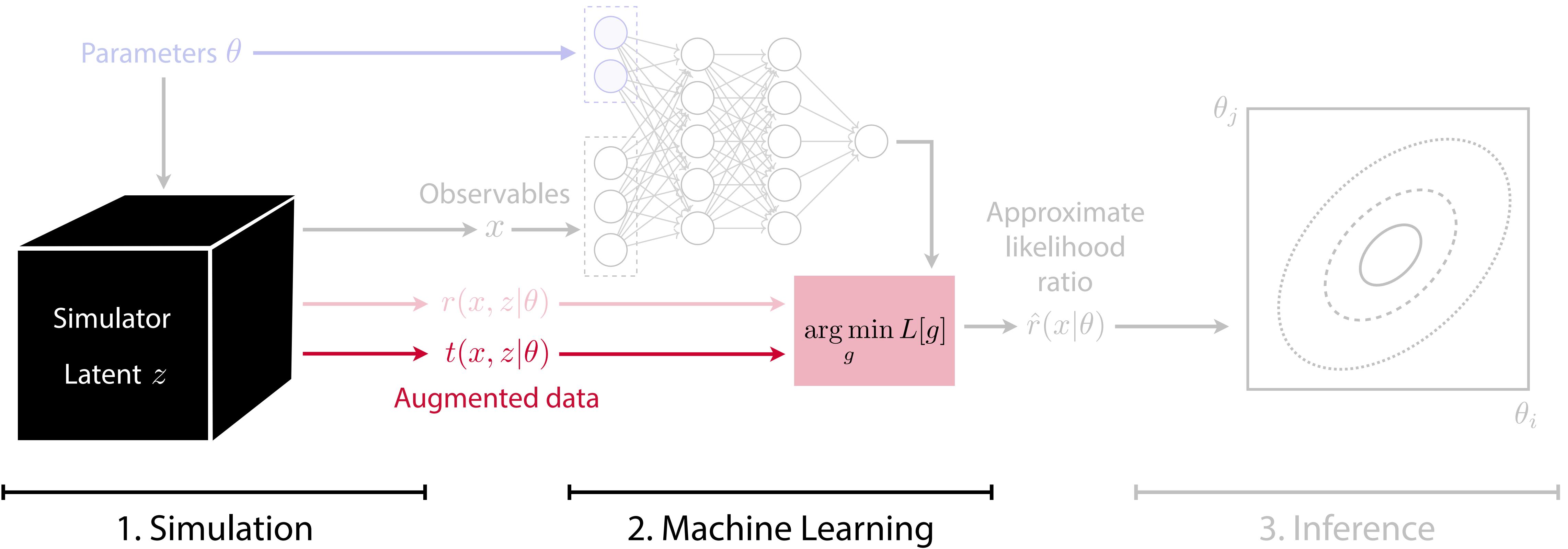


$$r(x|\theta_0, \theta_1) = \arg \min_{\hat{r}(x|\theta_0, \theta_1)} L_r[\hat{r}(x|\theta_0, \theta_1)]$$

Variational family
Extremization
Functional with integral
Loss function with finite sum over samples
Stochastic gradient descent

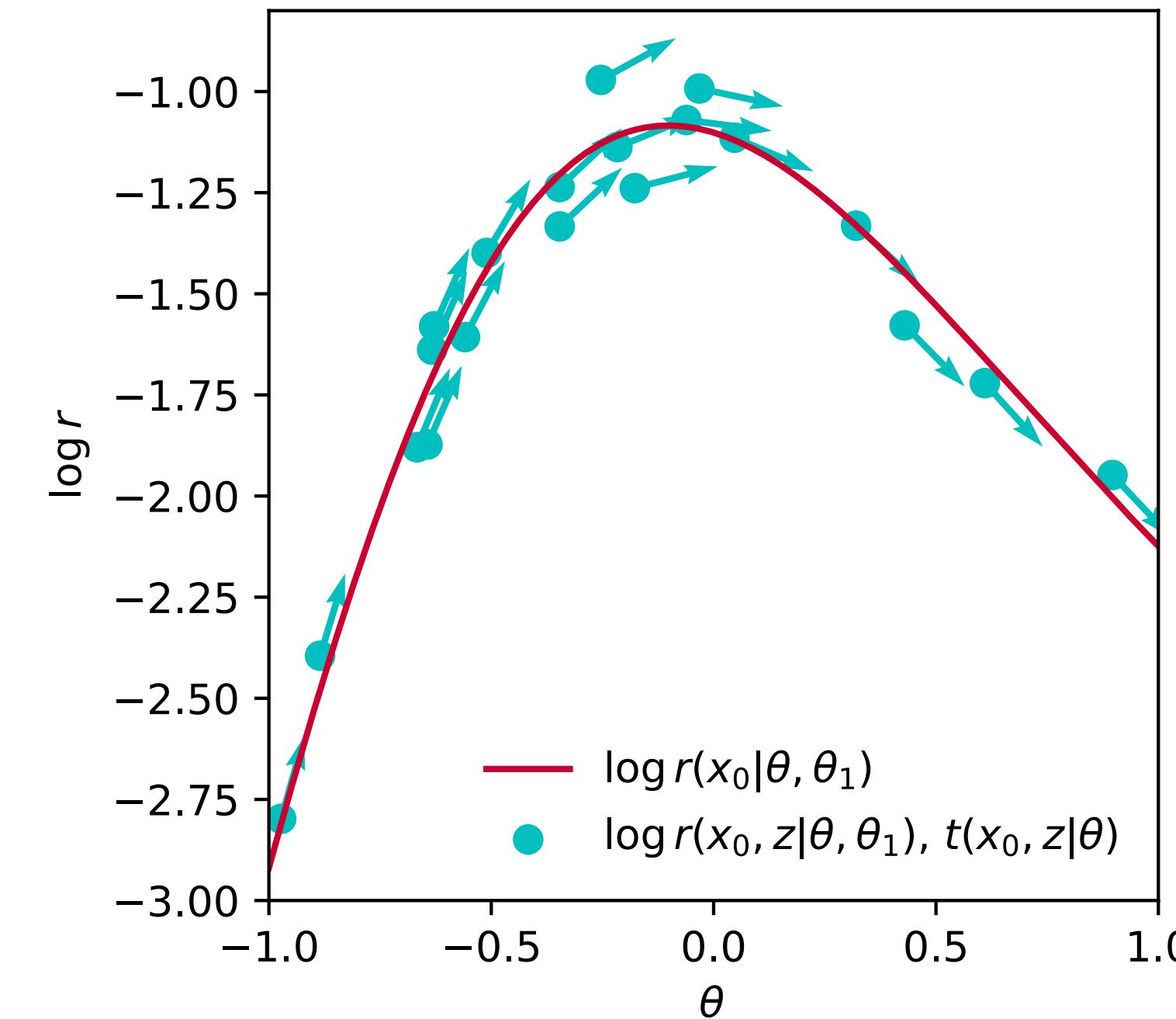
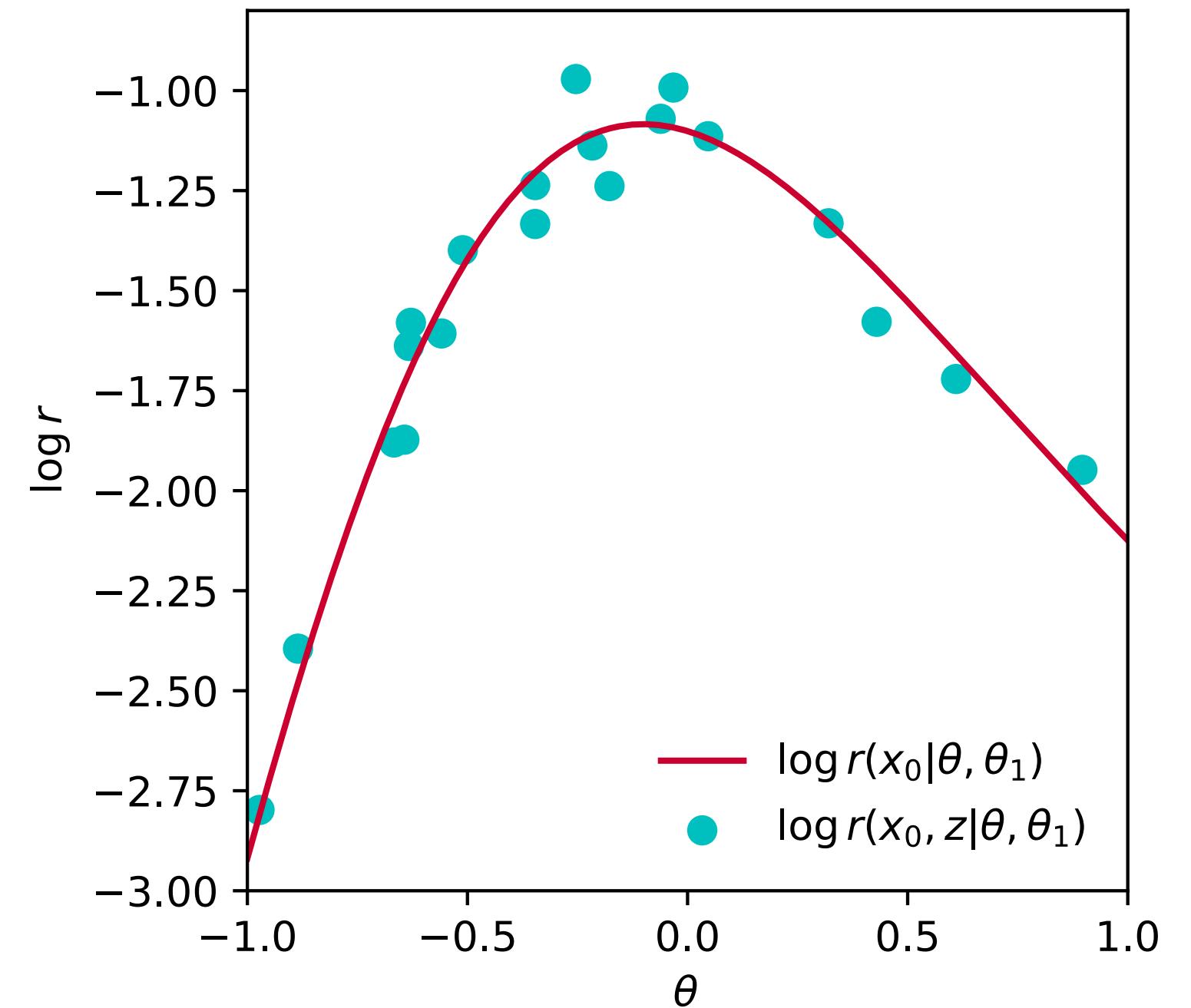
A sufficiently expressive neural network efficiently trained in this way with enough data will learn the likelihood ratio function $r(x|\theta_0, \theta_1)$!





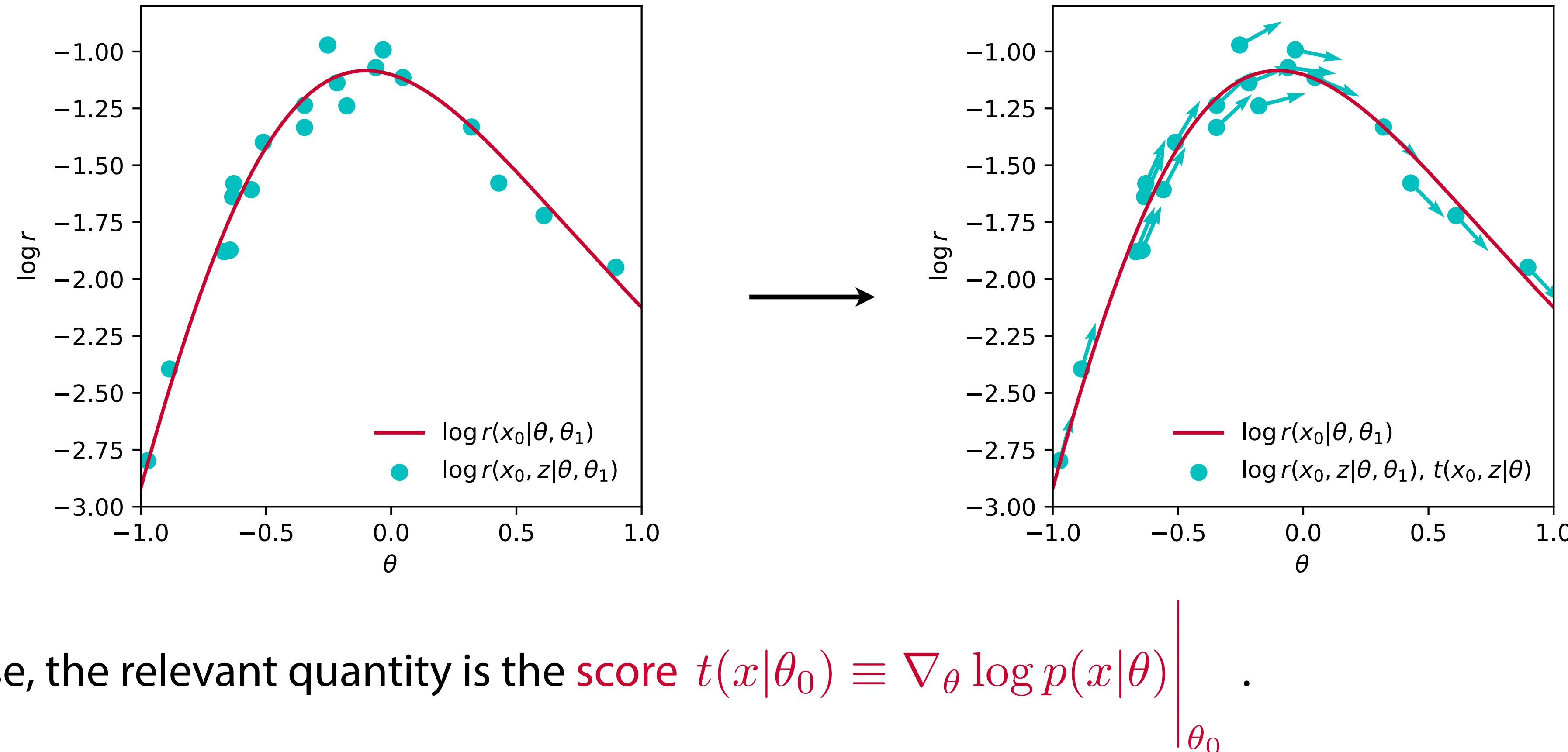
One more piece: the score

- Knowing derivative often helps fitting:



One more piece: the score

- Knowing derivative often helps fitting:



- In our case, the relevant quantity is the **score** $t(x|\theta_0) \equiv \nabla_{\theta} \log p(x|\theta) \Big|_{\theta_0}$.
- The score itself is intractable. But...

Learning the score

Similar to the joint likelihood ratio, from the simulator we can extract the **joint score**

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



We want the **score**

$$t(x|\theta_0) \equiv \nabla_{\theta} \log p(x|\theta) \Big|_{\theta_0}$$

Learning the score

Similar to the joint likelihood ratio, from the simulator we can extract the **joint score**

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



We want the **score**

$$t(x|\theta_0) \equiv \nabla_{\theta} \log p(x|\theta) \Big|_{\theta_0}$$

Given $t(x, z|\theta_0)$,
we define the functional

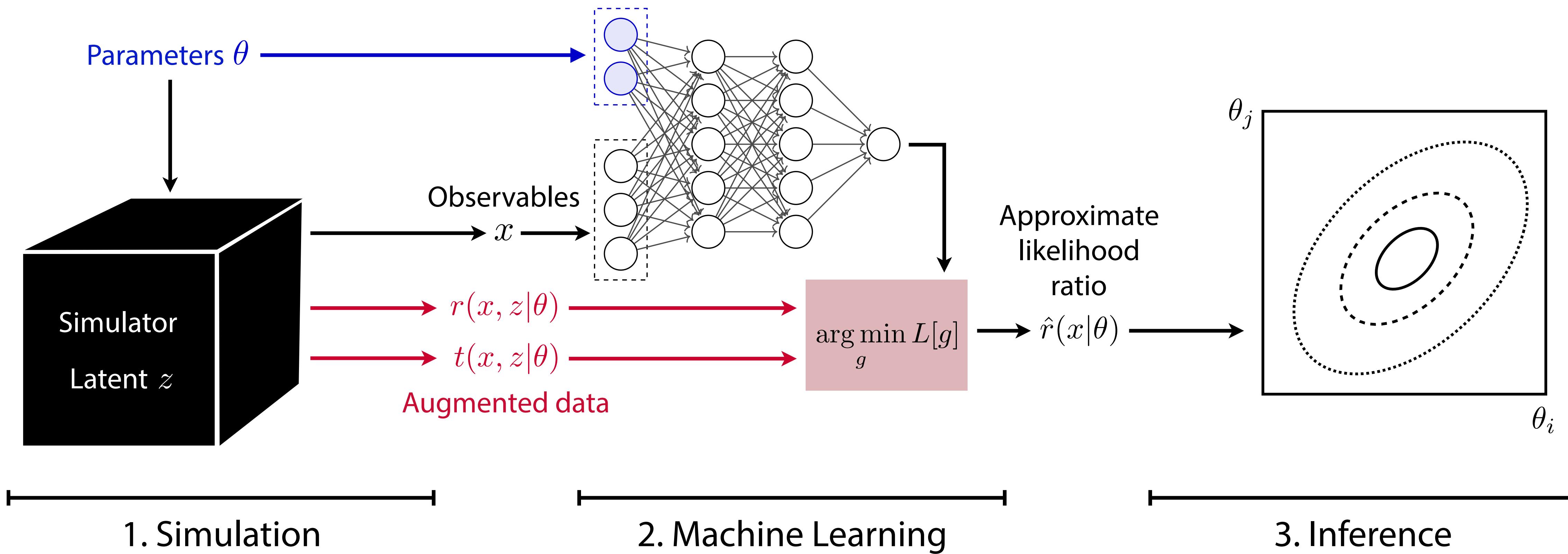
$$L_t[\hat{t}(x|\theta_0)] = \int dx \int dz \ p(x, z|\theta_0) \left[(\hat{t}(x|\theta_0) - t(x, z|\theta_0))^2 \right].$$

One can show it is minimized by

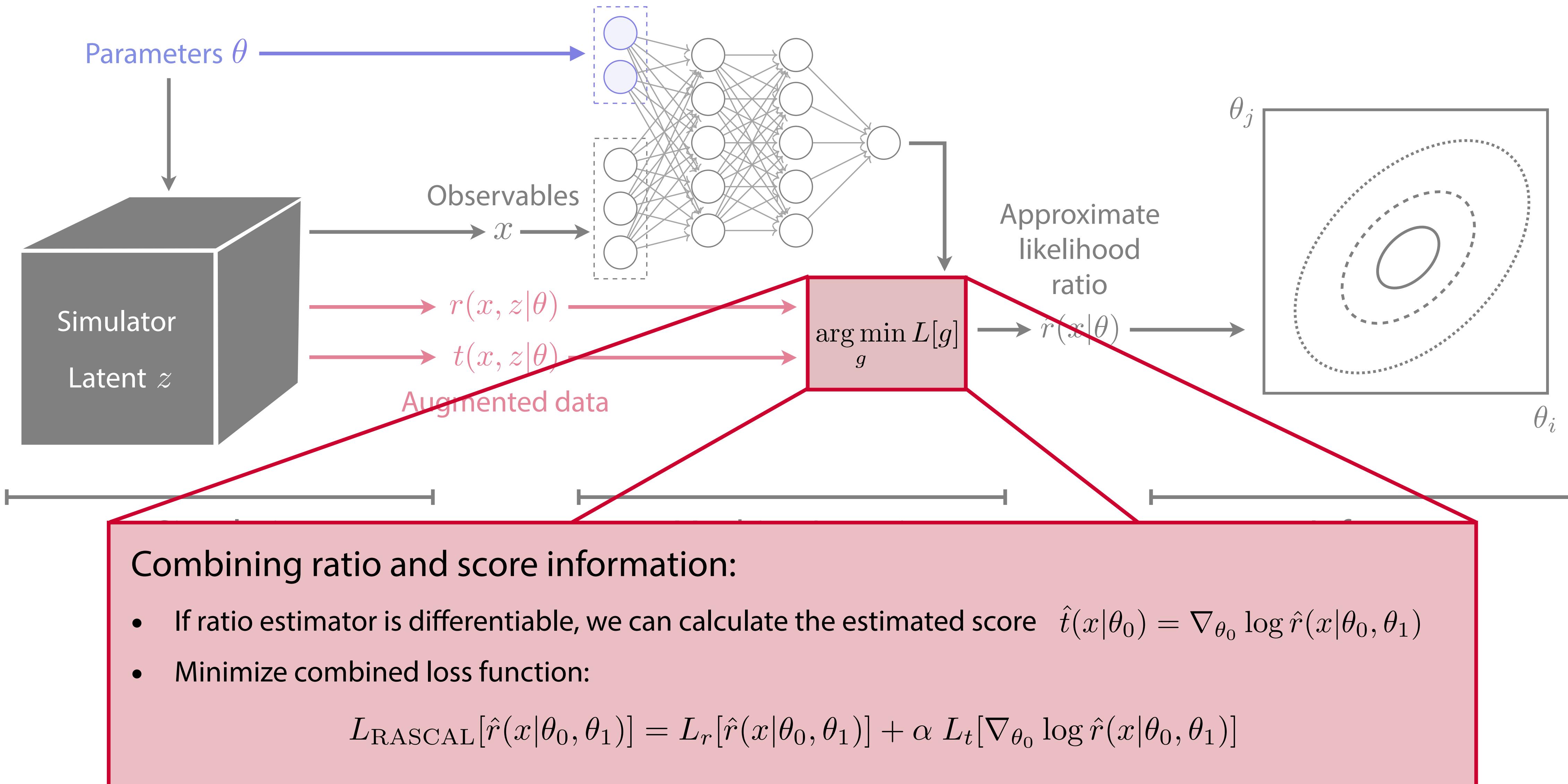
$$t(x|\theta_0) = \arg \min_{\hat{t}(x|\theta_0)} L_t[\hat{t}(x|\theta_0)].$$

Again, we implement this minimization through machine learning.

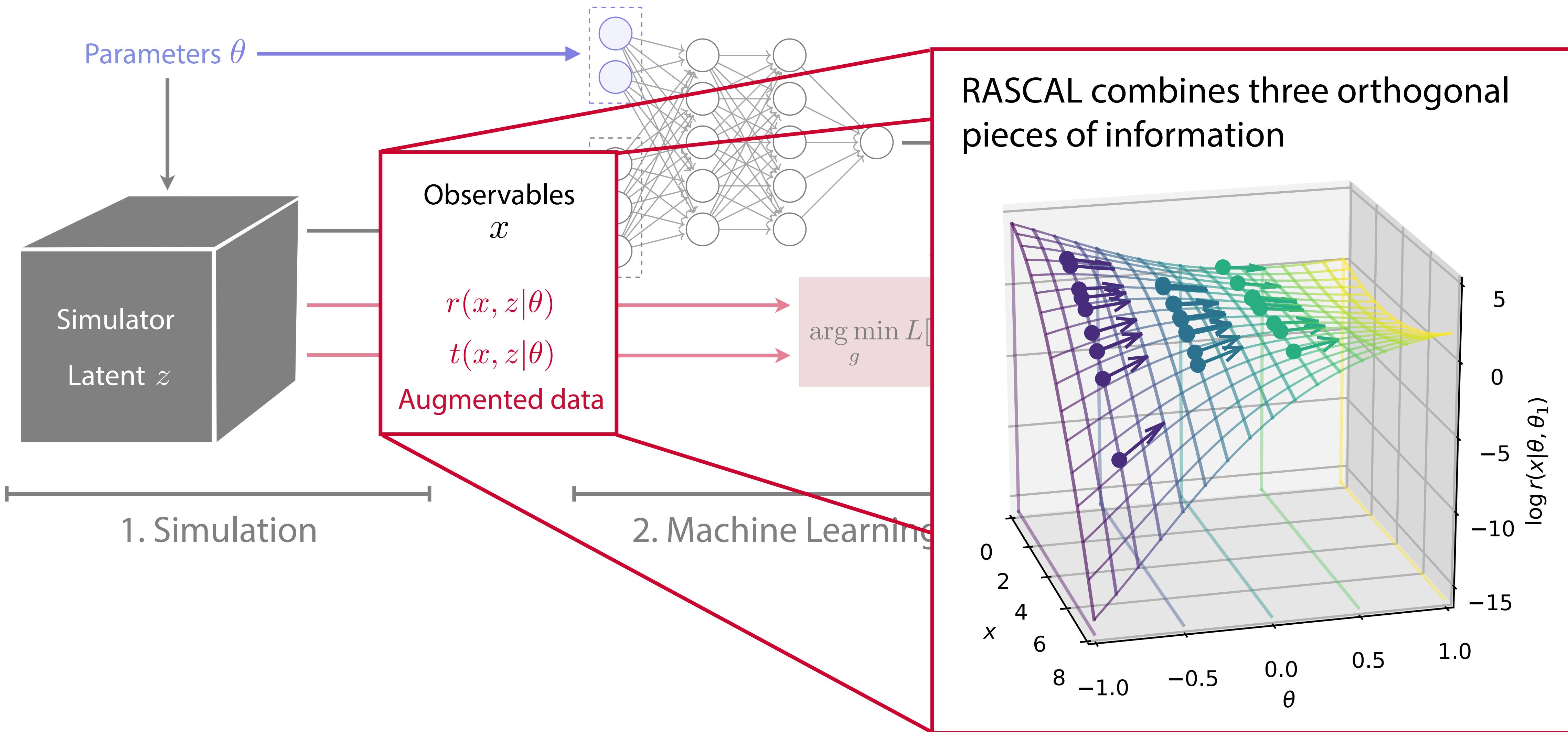
Putting the pieces together: RASCAL (Ratio and score approximate likelihood ratio)



Putting the pieces together: RASCAL (Ratio and score approximate likelihood ratio)



Putting the pieces together: RASCAL (Ratio and score approximate likelihood ratio)



Learning optimal observables

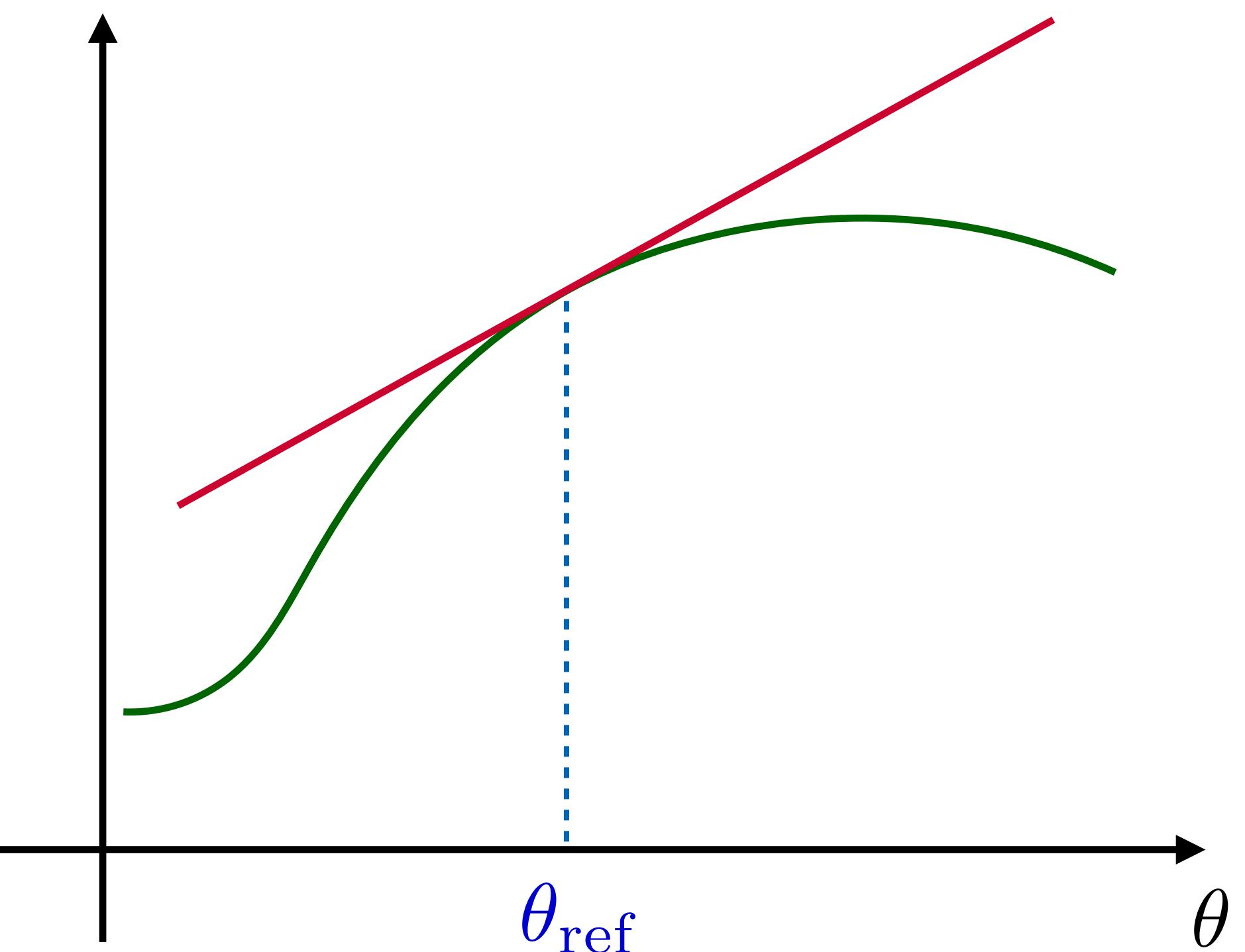
[JB, K. Cranmer, G. Louppe, J. Pavez 1805.00013, 1805.00020, 1805.12244]

The local model

[see also J. Alsing, B. Wandelt 1712.00012; J. Alsing, B. Wandelt, S. Freeney 1801.01497;
P. de Castro, T. Dorigo 1806.04743; J. Alsing, B. Wandelt 1903.01473]

Taylor expansion of $\log p(x|\theta)$ around θ_{ref} :

$$\begin{aligned}\log p(x|\theta) &= \log p(x|\theta_{\text{ref}}) \\ &+ \underbrace{\nabla_{\theta} \log p(x|\theta) \Big|_{\theta_{\text{ref}}} \cdot (\theta - \theta_{\text{ref}})}_{\equiv t(x|\theta_{\text{ref}})} \\ &+ \mathcal{O}((\theta - \theta_{\text{ref}})^2)\end{aligned}$$



The local model

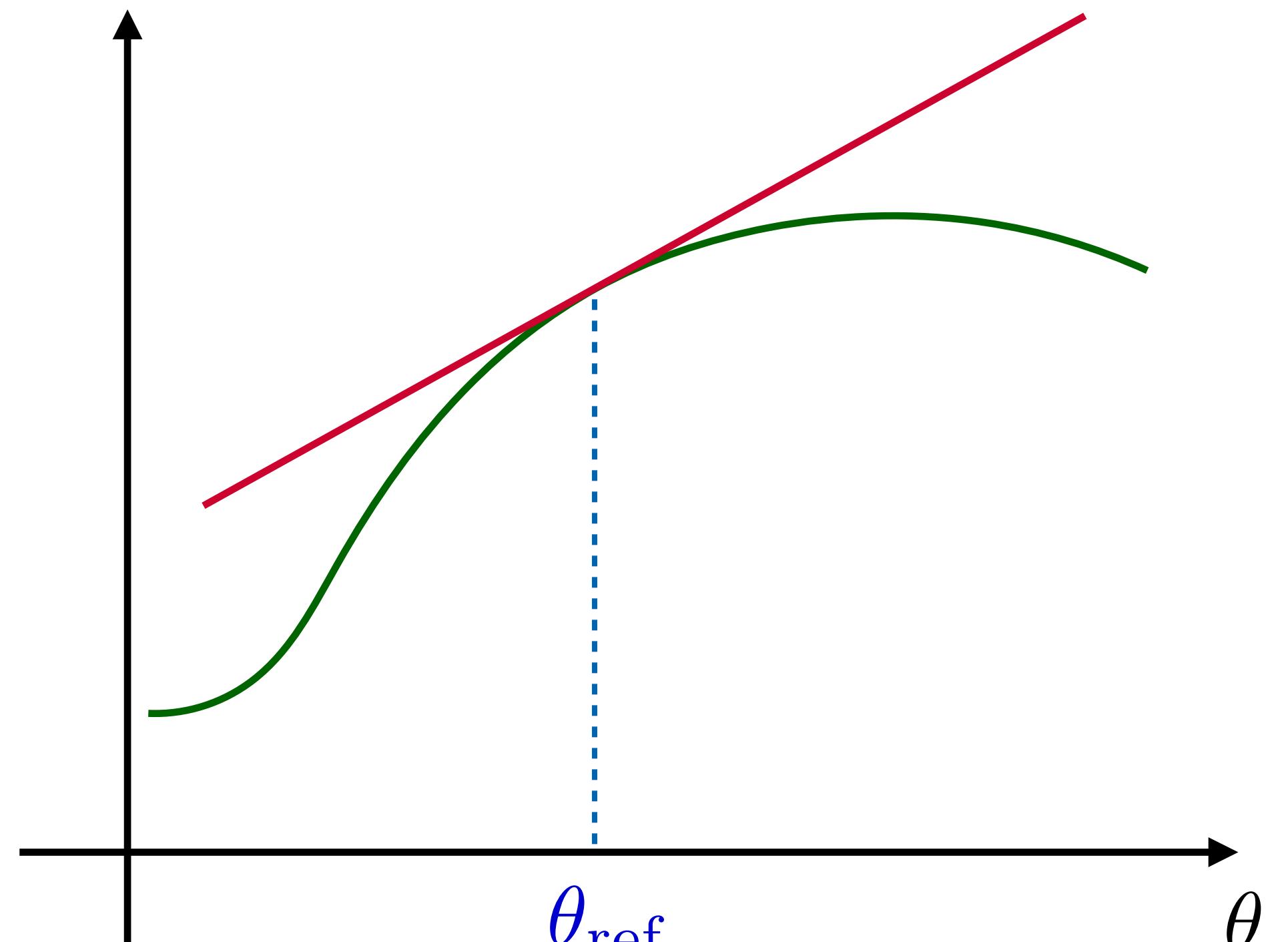
[see also J. Alsing, B. Wandelt 1712.00012; J. Alsing, B. Wandelt, S. Freeney 1801.01497;
P. de Castro, T. Dorigo 1806.04743; J. Alsing, B. Wandelt 1903.01473]

Taylor expansion of $\log p(x|\theta)$ around θ_{ref} :

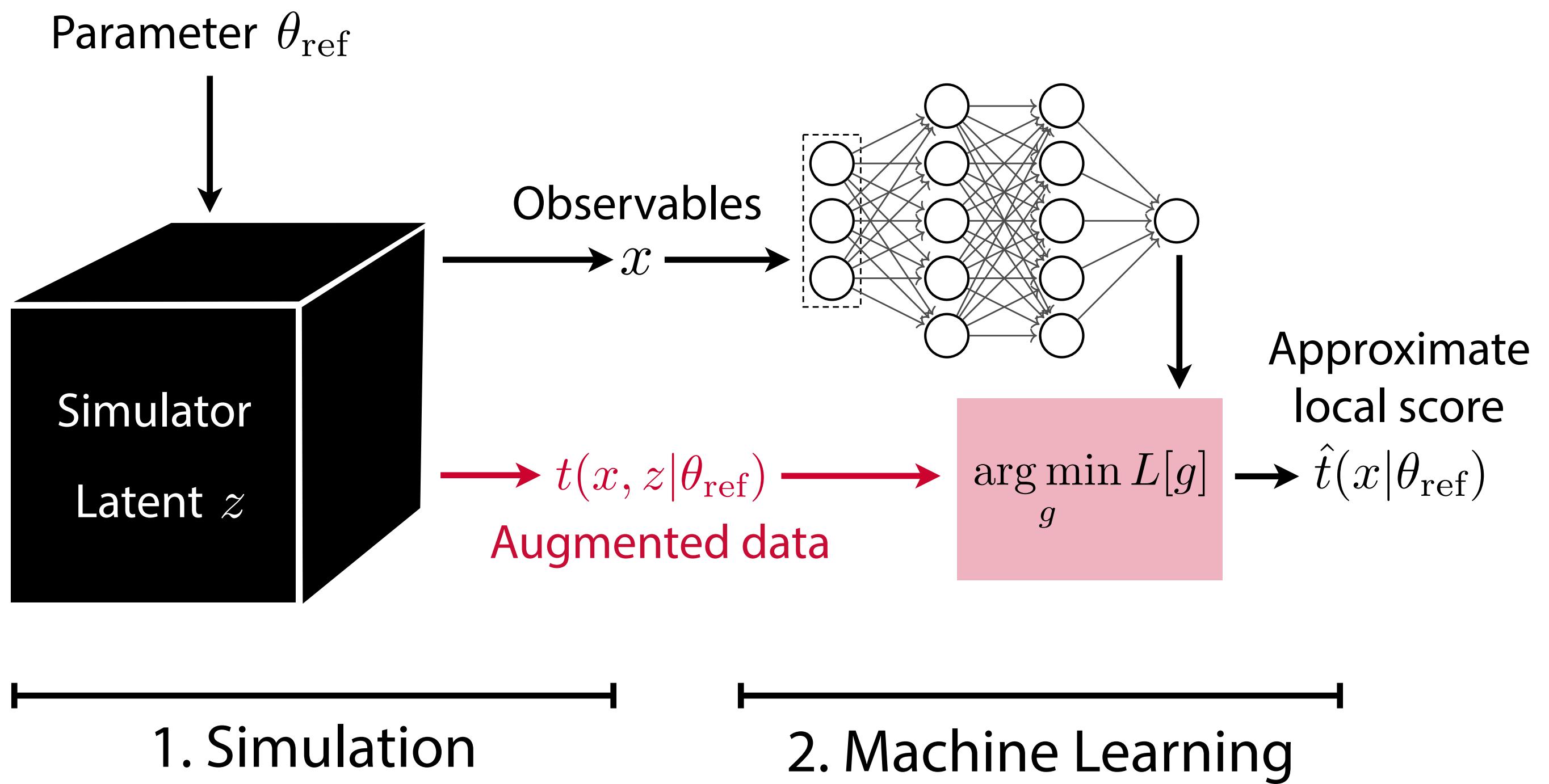
$$\begin{aligned}\log p(x|\theta) &= \log p(x|\theta_{\text{ref}}) \\ &+ \underbrace{\nabla_{\theta} \log p(x|\theta) \Big|_{\theta_{\text{ref}}} \cdot (\theta - \theta_{\text{ref}})}_{\equiv t(x|\theta_{\text{ref}})} \\ &+ \mathcal{O}((\theta - \theta_{\text{ref}})^2)\end{aligned}$$

In the neighborhood of θ_{ref} (e.g. close to the SM):

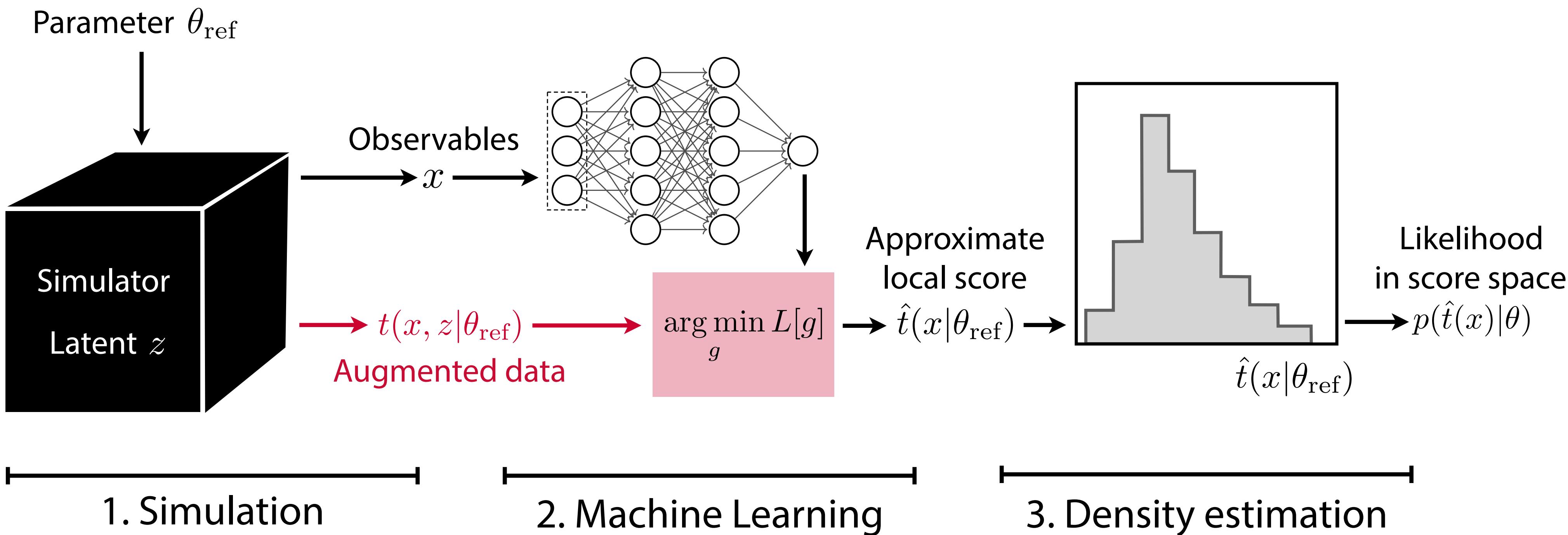
- the **score vector** $t(x|\theta_{\text{ref}})$ is the sufficient statistics
- knowing $t(x|\theta_{\text{ref}})$ is just as powerful as knowing the full function $\log p(x|\theta)$
- $t(x|\theta_{\text{ref}})$ is the most powerful observable



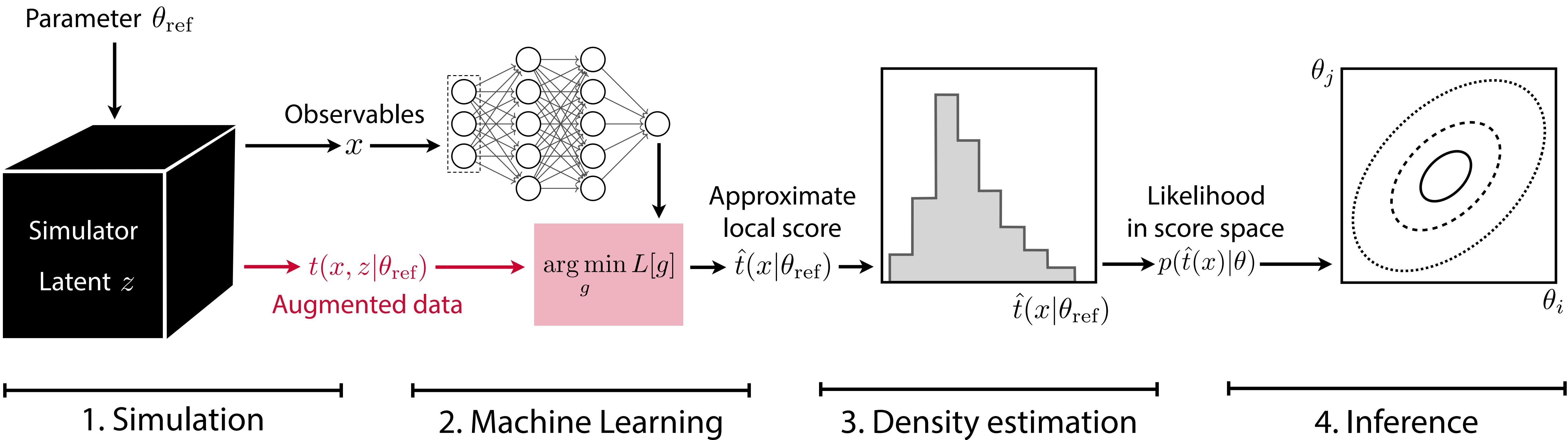
SALLY (Score approximates likelihood locally)



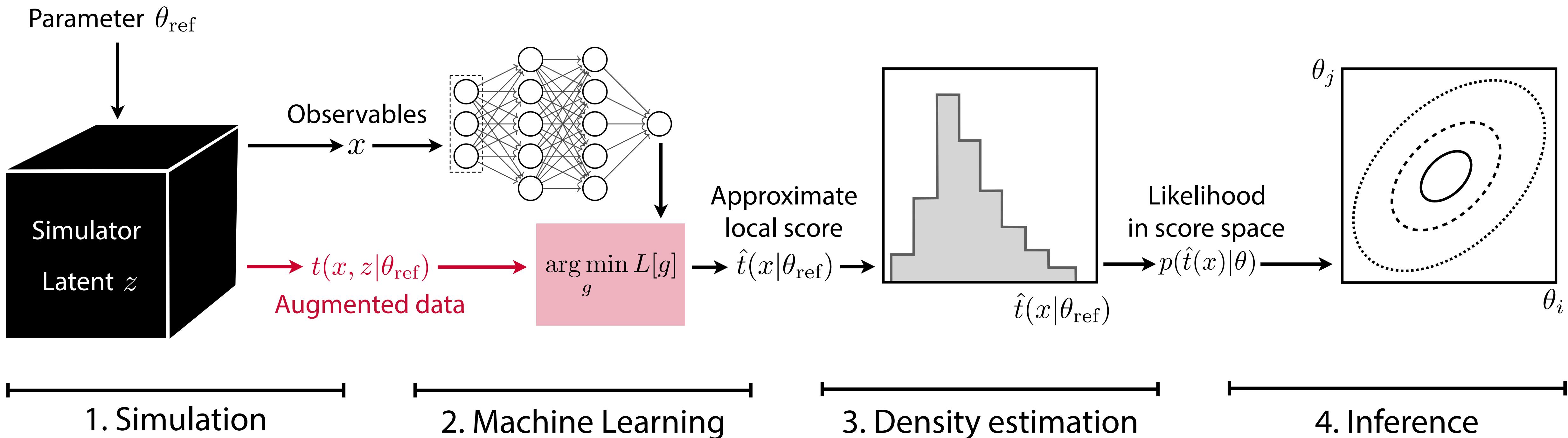
SALLY (Score approximates likelihood locally)



SALLY (Score approximates likelihood locally)



SALLY (Score approximates likelihood locally)

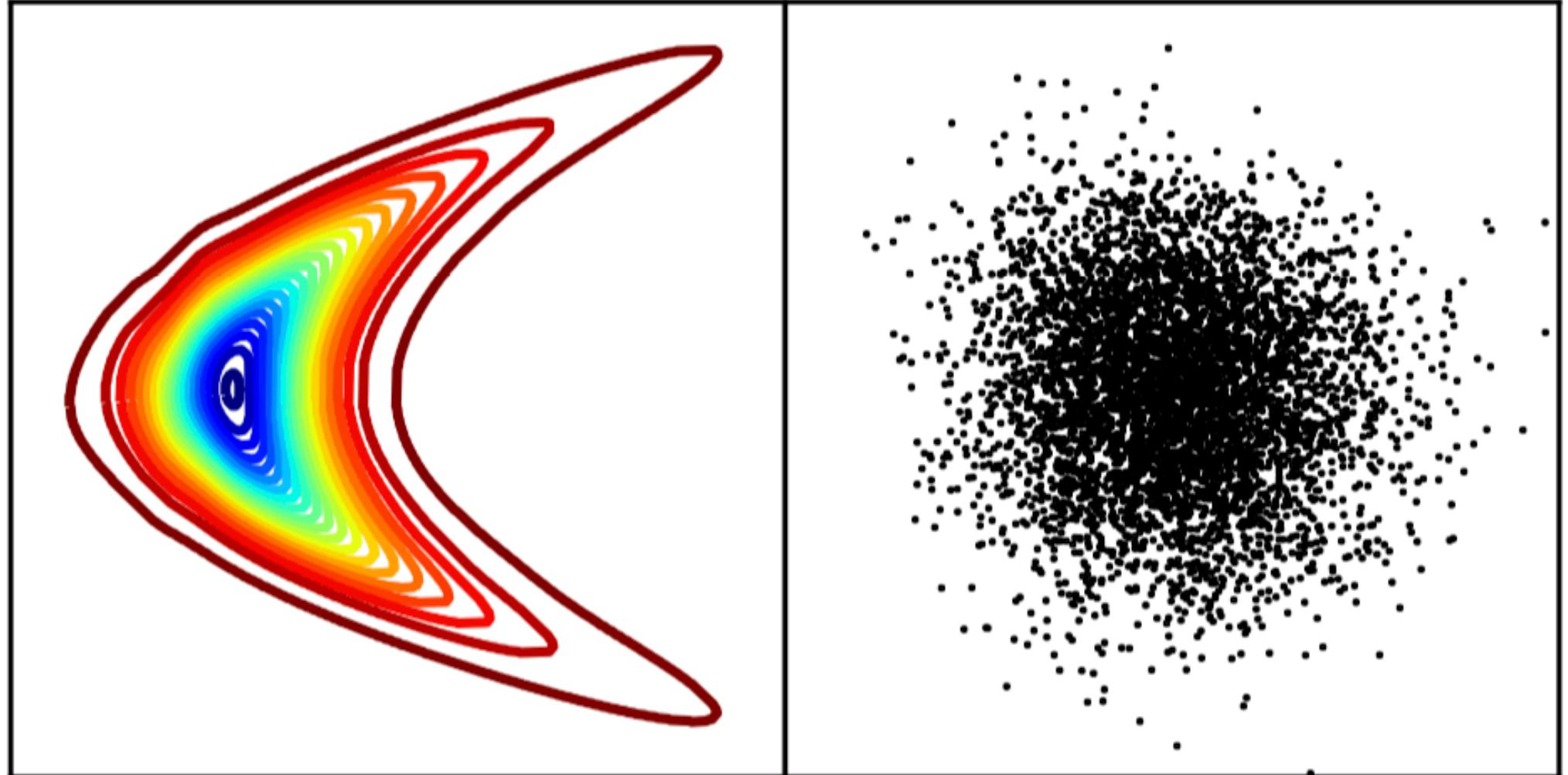


“The machine learning version of Optimal Observables”:

- Simpler & more robust than RASCAL
- Just as powerful close to θ_{ref} , but can lead to suboptimal limits further away

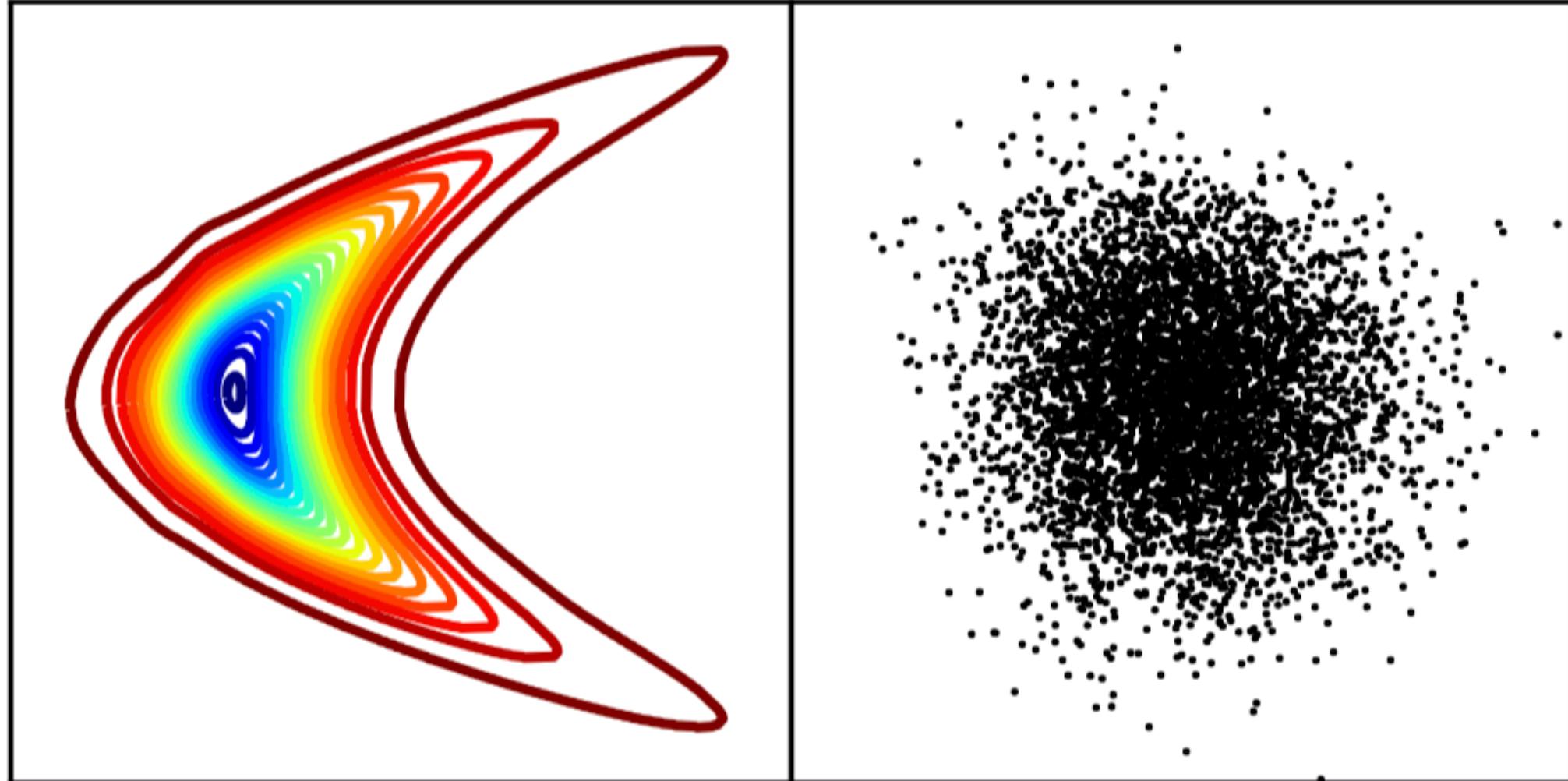
There's more...

- Many variations
 - Performance gains with cross-entropy-based loss
[M. Stoye, JB, K. Cranmer, G. Louppe, J. Pavez 1808.00973]
 - Combine with state-of-the-art neural density estimators,
e.g. Masked Autoregressive Flows
[G. Papamakarios, T. Pavlakou, I. Murray 1705.07057]



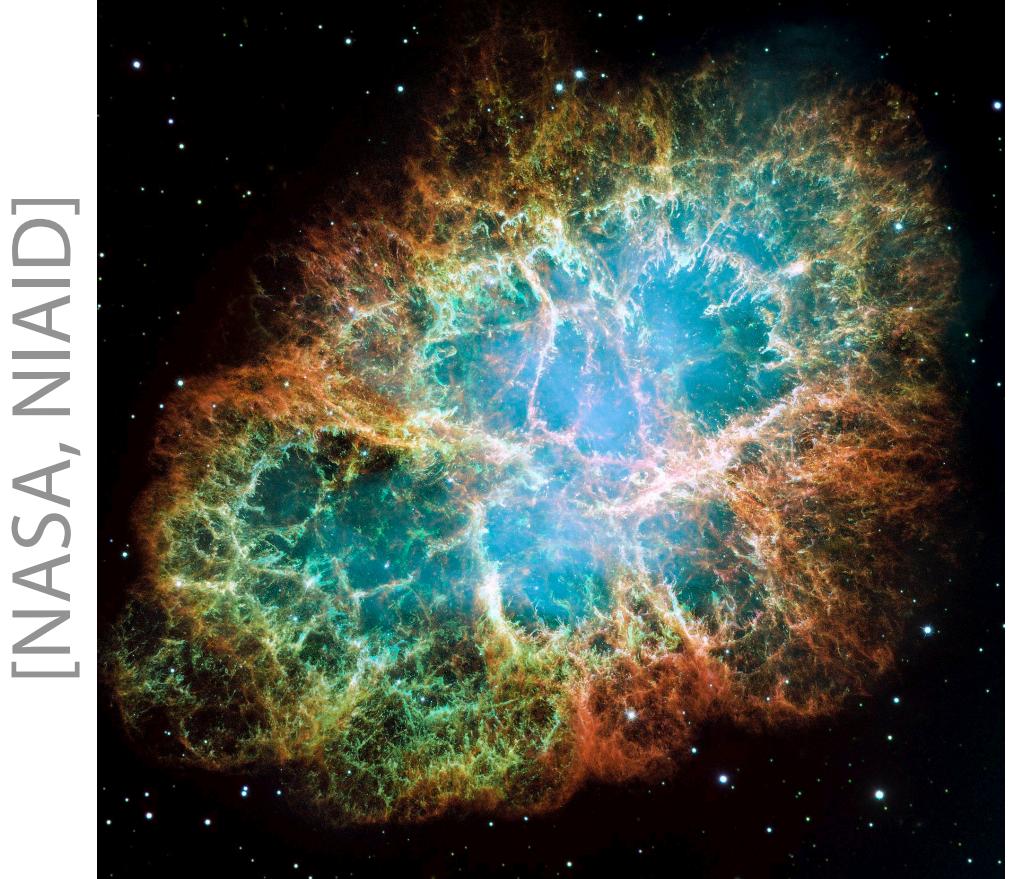
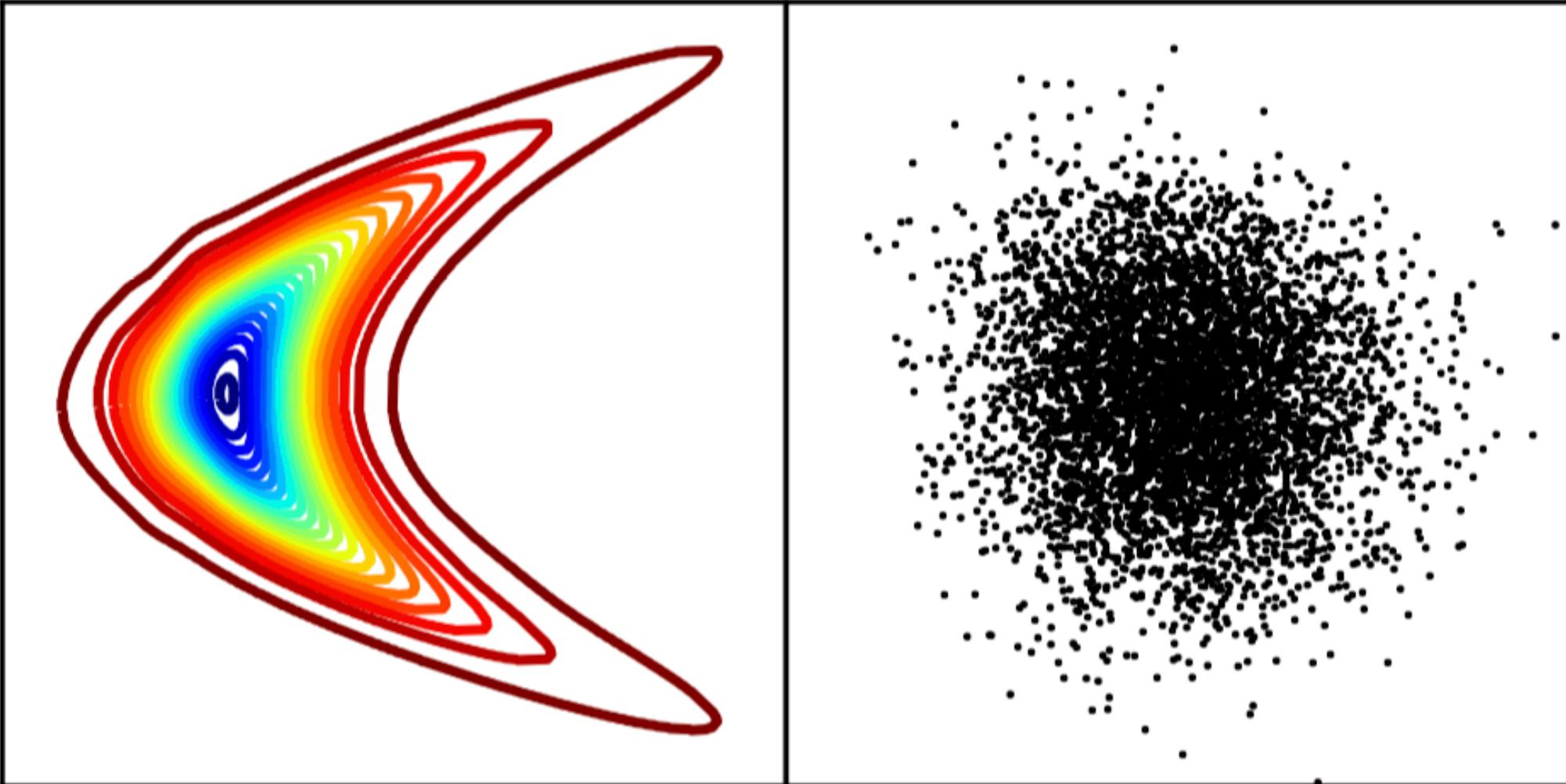
There's more...

- Many variations
 - Performance gains with cross-entropy-based loss
[M. Stoye, JB, K. Cranmer, G. Louppe, J. Pavez 1808.00973]
 - Combine with state-of-the-art neural density estimators,
e.g. Masked Autoregressive Flows
[G. Papamakarios, T. Pavlakou, I. Murray 1705.07057]
- Don't fully trust the simulator?
 - Nuisance parameters to model systematic uncertainties
 - Learn robustness with adversarial training
[G. Louppe, M. Kagan, K. Cranmer 1611.01046]
- Don't blindly trust the neural network?
 - Many diagnostic cross checks
 - Calibration / Neyman construction: badly trained network can lead to suboptimal limits, but not to wrong limits



There's more...

- Many variations
 - Performance gains with cross-entropy-based loss
[M. Stoye, JB, K. Cranmer, G. Louppe, J. Pavez 1808.00973]
 - Combine with state-of-the-art neural density estimators,
e.g. Masked Autoregressive Flows
[G. Papamakarios, T. Pavlakou, I. Murray 1705.07057]
- Don't fully trust the simulator?
 - Nuisance parameters to model systematic uncertainties
 - Learn robustness with adversarial training
[G. Louppe, M. Kagan, K. Cranmer 1611.01046]
- Don't blindly trust the neural network?
 - Many diagnostic cross checks
 - Calibration / Neyman construction: badly trained network can lead to suboptimal limits, but not to wrong limits
- Applications beyond particle physics



A sustainable software implementation

[JB, F. Kling, I. Espejo, K. Cranmer doi: 10.5281/zenodo.1489147]



The SCAILFIN Project
scailfin.github.io



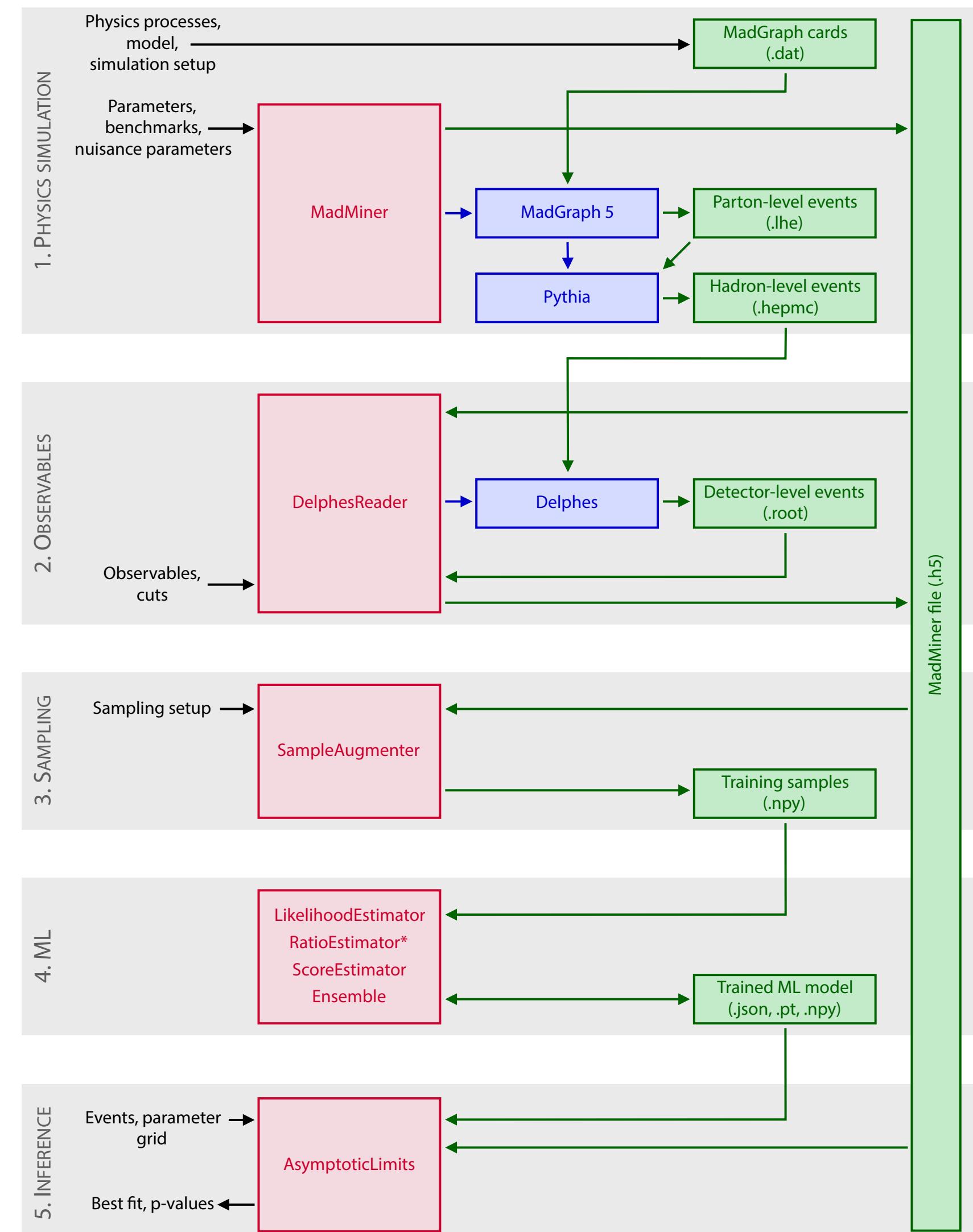


[S. Adams 2013]

Automation

We are developing **MadMiner**, which makes it straightforward to apply the new techniques to LHC problems

- Out of the box: Pheno-level analyses
 - MadGraph, Pythia, Delphes
 - Systematic uncertainties from PDF / scale variation
- Scalable to state-of-the-art experimental tools
 - Mostly requires bookkeeping of fully differential cross sections
- Modular interface
 - Extensive documentation
 - Embedded into Python / ML ecosystem



MadMiner

Johann Brehmer, Felix Kling, Irina Espejo, and Kyle Cranmer

Mining gold from MadGraph to improve limit setting in particle physics.

Note that this is an early development version. Do not expect anything to be stable. If you have any questions, please contact us at johann.brehmer@nyu.edu.

Note: Consider working with the `future` branch, where we develop a refactored and cleaned up version of MadMiner. The API and even the file format for trained neural networks is different from the `master` version (v0.2.x). We expect these changes to make it into MadMiner v0.3.0.

[pypi package](#) 0.2.8 [docs](#) passing [build](#) error [docker pulls](#) 2k [launch](#) [binder](#) [code style](#) black [License](#) MIT [DOI](#) 10.5281/zenodo.1489147

Introduction

Particle physics processes are usually modelled with complex Monte-Carlo simulations of the hard process, parton shower,

The screenshot shows the MadMiner documentation website. The left sidebar has a blue header with the logo and "latest" version. It contains sections for "SITES" (Introduction to MadMiner, Getting started, Using MadMiner, References) and "REFERENCE" (list of modules: core, delphes, fisherinformation, lhe, ml, morphing, plotting). The main content area has a "Docs » MadMiner" header and an "Edit on GitHub" button. The title "MadMiner" is in large bold letters, followed by the authors' names. Below that is a subtitle "An inference toolkit for LHC measurements". A note says "Note that this is a development version. Do not rely on anything being stable. If you have any questions, please contact us at johann.brehmer@nyu.edu". The "Sites" section is expanded, showing a bulleted list of the four "Getting started" links from the sidebar.

SITES

- [Introduction to MadMiner](#)
- [Getting started](#)
- [Using MadMiner](#)
- [References](#)

REFERENCE

- [madminer.core module](#)
- [madminer.delphes module](#)
- [madminer.fisherinformation module](#)
- [madminer.lhe module](#)
- [madminer.ml module](#)
- [madminer.morphing module](#)
- [madminer.plotting module](#)

Docs » MadMiner

[Edit on GitHub](#)

MadMiner

Johann Brehmer, Felix Kling, Irina Espejo, and Kyle Cranmer

An inference toolkit for LHC measurements

Note that this is a development version. Do not rely on anything being stable. If you have any questions, please contact us at johann.brehmer@nyu.edu.

Sites

- [Introduction to MadMiner](#)
- [Getting started](#)
- [Using MadMiner](#)
- [References](#)

Reference

Repository and tutorials:
github.com/johannbrehmer/madminer

Documentation:

madminer.readthedocs.io



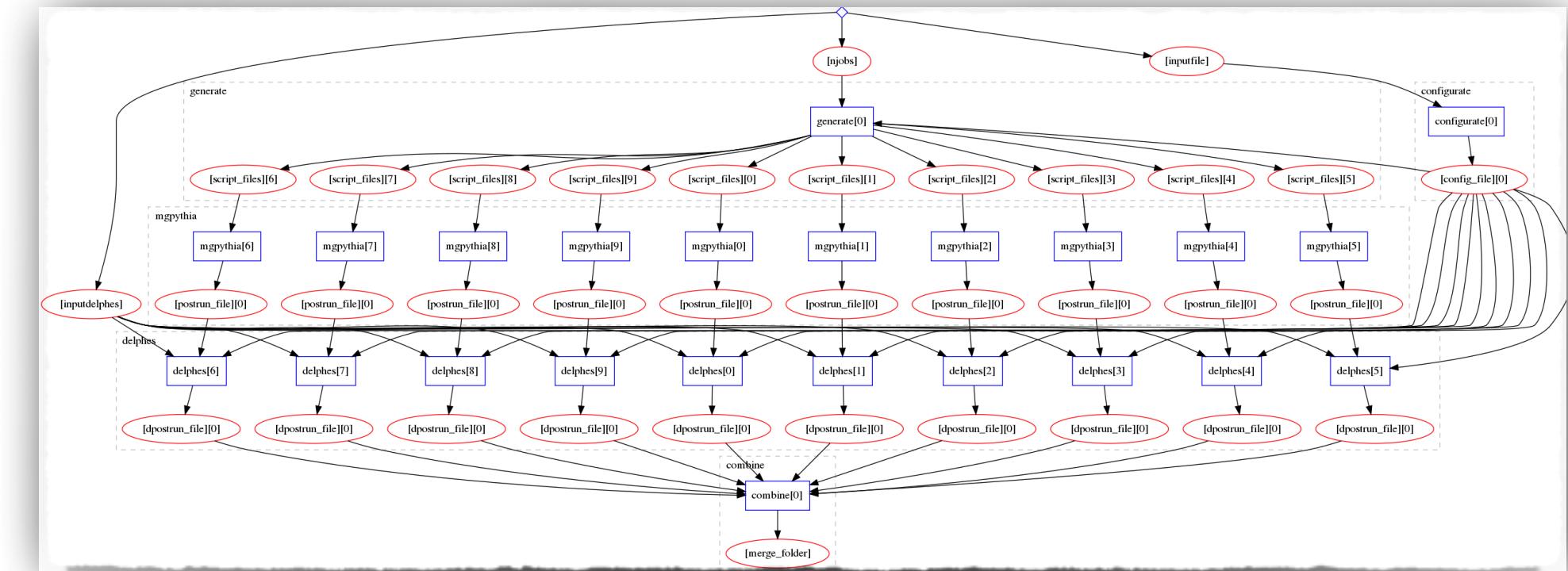


Help

Donate

Log in

Register



Installation:

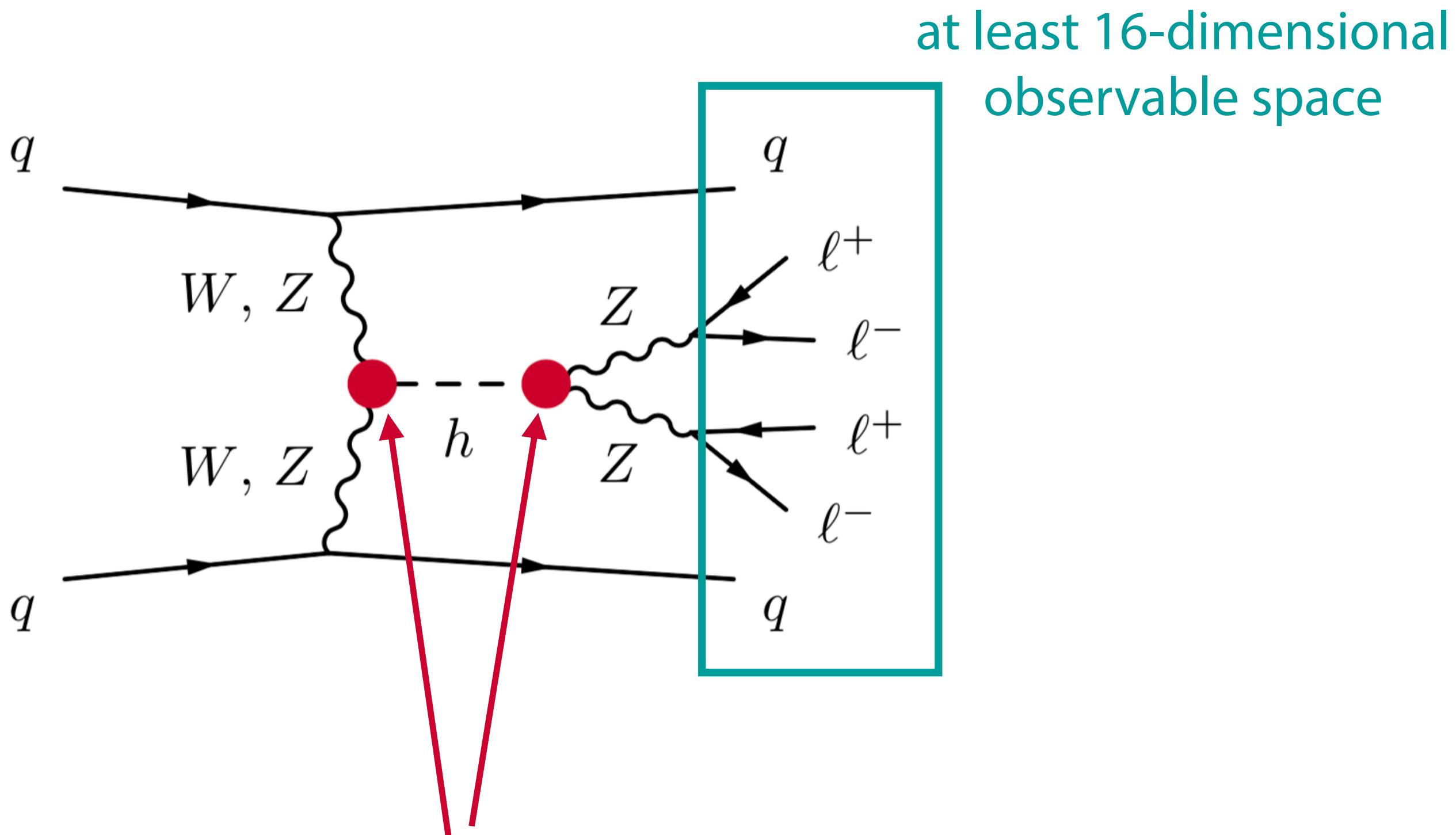
pip install madminer

Deployment with Docker, yadage, REANA: github.com/irinaespejo/workflow-madminer

In practice

Proof of concept: Higgs production in weak boson fusion

[JB, K. Cranmer, G. Louppe, J. Pavez
1805.00013, 1805.00020]



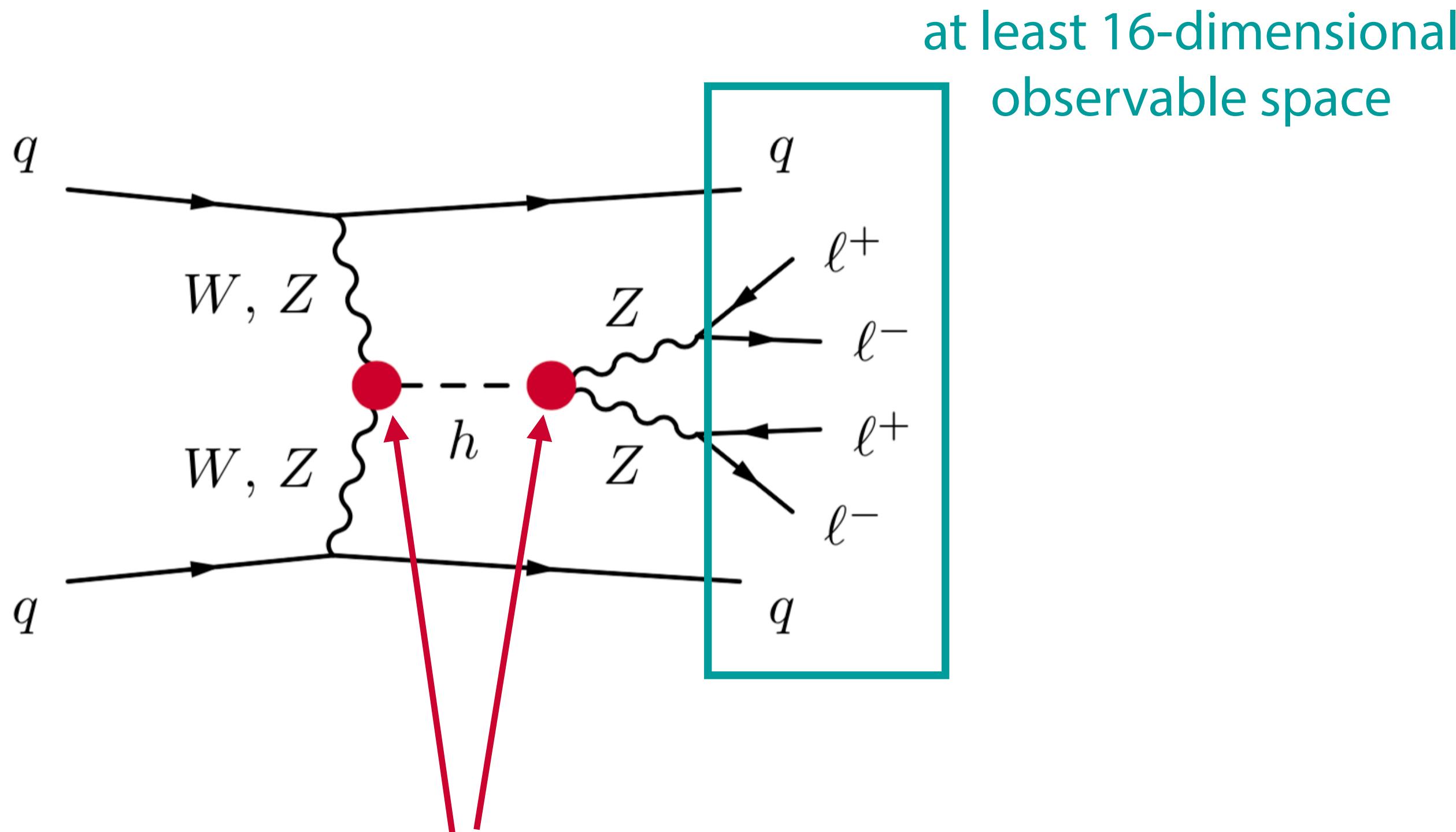
Exciting new physics might hide here!

We parameterize it with two EFT coefficients:

$$\mathcal{L} = \mathcal{L}_{\text{SM}} + \underbrace{\left[\frac{f_W}{\Lambda^2} \frac{i g}{2} (D^\mu \phi)^\dagger \sigma^a D^\nu \phi W_{\mu\nu}^a \right]}_{\mathcal{O}_W} - \underbrace{\left[\frac{f_{WW}}{\Lambda^2} \frac{g^2}{4} (\phi^\dagger \phi) W_{\mu\nu}^a W^{\mu\nu a} \right]}_{\mathcal{O}_{WW}}$$

Proof of concept: Higgs production in weak boson fusion

[JB, K. Cranmer, G. Louppe, J. Pavez
1805.00013, 1805.00020]



We parameterize it with two EFT coefficients:

$$\mathcal{L} = \mathcal{L}_{\text{SM}} + \underbrace{\frac{f_W}{\Lambda^2} \frac{ig}{2} (D^\mu \phi)^\dagger \sigma^a D^\nu \phi W_{\mu\nu}^a}_{\mathcal{O}_W} - \underbrace{\frac{f_{WW}}{\Lambda^2} \frac{g^2}{4} (\phi^\dagger \phi) W_{\mu\nu}^a W^{\mu\nu a}}_{\mathcal{O}_{WW}}$$

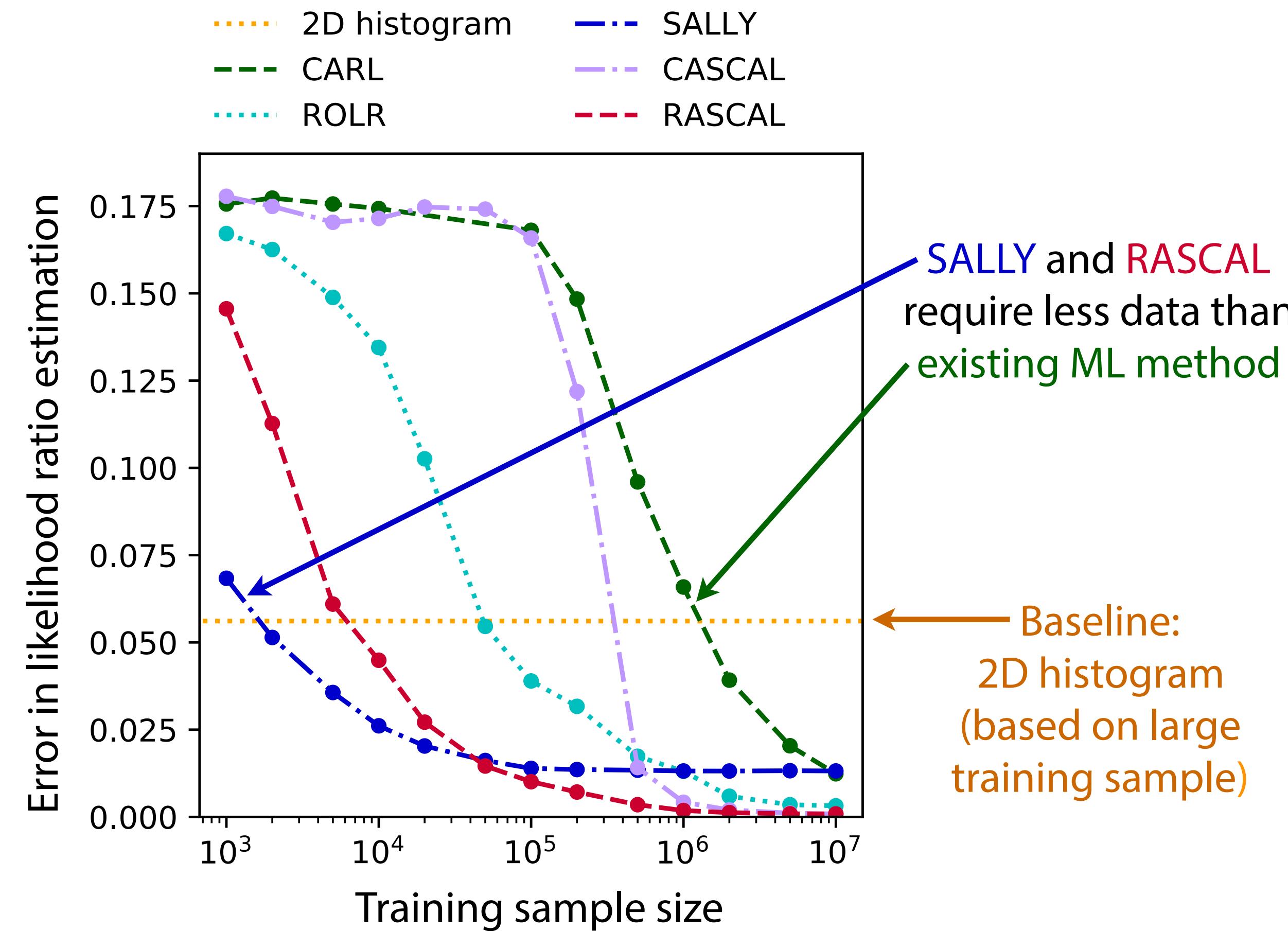
Goal: constrain the two EFT parameters

- new inference methods
- baseline: 2d histogram analysis of jet momenta & angular correlations

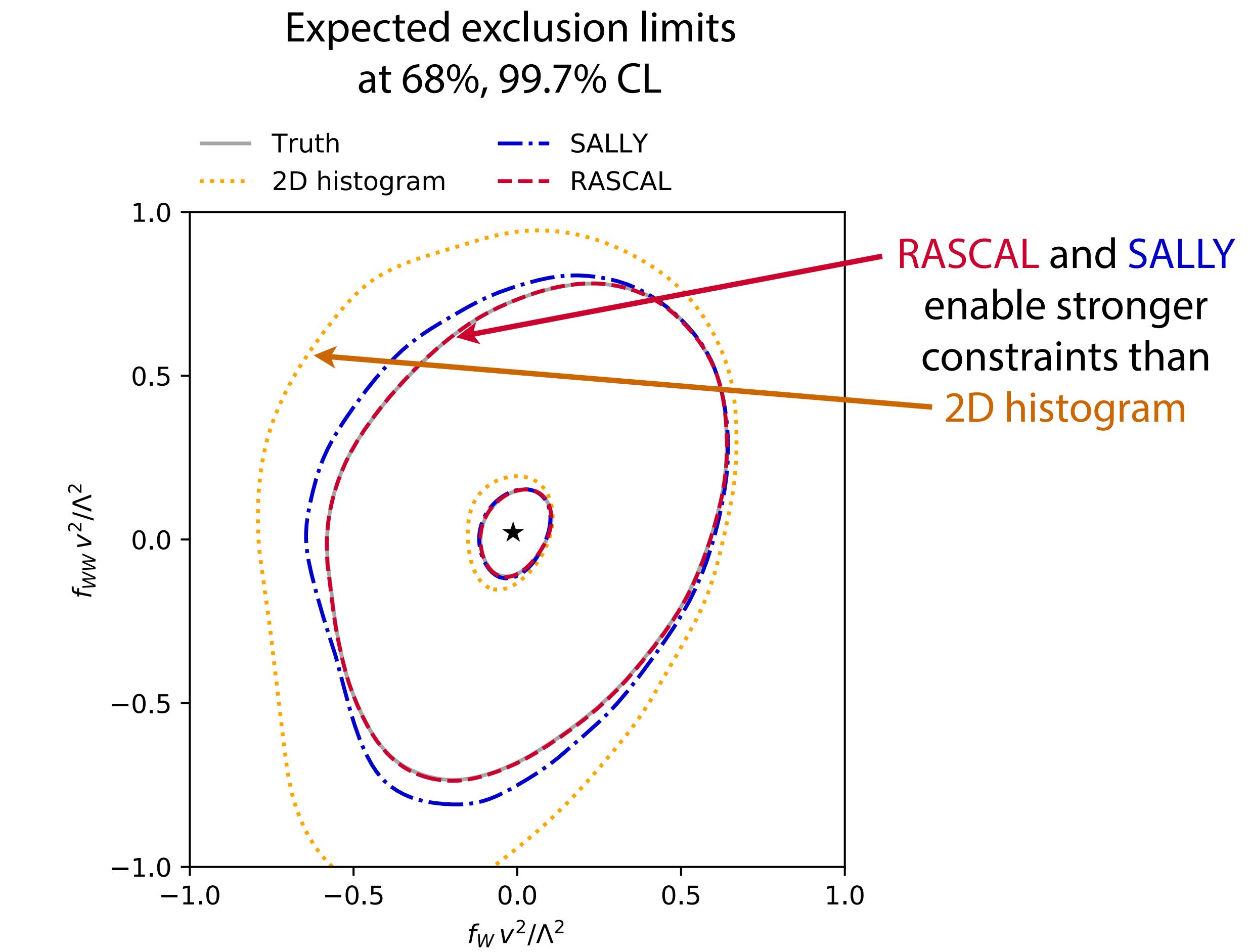
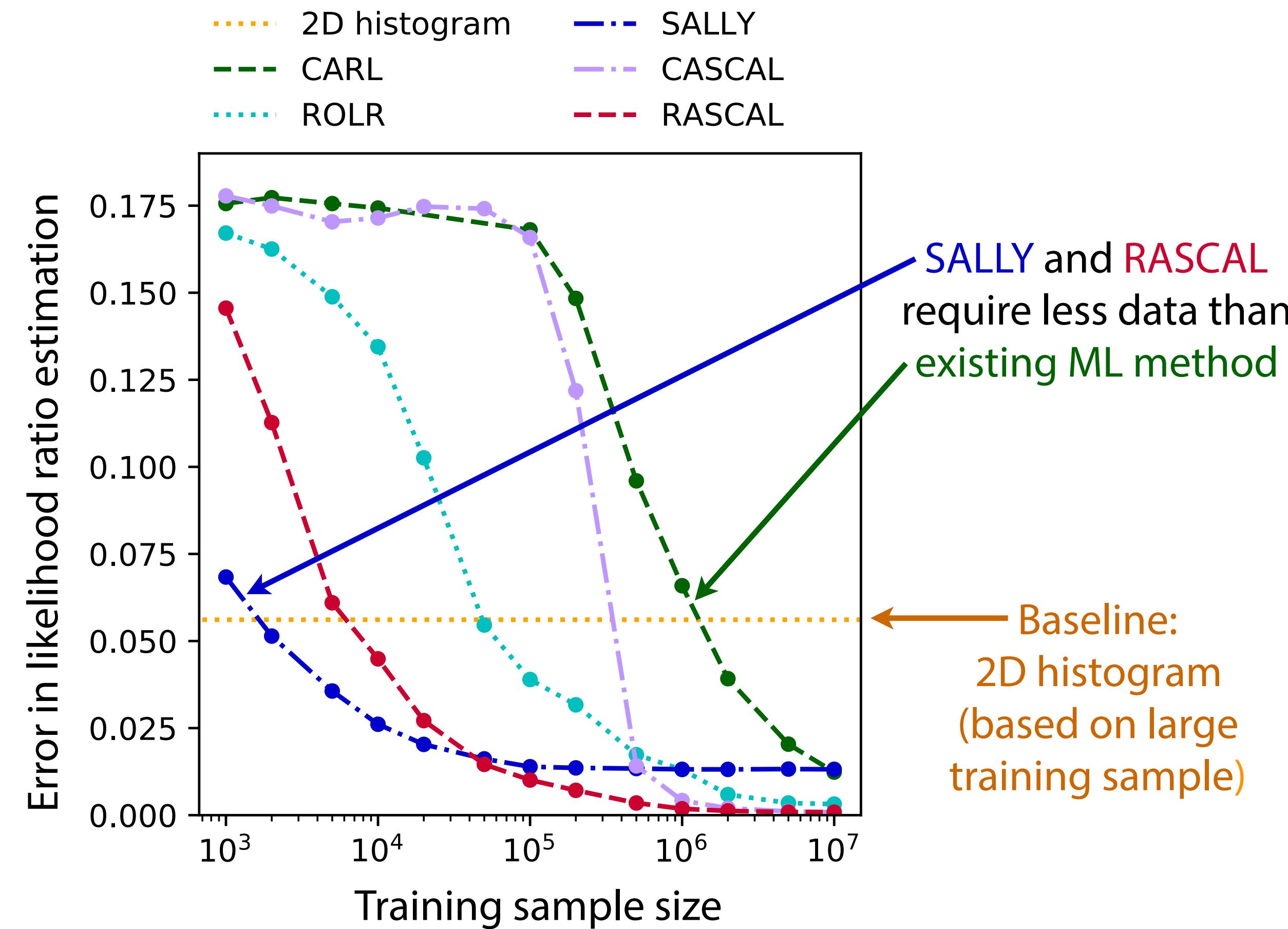
Two scenarios:

- Simplified setup in which we can compare to true likelihood
- “Realistic” simulation with approximate detector effects

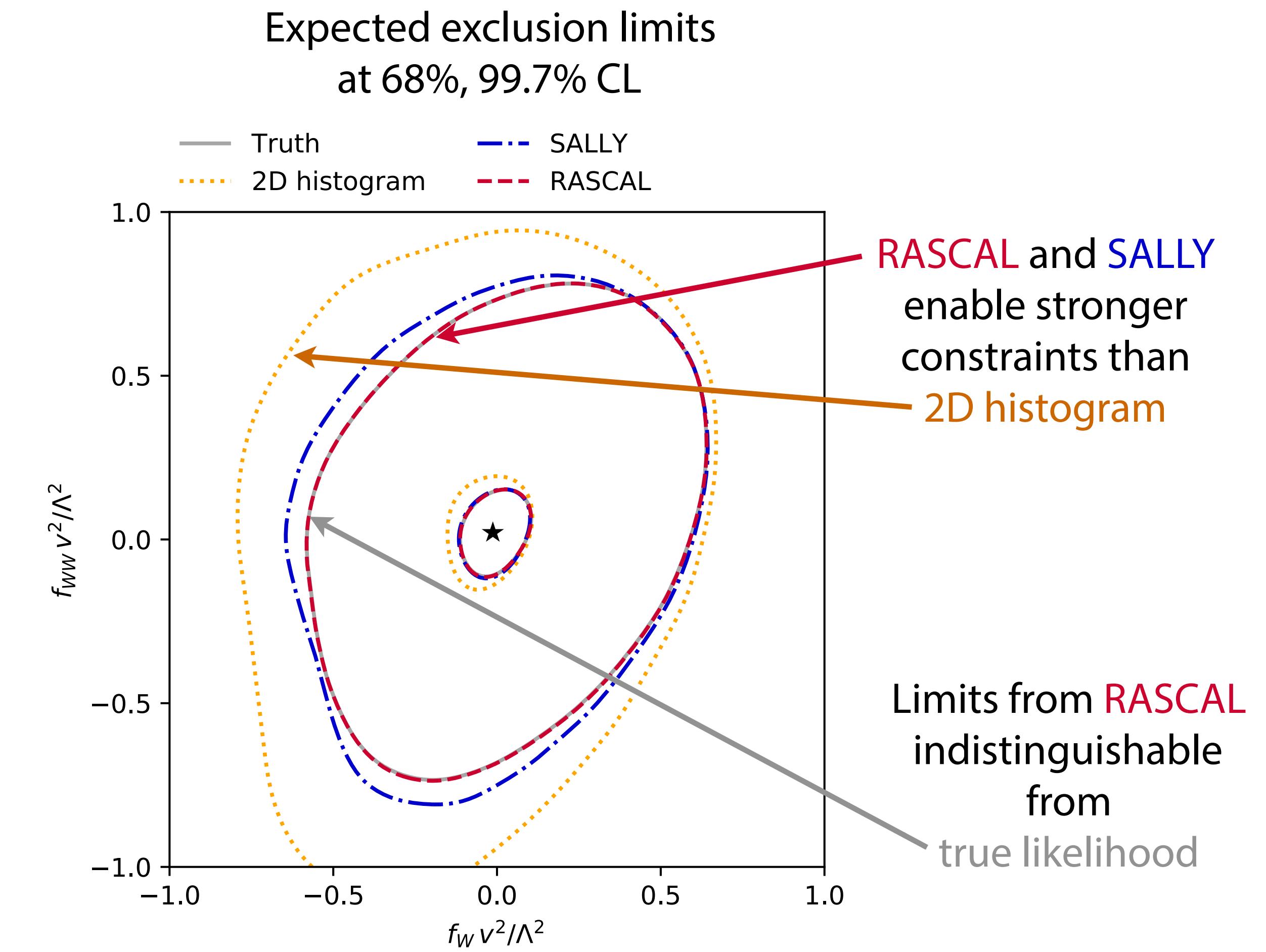
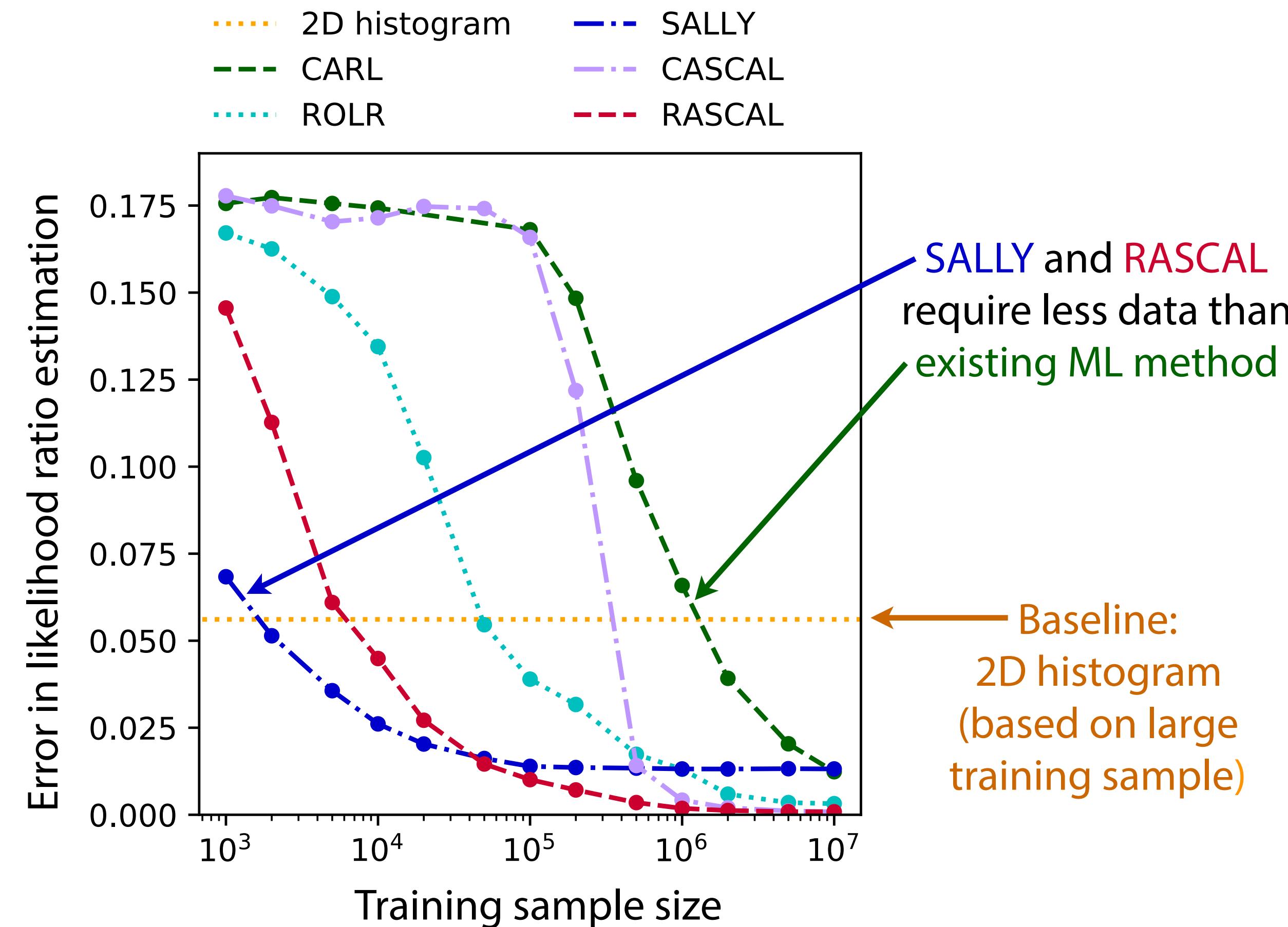
Stronger constraints with less training data



Stronger constraints with less training data



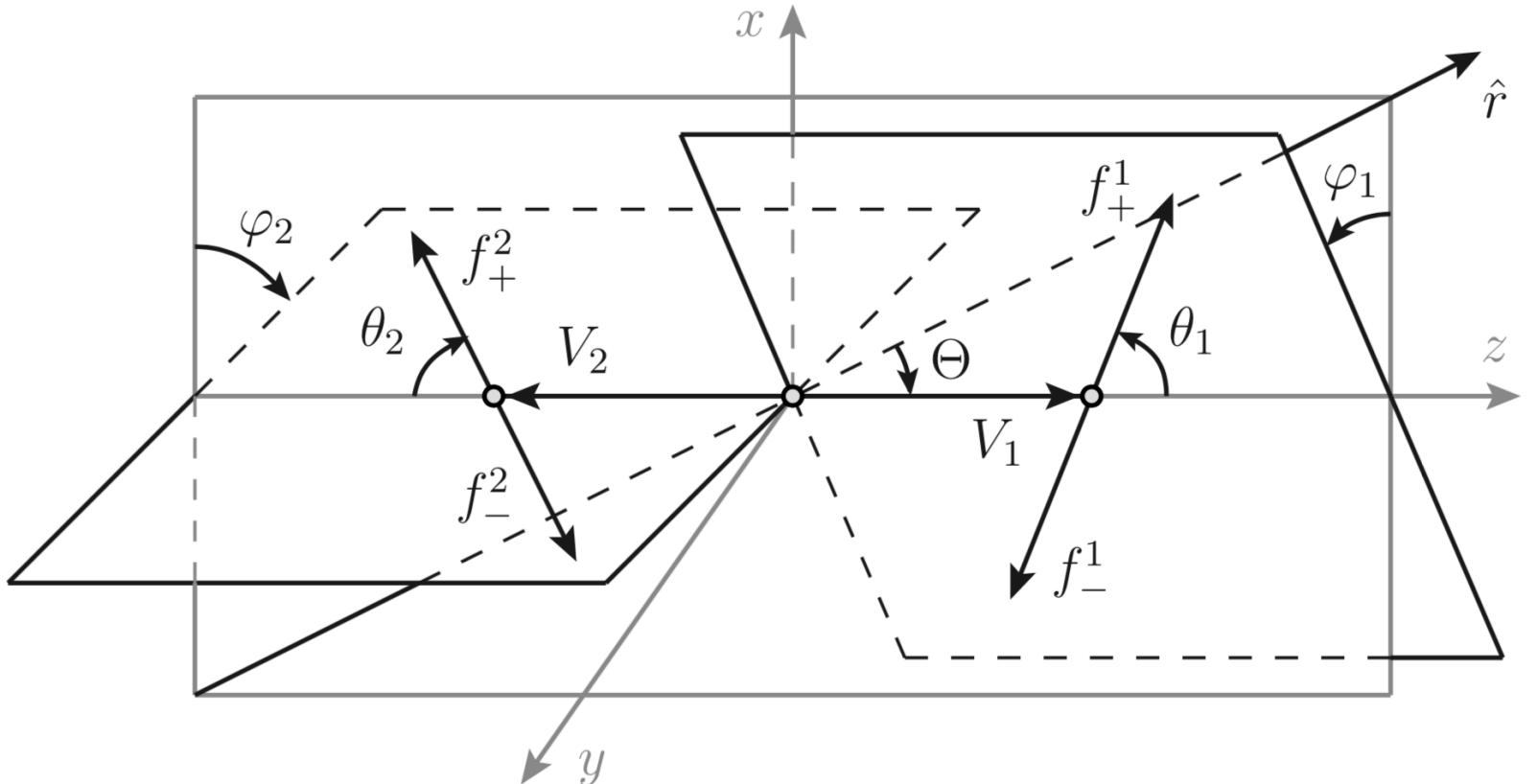
Stronger constraints with less training data



Diboson production

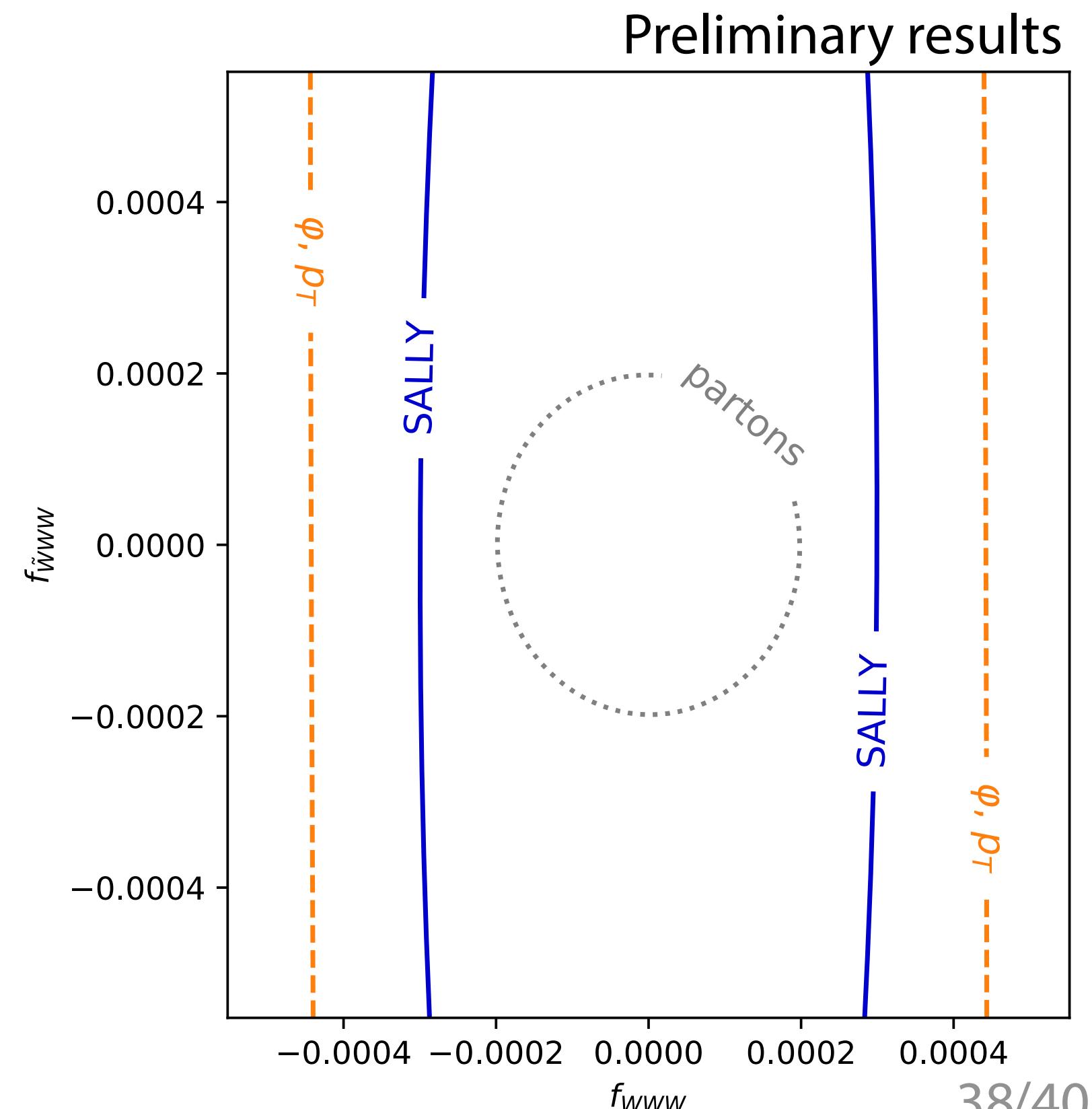
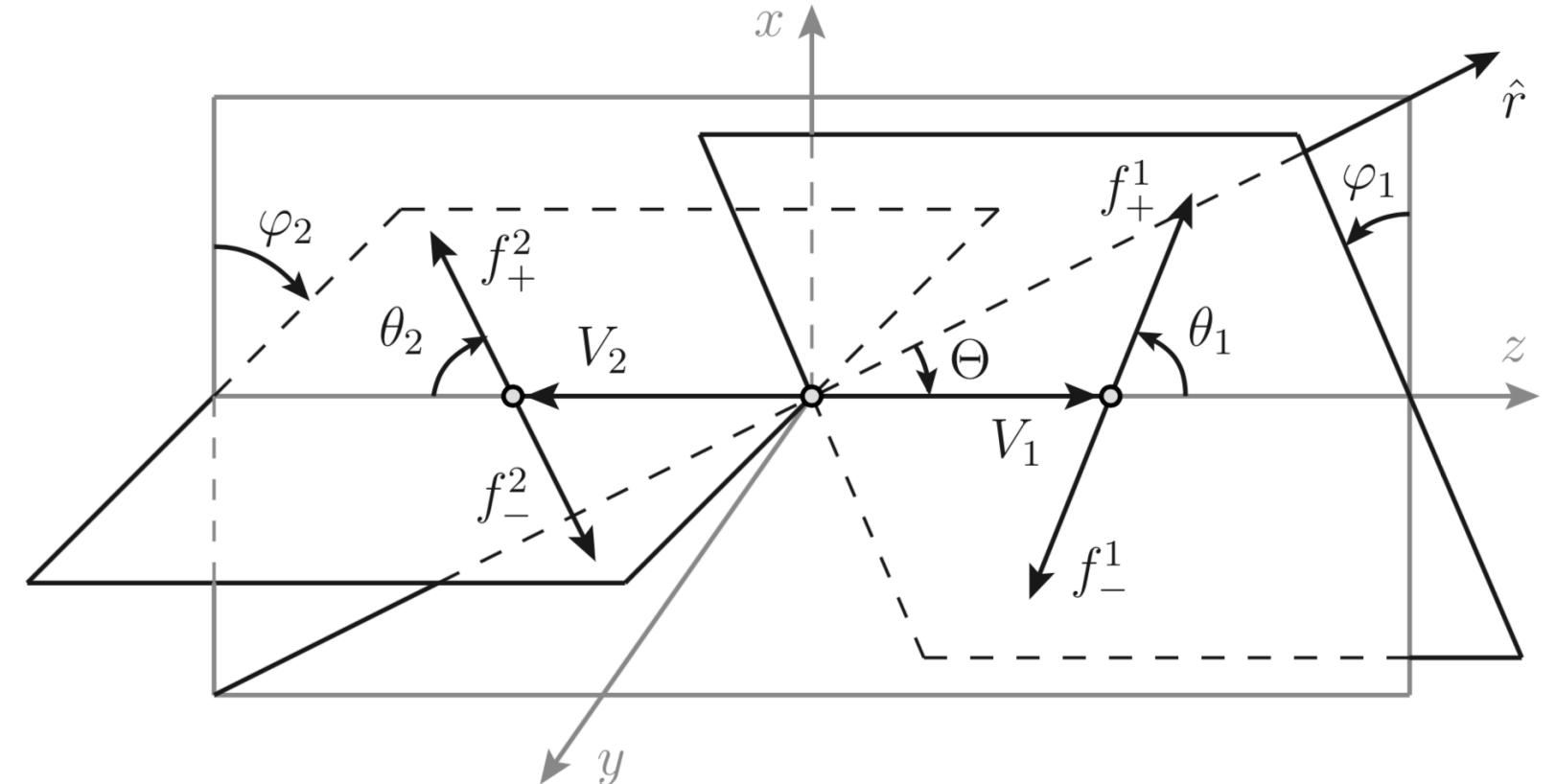
- In inclusive observables, the interference between SM and new physics amplitudes vanishes
⇒ Reduced sensitivity to new physics
- “Diboson interference resurrection”:
an **angular variable** φ can be constructed to be sensitive to this interference

[G. Panico, F. Riva, A. Wulzer 1708.07823]

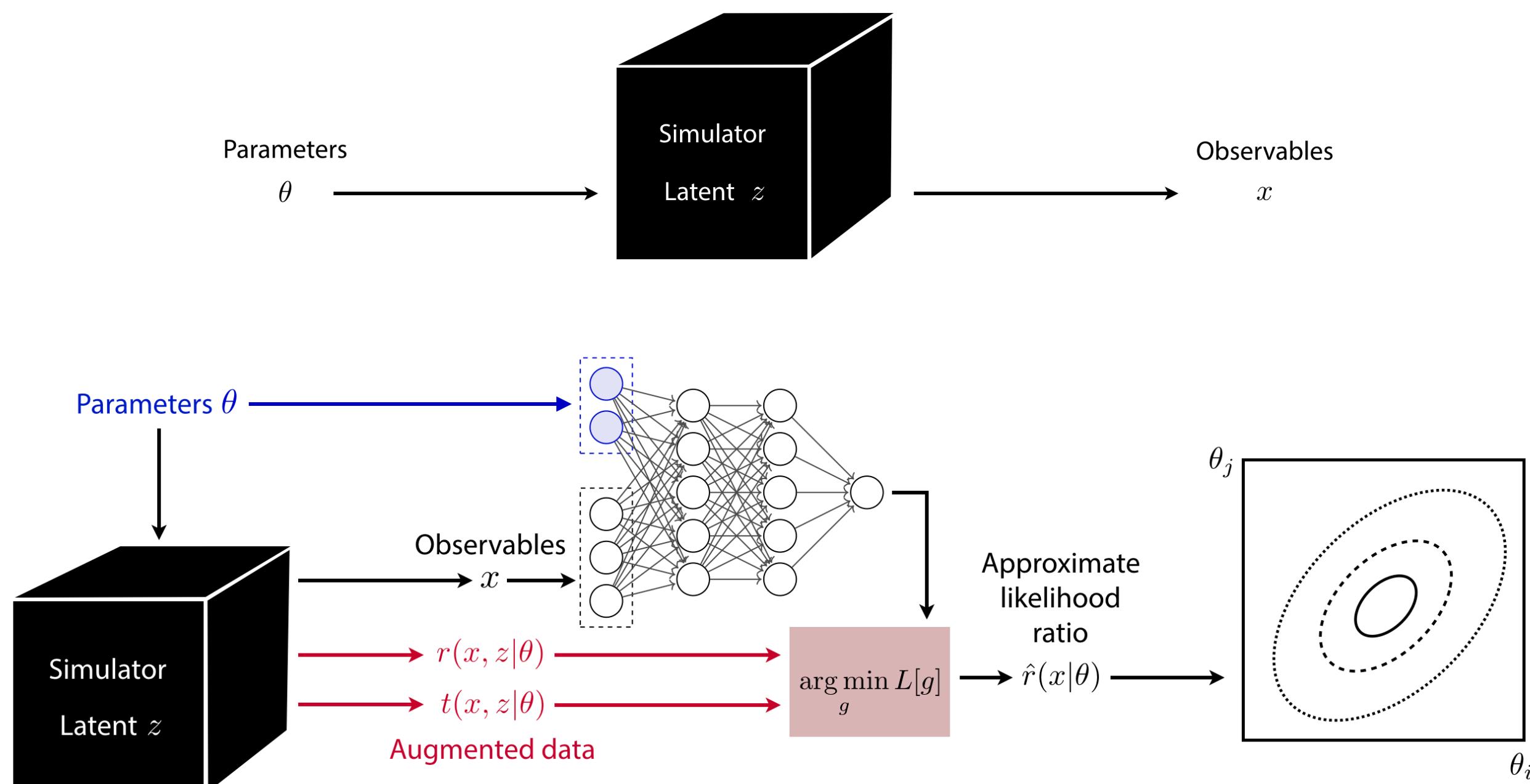


Diboson production

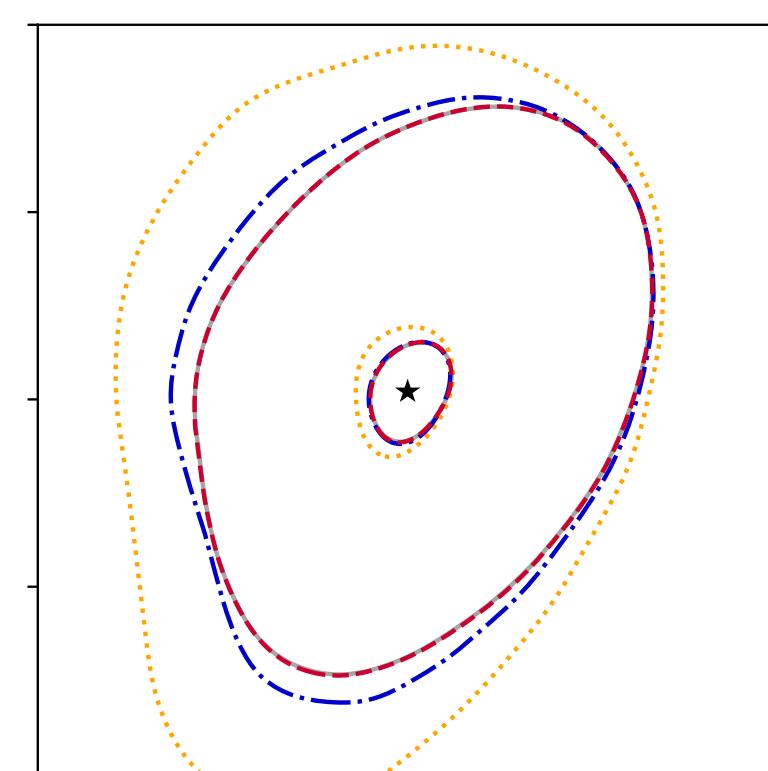
- In inclusive observables, the interference between SM and new physics amplitudes vanishes
⇒ Reduced sensitivity to new physics
- “Diboson interference resurrection”:
an **angular variable** φ can be constructed to be sensitive to this interference
[G. Panico, F. Riva, A. Wulzer 1708.07823]
- We test the ML approach in EFT measurements in $W\gamma \rightarrow \ell\nu \gamma$
[JB, K. Cranmer, M. Farina, F. Kling, D. Pappadopulo, J. Ruderman in progress]
- Preliminary results: we can extract more information
when we **analyze events with SALLY**
than with **histograms of φ and standard observables**



A new approach to LHC measurements



- LHC measurements with high-dimensional observations have “likelihood-free” structure
- New multivariate inference techniques:
Leverage information in matrix elements + power of machine learning to...
 - estimate the full likelihood function
 - learn optimal summary statistics
- First tests show potential to substantially increase sensitivity to new physics



References



Kyle Cranmer



Gilles Louppe



Juan Pavez



Markus Stoye



Felix Kling

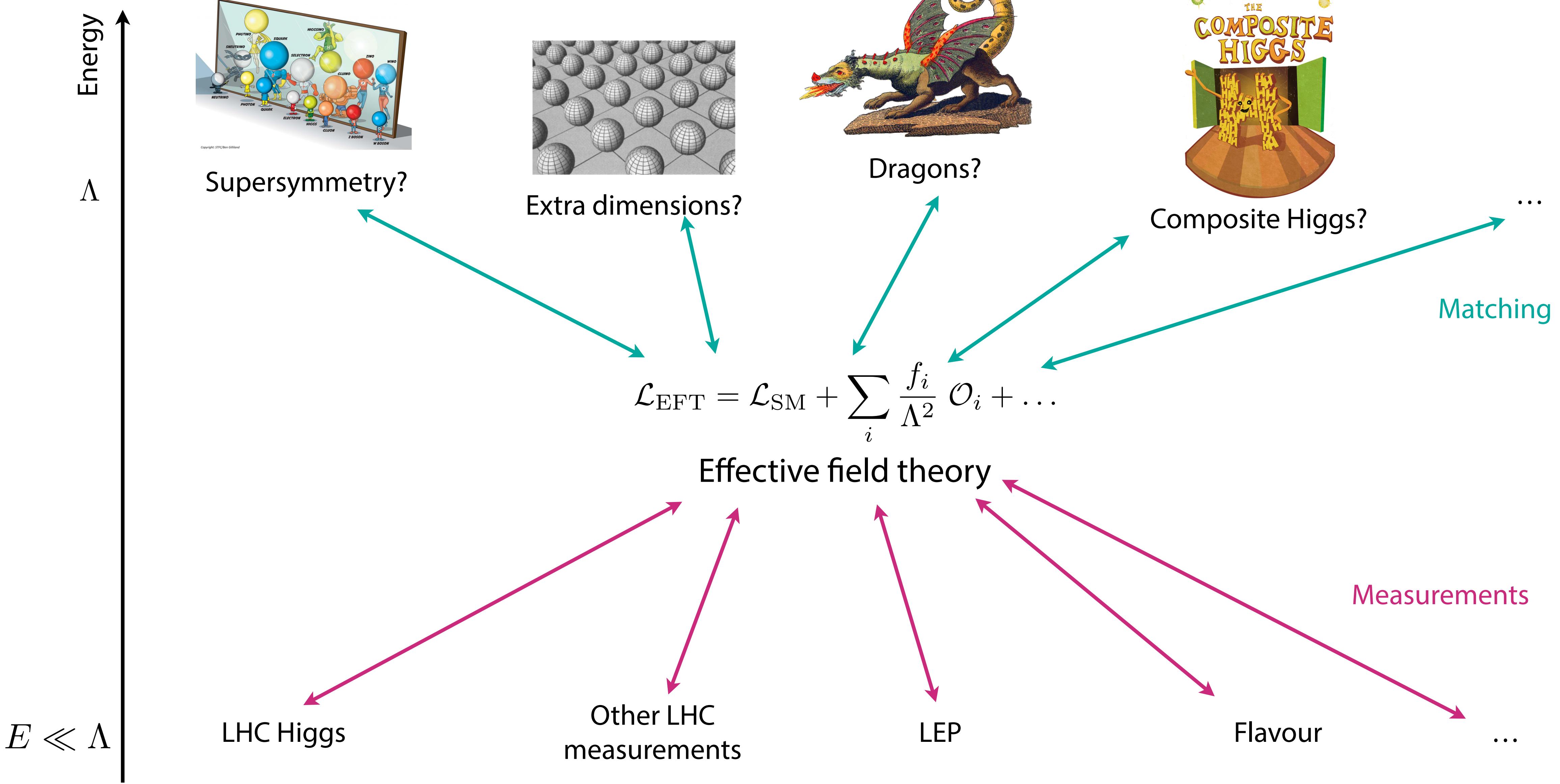


Irina Espejo

JB, KC, GL, JP:	Constraining Effective Field Theories with Machine Learning	[PRL, 1805.00013]
JB, KC, GL, JP:	A Guide to Constraining Effective Field Theories with Machine Learning	[PRD, 1805.00020]
JB, GL, JP, KC:	Mining gold from implicit models to improve likelihood-free inference	[1805.12244]
MS, JB, GL, JP, KC:	Likelihood-free inference with an improved cross-entropy estimator	[1808.00973]
JB, FK, IE, KC:	MadMiner: An inference toolkit for particle physics	[doi: 10.5281/zenodo.1489147]

Bonus material

Effective field theory



SMEFT (Standard Model Effective Field Theory)

[W. Buchmuller, D. Wyler 85;
B. Grzadkowski, M. Iskrzynski, M. Misiak, J. Rosiek 1008.4884; ...]

$$\mathcal{L}_{\text{EFT}} = \mathcal{L}_{\text{SM}} + \sum_i \frac{f_i}{\Lambda^2} \mathcal{O}_i + \dots$$

Wilson coefficients

- precise measurement of these parameters is one of the most important goals of the LHC
- can be translated to high-energy physics parameters

Operators

- all possible interactions between SM particles mediated by new physics
- fixed by SM particles + SM symmetries + expansion in $1/\Lambda$, independent of high-energy physics
- affect rates + kinematics

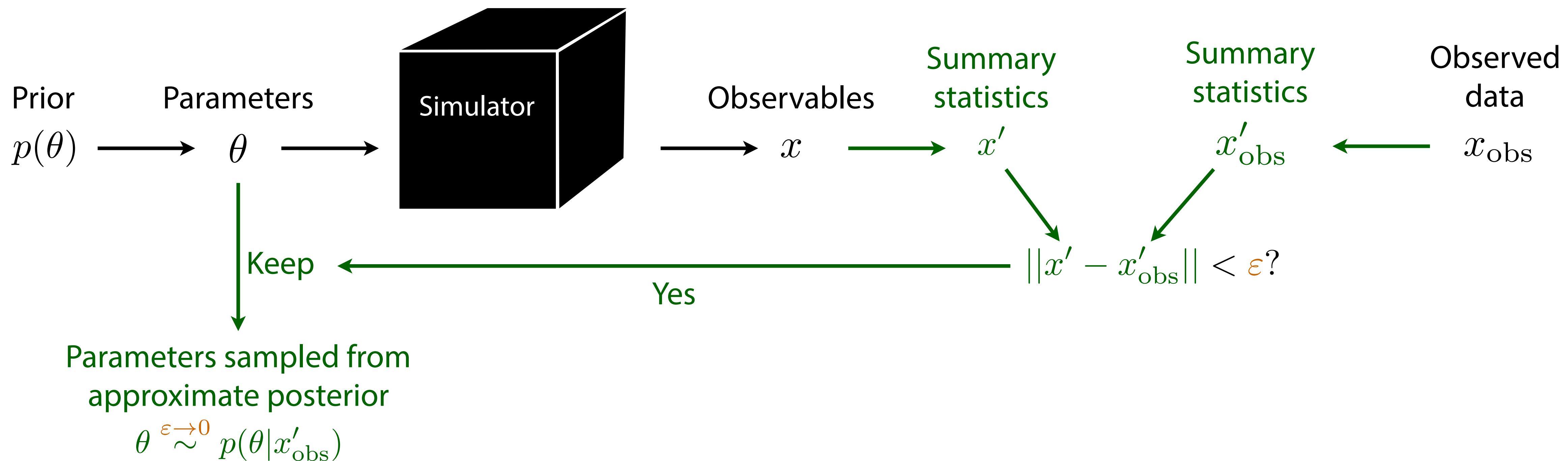
Higher-order terms

- suppressed by additional factors of E^2 / Λ^2

$\mathcal{O}_{\phi,1} = (D_\mu \phi)^\dagger \phi \phi^\dagger D^\mu \phi$	$\mathcal{O}_{GG} = (\phi^\dagger \phi) G_{\mu\nu}^a G^{\mu\nu a}$
$\mathcal{O}_{\phi,2} = \frac{1}{2} \partial_\mu (\phi^\dagger \phi) \partial^\mu (\phi^\dagger \phi)$	$\mathcal{O}_{BB} = -\frac{g'^2}{4} (\phi^\dagger \phi) B_{\mu\nu} B^{\mu\nu}$
$\mathcal{O}_{\phi,3} = \frac{1}{3} (\phi^\dagger \phi)^3$	$\mathcal{O}_{WW} = -\frac{g^2}{4} (\phi^\dagger \phi) W_{\mu\nu}^a W^{\mu\nu a}$
$\mathcal{O}_{\phi,4} = (\phi^\dagger \phi) (D_\mu \phi)^\dagger D^\mu \phi$	$\mathcal{O}_{BW} = -\frac{gg'}{4} (\phi^\dagger \sigma^a \phi) B_{\mu\nu} W^{\mu\nu a}$
	$\mathcal{O}_B = \frac{ig'}{2} (D^\mu \phi)^\dagger D^\nu \phi B_{\mu\nu}$
	$\mathcal{O}_W = \frac{ig}{2} (D^\mu \phi)^\dagger \sigma^a D^\nu \phi W_{\mu\nu}^a$

Approximate Bayesian Computation (ABC)

[D. Rubin 1984]



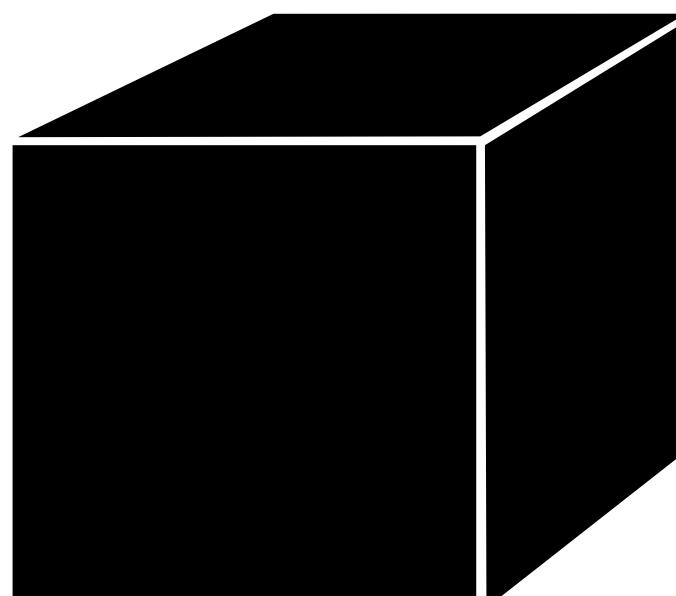
- How to choose x' ?
- How to choose ε ?
- No tractable posterior
- Need to run new simulations for new data or new prior

“Curse of dimensionality”
Precision vs efficiency tradeoff

Likelihood-free inference methods

Treat simulator as black box:

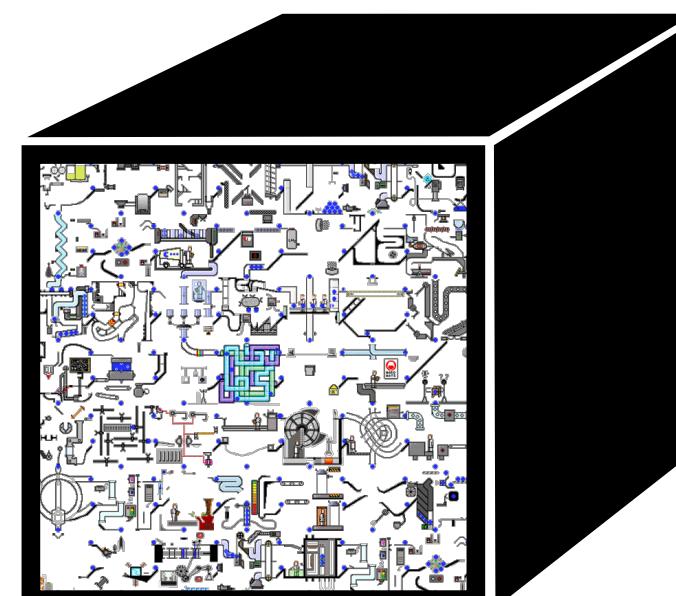
- Histograms of observables,
Approximate Bayesian Computation
Rely on summary statistics
- Machine learning techniques
Density networks, CARL, autoregressive models,
normalizing flows, ...



Use latent structure:

- Matrix Element Method, Optimal Observables,
Shower Deconstruction, Event Deconstruction
Neglect or approximate shower + detector, explicitly calculate
 \mathcal{Z} integral
- Mining gold from the simulator
Leverage matrix-element information + machine learning

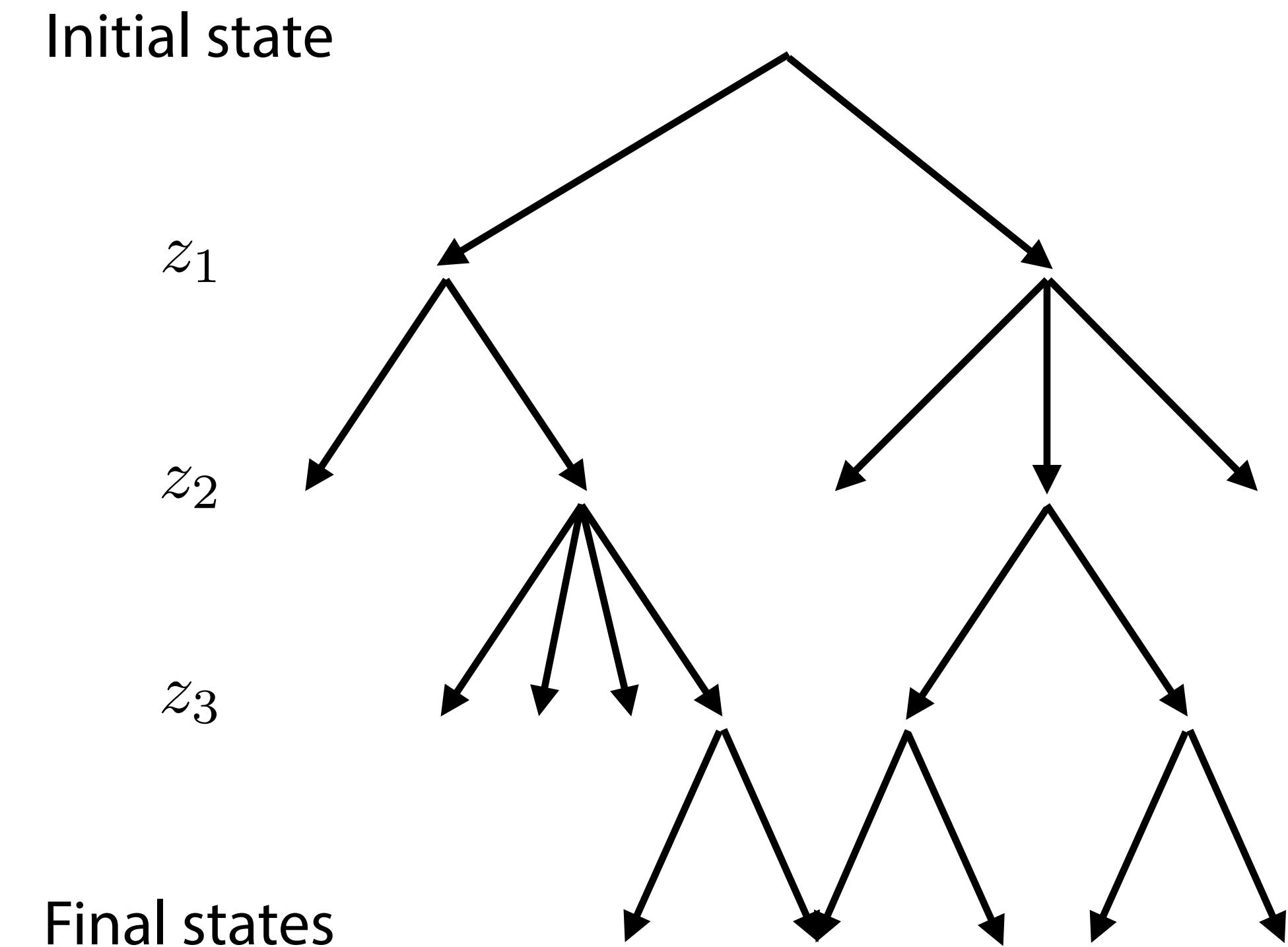
New!



Extracting the joint likelihood ratio from any simulation

- Computer simulation typically evolve along a tree-like structure of successive random branchings
- The probabilities of each branching $p_i(z_i|z_{i-1}, \theta)$ are often clearly defined in the code:

```
if random() > 0.1 + 2.5 * model_parameter:  
    do_one_thing()  
else:  
    do_another_thing()
```



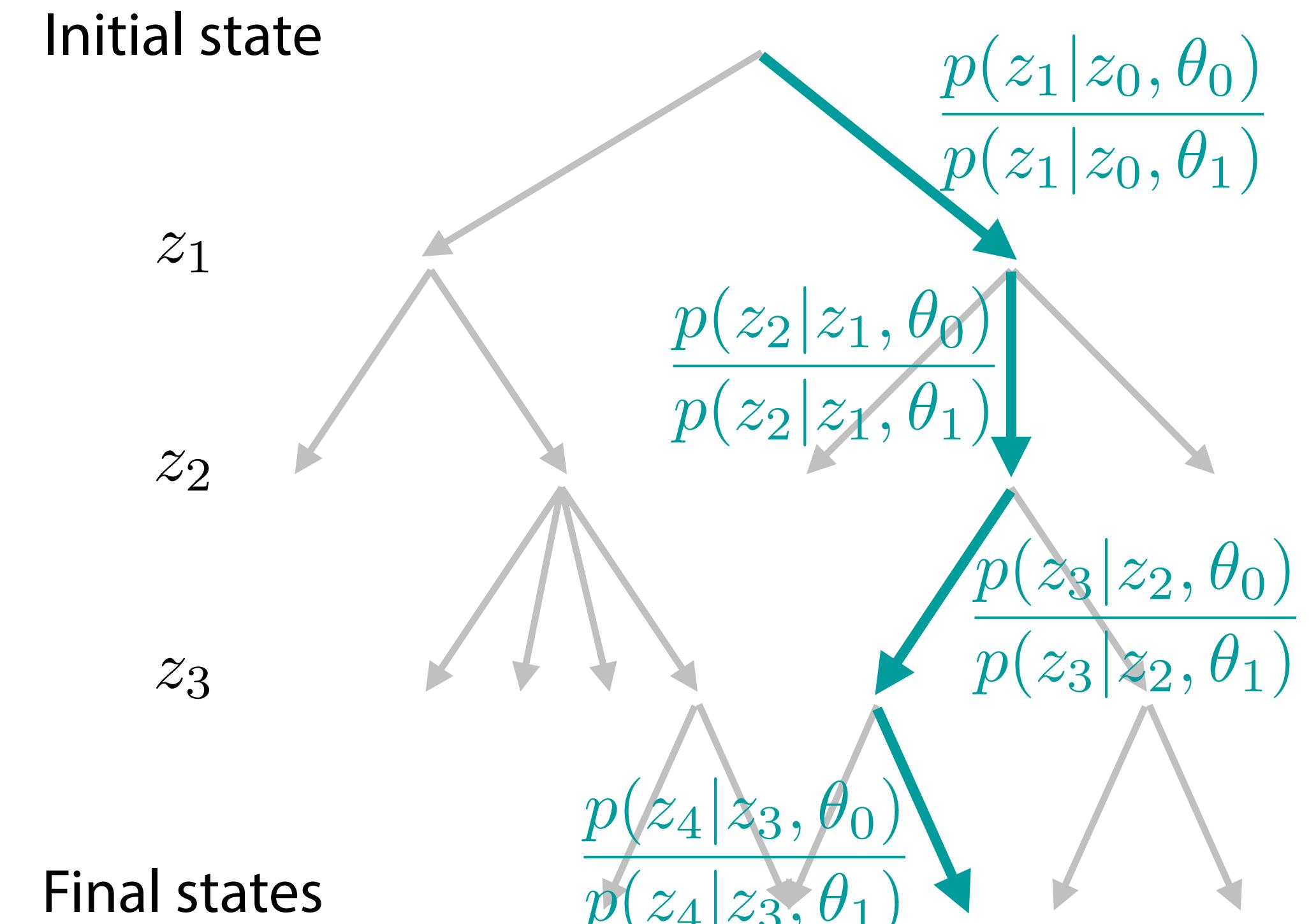
Extracting the joint likelihood ratio from any simulation

- Computer simulation typically evolve along a tree-like structure of successive random branchings
- The probabilities of each branching $p_i(z_i|z_{i-1}, \theta)$ are often clearly defined in the code:

```
if random() > 0.1 + 2.5 * model_parameter:  
    do_one_thing()  
else:  
    do_another_thing()
```

- For each run of the simulator, we can calculate the probability **of the chosen path** for different values of the parameters, and the “**joint likelihood ratio**”:

$$r(x, z|\theta_0, \theta_1) = \frac{p(x, z|\theta_0)}{p(x, z|\theta_1)} = \prod_i \frac{p(z_i|z_{i-1}, \theta_0)}{p(z_i|z_{i-1}, \theta_1)}$$



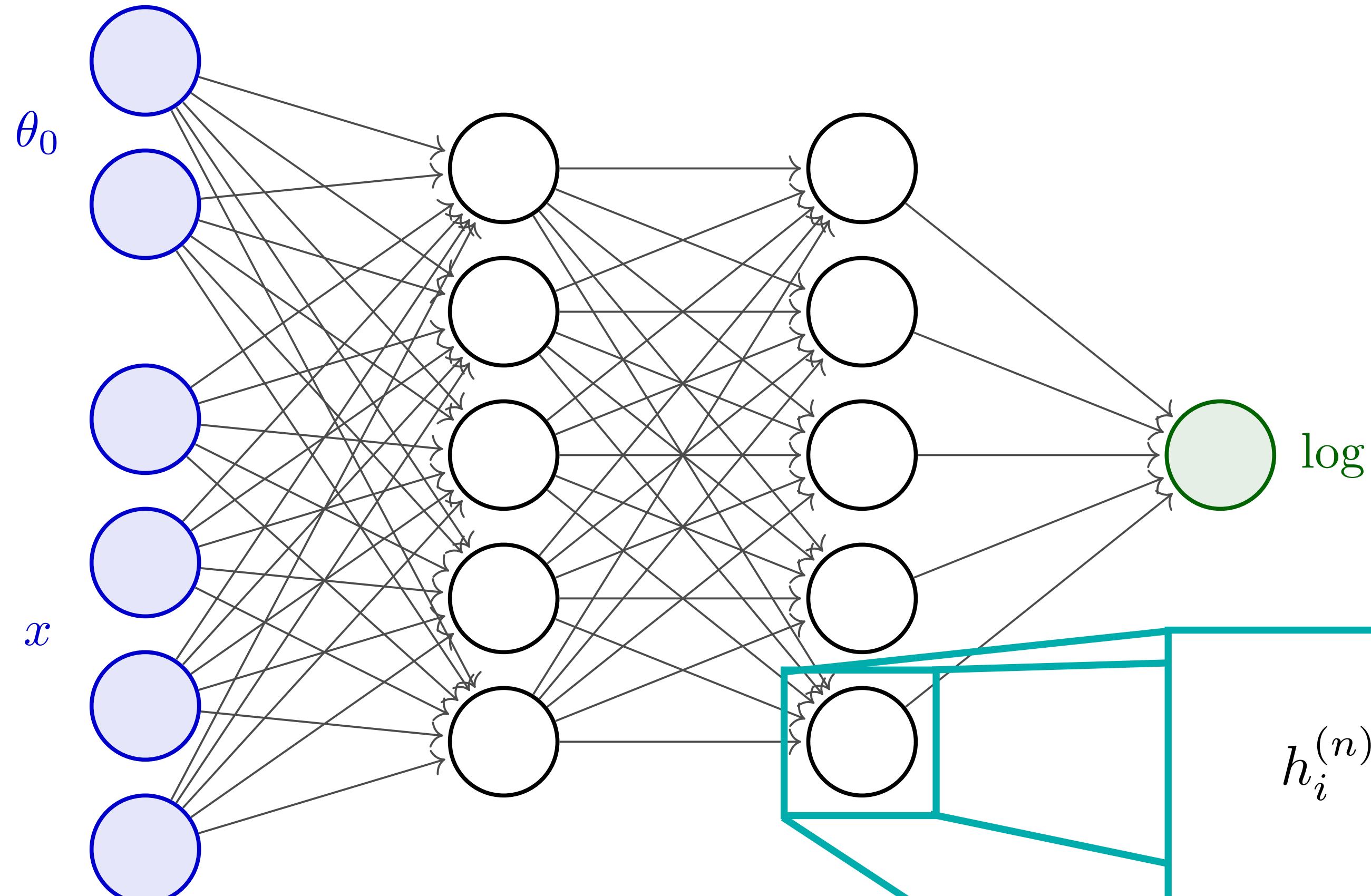
Variational calculus

$$\begin{aligned} L[\hat{g}(x)] &= \int dx dz \textcolor{red}{p}(x, z|\theta) |g(x, z) - \hat{g}(x)|^2 \\ &= \underbrace{\int dx \left[\hat{g}^2(x) \int dz \textcolor{red}{p}(x, z|\theta) - 2\hat{g}(x) \int dz \textcolor{red}{p}(x, z|\theta) g(x, z) + \int dz \textcolor{red}{p}(x, z|\theta) g^2(x, z) \right]}_{F(x)} \end{aligned}$$

$$0 = \frac{\delta F}{\delta \hat{g}} \Big|_{g^*} = 2\hat{g} \underbrace{\int dz \textcolor{red}{p}(x, z|\theta)}_{=\textcolor{red}{p}(x|\theta)} - 2 \int dz \textcolor{red}{p}(x, z|\theta) g(x, z)$$

$$g^*(x) = \frac{1}{\textcolor{red}{p}(x|\theta)} \int dz \textcolor{red}{p}(x, z|\theta) g(x, z)$$

Neural networks = universal function approximators



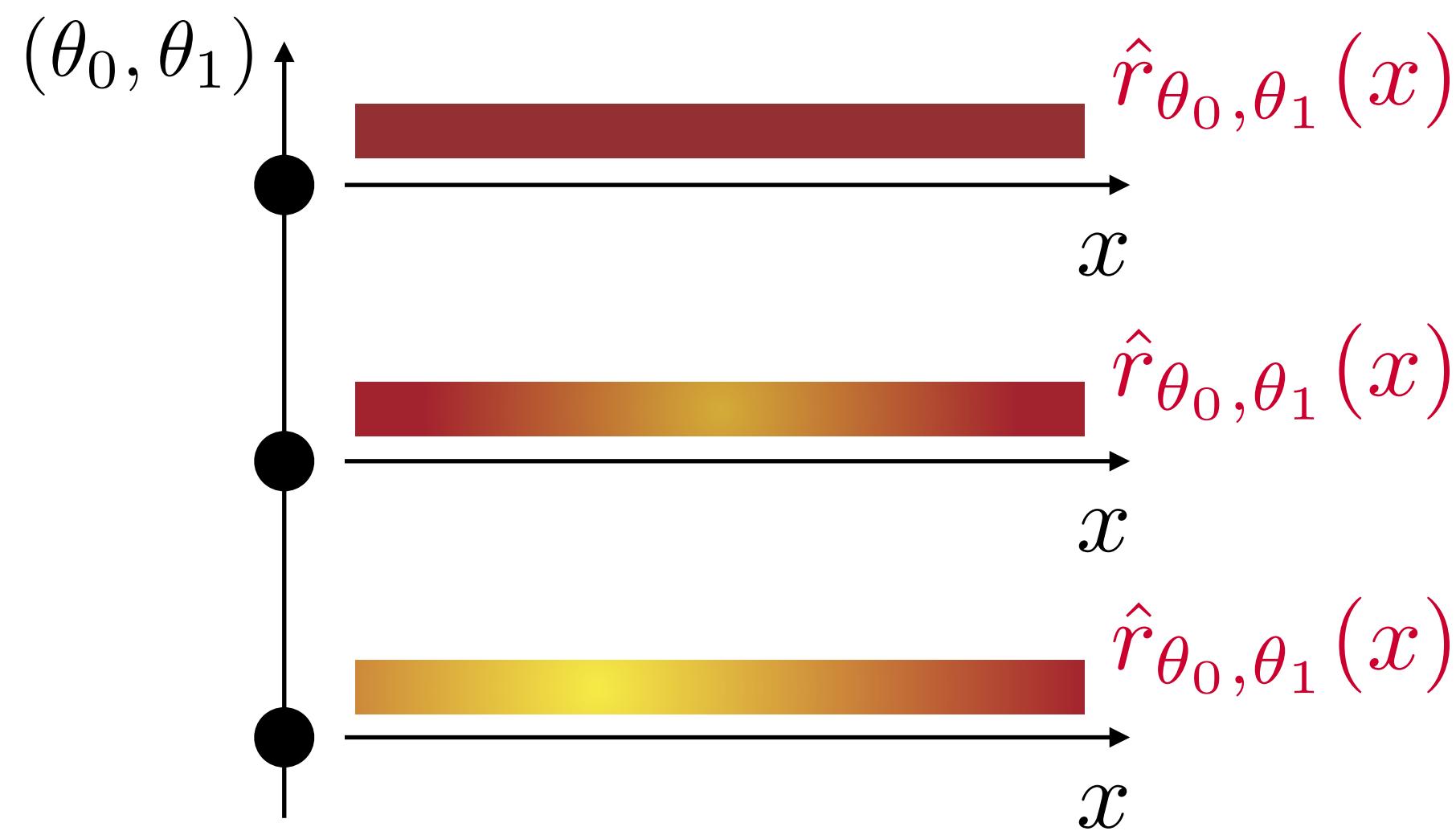
$$h_i^{(n)} = f \left(\sum_k w_{ik}^{(n)} h_k^{(n-1)} + b_i^{(n)} \right)$$

where the weights $w_{ik}^{(n)}, b_i^{(n)}$ are parameters “trained” by the optimizer

Two types of likelihood ratio estimators

A) Point by point:

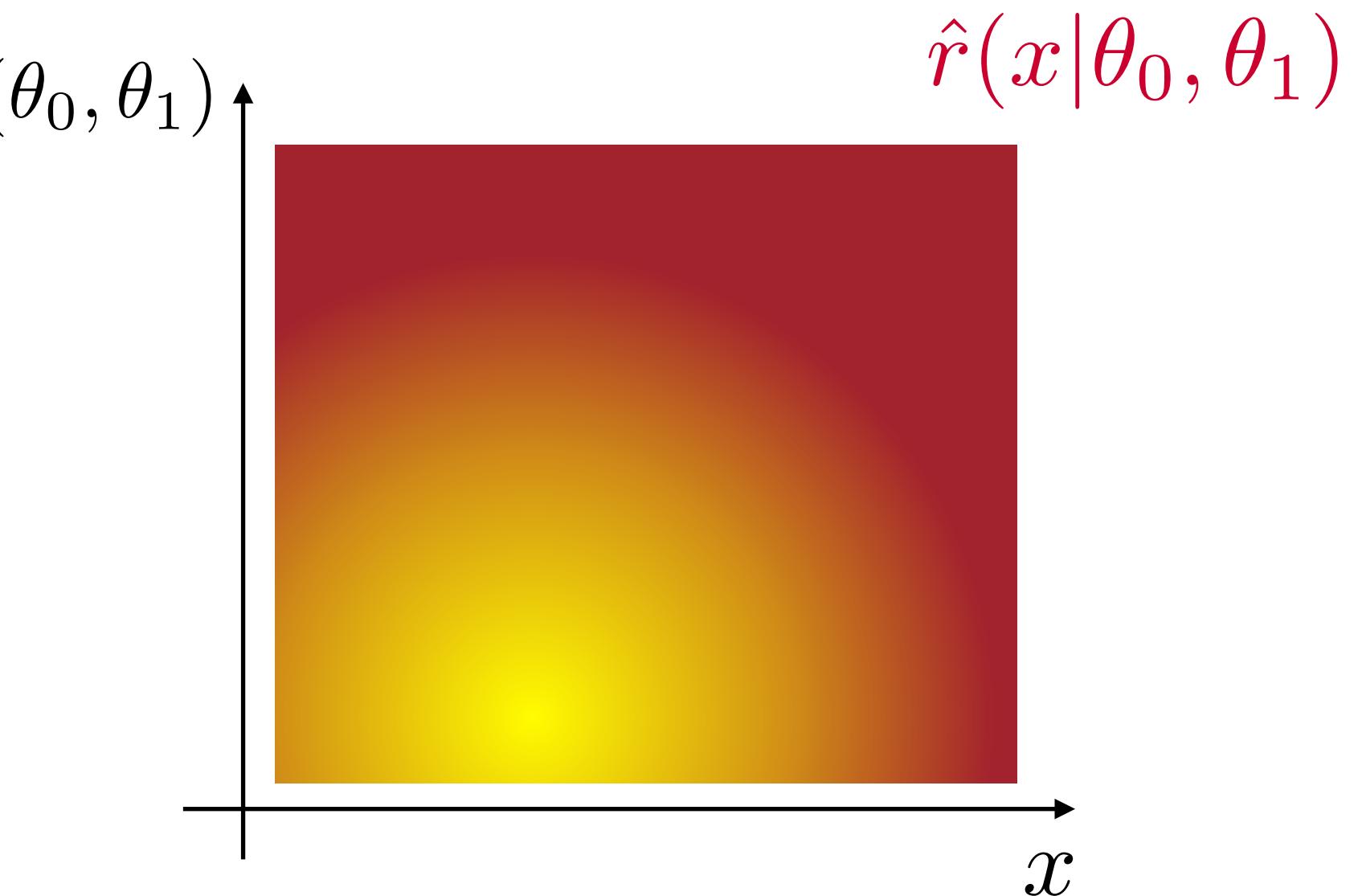
- first, define grid of parameter points $\{(\theta_0, \theta_1)\}$
- for each combination (θ_0, θ_1) ,
create separate estimator $\hat{r}_{\theta_0, \theta_1}(x)$
- final results can be interpolated between grid points



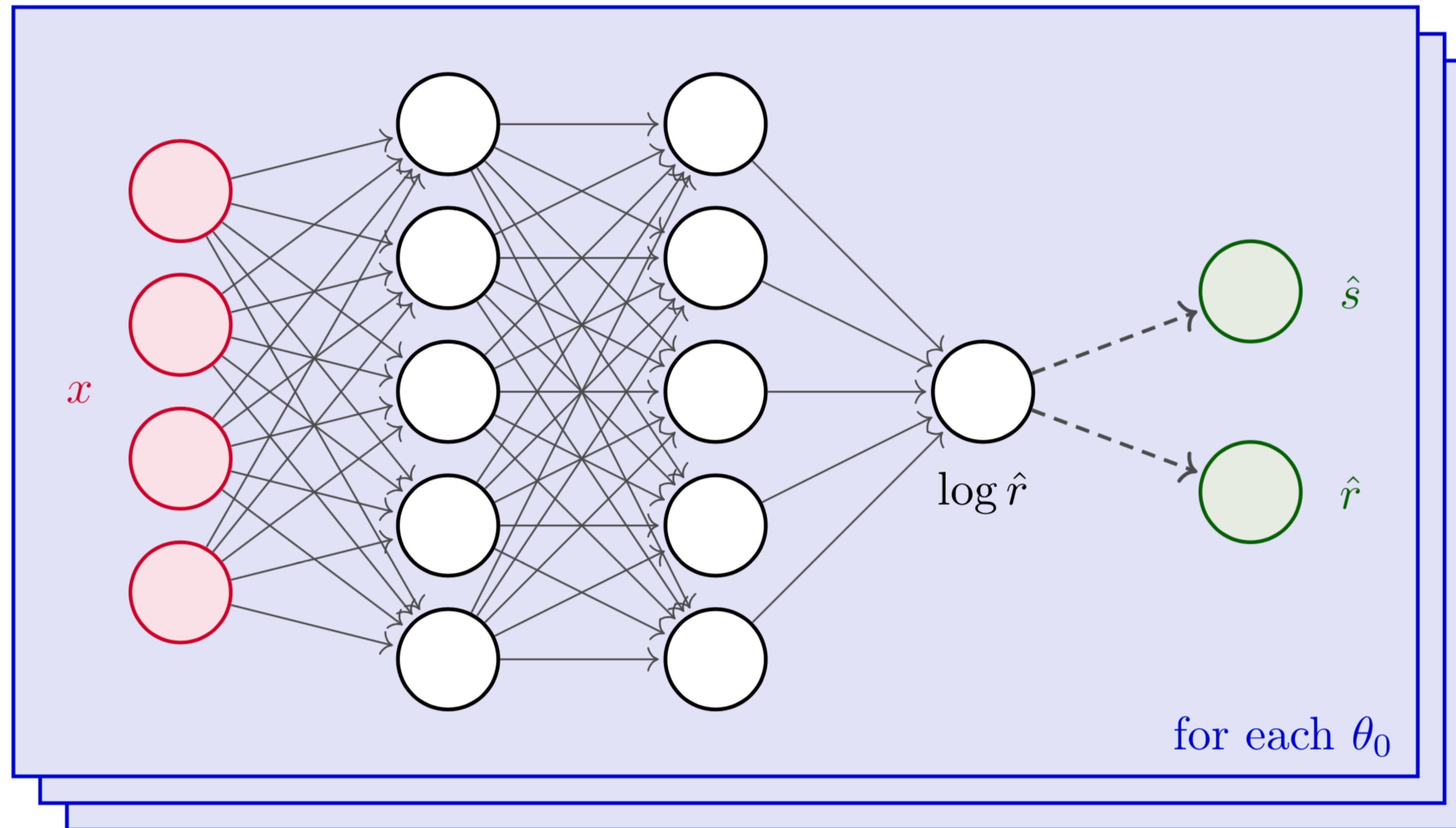
B) Parameterized:

[K. Cranmer, J. Pavez, G. Louppe 1506.02169;
P. Baldi et al. 1601.07913]

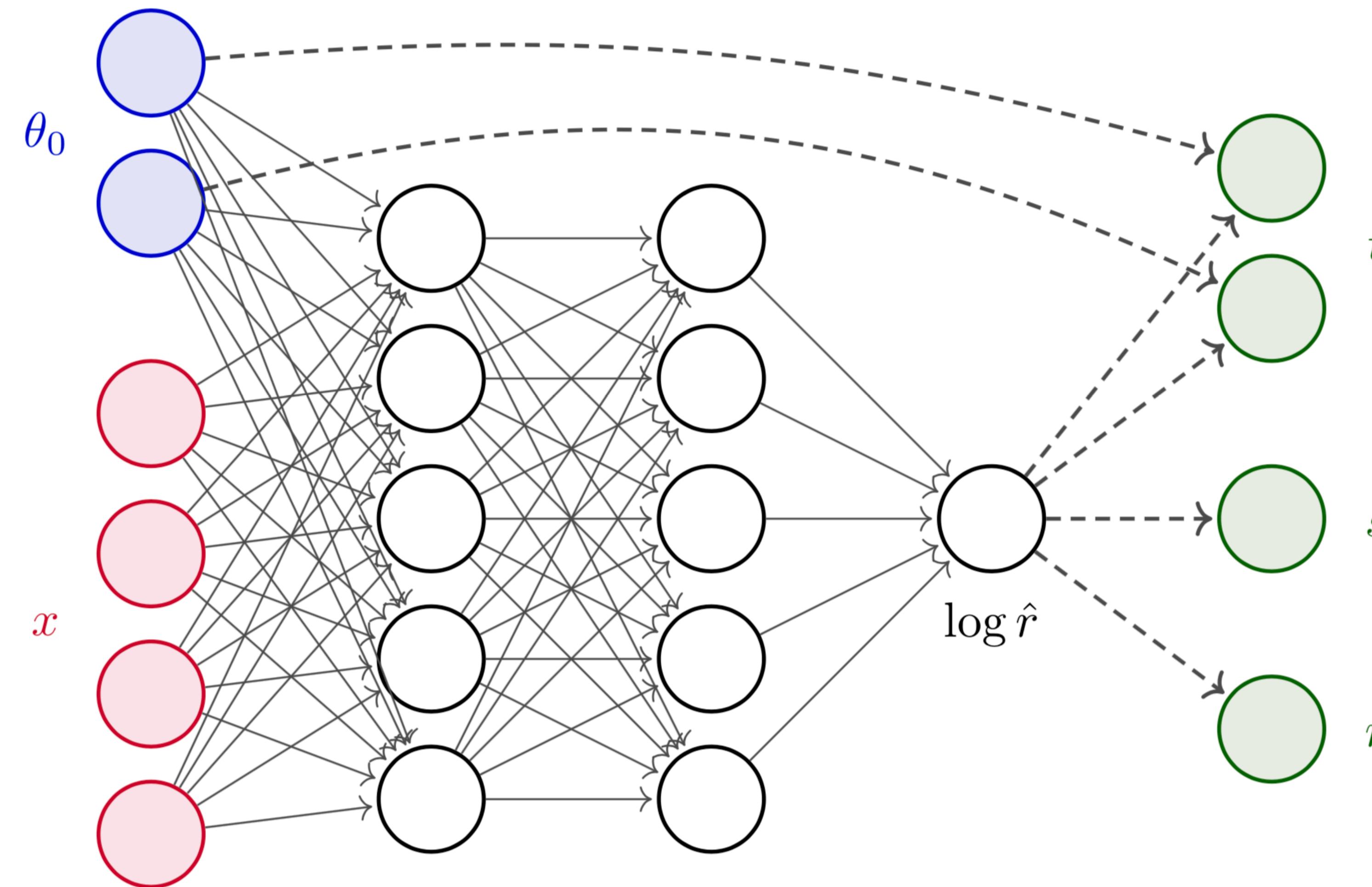
- create one estimator $\hat{r}(x|\theta_0, \theta_1)$
that is a function of θ_0 and θ_1
- no further interpolation necessary
- “borrows information” from close points



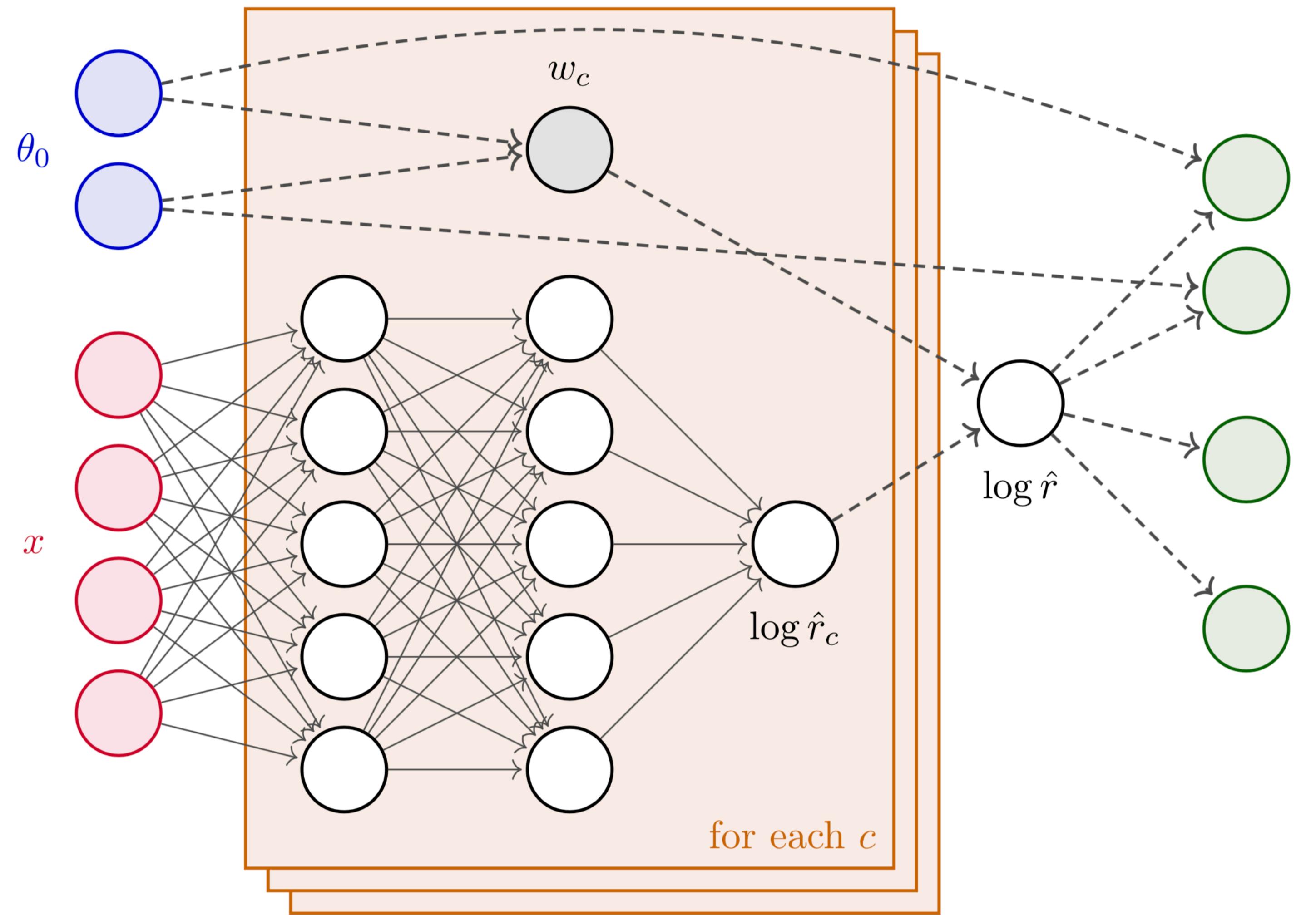
Point by point



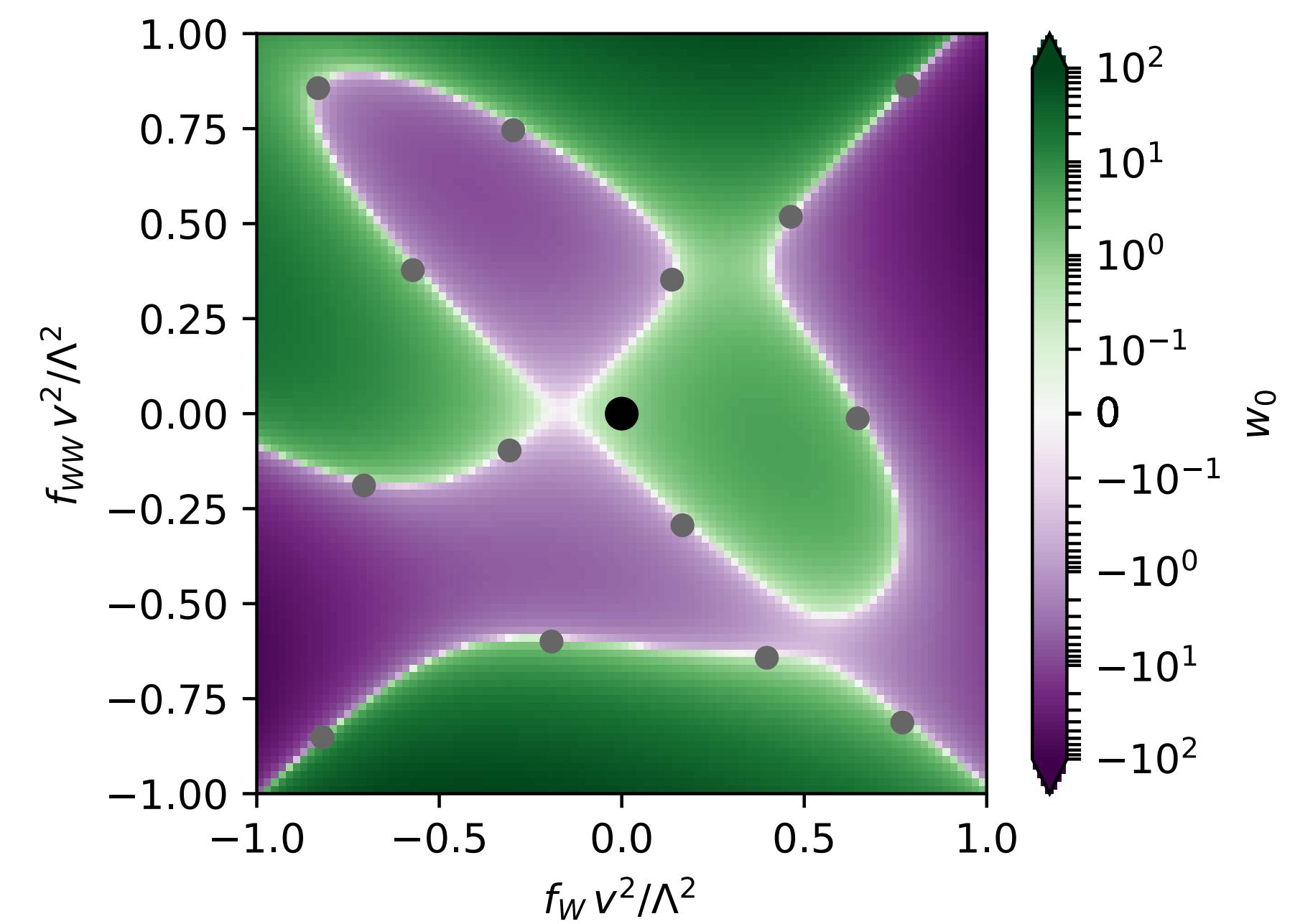
(Agnostic) parameterized estimators



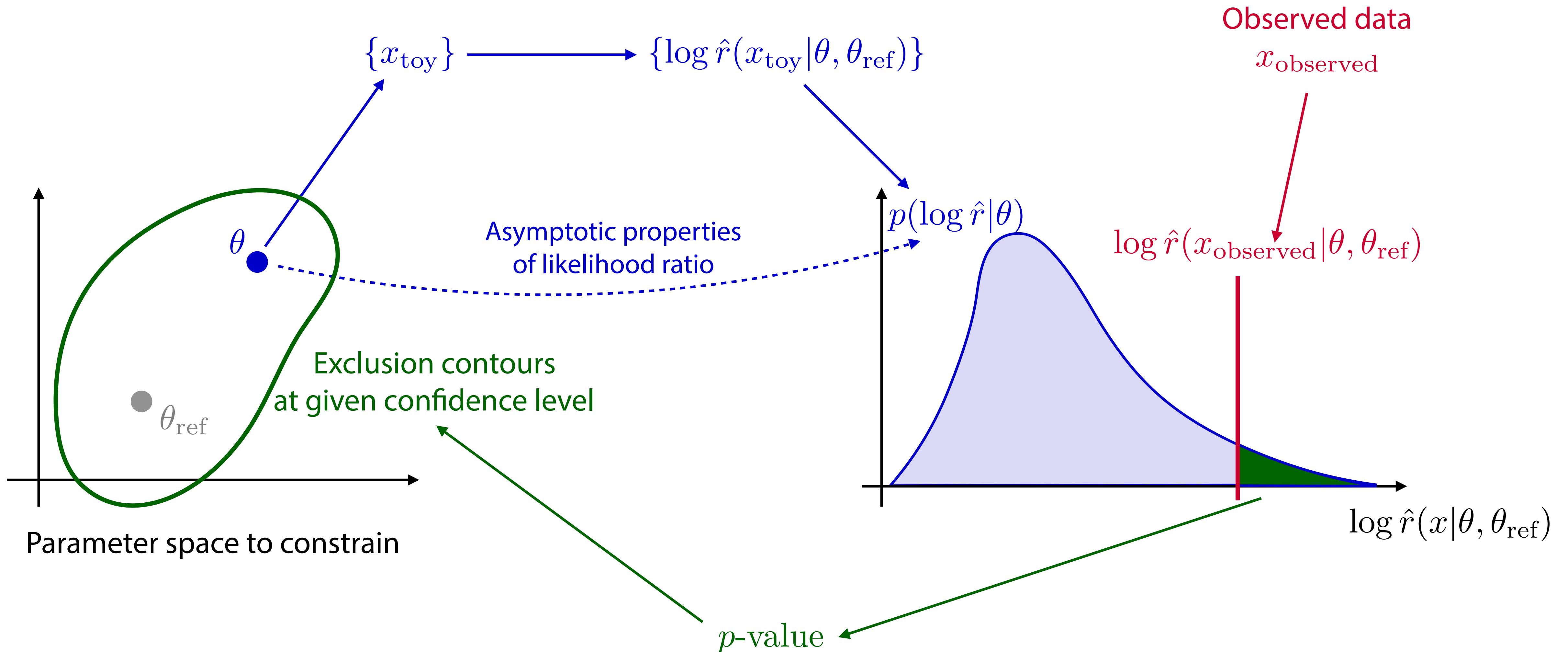
Morphing-aware parameterized estimators



$$\hat{r}(x|\theta_0, \theta_1) = \sum_c w_c(\theta_0) \hat{r}_c(x)$$



Limit setting (frequentist)



The likelihood ratio is the most powerful test statistic

IX. *On the Problem of the most Efficient Tests of Statistical Hypotheses.*

By J. NEYMAN, Nencki Institute, Soc. Sci. Lit. Varsoviensis, and Lecturer at the Central College of Agriculture, Warsaw, and E. S. PEARSON, Department of Applied Statistics, University College, London.

(Communicated by K. PEARSON, F.R.S.)

(Received August 31, 1932.—Read November 10, 1932.)

CONTENTS.

	PAGE.
I. Introductory	289
II. Outline of General Theory	293
III. Simple Hypotheses	

WBF Higgs to four leptons with detector effects

